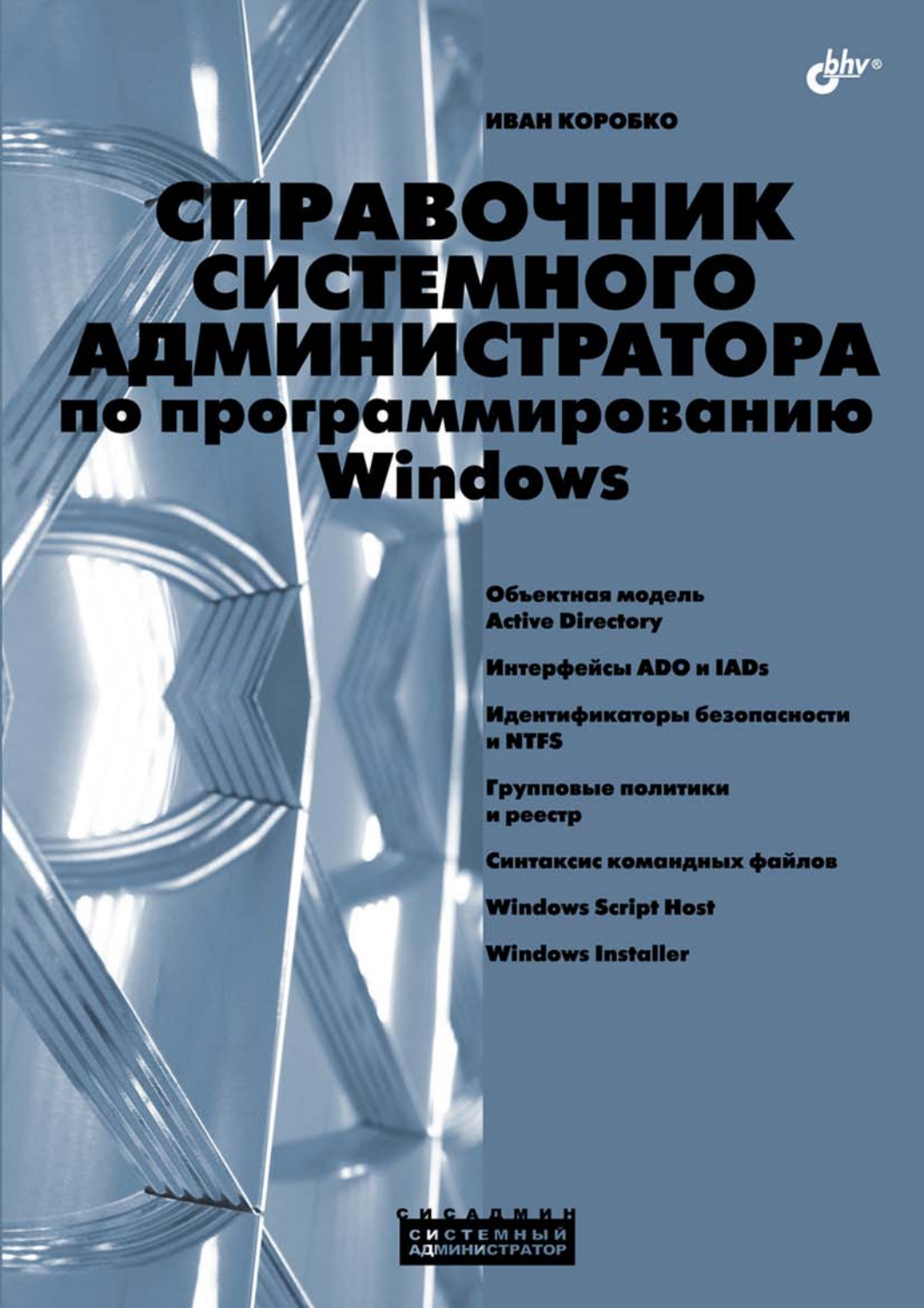

bhv
ИВАН КОРОБКО
СПРАВОЧНИК СИСТЕМНОГО АДМИНИСТРАТОРА
по программированию Windows
Объектная модель Active Directory
Интерфейсы ADO и IADs
Идентификаторы безопасности и NTFS
Групповые политики и реестр
Синтаксис командных файлов
Windows Script Host
Windows Installer
СИСАДМИН
СИСТЕМНЫЙ АДМИНИСТРАТОР

**Иван Коробко**

# СПРАВОЧНИК СИСТЕМНОГО АДМИНИСТРАТОРА по программированию Windows

Санкт-Петербург
«БХВ-Петербург»
2009

УДК 681.3.06
ББК 32.973.26-018.2
К68

**Коробко И. В.**

К68 Справочник системного администратора по программированию Windows. — СПб.: БХВ-Петербург, 2009. — 576 с.: ил. — (Системный администратор)

ISBN 978-5-9775-0296-2

Приведена исчерпывающая информация по устройству и управлению Active Directory. Описаны интерфейсы IADs, ADO, а также основные приемы программного управления каталогом на языке VBScript. Особое внимание уделено безопасности операционной системы: программному управлению NTFS, принципам построения и чтения идентификаторов безопасности и др. Подробно освещен вопрос управления групповыми политиками (ADM-файлы) и реестром (REG-файлы). Приведен синтаксис пакетных файлов (autorun.inf и др.), рассказано о технологии изменения дистрибутивов, созданных с помощью Windows Installer.

*Для системных администраторов и программистов*

УДК 681.3.06
ББК 32.973.26-018.2

**Группа подготовки издания:**

| | |
|---|---|
| Главный редактор | *Екатерина Кондукова* |
| Зам. главного редактора | *Игорь Шишигин* |
| Зав. редакцией | *Григорий Добин* |
| Редактор | *Юрий Якубович* |
| Компьютерная верстка | *Натальи Смирновой* |
| Корректор | *Наталия Першакова* |
| Дизайн серии | *Инны Тачиной* |
| Оформление обложки | *Елены Беляевой* |
| Зав. производством | *Николай Тверских* |

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 31.10.08.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 46,44.
Тираж 2000 экз. Заказ №
"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-9775-0296-2

© Коробко И. В., 2008
© Оформление, издательство "БХВ-Петербург", 2008

# Оглавление

**Об авторе** ..... **3**

**Предисловие** ..... **5**

**Введение** ..... **7**

Для кого эта книга? ..... 7
Какой язык программирования выбрать? ..... 7
Как пользоваться книгой? ..... 8
Какой редактор сценариев использовать? ..... 11
Какова структура книги? ..... 12

**Благодарности** ..... **15**

**Раздел 1. Основы построения Active Directory** ..... **17**

**Глава 1. Внутреннее устройство каталога Active Directory** ..... **19**

Основные термины и понятия ..... 19
Нормативная документация по Active Directory ..... 20
Общие сведения ..... 20
RFC по LDAP v2 ..... 21
RFC по LDAP v3 ..... 21
Особенности и характеристики Active Directory ..... 22
Active Directory как часть файловой системы ..... 23
Архитектура службы каталогов Active Directory ..... 24
DNS и Active Directory ..... 25
Службы каталогов Active Directory ..... 26
Глобальный каталог ..... 27
Провайдеры ADSI ..... 29
Схема объектной модели ADSI ..... 30
Объекты в Active Directory ..... 31
Доступ к объектам Active Directory ..... 31

Типы объектов Active Directory ..... 32
Типы данных объектов в Active Directory ..... 33

**Глава 2. Установка домена Windows 2003/2008 ..... 35**

Основные термины и понятия ..... 35
Подготовка к установке Active Directory ..... 36
Запуск мастера установки Active Directory ..... 36
Синтаксис файла ответов утилиты *dcpromo.exe* ..... 37
Описание процесса установки Active Directory ..... 41
Мастер установки домена Windows 2008 ..... 50

**Раздел 2. Принципы программного управления Active Directory ..... 53**

**Глава 3. Основы программного управления Active Directory ..... 55**

Основные термины и понятия ..... 55
Идентификаторы Active Directory ..... 56
Типы объектов в Active Directory ..... 57
Способы доступа к объектам Active Directory ..... 59
LDAP-путь к объекту ..... 60
Развернутая форма записи ..... 61
Сокращенная форма записи ..... 61
Способы доступа к каталогу Active Directory ..... 62
Диалект LDAP ..... 62
Диалект SQL ..... 66
Прилинкованный SQL-сервер ..... 69
Чтение данных из Active Directory ..... 75
Тип данных: длинное целое число (*VarType = 3*) ..... 77
Тип данных: дата-время (*VarType = 7*) ..... 79
Тип данных: строка (*VarType = 8*) ..... 82
Тип данных: объект (*VarType = 9*) ..... 85
Тип данных: булево значение (*VarType = 11*) ..... 87
Тип данных: массив элементов (*VarType = 8192+x*) ..... 88

**Глава 4. Объектная модель интерфейсов *IADs**** ..... 90

Основные термины и понятия ..... 90
Типы данных, поддерживаемые интерфейсами *IADs** ..... 91
Классификация интерфейсов *IADs** ..... 93
Основные интерфейсы ..... 93

Объектная модель *IADs* ..... 93
Объектная модель *IADsContainer* ..... 105
Объектная модель *IADsNamespaces* ..... 111
Объектная модель *IADsOpenDSObject* ..... 112
Интерфейсы свойств объектов ..... 113
Объектная модель *IADsPropertyList* ..... 114
Объектная модель *IADsPropertyEntry* ..... 120
Объектные модели *IADsPropertyValue* и *IADsPropertyValue2* ..... 124
Интерфейсы объектов ..... 127
Методы интерфейсов объектов ..... 128
Свойства интерфейсов объектов ..... 128
Использование *IADs**-интерфейсов на практике ..... 129
Способы доступа к объектам Active Directory ..... 131
Управление атрибутами с помощью интерфейса *IADs* ..... 133
Управление атрибутами с помощью интерфейса *IADsUser* ..... 133

**Глава 5. Объектная модель ADO DB ..... 135**

Основные термины и понятия ..... 135
Объекты ADO ..... 135
Объект *Connection* ..... 137
Объект *Command* ..... 146
Объект коллекции *Properties* ..... 153
Объект *Recordset* ..... 155

**Раздел 3. Объекты Active Directory ..... 169**

**Глава 6. Виртуальный объект RootDSE ..... 171**

Основные термины и понятия ..... 171
Назначение объекта RootDSE ..... 171
Объектная модель RootDSE ..... 172
Атрибуты, характеризующие структуру домена ..... 172
Атрибуты, характеризующие конфигурацию домена ..... 173
Атрибуты, характеризующие конфигурацию контроллера домена ..... 175
Управляющие объекты ..... 175
Получение сведений об объекте RootDSE ..... 178
Утилиты для просмотра характеристик RootDSE ..... 178
Программное управление RootDSE ..... 188
Определение имени домена ..... 189

**Глава 7. Атрибуты основных объектов Active Directory ..... 191**

Основные термины и понятия ..... 191
Условные обозначения ..... 192
Обязательные атрибуты для всех объектов Active Directory ..... 193
Основные объекты Active Directory ..... 194
Пользователь ..... 195
Группа безопасности ..... 204
Контейнер ..... 207
Компьютер ..... 208

**Глава 8. Управление учетной записью пользователя ..... 214**

Основные термины и понятия ..... 214
Производимые операции ..... 214
Создание пользователя ..... 215
Работа мастера. Теория ..... 216
Работа мастера. Практика ..... 219
Программное создание учетной записи пользователя ..... 224
Поля учетной записи, создаваемой мастером ..... 227
Удаление пользователя ..... 227
Удаление с помощью мастера ..... 227
Удаление программным способом ..... 227
Чтение свойств пользователя ..... 229
Вкладка *General* ..... 229
Вкладка *Address* ..... 233
Вкладка *Account* ..... 235
Вкладка *Profile* ..... 242
Вкладка *Telephones* ..... 245
Вкладка *Organization* ..... 247
Вкладка *Member Of* ..... 251
Вкладка *Dial-in* ..... 253
Вкладка *Environment* ..... 258
Вкладка *Sessions* ..... 261
Вкладка *Remote control* ..... 265
Вкладка *Terminal Services Profile* ..... 267
Вкладка *COM+* ..... 270

**Глава 9. Управление учетной записью группы безопасности ..... 271**

Основные термины и понятия ..... 271
Производимые операции ..... 271
Создание группы ..... 272

Работа мастера. Теория ....272
Работа мастера. Практика ....275
Программное создание группы ....277
Поля создаваемой мастером учетной записи ....278
Удаление группы ....279
Удаление с помощью мастера ....279
Удаление программным способом ....280
Чтение параметров группы безопасности ....280
Вкладка *General* ....281
Вкладка *Members* ....284
Вкладка *Member Of* ....285
Вкладка *Managed By* ....287

**Глава 10. Управление контейнером ....289**

Основные термины и понятия ....289
Создание контейнера ....289
Работа мастера. Теория ....290
Работа мастера. Практика ....291
Программное создание учетной записи контейнера ....293
Поля контейнера, создаваемого мастером ....293
Удаление контейнера ....294
Удаление с помощью мастера ....294
Удаление программным способом ....294
Чтение свойств контейнера ....296
Вкладка *General* ....296
Вкладка *Managed By* ....299
Вкладка *COM+* ....300
Вкладка *Group Policy* ....301

**Глава 11. Управление учетной записью компьютера ....304**

Основные термины и понятия ....304
Создание учетной записи компьютера ....304
Создание учетной записи компьютера в MMC-консоли ....305
Программное создание учетной записи компьютера ....312
Удаление компьютера ....315
Удаление с помощью мастера ....315
Удаление программным способом ....316
Чтение свойств компьютера ....316
Вкладка *General* ....317
Вкладка *Operating System* ....319

Вкладка *Member Of* ..... 321
Вкладка *Delegation* ..... 323
Вкладка *Location* ..... 325
Вкладка *Managed By* ..... 325
Вкладка *Dial-in* ..... 326
Публикация компьютера в домене ..... 331

**Раздел 4. Безопасность ..... 335**

**Глава 12. Идентификаторы безопасности Active Directory ..... 337**

Основные термины и понятия ..... 337
Идентификаторы в Active Directory ..... 337
Globally Unique Identifier ..... 338
Структура GUID ..... 338
Определение типа GUID ..... 339
Алгоритм создания GUID v.1 ..... 339
Получение нового GUID ..... 341
Особенности глобального идентификатора GUID ..... 342
Security Identifier ..... 342
Структура SID ..... 342
Чтение SID из Active Directory ..... 343
Определение SID пользователя с помощью утилиты *GetSID* ..... 346
Особенности SID ..... 347
Широко известные идентификаторы SID ..... 348

**Глава 13. Программное управление безопасностью файловой системы ..... 355**

Основные термины и понятия ..... 355
Виды файловых систем ..... 356
FAT ..... 356
HPFS ..... 357
VFAT ..... 357
FAT32 ..... 357
NTFS ..... 358
Главная файловая таблица ..... 359
Метафайлы NTFS ..... 359
Особенности NTFS ..... 362
Сравнение файловых систем ..... 362
Управление правами доступа на файлы и папки ..... 364
Организация доступа к параметрам безопасности ..... 364

Объектная модель NTFS ....364
Объектная модель *IADsSecurityUtility* ....365
Методы интерфейса *IADsSecurityUtility* ....366
Свойства интерфейса *IADsSecurityUtility* ....369
Объектная модель *IADsSecurityDescriptor* ....370
Объектная модель *IADsAccessControlList* ....373
Методы интерфейса *IADsAccessControlList* ....374
Свойства интерфейса *IADsAccessControlList* ....380
Объектная модель *IADsAccessControlEntry* ....381
Свойства интерфейса *IADsAccessControlEntry* ....382
Стандартный набор параметров безопасности ....387
Технология Access-based Enumerator ....388
Установка ABE ....389
Ограничения технологии ABE ....391
Способы настройки ABE ....392
Удаление ABE ....393
Недостатки технологии ABE ....393
Практика использования ABE ....393

**РАЗДЕЛ 5. УПРАВЛЕНИЕ КОМПЬЮТЕРОМ ....395**

**ГЛАВА 14. РЕЕСТР WINDOWS 2K ....397**

Основные термины и понятия ....397
Историческая справка ....397
Основы построения реестра ....398
Синтаксис REG-файлов ....400
Редакторы реестра ....401
Программное управление реестром ....402
Управление реестром с помощью WSH ....402
Управление реестром с помощью групповых политик ....404
Синтаксис ADM-файлов ....405
Управление интерфейсом групповых политик ....410
Примеры использования административных шаблонов на практике ....418

**ГЛАВА 15. WINDOWS SCRIPT HOST ....421**

Основные термины и понятия ....421
Сервер сценариев Windows Script ....421
Запуск сценариев WSH из командной строки ....422
Возможности WSH-сценариев ....424
Объектная модель WSH ....424

Объект *WScript* ....426
Свойства объекта *WScript* ....426
Методы объекта *WScript* ....430
Объект *WshArguments* ....433
Объект *WshShell* ....433
Метод *AppActivate()* ....435
Управление ярлыками методом *CreateShortcut()* ....436
Свойство *Environment* ....436
Метод *ExpandEnvironmentStrings()* ....441
Метод *LogEvent()* ....441
Метод *Popup()* ....442
Метод *Run()* ....444
Метод *SendKeys()* ....446
Свойство *SpecialFolders* ....448
Объект *WshShortcut* ....450
Создание ярлыка ....451
Чтение и изменение свойств ярлыка ....453
Удаление ярлыка ....453
Объект *WshURLShortcut* ....454
Объект *WshNetwork* ....455
Методы *AddWindowsPrinterConnection()* и *AddPrinterConnection()* ....456
Метод *RemovePrinterConnection()* ....458
Метод *EnumPrinterConnections()* ....459
Метод *SetDefaultPrinter()* ....459
Метод *MapNetworkDrive()* ....460
Метод *EnumNetworkDrives()* ....461
Метод *RemoveNetworkDrive()* ....461
Ошибки выполнения сценариев в WSH ....461

**ГЛАВА 16. ЗАГРУЗКА ОПЕРАЦИОННОЙ СИСТЕМЫ ....464**

Основные термины и понятия ....464
Процесс запуска ....465
Предварительная загрузка ....465
Загрузка ....466
Загрузка ядра ....467
Инициализация ядра ....467
Регистрация ....468
Варианты загрузки операционной системы ....468
Безопасный режим ....469
Режим протоколирования загрузки ....470
Режим VGA ....470
Загрузка последней удачной конфигурации ....471

Восстановление службы каталогов ..... 471
Режим отладки ..... 472
Обычный режим ..... 472
Файл Boot.ini ..... 472
Компоненты файла Boot.ini ..... 473
ARC-путь ..... 474
Параметры настройки загрузки ОС ..... 475

**Глава 17. Особенные файлы Windows ..... 486**

Основные термины и понятия ..... 486
Файл подкачки ..... 486
Размер файла подкачки ..... 486
Снятие ограничения в 4096 Мбайт на размер файла подкачки ..... 487
Размещение файла подкачки ..... 487
Включение нескольких файлов подкачки ..... 488
Файл Autorun.inf ..... 489
Синтаксис ..... 489
Пакетный файл ..... 493
Различие между BAT и CMD ..... 493
Формальные параметры ..... 493
Команды пакетного файла ..... 494

**Глава 18. Windows Installer ..... 502**

Основные термины и понятия ..... 502
Пакетная установка программного обеспечения ..... 502
Windows Installer ..... 503
Параметры командной строки для Msiexec ..... 504
Windows Installer SDK ..... 506
Доступ к MSI-файлам с помощью графической оболочки Orca.exe ..... 506
Структура MSI-файлов ..... 508
Доступ к MSI-файлам программным способом ..... 511
Работа с таблицами в Orca.exe ..... 511

**Приложения ..... 517**

**Приложение 1. Таблица ASCII ..... 519**

Назначение служебных символов ..... 520
Форматирование ..... 521

Передача данных....521
Разделительные знаки при передаче информации....522
Другие символы....522

**Приложение 2. Измерения в байтах....524**

**Приложение 3. Системы счисления....525**

Десятичная система счисления....525
Двоичная система счисления....525
Шестнадцатеричная система счисления....526

**Приложение 4. Переменные среды....528**

**Приложение 5. Буквенные сокращения стран....531**

**Приложение 6. Интегрирование собственных шаблонов в PrimalScript....542**

**Ответственность....547**

**Литература....548**

**Предметный указатель....549**

*Посвящается моей жене — Елене Ходовой*

# Об авторе

Коробко Иван Викторович, сертифицированный специалист MCP, в настоящее время менеджер ИТ-проектов. Автор более 35 статей в журнале "Системный администратор" и книги "Администрирование сетей Windows с помощью сценариев", вышедшей в 2007 г. в издательстве "БХВ-Петербург".

# Предисловие

Работа на сервере всегда требовала повышенного внимания от главного человека в сети — системного администратора. Любая ошибка в его работе может дорого обойтись, поэтому избежать ее — одна из приоритетных задач.

Автоматизация рутинных действий с помощью сценариев давно взята на вооружение программистами и администраторами. В настоящее время тенденции таковы, что системный администратор вынужден программировать.

Необходимо понимать, что системный администратор — не программист, и никогда им не будет. Его задача — взять имеющийся шаблон и переделать его под свои нужды. Именно поэтому и появился этот справочник: знание подхода и объектной модели позволит любому системному администратору быстро и просто создать нужный сценарий, который сделает необходимые изменения в каталоге Active Directory, реестре или политике безопасности.

Автор надеется, что эта книга будет полезной для системных администраторов и начинающих программистов, а для некоторых из них она станет настольной книгой.

# Введение

## Для кого эта книга?

Вы держите в руках справочник, призванный помочь системным администраторам, специалистам системной поддержки и начинающим программистам в создании различных скриптов, позволяющих автоматизировать рутинные операции. Для чтения книги не требуется высокого уровня квалификации. В начале каждой главы кратко описано ее содержание и приведен список основных терминов и понятий.

## Какой язык программирования выбрать?

Сценарии, созданные для управления серверами, в большинстве случаев будут запускаться на серверах. Любой системный администратор согласится, что на сервер не стоит ставить какое-либо программное обеспечение без особой необходимости, поэтому использовать встроенный язык программирования — самый простой и логичный ответ.

Операционной системой Windows поддерживается ряд языков программирования: VBScript, JScript, PowerShell (Monad), командные файлы (CMD, BAT). Полноценный доступ к Active Directory обеспечат вам лишь первые два. Несмотря на то, что PowerShell позволяет обеспечить работу с Active Directory, автор не рекомендует его использовать, поскольку по своей сути с его помощи создаются командные файлы, в которых сложно реализовать многие функции. Выбор между первыми двумя языками программирования остается за читателем. Поскольку автор предпочитает VBScript, то листинги всех сценариев в этой книге приведены на этом языке.

В некоторых случаях в сценариях должны присутствовать длинные строки, которые не удается разместить в одну строку в печатном издании. Такие

строки в листингах разбиты на две или несколько строк, а в начале перенесенной строки стоит знак ↳:

```
Очень длинная строка, которую никак невозможно разместить в одной строке
↳книги, но в сценарии нельзя разбивать
```

# Как пользоваться книгой?

Давайте рассмотрим решение задачи на примере. Пусть наша задача заключается в задании пути к профилю для всех пользователей в указанной папке.

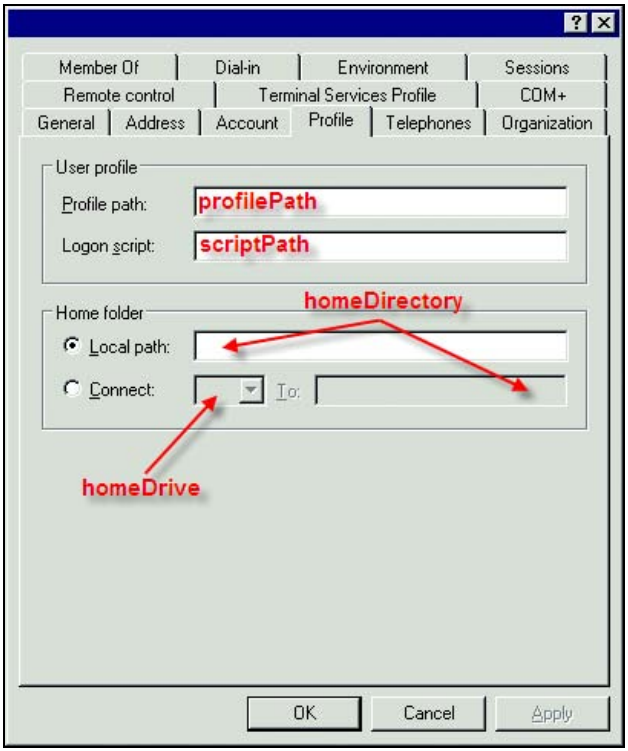



**Рис. В1.** Вкладка **Profile** учетной записи пользователя

Первое, что необходимо сделать, — это разделить поставленную задачу на несколько подзадач, а именно:

- получение доступа к Active Directory;

- поиск всех учетных записей пользователя в указанной папке;
- присвоение нужного значения в соответствующие учетные записи.

Поскольку речь идет об учетной записи пользователя, то необходимо обратиться к *главе 8* в *разделе 3 "Объекты Active Directory"*, которая называется *"Управление учетной записью пользователя"*. Путь к профилю назначается во вкладке **Profile**, поэтому в одноименном разделе — *"Вкладка Profile"* — ищем описание этого параметра и наглядную иллюстрацию (рис. 8.17). Для удобства мы дублируем этот рисунок здесь (рис. В1). На приведенном рисунке видно, что требуемому параметру **Profile path** соответствует атрибут `profilePath`.

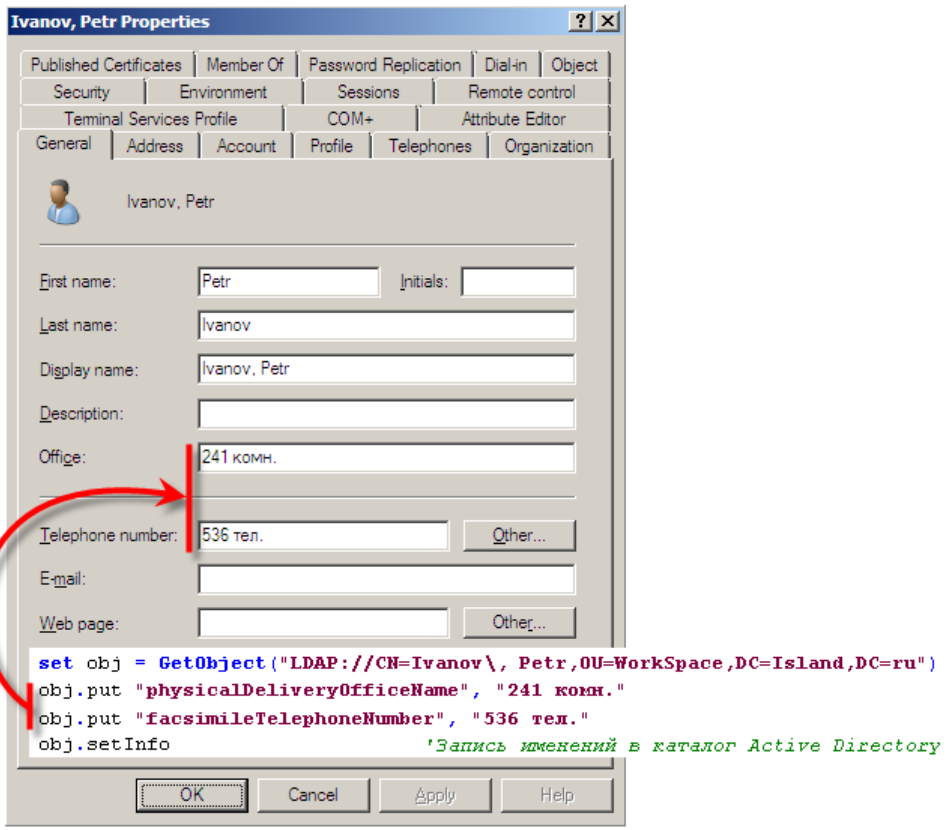



**Рис. В2.** Пример изменения значений атрибутов учетной записи пользователя

Для создания сценария обратимся к *разд. 2 "Принципы программного управления Active Directory"*, который целиком посвящен описанию объектных моделей. В зависимости от выбранного механизма решения обращаемся к *главе 4 "Объектная модель интерфейсов IADs*"* или *главе 5 "Объектная модель ADO DB"*.

В любой из глав необходимо найти сценарий записи данных в учетную запись. В *главе 4* такой сценарий находится в *подразделе "Метод SetInfo()" раздела "Методы интерфейса IADs"* (см. рис. 4.7, продублированный здесь как рис. В2). Таким образом, видоизменив сценарий в соответствии с поставленной задачей, получим код, представленный в листинге В1.

**Листинг В1. Изменение пути к профилю пользователя у одиночной учетной записи**

```
Set obj = GetObject("LDAP://" + path)
obj.Put "profilePath", "\\Island\Profile$\%UserName%"
obj.SetInfo
```

В приведенном листинге предполагается, что составной путь к учетной записи пользователя находится в переменной `path`, а путь к профилю равен \\Island\Profile$\%UserName%.

Чтобы использовать ADO DB для решения той же задачи, нужно обратиться к листингу 3.8 в *подразделе "Пример доступа к Active Directory" раздела "Диалект SQL"* в *главе 3*, который продублирован с небольшими изменениями в листинге В2.

**Листинг В2. Поиск всех учетных записей пользователей в указанной папке**

```
domain = "LDAP://OU=WorkSpace,DC=Island,DC=ru"

'Подключение к каталогу Active Directory
Set obj= CreateObject("ADODB.Connection")
obj.Provider = "ADsDSOObject"
obj.Open "Active Directory Provider"

'Запрос в каталог Active Directory
query = "SELECT distinguishedName FROM '" & domain & "' WHERE " & _
    " objectClass = 'person' and not objectClass = 'computer'"
```

```
'Вызов метода Execute, возвращающего объект Recordset
Set rs = obj.Execute(query)
rs.MoveFirst

Do Until rs.EOF
    path = rs.Fields("distinguishedName ").Value
    rs.MoveNext
Loop
rs.Close
```

В приведенном листинге последовательно определяется составной путь для всех учетных записей, находящихся в корневой папке WorkSpace домена Island.ru.

Для получения листинга, решающего поставленную выше задачу, необходимо в цикл `Do..Loop` листинга В2 вставить листинг В1.

# Какой редактор сценариев использовать?

В настоящее время существует множество редакторов, позволяющих создавать скрипты. Как правило, редакторы такого рода узкоспециализированы, то есть поддерживают всего один или два языка программирования. Редактор Primal Script (**http://www.primalscript.com**), созданный компанией SAPIEN (**http://www.sapien.com**), имеет встроенную поддержку более чем 30 языков программирования, начиная от HTML, XML и VBScript заканчивая узкоспециализированными, такими как KIXTart.

Несмотря на то, что он платный, читатель может бесплатно загрузить с сайта пробную версию, которая будет успешно функционировать в течение 30 дней с момента установки.

Еще одна причина, по которой рекомендуется использовать этот редактор — наличие библиотеки шаблонов, которую можно пополнять самостоятельно. О том, как это сделать, рассказано в *приложении 6*.

# Какова структура книги?

Книга состоит из 18 глав, которые логически сгруппированы в пять разделов.

***Раздел 1 "Основы построения Active Directory"*** объединяет главы, содержащие информацию об устройстве каталога Active Directory.

*Глава 1 "Внутреннее устройство каталога Active Directory"* описывает основы построения Active Directory. Рассматриваются существующие службы каталогов, основные объекты и соответствующие им типы данных. Приводятся ссылки на нормативные документы.

*Глава 2 "Установка домена Windows 2003/2008"* знакомит читателя с процессом создания домена Active Directory двумя способами — вручную и автоматически. Подробно рассмотрена работа мастера Active Directory Installation Wizard и описаны основные поля в файле ответов, который используется для автоматической установки.

***Раздел 2 "Принципы программного управления Active Directory"*** целиком посвящен описанию объектных моделей, используя которые можно получить доступ к объектам каталога Active Directory.

*Глава 3 "Основы программного управления Active Directory"* рассказывает об поддерживаемых объектах. Подробно освещен вопрос, касающийся способов доступа к каталогу, определения типа данных считываемой переменной.

*Глава 4 "Объектная модель интерфейсов IADs*"* включает в себя обзор объектной модели интерфейсов IADs*, который сопровождается многочисленными примерами. Рассказывается о манипуляции с объектами: создание, удаление, перемещение, копирование, изменение и чтение свойств объектов.

*Глава 5 "Объектная модель ADO BD"* расскажет читателю об объектной модели ADO — альтернативном способе доступа к каталогу Active Directory. Так же, как и в предыдущей главе, в ней присутствует много наглядных примеров.

***Раздел 3 "Объекты Active Directory"*** содержит главы, в которых описаны принципы управления основных объектов каталога Active Directory: пользователь, группа, папка и др. А также виртуальный объект RootDSE.

*Глава 6 "Виртуальный объект RootDSE"* посвящена описанию объектной модели RootDSE, предназначенной для получения доступа к пространству имен домена.

*Глава 7 "Атрибуты основных объектов Active Directory"* содержит описание объектных моделей основных объектов каталога Active Directory. К ним относятся учетные записи пользователей, групп, компьютеров и контейнеров. Атрибуты, характерные для всех учетных записей объектов, приведены отдельно, чтобы не путать читателя.

*Глава 8 "Управление учетной записью пользователя"* расскажет читателю об управлении учетной записью пользователя с помощью мастера и сценариев. Приводится сопоставление названий полей мастера и соответствующих им атрибутов в каталоге Active Directory.

*Глава 9 "Управление учетной записью группы безопасности"* посвящена принципам управления учетной записью группы безопасности с помощью мастера и сценариев. В ней также приведены сопоставления названий полей мастера и атрибутов Active Directory, описаны нюансы управления объектом.

*Глава 10 "Управление контейнером"* познакомит читателя с принципами управления контейнером Active Directory с помощью мастера и сценариев. Приведены сопоставления названий полей мастера и атрибутов Active Directory, описаны нюансы управления этим объектом.

*Глава 11 "Управление учетной записью компьютера"* содержит обзор принципов управления учетной записью компьютера в Active Directory с помощью мастера и сценариев. Сопоставлены названия полей мастера и атрибуты Active Directory, описаны нюансы управления объектом.

***Раздел 4 "Безопасность"*** посвящен управлению безопасностью объектов Active Directory и файловой системы.

*Глава 12 "Идентификаторы безопасности Active Directory"* рассказывает о различных идентификаторах безопасности (GUID, SID). Читатель познакомится с принципами их построения и алгоритмами их преобразования из бинарного значения в общепринятый вид.

*Глава 13 "Программное управление безопасностью файловой системы"* содержит информацию о существующих файловых системах, их достоинствах и недостатках. Особое внимание уделено файловой системе NTFS, проблемам безопасности и программному управлению доступом к ресурсам.

***Раздел 5 "Управление компьютером"*** посвящен программному управлению как серверами, так и рабочими станциями.

*Глава 14 "Реестр Windows 2k"* содержит описание основ построения реестра, основных подходов программного управления, групповых политик.

*Глава 15 "Windows Script Host"* целиком посвящена объектной модели Windows Script Host (WSH). Здесь читатель найдет описание методики получения информации о дисках, чтения и записи файлов, управления принтерами. Теоретические данные сопровождаются множеством сценариев, иллюстрирующих описание объектной модели.

*Глава 16 "Загрузка операционной системы"* познакомит читателя с процессом загрузки операционной системы Windows 2k. Изучив ее, читатель сможет решать проблемы, возникающие на этапе загрузки компьютера.

*В главе 17 "Особенные файлы Windows"* рассказано о некоторых специфических файлах, используемых в любой версии Windows 2k: файле подкачки и файле Autorun.inf. В этой главе приведено описание пакетных файлов (CMD, BAT), позволяющих автоматизировать некоторые процессы.

*Глава 18 "Windows Installer"* позволит узнать, как самостоятельно изменять дистрибутивы программ, построенных на основе Windows Installer, чтобы обеспечить их автоматическую установку в заданном режиме.

Помимо основного материала книга содержит ряд приложений, в которых собрана справочная информация, часто необходимая администратору.

# Благодарности

Создание этой книги — труд не одного дня. В ней сосредоточены знания, которые я накапливал в течение 6 лет. Хочу поблагодарить Анатолия Пуртова, Филиппа Зыкова, Юрия Тачилина и, конечно, мою жену — Елену Ходову за неоценимую помощь в создании этой книги и терпение.

Отдельное спасибо — Игорю Шишигину, который подал идею создания этой книги, и Юрию Якубовичу — замечательному редактору.



# Раздел 1

# Основы построения Active Directory

Глава 1



# Внутреннее устройство каталога Active Directory

В этой главе кратко описываются основы построения Active Directory. Рассматриваются основные службы каталогов, подробно рассказывается об объектах и соответствующих им типах данных. Приводятся нормативные документы, касающиеся структуры каталога Active Directory и основных правил доступа к нему.

## Основные термины и понятия

- **Active Directory** — служба каталогов, обеспечивающая централизованное управление сетью. Она содержит информацию об объектах сети и обеспечивает к ней доступ пользователей, компьютеров и приложений в соответствии с установленными правилами безопасности.
- **Глобальный каталог** — централизованное хранилище информации об объектах дерева или домена.
- **Лес** — объединение одного или нескольких деревьев. Деревья в лесу обеспечивают однородный доступ к схеме и правилам совместной работы объектов.
- **Дерево** — группировка или иерархия одного или нескольких доменов Windows 2000, предоставляющих совместный доступ к объектам.
- **Доверительные отношения** — связующее звено между двумя или бóльшим числом доменов. При этом каждый доверяющий домен предоставляет аутентификацию входа в систему доверяемому.
- **Домен** (Windows 2000) — логическая группировка сетевых компьютеров, объединенных общей базой данных каталога, которая содержит учетные записи объектов и правила безопасности.

- **Контроллер домена** — это компьютер под управлением Windows Server, хранящий реплику раздела каталога.
- **Пространство имен** — набор правил именования, обеспечивающих иерархическую структуру.
- **Схема Active Directory** — формальное описание объектов в хранилище Active Directory.

# Нормативная документация по Active Directory

Нормативная документация (RFC, Request for Comments) по Active Directory содержит информацию о внутренней структуре каталога и поддерживаемых протоколов (provider); описание типов данных объектов и т. д.

## Общие сведения

RFC (Request for Comments, запрос комментариев) — пронумерованный информационный документ, содержащий технические спецификации и стандарты, широко применяемые в сети Интернет. Первый RFC опубликован 7 апреля 1969 г. и называется "Host Software". Несмотря на название, запросы комментариев RFC сейчас рассматриваются как стандарты Интернета.

Согласно RFC 2026, жизненный цикл стандарта выглядит следующим образом. Сначала рассматривается *Черновик* (Internet Draft). Он не имеет официального статуса и удаляется из базы через шесть месяцев после последнего изменения. Затем черновик получает статус *Предложенного стандарта* (Proposed Standard) и свой номер RFC. Наличие программной реализации стандарта желательно, но не обязательно. Следующая стадия — *Черновой стандарт* (Draft Standard). В этот документ могут вноситься незначительные поправки. Высший уровень — *Стандарт Интернета* (Internet Standard). Это спецификации с большим успешным опытом применения. Параллельно с нумерацией RFC такие документы имеют собственную нумерацию STD. Список стандартов имеется в документе STD 1 (RFC 3700). Из более чем трех тысяч RFC этого уровня достигли всего несколько десятков.

Многие старые документы RFC замещены более новыми версиями с измененными номерами или вышли из употребления. Такие документы получают статус *Исторических* (Historic).

RFC — это не только стандарты, но и концепции, введения в новые направления в исследованиях, исторические справки, результаты экспериментов, руководства по внедрению технологий, предложения и рекомендации по развитию существующих стандартов и другие новые идеи в информационных технологиях. Можно условно выделить следующие серии спецификаций:

- *экспериментальные* (Experimental) спецификации содержат информацию об исследованиях в ИТ-сфере, например, ими могут быть прототипы, реализующие новые концепции;
- *информационные* (Informational) RFC предназначены для ознакомления общественности. Они не являются ни стандартами, ни результатом консенсуса или рекомендациями. Некоторые черновики, не получившие статуса Предложенного стандарта, но представляющие интерес, могут быть опубликованы как Информационные RFC;
- серия *лучший современный опыт* (Best Current Practice) содержит рекомендации по реализации стандартов, в том числе от сторонних производителей ПО.

Почти все стандарты разрабатываются под эгидой каких-либо организаций, например, W3C, IETF, консорциума Юникода и др.

## RFC по LDAP v2

- Lightweight Directory Access Protocol (RFC 1777)
- The String Representation of Standard Attribute Syntaxes (RFC 1778)
- A String Representation of Distinguished Names (RFC 1779)
- Using the OSI Directory to Achieve User Friendly Naming(RFC 1781)
- Internet X.509 Public Key Infrastructure Operational Protocols - LDAPv2 (RFC 2559)
- Lightweight Directory Access Protocol version 2 (LDAPv2) to Historic Status (RFC 3494)

## RFC по LDAP v3

- LDAPv3 Protocol (RFC 2251)
- LDAPv3 Attribute Syntax Definitions (RFC 2252)
- LDAPv3 UTF-8 String Representation of Distinguished Names (RFC 2253)

- LDAPv3 String Representation of LDAP Search Filters (RFC 2254)
- LDAPv3 URL Format (RFC 2255)
- A Summary of the X.500(96) User Schema for use with LDAPv3 (RFC 2256)
- Authentication Methods for LDAP (RFC 2829)
- LDAPv3 Extension for Transport Layer Security (RFC 2830)
- LDAPv3 Technical Specification (RFC 3377)
- IANA Considerations for LDAP (RFC 3383)
- Lightweight Directory Access Protocol version 3 (LDAPv3): All Operational Attrib (RFC 3673)

# Особенности и характеристики Active Directory

Active Directory (AD) — это служба каталогов, обеспечивающая централизованное управление сетью. Она содержит информацию о различных объектах в сети и обеспечивает доступ к ним.

Active Directory обладает следующими особенностями.

- *Масштабируемость.* В отличие от большинства других баз данных, которые являются реляционными, — Active Directory является иерархической. В реляционных базах данных взаимосвязи между записями формируются с помощью ключей, которые хранятся совместно с данными. В иерархической базе данных взаимосвязи между записями имеют характер "родитель-потомок": все записи, за исключением корневой, обладают родительской записью. У каждой родительской записи может быть один или несколько потомков. Иерархическая база данных позволяет хранить большое количество объектов, при этом обеспечивается быстрый доступ к необходимым объектам.
- *Поддержка открытых стандартов.* Active Directory включает в себя концепцию пространства имен Интернета со службой каталогов Windows NT, что позволяет объединять и управлять различными пространствами имен в разноразрядных аппаратных и программных средах. Для управления пространством имен Active Directory используется библиотека интерфейса службы активного каталога (Active Directory Service Interface — ADSI).

- *Поддержка стандартных форматов имен.* Active Directory поддерживает несколько форматов имен и позволяет приложениям и пользователям получать доступ к каталогу, применяя наиболее удобный для них формат (табл. 1.1).

***Таблица 1.1.*** *Форматы имен, используемые в Active Directory*

| Формат | RFC | Описание |
|---|---|---|
| UPN | RFC 822 | Формат основного имени пользователя (User Principial Name, UPN). UPN-имена известны как адреса электронной почты. Active Directory обеспечивает "дружественные" имена в этом формате. В качестве имени для регистрации в сети пользователь может использовать как имя учетной записи SAM, так и имя в формате RFC 822. Пример: PIvanov@Island.ru |
| RDN (LDAP) | RFC 1779, RFC 2247 | Имена в формате LDAP URL, также называемые RDN-именами (Relative Distinguished Name), имеют более сложную структуру по сравнению с UPN-именами. Имена LDAP в формате RDN состоят из нескольких частей:<br>• CN расшифровывается как Common Name (общее имя);<br>• OU означает организационную единицу (Organization Unit);<br>• DC — класс объекта домена (Domain Object Class). Часть имени "DC=" обеспечивает подключение к виртуальному объекту RootDSE, с помощью которого осуществляется подключение к домену.<br>Пример: LDAP://ISLAND.RU/CN=PIvanov,OU=WorkSpace или LDAP://CN=PIvanov,OU=WorkSpace,DC=Island,DC=ru |
| UNC | RFC 1123 | UNC (Universal Naming Convention) имеет вид Hoda.Island.ru/WorkSpace/PIvanov. (Здесь Hoda — это имя контроллера домена.) В Active Directory данный формат является путем к объекту в иерархической структуре |

# Active Directory как часть файловой системы

База данных Active Directory по умолчанию хранится на контроллере домена (%WinDir%\System32) в файле Ntds.dit.

База данных имеет следующие особенности:

- размер файла Ntds.dit увеличивается фиксированными порциями, занимая отдельные страницы, что позволяет избежать их разбиения. Поэтому при добавлении объектов размер базы данных Active Directory кажется больше, чем он есть на самом деле;
- файлы базы данных Active Directory всегда открыты;
- размер открытого файла не обновляется. Фактический размер базы данных определяют по отображаемому свободному пространству на диске;
- в подключенном состоянии база данных не может быть дефрагментирована с помощью утилиты ntdsutil.exe.

# Архитектура службы каталогов Active Directory

Чтобы понять, как хранятся и обрабатываются данные в Active Directory, необходимо знать порядок взаимодействия ее компонентов. Службу каталогов можно представить в виде многоуровневой структуры (рис. 1.1).

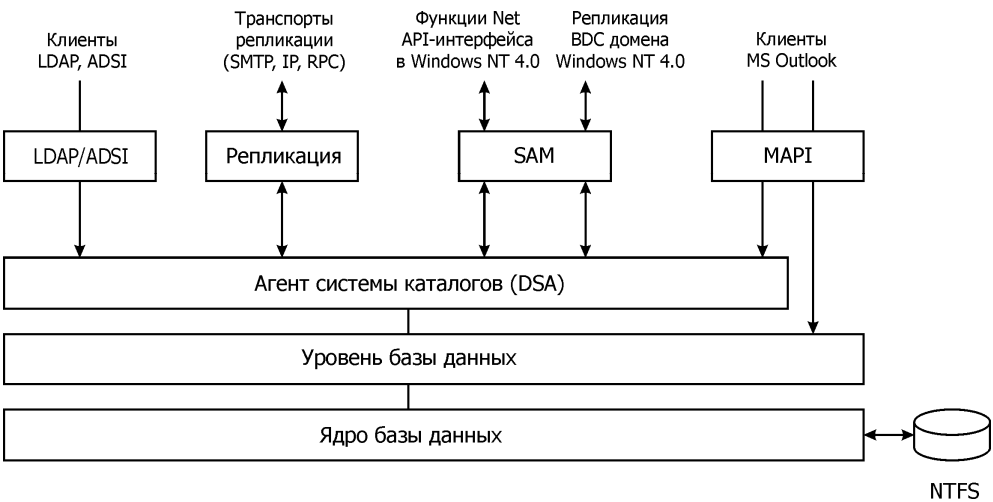



**Рис. 1.1.** Многоуровневая структура службы каталогов

# DNS и Active Directory

Для иерархического именования доменов и компьютеров в Active Directory используется служба DNS (Domain Name System), поэтому объекты доменов и компьютеров являются частью иерархии доменов DNS и Active Directory. Несмотря на то, что имена в обеих системах идентичны, они относятся к разным пространствам имен. Взаимодействие DNS-имен доменов и их IP-адресов в Active Directory реализовано в соответствии с общепринятыми соглашениями об именовании в DNS.

Domain Name System — иерархическая система именования, которая используется для идентификации хостов. Основная функция DNS заключается в прямом и обратном разрешении имен компьютеров в IP-адреса.

База данных DNS представляет собой древовидную структуру, называемую пространством имен доменов (domain name space).

В Windows 2000 полное доменное имя (Fully Qualified Domain Name, FQDN) компьютера состоит из двух частей (рис. 1.2):

1. Имя DNS-узла. Крайняя левая часть FQDN — это полноценное имя DNS-узла, идентифицирующее учетную запись компьютера, хранящуюся в Active Directory. Кроме того, это имя локальной учетной записи компьютера в диспетчере безопасности учетных записей (Security Account Manager, SAM) на рабочей станции или рядовом сервере (не на контроллере домена). По умолчанию имя DNS-узла также используется в качестве NetBIOS-имени. Это делается для совместимости с доменами на основе Windows NT и с рабочими станциями под управлением Windows 9*x*.
2. Основной суффикс имени DNS домена. По умолчанию — это домен Windows, к которому относится данный компьютер.




**Рис. 1.2.** Схема образования полного доменного имени (FQDN)

Кроме DNS-имен компьютеров, контроллеры домена Active Directory идентифицируются по видам предоставляемых ими служб: серверы протокола

LDAP (Lightweight Directory Access Protocol); контроллеры доменов; сервер глобального каталога GC (Global Catalog). Получив указание на имя домена и службу, DNS-сервер ищет контроллер со службой нужного типа в данном домене.

## Службы каталогов Active Directory

Существует несколько уровней служб, интерфейсов и протоколов, которые образуют полный спектр служб для управления каталогом. Три уровня служб содержат все данные для нахождения записей в базе данных каталога. Выше уровня служб находятся протоколы и API-интерфейсы, обеспечивающие взаимодействие при репликации или между клиентами и службами каталогов. Репликация выполняется между службами каталогов.

Основные службы-компоненты Active Directory (см. рис. 1.1):

- *агент системы каталогов* (Directory System Agent, DSA) формирует иерархию каталога на основе отношений "родитель-потомок" и обеспечивает интерфейс прикладного программирования (Application Programming Interfece, API) для запросов на доступ к каталогу;
- *уровень базы данных* — промежуточный уровень абстракций между базой данных и приложениями;
- *ядро базы данных* (Extensible Storage Engine, ESE), работающее непосредственно с записями хранилища каталогов, различает объекты по атрибуту относительно составного каталога;
- *хранилище данных* — файл базы данных Ntds.dit. С этим файлом работает только ядро базы данных. Обращаться напрямую к нему можно с помощью программы ntdsutil.exe, которая находится в папке Support/Tools на диске с дистрибутивом операционной системы Windows 200x Server.

Клиенты могут получить доступ к Active Directory, используя один из следующих механизмов:

- LDAP/ADSI — клиенты, поддерживающие протокол LDAP (Lightweight Directory Access Protocol), используют его для доступа к агенту системы каталогов. Интерфейсы службы каталогов Active Directory (Active Directory Service Interface — ADSI) служат для абстрагирования от LDAP интерфейса прикладного программирования (API), представляя COM-

интерфейсы для взаимодействия с Active Directory. Однако нужно помнить, что в Active Directory используется только LDAP;

- MAPI — при обмене сообщениями и коллективной работе клиенты MS Outlook подключаются к агенту системы каталогов, используя механизм вызова удаленных процедур MAPI (Messaging Application Programming Interface) посредством интерфейса доступа к адресной книге;
- SAM — клиенты MS Windows NT 4.0 и более ранних версий, Windows 9x подключаются к агенту системы каталогов (DSA) через SAM (Security Account Manager);
- REPL — в процессе репликации каталогов агенты системы каталогов (DSA) Active Directory взаимодействуют через интерфейс RPC (Remote Procedure Call).

Существуют четыре способа доступа к Active Directory (см. рис. 1.1). С точки зрения программного управления Active Directory, системного администратора интересует только ADSI. ADSI поддерживает такие языки программирования, как C/C++, VB/VBScript, JScript, WSH. Он представляет объекты Active Directory в виде COM-объектов, а управление осуществляется с помощью COM-интерфейсов. Провайдеры ADSI отображают объекты ADSI в соответствующие пространства имен, то есть они преобразуют вызовы COM-интерфейсов к запросам API конкретной службы каталогов.

## Глобальный каталог

Глобальный каталог (Global Catalog, GC) — это контроллер домена, в котором существуют три доступных для записи раздела: хранилища домена, схемы и конфигурации (листинг 1.1 и рис. 1.3). Каталог автоматически создается при репликации Active Directory. Все разделы каталогов на сервере глобального каталога хранятся в одной базе данных каталога Ntds.dit. Глобальный каталог хранит сведения обо всех лесах, поэтому его можно использовать для поиска любых объектов в лесу без переадресации на другие серверы. Если запрос осуществляется по порту 389 (стандартный порт протокола LDAP), то в случае неудачного поиска запрос будет последовательно передаваться другим контроллерам домена. Если обращение идет по стандартному порту глобального каталога (GC) 3268, то поиск осуществляется по всем разделам

леса. Для безопасного доступа к службам следует использовать порты, применяющие SSL (табл. 1.2).

***Таблица 1.2.*** *Описание портов, поддерживающихся ADSI*

| Порт | Описание |
|---|---|
| 389 | Порт для открытых запросов LDAP |
| 636 | Порт для запросов LDAP с использованием протокола SSL |
| 3268 | Порт для открытых запросов GC |
| 3269 | Порт для запросов GC с использованием протокола SSL |

**Листинг 1.1. Получение имени глобального каталога**

```
temp = ""
Set obj = GetObject("GC:")
For Each el In obj
    temp = temp & el.name
Next
MsgBox temp
```




**Рис. 1.3.** Определение доступных имен глобального каталога

Сценарий определения имени глобального каталога, приведенный в листинге 1.1, работает по следующему алгоритму: в первой строке неявно объявляется пустая переменная `temp`, в которую будут накапливаться доступные имена глобального каталога. Затем с помощью функции `GetObject()` получают доступ к глобальному каталогу с помощью провайдера GC. С третьей по пятую строку осуществляется считывание доступных пространств имен глобального каталога. На последней строке с помощью команды `MsgBox` осуществляется вывод информации на экран.

# Провайдеры ADSI

Интерфейс ADSI поддерживает несколько провайдеров, позволяющих осуществлять программное администрирование. Их список приведен в табл. 1.3.

***Таблица 1.3.*** *Поддерживаемые ADSI-провайдеры*

| Название | Протокол | Описание |
|---|---|---|
| LDAP Provider | "LDAP:" | Администрирование Active Directory, Microsoft Exchange Server |
| WinNT Provider | "WinNT:" | Администрирование Windows NT, Windows 200x, Windows XP |
| NDS Provider | "NDS:" | Администрирование Novell NetWare Directory Service |
| NWCOMPAT | "NWCOMPAT:" | Администрирование Novell NetWare 3.1 |
| IIS | "IIS: | Протокол IIS предназначен для управления WWW- и FTP-узлами по протоколу HTTP |

В зависимости от установленных на сервере компонентов список доступных провайдеров может отличаться от приведенного в табл. 1.3. Для определения всех доступных провайдеров в сети используют службу ADs. С помощью функции `GetObject()` получают доступ к коллекции провайдеров (листинг 1.2 и рис. 1.4).

**Листинг 1.2. Определение доступных в домене служб ADs**

```
temp = ""
Set obj = GetObject("ADs:")
For Each el In obj
    temp = temp & el.name & vbNewLine
Next
MsgBox temp
```

Из всех перечисленных провайдеров мы рассмотрим только LDAP. Провайдер LDAP выполняется на клиенте ADSI и обеспечивает доступ к Active Di-

rectory. Кроме служб каталогов Active Directory Windows 2k, провайдер LDAP обеспечивает доступ к Microsoft Exchange Server, Microsoft Commercial Internet System (MCIS) Address Book Server и т. д.




**Рис. 1.4.** Доступные в домене службы ADs

Для программного управления Active Directory чаще всего используется именно LDAP. В качестве языка программирования обычно выбирают VBScript. Компания Microsoft специально для этих целей разработала язык программирования и интегрировала его во все современные версии Windows. В настоящее время появился альтернативный инструмент — командная оболочка PowerShell.

## Схема объектной модели ADSI

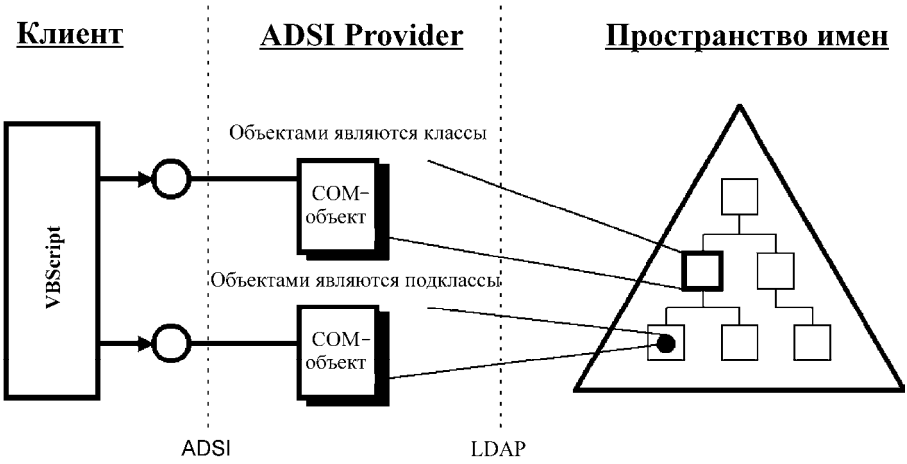



**Рис. 1.5.** Схема объектной модели ADSI

Схема объектной модели ADSI состоит из трех частей (рис. 1.5). Используя клиент — язык программирования, — сценарий получает доступ к COM-объектам. Объекты могут быть классами или подклассами. Взаимодействие между ними не показано, чтобы не загромождать рисунок. С помощью COM-объекта через протокол LDAP сценарий получает доступ к указанному объекту доступного пространства. Пространство имен условно обозначено треугольником; квадратами обозначены классы, а подклассы — кругами.

# Объекты в Active Directory

## Доступ к объектам Active Directory

Доступ к объектам Active Directory можно получить с помощью семейства интерфейсов `IADs*` (листинг 1.3а) или ADODB-соединения (листинг 1.3б). В зависимости от поставленных задач необходимо использовать тот или иной способ.

**Листинг 1.3а. Шаблон использования функции `GetObject()`**

```
'Использование функции GetObejct()
Set obj = GetObject("LDAP://PATH_TO_OBJECT")
MsgBox obj.PROPERTY
```

**Листинг 1.3б. Шаблон использования ADODB-соединения**

```
'Создание ADODB-соединения
Set objConn = CreateObject("ADODB.Connection")
Set objCom = CreateObject("ADODB.Command")
objConn.Provider = "ADsDSOObject"
objConn.Open "Active Directory Provider"
Set objCom.ActiveConnection=objConn
'составление и обработка SQL-запроса
Query = "SELECT PROPERTY FROM 'LDAP://PATH_TO_OBJECT'"
Set st = objConn.Execute(Query)
'Вывод результата типа данных переменной
MsgBox = st.Fields("PROPERTY").Value
```

Листинг, в котором использовано ADODB-соединение, в несколько раз длиннее аналогичного сценария, написанного с использованием `IADs*`-интерфейсов. Основная причина, по которой ADODB-соединения все же используются — это предоставляемая ими возможность поиска объектов по заданным критериям. Функция `GetObject()`, относящаяся к семейству `IADs*`-интерфейсов, может быть использована для поиска объектов, однако ее возможности очень ограничены и их практически не используют. Тем не менее, преимущества сценариев, созданных на основе `IADs*`, нельзя сбрасывать со счетов, хотя бы потому, что обращение к виртуальному объекту RootDSE для определения имени текущего домена невозможно с помощью ADODB-соединения.

Со времен Windows NT семейство интерфейсов `IADs*` было модернизировано. Уже в Windows 2k появились новые интерфейсы, позволяющие эффективно и быстро управлять объектами каталога Active Directory.

В книге будут параллельно рассмотрены оба варианта подключения к каталогу Active Directory.

## Типы объектов Active Directory

В Active Directory поддерживается несколько типов объектов, каждый из которых имеет индивидуальный набор свойств. Для доступа к каталогу используется провайдер LDAP. В табл. 1.4 перечислены характерные объекты Active Directory. Для совместимости доменов, построенных на основе Windows NT и Windows 2k, Active Directory обеспечивает доступ к некоторым из них с помощью провайдера WinNT.

***Таблица 1.4.*** *Типы объектов Active Directory*

| Тип объекта | Создается с Active Directory | Поддержка WnNT | Описание |
|---|---|---|---|
| RootDSE | Да | Нет | Виртуальный объект, с помощью которого обеспечивается доступ ко всем объектам каталога Active Directory |
| Computer | Да | Да | Учетная запись рабочей станции, зарегистрированной в домене |

***Таблица 1.4*** *(окончание)*

| Тип объекта | Создается с Active Directory | Под-держка WnNT | Описание |
|---|---|---|---|
| Contact | Нет | Нет | Контакт, использующийся в почте. Созданные контакты отображаются в почтовом клиенте, например, в Microsoft Oulook |
| Group | Да | Да | Группа безопасности в Active Directory |
| InetOrgPerson | Нет | Нет | Учетная запись пользователя, исполь-зуемая Active Directory (LDAP) |
| OU | Да | Нет | Папка (OU, Organizational Unit) в Active Directory |
| SharedFolder | Нет | Да | Сетевая папка, опубликованная в Active Directory |
| User | Да | Да | Учетная запись пользователя, исполь-зуется для совместимости с Windows NT |
| Printer | Нет | Да | Сетевой принтер, опубликованный в Active Directory |

## Типы данных объектов в Active Directory

Active Directory использует 15 различных типов данных, которые описаны в табл. 1.5. Несмотря на их многообразие, при получении значений все параметры либо автоматически, либо вручную преобразуются к строке или представляют собой массив строк. Например, значение типа `Integer` автоматически преобразуется в строку, состоящую из последовательности символов-цифр (`char`).

***Таблица 1.5.*** *Типы данных в Active Directory*

| LDAP | Тип данных ANSI | RFC | Описание |
|---|---|---|---|
| Boolean | Boolean | 2252, п. 6.4 | Булево значение: `TRUE` / `FALSE` |

***Таблица 1.5*** *(окончание)*

| LDAP | Тип данных ANSI | RFC | Описание |
|---|---|---|---|
| Enumeration | Integer | 2252, п. 6.16 | Целое положительное число. Используется для нумерации |
| Integer | Integer | 2252, п. 6.16 | Целое число |
| Large Integer | Integer | 2252, п. 6.16 | Длинное целое число. Для чтения используются свойства `HighPart` и `LowPart` |
| Object (DS–DN) | DN | 2252, п. 6.9; 2253 | Составной путь к объекту |
| Object (Presentation Address) | Address | 2252, п. 6.28; 1278 | WWW-адрес |
| Object (Replica Link) | Binary | 2252, п. 6.2 | Бинарный формат записи данных |
| String (IA5) | IA5 string | 2252, п. 6.15 | Строка 7-bit US ASCII |
| String (Numeric) | Numeric String | 2252, п. 6.23 | Число, преобразованное в строку символов |
| String (Object Identifier) | OID | 2252, п. 6.15 | Идентификатор безопасности (SID, GUID) |
| String (Octet) | Binary | 2252, п. 6.2 | Бинарные данные, записанные в виде строки (`0x**`) |
| String (Printable) | Printable String | 2252, п. 6.29 | Строка, предназначенная для вывода на экран или печать |
| String (Unicode) | Directory String | 2252, п. 6.10 | Строка в формате Unicode |
| String (UTC Time) | UTC Time | 2252, п. 6.31 | Строка в формате UTC-Time |
| String (Generalized–Time) | Generalized Time | 2252, п. 6.14 | Время с точностью до 100 мс, прошедшее с 1 января 1601 года |

ГЛАВА 2



# Установка домена Windows 2003/2008

В этой главе рассмотрен процесс создания домена Active Directory двумя способами — вручную и автоматически. Подробно рассмотрена работа мастера Active Directory Installation Wizard и описаны основные поля в файле ответов, который используется для автоматической установки. Приведен синтаксис этого файла с комментариями.

## Основные термины и понятия

- **Active Directory** — служба каталогов, обеспечивающая централизованное управление сетью. Содержит информацию об объектах сети и обеспечивает доступ пользователей, компьютеров и приложений к сети в соответствии с установленными правилами безопасности.
- **DNS** (Domain Name Server) — служба, предназначенная для расшифровки IP-адресов в имена хостов.
- **RootDSE** — виртуальный объект, предназначенный для доступа к вершине пространства имен любого домена.
- **TCP/IP** (Transmission Control Protocol/Internet Protocol) — собирательное название для сетевых протоколов разных уровней, используемых в сетях.
- **Домен** (Windows 2k) — логическая группировка сетевых компьютеров, объединенных общей базой данных каталога, которая содержит учетные записи объектов и правил безопасности.
- **Контроллер домена** — это компьютер под управлением Windows Server, хранящий реплику раздела каталога.

# Подготовка к установке Active Directory

Для успешной установки Active Directory в Windows должны быть соблюдены следующие условия:

- операционная система должна принадлежать к семейству серверных ОС, то есть Windows 2000 Server, Windows 2003 Server или Windows 2008 Server;
- файловая система, на которую устанавливается Active Directory, — NTFS. Это обусловлено тем, что система безопасности в файловой системе FAT32 отсутствует (см. *разд. "Виды файловых систем"* в *главе 13*). При необходимости FAT32 можно преобразовать в NTFS с помощью утилиты convert.exe, входящей в комплект операционной системы;
- компьютер должен иметь доступ к DNS-серверу. Если это условие не выполнено, то мастер установки Active Directory установит службу DNS на компьютере и поместит DNS-базу в Active Directory, а другие варианты установки Active Directory будут недоступны. Рекомендуется хранить DNS-базу в отдельном файле. Для выполнения этого условия необходимо заранее установить и настроить DNS;
- на сервере должны быть установлены драйверы сетевого адаптера и выполнена настройка протокола TCP/IP. Для сервера рекомендуется назначить статический IP-адрес. Если DNS-сервер присутствует в сети, его адрес также необходимо указать в свойствах протокола TCP/IP.

# Запуск мастера установки Active Directory

Основная функция мастера Active Directory Installation Wizard — настройка сервера для его функционирования в качестве контроллера домена, в зависимости от текущего состояния Active Directory. Если в сети конфигурируется первый контроллер домена, то при повышении статуса сервера до уровня контроллера домена создается новый каталог Active Directory. Настраиваемый компьютер будет содержать первый домен первого дерева первого леса.

Рассмотрим процесс установки каталога Active Directory на Windows Server 2003. Как и в предыдущих версиях Windows, запуск мастера осуществляется запуском утилиты dcpromo.exe.

Мастер может работать в трех режимах:

- в *стандартном* режиме работа мастера состоит из нескольких шагов. На каждом из них администратор задает исходные данные, касающиеся конфигурации домена. Затем на основе представленной информации осуществляется установка и настройка Active Directory;
- в *расширенном* режиме имеется возможность выбирать контроллер домена, с которого будет выполняться репликация;
- в *автоматическом* режиме настройки берутся из файла ответов. Все ответы на задаваемые мастером в обычном или расширенном режимах вопросы хранятся в файле ответов. Использование файла ответов экономит системному администратору время и снижает влияние человеческого фактора.

О работе мастера в расширенном и обычном режиме можно прочитать в любом справочнике системного администратора. Остановимся подробнее на установке Active Directory в автоматическом режиме.

Утилита dcpromo.exe может быть запущена со следующими ключами:

- `/unattend:<path to file>` позволяет запустить мастера с файлом ответов;
- `/adv` включает расширенный режим работы мастера;
- `/?` заставляет утилиту вывести справку и закончить выполнение.

# Синтаксис файла ответов утилиты *dcpromo.exe*

Для автоматизации работы мастера рекомендуется создать файл ответов (answer file), который представляет собой текстовый файл в кодировке ASCII. В нем содержатся значения параметров, считываемые мастером во время его работы (листинг 2.1).

**Листинг 2.1. Шаблон файла ответов создания домена**

```
[DCINSTALL]
InstallDNS = Yes
NewDomain = Forest
NewDomainDNSName = <DNS-имя создаваемого домена>
```

```
DomainNetBiosName = <Сокращенное имя домена>
SiteName = <Имя сайта>
ReplicaOrNewDomain = domain
ForestLevel = <Функциональный уровень леса>
DomainLevel = <Функциональный уровень домена>
DatabasePath = <Локальный путь к базе данных>
LogPath = <Локальный путь к файлу отчетов>
RebootOnCompletion = Yes
SYSVOLPath = <Локальный путь к папке System Volume>
SafeModeAdminPassword = <Пароль для входа учетной записи с
              административными правами в случае сбоя работы домена>
```

В шаблоне фигурирует ряд параметров, описание которых приведено в табл. 2.1.

***Таблица 2.1.*** *Описание параметров файла ответов для создания Active Directory*

| Параметр | Описание |
|---|---|
| `InstallDNS` | Установка службы DNS на сервере. Принимает значение Yes или No. Принимаемое значение зависит от состояния домена. При создании нового леса по умолчанию присваивается значение Yes |
| `NewDomain` | Указывает тип создаваемого домена:<br>• Tree — новый домен является корнем в существующем лесу;<br>• Child — новый домен является дочерним по отношению к существующему;<br>• Forest (значение по умолчанию) — новый домен является первым в новом лесу деревьев домена |
| `NewDomainDNSName` | Новое имя DNS-сервера в формате Fully Qualified Domain Name (FQDN[1]). Пример: island.ru. Значение по умолчанию не определено |

[1] См. RFC 1035 "Domain names — implementation and specification"; RFC 1123 "Requirements for Internet Hosts — Application and Support"; RFC 2181 "Clarifications to the DNS Specification".

***Таблица 2.1*** *(продолжение)*

| Параметр | Описание |
|---|---|
| `DomainNetBiosName` | Сокращенное имя домена. Как правило, ему соответствует первая часть FQDN-имени. Пример: island. Значение по умолчанию не определено |
| `SiteName` | Имя сайта, которое будет назначено в новом лесу. По умолчанию Default-First-Site-Name |
| `ReplicaOrNewDomain` | Режим создания домена. Параметр используется только при создании нового домена и может принимать следующие значения:<br>• Replica — сервер становится дополнительным контроллером домена;<br>• ReadOnlyReplica — сервер конвертируется в RODC (Read-only Domain Controller, контроллер домена, доступный только для чтения);<br>• Domain — сервер преобразуется в первый контроллер домена |
| `ForestLevel` | Значение параметра — число, определяющее функциональный уровень леса в новом домене:<br>• 0 = Windows 2000 Server;<br>• 2 = Windows Server 2003;<br>• 3 = Windows Server 2008.<br>По умолчанию принимает значение 0 |
| `DomainLevel` | Значение параметра — число, определяющее функциональный уровень домена:<br>• 0 = Windows 2000 Server;<br>• 2 = Windows Server 2003;<br>• 3 = Windows Server 2008.<br>Назначаемое значение основывается на функциональном уровне существующего леса в новом домене. По умолчанию значение не определено |

***Таблица 2.1*** *(окончание)*

| Параметр | Описание |
|---|---|
| DatabasePath | Полный путь (Fully Qualified Path Name, но не UNC-путь) к папке на жестком диске локального компьютера, в котором создана база AD DS (файл Ntds.dit). Если папка существует, то она должна быть пуста. Если папка отсутствует на жестком диске — она будет создана. Для успешного создания базы данных необходимо как минимум 200 Мбайт свободного места на жестком диске. Значение по умолчанию %SYSTEMROOT%\Ntds, где Ntds — имя файла базы данных без расширения |
| LogPath | Полный путь (но не UNC-путь) к папке на жестком диске локального компьютера, в котором будет создан журнал событий базы данных Active Directory. Значение по умолчанию %SYSTEMROOT%\Ntds. Пример: C:\Windows\Ntds |
| RebootOnCompletion | Перезагрузка сервера после завершения работы мастера. Принимает значение Yes (по умолчанию) или No |
| SYSVOLPath | Полный путь (но не UNC-путь) к папке Sysvol на жестком диске локального компьютера. Значение по умолчанию %SYSTEMROOT%\Sysvol |
| SafeModeAdminPassword | Пароль для входа в Directory Service Restore Mode в случае нарушения работы контроллера домена. Можно указать либо пароль, либо NONE (по умолчанию). Пустые пароли не допускаются |

Пример файла ответов Island.txt для создания корня домена island.ru на Windows 2003 Server приведен в листинге 2.2.

**Листинг 2.2. Создание файла ответов Island.txt**

```
[DCINSTALL]
InstallDNS = YES
NewDomain = FOREST
NewDomainDNSName = ISLAND.RU
DomainNetBiosName = ISLAND
SiteName = Default-First-Site-Name
ReplicaOrNewDomain = DOMAIN
```

```
ForestLevel = 2
DomainLevel = 2
DatabasePath = %systemroot%\NTDS
LogPath = %systemroot%\NTDS
RebootOnCompletion = YES
SYSVOLPath = %systemroot%\SYSVOL
SafeModeAdminPassword = NONE
```

# Описание процесса установки Active Directory

Рассмотрим параллельно работу мастера в стандартном и автоматическом режимах, основываясь на информации, приведенной в файле ответов Island.txt в листинге 2.2.

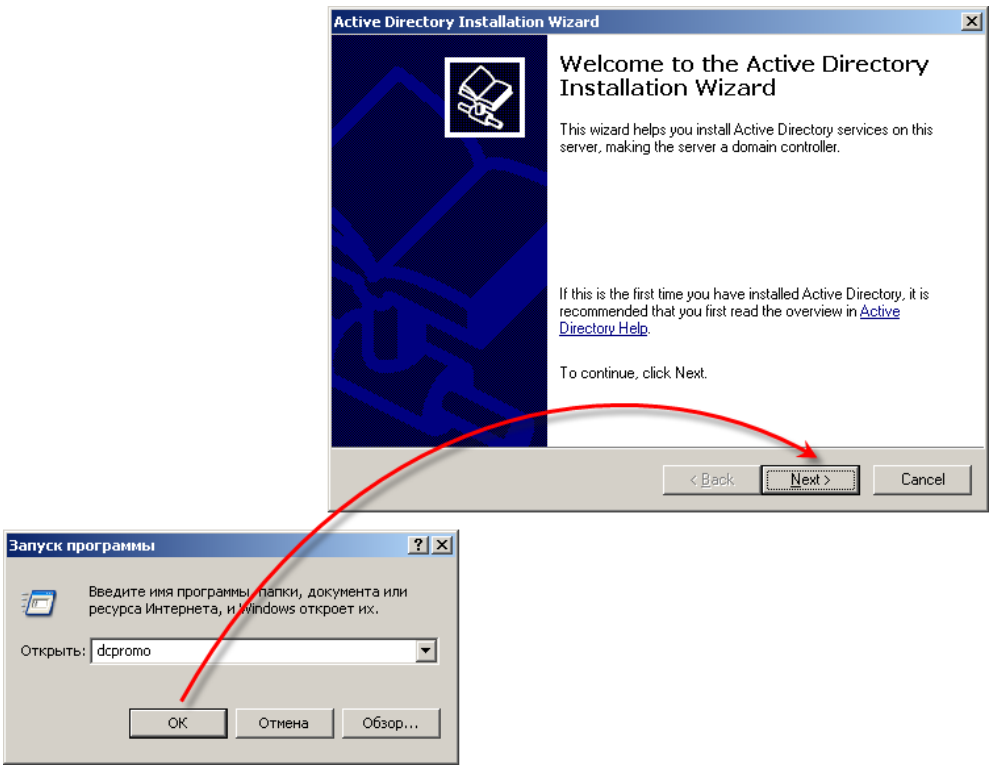



**Рис. 2.1.** Запуск утилиты dcpromo.exe

На **первом** шаге осуществляется запуск мастера вызовом утилиты dcpromo.exe (рис. 2.1) из меню **Пуск | Выполнить**. Для запуска мастера в автоматическом режиме с помощью файла C:\Island.txt используйте команду:

```
dcpromo.exe /unattend:C:\Island.txt
```

На **втором** шаге выводится информация, касающаяся совместимости Windows 2003 Server с другими типами операционных систем с точки зрения безопасности (рис. 2.2).

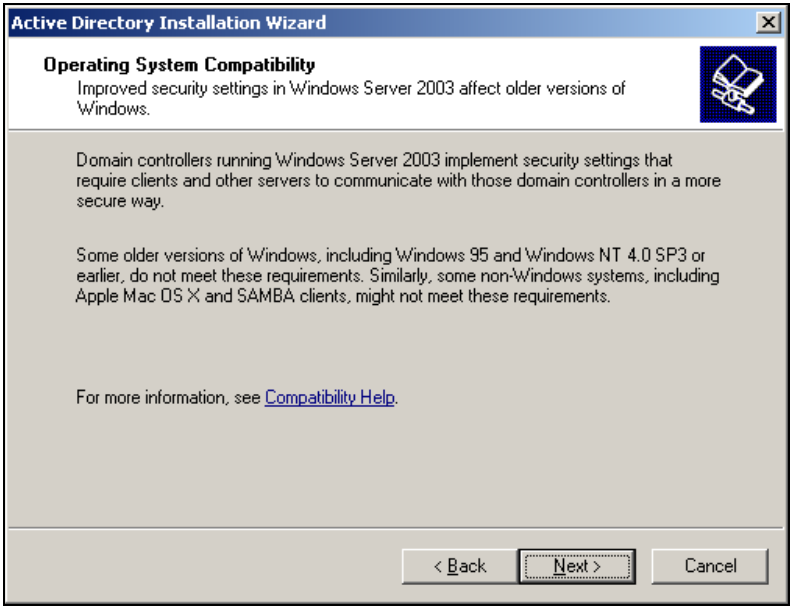



**Рис. 2.2.** Совместимость операционных систем

На **третьем** шаге осуществляется проверка настройки и конфигурации сетевой карты и протокола TCP/IP. Если система проверки обнаружит какие-либо ошибки, то на экран будет выведено соответствующее информационное окно (рис. 2.3).

На **четвертом** шаге (рис. 2.4) определяется тип создаваемого контроллера домена:

- выбор значения **Domain Controller for a new Domain** влечет установку контроллера домена в новом домене. В файле ответов этому параметру соответствует установка `ReplicaOrNewDomain = Domain`;

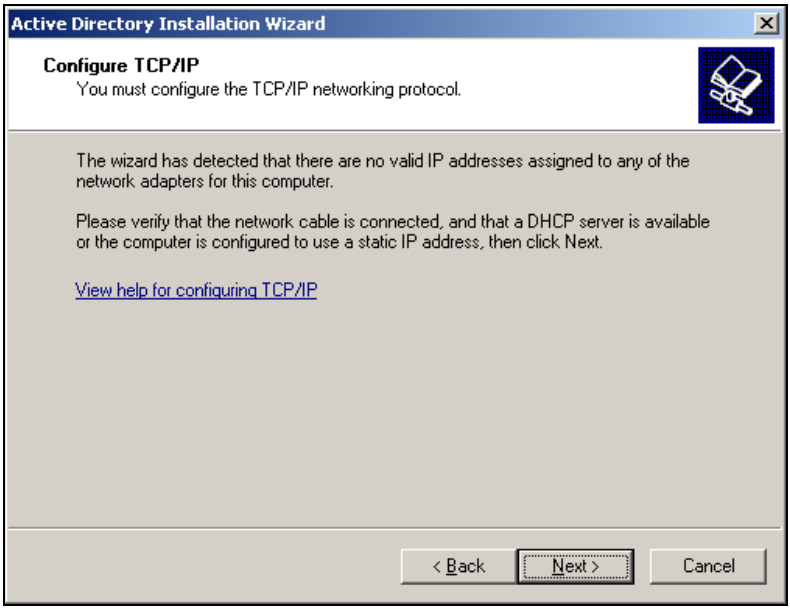



**Рис. 2.3.** Проверка настроек протокола TCP/IP

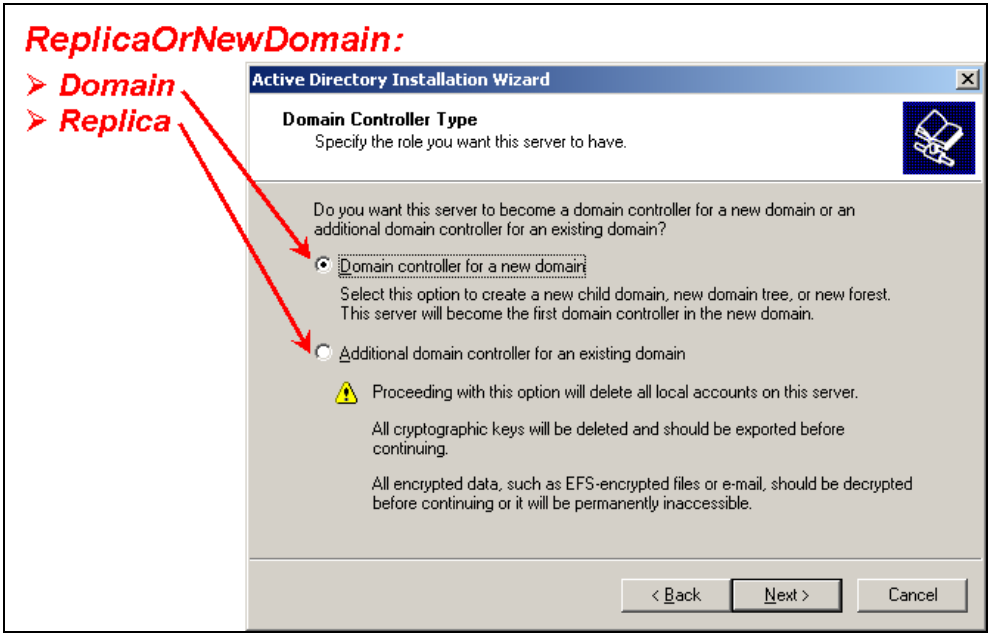



**Рис. 2.4.** Выбор типа контроллера домена

- значение **Additional domain controller for an existing domain** заставляет установить дополнительный контроллер домена в существующем домене. В файле ответов выбору этого параметра соответствует строка `ReplicaOrNewDomain = Replica`.

На **пятом** шаге определяется тип создаваемого домена (рис. 2.5). В файле ответов тип домена устанавливается параметром `NewDomain`, который принимает одно из трех значений (см. табл. 2.1).

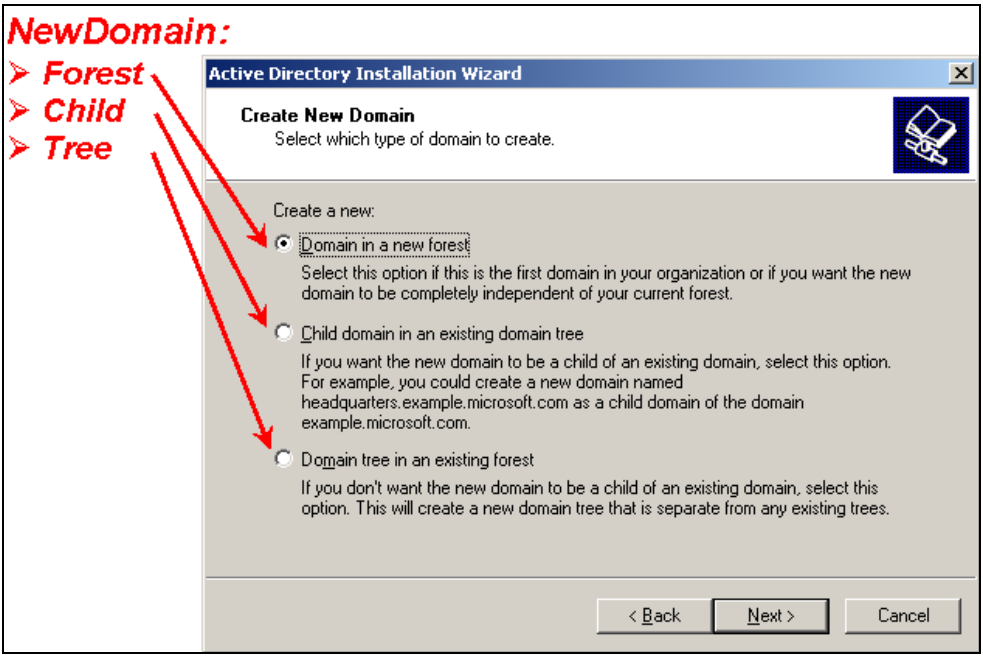



**Рис. 2.5.** Определение типа домена

На следующих двух шагах (**шестом** и **седьмом** шаге) определяются имя домена и соответствующее ему NetBIOS-имя — сокращенное имя домена (рис. 2.6). В файле ответов им соответствуют параметры `NewDomainDNSName` и `NetBIOSName`.

На **восьмом** и **девятом** шагах указывается физическое расположение базы Active Directory, журнала событий и папки SYSVOL. В файле ответов им соответствуют параметры `DatabasePath`, `LogPath` и `SYSVOLPath` (рис. 2.7). Для создания базы необходимо как минимум 200 Мбайт. Если это условие не вы-

полнено, то при создании каталога Active Directory мастер выдаст сообщение об ошибке.

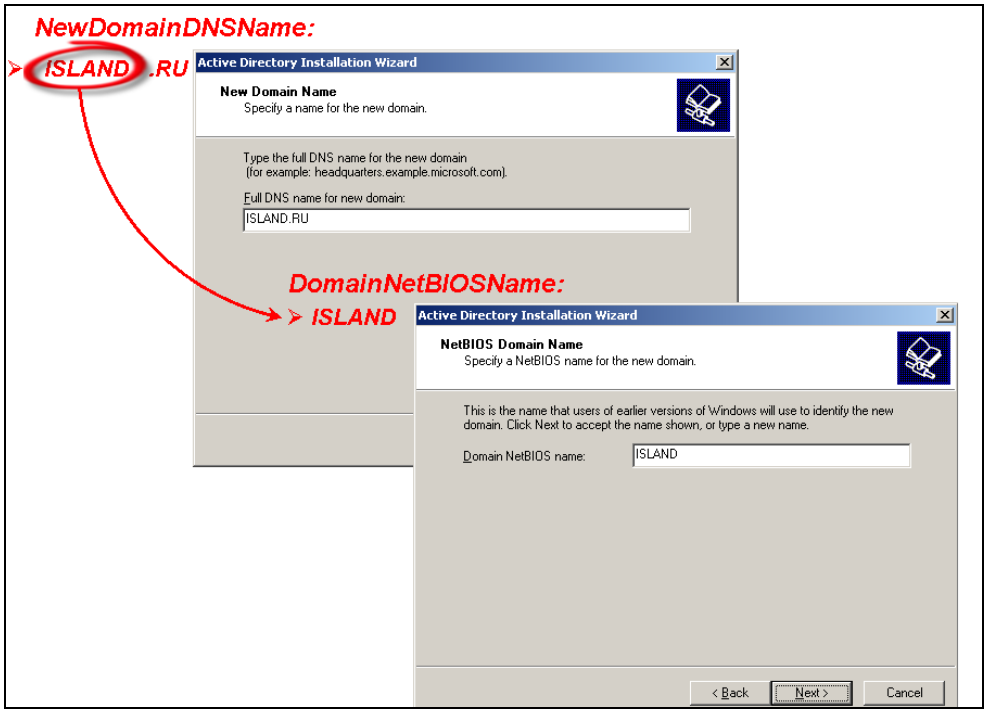



**Рис. 2.6.** Определение DNS- и NetBIOS-имени домена

**На десятом** шаге осуществляется проверка службы DNS и выдается отчет (рис. 2.8). В случае если служба DNS до этого не была установлена и сконфигурирована, нужно настроить ее как в ручном, так и в автоматическом режиме работы мастера. Рекомендуется настраивать службу DNS автоматически, для чего в файле ответов необходимо указать значение Yes для параметра `InstallDNS`.

На предпоследнем, **одиннадцатом** шаге необходимо выбрать режим работы нового домена. С помощью этой настройки администратор определяет, будет ли совместим создающийся домен с доменами на основе Windows NT. По умолчанию выбран второй вариант (рис. 2.9), в котором эта поддержка обеспечена. В файле ответов режим работы устанавливается параметрами `ForestLevel` и `DomainLevel`. Их значения в большинстве случаев совпадают.

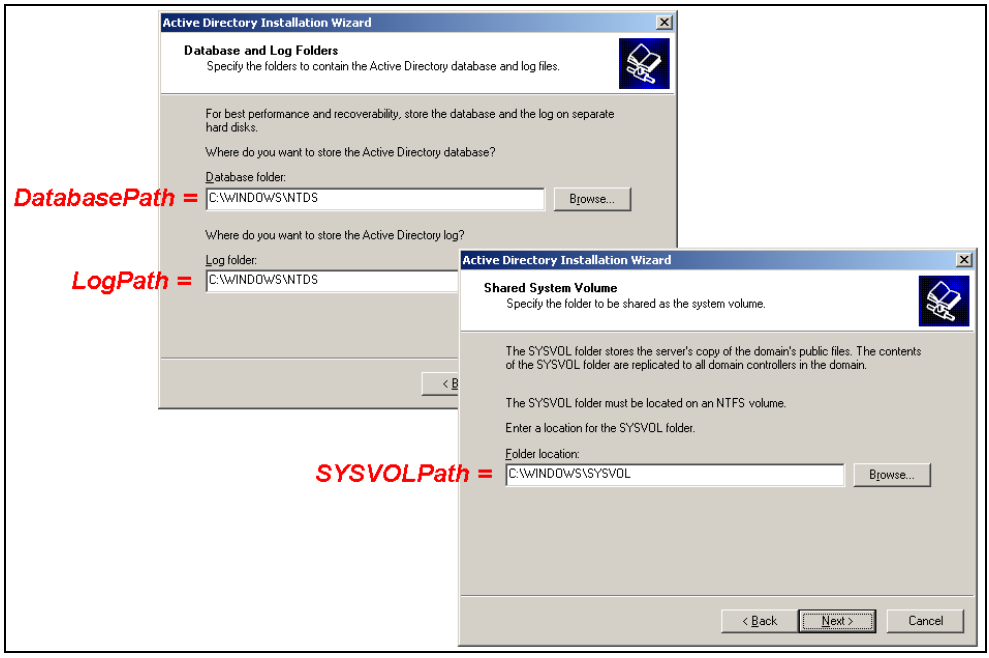



**Рис. 2.7.** Определение физического местоположения базы

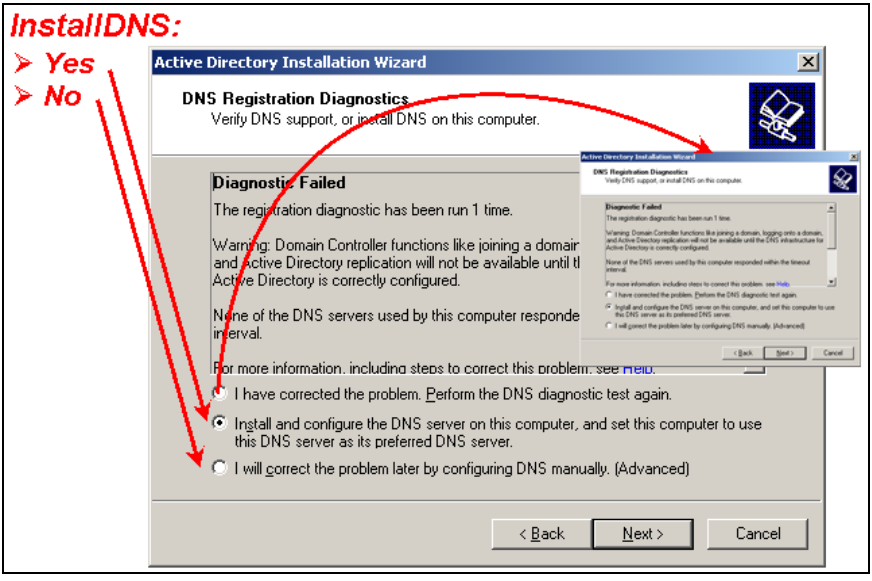



**Рис. 2.8.** Настройка службы DNS

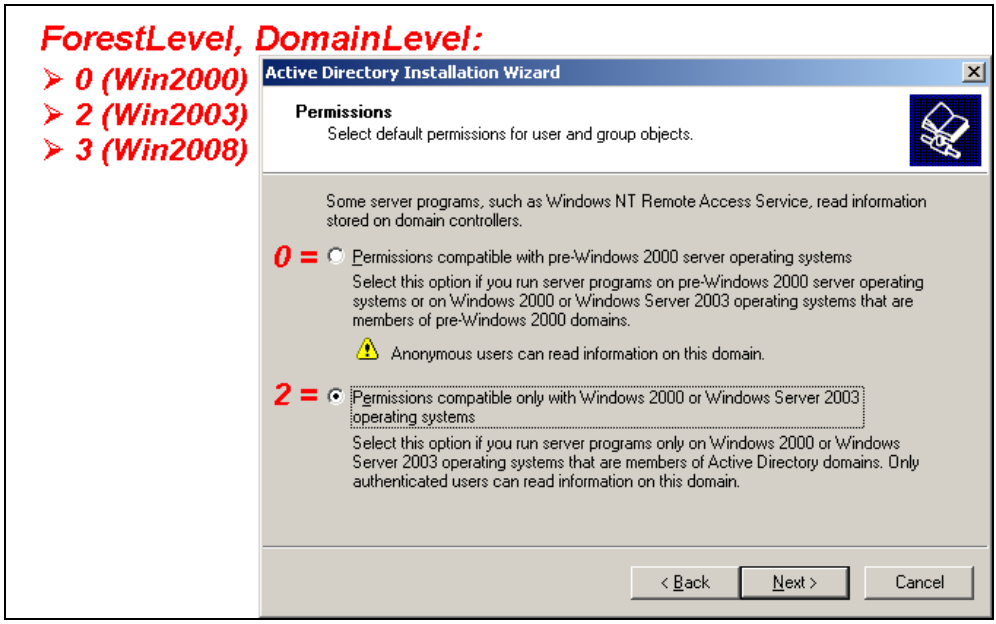



**Рис. 2.9.** Настройка совместимости создаваемого домена с доменами на основе Windows NT

На завершающем, **двенадцатом** шаге необходимо ввести пароль, который будет использован для входа в случае неполадок в режим восстановления домена (рис. 2.10). В файле ответов пароль задается параметром `SafeModeAdminPassword`.

Не рекомендуется назначать какой-либо пароль (поле остается пустым), поскольку он будет храниться в незашифрованном виде. После установки необходимо сразу назначить безопасный пароль для ограничения доступа к контроллеру домена.

После того как все шаги успешно пройдены, мастер выведет диалоговое окно, содержащее отчет о заданных системным администратором параметрах создаваемого домена. После нажатия на кнопку **Next** осуществляется создание базы Active Directory (рис. 2.11). По завершению процесса мастер выведет сообщение об успешном создании домена.

После завершения установки и настройки Active Directory выводится соответствующее диалоговое окно, а затем предложение перезагрузиться (рис. 2.12), чтобы все сделанные изменения вступили в силу.

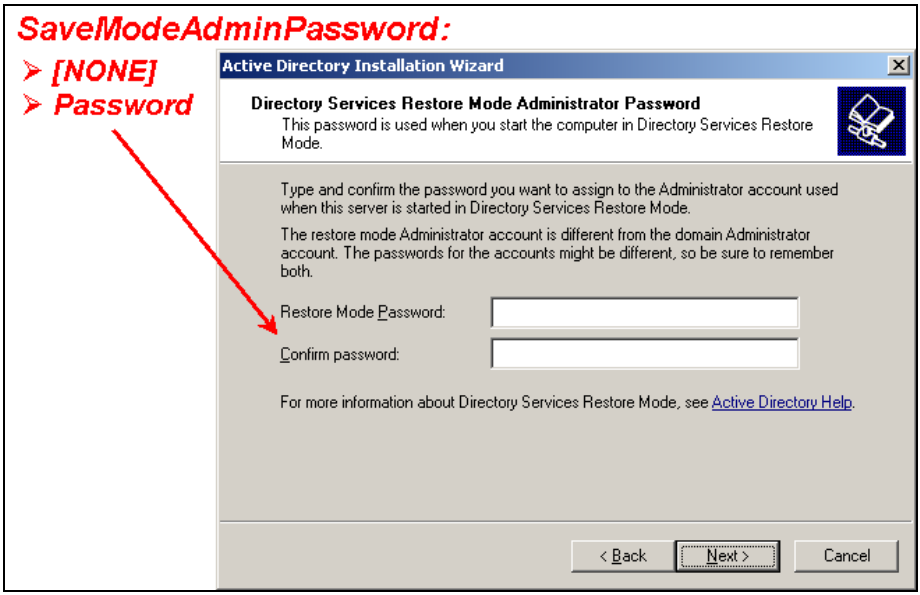



**Рис. 2.10.** Назначение пароля, необходимого для запуска режима восстановления домена

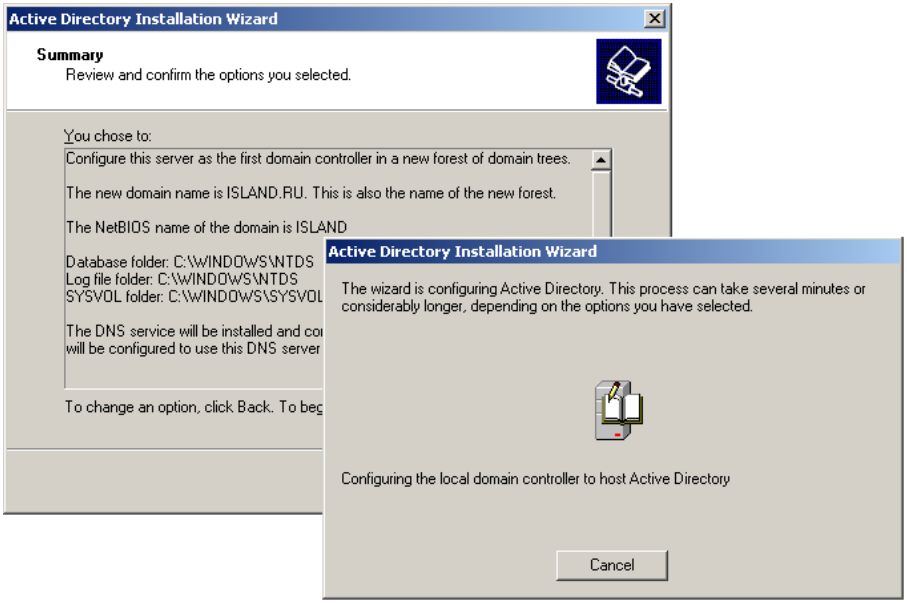



**Рис. 2.11.** Проверка заданных параметров и создание базы Active Directory

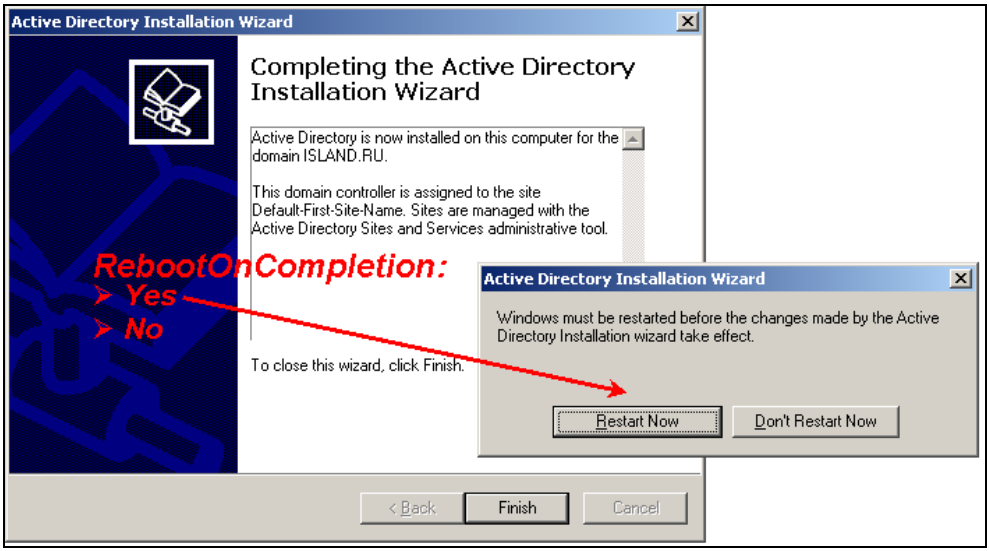



**Рис. 2.12.** Завершение работы мастера настройки Active Directory

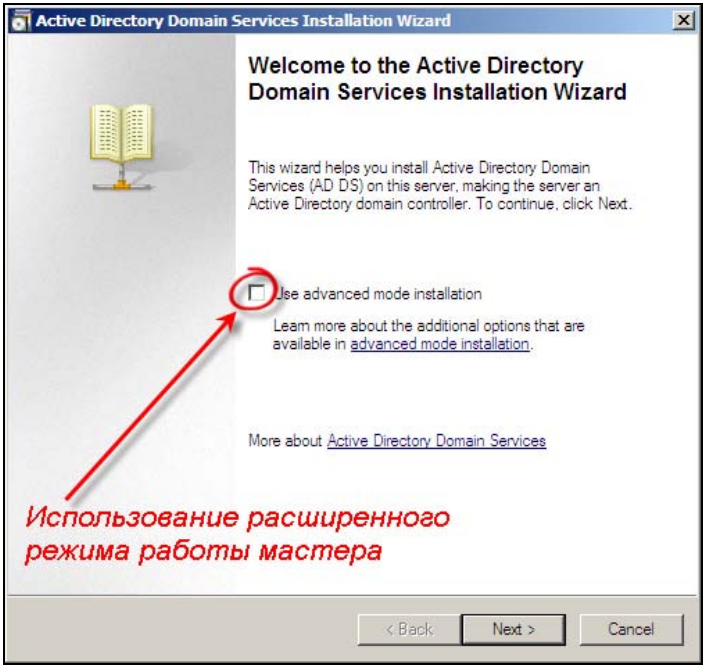



**Рис. 2.13.** Запуск мастера создания каталога Active Directory в расширенном режиме в Windows 2008

## Мастер установки домена Windows 2008

В Windows 2008 мастер создания домена существенно переработан. Во-первых, в новой версии мастера на первом же шаге можно продолжить его работу в расширенном режиме, установив соответствующий флажок (рис. 2.13). Теперь нет необходимости запускать мастер с ключом `/adv`.

Во-вторых, отсутствует седьмой шаг, в котором необходимо задавать NetBIOS-имя нового домена (см. рис. 2.6). Теперь оно определяется по заданному FQDN-имени в автоматическом режиме.

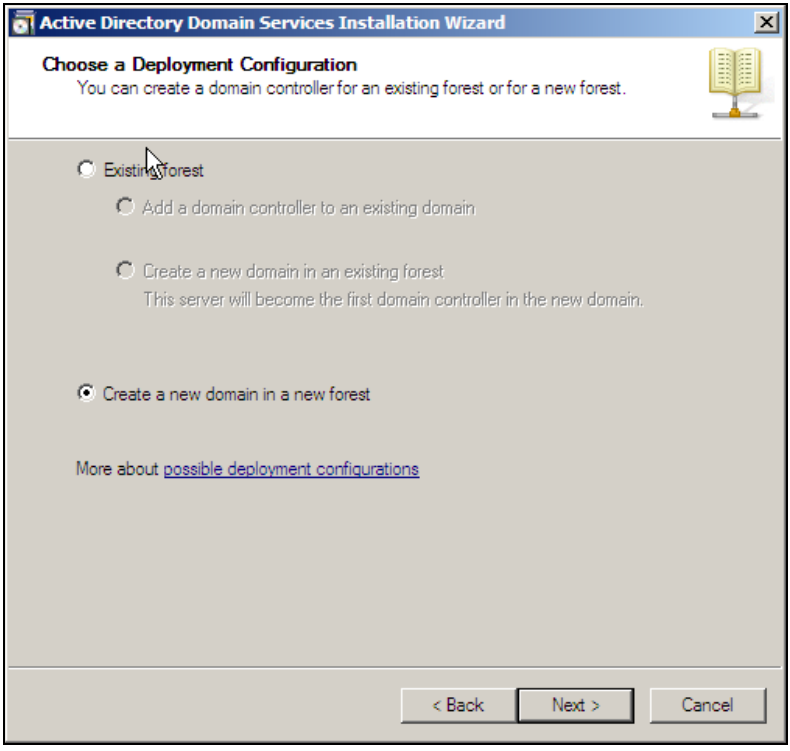



**Рис. 2.14.** Создание нового домена в Windows 2008

В-третьих, оптимизирован процесс выбора роли домена. Два шага работы мастера в Windows 2003 (шаги четыре и пять) объединены в один (рис. 2.14).

В-четвертых, в Windows 2008 появилась новая роль домена — Read-Only Domain Controller (RODC). При выборе этого варианта (рис. 2.15) контроллер домена

предназначен только для чтения. В файле ответа при назначении этой роли новому серверу необходимо указать `ReplicaOrDomain = ReadOnlyReplica`.

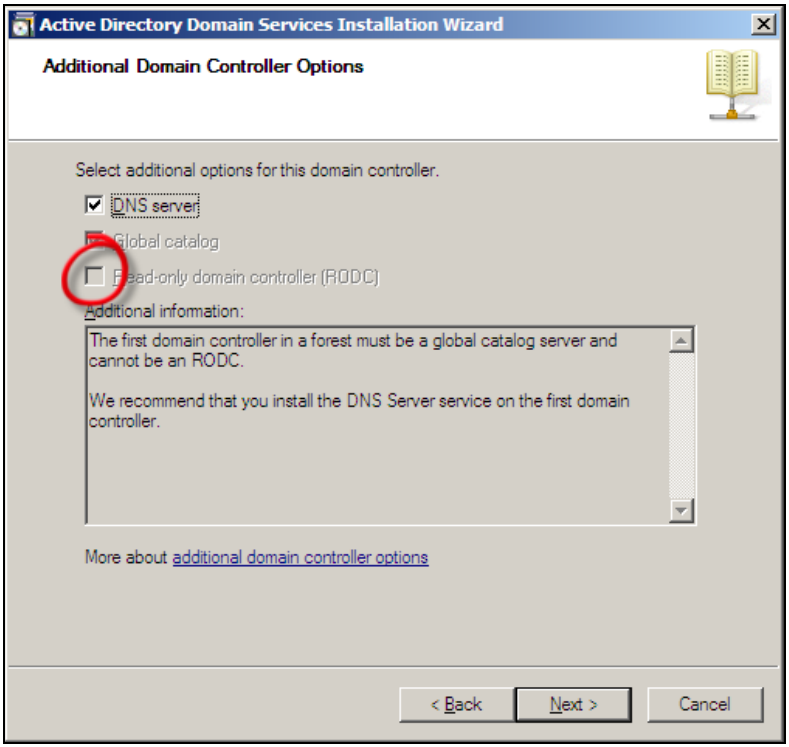



**Рис. 2.15.** Выбор назначаемой серверу роли в Windows 2008



# Раздел 2

# Принципы программного управления Active Directory

## Глава 3



# Основы программного управления Active Directory

В этой главе описаны объекты, их идентификаторы и соответствующие им типы данных в каталоге Active Directory. Рассказано о способах доступа к каталогу, подробно рассмотрен вопрос определения типа данных той или иной переменной.

## Основные термины и понятия

- **ADO** (ActiveX Data Objects) — интерфейс программирования приложений для доступа к данным, основанный на технологии компонентов ActiveX. ADO позволяет представлять данные из разнообразных источников, в том числе и из Active Directory.
- **ADSI** (Active Directory Service Interfaces) — это интерфейс программирования приложений, предназначенный для доступа к различным службам каталогов, в первую очередь к Active Directory.
- **`IADs*`** — набор интерфейсов, который предназначен для работы с любыми ADSI-объектами и описывает их возможности, в том числе поддерживаемые ими свойства и методы.
- **LDAP** (Lightweight Directory Access Protocol) — протокол, используемый для доступа к древовидной базе данных Directory Services.
- **SQL** (Structured Query Language) — универсальный язык структурированных запросов, используемый для создания, модификации и управления базами данных. SQL является информационно-логическим языком, а не языком программирования.

- **UNC** (Universal Naming Convention) — полное имя ресурса в сети, соответствующее соглашению об универсальном назначении имен и включающее в себя имена сервера и совместно используемого ресурса; для каталогов или файлов имя может также включать полный путь к ним.

# Идентификаторы Active Directory

Управление объектами Active Directory осуществляется с помощью провайдера LDAP, который предоставляет доступ к объектам каталога. Каждый объект имеет набор свойств, характерный исключительно для объектов данного типа. Группы объектов характеризуются идентификатором, по которому логически разделяют объекты. Это позволяет упростить программное управление Active Directory. Существует несколько типов идентификаторов объектов: CN, DC, OU (табл. 3.1).

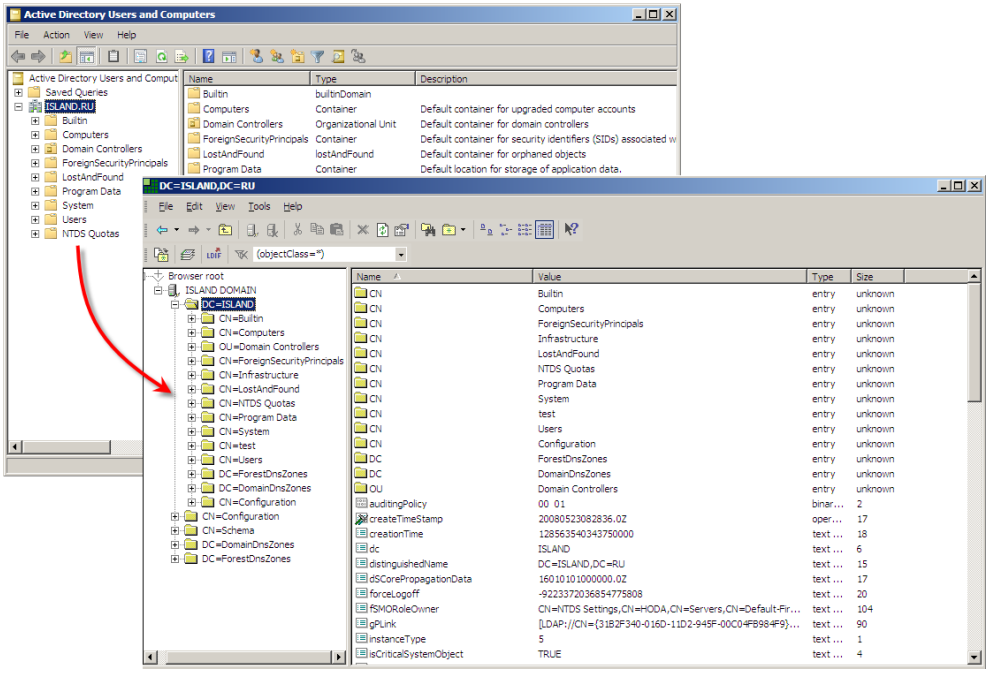



**Рис. 3.1.** Идентификаторы Active Directory

***Таблица 3.1.*** *Типы идентификаторов объектов провайдера LDAP*

| Сокращение | Описание |
|---|---|
| DC (Domain Component) | Метка доменного имени |
| OU (Organization Unit) | Папка — организационная единица; идентификаторы учетных записей, создаваемые в Active Directory по умолчанию |
| CN (Common Name) | Идентификатор учетной записи пользователя или группы безопасности |

В стандартной оснастке MMC-консоли идентификаторы объектов не видны, однако они являются неотъемлемой частью объектов. Для просмотра идентификаторов необходимо использовать специфическое программное обеспечение, например, SysInternals AD Explorer, Microsoft ADSIEdit или Softerra LDAP Browser (рис. 3.1).

# Типы объектов в Active Directory

Каталог Active Directory поддерживает 8 основных типов объектов (табл. 3.2), каждый из которых описывается набором свойств. Часть свойств повторяется, часть — индивидуальна. Значения некоторых свойств можно просмотреть и изменить с помощью встроенного в Active Directory мастера, а значения других — только с помощью специализированного программного обеспечения или сценария.

Тип объекта определяется набором параметров, которые хранятся в Active Directory в массиве `objectClass`. Некоторые значения присутствуют везде, например Top, другие — строго идентифицируют объект. Избыточность значений необходима для совместимости доменов, построенных на основе Windows 2k и Windows NT. При составлении поисковых запросов в качестве фильтра используют только индивидуальные значения массива `objectClass` для однозначной идентификации. Значения многих элементов объектов разных типов пересекаются.

***Таблица 3.2.*** *Взаимосвязь типов объектов Active Directory и значений параметра* `objectClass`

| Комментарий | Тип объекта | Значение `objectClass` | Фрагмент поискового запроса |
|---|---|---|---|
| Учетная запись компьютера | Computer | Top<br>Person<br>Organizational-Person<br>User<br>Computer | `objectClass= 'Computer'` |
| Контакт, используется в почтовых приложениях | Contact | Top<br>Person<br>OrganizationalPerson<br>Contact | `objectClass='Contact'` |
| Группа безопасности | Group | Top<br>Group | `objectClass='Group'` |
| Учетная запись пользователя, не совместимая с доменами Windows 2k | InetOrgPerson | Top<br>Person<br>Organizational-Person<br>User<br>InetOrgPerson | `objectClass= 'InetOrgPerson'` |
| Папка дерева каталогов Active Directory | OU | Top<br>OrganizationalUnit | `objectClass= 'OrganizationalUnit'` |
| Опубликованный в Active Directory сетевой принтер | Printer | Top<br>Leaf<br>ConnectionPoint<br>PrintQueue | `objectClass= 'PrintQueue'` |

***Таблица 3.2*** *(окончание)*

| Комментарий | Тип объекта | Значение `objectClass` | Фрагмент поискового запроса |
|---|---|---|---|
| Опубликованная в Active Directory сетевая папка | Shared Folder | Top<br>Leaf<br>ConnectionPoint<br>Volume | `objectClass='Volume'` |
| Учетная запись пользователя, совместимая с доменами Windows NT | User | Top<br>Person<br>OrganizationalPerson<br>User | `objectClass='User' and not objectClass='Computer'` |
| Контейнер (папка), создаваемый в каталоге Active Directory по умолчанию | Container | Top<br>Container | `objectClass='Container'` |
| Корень домена | RootDSE | Top<br>Domain<br>DomainDNS | `objectClass='DomainDNS'` |

В расширенном режиме становятся доступны еще несколько типов объектов, которые выполняют сервисные функции. Описание этих объектов выходит за рамки книги.

# Способы доступа к объектам Active Directory

Доступ к объектам Active Directory можно получить с помощью семейства `IADs*`-интерфейсов и ADO. В зависимости от поставленных задач необходимо использовать тот или иной способ: для чтения свойств объектов идеально подходит семейство `IADs*`-интерфейсов, для реализации поиска объектов — ADO.

Самый простой способ доступа к Active Directory реализуется с помощью пары `IADs*`-интерфейсов, а именно интерфейсов `IADs` и `IADsContainer`.

# LDAP-путь к объекту

Совокупность идентификаторов объектов DC, OU, CN образует полное составное имя — DN (Distinguished Name), а имя самого объекта относительно содержащего его контейнера — относительное составное имя — RDN (Relative Distinguished Name).

Полное составное имя объекта включает в себя имя объекта и всех его родителей, начиная с корня домена.

Существуют две формы доступа к ADSI: развернутая и сокращенная. Рассмотрим принципы построения путей к ресурсу обоими способами на примере доменов domain1.ru, domain2.ru и domain3.ru (рис. 3.2).

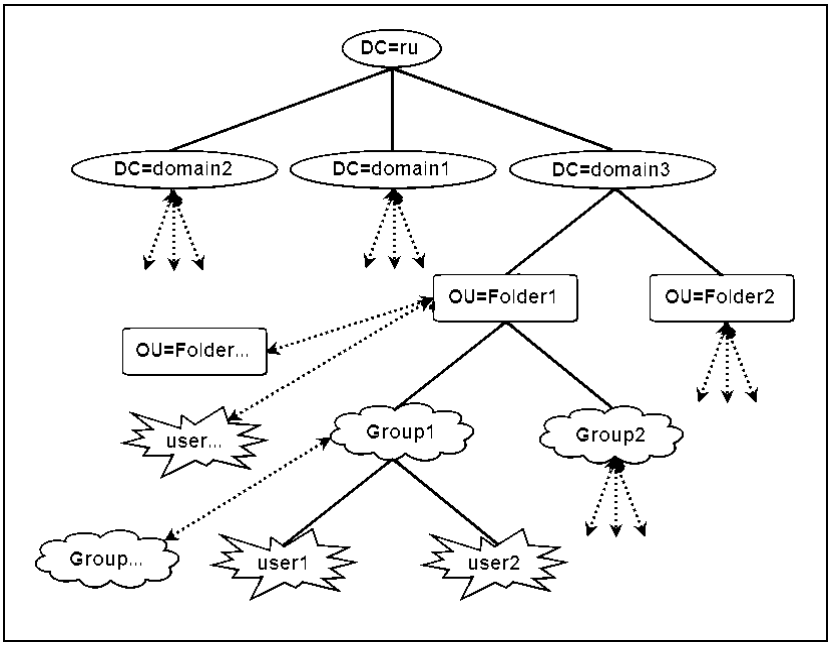



**Рис. 3.2.** Древовидная структура Active Directory

## Развернутая форма записи

При развернутой форме записи строка связывания начинается с описания верхнего элемента структуры. Затем происходит переход вниз по иерархии (листинг 3.1). Помните: при формировании пути к объекту следует исключать пробелы и переносы строк. В листингах 3.1 и 3.3 они присутствуют только для того, чтобы показать длинную строку.

**Листинг 3.1. Шаблон доступа к объектам каталога. Полная форма записи**

```
Set obj = GetObject
↳("LDAP://DC=Domain_name1/DC=Domain_name2/DC=Domain_name3/
↳OU=OU_Name_Level1/OU=OU_Name_Level2/…/OU=OU_Name_Levelμ/CN=CN_Name")
```

Здесь строка `DC=Domain_name1/DC=Domain_name2/DC=Domain_name3` образует полное имя контроллера домена, а элементы `OU=OU_Name_Level1/OU=OU_Name_Level2/…/OU=OU_Name_Levelμ` представляют собой вложенные друг в друга элементы в количестве μ штук. В развернутой форме доступа объект `CN` является "дном колодца" в иерархии.

В листинге 3.2 приведен пример запроса к объекту `CN=User2` с помощью провайдера LDAP в развернутой форме доступа (см. рис. 3.2).

**Листинг 3.2. Запрос к объекту `CN=User2`. Полная форма записи**

```
Set obj = GetObject( _
              "LDAP://DC=RU/DC=Domain3/OU=Folder1/OU=Group1/CN=User2")
```

## Сокращенная форма записи

Сокращенная форма записи характеризуется тем, что строка запроса строится в соответствии с обратной иерархией структуры организации (листинг 3.3).

**Листинг 3.3. Шаблон доступа к объектам каталога. Сокращенная форма записи**

```
Set obj=GetObject("LDAP://CN=CN_Name,OU=OU_Name_Levelμ,…,
↳OU=OU_Name_Level2,OU=OU_Name_Level1,DC==Domain_name3,
↳DC=Domain_name2,DC=Domain_name1")
```

Запрос к объекту `CN=User2` с помощью сокращенной формы доступа показан в листинге 3.4.

**Листинг 3.4. Запрос к объекту `CN=User2`. Сокращенная форма записи**

```
Set obj=GetObject( _
          "LDAP://CN=User2,OU=Group1,OU=Folder1,DC=Domain3,DC=RU")
```

На практике почти всегда используется сокращенная форма записи. В качестве пути обычно участвует значение поля `distinguishedName` объекта.

# Способы доступа к каталогу Active Directory

Запросы к службам каталогов в ADSI могут выполняться на двух диалектах — LDAP (RFC 2254) и SQL (стандарт ANSI SQL). Опытные программисты для увеличения скорости работы сценариев используют подключение к каталогу Active Directory через прилинкованный SQL-сервер.

## Диалект LDAP

Фильтры запроса диалекта LDAP определяются в соответствии с правилами синтаксиса запросов LDAP, описанными в RFC 2254.

Запрос состоит из нескольких частей:

- RDN-путь к объекту;
- фильтр поиска;
- атрибуты объекта, значения которых необходимо получить;
- область поиска.

Перечисленные составные части разделены символом "точка с запятой". Рассмотрим подробнее каждую их составных частей.

### RDN-путь к объекту

Формат относительного составного пути (RDN) к объектам каталога Active Directory описан в табл. 1.1 в *главе 1*.

## Фильтр поиска

Вторую часть запроса составляет фильтр поиска.

Для поисковых фильтров в LDAP предусмотрено два варианта синтаксиса. Первый вариант использует только один фильтр:

```
(<имя_атрибута><оператор><значение>)
```

Второй вариант использует несколько фильтров, объединенных логическим оператором:

```
(<оператор><фильтр1><фильтр2>...<фильтрN>)
```

На практике в большинстве случаев используется второй вариант фильтра. При написании фильтра используются операторы, список которых приведен в табл. 3.3.

***Таблица 3.3.*** *Операторы фильтра поиска*

| Оператор | Значение | Оператор | Значение |
|---|---|---|---|
| = | Эквивалентно | & | И |
| ~= | Примерно равно | \| | ИЛИ |
| <= | Меньше или равно | ! | НЕ |
| >= | Больше или равно | * | ВСЕ |

Приведем несколько характерных примеров фильтров:

- `(objectClass=*)`. Все объекты имеют принадлежность к одному из классов. Данный фильтр возвратит все объекты в области поиска, заданной с помощью четвертого атрибута;
- `&((objectClass=person)(!objectClass=computer))`. Будут возвращены все объекты, принадлежащие к классу `person` и не принадлежащие к классу `computer`. Такими объектами являются учетные записи пользователей;
- `&((description=test)(|(cn=User*)(cn=Group1)))`. Будут возвращены все объекты с описанием test и именем, начинающимся со слова User, а также объект Group1.

Встречаются случаи, в которых необходимо в качестве значений указывать такие служебные символы, как звездочка, скобка и др. Чтобы реализовать эту

возможность, вместо символа необходимо указать соответствующее ему кодовое значение, приведенное в табл. 3.4.

***Таблица 3.4.*** *Зарезервированные имена, используемые в фильтрах поиска*

| Символ | Значение | Символ | Значение |
|---|---|---|---|
| * | \2a | \ | \5c |
| ( | \28 | NUL | \00 |
| ) | \29 | / | \2f |

## Атрибуты объекта

Все атрибуты, значения которых необходимо получить, следует указать через запятую согласно объектной модели объектов, поиск которых выполняется. Регистр символов в данном случае не играет значения.

## Область поиска

Поиск объекта может быть осуществлен в одной из трех областей:

- `Base` — поиск осуществляется по корневому объекту, указанному в первой части запроса. Всегда возвращается либо один объект, либо пустой набор объектов. Эта область поиска используется чаще всего для проверки на существование объекта, указанного в запросе;
- `OneLevel` — поиск осуществляется только по непосредственным дочерним объектам контейнерного объекта, указанного в первой части запроса. Поиск по вложенным объектам более низких уровней не производится. В поиск не попадет и сам контейнерный объект;
- `SubTree` — поиск осуществляется по всем вложенным объектам. В поиск при этом не попадает сам контейнерный объект.

## Пример доступа к Active Directory

Рассмотрим пример использования диалекта LDAP. В листинге 3.5 осуществляется поиск всех учетных записей пользователей, начинающихся с указанной в параметре `Letter` буквы. Учетные записи пользователей принадлежат к классу `Person`. Поскольку компьютеры тоже принадлежат к классу `Person`, то

добавляется фильтр, который отсеивает объекты, принадлежащие к классу `Computer`. Наконец, фамилии пользователей, начинающиеся с буквы D, например, в фильтре задаются следующим образом: `sn = d*`.

**Листинг 3.5. Пример использования диалекта LDAP**

```
' Буква, с которой начинается фамилия пользователя
Letter = "d"

' Определение имени текущего домена
set rootDSE_ = GetObject("LDAP://RootDSE")
domain = "LDAP://" & rootDSE_.Get("defaultNamingContext")

'Подключение к каталогу Active Directory
Set obj= CreateObject("ADODB.Connection")
obj.CommandTimeout = 120
obj.Provider = "ADsDSOObject"
obj.Open "Active Directory Provider"

'Запрос в каталог Active Directory
ObjPath = "<" & domain & ">"
ObjFilter = "(&(sn=" & Letter & _
    "*)(objectClass=person)(!objectClass=computer))"
objFields = "cn, samaccountname"
objScope = "subtree"
query = ObjPath & "; "& ObjFilter & "; "& objFields & "; "& objScope

'Вызов метода Execute, возвращающего объект RecordSet
Set rs = obj.Execute (query)
rs.MoveFirst
WScript.Echo "Количество найденных записей: " & rs.RecordCount

Do Until rs.EOF
    f01 = rs.Fields("cn").Value
    f02 = rs.Fields("samaccountname").Value
```

```
    WScript.Echo f01 & vbNewLine & f02
    rs.MoveNext
Loop
rs.Close
```

## Диалект SQL

SQL-диалект является усеченной версией структурированного языка запросов Structured Query Language. Несмотря на это, возможности SQL-диалекта позволяют осуществлять поиск на основе запросов и выводить данные в отсортированном виде, в отличие от LDAP-диалекта. Синтаксис запроса выглядит следующим образом:

```
SELECT список_полей FROM имя_таблицы
[WHERE критерий ORDER BY поле [ASC | DESC]]
```

Запрос состоит из нескольких составных частей. По внешнему виду он очень похож на классический SQL. Так же, как и в диалекте LDAP, синтаксис можно для удобства разделить на те же части:

- атрибуты объекта, значения которых необходимо получить;
- LDAP-путь к объекту;
- фильтр поиска;
- сортировка выводимых данных.

Отличие состоит в том, что с помощью SQL-диалекта нельзя задать глубину поиска. Несмотря на это, выводимый результат можно отсортировать по возрастанию или убыванию. Рассмотрим подробнее каждую из частей.

### Атрибуты объекта, значения которых необходимо получить

SQL-запрос, предназначенный для чтения данных из базы данных, всегда начинается с ключевого слова `SELECT`. Сразу за ним следует список параметров, разделенных запятыми, значения которых необходимо получить (листинг 3.6а). Если необходимо выбрать значения всех доступных параметров, то в качестве аргумента `SELECT` указывается ключевое слово `ALL` или символ звездочки — "`*`" (листинг 3.6б).

**Листинг 3.6а. Чтение явно указанных параметров**

```
SELECT cn, samAccountName, distinguishedName FROM…
```

**Листинг 3.6б. Чтение всех неявно указанных параметров**

```
SELECT * FROM…
```

## LDAP-путь к объекту

В официальной документации по SQL зафиксировано, что после ключевого слова `SELECT` следует `FROM`, за которым должна быть указана таблица базы данных, к которой осуществляется подключение. В случае с каталогом ADSI достаточно указать LDAP-путь (листинг 3.7), который описан в табл. 1.1 в *главе 1*.

**Листинг 3.7. Описание точки входа в каталог Active Directory**

```
SELECT … FROM 'LDAP://OU=Workspace,DC=Island,DC=Ru'
```

Обратите внимание, что путь к объекту или контейнеру в Active Directory указывается в одинарных кавычках — апострофах.

## Фильтр поиска

Фильтр поиска задается ключевым словом `WHERE`. В качестве аргумента выступает выражение, сформированное по определенным правилам. Фильтр поиска поддерживает те же самые операторы фильтра поиска, что и в диалекте LDAP (табл. 3.5). Выражение выглядит следующим образом:

```
Where field1='value1' operator field1='value1' …
```

Вместо `field1` сюда нужно подставить название параметра искомого объекта, а вместо `value` — его текстовое значение. Параметр `operator` — один из операторов, перечисленных в табл. 3.5.

***Таблица 3.5.*** *Логические операторы*

| Оператор | Значение |
|---|---|
| AND | логическое И |
| OR | логическое ИЛИ |
| NOT | логическое НЕ |

## Сортировка выводимых данных

В разделе ORDER BY указывается атрибут, по значению которого будет выполнена сортировка возвращаемых результатов. Active Directory поддерживает сортировку только по одному атрибуту. Однако можно указать порядок сортировки при помощи ключевых слов ASC и DESC. По умолчанию используется ASC.

Оператор объединения JOIN в запросах к Active Directory не поддерживается. Если запрос производится из SQL Server, а Active Directory используется в качестве внешнего источника данных, то можно производить объединение таблиц SQL Server и информации, возвращаемой Active Directory.

## Пример доступа к Active Directory

Создадим пример, осуществляющий те же функции, что и пример из листинга 3.5. Он приведен в листинге 3.8.

**Листинг 3.8. Пример использования диалекта SQL**

```
' Буква, с которой начинается фамилия пользователя
Letter = "d"

' Определение имени текущего домена
set rootDSE_ = GetObject("LDAP://RootDSE")
domain = "LDAP://" & rootDSE_.Get("defaultNamingContext")

'Подключение к каталогу Active Directory
Set obj= CreateObject("ADODB.Connection")
obj.CommandTimeout = 120
```

```
obj.Provider = "ADsDSOObject"
obj.Open "Active Directory Provider"

'Запрос в каталог Active Directory
query = "SELECT cn, sAMAccountName FROM '" & domain &
    "' WHERE sn='" & Letter &
        "*' and objectClass = 'person' and not objectClass = 'computer'"
WScript.Echo query
'Вызов метода Execute, возвращающего объект RecordSet
Set rs = obj.Execute(query)
rs.MoveFirst
WScript.Echo "Количество найденных записей: " & rs.RecordCount

Do Until rs.EOF
    f01 = rs.Fields("cn").Value
    f02 = rs.Fields("sAMAccountName").Value
    WScript.Echo f01 & vbNewLine & f02
    rs.MoveNext
Loop
rs.Close
```

## Прилинкованный SQL-сервер

Доступ к Active Directory с помощью `ADsDSOObject` обеспечивает возможность любых манипуляций со свойствами ее объектов: их можно создавать, удалять, изменять свойства. Основной недостаток использования провайдера `ADsDSOObject` — низкая скорость получения данных. Например, если читать из Active Directory свойства 500—600 объектов, отфильтровывая их из общей массы, то эта процедура занимает по времени 1—3 минуты. Увеличить скорость чтения данных можно, используя в качестве провайдера прилинкованный к Active Directory SQL-сервер. Скорость решения аналогичной задачи сокращается до нескольких секунд.

Создавая прилинкованный сервер, необходимо помнить о нескольких недостатках данного метода:

- с помощью прилинкованного сервера можно осуществлять быстрый поиск и чтение данных, однако запись данных невозможна;

- прилинкованный сервер имеет ограничение по количеству выводимых записей — по умолчанию это 1000 записей. Это ограничение — изменяемый параметр, но все-таки необходимо о нем помнить при формировании запросов;
- с помощью прилинкованного сервера невозможно считывать значения массивов, так как SQL-сервер не знает о существовании такого типа данных.

Прилинкованный сервер можно создать двумя способами: программным и вручную.

## Создание прилинкованного сервера вручную

Чтобы создать прилинкованный сервер вручную, необходимо войти в SQL Server Enterprise Manager, выбрать в левой панели пункт **SQL Server Group |** <Сервер> **| Security | Linked Server** и в контекстном меню выбрать **New Linked Server…**. В открывающемся после этого диалоговом окне (рис. 3.3) присутствуют следующие параметры:

- **Linked server** — произвольное имя создаваемого сервера. Максимальная длина имени 128 символов;
- **Provider name** — имя провайдера. Выбирается из списка, например — Microsoft Jet 4.0 OLE DB Provider. Максимальная длина этого обязательного параметра — 128 символов. Остальным параметрам значения присваиваются по необходимости;
- **Product name** — имя, используемое в качестве комментария о прилинкованном сервере размером до 128 символов;
- **Data source** — имя источника данных или его IP-адрес. Максимальная длина — 4000 символов;
- **Provider string** — строка соединения, в которой могут быть заданы имя, пароль и другие характеристики доступа, например — `Encrypt Password=False;Integrated Security=SSPI;`. Максимальная длина — 4000 символов.

На вкладке **Security** осуществляется настройка параметров безопасности. Для прилинкованного сервера к Active Directory настройки можно оставить по умолчанию: **Be made using the login's current security context**.

На вкладке **Server Options** удостоверьтесь, что отмечены опции **Data Access** и **Use Remote Collation**. Использование RPC в случае соединения с Active Directory не обязательно.

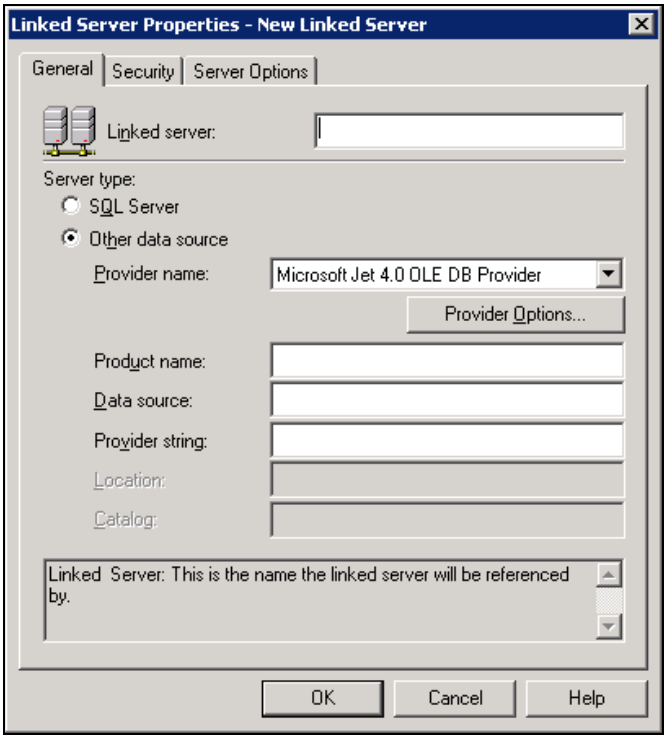



**Рис. 3.3.** Создание прилинкованного сервера

Для создания прилинкованного к Active Directory сервера используется провайдер OLE DB Provider for Microsoft Directory Services (`ADsDSOObject`). Именно его необходимо выбрать в списке **Provider name** во вкладке **Settings**. В поле **Data source** указывают `adsdatasource`. Рекомендуется добавить описание созданного сервера в поле `Product name`, например Active Directory Service Interfaces. Остальные настройки остаются без изменений.

Если по какой-то причине провайдер OLE DB Provider for Microsoft Directory Services отсутствует в предлагаемом списке, необходимо в реестре изменить значение параметра **AllowInProcess** (тип DWORD) в ветви `HKLM\Software\Microsoft\MSSQLServer\Providers\ADSDSOObject` на 1 (рис. 3.4).

Существует альтернативный вариант создания прилинкованного сервера с помощью Microsoft Jet 4.0 OLE DB Provider. Все настройки остаются прежними, за исключением поля **Provider string**: `Provider=ADsDSOObject;Encrypt Password=False;Integrated Security=SSPI; Mode=Read; Bind Flags=0; ADSI Flag=-2147483648.`

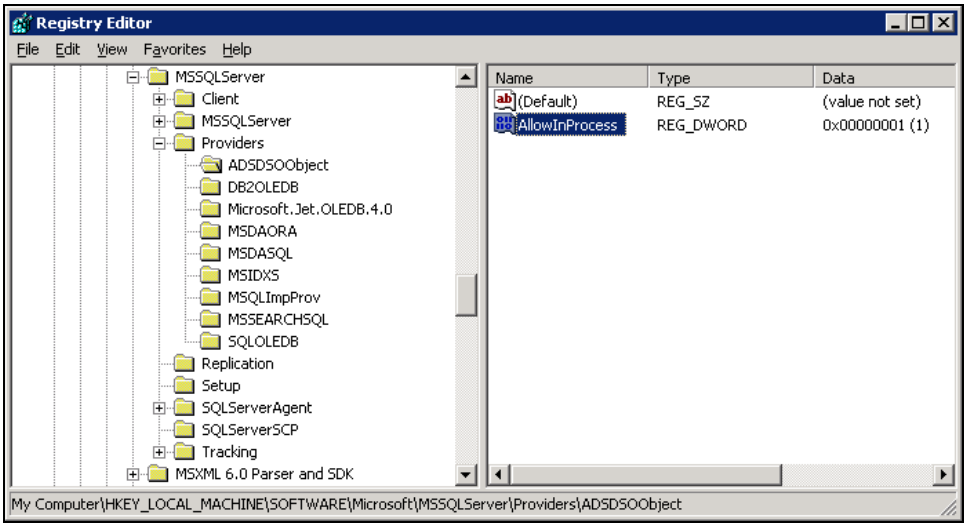



**Рис. 3.4.** Провайдер OLE DB Provider for Microsoft Directory Services в реестре

В созданном прилинкованном сервере находятся два объекта: Tables и Views. При попытке посмотреть их содержимое провайдер данных выдаст сообщение об ошибке:

```
"OLE DB Provider for Microsoft Directory Services" does not provide either the tables or views. Therefore you will get the error "Error 7301: Could not obtain a required interface from OLEDB provider ADsDSOObject
```

Не обращайте на эту ошибку внимания и переходите к созданию запроса.

## Создание прилинкованного сервера программным способом

Для создания прилинкованного сервера программным способом используется функция `sp_addlinkedserver`, синтаксис которой приведен в листинге 3.9.

**Листинг 3.9. Синтаксис функции `sp_addlinkedserver`**

```
sp_addlinkedserver [ @server = ] 'server'
[ , [ @srvproduct = ] 'product_name' ]
[ , [ @provider = ] 'provider_name' ]
[ , [ @datasrc = ] 'data_source' ]
[ , [ @location = ] 'location' ]
```

```
[ , [ @provstr = ] 'provider_string' ]
[ , [ @catalog = ] 'catalog' ]
```

Для ее выполнения нужно задать следующие параметры:

- `server` — имя создаваемого прилинкованного сервера;
- `product_name` — его описание;
- `provider_name` — имя провайдера, к которому прилинковывается сервер;
- `data_source` — источник данных — имя или IP-адрес ресурса;
- `location` — местонахождение базы как интерпретатора OLE-DB провайдера;
- `provider_string` — строка соединения, содержащая параметры создаваемого соединения, указывается при необходимости;
- `catalog` — имя каталога, который будет использоваться как временный во время соединения с провайдером.

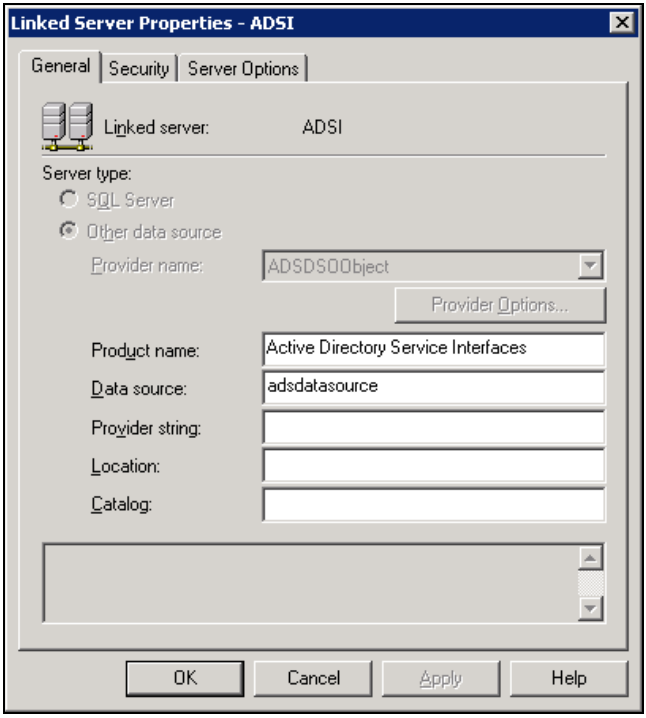



**Рис. 3.5.** Прилинкованный к Active Directory сервер

На практике запрос к Active Directory, формирующий прилинкованный сервер, может выглядеть следующим образом (на рис. 3.5 показаны свойства созданного сервера):

```
sp_addlinkedserver 'ADSI', 'Active Directory Service Interfaces',
'ADSDSOObject', 'adsdatasource'
```

## Использование прилинкованного сервера

Для подключения к SQL-серверу на VBScript используется ADODB-соединение, однако связь с каталогом Active Directory идет не напрямую через `ADsDSOObject`, а с помощью прилинкованного сервера. С помощью функции `CreateObject` создают два объекта: `Adodb.Connection` для соединения с SQL-сервером и `Adodb.Recordset` — для получения данных от прилинкованного к нему сервера. В строке соединения не нужно указывать имя базы, поскольку ни к одной из них подключение не осуществляется (листинг 3.10).

**Листинг 3.10. Чтение данных с помощью прилинкованного сервера**

```
set a = CreateObject("Adodb.Connection")
str = "driver={sql server};server=SQLServer;trusted_connected=yes;"
a.Open(str)
a.cursorLocation = 3

set rs = CreateObject("Adodb.Recordset")
str_q = "select * from OpenQuery (ADSI, 'select cn "
str_q = str_q & "from ''LDAP://DC=firm,DC=ru'' "
str_q = str_q & "where objectclass=''computer'' ')"
rs.Open str_q, a, 1, 2, 1
rs.MoveFirst
Do Until rs.EOF
    Wscript.Echo rs.Fields(0).Value
    rs.MoveNext
Loop
```

Если созданный сценарий запускается от имени локального, а не доменного пользователя, то SQL-сервер выдаст сообщение об ошибке:

```
An error occurred while preparing a query for execution against OLE DB
provider 'ADsDSOObject'. OLE DB error trace [OLE/DB Provider
'ADsDSOObject' CommandPrepare::Prepare returned 0x80040e14]
```

# Чтение данных из Active Directory

Алгоритм чтения значения какого-либо параметра зависит от типа данных, к которому он относится. Для определения типа данных в VBScript используется функция `VarType()`, имеющая следующий синтаксис:

```
Value = VarType(VarName)
```

где `VarName` — переменная, тип которой необходимо определить, а `Value` — возвращаемое функцией значение (листинг 3.11а и 3.11б). Расшифровка возможных значений приведена в табл. 3.6.

***Таблица 3.6.*** *Значения, возвращаемые функцией* `VarType()`

| Константа | Значение | Описание |
|---|---|---|
| vbEmpty | 0 | Empty (пустое значение) |
| vbNull | 1 | Null (не содержит данных) |
| vbInteger | 2 | Integer |
| vbLong | 3 | Long |
| vbSingle | 4 | Single (число с плавающей точкой) |
| vbDouble | 5 | Double (число с плавающей точкой) |
| vbCurrency | 6 | Currency |
| vbDate | 7 | Date |
| vbString | 8 | String |
| vbObject | 9 | Object |
| vbError | 10 | Error |
| vbBoolean | 11 | Boolean |
| vbVariant | 12 | Variant (используется только с массивами) |
| vbDataObject | 13 | Object |
| vbByte | 17 | Byte |
| vbArray | 8192 | Array |

При определении типа данных массива функция `VarType()` возвращает сумму. Первое слагаемое, равное 8192, идентифицирует массив данных, а второе определяет тип данных его элементов. Например, если массив состоит из целых чисел (`Integer`), то функция `VarType()` возвратит значение 8194 = 8192 + 2. Поскольку элементами массива не могут быть пустые значения (`vbEmpty`), то функция `VarType()` никогда не возвратит значение 8192.

**Листинг 3.11а. Использование функции `VarType()`. Функция `GetObject()`**

```
'Создание ADODB-соединения
Set obj = GetObject("LDAP://PATH_TO_OBJECT")
a = obj.PROPERTY
MsgBox VarType(a)
```

**Листинг 3.11б. Использование функции `VarType()` в ADODB-соединениях**

```
'Создание ADODB-соединения
Set objConn = CreateObject("ADODB.Connection")
Set objCom = CreateObject("ADODB.Command")
objConn.Provider = "ADsDSOObject"
objConn.Open "Active Directory Provider"
Set objCom.ActiveConnection=objConn
'составление и обработка SQL-запроса
Query = "SELECT PROPERTY FROM 'LDAP://PATH_TO_OBJECT'"
Set st = objConn.Execute(Query)
'Вывод результата типа данных переменной
a = st.Fields("PPOPERTY").Value
MsgBox VarType(a)
```

В приведенных сценариях переменная *PROPERTY* — название параметра, значение которого необходимо получить и определить его тип данных, а *PATH_TO_OBJECT* — LDAP-путь к объекту, например, `DC=Island,DC=ru`.

Перечислим все встречающиеся в Active Directory типы данных:

- длинное число, `Long`, имеет `VarType = 3`;
- дата-время, `Date`, — `VarType = 7`;
- строка, `String`, — `VarType = 8`;

- объект, `Object`, — `VarType = 9`;
- булево значение, `Boolean`, — `VarType = 11`;
- массив элементов, `Array`, — `VarType = 8192+x`, где `x` — одно из перечисленных выше значений.

## Тип данных: длинное целое число (*VarType = 3*)

Длинному целому числу в интерпретации VBScript соответствует тип данных `Long`. При чтении данных значения типа `Long` автоматически преобразуются к значениям, имеющим тип `String`. В табл. 3.7 приведены часто используемые параметры основных объектов Active Directory: учетная запись пользователя (person), учетная запись группы (group) и папка (organizational unit).

***Таблица 3.7.*** *Параметры объектов (`Long`), создаваемых по умолчанию*

| Переменная | Комментарий | Пример |
|---|---|---|
| **Учетная запись пользователя** (`ObjectClass = Person`) | | |
| `codePage` | Кодовая страница. Появилась в Windows 2003, в настоящее время не используется | 0 |
| `countryCode` | Код страны. Появилась в Windows 2003, в настоящее время не используется | 0 |
| `instanceType` | Статус объекта на сервере, который может принимать значения 1, 2, 4, 8, 16 и 32. По умолчанию принимает значение 4, остальные значения используются крайне редко | 4 |
| `primaryGroupID` | Идентификатор группы безопасности, назначенной по умолчанию (см. вкладку **MemberOf** свойств пользователя). Идентификатор назначаемой по умолчанию группы Domain Users равен 513 | 513 |
| `sAMAccountType` | Информация о типе объекта | 805306368 |
| `userAccountControl` | Значение, устанавливающее правила регистрации пользователя, регистрирующегося в сети под текущей учетной записью | 512 |

***Таблица 3.7*** *(окончание)*

| Переменная | Комментарий | Пример |
|---|---|---|
| **Учетная запись группы** (`ObjectClass = Group`) | | |
| `groupType` | Суммарное значение, определяющее область действия группы и ее тип | –2147483643 |
| `instanceType` | Статус объекта, определяющий действия, которые могут быть выполнены в домене над объектом. В нормальном режиме равен 4 (чтение и запись значений атрибутов) | 4 |
| `sAMAccountType` | Информация о типе объекта | 268435456 |
| **Учетная запись папки** (`ObjectClass = OrganizationalUnit`) | | |
| `instanceType` | Статус объекта, определяющий действия, которые могут быть выполнены в домене над объектом. В нормальном режиме равен 4 (чтение и запись значений атрибутов) | 4 |

Чтение данных осуществляется с помощью функции `GetObject()` (листинг 3.12а) или ADODB-соединения (листинг 3.12б). В данном случае оба способа равнозначны, и предпочтительно использовать вариант с функцией `GetObject()`. В приведенных в листингах примерах мы получаем значение параметра `sAMAccountType` группы Test, расположенной в папке WorkSpace корня домена.

**Листинг 3.12а. Чтение значений типа `Long` с помощью функции `GetObject()`**

```
Set obj = GetObject("LDAP://CN=Test,OU=WorkSpace,DC=Island,DC=ru")
Wscript.Echo obj.sAMAccountType
```

**Листинг 3.12б. Чтение значений типа `Long` с помощью ADODB-соединения**

```
'Создание ADODB-соединения
Set objConn = CreateObject("ADODB.Connection")
Set objCom = CreateObject("ADODB.Command")
objConn.Provider = "ADsDSOObject"
objConn.Open "Active Directory Provider"
```

```
Set objCom.ActiveConnection=objConn
'составление и обработка SQL-запроса
Query = "SELECT sAMAccountType FROM" & _
        "'LDAP://CN=Test,OU=WorkSpace,DC=Island,DC=ru'"
Set st = objConn.Execute(Query)
'Вывод результата
wscript.echo st.Fields("sAMAccountType").Value
```

## Тип данных: дата-время (*VarType = 7*)

В Active Directory присутствует ряд обязательных для всех объектов атрибутов (см. *разд. "Типы данных объектов Active Directory" в главе 1*), в том числе и параметры, характеризующие дату и время создания и изменения объектов (табл. 3.8). Данные в формате дата-время (`Date`), хранящиеся в Active Directory в формате Generalized Time (GT), в процессе чтения автоматически преобразуются в строку. Несмотря на это, для получения понятного всем значения необходимо преобразовать это значение в соответствии с определенным алгоритмом, проиллюстрированным на рис. 3.6.

***Таблица 3.8.*** *Параметры типа* `Date` *всех объектов*

| Переменная | Комментарий | Пример | ADODB-соединение | Функция `GetObject()` |
|---|---|---|---|---|
| **Все типы объектов** (`ObjectClass = *`) | | | | |
| `createTimeStamp` | Время создания объекта | 20070222080007.0Z | + | – |
| `modifyTimeStamp` | Время последнего изменения объекта | 20080507120605.0Z | + | – |
| `whenCreated` | Время создания объекта | 20070222080007.0Z | + | + |
| `whenChanged` | Время последнего изменения объекта | 20080507120605.0Z | + | + |

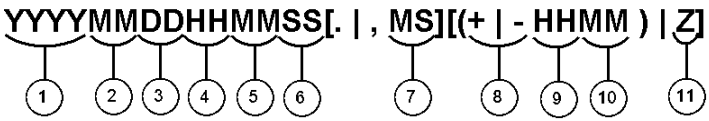



**Рис. 3.6.** Структура данных в формате Generalized Time

Параметр в формате Generalized Time в общем виде имеет структуру, показанную на рис. 3.6. На рисунке цифрами обозначены: 1 — год; 2 — месяц; 3 — день месяца; 4 — часы в 24-часовом формате; 5 — минуты; 6 — секунды; 7 — через точку или запятую приводятся десятые доли секунды; 8 — расположение текущего часового пояса относительно Гринвича; 9 — сдвиг в часах относительно Гринвича; 10 — сдвиг в минутах относительно Гринвича; 11 — указано международное время (GMT).

Значение параметра в формате GT состоит из трех частей: обязательной (пункты 1—6) и двух необязательных (пункт 7 и пункты 8—11). В первой (обязательной) части указывается время с точностью до секунды. Например, значению 20080410090637 в формате GT соответствует 10 апреля 2008 года, 9 часам 6 минутам и 37 секундам местного времени.

Первая необязательная часть (7) позволяет уточнить время с точностью до десятой доли секунды. Отделить десятые доли от основной части можно с помощью точки или запятой: Например, 20080410090637,4 или 20080410090637.4. Оба значения соответствуют дате 10 апреля 2008 года, 9 часам 6 минутам 37,4 секундам.

Чтобы указать международное время используется вторая необязательная часть (пункты 8—11). Существуют два способа: указать либо символ Z (от Zero, что обозначает Гринвичский меридиан) в конце строки, либо — сдвиг местного времени относительно международного (то есть все того же времени по Гринвичу). Например, значения 20080410040637Z и 20080410090637–0500 идентичны, различается лишь форма записи. Значение соответствует 4 часам 6 минутам и 37 секундам по Гринвичу. При этом из второй записи видно, что время сдвига относительно Гринвичского меридиана равно −5 часам.

В Active Directory значения параметров в формате Generalized Time формируются по шаблону, представленному на рис. 3.7.

На рисунке цифрами обозначены: 1 — год; 2 — месяц; 3 — день месяца; 4 — часы в 24-часовом формате; 5 — минуты; 6 — секунды; 7 — через точку приводятся десятые доли секунды (в Active Directory это значение во всех

случаях равно нулю); 8 — Z (от Zero), указывает, что время приведено в международном формате.

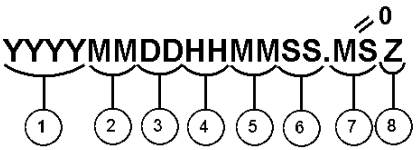



**Рис. 3.7.** Структура данных в формате Generalized Time для Active Directory

Значения параметров `createTimeStamp` и `modifyTimeStamp` хранятся в Active Directory в формате operational attribute (см. рис. 3.6), в то время как значения параметров `whenChanged` и `whenCreated` хранятся в обычном текстовом виде. Последние два параметра, дублирующие значения первых двух, предназначены для совместимости с доменами, построенными на основе Windows NT. Поскольку с помощью функции `GetObject()` невозможно считать значения данных, хранящихся в формате operational attribute, для их получения рекомендуется использовать ADODB-соединение.

В листинге 3.13а приведен пример чтения значения параметра `whenCreate` с помощью функции `GetObject()`, а в листинге 3.13б — пример чтения значения параметра `createTimeStamp` (рис. 3.8).

**Листинг 3.13а. Чтение параметров в формате UTC Time с помощью функции `GetObject()`**

```
' Только для параметров whenCreated, whenChanged
Path = "LDAP://CN=Petr\, Ivanov,OU=WorkSpace,DC=Island,DC=ru"
Set obj = GetObject(Path)
MsgBox obj.Get("whenCreated")
```

**Листинг 3.13б. Чтение параметров в формате UTC Time с использованием ADODB-соединения**

```
'Создание ADODB-соединения
Set objConn = CreateObject("ADODB.Connection")
Set objCom = CreateObject("ADODB.Command")
```

```
objConn.CommandTimeout = 120
objConn.Provider = "ADsDSOObject"
objConn.Open "Active Directory Provider"
Set objCom.ActiveConnection=objConn
'составление и обработка SQL-запроса
Query = "SELECT createTimeStamp FROM " & _
        "'LDAP://CN=Petr\, Ivanov,OU=WorkSpace,DC=Island,DC=ru'"
Set st = objConn.Execute(Query)
'Вывод результата
MsgBox st.Fields("createTimeStamp").Value
```

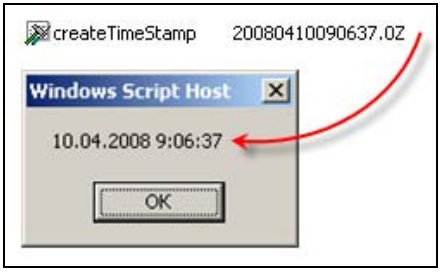



**Рис. 3.8.** Чтение параметра `createTimeStamp` в формате Generalized Time

## Тип данных: строка (*VarType = 8*)

Большинство параметров различных объектов в каталоге Active Directory хранятся в виде строк (табл. 3.9).

***Таблица 3.9.*** *Параметры типа* `String` *в различных объектах*

| Переменная | Комментарий | Пример |
|---|---|---|
| **Учетная запись пользователя** (`ObjectClass = Person`) | | |
| `cn` | Составное имя пользователя (`cn` — сокращение от Canonical Name), формирующееся из фамилии (поле `givenName`), имени (поле `sn`) и необязательного инициала (поле `initials`) | Petr, Ivanov |

***Таблица 3.9*** *(продолжение)*

| Переменная | Комментарий | Пример |
|---|---|---|
| **Учетная запись пользователя** (`ObjectClass = Person`) | | |
| `displayName` | Отображаемое имя. Его значение идентично составному имени пользователя (поле `cn`). Используется для совместимости с доменами Windows NT | Petr, Ivanov |
| `distinguishedName` | Составной путь (DN) к объекту в соответствии с RFC 1779, 2247 | CN=Petr\, Ivanov, CN=WorkSpace, DC=Island,DC=Ru |
| `givenName` | Имя учетной записи пользователя. Является обязательным параметром | Ivanov |
| `mail` | Адрес электронной почты в формате UPN. Заполняется после создания ящика электронной почты для учетной записи пользователя | PIvanov@island.ru |
| `sAMAccountName` | Имя пользователя для входа в сеть. Используется для совместимости пространств имен Active Directory с доменами на основе Windows NT | PIvanov |
| `sn` | Имя пользователя (`sn` — сокращение от Second Name) | Petr |
| `userPrincipalName` | UPN-имя пользователя, описанное в RFC 822. Это основной формат имен, используемый в Active Directory | PIvanov@island.ru |
| `initials` | Инициалы пользователя. Необязательный параметр длиной до 6 символов | V |
| **Учетная запись группы** (`ObjectClass = Group`) | | |
| `cn` | Имя группы | Guests |
| `distinguishedName` | Составной путь (DN) к объекту в соответствии с RFC 1779 и 2247 | CN=Guests,CN=Builtin, DC=Island,DC=Ru |

***Таблица 3.9*** *(окончание)*

| Переменная | Комментарий | Пример |
|---|---|---|
| **Учетная запись группы** (`ObjectClass = Group`) | | |
| `sAMAccountName` | Имя группы, используемое для совместимости с доменом Windows NT. Обычно совпадает с `cn` | Guests |
| **Учетная запись папки** (`ObjectClass = OrganizationalUnit`) | | |
| `distinguishedName` | Составной путь (DC) к объекту в соответствии с RFC 1779 и 2247 | OU=Сотрудники, DC=Island,DC=Ru |
| `name` | Отображаемое имя папки. Как правило совпадает со значением поля `ou` | Сотрудники |
| `ou` | Имя папки, участвующее в составном пути к папке (то есть в поле `distinguishedName`) | Сотрудники |

Чтение данных осуществляется одним из ранее описанных способов: с помощью `IADs*`-интерфейсов (листинг 3.14а) или ADODB-соединения (листинг 3.14б). В обоих листингах решается одна и та же задача — чтение параметра `cn` группы test в папке group, расположенной в корне домена.

**Листинг 3.14а. Чтение значения строковых параметров**

```
Set obj = GetObject("LDAP://CN=Test,OU=Group,DC=Island,DC=Ru")
WScript.Echo obj.cn
```

**Листинг 3.14б. Чтение значения строковых параметров с помощью ADODB–соединения**

```
'Создание ADODB-соединения
Set objConn = CreateObject("ADODB.Connection")
Set objCom = CreateObject("ADODB.Command")
objConn.Provider = "ADsDSOObject"
objConn.Open "Active Directory Provider"
```

```
Set objCom.ActiveConnection=objConn
'составление и обработка SQL-запроса
Query = "SELECT cn FROM 'LDAP://CN=Test,OU=Group,DC=Island,DC=Ru'"
Set st = objConn.Execute(Query)
'Вывод результата
wscript.echo st.Fields("cn").Value
```

## Тип данных: объект (*VarType = 9*)

Хранящаяся в Active Directory информация, соответствующая типу данных Large Integer, воспринимается интерпретатором VBScript как объект (табл. 3.10).

Алгоритм чтения параметров с префиксом `uSN` отличается от методики чтения остальных параметров. Это объясняется тем, что значение этих параметров — число, автоматически преобразованное в символы (char), в то время как остальные — это зашифрованные значения формата дата-время.

***Таблица 3.10.*** *Параметры типа* `Object` *в различных объектах*

| Переменная | Комментарий | Пример |
|---|---|---|
| **Все типы учетных записей** (`ObjectClass = *`) | | |
| `uSNChanged` | Порядковый номер изменений свойств объекта, отсчитывающийся с момента создания объекта. Количество изменений, произошедших с объектом, равно разности `uSNChanged - uSNCreated` | 794175 |
| `uSNCreated` | Порядковый номер изменений, сделанных в Active Directory на момент создания объекта | 794169 |
| **Учетная запись пользователя** (`ObjectClass = Person`) | | |
| `accountExpires` | Дата и время окончания срока действия учетной записи. После наступления этой даты учетная запись пользователя будет автоматически заблокирована. Если значение равно 0 — функция дезактивирована | 9223372036854775807 |

***Таблица 3.10*** *(окончание)*

| Переменная | Комментарий | Пример |
|---|---|---|
| **Учетная запись пользователя** (`ObjectClass = Person`) | | |
| `lastLogon` | Дата и время последней регистрации пользователя в сети. При попытке получить значение с помощью функции `GetObejct()` возникает ошибка | 128555016510096179 |
| `pwdLastSet` | Дата и время последней смены пароля | 128527377084375000 |

Все данные типа `Object`, хранящиеся в Active Directory, считываются с помощью функции `Get()`. Если запрошенное значение представляет собой дату-время, то результат состоит из двух частей: `HighPart` (Дата) и `LowPart` (Время). Пример чтения показан в листинге 3.15а.

**Листинг 3.15а. Получение значение типа `Object`**

```
Set obj = GetObject("LDAP://CN=Ivan\, Petrov,CN=Users,DC=Island,DC=Ru")
Set objUSN = obj.Get("uSNChanged")
Wscript.Echo Abs(objUSN.HighPart * 2^32 + objUSN.LowPart)
```

Для чтения некоторых параметров, например `lastLogon`[1], применяется другая методика чтения. Значение таких параметров — промежуток времени с 1 января 1601 года по настоящий момент с точностью до 100 миллисекунд. Преобразование данных может осуществляться с помощью функции `ConvertTime`, приведенной в листинге 3.15б.

**Листинг 3.15б. Чтение параметра `lastLogon` (`accountExpires`, `pwdLastSet`, `pwdLastSet`)**

```
Set obj = GetObject("LDAP://CN=Ivan\, Petrov,CN=Users,DC=Island,DC=Ru")
Set objUSN = obj.Get("lastLogonTimestamp")
Wscript.Echo ConvertTime(objUSN)
```

[1] Эта методика также используется для чтения параметров `accountExpires`, `uSNChanged`, `lastLogonTimestamp`, `pwdLastSet`, `badPasswordTime`.

```
' Функция преобразования времени
Function ConvertTime(objDate)
Set objShell = CreateObject("Wscript.Shell")
Key="HKLM\System\CurrentControlSet\Control\TimeZoneInformation" & _
    "\ActiveTimeBias"
TempKey = objShell.RegRead(Key)
If (VarType(TempKey) = 3) Then
    Temp = TempKey
ElseIf (VarType(TempKey)=12) Then
    Temp = 0
    For k = 0 To UBound(TempKey)
        Temp = Temp + (TempKey(k) * 256^k)
    Next
End If

lngHigh = objDate.HighPart
lngLow = objDate.LowPart
If (lngLow < 0) Then
    lngHigh = lngHigh + 1
End If
If (lngHigh = 0) And (lngLow = 0 ) Then
    dtmDate = #1/1/1601#
Else
    dtmDate = #1/1/1601# + _
        (((lngHigh * (2 ^ 32))+ lngLow)/6/10^8 - Temp)/1440
End If
ConvertTime=dtmDate
End Function
```

## Тип данных: булево значение (*VarType = 11*)

Параметры, содержащие булево значение, встречаются крайне редко. Возвращаемые значения такого параметра — `TRUE` или `FALSE`. Обратите внимание, что при использовании проверки в инструкции `If…End If` значение переменной чувствительно к регистру:

```
If a = TRUE Then
```

## Тип данных: массив элементов (*VarType = 8192+x*)

Возвращаемое значение функции `VarType()` при определении типа "массив" складывается из двух составляющих: `8192` (array) `+ x`. В данном выражении `x` — число, обозначающее тип данных элементов массива. Так, если элементы массива — строки, то функция `VarType()` возвращает значение 8200 = 8192 + 8; если массив состоит из бинарных элементов, то возвращаемое значение — 8209 = 8192 + 17.

Массивы, возвращаемые Active Directory, как правило, состоят из неопределенных элементов `Variant` (`VarType = 12`). Функция в этом случае возвращает значение 8204 = 8192 + 12. Элементами массива в данном случае являются либо строки, либо числа, которые при получении значений элементов автоматически преобразуются в строки (листинг 3.16). Чтение массива осуществляется с помощью цикла `For Each…Next`.

**Листинг 3.16. Чтение массива строк с помощью ADODB-соединения**

```
'Создание ADODB-соединения
Set objConn = CreateObject("ADODB.Connection")
Set objCom = CreateObject("ADODB.Command")
objConn.CommandTimeout = 120
objConn.Provider = "ADsDSOObject"
objConn.Open "Active Directory Provider"
Set objCom.ActiveConnection=objConn
'составление и обработка SQL-запроса
Query = "SELECT OBJECTCLASS " & _
        "FROM 'LDAP://CN=Test,OU=Group,DC=Island,DC=Ru'"
Set st = objConn.Execute(Query)
'Вывод результата
For Each el In st.Fields("OBJECTCLASS").Value
    WScript.Echo el
Next
```

В Active Directory присутствуют два массива бинарных элементов, которые необходимо выделить отдельно: SID (параметр `objectSID`) и GUID (параметр `objectGUID`) объекта. Элементы этих массивов — байты, поэтому функция

`VarType()` возвращает значение 8209 = 8192 + 17. После того, как элементы массива считаны, их необходимо преобразовать в понятный всем вид (рис. 3.9) с помощью функции `Hex()`.

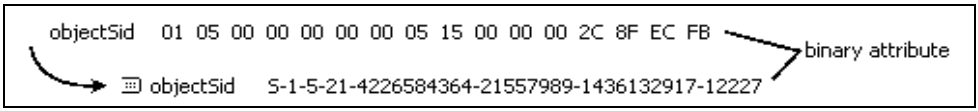



**Рис. 3.9.** Преобразование SID-объекта

Подробная информация о SID и GUID приведена в *главе 12*.

## Глава 4



# Объектная модель интерфейсов *IADs**

В этой главе описана объектная модель интерфейсов `IADs*` — основного семейства интерфейсов, предназначенных для программного управления каталогом Active Directory. С их помощью можно управлять всеми объектами и производить весь спектр манипуляций: создание, удаление, перемещение, копирование, изменение и чтение свойств объектов. Наряду с объектной моделью приведено описание типов данных, которые поддерживаются всеми без исключений интерфейсами `IADs*`.

## Основные термины и понятия

- **`IADs*`** (Interface to Active Directory) — набор интерфейсов, описывающий возможности объектов, в том числе поддерживаемые ими свойства и методы для любых ADSI-объектов.
- **LDAP** (Lightweight Directory Access Protocol) — протокол, используемый для доступа к древовидной базе данных Active Directory.
- **WinNT** (Windows NT) — протокол доступа к базе данных домена Windows NT. В настоящее время может быть использован для частичного доступа к объектам Active Directory.
- **Объектная модель** — совокупность свойств и методов, однозначно описывающая характеристики объекта и действия, которые можно произвести с объектом.
- **Составной путь** — путь в иерархии Active Directory, построенный на основе RFC 1779 и RFC 2247 (см. табл. 1.1 в *главе 1*). В составном пути участвуют идентификаторы объектов. Пример: LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=Ru.

# Типы данных, поддерживаемые интерфейсами *IADs**

Для получения и присвоения значений атрибутам различных объектов необходимо задавать их тип данных. В табл. 4.1 приведен список используемых типов данных и их описание. Будьте внимательны при задании параметра, указывающего тип данных. Если будет указан неверный тип, то значение переменной не будет получено. Чтобы избежать ошибки, рекомендуется использовать специально предусмотренную константу `ADSTYPE_UNKNOWN`, имеющую значение 26.

***Таблица 4.1.*** *Типы данных, поддерживаемые интерфейсами семейства* `IADs*`

| *Константа* | Значение | Комментарий |
|---|---|---|
| `ADSTYPE_INVALID` | 0 | Неверный тип данных |
| `ADSTYPE_DN_STRING` | 1 | Составной путь к объекту в каталоге Active Directory, например, `OU=WorkSpace,CN=Island,CN=Ru` |
| `ADSTYPE_CASE_EXACT_STRING` | 2 | Строка, символы которой чувствительны к регистру |
| `ADSTYPE_CASE_IGNORE_STRING` | 3 | Строка, символы которой не чувствительны к регистру |
| `ADSTYPE_PRINTABLE_STRING` | 4 | Строка, выводимая на экран или на печать |
| `ADSTYPE_NUMERIC_STRING` | 5 | Строка, состоящая из цифр. Например, 513 |
| `ADSTYPE_BOOLEAN` | 6 | Булево значение: `TRUE / FALSE` |
| `ADSTYPE_INTEGER` | 7 | Целое число |
| `ADSTYPE_OCTET_STRING` | 8 | Строка, состоящая из бинарных элементов |
| `ADSTYPE_UTC_TIME` | 9 | Дата и время в формате UTC (Universal Coordinated Time) |
| `ADSTYPE_LARGE_INTEGER` | 10 | Длинное целое число |

***Таблица 4.1*** *(окончание)*

| *Константа* | Значение | Комментарий |
|---|---|---|
| ADSTYPE_PROV_SPECIFIC | 11 | Специфическая для провайдера ADSI строка |
| ADSTYPE_OBJECT_CLASS | 12 | Не используется |
| ADSTYPE_CASEIGNORE_LIST | 13 | Массив нечувствительных к регистру строк |
| ADSTYPE_OCTET_LIST | 14 | Массив из бинарных строк |
| ADSTYPE_PATH | 15 | UNC-путь к каталогу |
| ADSTYPE_POSTALADDRESS | 16 | Строка, содержащая почтовый адрес |
| ADSTYPE_TIMESTAMP | 17 | "Слепок времени" — время в секундах, прошедшее с 1 января 1601 года |
| ADSTYPE_BACKLINK | 18 | Обратная ссылка |
| ADSTYPE_TYPEDNAME | 19 | Вводимое пользователем значение |
| ADSTYPE_HOLD | 20 | Жестко фиксированная структура данных, заданная на низком уровне |
| ADSTYPE_NETADDRESS | 21 | Строка, содержащая сетевой адрес |
| ADSTYPE_REPLICAPOINTER | 22 | Указатель на точку репликации |
| ADSTYPE_FAXNUMBER | 23 | Строка, содержащая номер факса |
| ADSTYPE_EMAIL | 24 | UPN-имя объекта, например, PIvanov@Island.ru |
| ADSTYPE_NT_SECURITY_DESCRIPTOR | 25 | Описатель параметра безопасности объекта |
| ADSTYPE_UNKNOWN | 26 | Неизвестный тип данных |
| ADSTYPE_DN_WITH_BINARY | 27 | Ссылка в виде составного пути на схему в бинарном виде |
| ADSTYPE_DN_WITH_STRING | 28 | Ссылка в виде составного пути на схему в строковом виде |

# Классификация интерфейсов *IADs**

Объектная модель состоит из набора COM-классов. Свойства и методы этих классов используются для манипулирования объектами службы каталогов. Наборы и свойства реализуются с помощью `IADs*`-интерфейсов, которые классифицируются следующим образом:

- *основные интерфейсы* используются для получения доступа и управления объектом;
- *интерфейсы свойств объектов* предназначены для динамического получения списка свойств указанного объекта и соответствующих им свойств;
- *интерфейсы объектов* применяются для управления свойствами объектов.

## Основные интерфейсы

В семействе интерфейсов `IADs*` можно выделить четыре основных:

- `IADs` определяет основные возможности объекта, такие как свойства и методы, поддерживаемые любыми объектами ADSI;
- `IADsContainer` обеспечивает манипуляции с объектами: создание, удаление, копирование и перемещение;
- `IADsNamespaces` предназначен для получения списка поддерживаемых интерфейсов;
- `IADsOpenDSObject` используется для соединения с указанной базой данных, в частности, с Active Directory.

## Объектная модель *IADs*

Интерфейс `IADs` предназначен для управления объектами совместно с интерфейсом `IADsContainer`. Он определяет основные возможности объекта: его свойства и методы. Примерами объектов Active Directory являются учетные записи пользователей и компьютеров, сервисы, контейнеры и т. д.

Каждым объектом Active Directory поддерживается этот интерфейс и обеспечивается:

- идентификация объекта по имени, принадлежность к классу и путь;

- идентификация и управление объектами: создание, удаление, изменение свойств;
- загрузка списка атрибутов объектов и соответствующих им значений в оперативную память компьютера.

Объекты, реализующие интерфейс `IADs`, обладают шестью свойствами (см. табл. 4.2) и семью методами (см. табл. 4.4).

## Свойства интерфейса *IADs*

В табл. 4.2 приведены основные свойства интерфейса `IADs`. Они однозначно идентифицируют объект: его имя, составной путь, принадлежность к классу и т. д.

***Таблица 4.2.*** *Свойства интерфейса* `IADs`

| Свойство | Описание |
|---|---|
| `ADsPath` | Возвращает путь к указанному объекту Active Directory |
| `Class` | Возвращает класс, к которому относится объект |
| `GUID` | Возвращает глобальный идентификатор |
| `Name` | Возвращает составное имя объекта в LDAP-формате |
| `Parent` | Возвращает составное имя родительского контейнера |
| `Shema` | Возвращает составной путь объекта к схеме Active Directory |

### Свойство *ADsPath*

Это свойство возвращает указанный путь к объекту в каталоге Active Directory (рис. 4.1). Возвращаемое значение зависит от полученной на объект ссылки.

В приведенных на рис. 4.1 строках `ADsPath` — путь к объекту в Active Directory, который может быть задан с помощью провайдеров LDAP, GC или устаревшего WinNT.

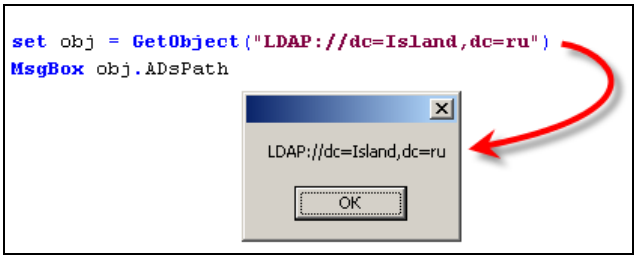



**Рис. 4.1.** Свойство `ADsPath`

## Свойство *Class*

Свойство `Class` возвращает название класса (рис. 4.2), к которому относится данный объект согласно схеме Active Directory. Несмотря на то, что любой объект принадлежит к нескольким классам одновременно, свойство возвращает последнее значение массива `objectClass` (табл. 4.3).

***Таблица 4.3.*** *Значения, возвращаемые свойством* `Class`

| **Название объекта** | **Тип объекта в Active Directory** | **Возвращаемое свойством `Class` значение** |
|---|---|---|
| Учетная запись компьютера | Computer | Computer |
| Контакт, используется в почтовых приложениях | Contact | Contact |
| Группа безопасности | Group | Group |
| Учетная запись пользователя, не совместимая с доменами Windows NT | InetOrgPerson | InetOrgPerson |
| Контейнер дерева каталогов Active Directory | OU | organizationalUnit |
| Опубликованный в Active Directory сетевой принтер | Printer | PrintQueue |
| Опубликованная в Active Directory сетевая папка | Shared Folder | Volume |
| Учетная запись пользователя, совместимая с доменами Windows NT | User | User |

***Таблица 4.3*** *(окончание)*

| **Название объекта** | **Тип объекта в Active Directory** | **Возвращаемое свойством `Class` значение** |
|---|---|---|
| Контейнер, создаваемый в каталоге Active Directory по умолчанию | Container | Container |
| Корень домена | RootDSE | DomainDNS |

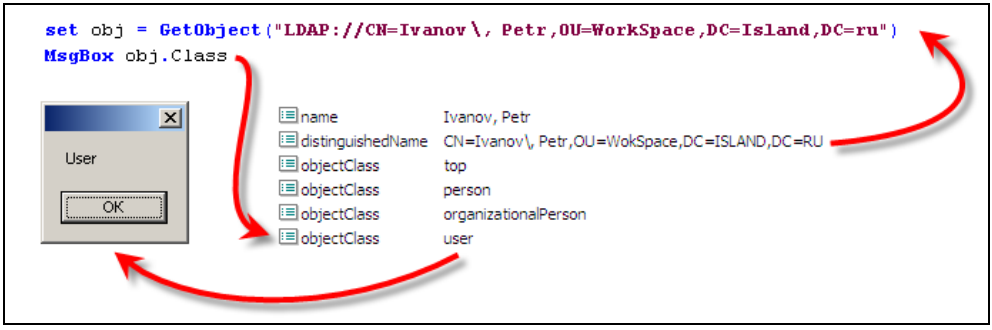



**Рис. 4.2.** Свойство `Class`

## Свойство *GUID*

Это свойство возвращает GUID (глобальный уникальный идентификатор объекта) в преобразованном виде (рис. 4.3).

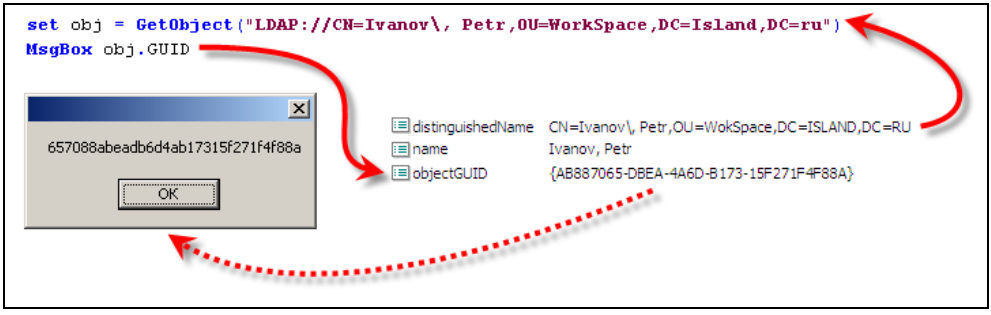



**Рис. 4.3.** Свойство `GUID`

В Active Directory значение GUID хранится в двоичном виде. Подробнее о GUID и их преобразовании написано в *главе 12*. Обратите внимание, что выводимое значение не разбито по группам в два байта.

## Свойство *Name*

Это свойство возвращает относительное составное имя объекта (рис. 4.4) в LDAP-формате (Relative Distinguished Name, RDN). Оно является составной частью Distinguished Name — составного имени в формате LDAP.

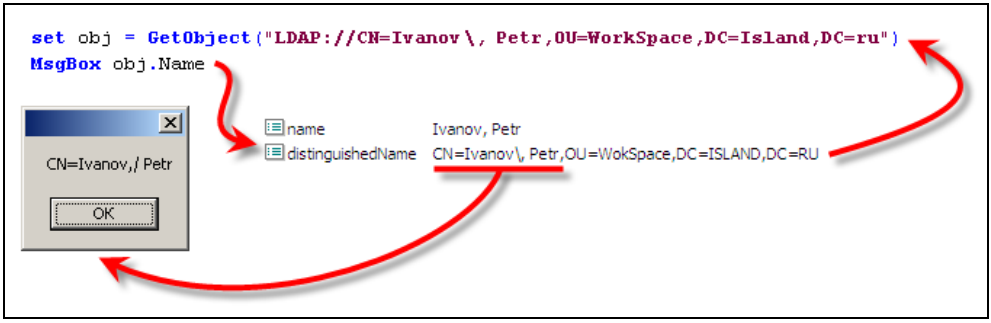



**Рис. 4.4.** Свойство `Name`

## Свойство *Parent*

Это свойство возвращает относительное составное имя родительского объекта — имя папки, в которой находится объект (рис. 4.5). Значение, как и в случае со свойством `Name`, является составной частью Distinguished Name.

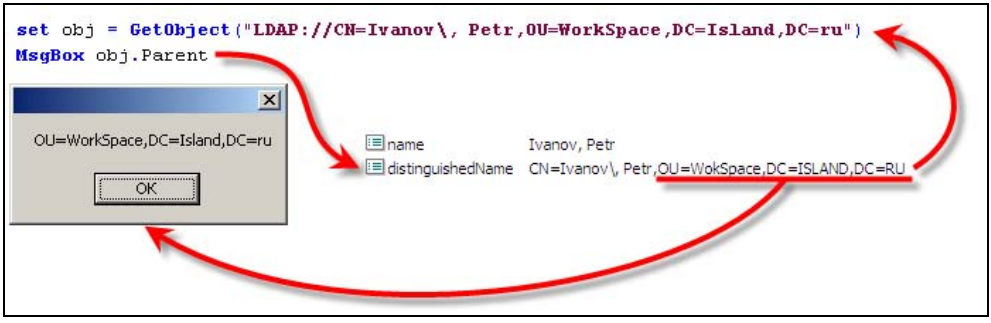



**Рис. 4.5.** Свойство `Parent`

Сумма (по-научному называемая конкатенацией) значений параметров `obj.Name` и `obj.Parent` равна значению Distinguished Name объекта.

### Свойство *Schema*

Возвращает LDAP-путь к схеме (рис. 4.6) в соответствии с шаблоном LDAP://shema/ObjectType, где ObjectType — тип объекта. Возможные возвращаемые значения приведены в табл. 3.6 в *главе 3*. Путь указывает на ту часть схемы каталога Active Directory, которой соответствует тип данного объекта. В ней описаны все поля для данного типа объектов, их тип данных и другие характеристики.

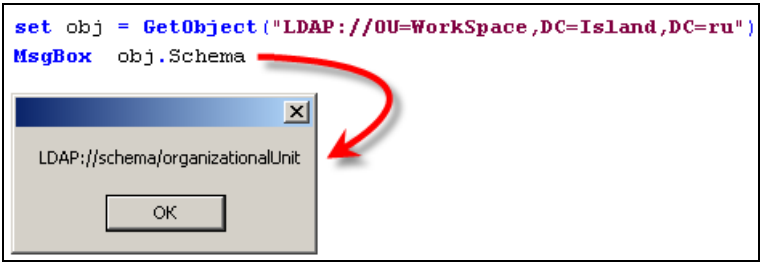



**Рис. 4.6.** Свойство `Schema`

## Методы интерфейса *IADs*

В табл. 4.4 приведены методы интерфейса `IADs`. Используя эти методы, можно создавать, удалять и изменять свойства всех типов объектов каталога Active Directory.

***Таблица 4.4.*** *Методы интерфейса* `IADs`

| Метод | Описание |
|---|---|
| `SetInfo()` | Записывает информацию о сделанных изменениях в каталог Active Directory |
| `GetInfo()` | Позволяет загрузить значения всех параметров объекта в оперативную память компьютера |
| `GetInfoEx()` | Загружает в память значения массива строковых или числовых атрибутов указанного объекта каталога Active Directory |

***Таблица 4.4*** *(окончание)*

| Метод | Описание |
|---|---|
| `Get()` | Позволяет получить значение строкового или числового атрибута объекта Active Directory в оперативную память. Имя свойства задается в качестве параметра |
| `GetEx()` | Загружает массив строковых значений указанного атрибута объекта Active Directory в оперативную память |
| `Put()` | Записывает новое значение атрибута в оперативную память компьютера. Используется в случаях, когда объект только что создан, но еще не записан в каталог Active Directory |
| `PutEx()` | Присваивает значение атрибута в кэше свойств объекта, выполняя одно из указанных действий |

## Метод *SetInfo()*

Метод `SetInfo()` используется для записи данных в каталог Active Directory. На практике он используется при создании объектов и при записи измененных свойств одного или нескольких атрибутов в каталог. Приведем шаблон использования метода `SetInfo()` в листинге 4.1.

**Листинг 4.1. Шаблон использование метода `SetInfo()`**

```
path = "…"
Set obj = GetObject(path)
…
obj.SetInfo()
```

На рис. 4.7 приведена схема изменения двух атрибутов учетной записи пользователя: размещения и номер телефона. Оба поля — `Office` и `Telephone Number` соответственно, — находятся во вкладке **General**. В каталоге Active Directory им соответствуют поля `facsimileTelephoneNumber` и `physicalDeliveryOfficeName`.

## Методы *GetInfo()* и *GetInfoEx()*

Действия методов `GetInfo()` и `GetInfoEx()` схожи. Оба метода используются для чтения атрибутов объектов в Active Directory. Принципиальная разница

заключатся в том, что с помощью метода `GetInfo()` в кэш загружают значения всех атрибутов, а с помощью `GetInfoEx()` — только значение указанного атрибута.

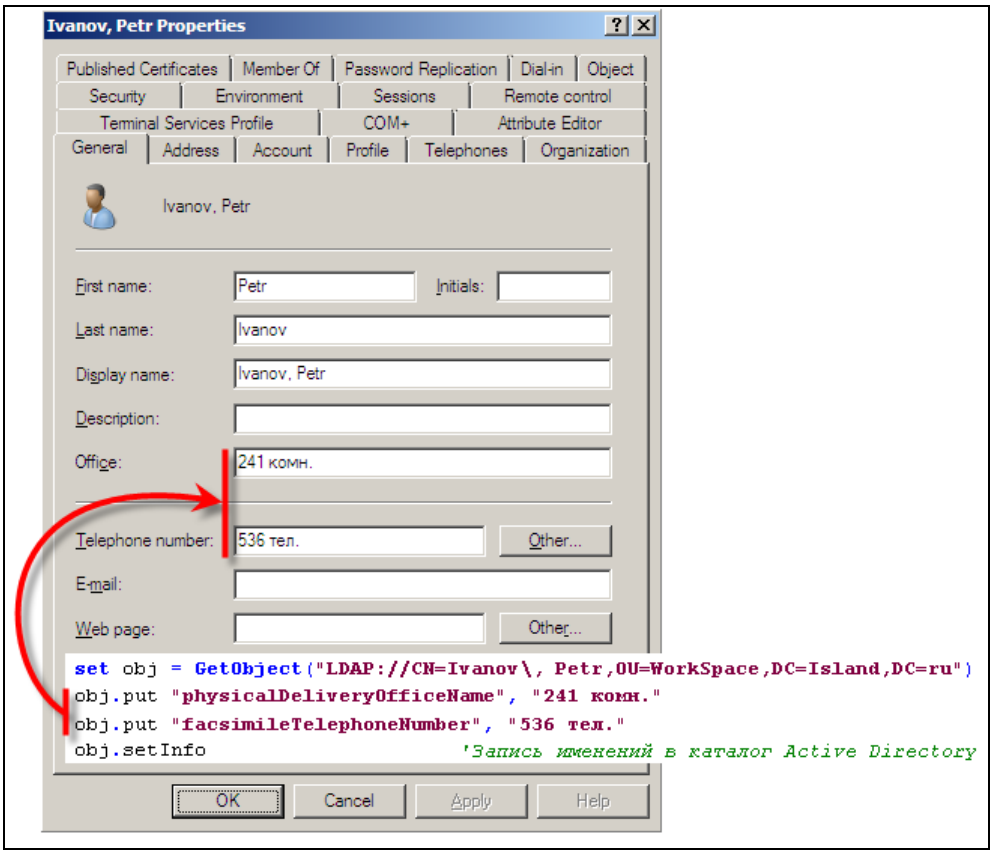



**Рис. 4.7.** Метод `SetInfo()`

Метод `GetInfo()` может быть вызван явным (листинг 4.2а) и неявным образом (листинг 4.2б). Почти всегда он вызывается неявно методом `Get()`, а `GetInfoEx()` — методом `GetEx()`.

**Листинг 4.2а. Шаблон явного вызова метод `GetInfo()`**

```
path = "..."
Set obj = GetObject(path)
```

```
obj.GetInfo()
obj.Get("PROPERTY")
…
obj.SetInfo()
```

**Листинг 4.2б. Шаблон неявного вызова метод `GetInfo()`**

```
path = "…"
Set obj = GetObject(path)
obj.Get("PROPERTY")
…
obj.SetInfo()
```

Как видно из листингов, они отличаются всего лишь одной строкой: `obj.GetInfo()`.

Явный вызов методов осуществляется при необходимости получить значение атрибута после его изменения без повторного обращения к объекту с помощью метода `GetObject()`. Подробнее о методе `GetObject()` написано далее в одноименном разделе этой главы.

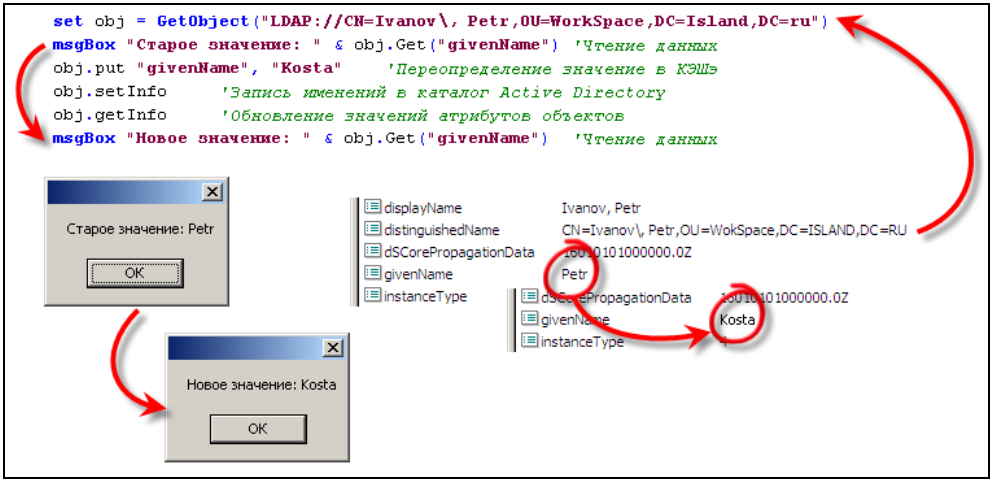



**Рис. 4.8.** Метод `GetInfo()`

На рис. 4.8 приведена схема работы сценария для изменения данных учетной записи пользователя. В первой строке сценария осуществляется обращение к объекту (к учетной записи), после чего во второй строке выводится на экран значение поля `givenName`. Затем в следующих трех строках осуществляется запись данных в каталог Active Directory и обновление кэша. И, наконец, новое измененное значение параметра `givenName` выводится на экран.

С помощью метода `GetInfoEx()` определяют, присвоено ли какое-нибудь значение тому или иному атрибуту. Если попытаться получить значение атрибута, которому не присвоено значение, с помощью метода `GetInfo()`, то интерпретатор выведет сообщение об ошибке: "The Active Directory property cannot found in the cache". Это объясняется тем, что в кэш загружаются только атрибуты с присвоенными значениями.

## Методы *Get()* и *GetEx()*

Оба метода предназначены для чтения значений атрибутов объекта в зависимости от их типов данных. Если атрибут имеет строковый (рис. 4.9) или приводимый к нему тип данных, то для чтения используется метод `Get()`, если же это массив — метод `GetEx()`.

Тип данных того или иного параметра можно определить с помощью функции `VarType()`. Как использовать эту функцию, рассказано в *разд. "Чтение данных из Active Directory" главы 3*.

В обоих случаях осуществляется неявный вызов метода `GetInfo()`. Для увеличения скорости работы сценария рекомендуется использовать в явном виде метод `GetInfoEx()`.

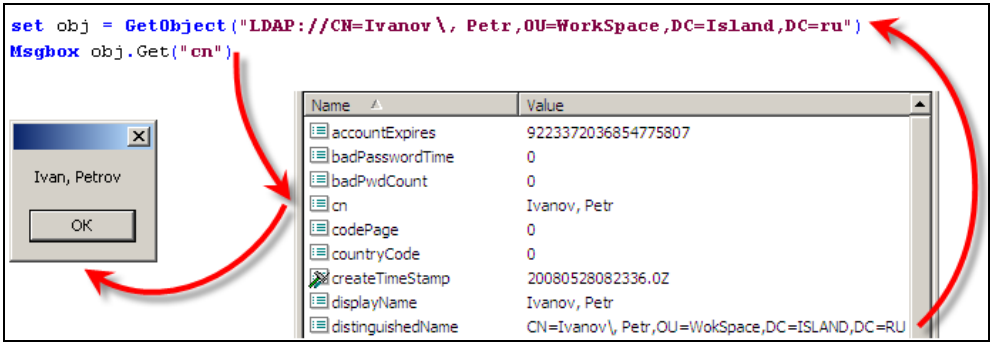



**Рис. 4.9.** Метод `Get()`

Несмотря на то, что для чтения значений массивов предназначена функция `GetEx()`, с помощью функции `Get()` также можно получить эти данные. Использование метода `GetEx()` проиллюстрировано на рис. 4.10.

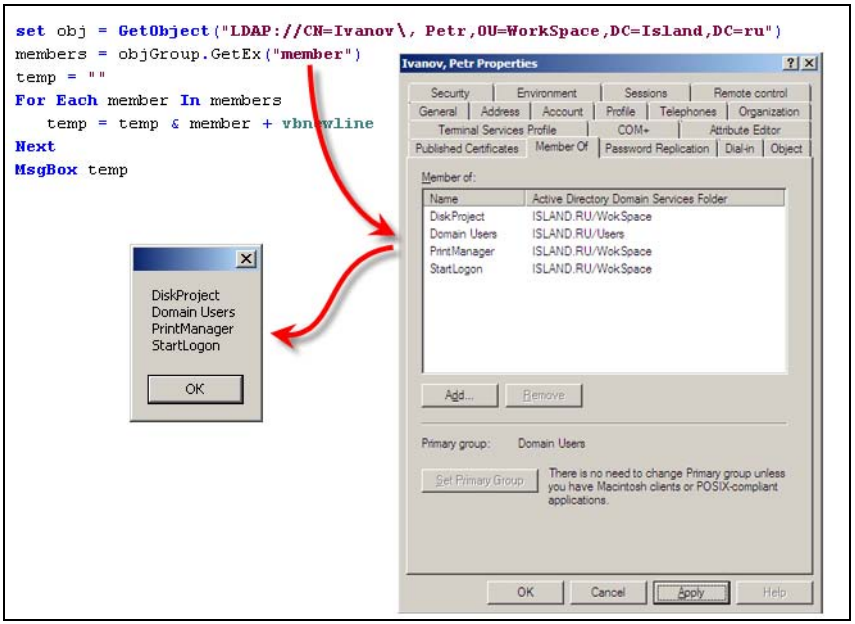



**Рис. 4.10.** Метод `GetEx()`

## Методы *Put()* и *PutEx()*

Метод `Put()` используется для присвоения значений атрибутам объекта в кэше. Шаблон доступа приведен в листинге 4.3. Этот метод вызывается непосредственно перед записью данных в каталог Active Directory методом `SetInfo()`.

**Листинг 4.3. Шаблон вызова метода `Put()`**

```
path = "…"
Set obj = GetObejct(path)
…
obj.Put "PROPERTY", "VALUE"
obj.SetInfo()
```

Особое внимание стоит уделить методу `PutEx()`. Как и метод `Put()`, он предназначен для присвоения значения атрибута в кэше свойств объекта. При этом в зависимости от значения первого параметра метода `PutEx()` с указанным в качестве второго параметра атрибутом выполняется одно из действий, перечисленных в табл. 4.5.

***Таблица 4.5.*** *Действия, выполняемые методом* `PutEx()`

| Значение первого параметра | Выполняемое действие |
|---|---|
| 1 | Обнуление значения указанного свойства. Если тип данных атрибута — массив, то уничтожаются все элементы массива |
| 2 | Обновление значения с заменой его указанным новым значением |
| 3 | Добавление нового значения атрибута к существующему. Данное действие характерно только для массивов данных |
| 4 | Удаление указанного значения из существующего набора значений. Данное действие характерно только для массивов данных |

Шаблон вызова метода `PutEx()` приведен в листинге 4.4.

**Листинг 4.4. Шаблон вызова метода `PutEx()`**

```
path = "…"
Set obj = GetObejct(path)
…
obj.PutEx "FLAG", "PROPERTY", "VALUE"
obj.SetInfo()
```

В приведенном листинге параметр *FLAG* идентифицирует действие, производимое со значением (*VALUE*) указанного параметра (*PROPERTY*), в соответствии с табл. 4.5.

На рис. 4.11 приведена схема добавления учетной записи пользователя в новую группу, при этом в каталоге Active Directory происходит добавление

элемента в массив `Member`. Для выполнения поставленной задачи используется функция `PutEx()` с первым параметром `FLAG=3`.

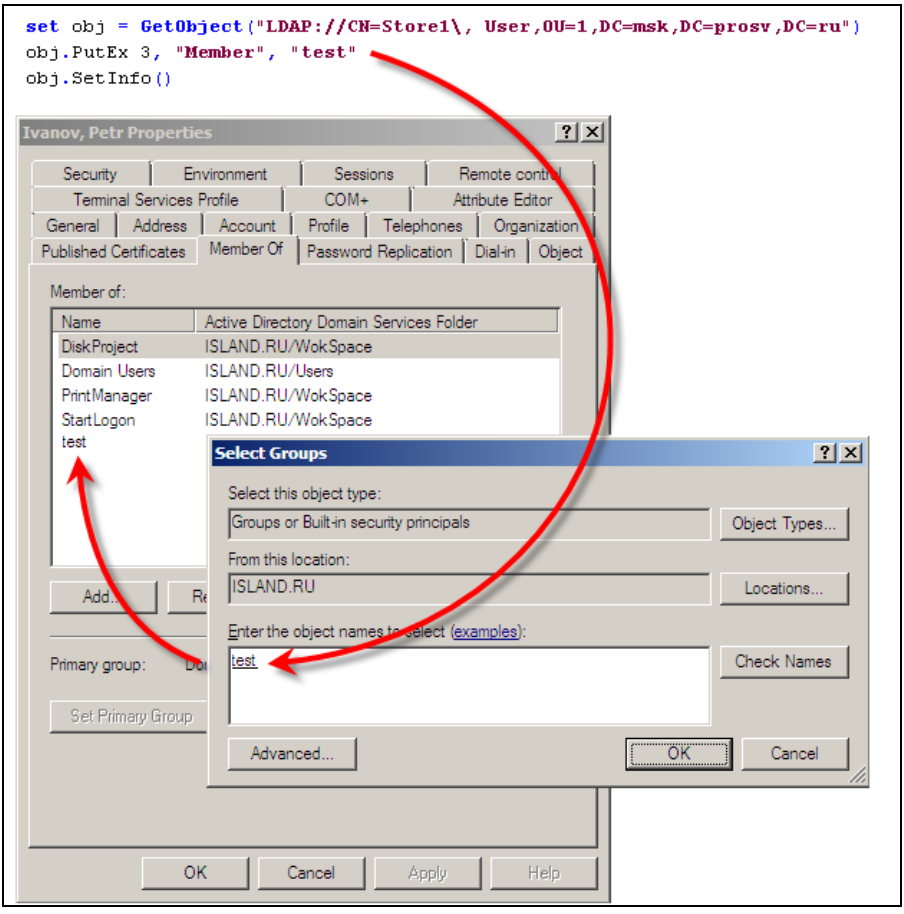



**Рис. 4.11.** Метод `PutEx()`

## Объектная модель *IADsContainer*

Интерфейс `IADsContainer` поддерживает набор свойств и методов для контейнеров в каталоге Active Directory. Чаще всего свойства и методы этого интерфейса используются для решения двух задач:

- получение доступа к элементам в указанном контейнере;

- осуществление манипуляций с объектом: создание, удаление, копирование или перемещение.

## Свойства интерфейса *IADsContainer*

Контейнер `IADsContainer` поддерживает всего лишь одно свойство — `Filter`, которое позволяет отфильтровать элементы указанного класса или классов.

### Свойство *Filter*

В домене существует огромное количество объектов, поэтому для увеличения скорости работы сценария целесообразно использовать фильтр. В качестве фильтра может быть задано значение одного из элементов массива `objectClass`, причем в списке присутствуют только те элементы, которые находятся непосредственно в указанном контейнере, а не во всем нижестоящем дереве. Возможные варианты были приведены в табл. 4.3. Шаблон использования этого свойства приведен в листинге 4.5.

**Листинг 4.5. Шаблон использования свойства `Filter`**

```
path = "…"
Set obj = GetObejct(path)
obj.Filter = Array("TYPE")
For Each element In obj
    t = element.PROPERTY
Next
```

В приведенном шаблоне *`TYPE`* — это тип объекта (см. табл. 4.3), а *`PROPERTY`* — название свойства согласно объектной модели выбранного объекта.

## Методы интерфейса *IADsContainer*

Краткое описание методов, поддерживаемых интерфейсом `IADsContainer`, приведено в табл. 4.6.

***Таблица 4.6.*** *Методы интерфейса* `IADsContainer`

| Метод | Описание |
|---|---|
| `GetObject()` | Возвращает ссылку на объект в указанном контейнере |

***Таблица 4.6*** *(окончание)*

| Метод | Описание |
|---|---|
| `Create()` | Создает объект указанного типа в данном контейнере |
| `Delete()` | Удаляет объект в контейнере |
| `CopyHere()` | Создает копию объекта |
| `MoveHere()` | Перемещает объект в указанный контейнер |

## Метод *GetObject()*

Этот метод позволяет получить ссылку на объект. В качестве параметра используется составной путь к объекту и провайдер доступа к нему в соответствии с шаблоном, приведенным в листинге 4.6.

**Листинг 4.6. Шаблон использования метода `GetObject()`**

```
path = "PROVIDER://PATH_TO_OBJECT"
Set obj = GetObject(path)
Set adobj = obj.GetObject(OBJECT_TYPE, VALUE)
…
```

В качестве значения *PROVIDER* может быть указан один из приведенных в табл. 4.7 протоколов, значение параметра *PATH_TO_OBJECT* — составной путь к объекту. *OBJECT_TYPE* — тип объекта, например "user", "group", "computer" и т. д., *VALUE* — выражение вида "cn=*OBJECT_NAME*", где *OBJECT_NAME* — имя объекта, например aivanov, domain users, 1001pc.

***Таблица 4.7.*** *Протоколы доступа и их назначение*

| Протокол | Название | Описание |
|---|---|---|
| GC: | Global Catalog | Обеспечивает доступ к глобальному каталогу домена |
| LDAP: | Lightweight Directory Access Protocol | Обеспечивает доступ к объектам указанного леса в домене Windows 2k |
| WinNT: | Windows NT | Обеспечивает доступ к объектам домена Windows NT. Устаревший протокол |

***Таблица 4.7*** *(окончание)*

| Протокол | Название | Описание |
|---|---|---|
| IIS: | Internet Information Service | Обеспечивает доступ к объектам сайта и виртуальным каталогам, созданным в IIS |

После получения доступа к объекту осуществляется чтение характеристик объекта или другие манипуляции: удаление, создание и т. д. Чтение характеристик осуществляется с помощью свойств и методов интерфейса `IADs`, операции с объектами (создание, удаление, перемещение, копирование и др.) осуществляются с помощью `IADsContainer`.

## Метод *Create()*

Метод `Create()` позволяет создавать новые элементы в указанном контейнере. Вызывая метод, необходимо указать два обязательных параметра: имя класса создаваемого объекта и его относительное составное имя. Шаблон создания объекта приведен в листинге 4.7. Значение первого параметра *`CLASS`* — идентификатор объекта (см. табл. 4.3). Второй параметр *`RDN_NAME`* задает имя объекта, возвращаемое при помощи свойства `IADs.Name`, например, `cn = Petr, Ivanov`. Объект создается в кэше, поэтому для его записи в каталог необходимо вызвать метод `SetInfo()`.

**Листинг 4.7. Шаблон использования метода `Create()`**

```
path = "…"
Set obj = GetObject(path)
obj.Create("CLASS", "RDN_NAME")
obj.Put …
…
obj.setInfo
```

## Метод *Delete()*

С помощью метода `Delete()`удаляют указанный объект из каталога Active Directory. При вызове метода необходимо указать два обязательных параметра, аналогичных параметрам метода `Create()`. Первый параметр — класс

удаляемого объекта. Если класс отключен, то вместо него можно указать значение `NULL`. Второй параметр — относительное составное имя объекта.

В листинге 4.8 приведен шаблон удаления объекта с помощью метода `Delete()`. Обратите внимание, что в отличие от метода `Create()`, при использовании метода `Delete()` метод `SetInfo()` не вызывается.

**Листинг 4.8. Шаблон использования метода `Delete()`**

```
path = "…"
Set obj = GetObject(path)
obj.Create("CLASS", "RDN_NAME")
```

### Метод *CopyHere()*

Метод `CopyHere()` позволяет скопировать элемент внутри данного контейнерного объекта. Сразу отметим, что использовать его можно только при подключении к службе каталогов NDS (см. *разд. "Провайдеры ADSI" в главе 1*). При использовании любого другого провайдера попытка вызвать этот метод вернет ошибку "Не поддерживается".

### Метод *MoveHere()*

Перемещение объекта в каталоге Active Directory осуществляется с помощью метода `MoveHere()`. Для обеспечения процесса надо получить доступ к объекту, который необходимо перенести, и доступ к контейнеру, в который он переносится. В листинге 4.9 приведен шаблон перемещения объекта из одной папки (`path1`) в другую (`path2`).

**Листинг 4.9. Шаблон использования метода `MoveHere()`**

```
path1 = "…"
objname = "…"
set obj1 = GetObject(objname & path1)
path2 = "…"
set obj2 = GetObject(path2)
obj2.MoveHere obj1.adsPath, vbNullString
obj.Create("CLASS", "RDN_NAME")
```

На практике метод `MoveHere()` часто используется для переименования объектов. В этом случае вторым параметром является относительное составное имя объекта (рис. 4.12).

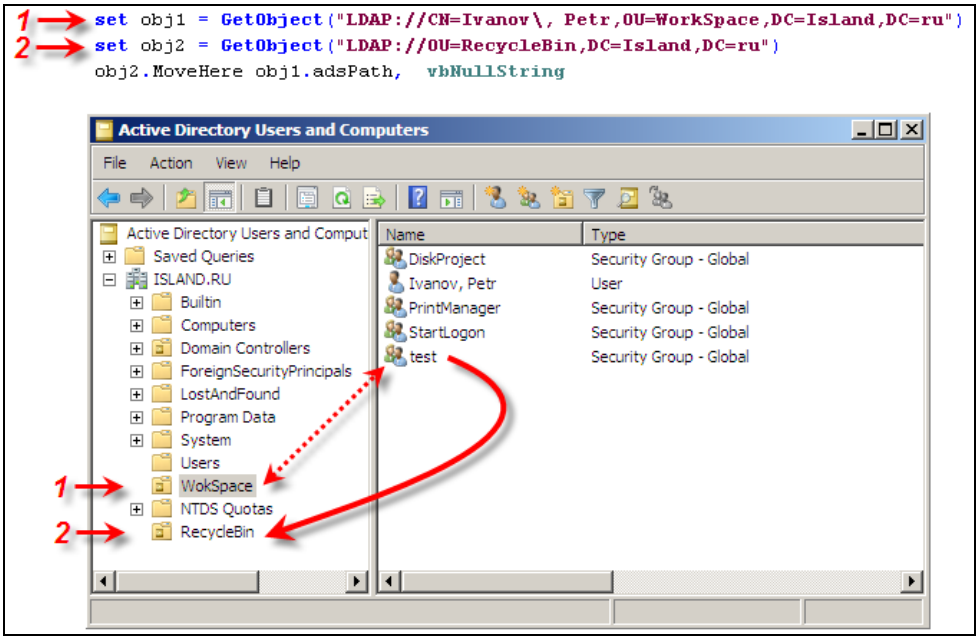



**Рис. 4.12.** Перемещение объектов с помощью функции `MoveHere()`

**Листинг 4.10. Перемещение объекта на примере учетной записи пользователя**

```
Set oRoot = GetObject("LDAP://rootDSE")
Set oUser = GetObject("LDAP://cn=Ivanov\, Petr," & _
                oRoot.Get("defaultNamingContext"))
Set oOu = GetObject("LDAP://ou=Дирекция," & _
                oRoot.Get("defaultNamingContext"))
oOu.MoveHere oUser.ADsPath, vbNullStaring
```

В листинге 4.10 учетная запись пользователя перемещается из корневого каталога Active Directory в контейнер Дирекция.

Если объект нужно перенести и переименовать одновременно, то в качестве второго аргумента функции `MoveHere` используется новое имя объекта:

```
oOu.MoveHere oUser.ADsPath, "Petrov, Vasily"
```

# Объектная модель *IADsNamespaces*

Один из основных объектов — `IADsNamespaces`, позволяющий получить список доступных протоколов, с помощью которых можно получить доступ к Active Directory. На практике обращение к этому объекту происходит крайне редко. Объект обладает двумя методами и не поддерживает свойств.

## Методы интерфейса *IADsNamespaces*

Объекты, реализующие интерфейс `IADsNamespaces`, поддерживают два метода[1], краткая характеристика которых приведена в табл. 4.8. Необходимо отметить, что оба метода вызываются в неявном виде, то есть конструкций `.Get()` или `.Put()` в приведенных листингах вы не увидите.

***Таблица 4.8.*** *Методы интерфейса* `IADsNamespaces`

| Метод | Описание |
|---|---|
| `Get()` | Получение доступных пространств имен |
| `Put()` | Получение доступа к указанному пространству имен |

### Метод *Get()*

Метод `Get()` используется для определения списка доступных провайдеров. Для этого используется вызов объекта ADs в явном виде. Пример получения списка доступных пространств имен приведен в листинге 4.11.

[1] Написанное здесь не вполне точно. Неявные методы для чтения и установки значения поддерживаются всеми объектами VBScript, и приписывать их исключительно интерфейсу `IADsNamespaces` нелогично. — *Прим. ред.*

**Листинг 4.11. Получение доступных пространств имен с помощью метода `Get()`**

```
Set obj = GetObject("ADs:")
For Each provider In obj
    WScript.Echo provider.Name
Next
```

### Метод *Put()*

Самый используемый метод в программировании ADSI — метод `Put()`. Практически все сценарии, связанные с программным управлением каталога Active Directory, используют этот метод. Пример использования метода `Put()` приведен в листинге 4.12.

**Листинг 4.12. Подключение к пространству имен LDAP с помощью метода `Put()`**

```
Set obj = GetObject("LDAP://OU=WorkSpace,DC=Island,DC=ru")
```

## Объектная модель *IADsOpenDSObject*

Объект `IADsOpenDSObject` предназначен для авторизованного подключения к каталогу Active Directory. Он используется для соединения с указанной базой данных, в частности — с Active Directory.

### Методы интерфейса *IADsOpenDSObject*

Объект `IADsOpenDSObject` поддерживает всего один метод — `OpenDSObject()`, используемый для авторизованного подключения к каталогу Active Directory с привилегиями альтернативной учетной записи. Зачастую она должна быть наделена административными правами.

### Метод *OpenDSObject()*

По своей сути метод `OpenDSObject()` делает то же, что и вызов метода `GetObject("OBJECT_PATH")`. Принципиальное отличие заключается в том, что с помощью метода `OpenDSObject` можно создать соединение с каталогом Ac-

tive Directory от имени определенного пользователя (листинг 4.13). Вызов метода `OpenDSObject()` имеет следующий синтаксис:

```
OpenDSObject(ObjectPath, UserLogin, UserPwd, AuthFlag)
```

Здесь используются следующие параметры:

- `ObjectPath` — составной путь к объекту в соответствии с RFC 1779 и 2247;
- `UserLogin` — имя учетной записи в домене, например, "Island\PIvanov";
- `UserPwd` — пароль учетной записи в незашифрованном виде. После создания сценария настоятельно рекомендуется его зашифровать;
- `AuthFlag` — уровень аутентификации учетной записи пользователя. Для доступа под именем системного администратора необходимо использовать константу `ADS_SECURE_AUTHENTICATION`, равную 1.

**Листинг 4.13. Создание соединения с Active Directory от имени указанной учетной записи пользователя**

```
set root = GetObject("LDAP:")
set obj = Root.OpenDSObject("LDAP://OU=WorkSpace,DC=Island,DC=ru", _
            "Island\PIvanov", "123456789", 100)
WScript.Echo obj.cn
```

# Интерфейсы свойств объектов

Четыре интерфейса предназначены для работы со свойствами объекта:

- `IADsPropertyList` предназначен для перечисления свойств объектов Active Directory;
- `IADsPropertyEntry` служит для работы со свойствами объекта Active Directory в кэше;
- `IADsPropertyValue` используется непосредственно для работы со значениями свойств, имеющих предопределенные типы данных;
- `IADsPropertyValue2` также используется непосредственно для работы со значениями свойств, но поддерживает определенные пользователем типы данных.

# Объектная модель *IADsPropertyList*

`IADsPropertyList` — один из основных интерфейсов, предназначенный для работы со свойствами объектов Active Directory. Почти всегда этот интерфейс вызывается неявно, однако в некоторых случаях использование интерфейса в явном виде оправдано. Ярким примером явного использования этого интерфейса может служить сценарий для получения исчерпывающей информации о свойствах указанного объекта — о названиях полей, соответствующих им типов данных, определения количества заданных свойств и т. д.

Использовать свойства и методы этого интерфейса можно только после вызова одного из методов `GetInfo()`, `GetInfoEx()`, `Get()` или `GetEx()`. Это объясняется тем, что данные `IADsPropertyList` берутся не из каталога Active Directory, а из кэша.

## Свойства *IADsPropertyList*

`IADsPropertyList` поддерживает лишь одно свойство — `PropertyCount`, с помощью которого определяют количество атрибутов объекта в каталоге Active Directory.

### Свойство *PropertyCount*

`PropertyCount` — единственное свойство интерфейса `IADsPropertyList`. Оно возвращает количество свойств, поддерживаемых указанным объектом (листинг 4.14).

**Листинг 4.14. Определение количества свойств объекта**

```
On Error Resume Next
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
WScript.Echo obj.PropertyCount
```

## Методы *IADsPropertyList*

В интерфейсе `IADsPropertyList` определено восемь методов (табл. 4.9).

***Таблица 4.9.*** *Методы интерфейса* `IADsPropertyList`

| Методы | Описание |
|---|---|
| `GetPropertyItem()` | Возвращает значение указанного по имени свойства объекта |
| `Item()` | Возвращает значение свойства объекта, указанного по порядковому номеру свойства или его имени |
| `Next()` | Переставляет курсор на следующий атрибут анализируемого объекта |
| `PurgePropertyList()` | Очищает кэш в оперативной памяти компьютера, на котором выполняется сценарий |
| `PutPropertyItem()` | Присваивает новое значение указанному по имени свойству объекта Active Directory |
| `Reset()` | Устанавливает позицию курсора на исходную позицию |
| `ResetPropertyItem()` | Удаляет из кэша указанное свойство по его имени или номеру |
| `Skip()` | Перемещает курсор в списке свойств на указанное количество позиций |

## Метод *GetPropertyItem()*

Метод `GetPropertyItem()` возвращает значение указанного атрибута объекта:

```
GetPropertyItem(FieldName, FieldType)
```

При вызове метода используются два аргумента:

- `FieldName` — имя атрибута объекта;
- `FieldType` — тип данных значения атрибута.

В большинстве случаев тип данных значения атрибута неизвестен, поэтому в качестве значения параметра `FieldType` задают специальное значение `ADSTYPE_UNKNOWN`, равное 26.

Метод `GetPropertyItem()` возвращает данные в виде ссылки на объект `IADsPropertyEntry`, который можно использовать для получения информации о значении свойства. Его использование проиллюстрировано в листинге 4.15.

**Листинг 4.15. Использование метода `GetPropertyItem()`**

```
Const ADSTYPE_UNKNOWN = 26
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
Set entry = obj.GetPropertyItem("cn", ADSTYPE_UNKNOWN)
For Each element In entry.Values
    Set value = element
    Wscript.Echo value.CaseIgnoreString
Next
```

## Метод *Item()*

Назначение метода `Item()` аналогично методу `GetPropertyItem()`. Отличие заключается в том, что этот метод позволяет получать значение свойства, указав как его номер (листинг 4.16а), так и его имя (листинг 4.16б). Тип свойства передавать при этом не надо.

**Листинг 4.16а. Чтение значения свойства, указанного по имени**

```
Set ent=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
ent.GetInfo
Set obj = ent.Item("cn")
WScript.Echo obj.Name
WScript.Echo obj.ADsType
```

**Листинг 4.16б. Чтение значения свойств, указанных по номеру**

```
Const ADSTYPE_UNKNOWN = 26
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
For i = 0 To obj.PropertyCount - 1
    Set Entry = obj.Item(i)
    a = "Свойство: " & CStr(i+1)
    b = "Имя: " & Entry.name
    c = "Тип данных: " & Entry.AdsType
    WScript.Echo a & vbNewLine & b & vbNewLine & c
Next
```

## Метод *Next()*

Метод `Next()` позволяет вернуть следующее свойство в виде объекта `IADsPropertyEntry`. Обычно он используется для перебора всех свойств какого-либо объекта Active Directory (листинг 4.17). Рисунок 4.13 иллюстрирует значения свойств объекта.

**Листинг 4.17. Использование метода `Next()` для перебора всех элементов объекта**

```
On Error Resume Next
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
Do While (Err.number=0)
    Set Entry = obj.Next()
    WScript.Echo "Свойство: "& Entry.Name & vbNewLine & _
        "Тип данных: " & Entry.AdsType
Loop
```

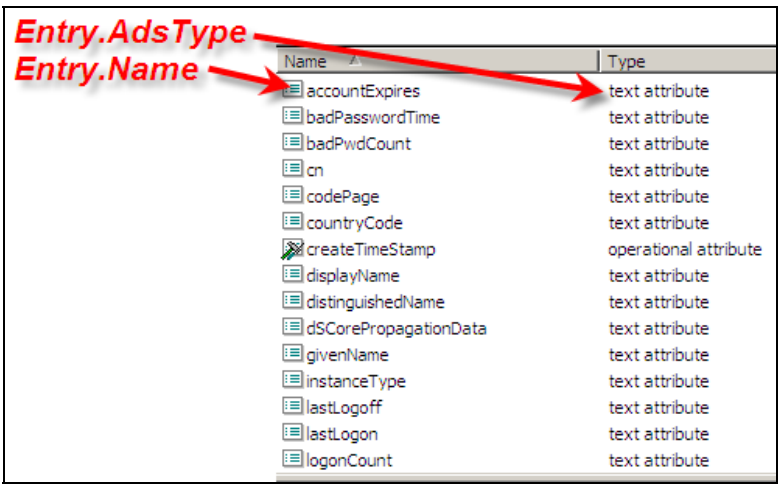



**Рис. 4.13.** Список атрибутов объекта и соответствующих им типов данных

Получение значений всех свойств выглядит немного сложнее и производится средствами интерфейсов `IADsPropertyValue` и `IADsPropertyValue2`, которые описаны далее.

### Метод *PurgePropertyList()*

С помощью метода `PurgePropertyList()` очищают кэш (листинг 4.18). Как видно из приведенного листинга, он по своему действию противоположен методу `GetInfo()`, который загружает данные из каталога Active Directory в оперативную память.

При вызове этого метода каталог Active Directory никак не затрагивается. Метод работает только с представлением данного объекта в оперативной памяти.

**Листинг 4.18. Использование метода `PurgePropertyList()` для очистки кэша**

```
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
…
obj.PurgePropertyList
```

### Метод *PutPropertyItem()*

По своим возможностям метод `PutPropertyItem()` очень похож на метод `Put()` интерфейса `IADs`. Он позволяет присвоить новое значение параметру выбранного объекта Active Directory. Так же, как и метод `Put()`, `PutPropertyItem()` работает с кэшем, в который загружены параметры объекта Active Directory. После его вызова необходимо использовать метод `SetInfo()` для записи нового значения в каталог. В качестве параметра метод `PutPropertyInfo()` принимает объект `PropertyEntry`, в котором должно быть настроено новое значение для свойства.

На практике этот метод используется очень редко, поскольку его использование требует более длинного кода, чем метод `Put()`. К тому же, необходимо четко знать тип данных назначаемого объекта, а при использовании метода `Put()` в этом нет необходимости.

### Метод *Reset()*

Метод `Reset()` позволяет установить курсор в начало списка свойств выбранного объекта. Обычно он используется после одного или нескольких вызовов метода `Next()`, чтобы вернуться к исходному состоянию.

### Метод *ResetPropertyItem()*

С помощью метода `ResetPropertyItem()` обнуляют значение указанного свойства, находящегося в оперативной памяти компьютера (листинг 4.19), на котором выполняется сценарий. Ему можно передать как имя свойства, так и его номер. Этот метод используется только для экономии оперативной памяти. На каталог Active Directory он никак не влияет.

**Листинг 4.19. Использование метода `ResetPropertyItem()`**

```
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
ResetPropertyItem("cn")
…
Set obj = Nothing
```

Если в сценарии дописать еще одну строку: `obj.cn`, то при выполнении сценария будет выведено сообщение об ошибке (рис. 4.14), поскольку значение параметра `cn` удалено из оперативной памяти компьютера. Возникновение такой ошибки наглядно иллюстрирует работу метода `ResetPropertyItem()`.

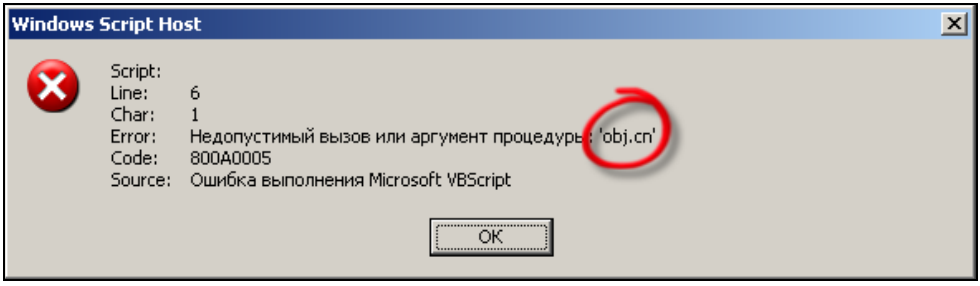



**Рис. 4.14.** Ошибка, возникающая при обращении к свойству после вызова метода `ResetPropertyItem()`

### Метод *Skip()*

С помощью метода `Skip()` перемещают курсор в списке свойств объекта на указанное количество позиций. В качестве параметра этого метода выступает число, на которое осуществляется сдвиг курсора. Обычно `Skip()` использует-

ся вместо метода `Next()` или вместе с ним для сокращения количества выполняемых операций (листинг 4.20).

**Листинг 4.20. Использование метода `Skip()`**

```
On Error Resume Next
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
Do While (Err.Number=0)
    Set Entry = obj.Skip(1)
    If Ent.Name = "uSNCreated" then
        WScript.Echo "Свойство uSNCreated" & vbNewLine & _
            "Тип данных: " & Ent.AdsType
    End if
Loop
```

# Объектная модель *IADsPropertyEntry*

Интерфейс `IADsPropertyEntry` предназначен для работы со значениями свойств объекта Active Directory. Его можно использовать для считывания существующего значения указанного свойства или для назначения нового значения.

## Свойства интерфейса *IADsPropertyEntry*

Для этого интерфейса предусмотрено 4 свойства, которые описаны в табл. 4.10.

***Таблица 4.10.*** *Свойства интерфейса* `IADsPropertyEntry`

| Свойство | Описание |
|---|---|
| `Name` | Имя свойства, соответствующее названию атрибута объекта Active Directory |
| `ADsType` | Тип свойства в соответствии со значением константы |
| `ControlCode` | Тип выполняемой операции |
| `Values` | Значение указанного свойства |

### Свойство *Name*

При переборе объектов с помощью свойства `Name` определяют имя объекта согласно схеме Active Directory. Приведем пример, в котором определяются имена всех атрибутов, имеющих указанный тип данных (листинг 4.21), в данном случае — тип `ADSTYPE_UTC_TIME`, которому соответствует значение 9 свойства `AdsType` (см. табл. 4.1).

**Листинг 4.21. Определение всех названий атрибутов объекта типа `UTC_TIME`**

```
On Error Resume Next
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
Do While Err.Number=0
    Set Entry = obj.Next()
    If Entry.ADsType=9 then
        WScript.Echo "Свойство: "& Entry.Name
    End If
Loop
```

### Свойство *ADsType*

Определяет тип считываемого объекта согласно схеме Active Directory. Свойство `ADsType` возвращает число, значение которого соответствует тому или иному типу данных. В листинге 4.22 приведен сценарий, возвращающий соответствующий типу данных числовой код (см. табл. 4.1).

**Листинг 4.22. Определение типа данных указанного атрибута**

```
Const ADSTYPE_UNKNOWN = 26
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
Set ent = obj.Item("createTimeStamp")
WScript.Echo ent.ADsType
```

### Свойство *ControlCode*

С помощью свойства `ControlCode` устанавливается режим обновления указанного свойства выбранного объекта.

Допустимые значения перечислены в табл. 4.11, а пример использования этого свойства приведен в листинге 4.23.

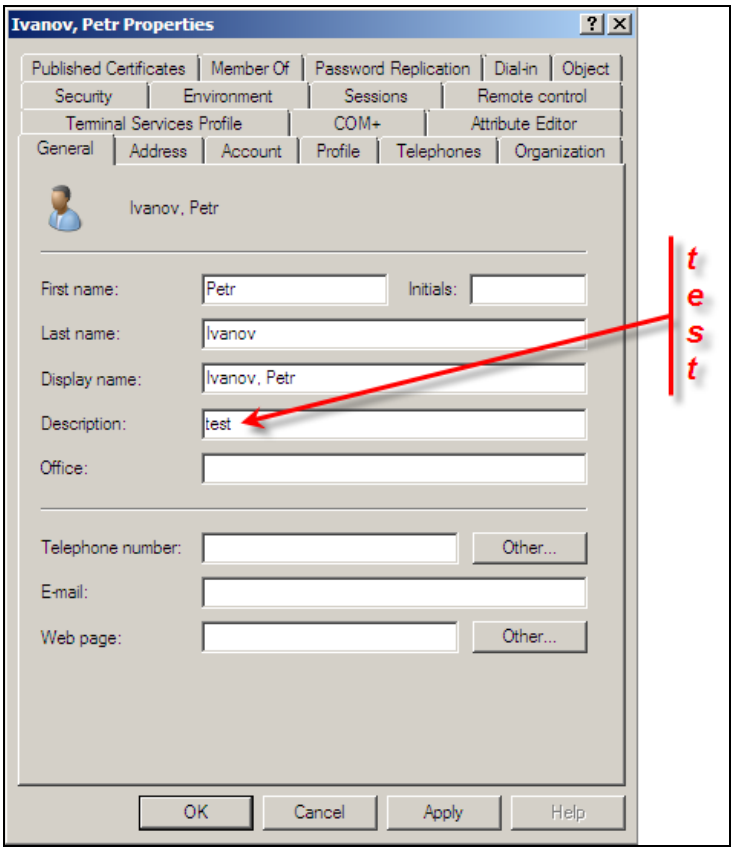



**Рис. 4.15.** Присвоение нового значения атрибуту `description`

***Таблица 4.11.*** *Значения свойства* `ControlCode`

| Константа | Значение | Описание |
|---|---|---|
| `ADS_PROPERTY_CLEAR` | 1 | Очистить свойство, удалив все указанные для него ранее значения |
| `ADS_PROPERTY_UPDATE` | 2 | Заменить существующее значение свойства другим значением |

***Таблица 4.11*** *(окончание)*

| Константа | Значение | Описание |
|---|---|---|
| ADS_PROPERTY_APPEND | 3 | Добавить указанное значение к существующим |
| ADS_PROPERTY_DELETE | 4 | Убрать указанное значение из списка значений свойства |

**Листинг 4.23. Присвоение нового значения атрибута с помощью свойства `ControlCode`**

```
Set ent=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
ent.GetInfo
Set obj = ent.Item("description")
obj.Values=Array("t","e","s","t")
obj.ControlCode=3
ent.PutPropertyItem(obj)
ent.SetInfo
```

На рис. 4.15 проиллюстрирован результат выполнения кода из листинга 4.23.

## Свойство *Values*

С помощью свойства `Values` определяют значения свойства. По своей сути это массив, элементы которого — символы, в сумме составляющие значение параметра. В листинге 4.24 приведен пример чтения значения атрибута `sAMAccountName` учетной записи пользователя.

**Листинг 4.24. Чтение значения атрибута объекта с помощью свойства `Values`**

```
Const ADSTYPE_UNKNOWN = 26
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
Set entry = obj.GetPropertyItem("sAMAccountName", ADSTYPE_UNKNOWN)
For Each element In entry.Values
    Set value = element
    WScript.Echo value.CaseIgnoreString
Next
```

# Объектные модели *IADsPropertyValue* и *IADsPropertyValue2*

Интерфейсы `IADsPropertyValue` и `IADsPropertyValue2` используются непосредственно для чтения значений указанных атрибутов. Каждому объекту `IADsPropertyValue` соответствует конкретное значение свойства. Массив значений (то есть массив объектов `IADsPropertyValue`) передается при помощи свойства `Values` объекту `IADsPropertyEntry`.

## Свойства *IADsPropertyValue*

Объекты, реализующие интерфейс `IADsPropertyValue`, поддерживают 13 свойств, причем 12 из них используются для чтения атрибута того или иного типа данных (табл. 4.12).

***Таблица 4.12.*** *Свойства* `IADsPropertyValue`

| Свойство | Описание |
|---|---|
| `AdsType` | Тип данных свойства. Возвращаемые коды описаны в табл. 4.1 |
| `DNString` | Составной путь к объекту в каталоге Active Directory, например, `OU=WorkSpace,CN=Island,CN=Ru` |
| `CaseExtractString` | Строка, символы которой чувствительны к регистру |
| `CaseIgnoreString` | Строка, символы которой не чувствительны к регистру |
| `PrintableString` | Строка, предназначенная для отображения на экране или для печати на принтере |
| `NumericString` | Строка, символы которой — цифры, например, 10237645 |
| `Boolean` | Булево значение, то есть `TRUE` или `FALSE` |
| `Integer` | Целое число |
| `OctetString` | Строка в бинарном виде |
| `UTCTime` | Данные в формате UTC Time |
| `LargeInteger` | Длинное целое число |
| `SecurityDescription` | Параметры безопасности объекта |

Поскольку все свойства, кроме первого, предназначены для чтения того или иного типа данных, рассмотрим эти свойства вместе.

### Свойство *AdsType*

Свойство `AdsType` предназначено для определения типа данных указанного значения. В коде, приведенном в листинге 4.25, тип данных возвращается инструкцией `element.AdsType`.

**Листинг 4.25. Использование свойства `AdsType`**

```
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
Set entry = obj.Item("cn")
For Each element In entry.Values
    WScript.Echo element.AdsType
Next
```

### Свойства, обеспечивающие чтение данных

В зависимости от типа данных, соответствующего тому или иному атрибуту, осуществляется чтение его значения с помощью одного из указанных в табл. 4.12 свойств. Так, для чтения параметра `distinguishedName` (листинг 4.26), соответствующего типу `ADSTYPE_DN_STRING = 1`, необходимо использовать свойство `DNString` (см. табл. 4.1 и табл. 4.12).

**Листинг 4.26. Использование свойства `DNString` для чтения атрибута типа `ADSTYPE_DN_STRING`**

```
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.getinfo
Set entry = obj.Item("distinguishedName")
For Each element In entry.Values
    WScript.Echo element.DNString
Next
```

## Методы *IADsPropertyValue*

В интерфейсе `IADsPropertyValue` определено 3 метода (табл. 4.13).

***Таблица 4.13.*** *Методы* `IADsPropertyValue`

| Свойство | Описание |
|---|---|
| `Clear()` | Позволяет очистить все значения, настроенные ранее для объекта `PropertyValue` |
| `GetObjectProperty()` | Позволяет получить значение атрибута объекта Active Directory |
| `PutObjectProperty()` | Позволяет настроить значение для атрибута объекта Active Directory |

### Метод *Clear()*

С помощью метода `Clear()` очищают значения, настроенные заранее для объекта `IADsPropertyValue`. На практике этот метод используется достаточно редко.

### Метод *GetObjectProperty()*

Метод `GetObjectProperty()` используется для получения значения указанного атрибута. В качестве аргумента указывается число, соответствующее типу данных атрибута (см. табл. 4.1). В приведенном в листинге 4.27 коде параметру `modifyTimeStamp` соответствует тип данных `ADSTYPE_UTC_TIME = 9`.

**Листинг 4.27. Использование метода `GetObjectProperty()`**

```
Const ADSTYPE_UTC_TIME = 9
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
Set entry = obj.Item("modifyTimeStamp")
For Each element In entry.Values
    WScript.Echo element.GetObjectProperty(ADSTYPE_UTC_TIME)
Next
```

### Метод *PutObjectProperty()*

Метод `PutObjectProperty()`, имеющий два аргумента, используется для присвоения нового значения переменной (листинг 4.28). Первым аргументом указывается тип данных переменной, а вторым — новое значение атрибута в текстовом виде. При этом свойство `ControlCode` контролирует способ изменения значения.

**Листинг 4.28. Использование метода `PutObjectProperty()`**

```
NewValue = "Петр Иванов" ' Новое значение поля description
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
Set entry = obj.Item("description")
For Each element In entry.Values
    element.PutObjectProperty ADSTYPE_CASE_IGNORE_STRING, NewValue
Next
obj.SetInfo
```

# Интерфейсы объектов

Существуют пять основных интерфейсов объектов, которые предназначены для управления свойствами объектов (табл. 4.14). Изначально `IADs*`-интерфейсы были предназначены для управления доменом под управлением Windows NT. При проектировании Active Directory было необходимо обеспечить максимальную совместимость `IADs*`-интерфейса с новым каталогом. В результате в Active Directory появились поля-дубли (см. *разд. "Типы объектов Active Directory"* в *главе 1*).

В Windows NT домены управлялись с помощью провайдера WinNT, и при этом все объекты рассматривались как локальные. В Windows 2k все ресурсы рассматриваются как сетевые. Именно этот факт в настоящее время делает часть `IADs*`-интерфейсов практически неиспользуемыми, поскольку с помощью интерфейсов объектов можно получить значения ограниченного числа параметров (лишь параметров, используемых в доменах Windows NT), в то время как с помощью интерфейсов свойств можно получить весь спектр существующих параметров.

***Таблица 4.14.*** *Интерфейсы объектов*

| Интерфейс | Описание |
|---|---|
| `IADsComputer` | Чтение свойств учетной записи компьютера |
| `IADsDomain` | Определение характеристик домена |
| `IADsGroup` | Управление группами безопасности |
| `IADsOu` | Управление контейнерами |
| `IADsUser` | Управление учетной записью пользователя |

## Методы интерфейсов объектов

Все интерфейсы объектов поддерживают одни и те же методы — методы интерфейса `IADs`: `GetInfo()`, `SetInfo()`, `Get()`, `GetEx()`, `GetInfoEx()`, `Put()`, `PutEx()`. Использование этих методов было подробно рассмотрено в *разд. "Объектная модель IADs"*.

## Свойства интерфейсов объектов

Интерфейсы объектов поддерживают лишь небольшую часть свойств, которыми обладает объект. Это связано с тем, что изначально эти интерфейсы были разработаны для работы с доменами Windows NT. В настоящее время бóльшая часть из них утратила свой смысл, поэтому они рассматриваться не будут.

Отметим, что для всех объектов действительны свойства, приведенные в табл. 4.15.

***Таблица 4.15.*** *Общие свойства интерфейсов объектов*

| Свойство | Описание |
|---|---|
| `AdsPath` | Относительный составной путь к объекту |
| `Class` | Класс объекта. Последнее значение массива `objectClass` |
| `GUID` | Глобальный уникальный идентификатор объекта (см. *разд. "Globally Unique Identifier"* в *главе 12*) |
| `Name` | Имя объекта |
| `Parent` | Составное имя родительской папки объекта |

# Использование *IADs*-интерфейсов на практике

Для иллюстрации работы с интерфейсами семейства `IADs*` приведем сценарий (листинг 4.29), в котором выводятся все атрибуты и их значения для указанной учетной записи пользователя на экран. Алгоритм работы сценария следующий. На первом этапе с помощью функции `GetObject()` получается доступ к пространству имен объекта. Затем с помощью метода `GetInfo()` загружаются все данные из Active Directory в оперативную память компьютера. С помощью цикла `For…Next` осуществляется чтение всех названий атрибутов (`ent.Name`) и их свойств (`ent.AdsType`). По возвращенному типу свойств (`ent.AdsType`) на основе данных, приведенных в табл. 4.1 и 4.10, используя инструкцию `Select…Case`, вызывается свойство, характерное для данного типа объекта. Наконец, следует вывод полученной информации на экран (рис. 4.16). Каждое свойство, его значение и тип данных выводятся в своем диалоговом окне. Отметим, что данный сценарий является простейшей реализацией поставленной задачи, и не все параметры отображаются корректно (значения SID, GUID). Это сделано намеренно, чтобы не загромождать сценарий и показать способы взаимодействия семейства `IADs*`-интерфейсов.

**Листинг 4.29. Определение атрибутов и их значения для указанного объекта**

```
On Error Resume Next
Set obj=GetObject("LDAP://CN=Ivanov\, Petr,OU=WorkSpace,DC=Island,DC=ru")
obj.GetInfo
For i = 0 To obj.PropertyCount-1
    set ent = obj.Item(i)
    c = ent.Name + vbNewLine
    c = c + CStr(ent.AdsType) + vbNewLine
    select case ent.AdsType
    case 1
        Set o = obj.Item(ent.Name)
        For Each element In o.Values
            c = c & element.DNString + vbNewLine
        Next
    case 2
        Set o = obj.Item(ent.Name)
```

```
    For Each element In o.Values
        c = c + element.CaseExtractString + vbNewLine
    Next
case 3
    Set o = obj.Item(ent.Name)
    For Each element In o.Values
        c = c + element.CaseIgnoreString + vbNewLine
    Next
case 4
    Set o = obj.Item(ent.Name)
    For Each element In o.Values
        c = c + element.PrintableString + vbNewLine
    Next
case 5
    Set o = obj.Item(ent.name)
    For Each element In o.Values
        c = c + element.NumericString + vbNewLine
    Next
case 6
    Set o = obj.Item(ent.Name)
    For Each element In o.Values
        c = c + CStr(element.Boolean) + vbNewLine
    Next
case 7
    Set o = obj.Item(ent.Name)
    For Each element In o.Values
        c = c + CStr(element.Integer) + vbNewLine
    Next
case 8
    Set o = obj.Item(ent.Name)
    For Each element In o.Values
        c = c + CStr(element.OctetString) + vbNewLine
    Next
case 9
    Set o = obj.Item(ent.Name)
```

```
        For Each element In o.Values
            c = c + CStr(element.UTCTime) + vbNewLine
        Next
    case 10
        Set o = obj.Item(ent.Name)
        For Each element In o.Values
            set r=element.LargeInteger
            c = c + CStr(abs(r.HighPart*2^32 + r.LowPart)) + vbNewLine
        Next
    End Select
    WScript.Echo c
Next
```

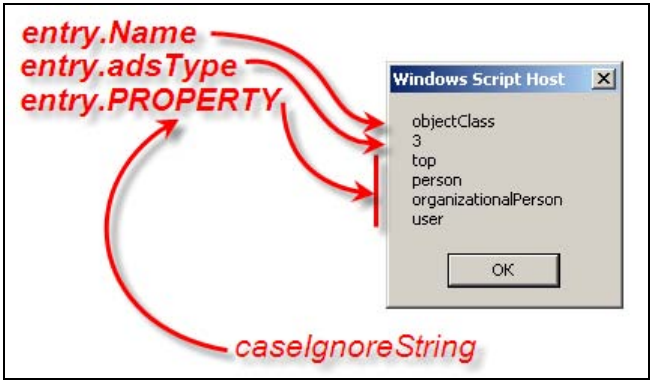



**Рис. 4.16.** Чтение атрибута пользователя

# Способы доступа к объектам Active Directory

Доступ к свойствам объектов каталога Active Directory можно получить с помощью одного из двух интерфейсов: `IADs` и специализированного интерфейса (табл. 4.16).

***Таблица 4.16.*** *Некоторые интерфейсы объектов*

| Интерфейс | Объект | Комментарий |
|---|---|---|
| `IADsUser` | Person | Учетная запись пользователя |
| `IADsComputer` | Computer | Учетная запись компьютера |
| `IADsGroup` | Group | Учетная запись группы |
| `IADsOU` | Organizational Unit | Контейнер (папка) |

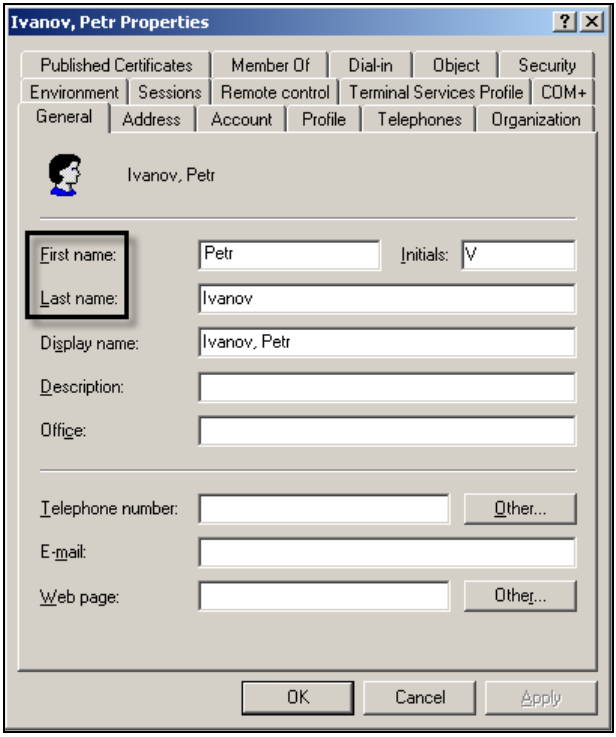



**Рис. 4.17.** Учетная запись пользователя

В этой книге уделено основное внимание интерфейсу `IADs`, поскольку с его помощью можно получить доступ к любому из объектов, независимо от его типа. Однако упомянуть о возможности использования узкоспециализированных интерфейсов необходимо, поскольку имена атрибутов Active Directory, используемых при работе с интерфейсом `IADs`, и свойства специализированных объектов `IADsUser`, `IADsComputer` и т. д.) различны.

Рассмотрим учетную запись пользователя Ivanov, Petr (рис. 4.17) и исправим его имя и фамилию двумя разными способами.

## Управление атрибутами с помощью интерфейса *IADs*

Для управления атрибутами любого объекта с помощью интерфейса `IADs` используются методы `Get()` и `Put()`. Метод `Get()` применяется для чтения атрибутов объекта. В листинге 4.30 он возвращает значения атрибутов, которые отображаются методом `WScript.Echo()` на экране.

**Листинг 4.30. Чтение параметров учетной записи пользователя. Интерфейс `IADs`**

```
Set objUser = GetObject( _
       "LDAP://CN=Ivanov\, Petr,OU=Workspace,DC=ISLAND,DC=RU")
WScript.Echo objUser.Get("givenName")
WScript.Echo objUser.Get("sn")
```

С помощью метода `Put()` атрибутам присваивают новые значения. После его использования нужно применить метод `SetInfo()` для записи изменений в каталог Active Directory (листинг 4.31).

**Листинг 4.31. Изменение параметров учетной записи пользователя. Интерфейс `IADs`**

```
Set objUser = GetObject( _
        "LDAP://CN=Ivanov\, Petr,OU=Workspace,DC=ISLAND,DC=RU")
objUser.Put "givenName", "Nikolay"
objUser.Put "sn", "Sidorov"
objUser.SetInfo
```

## Управление атрибутами с помощью интерфейса *IADsUser*

С помощью интерфейса `IADsUser` также можно и считывать, и записывать данные. Для этого необходимо после получения к нему доступа с помощью функции `GetObject()` считывать данные, используя соответствующее нуж-

ному параметру свойство объекта (листинг 4.32). Для изменения данных необходимо выбранному свойству присвоить новое значение. Чтобы обеспечить запись данных в каталог, также необходимо использовать метод `SetInfo()` (листинг 4.33).

**Листинг 4.32. Чтение параметров учетной записи пользователя. Интерфейс `IADsUser`**

```
Set objUser = GetObject( _
        "LDAP://CN=Ivanov\, Petr,OU=Workspace,DC=ISLAND,DC=RU")
wscript.Echo objUser.FirstName
wscript.Echo objUser.LastName
```

**Листинг 4.33. Изменение параметров учетной записи пользователя. Интерфейс `IADsUser`**

```
Set objUser = GetObject( _
       "LDAP://CN=Ivanov\, Petr,OU=Workspace,DC=ISLAND,DC=RU")
objUser.FirstName = "Nikolay"
objUser.LastName = "Sidorov"
objUser.SetInfo
```

ГЛАВА 5



# Объектная модель ADO DB

В этой главе описаны объектные модели объектов ADO — альтернативного способа соединения с каталогом Active Directory. Все свойства и методы объектной модели ADO приведены с учетом их поддержки провайдером `ADsDSOObject`. Приведено множество наглядных примеров и схем.

## Основные термины и понятия

- **ADO** (ActiveX Data Objects) — интерфейс программирования приложений для доступа к данным, основанный на технологии компонентов ActiveX. ADO позволяет представлять данные из разнообразных источников, в том числе и из Active Directory.
- **`ADsDSOObject`** — провайдер, используемый для доступа к полям каталога Active Directory через ADO.
- **SQL** (Structured Query Language) — универсальный язык структурированных запросов, используемый для создания, модификации и управления базами данных. SQL является информационно-логическим языком, а не языком программирования.

## Объекты ADO

В ADSI реализовано несколько технологий поиска. В 90% случаев поиск объектов в каталоге Active Directory осуществляется с помощью библиотеки Microsoft ActiveX Data Object (ADO).

С помощью библиотеки ADO можно получить доступ к огромному количеству типов баз данных, например, dBASE, Access, Excel, Oracle, Paradox, MS

SQL Server, Sybase, FoxPro, Active Directory Service, Microsoft Jet, Interbase, Informix, MySQL, в том числе и к Active Directory.

Объектная модель ADO полностью поддерживает OLE Automation, поэтому эту технологию можно использовать из любых языков программирования, которые поддерживают OLE Automation, в том числе из VBA, VBScript, JScript и т. п.

Для реализации подключения к той или иной базе необходимо наличие установленного OLE-провайдера. Для подключения к каталогу Active Directory все необходимые драйверы уже интегрированы в дистрибутив операционной системы и устанавливаются по умолчанию.

ADO поддерживает семь объектов, краткое описание которых приведено в табл. 5.1. На практике же используют в основном `ADODB.Connection` для подключения к службе каталогов, `ADODB.Command` — для формулирования запроса и его выполнения, а `ADODB.Recordset` — для получения результата.

***Таблица 5.1.*** *Объекты ADO*

| Объект | Описание |
|---|---|
| `ADODB.Connection` | Создает соединение с базой данных. В данном случае в качестве базы выступает каталог Active Directory |
| `ADODB.Command` | Обеспечивает выполнение сформированных запросов |
| `ADODB.Parameter` | Определяет различные параметры объектов |
| `ADODB.Recordset` | Устанавливает позицию в таблице, определенную командой объекта или SQL-запросом |
| `Fields` | Выделяет колонку в указанной таблице |
| `Property` | Служит для определения значения в выбранной колонке |
| `Error` | Предназначен для обработки ошибок |

Все объекты, перечисленные в таблице, тесно связаны между собой. Кроме того, каждый из перечисленных объектов имеет ряд свойств и методов, которые зачастую дублируются, как в объектах `Connection` и `Command`.

Рассматривать все свойства и методы всех объектов не имеет смысла, поскольку это только запутает администратора, поэтому приведем шаблон доступа к Active Directory, который прошел проверку временем, и опишем используемые свойства и методы.

Для доступа к каталогу Active Directory достаточно использовать объект `Connection` (листинг 5.1). Получив доступ к объекту, можно осуществлять не только чтение значений атрибутов, но и выполнять поиск объектов.

**Листинг 5.1. Использование объекта `Connection`**

```
' Пример чтения свойства Version объекта Connection
set obj = CreateObject("ADODB.Connection")
msgBox obj.Version
```

# Объект *Connection*

Объект `Connection`, для использования которого нужна библиотека msadoxx.dll, служит для создания соединения с базой данных. Подключение к объекту в VBScript осуществляется с помощью стандартного приема:

```
set obj = CreateObject("ADODB.Connection")
```

## Методы объекта *Connection*

Объект `Connection` поддерживает, помимо прочих, четыре метода: `Open()`, `Close()`, `Cancel()`, `Execute()`. Их описание приведено в табл. 5.2.

***Таблица 5.2.*** *Методы объекта* `Connection`

| Методы | Описание |
|---|---|
| `Open()` | Открывает соединение с базой данных |
| `Execute()` | Выполняет запросы, хранимые процедуры или любые другие команды, доступные провайдеру базы данных |
| `Close()` | Закрывает соединение с базой данных |
| `Cancel()` | Отменяет выполнение последнего вызова `Execute()` или `Open()`, если действие еще не завершено |

### Метод *Open()*

Метод `Open()` открывает сеанс (листинг 5.2) подключения к источнику данных. Соответствующая команда имеет следующий синтаксис:

```
Open(ConnectionString, UserID, Password, Options)
```

При обращении к методу используются четыре параметра:

- `ConnectionString` определяет параметры подключения к источнику данных. Если указано значение свойства `ConnectionString` объекта `Connection`, то оно используется как значение по умолчанию;
- `UserID` — имя учетной записи пользователя, используемое для создания соединения;
- `Password` — пароль пользователя;
- `Options` — описывает способ подключения к источнику данных. По умолчанию создается синхронное подключение (−1). Для асинхронного подключения параметру присваивают значение 16.

**Листинг 5.2. Использование метода `Open()`**

```
set obj = CreateObject("ADODB.Connection")
…
obj.Open "Active Directory Provider"
```

Все указанные параметры необязательны для использования. В том случае, если имя и пароль учетной записи пользователя заданы и с помощью параметра `ConnectionString`, и при вызове метода `Open()`, то используются параметры, заданные при вызове метода.

Закончив операции с открытым подключением, используйте метод `Close()` для освобождения всех связанных системных ресурсов.

## Метод *Execute()*

Метод `Execute()` выполняет запросы, запускает хранимые процедуры или любые другие команды, доступные провайдеру базы данных. Он возвращает объект `Recordset`, доступный только для чтения курсором Forward-only, если переданная команда возвращает записи. Для использования объекта `Recordset`, доступного для записи, необходимо создать его в явном виде, а затем воспользоваться его свойствами и методами.

Команда `Execute()` имеет следующий синтаксис:

```
Execute(CommandText, RecordsAffected, Options)
```

При ее вызове используются следующие параметры:

- `CommandText` — строка, содержащая имя таблицы или хранимой процедуры, команду SQL или другую команду провайдера. При создании соединений с Active Directory обычно используется оператор SQL `SELECT`;

- `RecordsAffected` — целое число, определяющее число записей, затронутых командой. Заполняется провайдером;
- `Options` — целое число, определяющее тип команды, то есть уточняет значение параметра `CommandText`. Параметр может принимать следующие значения:
  - 1 означает, что параметр `CommandText` содержит текстовое определение команды или хранимой процедуры;
  - 2 показывает, что параметр `CommandText` содержит таблицу, для возвращения всех записей которой создается SQL-запрос;
  - 4 значит, что параметр `CommandText` содержит хранимую процедуру;
  - 8 означает, что тип команды неизвестен (по умолчанию);
  - 16 задает асинхронное выполнение команды;
  - 128 указывает не возвращать строки.

  При создании соединения с Active Directory параметр `Options` нужно устанавливать равным двум (2).

### Метод *Close()*

Команда `Close()`, не имеющая аргументов, закрывает соединение с источником данных (листинг 5.3). Закрытие соединения не приводит к удалению объекта из памяти. Можно изменить его свойства, а затем открыть снова. При закрытии подключения закрываются также все активные наборы записей (объекты `Recordset`) для данного подключения. Объекты `Command`, связанные с данным подключением, после закрытия подключения, не будут связаны с данным объектом `Connection`.

**Листинг 5.3. Использование метода `Close()`**

```
set obj = CreateObject("ADODB.Connection")
…
obj.Close
set obj = Nothing
```

### Метод *Cancel()*

Отменяет выполнение последнего вызова `Execute()` или `Open()`, если осуществляемое ими действие еще не завершено.

## Свойства объекта *Connection*

Объект `Connection` имеет ряд очень важных свойств, краткое описание которых приведено в табл. 5.3.

***Таблица 5.3.*** *Свойства и методы объекта* `Connection`

| Свойства | Описание |
|---|---|
| `ConnectionTimeout` | Устанавливает или возвращает время ожидания подключения к базе данных, в секундах. По умолчанию значение равно 15 с |
| `CommandTimeout` | Устанавливает или возвращает время ожидания выполнения команды, в секундах. По умолчанию значение равно 30 с |
| `ConnectionString` | Определяет параметры подключения к базе данных |
| `Provider` | Устанавливает или возвращает имя провайдера, используемого для подключения к базе данных |
| `State` | Возвращает состояние объекта `ADO.Connection` |
| `Mode` | Определяет режим доступа для изменения данных в сеансе. По умолчанию значение равно 0 |
| `Version` | Возвращает версию ADO |

### Свойство *ConnectionTimeout*

Это свойство устанавливает или возвращает время в секундах, в течение которого ожидается подключение к базе данных. Данный параметр по умолчанию равен 15 с. Изменять значение этого свойства (листинг 5.4) рекомендуется только при наличии проблем, возникающих из-за плотного сетевого трафика или высокой загрузки сервера.

**Листинг 5.4. Пример изменения свойства `ConnectionTimeout`**

```
set obj = CreateObject("ADODB.Connection")
obj.CommandTimeout = 60
```

Если время истекло, а соединение не установлено, то интерпретатор языка программирования выдаст соответствующую ошибку. Если значение

ConnectionTimeout равно нулю, то ADO будет ждать отклика базы данных бесконечно. Это свойство доступно для записи только для закрытого соединения.

### Свойство *CommandTimeout*

Устанавливает или возвращает время в секундах, в течение которого ожидается выполнение команды. Значение параметра по умолчанию — 30 с. Рекомендуется изменять значение этого свойства (листинг 5.5) только в случае появления проблем, возникающих из-за плотного сетевого трафика или высокой загрузки сервера.

**Листинг 5.5. Пример изменения значения свойства CommandTimeout**

```
set obj = CreateObject("ADODB.Connection")
obj.CommandTimeout = 60
```

Если заданное время ожидания истекло, а выполнение команды ADO не закончилось, то ее выполнение прерывается и выводится соответствующее сообщение об ошибке. Если значение CommandTimeout равно нулю, то ADO ждет выполнения команды бесконечно.

Установка CommandTimeout объекта Connection никак не влияет на одноименный параметр объекта Command.

### Свойство *ConnectionString*

Свойство ConnectionString — одно из основных свойств, определяющее правила подключения к базе данных. Значение представляет собой пять параметров, разделенных символом точка с запятой, при этом порядок их расположения не важен:

- User ID — учетная запись пользователя в формате DOMAIN\LOGIN, например: ISLAND\PIvanov;
- Password — пароль учетной записи пользователя;
- Data Source — имя сервера;
- Initial Catalog — имя базы данных;
- Integrated Security — настройки безопасности.

Приведенный список параметров не претендует на полноту. На практике при создании соединения с Active Directory используют коллекцию свойств `Properties`.

### Свойство *Provider*

Значение этого обязательного свойства — имя провайдера, используемого для подключения к базе данных. По умолчанию, если значение не задано, в качестве провайдера используется `MSDASQL`. Для подключения к Active Directory используется провайдер `ADsDSOObject` (листинг 5.6). Он может быть задан как с помощью этого свойства, так и с помощью `ConnectionString`, поскольку подключение к каталогу Active Directory осуществляется по OLE DB.

**Листинг 5.6. Использование свойства `Provider`**

```
set obj = CreateObject("ADODB.Connection")
obj.Provider = "ADsDSOObject"
```

Свойство `Provider` используется только при подключении по OLE DB. При подключении по ODBC все необходимые для соединения параметры указываются через параметр `Connection.String`.

### Свойство *State*

Это свойство возвращает состояние объекта `ADO.Connection`. Обычно оно используется для диагностирования проблем, связанных с доступом к базе данных. Принимает это свойство одно из перечисленных значений:

- 0 — объект закрыт;
- 1 — объект открыт;
- 2 — объект соединяется;
- 4 — объект выполняет команду;
- 8 — объект выполняет выборку строк.

Существуют правила хорошего тона создания сценариев, которые требуют, чтобы дальнейшая работа сценария осуществлялась только после удачного завершения ранее стоящей функции. Для обработки события рекомендуется использовать свойство `State` (листинг 5.7). В листинге приведенного сценария осуществляется ожидание, пока объект не будет открыт.

**Листинг 5.7. Использование свойства `State`**

```
set obj = CreateObject("ADODB.Connection")
…
Obj.Execute ("SELECT name, description FROM…")
While obj.State<>1
    WScript.Sleep 500
Wend
…
```

### Свойство *Mode*

Определяет режим доступа для изменения данных во время сеанса связи. Параметр `Mode` может принимать одно из перечисленных значений:

- 0 — режим доступа не установлен или не определен (по умолчанию);
- 1 — доступ только для чтения;
- 2 — доступ только для записи;
- 3 — доступ для чтения и записи;
- 4 — не разрешено открывать соединение на чтение другим пользователям;
- 8 — не разрешено открывать соединение на запись другим пользователям;
- 12 — не разрешено открывать соединение другим пользователям;
- 16 — разрешает открывать соединение с любым видом доступа другим пользователям.

### Свойство *Version*

Это свойство определяет и выводит версию провайдера, используемого для соединения с базой данных (листинг 5.8).

**Листинг 5.8. Использование свойства `Version`**

```
set obj = CreateObject("ADODB.Connection")
msgBox obj.Version
```

## Коллекции объекта *Connection*

Объект `Connection` имеет всего две коллекции, назначение которых приведено в табл. 5.4.

***Таблица 5.4.*** *Коллекции объектов* `Connection`

| Коллекция | Описание |
|---|---|
| `Properties` | Содержит коллекцию динамических свойств соединения |
| `Errors` | Содержит обработчик событий, произошедших с объектом Connection |

## Коллекция *Properties*

Коллекция `Properties` позволяет определить ряд свойств соединения, называемых *динамическими свойствами*. Самые важные из них описаны в табл. 5.5.

***Таблица 5.5.*** *Динамические свойства объекта* `Connection`

| Свойство | Описание |
|---|---|
| User ID | Учетная запись пользователя домена, обладающая соответствующими полномочиями |
| Password | Пароль в текстовом виде |
| Encrypt Password | Шифрование пароля при передаче |
| ADSI Flag | Параметры опознавания заданной учетной записи |

В табл. 5.5 приведена лишь малая часть поддерживаемых провайдером Active Directory динамических свойств. Работа с коллекцией `Properties` будет описана более подробно далее в *разд. "Объект коллекции Properties"*, а сейчас мы опишем перечисленные свойства более подробно.

- В качестве значения свойства **User ID** задается доменное имя учетной записи пользователя для подключения к каталогу Active Directory. Этот прием часто используется для выполнения сценария от имени администратора при условии, что сам сценарий запущен от имени стандартного пользователя. Такая ситуация возникает, например, при использовании сценариев для регистрации пользователей в сети (logon script).

Имя пользователя задается в формате DOMAIN\LOGIN, где DOMAIN — NetBIOS-имя домена, а LOGIN — значение поля `sAMAccountName`, например, ISLAND\PIvanov.

- Значение свойства **Password** — пароль учетной записи пользователя, хранящийся в теле сценария в незашифрованном виде. Хранение незашифрованных паролей является грубейшим нарушением мер информационной безопасности. Необходимо с помощью различных утилит создавать воспринимаемые встроенным интерпретатором VBScript файлы с расширением VBE (encrypted VBS), в которых данные хранятся в бинарном виде. Одна из таких утилит входит в редактор SAPIEN PrimalScript.
- Указанный в предыдущем свойстве пароль необходимо передать по сети в каталог Active Directory. Для его безопасной передачи настоятельно рекомендуется воспользоваться шифрованием паролей, указав в качестве значения свойства **Encrypt Password** — `TRUE` или 1.
- Флаговое значение **ADSI Flag** определяет аутентификацию пароля в Active Directory. Этот параметр использовался исключительно для доступа в Active Directory с помощью устаревшего протокола WinNT, описание которого выходит за рамки этой книги.

### Коллекция *Errors*

Любая инструкция, использующая объекты ADO, может сгенерировать одну или более ошибок провайдера. Когда происходит ошибка, в коллекцию `Errors` могут быть помещены один или более объектов `Error`. Если следующая подобная инструкция также сгенерирует ошибку, коллекция будет очищена и заполнена заново. Каждый объект `Error` представляет определенную ошибку провайдера, но не ошибку ADO.

Для доступа к конкретному элементу коллекции `Errors` используйте свойство `Item`, для получения количества элементов — свойство `Count`. Используйте метод `Clear()`, чтобы вручную очистить коллекцию `Errors`.

Некоторые свойства и методы возвращают предупреждения, которые появляются как объекты `Error` в коллекции `Errors`, при этом не останавливая выполнение программы.

Объект `Error` содержит информацию об ошибках доступа к данным, которые возникли при выполнении конкретной операции провайдера. Вы можете обратиться к свойствам объекта `Error`, чтобы получить информацию о каждой ошибке (табл. 5.6).

***Таблица 5.6.*** *Свойства объекта* `Error`

| Свойство | Описание |
|---|---|
| `Description` | Содержит строку, определяющую короткое описание ошибки. Это свойство используется по умолчанию |
| `Number` | Содержит уникальный код (целое число), определяющий тип ошибки |
| `Source` | Идентифицирует имя объекта, который вызвал ошибку (строка) |
| `SQLState` | Содержит строку из пяти символов, которая указывает код ошибки по стандарту SQL ANSI |
| `NativeError` | Содержит определенный провайдером код ошибки (целое число) |

Если вы создаете сценарии на VBScript, то необходимо использовать встроенный обработчик ошибок.

## Объект *Command*

Объект `Command` используется для создания и выполнения различных команд и запросов. Хотя запрос можно выполнить и без команды `Command` — с помощью метода `Execute()` объекта `Connection` или метода `Open()` объекта `Recordset`, объект `Command` незаменим при выполнении запроса с параметрами и очень удобен, когда необходимо сохранить запрос для его повторного использования.

Объект `Command` может быть использован как в паре с объектом `Connection`, так и без него, поскольку строку подключения можно задать непосредственно в свойстве `ActiveConnection`.

Подключение к объекту в VBScript осуществляется с помощью стандартного приема:

```
set ComObj = CreateObject("ADODB.Command")
```

Рассмотрим свойства и методы объекта `Command`.

## Методы объекта *Command*

В табл. 5.7 приведены часто используемые методы с их кратким описанием.

***Таблица 5.7.*** *Методы объекта* `Command`

| Метод | Описание |
|---|---|
| `Execute()` | Выполняет SQL-запрос, команду или выполняемую процедуру |
| `Cancel()` | Прерывает выполнение текущей команды |
| `CreateParameter()` | Создает новый объект `Parameter` |

Поскольку в каталоге Active Directory невозможно создавать свои параметры запросов, то метод `CreateParameter` рассматриваться не будет.

### Метод *Execute()*

Выполняет запрос, команду SQL, хранимую процедуру или другую команду, доступную провайдеру. Метод аналогичен одноименному методу `Execute()` объекта `Connection`. Все параметры являются необязательными. Параметр `Parameters` представляет собой массив параметров запроса типа `Variant`, передаваемый оператору SQL.

Команда `Execute()` имеет следующий синтаксис:

```
Execute(CommandText, Parameters, Options)
```

При его вызове используются следующие параметры:

- `CommandText` — строка, содержащая команду SQL, имя таблицы, хранимой процедуры или другую команду провайдера. При создании соединения с Active Directory строкой является запрос `SELECT`;
- `Parameters` — массив параметров типа `Variant`, передаваемых оператору SQL;
- `Options` — целое число, определяющее тип команды. Принимает следующие значения:
  - 1 — текстовое определение команды или хранимой процедуры;
  - 2 — имя таблицы. В этом случае будет создан SQL-запрос, который вернет все строки указанной таблицы;

- 4 — хранимая процедура;
- 8 — тип команды неизвестен (по умолчанию);
- 16 — асинхронное выполнение команды;
- 128 — не возвращать строки.

При создании соединения с каталогом Active Directory обычно указывают только первый параметр — SQL-запрос к пространству имен домена. В листинге 5.9 приведен пример получения доступа к пространству имен корневого каталога Active Directory.

**Листинг 5.9. Получение доступа к объекту RootDSE**

```
'Определение LDAP-имени домена
set root = GetObject("LDAP://RootDSE")
domain = "LDAP://" & root.Get("defaultNamingContext")

'Подключение к объекту ADODB.Connection
Set obj = CreateObject("ADODB.Connection")
obj.CommandTimeout = 120
obj.Provider = "ADsDSOObject"
obj.Open "Active Directory Provider"

'Подключение к объекту ADODB.Command
Set objCommand = CreateObject("ADODB.Command")
Set objCommand.ActiveConnection = obj
set RObj=obj.Execute("SELECT * FROM '" & domain & "'")
…
```

### Метод *Cancel()*

Этот метод, как и в случае объекта `Connection`, отменяет выполнение последнего вызова `Execute()` или `Open()`, если действие еще не завершено.

## Свойства объекта *Command*

В табл. 5.8 приведены свойства объекта `Command` и их краткое описание.

***Таблица 5.8.*** *Методы объекта* `Command`

| Свойство | Описание |
|---|---|
| `ActiveConnection` | Определяет объект `Connection`, с которым связан данный объект `Command` |
| `CommandText` | Строка, в которой определяется текст команды |
| `CommandTimeout` | Устанавливает или возвращает количество секунд ожидания выполнения команды |
| `CommandType` | Определение типа выполняемой команды: процедура, команда, SQL-запрос и др. |
| `Prepared` | Указывает, сохранять ли откомпилированную версию запроса |
| `State` | Возвращает состояние объекта `ADO.Command` |

## Свойство *ActiveConnection*

С помощью свойства `ActiveConnection` осуществляется связывание объектов `ADO.Command` и `ADO.Connection` (листинг 5.10). Значением этого свойства может быть строка, идентичная значению свойства `ConnectionString` объекта `Connection`.

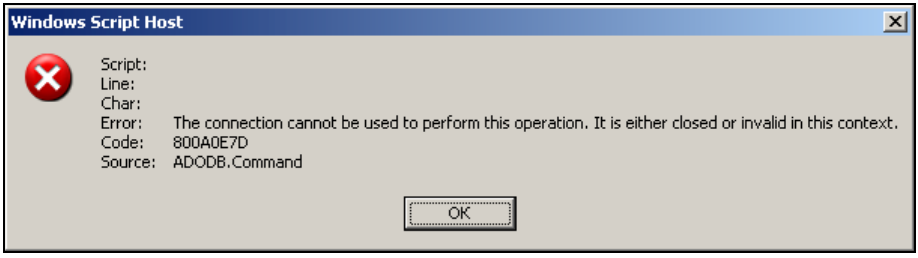



**Рис. 5.1.** Некорректный вызов метода `ActiveConnection`

**Листинг 5.10. Использование свойства `ActiveConnection`**

```
Set obj = CreateObject("ADODB.Connection")
obj.Provider = "ADsDSOObject"
```

```
obj.Open "Active Directory Provider"
Set ComObj = CreateObject("ADODB.Command")
Set ComObj.ActiveConnection = obj
```

Перед выполнением метода `Execute` объекта `Command` обязательно должно быть определено свойство `ActiveConnection`, иначе будет выведено сообщение об ошибке (рис. 5.1).

### Свойство *CommandText*

Строка, определяющая текст команды, оператор SQL, имя таблицы или вызов хранимой процедуры. В зависимости от значения свойства `CommandType` ADO может автоматически изменить свойство `CommandText`.

Для соединения с Active Directory это свойство обычно не используется.

### Свойство *CommandTimeout*

Устанавливает или возвращает время ожидания (в секундах) выполнения команды. По умолчанию `CommandTimeout` равен 30. Обычно значение этого свойства увеличивают при плотном сетевом трафике или большой нагрузке на серверах.

Если `CommandTimeout` равен нулю, то ADO будет пытаться соединиться в течение неограниченного количество времени.

Установка свойства `CommandTimeout` объекта `Connection` никак не связана с установкой свойства `CommandTimeout` объекта `Command`.

### Свойство *CommandType*

Значением свойства `CommandType` определяют тип выполняемой команды. Для ADODB-соединения, созданного для чтения данных из Active Directory, значение этого параметра всегда равно двум (2), поскольку с базой данных можно связаться, только выполнив SQL-запрос. По умолчанию тип команды не известен и равен восьми (8). Вот возможные значения свойства `CommandType`:

- −1 — тип команды не определен;
- 1 — текстовое определение команды или хранимой процедуры;
- 2 — создан SQL-запрос, который вернет заданные параметры;

- 4 — хранимая процедура;
- 8 — тип команды не известен. Назначается по умолчанию.

Если используется значение по умолчанию, то есть `CommandType = 8`, то команды выполняются несколько медленнее, поскольку необходимо потратить время на запрос ADO к провайдеру, чтобы определить тип выполняемой команды.

Если свойство `CommandType` не соответствует типу команды в свойстве `CommandText`, при вызове метода `Execute()` происходит ошибка.

### Свойство *Prepared*

Значение булева свойства `Prepared` (`True` или `False`) указывает, сохранить ли откомпилированную версию команды перед первым выполнением. Это может замедлить первое выполнение команды, но ускорит любое последующее выполнение. По умолчанию принимает значение `False`.

### Свойство *State*

Это свойство возвращает состояние объекта `ADO.Command`. Оно принимает одно из следующих значений:

- 0 — объект закрыт;
- 1 — объект открыт;
- 2 — объект соединяется;
- 4 — объект выполняет команду;
- 8 — объект выполняет выборку строк.

Свойство `State` может принимать комбинацию значений, например, пять (5 = 4 + 1). Это обозначает, что объект открыт и при этом осуществляется выполнение команды. Это признак асинхронного выполнения заданной инструкции.

## Коллекции объекта *Command*

Объект `Command` поддерживает две коллекции. Краткое описание коллекций приведено в табл. 5.9.

***Таблица 5.9.*** *Коллекции объекта* `Command`

| Коллекция | Описание |
|---|---|
| `Parameters` | Параметры команды |
| `Properties` | Динамические свойства соединения |

### Коллекция *Parameters*

Коллекция параметров содержит набор параметров объектов `Parameter`. У нее имеется несколько свойств и методов:

- `Count` — содержит количество объектов в коллекции;
- `Item(`*`index`*`)` — возвращает элемент коллекции по имени или порядковому номеру *`index`*;
- `Append(`*`object`*`)` — добавляет предварительно созданный объект *`object`* в коллекцию;
- `Delete(`*`index`*`)` — удаляет элемент из коллекции по имени или порядковому номеру *`index`*;
- `Refresh()` — обновляет коллекцию, чтобы отразить доступные объекты, определенные провайдером.

### Коллекция *Properties*

Коллекция `Properties` включает 63 свойства. Рассматривать их все не имеет смысла. Остановимся лишь на тех, которые часто используют на практике (табл. 5.10) при подключении к Active Directory.

***Таблица 5.10.*** *Часто используемые свойства коллекции* `Properties` *объекта* `Command`

| Свойство | Описание |
|---|---|
| Page size | Указывает количество записей, считываемое с каждого объекта. По умолчанию равно 10. Рекомендуется назначить большее значение, скажем, 100 |

***Таблица 5.10*** *(окончание)*

| Свойство | Описание |
|---|---|
| Timeout | Время в секундах, в течение которого считываются данные из каталога. По умолчанию равно 30 с. Как правило, этого времени мало, поэтому рекомендуется уставить значение 0 или любое другое значение, определенное эмпирическим путем. При установке значения 0 поиск будет осуществляться без ограничения времени |
| Cache Results | Управляет кэшированием результатов поиска. Первоначально скорость работы сценария снижается, поскольку на кэширование данных необходимо время. В целом, сценарий работает быстрее, поскольку считывает информацию в случае необходимости из кэша, а не из каталога. Это позволяет не только экономить время, но и снизить сетевой трафик<br><br>Принимает значение `TRUE` (по умолчанию) и `FALSE`. Рекомендуется отключать кэширование, поскольку при частых обновлениях Active Directory данные в кэше будут обновляться лишь по истечению определенного срока |
| SearchScope | Задает область поиска объектов:<br><br>• 0 (Base) — поиск осуществляется по корневому объекту, указанному в первой части запроса. Всегда возвращается либо один объект, либо пустой набор объектов. Эта область поиска используется чаще всего для проверки существования объекта, указанного в запросе;<br><br>• 1 (OneLevel) — поиск осуществляется только по непосредственным дочерним объектам контейнерного объекта, указанного в первой части запроса. Поиск по вложенным объектам более низких уровней не производится. При этом в результаты поиска не попадает и сам контейнерный объект;<br><br>• 2 (SubTree) — поиск осуществляется по всем вложенным объектам. При этом в результаты поиска не попадает и сам контейнерный объект |

## Объект коллекции *Properties*

Самый наглядный пример использования коллекции `Properties` — получение всех доступных свойств объекта — приведен в листинге 5.11. Несмотря на то, что задача достаточно специфическая, при создании различных сценариев такого рода информация иногда бывает просто необходима.

**Листинг 5.11. Просмотр всех свойств коллекции `Properties`**

```
Set obj= CreateObject("ADODB.Connection")
obj.Provider = "ADsDSOObject"
obj.Open "Active Directory Provider"
…
For Each elem In obj.Properties
    A = "Свойство: " & elem.Name
    B = "Значение: " & elem.Value
    C = "Тип данных: " & elem.Type
    ' D = "Атрибут: " & elem.Attributes
    WScript.Echo A & vbNewLine & B & vbNewLine & C & vbNewLine
Next
```

В результате работы сценария выводится сообщение (рис. 5.2), в котором первая строка — название свойства, вторая — его значение, третья — тип данных (см. табл. 3.6 в *главе 3*).




**Рис. 5.2.** Свойство `Provider` коллекции `Properties`

Переназначение параметров осуществляется двумя равноценными способами, в зависимости от удобства в том или ином случае (листинги 5.12а и 5.12б). Способы отличаются исключительно формой записи. В приведенных листингах *`PROPERTY`* — название свойства, например, одно из приведенных в табл. 5.10 свойств, а *`VALUE`* — присваиваемое значение.

**Листинг 5.12а. Присвоение значения параметрам коллекции `Properties`**

```
Set ComObj = CreateObject("ADODB.Command")
ComObj.Properties("PROPERTY") = VALUE
```

**Листинг 5.12б. Присвоение значения параметрам коллекции `Properties`**

```
Set ComObj = CreateObject("ADODB.Command")
ComObj.Properties.Item("PROPERTY").Name = VALUE
```

# Объект *Recordset*

Объект `Recordset` предназначен для чтения значений указанных параметров из базы данных. Объект `Recordset` создается следующим образом:

```
Set RObj = CreateObject("ADODB.Recordset")
```

Не все свойства и методы объекта `Recordset` поддерживаются всеми провайдерами. Поскольку в книге рассматривается Active Directory Provider, то мы опишем только свойства и методы, поддерживаемые этим провайдером.

## Свойства объекта *Recordset*

В табл. 5.11 приведены свойства объекта `Recordset`, используемые Active Directory Provider.

***Таблица 5.11.*** *Свойства объекта* `Recordset`

| Свойство | Описание |
|---|---|
| `ActiveConnection` | Объект `Connection`, с которым связан данный объект `Recordset` |
| `CommandText` | Строка, в которой определяется текст команды |
| `CursorType` | Тип курсора в открываемом объекте `Recordset` |
| `LockType` | Тип блокировки в открываемом объекте `Recordset` |
| `Source` | Источник данных объекта `Recordset` |
| `BOF` | Флаг, указывающий на нахождение курсора в начале списка записей |
| `EOF` | Флаг, указывающий на нахождение курсора в конце списка записей |
| `MaxRecords` | Возвращаемое провайдером количество записей |
| `RecordCount` | Количество выбранных с помощью запроса записей |

### Свойство *ActiveConnection*

С помощью свойства `ActiveConnection` осуществляется связывание объектов `ADO.Connection` и `ADO.Recordset` (рис. 5.3). Значением этого свойства может быть строка, идентичная значению свойства `ConnectionString` объекта `Connection`, или же сам объект типа `ADO.Connection`.

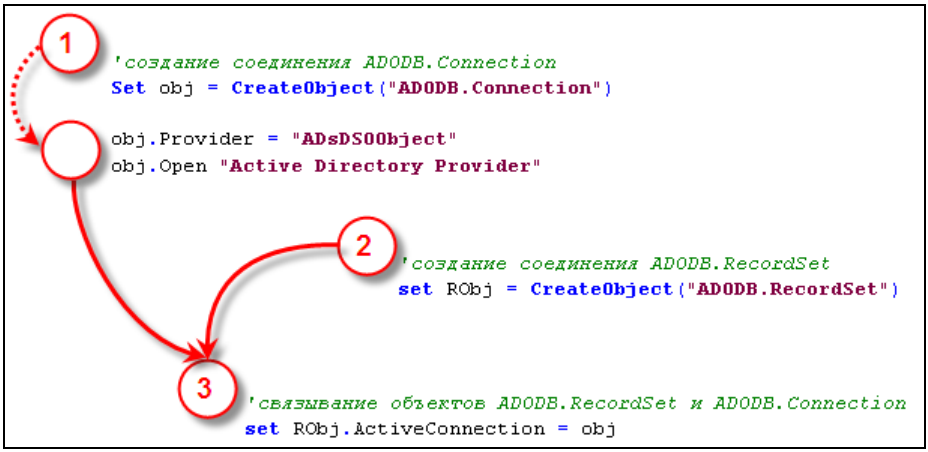



**Рис. 5.3.** Связывание объектов `ADO.Connection` и `ADO.Recordset`

### Свойство *CommandText*

Строка, определяющая текст команды, например, оператор SQL, имя таблицы или вызов хранимой процедуры. В зависимости от установки свойства `CommandType` ADO может автоматически изменить свойство `CommandText`.

Для соединения с Active Directory обычно это свойство не используется.

### Свойство *CursorType*

С помощью свойства `CursorType` устанавливается тип курсора для `Recordset`, который будет открыт. Тип курсора можно изменять. Возможные значения:

- −1 — тип курсора не определен (по умолчанию);
- 0 — определяет курсор типа forward-only, который аналогичен по действию статическому курсору (`CursorType = 3`). Разница заключается в том,

что записи могут прокручиваться только вперед. Это оптимизирует выполнение, поскольку позволяется лишь один проход по списку записей;

- 1 — определяет keyset-курсор, который аналогичен по действию динамическому курсору (`CursorType = 2`). Разница в том, что записи, добавляемые другими пользователями после получения результата, не появляются в списке записей. При этом удаляемые другими пользователями записи в `Recordset` становятся недоступными. Изменения атрибутов записей доступны в полном объеме;
- 2 — определяет динамический курсор. При просмотре набора записей в нем отражаются все добавления, изменения и удаления объектов, совершенные другими пользователями;
- 3 — определяет статический курсор. Возвращает статическую копию набора записей на момент выполнения запроса. Добавления, изменения или удаления, сделанные другими пользователями, не видны.

### Свойство *LockType*

Свойство `LockType` устанавливает тип блокировки, налагаемой создаваемым объектом `Recordset` на возвращаемые им записи. Принимает следующие значения:

- −1 — тип блокировки не определен (по умолчанию);
- 1 — только для чтения;
- 2 — пессимистическая блокировка. Провайдер гарантирует успешное редактирование записей. Запись блокируется сразу после начала редактирования и до момента сохранения записей;
- 3 — оптимистическая блокировка. Провайдер осуществляет блокировку записей только в момент сохранения изменений, то есть когда вызывается метод `Update()`;
- 4 — оптимистические пакетные модификации. Требуется для пакетного режима модификации (отложенное сохранение записей).

### Свойство *Source*

Значение свойства `Source` — ссылка на объект `Command`, строка с командой SQL, имя таблицы или хранимой процедуры. Последние два варианта не используются при создании прямого соединения с Active Directory. Рассмотрим

использование команды SQL, а если быть совсем точным — SQL-запроса, в качестве значения свойства `Source`.

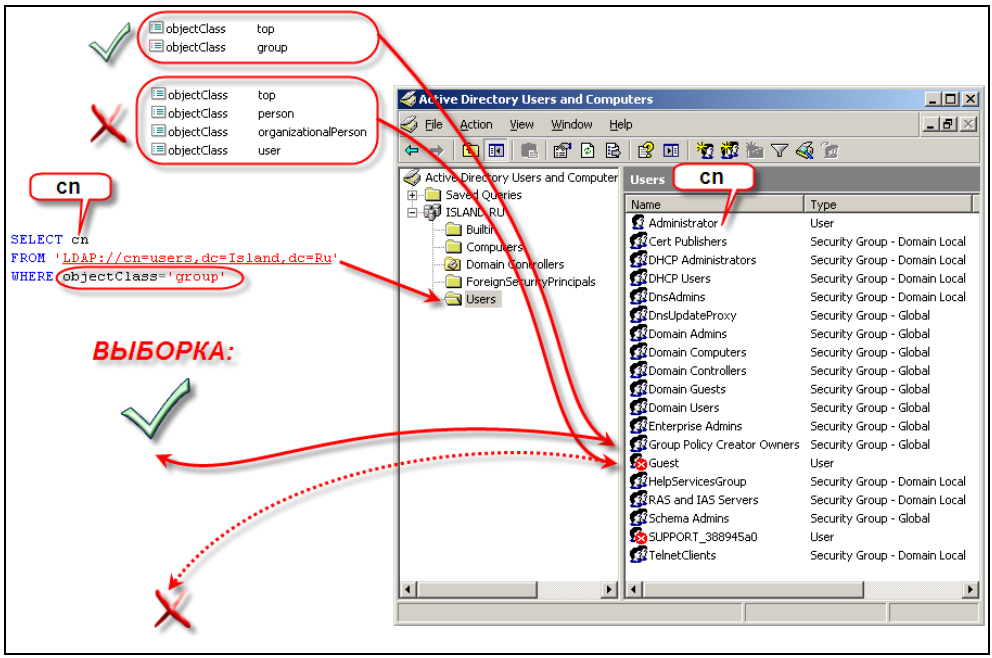



**Рис. 5.4.** Выполнение SQL-запроса

На рис. 5.4 показан процесс выполнения SQL-запроса, по которому осуществляется выборка объектов в папке Users, у которых `objectClass = 'group'`. В результате поиска выводится список значений атрибута `cn` (canonical name) для объектов, удовлетворяющих поставленному условию.

## Свойства *BOF* и *EOF*

С помощью свойств `BOF` и `EOF` проверяют, установлен ли курсор в начало и в конец списка соответственно. Чаще всего ими пользуются для организации цикла, просматривающего все элементы контейнера. Обычно эти свойства используются совместно с методами `moveFirst`, `moveNext`. Пример использования свойства `EOF` приведен далее в *разд. "Методы объекта Recordset"* (см. листинг 5.14).

### Свойство *MaxRecords*

С помощью этого свойства изменяют максимальное количество записей, возвращаемых провайдером. По умолчанию `MaxRecords` равно нулю, то есть провайдер возвращает все удовлетворяющие запросу записи (или указанное в запросе количество записей). Несмотря на это, при обращении к Active Directory провайдер LDAP возвращает только 1000 записей. Об этом ограничении всегда необходимо помнить.

### Свойство *RecordCount*

Свойство `RecordCount` возвращает количество выбранных с помощью SQL-запроса записей (листинг 5.13).

**Листинг 5.13. Определение количества выбранных запросом записей**

```
'создание соединения ADODB.Connection
Set obj = CreateObject("ADODB.Connection")
obj.Provider = "ADsDSOObject"
obj.Open "Active Directory Provider"
'создание соединения ADODB.Recordset
set RObj = CreateObject("ADODB.Recordset")
'связывание объектов ADODB.Recordset и ADODB.Connection
set RObj.ActiveConnection = obj
'формирование SQL-запроса
RObj.Source ="SELECT cn FROM 'LDAP://dc=Island,dc=Ru' " & _
        "WHERE objectClass='user' and NOT objectClass='computer'"
'выполнение запроса
RObj.Open
WScript.Echo RObj.RecordCount
```

## Методы объекта *Recordset*

В табл. 5.12 приведены методы объекта `Recordset`, используемые Active Directory Provider.

***Таблица 5.12.*** *Методы объекта* `Recordset`

| Метод | Описание |
|---|---|
| `Open()` | Открыть соединение |
| `Close()` | Закрыть соединение `Recordset` |
| `ReQuery()` | Обновить данные в объекте `Recordset`, выполнив запрос заново |
| `Update()` | Обновить данные в базе данных |
| `Cancel()` | Отменить выполняемый запрос |
| `Move()` | Сместиться на указанное количество записей относительно указанной позиции |
| `MoveFirst()` | Установить курсор в начало таблицы |
| `MoveNext()` | Установить курсор на следующий элемент списка записей |
| `MovePrevious()` | Установить курсор на предыдущий элемент списка записей |
| `Save()` | Сохранить полученные данные в файл |

## Метод *Open()*

С помощью метода `Open()` запускают выполнение запроса:

```
Open(Source, ActiveConnection, CursorType, LockType, Options)
```

Метод принимает пять необязательных параметров:

- `Source` — источник данных: SQL-запрос или ссылка на объект `Command`;
- `ActiveConnection` — ссылка на объект `Connection` или строка параметров подключения — `ConnectionString`;
- `CursorType` — тип курсора. По умолчанию `Recordset` будет открыт с курсором forward-only read-only на стороне сервера. Возможные типы курсоров описаны в *разд. "Свойство CursorType"*;
- `LockType` — тип блокировки записей. Типы блокировок описаны в *разд. "Свойство LockType"*;
- `Options` — целое число, определяющее тип команды, а именно показывающее, как надо интерпретировать параметр `Source` и задающий неко-

торые характеристики возвращаемого набора записей (при необходимости значения суммируются):

- 1 — текстовое определение команды или хранимой процедуры;
- 2 — имя таблицы, для возвращения всех записей которой автоматически создается SQL-запрос;
- 4 — хранимая процедура;
- 8 — тип команды не известен (по умолчанию);
- 16 — режим асинхронного выполнения команды. Нельзя сочетать с типом команды, возвращающей 512;
- 32 — режим, в котором строки, не находящиеся в кэше, будут выбраны асинхронно. Размер кэша определяется свойством `CacheSize`;
- 64 — режим, в котором никогда не осуществляется блокировка при поиске;
- 256 — имя файла, в котором хранится ранее сохраненный набор записей;
- 512 — имя таблицы, все строки которой возвращаются в наборе записей. При использовании этой опции работа с таблицами в ряде случаев осуществляется быстрее; при работе с Active Directory не используется.

### Метод *Close()*

Этот метод закрывает соединение, используемое объектом `Recordset`.

### Метод *ReQuery()*

Этот метод обновляет данные в объекте `RecordSet` путем повторного выполнения запроса. Он эквивалентен последовательному вызову методов `Close()` и `Open()`. Метод `ReQuery()` использует те же параметры, что и метод `Open()`, описанный ранее в *разд. "Метод Open()"*.

### Метод *Update()*

С помощью метода `Update()` осуществляется обновление данных в каталоге Active Directory данными, находящимися в кэше.

### Метод *Cancel()*

Отменяет последний вызов `Open()`, если процесс еще не завершен.

### Метод *Move()*

Перемещает курсор на указанное количество записей (рис. 5.5):

```
Move(NumberOfRecords, StartPosition)
```

При его вызове необходимо указать два параметра:

- `NumberOfRecords` — целое число, определяющее количество записей, на которое осуществляется перемещение курсора;
- `StartPosition` — позиция, с которой необходимо выполнить перемещение.

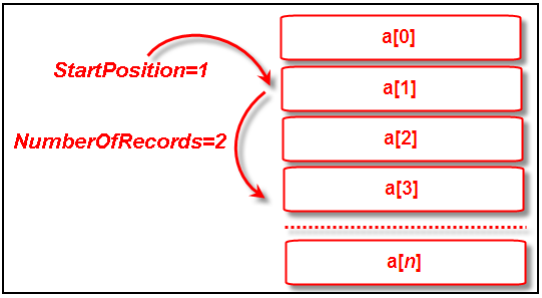



**Рис. 5.5.** Использование метода `Move()`

### Методы *MoveFirst()*, *MoveNext()*, *MovePrevious()*

Все три метода осуществляют перемещение курсора (рис. 5.6), соответственно, в начало списка, на одну позицию вперед по списку, на одну позицию назад по списку. Как правило, для перебора всех выбранных записей (листинг 5.14) используется пара методов `MoveFirst()` и `MoveNext()`. Метод `MovePrevious()` используется крайне редко.

**Листинг 5.14. Использование методов `MoveFirst()` и `MoveNext()`**

```
...
RObj.Open
RObj.MoveFirest
```

```
While Not Robj.EOF
    …
    RObj.MoveNext
Wend
```

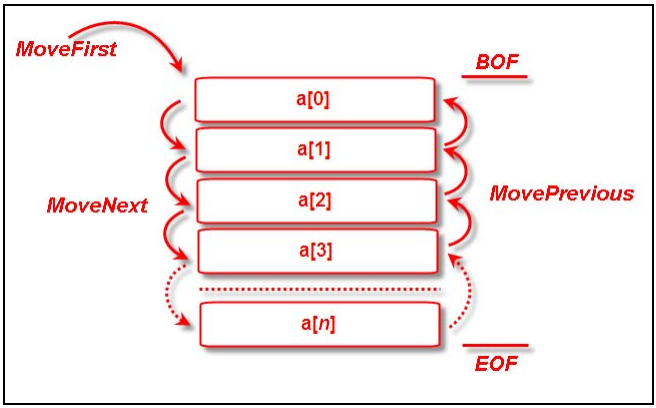



**Рис. 5.6.** Использование методов `MoveFirst()`, `MoveNext()`, `MovePrevious()`

## Метод *Save()*

С помощью метода `Save()` реализовано сохранение набора записей в файл (листинг 5.15). Метод может быть вызван только для открытого набора записей. После вызова метода текущей становится первая строка набора записей. Метод `Save()` имеет следующий синтаксис:

```
Save(FileName, Format)
```

При вызове можно указывать два параметра:

- `FileName` — полное имя файла-отчета;
- `Format` — формат, в котором будут сохранены данные. Принимает одно из следующих значений:
  - 0 — формат Microsoft Advanced Data TableGram (ADTG). Это значение параметра задано по умолчанию. Данные записываются в бинарном виде в файл с расширением DAT;
  - 1 — XML ADO в кодировке UTF-8. Пример файла приведен на рис. 5.7;

- 2 — собственный формат провайдера. Данная функция недоступна, поскольку провайдер Active Directory не поддерживает собственных форматов сохранения.

Оба параметра являются необязательными. Если параметр `FileName` не указан при первой записи файла, то в качестве имени файла используется значение свойства `Source` объекта `Recordset`. При последующих записях используется предыдущее имя файла, если не указано новое.

**Листинг 5.15. Сохранение результатов поиска в XML-файл**

```
'Создание соединения ADODB.Connection
Set obj = CreateObject("ADODB.Connection")
obj.Provider = "ADsDSOObject"
obj.Open "Active Directory Provider"

'Создание соединения ADODB.Recordset
set RObj = CreateObject("ADODB.Recordset")

'Связывание объектов ADODB.Recordset и ADODB.Connection
set RObj.ActiveConnection = obj

'Формирование SQL-запроса
RObj.Source ="SELECT cn FROM 'LDAP://cn=users,dc=Island,dc=Ru' " & _
        "WHERE objectClass='group'"
'Выполнение запроса
RObj.Open
RObj.Save "c:\ADO.xml", 1
```

Метод `Save()` может записывать только новые файлы, если при вызове параметр `FileName` не опущен. При попытке перезаписать файл, явно указав имя существующего файла, возникает ошибка. Для решения проблемы необходимо вставить обработчик ошибок (листинг 5.16): проверять, имеется ли уже такой файл, и удалять его перед использованием метода `Save()`.

**Листинг 5.16. Проверка на наличие файла**

```
...
RObj.Open
```

```
FileName = "c:\ADO.xml"
set fso = CreateObject("Scripting.FileSystemObject")
    If fso.FolderExists(FileName) = True Then
        fso.DeleteFile(FileName)
    End If
RObj.Save FileName, 1
```

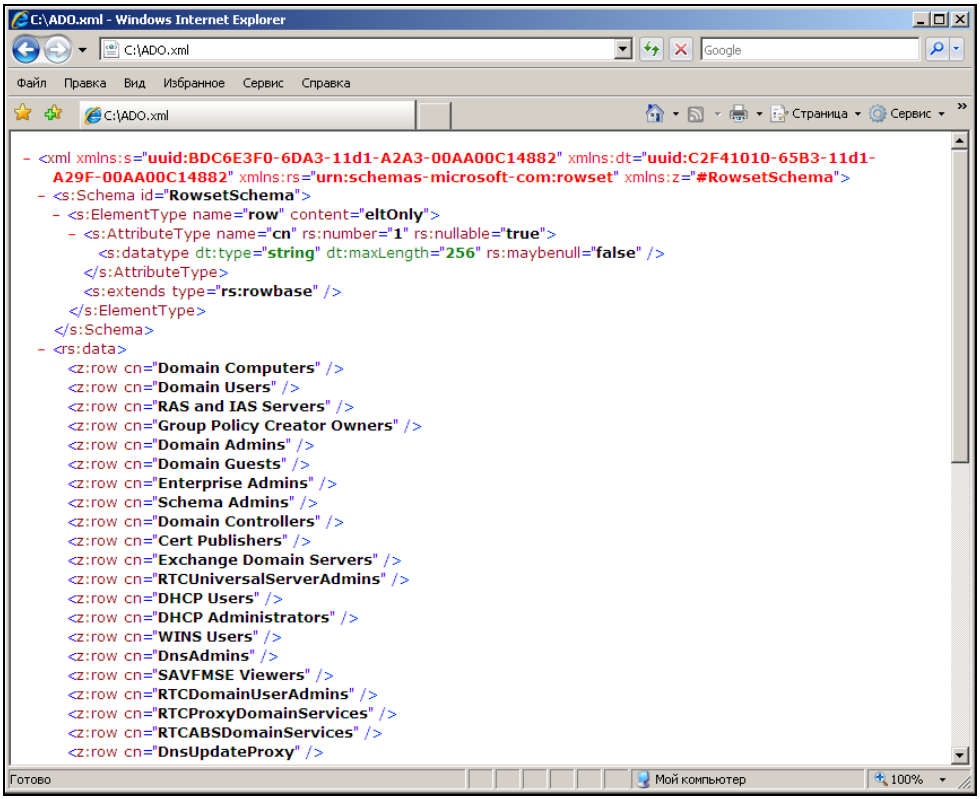



**Рис. 5.7.** Файл-отчет ADO.xml

## Коллекции объекта *Recordset*

Объект `Recordset` поддерживает две коллекции: `Fields` и `Properties`, предназначенные для управления значениями атрибутов. Каждая из коллекций `Fields` и `Properties` поддерживает по два свойства, описанные в табл. 5.13.

***Таблица 5.13.*** *Свойства коллекций* `Fields` *и* `Properties`

| Свойство | Описание |
|---|---|
| `Count` | Количество объектов в коллекции |
| `Item` | Значение указанного элемента |

## Свойство *Count*

Свойство `Count` возвращает количество объектов коллекции. Для коллекции `Fields` это количество определяется перечисленными после ключевого слова `SELECT` полями (рис. 5.8).

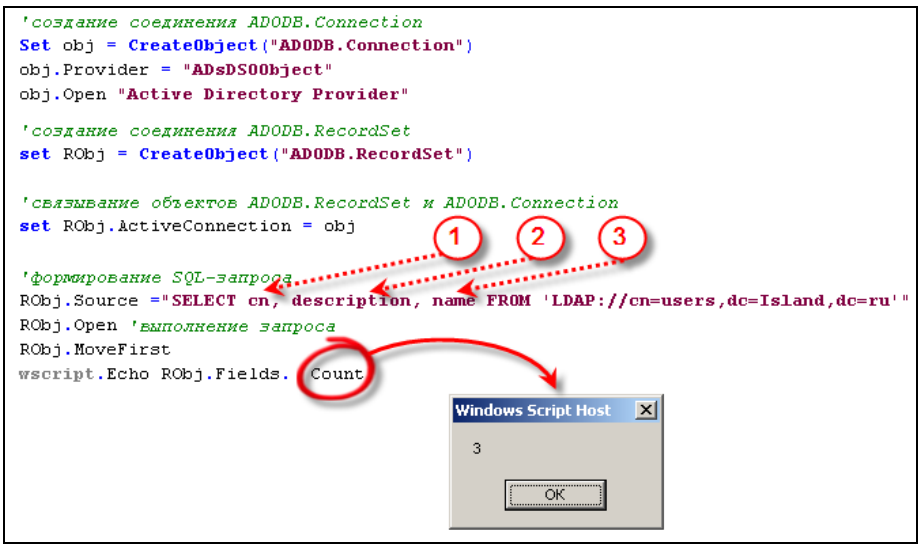



**Рис. 5.8.** Определение количества элементов коллекции

## Свойство *Item*

Свойство `Item` позволяет получить доступ к указанному элементу коллекции, указав его по имени или по номеру. Порядковый номер свойства определяется по SQL-запросу из выражения, стоящего сразу за ключевым словом `SELECT`. Необходимо помнить, что нумерация элементов начинается с нуля.

Например, после выполнения запроса, приведенного на рис. 5.8, обращение к полю `cn` можно записать двумя способами: `RObj.Fields.Item("cn")` или `RObj.Fields.Item(0)`.

В коллекции `Fields` после получения доступа к указанному полю набора записей с помощью свойства `Item` дальнейшие манипуляции с ним осуществляются как с объектом `Field`.

## Объект *Field*

С помощью свойств объекта `Field` можно определять различные характеристики поля, в частности, его значение. Свойства этого объекта описаны в табл. 5.14.

***Таблица 5.14.*** *Свойства объекта* `Field`

| Свойство | Описание |
|---|---|
| `Name` | Название поля, например, cn |
| `Value` | Значение поля, например, Ivanov, Petr |
| `OriginalValue` | Значение поля в каталоге Active Directory, то есть значение, полученное при чтении набора записей или в момент последнего вызова метода `Update()` |
| `DefinedSize` | Максимальный размер поля |
| `Type` | Тип поля (соответствующие константы приведены в табл. 3.6 в *главе 3*) |

Приведем пример чтения параметра `cn` в листинге 5.17. Обратите внимание на то, что если какое-либо свойство объекта `Field` возвращает число, то его необходимо преобразовать в строку. В противном случае при выводе на экран будет выведено сообщение об ошибке.

**Листинг 5.17. Определение характеристик заданного поля**

```
'создание соединения ADODB.Connection
Set obj = CreateObject("ADODB.Connection")
obj.Provider = "ADsDSOObject"
obj.Open "Active Directory Provider"
```

```
'создание объекта ADODB.Recordset
set RObj = CreateObject("ADODB.Recordset")

'связывание объектов ADODB.Recordset и ADODB.Connection
set RObj.ActiveConnection = obj

'формирование SQL-запроса
RObj.Source = "SELECT name " & _
        "FROM 'LDAP://cn=Ivanom\, Petr,ou=WorkSpace,dc=Island,dc=ru'"
RObj.Open
WScript.Echo "Значение: " & RObj.Fields(0).value
WScript.Echo "Тип данных: " & CStr(RObj.Fields(0).type)
WScript.Echo "Максимальная длина: " & CStr(RObj.Fields(0).DefinedSize)
```



# Раздел 3

# Объекты Active Directory

## Глава 6



# Виртуальный объект RootDSE

В этой главе подробно описывается виртуальный объект RootDSE, предназначенный для получения доступа к пространству имен домена, рассмотрена его объектная модель и приведены примеры его использования.

## Основные термины и понятия

- **RootDSE** — виртуальный объект, предназначенный для доступа к вершине пространства имен любого домена.
- **LDAP** (Lightweight Directory Access Protocol) — протокол, используемый для доступа к древовидной базе данных Directory Servers.
- **OID** (Object Identifier) — идентификатор безопасности объекта. В частности, это SID, GUID, UUID и др.
- **FQDN** (Fully Qualified Domain Name) — полное доменное имя или DNS-имя, составные части которого разделяются точками. Пример: island.ru.
- **UNC** (Universal Naming Convention) — полное имя ресурса в сети, соответствующее соглашению об универсальном назначении имен и включающее в себя имя сервера и совместно используемого ресурса; для каталогов или файлов могут также включать полный путь к этому ресурсу.
- **RDN** (Relative Distinguished Name) — относительное составное имя согласно FRC 2247. Пример: `DC=island,DC=ru`.

## Назначение объекта RootDSE

RootDSE является вершиной логического пространства имен и, к тому же, вершиной дерева, по которому осуществляется поиск с помощью LDAP. В этом виртуальном объекте публикуется информация о LDAP-сервере, в том

числе сведения о поддерживаемой им версии LDAP, механизмах протокола SASL (Simple Authentication and Security Layer), элементах управления.

Первоначально объект RootDSE был определен в RFC 2251 как часть спецификации LDAP версии 3. Поэтому его имя отсутствует в пространстве имен Active Directory. Каждый контроллер домена самостоятельно поддерживает этот искусственный объект.

Для подключения к RootDSE используется бессерверное подключение. В таких случаях для определения имени текущего домена используется локатор контроллеров домена. Доступ к RootDSE осуществляется анонимно.

# Объектная модель RootDSE

Рассмотрим подробно атрибуты, поддерживаемые объектом RootDSE. Все атрибуты можно разделить на 3 группы, характеризующие:

- *структуру домена* — доступные пространства имен;
- *конфигурацию домена* (LDAP) — поддерживаемые версии протокола LDAP, тип домена, текущее время и т. д.;
- *конфигурацию контроллера домена* — FQDN-путь контроллера домена, имя альтернативного сервера в случае нарушения функционирования домена и т. п.

## Атрибуты, характеризующие структуру домена

- `namingContexts` — массив, содержащий пути к каталогам, обеспечивающим функционирование домена. Массив состоит из пяти элементов, которые описывают пути к домену, к конфигурации домена, к схеме и DNS-зонам домена и леса. Часть значений элементов дублируется следующими параметрами: `defaultNamingContext`, `schemaNamingContext`, `configurationNamingContext`, `rootDomainNamingContext`. Это схематически отображено на рис. 6.1.
- `defaultNamingContext` — значение, которое определяет пространство имен по умолчанию. Обычно значением этого атрибута является составное имя домена, для которого данный контроллер является полномочным, например, `DC=Island,DC=ru`. Это значение обычно является первым элементом массива `namingContexts`.

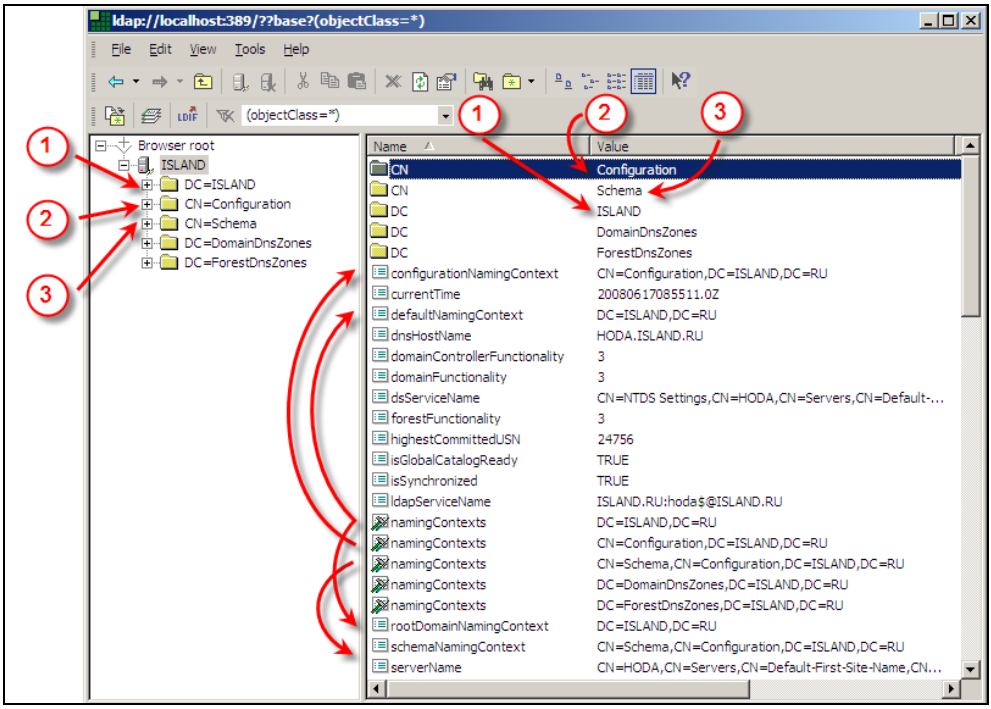



**Рис. 6.1.** Взаимосвязь составных имен домена

- `schemaNamingContext` — параметр, описывающий местоположение схемы домена. Является одним из членов массива `namingContexts`.
- `configurationNamingContext` — FQDN-путь к разделу конфигурации леса домена. Значение также является элементом массива `namingContexts`.
- `rootDomainNamingContext` — составное (FQDN) имя домена (раздела каталога), который был создан первым в данном лесе. Этот домен выполняет функции корня леса. Значение обычно является первым элементом массива `namingContexts`.

## Атрибуты, характеризующие конфигурацию домена

- `currentTime` — текущее время в формате Generalized Time.
- `subSchemaSubEntry` — составное имя местоположения настроек, необходимых для администрирования схемы Active Directory. Обычно это

CN=Aggregate,CN=Schema,CN=Configuration,DC=..., где DC=... — имя текущего домена.

- domainFunctionality — число, идентифицирующее режим работы домена:
  - 0 — Windows 2000 Domain Mode;
  - 1 — Windows Server 2003 Interim Domain Mode;
  - 2 — Windows Server 2003 Domain Mode;
  - 3 — Windows Server 2008 Domain Mode.
- forestFunctionality — число, которому соответствует версия леса:
  - 0 — Windows 2000 Forest Mode;
  - 1 — Windows Server 2003 Interim Forest Mode;
  - 2 — Windows Server 2003 Forest Mode;
  - 3 — Windows Server 2008 Forest Mode.
- supportedLDAPVersion — массив, элементы которого — версии протокола LDAP, поддерживаемые серверами. В настоящее время используется последняя вышедшая версия LDAP v3.1.
- supportedLDAPPolicies — идентификаторы объектов, указывающих используемые доменные групповые политики безопасности (GPO, Group Policy Object).
- supportedControl — идентификаторы объектов (известные как OID), которые указывают на элементы управления LDAP, поддерживаемые сервером. Этот атрибут отсутствует, если сервер не поддерживает никаких элементов управления. Подробнее об управляющих атрибутах рассказано в *разд. "Управляющие объекты"* в этой главе.
- supportedCapabilities — идентификаторы объектов (OID), совместимых с контроллером домена. Подробнее об управляющих атрибутах рассказано в *разд. "Управляющие объекты"* в этой главе.
- supportedExtension — идентификаторы объектов (OID), которые указывают на дополнительные операции, поддерживаемые сервером. Этот атрибут отсутствует, если сервер не поддерживает никаких расширений. По умолчанию серверы Active Directory (контроллеры доменов) не поддерживают каких-либо расширений.
- supportedSASLMechanisms — имена поддерживаемых сервером механизмов SASL (Simple Authentication and Security Layer), которые обеспечи-

вают выбор процедуры проверки подлинности при создании соединения и (при необходимости) механизма шифрования. Этот атрибут отсутствует, если сервер не поддерживает ни один из перечисленных механизмов.

## Атрибуты, характеризующие конфигурацию контроллера домена

- `dnsHostName` — DNS-имя контроллера домена.
- `serverName` — полное имя (FQDN) контроллера домена.
- `altServer` — URL-адреса серверов, к которым можно подключаться в случае недоступности текущего сервера. Этот атрибут отсутствует, если серверу не известны такие серверы. Для серверов Active Directory этот атрибут отсутствует по умолчанию.
- `ldapServiceName` — имя основного LDAP-сервера, используемого для взаимной проверки подлинности, в формате <DNS-имя главного домена леса>:<Главное имя протокола Kerberos>.
- `domainControllerFunctionality` — число, характеризующее операционную систему контроллера домена. Принимает следующие значения:
  - 0 — Windows 2000 Mode;
  - 2 — Windows Server 2003 Mode;
  - 3 — Windows Server 2008 Mode.
- `isSynchronized` — булево значение, информирующее о том, завершил (`True`) или нет (`False`) контроллер домена процесс начальной синхронизации с партнерами по репликации.
- `isGlobalCatalogReady` — булево значение, информирующее о том, готов ли контроллер домена оповещать о себе в качестве глобального каталога.
- `dsServiceName` — полное составное имя (DN) каталога, в котором хранятся параметры настройки службы директорий NTDS (NT Directory Service).

# Управляющие объекты

Использование управляющих объектов LDAP определено в RFC 2247 (LDAP v.3) как способ расширения возможностей без нарушения лежащего в ее основе

протокола. В настоящее время реализовано достаточно много таких объектов. Часть из них используются при поиске информации в каталоге, другие необходимы для выполнения различных операций. Управляющие объекты очень важны для нормальной жизнедеятельности домена. Нарушение их работы может оказаться фатальным для функционирования домена. Все управляющие объекты имеют статус "критические". При выполнении запроса обязательно осуществляется обращение к этим объектам. При невыполнении этого условия сценарий выдает ошибку и его работа прекращается.

Полный список управляющих объектов приведен в табл. 6.1.

***Таблица 6.1.*** *Управляющие объекты LDAP в Active Directory*

| Имя | OID | Описание |
|---|---|---|
| Paged Results | 1.2.840.113556.1.4.319 | Указывает серверу на необходимость возвращать результаты поиска постранично |
| Cross Domain Move | 1.2.840.113556.1.4.521 | Используется для перемещения объектов между доменами |
| DIRSYNC | 1.2.840.113556.1.4.841 | Предназначен для поиска объектов, измененных в течение заданного периода времени |
| Domain Scope | 1.2.840.113556.1.4.1339 | Запрещает серверу осуществлять перенаправление запроса при его обработке |
| Extended | 1.2.840.113556.1.4.529 | Возвращает составное имя объекта, а также его GUID и SID |
| Lazy Commit | 1.2.840.113556.1.4.619 | Указывает серверу на необходимость сообщать об успешном выполнении запроса после записи производимых в каталоге изменений в память, но до их записи на диск. Это значительно ускоряет обработку запросов на внесение большого количества изменений |
| Change Notification | 1.2.840.113556.1.4.528 | Используется клиентами во время регистрации для получения уведомлений об изменении каталога |

***Таблица 6.1*** *(продолжение)*

| Имя | OID | Описание |
|---|---|---|
| Permissive Modify | 1.2.840.113556.1.4.1413 | Позволяет повторно добавить определенное значение атрибута или удалить атрибут, не имеющий значения. По умолчанию попытка выполнить указанные операции завершается выводом сообщения об ошибке |
| SD Flags | 1.2.840.113556.1.4.801 | Используется для передачи серверу флагов управления параметрами защиты |
| Search Options | 1.2.840.113556.1.4.1340 | Предназначен для передачи серверу флагов управления определенными параметрами поиска |
| Show Deleted Objects | 1.2.840.113556.1.4.417 | Указывает серверу на необходимость вернуть данные удаленных объектов, соответствующих определенному критерию |
| Server-side Sort Request | 1.2.840.113556.1.4.473 | Предписывает серверу отсортировать результаты поиска |
| Server-side Sort Response | 1.2.840.113556.1.4.474 | Сервер возвращает объект в ответ на запрос сортировки |
| Tree Delete | 1.2.840.113556.1.4.805 | Предназначен для удаления части дерева каталога, включая все дочерние объекты |
| Verify Name | 1.2.840.113556.1.4.1338 | Позволяет указать конкретный сервер глобального каталога для проверки атрибутов, используемых при выполнении операций добавления и модификации объектов |
| VLV Request | 2.16.840.1.113730.3.4.9 | Позволяет запросить представление результатов поиска в виде виртуального списка. Появился в Windows Server 2003 |

***Таблица 6.1*** *(окончание)*

| Имя | OID | Описание |
|---|---|---|
| VLV Response | 2.16.840.1.113730.3.4.10 | Содержит ответ от сервера, у которого запрашивалось представление результатов поиска в виде виртуального списка. Появился в Windows Server 2003 |
| Attribute Scoped Query | 1.2.840.113556.1.4.1504 | Указывает, что запрос должен быть основан на атрибуте. Появился в Windows Server 2003 |
| Search Stats | 1.2.840.113556.1.4.970 | Позволяет получить статистическую информацию о запросе LDAP |
| Incremental Multivalue Retrieval | 1.2.840.113556.1.4.802 | Применяется для получения части значений составного атрибута (вместо всех его значений за один раз). Появился в Windows Server 2003 |

# Получение сведений об объекте RootDSE

Для получения информации о домене существует несколько разных способов. Данные о конфигурации объекта RootDSE можно получить с помощью одной из утилит (графической или запускаемой из командной строки) или с помощью сценария.

## Утилиты для просмотра характеристик RootDSE

В стандартный набор Windows Server входит стандартная утилита, запускаемая из командной строки — ldp.exe. Во времена Windows NT использовалась утилита Active Directory Viewer, входившая в комплект SDK для Active Directory Services Interface. Особенность и ценность этого продукта состоит в том, что это единственный из всех существующих программных продуктов, поддерживающих оба протокола доступа к Active Directory — LDAP и WinNT.

В настоящее время компания Microsoft выпустила бесплатную графическую утилиту Active Directory Explorer, которая намного удобнее своей предшественницы, однако поддерживает только протокол LDAP.

Существует также два очень удачных продукта компании Softerra для работы с Active Directory: LDAP Browser и LDAP Administrator. Принципиальная разница между этими двумя утилитами заключается в том, что бесплатный LDAP Browser предназначен только для чтения данных из Active Directory, как Active Directory Viewer или Active Directory Explorer, а платный LDAP Administrator позволяет вносить изменения не только в каталог Active Directory, но и в саму схему базы данных. Это очень мощное и удобное средство для работы с Active Directory.

## Active Directory Viewer (Microsoft)

Active Directory Viewer (AD Viewer) — графическая утилита, позволяющая выполнять операции чтения и модифицирования данных, а также осуществлять поиск в любых совместимых каталогах, таких как Active Directory, Exchange Server, Netscape Directory, Netware Directory.

Active Directory Viewer входит в состав SDK для Active Directory Services Interface, который можно бесплатно загрузить с сайта Microsoft: **http://www.microsoft.com/ntserver/nts/downloads/other/adsi25**.

По умолчанию AdsVw.exe устанавливается в папку %Program Files%\Microsoft\ ADSI Resource Kit, Samples and Utilities\ADsVw.

Программа работает в двух режимах: ObjectViewer и Query (рис. 6.2). Режим ObjectViewer используется для просмотра объектной модели выбранного провайдера, а режим Query — для поиска объектов.

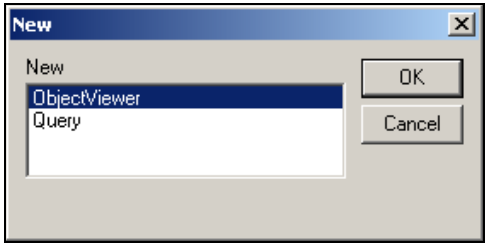



**Рис. 6.2.** Доступные режимы работы Active Directory Viewer

### Режим ObjectViewer

После выбора режима работы ObjectViewer появится диалоговое окно (рис. 6.3), в котором необходимо указать настройки подключения к базе данных. Существуют два провайдера, обеспечивающих подключение к Active Directory: LDAP или WinNT.

Для получения доступа к каталогу Active Directory необходимо указать RDN-путь к каталогу и параметры учетной записи, обладающей правами администратора (имя и пароль). Если приложение установлено на сервере, то параметры **Open As** и **Password** могут быть оставлены пустыми. Путь к каталогу должен быть построен в соответствии с шаблоном `<ProviderName:>//<Path>`

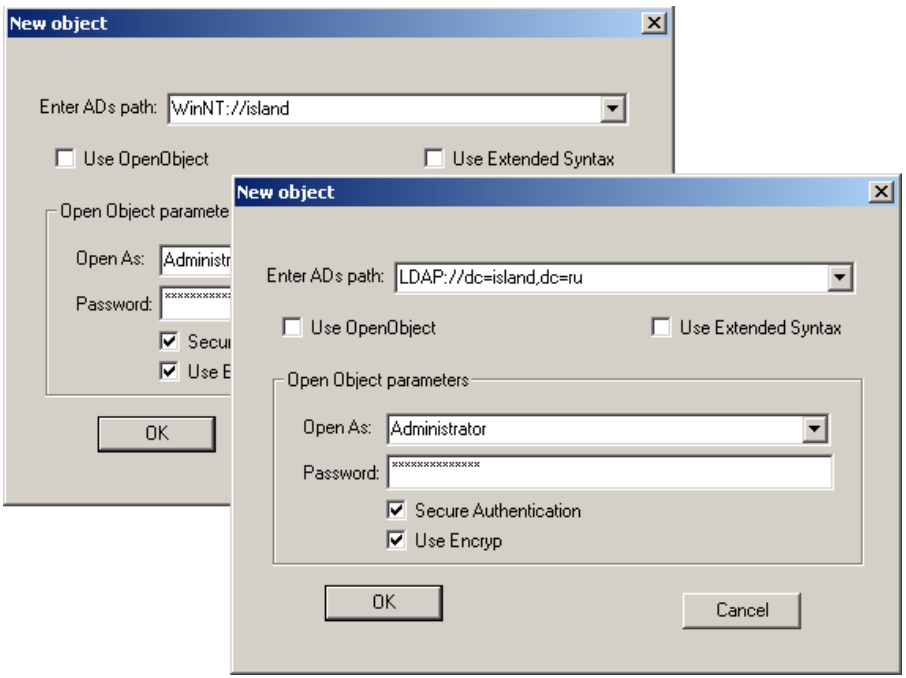



**Рис. 6.3.** Получение доступа к каталогу

Обратите внимание на две особенности получения доступа к AD:

- при вводе пути в поле **Enter ADs path** не должно быть пробелов;
- "слэши" должны быть прямыми — "/";
- имя провайдера чувствительно к регистру: WinNT, LDAP.

Невыполнение хотя бы одного из перечисленных условий приведет к ошибке — соединение с каталогом Active Directory не будет установлено. Если все параметры соединения указаны верно, то в появившемся диалоговом окне появится содержимое домена (рис. 6.4).

Обратите внимание на структуру данных Active Directory, приведенную в левой части экрана. При подключении с помощью протокола LDAP она иерархическая, а в случае WinNT — нет, и все учетные записи представлены единым списком. Это принципиальное различие между WinNT и LDAP.

В правой части — характеристики выбранного объекта, на котором установлен курсор. Списки свойств объекта и соответствующих им значений приведены в полях **Properties** и **Property Value**.

С помощью кнопок **Change**, **Clear**, **Append** и **Delete** можно воздействовать на объектную модель каталога — изменять, очищать, добавлять и удалять поля в свойствах объектов соответственно.

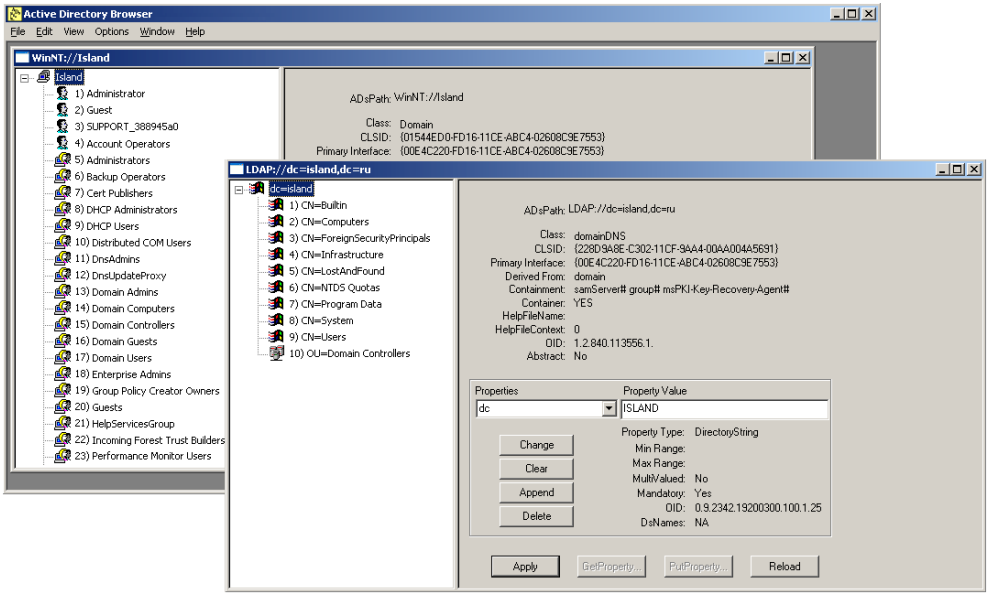



**Рис. 6.4.** Общий вид программы Active Directory Viewer

## Режим Query

Режим Query используется в случае необходимости поиска объектов в каталоге Active Directory.

## Active Directory Explorer

Компания Sysinternals (**http://www.sysinternals.com**) в 2007 году выпустила утилиту AD Explorer (рис. 6.5), с помощью которой можно не только смотреть, но и редактировать Active Directory. В отличие от Active Directory Viewer (Microsoft) и LDAP Browser (Softerra), программа не требует установки. В настоящее время компания Sysinternals принадлежит Microsoft (**http://technet.microsoft.com/en-us/sysinternals**).

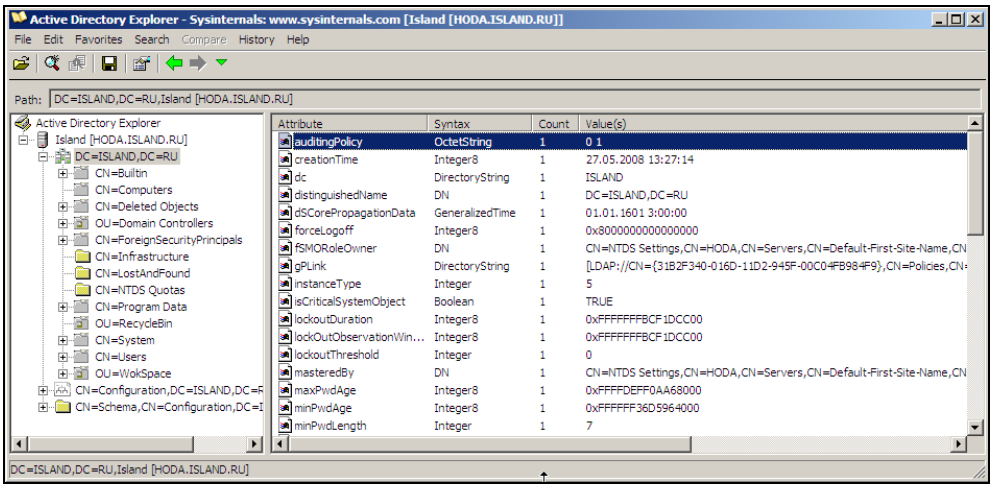



**Рис. 6.5.** Внешний вид Active Directory Explorer

Среди достоинств этой программы стоит отметить удобство подачи информации и возможность импорта/экспорта данных из Active Directory.

После первого запуска программы автоматически запускается мастер подключения к каталогу Active Directory (рис. 6.6). В отличие от всех остальных программ такого рода, для подключения достаточно указать имя домена. Будет ли это NetBIOS или DNS-имя — не играет роли.

При необходимости только чтения данных поля **User** и **Password** могут быть не заполнены. Для обеспечения возможности изменения данных в Active Directory в соответствующих полях должны быть указаны имя и пароль администратора сети.

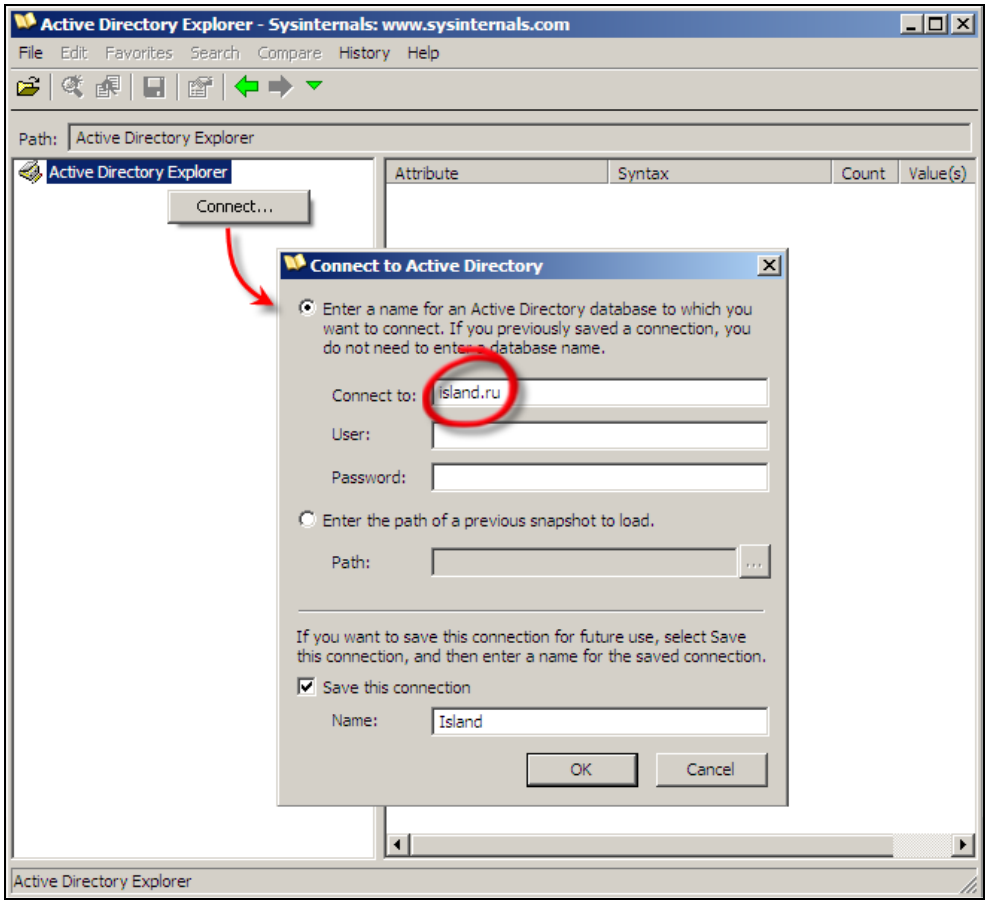



**Рис. 6.6.** Настройка соединения с Active Directory в AD Explorer

## LDAP Browser 2.6 (Softerra)

LDAP Browser 2.6 — программа, предназначенная исключительно для просмотра структуры каталога Active Directory и поиска данных по заданным атрибутам. Программу можно бесплатно загрузить с сайта **http://www.ldapbrowser.com** в разделе Download.

По своим возможностям программа превосходит Active Directory Viewer, да и в использовании LDAP Browser гораздо удобнее. В процессе создания соединения с каталогом могут быть заданы фильтры, параметры административной учетной записи, номер TCP-порта и др. Настройка программы нетри-

виальна, поэтому остановимся на данном вопросе подробнее. После запуска программы в ней уже присутствуют три тестовых домена, которые вряд ли пригодятся, поэтому их можно удалить. Для получения доступа к интересующему домену необходимо запустить соответствующий мастер, нажав <Ctrl>+<N> или выбрав в контекстном меню **Browse Root | New Profile** (рис. 6.7). Далее нужно пошагово отвечать на вопросы, задаваемые мастером настройки соединения.

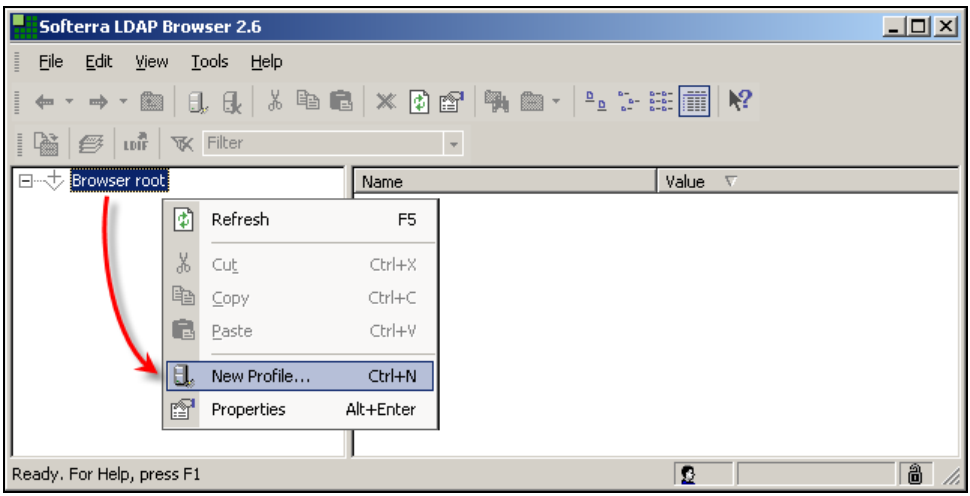



**Рис. 6.7.** Создание нового профиля

1. Ввести название нового профиля. Рекомендуется для удобства сделать его совпадающим с названием домена. По умолчанию предлагается имя New Profile (рис. 6.8).
2. Если программа устанавливается на контроллере домена, то значение параметра **Host** изменять не надо (то есть просто оставить значение localhost, предлагаемое по умолчанию). Если же программа установлена на локальном компьютере, то значение параметра **Host** нужно установить равным имени домена в соответствии с RFC 1123, например, domain.ru. Остальные параметры можно оставить без изменений (рис. 6.9).
3. После нажатия кнопки **Далее** мастер выведет сообщение о том, что параметр Base DN не задан (рис. 6.9). Необходимо продолжить работу мастера, нажав на кнопку **Yes** (рис. 6.10).

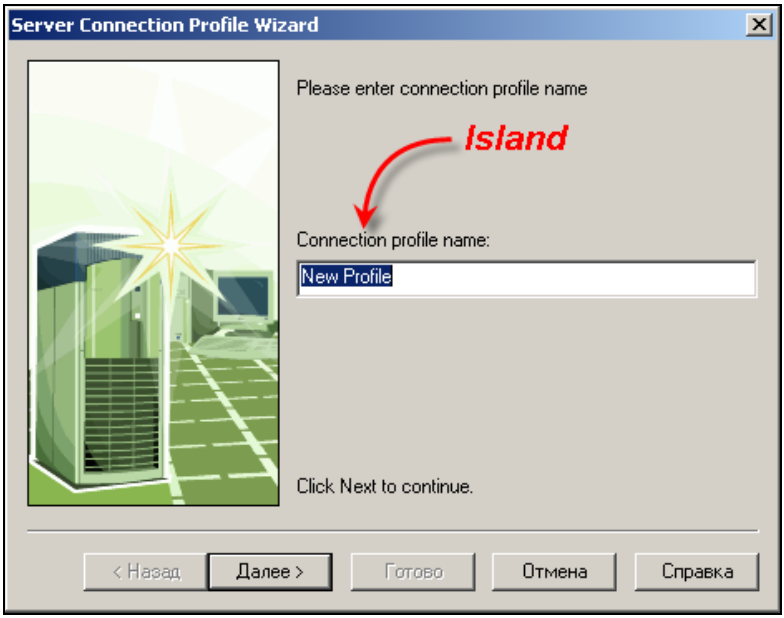



**Рис. 6.8.** Создание нового профиля, шаг 1

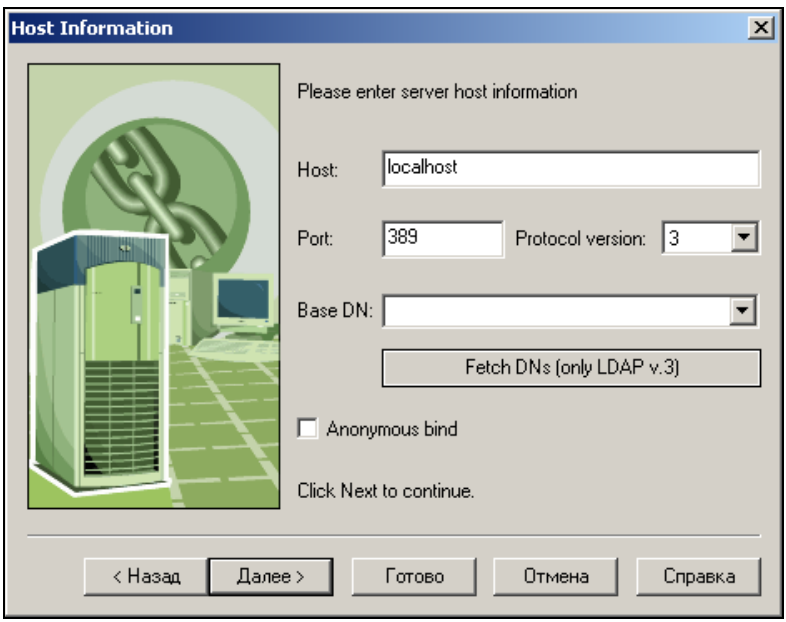



**Рис. 6.9.** Создание нового профиля, шаг 2

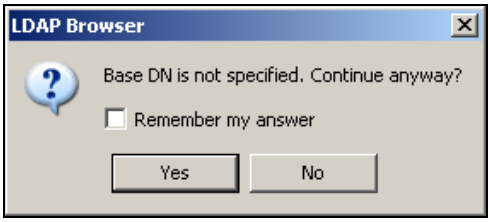



**Рис. 6.10.** Создание нового профиля, шаг 3

4. Мастер предлагает ввести имя и пароль учетной записи, от имени которой будет осуществляться доступ к Active Directory. В поле **User DN** введите учетное имя пользователя в сети (login без указания домена), в поле **Password** — пароль. Рекомендуется включить опцию **Save password** (рис. 6.11).

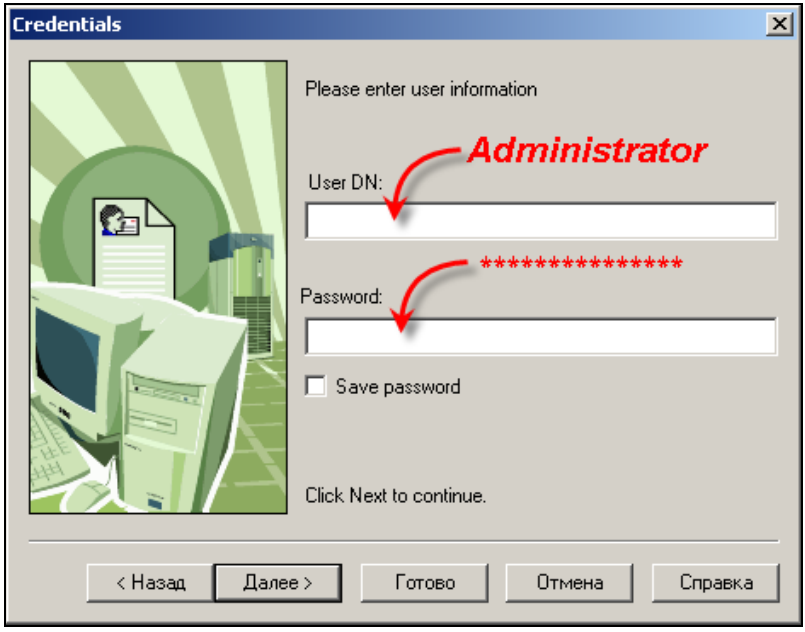



**Рис. 6.11.** Создание нового профиля, шаг 4

5. Если была установлена опция **Save password**, то на следующем шаге мастер запросит подтверждение пароля (рис. 6.12).

6. Последний шаг — настройка фильтра и временных характеристик. Не рекомендуется изменять значения настроек соединения, за исключением поля **Entry count limit** (рис. 6.13). Практика показывает, что количество объектов часто превышает 1000, поэтому рекомендуется установить значение, большее в несколько раз, например, 10 000.

После завершения работы мастера будет предпринята попытка подключения к домену. В случае успеха программа отобразит структуру каталога подключенного домена.

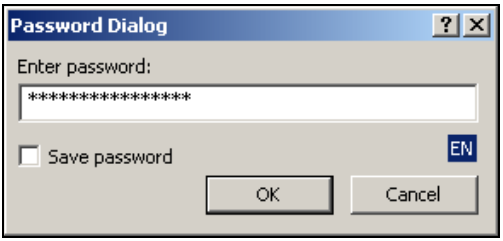



**Рис. 6.12.** Создание нового профиля, шаг 5

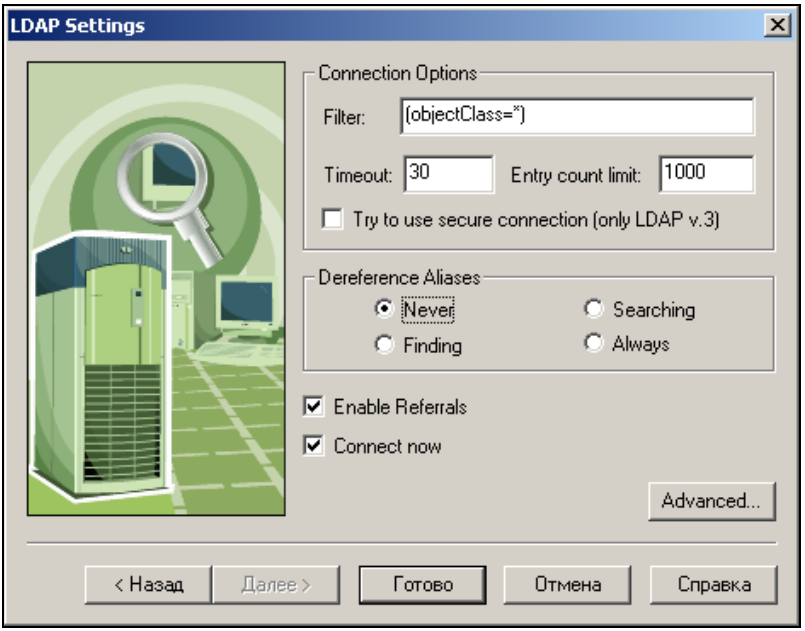



**Рис. 6.13.** Создание нового профиля, шаг 6

На рис. 6.14 приведен пример конфигурации домена Island.Ru с контроллером домена Hoda. Проанализируем информацию на рисунке: кроме перечисленных имен домена и контроллера домена можно заметить, что домен и лес построены на основе Windows 2003. Сам домен поддерживает две версии протокола LDAP — 2 и 3.

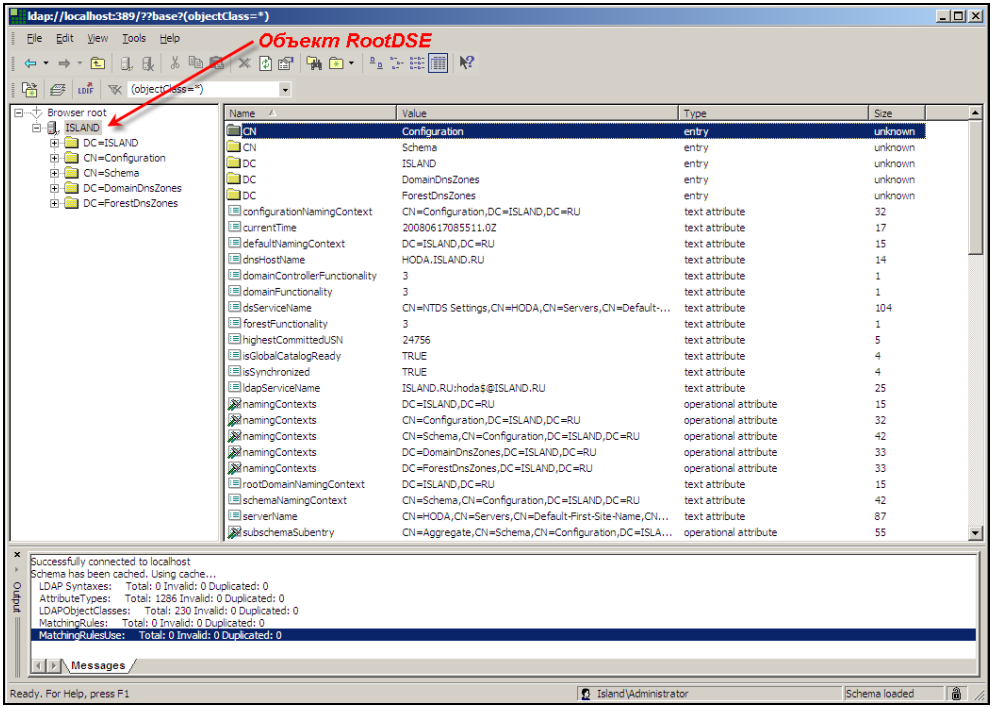



**Рис. 6.14.** Общий вид программы LDAP Browser

## Программное управление RootDSE

Список параметров и их значений программным способом можно получать с помощью интерфейсов `IADsPropertyList` и `IADsPropertyEntry`, подробно описанных в *разд. "Использование IADs-интерфейсов на практике"* в *главе 5*.

# Определение имени домена

На практике виртуальный каталог RootDSE используется для определения текущего имени домена, а также косвенным образом для определения принадлежности сервера к тому или иному сайту. Рассмотрим подробнее процедуру определения имени домена. Как известно, существуют три имени домена:

- FQDN — составное имя домена, например, `DC=ISLAND,DC=RU`;
- DSN — составное имя, в котором части разделены точкой, например, `ISLAND.RU`;
- NetBIOS-имя — первая часть составного имени домена, например, `ISLAND`.

С помощью RootDSE напрямую можно определить только FQDN-имя домена. Для этого необходимо получить значения параметров `defaultNamingContext`, `rootDomainNamingContext` или значение первого элемента `namingContexts`. На практике используют значение параметра `defaultNamingContext`. Приведем пример, в котором имя домена получается на языке VBScript (листинг 6.1).

**Листинг 6.1. Определение имени домена. VBSctipt**

```
Set objRootDSE = GetObject("LDAP://rootDSE")
FQDNDomainName = objRootDSE.Get("defaultNamingContext")
WScript.Echo FQDNDomainName
```

DNS-имя домена может быть определено на основе FQDN-имени домена (листинг 6.2), а также с помощью встроенного в Windows объекта `ADSystemInfo`.

**Листинг 6.2. Определение DNS-имени домена на основе FQDN-имени**

```
Set objRootDSE = GetObject("LDAP://rootDSE")
LDAPName = objRootDSE.Get("defaultNamingContext")
LDAPArray = Split(LDAPName, ",")
For Each el In LDAPArray
    DNSName = DNSName + Right(el, Len(el)-3) + "."
Next
WScript.Echo Left(DNSName, Len(DNSName)-1)
```

NetBIOS-имя домена определяется по такому же принципу, как и DNS-имя. Различие состоит в том, что после расчленения составного пути на части берется только первая часть (листинг 6.3).

**Листинг 6.3. Определение NetBIOS-имени домена на основе FQDN-имени**

```
Set objRootDSE = GetObject("LDAP://rootDSE")
LDAPName = objRootDSE.Get("defaultNamingContext")
LDAPArray = Split(LDAPName, ",")
el = LDAPArray(0)
WScript.Echo Right(el, Len(el)-3)
```

Глава 7



# Атрибуты основных объектов Active Directory

В этой главе описываются объектные модели основных объектов каталога Active Directory. К ним относятся учетные записи пользователей, групп, компьютеров и контейнеров. Атрибуты, характерные для всех учетных записей объектов, приведены отдельно, чтобы не путать читателя.

## Основные термины и понятия

- **GUID** (Globally Unique Identifier) — это созданная Microsoft реализация стандарта, описывающего глобальный идентификатор безопасности, уникальный во всем мире.
- **SID** (Security Identifier) — идентификатор безопасности. Как правило, используется для однозначной идентификации группы или учетной записи пользователя в домене.
- **Объектная модель** — совокупность свойств и методов, однозначно описывающая характеристики объекта и действия, которые можно с ним проделать.
- **Тип данных** — фундаментальное понятие теории программирования. Определяет множество значений; набор операций, которые можно применять к этим значениям; способ реализации хранения значений и выполнения операций. Любые данные, которыми оперируют программы, относятся к определенным типам.

# Условные обозначения

Для удобства восприятия информации и сокращения места во всех таблицах в колонках "По умолчанию" и "Тип данных" этой главы использованы сокращения. Их расшифровки приведены в табл. 7.1 и 7.2 соответственно.

***Таблица 7.1.*** *Расшифровка обозначений, используемых в столбце "По умолчанию"*

| Условное обозначение | Описание |
|---|---|
| + | Значение атрибуту присваивается при его создании с помощью сценария или мастера |
| – | Значение атрибуту присваивается после создания объекта |
| = | Значение атрибуту присваивается и изменяется автоматически |

***Таблица 7.2.*** *Расшифровка обозначений, используемых в столбце "Тип данных"*

| Условное обозначение | Тип данных | Описание | Пример |
|---|---|---|---|
| 3 | Long | Длинное число | 13 |
| 7 | Date | Дата-время | 20080410090637Z |
| 8 | String | Строка | Petr, Ivanov |
| 9 | Large Integer | Длинное число | 128555016510096179 |
| 11 | Bool | Булево значение | `True / False` |
| * | Array | Массив объектов: VarType() = 8192 + 12 | top; user; person |
| ** | Array | Массив бинарных элементов: VarType() = 8192 + 17 | F7 C5 CC CD 1B 4D A4 44 B4 24 72 43 4C 85 C0 98 |

# Обязательные атрибуты для всех объектов Active Directory

Все объекты, создаваемые в Active Directory с помощью мастера или программным способом, имеют ряд обязательных атрибутов (табл. 7.3). Часть из них может быть изменена, например, расположение объекта в иерархии каталога (поля `distinguishedName`, `name` и пр.), другая часть контролируется базой данных в автоматическом режиме (поля `createTimeStamp`, `uSNChanged` и пр.).

***Таблица 7.3.*** *Обязательные атрибуты всех объектов Active Directory*

| Поле в Active Directory | Тип данных | Описание |
|---|---|---|
| `codePage` | 3 | Код страны. Используется для переключения языка интерфейса. Появился в Windows 2003. В настоящее время не используется |
| `countryCode` | 3 | Код страны. Используется для переключения языка интерфейса. Появился в Windows 2003. В настоящее время не используется |
| `createTimeStamp` | 7 | Дата и время создания учетной записи сотрудника в формате UTC Time |
| `description` | 8 | Комментарий, описывающий объект |
| `distinguished-Name` | 8 | Составной путь учетной записи объекта в каталоге Active Directory |
| `instanceType` | 3 | Значением параметра описывается статус объекта на сервере. По умолчанию принимает значение 4. Возможные значения — 1, 2, 4, 8, 16, 32 |
| `modifyTimeStamp` | 7 | Дата и время последнего изменения учетной записи сотрудника в формате UTC Time |
| `name` | 8 | Отображаемое имя объекта |
| `objectCategory` | 8 | Составной путь к схеме с описанием объекта. Пример: `CN=`***`OBJECT_TYPE`***`,CN=Schema, CN=Configuration,DC=…` |

***Таблица 7.3*** *(окончание)*

| Поле в Active Directory | Тип данных | Описание |
|---|---|---|
| `objectClass` | * | Список классов, к которым относится объект (см. табл. 1.4 в *главе 1*). |
| `objectGUID` | ** | GUID-объекта (глобальный уникальный идентификатор объекта) |
| `subSchema-SubEntry` | 8 | Составной путь к подсхеме Active Directory |
| `uSNChanged` | 9 | Порядковый номер изменений свойств объекта, отсчитывающийся с момента его создания. Количество изменений, произошедших с объектом, равно разности `uSNChanged – uSNCreated` |
| `uSNCreated` | 9 | Порядковый номер изменений, сделанных в Active Directory на момент создания объекта |
| `whenChanged` | 7 | Дата и время изменения атрибутов объекта. Значение совпадает со значением параметра `modifyTimeStamp`. Используется для совместимости с доменами Windows NT |
| `whenCreated` | 7 | Дата и время создания объекта. Значение совпадает со значением параметра `createTimeStamp`. Используется для совместимости с доменами Windows NT |

# Основные объекты Active Directory

В этой главе мы рассмотрим атрибуты основных учетных записей, используемых в Active Directory:

- ❒ Пользователь;
- ❒ Группа
- ❒ Контейнер;
- ❒ Компьютер.

Более подробно работа с соответствующими объектами будет описана в *главах 8—11*.

## Пользователь

Учетная запись пользователя (рис. 7.1) — один из основных типов объектов, используемых в Active Directory. По сравнению с другими учетными записями она содержит больше всего полей. В табл. 7.4 приведена следующая информация:

- в колонке "По умолчанию" указано, присваивается ли значение этому атрибуту во время создания объекта. Возможные значения расшифрованы в табл. 7.1;
- в колонке "Тип данных" приведено число, возвращаемое функцией `VarType()`. Данная информация необходима для чтения того или иного атрибута в сценарии (табл. 7.2);

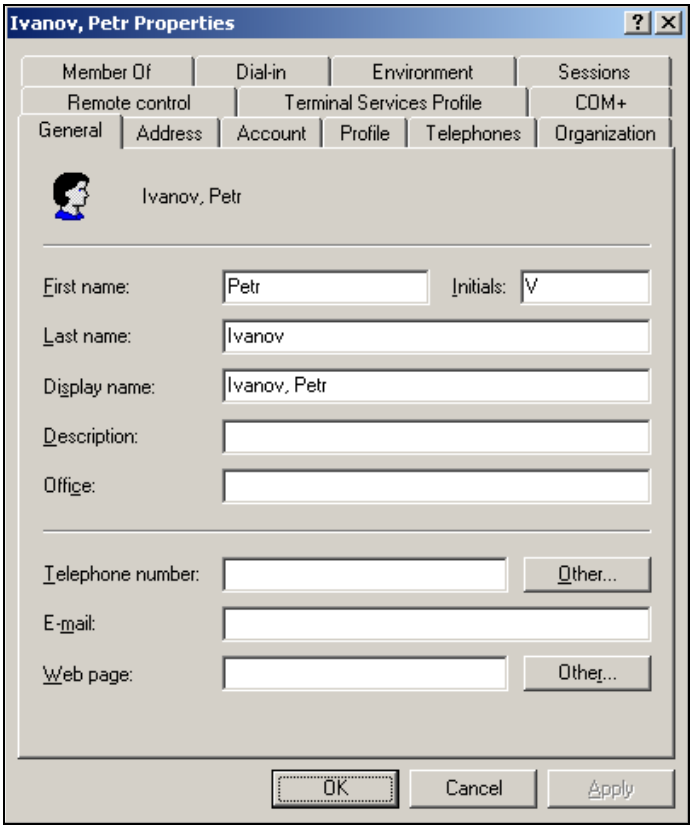



**Рис. 7.1.** Учетная запись пользователя (по умолчанию)

- в колонке "Вкладка" дано название вкладки, в которой с помощью MMC-консоли можно увидеть значение атрибута в явном виде (*разд. "Чтение свойств пользователя", глава 8*);
- в колонке "Описание" приведена краткая информация об атрибуте.

***Таблица 7.4.*** *Атрибуты учетной записи пользователя*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| accountExpires | + | 9 | **Account** | Время в секундах с 1 января 1961 года до момента выключения учетной записи. Если значение равно 0 — срок действия учетной записи не ограничен |
| allowLogon | – | 11 | **Terminal Services Profile** | Управление запретом для пользователя открывать терминальную сессию |
| badPasswordTime | – | 3 | – | Время в секундах с 1 января 1961 года до момента неправильного ввода пароля при входе в сеть |
| badPwdCount | – | 3 | – | Количество неудачных попыток ввода пароля при регистрации пользователя в сети, произведенных подряд. Если значение этого параметра превышает заданное групповой политикой, то ученая запись пользователя блокируется |
| c | – | 3 | **Address** | Международное сокращение названия страны (см. *приложение 5*) |
| cn | + | 8 | **General** | Отображаемое полное имя пользователя |
| co | – | 3 | **Address** | Название страны в текстовом виде. Выбирается администратором из списка |

***Таблица 7.4*** *(продолжение)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| company | – | 8 | **Organization** | Название компании-работодателя или дочерней фирмы |
| connectClient-DriversAtLogon | – | 11 | **Environment** | Управление подключением сетевых дисков при входе в терминальную сессию. Список подключаемых сетевых дисков соответствует списку сетевых дисков в папке Мой компьютер |
| connectClient-PrintersAtLogon | – | 11 | **Environment** | Управление подключением сетевыми принтерами при входе в терминальную сессию. Список подключаемых принтеров соответствует списку принтеров в папке Принтеры, находящейся в Панели управления |
| defaultToMain-Printers | – | 11 | **Environment** | При установке значения TRUE принтеры по умолчанию в терминальной сессии и обычном режиме совпадают |
| department | – | 8 | **Organization** | Подразделение, в котором работает сотрудник |
| description | – | 8 | **General** | Комментарий. Обычно ФИО сотрудника |
| directReports | – | * | **Organization** | Список учетных записей подчиненных сотрудников. Назначается в разделе **Organization \| Manager**. На экране выводится отображаемое имя, в каталоге Active Directory хранится полный составной путь (distinguishedName) к учетной записи пользователя |

***Таблица 7.4*** *(продолжение)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| displayName | + | 8 | **General** | Отображаемое имя |
| enableRemote-Control | – | 3 | **Remote control** | Управление удаленным подключением терминальной сессии. Значение параметра складывается из нескольких составляющих |
| facsimile-TelephoneNumber | – | 8 | **Telephones** | Номер факса сотрудника |
| givenName | + | 8 | **General** | Имя сотрудника |
| homeDirectory | – | 8 | **Profile** | UNC-путь к сетевой домашней папке пользователя |
| homeDrive | – | 8 | **Profile** | Буква, к которой монтируется сетевой диск |
| homePhone | – | 8 | **Telephones** | Домашний телефон сотрудника |
| info | – | 8 | **Telephones** | Служебная информация о сотруднике |
| initials | – | 8 | **General** | Инициал сотрудника |
| ipPhone | – | 8 | **Telephones** | Номер IP-телефона |
| l | – | 8 | **Address** | Название города |
| lastLogoff | – | 7 | – | Дата и время последнего выхода пользователя из сети (выключения компьютера или завершения сеанса работы) в формате UTC Time |
| lastLogon | – | 7 | – | Дата и время последней регистрации пользователя в сети (включения компьютера или начала сеанса работы) в формате UTC Time |

***Таблица 7.4*** *(продолжение)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| logonCount | – | 3 | – | Количество регистраций сотрудника в сети под этой учетной записью |
| logonHours | – | 11 | – | График доступа учетной записи пользователя в сеть по часам |
| mail | – | 8 | **General** | Адрес электронной почты в формате UPN |
| manager | – | 8 | **Organization** | Руководитель подразделения в виде составного пути к его учетной записи. Во вкладке выводится отображаемое имя, а в каталоге Active Directory хранится полный составной путь (distinguishedName) к учетной записи пользователя |
| maxConnection-Time | – | 8 | **Sessions** | Максимальное время работы сеанса терминальной сессии |
| max-Disconnection-Time | – | 8 | **Sessions** | Максимальное время, через которое закрытая терминальная сессия будет завершена |
| maxIdleTime | – | 8 | **Sessions** | Время завершения терминальной сессии с момента ухода компьютера, на котором она запущена, в спящий режим |
| memberOf | – | * | **MemberOf** | Список учетных записей групп, членами которых является пользователь. На экране выводится отображаемое имя, в каталоге Active Directory хранится полный составной путь (distinguishedName) к учетной записи группы |

***Таблица 7.4*** *(продолжение)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| mobile | – | 8 | **Telephones** | Номер мобильного телефона |
| msNPAllowDialin | – | 11 | **Dial-in** | Разрешение или запрет подключения пользователя к сети через соединение Dial-up. Принимает одно из трех значений: TRUE, FALSE и "пустое значение" |
| msNPCalling-StationID | – | 8 | **Dial-in** | Управление проверкой пользователя по идентификатору во время регистрации в сети через соединение Dial-up |
| msRADIUS-CallingNumber | – | 8 | **Dial-in** | Телефонный номер, по которому перезванивает модем провайдера для соединения с пользователем. Активно, если значение параметра msRADIUSServiceType равно 4 |
| msRADIUS-FramedIPAddress | – | 8 | **Dial-in** | IP-адрес, используемый для подключения через соединение Dial-up |
| msRADIUS-FramedRoute | – | 8 | **Dial-in** | Установка параметров маршрутизации для подключения к другой подсети. Значение параметра соответствует шаблону ooo.ooo.ooo.ooo / xxx.xxx.xxx.xxx, где ooo.ooo.ooo.ooo — маска подсети, к которой необходимо выполнить преобразование, а xxx.xxx.xxx.xxx — преобразуемая маска подсети |

***Таблица 7.4*** *(продолжение)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| msRADIUS-ServiceType | – | 8 | **Dial-in** | Настройка обратной связи (callback). При установленной настройке соединение инициализируется провайдером, а не пользователем |
| msRASaved-AllowDialin | – | 8 | – | Предыдущее значение параметра msNPAllowDialin |
| msRASaved-CallingNumber | – | 8 | – | Предыдущее значение параметра msRADIUSCallingNumber |
| msRASaved-FramedIPAddress | – | 8 | – | Предыдущее значение параметра msRADIUSFramedIPAddress |
| msRASaved-FramedRoute | – | 8 | – | Предыдущее значение параметра msRADIUSFramedRoute |
| MsTSHome-Directory | – | 8 | **Terminal Services Profile** | UCN-путь к каталогу, назначаемому в качестве домашнего при подключении к терминальной сессии (аналогичен homeDirectory) |
| msTSHomeDrive | – | 8 | **Terminal Services Profile** | Буква сетевого диска, на которую будет монтироваться домашний каталог при входе в терминальную сессию (аналогичен homeDrive) |
| msTSInitial-Program | – | 8 | **Environment** | Название файла, в котором находится сценарий, запускаемый при входе в терминальную сессию (аналогичен scriptPath) |
| msTSProfilePath | – | 8 | **Terminal Services Profile** | UCN-путь папки профиля учетной записи для входа в терминальную сессию (аналогичен profilePath) |

***Таблица 7.4*** *(продолжение)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| `MsTSWork-Directory` | – | 8 | **Environment** | UNC-путь к каталогу, назначаемому рабочим каталогом пользователя при входе в терминальную сессию |
| `objectSid` | + | ** | – | SID-объекта (уникальный идентификатор безопасности) |
| `otherFacsimile-TelephoneNumber` | – | * | **Telephones** | Дополнительные номера факсимильных аппаратов |
| `otherHomePhone` | – | * | **Telephones** | Дополнительные номера домашних телефонов |
| `otherIpPhone` | – | * | **Telephones** | Дополнительные номера IP-телефонов |
| `otherMobile` | – | * | **Telephones** | Дополнительные номера мобильных телефонов |
| `otherPager` | – | * | **Telephones** | Дополнительные номера пейджера |
| `otherTelephone` | – | * | **General** | Дополнительные номера телефонов |
| `pager` | – | 8 | **Telephones** | Номер пейджера |
| `physical-Delivery-OfficeName` | – | 8 | **General** | Номер офиса или комнаты (рабочего места, компьютера) |
| `postalCode` | – | 3 | **Address** | Код страны, города |
| `postOfficeBox` | – | 8 | **Address** | Почтовый индекс |
| `primaryGroupID` | = | 3 | **Members** | Идентификационный номер основной группы. По умолчанию назначена группа Domain Users (`primaryGroupID = 513`) |

***Таблица 7.4*** *(продолжение)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| profilePath | – | 8 | **Profile** | UNC-путь к сетевой папке с профилями |
| pwdLastSet | + | 11 | **Account** | Если параметр равен нулю, то при следующем входе в систему пользователь должен изменить свой пароль. В противном случае он хранит время последней смены пароля |
| sAMAccountName | + | 8 | **Account** | Имя пользователя для входа в сеть. Используется для совместимости с доменами Windows NT |
| sAMAccountType | + | 3 | – | Тип объекта |
| scriptPath | – | 8 | **Profile** | Полный путь к файлу, в котором находится сценарий регистрации пользователей в сети. Если указано только имя файла, то сценарий находится в папке NetLogon |
| sn | + | 8 | **General** | Фамилия сотрудника |
| st | – | 8 | **Address** | Штат, область |
| streetAddress | – | 8 | **Address** | Адрес сотрудника: название улицы, номер дома |
| subSchema-SubEntry | + | 8 | – | Составной путь к контейнеру схемы Active Directory, в котором находится описание атрибутов объекта "person" |
| telephoneNumber | – | 8 | **General** | Номер телефона |

***Таблица 7.4*** *(окончание)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| title | – | 8 | **Organization** | Должность сотрудника |
| url | – | * | **General** | Дополнительные Web-адреса страниц сотрудника |
| userAccount-Control | + | 3 | **Account** | Число, представляющее собой сумму флагов. Значением устанавливаются правила безопасности: выключение учетной записи пользователя, использование зашифрованного пароля, управление возможностью изменения пароля и др. |
| userPrincipal-Name | + | 8 | **Account** | Полное имя пользователя в формате UPN |
| userWorkstations | – | * | – | Список компьютеров, с которых пользователь может регистрироваться в сети |
| wWWHomePage | – | 8 | **General** | Web-адрес страницы сотрудника |

## Группа безопасности

Учетная запись группы (рис. 7.2) — один из основных компонентов, обеспечивающих безопасность функционирования домена. Принадлежность учетных записей пользователей к иерархической системе групп обеспечивает четкое разделение привилегий пользователей в сети.

В табл. 7.5 приведены обязательные атрибуты учетной записи группы безопасности, используемые для программного управления объектами этого типа.

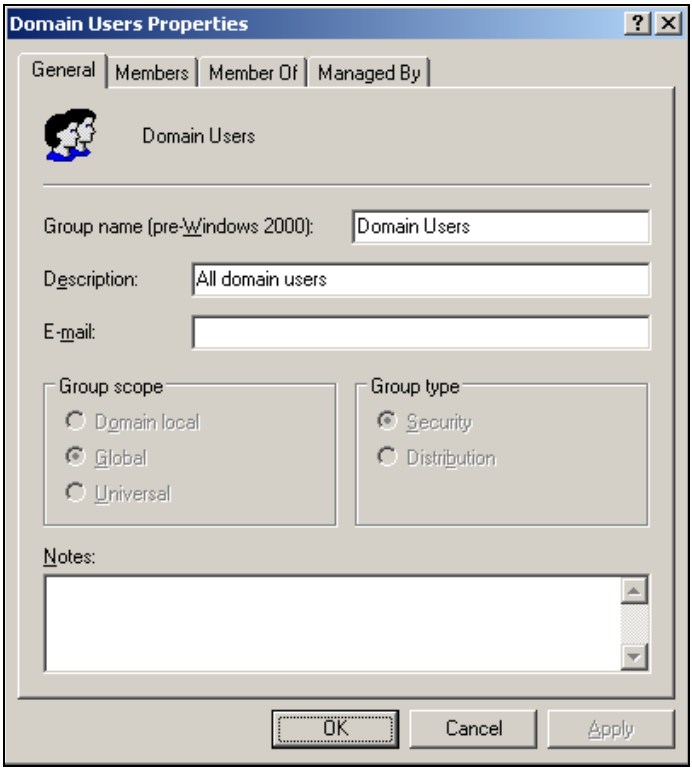



**Рис. 7.2.** Учетная запись группы (по умолчанию)

***Таблица 7.5.*** *Атрибуты учетной записи группы*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| `cn` | + | 8 | **General** | Полное имя объекта |
| `description` | – | 8 | **General** | Комментарий к учетной записи группы |
| `groupType` | + | 8 | **General** | Тип группы. Значение является суммой двух составляющих, которыми закодированы область действия и тип группы |

***Таблица 7.5*** *(окончание)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| notes | – | 8 | **General** | Служебная информация о группе |
| mail | – | 8 | **General** | Адрес электронной почты в UPN-формате (RFC 822). Появляется после создания электронного почтового ящика в MS Exchange |
| managedBy | – | 8 | **Managed By** | Ответственный за группу, назначаемый системным администратором. Во вкладке выводится отображаемое имя, в каталоге Active Directory хранится полный составной путь (distinguishedName) учетной записи пользователя |
| member | – | * | **Member** | Список учетных записей пользователей или групп — членов данной группы. На экране выводятся отображаемые имена объектов, в каталоге Active Directory хранится полный составной путь (distinguishedName) к объектам |
| memberOf | – | * | **Member Of** | Список учетных записей групп, в которые входит данная группа. На экране выводятся отображаемые имена групп, в каталоге Active Directory хранится полный составной путь (distinguishedName) к объектам |
| objectSid | + | ** | – | SID-объекта (уникальный идентификатор безопасности) |
| sAMAccountName | + | 8 | **General** | Имя группы, используемое для совместимости с доменами Windows NT |

## Контейнер

Контейнер (рис. 7.3) или, как его еще называют, папка, обеспечивает не только иерархическую структуру Active Directory, но и работу политик с объектами, находящимися в папке. Несмотря на небольшое количество атрибутов (табл. 7.6), контейнер является очень важным объектом.

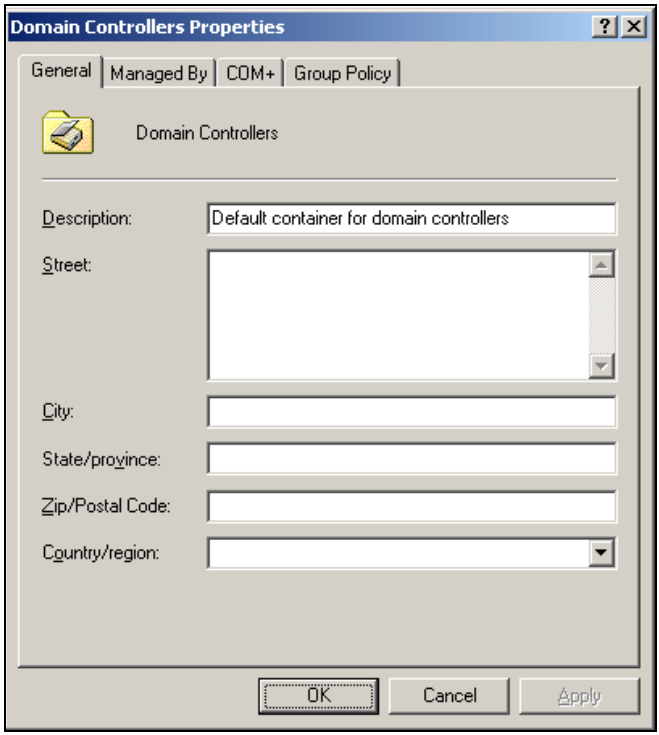



**Рис. 7.3.** Учетная запись контейнера (по умолчанию)

***Таблица 7.6.*** *Атрибуты учетной записи контейнера*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| cn | + | 8 | **General** | Полное имя объекта |

***Таблица 7.6*** *(окончание)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| `description` | – | 8 | **General** | Описание контейнера/папки |
| `showInAdvanced-ViewOnly` | – | 1<br>1 | – | При установке значения `TRUE` контейнер отображается только в расширенном режиме просмотра Active Directory |

## Компьютер

Компьютер (рис. 7.4) — единственная из основных учетных записей, которая почти всегда создается косвенным способом. Несмотря на кажущуюся простоту, это достаточно сложный объект для создания программным способом. Но изменить его свойства так же просто, как и у любого другого объекта. В табл. 7.7 приведены атрибуты учетной записи компьютера. Напомним, что в табл. 7.3 приводятся атрибуты, характерные для всех объектов Active Directory.

***Таблица 7.7.*** *Атрибуты учетной записи компьютера*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| `accountExpires` | + | 9 | – | Время в секундах с 1 января 1961 года до момента выключения учетной записи. Если значение равно 0 — срок учетной записи не ограничен |
| `badPasswordTime` | – | 3 | – | Время в секундах с 1 января 1961 года до момента неправильного ввода пароля при входе в сеть |

***Таблица 7.7*** *(продолжение)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| badPwdCount | – | 3 | – | Количество неудачных попыток входа пользователя при регистрации в сети, произведенных подряд. Если значение этого параметра превышает заданное групповой политикой, то учетная запись объекта блокируется |
| cn | + | 8 | **General** | Отображаемое полное имя компьютера |
| description | – | 8 | **General** | Описание компьютера. Например, его местоположение или роль: для верстки, редактирования, видеомонтажа и т. д. |
| displayName | + | 8 | **General** | Отображаемое имя в сети |
| dNSHostName | + | 8 | **General** | DNS-имя компьютера |
| lastLogoff | – | 7 | – | Дата и время выключения компьютера или завершения сеанса работы последнего пользователя в формате UTC Time |
| lastLogon | – | 7 | – | Дата и время включения компьютера в формате UTC Time |
| location | – | 8 | **Location** | Расположение учетной записи компьютера. Назначается администратором вручную |
| logonCount | – | 3 | – | Количество регистраций компьютера в сети под одной из доступных учетных записей пользователя |

***Таблица 7.7*** *(продолжение)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| managedBy | – | 8 | **Managed By** | Пользователь, который может управлять учетной записью компьютера. Назначается системным администратором. Во вкладке выводится отображаемое имя, в каталоге Active Directory хранится полный составной путь (distinguishedName) к учетной записи пользователя |
| msNPAllowDialin | – | 11 | **Dial-in** | Разрешение/запрет подключения к компьютеру в сети по Dial-up. Принимает одно из трех значений: TRUE, FALSE и "пустое значение" |
| msNPCalling-StationID | – | 8 | **Dial-in** | Управление проверкой пользователя по идентификатору |
| msNPSavedCalling-StationID | – | 8 | **Dial-in** | Предыдущее значение параметра msNCallingStationID |
| msRADIUSCalling-Number | – | 8 | **Dial-in** | Телефонный номер, по которому перезванивает модем провайдера для соединения с пользователем. Режим перезванивания активен, если значение параметра msRADIUSServiceType равно 4 |
| msRADIUSFramed-IPAddress | – | 8 | **Dial-in** | IP-адрес, используемый для подключения по Dial-up |
| msRADIUSFramed-Route | – | 8 | **Dial-in** | Установка параметров маршрутизации для подключения к другой подсети |

***Таблица 7.7*** *(продолжение)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| | | | | Значение параметра соответствует шаблону ooo.ooo.ooo.ooo / xxx.xxx.xxx.xxx, где ooo.ooo.ooo.ooo — маска подсети, к которой необходимо выполнить преобразование, xxx.xxx.xxx.xxx — преобразуемая маска подсети |
| msRADIUSService-Type | – | 8 | **Dial-in** | Настройка обратной связи (callback). При установленной настройке соединение инициализируется провайдером, а не пользователем |
| msRASavedCalling-Number | – | 8 | – | Предыдущее значение параметра msRADIUSCallingNumber |
| msRASavedFramed-IPAddress | – | 8 | – | Предыдущее значение параметра msRADIUSFramedIPAddress |
| msRASavedFramed-Route | – | 8 | – | Предыдущее значение параметра msRADIUSFramedRoute |
| objectSid | + | ** | – | SID-объекта (уникальный идентификатор безопасности) |
| operatingSystem | + | 8 | **Operating System** | Операционная система, установленная на компьютере. Пример: Windows XP Professional |
| operatingSystem-ServicePack | + | 8 | **Operating System** | Версия пакета исправлений (Service Pack), установленного на компьютере. Пример: Service Pack 3 |

***Таблица 7.7*** *(окончание)*

| Поле в Active Directory | По умолчанию | Тип данных | Вкладка | Описание |
|---|---|---|---|---|
| operatingSystem-Version | + | 8 | **Operating System** | Версия и номер сборки операционной системы. Пример: 5.1 (2600) |
| primaryGroupID | = | 3 | **Member Of** | Идентификационный номер основной группы. По умолчанию назначена группа Domain Users (primaryGroupID=513) |
| pwdLastSet | + | 11 | – | При значении TRUE при следующем входе в систему пользователь должен изменить свой пароль |
| sAMAccountName | + | 8 | **General** | Имя компьютера для входа в сеть. Используется для совместимости с доменами Windows NT |
| sAMAccountType | + | 3 | – | Тип объекта. Используется для совместимости с доменами Windows NT |
| servicePrincipal-Name | + | * | – | Имена компьютера, по которым можно к нему обратиться по сети |
| userAccountControl | + | 3 | – | Число, представляющее собой сумму флагов, которые задают правила безопасности |
| userParemeters | + | 8 | **Delegation**[1] | Настройка параметров доверительных отношений системы безопасности |

[1] Вкладка **Delegation** по умолчанию отсутствует. Чтобы она появилась, необходимо связать учетную запись со службой Service Principal Name (SPN).

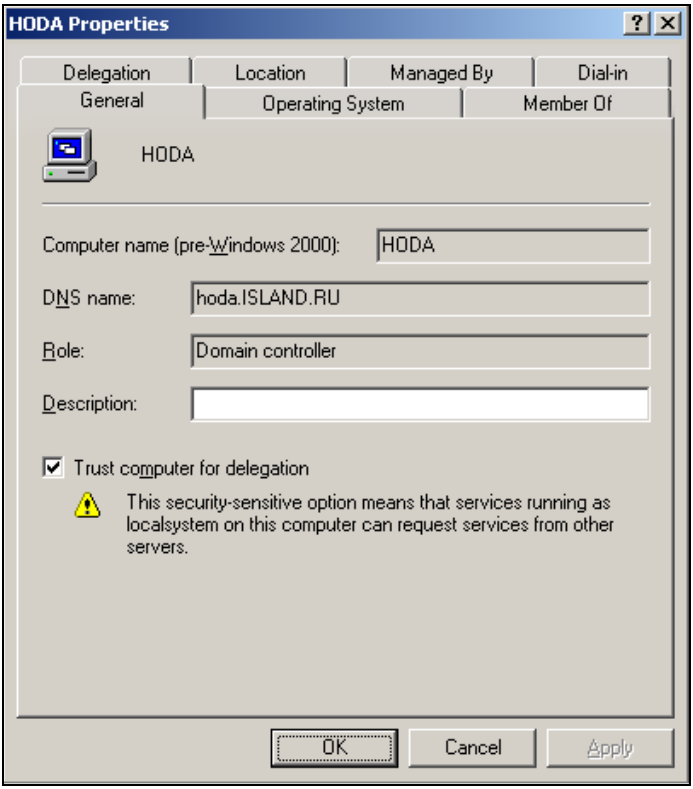



**Рис. 7.4.** Учетная запись компьютера (по умолчанию)

Глава 8



# Управление учетной записью пользователя

В этой главе рассказано об управлении учетной записью пользователя с помощью мастера и сценария. Приводится сопоставление названий полей мастера и соответствующих им атрибутов в каталоге Active Directory.

## Основные термины и понятия

- **Home Directory** — сетевая папка, расположенная на сервере, содержимое которой подключается в качестве сетевого диска, которому назначается указанная в свойствах учетной записи пользователя буква. Это происходит каждый раз при входе пользователя в сеть.
- **Login** — сокращенное имя учетной записи пользователя для регистрации в сети.
- **Logon Script** — сценарий регистрации пользователя в сети. Располагается в папке NetLogon любого из контроллеров домена. Для выполнения сценария его имя должно быть указано в свойствах учетной записи пользователя.
- **Profile** — сетевая папка, в которой находится выгружаемый на рабочую станцию профиль пользователя во время его регистрации в сети.

## Производимые операции

Учетная запись пользователя является одним из основных объектов Active Directory, который можно создать, удалить, переместить, изменить свойства

и т. д. Уделим особое внимание созданию, удалению и изменению свойств учетной записи пользователя. Остальные возможные операции с объектами подробно рассмотрены в *главах 6 и 7*. Все операции рассмотрены как на примере работы мастера, так и сценария. Особое внимание уделено взаимосвязям полей мастера и атрибутов Active Directory.

# Создание пользователя

Рассмотрим работу мастера, создающего учетную запись пользователя, обращая внимание на изменения, происходящие в Active Directory. Во время работы мастера системный администратор определяет минимальное количество параметров, необходимых для создания учетной записи пользователя: отображаемое имя и имя в пользователя в сети, а также некоторые параметры безопасности. Остальные поля заполняются после завершения процесса создания учетной записи.

Запуск мастера осуществляется из MMC-консоли **Active Directory Users and Computers** выбором пункта **New | User** контекстного меню папки (рис. 8.1), в которой необходимо создать учетную запись.

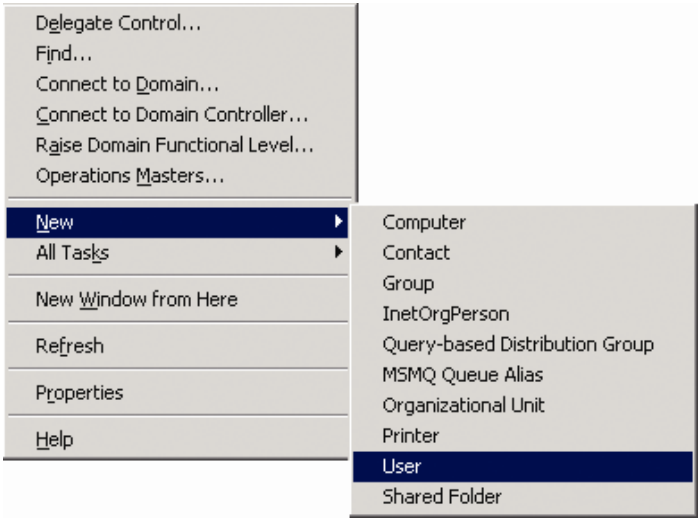



**Рис. 8.1.** Запуск мастера **Создание учетной записи пользователя**

# Работа мастера. Теория

Работа мастера при создании учетной записи пользователя состоит из трех шагов. На первом шаге указываются имя пользователя и логин, на втором — устанавливаются правила безопасности учетной записи, на третьем — осуществляется подтверждение заданных ранее параметров. Рассмотрим работу мастера подробнее.

## Шаг 1. Назначение имени пользователя в сети

Во время создания учетной записи пользователя на первом шаге (рис. 8.2) необходимо указать фамилию, имя и отчество пользователя.

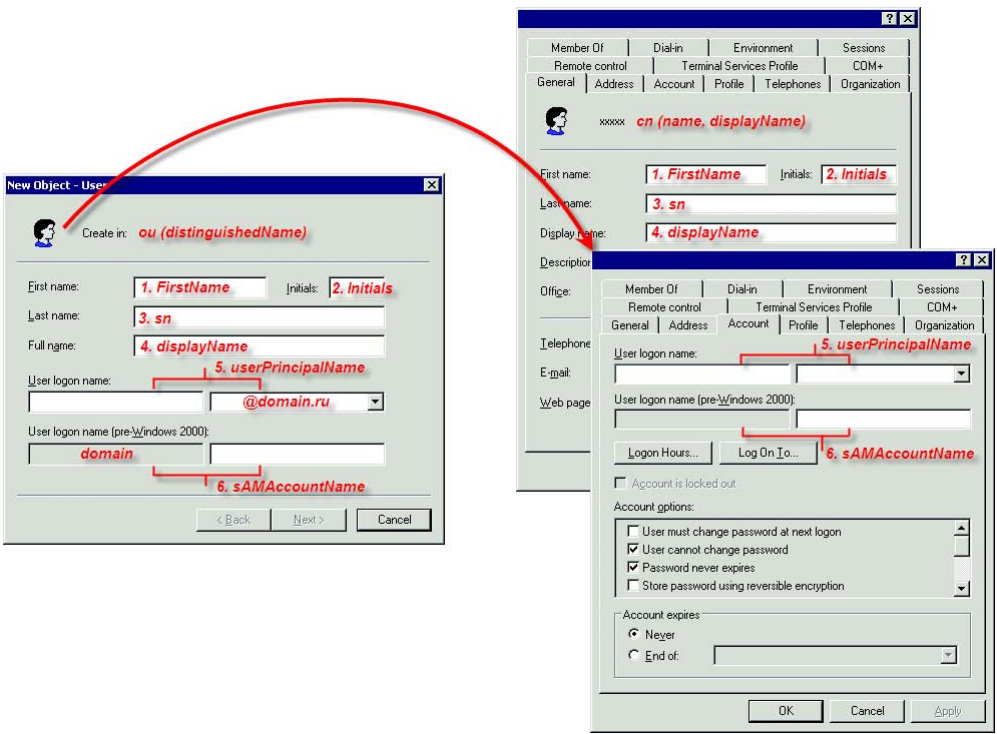



**Рис. 8.2.** Создание учетной записи пользователя. Шаг 1

Необязательно задавать все три параметра: достаточно задать один из них. Полное имя (поле **Full name** на рис. 8.2) формируется на основе Ф.И.О.,

заданных с помощью первых трех полей. Хотя его значение формируется автоматически, администратор может изменить его на любое другое. Первые четыре параметра могут быть изменены и позже. Для этого необходимо войти в свойства пользователя на вкладку **General**, отображаемую по умолчанию.

Вторая группа параметров — имя пользователя в сети и дополнительное имя для совместимости с доменами, построенными на основе Windows NT. В Active Directory им соответствуют параметры `userPrincipalName` и `sAMAccountName`. Каждое из этих имен состоит из двух частей: имени пользователя и текущего домена. Имя текущего домена определяется автоматически и не может быть изменено.

В табл. 8.1 приведено описание полей, задействованных на первом шаге работы мастера.

***Таблица 8.1.*** *Параметры учетной записи пользователя. Шаг 1*

| Параметр мастера | Поле в Active Directory | Тип данных | Вкладка в свойствах объекта USER | Описание |
|---|---|---|---|---|
| **First name** | `firstName` | String | **General** | Имя сотрудника |
| **Initials** | `initials` | String | **General** | Инициалы (до 6 символов) |
| **Last name** | `sn` | String | **General** | Фамилия сотрудника |
| **Full name** | `displayName` | String | **General** | Полное имя сотрудника (на основе первых трех полей) |
| **User logon name** | `userPrincipalName` | String | **Account** | Имя регистрации пользователя в сети |
| **User logon name (pre-Windows 2000)** | `sAMAccountName` | String | **Account** | Имя регистрации пользователя в сети для учетных записей Windows NT |

## Шаг 2. Настройка политики безопасности учетной записи пользователя

На втором этапе работы мастера администратор задает пароль пользователя, длина которого определяется с помощью доменной политики безопасности **Minimum Password Length**, расположенной в разделе **Computer Configuration | Windows Settings | Security Settings | Account Policies | Password Policy** оснастки групповых политик. Так определяется политика безопасности учетной записи.

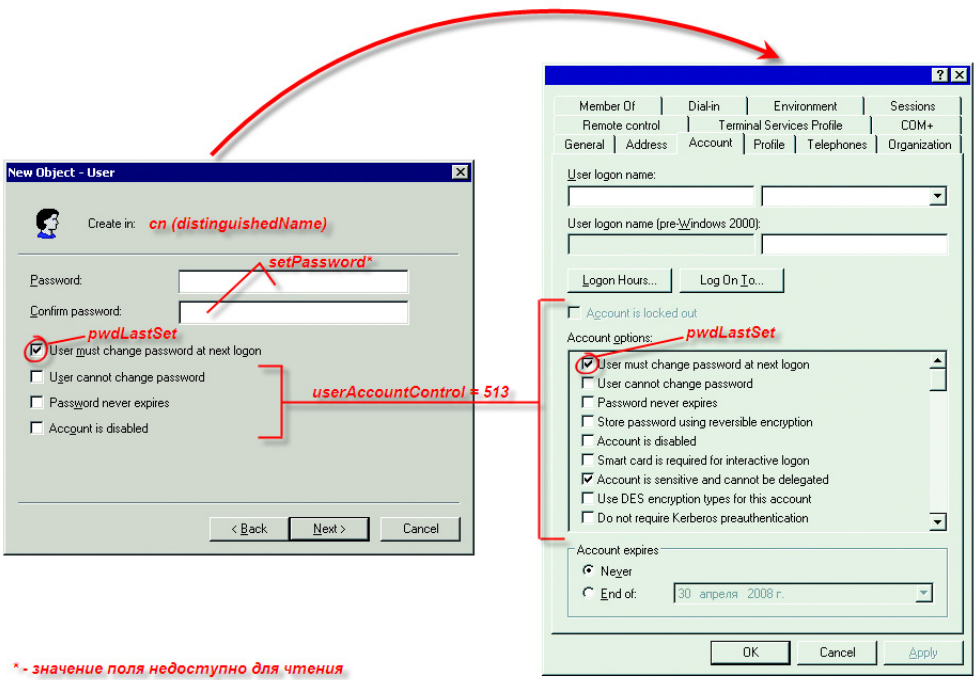



**Рис. 8.3.** Создание учетной записи пользователя. Шаг 2

На рис. 8.3 показано, какими параметрами Active Directory может управлять системный администратор на этом этапе создания учетной записи пользователя. К этим параметрам относятся:

- `setPassword`. Значение этого обязательного параметра — пароль, хранящийся в зашифрованном виде в Active Directory. С помощью сценария

его можно только записать. Считать существующее значение невозможно — оно зашифровано необратимым алгоритмом;

- `pwdLastSet`. Данному атрибуту каталога Active Directory соответствует параметр мастера **User must change password at next logon** (см. рис. 8.3). С его помощью управляют возможностью принудительного изменения пароля учетной записи пользователя при следующем его входе в сеть. По умолчанию принимает значение 0 (сменить пароль);
- `userAccountControl`. Оставшиеся три параметра (**User cannot change password**; **Password never expires**; **Account is disabled**) формируют в совокупности значение параметра `userAccountControl`. Подробное описание возможных значений параметра см. в табл. 8.7. По умолчанию `userAccountControl` равно 513 (стандартная учетная запись пользователя).

Все перечисленные параметры, кроме `setPassword`, можно изменить на вкладке **Account** учетной записи пользователя.

### Шаг 3. Проверка параметров создаваемой учетной записи пользователя

После завершения второго шага можно перейти к третьему, завершающему этапу, на котором осуществляется проверка заданных параметров. Здесь выводится следующая информация для контроля введенных параметров:

- местоположение пользователя в Active Directory;
- отображаемое имя пользователя в сети;
- имя пользователя для регистрации в сети;
- параметры безопасности, которые устанавливаются на втором шаге работы мастера.

Пример выводимой информации приведен в *разд. "Работа мастера. Практика"* на рис. 8.7.

## Работа мастера. Практика

Создадим учетную запись пользователя с помощью мастера. Исходные данные приведены в табл. 8.2.

***Таблица 8.2.*** *Исходные данные для создания учетной записи пользователя*

| Поля мастера | Значение | Комментарий |
|---|---|---|
| **Create In** | ISLAND.RU\ WorkSpace | Папка WorkSpace находится в корневом каталоге домена ISLAND |
| ***Шаг 1*** | | |
| **First name** | Ivanov | Фамилия |
| **Initials** | V | Инициал |
| **Last name** | Petr | Отчество |
| **Full name** | Ivanov, Petr | Отображаемое имя |
| **User logon name** | PIvanov | Имя для входа в сеть |
| **User logon name (pre-Windows 2000)** | | Имя для входа в сеть (для совместимости с Windows NT доменами) |
| ***Шаг 2*** | | |
| **Password, Confirm password** | 123456789 | Пароль |
| **User must change password at next logon** | ДА | Пользователь должен изменить пароль после первого входа в сеть |
| **User cannot change password** | НЕТ | Пользователь не может изменить пароль |
| **Password never expires** | НЕТ | Действие пароля неограниченно |
| **Account is disabled** | НЕТ | Учетная запись выключена |

Рассмотрим пошагово создание учетной записи. После входа на контроллер домена в сети необходимо запустить оснастку управления **Active Directory Users and Computers**. В появившемся окне необходимо установить курсор в папке WorkSpace и нажатием правой кнопки мыши вызвать контекстное ме-

ню (рис. 8.4). В нем выбрать **New | User**. Запустится мастер установки учетной записи пользователя.

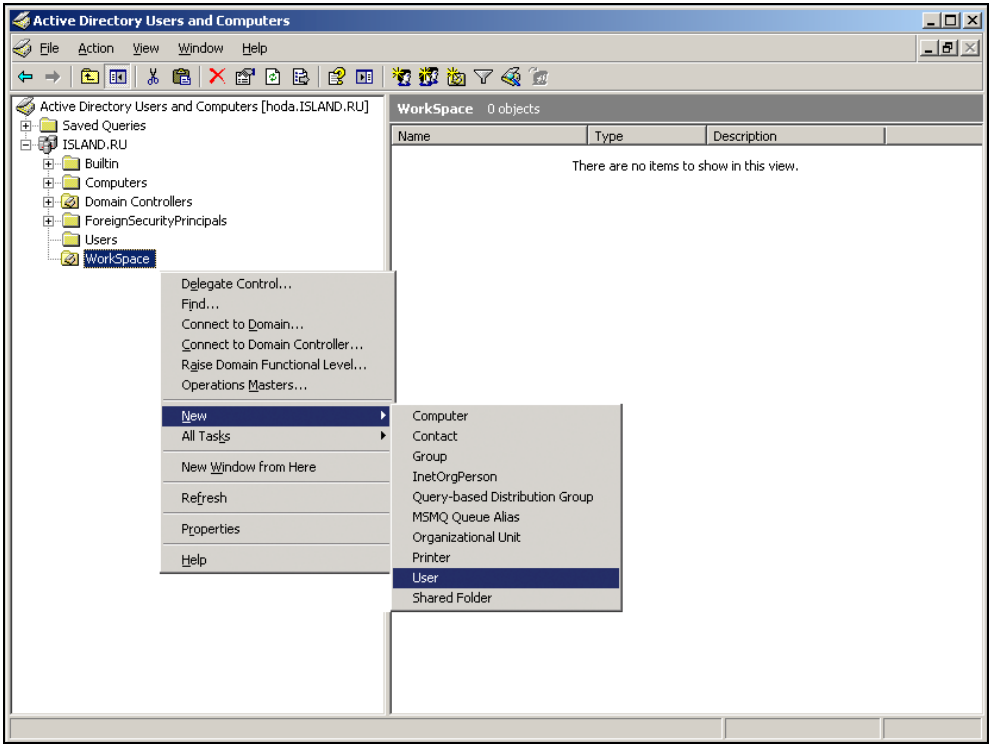



**Рис. 8.4.** Запуск мастера создания учетной записи пользователя

В появившемся диалоговом окне заполните поля в соответствии с данными, приведенными в колонке "Значение" табл. 8.2. Обратите внимание, что после заполнения первых трех полей при переходе на четвертое поле оно заполняется автоматически (рис. 8.5).

Имя пользователя для регистрации в сети задается системным администратором. На основе введенных данных автоматически генерируется имя для поддержки доверительных отношений (pre-Windows 2000), которое можно изменить на требуемое.

На втором шаге необходимо ввести пароль и подтвердить его. Остальные четыре параметра безопасности оставим без изменений (рис. 8.6).

По назначению пароля существуют следующие рекомендации:

- пароли чувствительны к регистру символов;
- используйте цифры в паролях;
- рекомендуемая длина пароля — не менее 7 символов;
- в пароле не могут быть использованы символы `"/\|:;,+*?<>`;
- не используйте в пароле имя учетной записи пользователя;
- не используйте слишком простых паролей, например, `0123456789`.

Контроль над выполнением некоторых из перечисленных требований может быть возложен на доменные групповые политики.

На последнем этапе создания учетной записи осуществляется контроль установленных параметров (рис. 8.7).

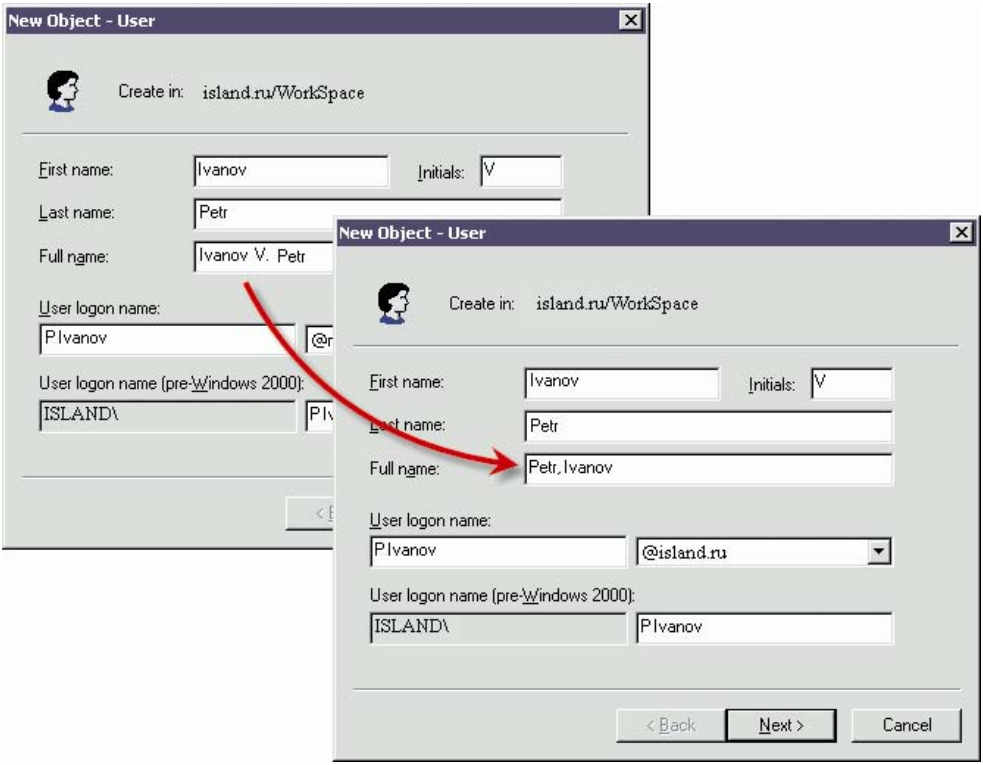



**Рис. 8.5.** Работа мастера. Шаг 1

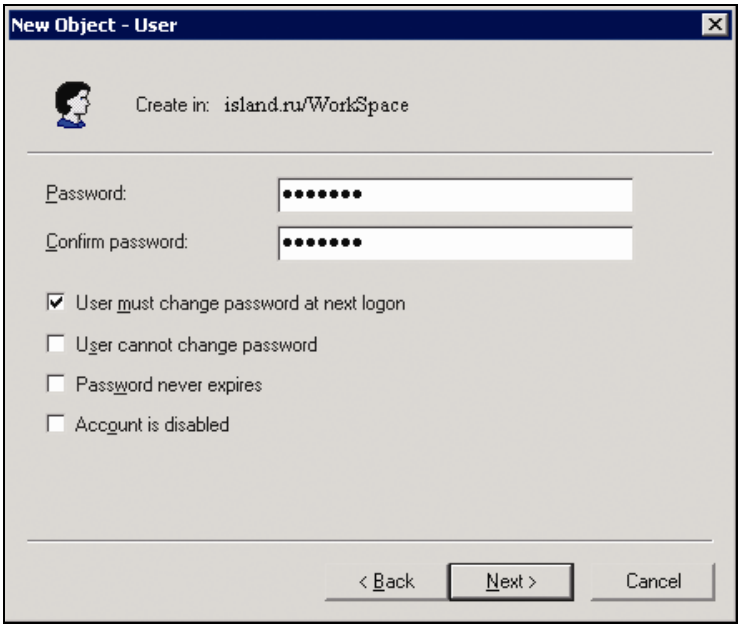



**Рис. 8.6.** Работа мастера. Шаг 2

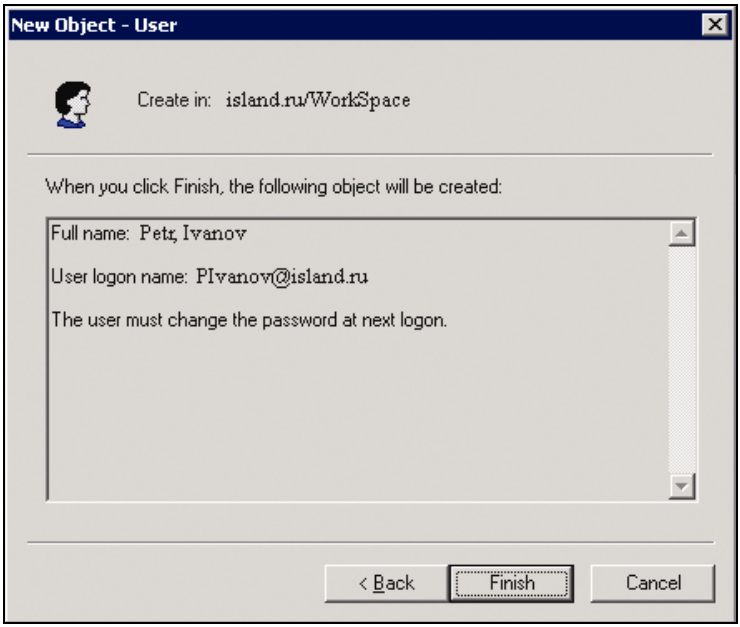



**Рис. 8.7.** Работа мастера. Шаг 3

# Программное создание учетной записи пользователя

При создании учетной записи пользователя программным способом (с помощью VBScript), как и при работе мастера, необходимо задать несколько параметров. В листинге 8.1 приведен сценарий, позволяющий создать учетную запись пользователя. Результат действия сценария аналогичен работе мастера, который был рассмотрен в предыдущем разделе. В табл. 8.3 приведены параметры и соответствующие им значения, которые необходимо задать в Active Directory.

***Таблица 8.3.*** *Параметры сценария для создания учетной записи пользователя*

| Параметр | Комментарий | Способ назначения |
|---|---|---|
| `Cn = Petr\, Ivanov` | Отображаемое имя | Явный |
| `sAMAccountName = PIvanov` | Имя в сети для совместимости с доменами Windows NT | Явный |
| `sn = Ivanov` | Фамилия | Явный |
| `givenName = Petr` | Имя | Явный |
| `initials = V` | Инициалы | Явный |
| `userPrincipalName = PIvanov@island.ru` | Имя в сети Windows 2k домена | Явный |
| `setPassword = 123456789` | Назначаемый пароль | Явный |
| `pwdLastSet = 0` | Изменить пароль при следующем входе в сеть | Явный |
| `distinguishedName = CN=Petr\, Ivanov, OU=WorkSpace,DC=island,DC=ru` | Путь к создаваемой учетной записи | Неявный |
| `objectClass = person` | Тип объекта | Неявный |

**Листинг 8.1. Создание учетной записи пользователя**

```
' Определение RDN-имени (LDAP-имени) домена
set RootDSE = GetObject("LDAP://RootDSE")
Domain = rootDSE.Get("defaultNamingContext")
```

```
' Подключение к контейнеру и создание объекта
Set objUsers = GetObject("LDAP://OU=WorkSpace," & Domain)
Set objNewUser = objUsers.Create("user", "cn=Petr\, Ivanov")

' Назначение параметров учетной записи
objNewUser.Put "sAMAccountName", "PIvanov"
objNewUser.Put "sn", "Ivanov"
objNewUser.Put "givenName", "Petr"
objNewUser.Put "Initials", "V"
objNewUser.Put "userPrincipalName", "PIvanov@" & DetectDNSName(Domain)
' Запись данных в каталог Active Directory
objNewUser.SetInfo

' Включение учетной записи и назначение пароля
objNewUser.AccountDisabled = False
objNewUser.SetPassword("1234567890")
objNewUser.SetInfo

Function DetectDNSName(Domain)
' Определение DNS-имени домена
    LDAPArray = Split(Domain,",")
    For Each el In LDAPArray
        DNSName = DNSName + Right(el,Len(el)-3) + "."
    Next
    DetectDNSName = Left(DNSName, Len(DNSName)-1)
End Function
```




**Рис. 8.8.** Ошибка работы сценария VBScript

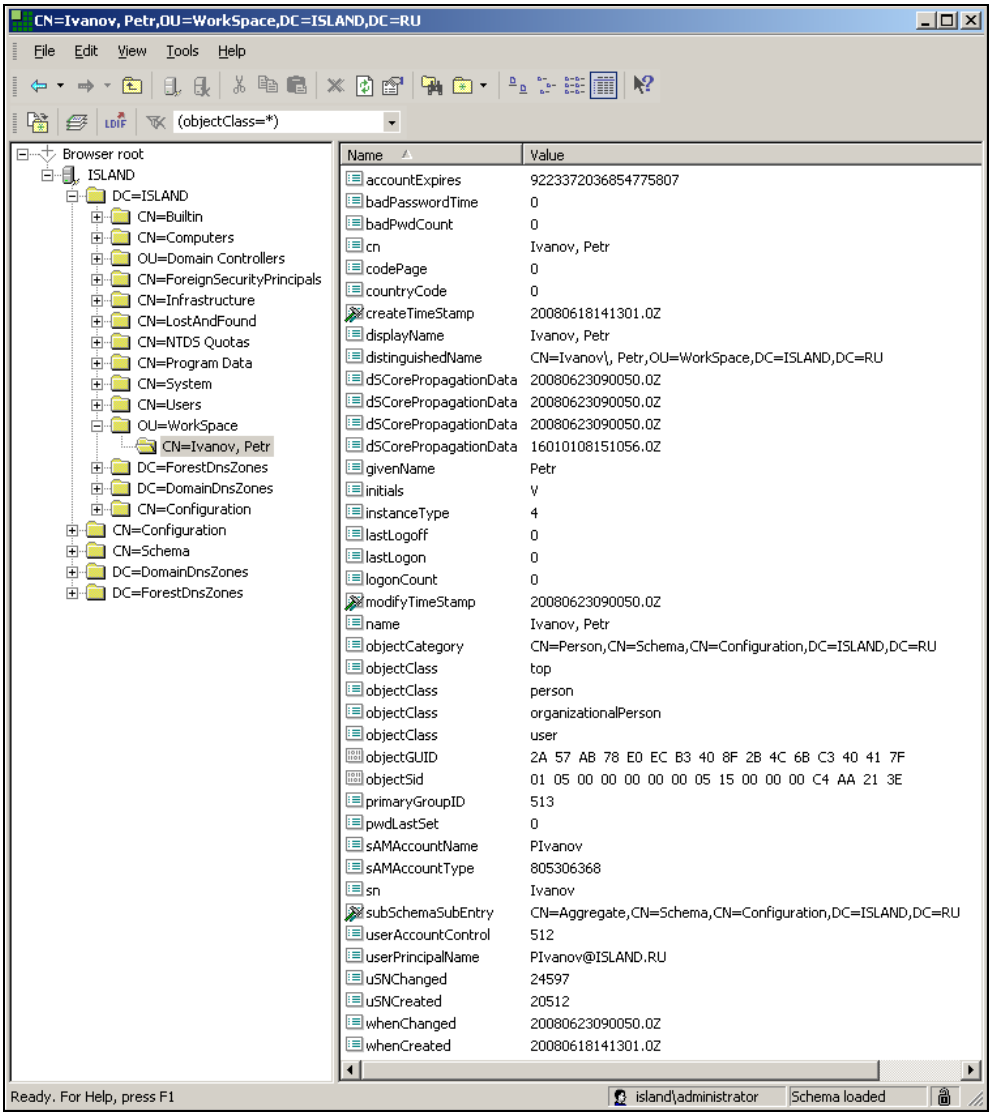



**Рис. 8.9.** Параметры учетной записи пользователя, созданной с помощью мастера

В приведенном сценарии есть несколько особенностей, на которые необходимо обратить внимание.

- Для того чтобы в имени учетной записи (поле `cn`) присутствовала запятая, необходимо при присвоении значения перед запятой поставить косую черту (\): `cn=Petr\, Ivanov`. В противном случае при выполнении инст-

рукции создания нового пользователя интерпретатор VBScript выдаст сообщение об ошибке (рис. 8.8).

- Значение параметра `userPrincipalName` содержит суффикс, в данном случае @island.ru. Им является DNS-имя текущего домена. В сценарии используется функция `DetectDNSName()`, которая преобразует LDAP-имя в DNS-имя.
- Значение свойства `AccountDisabled` не может быть изменено, пока объект не существует, поэтому метод `SetInfo()` в скрипте используется дважды.

## Поля учетной записи, создаваемой мастером

Во время создания учетной записи пользователя в Active Directory создается 34 атрибута (рис. 8.9). Часть атрибутов задается явным образом, другая часть — неявным. Остальные задаются в автоматическом режиме.

# Удаление пользователя

Рассмотрим удаление учетной записи пользователя с помощью мастера и программным способом.

## Удаление с помощью мастера

Удаление учетной записи пользователя осуществляется из MMC-оснастки Active Directory. Чтобы удалить объект в Active Directory, необходимо либо установить на него курсор и нажать клавишу <Delete> на клавиатуре, либо вызвать контекстное меню объекта и выбрать в нем пункт **Delete** (рис. 8.10).

Поскольку объекты удаляются безвозвратно, то при удалении объекта запрашивается подтверждение. После удаления объект можно восстановить только из резервной копии Active Directory.

## Удаление программным способом

В сценарии для удаления объекта необходимо указать тип удаляемого объекта и путь к этому объекту в каталоге Active Directory. Оба параметра в сумме составляют значение `distinguishedName`.

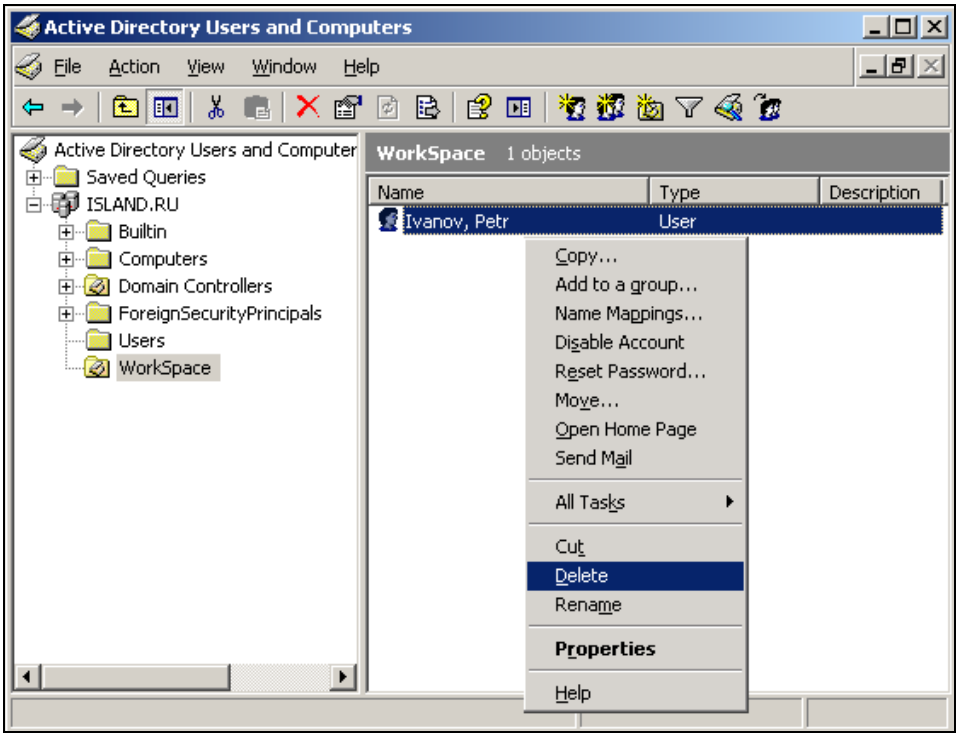



**Рис. 8.10.** Удаление учетной записи пользователя

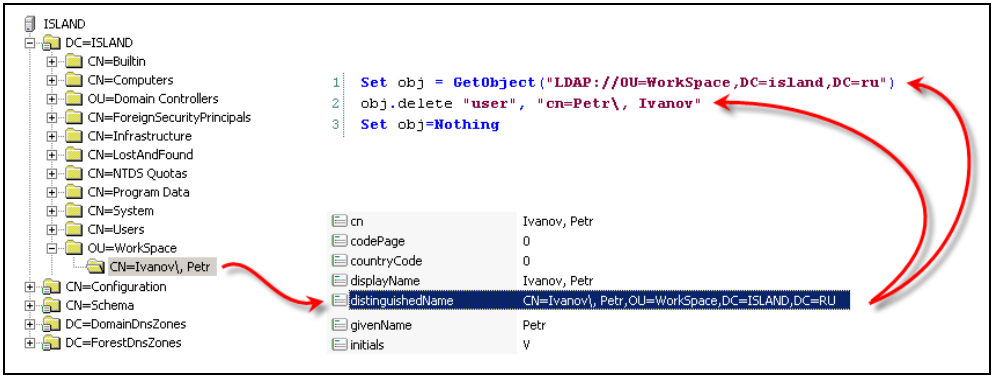



**Рис. 8.11.** Структура параметра `distinguishedName`

Например, если необходимо удалить учетную запись Petr, Ivanov (рис. 8.11), то следует определить значение параметра `distinguishedName` и выделить

имя объекта (значение `cn`) и папку, в которой находится объект (остальные параметры RDN).

Удаление объекта осуществляется при вызове метода `Delete()`. Метод `SetInfo()` при этом не используется. При попытке его вызова интерпретатор VBScript выдаст сообщение об ошибке.

# Чтение свойств пользователя

В этом разделе будут рассмотрены параметры, отображаемые в 12-ти из 13-ти вкладок, которые видны в стандартном режиме просмотра Active Directory.

## Вкладка *General*

На вкладке **General** задаются параметры, касающиеся непосредственно сотрудника, которому соответствует созданная учетная запись. К этим параметрам относятся имя, фамилия, отображаемое имя в сети и контактная информация: телефоны, размещение, адрес электронной почты, ссылки на Web-страницы. Вкладка **General** (рис. 8.12) открывается по умолчанию при вызове свойств учетной записи любого объекта из Active Directory, например, группы или пользователя. В качестве значений параметров указаны названия соответствующих им полей в Active Directory. В табл. 8.4 эта информация дополнена описанием соответствующих им типов данных.

***Таблица 8.4.*** *Соответствие полей на вкладке* ***General*** *параметрам в Active Directory*

| **Поле на вкладке *General*** | **Тип** | **Поле в Active Directory** | **Тип** |
|---|---|---|---|
|  | — | `cn`<br>`name` | String |
| **First name** | InputBox | `givenName` | String |
| **Initials** | InputBox | `initials` | String |
| **Last name** | InputBox | `sn` | String |
| **Display name** | InputBox | `displayName` | String |

***Таблица 8.4*** *(окончание)*

| Поле на вкладке *General* | Тип | Поле в Active Directory | Тип |
|---|---|---|---|
| **Description** | InputBox | description | String |
| **Office** | InputBox | physicalDeliveryOfficeName | String |
| **Telephone number** | InputBox | telephoneNumber | String |
| **Other** | Button | otherTelephone | Array |
| **E-mail** | InputBox | mail | String |
| **Web page** | InputBox | wWWHomePage | String |
| **Other** | Button | url | Array |

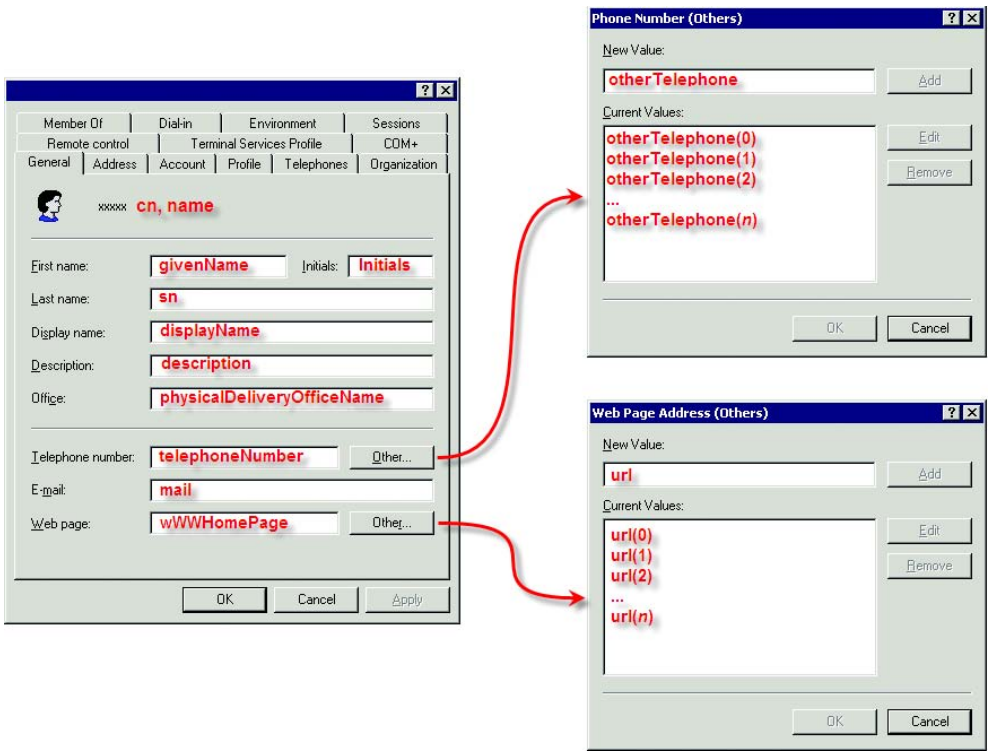



**Рис. 8.12.** Вкладка **General**

## Описание полей

Рассмотрим подробно каждый из задаваемых параметров.

### *User Name*

Имя пользователя отображается около значка, изображающего человечка:  .

На вкладке **General** его невозможно изменить. В Active Directory этому параметру соответствуют два параметра, значения которых совпадают: `name` и `cn`. Для изменения этих параметров необходимо переименовать пользователя.

### *Display name*

Значение параметра **Display name** — отображаемое имя пользователя, которое видит администратор, войдя в Active Directory. Это же имя присутствует в адресной книге почтового клиента, например, Microsoft Outlook. Значение этого параметра фиксируется в параметре `displayName` в Active Directory.

Значение этого обязательного параметра складывается из суммы значений трех параметров: **First name**, **Initials** и **Last name** по шаблону:

```
a b. c
```

Здесь `a` — имя пользователя, `b` — инициал (6 символов), `c` — фамилия. Хотя бы один из этих трех параметров должен быть задан, однако по отдельности каждый из них не является обязательным.

### *First name*

Необязательное поле **First name** содержит имя сотрудника, которому соответствует настоящая учетная запись. В Active Directory этому параметру соответствует поле `givenName`.

### *Initials*

Поле **Initials** — необязательный параметр длиной не более 6 символов. Значению этого параметра соответствует инициал пользователя. Ему соответствует одноименный атрибут Active Directory. Чаще всего этот параметр не заполняют.

### Last name

Значение параметра **Last name** — фамилия сотрудника, для которого создана учетная запись. В Active Directory ему соответствует параметр `sn`. Как правило, из трех параметров задают только два: имя и фамилию.

### Description

Поле **Description** обычно заполняют каким-либо комментарием. Например, если имя и фамилия пользователя в домене задаются латинской транслитерацией, то в это поле можно занести ФИО на русском языке. В Active Directory данные хранятся в одноименном поле.

### Office

Указывается физическое месторасположение пользователя: комната, офис и т. д. Параметру **Office** в Active Directory соответствует параметр `physicalDeliveryOfficeName`.

### Telephone number

В поле **Telephone number** обычно хранится номер телефона сотрудника. В Active Directory ему соответствует поле `telephoneNumber`.

Если у сотрудника несколько телефонных номеров, то их можно занести в список, нажав кнопку **Other**, расположенную рядом с полем **Telephone number** на вкладке **General**. В появившемся диалоговом окне (см. рис. 8.12) с помощью мастера добавляют номера телефонов в список. В Active Directory списку соответствует массив `otherTelephone`, элементы которого — строки.

### E-mail

Этот параметр (и соответствующий атрибут `mail` в Active Directory) автоматически заполняется в соответствии с форматом UPN (RFC 822) при создании почтового ящика для учетной записи пользователя. По умолчанию значение полю не присвоено.

### Web page

В поле **Web page** указывают ссылку на Web-страницу сотрудника. В Active Directory ему соответствует поле `wWWHomePage`.

Если у сотрудника несколько ссылок на Web-сайты, то их можно занести в список, нажав кнопку **Other**, относящуюся к полю **Web page** вкладки **General**. В появившемся диалоговом окне (см. рис. 8.12) с помощью мастера добавляют ссылки на сайты. В Active Directory списку соответствует массив `url`.

## Вкладка *Address*

На вкладке **Address** (рис. 8.13) сосредоточены параметры, касающиеся местоположения человека, которому соответствует учетная запись пользователя: его почтовый адрес, включая регион, индекс, город. Все перечисленные параметры необязательны, и в мастере создания учетной записи они недоступны.

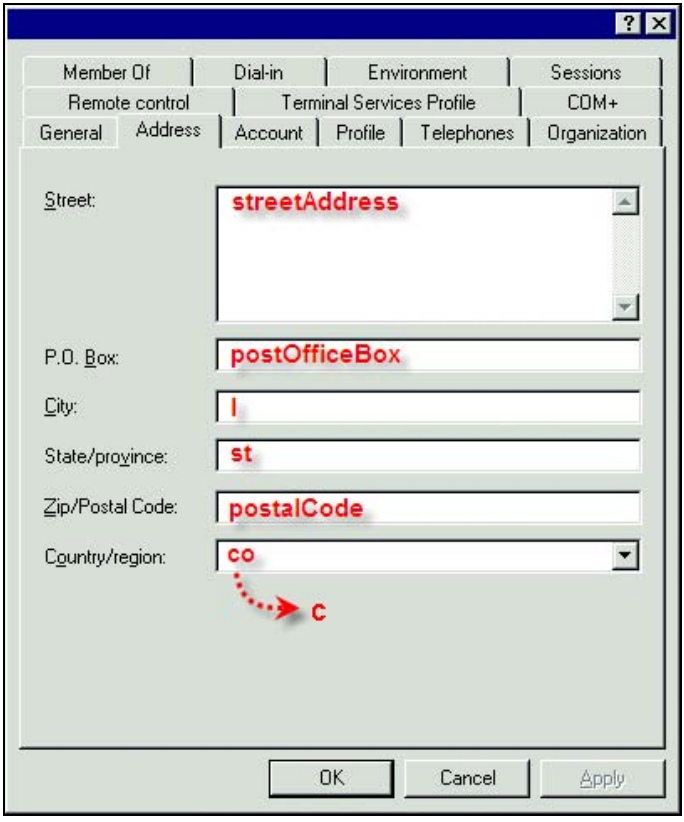



**Рис. 8.13.** Вкладка **Address**

В табл. 8.5 приведены соответствия описанных параметров полям в Active Directory.

***Таблица 8.5.*** *Соответствия полей на вкладке* ***Address*** *параметрам в Active Directory*

| Поле на вкладке *Address* | Тип | Поле в Active Directory | Тип |
|---|---|---|---|
| **Street** | InputBox | `streetAddress` | String |
| **P.O. Box** | InputBox | `postOfficeBox` | String |
| **City** | InputBox | `l` | String |
| **State/province** | InputBox | `st` | String |
| **Zip/Postal Code** | InputBox | `postalCode` | String |
| **Country/region** | InputBox | `co` | String |
| | ComboBox | `c` | String |

## Описание полей

Опишем смысл, стоящий за полями вкладки **Address**, и способы работы с ними.

### *Street*

Параметру **Street** соответствует поле `streetAddress` в Active Directory. Значение поля — адрес местонахождения сотрудника с указанием улицы, номера дома, корпуса, номера офиса. Параметр `streetAddress` — строка, которая разбивается на подстроки с помощью перевода каретки на новую строку. На VBScript для этого используется переменная `vbNewLine` или выражение `chr(13) & chr(10)`.

### *P.O. Box*

Значением параметра **P.O. Box** является почтовый индекс. Несмотря на то, что в России индекс — число, тип данных этого параметра, которому в Active Directory соответствует поле `postOfficeBox`, — строка. Это сделано так, поскольку в некоторых странах почтовый индекс включает в себя буквенный код.

### *City*

В поле **City** указывается название города, в котором находится сотрудник. Этому полю соответствует строковый параметр `l`.

### *State/province*

В этом поле указываются штат, провинция или область в зависимости от административного деления, используемого в стране проживания сотрудника. Полю **State/province** соответствует строковый параметр `st`.

### *Zip/Postal Code*

Здесь задается код города в строковом виде. В Active Directory этому полю соответствует параметр `postalCode`.

### *Country/region*

Регион, выбирается из заданного списка. При выборе региона в Active Directory создаются два поля — `co` и `c`. Выбранное значение страны из списка в текстовом виде фиксируется в параметре `co`. Параллельно создается еще одно поле — `c`, значение которого — сокращенное название страны. Для России это RU, для США — US и т. д. Полный список кодов стран приведен в *приложении 5*.

## Вкладка *Account*

На вкладке **Account** (рис. 8.14) собраны настройки, характеризующие правила доступа пользователя к сети, включая имя входа в сеть. В табл. 8.6 приведены описания полей вкладки **Account** и атрибуты, соответствующие им в Active Directory. Для поля `userAccountControl` в скобках указаны значения, соответствующие установке только этого флажка.

***Таблица 8.6.*** *Соответствия полей на вкладке* ***Account*** *параметрам в Active Directory*

| **Поле на вкладке *Account*** | **Тип** | **Поле в Active Directory** | **Тип** |
|---|---|---|---|
| **User logon name** | InputBox | `userPrincipalName` | String |
| | ListBox | | |

***Таблица 8.6*** *(продолжение)*

| **Поле на вкладке *Account*** | **Тип** | **Поле в Active Directory** | **Тип** |
|---|---|---|---|
| **User logon name (pre-Windows 2000)** | InputBox (Read)<br>InputBox | `sAMAccountName` | String |
| **Logon Hours** | Button | `logonHours` | Binary |
| **Log On To** | Button | `userWorkStations` | String |
| **Account is locked out** | CheckBox | `userAccountControl (16)` | Integer |
| **User must change password at next logon** | CheckBox | `pwdLastSet` | Integer |
| **User cannot change password** | CheckBox | `userAccountControl (64)` | Integer |
| **Password never expires** | CheckBox | `userAccountControl (65536)` | Integer |
| **Store password using reversible encryption** | CheckBox | `userAccountControl (128)` | Integer |
| **Account is disabled** | CheckBox | `userAccountControl (2)` | Integer |
| **Smart card is required for interactive logon** | CheckBox | `userAccountControl (262144)` | Integer |
| **Account is sensitive and cannot delegated** | CheckBox | `userAccountControl (1048576)` | Integer |
| **Use DES encryption types for this account** | CheckBox | `userAccountControl (2097152)` | Integer |
| **Do not require Kerberos preauthentication** | CheckBox | `userAccountControl (4194304)` | Integer |

***Таблица 8.6*** *(окончание)*

| Поле на вкладке *Account* | Тип | Поле в Active Directory | Тип |
|---|---|---|---|
| **Account expires** | Radio | `userAccountControl` | Integer |
| | ListBox | `AccountExpires` | Integer |

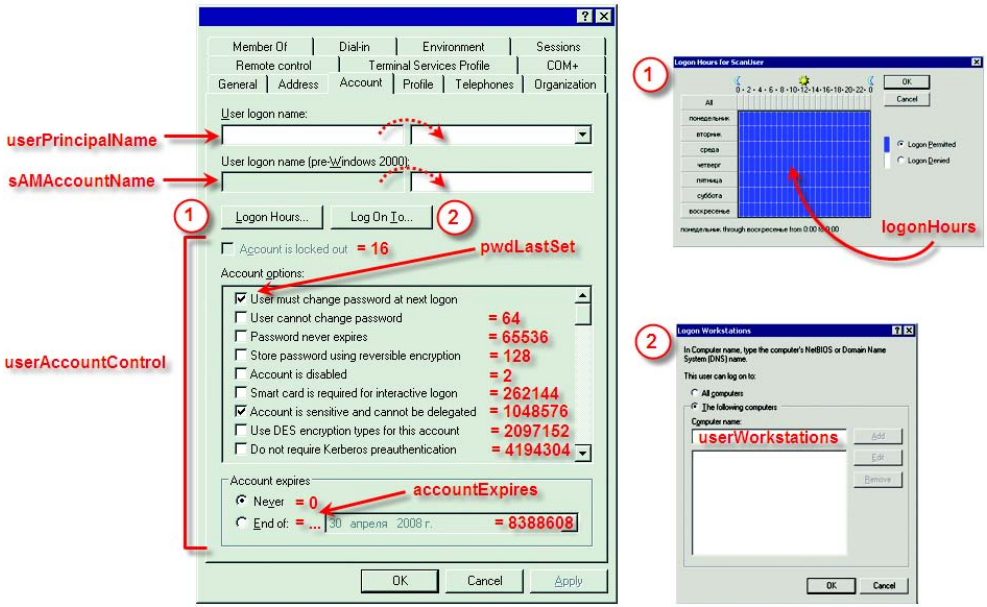



**Рис. 8.14.** Вкладка **Account**

## Описание полей

Опишем подробнее элементы управления, присутствующие на вкладке **Account**.

### *User logon name*

Для совместимости с доменами Windows NT назначаются два имени пользователя, которые могут не совпадать.

Первое — UPN-имя, задаваемое в поле **User logon name**, которому в каталоге Active Directory соответствует поле `userPrincipalName`. Оно имеет формат user@domain, где domain — DNS-имя домена, например Island.RU; user — имя пользователя в сети. Для удобства назначения имен UPN-имя разделено на две части (рис 8.14).

Второе — SAM-имя, которое использовалось в доменах на основе Windows NT. Оно задается в поле **User logon name (pre-Windows 2000)**. Структура SAM-имени следующая: domain\user, где domain — сокращенное имя домена, например Island, user — имя пользователя. Для удобства назначения имени поле также разбито на две части. В Active Directory хранится только имя пользователя, находящееся в поле `sAMAccountName`. Первая часть SAM-имени однозначно вычисляется из DNS-имени домена. Разделение имен пользователя показано на рис. 8.14 пунктирными стрелками.

### *Logon Hours*

С помощью атрибута `logonHours`, хранящегося в каталоге Active Directory в бинарном виде, можно корректировать график, в соответствии с которым пользователь может регистрироваться в сети.

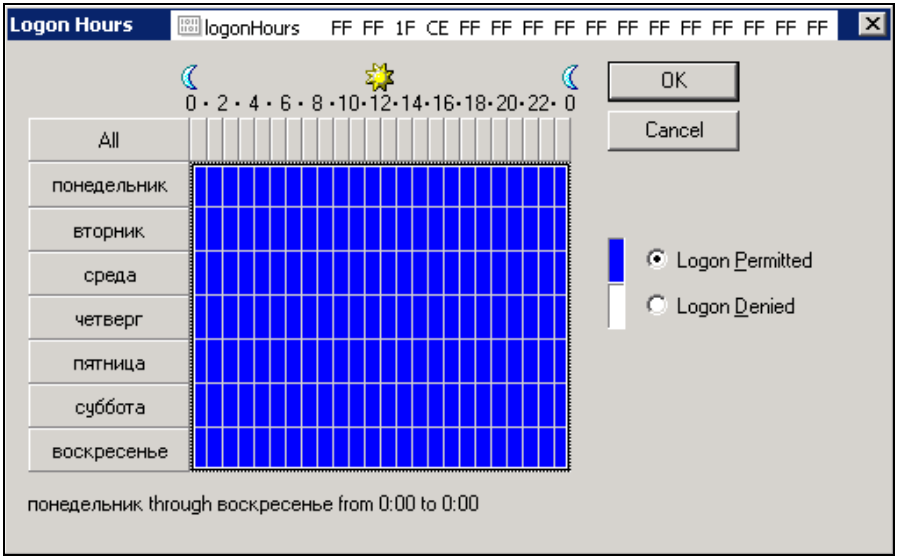



**Рис. 8.15.** Диалоговое окно **Logon Hours**

Значению, состоящему из одних FF (по умолчанию), соответствует круглосуточный доступ с понедельника по воскресенье, значению из нулей — запрет доступа в любое время. На вкладке **Account** присутствует кнопка **Logon Hours**, вызывающая диалоговое окно (рис. 8.15), которое позволяет просмотреть и изменить этот график интерактивно.

### *Log On To*

При нажатии кнопки **Log On To** вызывается диалоговое окно **Logon Workstations** (рис. 8.16), в котором отображается список компьютеров, с которых может осуществляться регистрация данного пользователя в сети.

Введенные администратором значения преобразуются в строку с разделителем. Итоговое значение фиксируется в Active Directory в строковом параметре `userWorkstations`. В качестве разделителя используется запятая.

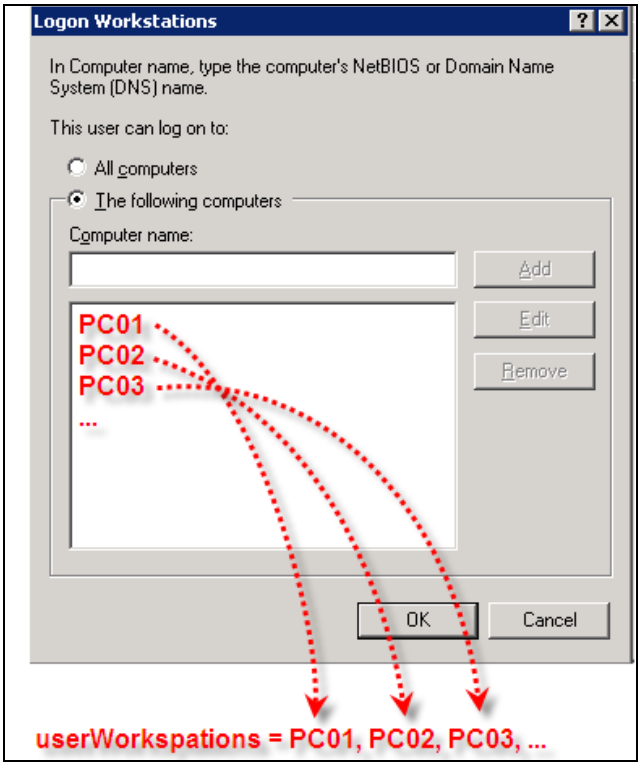



**Рис. 8.16.** Диалоговое окно **Logon Workstations**

### *User must change password at next logon*

Этот флажок входит в группу **Account options**, однако он хранится в Active Directory отдельно в параметре `pwdLastSet`. Установки остальных флажков этой группы хранятся в одном параметре Active Directory и будут описаны далее в *разд. "Группа Account options"*.

Параметр **User must change password at next logon** по умолчанию включен. В Active Directory ему соответствует параметр `pwdLastSet`, который равен 0, когда флажок отмечен. Если же флажок не отмечен, то параметр `pwdLastSet` хранит время с точностью 100 наносекунд, прошедшее с 1 января 1601 года до последней смены пароля пользователем. Техника работы с такими полями описана в листингах 3.15а и 3.15б в *главе 3*.

### *Account expires*

Выбору опции в группе параметров **Account expires** отвечает значение параметра `accountExpires` в Active Directory. Если выбрана опция **Never**, то `accountExpires` равен 0. В противном случае `accountExpires` хранит дату, после наступления которой с помощью данной учетной записи будет невозможно войти в сеть. Так же, как и для параметра `pwdLastSet`, дата хранится как время, прошедшее с 1 января 1601 года, с точностью 100 наносекунд.

### Группа *Account options*

Все параметры данной группы, за исключением первого (рис. 8.14), составляют значение параметра `userAccountControl`, которое образуется путем суммирования всех установленных значений. Однако в табл. 8.6 приведены только те значения, которые можно изменить явным образом с помощью вкладки **Account**. В табл. 8.7 приведены все значения флагов, составляющие в совокупности параметр `userAccountControl`, и описание этих значений.

***Таблица 8.7.*** *Флаговые значения параметра* `userAccountControl`

| Флаг | Значение | Описание |
|---|---|---|
| `SCRIPT` | 1 | Запуск сценария входа |
| **`ACCOUNTDISABLED`**[1] | 2 | Отключение учетной записи пользователя |

[1] Значения в таблице, выделенные жирным шрифтом, могут быть назначены явным образом на вкладке **Account** (рис. 8.14).

***Таблица 8.7*** *(продолжение)*

| Флаг | Значение | Описание |
|---|---|---|
| UNKNOWN | 4 | — |
| HOMEDIR_REQUIRED | 8 | Требуется домашняя папка |
| **LOCKOUT** | 16 | Учетная запись пользователя заблокирована |
| PASSWD_NOTREQD | 32 | Пароль не требуется |
| **PASSWD_CANT_CHANGE** | 64 | Пользователь не может изменить пароль самостоятельно |
| **ENCRYPTED_TEXT_PWD_ALLOWED** | 128 | Пользователь может отправить зашифрованный пароль |
| TEMP_DUBLICATE_ACCOUNT | 256 | Учетная запись для пользователя, чьи основные данные хранятся в другом домене |
| NORMAL_ACCOUNT | 512 | Учетная запись, соответствующая обычному пользователю (используется по умолчанию) |
| UNKNOWN | 1024 | — |
| INTERDOMAIN_TRUST_ACCOUNT | 2048 | Разрешение доверять учетную запись домена другому домену |
| WORKSTATION_TRUST_ACCOUNT | 4068 | Учетная запись для компьютера с ядром Windows 2k |
| SERVER_TRUST_ACCOUNT | 8192 | Учетная запись для контроллера домена, являющегося членом домена |
| UNKNOWN | 16384 | — |
| UNKNOWN | 32768 | — |
| **DONT_EXPIRE_PASSWORD** | 65536 | Срок действия установленного пароля не истекает никогда |

***Таблица 8.7*** *(окончание)*

| Флаг | Значение | Описание |
|---|---|---|
| MSN_LOGON_ACCOUNT | 131072 | Учетная запись входа MSN |
| **SMARTCARD_REQUIRED** | 262144 | Для регистрации в сети требуется смарт-карта |
| TRUSTED_FOR_DELEGATION | 524288 | Учетная запись пользователя или компьютера, из-под имени которой выполняется служба, которой доверяется делегирование Kerberos |
| **NOT_DELEGATED** | 1048576 | Делегирование каких-либо полномочий Kerberos службе отключено |
| **USER_DES_KEY_ONLY** | 2097152 | Для шифрования ключей используется DES-шифрование |
| **DONT_REQ_PREAUTH** | 4194304 | Для входа в сеть не требуется предварительная проверка Kerberos |
| **PASSWORD_EXPIRED** | 8388608 | Срок действия пароля истек |
| TRUSTED_TO_AUTH_FOR_DELEGATION | 16777216 | Учетной записи разрешено безопасное делегирование. Данный параметр разрешает службе использовать учетные данные и проходить проверку подлинности от имени пользователя для других удаленных серверов сети |

## Вкладка *Profile*

На вкладке **Profile** приведены настройки профиля пользователя и его домашнего каталога (рис. 8.17). В табл. 8.8 приведены соответствия полей мастера атрибутам в каталоге Active Directory.

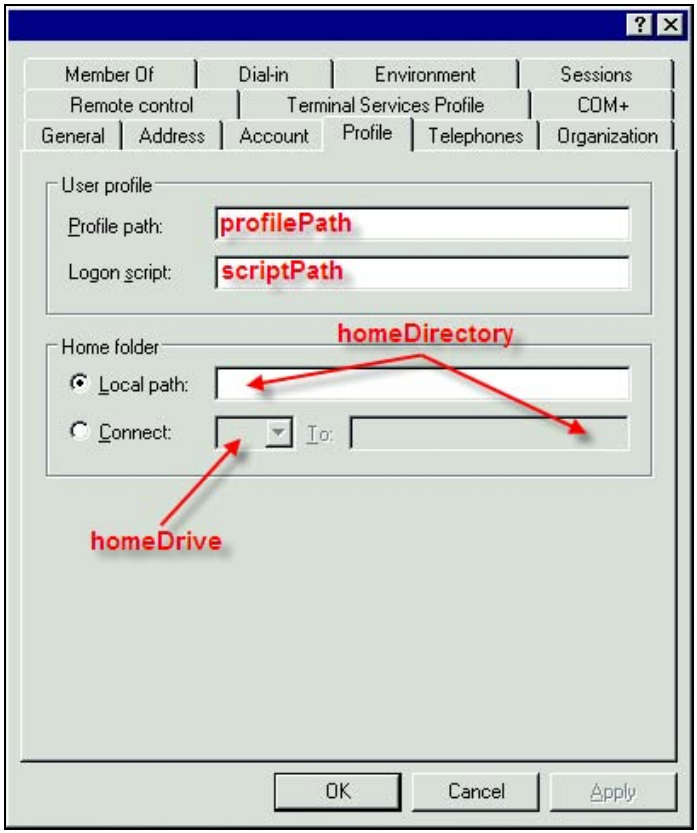



**Рис. 8.17.** Вкладка **Profile**

***Таблица 8.8.*** *Соответствие полей на вкладке* ***Profile*** *параметрам в Active Directory*

| **Поле на вкладке *Profile*** | **Тип** | **Параметр в Active Directory** | **Тип** |
|---|---|---|---|
| **Profile path** | InputBox | `profilePath` | String |
| **Logon script** | InputBox | `scriptPath` | String |
| **Local path** | InputBox | `homeDirectory` | String |
| **Connect** | CheckBox | `homeDrive` | String |
| | InputBox | `homeDirectory` | String |

## Описание полей

Опишем поля на вкладке **Profile** более подробно.

### *Profile path*

Полю **Profile path** соответствует строковый параметр `profilePath` в Active Directory. В качестве значения указывают путь к перемещаемому профилю в виде UNC-пути: \\Server\ShareName\UserNameFolder. Необходимо отметить, что здесь ShareName — имя опубликованной папки в сети на сервере Server. В качестве значения UserNameFolder как правило используют переменную `%username%` (см. *приложение 4*), которая автоматически расшифровывается в сокращенное имя пользователя в сети. В Active Directory значению переменной `%username%` соответствует параметр `sAMAccountName`.

### *Logon script*

В Active Directory полю **Logon script** соответствует значение строкового параметра `scriptPath`.

Сценарий входа, или, как его еще называют, сценарий регистрации пользователей в сети, можно назначить по профилю или с помощью групповых политик (Group Policy). Windows 2k всегда ищет сценарий на домене контроллера в папке %SystemRoot%\SYSVOL\sysvol\DomainName\Scripts, где DomainName — DNS-имя домена, например Island.ru. В сети доступ в этой папке осуществляется через UNC-путь \\DomainName\Netlogon. В качестве сценария может быть использован файл с расширением bat (сценарий на базе командной строки), vbs (Microsoft Visual Basic Script) или js (Microsoft Java Script). При создании сценариев можно использовать различные переменные среды.

### *Home folder*

Домашний каталог предназначен для хранения личных файлов пользователя на сервере. При этом не важно, с какой рабочей станции выполнен вход в сеть. Этот ресурс подключается автоматически.

Домашний каталог может быть задан одним из двух способов:

- как локальный путь, например, C:\Storage\Profiles\PIvanov;
- как сетевой диск, который будет монтироваться каждому сотруднику после регистрации в сети.

### *Local path*

Если домашний каталог должен храниться локально на рабочей станции, то необходимо указать локальный путь, которому в Active Directory соответствует поле `homeDirectory`, при этом значение поля `homeDrive=" "`. Формат пути обычный: C:\FolderName. После регистрации пользователя в сети указанная папка будет создана локально на компьютере пользователя.

### *Connect*

Для того чтобы пользователю после регистрации компьютера в сети монтировалась папка с домашним каталогом на сетевой диск с указанным именем, необходимо перейти в режим **Connect**. В отличие от предыдущего режима, необходимо указать два параметра — имя диска из списка, которому в Active Directory соответствует текстовый атрибут `homeDrive`, и UNC-путь к подключаемой папке, которому соответствует строковый параметр `homeDirectory`. После того как задан UNC-путь, проверяется как его соответствие формату \\Server\ShareName\FolderName, так и наличие папки, к которой предоставлен сетевой доступ.

## Вкладка *Telephones*

На вкладке **Telephones** (рис. 8.18) приведены телефонные номера различных средств связи: домашний, рабочий, факс и др. В табл. 8.9 приведены соответствия параметров полям в Active Directory.

***Таблица 8.9.*** *Соответствие полей на вкладке* ***Telephones*** *параметрам в Active Directory*

| **Поле на вкладке *Telephones*** | **Тип** | **Поле в Active Directory** | **Тип** |
|---|---|---|---|
| **Home** | InputBox | `homePhone` | String |
| **Other** | Button | `otherHomePhone` | Array |
| **Pager** | InputBox | `pager` | String |
| **Other** | Button | `otherPager` | Array |
| **Mobile** | InputBox | `mobile` | String |
| **Other** | Button | `otherMobile` | Array |

***Таблица 8.9*** *(окончание)*

| Поле на вкладке *Telephones* | Тип | Поле в Active Directory | Тип |
|---|---|---|---|
| **Fax** | InputBox | `facsimileTelephoneNumber` | String |
| **Other** | Button | `otherFacsimileTelephoneNumber` | Array |
| **IP phone** | InputBox | `ipPhone` | String |
| **Other** | Button | `otherIpPhone` | Array |
| **Notes** | InputBox | `info` | String |

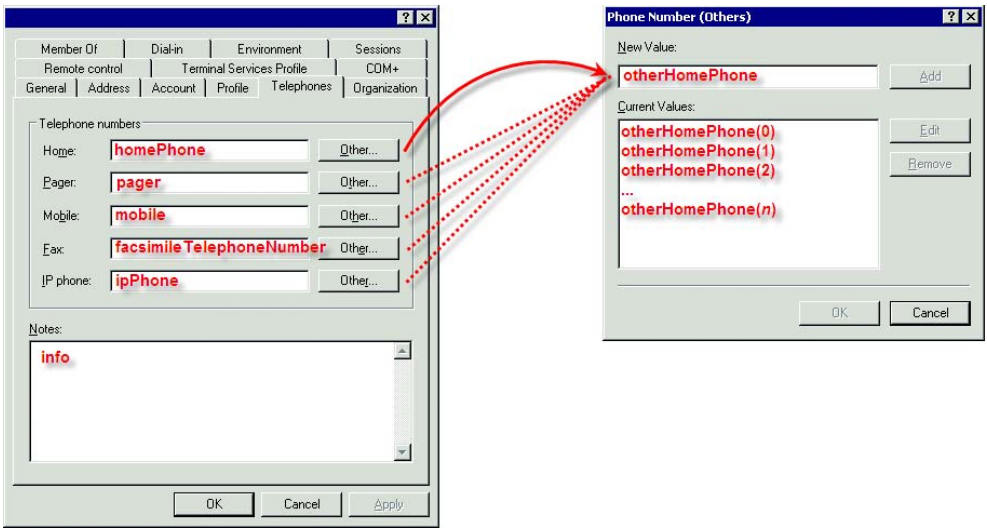



**Рис. 8.18.** Вкладка **Telephones**

## Описание полей

Первые пять полей, в которых задаются номера телефонов, имеют одинаковую структуру. Кнопки, соответствующие им, вызывают единообразные диалоговые окна, в которые данные заносятся одинаково. Поэтому эти поля будут рассмотрены в совокупности.

### Группа *Telephone numbers*

Поскольку все пять параметров (рис. 8.18) имеют идентичную структуру, рассмотрим подробно один из них, например **Home**. Назначение всех полей приведено в табл. 8.10.

Каждому из этих полей соответствует тип данных "строка" (string). Для занесения списка телефонов, соответствующего одному и тому же полю, можно воспользоваться формой, запускаемой нажатием кнопки **Other** напротив каждого параметра. В результате формируется список. В Active Directory ему соответствует массив, элементы которого — строки. Имя массива формируется на основе соответствующего ему поля на вкладке **Telephones** путем добавления префикса `other`. Таким образом, параметр **Home** хранится в строковом поле `homePhone`, а дополнительные домашние телефоны, показываемые при нажатии на кнопку **Other**, хранятся в массиве `otherHomePhone`.

***Таблица 8.10.*** *Назначение полей вкладки* ***Telephones***

| Поле | Назначение |
|---|---|
| **Home** | Домашний телефон сотрудника |
| **Pager** | Номер пейджера сотрудника |
| **Mobile** | Номер мобильного телефона сотрудника |
| **Fax** | Номер телефона-факса |
| **IP phone** | Номер IP-телефона |

### *Notes*

Полю **Notes** соответствует параметр `info` в Active Directory. В этом поле можно указать любой комментарий, касающийся сотрудника. Обратите внимание, что тип данных этого параметра — строка, а не массив, элементы которого строки. Многострочность значения параметра `info` достигается переводом каретки на новую строку. На VBScript для этого используется константа `vbNewLine` или выражение `chr(13) & chr(10)`.

## Вкладка *Organization*

На вкладке **Organization** задаются параметры, характеризующие статус сотрудника в иерархии предприятия. Здесь указываются такие параметры, как

должность, подразделение, компания, начальник и т. д. На рис. 8.19 приведен внешний вид вкладки **Organization** с соответствующими именами полей в Active Directory. В табл. 8.11 дана необходимая дополнительная информация.

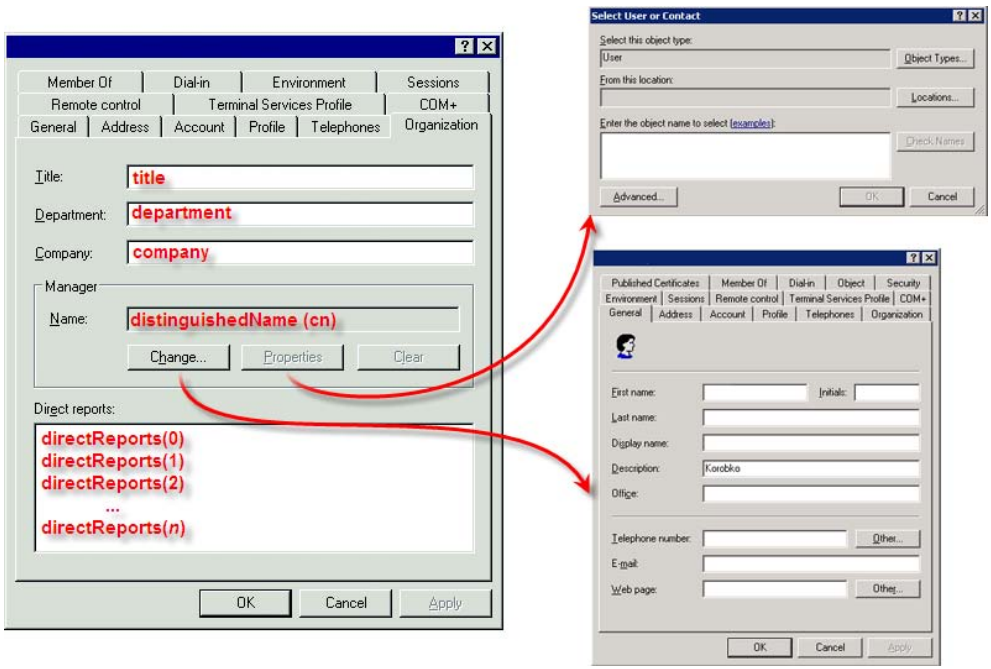



**Рис. 8.19.** Вкладка **Organization**

***Таблица 8.11.*** *Соответствие полей на вкладке* ***Organization*** *параметрам в Active Directory*

| **Поле на вкладке *Organization*** | **Тип** | **Поле в Active Directory** | **Тип** |
|---|---|---|---|
| **Title** | InputBox | `title` | String |
| **Department** | InputBox | `department` | String |
| **Company** | InputBox | `company` | String |
| **Manager: Name** | TextBox | `distinguishedName` (в Active Directory) `cn` (отображается в окне) | String |

***Таблица 8.11*** *(окончание)*

| Поле на вкладке *Organization* | Тип | Поле в Active Directory | Тип |
|---|---|---|---|
| **Direct reports** | TextArea (только чтение) | `directReports` | array |

## Описание полей

Рассмотрим подробно каждый из задаваемых параметров.

### *Title*

Значение параметра **Title** — должность пользователя согласно штатному расписанию предприятия. Этому полю в Active Directory соответствует одноименный текстовый параметр.

### *Department*

Значение этого поля — отдел, в котором работает сотрудник. В Active Directory параметру **Department** соответствует одноименный текстовый параметр. Максимальная длина строки — 255 символов.

### *Company*

Этот параметр обычно используется в очень больших компаниях. Сюда заносят название фирмы или ее филиала. Это актуально, если у фирмы разветвленная сеть филиалов. Полю **Company** в Active Directory соответствует одноименный текстовый параметр.

### Группа *Manager*

Блок **Manager** предназначен для определения руководителя сотрудника и состоит из неизменяемого поля **Name** и трех кнопок: **Change**, **Properties**, **Clear**. В текстовом поле **Name** отображается `cn` сотрудника, однако в Active Directory в соответствующем поле `manager` хранится полный путь к учетной записи пользователя в формате LDAP. Это ни что иное, как `distinguishedName` учетной записи пользователя, являющегося руководителем.

Из трех кнопок по умолчанию доступна только кнопка **Change**. С ее помощью администратор указывает учетную запись руководителя сотрудника. После того как указан руководитель, становятся доступными оставшиеся две кнопки: **Properties** и **Clear**. С помощью кнопки **Properties** вызывается стандартное диалоговое окно учетной записи руководителя, элементы которого подробно рассматриваются в этой главе. С помощью кнопки **Clear** удаляют значение поля `manager` и возвращаются к настройкам по умолчанию.

### *Direct reports*

Это неизменяемое поле, в котором приводится список прямых подчиненных, назначаемых в разделе **Manager** (рис. 8.20).

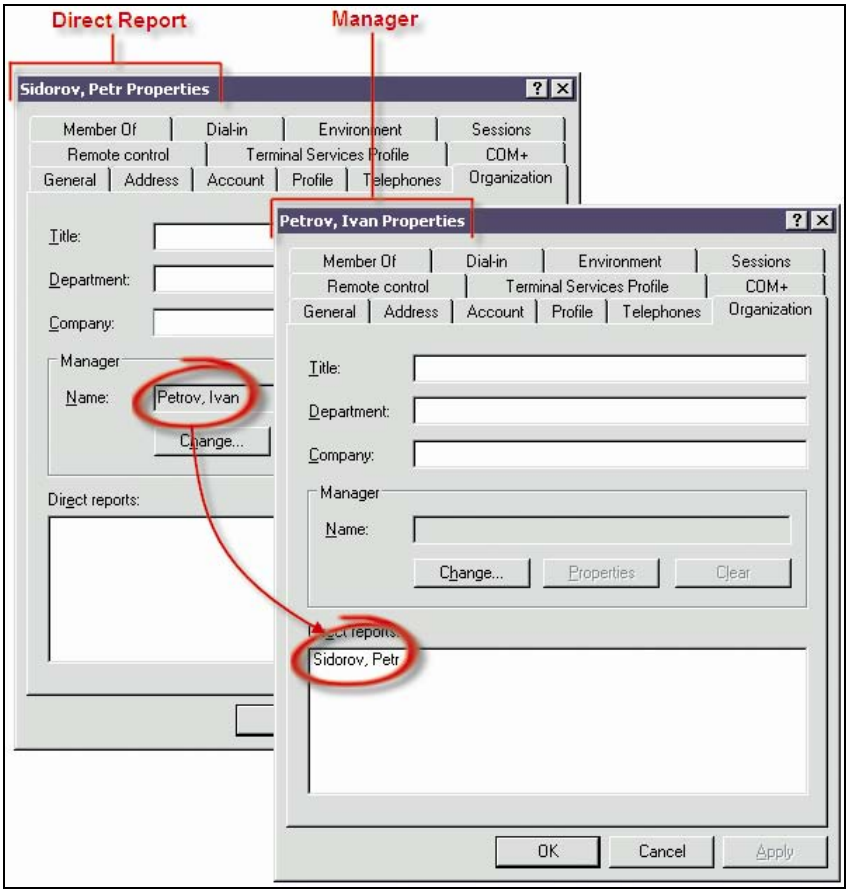



**Рис. 8.20.** Управление наполнением раздела **Direct reports**

В Active Directory этому полю соответствует параметр `directReports`, который представляет собой массив. Элементы этого массива — строки, каждая из которых является `distinguishedName` учетной записи подключенного пользователя. При отображении на вкладке оно преобразуется в `cn`.

## Вкладка *Member Of*

На вкладке **Member Of** формируется список групп (рис. 8.21), членом которых является текущий пользователь, а также назначается Primary group (основная группа). Список атрибутов Active Directory, соответствующих полям на вкладке **Member Of**, приведен в табл. 8.12.

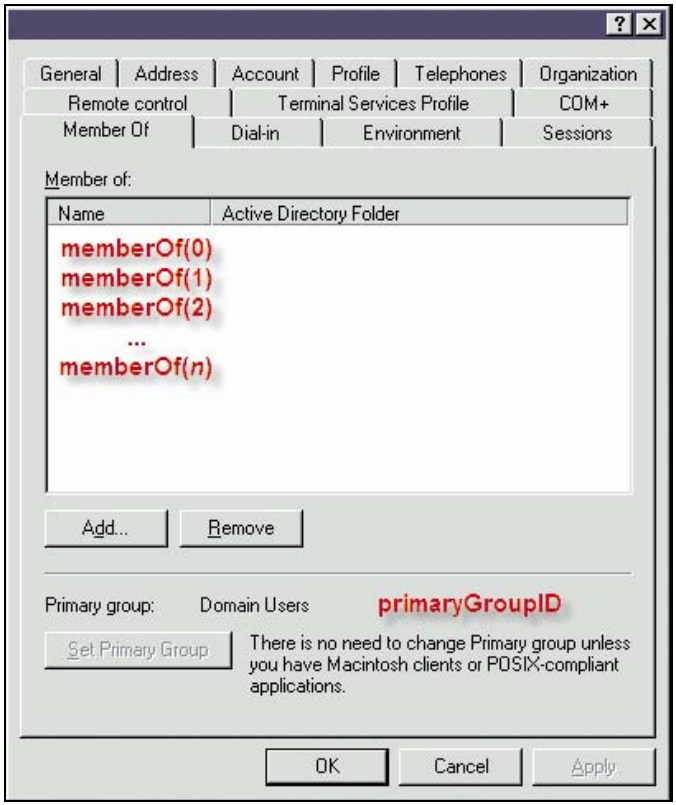



**Рис. 8.21.** Вкладка **Member Of**

***Таблица 8.12.*** *Соответствия полей на вкладке* ***Member Of*** *параметрам Active Directory*

| Поле на вкладке *Member Of* | Тип | Поле в Active Directory | Тип |
|---|---|---|---|
| **Member of** | ListBox, Button | `memberOf` | Array |
| **Primary group** | Button | `primaryGroupID` | String |

## Описание полей

Опишем приемы работы с полями на вкладке **Member Of** и смысл этих полей.

### *Member of*

Для управления членством пользователя в группах безопасности Active Directory используются две кнопки, находящиеся под списком групп: **Add** (Добавить) и **Remove** (Удалить). По умолчанию пользователь входит в группу Domain Users. Эта группа не отображается в списке.

Механизм управления следующий. Для добавления пользователя в какую-либо группу необходимо нажать кнопку **Add**. В появившемся диалоговом окне (рис. 8.22) осуществляется поиск объектов по заданным критериям. В поле **Enter the object names to select** указывается одно из имен пользователя (`cn` или `sAMAccountName`), при этом фиксируется значение поля `distinguishedName`, в то время как в списке отображается `cn` этого объекта.

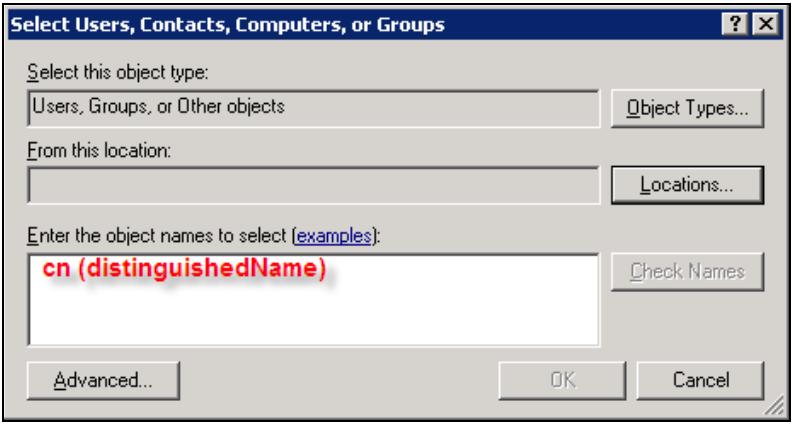



**Рис. 8.22.** Поиск объектов в Active Directory по заданному критерию

### *Primary Group*

Единственная группа, членом которой является пользователь после создания его учетной записи — группа Domain Users, и она же является основной (Primary Group). Значение соответствующего параметра `primaryGroupID` — идентификационный номер группы. Для группы Domain Users параметр `primaryGroupID` равен 513.

Рассмотрим пример. Пусть пользователь Test_User входит в группу Test_Group, и SID этой группы — `S-1-5-21-42226584364-21557989-1436132917-12213`. Необходимо определить значение параметра `primaryGroupID`, если основная группа — Test_Group. Решение просто: последний раздел SID и является идентификатором основной группы, поэтому параметр `PrimaryGroupID = 12213` (рис. 8.23).

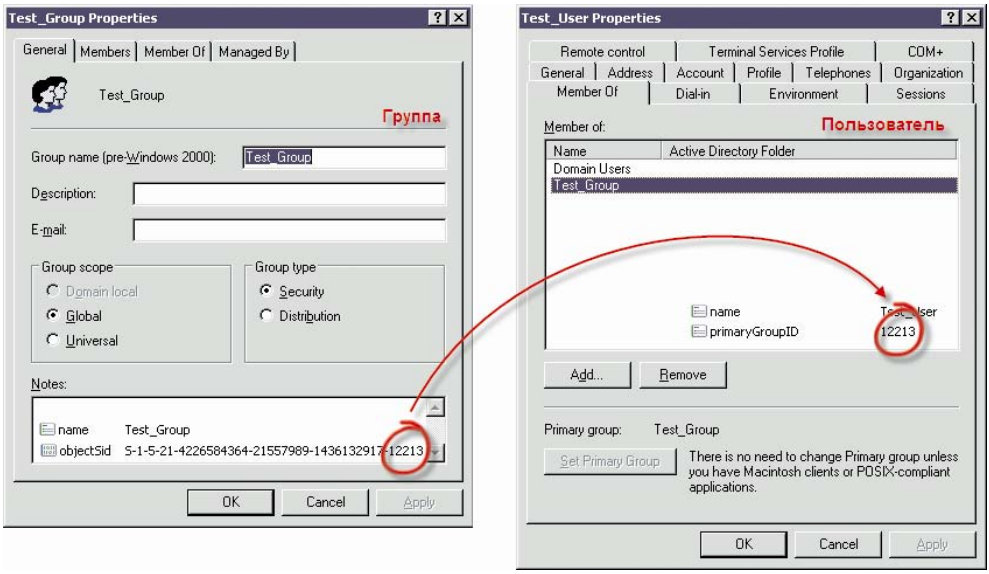



**Рис. 8.23.** Определение значения параметра `primaryGroupID`

## Вкладка *Dial-in*

На вкладке **Dial-in** сосредоточены настройки Dial-up-соединения учетной записи пользователя (рис. 8.24).

В табл. 8.13 показано соответствие полей вкладки атрибутам Active Directory.

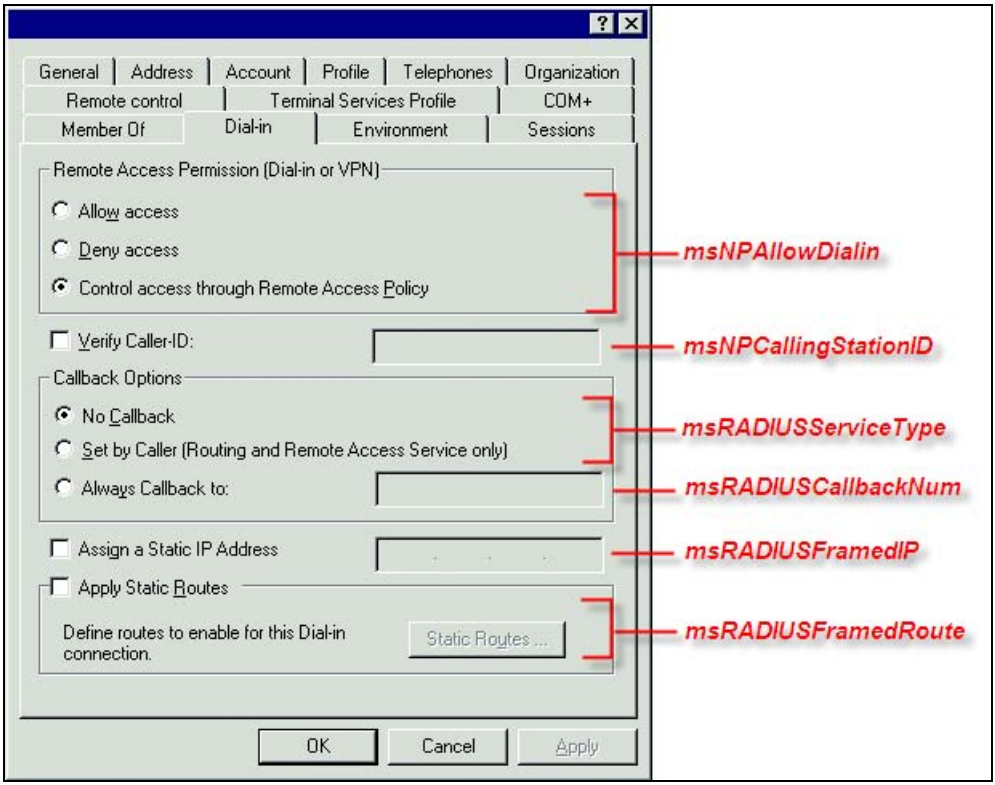



**Рис. 8.24.** Вкладка **Dial-in**

***Таблица 8.13.*** *Соответствия полей на вкладке* ***Dial-in*** *параметрам Active Directory*

| **Поле на вкладке *Dial-in*** | **Тип** | **Поле в Active Directory** | **Тип** |
|---|---|---|---|
| **Allow Access** | Radio | `msNPAllowDialin` | Boolean |
| **Deny Access** | | | |
| **Control access through Remote Access Policy** | | | |

***Таблица 8.13*** *(окончание)*

| Поле на вкладке *Dial-in* | Тип | Поле в Active Directory | Тип |
|---|---|---|---|
| **Verify Caller-ID** | CheckBox | msNPCallingStationID | String |
| **No Callback** | Radio | msRADIUSServiceType | String |
| **Set by Caller** | | | |
| **Always Callback to** | | msRADIUSCallbackNum | String |
| **Assign Static IP Address** | CheckBox | msRADIUSFramedIP | String |
| **Apply Static Routes** | CheckBox | msRADIUSFramedRoute | String |

## Описание полей

Опишем присутствующие на вкладке поля подробнее.

### Группа *Remote Access Permission (Dial-in or VPN)*

С помощью группы параметров **Remote Access Permission (Dial-in or VPN)** осуществляется настройка контроля за удаленным соединением пользователя с сетью. Переключатель может быть установлен в одно из следующих положений:

- вариант **Allow Access** дает возможность пользователю беспрепятственно подключаться к сети с помощью соединения Dial-up или VPN. При этом параметру `msNPAllowDialin` в Active Directory присвоено значение `True`;
- позиция **Deny Access** запрещает пользователю удаленно подключаться к сети при любых условиях. Значение параметра `msNPAllowDialin` в этом случае равно `False`;
- выбор **Control access through Remote Access Policy** дает возможность регулировать настройки удаленного доступа пользователя к сети с помощью групповой политики Remote Access Policy. При этом программно назначают `msNPAllowDialin = " "`.

По умолчанию выбран вариант **Control access through Remote Access Policy**.

### *Verify Caller-ID*

Включив возможность проверки Caller-ID звонящего пользователя, можно создать дополнительную защиту от несанкционированного вмешательства. По умолчанию опция **Verify Caller-ID** отключена. В Active Directory значение этого параметра хранится в текстовом виде в переменной `msNPCallingStationID`.

### Группа *CallBack Options*

Одной из дополнительных мер по защите от несанкционированного доступа к сети является возможность автоматического перезвона. Таким образом, при попытке пользователя подключиться к сети осуществляется обрыв связи. После этого система сама пытается установить соединение с удаленным компьютером. Если какой-либо способ обратной связи активирован, то злоумышленник не сможет подключиться к сети от имени этого пользователя, поскольку кроме имени и пароля задействован третий параметр — телефон. Группа **CallBack Options** позволяет выбрать один из следующих вариантов:

- пункт **No Callback** отключает возможность перезванивания, и подключение идет исключительно со стороны пользователя. Этот параметр выбран по умолчанию. В Active Directory необходимо, чтобы параметры `msRADIUSServiceType` и `msRADIUSCallbackNum` были обнулены;
- опция **Set by Caller (Routing and Remote Access Service only)** включает режим, в котором подключающийся пользователь в настройках своего удаленного соединения должен выбрать способ подключения: с включенной функцией Callback или нет. При активации этой функции пользователь должен указать номер телефона, по которому модем свяжется с его компьютером.

  При выборе этой опции в Active Directory необходимо обнулить параметр `msRADIUSCallbackNum`, а параметру `msRADIUSServiceType` присвоить значение 4;
- вариант **Always Callback to** заставляет систему при попытке подключения перезванивать удаленному пользователю по указанному в следующем за переключателем поле телефонному номеру. В Active Directory этот телефонный номер хранится в атрибуте `msRADIUSCallbackNum`. Для включения режима автоматического перезванивания значение параметра `msRADIUSServiceType` должно быть обнулено.

### Assign Static IP Address

В момент подключения удаленного пользователя служба DHCP динамически выделяет IP-адрес. Если по каким-либо причинам должен быть указан статический адрес подключения, то необходимо поставить галочку **Assign Static IP Address** и ввести IP-адрес в сделавшееся доступным поле. В Active Directory ему соответствует поле `msRADIUSFramedIP`, значение которого имеет вид: `***.***.***.***`. По умолчанию статический адрес не назначается.

### Apply Static Routers

Для подключения пользователя из другой подсети необходимо произвести маршрутизацию.

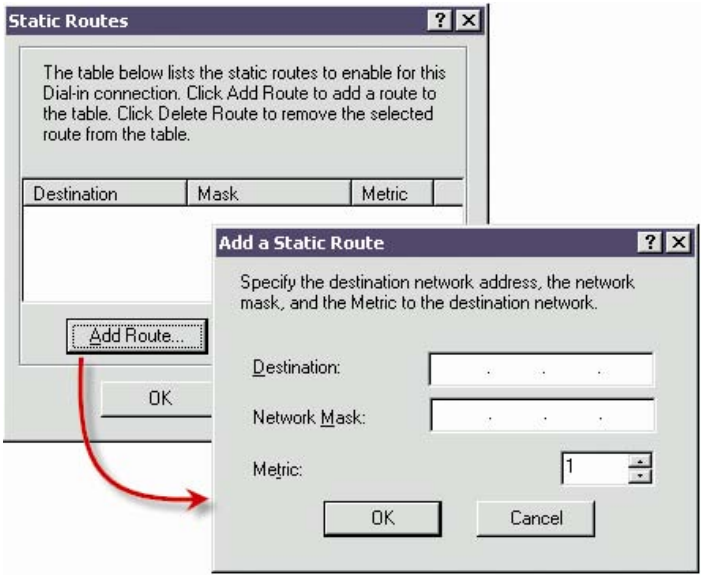



**Рис. 8.25.** Параметры статической маршрутизации

Для этого необходимо активировать группу параметров **Apply Static Routers** и, нажав на кнопку **Static Route** (Статическая маршрутизация), в появившемся окне (рис. 8.25) ввести необходимые значения. Полученные данные сохраняются в поле `msRADIUSFramedRoute`. Элементы массива `msRADIUSFramedRoute` представляют собой строки, сформированные по шаблону:

`ooo.ooo.ooo.ooo/xxx.xxx.xxx.xxx`

Здесь ooo.ooo.ooo.ooo — подсеть, к которой необходимо преобразовать, xxx.xxx.xxx.xxx — преобразуемая подсеть.

По умолчанию статическая маршрутизация отключена.

### Резервная копия параметров вкладки *Dial-in*

Все настройки, перечисленные на вкладке **Dial-in**, имеют резервную копию для восстановления, которая также хранится в Active Directory. Каждому полю соответствует дополнительный параметр, имя которого образовано добавлением в него слова Saves и заменой RADIUS на RAS (табл. 8.14).

***Таблица 8.14.*** *Соответствия параметров вкладки Dial-In их резервным копиям*

| Значение параметра | Резервная копия |
|---|---|
| msNPAllowDialin | — |
| msNPCallingStationID | msNPSavesCallingStationID |
| msRADIUSServiceType | — |
| msRADIUSCallbackNum | msRASSavesCallbackNum |
| msRADIUSFramedIP | msRASSavesFramedIP |
| msRADIUSFramedRoute | msRASSavesFramedRoute |

При изменении какого-либо параметра сначала осуществляется копирование имеющегося значения в соответствующий ему параметр для сохранения, и лишь затем назначается новое значение.

## Вкладка *Environment*

На вкладке **Environment**[1] (рис. 8.26) осуществляется настройка входа пользователя в терминальную сессию, то есть конфигурация сценария входа.

[1] Для реализации программного управления настройками терминальной сессии необходимо установить Windows Server 2003 Administrative Tools Pack на компьютер, с которого запускается сценарий. Этот пакет можно бесплатно загрузить с сайта компании Microsoft **http://download.microsoft.com/download/c/7/5/c750f1af-8940-44b6-b9eb-d74014e552cd/adminpak.exe**.

Отметим, что терминальные сессии для обычных сотрудников настраиваются достаточно редко. В табл. 8.15 показано соответствие полей на вкладке **Environment** атрибутам в каталоге Active Directory.

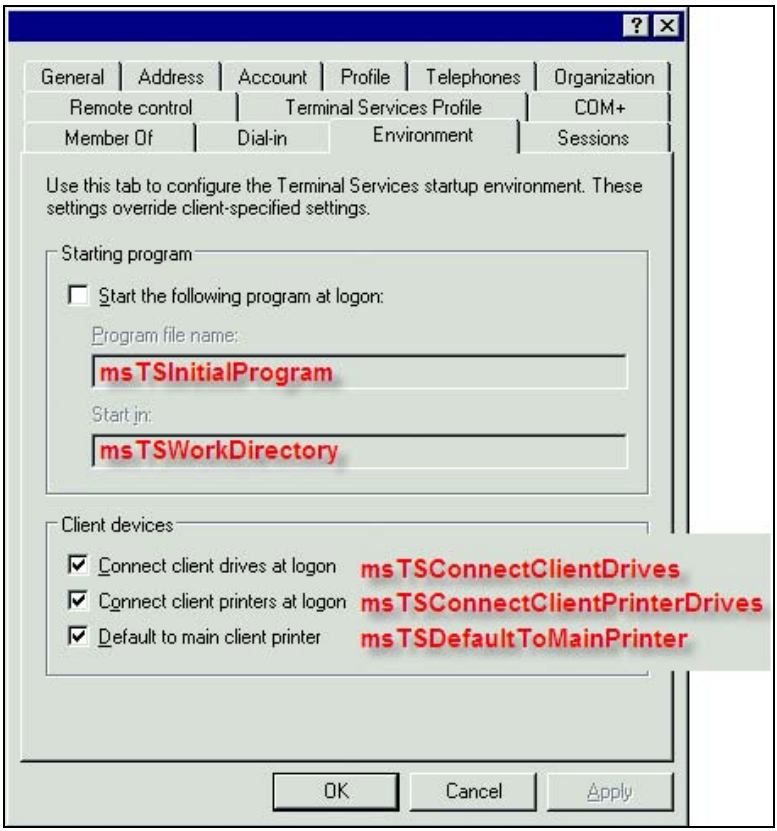



**Рис. 8.26.** Вкладка **Environment**

***Таблица 8.15.*** *Соответствие полей на вкладке* ***Environment*** *атрибутам в Active Directory*

| **Поле на вкладке *Environment*** | **Тип** | **Параметр в Active Directory** | **Тип** |
|---|---|---|---|
| **Program file name** | InputBox | `msTSInitialProgram` | String |

***Таблица 8.15*** *(окончание)*

| Поле на вкладке *Environment* | Тип | Параметр в Active Directory | Тип |
|---|---|---|---|
| **Start in** | InputBox | `msTSWorkDirectory` | String |
| **Connect client drives at logon** | CheckBox | `msTSConnectClientDrives` | Boolean |
| **Connect client printers at logon** | CheckBox | `msTSConnectClientPrinterDrives` | Boolean |
| **Default to main client printer** | CheckBox | `msTSDefaultToMainPrinter` | Boolean |

## Описание полей

Поля, присутствующие на вкладке, требуют дополнительного описания.

### Группа *Starting program*

В группе параметров **Starting program** указывается сценарий, который будет выполняться при входе в терминальную сессию. Первый параметр-флажок предназначен для активации сценария загрузки, остальные два указывают имя и местоположение сценария.

Флажок **Start the following program at logon** отвечает за использование сценария загрузки при входе в терминальную сессию. Активация флажка делает доступными изменения в следующих полях ввода:

- поле **Program file name** определяет имя файла со сценарием регистрации, который будет запущен при входе пользователя в терминальную сессию. В Active Directory этому параметру соответствует строковый параметр `msTSInitialProgram`;
- поле **Start in** указывает месторасположение запускаемого сценария. В Active Directory этому параметру соответствует строковый параметр `msTSWorkDirectory`.

### Группа *Client devices*

С помощью группы параметров **Client devices** осуществляется управление дисками и принтерами, подключаемыми при создании терминальной сессии.

В этой группе находятся три параметра: один из них управляет подключением дисков, два оставшихся — принтеров. Все параметры в этой группе принимают только булево значение (`True`/`False`) и включены по умолчанию (`True`).

- Если флажок **Connect client drives at logon** отмечен, то при входе в терминальную сессию монтируются те же диски, что и в обычной сессии пользователя. Соответствующий атрибут `msTSConnectClientDrives` в Active Directory по умолчанию равен `True`.
- Если флажок **Connect client printers at logon** помечен, то при входе в терминальную сессию осуществляется подключение тех же принтеров, что и в нормальной сессии этого пользователя. Этому параметру в Active Directory соответствует параметр `msTSConnectClientPrinterDrives`, принимающий булево значение.
- Если отмечен флажок **Default to main client printer**, то после загрузки и подключения принтеров принтер по умолчанию назначается тот же, что и в нормальной сессии. В Active Directory этому параметру соответствует атрибут `msTSDefaultToMainPrinter`.

## Вкладка *Sessions*

На вкладке **Sessions** (рис. 8.27) осуществляется настройка правил работы сессий, а именно различных временных показателей.

В табл. 8.16 приведено соответствие полей вкладки **Sessions** атрибутам в каталоге Active Directory.

***Таблица 8.16.*** *Соответствие полей на вкладке* ***Sessions*** *атрибутам Active Directory*

| **Поле на вкладке *Sessions*** | **Тип** | **Поле в Active Directory** | **Тип** |
|---|---|---|---|
| **End a disconnected session** | ComboBox | `msTSMaxDisconnectionTime` | String |
| **Active session limit** | ComboBox | `msTSMaxConnectionTime` | String |

***Таблица 8.16*** *(окончание)*

| **Поле на вкладке *Sessions*** | **Тип** | **Поле в Active Directory** | **Тип** |
|---|---|---|---|
| **Idle session limit** | ComboBox | `msTSMaxIdleTime` | String |
| **When a session limit is reached or connection is broken** | Radio | `msTSBrokenConnectionAction` | Boolean |
| **Allow reconnection** | Radio | `msTSReconnectionAction` | Boolean |

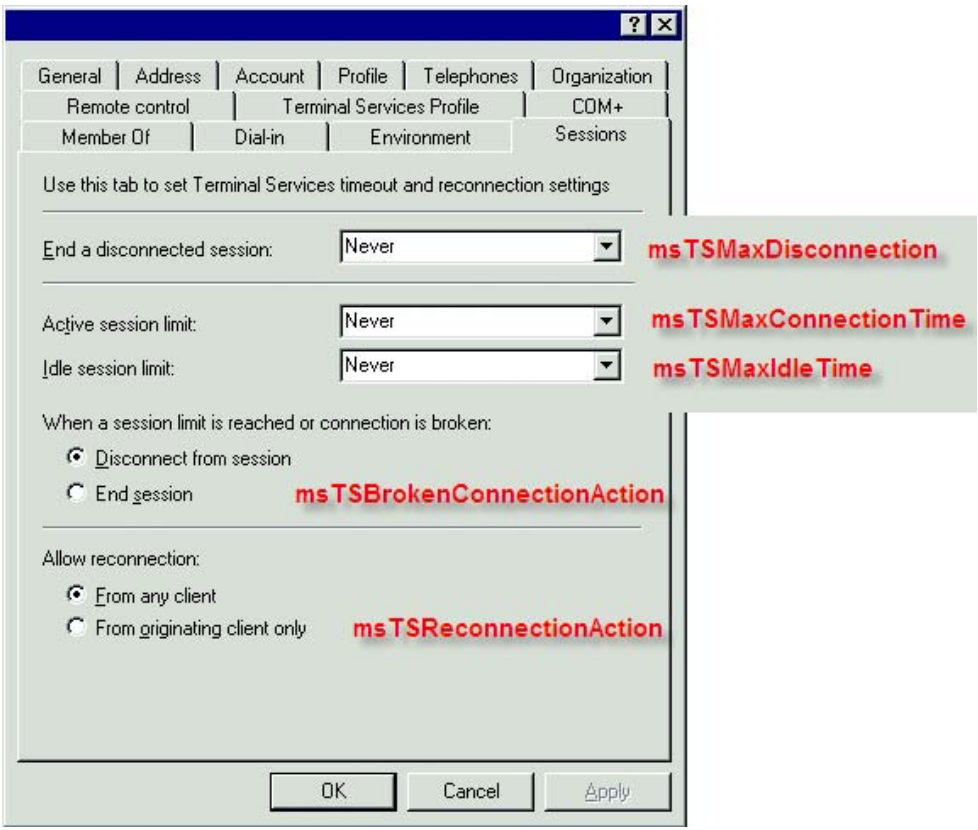



**Рис. 8.27.** Вкладка **Sessions**

## Описание полей

Приведем более подробное описание элементов управления, присутствующих на вкладке **Sessions**.

### *End a disconnected session*

При закрытии окна терминальной сессии с помощью соответствующей кнопки в правом верхнем углу окна отключение от нее не происходит. Про такую сессию обычно говорят, что она "висит". Корректное завершение терминальной сессии — завершение сеанса работы ОС с помощью пункта **LogOn/LogOff** меню **Пуск**. "Висящая" сессия иногда бывает полезна, однако в большинстве случаев это отрицательно сказывается на возможностях других пользователей подключаться к серверу с помощью терминальной сессии. Чтобы обеспечить завершение сессии при некорректном выходе из нее, в поле **End a disconnected session** выбирают одно из предлагаемых в списке значений (рис. 8.28). Следует иметь в виду, что при плохой связи сервер может принять сессию за "висящую", когда она на самом деле подключена, так что очень маленькое значение этого параметра может приводить к разрывам связи.

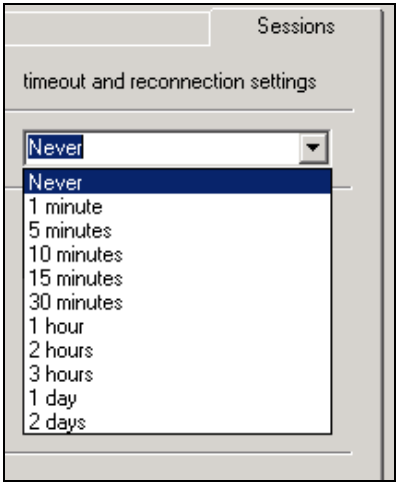



**Рис. 8.28.** Возможные значения временных интервалов на вкладке **Sessions**

По умолчанию значение не задано. Указанный временной интервал фиксируется в атрибуте Active Directory `msTSMaxDisconnectionTime`.

### Active session limit

С помощью параметра **Active session limit** устанавливают так называемое время жизни сессии. По истечению указанного срока терминальная сессия закрывается с соответствующим оповещением. По умолчанию этот срок не задан, что соответствует бесконечной сессии. В Active Directory параметру **Active session limit** соответствует поле `msTSMaxConnectionTime`. Диапазон допустимых значений такой же, как у поля **End a disconnected session** (рис. 8.28).

### Idle session limit

Если необходимо закрывать терминальную сессию после перехода компьютера в спящий режим на время, большее указанного, измените значение параметра **Idle session limit** на требуемое. Список возможных значений приведен на рис. 8.28. В Active Directory параметру **Idle session limit** соответствует строковое поле `msTSMaxIdleTime`.

### When session limit is reached or connection is broken

С помощью параметра `msTSBrokenConnectionAction` в Active Directory определяется поведение сессии, когда закрыто соединение или превышено ограничение сессий. По умолчанию выбран вариант **Diconnected from session**, которому соответствует булево значение `True` параметра `msTSBrokenConnectionAction`. Варианту **End session** на вкладке `Sessions` соответствует значение `False` атрибута Active Directory.

### Allow reconnection

Параметр `msTSReconnectionAction` в Active Directory определяет поведение сессии при попытке какого-либо пользователя войти в нее вновь при условии, что она была закрыта, но еще не отключена. По умолчанию установлено **From any client** (с любого клиента), которому соответствует значение `True` параметра `msTSReconnectionAction`. Варианту **From originating client only** (только с клиента, с которого была создана сессия) соответствует значение `False`.

## Вкладка *Remote control*

Управление удаленным подключением терминальной сессии осуществляется на вкладке **Remote control** (рис. 8.29).

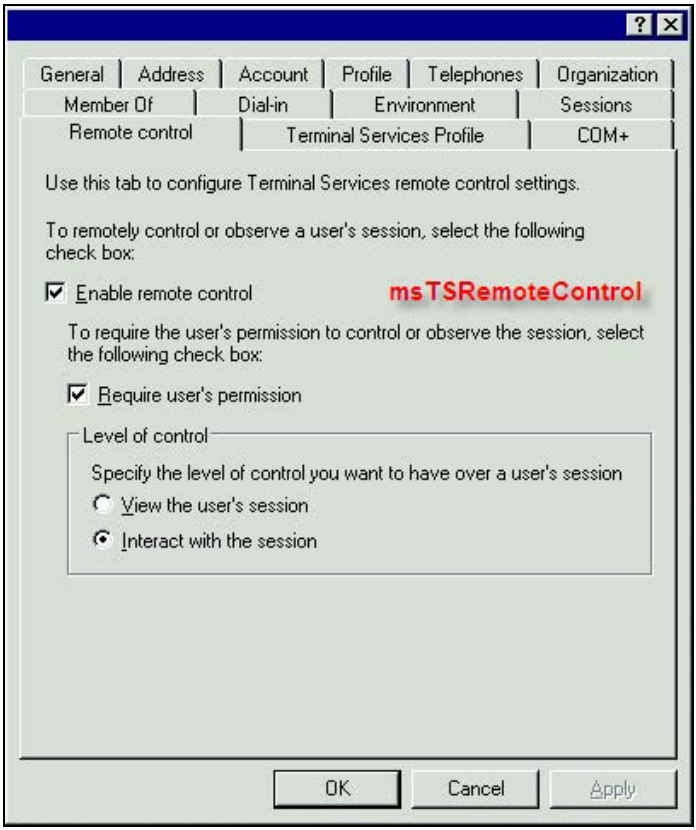



**Рис. 8.29.** Вкладка **Remote control**

Все параметры на вкладке устанавливаются с помощью флажков и переключателя, при этом в Active Directory изменяется только один параметр — `msTSRemoteControl`. По умолчанию значение этого параметра — 1.

В табл. 8.17 приведен список возможных значений параметра `msTSRemoteControl` и комментарии.

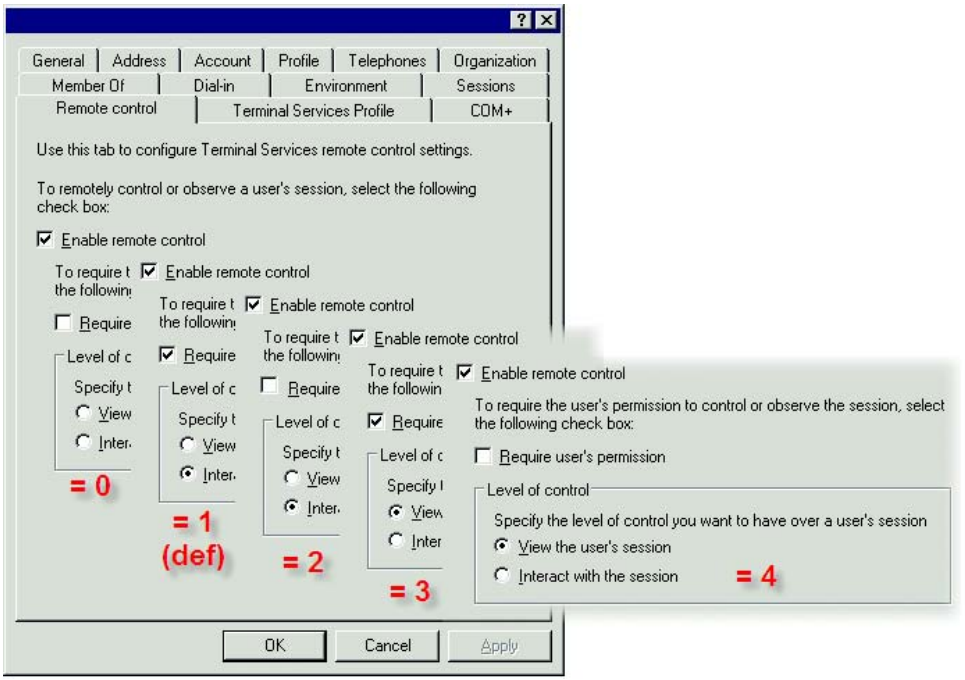



**Рис. 8.30.** Иллюстрация возможных значений параметров на вкладке **Remote control**

***Таблица 8.17.*** *Значения параметра* `msTSRemoteControl`

| **Значение** `msTSRemoteControl` | **Тип** |
|---|---|
| 0 | Удаленное управление выключено. Параметр **Enable Remote Control** не активен. Остальные параметры недоступны для управления |
| 1 (по умолчанию) | Включены следующие возможности удаленного управления терминальной сессией:<br>• **Require user's permission** (включена идентификация пользователя);<br>• **Interact with the session** (сессия интерактивна) |
| 2 | Сессия работает под именем anonymous. Уровень контроля сессии — интерактивный (**Interact with the session**) |

***Таблица 8.17*** *(окончание)*

| **Значение** `msTSRemoteControl` | **Тип** |
|---|---|
| 3 | Уровень контроля — **View the user's session** (просмотр сессии пользователя). Для обеспечения этого уровня безопасности необходимо, чтобы пользователь был идентифицирован, поэтому опция **Require user's permission** активна |
| 4 | Уровень контроля — **View the user's session** (просмотр сессии пользователя), а опция **Require user's permission** не активна. Пользователь может просматривать, но не может изменять сессию |

На рис. 8.30 показаны все возможные комбинации значений, указанные в табл. 8.17.

## Вкладка *Terminal Services Profile*

Управление профилями терминальной сессии осуществляется на вкладке **Terminal Services Profile**. Наполнение этой вкладки очень похоже на содержание вкладки **Profile**, однако есть и отличия.

В табл. 8.18 указаны соответствия полей вкладки **Terminal Services Profile** атрибутам в Active Directory.

***Таблица 8.18.*** *Соответствие полей на вкладке* ***Terminal Services Profile*** *параметрам Active Directory*

| **Поле на вкладке** ***Terminal Services Profile*** | **Тип** | **Параметр в Active Directory** | **Тип** |
|---|---|---|---|
| **Profile Path** | InputBox | `msTSProfilePath` | String |
| **Local path** | InputBox | `msTSHomeDirectory` | String |

***Таблица 8.18*** *(окончание)*

| **Поле на вкладке *Terminal Services Profile*** | **Тип** | **Параметр в Active Directory** | **Тип** |
|---|---|---|---|
| **Connect** | CheckBox | `msTSHomeDrive` | String |
| | InputBox | `msTSHomeDirectory` | String |
| **Deny this user permissions to log on to any Terminal Server** | CheckBox | `msTSAllowLogon` | Boolean |

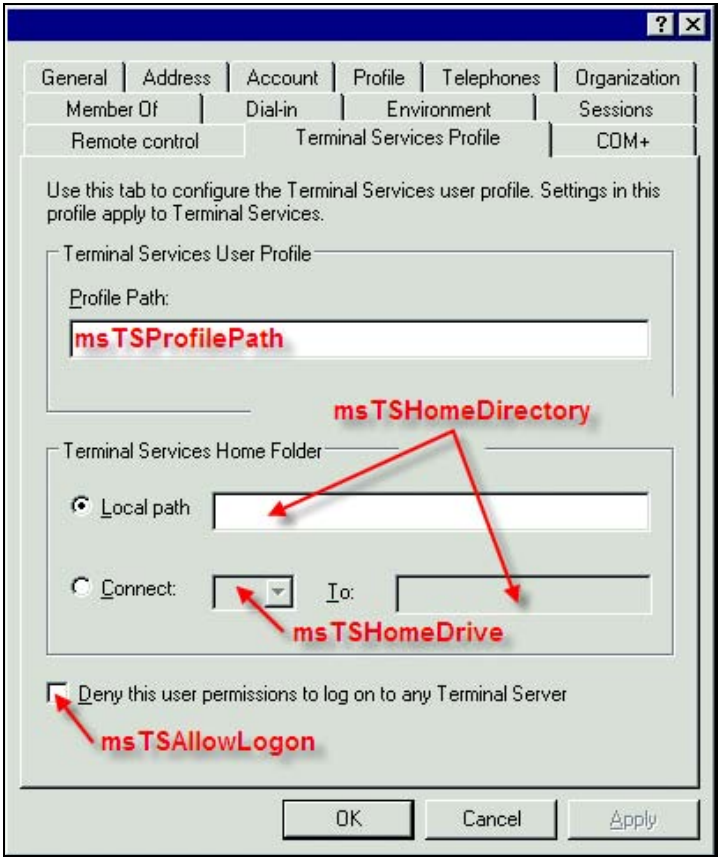



**Рис. 8.31.** Вкладка **Terminal Services Profile**

## Описание полей

Опишем подробнее присутствующие на вкладке поля.

### Группа *Terminal Service User Profile*

На вкладке **Terminal Service Profile** в группе параметров **Terminal Service User Profile**, в отличие от аналогичной группы на вкладке **Profile**, присутствует всего лишь один параметр. Второй параметр, находившийся ранее на вкладке **Profile**, перенесен на вкладку **Environment**.

#### *Profile Path*

Параметру **Profile Path** в Active Directory соответствует `msTSProfilePath`, который имеет формат строки. В качестве значения указывают путь к перемещаемому профилю в виде UCN-пути: \\Server\ShareName\UserNameFolder, где ShareName — имя опубликованной в сети папки на сервере Server. В качестве UserNameFolder обычно используют переменную `%username%` (см. *приложение 4*), которая автоматически преобразуется в сокращенное имя пользователя, используемое для входа в сеть. В Active Directory переменной `%username%` соответствует значение параметра `sAMAccountName`.

### Группа *Terminal Service Home Folder*

Группа параметров **Terminal Service User Profile** полностью идентична группе **Home folder** на вкладке **Profile**. В этой группе задается путь к домашнему каталогу.

Домашний каталог может быть задан двумя способами:

- как локальный путь, например, C:\Storage\Profiles\PIvanov;
- как сетевой диск, который будет монтироваться каждому сотруднику после регистрации в сети.

#### *Local Path*

Если домашний каталог должен храниться локально на рабочей станции, то необходимо указать локальный путь, которому в Active Directory соответствует атрибут `msTSHomeDirectory`, при этом значение атрибута `msTSHomeDrive=" "`. Формат пути такой: C:\FolderName. После регистрации

пользователя в сети указанная папка будет создана локально на компьютере пользователя.

### *Connect*

Для того чтобы пользователю после регистрации компьютера в сети монтировалась папка с домашним каталогом на сетевой диск с указанным именем, необходимо перейти в режим **Connect**. В отличие от предыдущего режима, необходимо указать два параметра — имя диска из списка, которому в Active Directory соответствует текстовое поле `msTSHomeDrive`, и UNC-путь к подключаемой папке. Этому пути соответствует строковый параметр `msTSHomeDirectory`. После того как задан UNC-путь, осуществляется его проверка как на соответствие формату \\Server\ShareName\FolderName, так и на наличие папки, к которой предоставлен сетевой доступ.

### *Deny this user permissions to log on to any Terminal Server*

На вкладку **Terminal Services Profile** добавлен еще один параметр, который позволяет запретить пользователю удаленно входить на сервер.

Значение **Deny this user permissions to log on to any Terminal Server** в Active Directory сохраняется в булевом атрибуте `msTSAllowLogon`. По умолчанию эта опция включена, а параметру `msTSAllowLogon` присвоено значение `False`.

## Вкладка *COM+*

Вкладка **COM+** не рассматривается, поскольку для управления параметрами, присутствующими на этой вкладке, необходимо использовать расширенный режим работы Active Directory.

Глава 9



# Управление учетной записью группы безопасности

В этой главе описаны принципы управления учетной записью группы безопасности с помощью мастера и сценария. Приведены сопоставления названий полей мастера и атрибутов Active Directory, описаны нюансы управления объектом.

## Основные термины и понятия

- **Security group** — учетная запись, содержащая в себе другие учетные записи: групп или пользователей. Права и привилегии, предоставленные группе, автоматически распространяются на ее членов. Группы безопасности используется для определения разрешений доступа к файлам и другим ресурсам.
- **Group scope** — область действия группы, которая определяет, каким образом может быть использована группа. Существуют две области действия групп: локальная, действующая в пределах локального компьютера, и глобальная, область действия которой — домен;
- **Тип группы** — параметр, определяющий, является ли группа группой рассылки или же группой безопасности. Существуют два типа групп: Security (безопасности) и Distribution (распространения). В настоящее время, в связи с переходом на домены Windows 2k, понятие Distribution нивелировалось.

## Производимые операции

На основе членства объектов в группах построена система разделения прав доступа к различным ресурсам. Управление учетными записями групп —

первоочередная задача системного администратора. Для автоматизации данного процесса необходимо четко представлять себе взаимосвязь полей мастера и атрибутов каталога Active Directory.

# Создание группы

Рассмотрим работу мастера создания группы безопасности с точки зрения изменений, происходящих в Active Directory. Во время его работы системным администратором назначаются имя и тип группы. Остальные параметры группы можно изменить после окончания работы мастера.

Запуск мастера осуществляется из MMC-консоли **Active Directory Users and Computers** выбором пункта **New | Group** из контекстного меню папки (рис. 9.1), в которой необходимо создать учетную запись. Таким образом определяется местоположение объекта.

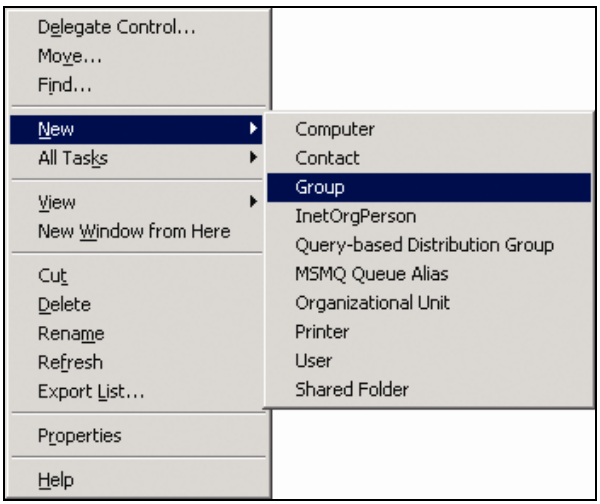



**Рис. 9.1.** Запуск мастера **Создание новой группы безопасности**

# Работа мастера. Теория

Как ни странно, работа мастера состоит всего из одного шага (рис. 9.2). Во время создания группы безопасности необходимо назначить имя и тип груп-

пы, определить область действия и имя группы для совместимости с доменом Windows NT (табл. 9.1).

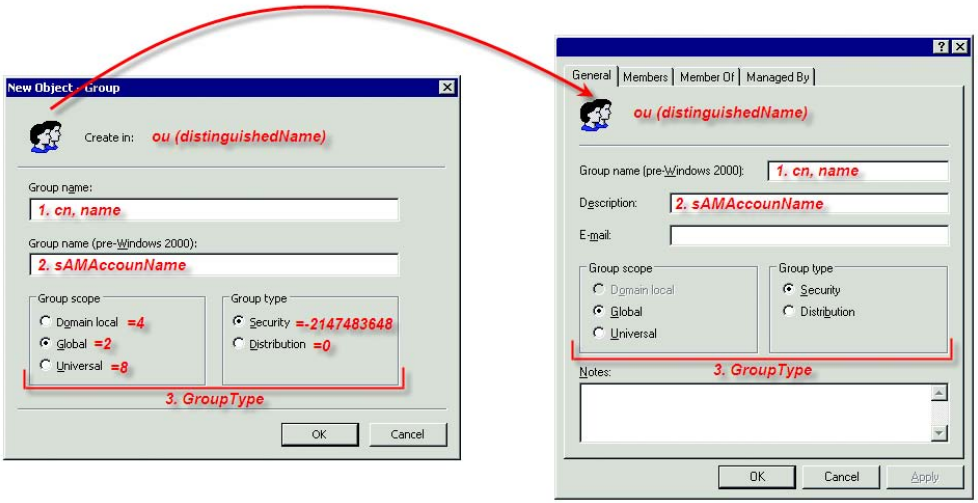



**Рис. 9.2.** Работа мастера **Создание новой группы безопасности**

***Таблица 9.1.*** *Параметры учетной записи группы*

| Параметр мастера | Поле в Active Directory | Тип данных | Описание |
|---|---|---|---|
| **Group name** | `cn`, `name` | String | Название группы |
| **Group name (pre-Windows 2000)** | `sAMAccountname` | String | Название группы для совместимости с доменом Windows NT |
| **Group scope** | `groupType` | Radio Button | Настройка области действия и типа группы |

## *Create in*

Во время создания учетной записи группы местоположение объекта (отображаемое в неизменяемом поле **Create in**) выбирает администратор. Таким образом, значение параметра `distinguishedName` задается неявным образом.

В свойствах группы отображается имя папки — преобразованное значение параметра `distinguishedName`.

### *Group name*

**Group name** — один из обязательных параметров: имя группы, используемое в доменах Windows 2k. В Active Directory этому параметру соответствует поле `cn`. Значение параметра `name` назначается автоматически в соответствии со значением `cn`.

Впоследствии значение этого параметра можно изменить, войдя в свойства созданной группы на вкладке **General** (рис. 9.2).

### *Group name (pre-Windows 2000)*

С помощью параметра **Group name (pre-Windows 2000)** назначается имя группы, необходимое для совместимости с доменами на основе Windows NT. В Active Directory этому параметру соответствует поле `sAMAccountName`. При назначении параметра **Group name** значение **Group name (pre-Windows 2000)** заполняется автоматически, так что по умолчанию значения параметров совпадают. При необходимости они могут быть изменены.

### Группы *Group scope* и *Group type*

Этим группам соответствует один и тот же параметр в Active Directory — `groupType`. Каждой выбранной настройке обоих групп сопоставлено индивидуальное значение. В каталоге хранится сумма значений, полученных из обоих разделов (табл. 9.2). По умолчанию `groupType = -2147483646`, что соответствует в группе **Group Scope** значению **Global**, а в **Group Type** — **Security**.

***Таблица 9.2.*** *Расшифровка значений параметра* `groupType`

| Тип группы | Значение | Тип группы | Значение |
|---|---|---|---|
| Global Security Group | -2147483646 | Global Distribution Group | 2 |
| Local Security Group | -2147483644 | Local Distribution Group | 4 |

***Таблица 9.2*** *(окончание)*

| Тип группы | Значение | Тип группы | Значение |
|---|---|---|---|
| BuiltIn Group | –2147483643 | Universal Distribution Group | 8 |
| Universal Security Group | –2147483640 | | |

## Работа мастера. Практика

Создадим учетную запись группы с помощью мастера. Исходные данные приведены в табл. 9.3.

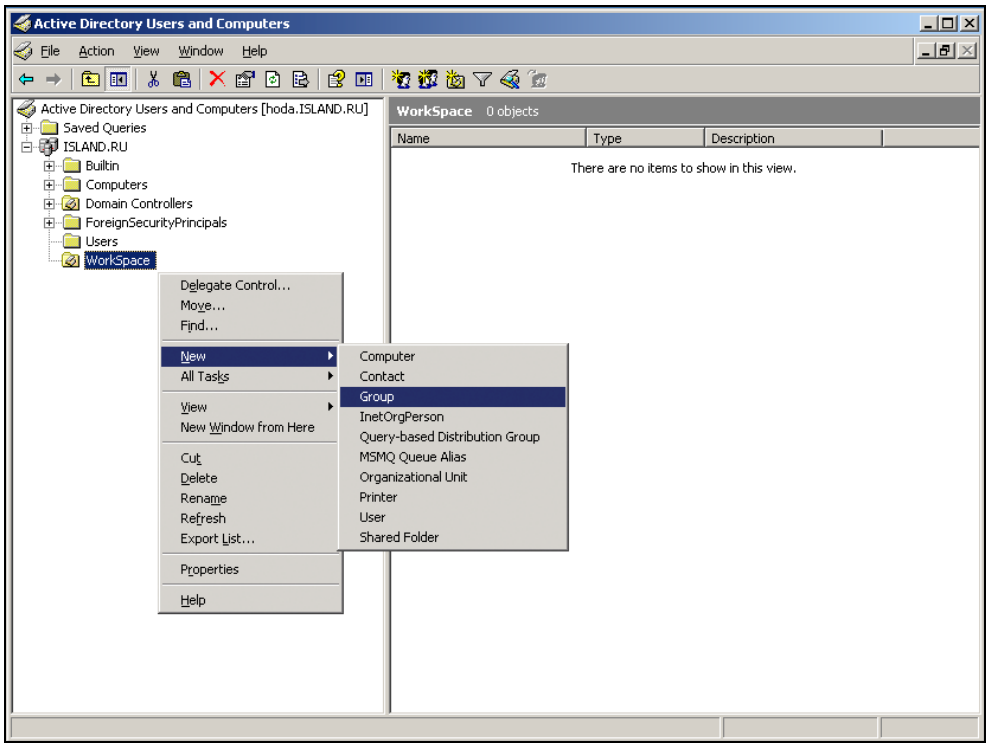



**Рис. 9.3.** Запуск мастера создания учетной записи группы

***Таблица 9.3.*** *Исходные данные для создания учетной записи группы*

| Поля мастера | Значение | Описание |
|---|---|---|
| **Create In** | ISLAND.RU\WORKSPACE | Учетная запись группы будет создана в контейнере Work-Space, находящемся в корневом каталоге домена ISLAND.RU |
| **Group name** | Test | Название группы в сети |
| **Group name (pre-Windows 2000)** | | Название группы в сети для совместимости с доменами Windows NT |
| **Group scope** | Global | Тип группы |
| **Group type** | Security | |

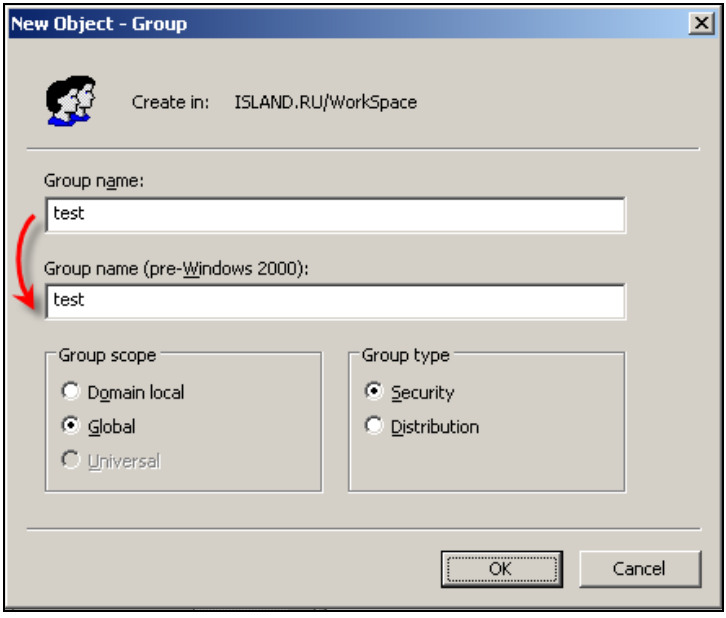



**Рис. 9.4.** Работа мастера

Рассмотрим пошагово создание учетной записи группы. После входа на контроллер домена запустите оснастку управления **Active Directory Users and Computers**. В появившемся окне установите курсор на созданной ранее пап-

ке WorkSpace и нажатием правой кнопки мыши вызовите контекстное меню (рис. 9.3). В нем выберите **New | Group**. Запустится мастер создания учетной записи группы.

В появившемся диалоговом окне (рис. 9.4) заполните поля в соответствии с данными, приведенными в колонке "Значение" в табл. 9.3. Обратите внимание, что поле **Group name (pre-Windows 2000)** заполняется автоматически, но может быть изменено вручную в случае необходимости.

Характеристики группы в соответствии с табл. 9.3 требуется оставить по умолчанию, а именно **Global Security group**.

## Программное создание группы

Во время создания учетной записи группы программным способом (VBScript), как и при работе мастера, необходимо жестко задать несколько параметров. В листинге 9.1 приведен сценарий, позволяющий создать учетную запись группы, аналогичную созданной мастером. Параметры сценария приведены в табл. 9.4.

***Таблица 9.4.*** *Параметры сценария для создания учетной записи пользователя*

| Параметр | Описание | Способ назначения |
|---|---|---|
| `cn = Test` | Отображаемое имя | Явный |
| `sAMAccountName = Test` | Имя в сети для совместимости с доменами Windows NT | Явный |
| `groupType = -2147483646` | Область действия и тип группы | Явный |
| `distinguishedName = CN=Test,OU=WorkSpace,DC=Island,DC=ru` | Путь к создаваемой учетной записи группы | Неявный |

**Листинг 9.1. Создание учетной записи группы**

```
set RootDSE = GetObject("LDAP://RootDSE")
Domain = rootDSE.Get("defaultNamingContext")
```

```
Set objOU = GetObject("LDAP://OU=WorkGroup," & Domain)
Set objNewGrp = objOU.Create("group", "cn=Test")

objNewGrp.Put "sAMAccountName", "Test"
objNewGrp.Put "groupType", "-2147483646"
objNewGrp.SetInfo
```

В приведенном сценарии есть одна особенность: если значение параметра `groupType` не задано в сценарии, то в учетной записи группы этот параметр будет создан со значением по умолчанию: `groupType = -2147483646`.

## Поля создаваемой мастером учетной записи

Во время создания учетной записи группы в Active Directory появляется 18 новых атрибутов (рис. 9.5).

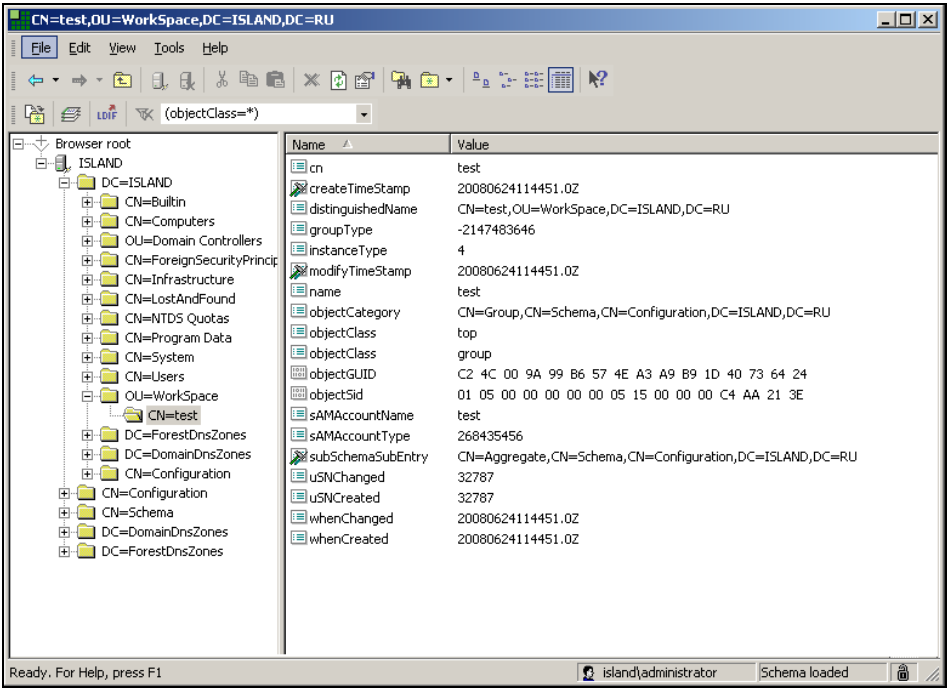



**Рис. 9.5.** Параметры учетной записи группы, созданной с помощью мастера

Часть из них задается явным образом, другая часть — неявным. Некоторые назначаются автоматически.

# Удаление группы

Рассмотрим удаление учетной записи группы безопасности программным способом и с помощью мастера. Особое внимание уделим взаимосвязи атрибутов Active Directory полям мастера.

## Удаление с помощью мастера

Удаление группы безопасности осуществляется из MMC-оснастки Active Directory. Чтобы удалить объект в Active Directory, необходимо установить на него курсор и, вызвав контекстное меню объекта, выбрать в нем **Delete** (рис. 9.6).

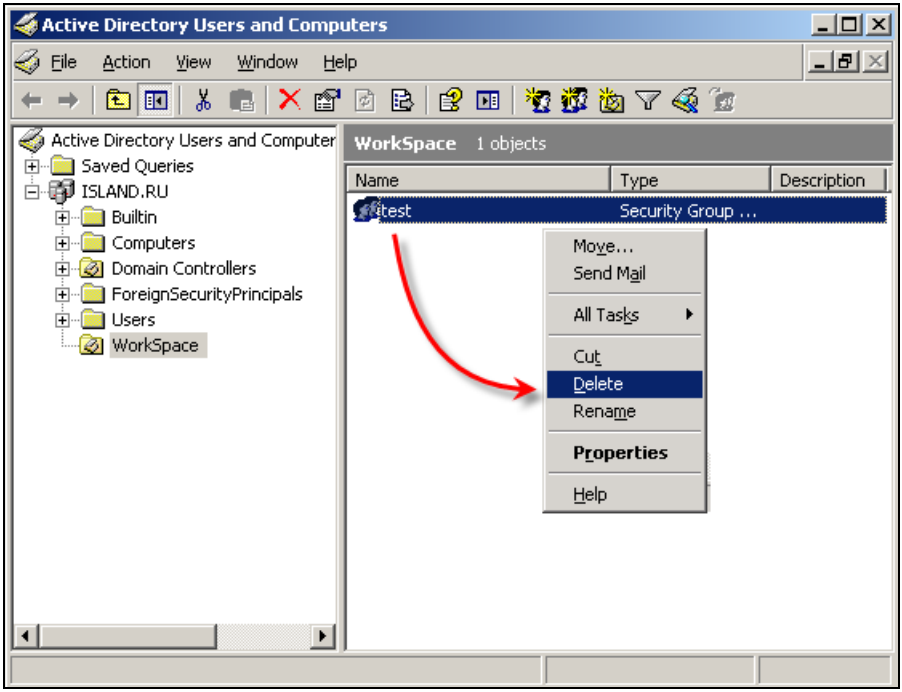



**Рис. 9.6.** Удаление группы безопасности

Поскольку объекты удаляются безвозвратно, то для удаления объекта системой запрашивается подтверждение. После удаления объект можно восстановить только из резервной копии Active Directory.

## Удаление программным способом

В сценарии удаления объекта необходимо указать тип удаляемого объекта и составной путь к этому объекту в каталоге Active Directory. Оба параметра в сумме составляют значение `distinguishedName`. Например, если требуется удалить группу Test (рис. 9.7), то необходимо определить значение параметра `distinguishedName`. Затем это значение надо разделить на имя объекта (значение `cn`) и папку, в которой он находится.

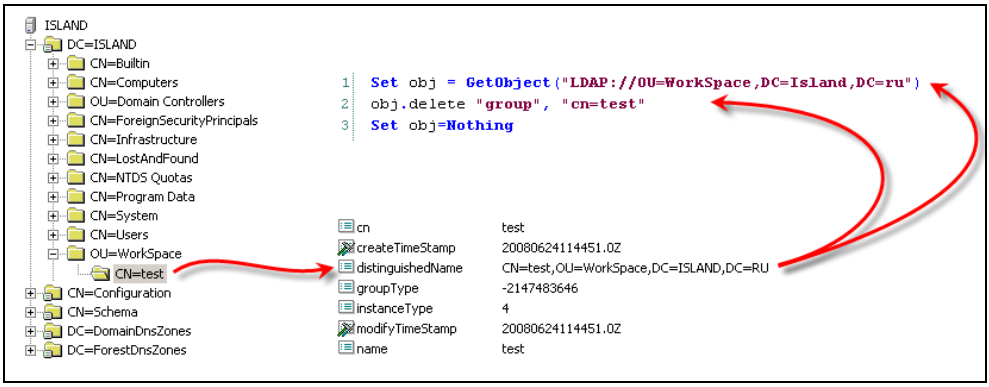



**Рис. 9.7.** Структура параметра `distinguishedName`

Удаление объекта осуществляется при вызове метода `Delete()`. Метод `SetInfo()` не используется. При попытке его вызова интерпретатор VBScript выдаст сообщение об ошибке.

# Чтение параметров группы безопасности

В этом разделе будут рассмотрены параметры, отображаемые в 4 вкладках, которые видны в стандартном режиме просмотра Active Directory.

## Вкладка *General*

На вкладке **General** назначаются параметры, характеризующие название и тип группы. Эта вкладка (рис. 9.8) отображается по умолчанию при вызове свойств группы безопасности. В качестве значений параметров указаны названия соответствующих им полей в Active Directory. В табл. 9.5 эта информация дополнена описанием соответствующих им типов данных.

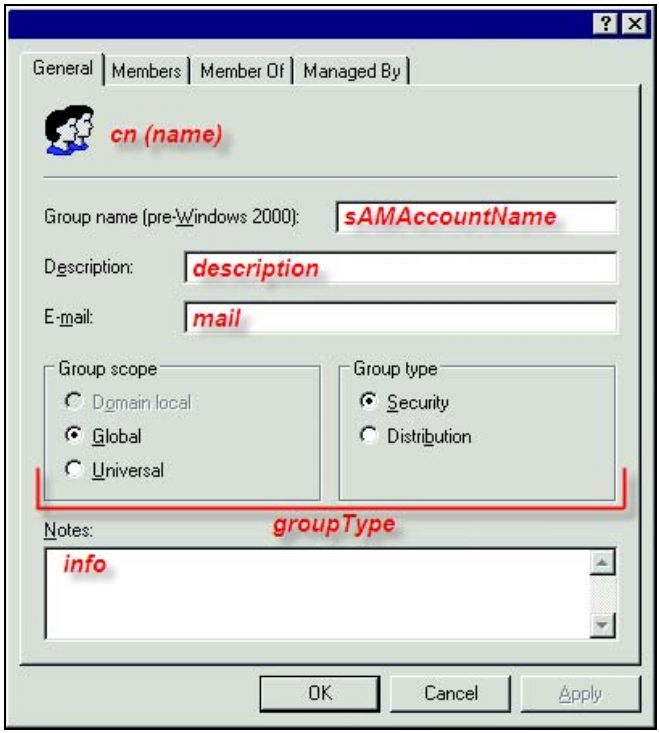



**Рис. 9.8.** Вкладка **General**

***Таблица 9.5.*** *Соответствие полей на вкладке* ***General*** *атрибутам Active Directory*

| **Поле на вкладке *General*** | **Тип** | **Атрибут Active Directory** | **Тип** |
|---|---|---|---|
|  | InputBox | `cn`<br>`name` | String |

***Таблица 9.5*** *(окончание)*

| Поле на вкладке *General* | Тип | Атрибут Active Directory | Тип |
|---|---|---|---|
| **Group name (pre-Windows 2000)** | InputBox | `sAMAccountName` | String |
| **Mail** | InputBox | `Mail` | String |
| **Group scope** | Radio | `groupType` | String |
| **Group type** | Radio | | String |
| **Notes** | InputBox | `info` | String |

## Описание полей

Рассмотрим подробно каждый из задаваемых параметров.

### *Group Name*

Имя группы отображается около значка с изображением  .

На вкладке **General** его невозможно изменять. В Active Directory имени группы соответствуют два атрибута, значения которых совпадают: `name` и `cn`. Для изменения этих атрибутов необходимо переименовать группу.

### *Group name (pre-Windows 2000)*

Параметр **Group name (pre-Windows 2000)** используется для совместимости созданной учетной записи группы с доменами, построенными на основе Windows NT. В Active Directory ему соответствует строковый параметр `sAMAccountName`, значение которого по умолчанию совпадает с именем группы для домена Windows 2k. При необходимости оно может быть изменено как при создании учетной записи группы, так и после.

### *Mail*

Адрес электронной почты группы безопасности, отображаемый в поле **Mail**, заполняется автоматически при создании почтового ящика для группы, одна-

ко может быть задан и вручную. Этому полю в Active Directory соответствует одноименный строковый параметр `mail`.

## Группа *Group scope*

Различают три области действий (**Group scope**) группы: локальная для домена, глобальная и универсальная. По умолчанию группа имеет глобальную область действия (`groupType = 2`). Каждому из трех значений области соответствует индивидуальное значение параметра `groupType` в Active Directory (табл. 9.6).

***Таблица 9.6.*** *Значения параметра* `groupType`*. Область действия группы*

| Область действия группы | Значение параметра `groupType` | Описание |
|---|---|---|
| **Domain local** | 4 | Область действия группы — домен |
| **Global** | 2 | Глобальная группа с областью действий, охватывающей два и более доменов, при условии, что между доменами установлены доверительные отношения |
| **Universal** | 8 | Универсальная группа |

## Группа *Group type*

Существуют два типа групп в доменах Windows 2k — **Security** (безопасности) и **Distribution** (распространения). По умолчанию используется группа безопасности. Каждому из типов групп соответствует индивидуальное значение параметра `groupType` (табл. 9.7).

***Таблица 9.7.*** *Значения параметра* `groupType`*. Тип группы*

| Область действия группы | Значение параметра `groupType` | Описание |
|---|---|---|
| **Security** | –2147483648 | Группа безопасности |
| **Distribution** | 0 | Группа распространения |

Значения обеих групп параметров фиксируются в одном и том же атрибуте — `groupType`. Здесь нет никаких недоразумений: итоговым значением атрибута является сумма значений, соответствующих настройкам типа и области действия группы. Например, если группа создана без изменения этих настроек, то есть является Global Security group, то `groupType = 2 + (-2147483648) = -2147483646`.

### *Notes*

Параметру `info` в Active Directory соответствует поле **Notes**. В качестве значения может быть указан любой комментарий, касающийся группы. Обратите внимание, что тип данных этого параметра — строка, а не массив, элементы которого строки. Многострочности значения параметра `info` достигают переводом каретки на новую строку. В VBScript для этого используется константа `vbNewLine` или выражение `chr(13) & chr(10)`.

## Вкладка *Members*

На вкладке **Members** (рис. 9.9) формируется список членов группы безопасности. Ими могут быть пользователи и группы.

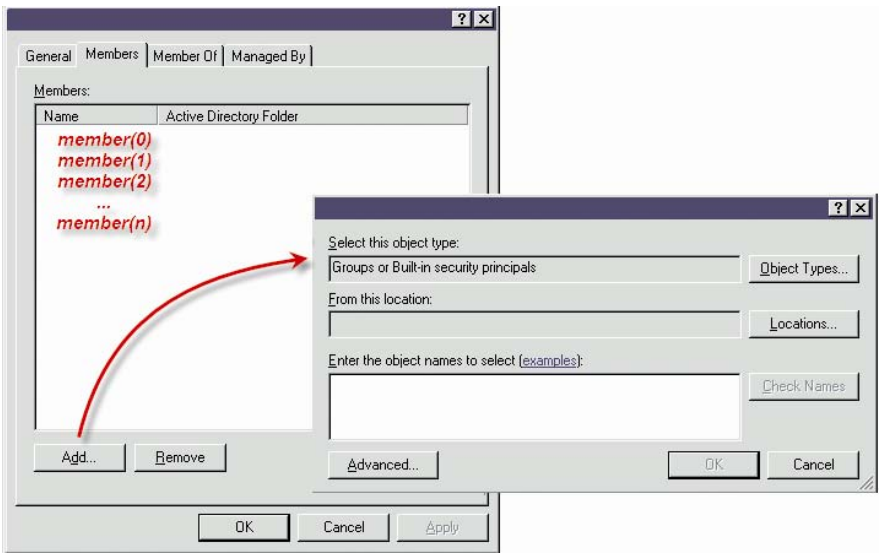



**Рис. 9.9.** Вкладка **Members**

В табл. 9.8 указаны соответствующие атрибуты Active Directory.

***Таблица 9.8.*** *Соответствие полей на вкладке* ***Members*** *атрибутам Active Directory*

| Поле на вкладке *Members* | Тип | Атрибут Active Directory | Тип |
|---|---|---|---|
| **Member** | ListBox, Button | `member` | Array |

## Описание полей

Основным элементом управления на вкладке является список групп. Также присутствуют две кнопки для добавления и удаления элементов списка.

### *Member*

Членом группы безопасности может быть как группа, так и пользователь. Для включения объектов в группу воспользуйтесь мастером, вызываемым кнопкой **Add**, находящейся под списком, для исключения из списка — кнопкой **Remove**.

Рассмотрим подробнее механизм включения пользователей и групп в группу безопасности. Для добавления пользователя или группы нажмите кнопку **Add** и в появившемся диалоговом окне (рис. 9.9) введите фрагмент имени группы или пользователя, которого необходимо найти. Для запуска механизма поиска нажмите на кнопку **Check Names**. В поле **Enter the object names to select** указывается имя группы безопасности (`cn` или `sAMAccountName`), при этом в списке фиксируется значение поля `distinguishedName`, в то время как отображается `cn` этого объекта.

Для исключения группы или пользователя из содержащей их группы необходимо поставить курсор на удаляемый объект и нажать кнопку **Remove**, после чего подтвердить свой выбор.

# Вкладка *Member Of*

На вкладке **Member Of** (рис. 9.10) формируется список групп, членом которой является данная группа безопасности. Их соответствие атрибутам Active Directory описано в табл. 9.9.

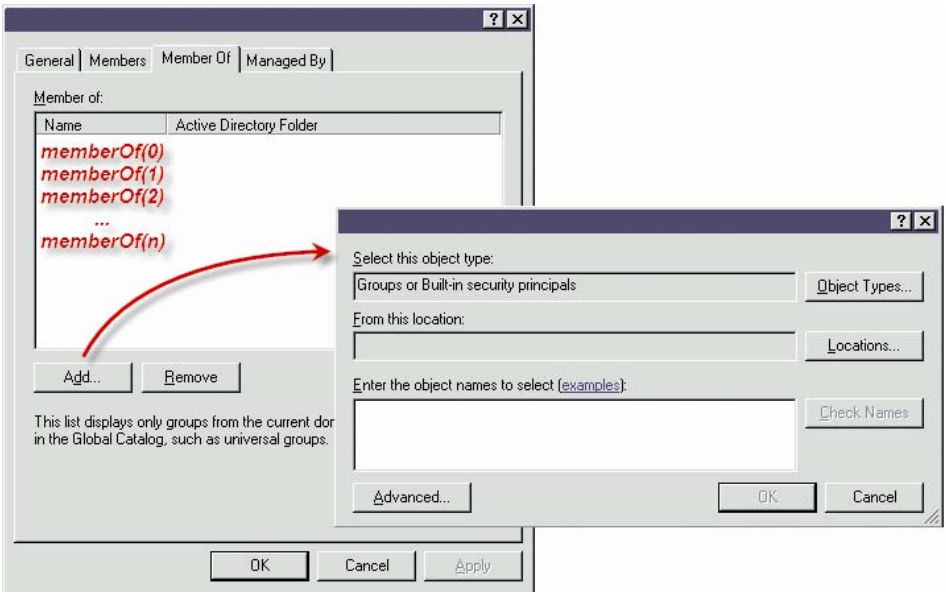



**Рис. 9.10.** Вкладка **Member Of**

***Таблица 9.9.*** *Соответствие полей на вкладке* ***Member Of*** *атрибутам Active Directory*

| **Поле на вкладке *Member Of*** | **Тип** | **Атрибут Active Directory** | **Тип** |
|---|---|---|---|
| **Member of** | ListBox, Button | `memberOf` | Array |

## Описание полей

Родительские группы представлены в списке, для управления которым служат две кнопки.

### *Member of*

Группа безопасности может входить в другую группу безопасности. Для включения редактируемой группы в другую необходимо воспользоваться кнопкой **Add**, а для исключения данной группы из других групп — кнопкой **Remove**.

Рассмотрим подробнее механизм включения группы в другие группы. После нажатия кнопки **Add** появляется диалоговое окно (рис. 9.10), в котором осу-

ществляется поиск групп по заданным критериям. В поле **Enter the object names to select** указывается имя группы безопасности (`cn` или `sAMAccountName`), при этом в списке фиксируется значение поля `distinguishedName`, в то время как отображается `cn` этого объекта.

Для исключения группы необходимо установить курсор на удаляемую группу и нажать кнопку **Remove**, после чего подтвердить свои действия.

## Вкладка *Managed By*

На вкладке **Managed By** (рис. 9.11) назначается "управляющий" группой — человек, которому даются права для изменения свойств группы, ее членов. В этой вкладке можно изменить всего лишь одно поле: **Name**, которому в Active Directory соответствует поле `manager`. Остальные поля, включая флажок **Manager can update membership list**, недоступны до назначения ответственного лица.

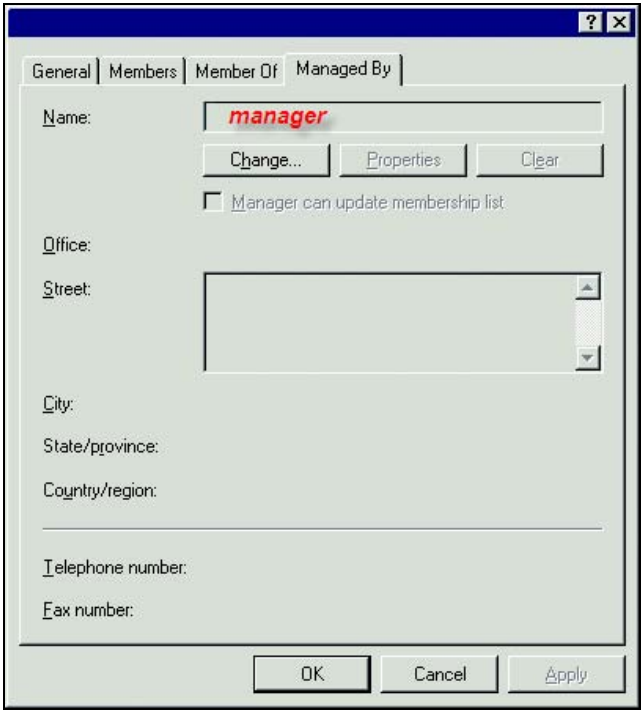



**Рис. 9.11.** Вкладка **Managed By**

## Описание полей

### *Name*

Нажатием на кнопку **Change** назначают пользователя домена, который получает расширенные права по управлению свойствами группы. В Active Directory этому полю соответствует параметр `manager`. Для программиста важно знать, что в текстовом поле **Name** отображается `cn` сотрудника, однако в Active Directory в поле `manager` хранится `distinguishedName` — составной путь к учетной записи пользователя в формате LDAP.

После назначения расширенных прав пользователю становятся доступны оставшиеся две кнопки: **Properties** и **Clear**. С помощью кнопки **Properties** можно вызывать стандартное диалоговое окно учетной записи пользователя. С помощью кнопки **Clear** удаляют значение поля `manager` и возвращаются к настройкам по умолчанию.

### *Manager can update membership list*

После назначения ответственного лица становится доступна опция **Manager can update membership list**, дающая возможность ответственному лицу управлять членством этой группы. Изменения этого флажка отражаются и на вкладке **Security** (Безопасность), отображаемой в режиме расширенного просмотра каталога Active Directory.

### *Office, Street…*

Во всех остальных полях выводится информация о пользователе, обладающем расширенными правами. Она включает в себя следующие данные:

- **Office** — номер офиса, в котором находится сотрудник;
- **Street** — улица и номер дома, где расположен офис сотрудника;
- **City** — город, поселок, деревня, где расположена компания;
- **State/province** — штат или область, то есть более мелкое, чем страна, деление;
- **Country/region** — страна или регион расположения предприятия;
- **Telephone number** — контактный телефон для связи;
- **Fax number** — номер факсимильного аппарата. Значение может совпадать с предыдущим полем.

## Глава 10



# Управление контейнером

В этой главе описаны принципы управления контейнером Active Directory с помощью мастера и сценариев. Приведены сопоставления названий полей мастера и атрибутов Active Directory, описаны нюансы управления этим объектом.

## Основные термины и понятия

- **Organizational Unit** — объект, с помощью которого в каталоге Active Directory построена иерархическая структура. Контейнер является единственным объектом, который может включать в себя все остальные типы объектов, в том числе — сам себя.
- **GPO** (Group Policy Object) — это набор правил, в соответствии с которыми производится настройка рабочей среды. Объект групповой политики (Group Policy Object, GPO) состоит из двух физически раздельных составляющих: контейнера групповой политики (Group Policy Container, GPC) и шаблона групповой политики (Group Policy Template, GPT).

## Создание контейнера

Рассмотрим работу мастера создания контейнера, обращая внимания на изменения, происходящие в Active Directory. Во время его работы системным администратором явным образом назначается имя папки и неявно — местоположение объекта. Остальные параметры, например, описание, можно изменить после окончания работы мастера.

Запуск мастера осуществляется из MMC-консоли **Active Directory Users and Computers** выбором пункта **New | Organizational Unit** из контекстного меню папки (рис. 10.1), в которой необходимо создать подпапку. Таким образом определяется местоположение объекта.

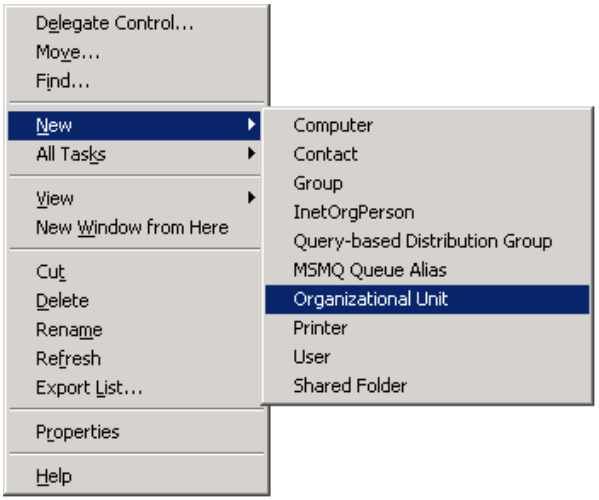



**Рис. 10.1.** Запуск мастера **Создание контейнера**

## Работа мастера. Теория

Работа мастера состоит из одного шага (рис. 10.2). Во время создания контейнера необходимо назначить лишь его имя (табл. 10.1).

***Таблица 10.1.*** *Параметры контейнера*

| Поле мастера | Атрибут Active Directory | Тип данных | Описание |
|---|---|---|---|
| **Name** | `ou, name` | String | Имя контейнера |

### *Name*

Поле **Name** — единственный и обязательный параметр, который необходимо задать для создания контейнера.

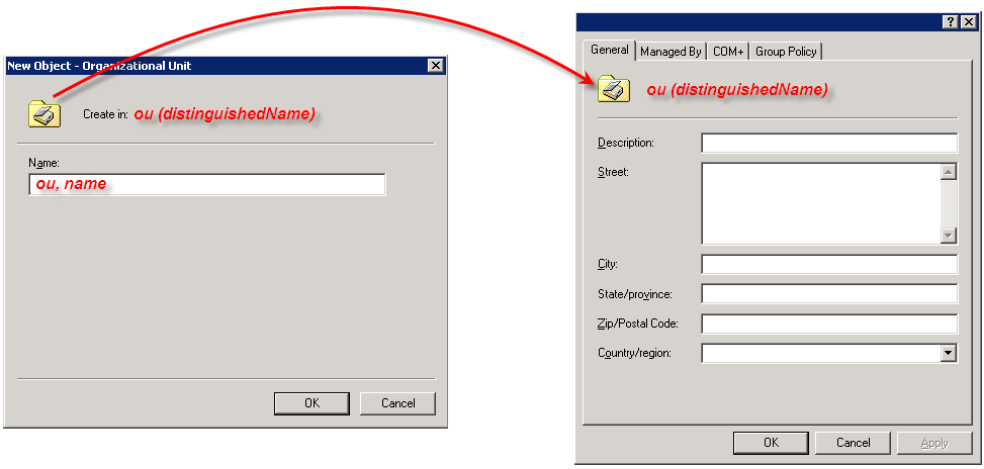



**Рис. 10.2.** Работа мастера **Создание нового контейнера**

Поскольку структура домена Windows NT была линейной, то понятия "контейнер" в ней не существовало. По этой причине никаких параметров, обеспечивающих совместимость с доменом Windows NT, нет.

## Работа мастера. Практика

Создадим контейнер WorkSpace в корневом каталоге Active Directory. Исходные данные приведены в табл. 10.2.

***Таблица 10.2.*** *Исходные данные для создания контейнера*

| Поля мастера | Значение | Описание |
|---|---|---|
| **Create In** | ISLAND.RU | Учетная запись группы будет создана в корневом каталоге Active Directory |
| **Name** | WorkSpace | Имя создаваемого контейнера |

Рассмотрим пошагово создание учетной записи контейнера. После входа на контроллер домена в сети запустите оснастку управления **Active Directory Users and Computers**. В появившемся окне установите курсор на контейнер, в котором необходимо создать папку. В данном случае нужно выделить кор-

невой каталог домена (рис. 10.3). Нажмите правую кнопку мыши и в открывшемся меню выберите пункт **New | Organizational Unit**.

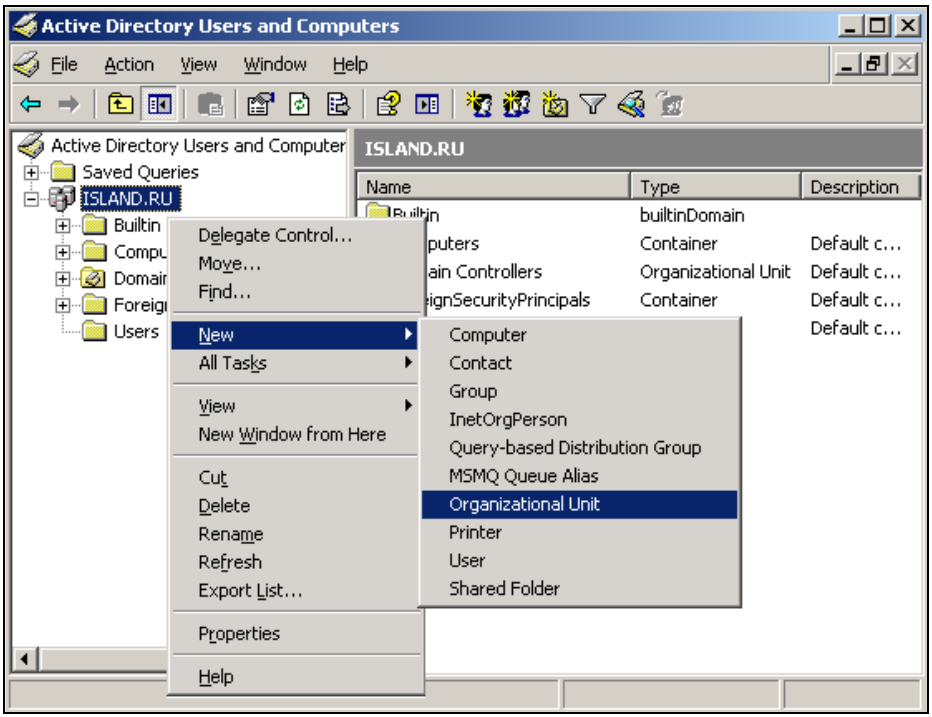



**Рис. 10.3.** Запуск мастера создания контейнера

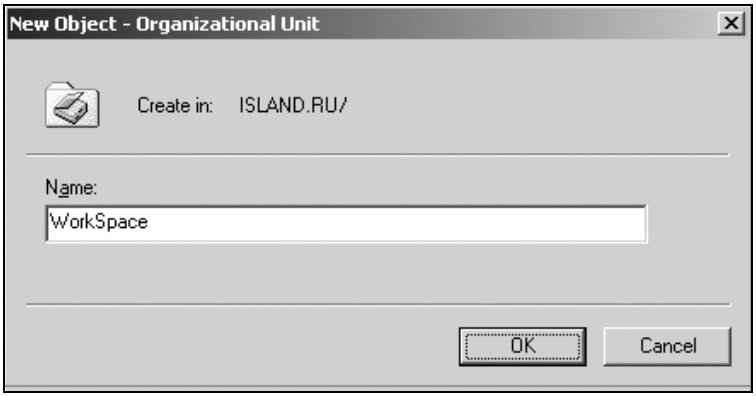



**Рис. 10.4.** Работа мастера создания контейнера

В появившемся диалоговом окне введите имя создаваемого контейнера (рис. 10.4).

## Программное создание учетной записи контейнера

Для создания контейнера программным способом (VBScript), как при работе мастера, необходимо жестко задать несколько параметров. В листинге 10.1 приведен сценарий, позволяющий создать аналогичный контейнер, созданный мастером. Параметры сценария приведены в табл. 10.3.

***Таблица 10.3.*** *Параметры сценария для создания контейнера*

| Параметр | Описание | Способ назначения |
|---|---|---|
| `ou = WorkSpace` | Отображаемое имя | Явный |
| `distinguishedName = DC=Island,DC=ru` | Путь к создаваемому контейнеру | Неявный |

**Листинг 10.1. Создание контейнера**

```
set RootDSE = GetObject("LDAP://RootDSE")
Domain = rootDSE.Get("defaultNamingContext")

Set objUsers = GetObject("LDAP://" & Domain)
Set objNewUser = objUsers.Create("OrganizationalUnit", "OU=WorkGroup")
objNewUser.SetInfo
```

## Поля контейнера, создаваемого мастером

В каталоге Active Directory созданному контейнеру соответствует 15 атрибутов (рис. 10.5). Все из них, кроме одного (`ou`, равного `name`), задаются неявным образом.

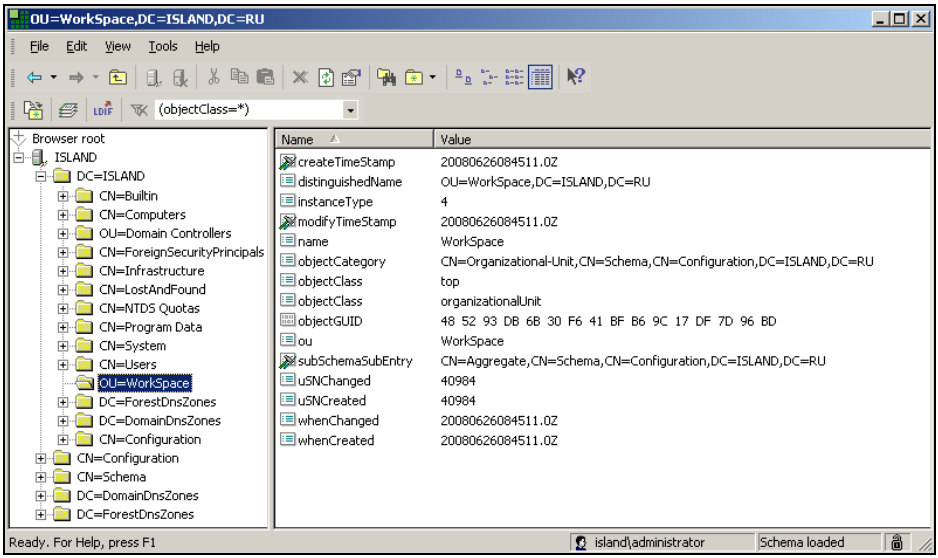



**Рис. 10.5.** Параметры контейнера, созданного с помощью мастера

# Удаление контейнера

Рассмотрим удаление контейнера с помощью мастера и сценария. Особое внимание уделим их взаимосвязи.

## Удаление с помощью мастера

Удаление контейнера осуществляется из MMC-оснастки Active Directory. Чтобы удалить объект в Active Directory, необходимо установить на него курсор и, вызвав его контекстное меню, выбрать пункт **Delete** (рис. 10.6).

Поскольку объекты удаляются безвозвратно, то для его удаления система запрашивает подтверждение. После удаления контейнер можно восстановить только из резервной копии Active Directory.

## Удаление программным способом

В сценарии удаления объекта необходимо указать его тип и составной путь к его родительской папке в формате RDN.

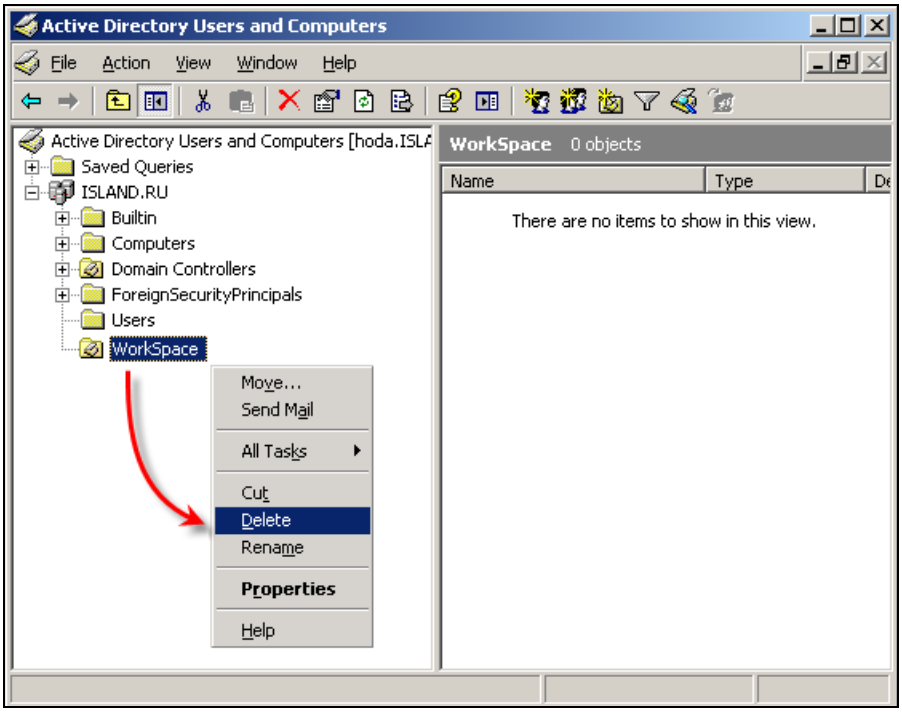



**Рис. 10.6.** Удаление группы безопасности

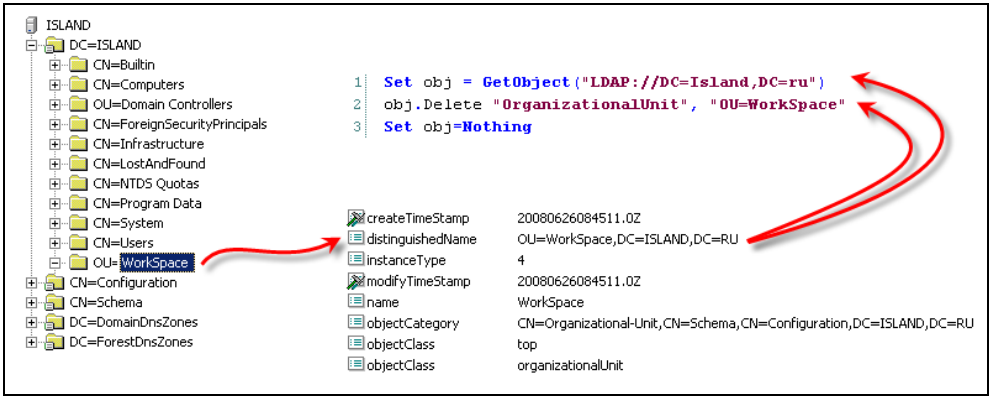



**Рис. 10.7.** Структура параметра `distinguishedName`

Оба перечисленных параметра в сумме составляют значение `distinguishedName`. Например, если требуется удалить контейнер WorkSpace

(рис 10.7), то необходимо определить значение параметра `distinguishedName`. Затем из этого значения следует узнать имя объекта (значение `ou` или `name`) и имя его родительской папки. В данном случае — это корень домена.

Удаление объекта осуществляется с помощью метода `Delete()`, при этом метод `SetInfo()` для сохранения изменений не используется. При попытке его вызова интерпретатор VBScript выдаст сообщение об ошибке.

# Чтение свойств контейнера

В этом разделе будут рассмотрены параметры, отображаемые в 4 вкладках, которые видны в стандартном режиме просмотра Active Directory.

## Вкладка *General*

На вкладке **General** назначаются все основные параметры контейнера (рис. 10.8): описание, размещение. В большинстве случаев значение присваивается только полю **Description**, в котором дается описание контейнера. На рисунке в качестве значений параметров указаны названия соответствующих им полей в Active Directory. В табл. 10.4 приведено соответствие полей мастера атрибутам в каталоге Active Directory.

***Таблица 10.4.*** *Соответствие полей на вкладке* ***General*** *параметрам в Active Directory*

| **Поле на вкладке *General*** | **Тип** | **Поле в Active Directory** | **Тип** |
|---|---|---|---|
|  | | `ou`<br>`name` | String |
| **Description** | InputBox | `description` | String |
| **Street** | InputBox | `street` | String |
| **City** | InputBox | `l` | String |
| **State/province** | InputBox | `st` | String |
| **Zip/Postal Code** | InputBox | `postalCode` | String |

***Таблица 10.4*** *(окончание)*

| Поле на вкладке *General* | Тип | Поле в Active Directory | Тип |
|---|---|---|---|
| **Country/region** | InputBox | co | String |
| | ComboBox | c | String |

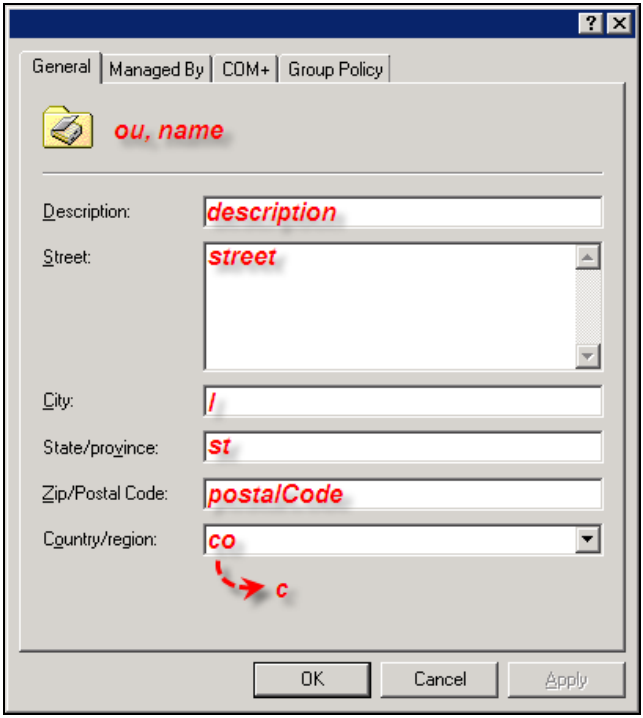



**Рис. 10.8.** Вкладка **General**

## Описание полей

Опишем более подробно присутствующие на вкладке поля.

### *Container Name*

Название контейнера (папки) отображается около значка с изображением  .

На вкладке **General** его невозможно изменить. В Active Directory названию контейнера соответствуют два атрибута, значения которых дублируются — `name` и `ou`. Для изменения параметров необходимо переименовать контейнер (пункт **Rename** контекстного меню контейнера, см. рис. 10.6).

### *Description*

Значением данного поля является описание контейнера, его содержание. В каталоге Active Directory полю **Description** соответствует одноименный атрибут.

### *Street*

Параметру **Street** соответствует поле `streetAddress` в Active Directory. Значение поля — адрес местонахождения, соответствующего контейнеру, с указанием улицы, номера дома, корпуса, номера офиса. Параметр `streetAddress` — строка, которая разбивается на подстроки с помощью перевода каретки на новую строку. В VBScript для этого используется константа `vbNewLine` или выражение `chr(13) & chr(10)`.

### *City*

В поле **City** указывается название города, к которому относится контейнер. Этому полю соответствует строковый параметр `l`.

### *State/province*

Здесь указывается штат, провинция или область, в зависимости от административного деления. Полю **State/province** соответствует строковый параметр `st`.

### *Zip/Postal Code*

В поле **Zip/Postal Code** задается почтовый индекс города, соответствующего контейнеру, в строковом виде. В Active Directory этому полю соответствует параметр `postalCode`.

### *Country/region*

Регион выбирается из заданного списка. При выборе региона в Active Directory создается два поля — `co` и `c`. Выбранное значение страны из списка

в текстовом виде фиксируется в тестовом виде в параметре `co`. Параллельно создается еще одно поле — `c`, значение которого — сокращенное название страны. Например, для России это RU, для США — US и т. д. Полный список кодов приведен в *приложении 5*.

## Вкладка *Managed By*

На вкладке **Managed By** (рис. 10.9) назначается "управляющий" контейнером — пользователь, которому даются права для изменения свойств контейнера. На этой вкладке можно изменить всего лишь одно поле **Name**, которому в Active Directory соответствует поле `manager`. Для этого надо воспользоваться кнопкой **Change**. Для удаления уже назначенного управляющего контейнером служит кнопка **Clear**.

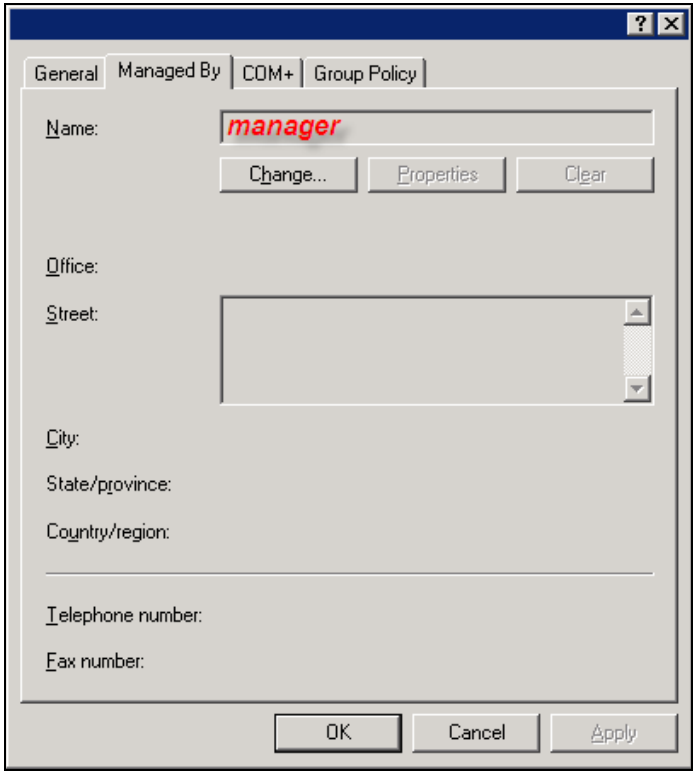



**Рис. 10.9.** Вкладка **Managed By**

При помощи кнопки **Properties** можно посмотреть учетную запись пользователя, назначенного управляющим. Остальные поля недоступны и служат для отображения информации о пользователе.

Несмотря на то, что в поле **Name** отображается значение поля `cn` учетной записи пользователя, в каталоге Active Directory значением атрибута `manager` является полный составной путь к учетной записи пользователя, то есть его `distinguishedName`.

## Вкладка *COM+*

Вкладка COM+ (рис. 10.10) не рассматривается, поскольку для управления этой вкладкой необходимо использовать расширенный режим работы Active Directory.

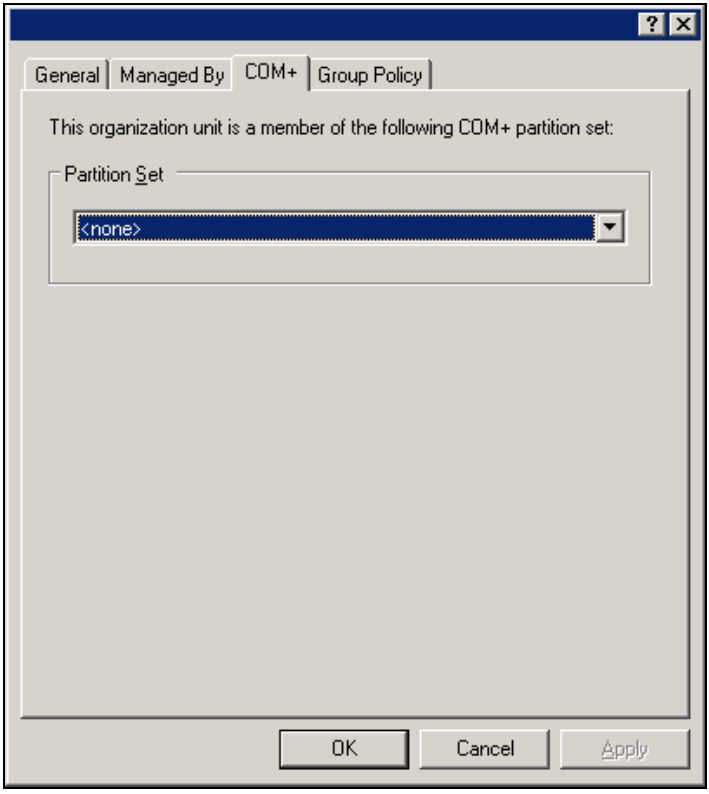



**Рис. 10.10.** Вкладка **COM+**

## Вкладка *Group Policy*

На вкладке **Group Policy** (рис. 10.11) осуществляется назначение групповых политик, которые будут действовать на все объекты, находящиеся в этом контейнере.

Групповые политики могут быть применены к сайту, домену или контейнеру (объект типа `SDOU`). Одна и та же групповая политика может действовать на несколько контейнеров.

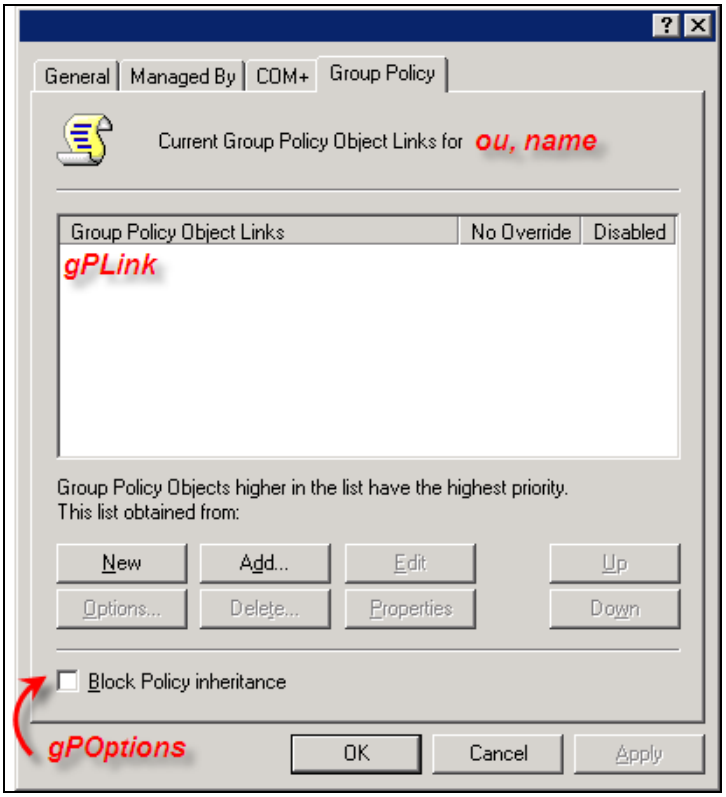



**Рис. 10.11.** Вкладка **Group Policy**

В табл. 10.5 приведено соответствие полей в окне свойств контейнера атрибутам в каталоге Active Directory.

***Таблица 10.5.*** *Соответствие полей на вкладке* ***Group Policy*** *атрибутам Active Directory*

| Поле на вкладке *Group Policy* | Тип | Поле в Active Directory | Тип |
|---|---|---|---|
|  | | `ou`<br>`name` | String |
| **GPO Policy Object Links** | ListBox | `gPLink` | String |
| **Block Policy inheritance** | CheckBox | `gPOprions` | Boolean |

## Описание полей

Рассмотрим подробно каждый из задаваемых параметров.

### *Current Group Policy Links*

Справа от иконки, изображающей групповую политику, указан контейнер, на который она будет действовать. В качестве имени, как и на вкладке **General**, отображается значение атрибута `ou` или `name`.

### *Group Police Object Links*

Список **Group Police Object Links** отображает все политики, которые действуют на находящиеся в контейнере объекты. Несмотря на то, что он должен бы быть сформирован на основе массива, все данные хранятся в атрибуте `gPLink` в строковом виде в соответствии со следующим шаблоном:

```
[LDAP://CN={GUID},CN=Policies,CN=System,DC=…;0]
[LDAP://CN={GUID},CN=Policies,CN=System,DC=…;0]…
```

При этом разрыва строк в значении атрибута нет, а шаблон разбит на строки лишь для удобства чтения. Можно сказать, что приведенный шаблон — псевдомассив, элементы которого содержатся в квадратных скобках и представляют собой ссылки на хранилище политик в каталоге Active Directory. Обратите внимание, что в окне свойств контейнера отображается название политики (поле `name`), а в значении атрибута фигурирует GUID объекта.

### Block Policy inheritance

Параметр **Block Policy inheritance** отвечает за область действия указанных групповых политик внутри контейнера. Параметру **Block Policy interface** соответствует булево значение атрибута `gPOptions` каталога Active Directory. По умолчанию указанные в списке политики безопасности действуют на все вложенные в контейнер объекты: учетные записи групп, пользователей, другие (дочерние) контейнеры (`gPOptions = 0`). Если же параметр отмечен флажком (`gPOptions = 1`), политики действуют только на объекты, находящиеся непосредственно в этом контейнере.

Глава 11



# Управление учетной записью компьютера

В этой главе описаны принципы управления учетной записью компьютера в Active Directory с помощью мастера и сценария. Сопоставлены названия полей мастера и атрибуты Active Directory, описаны нюансы управления объектом.

## Основные термины и понятия

- **Computer** — объект в составе каталога Active Directory. Для удобства администрирования объекты данной категории объединяются в группы и помещаются в контейнеры. Как и к учетным записям пользователей, к ним применяются доменные политики (GPO), с помощью которых регламентируется доступ к сетевым ресурсам.
- **GPO** (Group Policy Object) — это набор правил, в соответствии с которыми производится настройка рабочей среды. Объект групповой политики (Group Policy Object, GPO) состоит из двух физически раздельных составляющих: контейнера групповой политики (Group Policy Container, GPC) и шаблона групповой политики (Group Policy Template, GPT).

## Создание учетной записи компьютера

Чтобы компьютер мог войти в состав домена, он должен иметь безопасный канал связи с одним из контроллеров домена. Процесс создания данного канала называется публикацией.

Создать учетную запись компьютера можно несколькими способами:

- с помощью мастера в MMC-консоли **Active Directory Users and Computers**;
- программным способом с помощью соответствующего сценария;
- вручную зарегистрировать компьютер в домене (Active Directory) непосредственно с консоли рабочей станции.

Рассмотрим работу мастера создания учетной записи компьютера, акцентируя внимание на изменениях, происходящих в Active Directory. Во время его работы системным администратором явным образом назначается имя компьютера, его роль в сети и ряд других параметров, а также, неявно, — местоположение соответствующего объекта. Остальные параметры, например, описание, можно изменить после окончания работы мастера.

## Создание учетной записи компьютера в MMC-консоли

Создание учетной записи в Active Directory не подменяет регистрации компьютера в домене.

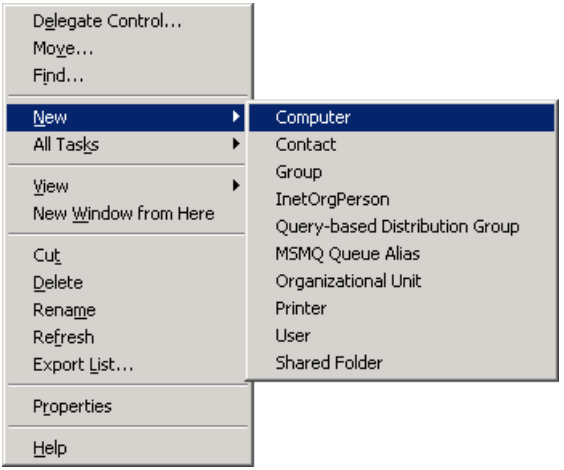



**Рис. 11.1.** Запуск мастера **Создание учетной записи компьютера**

Создав учетную запись компьютера в Active Directory, системный администратор может дать привилегии какой-либо группе, участники которой могут

регистрировать компьютеры в домене с рабочей станции. Это делается исключительно для того, чтобы дать возможность сотрудникам службы поддержки регистрировать компьютеры в домене, но не наделять их для этого административными полномочиями.

Запуск мастера осуществляется из MMC-консоли **Active Directory Users and Computers** выбором пункта **New | Computer** из контекстного меню папки (рис. 11.1), в которой необходимо создать новую учетную запись компьютера. Таким способом определяется местоположение объекта — его RDN-путь.

## Работа мастера. Теория

Работа мастера состоит из нескольких шагов. В зависимости от выбранных опций и надстроек, установленных на контроллере домена, количество шагов может изменяться. По умолчанию создание учетной записи компьютера состоит из одного шага. Однако, в зависимости от установленных на контроллере домена приложений, количество шагов может варьироваться. В этом разделе будут рассмотрены все возможные варианты.

### Шаг 1. Назначение имени новой учетной записи

Традиционно на первом шаге работы мастера (рис. 11.2) необходимо назначить имя компьютера и определить его роль в домене.

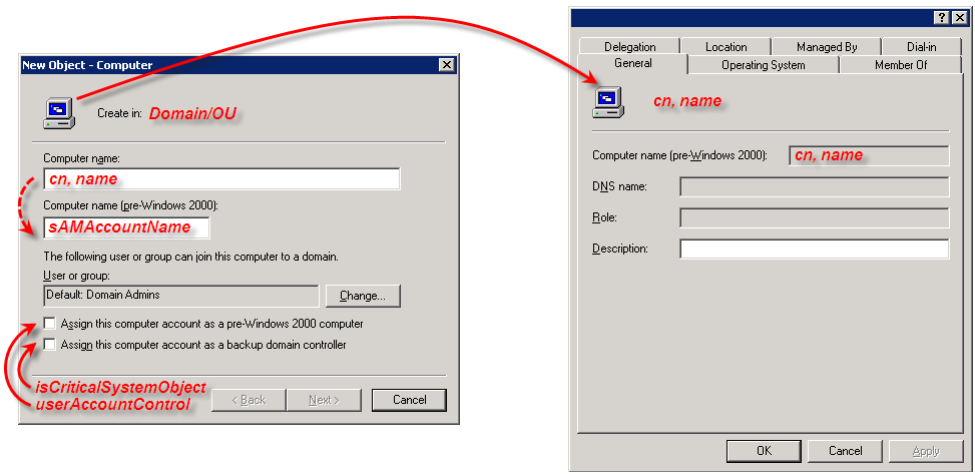



**Рис. 11.2.** Работа мастера **Создание учетной записи компьютера**. Шаг 1

В табл. 11.1 приведено соответствие полей мастера атрибутам каталога Active Directory.

***Таблица 11.1.*** *Поля мастера создания учетной записи компьютера*

| Параметр мастера | Поле в Active Directory | Тип данных | Описание |
|---|---|---|---|
| **Computer name** | `ou`, `name` | String | Имя учетной записи компьютера в домене |
| **Computer name (pre-Windows 2000)** | `sAMAcoountName` | String | Имя учетной записи компьютера в домене. Используется для совместимости с доменами Windows NT |
| **User or group** | — | ACE | Объект (пользователь или группа), обладающий полномочиями для регистрации компьютера в домене |
| **Assign this computer account as a pre-Windows 2000 computer** | `userAccountControl` | Integer | Режим совместимости учетной записи компьютера для доменов Windows NT |
| **Assign this computer account as a backup domain controller** | `IsCriticalSystem Object` | Boolean | Роль учетной записи компьютера: сервер /рабочая станция или резервный контроллер домена |

В этом диалоговом окне также осуществляется назначение группы или пользователя, который может зарегистрировать компьютер в домене (см. *разд. "Публикация компьютера в домене"* далее *в этой главе*). Для упрощения регистрации компьютеров в домене системный администратор может делегировать право регистрации компьютеров в нем пользователям какой-либо группы. Ими могут быть сотрудники службы поддержки, опытные пользователи, локальные администраторы рабочих станций и т. д. По умолчанию назначена группа Domain Admins.

В этом же диалоговом окне указывается роль учетной записи компьютера: сервер/рабочая станция или резервная копия контроллера домена. В зависимости от выбранной роли учетная запись либо приобретает статус "критического объекта", либо нет. По умолчанию создается учетная запись локального компьютера, поэтому `isCriticalSystemObject = False`.

### Шаг 2. Создание управляемого объекта

Если создаваемая учетная запись компьютера является управляемой, то необходимо установить флажок напротив надписи **This is a managed computer** и ввести его GUID (рис. 11.3), который можно найти в БИОСе. Формат ввода — {xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxx}. При присвоении значения параметру `netBootGUID` осуществляется проверка на правильность ввода данных. По умолчанию значение данному параметру не присвоено.

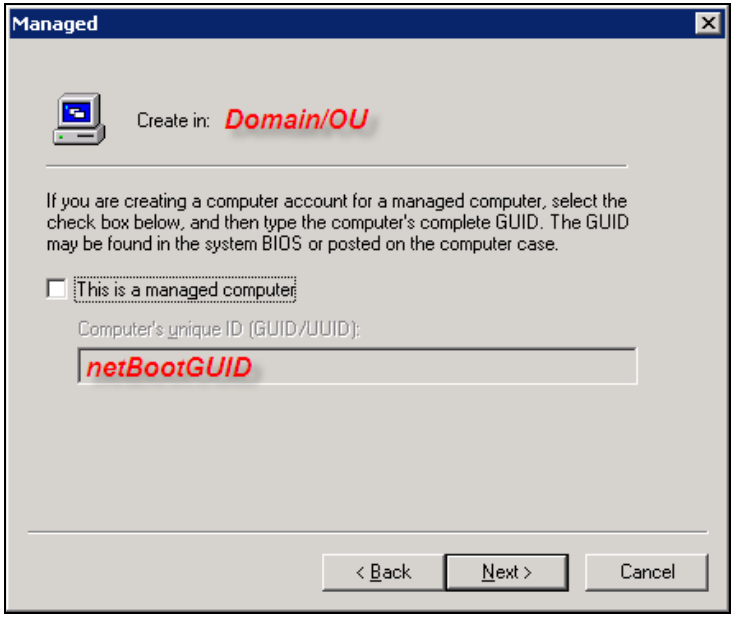



**Рис. 11.3.** Работа мастера **Создание учетной записи компьютера**. Шаг 2

### Шаг 3. Назначение управляющего объекта

Третий шаг выполняется только при условии, что на предыдущем, втором, шаге создаваемая учетная запись отмечена как управляемая.

На этом шаге назначается компьютер, который будет управлять создаваемой учетной записью (рис. 11.4).

В качестве имени управляющего компьютера задают один из серверов, зарегистрированных в домене. Если не задан ни один из них, то устанавливается значение **Any available remote installation server** (Любой доступный сервер удаленной установки).

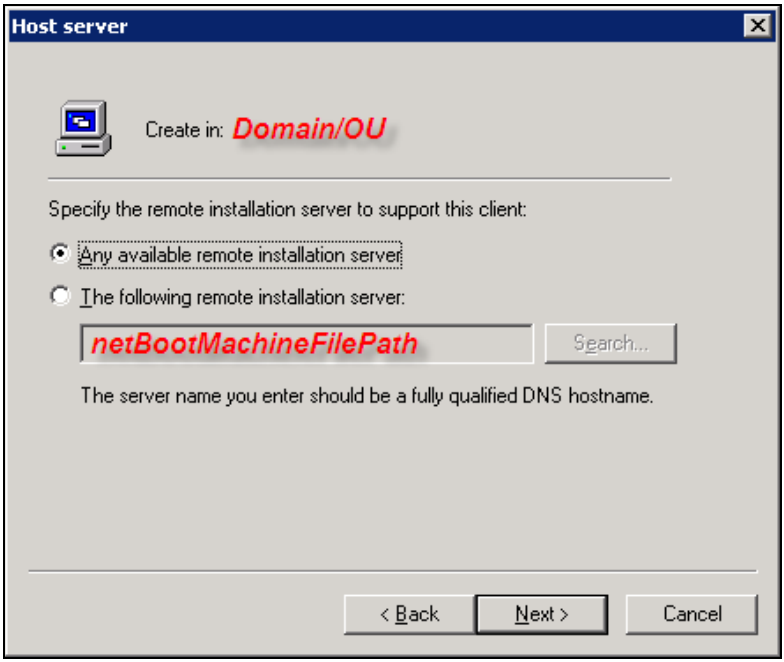



**Рис. 11.4.** Работа мастера **Создание учетной записи компьютера**. Шаг 3

## Шаг 4. Проверка параметров создаваемой учетной записи компьютера

На последнем этапе выводится диалоговое окно, в котором представлена для подтверждения вся введенная с помощью мастера информация. Здесь выводится информация об имени создаваемой учетной записи, а также, если были заданы, то и GUID этого компьютера и имя управляющего сервера. Все значения параметров, заданных во время выполнения мастера, хранятся в соответствующих свойствах создаваемой учетной записи компьютера.

## Работа мастера. Практика

Создадим учетную запись компьютера 1001PC с помощью мастера. При этом члены группы support должны обладать достаточными правами для регистрации компьютеров в домене с рабочей станции. Исходные данные приведены в табл. 11.2.

***Таблица 11.2.*** *Исходные данные для создания учетной записи компьютера*

| Поля мастера | Значение | Комментарий |
|---|---|---|
| **Create in** | Island\WorkSpace | Создается учетная запись компьютера во встроенной папке computers, находящейся в корневом каталоге Active Directory |
| **Computer name** | 1001pc | Имя новой учетной записи компьютера |
| **Computer name (pre-Windows 2000)** | 1001PC | Имя новой учетной записи компьютера для совместимости с доменами Windows NT |
| **Assign this computer account as a pre-Windows 2000 computer** | FALSE (по умолчанию) | Управление возможностью использовать этот компьютер для совместимости с доменами Windows NT |
| **Assign this computer account as a backup domain controller** | FALSE (по умолчанию) | Управление ролью новой учетной записи. Значение параметра остается по умолчанию: роль компьютера — сервер/рабочая станция |
| **User or group** | Island/support | Группа безопасности, члены которой могут зарегистрировать рабочую станцию в домене |

На контроллере домена необходимо запустить оснастку управления **Active Directory Users and Computers**. В появившемся окне установите курсор на папку WorkSpace, в которой необходимо создать новую учетную запись. Для этого нажатием правой кнопки мыши вызовите контекстное меню (рис. 11.5) и в нем выберите **New | Computer**. Запустится мастер создания учетной записи компьютера.

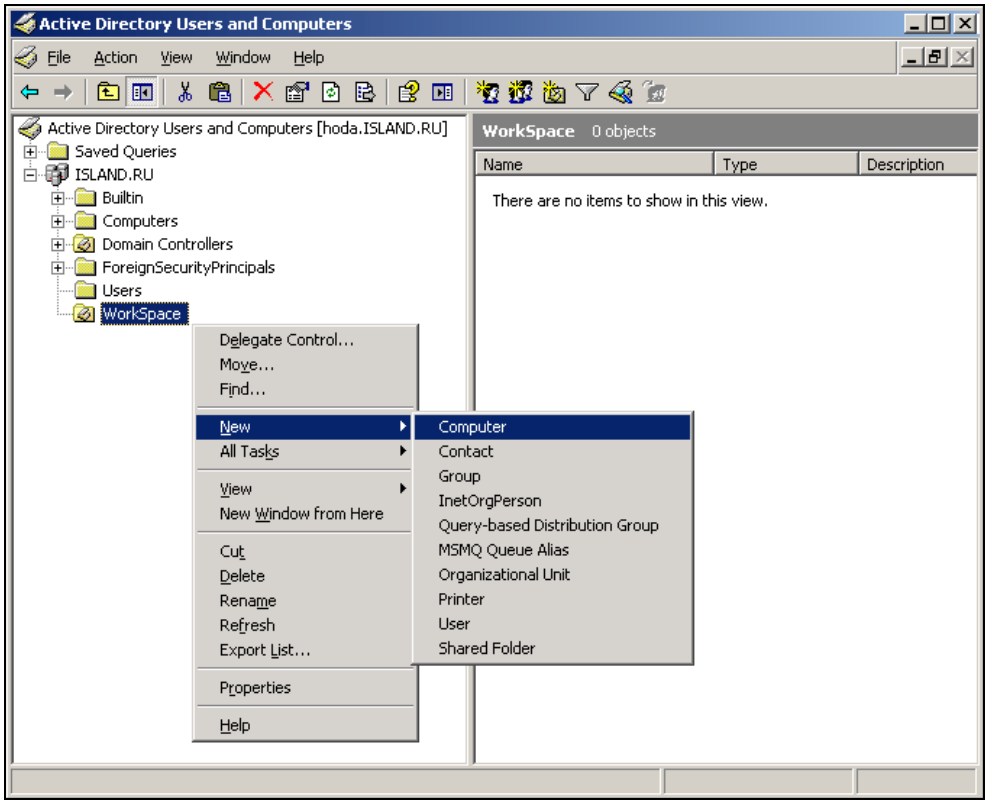



**Рис. 11.5.** Запуск мастера создания учетной записи компьютера

В появившемся диалоговом окне (рис. 11.6) необходимо указать имя компьютера в поле **Computer name**, в данном случае, согласно данным из табл. 11.2, — 1001pc. Значение в поле **Computer name (pre-Windows 2000)** появится автоматически. Его можно изменить в случае необходимости.

Для смены группы, члены которой могут ввести в домен данную рабочую станцию, необходимо нажать на кнопку **Change** и в появившемся диалоговом окне набрать support (рис. 11.6). По умолчанию назначена группа Domain Admins.

В свойствах безопасности учетной записи компьютера на вкладке **Security** добавится группа support, которой будут назначены права **Read** и **Allow to Authenticate**. Вкладка **Security** доступна в свойствах учетной записи MMC-консоли **Active Directory Users and Computers** при включенном расширенном режиме просмотра каталога. Для его включения необходимо войти в меню **View** и установить флажок напротив пункта **Advanced Feature**.

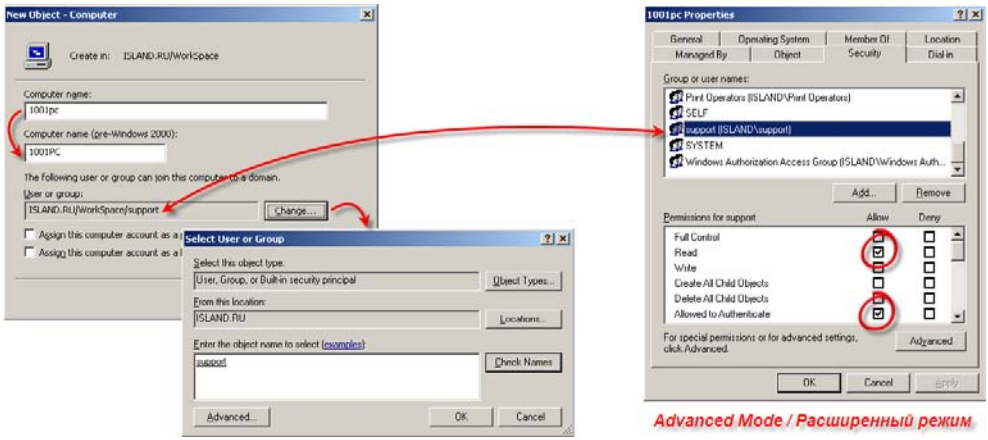



**Рис. 11.6.** Работа мастера. Шаг 1

На этом процесс создания учетной записи компьютера на контроллере домена без дополнительных настроек завершен.

## Программное создание учетной записи компьютера

Во время создания учетной записи компьютера программным способом (VBScript), как и при работе мастера, необходимо жестко задать несколько параметров. В листинге 11.1 приведен сценарий, позволяющий создать учетную запись компьютера. Результат действия сценария аналогичен работе мастера, который был рассмотрен в предыдущем параграфе. В табл. 11.3 приведены параметры создаваемой учетной записи и соответствующие им значения, которые необходимо задать в сценарии.

***Таблица 11.3.*** *Параметры сценария для создания учетной записи компьютера*

| Параметр | Комментарий | Способ назначения |
|---|---|---|
| `cn = 1001PC` | Отображаемое имя | Явный |
| `sAMAccountName = 1001PC$` | Имя в сети для совместимости с доменами Windows NT | Явный |

***Таблица 11.3*** *(окончание)*

| Параметр | Комментарий | Способ назначения |
|---|---|---|
| `userAccountControl = 4218` | Аутентификация пользователя. По умолчанию принимает значение 4218. Если необходимо эмулировать установку флажка в мастере создания напротив надписи **Assign this computer account as a pre-Windows 2000 computer**, то необходимо параметру `userAccount-Control` присвоить значение, равное 8224 | Явный |
| `distinguishedName = CN=1001PC, OU=WorkSpace, DC=island, DC=ru` | Путь к создаваемой учетной записи | Неявный |
| `objectClass = computer` | Тип объекта | Неявный |

**Листинг 11.1. Создание учетной записи пользователя**

```
' Определение RDN-имени (LDAP-имени) домена
Set RootDSE = GetObject("LDAP://RootDSE")
Domain = rootDSE.Get("defaultNamingContext")

' Подключение к контейнеру и создание объекта
Set objUsers = GetObject("LDAP://OU=WorkSpace," & Domain)
Set objNewUser = objUsers.Create("computer", "cn=1001pc")

' Назначение параметров учетной записи
objNewUser.Put "sAMAccountName", "1001PC$"
objNewUser.Put "userAccountControl", 4218

' Запись данных в каталог Active Directory
objNewUser.SetInfo
```

Обратите внимание: значение параметра `sAMAccountName` равно значению параметра `cn` со знаком доллара (`$`) на конце (рис. 11.7).

## Поля учетной записи, создаваемой мастером

После завершения процесса создания учетной записи в Active Directory хранится 30 атрибутов. Часть из них создается и изменяется системой в автоматическом режиме, другую часть администратор задает явным и неявным способом.

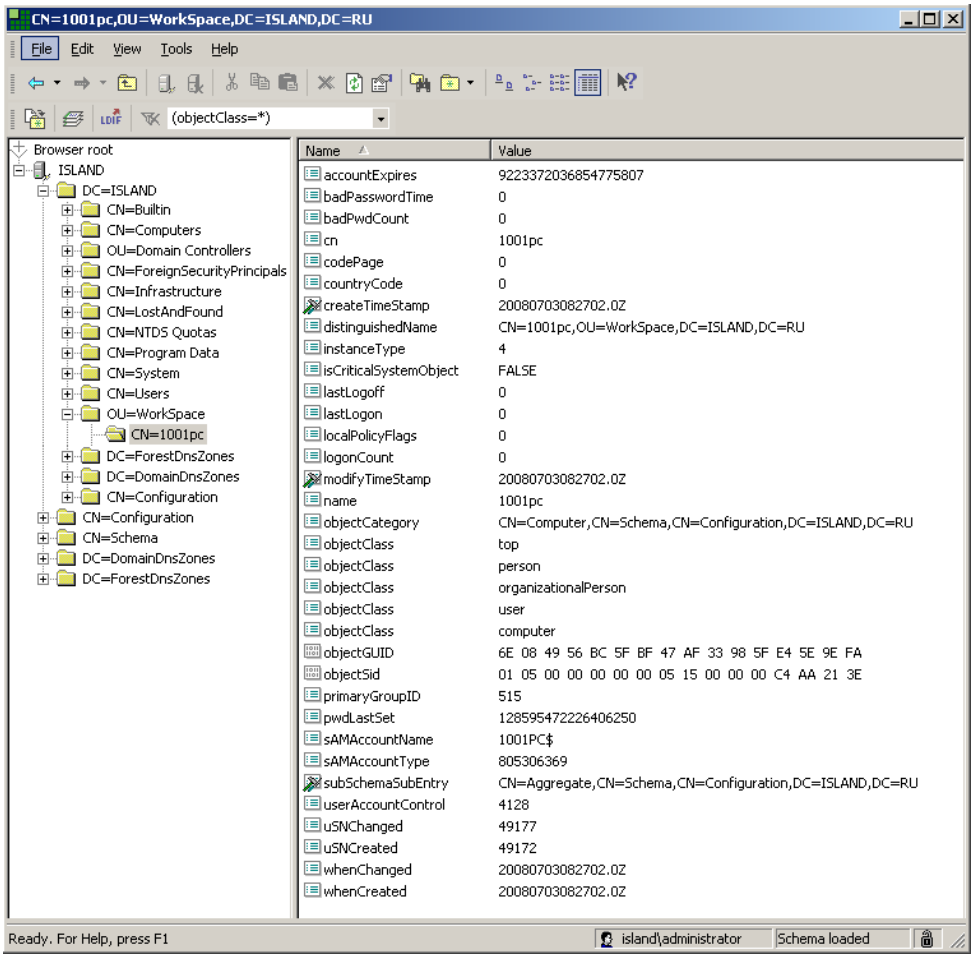



**Рис. 11.7.** Результат назначения имени компьютера

Оставшиеся параметры, отсутствующие на рис. 11.7, можно изменить только после завершения процесса создания учетной записи компьютера. К таким параметрам, например, относятся расположение объекта в каталоге, значение основной группы (primary group) и т. д.

# Удаление компьютера

Рассмотрим удаление учетной записи компьютера программным способом и с помощью мастера.

## Удаление с помощью мастера

Удаление учетной записи компьютера осуществляется в MMC-оснастке Active Directory.

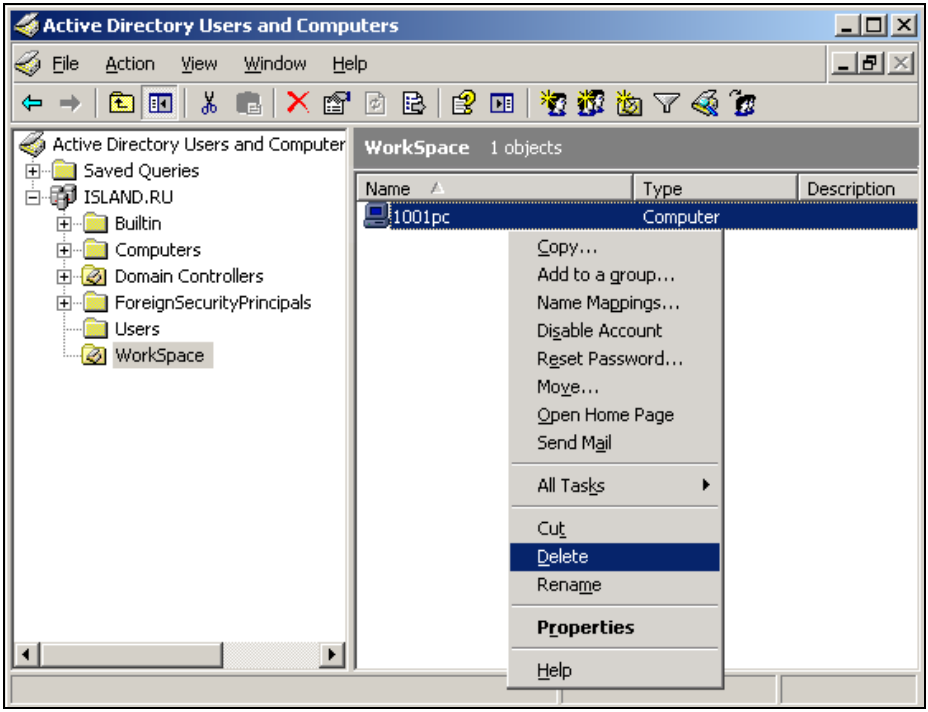



**Рис. 11.8.** Удаление учетной записи компьютера

Чтобы удалить объект из Active Directory, необходимо установить на него курсор и нажать клавишу <Delete> на клавиатуре или, вызвав контекстное меню объекта, выбрать в нем **Delete** (рис. 11.8).

Поскольку объекты удаляются безвозвратно, то для совершения операции запрашивается подтверждение. После удаления объект можно восстановить только из резервной копии.

## Удаление программным способом

В сценарии, который удаляет объект, необходимо указать два параметра: тип объекта и его местоположение. Оба эти параметра содержатся в относительном составном имени объекта — RDN-пути, которому соответствует значение атрибута `distinguishedName` в каталоге Active Directory. В свою очередь, имени объекта соответствует параметр `cn` (рис. 11.9).

Удаление объекта осуществляется вызовом метода `Delete()`. В данном случае метод `SetInfo()` не используется. При попытке его использования интерпретатор VBScript выдаст сообщение об ошибке.

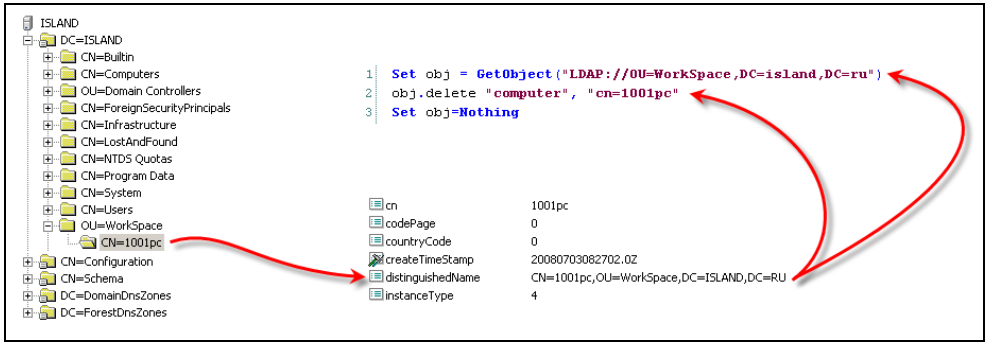



**Рис. 11.9.** Структура параметра `distinguishedName`

# Чтение свойств компьютера

В этом разделе будут рассмотрены параметры, отображаемые на 7 вкладках, которые видны в стандартном режиме просмотра Active Directory. В расширенном режиме просмотра к ним добавляются две вкладки: **Object** и **Security**.

## Вкладка *General*

На вкладке **General** (рис. 11.10) выводятся основные параметры учетной записи компьютера: имя компьютера, роль и т. д. Они рассмотрены в табл. 11.4.

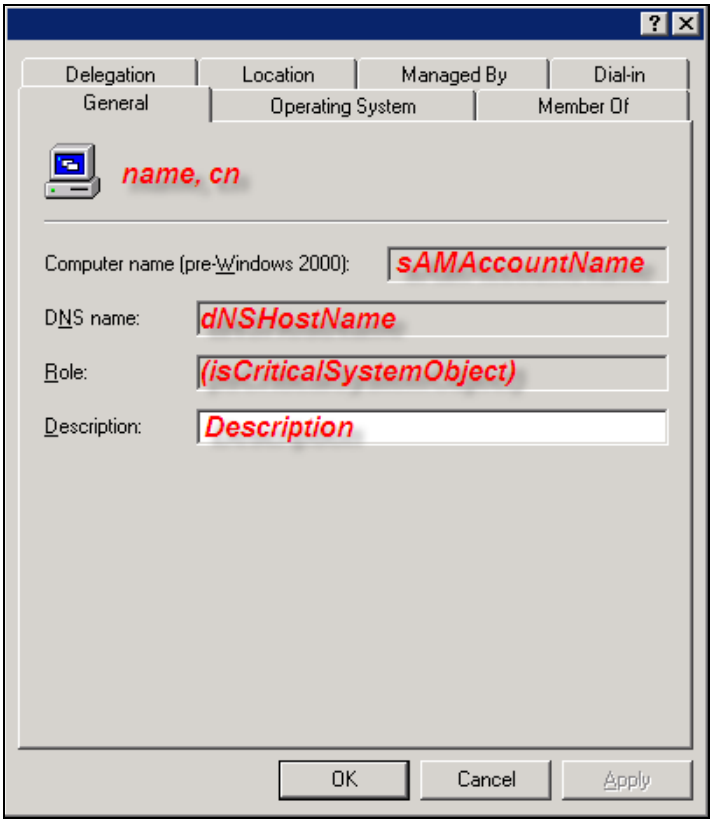



**Рис. 11.10.** Вкладка **General**

***Таблица 11.4.*** *Соответствие полей на вкладке* ***General*** *параметрам в Active Directory*

| **Поле на вкладке *General*** | **Тип** | **Поле в Active Directory** | **Тип** |
|---|---|---|---|
|  | — | `cn`<br>`name` | String |

***Таблица 11.4*** *(окончание)*

| Поле на вкладке *General* | Тип | Поле в Active Directory | Тип |
|---|---|---|---|
| **Computer name (pre-Windows 2000)** | InputBox (только чтение) | `sAMAccountName` | String |
| **DNS name** | InputBox (только чтение) | `dNSHostName` | Boolean |
| **Role** | InputBox (только чтение) | Определяется значением поля `isCriticalSystemName` | String |
| **Description** | InputBox | `description` | String |

## Описание полей

Рассмотрим подробно каждый из отображаемых параметров.

### *Computer name*

Отображаемое имя компьютера (значение атрибута `name` в каталоге Active Directory) выводится рядом со значком компьютера  .

На вкладке **General** его невозможно изменить. В Active Directory этому параметру соответствуют два параметра `name` и `cn`, значения которых совпадают. Для изменения этих параметров необходимо переименовать компьютер.

### *Computer name (pre-Windows 2000)*

В этом неизменяемом поле отображается значение текстового атрибута `sAMAccountName`. Это имя компьютера, совместимое с доменами Windows NT. В связи с этим в каталоге Active Directory хранится значение, в конце которого добавлен знак доллара (`$`). При отображении в поле знак доллара не выводится.

### *DNS name*

После создания учетной записи компьютера в сети данное строковое поле пусто. Ему присваивается значение только после регистрации рабочей стан-

ции в сети. Пример выводимого значения: 1001pc.island.ru. Данное значение невозможно изменить ни при каких условиях.

### Role

В поле **Role** выводится одна из двух ролей компьютера: Workstation or server или Domain controller. В каталоге Active Directory выводимое значение определяется булевым атрибутом `isCriticalSystemName`. Если значение атрибута `False`, то в поле **Role** отображается Workstation or server, если `True` — Domain controller.

### Description

Полю **Description**, единственному доступному для изменения полю на этой вкладке, в каталоге Active Directory соответствует одноименный строковый параметр. В качестве значения обычно указывают служебную информацию, например, номер комнаты, в которой находится рабочая станция.

## Вкладка *Operating System*

На вкладке **Operating System** (рис. 11.11) отображаются параметры, характеризующие установленную на рабочей станции операционную систему. Информацию из данной вкладки рекомендуется использовать для быстрого решения задачи инвентаризации парка компьютеров в сети.

***Таблица 11.5.*** *Соответствие полей на вкладке* ***Operating System*** *атрибутам Active Directory*

| **Поле на вкладке *Operating System*** | **Тип** | **Атрибут Active Directory** | **Тип** |
|---|---|---|---|
| **Name** | InputBox (read only) | `operatingSystem` | String |
| **Version** | InputBox (read only) | `operatingSystemVersion` | String |
| **Service pack** | InputBox (read only) | `operatingSystemServicePack` | String |

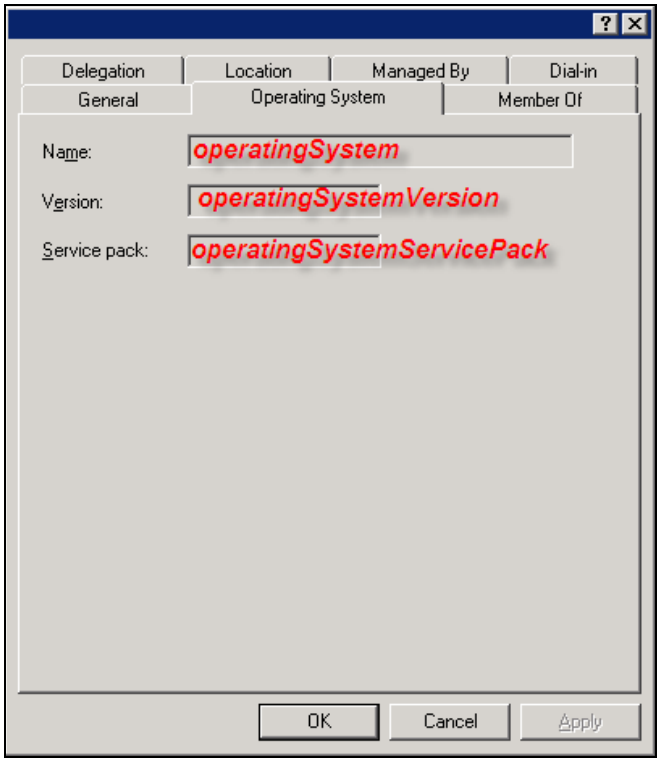



**Рис. 11.11.** Вкладка **Operating System**

## Описание полей

Рассмотрим подробно каждый из показанных параметров.

### *Name*

Значение поля **Name** — версия установленной на клиентской машине операционной системы, например — Windows 2000 Professional. Значение хранится в текстовом атрибуте `operatingSystem`.

### *Version*

Значение поля **Version** — номер версии сборки операционной системы. Например: 5.0 (2195). Значение хранится в текстовом атрибуте `operatingSystemVersion`.

### *Service Pack*

Значение поля **Service Pack** — версия установленного пакета исправлений клиентской операционной системы.

Например: Service Pack 4. Значение хранится в текстовом атрибуте `operatingSystemServicePack`.

## Вкладка *Member Of*

На вкладке **Member Of** формируется список групп (рис. 11.12), членом которых является пользователь, а также назначается Primary Group (основная группа).

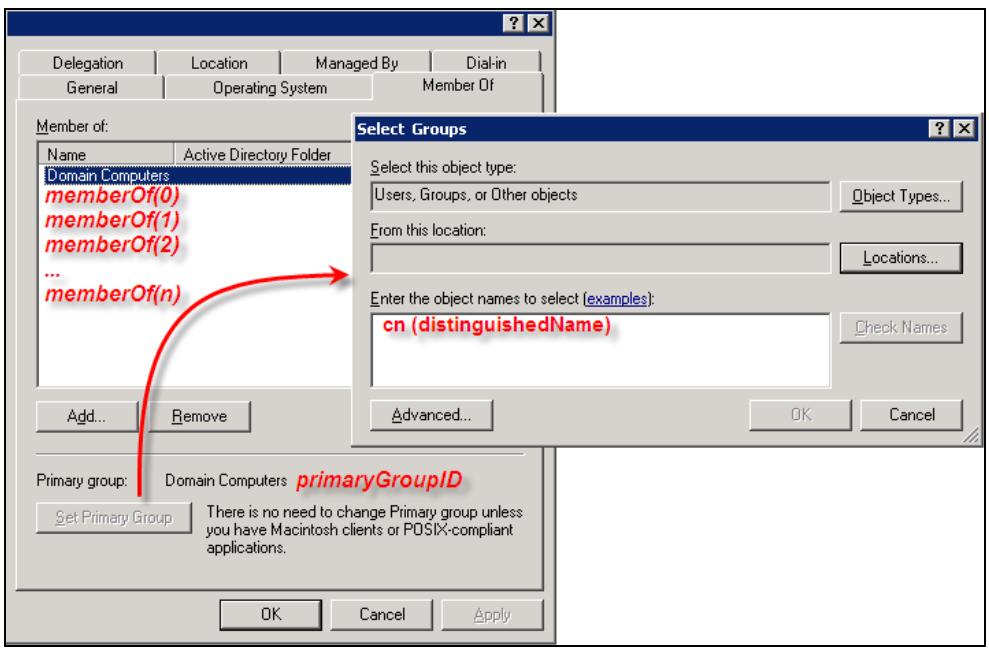



**Рис. 11.12.** Вкладка **Member Of**

Список атрибутов Active Directory, соответствующих полям на вкладке **Member Of**, приведен в табл. 11.6.

***Таблица 11.6.*** *Соответствие полей на вкладке* ***Member Of*** *атрибутам Active Directory*

| Поле на вкладке *Member Of* | Тип | Поле в Active Directory | Тип |
|---|---|---|---|
| **Member of** | ListBox, Button | `memberOf` | Array |
| **Primary group** | Button | `primaryGroupID` | String |

## Описание полей

Опишем отображаемую на вкладке информацию подробнее.

### *Member of*

Для управления членством пользователя в группах безопасности Active Directory используются две кнопки, находящиеся под списком групп: **Add** для добавления и **Remove** для удаления. По умолчанию учетная запись компьютера входит в группу **Domain Computers**, идентификатор которой 515.

Механизм управления группами безопасности следующий. Для добавления учетной записи компьютера в какую-либо группу необходимо нажать кнопку **Add**. В появившемся диалоговом окне (рис. 11.12) осуществляется поиск объектов по заданным критериям. В поле **Enter the object names to select** указывается фрагмент отображаемого имени (поле `name`) или одно из имен группы (`cn` или `sAMAccountName`), при этом в массиве фиксируется значение поля `distinguishedName`, в то время как в списке отображается `cn` этого объекта.

### *Primary Group*

Единственная группа, членом которой является группа после создания его учетной записи — Domain Computers, и она же является основной. Значение параметра `primaryGroupID` учетной записи компьютера — идентификационный номер группы. Для группы Domain Computers параметр `primaryGroupID` равен 515.

Рассмотрим пример. Пусть учетная запись компьютера 1001PC входит в группу Test_Group, которая имеет SID `S-1-5-21-42226584364-21557989-1436132917-12213`. Необходимо определить значение параметра

primaryGroupID, если основная группа — группа Test_Group. Для этого надо посмотреть на последний раздел SID, который и является идентификатором основной группы, поэтому параметр primaryGroupID равен 12213 (рис. 11.13).

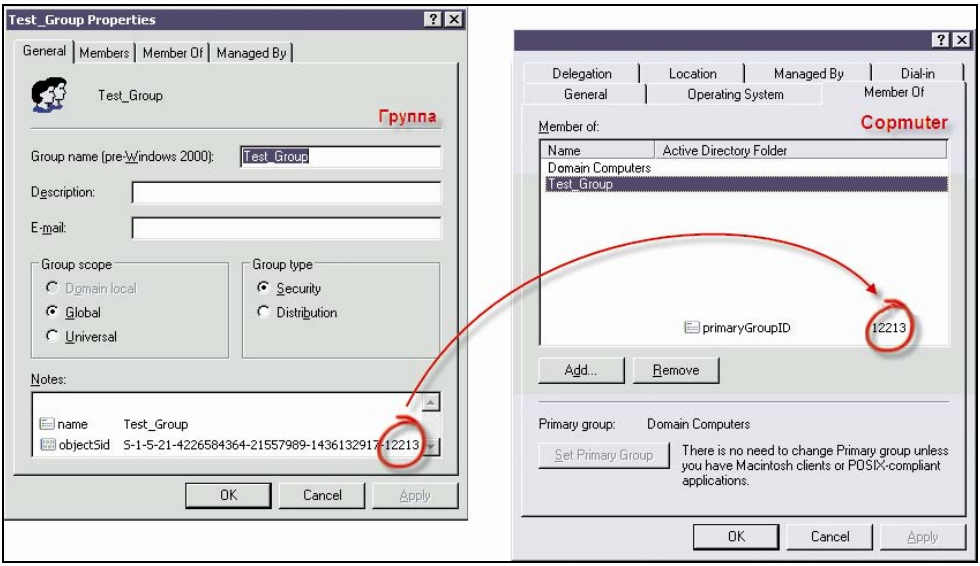



**Рис. 11.13.** Определение значения параметра primaryGroupID

## Вкладка *Delegation*

Вкладка **Delegation** (рис. 11.14) появляется, только если установлен терминальный сервер. В ней настраиваются доверительные отношения компьютера, характеризующиеся числовым атрибутом userAccountControl. Его значение — сумма параметров, характеризующих в целом параметры безопасности данной учетной записи.

Администратор может выбрать один из следующих вариантов, определяющих доступ служб к данной учетной записи компьютера:

- **Do not trust this computer for Delegation** (по умолчанию) — удаленное подключение к данному компьютеру запрещено;

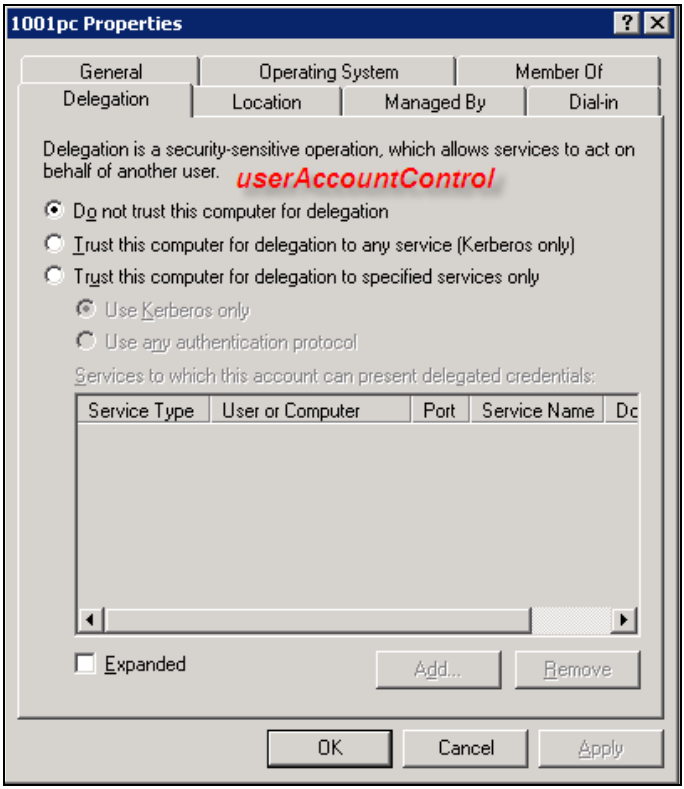



**Рис. 11.14.** Определение значения параметра `userAccountControl` на вкладке **Delegation**

- **Trust this computer for delegation to any service (Kerberos Only)** — разрешить удаленное подключение к компьютеру любым сервисам с помощью Kerberos;
- **Trust this computer for delegation to specified services (Use Kerberos only)** — разрешить удаленное подключение к компьютеру указанным в списке сервисам с помощью протокола аутентификации Kerberos;
- **Trust this computer for delegation to specified services (Use any authentication protocol)** — разрешить удаленное подключение к компьютеру указанным в списке сервисам с помощью любого доступного протокола аутентификации.

Все возможные значения, принимаемые параметром `userAccountControl`, подробно рассмотрены в *главе 8* (см. табл. 8.6 и 8.7).

## Вкладка *Location*

На вкладке **Location** в одноименном поле указывается местоположение учетной записи компьютера (рис. 11.15). Этому полю в каталоге Active Directory соответствует одноименный текстовый атрибут — `location`.

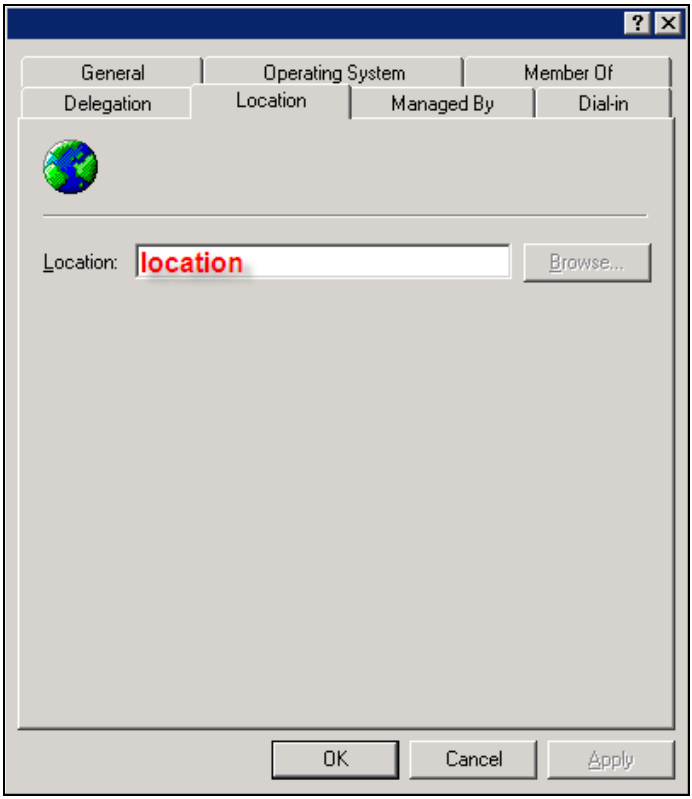



**Рис. 11.15.** Определение значения параметра **Location**

## Вкладка *Managed By*

На вкладке **Managed By** (рис. 11.16) назначается "управляющий" компьютером — учетная запись пользователя, которому даются права для изменения свойств данной учетной записи. В этой вкладке можно изменить всего лишь одно поле: **Name**. В каталоге Active Directory ему соответствует поле

managedBy. Остальные поля недоступны, и в них лишь выводится информация об установленной учетной записи.

Несмотря на то, что в поле **Name** отображается значение поля cn учетной записи пользователя, в каталоге Active Directory значением атрибута managedBy является составной относительный путь к учетной записи пользователя, то есть его distinguishedName.

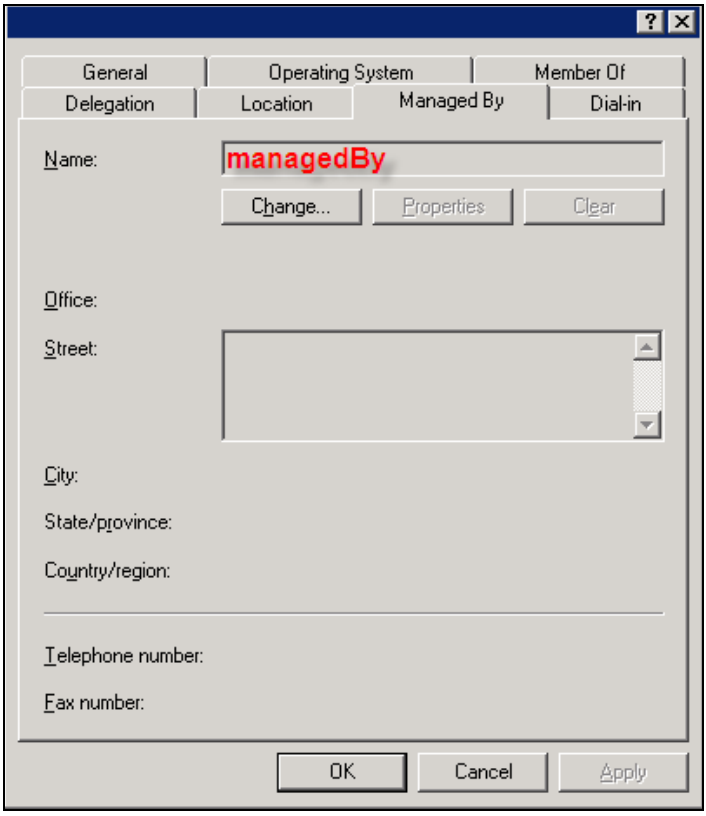



**Рис. 11.16.** Вкладка **Managed By**

## Вкладка *Dial-in*

На вкладке **Dial-in** собраны настройки Dial-up-соединения учетной записи компьютера (рис. 11.17).

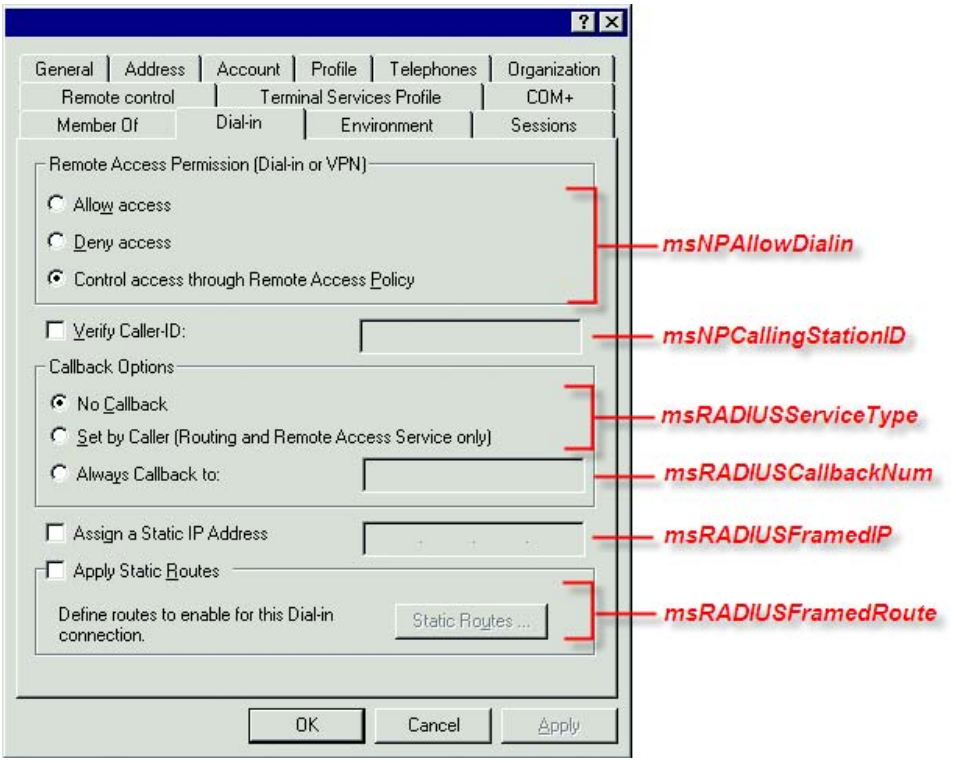



**Рис. 11.17.** Вкладка **Dial-in**

В табл. 11.7 приведены соответствия полей вкладки атрибутам Active Directory.

***Таблица 11.7.*** *Соответствие полей на вкладке* ***Dial-in*** *параметрам в Active Directory*

| **Поле на вкладке *Dial-in*** | **Тип** | **Поле в Active Directory** | **Тип** |
|---|---|---|---|
| **Allow access** | Radio | `msNPAllowDialin` | Boolean |
| **Deny access** | | | |
| **Control access through Remote Access Policy** | | | |

***Таблица 11.7*** *(окончание)*

| Поле на вкладке *Dial-in* | Тип | Поле в Active Directory | Тип |
|---|---|---|---|
| **Verify Caller-ID** | CheckBox | msNPCallingStationID | String |
| **No Callback** | Radio | msRADIUSServiceType | String |
| **Set by Caller** | | | |
| **Always Callback to** | | msRADIUSCallbackNum | String |
| **Assign Static IP Address** | CheckBox | msRADIUSFramedIP | String |
| **Apply Static Routes** | CheckBox | msRADIUSFramedRoute | String |

## Описание полей

Рассмотрим соответствие полей и атрибутов Active Directory более подробно.

### Группа *Remote Access Permission (Dial-in or VPN)*

С помощью группы параметров **Remote Access Permission (Dial-in or VPN)** осуществляется настройка возможности удаленно подключаться к компьютеру по сети. В группе может быть выбран один из следующих параметров:

- **Allow Access** — разрешить удаленные подключения. Этот вариант дает возможность пользователям беспрепятственно подключаться к компьютеру с помощью Dial-up- или VPN-соединений. При этом параметру `msNPAllowDialin` в Active Directory присвоено значение `True`;
- **Deny Access** — запретить удаленные подключения. В этом случае пользователи ни при каких условиях не смогут удаленно подключиться к компьютеру. Соответствующее значение параметра `msNPAllowDialin` равно `False`;
- **Control access through Remote Access Policy** (по умолчанию) — возможность удаленного доступа к компьютеру регулируется с помощью групповой политики Remote Access Policy. Это соответствует значению атрибута `msNPAllowDialin = " "`.

По умолчанию выбран вариант **Control access through Remote Access Policy**.

### Verify Caller-ID

Включив возможность проверки ID подключаемого пользователя к компьютеру, можно создать дополнительную защиту от несанкционированного вмешательства. По умолчанию опция Verify Caller-ID отключена. В Active Directory значение этого параметра хранится в текстовом виде в переменной `msNPCallingStationID`.

### Callback Options

Одной из дополнительных мер по защите от несанкционированного доступа к сети является возможность автоматического перезвона. Таким образом, при попытке пользователя подключиться к компьютеру осуществляется обрыв связи. После этого система сама пытается установить соединение с удаленным компьютером. Если какой-либо способ обратной связи активирован, то злоумышленник не сможет подключиться к сети от имени этого пользователя, поскольку кроме имени и пароля задействован третий параметр — телефон. Группа **Callback Options** позволяет выбрать один из следующих вариантов поведения:

- **No Callback** (выбран по умолчанию) означает, что подключение идет исключительно со стороны компьютера. В Active Directory это соответствует обнуленным параметрам `msRADIUSServiceType` и `msRADIUSCallbackNum`;
- **Set by Caller (Routing and Remote Access Service only)** показывает, что пользователь, пытающийся подключиться к компьютеру, должен выбрать в настройках своего удаленного соединения способ подключения: с включенной функцией обратного звонка или нет. При активации этой функции на клиентском компьютере указывается номер телефона, по которому модем свяжется с компьютером.

  Для выбора этой опции программным путем в Active Directory необходимо обнулить параметр `msRADIUSCallbackNum`, а параметру `msRADIUSServiceType` присвоить значение `4`;
- **Always Callback** включает режим перезванивания, в котором система будет всегда перезванивать удаленному компьютеру по указанному в соседнем поле телефонному номеру. Этот номер сохраняется в атрибуте `msRADIUSCallbackNum`, а значение параметра `msRADIUSServiceType` должно быть обнулено.

### Assign a Static IP Address

Обычно в момент подключения удаленного компьютера служба DHCP динамически выделяет IP-адрес. Если по каким-либо причинам должен быть указан статический адрес подключения, то необходимо активировать параметр **Assign a Static IP Address**. В Active Directory ему соответствует атрибут `msRADIUSFramedIP`, значение которого имеет вид `***.***.***.***`. По умолчанию статический адрес не назначается.

### Apply Static Routes

Для подключения компьютера из другой подсети необходимо произвести маршрутизацию.

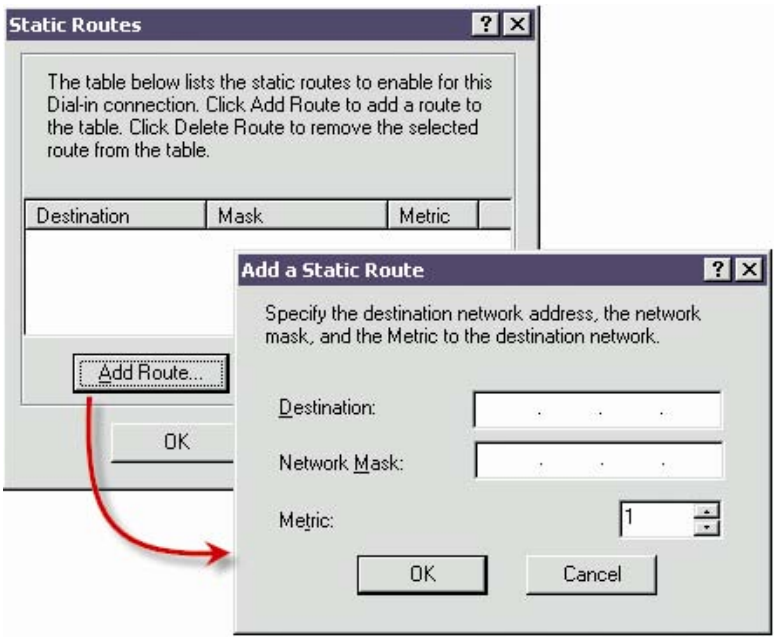



**Рис. 11.18.** Параметры статической маршрутизации

Для этого необходимо активировать группу параметров **Apply Static Routes** и, нажав на кнопку **Static Route** (Статическая маршрутизация), в появившемся окне (рис. 11.18) ввести необходимые значения. Полученные данные сохраняются в поле `msRADIUSFramedRoute`. Элементы массива

msRADIUSFramedRoute представляют собой строки, сформированные по шаблону:

ooo.ooo.ooo.ooo/xxx.xxx.xxx.xxx

В этом шаблоне ooo.ooo.ooo.ooo — подсеть, к которой необходимо преобразовать, а xxx.xxx.xxx.xxx — преобразуемая подсеть.

По умолчанию статическая маршрутизация отключена.

### Резервная копия параметров вкладки *Dial-in*

Все настройки, перечисленные на вкладке **Dial-in**, имеют резервную копию для восстановления, которая также хранится в Active Directory. Каждому полю соответствует дополнительный параметр, имя которого образовано добавлением в него слова Saves и заменой слова RADIUS на RAS (табл. 11.8).

***Таблица 11.8.*** *Соответствие параметров вкладки* ***Dial-in*** *их резервным копиям*

| Значение параметра | Резервная копия |
|---|---|
| msNPAllowDialin | — |
| msNPCallingStationID | msNPSavesCallingStationID |
| msRADIUSServiceType | — |
| msRADIUSCallbackNum | msRASSavesCallbackNum |
| msRADIUSFramedIP | msRASSavesFramedIP |
| msRADIUSFramedRoute | msRASSavesFramedRoute |

При изменении какого-либо параметра сначала текущее значение копируется в соответствующий ему резервный параметр, и лишь затем назначается новое значение.

## Публикация компьютера в домене

Публикация рабочей станции в Active Directory осуществляется с локальной станции. Для этого необходимо на локальной станции войти в свойства объекта My computer (Мой компьютер). Перейдя на вкладку **Computer Name** (Имя компьютера), нажмите на кнопку **Change** (Изменить). В появившемся диалоговом окне (рис. 11.19) необходимо в группе параметров **Member of**

(Членство в) установить режим **Domain** (Домене), и в сделавшемся доступным для редактирования поле ввести сокращенное или DNS-имя домена.

При попытке подключиться к домену контроллер домена запросит имя и пароль учетной записи пользователя, обладающей административными правами. Если эти данные введены верно, будет выведено сообщение об успешной регистрации компьютера в домене и необходимости перезагрузиться, чтобы сделанные в операционной системе изменения вступили в силу.

При таком способе регистрации учетной записи рабочей станции в домене новая учетная запись будет создана в контейнере Computers, находящемся в корневом каталоге Active Directory. Учетные записи контроллеров домена, соответственно, автоматически помещаются в контейнере Domain Controllers.

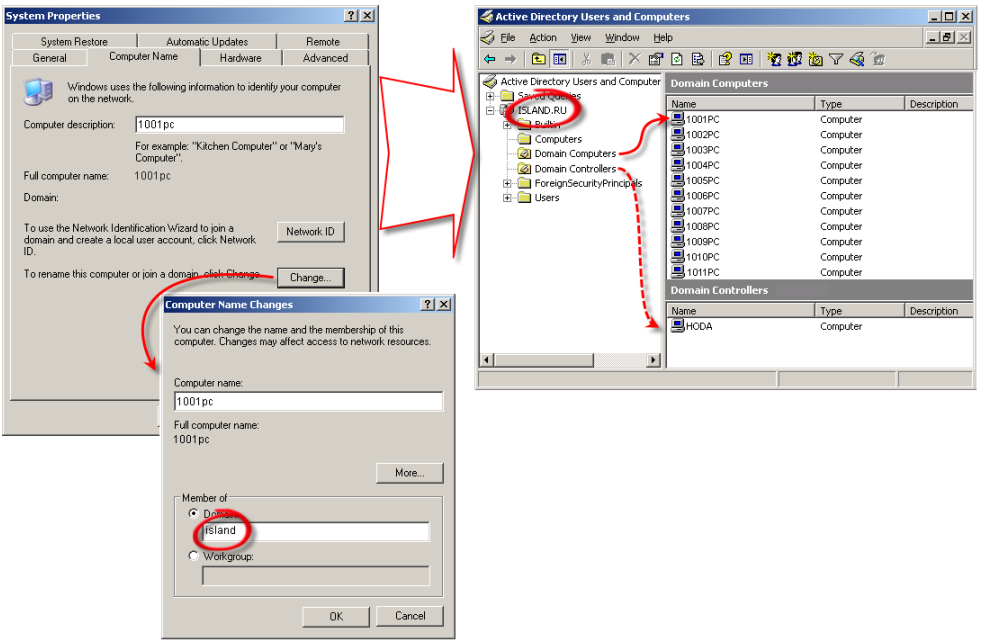



**Рис. 11.19.** Публикация учетной записи компьютера в Active Directory

После регистрации компьютера в Active Directory созданная учетная запись получит ряд атрибутов, значения которых не назначаются при создании объекта с помощью мастера в MMC-консоли или программным способом. На рис. 11.20 эти атрибуты отмечены звездочкой.

| | Name | Value |
|---|---|---|
| | accountExpires | 9223372036854775807 |
| | badPasswordTime | 0 |
| | badPwdCount | 0 |
| | cn | HODA |
| | codePage | 0 |
| | countryCode | 0 |
| | createTimeStamp | 20080523084906.0Z |
| | distinguishedName | CN=HODA,OU=Domain Controllers,DC=ISLAND,DC=RU |
| * | dNSHostName | hoda.ISLAND.RU |
| * | frsComputerReferen... | CN=HODA,CN=Domain System Volume (SYSVOL share),CN=File Replication Service... |
| | instanceType | 4 |
| | isCriticalSystemObject | TRUE |
| | lastLogoff | 0 |
| * | lastLogon | 128600569037656250 |
| | localPolicyFlags | 0 |
| | logonCount | 36 |
| | modifyTimeStamp | 20080623080354.0Z |
| | name | HODA |
| | objectCategory | CN=Computer,CN=Schema,CN=Configuration,DC=ISLAND,DC=RU |
| | objectClass | top |
| | objectClass | person |
| | objectClass | organizationalPerson |
| | objectClass | user |
| | objectClass | computer |
| | objectGUID | AE 7A FA 6F 41 CC 1B 45 85 50 5A 90 6B B2 96 43 |
| | objectSid | 01 05 00 00 00 00 00 05 15 00 00 00 C4 AA 21 3E |
| * | operatingSystem | Windows Server 2003 |
| * | operatingSystemSer... | Service Pack 2 |
| * | operatingSystemVers... | 5.2 (3790) |
| | primaryGroupID | 516 |
| | pwdLastSet | 128586818345781250 |
| * | rIDSetReferences | CN=RID Set,CN=HODA,OU=Domain Controllers,DC=ISLAND,DC=RU |
| | sAMAccountName | HODA$ |
| | sAMAccountType | 805306369 |
| * | serverReferenceBL | CN=HODA,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,... |
| * | servicePrincipalName | ldap/hoda.ISLAND.RU/ForestDnsZones.ISLAND.RU |
| * | servicePrincipalName | ldap/hoda.ISLAND.RU/DomainDnsZones.ISLAND.RU |
| | servicePrincipalName | DNS/hoda.ISLAND.RU |
| | servicePrincipalName | GC/hoda.ISLAND.RU/ISLAND.RU |
| | servicePrincipalName | HOST/hoda.ISLAND.RU/ISLAND |
| | servicePrincipalName | HOST/HODA |
| | servicePrincipalName | HOST/hoda.ISLAND.RU |
| | servicePrincipalName | HOST/hoda.ISLAND.RU/ISLAND.RU |
| | servicePrincipalName | E3514235-4B06-11D1-AB04-00C04FC2DCD2/ff820753-096f-4dd8-867a-2a350eac4... |
| | servicePrincipalName | ldap/ff820753-096f-4dd8-867a-2a350eac4a0a._msdcs.ISLAND.RU |
| | servicePrincipalName | ldap/hoda.ISLAND.RU/ISLAND |
| | servicePrincipalName | ldap/HODA |
| | servicePrincipalName | ldap/hoda.ISLAND.RU |
| | servicePrincipalName | ldap/hoda.ISLAND.RU/ISLAND.RU |
| * | servicePrincipalName | NtFrs-88f5d2bd-b646-11d2-a6d3-00c04fc9b232/hoda.ISLAND.RU |
| | subSchemaSubEntry | CN=Aggregate,CN=Schema,CN=Configuration,DC=ISLAND,DC=RU |
| | userAccountControl | 532480 |
| | uSNChanged | 24580 |
| | uSNCreated | 12290 |
| | whenChanged | 20080623080354.0Z |
| | whenCreated | 20080523084906.0Z |

**Рис. 11.20.** Параметры учетной записи компьютера, зарегистрированного в домене



# Раздел 4

# Безопасность

Глава 12



# Идентификаторы безопасности Active Directory

В этой главе описаны виды идентификаторов безопасности (GUID, SID), принципы их построения и алгоритмы их преобразования из бинарного значения в общепринятый вид.

## Основные термины и понятия

- **GUID** (Globally Unique Identifier) — глобально уникальный 128-битный идентификатор. Его главная особенность — уникальность; общее количество различных ключей настолько велико ($2^{128}$ или примерно $3{,}4028\times10^{38}$), что вероятность генерации двух совпадающих ключей ничтожно мала. GUID — это частная реализация компанией Microsoft стандарта, имеющего название Universally Unique Identifier (UUID).
- **SID** (Security Identifier) — уникальная бинарная структура данных переменной длины, однозначно идентифицирующая объект: учетную запись пользователя, группы, домена, компьютера и т. д.

## Идентификаторы в Active Directory

Из всех видов идентификаторов, существующих в Active Directory, используются только два из них: SID и GUID.

При создании новой учетной записи пользователя или группы в Active Directory ее SID сохраняется в атрибуте объекта `objectSID`. Новому объекту присваивается глобально уникальный идентификатор (GUID), имею-

щий 128-битное значение, неповторяющийся не только в домене, но и во всем мире. Кроме объектов-пользователей и объектов-групп GUID есть у всех объектов, создающихся в Active Directory. GUID каждого объекта хранится в его атрибуте `objectGUID`.

# Globally Unique Identifier

## Структура GUID

GUID — это 16-байтный (128-битный) идентификатор, описанный в стандарте RFC 4122, озаглавленном A Universally Unique IDentifier (UUID) URN Namespace. Условно идентификатор разбивают на 4 части (табл. 12.1), а при записи в текстовом виде последнюю часть разбивают дополнительно еще на две. Это делается для упрощения определения типа идентификатора: первый байт третьего 64-битного раздела (в текстовом виде это как раз 4 символа), определяет тип GUID (табл. 12.2). Рассмотрим GUID, записанный в текстовом виде следующим образом: `d50b151a-02c0-4dd0-a59e-f1fb61614d6b`. Эту запись можно понимать как ряд чисел, записанных в шестнадцатеричной системе счисления, с которой можно ознакомиться в *приложении 3*.

Проанализируем первый байт четвертого блока, а именно a5. Выполнив преобразование из 16-ричной системы в двоичную, получим a5=> 10100101. Таким образом, на основе данных в табл. 12.2, проанализировав первые три символа, приходим к выводу, что это "стандартный" GUID,

***Таблица 12.1.*** *Структура GUID*

| Номер части | Бит | Байт | Количество символов в блоке |
|---|---|---|---|
| 1 | 32 | 4 | 8 |
| 2 | 16 | 2 | 4 |
| 3 | 16 | 2 | 4 |
| 4 | 64 | 8 | 4 |
| | | | 12 |

***Таблица 12.2.*** *Расшифровка значения первого байта последнего блока GUID*

| Значение | Описание |
|---|---|
| 000 | Совместимо с Networking Computing System (NCS) |
| 100 или 101 | Стандарт |
| 110 | Microsoft COM. Сюда также относится очень важный GUID — unknown device (неизвестное устройство) |
| 111 | Зарезервировано |

## Определение типа GUID

В настоящее время существуют 5 версий GUID (табл. 12.3). Номер версии GUID — первое число третьей части GUID:

```
c9802770-e0af-11dc-95ff-0016368d1a02
```

Как видно из таблицы — этот GUID сгенерирован на основе временного штампа.

***Таблица 12.3.*** *Версии GUID*

| Версия | Описание |
|---|---|
| 1 | GUID, созданный на основе временного штампа (time based GUID) |
| 2 | DCE версия безопасности |
| 3 | GUID, основанный на имени (MD5 hash) |
| 4 | GUID, сгенерированный случайным образом |
| 5 | GUID, основанный на имени (SHA-1 hash) |

В основном используются две версии GUID: первая (v.1) и четвертая (v.4).

## Алгоритм создания GUID v.1

В формировании GUID участвуют три составляющие:

- временной штамп;

- временной идентификатор;
- сетевой идентификатор.

### Временной штамп

Временной штамп — это 60-битное значение времени с точностью 100 наносекунд, прошедшего с 1 января 1601 года.

### Временной идентификатор

Идентификатор времени — 16-битное значение, которое генерируется случайным образом, чтобы значения GUID, созданные в мире в один и тот же миг, отличались. Идентификатор времени является дополнительной защитой в обеспечении уникальности созданного.

### Сетевой идентификатор

В качестве третьего уникального числа, участвующего в формировании GUID, используется MAC-адрес (стандарт IEEE 802), известный еще как MAC-адрес используемого сетевого адаптера.

### Правила формирования GUID

Рассмотрим правила построения GUID. Номер складывается из трех частей, которые переплетены друг с другом. Опишем подробнее, как это устроено. Допустим, сейчас 21 февраля 2008 года, 22 часа 03 минуты. Этому времени соответствует 60-битный штамп `1dce0afc9802770`, который в свою очередь разбивается на три части (рис. 12.1). Первая часть состоит из первых трех символов — `1dc`. Вместе с идентификатором GUID (дающим первый символ, см. табл. 12.3) они составляют третий раздел GUID. Первый символ принимает значение от одного до пяти, и единице соответствует GUID, сгенерированный на основе временного штампа. Следующие четыре символа штампа времени составляют второй раздел GUID. Оставшаяся же часть — первый раздел GUID.

Четвертый раздел GUID — временной идентификатор, сгенерированный произвольным образом. В данном случае он равен `95ff`. Как видно, он занимает 16 бит или 2 байта.

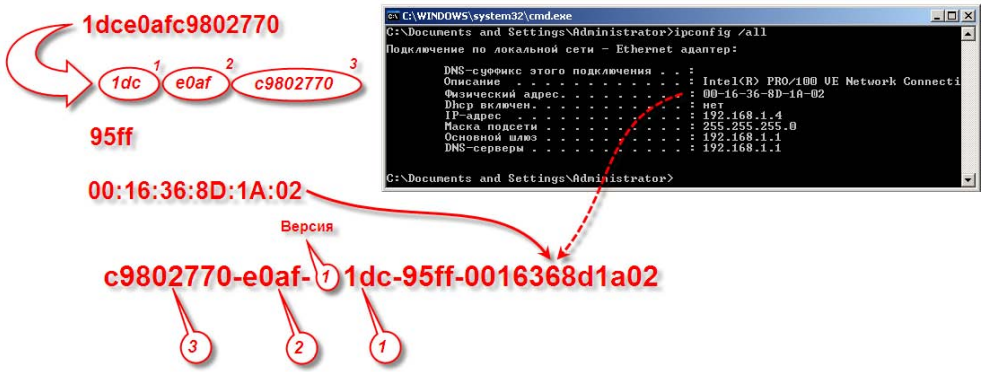



**Рис. 12.1**. Структура GUID

Последний, пятый раздел, — MAC-адрес активного сетевого адаптера. MAC-адрес адаптера легко определить с помощью команды ipconfig /all. В нашем случае это `0016368d1a02`.

Итак, сгенерированный GUID — `c9802770-e0af-11dc-95ff-0016368d1a02`.

## Получение нового GUID

Для создания нового GUID используют утилиту Uuidgen.exe, входящую в состав Microsoft SDK. После установки пакета утилита находится на диске в папке %ProgramFiles%\Microsoft SDK\Bin. После запуска утилита возвращает в консоль сгенерированный GUID.

Для работы утилиты необходим пакет .Net Framework, который сейчас является неотъемлемой частью операционной системы.

Существует альтернативный способ генерации GUID с помощью объекта `Scriptlet.TypeLib`, входящего в состав любой операционной системы Windows, который также позволит сгенерировать GUID (листинг 12.1).

**Листинг 12.1. Генерация GUID (VBScript)**

```
Set TypeLib = CreateObject("Scriptlet.TypeLib")
WScript.Echo TypeLib.Guid
Set TypeLib = Nothing
```

## Особенности глобального идентификатора GUID

Глобальный идентификатор GUID имеет следующие особенности:

- генерируемое значение GUID уникально во всем мире;
- идентификатор не изменяется на протяжении всего времени существования объекта.

# Security Identifier

## Структура SID

Идентификатор безопасности SID состоит из нескольких частей (рис. 12.2). Выражение SID можно записать формулой (12.1):

$$S - R - X - Y_1 - Y_2 - \ldots - Y_{n-1} - Y_n . \qquad (12.1)$$

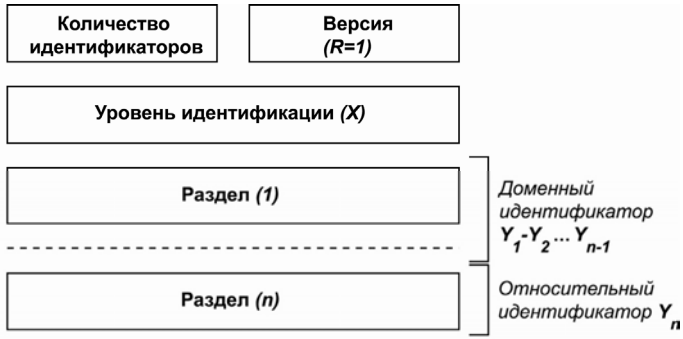



**Рис. 12.2.** Структура SID

Каждая из букв несет в себе смысловую нагрузку:

- $S$ — идентифицирует SID. Идентификатор всегда начинается с этой буквы;
- $R$ — указывает на версию структуры идентификатора. Поскольку SID никогда не может быть изменен, этот параметр всегда равен 1;
- $X$ — показывает наивысший уровень идентификации, которым обладает созданный объект. Значение этого параметра варьируется от 0 до 5 (табл. 12.4). Например, параметр $X$ в SID любой группы безопасности

или учетной записи пользователя в домене имеет значение 5. Исключение составляет группа **Everyone** (Все) — для нее $X = 0$;

- $Y_1 - Y_2 - \ldots - Y_{n-1}$ — идентификатор домена;
- $Y_n$ — относительный идентификатор (RID, Relative ID), который показывает порядковый номер объекта с момента создания Active Directory. Для встроенных объектов зарезервирован диапазон RID от 0 до 999. Например, для администратора это 500, для гостя — 501. Первой учетной записи, созданной в любой локальной системе или домене NT/2k, присваивается RID 1000, а каждому последующему объекту — следующий за ним порядковый номер (1001, 1002, 1003 и т. д.), причем даже при удалении объекта его номер уже никогда не используется при создании новых объектов.

***Таблица 12.4.*** *Расшифровка уровней идентификации SID*

| Идентификатор | Значение |
|---|---|
| SECURITY_NULL_SID_AUTHORITY | 0 |
| SECURITY_WORLD_SID_AUTHORITY | 1 |
| SECURITY_LOCAL_SID_AUTHORITY | 2 |
| SECURITY_CREATOR_SID_AUTHORITY | 3 |
| SECURITY_NON_UNIQUE_AUTHORITY | 4 |
| SECURITY_NT_AUTHORITY | 5 |

## Чтение SID из Active Directory

В Active Directory SID хранится в атрибуте `objectSID` как массив, состоящий из бинарных элементов (рис. 12.3). Размер любого SID в Active Directory — 28 байт. Рассмотрим процесс преобразования бинарной записи идентификатора безопасности в традиционный (строковый) вид. Для лучшего понимания этого раздела вам может быть полезно ознакомиться с различными системами счисления, описанными в *приложении 3*.

В бинарной форме идентификатор безопасности можно записать в общем виде так:

$$A - B - C - Z_1 - Z_2 - \ldots - Z_{n-1} - Z_n . \qquad (12.2)$$

Рассмотрим преобразование идентификатора безопасности из бинарного вида в строку на примере доменной учетной записи Test (см. рис. 12.3).

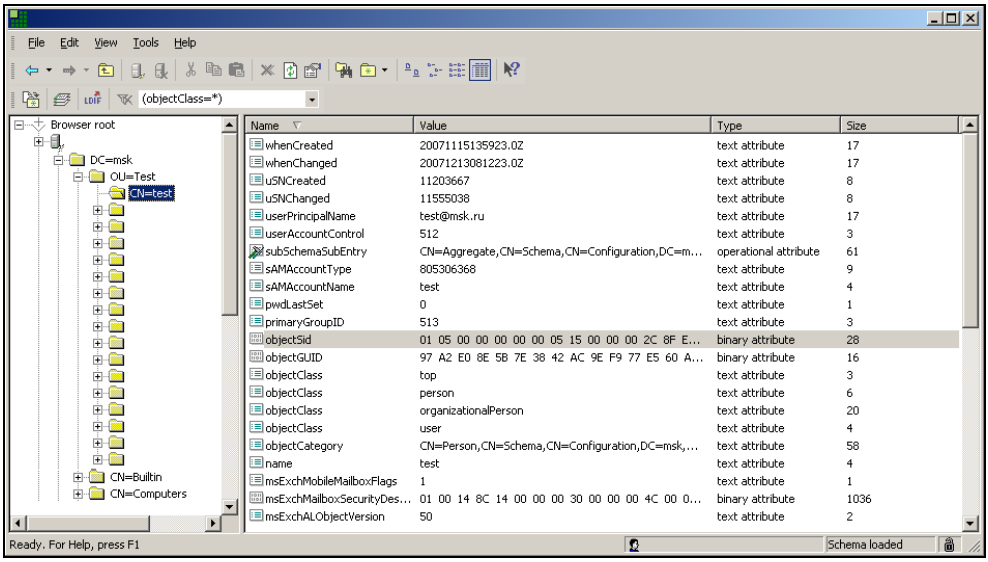



**Рис. 12.3.** SID в Active Directory

Каждый элемент массива — это 1 байт или 8 бит. Полностью SID, часть которого можно увидеть на рис. 12.3, равен `01 05 00 00 00 00 00 05 15 00 00 00 2C 8F EC FB E5 F2 48 01 35 A6 99 55 26 1B 00 00`.

Общая структура SID и алгоритм преобразования приведены в табл. 12.5. Опишем эту процедуру подробнее. Прежде всего, для преобразования массив, представляющий SID, разбивают на группы по 4 байта. Исключение составляют первые 8 байт. Они разбиваются на три группы: первые две по байту и оставшиеся 6 байт. Разбиение массива показано[1] в формуле (12.3). Такое разбиение обусловлено форматом традиционной формы записи, описанным в формуле (12.1).

$$\begin{array}{l} \underline{\underline{01}} \\ \underline{\underline{05}} \\ \underline{\underline{00\ 00\ 00\ 00\ 00\ 05}} \\ \underline{15\ 00\ 00\ 00} \\ \underline{2C\ 8F\ EC\ FB} \\ \underline{E5\ F2\ 48\ 01} \\ \underline{35\ A6\ 99\ 55} \\ \underline{26\ 1B\ 00\ 00} \end{array} \qquad (12.3)$$

[1] В формуле (12.3) первые 8 байт, распределенные нестандартным образом, подчеркнуты два раза. Остальная информация разделена на группы по 4 байта. Каждая из них подчеркнута одинарной линией.

В первом байте хранится версия структуры идентификатора. В формуле (12.1) это параметр $R$, в формуле (12.2) — $A$. Как уже отмечалось ранее, он всегда равен 1.

Во втором байте (табл. 12.5) хранится количество групп по 4 байта, которые следуют за первыми тремя группами. В строковом представлении идентификатора безопасности это число явным образом не присутствует.

В следующих 6 байтах (48 битах) хранится значение наивысшего уровня идентификации (см. табл. 12.4 для расшифровки значений этого параметра).

Преобразование оставшихся групп осуществляется по следующему принципу. Каждая группа перезаписывается с конца в шестнадцатеричной форме. Для нашего примера это сделано в формуле (12.4).

```
0 x 01
0 x 05
0 x 00 00 00 00 00 05
0 x 00 00 00 15
0 x FB EC 8F 2C
0 x 01 48 F2 E5
0 x 55 99 A6 35
0 x 00 00 1B 26
```
(12.4)

Перед записью добавляют символ S, свидетельствующий о том, что это идентификатор безопасности. Второй байт исключают, поскольку он несет лишь информационную нагрузку о количестве идентификационных групп. Далее полученные числа преобразуются из шестнадцатеричной системы счисления в десятичную, как показано в формуле (12.5). На этом преобразование закончено.

```
S – 1 – 5 – 21 – 4226584364 – 21557989 – 1436132917 – 6950
```
(12.5)

***Таблица 12.5.*** *Поэтапное преобразование SID*

| Бинарный вид | Канонический (строковый) вид | Положение в бинарной записи | Комментарий |
|---|---|---|---|
| 01 | S-1 | 1-й байт (1 байт) | К идентификатору добавляется признак SID — S |
| 05 | 5 разделов (0x5) | 2-й байт (1 байт) | Не участвует в формуле SID. Несет информационную нагрузку о количестве идентификационных групп |

***Таблица 12.5*** *(окончание)*

| Бинарный вид | Канонический (строковый) вид | Положение в бинарной записи | Комментарий |
|---|---|---|---|
| 00 00 00 00 00 05 | 05 (0x00000005) | 3—8 байты (6 байт) | Уровень идентификации SID, расшифрованный в табл. 5.4 |
| 15 00 00 00 | 21 (0x00000015) | 9—12 байты (4 байта) | Перед преобразованием из шестнадцатеричной системы записывают байты в обратном порядке |
| 2C 8F EC FB | 4226584364 (0xFBEC8F2C) | 13—16 байты (4 байта) | |
| E5 F2 48 01 | 21557989 (0x0148F2E5) | 17—20 байты (4 байта) | |
| 35 A6 99 55 | 1436132917 (0x5599A635 ) | 21—24 байты (4 байта) | |
| 26 1B 00 00 | 6950 (0x00001B26) | 25—28 байты (4 байта) | |

## Определение SID пользователя с помощью утилиты *GetSID*

Чтобы узнать SID пользователя, можно воспользоваться утилитой GetSID, входящей в состав Windows 2000 Resource Kit Tools. Утилиту можно загрузить с сайта Microsoft: **http://download.microsoft.com/download/win2000platform/Getsid/1.0/NT5/EN-US/getsid.exe**.

В первую очередь утилита GetSID.exe предназначена для сравнения SID разных пользователей, однако она может с успехом использоваться и для получения SID конкретного пользователя.

После завершения процесса установки утилиту и сопутствующую документацию можно найти в каталоге %ProgramFiles%\Resource Kit.

Утилита GetSID.exe запускается из командной строки и имеет следующий синтаксис:

```
GetSID.exe \\server1 account \\server2 account
```

Для определения SID рекомендуется задать две одинаковые пары параметров: сервер и имя учетной записи в сети. Если необходимо определить SID

доменной учетной записи, то в качестве сервера следует указать либо имя контроллера домена, либо имя домена в сокращенной форме.

Например, необходимо определить SID пользователя test в домене `MSK`. Для этого выполним команду:

```
GetSID.exe \\MSK test \\MSK test
```

В результате выполнения этой команды на экран будет выведено сообщение, показанное на рис. 12.4.

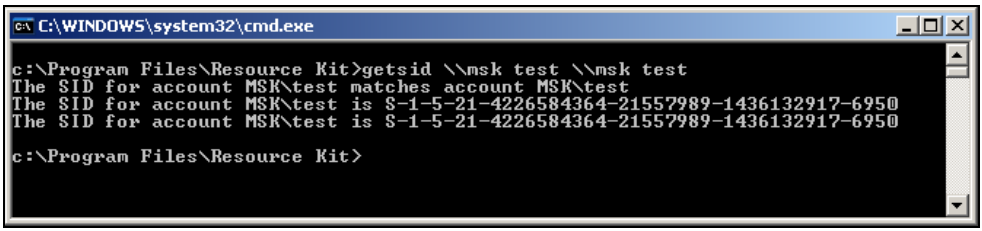



**Рис. 12.4.** Определение SID пользователя с помощью утилиты GetSID.exe.

## Особенности SID

Идентификатор безопасности имеет несколько очень важных особенностей:

- каждый объект имеет персональный идентификатор;
- изменить SID объекта в пределах одного домена невозможно;
- идентификаторы удаленных объектов никогда более не используются;
- SID встроенной доменной учетной записи (табл. 12.6) или Windows одни и те же вне зависимости от версии продукта.

При взаимодействии между объектами в Active Directory и Windows идентификация объектов осуществляется с помощью SID. При переименовании объекта идентификатор безопасности остается прежним. При создании объекта ему присваивается новый SID. Таким образом, если доменная учетная запись пользователя User1 переименована в User2, то с точки зрения Active Directory у объекта изменено только свойство, поскольку SID остался неизменным. Если же пользователь был удален, а затем создан заново с тем же именем, то с точки зрения Active Directory появился новый объект. Старый безвозвратно удален.

## Широко известные идентификаторы SID

Под широко известными SID (Well-known SID) понимают группу идентификаторов безопасности, идентифицирующую общих пользователей и общие группы безопасности. Их значения одинаковы во всех операционных системах. Они перечислены в табл. 12.6.

***Таблица 12.6.*** *Широко известные идентификаторы*

| SID | Объект | Комментарий |
|---|---|---|
| S-1-0 | Null Authority | Защитный код |
| S-1-0-0 | Nobody | Нет участника безопасности |
| S-1-1 | World Authority | Глобальный источник идентификатора безопасности |
| S-1-1-0 | Everyone | Группа, в которую входят все пользователи, даже анонимные пользователи и гости. Принадлежность контролируется операционной системой |
| S-1-2 | Local Authority | Источник идентификатора безопасности |
| S-1-3 | Creator Authority | |
| S-1-3-0 | Creator Owner | Замещающий элемент в наследуемой записи управления доступом (ACE). При наследовании ACE система замещает этот SID идентификатором SID создателя объекта |
| S-1-3-1 | Creator Group | Замещающий элемент в наследуемой записи управления доступом (ACE). При наследовании ACE система замещает этот SID идентификатором SID основной группы создателя объекта. Основная группа используется только подсистемой POSIX |
| S-1-3-2 | Creator Owner Server | Появился в Windows 2003 Server |
| S-1-3-3 | Creator Group Server | |

***Таблица 12.6*** *(продолжение)*

| SID | Объект | Комментарий |
|---|---|---|
| S-1-4 | Non-unique Authority | Источник идентификатора безопасности |
| S-1-5 | NT Authority | |
| S-1-5-1 | Dialup | Группа, в которую входят все пользователи, вошедшие в систему с использованием удаленного доступа. Принадлежность контролируется операционной системой |
| S-1-5-2 | Network | Группы, в которые входят все пользователи, вошедшие в систему с использованием сетевого подключения. Принадлежность контролируется операционной системой |
| S-1-5-3 | Batch | |
| S-1-5-4 | Interactive | |
| S-1-5-5-X-Y | Logon Session | Сеанс входа в систему. Значения X и Y для этих идентификаторов SID меняются в каждом сеансе |
| S-1-5-6 | Service | Группы, в которые входят все участники безопасности, вошедшие в систему в качестве службы. Принадлежность контролируется операционной системой |
| S-1-5-7 | Anonymous | |
| S-1-5-8 | Proxy | Появился в Windows 2003 Server |
| S-1-5-9 | Enterprise Domain Controllers | Группа, в которую входят все контроллеры доменов в лесу, в котором используется служба каталога Active Directory. Принадлежность контролируется операционной системой |
| S-1-5-10 | Principal Self | Замещающий элемент в наследуемом элементе управления доступом (ACE) на объекте учетной записи или объекте группы в Active Directory. При наследовании ACE система замещает этот SID идентификатором SID соответствующей учетной записи объекта |
| S-1-5-11 | Authenticated Users | Группа, в которую входят все пользователи, идентификаторы которых были проверены при входе в систему. Принадлежность контролируется операционной системой |

***Таблица 12.6*** *(продолжение)*

| SID | Объект | Комментарий |
|---|---|---|
| S-1-5-12 | Restricted Code | Данный SID зарезервирован для использования в будущем |
| S-1-5-13 | Terminal Server Users | Группа, в которую входят все пользователи, вошедшие в систему через сервер службы терминалов. Принадлежность контролируется операционной системой |
| S-1-5-18 | Local System | Учетная запись службы, используемая операционной системой |
| S-1-5-19 | NT Authority | Локальная служба |
| S-1-5-20 | NT Authority | Сетевая служба |
| S-1-5-*domain*-500 | Administrator | Учетная запись администратора системы. По умолчанию только эта запись обеспечивает полный контроль системы |
| S-1-5-*domain*-501 | Guest | Учетная запись для лиц, не имеющих индивидуальной учетной записи. Для данной учетной записи пароль не требуется. По умолчанию учетная запись Guest отключена |
| S-1-5-*domain*-502 | KRBTGT (Kerberos Ticket Granting Ticket) | Учетная запись, используемая службой Центр распространения ключей (Key Distribution Center, KDC) |
| S-1-5-*domain*-512 | Domain Admins | Глобальная группа, членам которой разрешено управлять доменом. По умолчанию группа Domain Admins является членом группы Administrators на всех компьютерах, входящих в домен, включая контроллеры доменов. Группа Domain Admins по умолчанию является владельцем любого объекта, созданного любым ее членом |
| S-1-5-*domain*-513 | Domain Users | Глобальная группа, в которую по умолчанию входят все учетные записи домена. При создании учетной записи в домене она по умолчанию добавляется в эту группу |
| S-1-5-*domain*-514 | Domain Guests | Глобальная группа, в которую по умолчанию входит только один член — встроенная учетная запись гостя, соответствующая данному домену |

***Таблица 12.6*** *(продолжение)*

| SID | Объект | Комментарий |
|---|---|---|
| S-1-5-*domain*-515 | Domain Computers | Глобальная группа, в которую входят все клиентские и серверные компьютеры, входящие в домен |
| S-1-5-*domain*-516 | Domain Controllers | Глобальная группа, в которую входят все контроллеры данного домена. По умолчанию новые контроллеры домена добавляются в эту группу |
| S-1-5-*domain*-517 | Cert Publishers | Глобальная группа, в которую входят все компьютеры, на которых работает центр сертификации предприятия. Издателям сертификатов разрешено публиковать сертификаты для объектов (пользователей) в Active Directory |
| S-1-5-*root domain*-518 | Schema Admins | Универсальная группа в домене с основным режимом; глобальная группа в домене со смешанным режимом. Данной группе разрешено вносить изменения в схему в Active Directory. По умолчанию единственным членом группы является учетная запись Administrators для корневого домена леса |
| S-1-5-*root domain*-519 | Enterprise Admins | Универсальная группа в домене с основным режимом; глобальная группа в домене со смешанным режимом. Данной группе разрешено вносить изменения на уровне леса в Active Directory, например, добавлять дочерний домен. По умолчанию единственным членом группы является учетная запись Administrators для корневого домена леса |
| S-1-5-*domain*-520 | Group Policy Creator Owners | Глобальная группа, которой разрешено создавать новые объекты групповой политики в Active Directory. По умолчанию единственным членом группы является Administrators |
| S-1-5-*domain*-533 | RAS and IAS Servers | Локальная группа домена. Серверы этой группы имеют ограничения на чтение учетных записей и доступ к чтению информации о входе в систему для объектов Domain User в локальной группе домена Active Directory. По умолчанию в этой группе нет членов |

***Таблица 12.6*** *(продолжение)*

| SID | Объект | Комментарий |
|---|---|---|
| S-1-5-32-544 | Administrators | Встроенная группа. После первоначальной установки операционной системы единственным членом этой группы является учетная запись Administrator. Когда компьютер присоединяется к домену, группа Domain Admins добавляется к группе Administrators. Когда сервер становится контроллером домена, группа Enterprise Admins также добавляется к группе Administrators |
| S-1-5-32-545 | Users | Встроенная группа. После первоначальной установки операционной системы единственным членом этой группы является группа Allow Authenticated. Когда компьютер присоединяется к домену, группа Domain Users добавляется к группе Users на этом компьютере |
| S-1-5-32-546 | Guests | Встроенная группа. По умолчанию единственным членом группы является учетная запись Guest. Группа Guests предоставляет возможность периодическим или однократным пользователям входить в систему с ограниченными правами встроенной в компьютер учетной записи Guest |
| S-1-5-32-547 | Power Users | Встроенная группа. По умолчанию в этой группе нет членов. Членам группы Power Users разрешено создавать локальных пользователей и группы; изменять и удалять созданные ими учетные записи; удалять пользователей из групп Power Users, Users и Guests. Также членам группы Power Users разрешается устанавливать программы; создавать, удалять локальные принтеры и управлять ими; создавать и удалять общие файловые ресурсы |
| S-1-5-32-548 | Account Operators | Встроенная группа, существующая только на контроллерах доменов. По умолчанию в этой группе нет членов. |

***Таблица 12.6*** *(продолжение)*

| SID | Объект | Комментарий |
|---|---|---|
| | | По умолчанию членам группы Account Operators разрешено создавать, изменять и удалять учетные записи пользователей, групп и компьютеров во всех контейнерах и подразделениях Active Directory, за исключением контейнера Builtin и подразделения Контроллеры домена. Членам группы Account Operators не разрешено ни вносить изменения в группы Administrators и Domain Admins, ни изменять учетные записи членов этих групп |
| S-1-5-32-549 | Server Operators | Встроенная группа, существующая только на контроллерах доменов. По умолчанию в этой группе нет членов. Членам группы Server Operators разрешается входить на сервер в интерактивном режиме; создавать и удалять общие сетевые ресурсы; запускать и останавливать службы; делать резервные копии файлов и восстанавливать их; форматировать жесткий диск компьютера; завершать работу компьютера |
| S-1-5-32-550 | Print Operators | Встроенная группа, существующая только на контроллерах доменов. По умолчанию единственным членом является группа Domain Users. Членам группы Print Operators разрешено управлять принтерами и очередями документов |
| S-1-5-32-551 | Backup Operators | Встроенная группа. По умолчанию в этой группе нет членов. Членам группы Backup Operators разрешено делать резервные копии всех файлов на компьютере и восстанавливать их независимо от разрешений, защищающих эти файлы. Также членам группы Backup Operators разрешается входить в систему и завершать работу компьютера |
| S-1-5-32-552 | Replicators | Встроенная группа, использующаяся службой репликации файлов на контроллерах доменов. По умолчанию в этой группе нет членов. Запрещается добавлять пользователей в эту группу |

***Таблица 12.6*** *(окончание)*

| SID | Объект | Комментарий |
|---|---|---|
| S-1-5-32-554 | BUILTIN\Pre-Windows 2000 Compatible Access | Псевдоним, добавленный операционной системой Windows 2000. Группа для обратной совместимости, предоставляющая право чтения информации обо всех пользователях и группах домена |
| S-1-5-32-555 | BUILTIN\Remote Desktop Users | Псевдоним. Членам этой группы обеспечивается право входа в систему в удаленном режиме |
| S-1-5-32-556 | BUILTIN\Network Configuration Operators | Псевдоним. Члены этой группы могут иметь некоторые привилегии администратора для управления настройкой возможностей работы в сети |
| S-1-5-32-557 | BUILTIN\Incoming Forest Trust Builders | Псевдоним. Членам этой группы разрешается создавать входящие доверительные отношения в лесу, в котором они находятся |
| S-1-5-32-557 | BUILTIN\Incoming Forest Trust Builders | Псевдоним. Членам этой группы разрешается создавать входящие односторонние доверительные отношения в лесу, в котором они находятся |
| S-1-5-32-558 | BUILTIN\ Performance Monitor Users | Псевдоним. У членов этой группы имеется удаленный доступ к монитору данного компьютера |
| S-1-5-32-559 | BUILTIN\ Performance Log Users | Псевдоним. У членов этой группы имеется удаленный доступ к планированию регистрации в журналах данных счетчиков производительности на этом компьютере |
| S-1-5-32-560 | BUILTIN\ Windows Authorization Access Group | Псевдоним. У членов этой группы имеется доступ к вычисленному атрибуту `tokenGroupsGlobalAndUniversal` на объектах Domain User |
| S-1-5-32-561 | BUILTIN\ Terminal Server License Servers | Псевдоним. Группа для серверов лицензирования серверов терминалов |

Глава 13



# Программное управление безопасностью файловой системы

В этой главе рассказано о существующих файловых системах, их достоинствах и недостатках. Особое внимание уделено файловой системе NTFS, проблемам безопасности и программному управлению доступом к ресурсам.

## Основные термины и понятия

- **Файловая система** — часть операционной системы, обеспечивающая запись и чтение файлов на дисковых носителях. Определяет логическую и физическую структуру файла, идентификацию и сопутствующие данные файла.
- **Файл подкачки** (Paging file) — специальный файл (Pagefile.sys), размещающийся в корневом каталоге одного или нескольких дисков компьютера и предназначенный для (временного) хранения страниц виртуальной памяти.
- **ABE** (Access-Based Enumeration) — технология, позволяющая отображать пользователю содержимое сетевых ресурсов, на которые он имеет права (чтение и выше).
- **ACE** (Access Control Entry) — запись управления доступом, то есть правило, которое задает разрешение или запрет на определенные действия с объектом.
- **ACL** (Access Control List) — упорядоченный список записей управления доступом.

- **FAT** (File Allocation Table) — таблица, содержащая сведения о расположении всех файлов на диске. Таблица расположена на диске, состояние которого она описывает.
- **MFT** (Master File Table) — таблица метаданных, в которой хранится информация о содержании томов NTFS.
- **NTFS** (New Technology File System) — файловая система, предназначенная для использования в операционных системах Windows NT и Windows 2k.
- **DACL** (Discretionary Access Control List) — список управления доступом, который указывает группы или индивидуальных пользователей, которым разрешен доступ к объектам.
- **SACL** (System Access Control List) — системный список управления доступом, указывающий условия, при которых генерируются события безопасности.

# Виды файловых систем

Файловая система является важной частью любой операционной системы, которая отвечает за организацию хранения и доступа к информации на каких-либо носителях. На сегодняшний день существуют следующие файловые системы на жестком диске:

- FAT (File Allocation Table);
- HPFS (High Performance File System);
- VFAT (Virtual FAT);
- FAT32 (File Allocation Table);
- NTFS (New Technology File System).

Существует также множество других файловых систем, применяющихся в различных клонах UNIX и в менее распространенных операционных системах, но Windows не поддерживает работу с ними.

## FAT

Файловая система FAT (File Allocation Table) была разработана Биллом Гейтсом и Марком МакДональдом в 1977 году и первоначально использова-

лась в MS-DOS 1.0, выпущенной в августе 1981 года. FAT была предназначена для работы с гибкими дисками размером менее 1 Мбайт, и первоначально не предусматривала поддержки жестких дисков. В настоящее время FAT поддерживает разделы размером до 2 Гбайт.

В файловой системе FAT используются следующие соглашения по именам файлов:

- имя файла или папки должно начинаться с буквы или цифры. Может содержать любой символ ASCII, за исключением пробела и символов `"/\[]:;|=,^*?`;
- длина имени не превышает восьми символов, а следующее за ним расширение — трех символов;
- файловая система не чувствительна к регистру вводимых символов.

## HPFS

Высокопроизводительная файловая система HPFS (High Performance File System) была представлена IBM в 1989 году в операционной системе OS/2 1.20. HPFS поддерживалась Windows NT до версии 3.51 включительно. По производительности эта файловая система значительно опережает FAT. HPFS позволяет использовать жесткие диски объемом до 2 Тбайт. Кроме того, она поддерживает разделы диска размером до 512 Гбайт и позволяет использовать имена файлов длиной до 255 символов (на каждый символ при этом отводится 2 байта).

## VFAT

Файловая система VFAT (Virtual FAT), реализованная в Windows NT 3.5, Windows 95 (DOS 7.0), — это файловая система FAT, включающая поддержку длинных имен файлов (Long File Name, LFN) в кодировке Unicode (каждый символ имени состоит из 2-х байт). VFAT использует ту же самую схему распределения дискового пространства, что и FAT, поэтому размер кластера определяется величиной раздела. В отличие от старой системы, новая поддерживает имена длиной до 255 символов.

## FAT32

FAT32 — усовершенствованная версия файловой системы VFAT, поддерживающая жесткие диски объемом до 2 Тбайт. Впервые файловая система

FAT32 была включена в состав Windows 95 OSR 2. В ней были расширены атрибуты файлов, позволяющие хранить время и дату создания, модификации и последнего доступа к файлу или папке.

Из-за требования совместимости с ранее созданными программами структура FAT32 содержит минимальные изменения. Главные отличия от предыдущих версий FAT состоят в следующем: блок начальной загрузки в разделах с FAT32 был увеличен до 2 секторов и включает в себя резервную копию загрузочного сектора. Это делает систему более устойчивой к возможным сбоям на диске.

## NTFS

NTFS (New Technology File System) — файловая система, предназначенная для использования в операционных системах семейства Windows NT, 2k. На сегодняшний день известно несколько версий NTFS (табл. 13.1). Иногда последние версии обозначают как v4.0, v5.0 и v5.1 в соответствии с версиями Windows NT, с которыми они поставляются.

***Таблица 13.1.*** *Версии файловой системы NTFS*

| Версия | Альтернативное название | Операционная система | Год |
|---|---|---|---|
| 1.0 | | Windows NT 3.1 | 1993 |
| 1.1 | | Windows NT 3.5 | 1994 |
| 1.2 | | Windows NT 3.51 | 1995 |
| | NTFS V 4.0 | Windows NT 4.0 | 1996 |
| 3.0 | NTFS V 5.0 | Windows 2000 | 2000 |
| 3.1 | NTFS V 5.1 | Windows XP | 2001 |
| | NTFS V 5.2 | Windows 2003 Server | 2003 |
| | NTFS V 6.0 | Windows Vista | 2005 |
| | NTFS V 7.0 | Windows 2008 Server | 2008 |

В NTFS по сравнению с другими файловыми системами значительно расширены возможности по управлению доступом к файлам и папкам, увеличено

количество атрибутов, реализована система отказоустойчивости, используются средства динамического сжатия файлов.

## Главная файловая таблица

MFT (Master File Table) — это метаданные, в которых хранится информация о содержимом тома NTFS. Данные хранятся в табличной форме. Строки таблицы соответствуют файлам и каталогам, а столбцы — их атрибутам. В ней также хранится служебная информация о конфигурации тома NTFS.

Каждый файл на томе NTFS представлен записью в таблице MFT. В NTFS зарезервированы первые 16 записей таблицы размером около 1 Мбайт для служебной информации.

Первая запись таблицы описывает непосредственно саму MFT. За ней следует ее зеркальная запись. Если первая запись MFT разрушена, NTFS считывает вторую запись, чтобы отыскать зеркальный файл MFT. Местоположение сегментов данных MFT и зеркального файла MFT хранится в секторе начальной загрузки. Копия сектора начальной загрузки находится в логическом центре диска. Третья запись MFT содержит файл регистрации, используемый для восстановления файлов. Записи с 4-й по 16-ю содержат различную служебную информацию. Семнадцатая и последующие записи главной файловой таблицы используются собственно файлами и каталогами на томе.

## Метафайлы NTFS

Как и любая другая система, NTFS делит все полезное место на кластеры — блоки данных, используемые единовременно. NTFS поддерживает почти любые размеры кластеров — от 512 байт до 64 Кбайт, неким стандартом же считается кластер размером 4 Кбайт.

Диск NTFS условно делится на две части. Первые 12% диска (рис. 13.1) отводятся под так называемую MFT-зону — пространство, в котором растет метафайл MFT (об этом ниже). Запись каких-либо данных в эту область невозможна. MFT-зона всегда держится пустой — это делается для того, чтобы самый главный, служебный файл (MFT) не фрагментировался при своем росте. Остальные 88% диска представляют собой обычное пространство для хранения файлов.




**Рис. 13.1.** Структура тома NTFS

Свободное место на диске складывается из всего физически свободного объема и незаполненных фрагментов MTF-зоны. Механизм использования MFT-зоны таков: когда файлы уже нельзя записывать в обычное пространство, MFT-зона просто сокращается (в текущих версиях операционных систем ровно в два раза), освобождая место для записи файлов. При освобождении места в обычной области MFT зона снова расширяется.

Метафайлы находятся в корневом каталоге диска с файловой системой NTFS — они начинаются с символа имени "$", хотя получить какую-либо информацию о них стандартными средствами сложно. Любопытно, что и для этих файлов указан вполне реальный размер — можно узнать, например, сколько операционная система тратит на каталогизацию всего вашего диска, посмотрев размер файла $MFT. В табл. 13.2 приведены используемые в данный момент метафайлы и их назначение.

***Таблица 13.2.*** *Используемые в NTFS метафайлы и их назначение*

| Номер записи | Файл | Назначение |
|---|---|---|
| 0 | $MFT | Файл, который содержит все объекты раздела, включая их атрибуты |
| 1 | $MFTMirr | Копия первых 16 записей MFT, размещенная посередине диска |
| 2 | $LogFile | Файл поддержки журналирования (transaction log) |
| 3 | $Volume | Файл, который содержит в себе различную служебную информацию — метку тома, версию файловой системы и т. д. |
| 4 | $AttrDef | Список стандартных атрибутов файлов на томе |

***Таблица 13.2*** *(окончание)*

| Номер записи | Файл | Назначение |
|---|---|---|
| 5 | . | Корневой каталог (root) |
| 6 | $Bitmap | Карта свободного места тома |
| 7 | $Boot | Содержит загрузочный сектор, а также номера кластеров, с которых начинается $MFT и $MFTMirr |
| 8 | $BadClus | Содержит так называемые поврежденные сектора (bad sectors). Заполнение этого файла осуществляется с помощью утилиты chkdsk, которая делает соответствующие метки |
| 9 | $Secure | База безопасности (acl), которая содержит два объекта:<br>• $sii — содержит идентификаторы безопасности (sid);<br>• $sdh — содержит описатели безопасности, а именно — назначенные права |
| 10 | $UpCase | Файл с таблицей соответствия заглавных и прописных букв в имен файлов на текущем томе, которая необходима для увеличения скорости поиска |
| 11 | $Extend | Системная папка, содержащая файлы 24, 25, 26 |
| 12..23 | Зарезервировано MFT | |
| 24 | $Extend\$Quota | Управление ограничениями на использование дискового пространства (квотирование) |
| 25 | $Extend\$ObjId | Контекстный идентификатор безопасности |
| 26 | $Extend\$Reparse | Дополнительная точка подключения к базе данных, необходимая для реализации процесса оптимизации базы |
| 27 | pagefile.sys | Служебный файл, создаваемый операционной системой, так называемый файл подкачки (pagefile). По сути он не является метафайлом |

# Особенности NTFS

NTFS была разработана как восстанавливаемая файловая система, использующая модель обработки транзакций. Каждая операция ввода-вывода, изменяющая файл на томе NTFS, рассматривается системой как транзакция и выполняется как неделимый блок. При модификации файла пользователем служба файловой системы фиксирует всю информацию, необходимую для повторения или отката транзакции. Если транзакция завершена успешно, производится модификация файла. Если нет — NTFS производит откат транзакции.

Несмотря на наличие защиты от несанкционированного доступа к данным, NTFS не обеспечивает необходимую конфиденциальность хранимой информации. Для получения доступа к файлам достаточно загрузить компьютер в DOS с дискеты и воспользоваться каким-нибудь сторонним драйвером NTFS.

Последняя версия NTFS — v3.1 (см. табл. 13.1). В новой версии NTFS были введены дополнительные атрибуты файлов; наряду с правом доступа введено понятие запрета доступа, позволяющее, например, при наследовании пользователем прав группы на какой-нибудь файл запретить ему возможность изменять его содержимое. Новая система позволяет вводить ограничения (квоты) на размер дискового пространства, предоставленного пользователям. Еще одна особенность — возможность проецировать любой каталог (как на локальном, так и на удаленном компьютере) в подкаталог на локальном диске. Среди нововведений — динамическое шифрование файлов и каталогов, повышающее надежность хранения информации. В состав операционных систем с NTFS v.3.1 входит файловая система с шифрованием, использующая алгоритмы шифрования с общим ключом (Encrypting File System, EFS). Если для файла установлен атрибут шифрования, то при обращении пользовательской программы к файлу для записи или чтения происходит прозрачное для программы кодирование и декодирование файла.

# Сравнение файловых систем

В табл. 13.3 сведены существенные плюсы и минусы распространенных в наше время систем, таких как FAT, FAT32 и NTFS.

***Таблица 13.3.*** *Сравнение файловых систем FAT и NTFS*

| Характеристика | FAT | FAT32 | NTFS |
|---|---|---|---|
| Системы, ее поддерживающие | DOS, Windows 9*x*, NT всех версий | Windows 98, Windows 2k | NT4, Windows 2k |
| Максимальный размер тома | 2 Гбайт | Не ограничен | Не ограничен |
| Максимальное число файлов на томе | Около 65 000 | Не ограничено | Не ограничено |
| Имя файла | С поддержкой длинных имен — 255 символов, системный набор символов | С поддержкой длинных имен — 255 символов, системный набор символов | 255 символов, любые символы любых алфавитов (65 тысяч разных начертаний) |
| Возможные атрибуты файла | Базовый набор | Базовый набор | Нет ограничений |
| Безопасность | – | – | + |
| Сжатие | – | – | + |
| Устойчивость к сбоям | Средняя | Низкая. Средства по оптимизации производительности файловой системы привели к появлению уязвимых мест | Высокая. Достигается интегрированной системой отказоустойчивости |
| Экономичность | Минимальная. Огромный размер кластеров на больших дисках | Улучшена за счет уменьшения размеров кластеров | Максимальная |
| Быстродействие | Высокое для малого числа файлов, но быстро уменьшается с ростом их количества | Полностью аналогично FAT. На дисках большого размера (десятки Гбайт) начинаются серьезные проблемы с общей организацией данных | Система не очень эффективна для разделов до 1 Гбайт и максимально эффективна на больших дисках |

# Управление правами доступа на файлы и папки

При создании новой папки или файла на жестком диске с файловой системой NTFS новому объекту присваиваются права доступа, которые впоследствии изменяются администратором в случае необходимости. Для программного управления правами доступа необходимо знать объектную модель NTFS. Управление правами осуществляется с помощью библиотеки ADsSequrity.dll, входящей в комплект поставки ADSI Resource Kit (**http://www.microsoft.com/ntserver/nts/downloads/other/ADSI25**). Для использования библиотеки ее необходимо зарегистрировать в операционной системе. Для этого следует выполнить команду:

```
regsvr32.exe /s ADsSequrity.dll
```

## Организация доступа к параметрам безопасности

Существуют три типа объектов, безопасностью которых можно управлять:

- объект Active Directory — учетная запись группы, пользователя, контейнер и т. д.;
- объект NTFS — файл или папка;
- реестр — куст или ветвь.

# Объектная модель NTFS

Рассмотрим устройство объектной модели NTFS. На рис. 13.2 изображена стандартная вкладка безопасности созданной папки.

В настоящее время для управления безопасностью объектов используется четыре интерфейса из семейства `IADs*`:

- `IADsSecurityUtility` — используется для получения доступа к спискам безопасности объекта;
- `IADsSecurityDescriptor` — предназначен для получения доступа к объектам ADSI;

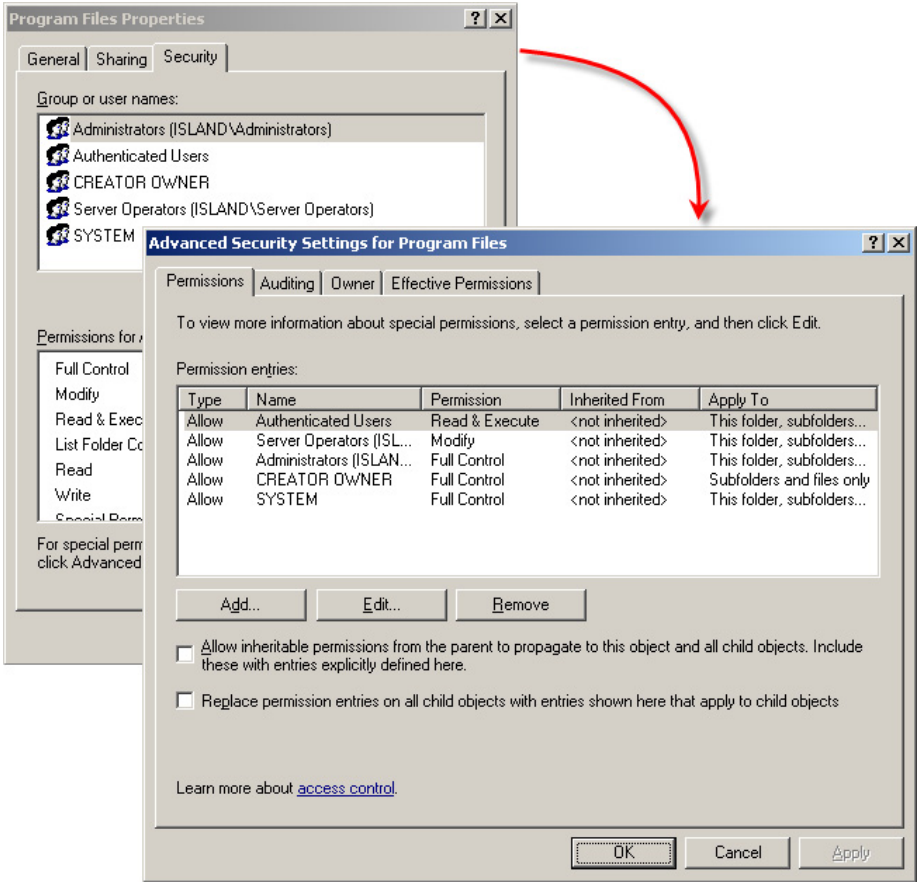



**Рис. 13.2.** Вкладка безопасности объекта (файла или папки)

- `IADsAccessControlList` — обеспечивает доступ к ACL-списку объектов (учетная запись пользователя или группы);
- `IADsAccessControlEntry` — обеспечивает доступ к характеристикам объекта безопасности (учетная запись пользователя или группы).

# Объектная модель *IADsSecurityUtility*

Один из основных объектов — `IADsSecurityUtility` — позволяет получить доступ к описателям системы безопасности основных объектов: файлов и папок NTFS, сетевых ресурсов (share), кустов и ветвей реестра.

# Методы интерфейса *IADsSecurityUtility*

Интерфейс `IADsSecurityUtility` поддерживает три метода, описание которых приведено в табл. 13.4.

***Таблица 13.4.*** *Методы интерфейса* `IADsSecurityUtility`

| Метод | Описание |
|---|---|
| `GetSecucityDescriptor()` | Возвращает описатель безопасности для указанного объекта |
| `SetSecucityDescriptor()` | Устанавливает описатель безопасности для указанного объекта |
| `ConvertSecurityDescriptor()` | Преобразует полученный описатель безопасности в другой формат |

## Метод *GetSecucityDescriptor()*

Метод предназначен для получения доступа к описателю безопасности объекта. В качестве объекта может быть файл или папка, сетевой ресурс (опубликованная в сети папка), куст или ветка реестра.

Для вызова метода `GetSecucityDescriptor()` используется шаблон:

```
Set sd = sec.GetSecurityDescriptor(pathToObject, pathFormat, sdFormat,
retrivesFormat)
```

В этом вызове метода значением параметра `pathToObject` является путь к объекту, формат которого зависит от типа объекта (табл. 13.5). Второй параметр, `pathFormat`, определяет тип объекта, к которому задан путь.

***Таблица 13.5.*** *Примеры возможных значений параметров* `pathToObject` *и* `PathFormat`

| Тип объекта | Параметр `pathToObject` | Параметр `pathFormat` |
|---|---|---|
| Файл | `\\1001pc\public\test\manual.doc` | 1 |
| Каталог | `\\hoda\software\viewers` | 1 |
| Сетевой ресурс | `\\hoda\software$` | 2 |
| Куст реестра | `HKEY_LOCAL_MACHINE` | 3 |
| Ветвь реестра | `HKLM\Software\Microsoft` | 3 |

Числовыми значениями параметров `sdFormat` и `retrivesFormat` определяется формат интерфейсов для входного и выходного потоков (табл. 13.6) соответственно.

***Таблица 13.6.*** *Примеры возможных значений параметров* `sdFormat` *и* `retrivesFormat`

| **Значение параметров `sdFormat`, `retrivesFormat`** | **Описание** |
|---|---|
| 1 | Идентификатор безопасности преобразуется в формат интерфейса `IADsSecurityUtility` |
| 2 | Идентификатор безопасности преобразуется в бинарный формат |
| 3 | Идентификатор безопасности преобразуется в шестнадцатеричную строку |

В листинге 13.1 приведен реальный пример использования метода `GetSecucityDescriptor()`.

**Листинг 13.1. Вызов метода `GetSecucityDescriptor()`**

```
Set sec = CreateObject("ADsSecurityUtility")
Set sd = sec.GetSecurityDescriptor("C:\test", 1, 1, 1)
```

## Метод *SetSecucityDescriptor()*

Метод предназначен для записи описателей безопасности объектов. В качестве объекта может выступать файл или папка, сетевой ресурс (опубликованная в сети папка), куст и ветка реестра.

Для вызова метода `GetSecucityDescriptor()` используется шаблон:

```
Set sd = sec.SetSecurityDescriptor(pathToObject, pathFormat, VarData,
sdFormat)
```

Здесь значением параметра `pathToObject` является путь к объекту, формат которого зависит от типа объекта (см. табл. 13.5). Второй параметр, `pathFormat`, определяет тип объекта, к которому задан путь. Третий параметр, `VarData`, представляет описатель безопасности и содержит изменения,

которые необходимо сделать. Четвертый параметр, `sdFormat`, определяет формат параметра `VarData` (см. табл. 13.6).

В листинге 13.2 приведен пример изменения владельца (owner) папки (рис. 13.3).

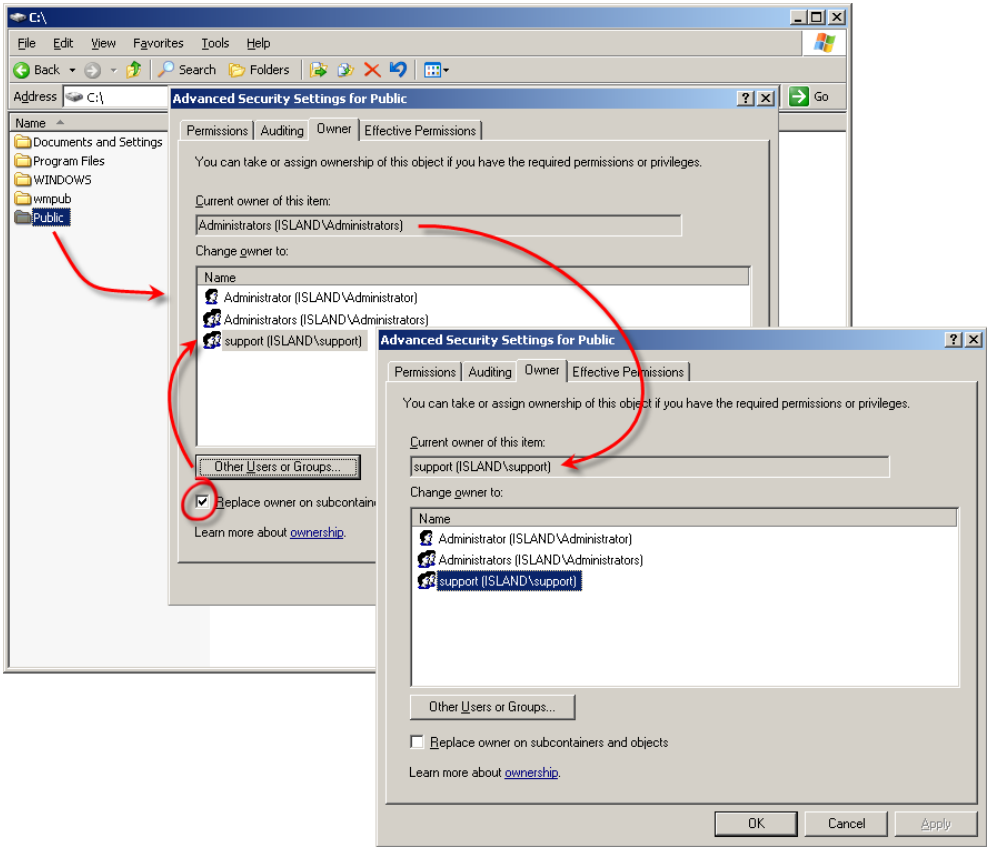



**Рис. 13.3.** Вкладка безопасности объекта (файла или папки)

**Листинг 13.2. Пример использования метода `SetSecucityDescriptor()`. Изменения владельца (owner) папки**

```
strPath="c:\public"
set sec = CreateObject("ADsSecurityUtility")
Set sd = sec.GetSecurityDescriptor(strPath, 1, 1)
```

```
a1 = sd.owner
sd.owner = "island\support"
sec.SetSecurityDescriptor strPath, 1, sd, 1
a2 = sd.owner
Wscript.Echo "Старый владелец: " & a1 & vbNewLine & _
             "Новый владелец: " & a2
```

### Метод *ConvertSecurityDescriptor()*

Метод предназначен для преобразования типа описателя объекта. Для вызова метода `ConvertSecurityDescriptor()` используется шаблон:

```
Set sd = sec.ConvertSecurityDescriptor(VarData, inFormat, outFormat)
```

В этом шаблоне `VarData` — данные описателя безопасности, которые необходимо преобразовать. Значения остальных двух параметров — `inFormat` и `outFormat`, соответственно, описывают входной и выходной форматы преобразуемых данных. Расшифровка возможных значений этих параметров приведена в табл. 13.7.

***Таблица 13.7.*** *Форматы данных идентификатора безопасности*

| Формат идентификатора | Описание |
|---|---|
| 1 | Формат интерфейса `IADsSecurityUtility` |
| 2 | Бинарный формат |
| 3 | Шестнадцатеричная строка |

## Свойства интерфейса *IADsSecurityUtility*

Интерфейс `IADsSecurityUtility` поддерживает единственное свойство — `SecurityMask`, с помощью которого определяется список доступных описателей безопасности.

### Свойство *SecurityMask*

Значением свойства `SecurityMask` (табл. 13.8) определяют область действия методов `GetSecurityDescriptor()` и `SetSecurityDescriptor()`.

По умолчанию свойство `SecurityMask` принимает комбинированное значение: `SecurityMask = 7`.

***Таблица 13.8.*** *Возможные значения свойства* `SecurityMask`

| Значение | По умолчанию | Описание |
|---|---|---|
| 1 | + | Чтение/запись владельца (owner) ресурса |
| 2 | + | Чтение/запись группы данных |
| 4 | + | Чтение/запись данных DACL |
| 8 | – | Чтение/запись данных SACL |

В листинге 13.3 приведен пример чтения и переопределения значения свойства `SecurityMask`.

**Листинг 13.3. Чтение и запись значения свойства `SecurityMask`**

```
set sec = CreateObject("ADsSecurityUtility")
sec.SecurityMask = 1
Wscript.Echo sec.SecurityMask
```

# Объектная модель *IADsSecurityDescriptor*

С помощью `IADsSecurityDescriptor` обеспечивается доступ к системе безопасности ADSI-объектов. В табл. 13.9 приведен список свойств объекта `IADsSecurityDescriptor`.

***Таблица 13.9.*** *Свойства интерфейса* `IADsSecurityDescriptor`

| Свойство | Описание |
|---|---|
| `Control` | Чтение/запись флага Security_Descriptor_Control |
| `DiscretionaryAcl` | Чтение/запись ассоциируемого с описателем безопасности DACL |
| `Group` | Чтение/запись группы-владельца объекта, ассоциируемого с данным описателем безопасности |

***Таблица 13.9*** *(окончание)*

| Свойство | Описание |
|---|---|
| GroupDefaulted | Чтение/запись группы-владельца объекта, ассоциируемого с данным описателем безопасности по умолчанию |
| Owner | Чтение/запись владельца данного объекта |
| OwnerDefaulted | Чтение/запись владельца данного объекта по умолчанию |
| Revision | Чтение/запись версии ACL |
| SystemAcl | Чтение/запись системного ACL, ассоциируемого с данным описателем безопасности |

Чтение и запись свойств осуществляется в соответствии с шаблоном, приведенным в листингах 13.5а и 13.5б соответственно.

**Листинг 13.4а. Шаблон использования свойства (чтение данных) интерфейса IADsSecurityDescriptor**

```
Set ou = GetObject("LDAP:// " & RDN_Path)
Set sd = ou.Get("ntSecurityDescriptor")
Wscript.Echo sd.Property
```

**Листинг 13.4б. Шаблон использования свойства (запись данных) интерфейса IADsSecurityDescriptor**

```
Set obj = GetObject("LDAP://" & RDN_Path)
Set sd = obj.Get("ntSecurityDescriptor")
Set val = sd
val.Property = Value
obj.put "ntSecurityDescriptor", val
obj.SetInfo
```

Несмотря на то, что этот интерфейс поддерживает около десятка свойств, на практике чаще всего используется только одно — Owner.

## Свойство *Owner*

Значением свойства Owner является учетная запись пользователя, создавшего данный объект.

Данное свойство доступно как для чтения (листинг 13.5а), так и для записи (листинг 13.5б). Для отображения владельца объекта в MMC-консоли необходимо войти в свойства этого объекта и выбрать вкладку **Owner** (рис. 13.4). Эта вкладка отображается только в расширенном режиме, который включается установкой флажка в пункте меню **View | Advanced Features**.

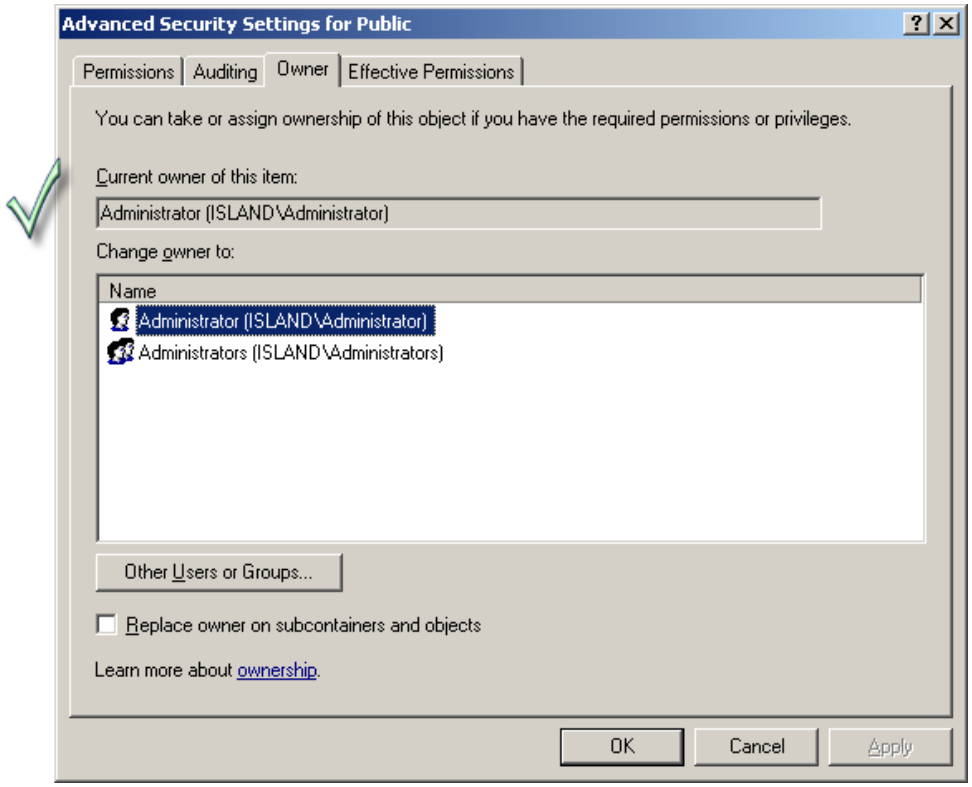



**Рис. 13.4.** Владелец объекта (во вкладке **Owner**)

**Листинг 13.5а. Чтение владельца (owner) папки**

```
Set obj = GetObject("LDAP://OU=WorkSpcase,DC=island,DC=ru")
Set sd = obj.Get("ntSecurityDescriptor")
Wscript.Echo sd.Owner
```

**Листинг 13.5б. Изменение владельца (owner) папки**

```
Set obj = GetObject("LDAP://OU=WorkSpcase,DC=island,DC=ru")
Set sd = obj.Get("ntSecurityDescriptor")
Set val = sd
val.Owner = "island\support"
obj.put "ntSecurityDescriptor", val
obj.SetInfo
```

# Объектная модель *IADsAccessControlList*

Интерфейс `IADsAccessControlList` используется для получения и работы со списком управления доступом объекта (рис. 13.5).

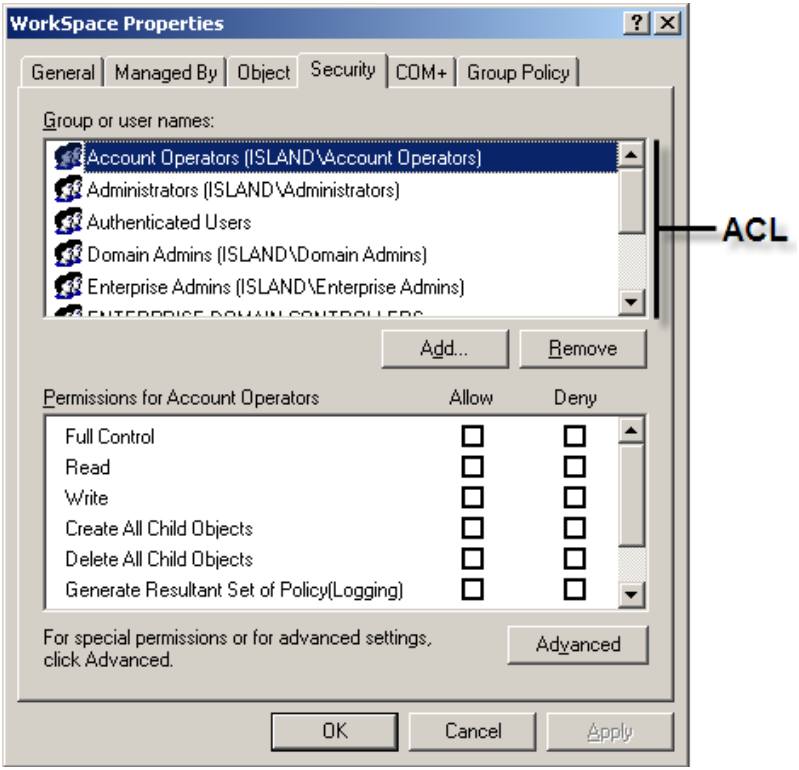



**Рис. 13.5.** Пример списка ACL

С его помощью можно удалять и добавлять объекты безопасности (записи управления доступом ACE) в список. На практике интерфейс `IADsAccessControlList` используется совместно с интерфейсом `IADsAccessControlEntry`.

## Методы интерфейса *IADsAccessControlList*

Посредством интерфейса `IADsAccessControlList` осуществляется управление списком учетных записей пользователей или групп, с помощью которых распределяют доступ к объекту. В табл. 13.10 приведены методы интерфейса `IADsAccessControlList`.

***Таблица 13.10.*** *Методы интерфейса* `IADsAccessControlList`

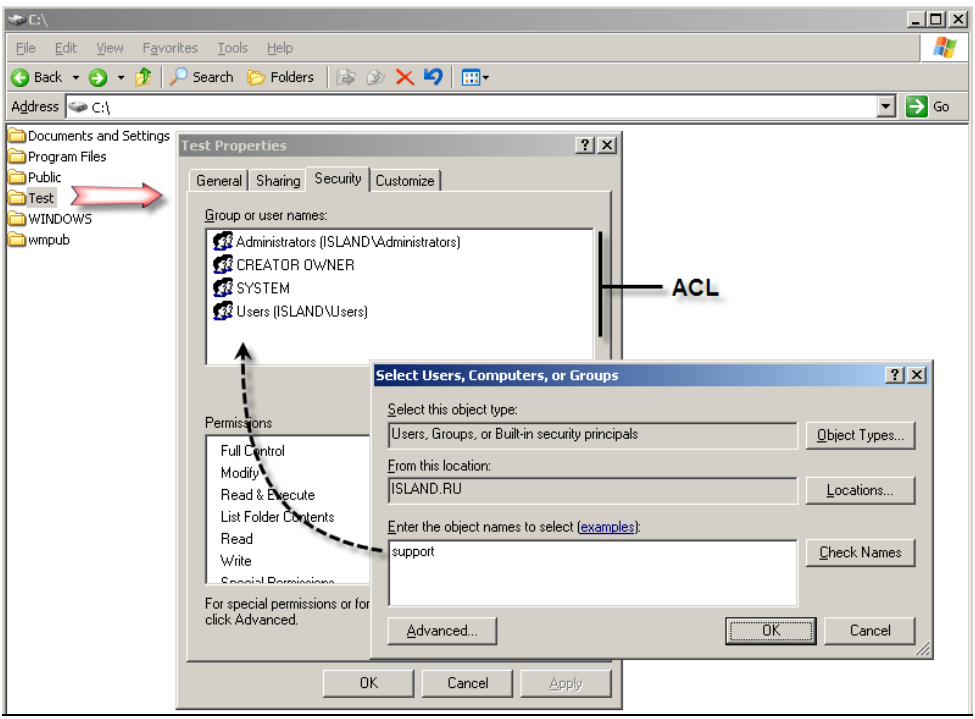
| Методы | Описание |
|---|---|
| `AddAce()` | Добавить ACE-объект в список ACL |
| `RemoveAce()` | Удалить ACE-объект из списка ACL |
| `CopyAccessList()` | Скопировать все ACE-объекты в список ACL |

### Метод *AddAce()*

Метод `AddAce()` используется для добавления нового объекта (учетной записи пользователя или группы) в список управления доступом (рис. 13.6, а). В работе сценария (листинг 13.6а) используются два метода объекта `IADsSecurityUtility`. На первом этапе с помощью метода `GetSecurityDescriptor()` получают доступ к описателю безопасности. После этого, используя `IADsAccessControlEntry` (см. далее *разд. "Объектная модель IADsAccessControlEntry"*), обеспечивают назначение параметров безопасности указанному объекту, в данном случае — группе island\support. Затем сделанные изменения фиксируют в файловой системе с помощью метода `SetSecurityDescriptor()`.

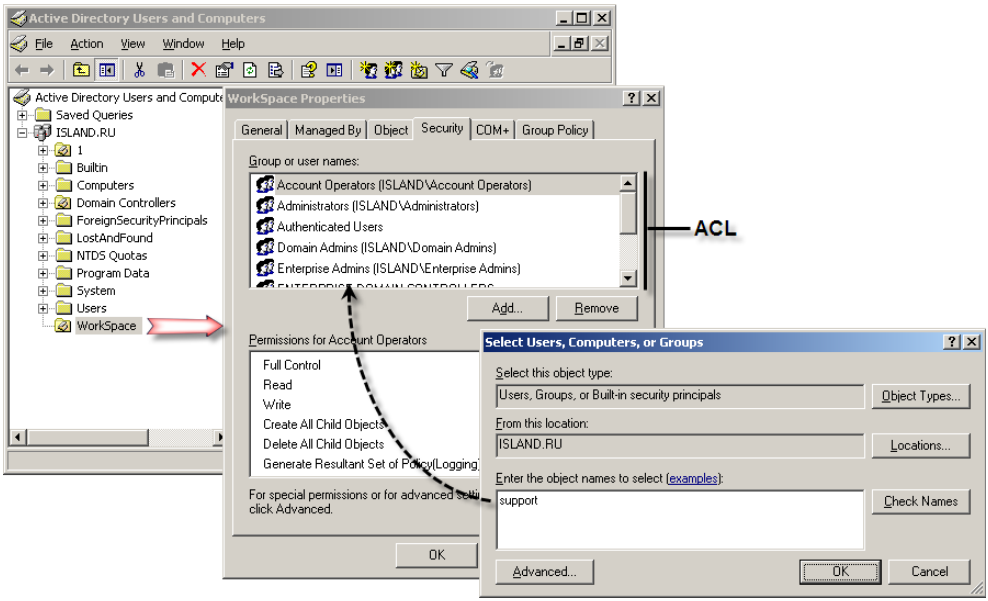
**Листинг 13.6а. Использование метода `AddAce()` в NTFS**

```
Set sec = CreateObject("ADsSecurityUtility")
Set sd = sec.GetSecurityDescriptor("c:\Test",1,1)
Set dacl = sd.DiscretionaryAcl
Set ace = CreateObject("AccessControlEntry")
ace.Trustee = "island\support"
ace.AccessMask = &h20000000
ace.AceType = &h0
ace.AceFlags = &h3
dacl.AddAce ace
sd.DiscretionaryAcl = dacl
sec.SetSecurityDescriptor "c:\Test", 1, sd, 1
```




**а**

**Рис. 13.6, а.** Добавление ACE-объекта в ACL-список. NTFS




б

**Рис. 13.6, б.** Добавление объекта ACE в список ACL. ADSI

Сценарий (листинг 13.6б) изменения параметров безопасности объектов в ADSI несколько сложнее. Однако идея, заложенная в нем, та же (рис. 13.6, б).

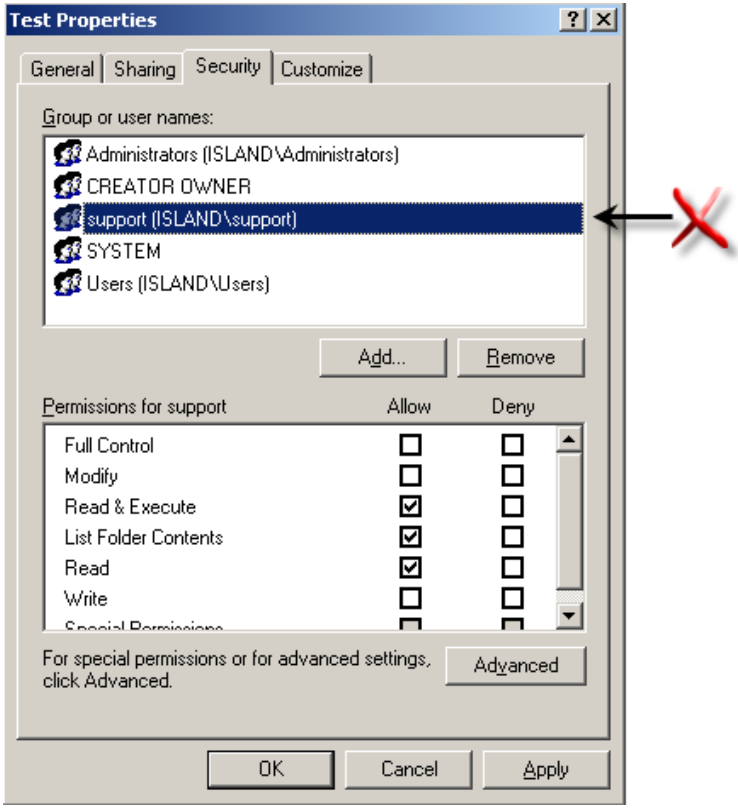
**Листинг 13.6б. Использование метода `AddAce()` в ADSI**

```
Set obj = GetObject("LDAP://OU=WorkSpcase,DC=island,DC=ru")
Set sd = obj.Get("ntSecurityDescriptor")
Set dacl = sd.DiscretionaryAcl
Set ace = CreateObject("AccessControlEntry")
ace.Trustee = "island\support"
ace.AccessMask = &h20000000
ace.AceType = &h0
ace.AceFlags = &h3
dacl.AddAce ace
```

```
sd.DiscretionaryAcl = dacl
ou.Put "ntSecurityDescriptor", sd
ou.SetInfo
```

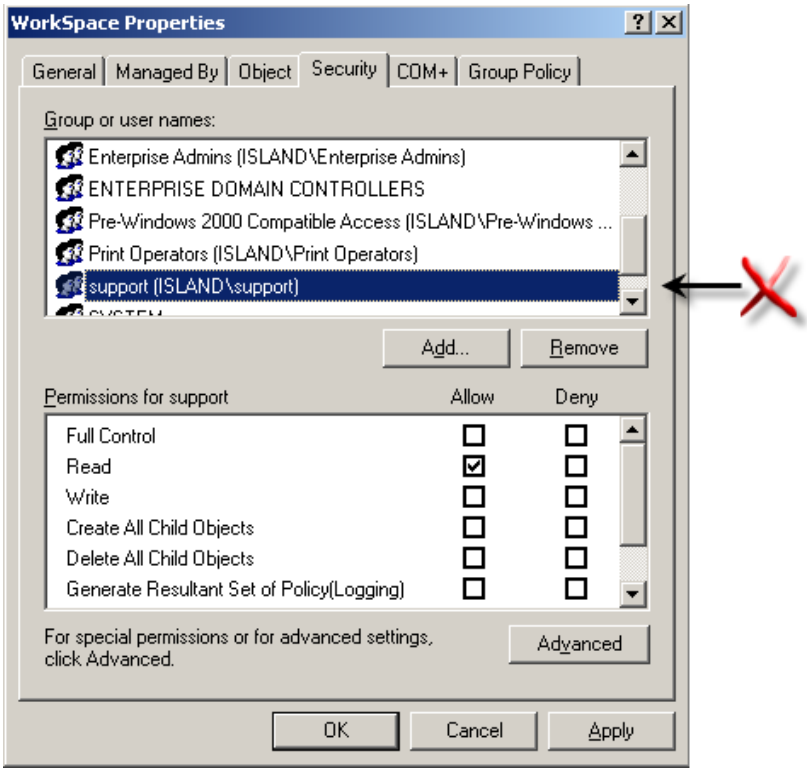
## Метод *RemoveAce()*

Метод `RemoveAce()` используется для удаления существующего объекта (учетной записи пользователя или группы) из списка ACL (рис. 13.7). В работе сценария (листинги 13.7а и 13.7б) используется только метод `GetSecurityDescriptor()`. В отличие от метода `AddAce()` в данном случае метод `SetSecurityDescriptor()` использовать не надо.




**а**

**Рис. 13.7, а.** Удаление объекта ACE из списка ACL. NTFS




**б**

**Рис. 13.7, б.** Удаление объекта ACE из списка ACL. ADSI

**Листинг 13.7а. Использование метода `RemoveAce()` в NTFS**

```
Set sec = CreateObject("ADsSecurityUtility")
Set sd = sec.GetSecurityDescriptor("c:\Test",1,1)
Set dacl = sd.DiscretionaryAcl
Set ace = CreateObject("AccessControlEntry")
For Each ace in dacl
    If StrComp(UCase(ace.Trustee), UCcase("island\support"))=0 Then
        dacl.RemoveAce ace
    End If
Next
sd.DiscretionaryAcl = dacl
```

**Листинг 13.7б. Использование метода `RemoveAce()` в ADSI**

```
Set obj = GetObject("LDAP://OU=WorkSpcase,DC=island,DC=ru")
Set sd = obj.Get("ntSecurityDescriptor")
Set dacl = sd.DiscretionaryAcl
Set ace = CreateObject("AccessControlEntry")
For Each ace in dacl
    If StrComp(UCase(ace.Trustee), UCase("island\support"))=0 Then
        dacl.RemoveAce ace
    End If
Next
sd.DiscretionaryAcl = dacl
ou.Put "ntSecurityDescriptor", sd
ou.SetInfo
```

## Метод *CopyAccessList()*

С помощью метода `CopyAccessList()` осуществляется копирование прав безопасности (всего списка ACL) из одного объекта в другой (листинг 13.8а). При использовании метода `CopyAccessList()` следует помнить, что осуществляется копирование списка (рис. 13.8), который, с точки зрения программирования, является многомерным массивом (листинг 13.8б), поскольку каждый элемент списка имеет несколько параметров.

**Листинг 13.8а. Использование метода `CopyAccessList()` в NTFS**

```
Set sec = CreateObject("ADsSecurityUtility")
Set sd = sec.GetSecurityDescriptor("c:\Test",1,1)
Set dacl = sd.DiscretionaryAcl
Set copyDacl = dacl.CopyAccessList()
Set sd = sec.GetSecurityDescriptor("c:\Public",1,1)
sd.DiscretionaryAcl = copyDacl
```

**Листинг 13.8б. Использование метода `CopyAccessList()` в ADSI**

```
Set obj = GetObject("LDAP://CN=Users,DC=msk,DC=prosv,DC=ru")
Set sd = obj.Get("ntSecurityDescriptor")
```

```
Set Dacl = sd.DiscretionaryAcl
Set CopyDacl = Dacl.CopyAccessList()

Set obj = GetObject("LDAP://OU=WorkSpace2, DC=Fabrikam,DC=com")
Set sd = obj.Get("ntSecurityDescriptor")
sd.DiscretionaryAcl = CopyDacl
obj.Put "ntSecurityDescriptor", Array(sd)
obj.SetInfo
```

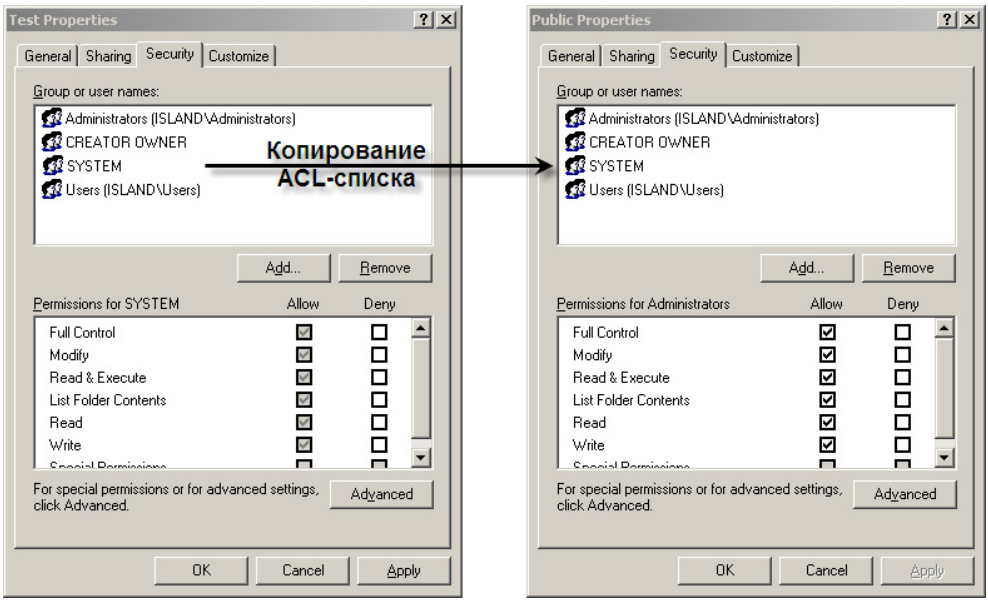



**Рис. 13.8.** Копирование списка ACL между объектами

## Свойства интерфейса *IADsAccessControlList*

В табл. 13.11 приведено описание основных свойств интерфейса `IADsAccessControlList`.

***Таблица 13.11.*** *Свойства интерфейса* `IADsAccessControlList`

| Свойство | Описание |
|---|---|
| `AceCount` | Определяет количество ACE-объектов в списке ACL |
| `AclRevision` | Выводит номер версии ACL |

### Свойство *AceCount*

Значение свойства `AceCount` — целое число, показывающее количество объектов ACE в списке ACL. В качестве объектов ACE рассматриваются учетные записи пользователей и групп. Пример использования свойства `AceCount` приведен в листинге 13.9.

**Листинг 13.9. Использование свойства `AceCount`**

```
Set sec = CreateObject("ADsSecurityUtility")
Set sd = sec.GetSecurityDescriptor("c:\Programs File",1,1)
Set dacl = sd.DiscretionaryAcl
WScript.Echo dacl.AceCount
```

### Свойство *AclRevision*

Значением свойства `AclRevision` является номер версии описателя безопасности, используемый для получения списка ACL. Пример использования свойства `AclRevision` приведен в листинге 13.10.

**Листинг 13.10. Использование свойства `AclRevision`**

```
Set sec = CreateObject("ADsSecurityUtility")
Set sd = sec.GetSecurityDescriptor("c:\Windows",1,1)
Set dacl = sd.DiscretionaryAcl
WScript.Echo dacl.AclRevision
```

## Объектная модель *IADsAccessControlEntry*

С помощью интерфейса `IADsAccessControlEntry` обеспечивают доступ к параметрам безопасности объекта (учетной записи пользователя, группы) в

списке ACL. Сам по себе этот объект невозможно вызвать. Алгоритм сценария для работы с ним следующий. Сперва получают доступ к объекту: папке или файлу в NTFS, объекту в Active Directory. Затем обеспечивают доступ к списку ACL, то есть к списку учетных записей групп и пользователей, которым обеспечен доступ к этому объекту. И, наконец, объекту, выбранному из списка, обеспечивается доступ к параметрам безопасности (ACE).

## Свойства интерфейса *IADsAccessControlEntry*

Каждая запись безопасности характеризуется несколькими параметрами безопасности (рис. 13.9), краткое описание которых приведено в табл. 13.12.

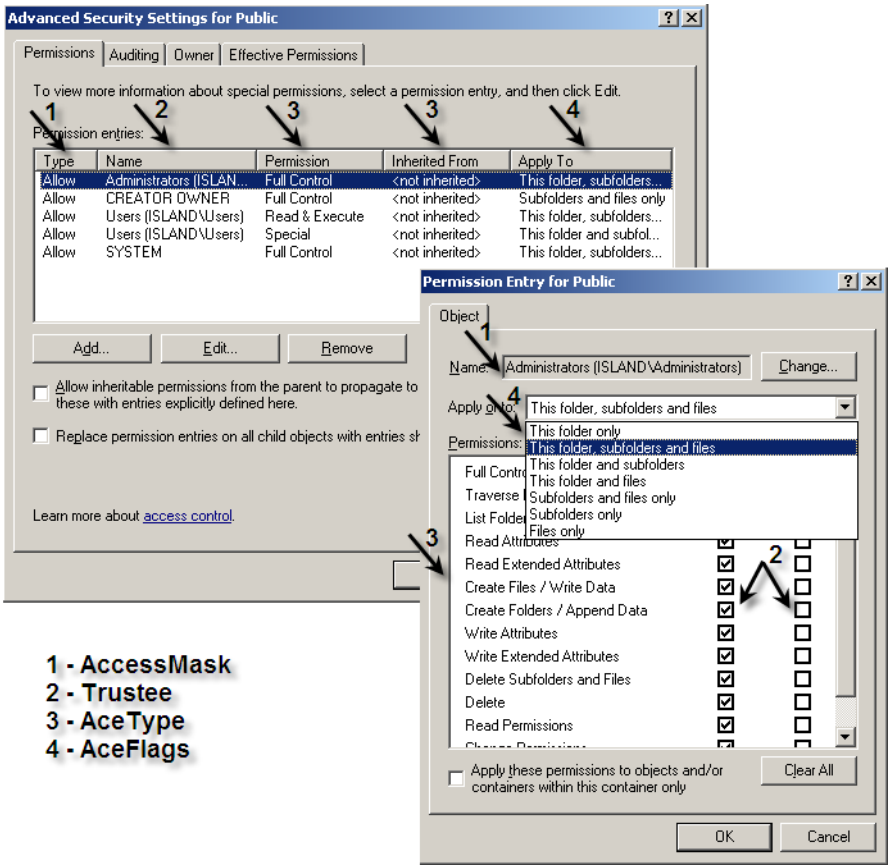



**Рис. 13.9.** Параметры безопасности ACE

***Таблица 13.12.*** *Свойства интерфейса* `IADsAccessControlList`

| Свойство | Описание |
|---|---|
| `Trustee` | Определение объекта, которому назначаются права. Пример: Island\support |
| `AccessMask` | Определяет уровень доступа к ресурсу (**Full Control**, **Modify**, **Read**, **Execute** и т. д.) |
| `AceFlags` | Определяет область действия прав (**This folder only**, **This folder and subfolders** и т. д.) и управляет наследованием прав на новые объекты |
| `AceType` | Определяет тип доступа (**Allow**, **Deny**, **Audit**) |

## Свойство *Trustee*

С помощью свойства `Trustee` (рис. 13.9) определяется имя объекта, параметры доступа к которому будут изменяться. Формат значения следующий: Domain\Object для домена или PCName\Object для компьютера. В обоих случаях Object — это `sAMAccountname` объекта, а остальные два параметра, соответственно, NetBIOS-имя домена или локального компьютера.

В листинге 13.11 приведен пример определения всех записей в списке ACL. Для доступа к выбранному объекту всегда используется метод `GetSecurityDescriptor()`.

**Листинг 13.11. Определение записей в списке ACL**

```
Set sec = CreateObject("ADsSecurityUtility")
Set sd = sec.GetSecurityDescriptor("c:\Test",1,1)
Set dacl = sd.DiscretionaryAcl

Set ace = CreateObject("AccessControlEntry")
For Each ace In dacl
    WScript.Echo ace.Trustee
Next
```

## Свойство *AccessMask*

Значением свойства `AccessMask` определяется уровень доступа к ресурсу для выбранного объекта.

Итоговым является складывающееся из составных частей суммарное значение. Составные значения приведены в табл. 13.13.

***Таблица 13.13.*** *Параметр* `AccessMask`

| Уровень доступа | &h1 | &h2 | &h4 | &h8 | &h10 | &h20 | &h40 | &h80 |
|---|---|---|---|---|---|---|---|---|
| **Full Control** (Полный доступ) | | | | | | | | |
| **Traverse Folder/ Execute Folder** (Обзор папок/ Выполнение файлов) | | | | | | + | | |
| **List Folder/ReadData** (Содержание папок/ Чтение данных) | + | | | | | | | |
| **Read Attributes** (Чтение атрибутов) | | | | | | | | + |
| **Read Extended Attributes** (Чтение дополнительных атрибутов) | | | | + | | | | |
| **Create Files/Write Data** (Создание файлов/ Запись данных) | | + | | | | | | |
| **Create Folder/Append Data** (Создание папок/ Дозапись данных) | | | + | | | | | |
| **Write Attributes** (Запись атрибутов) | | | | | | | | |
| **Write Extended Attributes** (Запись дополнительных атрибутов) | | | | | + | | | |
| **Delete Subfolders and Files** (Удаление подпапок и файлов) | | | | | | | + | |
| **Delete** (Удаление) | | | | | | | | |

***Таблица 13.13*** *(окончание)*

| Уровень доступа | &h1 | &h2 | &h4 | &h8 | &h10 | &h20 | &h40 | &h80 |
|---|---|---|---|---|---|---|---|---|
| **Read Permissions** (Чтение разрешений) | | | | | | | | |
| **Change Permissions** (Изменение разрешений) | | | | | | | | |
| **Take Ownership** (Смена владельца) | | | | | | | | |

| Уровень доступа | &h100 | &h10000 | &h20000 | &h40000 | &h80000 | &h100000 | &h10000000 | &h20000000 | &h40000000 | &h80000000 |
|---|---|---|---|---|---|---|---|---|---|---|
| **Full Control** | | | | | | + | | | | |
| **Traverse Folder/Execute Folder** | | | | | | + | | + | | |
| **List Folder/ReadData** | | | | | | + | | | | + |
| **Read Attributes** | | | | | | + | | + | | + |
| **Read Extended Attributes** | | | | | | + | | | | + |
| **Create Files/Write Data** | | | | | | + | | | + | |
| **Create Folder/Append Data** | | | | | | + | | | + | |
| **Write Attributes** | + | | | | | + | | | + | |
| **Write Extended Attributes** | | | | | | + | | | + | |
| **Delete Subfolders and Files** | | | | | | + | | | | |
| **Delete** | | + | | | | + | | | | |
| **Read Permissions** | | | + | | | + | + | + | + | + |
| **Change Permissions** | | | | + | | + | | | | |
| **Take Ownership** | | | | | | + | | | | |

## Свойство *AceFlags*

С помощью свойства `AceFlags` определяют область действия и порядок наследования объектов. При включенном наследовании в присваиваемом значении после h добавляется единица (табл. 13.14). В скобках указаны альтернативные значения.

***Таблица 13.14.*** *Параметр* `AceFlags`

| **Область действия** | **Inheritance** | |
|---|---|---|
| | **Not Inheritance** | **Parent Object** |
| **This folder only** (только эта папка) | &h0 (&h4) | &h10 (&h14) |
| **This folder, subfolder and files** (эта папка, подпапки и файлы) | &h3 (&h7) | &h13 (&h17) |
| **This folder and subfolder** (эта папка и подпапки) | &h2 (&h6) | &h12 (&h16) |
| **This folder and files** (эта папка и файлы) | &h1 (&h5) | &h11 (&h15) |
| **Subfolders and files only** (только подпапки и файлы) | &h0b (&h0f) | &h1b (&h1f) |
| **Subfolders only** (только подпапки) | &h0a (&h0e) | &h1a (&h1e) |
| **Files only** (только файлы) | &h0d (&h9) | &h1d (&h19) |
| **Nothing** (ничего) | &h0c(&h8) | &h1c(&h18) |

## Свойство *AceType*

С помощью свойства `AceType` обеспечивают установку флага **Allow / Deny / Audit** (табл. 13.15).

В листинге 13.12 приведен пример определения всех основных параметров существующей в ACL-списке учетной записи.

***Таблица 13.15.*** *Параметр* `AceType`

| **Тип** | **Флаг** |
|---|---|
| **Deny** | &h1 |
| **Allow** (Full Control) | &h0 |
| **Audit** | &h2 |

**Листинг 13.12. Чтение всех параметров ACL-списка**

```
Set sec = CreateObject("ADsSecurityUtility")
Set sd = sec.GetSecurityDescriptor("c:\Test",1,1)
Set dacl = sd.DiscretionaryAcl

Set ace = CreateObject("AccessControlEntry")
For Each ace In dacl
    WScript.Echo "Trustee " + CStr(ace.Trustee)
    WScript.Echo "AccessMask " + CStr(ace.AccessMask)
    WScript.Echo "AceType "+ CStr(ace.AceType )
    WScript.Echo "Aceflags " + CStr(ace.Aceflags)
Next
```

# Стандартный набор параметров безопасности

Для удобства читателей дополнительно приводится табл. 13.16, в которой указаны все параметры стандартных атрибутов, таких как **Full Control**, которые видит пользователь, открывая окно свойств безопасности папки или файла.

***Таблица 13.16.*** *Набор стандартных атрибутов*

| Установка атрибута | Фактические атрибуты | AccessMask | AceFlags |
|---|---|---|---|
| **Full Control** | **Full Control**<br>**Modify**<br>**Read & Execute**<br>**List Folder Contents**<br>**Read**<br>**Write** | &h10000000 | &h3 |
| **Modify** | **Modify**<br>**Read & Execute**<br>**List Folder Contents**<br>**Read**<br>**Write** | &h20, &h10000, &h80000000, &h40000000 | &h3 |
| **Read & Execute** | **Read & Execute**<br>**List Folder Contents**<br>**Read** | &h20000000, &h80000000 | &h2 |
| **List Folder Contents** | **List Folder Contents** | &h20000000, &h80000000 | &h3 |
| **Read** | **Read** | &h80000000 | &h3 |
| **Write** | **Write** | &h2, &h4, &h10, &h100 | &h3 |

# Технология Access-based Enumerator

Технология ABE (Access-based Enumerator) впервые стала доступной в Windows 2003 Server SP1. С ее помощью можно сделать невидимыми папки, к которым пользователь не имеет права доступа. Это не значит, что на папку

устанавливается атрибут "скрытый". Для пользователя, который не имеет прав как минимум на просмотр данной папки, она исчезает, хотя физически она присутствует на сервере и видна другим пользователям.

Возможности технологии ABE достаточно широки:

- высокий уровень защиты папок обеспечивает то, что пользователь может увидеть только те ресурсы, на которые ему предоставлены соответствующие права. Зачастую и название документа несет в себе очень важную информацию;
- ABE обеспечивает безболезненную миграцию сетевых ресурсов;
- использование этой технологии упрощает поиск необходимых документов, поскольку сотрудник видит только доступные ему документы;
- ABE отображает папки в сетевом окружении, основываясь на индивидуальных правах доступа пользователя к этим папкам (ACL).

## Установка ABE

Технология ABE встроена в операционную систему Windows 2003 Server SP1.

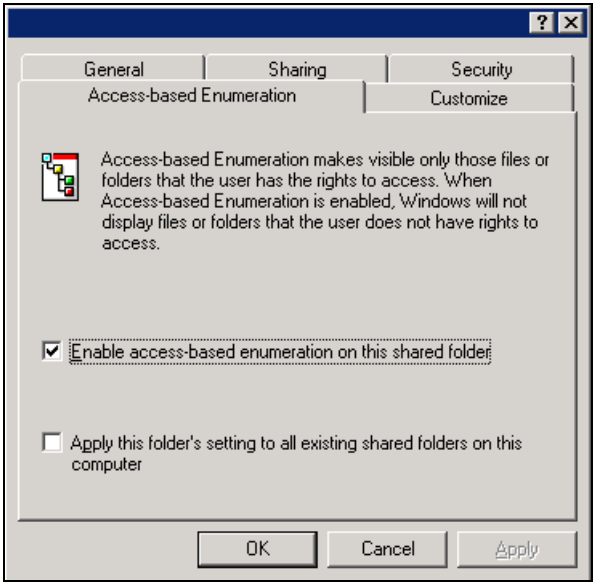



**Рис. 13.10.** Интеграция ABE в ОС

Для ее использования необходимо установить надстройку — графическую оболочку для управления отображением содержимого сетевых папок (рис. 13.10).

На сайте компании Microsoft доступен для загрузки установочный пакет ABEUI.msi версии 1.0.424.5 для Windows 2003 SP1, функционирующей на базе системы одного из трех типов: AMD64, I386. IA64. Выбрав необходимый тип платформы и загрузив файл из сети Интернет, приступим к установке ABE.

Работа мастера состоит из нескольких шагов. На первом шаге предстоит принять условия лицензионного соглашения. В противном случае продолжение процесса установки будет невозможно.

На следующем этапе необходимо указать каталог, в который будет установлена утилита (рис. 13.11). По умолчанию предлагается путь C:\Program Files\Microsoft Corporation\Windows Server 2003 Access-based Enumeration.

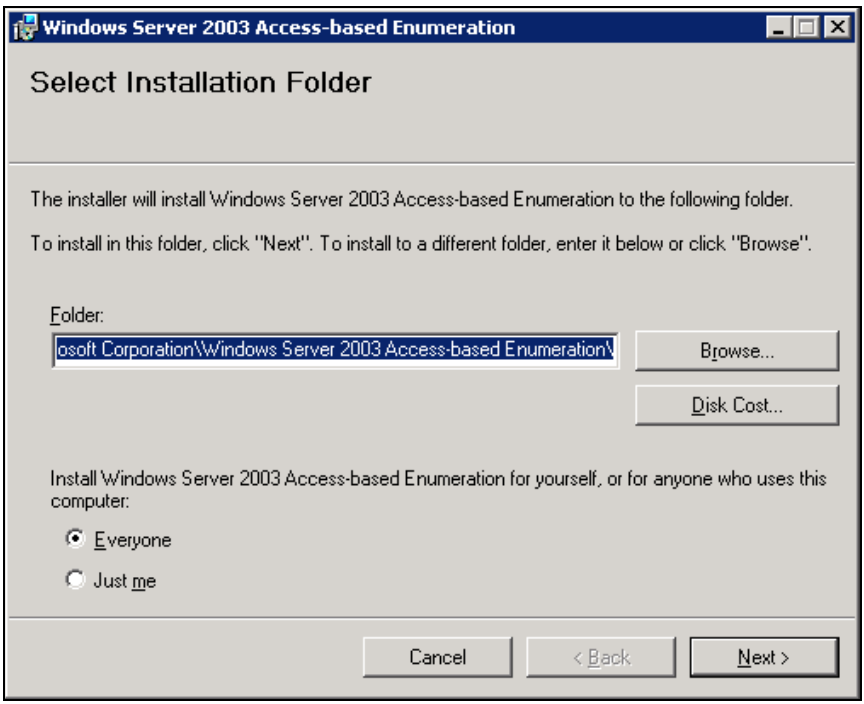



**Рис. 13.11.** Установка ABE. Выбор папки для установки

На последнем, третьем этапе установки необходимо выбрать, будет ли включен ABE сразу для всех папок, предоставленных в сетевое окружение на сервере, или каждая папка будет сконфигурирована отдельно (рис. 13.12).

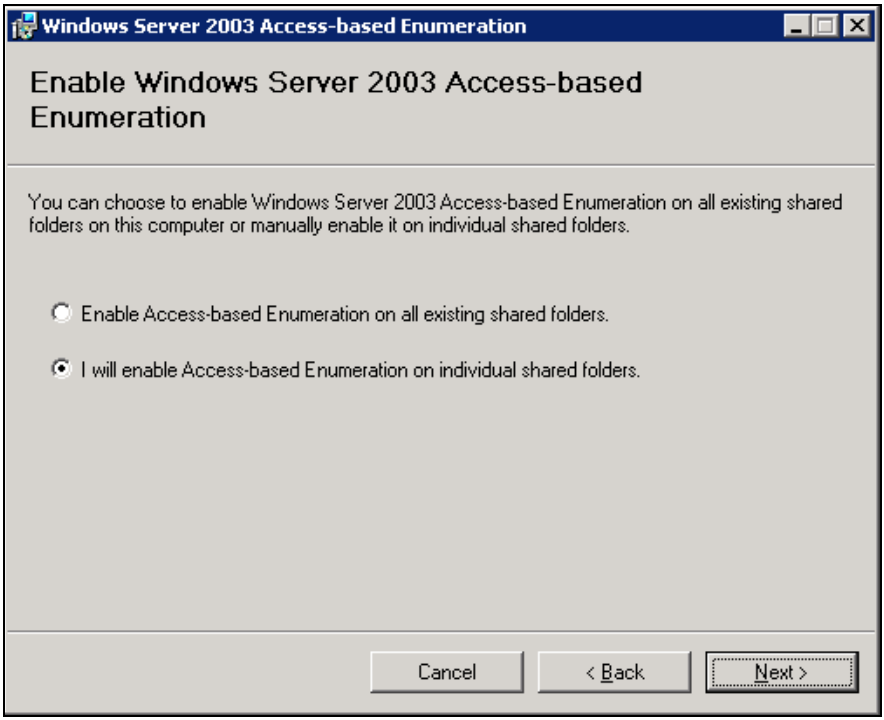



**Рис. 13.12.** Установка ABE. Настройка режима работы

## Ограничения технологии ABE

О существующих ограничениях технологии ABE на сайте Microsoft информация отсутствует. На практике опытным путем были обнаружены следующие ограничения:

- нельзя скрыть папки верхнего уровня (рис. 13.13), скрывать можно только папки второго и последующих уровней;
- установка возможна только на Windows Server 2003 SP1.




**Рис. 13.13.** Папки верхнего уровня

## Способы настройки ABE

Существует несколько способов настройки ABE для папки:

- графический интерфейс;
- утилита командной строки.

### Настройка ABE с помощью графического интерфейса

Вся настройка заключается во включении или отключении данного режима в свойствах папки, которая предоставлена в сетевой доступ (см. рис. 13.10).

### Настройка ABE с помощью командной строки

Настройка ABE в командной строке осуществляется с помощью утилиты abecmd.exe, которая находится в каталоге C:\WINDOWS\system32. Команда имеет следующие аргументы:

- `/enable` — включает ABE на сетевом ресурсе. Если дополнительно указан ключ `/all`, то осуществляется включение ABE на всех сетевых ресурсах данного сервера;
- `/disable` — выключает ABE на сетевом ресурсе. Если дополнительно указан ключ `/all`, то осуществляется выключение ABE на всех сетевых ресурсах данного сервера;
- `/server` — позволяет осуществлять управление ABE на удаленном сервере. Для того чтобы эта команда работала, необходима предварительная установка ABE на удаленном сервере под управлением Windows Server 2003 SP1;

- `/all` — осуществляется управление всеми сетевыми ресурсами на сервере;
- `<ShareName>` — имя папки, предоставленной в сетевое использование, для которой осуществляется настройка ABE.

Приведем пример командной строки, которая включает ABE на удаленном сервере MOON для сетевой папки TEST, расположенной на нем:

```
AbeCmd /enable /server MOON "TEST"
```

## Удаление ABE

Помните, что технология ABE встроена в операционную систему. Можно удалить лишь графическую оболочку. При этом все установки на существующие сетевые папки сохранятся. Графическая оболочка — всего лишь средство администрирования, а не сам функционал.

## Недостатки технологии ABE

Технология работает очень хорошо. Единственным ее недостатком является небольшая потеря производительности. Чем больше объектов в анализируемой папке, тем ниже скорость чтения данных. Microsoft заявляет, что при наличии 15 000 файлов время открытия папки увеличивается на 1—3 секунды. Чтобы избежать этого, рекомендуется создавать больше папок, делая файловую структуру анализируемой папки иерархической. При такой организации файловой системы падение скорости при открытии папки незаметно.

## Практика использования ABE

Чаще всего ABE применяется в больших сетях с распределенным доступом к сетевым ресурсам для значительного снижения сущностей распределенной файловой системы (DFS). Наилучший результат дает использование ABE совместно с DFS. Это позволит сформировать индивидуальное окружение для каждого пользователя, назначая нужные права доступа на объекты, минимизировать трудозатраты на администрирование и увеличить скорость работы сотрудников, обеспечив им быстрый поиск нужной информации в сети.



# Раздел 5

# Управление компьютером

## Глава 14



# Реестр Windows 2k

В этой главе кратко рассмотрены основы построения реестра, рассказано об основных подходах программного управления.

## Основные термины и понятия

- **Реестр** (registry) — это иерархическая база данных, содержащая настройки аппаратного и программного обеспечения компьютера.
- **Куст** (hive) — это подмножество разделов, подразделов и параметров реестра, которому сопоставлен набор вспомогательных файлов, содержащих их резервные копии.
- **Групповая политика** (group policy) — это набор правил, в соответствии с которыми производится настройка рабочей среды

## Историческая справка

Первый реестр появился в Windows NT. До этого в Microsoft Windows 3.*x* все настройки программного обеспечения хранились в конфигурационных файлах с расширением INI. Вся конфигурационная информация операционной системы находилась в двух файлах: System.ini и Win.ini. При установке любого приложения все его настройки сохранялись в одном из этих файлов. Приложения пользовались ограниченным количеством параметров. Оно определялось размером INI-файла, который не должен превышать 64 Кбайт. Чтобы обойти это ограничение, для каждой программы стал создаваться свой INI-файл, содержащий настройки. Со временем, из-за большого количества

конфигурационных файлов, производительность операционной системы значительно снижалась. В 1993 году была выпущена операционная система Microsoft Windows NT, в которой множество INI-файлов было заменено единой базой данных — реестром.

С точки зрения файловой системы реестр представляет собой файл с расширением DAT. Реестр хранится в файле Ntuser.dat, который находится в каталоге %WinDir%\Profiles. В Microsoft Windows принято использовать переменные среды. Переменная %WinDir% содержит полный путь к каталогу, в котором установлена ОС, например, C:\Windows.

## Основы построения реестра

Реестр состоит из разделов верхнего уровня, называемых кустами (hives):

- `HKEY_CLASSES_ROOT` (`HKCR`);
- `HKEY_CURRENT_USER` (`HKCU`);
- `HKEY_LOCAL_MACHINE` (`HKLM`);
- `HKEY_USER` (`HKU`);
- `HKEY_CURRENT_CONFIG` (`HKCC`).

Структура реестра такова: в каждом из кустов находятся ключи, образующие древовидную структуру, и в каждом из них могут содержаться параметры. Рассмотрим подробнее назначение кустов реестра.

В разделе `HKLM` находится информация об аппаратном и программном обеспечении, сведения о системе безопасности. Этот раздел — один из самых больших.

Раздел `HKCR` является виртуальной ссылкой на раздел `HKLM\Software\Classes`. В нем содержатся сведения обо всех расширениях файлов, определениях типов, ярлыках, привязке, классах идентификаторов и т. д.

Раздел `HKU` содержит настройки пользователя по умолчанию, в которые входят описания переменных среды, цветовых схем, шрифтов, сетевых настроек и т. д. Во время регистрации нового пользователя на рабочей станции на жестком диске для него создается новый профиль. Настройки, содержащиеся в профиле, копируются из куста `HKU`.

Раздел `HKCU` содержит сведения о текущем пользователе и имеет название, соответствующее значению идентификатора безопасности (SID) данного

пользователя. Каждый раз при перезагрузке компьютера этот раздел создается заново.

Раздел HKCC является ссылкой на текущий профиль оборудования, хранящийся в HKLM. С помощью профиля оборудования определяют список устройств, драйверы которых будут подгружены в данном сеансе работы пользователя. Профили изначально предназначены для ноутбуков.

Раздел HKCC не хранится в реестре, а динамически создается при загрузке операционной системы. Раздел содержит сведения о самонастраивающихся устройствах (Plug-and-Play).

Как и любая база данных, реестр поддерживает несколько типов данных (табл. 14.1).

***Таблица 14.1.*** *Типы данных, поддерживаемые реестром*

| Наименование | Тип данных | Описание |
|---|---|---|
| REG_NONE | Не определен | Зашифрованные данные |
| REG_SZ | Строка | Символьный текст |
| REG_EXPAND_SZ | Строка | Текст с переменными |
| REG_BINARY | Двоичный | Двоичные данные |
| REG_DWORD | Число | Численные данные |
| REG_DWORD_BIG_ENDIAN | Число | Данные с "не-интеловским" порядком байт |
| REG_LINK | Строка | Путь к файлу |
| REG_MULTI_SZ | Массив | Массив, элементы которого — строки |
| REG_RESOURCE_LIST | Строка | Список оборудования |
| REG_FULL_RESOURCE_ DESCRIPTOR | Строка | Идентификатор оборудования |
| REG_FULL_RESOURCE_LIST | Строка | Идентификатор оборудования |

Групповые политики, описанные в ADM-файлах, могут производить операции только со следующими типами данных: REG_SZ, REG_EXPAND_SZ, REG_DWORD.

# Синтаксис REG-файлов

Файлы с расширением REG предназначены для внесения изменений в реестр. С их помощью можно как добавлять, так и удалять данные. REG-файл представляет собой обычный текстовый файл в кодировке ASCII или Unicode.

Для экспорта данных из REG-файла в реестр используется команда `regedit.exe /s filename.reg`. Ключ `/s` предназначен для подавления запроса подтверждения на вносимые изменения в базе.

В начале любого REG-файла находится идентификатор. Для Windows 2k идентификатором является метка `Windows Registry Editor Version 5.00` (для Windows NT и 9*x* — `REGEDIT4`).

Общий вид REG-файла для добавления ключа:

```
Windows Registry Editor Version 5.00
[куст\путь к разделу]
"ключ"="тип:значение"
```

Если переменная имеет тип `REG_SZ`, то в этом случае тип данных не указывается явным образом. Пример REG-файла приведен в листинге 14.1.

Используемый для удаления раздела синтаксис отличается:

```
Windows Registry Editor Version 5.00
[-куст\путь к разделу]
```

А для удаления какого-либо параметра используется следующий формат:

```
Windows Registry Editor Version 5.00
[куст\путь к разделу]
"Параметр"=-
```

**Листинг 14.1. Добавление ключей в ветвь реестра Windows 2k**

```
Windows Registry Editor Version 5.00
[HKEY_CURRENT_USER\Identities]
"Migrated5"=dword:00000001
"Last Username"="Main Identity"
"Last User ID"="{DBE58550-FA28-4338-8C4C-26F838B95AFD}"
"Default User ID"="{DBE58550-FA28-4338-8C4C-26F838B95AFD}"
```

## Редакторы реестра

Используя редактор реестра, можно выполнить дополнительную настройку Windows. Перед тем как внести изменения в реестр, рекомендуется сделать резервные копии изменяемых ветвей.

В Windows 2k существуют два средства, предназначенных для работы с реестром: Regedit и Regedt32. Каждое из них обладает своим набором функций. Regedit — это средство редактирования реестра, входящее в состав всех версий Microsoft Windows. С его помощью можно удалять, сохранять, восстанавливать, выполнять поиск параметров и разделов реестра на локальном и удаленном компьютере. Отличительной чертой редактора является то, что изменения, сделанные в реестре, начинают действовать моментально.

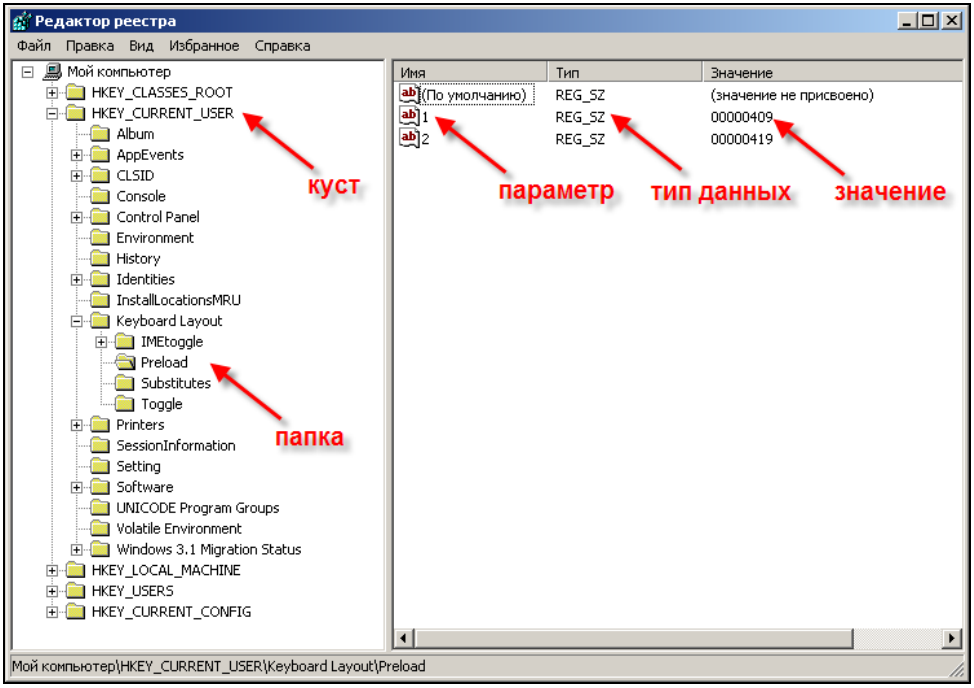



**Рис. 14.1.** Редактор реестра Regedt32

Regedt32 — редактор, появившийся в Windows 2k (рис. 14.1). Каждый раздел в данном редакторе располагается в отдельном окне. При удаленном доступе к реестру для редактирования доступны только два куста: `HKLM` и `HKU`. Редак-

тор позволяет менять права доступа для ветви реестра, что может использоваться, например, при перемещении пользователя из домена в домен. Для этого копируют профиль и изменяют права доступа на ветви. Regedt32 реализует изменения только после выгрузки редактируемого куста и закрытия редактора.

Запуск редакторов осуществляется с помощью утилит Regedit.exe и Regedt32.exe, входящих в Windows 2000 по умолчанию. В Windows XP оба редактора объединены в Regedit.exe, и Regedt32.exe просто вызывает этот редактор.

# Программное управление реестром

## Управление реестром с помощью WSH

Доступ к реестру из WSH-скриптов обеспечивается с помощью объекта `WshShell` и его методов `RegRead()`, `RegWrite()` и `RegDelete()`. Синтаксис доступа к реестру следующий:

- `WshShell.RegWrite "ключ реестра\ключ", "значение", "тип"`
- `WshShell.RegRead("ключ реестра\ключ")`
- `WshShell.RegDelete "ключ реестра\ключ"`

Как видно из приведенного синтаксиса, подразделы разделяются знаком обратной косой черты (\). Поскольку в первом аргументе команды `RegWrite` можно передать как имя раздела, так и название самого параметра, то признаком раздела считается обратная косая черта (\) в конце пути ветви реестра, например:

```
WshShell.RegWrite "HKCU\Test\", "";
```

В приведенном примере в кусте `HKCU` создается пустой раздел `Test`. Если же команда такая:

```
WshShell.RegWrite "HKCU\Test\Message", "Hello", "REG_SZ"
```

то создастся параметр `Message`, имеющий тип данных `REG_SZ` и значение Hello. При этом команда:

```
WshShell.RegWrite "HKCU\Test\Message\", "Hello", "REG_SZ"
```

создаст ключ реестра `HKCU\Test\Message` и запишет в него так называемое значение по умолчанию, равное Hello.

WSH-скрипт имеет расширение VBS и может использоваться как сценарий загрузки в Windows 2k. Полностью пример сценария приведен в листинге 14.2.

**Листинг 14.2. Запись данных в реестр с помощью WSH**

```
Set WshShell=WScript.CreateObject("WScript.Shell")
WshShell.RegWrite "HKCU\Test\Message", "Hello", "REG_SZ"
WScript.Echo "Реестр изменен!"
```

WSH сам по себе не позволяет выполнять перечисление параметров и подразделов внутри раздела. Сделать это позволяют API-функции. Для этого используется динамическая библиотека Regobj.dll, предоставляющая сценарию доступ к API-функциям через объектную модель. Библиотека поставляется с Microsoft Visual Basic, также может быть скопирована с Web-узла Microsoft: в поле "поиск" следует указать Regobj.dll.

Прежде чем использовать библиотеку, ее надо зарегистрировать командой:

```
"regsvr32 /s regobj.dll"
```

После этого доступ к ней осуществляется с помощью команды VBScript:

```
Set RegObject=WScript.CreateObject("RegObj.Registry")
```

После создания объекта используют метод `RegKeyFromString()`, чтобы получить объекты выбранного раздела. Полученные данные обрабатываются циклом `For Each` (листинг 14.3).

**Листинг 14.3. Чтение названий и значений подключей в указанной папке реестра**

```
Set RegObject=WScript.CreateObject("RegObj.Registry")
Set Rootkey=RegObject.RegKeyFromString(key)
Temp=""
For Each oVal In RootKey.SubKeys
    Temp = Temp & oVal.Name & vbTab & oVal.Value & vbCrlf
Next
WScript.Echo Temp
```

Свойство `SubKeys` предоставляет доступ ко всем разделам родительского раздела, который определяется методом `RegKeyFromString(key)`, где *key* —

путь к ветви реестра. Свойство `Name` содержит название раздела. Свойство `Values` возвращает набор, содержащий все параметры данного раздела. Переменная `vbCrlf` эквивалентна нажатию на клавишу <Enter>, `vbTab` эквивалента <Tab>.

Библиотека Regobj.dll не поддерживает новый тип параметров `REG_FULL_RESOURCE_DESCRIPTOR`, применяемый в Windows 2k. При попытке получить такой параметр возникнет ошибка времени выполнения.

С помощью этой же библиотеки можно получить доступ к удаленному компьютеру. Для этого используется метод `RemoteRegistry()` (листинг 14.4).

**Листинг 14.4. Доступ к реестру удаленного компьютера**

```
Set RegRemote=WScript.CreateObject("RegObj.Registry")
Set RegObject=RegRemote.RemoteRegistry(host)
Set Rootkey=RegObject.RegKeyFromString(key1)
```

Переменная `host` должна иметь вид \\имя_компьютера.

# Управление реестром с помощью групповых политик

Групповые политики (Group Policy, GP) представляют собой средство, обеспечивающее централизованное управление настройками рабочих станций и профилями пользователей. С их помощью определяют поведение операционной системы, рабочего стола, системы безопасности ОС, выключения компьютера, приложений и т. д. Групповые политики включают несколько компонентов (табл. 14.2).

***Таблица 14.2.*** *Компоненты групповых политик*

| Компонент | Описание |
|---|---|
| Administrative Templates | Политики, действие которых основано на изменении ветвей `HKLM`, `HCU` реестра |
| Security Settings | Настройка безопасности для домена, компьютеров и пользователей |

***Таблица 14.2*** *(окончание)*

| Компонент | Описание |
|---|---|
| Software Installation | Управление централизованной установкой программ из MSI-архивов |
| Internet Explorer Maintenance | Настройка браузера IE |
| Scripts | Настройка параметров при включении и выключении компьютера |
| Folder Redirection | Управление перенаправлением файлов и папок в сети |

С помощью политик можно изменить только ветви реестра `HKLM` и `HKU`. Консоль управления групповыми политиками (рис. 14.2) вызывают с помощью команды Gpedit.msc. Групповые политики логически делятся на две части: Computer Configuration и User Configuration. В разделе Computer Configuration находятся политики, посредством которых изменяется ветвь реестра `HKLM`, в разделе User Configuration, — соответственно, `HKU`. Информация о параметрах и конфигурации групповых политик собрана в `HKLM\Software\Policies` (предпочтительное расположение) или `HKLM\Software\Microsoft\Windows\CurrentVersion\Policies` для раздела Computer Configration; в `HKU\Software\Policies` (предпочтительное расположение) или `HKU\Software\Microsoft\Windows\CurrentVersion\Policies` для раздела User Configration.

## Синтаксис ADM-файлов

Административные шаблоны представляют собой текстовые файлы с расширением ADM. По умолчанию ADM-файлы находятся в каталоге %WinDir%\Inf. Рассмотрим возможности языка, с помощью которого создаются политики безопасности.

Синтаксис языка включает следующие ключевые компоненты:

- комментарии;
- строки;
- `CLASS`;
- `CATEGORY`;
- `POLICY`.

## Комментарии

Комментарии полезно использовать для документирования содержимого создаваемого шаблона (листинг 14.5). Комментарии предваряют точкой с запятой (`;`) или двумя прямыми косыми чертами (`//`). Их также можно помещать в конце строки.

**Листинг 14.5. Комментарий в ADM-файлах**

```
; Это комментарий
// И это комментарий
CLASS USER // Определение класса USER,
    // отвечающего за пользовательские настройки
CLASS MACHINE // Определение класса Machine,
    // отвечающего за общекомпьютерные настройки
```

## Строки

Всевозможные описания (описание политики, названия разделов, параметров и т. д.) рекомендуется располагать в специально предназначенном разделе `[strings]`. В тексте политики перед переменной, ссылающейся на текст в разделе `[strings]`, ставят два восклицательных знака `!!`. Каждая строка в разделе `[strings]` имеет формат `name="строка"`. Приведем два равнозначных примера (листинги 14.6 и 14.7). В первом примере не будем использовать преимущества строк.

**Листинг 14.6. Явное присвоение значения переменной**

```
CLASS Machine
POLICY "Пример политики"
…
END POLICY
```

**Листинг 14.7. Неявное присвоение значения переменной. Рекомендуемый метод**

```
CLASS Machine
POLICY !!Pname
…
```

```
END POLICY
[Strings]
Pname="Пример политики"
```

Использование строк позволяет создавать шаблоны групповых политик, что снижает трудозатраты и ускоряет процесс их создания.

## Ключевое слово *CLASS*

Первым элементом в файле, содержащем шаблон групповой политики, является ключевое слово `CLASS`, определяющее к какому типу относится описываемая ниже политика: компьютерная (Computer Configuration) или пользовательская (User Configuration). В одном файле допускается многократное использование ключевого слова `CLASS`. Синтаксис элемента `CLASS`:

```
CLASS Name
```

Здесь `Name` — параметр, принимающий одно из следующих двух значений: `USER` или `MACHINE`. Значение `USER` определяет, что политика пользовательская. Изменения будут вноситься в ветвь реестра `HKU`, настройку политики следует осуществлять в разделе User Configuration\Administrative Templates. Значение `MACHINE`, соответственно, определяет, что политика компьютерная и изменения будут вноситься в ветвь `HKLM` реестра, а настройку политики следует осуществлять в разделе Computer Configuration\Administrative Templates.

## Ключевое слово *CATEGORY*

После определения класса политики необходимо обозначить местоположение в папке Administrative Templates. Ключевым словом `CATEGORY` описываются подпапки. Описание категории необходимо закрыть ключевым словом `END CATEGORY`. Категории поддерживают вложенность. Синтаксис элемента `CATEGORY`:

```
CATEGORY Name
KEYNAME SubKey
…
END CATEGORY
```

Здесь `Name` — имя папки, которое будет отображаться в редакторе Group Policy, а `SubKey` — необязательный параметр, задающий путь к разделу реестра, относительно которого задаются все остальные параметры реестра. Для зада-

ния этих параметров используйте строковую переменную или строку, заключенную в кавычки.

Информация о параметрах и конфигурации групповых политик сохраняется:

- в ключе `HKLM\Software\Policies` (предпочтительное расположение) или `HKLM\Software\Microsoft\Windows\CurrentVersion\Policies` для раздела Computer Configuration;
- в ключе `HKU\Software\Policies` (предпочтительное расположение) или `HKU\Software\Microsoft\Windows\CurrentVersion\Policies` для раздела User Configuration.

Корневой ключ (`HKLM`, `HKU`) использовать в пути не следует, поскольку он уже описан ключевым словом `CLASS`. Если путь содержит пробелы, заключайте его в кавычки.

В разделе `CATEGORY` могут быть использованы следующие ключевые слова: `CATEGORY`, `END`, `KEYNAME`, `POLICY`.

**Листинг 14.8. Создание подпапок в групповой политике**

```
CLASS USER
CATEGORY "POLICIES"
    CATEGORY !!SubPol1
    KEYNAME "Software\Policies\SubPol1"
    …
    END CATEGORY

    CATEGORY !!SubPol2
    KEYNAME "Software\Policies\SubPol1"
    …
    END CATEGORY
END CATEGORY

[strings]
SubPol1="Policy1"
SubPol2="Policy2"
```

В приведенном примере (листинг 14.8) показано, как создать в указанной папке Policies две подпапки — Policy1 и Policy2 (рис. 14.2).

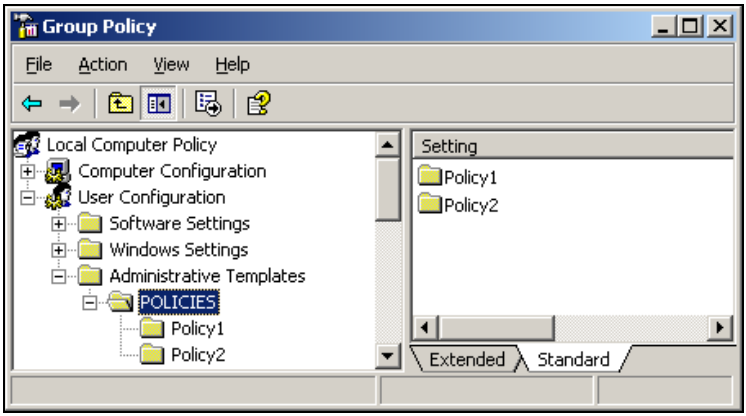



**Рис. 14.2.** Созданные папки групповых политик

## Ключевое слово *POLICY*

Ключевое слово `POLICY` используют для определения политики. В одну категорию может быть включено несколько разделов `POLICY`, каждый из которых должен заканчиваться инструкцией `END POLICY`. Синтаксис элемента `POLICY`:

```
POLICY Name
[KEYNAME SubKey]
EXPLAIN Help
VALUENAME Value
[PARTS]
...
END POLICY
```

Здесь используются следующие параметры:

- `Name` — название политики, отображаемое в редакторе Group Policy. Используйте строковую переменную или строку, заключенную в кавычки;
- `SubKey` — необязательный параметр, задающий расположение информации о политике. Информация о параметрах и конфигурации групповых политик сохраняется:
  - в ключе `HKLM\Software\Policies` (предпочтительное расположение) или `HKLM\Software\Microsoft\Windows\CurrentVersion\Policies` для раздела Computer Configuration;

- в ключе `HKU\Software\Policies` (предпочтительное расположение) или `HKU\Software\Microsoft\Windows\CurrentVersion\Policies` для раздела User Configuration.

Не используйте в пути корневой ключ (`HKLM`, `HKU`), поскольку он уже описан ключевым словом `CLASS`. Если путь содержит пробелы, заключайте его в кавычки. Ко всем политикам применяется последнее ключевое слово `KEYNAME`;

- `Help` — подробное описание политики, которое отображается в окне **Group Policy** в разделе **Extended** или на вкладке **Explain** свойств политики. Для начала новой строки используйте `\n`;
- `Value` — каждая политика содержит ключевое слово `VALUENAME`, которое связывает с ней значение реестра. По умолчанию редактор политик предполагает, что это значение типа `REG_DWORD` и имеет значение `1 (0x01)`, когда политика включена. Редактор политики удаляет значение, если политика выключена. В том случае, если необходимо сохранить это значение в реестре после удаления политики — используйте ключевые слова `VALUEON` и `VALUEOFF`, которые следуют непосредственно за словом `VALUENAME`:

```
VALUEON [NUMERIC] VValue
VALUEOFF [NUMERIC] VValue
```

По умолчанию тип данных значения `VValue` — `REG_SZ`. Если после ключевого слова указано `NUMERIC`, то тип данных значения `VValue` — `REG_DWORD`.

## Управление интерфейсом групповых политик

Программирование интерфейса настройки групповых политик сводится к созданию раскрывающихся списков, флажков, текстовых полей и т. д. Рассмотрим по порядку процедуры создания всех элементов интерфейса.

В первом варианте — самом простом, нет никаких управляющих элементов, кроме опции включения и выключения политики. В файле политики записан ключ реестра и значение, имеющее тип `REG_SZ` или `REG_DWORD`, которое может меняться в зависимости от того, включена политика или нет. Приведем пример, в котором при включенной политике переключение раскладки клавиатуры в диалоговом окне регистрации пользователя в сети устанавливает сочетание клавиш <Ctrl>+<Shift>, а при выключенной — <Alt>+<Shift>.

За переключение раскладки языка отвечает параметр `Hotkey`, принимающий значение 2 (<Ctrl>+<Shift>) или 1 (<Alt>+<Shift>). Он расположен в разделе HKEY_USER\Keyboard Layout\Toggle. Параметр `Hotkey` имеет тип данных `REG_SZ`. Приступим к созданию политики (листинг 14.9). Поскольку необходимо редактировать ветвь реестра `HKU`, то политика является пользовательской и будет создана в разделе User Configuration\Administrative Templates. Параметр `KEYNAME` раздела `CATEGORY` принимает значение `Keyboard Layout\Toggle`. В разделе `POLICY` параметр `VALUENAME` — `Hotkey`. `VALUEON` — `2`, а `VALUEOFF` — `1`.

**Листинг 14.9. ADM-файл политики**

```
CLASS USER
CATEGORY "Admin Policies"
    CATEGORY "Keyboard Toggle "
        KEYNAME ".Default\Keyboard Layout\Toggle"
        POLICY "Toggle Policy"
            EXPLAIN !!help
            VALUENAME "Hotkey"
            VALUEON 2
            VALUEOFF 1
        END POLICY
    END CATEGORY
END CATEGORY
[strings]
help="Управление переключением раскладки клавиатуры при регистрации пользователя в сети \n\n. При включенной политике переключение раскладки клавиатуры осуществляется с помощью комбинации клавиш CTRL+SHIFT, при выключенной - ALT+SHIFT."
```

Все остальные элементы интерфейса (флажки, выпадающие меню и т. д.) можно реализовать только в разделе `PART` — члене раздела `POLICY`. Синтаксис раздела `PART` следующий:

```
PART Name Type
Keywords
[KEYNAME SubKey]
[DEFAULT Default]
VALUENAME Value
END PART
```

Здесь используются следующие параметры:

- `Name` — название раздела, которое будет отображено в консоли во время редактирования политики;
- `Type` — соответствует одному из типов элементов управления (все они перечислены в табл. 14.3);
- `Keywords` — ключевые слова, меняющие поведение элементов. Они специфичны для каждого из типов, поэтому описаны далее при рассмотрении конкретных типов;
- `SubKey` — необязательный подключ `HKLM` или `HKU`. Не используйте в пути корневой ключ (`HKLM`, `HKU`), поскольку он уже описан ключевым словом `CLASS`. Если путь содержит пробелы, заключайте его в кавычки;
- `Default` — значение параметра по умолчанию. Когда администратор включает политику, то считываются параметры по умолчанию;
- `Value` — модифицируемое значение реестра; тип и значения данных полностью зависят от типа параметра.

***Таблица 14.3.*** *Доступные элементы управления в ADM-файлах*

| Тип | Описание | Тип данных |
|---|---|---|
| CHECKBOX | Отображает флажок. При установленном флажке принимает значение 1, при снятом — 0 | REG_DWORD |
| COMBOBOX | Отображает список с полем ввода | REG_SZ (по умолчанию), REG_EXPAND_SZ (при наличии метки EXPANDABLETEXT) |
| DROPDOWNLIST | Отображает раскрывающийся список с полем ввода. Пользователь может выбрать только одно из заданных значений | |
| EDITTEXT | Отображает текстовое поле, в котором можно вводить буквы и цифры | |
| LISTBOX | Отображает список с кнопками **Add** и **Remove** | – |
| NUMERIC | Отображает текстовое поле, позволяющее вводить только число | REG_DWORD |
| TEXT | Отображает текстовую строку (метку). Не сохраняет в реестре никаких значений. Отображается только в виде подсказок в диалоговом окне настройки групповой политики | |

## Тип *CHECKBOX*

С помощью ключевого слова CHECKBOX в политике безопасности отображается флажок. По умолчанию флажок сброшен. Если требуется, чтобы по умолчанию флажок был установлен, необходимо в теле раздела PART указать ключевое слово DEFCHECKED. При установленном флажке в реестр записывается значение 1, если он сброшен — 0.

Все переменные реестра соответствуют типу данных REG_SZ. Если необходимо записать данные в реестр в формате REG_DWORD, то следует после значений ключевых слов VALUEON и VALUEOFF добавить NUMERIC.

Вот полный синтаксис элемента CHECKBOX:

```
PART Name CHECKBOX
[DEFCHEKED]
VALUENAME ValueName
VALUEON [NUMERIC] Value1
VALUEOFF [NUMERIC] Value2
END PART
```

В нем используются такие параметры:

- Name — название раздела, которое будет отображено в консоли во время редактирования политики. Если в названии раздела встречаются пробелы, заключите его в кавычки, в противном случае значение будет считано до первого пробела;
- ValueName — название параметра, которое необходимо изменить;
- Value1 — значение, задаваемое при установленном флажке и включенной политике;
- Value2 — значение, задаваемое при снятом флажке и включенной политике.

## Тип *COMBOBOX*

С помощью ключевого слова COMBOBOX в диалоговое окно политики добавляют список с текстовым полем. Внутри COMBOBOX включен обязательный блок SUGGESTIONS, заканчивающийся инструкцией END SUGGESTIONS. В этом блоке перечислены элементы, заключенные в кавычки и разделенные пробелами.

```
PART Name COMBOBOX
SUGGESTIONS
```

```
"Suggestion1"  "Suggestion2" …  "Suggestionm"
END SUGGESTIONS
[DEFAULT Default]
[EXPANDABLETEXT]
[MAXLENGTH Max]
[NOSORT]
[REQUIRED]
VALUENAME ValueName
END PART
```

Кроме блока `SUGGESTIONS` могут использоваться следующие ключевые слова и параметры:

- `Name` — название раздела, которое будет отображено в консоли во время редактирования политики;
- `DEFAULT` — указывает значение списка по умолчанию;
- `EXPANDABLETEXT` — создает значение типа `REG_EXPAND_SZ`;
- `MAXLENGTH` — указывает максимальную длину строки;
- `NOSORT` — отключает сортировку списка редактором политик;
- `REQUIRED` — указывает, что значение обязательно;
- `VALUENAME` — Имя модифицируемого параметра реестра. По умолчанию имеет тип данных `REG_SZ`.

## Тип *DROPDOWNLIST*

С помощью ключевого слова `DROPDOWNLIST` в политику добавляют диалоговое окно — раскрывающийся список. Внутри `DROPDOWNLIST` должен присутствовать блок `ITEMLIST`, заканчивающийся инструкцией `END ITEMLIST`. В нем перечислены элементы раскрывающегося списка.

Синтаксис элемента `DROPDOWNLIST`:

```
PART Name DROPDOWNLIST
ITEMLIST
NAME Item VALUE Data
END ITEMLIST
[DEFAULT Default]
[EXPANDABLETEXT]
```

```
[NOSORT]
[REQUIRED]
VALUENAME Value
END PART
```

Здесь присутствуют следующие элементы:

- `Name` — название раздела, которое будет отображено в консоли во время редактирования политики;
- `Item` — имя каждого из элементов раскрывающегося списка;
- `Data` — значение, соответствующее отображаемому элементу `Item`;
- `DEFAULT` — значение по умолчанию в сформированном списке, которое считывается при включении политики;
- `EXPANDABLETEXT` — создает значение типа `REG_EXPAND_SZ`;
- `NOSORT` — отключает сортировку списка редактором политик;
- `REQUIRED` — указывает обязательное значение;
- `VALUENAME` — имя модифицируемого параметра реестра. По умолчанию имеет тип данных `REG_SZ`.

## Тип *EDITTEXT*

Ключевое слово `EDITTEXT` позволяет вводить в диалоговом окне политики произвольный текст. Синтаксис элемента `EDITTEXT`:

```
PART Name EDITTEXT
[DEFAULT Default]
[EXPANDABLETEXT]
[MAXLEN Max]
[NOSORT]
[REQUIRED]
VALUENAME Value
END PART
```

В описании поля используются следующие элементы:

- `Name` — название раздела, отображаемое в консоли во время редактирования политики;
- `DEFAULT` — задает значение списка по умолчанию;
- `EXPANDABLETEXT` — создает значение типа `REG_EXPAND_SZ`;

- MAXLEN — указывает максимальную длину строки;
- REQUIRED — показывает, что значение обязательно;
- VALUSENAME — имя модифицируемого параметра реестра, который по умолчанию имеет тип данных REG_SZ.

## Тип *LISTBOX*

Ключевое слово LISTBOX добавляет в диалоговое окно политик список с кнопками **Add** и **Remove**. Это единственный элемент, который может быть использован системным администратором для управления несколькими значениями, содержащимися в одном ключе. В разделе LISTBOX невозможно использовать опцию VALUENAME, поскольку он не связан с каким-либо конкретным значением.

Синтаксис элемента LISTBOX:

```
PART Name LISTBOX
[EXPANDABLETEXT]
[NOSORT]
[ADDITIVE]
[EXPLICITVALUE| VALUEPREFIX Prefix]
END PART
```

В описании списка используются следующие конструкции:

- Name — название раздела, которое отображается в консоли редактирования политики.
- EXPANDABLETEXT — создает значение типа REG_EXPAND_SZ;
- NOSORT — отключает сортировку списка редактором политик;
- ADDITIVE — считывает значения, уже хранящиеся в реестре;
- EXPLICITVALUE — ключевое слово, которое позволяет указать имя и данные значения. Отображаемый список имеет два столбца: один для имени, другой для данных. Невозможно использовать эту инструкцию совместно с VALUEPREFIX;
- VALUEPREFIX — указанный префикс определяет имена значений. Если указывается префикс, то редактор групповых политик добавляет к нему возрастающий номер. В том случае, если префикс пустой, то генерируются только возрастающие номера;

- Prefix — используется для создания последовательности имен. Если указан префикс, например Sample, то генерируются следующие имена значений: Sample1, Sample2, …, Sample*n*. Если префикс не указан, то генерируются только порядковые номера: 1, 2, 3,…, *n*.

## Тип *NUMERIC*

Ключевое слово NUMERIC позволяет вводить цифровые значения, используя элемент управления "счетчик", который увеличивает или уменьшает число. По умолчанию переменные имеют тип данных REG_DWORD, однако при использовании инструкции TXTCONVERT данные могут быть сохранены в формате REG_SZ.

Синтаксис элемента NUMERIC:

```
PART Name NUMERIC
[DEFAULT Default]
[MAX Max]
[MIN Min]
[REQUIRED]
[SPIN]
[TXTCONVERT]
VALUENAME Value
END PART
```

В описании используются следующие элементы:

- Name — название раздела, отображаемое в консоли редактирования политики;
- DEFAULT — значение элемента по умолчанию;
- MAX и MIN — максимальное и минимальное значения, по умолчанию равные 9999 и 0 соответственно;
- REQUIRED — указывает, что это обязательное значение;
- SPIN — устанавливает шаг арифметической прогрессии, который по умолчанию равен 1. Указание значения 0 делает счетчик невидимым;
- TXTCONVERT — инструктирует сохранять данные в формате REG_SZ вместо типа REG_DWORD, используемого по умолчанию;
- VALUNAME — задает имя модифицируемого параметра реестра, который по умолчанию имеет тип данных REG_DWORD.

### Тип *TEXT*

Ключевое слово `TEXT` добавляет в диалоговое окно статический текст. Синтаксис элемента `TEXT`:

```
PART Text TEXT
END PART
```

Единственный параметр в этой конструкции, `Text`, задает статический текст, добавляемый в диалоговое окно; если текст содержит пробелы, то его следует заключать в кавычки.

## Примеры использования административных шаблонов на практике

В корпоративной сети, как правило, существуют один или несколько файловых серверов, в которых хранятся дистрибутивы программного обеспечения, в том числе дистрибутив Microsoft Office.

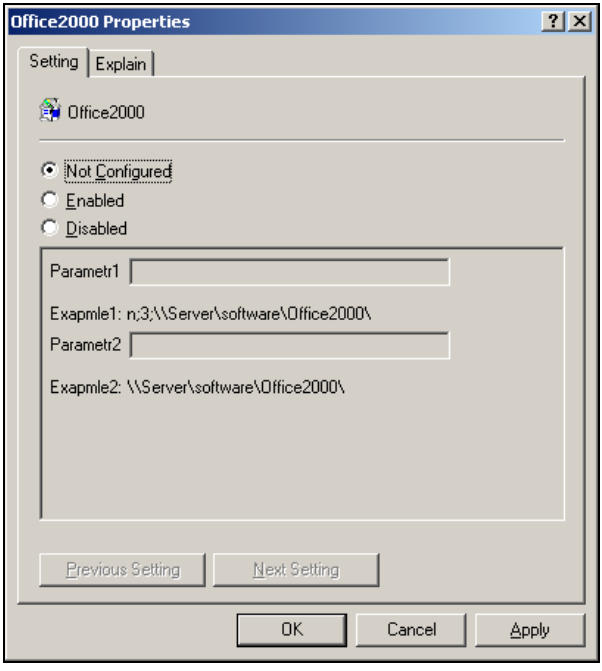



**Рис. 14.3.** Окно конфигурации политики

Если пользователю по какой-либо причине необходимо перейти с одной рабочей станции на другую, то при первой загрузке для его учетной записи создается новый профиль. При первом запуске одного из компонентов MS Office программа обращается к дистрибутиву, который расположен на файловом сервере. При изменении местоположения дистрибутива на сервере пользователю выдается сообщение о невозможности завершить процесс конфигурации программы и просьба обратиться к системному администратору. Во избежание этой неприятной ситуации рекомендуется использовать групповую политику, которая меняет в реестре пути к дистрибутиву MS Office (листинг 14.10, рис. 14.3).

**Листинг 14.10. Политика изменения пути к дистрибутиву Microsoft Office**

```
LASS Machine
CATEGORY "Office 2000"
KEYNAME "Software\Policies"

POLICY "Office2000"
EXPLAIN !!help

    PART "Parametr1" EDITTEXT
    KEYNAME "SOFTWARE\Classes\Installer\Products\
914000001E872D116BF00006799C897E\SourceList"
    VALUENAME LastUsedSource
    MAXLEN 100
    EXPANDABLETEXT
    END PART

PART !!help1 TEXT
END PART

    PART "Parametr2" EDITTEXT
    KEYNAME "SOFTWARE\Classes\Installer\Products\
914000001E872D116BF00006799C897E\SourceList\Net"
    VALUENAME 1
    MAXLEN 100
    EXPANDABLETEXT
    END PART
```

```
PART !!help2 TEXT
END PART
END POLICY
END CATEGORY

[strings]
help1="Exapmle1: n;3;\\Server\software\Office2000\"
help2="Exapmle2: \\Server\software\Office2000\"
help="\n\nCorrecting path to Microsoft Office 2000"
```

## Внедрение административных шаблонов

Для загрузки ADM-файла, содержащего политику, необходимо выполнить следующие действия:

1. Загрузить консоль групповых политик, выполнив команду `Gpedit.msc`.
2. В любом из разделов (Computer Configuration\Administrative Templates или User Configuration\Administrative Templates) вызвать контекстное меню, нажав правую кнопку мыши.
3. В меню выбрать **Add/Remove Templates**.
4. В диалоговом окне нажать кнопку **Add**. В появившемся окне, указав путь к файлу с расширением ADM, нажать кнопку **Open**.
5. Нажать кнопку **Close**, после чего программа проверит синтаксис файла; в случае синтаксической ошибки будет указана строка и предпочтительный вариант исправления этой ошибки.

## Удаление административного шаблона

Для удаления политики, подгруженной с помощью ADM-файла, необходимо выполнить следующую последовательность действий:

1. Загрузить консоль групповых политик, выполнив команду `Gpedit.msc`.
2. В любом из разделов (Computer Configuration\Administrative Templates или User Configuration\Administrative Templates) вызвать контекстное меню, нажав правую кнопку мыши.
3. В появившемся меню выбрать **Add/Remove Templates**.
4. В открывшемся диалоговом окне выбрать файл описания политики, которую необходимо удалить. Нажать кнопку **Remove**, затем **Close**.

ГЛАВА 15



# Windows Script Host

В этой главе приведена объектная модель Windows Script Host (WSH), который был предложен Microsoft в качестве инструмента для разработки сценариев. Сценарии WSH поддерживают два встроенных в Microsoft Windows языка программирования — Visual Basic Script Edition (VBScript) и Java Script Edition (JScript).

## Основные термины и понятия

- **Специальная папка** (Special folder) — папка на диске, которую операционная система использует в определенных, специфических для этой папки целях.
- **Сценарий** (Script) — последовательность команд на определенном языке программирования, предназначенных для выполнения какой-то цели, которой невозможно добиться одной командой. Чаще всего сценарии пишутся на интерпретируемых языках.
- **Ярлык** (Shortcut) — файл, воспринимаемый операционной системой как ссылка на другой файл, папку или ресурс в сети Интернет.

## Сервер сценариев Windows Script

Не все версии Windows обеспечивают одинаковый уровень поддержки для работы с административными сценариями. В табл. 15.1 отображена история развития WSH. В настоящее время используется версия WSH 5.7, которая вышла в свет 24 августа 2007 года. Она по умолчанию входит в дистрибутив Windows Vista. До WSH 5.7 может быть обновлена предыдущая версия для

операционных систем Windows 2000, Windows XP, Windows 2003 Server. Обновление можно скачать с сайта Microsoft.

***Таблица 15.1.*** *История версий WSH*

| Версия | Операционная система |
|---|---|
| WSH 1.0 | Windows 95, Windows 98, Windows NT 4.0 |
| WSH 2.0 | Windows 2000, Windows ME |
| WSH 5.6 | Windows XP, Windows 2003 Server |
| WSH 5.7 | Windows Vista |

В состав пакета Microsoft Windows Script 5.7 входят VBScript, JScript, Windows Script Host, Windows Script Components и Windows Script Runtime.Этого набора достаточно для создания работоспособных сценариев на VBScript или JScript.

## Запуск сценариев WSH из командной строки

Для выполнения сценариев с помощью сервера сценариев для командной строки (Cscript.exe) используется следующий синтаксис:

```
cscript [имя_сценария] [параметры_сервера] [аргументы_сценария]
```

При этом используются следующие параметры:

- `имя_сценария` — имя файла сценария, включая путь и расширение;
- `параметры_сервера` — ключи командной строки, задающие различные свойства сервера сценариев Windows. Параметр сервера всегда начинается с двух косых черт (//);
- `аргументы_сценария` — ключи командной строки, которые передаются в сценарий. Аргумент сценария всегда начинается с одной косой черты (/).

Все параметры являются необязательными, однако нельзя задать аргументы сценария, не задав сценарий. При отсутствии среди аргументов самого сценария программа Cscript.exe выведет описание синтаксиса команды и предусмотренные ключи. Ключи сервера сценариев для командной строки описаны в табл. 15.2.

***Таблица 15.2.*** *Параметры командной строки Cscript.exe*

| Параметр | Действие |
|---|---|
| `//B` | Включение пакетного режима, при котором не выводятся подсказки и сообщения об ошибках |
| `//D` | Включение отладчика |
| `//E:обработчик` | Обработчик, используемый для выполнения сценария |
| `//H:CScript` или `//H:WScript` | Регистрация Cscript.exe или Wscript.exe в качестве сервера сценариев, применяемого по умолчанию. Если ключ не задан, используется Wscript.exe |
| `//I` | Включение интерактивного режима, в котором выводятся подсказки и сообщения об ошибках. Применяется по умолчанию; отменяет ключ `//B` |
| `//Job:xxxx` | Выполнение задания `xxxx`, указанного в файле сценария WSF |
| `//Logo` | Отображение перед выполнением сценария эмблемы сервера сценариев Windows. Применяется по умолчанию |
| `//Nologo` | Скрытие перед выполнением сценария эмблемы сервера сценариев Windows; отменяет режим `//Logo` |
| `//S` | Сохранение параметров текущей командной строки для данного пользователя |
| `//T:nnnnn` | Указание максимального времени (в секундах), отведенного на выполнение сценария. Можно указать значение времени до 32 767 секунд. По умолчанию время выполнения не ограничено |
| `//X` | Запуск сценария в отладчике |
| `//?` | Вывод предусмотренных параметров командной строки и встроенной справки (аналогично запуску программы Cscript.exe без аргументов и без задания сценария) |

Параметр времени ожидания (`//T:nnnnn`) ограничивает максимальное время выполнения сценария. Если время выполнения превышает заданную величину, Cscript.exe прерывает работу обработчика и останавливает процесс.

Файлы с расширением WSF, которые можно использовать при работе с сервером сценариев Windows, позволяют вызвать несколько обработчиков сценариев и выполнить несколько заданий, в том числе написанных на разных языках сценариев.

# Возможности WSH-сценариев

WSH-сценарии позволяют осуществлять следующие действия:

- вывод информации в виде диалоговых сообщений;
- считывание информации из потока данных;
- управление процессами Microsoft Windows;
- работа с локальной сетью — подключение сетевых дисков и принтеров;
- работа с переменными среды;
- работа со специальными папками;
- создание ярлыков;
- работа с системным реестром;
- эмулирование нажатия клавиш;
- управление OLE-объектами.

# Объектная модель WSH

Объектная модель WSH представляет собой иерархически упорядоченную коллекцию объектов (рис. 15.1), содержащуюся в файле Wshom.ocx, который располагается в каталоге %WinDir%\System32.

Объектную модель WSH составляют следующие объекты:

- `WScript` — основной объект WSH, предназначенный для создания объектов и связи с ними, определения имени запускаемого скрипта и версии WSH; позволяет считывать данные с клавиатуры и выводить информацию на экран;
- `WshArguments` — обеспечивает доступ к параметрам командной строки запущенного сценария или ярлыка Microsoft Windows;

- `WshShell` — позволяет запускать процессы, создавать ярлыки, работать с системным реестром и специальными папками Microsoft Windows;
- `WshEnvironment` — служит для работы с переменными окружения;

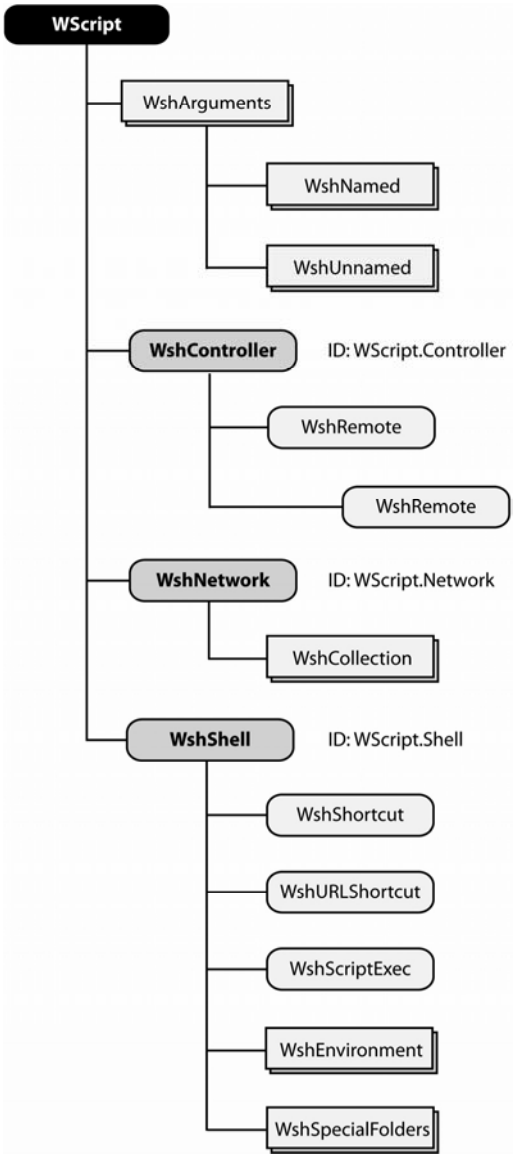



**Рис. 15.1.** Объектная модель WSH

- `WshSpecialFolders` — позволяет получить доступ к специальным папкам Windows;
- `WshNetwork` — используется для доступа к локальной сети; позволяет подключать сетевые устройства: принтеры и диски;
- `WshShortcut` — обеспечивает работу с ярлыками Windows;
- `WshUrlShortcut` — обеспечивает работу с ярлыками сетевых ресурсов Windows.

# Объект *WScript*

Свойства объекта `WScript` позволяют получить полный путь к использующемуся серверу сценариев (Wscript.exe или Cscript.exe), параметрам командной строки, с которыми он запущен (см. табл. 15.2) и т. д. В сценарии WSH-объекты используются без какого-либо предварительного описания. Для доступа к остальным объектам используют метод `CreateObject()` или определенное свойство другого объекта.

## Свойства объекта *WScript*

Поддерживаемые объектом `WScript` свойства приведены в табл. 15.3.

***Таблица 15.3.*** *Свойства объекта* `WScript`

| Свойство | Описание |
|---|---|
| `Arguments` | Содержит указатель на список параметров командной строки, с которой запущен сценарий |
| `FullName` | Содержит полный путь к исполняемому файлу сервера сценариев |
| `Name` | Содержит название объекта `WScript` (Windows Script Host) |
| `Path` | Содержит путь к каталогу, в котором расположен сервер сценариев |
| `ScriptFullName` | Содержит полный путь к исполняемому сценарию |
| `ScriptName` | Содержит имя запущенного сценария |

***Таблица 15.3*** *(окончание)*

| Свойство | Описание |
|---|---|
| `StdErr` | Позволяет запущенному сценарию записывать сообщения в стандартный поток для вывода ошибок |
| `StdIn` | Позволяет запущенному сценарию считывать данные из стандартного входного потока |
| `StdOut` | Позволяет запущенному сценарию записывать данные в стандартный выходной поток |
| `Version` | Содержит версию WSH |

## Свойство *Arguments*

Свойство `Arguments` используется для анализа скриптом параметров из командной строки. Используя это свойство, при выполнении сценария можно узнать заданные параметры, выяснить их количество.

В листинге 15.1 приведен файл who.vbs, который определяет, с какими параметрами был запущен сценарий (рис. 15.2).

**Листинг 15.1. Определение параметров запуска скрипта**

```
Set objArgs = WScript.Arguments
t = "Количество заданных параметров - " & objArgs.Count & chr(13) & _
    "Параметры:" & chr(13)
For Each arg in objArgs
    t = t & arg & chr(13)
Next
MsgBox t
```

Результат, показанный на рис. 15.2, получается при следующем вызове скрипта:

```
Who.vbs Petrov Nikolay
```

Аргументы сценария можно считывать другим способом (листинг 15.2). Оба способа равнозначны. Отметим лишь, что первый вариант компактнее, а второй — предпочтительнее, если необходимо использовать нумерацию объектов, поскольку в первом отсутствует счетчик.

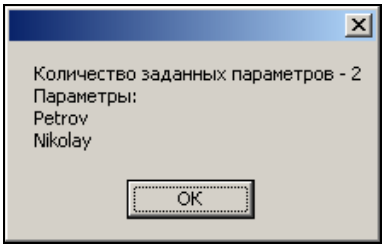



**Рис. 15.2.** Сообщение
о параметрах запуска сценария

**Листинг 15.2. Определение параметров запуска скрипта со счетчиком**

```
Set objArgs = WScript.Arguments
Coun = WScript.Arguments.Count
t = "Количество заданных параметров - " & Coun & chr(13) & _
    "Параметры:" & chr(13)
For i = 0 To Coun-1
    t = t & i+1 & ": " & objArgs(i) & chr(13)
Next
MsgBox t
```

Для определения значения заданного аргумента используется подход, пример которого представлен в листинге 15.3.

**Листинг 15.3. Определение значения заданного аргумента**

```
Set objArgs = WScript.Arguments
MsgBox objArgs(2)              ' Чтение третьего аргумента
```

Нумерация элементов начинается с 0. Приведенный пример не является лучшим образцом программирования, поскольку в нем отсутствует обработчик ошибок. Использование обработчика ошибок, имитация простейшего интеллекта у программы является хорошим тоном программирования. Создадим для этого примера обработчик ошибок. Ошибка может возникнуть, если предпринята попытка получить несуществующее значение (листинг 15.4).

**Листинг 15.4. Обработчик ошибок**

```
Set objArgs = WScript.Arguments
Coun = WScript.Arguments.Count
q = InputBox("Введите номер считываемого аргумента", _
    "Чтение номера аргумента") - 1
t = ""
If q > Coun Then
    t = objArgs(q)
Else
    t = "ERROR"
End if
MsgBox t
```

## Свойства *FullName*, *Name*, *Path*, *Version*

Набор свойств `FullName`, `Name`, `Path`, `Version` позволяет получить информацию о версии и месторасположении сервера сценариев (листинг 15.5, рис. 15.3).

**Листинг 15.5. Определение свойств интерпретатора WSH**

```
Product = WScript.Name & " " & WScript.Version & chr(13)
Path = WScript.Path & chr(13) & WScript.FullName
MsgBox Product & Path
```

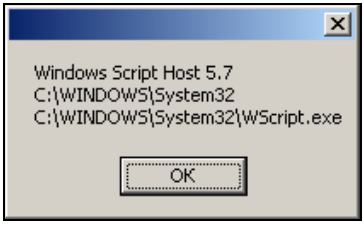



**Рис. 15.3.** Свойства интерпретатора WSH

## Свойства *StdErr*, *StdIn*, *StdOut*

Доступ к стандартным входным и выходным потокам с помощью свойств `StdErr`, `StdIn` и `StdOut` можно получить, если сценарий запущен в консоль-

ном режиме с помощью утилиты Cscript.exe. Если сценарий был запущен с помощью Wscript.exe, то при попытке обратиться к этим свойствам возникает ошибка `Invalid Handle`. Перечень ошибок, выдаваемых сценариями WSH, приведен в конце этой главы (см. табл. 15.20). При использовании сценариев регистрации пользователей в сети и ASP-файлов осуществляется работа с Wscript.exe, поэтому примеры работы с этими свойствами не приводятся.

## Методы объекта *WScript*

Объект `WScript` поддерживает несколько методов, которые описаны в табл. 15.4.

***Таблица 15.4.*** *Методы объекта* `WScript`

| Свойство | Описание |
|---|---|
| `CreateObject(ObjectID, Prefix)` | Создает объект, заданный параметром `ObjectID` |
| `GetObject(Object, StrProgID, Prefix)` | Активизирует объект, определяемый заданным файлом или параметром |
| `ConnectObject(ObjectID, Prefix)` | Устанавливает соединение с объектом, позволяющее писать функции-обработчики его событий. Имена этих функций должны начинаться с префикса `Prefix` |
| `DisconnectObject(ObjectID)` | Отсоединяет объект, с которым ранее была установлена связь |
| `Echo […[[Arg1], Arg2]…]` | Выводит текстовую информацию на консоль или в диалоговое окно. Является упрощенным аналогом функции `MsgBox()` в VBScript |
| `Quit(ErrorCode)` | Прерывает выполнение сценария с заданным параметром `ErrorCode` кодом выхода. Если параметр не задан, то Wscript.exe установит код выхода равным нулю |
| `Sleep(Time)` | Приостанавливает выполнение сценария на заданный промежуток времени. Время указывается в миллисекундах |

Приведем дополнительные пояснения и примеры наиболее часто используемых методов.

## Метод *CreateObject()*

Если объект не предоставляется WSH автоматически, как в случае `WScript`, то его необходимо создать с помощью метода `CreateObject()`, то есть загрузить его экземпляр в память и присвоить переменной-объекту ссылку. Используя этот метод, не забывайте об иерархии объектов. Первый параметр содержит ID объекта, например, `WScript.Shell`; второй параметр является необязательным и содержит строку префикса. Если префикс задан, то после создания этого объекта метод соединяет его исходящий интерфейс со сценарием. Такое соединение позволяет осуществлять обработку событий функциями обратного вызова, объявленными в сценарии. Например, если префиксом является `"ErrorCode"`, а у объекта срабатывает событие `OnBegin`, то WSH вызывает функцию `ErrorCode_OnBegin()`. Функция должна быть описана в теле файла. На практике префиксами пользуются крайне редко. Создание объекта:

```
Set objShell= WScript.CreateObject("WScript.Shell")
```

## Метод *GetObject()*

Наряду с `CreateObject()`, доступ к объектам также дает метод `GetObject()`, который получает объект из файла или из другого объекта, заданного параметром `ObjectID`. Этот метод используется в том случае, если экземпляр объекта уже имеется или его необходимо вызвать из файла. Первый параметр задает путь к файлу, содержащему объект или к объекту, а второй и третий параметры аналогичны по своей сути параметрам, используемым в методе `CreateObject()`. Второй параметр позволяет загружать объект с помощью записей реестра, а третий — связывать сценарий и внешний интерфейс. Пример вызова объекта:

```
Set rootDSE = WScript.GetObject("LDAP://RootDSE")
```

## Метод *ConnectObject()*

Метод `ConnectObject()` позволяет программно подключать обработку событий приложений. Для отключения данного обработчика событий служит метод `DisconnectObject()` (см. *разд. "Метод DisconnectObject()"*). Для вызова метода используется следующий синтаксис:

```
obj.ConnectObject strObject, strPrefix
```

В нем используются следующие аргументы:

- `strObject` — имя подключаемого объекта;
- `strPrefix` — префикс события функции.

В листинге 15.6 приведен пример обработки события закрытия приложения Microsoft Word. На первом этапе с помощью `CreateObject()` создается экземпляр объекта. Затем подключается обработчик событий запущенного приложения. И, наконец, создается сам обработчик. При закрытии приложения сценарием будет выведено соответствующее сообщение. В завершение сценария осуществляется отключение созданного обработчика ошибок с помощью метода `DisconnectObject()`.

**Листинг 15.6. Использование обработчика событий для созданного объекта**

```
Set obj = WScript.CreateObject("Word.Application")
obj.Visible = True      ' Отображение на экране Word
WScript.ConnectObject obj, "MyWord_"
Dim blnWordVisible      ' Объявление переменной blnWordVisible
                        ' для обработки события
blnWordVisible = True   ' По умолчанию считаем Word видимым
While blnWordVisible
    WScript.Sleep 500   ' Выполняем какие-то действия, в данном случае
                        ' просто ничего не делаем, и раз в полсекунды
                        ' проверяем значение blnWordVisible
Wend
WScript.DisconnectObject obj

Sub MyWord_Quit         ' Обработка события "закрытие приложения Word"
    blnWordVisible = False
    MsgBox "Программа Word закрыта"
End Sub
```

## Метод *DisconnectObject()*

При создании объекта методом `CreateObject()` или `GetObject()` он загружается в память и остается соединенным с WSH до конца сеанса. Сценарий может подключить свой обработчик событий, используя второй параметр метода `CreateObject()` или третий параметр `GetObject()`, или же вызвав метод `ConnectObject()`.

Объект может генерировать события, которые будут обслужены обработчиком событий сценария. Метод `DisconnectObject()` отсоединяет объект от процедуры обработки события во время работы сценария (листинг 15.6):

```
WScript.DisconnetObject(ObjectID)
```

или

```
WScript.DisconnetObject ObjectID
```

В данном случае переменная `ObjectID` — имя переменной объекта. При этом сам объект остается в памяти, и даже может быть в дальнейшем обратно подключен к обработке событий. Чтобы освободить занимаемую объектом память, надо удалить все ссылки на него. На VBScript удалить ссылку можно следующим образом (листинг 15.7):

```
ObjectID = Nothing
```

При этом надо иметь в виду, что, возможно, на объект есть и другие ссылки. Тогда выгружен из памяти он не будет.

**Листинг 15.7. Сброс значения переменной-объекта**

```
Set Wshell = WScript.CreateObject("WScript.Shell")
…
Wshell = Nothing
```

# Объект *WshArguments*

Этот объект обеспечивает доступ к параметрам командной строки запущенного сценария или ярлыка Microsoft Windows. Доступ к объекту `WshArguments` осуществляется с помощью свойства `Arguments` объекта `WScript`, которое было описано ранее.

# Объект *WshShell*

Этот объект позволяет запускать процессы, создавать ярлыки, работать с системным реестром и специальными папками Microsoft Windows. Для получения этого объекта надо вызвать метод `CreateObject()` объекта `WScript` с аргументом WScript.Shell:

```
Set Wshell = WScript.CreateObject("WScript.Shell")
```

Свойства и методы, поддерживаемые объектом `WshShell`, представлены в табл. 15.5 и 15.6.

***Таблица 15.5.*** *Свойства объекта* `WshShell`

| Свойство | Описание |
|---|---|
| `Environment` | Обеспечивает доступ к объекту `WshEnvironment`, содержащий переменные среды для Windows 9*x* и 2k |
| `SpecialFolders` | Обеспечивает доступ к объекту `SpecialFolders`, содержащий набор путей к специальным папкам Windows 9*x* и 2k |

***Таблица 15.6.*** *Методы объекта* `WshShell`

| Метод | Описание |
|---|---|
| `AppActivate(title)` | Активизирует заданное параметром `title` окно ранее запущенного приложения. Строка `title` задает название окна |
| `CreateShortcut (strPathName)` | Создает объект `WshShortcut` для связи с ярлыком Windows (*.lnk) или объект `WshURLShortcut` для связи с сетевым ярлыком (*.url). Параметр `strPathName` содержит полный путь к создаваемому или изменяемому ярлыку |
| `ExpandEnvironmentStrings (strString)` | Возвращает значение переменной среды текущего командного окна, заданной строкой `strString`. Имя переменной должно быть окружено знаками `%` |
| `LogEvent (intType, strMessage [,strTarget])` | Протоколирует произошедшие события в журнал событий Windows или в файл Wsh.log. Первый параметр определяет тип сообщения, второй — его содержание. Третий параметр (`strTarget`) может быть задан только в Windows 2k. С его помощью определяют название системы, в которой протоколируются события. По умолчанию это локальная система. Метод возвращает `True`, если событие успешно занесено в журнал событий, и `False` в противном случае |

***Таблица 15.6*** *(окончание)*

| Метод | Описание |
|---|---|
| `Popup(strText [,secWait] [,strTitle] [,nType])` | Выводит окно с заданным текстом `strText`. Аналог функции `MsgBox()` в VBScript. Параметр `secWait` определяет количество секунд, по истечении которых окно будет закрыто; `strTitle` задает заголовок окна; `nType` конфигурирует внешний вид окна (значок и кнопки) |
| `RegDelete(strName)` | Выполняют операции с реестром: удаление, чтение и запись данных соответственно (см. *разд. "Управление реестром с помощью WSH"* в *главе 14*) |
| `RegRead(strName)` | |
| `RegWrite(strName, Value [,strType])` | |
| `Run(strCommand [,WindowStyle], [,WaitOnReturn])` | Запускает стороннее приложение из сценария |
| `SendKeys(string)` | Эмулирует нажатие клавиш клавиатуры |

## Метод *AppActivate()*

С помощью метода `AppActivate()`, устанавливая фокус, активизируют ранее запущенное приложение; при этом никаких изменений размера окна не осуществляется (листинг 15.8). Для первоначального запуска приложения и определения размеров окна используйте метод `Run()`, который будет описан позднее. Для того чтобы определить, какое именно приложение необходимо активизировать, переданный параметр `title` поочередно сравнивается с названиями окон всех запущенных приложений. Если точного совпадения не найдено, то активизируется приложение, начало заголовка окна которого совпадает с `title`. Если такое окно с таким заголовком не найдено, то будет сделано активным приложение, у которого заголовок и строка `title` совпадают с конца строки.

**Листинг 15.8. Использование метода `AppActivate()`**

```
Set Wshell = WScript.CreateObject("WScript.Shell")
Wshell.Run("calc")
Wshell.AppActivate("Calculator")
```

## Управление ярлыками методом *CreateShortcut()*

Существуют два вида ярлыков: одни являются ссылками на файлы или папки, а другие — на интернет-страницы. Для создания ярлыков программ используют метод `CreateShortcut()`, который возвращает объект `WshShortcut` для ссылок на файлы и объект `WshURLShortcut` для ссылок на сайты. Тип возвращаемого объекта определяется расширением создаваемого ярлыка, которое должно быть равно LNK или URL соответственно. Более подробно работа с ярлыками описана далее в этой главе в *разд. "Объект WshShortcut"* и *"Объект WshURLShortcut"*.

## Свойство *Environment*

Свойство `Environment` предназначено для работы с переменными окружения. Это свойство возвращает объект `WshEnvironment`, который представляет собой коллекцию строковых значений. Чтобы получить доступ к этой коллекции, необходимо подключиться к пространству объекта-родителя `WScript.Shell`:

```
Set Wshell = CreateObject("WScript.Shell")
Set variables = Wshell.Environment([StrType])
```

Индекс `StrType` задает категорию переменной окружения. В Windows 2k переменные окружения разделены на несколько категорий — `System`, `User`, `Volatile`, `Process` (табл. 15.7).

***Таблица 15.7.*** *Переменные окружения и их категории*

| Имя | Описание | Категории | | | |
|---|---|---|---|---|---|
| | | System | Volatile | User | Process |
| COMSPEC | Путь к командному процессору (cmd.exe или command.com) | + | | | + |
| HOMEDIVE | Главный локальный диск. Как правило, C:\ | | | | + |
| HOMEPATH | Каталог по умолчанию для пользователей | | | | + |

***Таблица 15.7*** *(окончание)*

| Имя | Описание | Категории | | | |
|---|---|---|---|---|---|
| | | System | Volatile | User | Process |
| NUMBER_OF_PROCESSORS | Количество процессоров, установленных в материнской плате | + | | | + |
| OS | Название и версия ОС | + | | | + |
| PATH | Путь | + | | | + |
| PATHEXT | Расширения исполняемых файлов | + | | | + |
| PROCESSOR_ARCHITECTURE | Тип процессора | + | | | + |
| PROCESSOR_IDENTIFIER | Идентификатор процессора | + | | | + |
| PROCESSOR_LEVEL | Уровень процессора | + | | | + |
| PROCESSOR_REVISION | Версия процессора | + | | | + |
| PROMPT | Приглашение командной строки MS-DOS | | | | + |
| SYSTEMDRIVE | Логический диск, на котором установлена ОС | | | | + |
| SYSTEMROOT | Системный каталог, в котором расположена ОС | | | | + |
| TEMP | Каталог для хранения временных файлов | | + | + | + |
| TMP | Каталог для хранения временных файлов | | + | + | + |
| WINDIR | То же, что и SYSTEMROOT | | + | | + |

**Листинг 15.9. Чтение значения одной из переменных окружения**

```
Set Wshell = CreateObject("WScript.Shell")
Set variables = Wshell.Environment("Process")
WScript.Echo variables("WINDIR")
```




**Рис. 15.4.** Чтение переменной окружения

Опишем действия, выполняемые в сценарии (листинг 15.9), построчно. В первой строке создается ссылка на объект `WshShell`. Во второй выполняется обращение к свойству `Environment`, содержащему набор всех переменных окружения из категории `Process`. В последней строке осуществляется вывод значения переменной `WINDIR` из категории `Process` (рис. 15.4).

Свойства объекта `WshEnvironment` приведены в табл. 15.8.

***Таблица 15.8.*** *Свойства объекта* `Environment`

| Свойство | Описание |
|---|---|
| `Count` | Возвращает число переменных окружения |
| `Item` | Свойство, по умолчанию возвращающее переменную окружения, заданную по имени. В отличие от других применений одноименного свойства, задавать переменную окружения по номеру в данном случае нельзя |
| `Length` | Возвращает число переменных окружения. Используется для совместимости с JScript |

Чтобы считать все существующие на данной рабочей станции переменные окружения, можно запустить сценарий, приведенный в листинге 15.10. Отличие от предыдущего примера заключается в том, что, получив доступ ко всей коллекции, используя цикл, определяются имена и значения всех переменных среды окружения.

**Листинг 15.10. Чтение всех переменных окружения и их значений**

```
Set Wshell = CreateObject("WScript.Shell")
Set variables = Wshell.Environment("Process")
temp=""
For Each var In variables
    temp = temp + var + vbNewLine
Next
WScript.Echo temp
```

## Операции с переменными окружения средствами командной строки

С помощью сценария, написанного на WSH, можно управлять переменными окружения: удалять их и создавать.

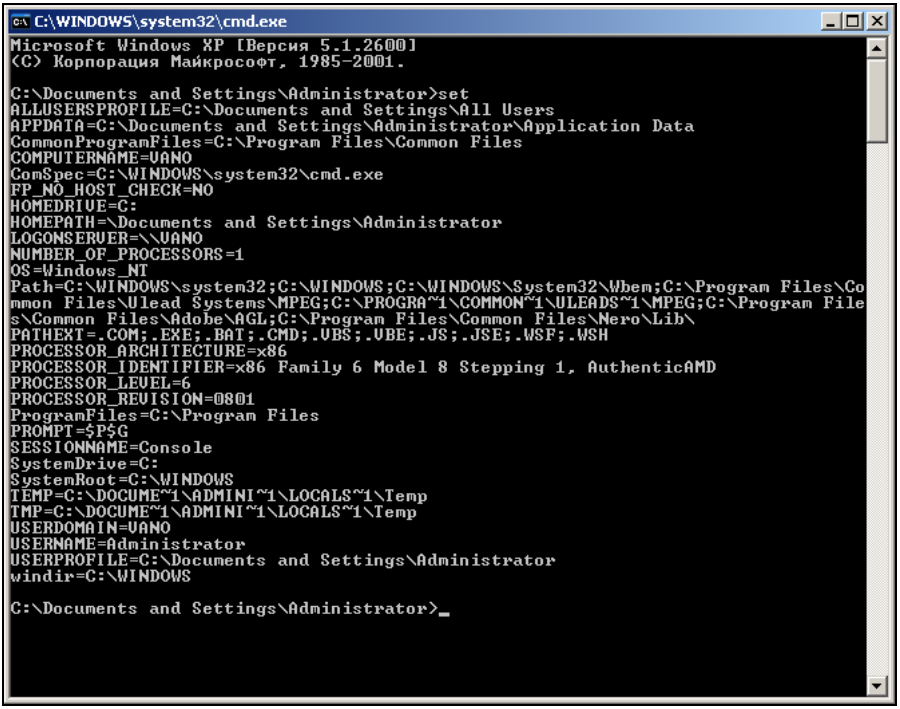



**Рис. 15.5.** Значения переменных окружения, полученных командой `Set`

В MS-DOS и командной строке Windows управление переменными осуществляется с помощью команды `set`, которая имеет следующий синтаксис:

```
set [Variable[=Value]]
```

В этой команде используются следующие параметры:

- `Variable` — имя переменной;
- `Value` — значение, сопоставляемое переменной среды.

При вызове команды `set` без параметров осуществляется вывод уже известных переменных и их значений (рис. 15.5).

## Создание переменных окружения с помощью сценария

Переменная окружения создается автоматически при присвоении ей значения в заранее оговоренной коллекции (листинг 15.11).

**Листинг 15.11. Создание переменной окружения**

```
Set Wshell = CreateObject("WScript.Shell")
Set variables = Wshell.Environment("Process")
variables("NewSet") = "This Is New Set Variable"
WScript.Echo variables("NewSet")
```

Если вызвать сценарий с помощью метода `Run()` из другого сценария, то переменная, объявленная в первом, будет доступна во втором сценарии.

## Удаление переменных окружения с помощью сценария

Удаление переменной окружения осуществляется с помощью метода `Remove()`. Этому методу доступны только локальные переменные окружения (листинг 15.12).

**Листинг 15.12. Удаление переменной окружения**

```
Set Wshell = CreateObject("WScript.Shell")
Wshell.Environment("Process").Remove("NewSet")
```

## Метод *ExpandEnvironmentStrings()*

Этот метод возвращает строку, в которой имена переменных среды текущего командного окна заменены соответствующими им значениями (листинг 15.13). Имена переменных среды должны быть окружены знаками `%`.

**Листинг 15.13. Получение значения указанной переменной окружения**

```
Set Wshell=CreateObject("WScript.Shell")
WScript.Echo Wshell.ExpandEnvironmentStrings("Каталог Windows: %Windir%")
```

Если операционная система установлена в папке Windows на диске C:, то скрипт выдаст диалоговое окно с сообщением "Каталог Windows: C:\Windows".

## Метод *LogEvent()*

Этот метод протоколирует произошедшие события. В Windows 2k события записываются в системный журнал. Каждая запись в файле отчетов содержит время события, его тип (табл. 15.9) и описание события.

***Таблица 15.9.*** *Расшифровка типов событий*

| Код | Значение | Описание |
|---|---|---|
| 0 | SUCCESS | Успех |
| 1 | ERROR | Ошибка |
| 2 | WARNING | Предупреждение |
| 4 | INFORMATION | Информационное сообщение |
| 8 | AUDIT_SUCCESS | Успех при аудите |
| 16 | AUDIT_FAILURE | Ошибка при аудите |

Метод `LogEvent()` имеет следующий синтаксис:

```
Wshell.LogEvent(intType, strMessage [,strTarget])
```

Первый параметр, `intType`, определяет тип сообщения, второй, `strMessage`, — его содержание. Необязательный параметр `strTarget` может быть задан только в Windows 2k. Им определяется название системы, в кото-

рой протоколируются события. По умолчанию это локальная система. Метод возвращает `True`, если событие успешно занесено в журнал, и `False` в противном случае.

В приведенном в листинге 15.14 примере выполняется функция `Function_1()`, которая возвращает 0 (`TRUE`) или 1 (`FALSE`). Если функция отработала успешно (`E_Code=0`), то в журнале событий появится запись `Status: OK`, в противном случае — `Status: ERROR`.

**Листинг 15.14. Протоколирование работы функции**

```
Set WshShell = CreateObject("WScript.Shell")
E_Code = Function_1()
If E_Code = 0 Then
    WshShell.LogEvent(0,"Status: OK","Event")
Else
    WshShell.LogEvent(1, "Status: ERROR", "Event")
End If
```

## Метод *Popup()*

Метод `Popup()` используется для вывода сообщений на экран. По своей сути этот метод является аналогом функции `MsgBox()` в VBScript. Вызов метода осуществляется посредством одноименной функции. Приведем ее синтаксис:

```
Wshell.Popup(strText [,secWait] [,strTitle] [,nType])
```

Переменная `strText` содержит текст, выводимый в диалоговом окне. Параметр `secWait` определяет количество секунд, по истечению которых окно будет закрыто; `strTitle` — заголовок окна; `nType` (табл. 15.10 и 15.11) — конфигурацию окна (значок и кнопки). Значения параметра `nType` полностью совпадают со значениями второго параметра функции `MsgBox()` в VBScript.

***Таблица 15.10.*** *Константы, определяющие вид иконки окна*

| Значение | Константа VBScript | Описание |
|---|---|---|
| 0 | – | Нет значка (по умолчанию) |
| 16 | vbCritical | Знак "Стоп" |

***Таблица 15.10*** *(окончание)*

| Значение | Константа VBScript | Описание |
|---|---|---|
| 32 | vbQuestion | Вопросительный знак |
| 48 | vbExclamation | Восклицательный знак |
| 64 | vbInformation | Знак "Информация" |

***Таблица 15.11.*** *Константы, определяющие набор кнопок окна*

| Значение | Константа VBScript | Описание |
|---|---|---|
| 0 | vbOKOnly | Кнопка **OK** (по умолчанию) |
| 1 | vbOKCancel | Кнопки **OK** и **Cancel** |
| 2 | vbOKRetryIgnore | Кнопки **OK**, **Retry**, **Ignore** |
| 3 | vbYesNoCancel | Кнопки **Yes**, **No**, **Cancel** |
| 4 | vbYesNo | Кнопки **Yes**, **No** |
| 5 | vbRetryCancel | Кнопки **Retry**, **Cancel** |

В методе `Popup()` можно комбинировать (то есть складывать) значения параметров, приведенные в табл. 15.10 и 15.11.

## Определение значения, возвращаемого методом *Popup()*

Значение, возвращаемое методом `Popup()`, зависит от кнопки, нажатой пользователем. Таким образом, анализируя возвращаемое значение, можно определить, какая из кнопок была нажата (табл. 15.12).

***Таблица 15.12.*** *Значения, возвращаемые методом* `Popup`

| Константа | Нажатая кнопка |
|---|---|
| -1 | Пользователь не нажал ни одну из кнопок в течение времени, заданного параметром `secWait` |

***Таблица 15.12*** *(оокнчание)*

| Константа | Нажатая кнопка |
|---|---|
| 1 | **OK** |
| 2 | **Cancel** |
| 3 | **Abort** |
| 4 | **Retry** |
| 5 | **Ignore** |
| 6 | **Yes** |
| 7 | **No** |

**Листинг 15.15. Использование метода `Popup()`**

```
Set WshShell = CreateObject("WScript.Shell")
result = WshShell.Popup("Операция завершена", 5, "Сообщение", 65)
WScript.Echo result
```

Сообщение, выводимое на экран (листинг 15.15, рис. 15.6), автоматически закроется через 5 секунд.

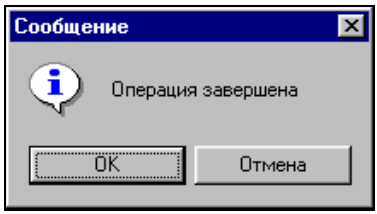



**Рис. 15.6.** Использование метода `Popup()`

## Метод *Run()*

С помощью метода `Run()` осуществляется запуск различных приложений из сценария (листинг 15.19).

Команда имеет следующий синтаксис:

```
Wshell.Run(strCommand [,WindowStyle], [,WaitOnReturn])
```

Передаваемые параметры имеют следующий смысл:

- `strCommand` — обязательный параметр, значение которого содержит путь к файлу запускаемого приложения. Путь к файлу, содержащий пробелы, необходимо заключать в кавычки. В противном случае метод выдаст ошибку "The system cannot find the file specified" (Система не может найти указанный файл). Метод `Run()` автоматически раскрывает переменные окружения;
- `WindowStyle` контролирует размер окна, в котором выполняется приложение (табл. 15.13). Этот параметр необязательный;

***Таблица 15.13.*** *Значения параметра* `WindowStyle`

| Значение | Константа VBScript | Описание |
|---|---|---|
| 0 | vbHide | Скрывает окно и активизирует другое (показывает его и передает ему фокус) |
| 1 | vbNormalFocus | Активизирует и показывает окно. Если процесс был уже запущен, то восстанавливается его предыдущий размер |
| 2 | vbMinimizedFocus | Активизирует и сворачивает окно. Фокус переходит к кнопке на панели задач |
| 3 | vbMaximizedFocus | Активизирует окно, разворачивает его на весь экран и передает ему фокус |
| 4 | vbNormalNoFocus | Вызывает окно в его предыдущем положении и размере |
| 5 | – | Активизирует окно уже запущенного процесса |
| 6 | vbMinimizedNoFocus | Сворачивает окно и активизирует следующее окно |
| 7 | – | Выводит окно в виде значка. Фокус окна сохраняется |
| 8 | – | Выводит окно в текущем состоянии. Активное окно остается таковым |

***Таблица 15.13*** *(окончание)*

| Значение | Константа VBScript | Описание |
|---|---|---|
| 9 | – | Активизирует окно и выводит его. Восстанавливает его первоначальный размер и положение |
| 10 | – | Устанавливает состояние окна в зависимости от состояния программы, запустившей это приложение |

- `WaitOnReturn` — необязательный параметр, принимающий значение `True` или `False`. Этот параметр определяет, будет ли сценарий ожидать завершения существующего процесса. Если он не задан или его значение равно `False`, то метод `Run()` возвращает значение 0. Код ошибки задается методом `Quit()`.

Метод `Run()` имеет существенный недостаток — им всегда порождается новый процесс, который необходимо каждый раз останавливать (это касается утилит, запускаемых из командной строки). Пример его использования приведен в листинге 15.16.

**Листинг 15.16. Использование метода `Run()`**

```
Set Wshell = CreateObject("WScript.Shell")
intReturn = Wshell.Run ("cmd.exe dir *.* > c:\test.txt")
Response.Write(intReturn) ' Код выше возвращает 0
Set Wshell = Nothing
```

## Метод *SendKeys()*

Метод `SendKeys` используется для эмуляции нажатия различных клавиш и управления запущенными приложениями. Каждая клавиша задается одним или несколькими символами. Некоторые символы, а именно `+^%~,[]{}`, являются служебными, и их необходимо заключать в фигурные скобки — `{}`.

Для задания неотображаемых клавиш, таких как <Enter> или <Tab>, используются специальные коды, перечисленные в табл. 15.14.

***Таблица 15.14.*** *Коды задания неотображаемых символов*

| Клавиша | Код | Клавиша | Код |
|---|---|---|---|
| <Alt> | % | <↑> | {Up} |
| <BackSpace> | {BackSpace}, {BS}, {BKSP} | <←> | {Left} |
| <Break> | {Break} | <↓> | {Down} |
| <Caps Lock> | {CapsLock} | <→> | {Right} |
| <Ctrl> | ^ | <F1> | {F1} |
| <Del>, <Delete> | {Del}, {Delete} | <F2> | {F2} |
| <End> | {End} | <F3> | {F3} |
| <Enter> | {Enter}, ~ | <F4> | {F4} |
| <Esc> | {Esc} | <F5> | {F5} |
| <Help> | {Help} | <F6> | {F6} |
| <Home> | {Home} | <F7> | {F7} |
| <Ins>, <Insert> | {Ins}, {Insert} | <F8> | {F8} |
| <Num Lock> | {NumLock} | <F9> | {F9} |
| <Page Down> | {PGDN} | <F10> | {F10} |
| <Page Up> | {PGUP} | <F11> | {F11} |
| <Print Screen> | {PRTSC} | <F12> | {F12} |
| <Scroll Lock> | {ScrollLock} | | |
| <Shift> | + | | |
| <Tab> | {Tab} | | |

Чтобы передать приложению комбинацию клавиш, включающую <Ctrl>, <Alt> или <Shift>, необходимо добавить соответствующий код перед клавишей, например, комбинации <Ctrl>+<C> соответствует "^C". Если комбинация состоит из двух и более клавиш, то символы следует заключать в круглые скобки, например, комбинации клавиш <Shift>+<E>+<C> соответствует +(EC). Для повтора эмуляции нажатия одной и той же клавиши необходимо

после клавиши указать количество раз повтора, например, если нужно сэмулировать нажатие кнопки <Tab> 10 раз, то аргумент метода `SendKeys()` должен выглядеть следующим образом: `{Tab 10}`.

Команда вызова метода `SendKeys()` имеет следующий синтаксис:

```
Wshell.SendKeys(string)
```

Пример использования метода `SendKeys` приведен в листинге 15.17.

**Листинг 15.17. Закрытие активного окна (эмуляция нажатия комбинации клавиш <Alt>+<F4>)**

```
Set Wshell = CreateObject("WScript.Shell")
Wshell.SendKeys(%{F4})
Set Wshell = Nothing
```

## Свойство *SpecialFolders*

Свойство `SpecialFolders` используется для получения доступа к специальным папкам Windows. Объект `WShell` содержит коллекцию путей `SpecialFolders` ко всем специальным папкам. Для получения пути к конкретной папке можно использовать константы, приведенные в табл. 15.15.

***Таблица 15.15.*** *Список специальных папок*

| Название папки | Описание |
|---|---|
| `AllUsersDesktop` | Хранят данные для всех пользователей |
| `AllUsersStartMenu` | |
| `AllUsersPrograms` | |
| `AllUsersStartup` | |
| `Desktop` | Рабочий стол |
| `Favorites` | Избранное |
| `Fonts` | Шрифты |
| `MyDocuments` | Мои документы |
| `NetHood` | Сетевое окружение |
| `Programs` | Программы |

***Таблица 15.15*** *(окончание)*

| Название папки | Описание |
|---|---|
| `Recent` | Корзина |
| `SendTo` | Отправить |
| `StartMenu` | Меню "Пуск" |
| `Startup` | Автозагрузка |
| `Templates` | Шаблоны |

Получение списка всех доступных специальных папок и соответствующих им путей показано в листинге 15.18 и на рис. 15.7. Члены набора `SpecialFolders` являются элементами массива.

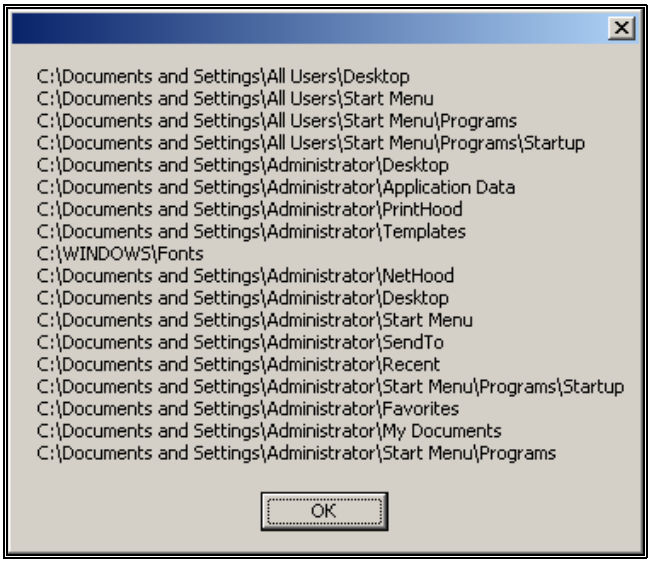



**Рис. 15.7.** Список доступных на рабочей станции специальных папок

**Листинг 15.18. Получение списка доступных специальных папок**

```
Set Wshell=CreateObject("WScript.Shell")
For Each variable In Wshell.SpecialFolders
```

```
    temp = temp+ variable +chr(13)
Next
Msgbox temp
```

Объект `Wshell.SpecialFolders` имеет два аналогичных свойства — `Count` (для VBScript) и `Length` (для JScript):

```
Set Wshell=CreateObject("WScript.Shell")
WScript.Echo "Всего спецпапок: " & Wshell.SpecialFolders.Count
```

Если необходимо получить путь к определенной специальной папке, то в запросе указывают ее конкретное название (см. табл. 15.15), как это сделано в листинге 15.19.

**Листинг 15.19. Получение пути к определенной специальной папке**

```
Set Wshell=CreateObject("WScript.Shell")
WScript.Echo "Путь к папке с рабочим столом:" & _
    Wshell.SpecialFolders(Desktop)
```

# Объект *WshShortcut*

Данный объект можно создать с помощью метода `CreateShortcut()` объекта `WshShell`. Свойства объекта `WshShortcut` приведены в табл. 15.16.

***Таблица 15.16.*** *Свойства объекта* `WshShortcut`

| Свойство | Описание |
|---|---|
| `Arguments` | Содержит строку, задающую параметры командной строки для ярлыка |
| `Description` | Задает описание ярлыка |
| `FullName` | Содержит полный путь к ярлыку |
| `HotKey` | Задает комбинацию горячих клавиш для ярлыка. Чтобы назначить комбинацию клавиш ярлыку, следует названия необходимых клавиш разделить символом "+", например `CTRL+ALT+W` |
| `IconLocation` | Указывает путь к иконке ярлыка |

***Таблица 15.16*** *(окончание)*

| Свойство | Описание |
|---|---|
| `TargetPath` | Устанавливает путь к файлу, на который указывает ярлык |
| `WindowStyle` | Определяет вид окна, в котором будет выводиться приложение. Значением этого свойства является целое число (табл. 15.17). |
| `WorkingDirectory` | Задает рабочий каталог приложения |

Для записи указанных свойств на диск после их установки в скрипте используется метод `Save()`. Приемы работы с этими свойствами мы рассмотрим на примерах, решающих часто возникающие задачи.

***Таблица 15.17.*** *Значения свойства* `WindowStyle`

| Значение | Описание |
|---|---|
| 1 | Стандартный размер окна, принятый по умолчанию. Окно располагается в центре экрана |
| 3 | Окно при запуске приложения будет развернуто на весь экран |
| 7 | Окно при запуске приложения будет свернуто |

## Создание ярлыка

Приведем пример, в котором используются все описанные свойства (листинг 15.20, рис. 15.8). Создадим на рабочем столе ярлык к стандартному приложению Windows Notepad.exe (Блокнот), которое расположено в каталоге %@WinDir%. Зададим клавиатурное сокращение <Ctrl>+<Alt>+<B>. После запуска приложения оно должно разворачиваться на весь экран.

**Листинг 15.20. Создание ярлыка Windows-приложения**

```
Set WshShell = CreateObject("WScript.Shell")
StrDesktop = WshShell.SpecialFolders("Desktop") ' определение пути
                                                ' к папке Desktop
```

```
Set oShk = WshShell.CreateShortcut(StrDesktop+"\\notepad.lnk")
oShk.Arguments = "1.txt"
oShk.Description = "Блокнот"
oShk.TargetPath = "%Windir%\\notepad.exe"
oShk.HotKey = "CTRL+ALT+B"
oShk.IconLocation = "notepad.exe, 0"
oShk.WindowStyle = 3
oShk.WorkingDirectory = "%Windir%"
oShk.Save()
WScript.Echo "Ярлык создан"
```

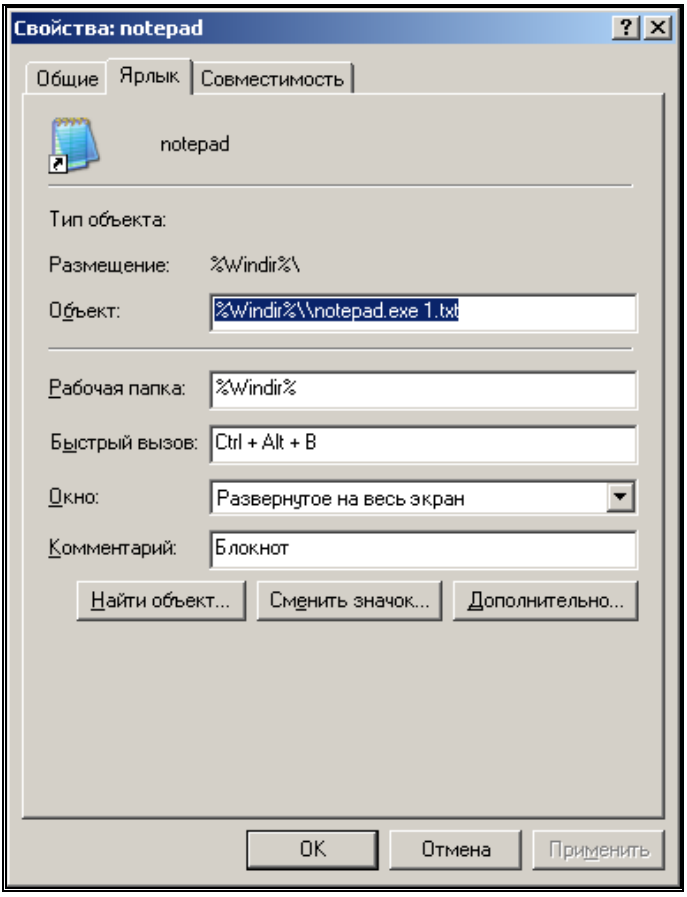



**Рис. 15.8.** Свойства ярлыка объекта

## Чтение и изменение свойств ярлыка

Метод `CreateShortcut()` позволяет не только создавать ярлыки, но и считывать и изменять их свойства. Сценарий работает по следующему алгоритму: сначала получают доступ к файлам с расширением LNK с помощью объекта `Scripting.FileSystemObject` и определяют тип файла, используя свойство `Type`. Если его тип равен Shortcut, то к свойствам ярлыка получают доступ с помощью инструкции `WshShell.CreateShortcut(oFile.path)`, где `oFile.path` — путь к ярлыку. Затем осуществляют чтение и запись данных (листинг 15.21).

**Листинг 15.21. Изменение свойств ярлыка Windows-приложения**

```
Set WshShell = CreateObject("WScript.Shell")
StrDesktop = WshShell.SpecialFolders("Desktop")
Path = StrDesktop + "\notepad.lnk"
Set fso = CreateObject("Scripting.FileSystemObject")
Set oFile = fso.GetFile(path)
If StrComp(oFile.type,"Shortcut",0) = 0 Then
    Set oShk=WshShell.CreateShortcut(oFile.path)
    If StrComp(UCase(oShk.Description), UCase("Блокнот")) = 0 Then
        t1 = oShk.Description
        oShk.Description = "Текстовый редактор"
        oShk.Save()
        t2 = oShk.Description
    End If
End If

WScript.Echo "Описание ярлыка " + ucase(t1) + " изменено на " + ucase(t2)
```

## Удаление ярлыка

Для удаления ярлыка используют метод `Delete()` объекта `Script.FileSystemObject` (листинг 15.22).

**Листинг 15.22. Удаление ярлыка**

```
Set WshShell=CreateObject("WScript.Shell")
StrDesktop = WshShell.SpecialFolders("Desktop") ' определение пути
                                                ' к папке Desktop
Set Path = StrDesktop + "\\notepad.lnk"
Set fso = CreateObject("Script.FileSystemObject")
Set oFile = fso.GetFile(path)
oFile.Delete
```

# Объект *WshURLShortcut*

У объекта `WshURLShortcut` только два свойства: `FullName` и `TargetPath`, которые полностью аналогичны одноименным свойствам метода `WshShortcut`.

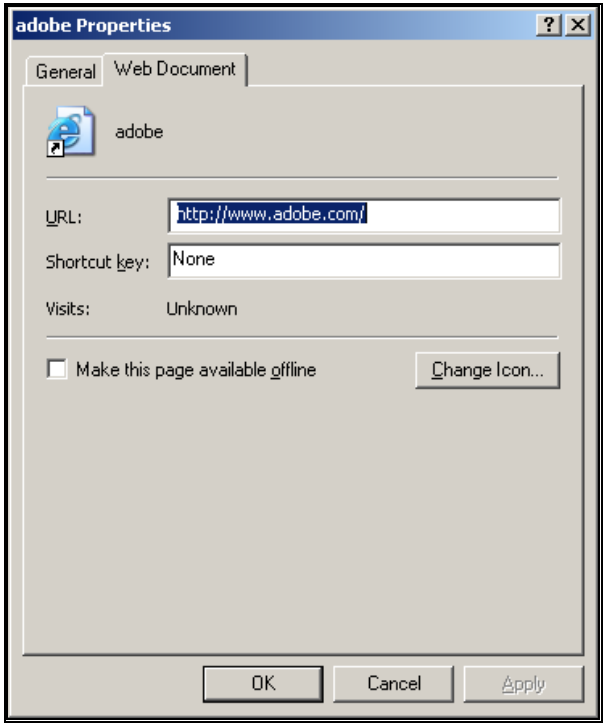



**Рис. 15.9.** Создание Web-ярлыка

Также поддерживается метод `Save()`, позволяющий сохранять ярлык в указанном каталоге (листинг 15.23, рис. 15.9).

**Листинг 15.23. Создание Web-ярлыка**

```
Set WshShell = CreateObject("WScript.Shell")
StrDesktop = WshShell.SpecialFolders("Desktop") ' определение пути
                                                ' к папке Desktop
Set oShk = WshShell.CreateShortcut(StrDesktop + "\\adobe.url")
oShk.TargetPath = "http://www.adobe.com"
oShk.Save()
WScript.Echo "Ярлык создан"
```

# Объект *WshNetwork*

Свойства и методы объекта `WshNetwork` (табл. 15.18 и 15.19) позволяют получить доступ к различным объектам сети, управлять сетевыми ресурсами (дисками и принтерами), считывать имя пользователя, название домена (листинг 15.24, рис. 15.10).

***Таблица 15.18.*** *Свойства, поддерживаемые объектом* `WshNetwork`

| Свойство | Описание |
|---|---|
| `UserName` | Получение имени текущего пользователя |
| `ComputerName` | Считывание имени рабочей станции |
| `UserDomain` | Определение имени домена, в котором находится рабочая станция |

**Листинг 15.24. Использование свойств объекта `WshNetwork`**

```
Set WshNetwork = WScript.CreateObject("WScript.Network")
Temp = "Рабочая станция: " + WshNetwork.ComputerName + chr(13)
Temp = Temp + "Пользователь: " + WshNetwork.UserName + chr(13)
Temp = Temp + "Домен: " + WshNetwork.UserDomain
WScript.Echo Temp
```

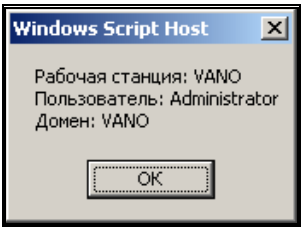



**Рис. 15.10.** Использование свойств объекта `WshNetwork`

***Таблица 15.19.*** *Методы, поддерживаемые объектом* `WshNetwork`

| Метод | Описание |
|---|---|
| `AddWindowsPrinterConnection()` | Подключение сетевого принтера для Windows |
| `AddPrinterConnection()` | Подключение сетевого принтера для Windows и DOS |
| `RemovePrinterConnection()` | Удаление сетевого принтера |
| `EnumPrinterConnections()` | Перечисление соединений с сетевыми принтерами |
| `SetDefaultPrinter()` | Установка принтера по умолчанию |
| `MapNetworkDrive()` | Подключение сетевого диска |
| `EnumNetworkDrives()` | Перечисление подключенных сетевых дисков |
| `RemoveNetworkDrive()` | Отключение сетевого диска |

## Методы *AddWindowsPrinterConnection()* и *AddPrinterConnection()*

Метод `AddWindowsPrinterConnection()` используется для подключения сетевых принтеров в среде Windows 2k.

```
Set WshNetwork=WScript.CreateObject("WScript.Network")
WshNetwork.AddWindowsPrinterConnection(strPath)
```

В качестве параметра `strPath` указывается путь к принтеру в формате UNC (\\Server\ShareName).

Если предполагается печать из программ, работающих в режиме эмуляции MS-DOS, нужно подключить локальный порт принтера к сетевому, использовав метод `AddPrinterConnection()` объекта `WshNetwork`:

```
Set WshNetwork = WScript.CreateObject("WScript.Network")
WshNetwork.AddPrinterConnection(strPort, strPath, _
                                [strConnect [, strUserName [, strPWD]]])
```

При этом нужно указать следующие параметры:

- `strPort` — локальный порт (например, `LPT1`);
- `strPath` — путь к принтеру в формате UNC (\\Server\ShareName);
- `strConnect` — необязательный параметр, характеризующий тип соединения: временный или постоянный; по умолчанию — `False` (временный);
- `strUserName` — необязательный параметр, который указывает имя пользователя, от имени которого монтируется принтер;
- `strPWD` — необязательный параметр, пароль пользователя.

Не рекомендуется использовать четвертый и пятый параметры, так как публикация паролей наносит серьезный вред системе безопасности сети.

## Обработка ошибок

Подключая сетевой принтер, рекомендуется использовать обработчик ошибок для обеспечения корректной работы скрипта и создания условий для решения сложившихся проблем в кратчайшие сроки (листинг 15.25).

**Листинг 15.25. Обработчик ошибок при подключении сетевых принтеров**

```
On Error Resume Next
Set WshNetwork = WScript.CreateObject("WScript.Network")
WshNetwork.AddWindowsPrinterConnection(Printer)
Select Case Err.Number
    Case 0
        Temp = "Сетевой принтер успешно подключен"
    Case -2147023688
        Temp = "Сетевой принтер невозможно подключить: ресурс " & _
                Printer &" не найден"
```

```
    Case -2147024811
        Temp = "Сетевой принтер уже подключен"
    Case Else
        Temp = "Ошибка " & Err.Number & ": " & Err.Descripion
End Select
MsgBox Temp
```

## Метод *RemovePrinterConnection()*

Этот метод используется для удаления подключения сетевого принтера (листинг 15.26).

```
Set WshNetwork = WScript.CreateObject("WScript.Network")
WshNetwork.RemovePrinterConnection(strName, [bForce], [bUpdateProfile])
```

При вызове передаются следующие параметры:

- `strName` — имя удаляемого принтера (`ShareName`) или порта (LPT1, LPT2), к которому подключен принтер;
- `bForce` — необязательный параметр; булево значение (`True` или `False`), определяющее, будет ли удалена связь, если принтер еще используется;
- `bUpdateProfile` — необязательный параметр, управляющий профилем пользователя: `True` — удаление записи о подключении принтера из профиля пользователя, `False` (по умолчанию) — нет.

**Листинг 15.26. Удаление сетевого принтера с порта LPT1**

```
Port_Name="LPT1"
On Error Resume Next
Set WshNetwork = WScript.CreateObject("WScript.Network")
WshNetwork.RemovePrinterConnection(Port_Name)
Select Case Err.Number
    Case 0
        Temp="Сетевой принтер успешно подключен"
    Case -2147023688
        Temp="Сетевой принтер невозможно подключить: ресурс " & _
             Printer &" не найден"
    Case -2147024811
        Temp="Сетевой принтер уже подключен"
```

```
    Case Else
        Temp="Ошибка " & Err.Number & ": "& Err.Descripion
End Select
MsgBox Temp
```

## Метод *EnumPrinterConnections()*

Метод `EnumPrinterConnections()` используется для определения подключенных к данной рабочей станции сетевых принтеров (листинг 15.27). Это свойство возвращает массив, членами которого являются элементы, содержащие информацию о подключенных сетевых принтерах:

```
Set WshNetwork = WScript.CreateObject("WScript.Network")
Set Property = WshNetwork.EnumPrinterConnections
```

После выполнения этого кода `Property` будет содержать массив с информацией о подключенных принтерах. Четные элементы данного массива содержат имена портов, к которым подключены принтеры, нечетные — имена принтеров в формате UNC (\\Server\ShareName). Нумерация элементов начинается с 0.

**Листинг 15.27. Определение всех принтеров, подключенных к рабочей станции**

```
Temp=""
Set WshNetwork = WScript.CreateObject("WScript.Network")
Set Property = WshNetwork.EnumPrinterConnections
Temp = ""
For i=0 to Property.Count-1 step 2
    Temp = Temp & Property(i) & " " & Property(i+1) & chr(13) & chr(10)
Next
MsgBox Temp
```

## Метод *SetDefaultPrinter()*

С помощью метода `SetDefaultPrinter()` на рабочей станции устанавливается принтер по умолчанию.

```
Set WshNetwork=WScript.CreateObject("WScript.Network")
WshNetwork.SetDefaultPrinter(strPath)
```

В параметре `strPath` должен быть указан путь к принтеру в формате UNC (\\Server\ShareName).

## Метод *MapNetworkDrive()*

С помощью метода `MapNetworkDrive()` осуществляется подключение сетевых дисков к рабочей станции (листинг 15.28). Вызов метода производится следующим образом:

```
WshNetwork=WScript.CreateObject("WScript.Network")
WshNetwork.MapNetworkDrive(strLetter,strPath)
```

Методу надо передать два параметра:

- `strLetter` — буква, на которую будет подключен сетевой ресурс;
- `strPath` — путь к сетевому ресурсу в формате UNC (\\Server\ShareName).

**Листинг 15.28. Предоставление доступа к сетевому ресурсу с использованием обработчика ошибок**

```
On Error Resume Next
WshNetwork = WScript.CreateObject("WScript.Network")
WshNetwork.MapNetworkDrive("Z:","\\Server\Multimedia")
ErrCheck Err.Number
…
Sub ErrCheck(Errors)
    Select Case Errors
        Case 0
            WScript.Echo "Подключение диска выполнено успешно"
        Case -2147024829
            WScript.Echo "Ошибка: сетевой ресурс не существует"
        Case -2147024811
            WScript.Echo "Ошибка: диск уже существует"
        Case Else
            WScript.Echo "Ошибка: "& Cstr(Errors)
    End Select
End Sub
```

## Метод *EnumNetworkDrives()*

Для получения списка подключенных дисков используется метод `EnumNetworkDrives()`, возвращающий массив с соответствующей информацией (листинг 15.29). Как и в методе `EnumPrinterConnections()`, четные элементы массива содержат локальное имя диска, а нечетные — соответствующие UNC-пути. Нумерация элементов начинается с нуля.

**Листинг 15.29. Использование метода `EnumNetworkDrives`**

```
Temp = ""
Set WshNetwork = WScript.CreateObject("WScript.Network")
Set Property = WshNetwork.EnumNetworkDrives
For i = 0 to Property.Count-1 Step 2
    If Property(i) <> "" Then
        Temp = Temp & Property(i) & " " & Property(i+1) & chr(13)+chr(10)
    End If
Next
MsgBox Temp
```

## Метод *RemoveNetworkDrive()*

Метод `RemoveNetworkDrive()` предназначен для отключения сетевого диска (листинг 15.30). Единственным его параметром является буква, которой сопоставлен отключаемый ресурс.

**Листинг 15.30. Отключение сетевого диска**

```
Set WshNetwork = WScript.CreateObject("WScript.Network")
WshNetwork.RemoveNetworkDrive("Z:")
```

# Ошибки выполнения сценариев в WSH

Некоторые ошибки, которые могут возникнуть при выполнении WSH-сценариев, и описание возможных причин их возникновения приведены в табл. 15.20.

***Таблица 15.20.*** *Ошибки выполнения сценария WSH*

| Сообщение об ошибке | Причина |
|---|---|
| A duplicate name for a named or unnamed element was encountered: xxx | Попытка повторного использования имени аргумента |
| Argument list too long | Эта ошибка чаще всего связана с запуском сценария с помощью технологии Drag-and-Drop: превышена максимальная длина командной строки |
| Can't save settings | Ошибка при сохранении файла с настройками сценария (*.wsh) |
| Environment variable <name> could not be removed | Вызов метода `Environment.Remove()` для несуществующей переменной среды |
| Invalid attempt to call Exec without a command | Вызов метода `WshShell.Exec()` без указания аргумента (команды для выполнения) |
| Invalid shortcut path name | Попытка создать ярлык с неправильным расширением файла (расширение должно быть LNK или URL) |
| Printer <name> not found | Неправильно указано имя принтера при вызове метода `SetDefaultPrinter()` |
| Protocol handler for <name> could not be found | Попытка установить ярлык на сетевой ресурс, использующий некорректно зарегистрированный или незарегистрированный обработчик протокола |
| Registry key <name> contains invalid root | Вызов метода `RegRead()` или `RegWrite()` для некорректного ключа реестра |
| Registry key <name> could not be opened | Вызов метода `RegRead()` для несуществующего ключа реестра |
| Registry key <name> could not be removed | Вызов метода `RegDelete()` для несуществующего ключа реестра |
| Remote script object can only be executed once | Попытка повторно запустить объект — удаленный сценарий |
| Shortcut <name> contains invalid syntax | Сохранение ярлыка на сетевой ресурс, имеющий некорректный URL |

***Таблица 15.20*** *(окончание)*

| **Сообщение об ошибке** | **Причина** |
|---|---|
| Shortcut <name> could not be saved | Попытка сохранить новый ярлык в файле, который уже существует и имеет атрибут "Только для чтения" |
| Shortcut <name> failed to execute protocol handler | Попытка установить ярлык на сетевой ресурс, использующий несуществующий обработчик протокола |
| Unable to execute remote script | Невозможно создать процесс — удаленный сценарий |
| Unable to find job <job identifier> | В WS-файле нет задания с идентификатором `<job identifier>` |
| Unable to wait for process | С помощью метода `Run()` дано указание ожидать завершение процесса, но из сценария нельзя определить, завершился ли он |

## Глава 16



# Загрузка операционной системы

В этой главе подробно рассмотрен процесс загрузки операционной системы Windows 2k и рассказано, как на него можно влиять.

## Основные термины и понятия

- **BSoD** (Blue Screen of Death, синий экран смерти) — сообщение о системной ошибке в операционных системах семейства Windows, после которой работа не может быть продолжена.
- **Ntldr** (NT Loader) — загрузчик операционной системы семейства Windows 2k, за исключением Windows Vista). Ntldr может быть запущен как с переносного устройства (CD, DVD, USB), так и с жесткого диска.
- **DEP** (Data Execution Prevention) — встроенное в семейство ОС Windows средство для предотвращения выполнения самомодифицирующегося кода, которое также следит за тем, чтобы программы не использовали защищенную область памяти. Если программа попытается запустить код из защищенной области, например, из памяти ядра, то она будет закрыта.
- **MBR** (Master Boot Record, главная загрузочная запись) — это первый физический сектор на жестком диске или другом устройстве хранения информации, которое разбито на логические диски (разделы). MBR содержит таблицу разделов (partition table) и небольшой фрагмент исполняемого кода.
- **HAL** (Hardware Abstraction Layer, слой аппаратных абстракций) — реализованный в операционной системе слой абстрагирования, находящийся между физическим уровнем аппаратного обеспечения и программным обеспечением, запускаемым на этом компьютере. Наличие слоя HAL по-

зволяет скрыть различия в аппаратном обеспечении компьютеров от основной части ядра операционной системы.

- **Save Mode** (Безопасный режим) — это специальный режим загрузки операционной системы, в котором загружается графическая оболочка, но отключена загрузка всех необязательных служб и драйверов, что максимально увеличивает вероятность успешной загрузки. Этот режим используют, если загрузку в обычном режиме завершить не удается.
- **ARC** (Advanced RISC Computing) **Path** — путь, указывающий на загрузочный раздел компьютера.

# Процесс запуска

Процесс загрузки операционной системы состоит из пяти этапов:

1. Предварительная загрузка (preboot sequence).
2. Загрузка (boot sequence).
3. Загрузка ядра (kernel load).
4. Инициализация ядра (kernel initialization).
5. Регистрация (logon).

## Предварительная загрузка

После включения компьютера инициализируется жесткий диск и осуществляется поиск загрузочного раздела жесткого диска. Предварительная загрузка проходит в четыре этапа:

1. BIOS определяет объем физической памяти и конфигурацию аппаратных устройств через PnP BIOS.
2. BIOS обнаруживает загрузочное устройство, загружает MBR (главную загрузочную запись) в память и запускает код, содержащийся в ней.
3. Исполняемый код MBR просматривает таблицу разделов в поиске активного раздела, загружает загрузочный сектор активного раздела в память и запускает его на выполнение.
4. Загружается и инициализируется загрузчик операционной системы Ntldr.

## Загрузка

После загрузки в память файла Ntldr собирается информация об аппаратном обеспечении и драйверах. При этом используются файлы Ntldr, Boot.ini и т. д. (табл. 16.1).

***Таблица 16.1.*** *Файлы, используемые в процессе загрузки ОС*

| Файл | Местоположение |
|---|---|
| Ntldr | %SystemDrive% |
| Boot.ini | %SystemDrive% |
| Bootsect.dos | %SystemDrive% |
| Ntdetect.com | %SystemDrive% |
| Ntbootdd.sys | %SystemDrive% |

Процесс загрузки можно разделить на четыре этапа:

1. **Начальная загрузка.** Загрузчик Ntldr переводит процессор в 32-разрядный режим, необходимый для выполнения дополнительных функций. Затем запускаются драйверы, встроенные в файл Ntldr, поэтому раздел может быть отформатирован как в FAT, так и в NTFS.
2. **Выбор операционной системы.** Во время загрузки Ntldr считывается файл Boot.ini и выводится на экран сконфигурированное в этом файле меню. Если файл отсутствует, то Ntldr пытается загрузиться из папки %SystemRoot% первого раздела первого диска. Подробнее о файле Boot.ini рассказано далее в этой главе.
3. **Поиск оборудования.** Поиск установленного оборудования осуществляется с помощью файла Ntdetect.com, который формирует список и передает его в Ntldr. Загрузчик в свою очередь записывает данные в реестр: `HKLM\HARDWARE`.
4. **Выбор конфигурации.** После окончания сбора информации об установленном аппаратном обеспечении выводится меню выбора конфигурации оборудования (Hardware Profile / Configuration Recovery Menu). Если существует только одна конфигурация, то меню не выводится.

## Загрузка ядра

После выбора конфигурации загружается файл Ntoskrnl.exe, который загружает ядро, инициализирует драйверы устройств и загружает службы. На этапе загрузки ядра Ntldr выполняет следующие действия:

1. Загружает файл Ntoskrnl.exe, не инициализируя его.
2. Загружает файл слоя абстрагирования от оборудования Hal.dll.
3. Загружает ключ реестра `HKLM\HARDWARE` из каталога %SystemRoot%\System32\Config\System.
4. Выбирает набор конфигурации, в котором содержится информация о драйверах, службах и т. д.
5. Загружает драйверы устройств, у которых значение параметра `Start` равно 0. Как правило, это драйверы низкого уровня, например, контроллер жесткого диска. Параметр `List` подраздела реестра `HKLM\System\CurrentControlSet\Control\ServiceGroupOrder` задает порядок их загрузки.

Файлы Ntoskrnl.exe и Hal.dll находятся в папке %SystemRoot%\System32.

## Инициализация ядра

После завершения загрузки инициализируется ядро и ему передается управление. В этот момент система отображает графический экран, в строке состояния которого можно увидеть ход загрузки. Этап инициализации состоит из четырех действий:

1. Создание раздела `HKLM\HARDWARE`. В этом разделе фиксируются данные, собранные в ходе поиска оборудования.
2. Ядром создается набор `Clone` на основе выбранного режима загрузки, фиксируемого в разделе реестра `HKLM\SYSTEM\Select`. В нем содержится несколько параметров: `Current`, `Default`, `Failed`, `LastKnownConfiguration`. В зависимости от способа загрузки каждый из параметров принимает соответствующее значение.
3. Загружаются и инициализируются драйверы устройств из списка в разделе реестра `HKLM\SYSTEM\CurrentControlSet\Services`. Физически драйверы находятся в папке %SystemRoot%\System32\Drivers.
4. Осуществляется запуск служб (service).

## Регистрация

Процесс регистрации начинается после завершения процесса инициализации ядра. Подсистемой загружается программа WinLogon.exe, которая в свою очередь запускает LocalSecurity Authority (Lsass.exe) и открывает окно Logon (вход в систему). После успешного входа в систему она создает копию управляющего набора `Clone` с именем `LastKnownGood`, которая может впоследствии использоваться для восстановления системы.

# Варианты загрузки операционной системы

Существует несколько вариантов загрузки операционной системы в случае сбоя для ликвидации последствий некорректной установки драйверов, обновлений операционной системы или других приложений, выхода оборудования из строя:

- безопасный режим (Save Mode);
- режим протоколирования загрузки (Enable Boot Logging);
- режим VGA (Enable VGA Mode);
- загрузка последней удачной конфигурации (Last Known Good Configuration);
- восстановление службы каталогов (Directory Services Restore Mode). Действителен только для контроллеров домена;

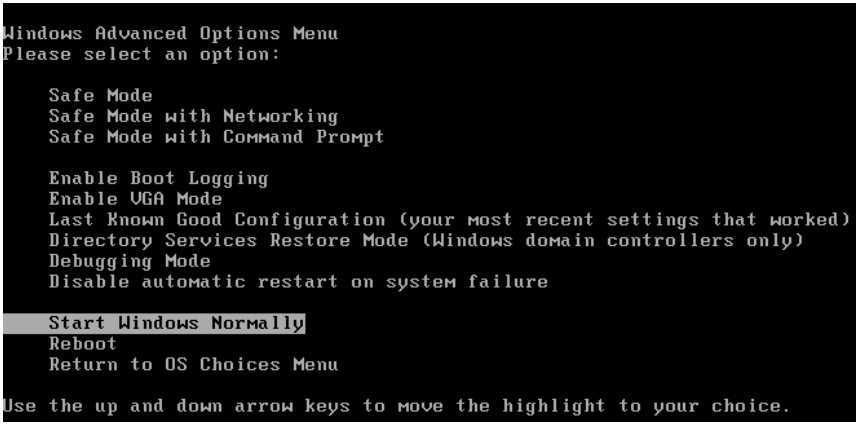



**Рис. 16.1.** Варианты загрузки операционной системы

- режим отладки (Debugging Mode);
- обычный режим (Boot Normally).

Использованный дополнительный режим загрузки записывается в переменную окружения SAFEBOOT_OPTION. Напомним, что назначенные переменные окружения можно увидеть в режиме командной строки с помощью команды `set`.

Для вызова на экран расширенного меню запуска операционной системы (рис. 16.1) необходимо нажать клавишу <F8> после выбора загружаемой операционной системы.

## Безопасный режим

В безопасном режиме загружаются только основные драйверы. Существует три вида безопасного режима:

- в безопасном режиме (Safe Mode) загружаются драйверы мыши, режима VGA для монитора, клавиатуры, жесткого диска, USB (рис. 16.2);

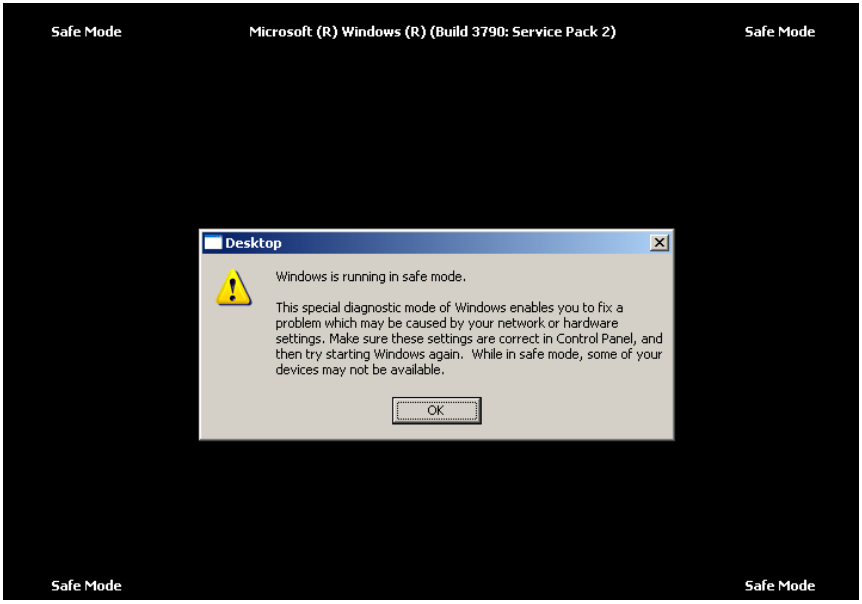



**Рис. 16.2.** Безопасный режим

- в безопасном режиме с поддержкой сети (Safe Mode with Networking) запускаются эти же драйверы, а также драйверы, необходимые для поддержки сетевых соединений;
- в безопасном режиме с командной строкой (Safe Mode with Command Prompt) графический режим заменяется командной строкой.

## Режим протоколирования загрузки

Во время загрузки операционной системы создается файл-отчет в каталоге %SystemRoot%\ntbtlog.txt. Во время загрузки операционной системы в любом из видов безопасного режима файл отчета создается автоматически.

Этот режим загрузки можно инициализировать с помощью ключа `/BOOTLOG` в файле Boot.ini (листинг 16.1).

**Листинг 16.1. Использование ключа /BOOTLOG в файле boot.ini**

```
[boot loader]
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
timeout=30
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft Windows XP
↳Professional RU" /noexecute=optin /fastdetect
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft Windows XP
↳Professional RU. Режим протоколирования" /noexecute=optin /fastdetect
↳/BOOTLOG
```

## Режим VGA

Операционная система загружается в штатном режиме. Исключение составляет видеоадаптер: вместо установленных драйверов подставляется стандартный VGA-драйвер.

Этот режим загрузки можно инициализировать с помощью ключа `/BASEVIDEO` в файле Boot.ini (листинг 16.2).

**Листинг 16.2. Использование ключа `/BASEVIDEO` в файле Boot.ini**

```
[boot loader]
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
```

```
timeout=30
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft Windows XP
↳Professional RU" /noexecute=optin /fastdetect
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft Windows XP
↳Professional RU. Режим VGA" /noexecute=optin /fastdetect /BASEVIDEO
```

## Загрузка последней удачной конфигурации

Во время загрузки используется информация из реестра о конфигурации устройств, с которой загрузка была успешно завершена в последний раз. Этот режим используется для запуска операционной системы после неудачной установки драйверов, которые вносят изменения в реестр. В Windows конфигурация хранится в ветви `HKLM\System\CurrentControlSet`. Информация обновляется каждый раз после удачной регистрации пользователя в операционной системе. Этот способ не позволяет восстановить поврежденные или удаленные файлы и не является заменой резервной копии и диска автоматического восстановления (Automatic Recovery Disk).

Этот режим загрузки можно инициализировать с помощью ключа `/LASTKNOWNGOOD` в файле Boot.ini (листинг 16.3).

**Листинг 16.3. Использование ключа `/LASTKNOWNGOOD` в файле Boot.ini**

```
[boot loader]
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
timeout=30
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft Windows XP
↳Professional RU" /noexecute=optin /fastdetect
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft Windows XP
↳Professional RU. Последняя удачная конфигурация " /noexecute=optin
↳/fastdetect /LASTKNOWNGOOD
```

## Восстановление службы каталогов

С помощью этого режима можно восстановить базу данных каталога Active Directory. Режим доступен только для контроллеров домена, на которых хранится база Active Directory.

## Режим отладки

Используя режим отладки, системный администратор может отследить ошибки, происходящие во время загрузки операционной системы. Этот вариант также можно использовать только для серверов.

## Обычный режим

В этом режиме выполняется штатная загрузка операционной системы. Подробно этот режим описан в *разд. "Процесс запуска"* ранее в этой главе.

# Файл Boot.ini

Файл Boot.ini используется для установки нескольких операционных систем на одном компьютере или для выбора различных вариантов загрузки одной и той же ОС. Этот файл располагается в каталоге %SystemDrive%. Несмотря на то, что его можно редактировать с помощью утилиты Msconfig.exe или в окне **Startup and Recovery** (Загрузка и восстановление), вызываемом из вкладки **Advanced** (Дополнительно) в окне свойств компьютера, некоторые функции остаются недоступны. Поэтому коррекция файла вручную по-прежнему актуальна.

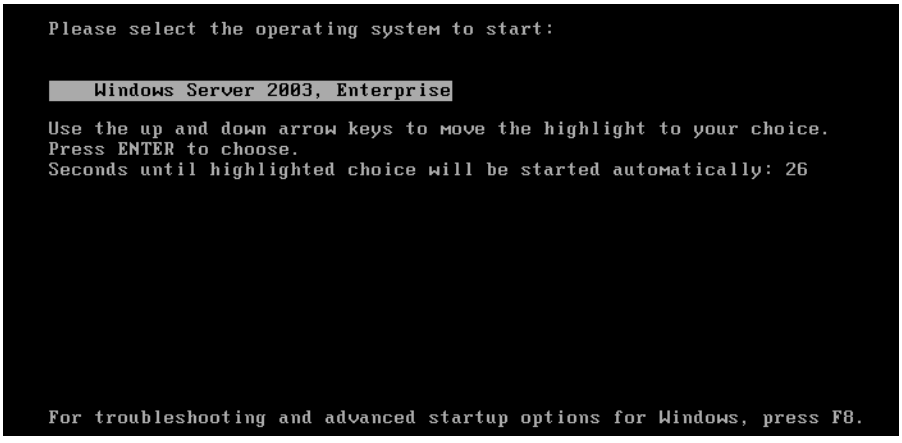



**Рис. 16.3.** Меню выбора режима загрузки ОС, генерируемое на основе файла Boot.ini

При загрузке операционной системы на основе инструкций в файле Boot.ini формируется меню выбора загрузки операционной системы (рис. 16.3).

По умолчанию этот файл имеет атрибуты "скрытый" и "только для чтения".

## Компоненты файла Boot.ini

Конфигурационный файл Boot.ini (листинг 16.4) состоит из двух блоков (секций):

- секция `[boot loader]` содержит информацию о времени ожидания выбора пользователем операционной системы и самой загружаемой по умолчанию операционной системе;
- в секции `[operating system]` перечислены все доступные для загрузки операционные системы и параметры их загрузки.

**Листинг 16.4. Пример файла Boot.ini**

```
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft Windows XP
↳Professional" /noexecute=optin /fastdetect
```

### Блок *[boot loader]*

В секции `[boot loader]` содержатся две строки вида `ключ=значение`, задающие параметры меню:

- значение, соответствующее ключу `timeout`, определяет время ожидания (в секундах), по истечении которого будет выполнена загрузка операционной системы, заданной с помощью параметра `default`. Если значение параметра `timeout` равно нулю, то меню выбора загрузки операционной системы не отображается. По умолчанию `timeout=30`;
- ключу `default` соответствует путь к загружаемой по умолчанию операционной системе. Этот путь должен совпадать с одним из путей, приведенных в секции `[operating system]`. Правила формирования пути описаны в *разд. "ARC-путь"*.

### Блок *[operating system]*

Секция `[operating system]` содержит несколько строк (по умолчанию две). Каждой операционной системе может соответствовать несколько строк. В зависимости от выбранной строки операционная система стартует в той или иной конфигурации.

Любая из строк строится по следующему шаблону:

```
ARC path="OSDisplayName" [/keys]
```

Максимально возможное число строк в этом разделе — 10, следовательно, этим же числом ограничено количество вариантов загрузки, доступных в загрузочном меню.

## ARC-путь

ARC-путь представляет собой путь, указывающий на загрузочный раздел компьютера. Путь состоит из нескольких составных частей, которые задают номер контроллера, диска и раздела (табл. 16.2).

***Таблица 16.2.*** *Составные части ACR-путей*

| Часть | Описание |
|---|---|
| `multi(x) \| scsi(x)` | В качестве параметра `x` указывается порядковый номер адаптера, с которого осуществляется загрузка. Если используется SCSI-адаптер с отключенным SCSI BIOS, то необходимо использовать ключевое слово `scsi`. Для всех других контроллеров, в том числе для SCSI с включенным SCSI BIOS, используется спецификатор `multi` |
| `disk(y)` | Параметр `y` всегда равен нулю (0), так как прерывание INT13 не может использоваться для идентификации нескольких дисковых контроллеров (для большинства BIOS) |
| `rdisk(z)` | Параметр `z` определяет порядковый номер жесткого диска, с которого производится загрузка. Нумерация начинается с нуля. Для SCSI-контроллера этот спецификатор игнорируется |
| `partition(a)` | Параметр `a` — порядковый номер раздела жесткого диска, с которого загружается операционная система. Нумерация начинается с 1, что соответствует диску C: в буквенном обозначении. В первую очередь нумеруются первичные разделы, затем идут логические диски. Не нумеруются расширенные разделы MS-DOS (тип 5) и неиспользуемые разделы (тип 0) |

После этих составных частей пути указывается имя директории, в которой находится Windows. Таким образом, ARC-путь выглядит, например, так:

```
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
```

При составлении ARC-путей надо иметь в виду, что ARC-путь, кроме имени папки, должен быть записан маленькими (строчными) буквами. В противном случае система не сможет загрузиться.

## Параметры настройки загрузки ОС

За всю историю развития файла Boot.ini, начиная от Windows NT и заканчивая Windows 2003 Server, перечень доступных ключей в значительной степени изменился. Список ключей, поддерживаемых той или иной операционной системой, приведен в табл. 16.3.

***Таблица 16.3.*** *Список ключей, используемых в файле Boot.ini*

| Параметр | Windows NT | Windows 2000 | Windows XP, Windows 2003 |
|---|---|---|---|
| /3GB | + | + | + |
| /BASEVIDEO | + | + | + |
| /BAUDRATE=число | + | + | + |
| /BOOTLOG | | + | + |
| /BOOTLOGO | | | + |
| /BREAK | | | + |
| /BURNMEMORY=число | | | + |
| /CHANNEL | | | + |
| /CLKLVL | | | + |
| /CMDCONS | | | + |
| /CRASHDEBUG | + | + | + |
| /DEBUG | + | + | + |
| /DEBUGPORT=COMn | + | + | + |

***Таблица 16.3*** *(продолжение)*

| Параметр | Windows NT | Windows 2000 | Windows XP, Windows 2003 |
|---|---|---|---|
| /EXECUTE | | | + |
| /FASTDETECT:COMn | | + | + |
| /INTAFFINITY | | | + |
| /HAL=имя_файла | + | + | + |
| /KERNEL=имя_файла | + | + | + |
| /LASTKNOWNGOOD | | | + |
| /MAXMEM=nn | + | + | + |
| /MAXPROCSPERCLUSTER= | | | + |
| /MININT | | | + |
| /NODEBUG | | + | + |
| /NOEXECUTE=уровень_политики | | | + |
| /NOGUIBOOT | | | + |
| /NOLOWMEM | | | + |
| /NOPAE | | | + |
| /NOSERIALMICE=COMn | + | | + |
| /NUMPROC=число | + | + | + |
| /ONECPU | + | | + |
| /PAE | | + | + |
| /PCILOCK | + | | + |
| /RDPATH= | | | + |
| /SAFEBOOT:параметр | | + | + |
| /SDIBOOT= | | | + |
| /SOS | + | + | + |
| /TIMERES= | | | + |

***Таблица 16.3*** *(окончание)*

| Параметр | Windows NT | Windows 2000 | Windows XP, Windows 2003 |
|---|---|---|---|
| /USERVA | | | + |
| /WIN95 | + | | + |
| /WIN95DOS | + | | + |
| /YEAR= | | | + |

Опишем эти ключи подробнее.

- Ключ /3GB применяется, начиная с NT 4.0 SP3, и предписывает перераспределять использование адресного пространства таким образом, чтобы приложениям выделялось 3 Гбайт, а системе — 1 Гбайт. По умолчанию приложениям и системе выделяется по 2 Гбайт. Ключ применим только в 32-битных операционных системах.
- Опция /BASEVIDEO заставляет после загрузки Windows использовать стандартные VGA-драйверы (рис. 16.4) для вывода изображения в графическом режиме.

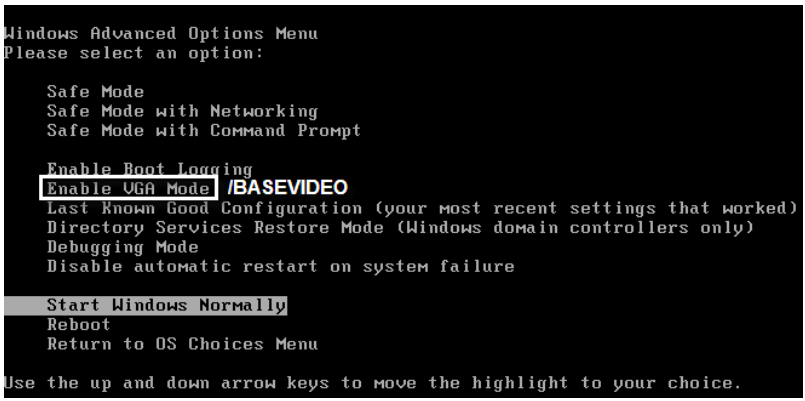



**Рис. 16.4.** Расширенное меню загрузки Windows и ключ /BASEVIDEO

- Число, указанное после ключа /BAUDRATE, определяет скорость передачи данных по COM-порту в отладочном режиме загрузки операционной системы. Для назначения имени порта используется параметр /DEBUGPORT, а

для определения скорости работы — /BAUDRATE. По умолчанию /BAUDRATE равен 19200 для нуль-модемного соединения и 9600 — для модемного. При использовании этого параметра автоматически активируется ключ /DEBUG.

- Использование параметра /BOOTLOG предписывает сохранять файл отчета о ходе загрузки операционной системы в файлы %SystemRoot%\Ntbtlog.txt (рис. 16.5).

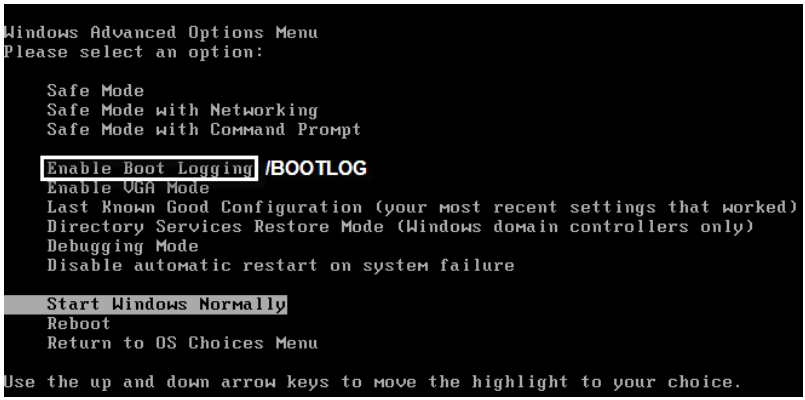



**Рис. 16.5.** Расширенное меню загрузки Windows и ключ /BOOTLOG

Приведем пример такого протокола:

```
Microsoft (R) Windows NT (R) Version 5.0
Loaded driver WINNTSystem32 toskrnl.exe
Loaded driver WINNTSystem32hal.dll
Loaded driver WINNTSystem32BOOTVID.dll
Loaded driver pci.sys
Loaded driver isapnp.sys
Loaded driver intelide.sys
...
```

- С помощью параметра /BOOTLOGO в Windows XP, Windows Server 2003 и старше можно заменить стандартную заставку (рис. 16.6). Новую заставку необходимо разместить в папке %SystemRoot%\boot.bmp. При этом к изображению предъявляются следующие требования: оно должно иметь размер 640×480 и 16 цветов. Вместе с этим параметром требуется использование параметра /NOGUIBOOT.




**Рис. 16.6.** Загрузка ОС

- Ключ `/BREAK` предписывает остановить инициализацию HAL в точке останова и ожидать подключения отладчика. Всегда используется совместно с ключом `/DEBUG`. При использовании без ключа `/DEBUG` будет отображен синий экран BSoD с кодом 0x00000078 (`PHASE0_EXCEPTION`).
- С помощью параметра `/BURNMEMORY` указывают количество оперативной памяти (в мегабайтах), которые операционная система не может использовать. Этот ключ является антиподом `/MAXMEM`.
- Ключ `/CHANNEL` используется для назначения номера канала IEEE 1394 в диапазоне от 0 (по умолчанию) до 62. Используется совместно с параметром `/DEBUGPORT=1394`.

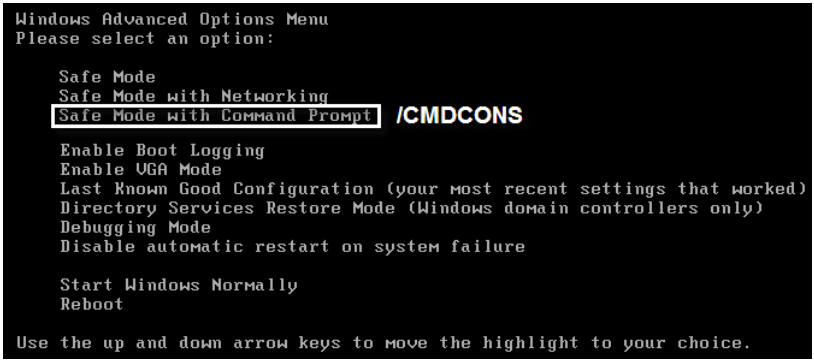



**Рис. 16.7.** Запуск консоли восстановления в расширенном меню загрузки Windows

- Параметр /CLKLVL указывает мультипроцессорному HAL на основе процессоров x86 (Halmps.dll) конфигурироваться для использования уровневого, а не синхронизирующегося по фронтам сигнала.
- Использование ключа /CMDCONS запускает консоль восстановления (Recovery Console). Эквивалентно по действию пункту **Safe Mode with Command Prompt** в расширенном меню загрузки Windows (рис. 16.7).
- При указании ключа /CRASHDEBUG на начальном этапе загрузки операционной системы запускается отладчик ядра в состоянии ожидания. Этот режим используется для анализа причин возникновения синего экрана BSoD. Используется совместно с ключом /DEBUG.
- Ключ /DEBUG активирует пошаговую загрузку операционной системы (рис. 16.8) в режиме ядра (kernel-mode).

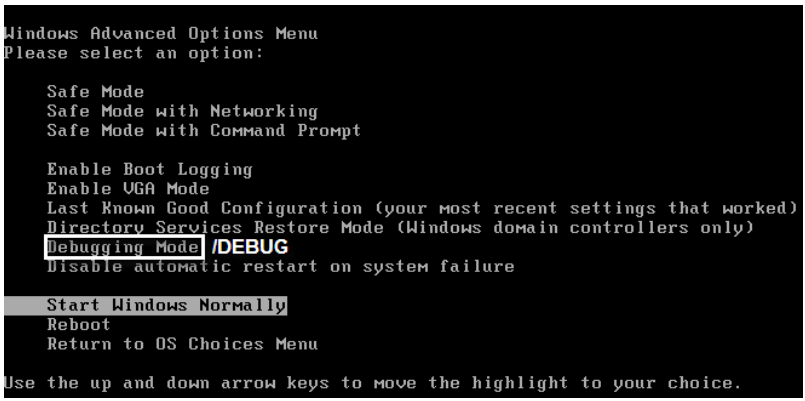



**Рис. 16.8.** Режим отладки в расширенном меню загрузки Windows

- Указание параметра /DEBUGPORT активирует загрузку модуля отладки ядра и определяет последовательный порт, используемый для отладки. По умолчанию таким портом является COM1. Для Windows XP и Windows Server 2003 может быть также использован порт IEEE 1394.

  При использовании этого параметра автоматически активируется ключ /DEBUG.

  При использовании последовательного порта необходимо указать его скорость с помощью параметра /BAUDRATE, а при использовании контроллера FireWire нужно задать /DEBUGPORT=1394 и номер используемого ка-

нала `/CHANNEL=x`, где `x` — число от 1 до 62, соответствующее использованному при настройке хоста каналу.

Примеры:

```
/DEBUGPORT = 1394 /CHANNEL = 25
/DEBUGPORT=COM3 /BAUDRATE=9600
```

- Настройка `/EXECUTE` выключает защиту исполнения от выполнения самомодифицирующегося кода (DEP). Более подробно эта возможность описана при рассмотрении ключа `/NOEXECUTE`.
- Опция `/FASTDETECT` запрещает использовать Ntdetect.com для определения установленных на компьютере компонентов. Вместо него это выполняет система Plug and Play. Этот ключ добавляется в файл Boot.ini по умолчанию.
- Ключ `/INTAFFINITY` позволяет использовать многопроцессорный HAL (Halmps.dll) для распределения прерываний. В результате нагрузка распределяется между процессорами автоматически. Без этого ключа HAL разрешает всем процессорам получать IRQ.
- Используя ключ `/HAL`, можно указать альтернативный файл, в котором хранится HAL (аппаратно зависимый слой). По умолчанию он хранится в Hal.dll.
- Параметром `/KERNEL` указывается альтернативный загрузчик ядра. По умолчанию ядро находится в файле Ntoskrnl.exe.

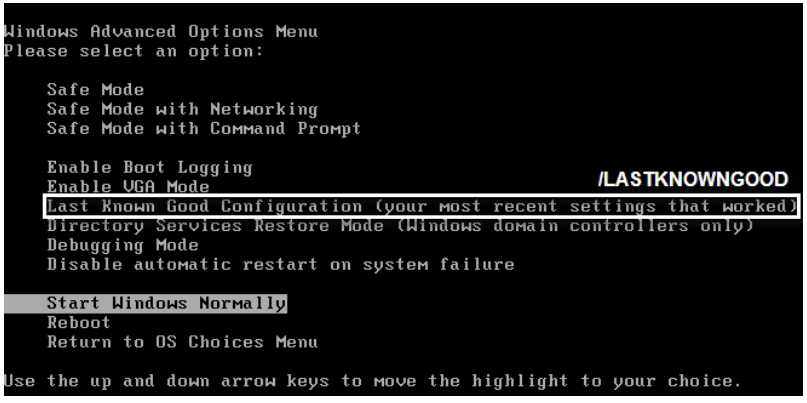



**Рис. 16.9.** Расширенное меню загрузки Windows и ключ `/LASTKNOWNGOOD`

- Ключ /LASTKNOWNGOOD позволяет восстановить последнюю удачную конфигурацию оборудования (рис. 16.9). Используется для быстрого отката при неудачной установке драйверов устройств.
- С помощью параметра /MAXMEM устанавливается максимальный размер (в мегабайтах) оперативной памяти, используемой операционной системой для работы.
- Параметр /MAXPROCSPERCLUSTER задает максимальное число процессоров на кластер. Многопроцессорный HAL в Windows 2k (Halmps.dll) предназначен для работы с многопроцессорными системами, составленными из кластеров малых мультипроцессорных систем. Например, если имеется 8-процессорная система, которая составлена из двух 4-процессорных кластеров, то ID каждого процессора должен быть определен ориентируемым кластером через HAL. Размер максимального кластера — 4 процессора. Значение по умолчанию этого параметра — 0 (система не основана на кластерах).
- Параметр /MININT используется для работы Windows PE (Preinstallation Environment).
- Ключ /NODEBUG блокирует вывод отладочной информации о ходе загрузки операционной системы. Имеет приоритет над командами /DEBUG, /DEBUGPORT и /BAUDRATE.
- Параметр /NOEXECUTE определяет уровень DEP (Data Execution Protection). Эта опция всегда включена в 64-битных системах, а в 32-битных может быть выключена. Она принимает одно из следующих значений:
  - OptIn (по умолчанию) — на компьютерах, оснащенных процессорами с поддержкой DEP, функция DEP включена для ограниченного числа системных файлов и программ, и защищаются только системные файлы Windows;
  - OptOut — функция DEP включена для всех процессов. В диалоговом окне **Система** в Панели управления можно вручную создать список приложений, для которых следует отключить DEP;
  - AlwaysOn — функция DEP включается для всей системы, и все без исключения процессы работают с выполнением проверок DEP;
  - AlwaysOff — функция DEP отключена для всей системы независимо от наличия аппаратной поддержки DEP. Процессор не работает в режиме расширенной адресации памяти (PAE), если в файле Boot.ini явно не указан параметр /PAE.

- Ключ `/NOGUIBOOT` запрещает использование VGA-драйвера на этапе загрузки системы — при этом не будет отображаться графическая заставка, и при сбоях не будут выводиться синие экраны BSOD, за показ которых также отвечает VGA-драйвер.
- Опция `/NOLOWMEM` запрещает использовать первые 4 Гбайт физической памяти. Для использования данного параметра необходимо указание параметра `/PAE`. Таким образом, все приложения и драйверы устройств будут загружаться в следующую за первыми четырьмя гигабайтами физическую память.

  Эта опция используется в системах с большой нагрузкой.
- Параметр `/NOPAE` запрещает использование расширенной адресации памяти (PAE) в системах с памятью более 4 Гбайт, поддерживающих PAE.
- Ключ `/NOSERIALMICE=[COMx | COMx,y,z...]` отключает обнаружение мыши на указанном COM-порту. Применяется при наличии компонентов, отличных от мыши, подключенных к последовательным портам во время загрузки. При использовании `/NOSERIALMICE` без указания порта обнаружение мыши отключается на всех имеющихся COM-портах.
- Указав `/NUMPROC=n`, задают количество `n` процессоров, используемых в мультипроцессорных системах.
- С помощью ключа `/ONECPU` можно принудительно использовать только один процессор в мультипроцессорных системах.
- Ключ `/PAE` включает использование механизма расширенной адресации памяти (PAE, Physical Address Extension) в принудительном порядке. При этом загружается ядро ntkrnlpa.exe, позволяющее системам x86 использовать до 64 Гбайт оперативной памяти (стандартный лимит процессоров архитектуры x86 — 4 Гбайт). При загрузке в режиме защиты от сбоев игнорируется.
- Ключ `/PCILOCK` выключает динамическое распределение IO/IRQ для PCI устройств и считывает настройку конфигурации из BIOS.
- Значение параметра `/RDPATH` — путь к образу системного диска (SDI, System Disk Image), который может находиться в сети. С этого образа осуществляется загрузка операционной системы. Совместно с этим ключом часто используется ключ `/RDIMAGEOFFSET`, значением которого является флаг загрузчика NTLDR, показывающий точку входа в файле-образа.

- С помощью параметра `/SAFEBOOT` задается один из трех режимов загрузки:
  - `/SAFEBOOT:MINIMAL` — безопасный режим без поддержки сети;
  - `/SAFEBOOT:NETWORK` — безопасный режим с поддержкой сети;
  - `/SAFEBOOT:DSREPAIR` — загрузка Windows Server в режиме восстановления службы каталога (Active Directory Services Repair) из резервной копии. Действительно только для контроллеров домена.

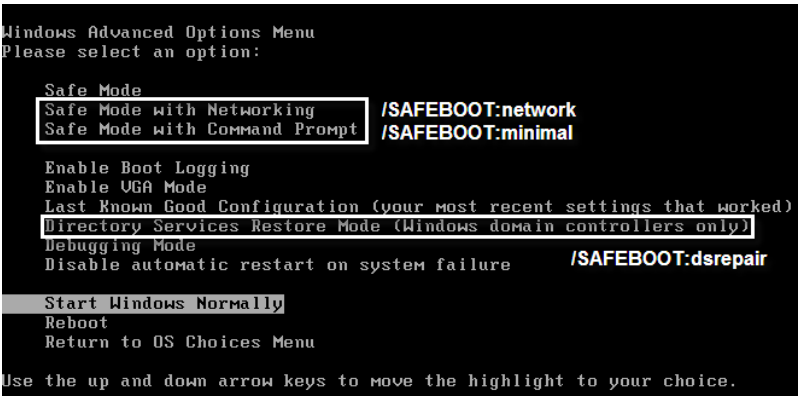



**Рис 16.10.** Безопасные режимы в расширенном меню загрузки Windows

При указании в скобках необязательной опции `ALTERNATESHELL` вместо Windows Explorer будет загружена альтернативная графическая оболочка, указанная в разделе реестра `HKLM\System\Current\ControlSet\SafeBoot\AlternateShell`.

Пример: `/SAFEBOOT:MINIMAL(ALTERNATESHELL)`. Опция `(ALTERNATESHELL)` указывается без пробела.

- Параметр `/SDIBOOT` используется для загрузки Windows с RAM-диска, на который развернуто содержимое файла-образа System Disk Image (SDI).
- Ключ `/SOS` формирует список загружаемых драйверов, содержащий время загрузки, название, версию (включая номер сборки), размер драйвера в ОЗУ и количество процессов. Рекомендуется использовать для выявления драйвера, вызывающего сбой.
- При помощи параметра `/TIMERES` в мультипроцессорных системах (Halmps.dll) устанавливают разрешение системного таймера. По умолча-

нию разрешение таймера 7,8 мс. Параметр измеряется в сотнях наносекунд (табл. 16.4).

***Таблица 16.4.*** *Поддерживаемые параметром* `/TIMERES` *значения*

| Сотен наносекунд | Миллисекунд |
|---|---|
| 9766 | 0,98 |
| 19532 | 2,0 |
| 39063 | 3,9 |
| 78125 | 7,8 |

- Ключ `/USERVA` используется только для 32-битных версий Windows XP, Windows Server 2003 и старше. Этот ключ позволяет более тонко настроить объем оперативной памяти, отдаваемой пользовательским программам. Значение параметра задает объем памяти (в мегабайтах), отдаваемый прикладным программам, и должно лежать между 2048 и 3072. При использовании этого ключа должен быть указан ключ `/3GB`.
- Ключ `/WIN95` применяется при использовании трех различных операционных систем: MS-DOS, Windows 9*x* и Windows 2k. Он создает ссылку в Ntldr.exe на загрузочный сектор Windows 9*x*, хранящийся в файле Bootsect.w40.
- Ключ `/WIN95DOS` также применяется при использовании трех различных операционных систем: MS-DOS, Windows 9*x* и Windows 2k. Он создает ссылку в Ntldr на загрузочный сектор MS-DOS, хранящийся в файле Bootsect.dos.
- Параметр `/YEAR` задает текущий год и предписывает игнорировать год, выдаваемый встроенными часами ПК. Это работает, начиная с Windows NT 4.0 SP4. Данный ключ используется в старых компьютерах для решения проблемы 2000 года.

ГЛАВА 17



# Особенные файлы Windows

В этой главе приведено описание синтаксисов нескольких основных файлов Windows, изложены основные принципы управления некоторыми из них.

## Основные термины и понятия

- **Файл подкачки** (paging file) — скрытый файл, используемый операционной системой для расширения размера доступной оперативной памяти.
- **Пакетный файл** — текстовый файл, предназначенный для последовательной обработки набора (пакета) программ. Используется для автоматизации рутинных действий.

## Файл подкачки

Файл подкачки (paging file) — скрытый файл, расположенный на жестком диске компьютера, используемый операционными системами семейства Windows для увеличения размера оперативного запоминающего устройства (ОЗУ). В сумме физическая память и файл подкачки составляют виртуальную память.

### Размер файла подкачки

По умолчанию размер файла подкачки определяется следующим образом:

- если общий объем ОЗУ меньше 2 Гбайт, то размер файла подкачки в 1,5 раза превышает объем установленного на компьютере ОЗУ, но не превышает 2 Гбайт;

- если общий объем ОЗУ больше или равен 2 Гбайт, то размер файла подкачки равен 2 Гбайт.

Максимальный размер файла подкачки равен 4 Гбайт. Это ограничение обусловлено архитектурой процессора x86 и действует на каждый том. Если необходим файл подкачки большего размера, то можно создать несколько файлов подкачки на одном или нескольких дисках.

На серверах с установленной операционной системой Windows Server 2003 SP1 и выше можно создать один файл подкачки, размер которого превышает 4 Гбайт.

## Снятие ограничения в 4096 Мбайт на размер файла подкачки

Чтобы создать файл подкачки большего размера, необходимо загрузить ядро операционной системы в расширенном режиме. Это может быть сделано автоматически, если в Windows 2003 Server используется устройство "горячего" подключения памяти, или принудительно, для чего нужно добавить в файл Boot.ini параметр PAE (рис. 17.1). Подробная информация о синтаксисе файла Boot.ini приведена в *разд. "Файл Boot.ini"* в *главе 16*.

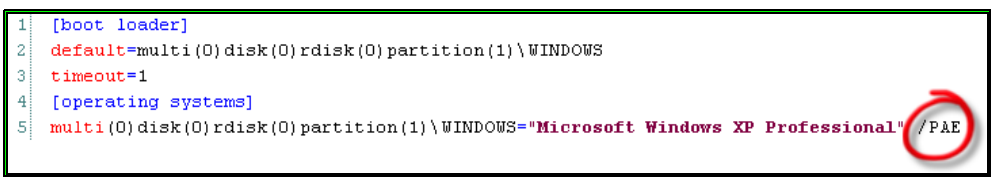



**Рис. 17.1.** Файл Boot.ini

## Размещение файла подкачки

Для увеличения быстродействия системы файл подкачки рекомендуется переместить на отдельный раздел жесткого диска. По умолчанию он находится в загрузочном разделе. В этом разделе, как правило, расположены папки Windows и Program Files.

При таком размещении разделяются два потока ввода-вывода: операционной системы в системные папки (например, C:\Windows) и файла подкачки.

Однако при такой схеме размещения операционная система не сможет создать дамп памяти (memory.dmp) в случае критической ошибки в ядре операционной системы.

Оптимальным решением является создание одного файла подкачки в загрузочном разделе, а второго — в разделе, который менее всего используется, в идеале — на отдельном жестком диске. По умолчанию операционная система использует менее загруженный файл подкачки. Таким образом, в случае критической ошибки будет создан дамп памяти. При этом общее быстродействие операционной системы увеличится за счет разделения потоков ввода-вывода. Файл подкачки на загрузочном разделе должен быть номинален.

## Включение нескольких файлов подкачки

Создание нескольких файлов подкачки позволит разнести потоки ввода-вывода операционной системы и файла подкачки, а также превысить установленное ограничение на размер файла подкачки в 4 Гбайт.

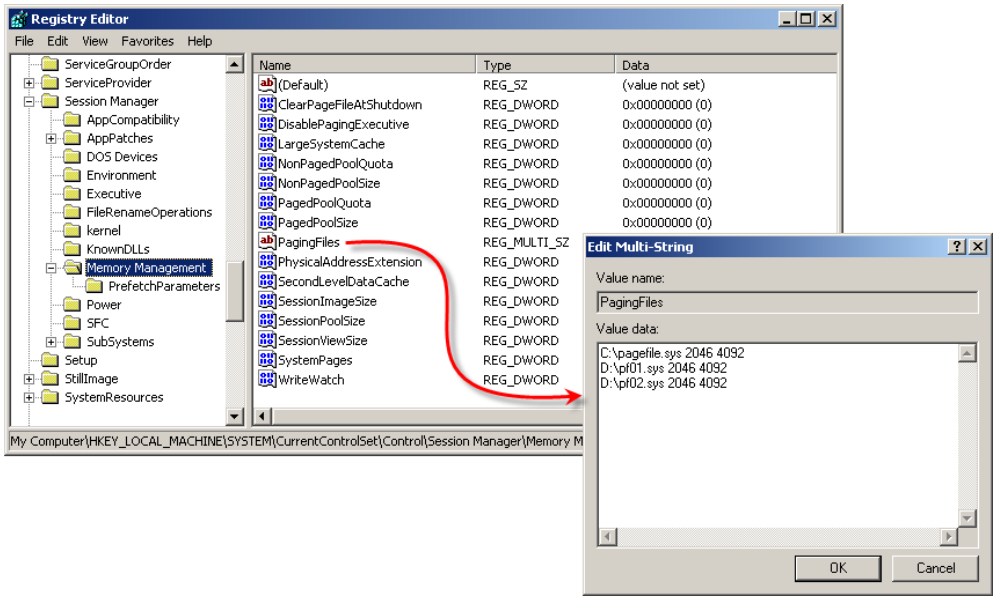



**Рис. 17.2.** Параметр `PagingFiles`

Расположение и имя файла подкачки определяется значением многострочного (`REG_MULTI_SZ`) параметра `PagingFiles`, расположенного в реестре (рис. 17.2) в разделе
`HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management`.

Каждая строка параметра `PagingFiles` состоит из трех частей: полного пути к файлу подкачки, минимального и максимального размера в мегабайтах. Для создания нескольких файлов подкачки необходимо добавить несколько строк.

В листинге 17.1 приведен пример добавления еще двух файлов подкачки к существующему по умолчанию (рис. 17.2).

**Листинг 17.1. REG-файл создания нескольких файлов подкачки**

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session
↳Manager\Memory Management]

"PagingFiles"=hex(7):43,00,3a,00,5c,00,70,00,61,00,67,00,65,00,66,00,69,\
00,6c,00,65,00,2e,00,73,00,79,00,73,00,20,00,32,00,30,00,34,00,36,00,20,\
00,34,00,30,00,39,00,32,00,00,00,44,00,3a,00,5c,00,70,00,66,00,31,00,2e,\
00,73,00,79,00,73,00,20,00,32,00,30,00,34,00,36,00,20,00,34,00,30,00,39,\
00,32,00,00,00,44,00,3a,00,5c,00,70,00,66,00,32,00,2e,00,73,00,79,00,73,\
00,20,00,32,00,30,00,34,00,36,00,20,00,34,00,30,00,39,00,32,00,00,00,00,\
00
```

Данные в параметре `PagingFiles` хранятся в бинарном виде в Unicode-кодировке. Каждый символ в этой кодировке состоит из 2-х байт. В первом байте записан код символа, а во втором — сдвиг в таблице Unicode. Для латинских букв он равен 00.

# Файл Autorun.inf

Файл Autorun.inf используется для запуска программы установки после вставки компакт-диска в привод, автоматического запуска слайд-шоу или проигрывателя. Местоположение файла — корневая папка сменного диска. Большинство файлов крайне просты, однако возможности файла больше, чем чаще всего применяют на практике. Расскажем о них подробнее.

## Синтаксис

Любой файл Autorun.inf начинается с раздела `[Autorun]`, в котором содержатся все параметры, описывающие внешний вид, контекстное меню, метку тома

диска и др. Всего таких параметров шесть: `Action`, `Icon`, `Label`, `ShellExecute` или `Open`, `Shell`. Рассмотрим каждый из этих параметров подробнее.

## Параметры *Open* и *ShellExecute*

В файле Autorun.inf в разделе `[Autorun]` всегда присутствует обязательный параметр `Open` или `ShellExecute`:

```
Open = [path\]filename [[[value1] [value2],…[ valuen]]
ShellExecute = [path\]filename [[[value1] [value2],…[ valuen]]
```

Они схожи по действию, но не являются полными синонимами. С их помощью определяют приложение, которое будет запущено при попытке открыть диск в Проводнике при условии, что включен режим автоматического воспроизведения диска. Однако в параметре `Open` допустимо использовать только имя исполняемого файла, тогда как параметр `ShellExecute` может содержать произвольный файл, и программа, используемая для работы с ним, определяется из настроек системы. Примеры:

```
Open = MosMap.exe
ShellExecute = program.msi /s
```

## Параметр *Icon*

Этот параметр задает значок, отображаемый в проводнике Windows (рис. 17.3):

```
Icon = filename[,index]
```

Обычно файлы, содержащие иконки, имеют расширение ICO. Также воспринимаются файлы с расширениями BMP, EXE, DLL. Если файл имеет расширение EXE или DLL, то необходимо указать порядковый номер отображаемой иконки. Нумерация иконок в файле начинается с нуля. Примеры:

```
Icon = AutoRun.Inf
Icon = MosMap.exe,1
```

## Параметр *Label*

С помощью параметра `Label` присваивается метка тома диска:

```
Label = LABEL TEXT
```

Метка тома видна в папке Мой компьютер (см. рис. 17.3). Пример:

```
Label = Карта г.Москвы
```

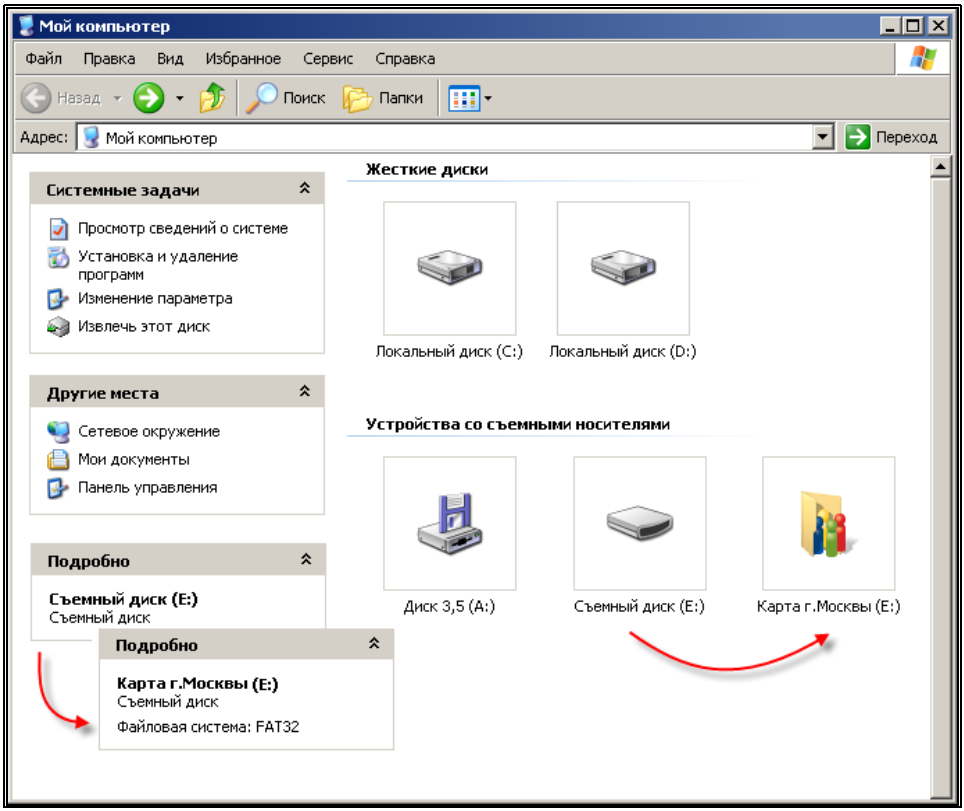



**Рис. 17.3.** Изменение внешнего вида диска с помощью файла Autorun.inf

## Параметр *Action*

Указанный в параметре `Action` текст отображается первым пунктом в диалоговом окне **Автозапуск** (AutoPlay) программ (рис. 17.4):

```
Action = COMMENT TEXT
```

Используется в Windows XP SP2 и выше. Пример:

```
Action = Запуск приложения "Карта г. Москвы"
```

## Параметр *Shell*

Команда, назначаемая по умолчанию в контекстном меню диска:

```
Shell = SubMenuName
```

По умолчанию на жестких дисках установлено значение **Открыть** (Open), а на CD-ROM/DVD-ROM, как правило, **Автозапуск** (Autorun). Примеры:

```
Shell = ReadMe
Shell =
```

Если значение параметра `Shell` не указано, как в последнем примере, то при попытке чтения диска будет выведено диалоговое окно **Автозапуск**.

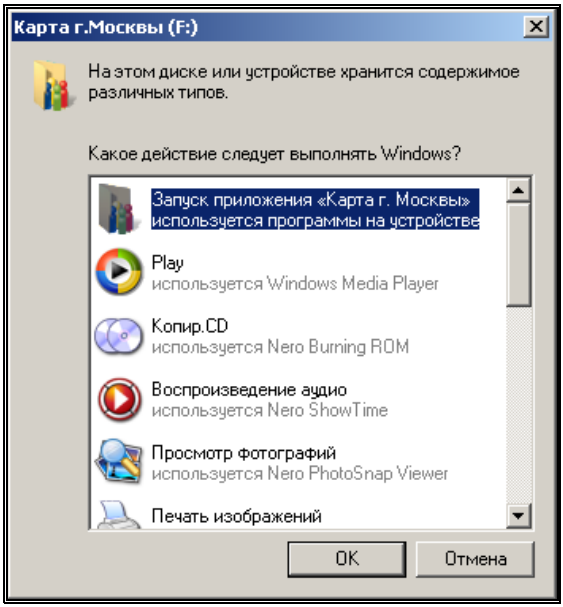



**Рис. 17.4.** Диалоговое окно **Автозапуск**

## Дополнительные возможности *Shell*

С помощью команды `Shell` можно добавить в контекстное меню новые пункты. Для этого необходимо задать два параметра: название подменю и путь к файлу, который будет запускаться при выборе этого пункта.

```
Shell\SubMenuName = Value
Shell\SubMenuName\Command = [path\]filename
```

Пример:

```
Shell\C01 = ReadMe
Shell\C01\Command = Notepad.exe Readme.txt
```

# Пакетный файл

Пакетный файл представляет собой неформатированный текстовый файл, содержащий одну или несколько команд и имеющий расширение BAT или CMD. Пакетные файлы, также называемые программами пакетной обработки или сценариями, позволяют упростить выполнение утомительных и часто используемых операций.

Такая технология позволяет пользователю создавать программы пакетной обработки, выполняющие несколько последовательно записанных команд.

Пакетные файлы предоставляют гибкие возможности создания сценариев, с их помощью возможно написание и маленьких линейных последовательностей команд, и сложных сценариев с условными ветвлениями и циклами. Технология файлов пакетной обработки используется во многих ОС и оболочках, среди которых Windows, OS/2, Mac OS и Linux-подобные системы.

## Различие между BAT и CMD

Термин "пакетный файл" является устоявшимся и может подразумевать пакетный либо командный файл. Несмотря на внешнее сходство процедур создания и применения, пакетные и командные файлы имеют множество различий. Пакетные файлы (BAT) использовались еще в ранних версиях MS-DOS как средство объединения нескольких команд в одну. Эти файлы выполнялись командным интерпретатором command.com и были основным средством автоматизации.

Начиная с Windows NT используются командные файлы (CMD), но пакетные также поддерживаются.

Командные файлы являются усовершенствованной технологией пакетных и выполняются интерпретатором Cmd.exe. Пакетные файлы поддерживаются вплоть до Windows Vista, но обладают ограниченным набором команд по сравнению с командными файлами и выполняются медленнее.

Для Windows NT, 2k и выше рекомендуется использование командных файлов, для более старых версий — файлов пакетной обработки, так как ничего другого не предусмотрено.

## Формальные параметры

При запуске любого пакетного файла (BAT, CMD) может быть указано до 9 аргументов командной строки. Для распознавания заданных значений

внутри файла используется до десяти формальных параметров, обозначаемых `%0...%9`.

Параметр `%0` всегда возвращает имя запущенного сценария. При запуске сценария example01.bat, содержащего строку `echo FileName: %0`, будет возвращена строка `FileName: example01.bat` (рис. 17.5).

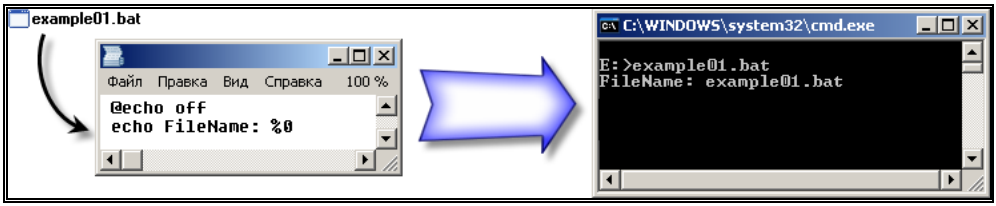



**Рис. 17.5.** Пример использования формального параметра `%0`

Остальные параметры (`%1...%9`) содержат аргументы командной строки. Например, при запуске файла (рис. 17.6) указываются два аргумента командной строки, которые впоследствии выводятся на экран консоли.

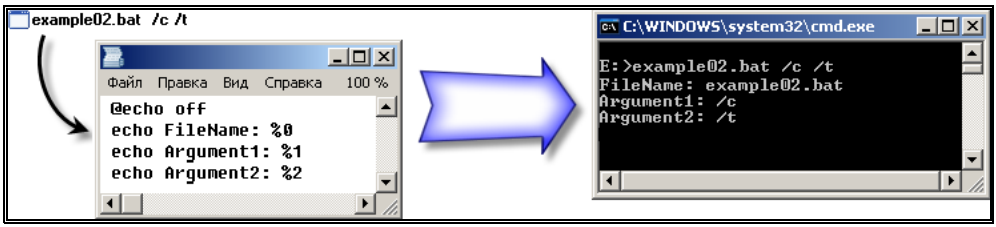



**Рис. 17.6.** Пример использования формальных параметров `%0...%2`

## Команды пакетного файла

Пакетные файлы поддерживают следующие команды:

- `call` обеспечивает последовательное выполнение команд пакетного файла;
- `echo` выводит сообщения на экран;
- `endlocal` сбрасывает локальное окружение, установленное для пакетного файла;

- `for` обеспечивает пакетную обработку наборов;
- `goto` осуществляет переход на указанную метку;
- `if` позволяет проверять различные условия и менять в зависимости от результата ход выполнения;
- `pause` приостанавливает действие программы до нажатия какой-либо клавиши на клавиатуре;
- `rem` служит для создания комментариев в листинге файла;
- `setlocal` создает локальное окружение для пакетного файла;
- `shift` сдвигает формальные параметры.

Написание команд является регистронезависимым, однако чаще всего употребляют строчные буквы.

## Команда *call*

Команда `call` используется для вызова одного пакетного файла из другого без завершения выполнения первого файла (рис. 17.7). В приведенном примере сперва осуществляется вывод сообщения, затем вызывается второй файл (second.bat). Во время его выполнения выводится сообщение из второго файла. По окончанию работы второго файла осуществляется передача управления обратно первому файлу и вывод третьего сообщения.

Команда `call` допускает использование меток в качестве адресата вызова:

```
call:Label
```

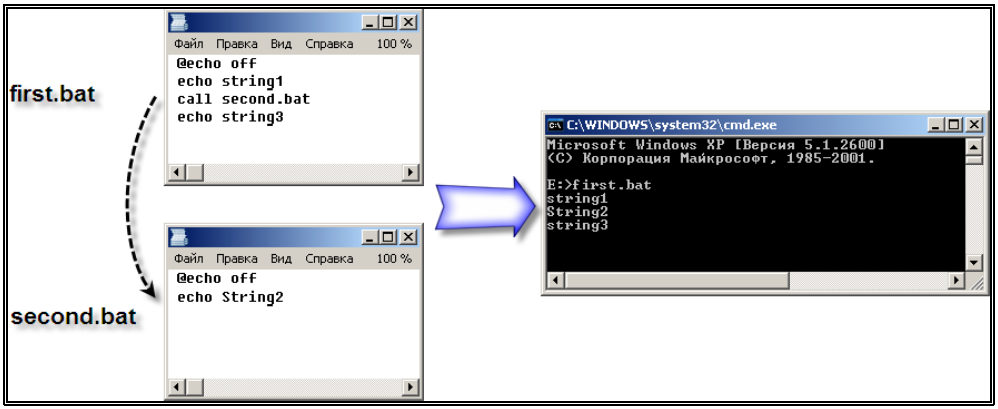



**Рис. 17.7.** Работа команды `call`

При вызове создается новый контекст текущего пакетного файла с заданными аргументами и управление передается на инструкцию, расположенную сразу после метки. Для выхода из такого пакетного файла необходимо дважды достичь его конца. Первый выход возвращает управление на инструкцию, расположенную сразу после строки `call`, а второй выход завершает выполнение пакетного файла. Команда `goto :eof` позволяет выполнить быстрый возврат из пакетного файла.

Также совместно с командой `call` могут быть использованы пакетные параметры (`%1…%9`) или переменные (например, `%str1%`).

## Команда *echo*

С помощью команды `echo` управляют режимом вывода команд из пакетного файла на экран. По умолчанию включен режим отображения команд на консоли (`echo on`), однако это не всегда удобно, поэтому командные файлы часто начинают с команды `@echo off`. Символ `@` в начале этой строки подавляет и ее вывод на экран. Дополнительно команда `echo` может быть использована для вывода сообщения в консоли (рис. 17.7). В тексте сообщений нельзя использовать символы перенаправления (> или <) или создания канала (`|`).

Синтаксис команды: `ECHO [ON | OFF | <Message>]`

## Команда *endlocal*

С помощью команды `endlocal` обнуляется локальное окружение в пакетной программе и происходит восстановление значений переменных среды, имевшихся на момент выполнения команды `setlocal`. Команда `endlocal` всегда вызывается неявно при завершении работы пакетного файла. Пример использования см. в *разд. "Команда setlocal"*.

## Команда *for*

Встроенная команда `for` используется для пакетной обработки набора, например, набора файлов или папок. Команда (цикл `for`) имеет следующий синтаксис для пакетной обработки:

```
for %%<C> in <ARRAY> do <COMMAND>
```

Для работы в интерактивном режиме нужно использовать один знак `%`:

```
for %<C> in <ARRAY> do <COMMAND>
```

В приведенных шаблонах `<C>` может быть любым символом, кроме цифр, чтобы исключить путаницу с параметрами `%0`, `%1`, …, `%9` пакетного файла. Этот параметр регистрозависим. Задание двух символов `%%` необходимо, так как после отбрасывания формальных параметров (`%0`…`%9`) остается только один символ процента.

Команда `for` просматривает список элементов, заданных после ключевого слова `in`, и выполняет с текущим элементом (`%%<C>`) действие, указанное после ключевого слова `do` (рис. 17.8). В приведенном на рисунке листинге осуществляется пофайловое копирование файлов с расширением LOG из системного каталога (%SystemRoot%) в папку C:\1.

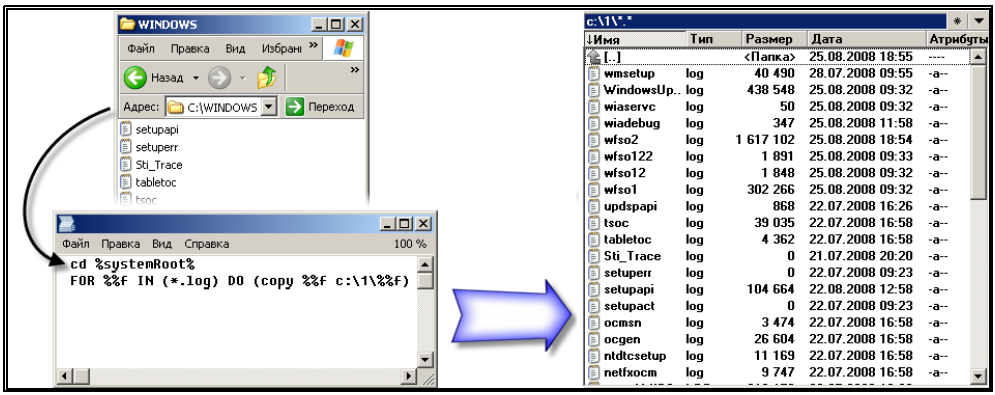



**Рис. 17.8.** Работа цикла `for`

Команда `for` позволяет также организовывать более сложные циклы, просматривать содержимое файлов и т. п., но синтаксис этих операций сложен и они редко используются. Если это все же необходимо, обратитесь к специальной литературе.

## Команда *goto*

Команда `goto` позволяет переместить выполнение программы на команду, обозначенную меткой. Метка может стоять в любом месте файла и состоит из двоеточия в первой позиции строки, за которой следует буквенно-цифровая комбинация. Значащими в метке являются только первые 8 символов.

Команда `goto` имеет следующий синтаксис:

```
goto <LABEL>
…
:<LABEL>
```

или

```
:<LABEL>
…
goto <LABEL>
```

В этих примерах `<LABEL>` — метка, предваряемая двоеточием, при переходе активного курсора. Если по ошибке метка не задана, то пакетный режим будет прерван, а выполнение сценария завершено.

Несмотря на то, что функция перевода текущей команды может быть удобной во многих случаях, ею не рекомендуют пользоваться (или по крайней мере злоупотреблять), поскольку множество меток легко запутает программиста; к тому же использование меток сильно снижает скорость работы.

Самым подходящим использованием меток является создание многострочных комментариев (листинг 17.2), не отображаемых в ходе выполнения программы.

**Листинг 17.2. Создание многострочных комментариев с помощью оператора `goto`**

```
goto LBL
Комментарий: строка 1
Комментарий: строка 2
…
Комментарий: строка n
:LBL
```

## Команда *if*

Условный оператор `if` используется для пакетной обработки. При выполнении заданного в нем условия выполняется определенное им действие. Синтаксис оператора `if` следующий:

```
if <CONDITION> <COMMAND>
```

Условие <CONDITION> может быть задано одним из следующих способов:

- ERRORLEVEL <NUMBER> используется для обработки ошибок;
- <STRING1>==<STRING2> проверяет идентичность строк;
- exists <OBJECT> позволяет выяснить, существует ли файл или папка;
- not <CONDITION> служит для отрицания заданного условия.

Вместо <COMMAND> подставляется команда, которую надо исполнить лишь в случае, если заданное условие выполнено.

В листинге 17.3 проверяется переменная окружения %OS%, содержащая название операционной системы, и если ее значение равно Windows_NT, выводится поясняющий текст.

**Листинг 17.3. Пример использования условного оператора if**

```
if "%OS%" == "Windows_NT" echo Используется система Windows 2K
```

## Команда *setlocal*

Командой setlocal определяется начало области задания для определения переменных среды в пакетном файле. Локальное окружение используется до тех пор, пока не встретится команда endlocal или не будет достигнут конец пакетного файла. В пределах одного командного файла можно использовать команды setlocal и endlocal несколько раз.

Команда setlocal имеет следующий синтаксис:

```
setlocal [enableextensions | disableextensions] [enabledelayedexpansion |
↳disabledelayedexpansion]
```

Возможно задавать следующие аргументы:

- enableextensions включает расширения командного процессора до появления соответствующей команды endlocal;
- disableextensions выключает расширения командного процессора до появления соответствующей команды endlocal;
- enabledelayedexpansion включает расширения переменной среды с задержкой до появления соответствующей команды endlocal;
- disabledelayedexpansion выключает расширения переменных среды с задержкой до появления соответствующей команды endlocal.

На практике перечисленные аргументы используются достаточно редко. На рис. 17.9 проиллюстрирован механизм использования команд `setlocal` и `endlocal`.

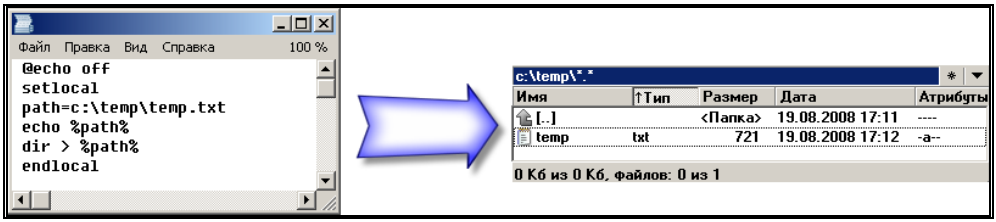



**Рис. 17.9.** Использование команд `setlocal` и `endlocal`

## Команда *pause*

Эта команда приостанавливает выполнение пакетной программы и выводит сообщение, предлагающее пользователю нажать любую клавишу (рис. 17.10).

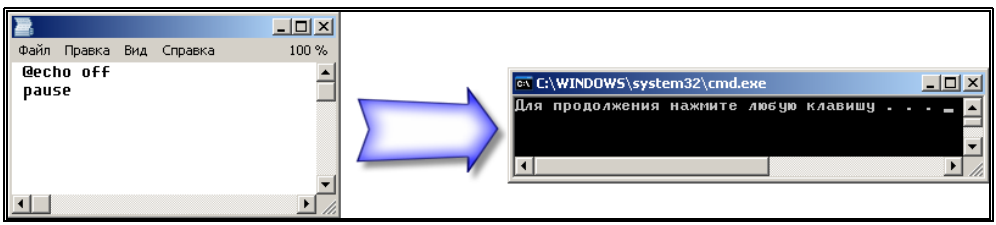



**Рис. 17.10.** Использование команды `pause`

## Команда *rem*

Команда `rem` добавляет комментарии в пакетные файлы (рис. 17.11). Она не выводит комментарии на экран. Их можно посмотреть только с помощью редактора сценария, например, notepad или PrimalScript. Команда `rem` имеет следующий синтаксис:

```
rem [TEXT]
```

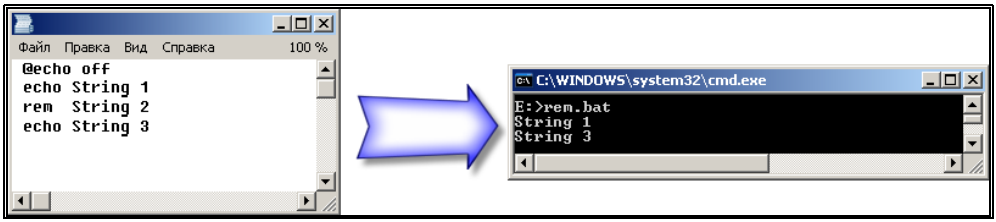



**Рис. 17.11.** Использование команды `rem`

## Команда *shift*

С помощью команды `shift` осуществляется сдвиг формальных параметров. Команда `shift` поддерживает параметр командной строки `/n`, который указывает команде начинать сдвиг с `n`-го аргумента, где `n` — число от нуля до восьми. Например, `shift /2` запишет значение переменной `%3` в переменную `%2`, `%4` — в `%3` и т. д. Значения переменных `%0` и `%1` останутся неизменными.

Команда `shift` изменяет значения замещаемых параметров `%0`—`%9`, копируя значение каждого параметра в предыдущий. Такой прием оказывается полезным при создании пакетных файлов, выполняющих одну и ту же операцию над любым числом параметров.

Используя эту команду можно назначить более 9 аргументов при запуске приложения, при этом все параметры, следующие за девятым (то есть за переменной `%9`), будут последовательно помещены в эту переменную `%9`.

## Глава 18



# Windows Installer

В этой главе подробно рассказано об обеспечении процесса автоматической установки программного обеспечения на основе Windows Installer разными способами, начиная с назначения параметров командной строки и заканчивая изменением базы данных внутри дистрибутива.

## Основные термины и понятия

- **SDK** (Software Development Kit) — набор средств разработки, утилит и документации, который позволяет разработчикам создавать и управлять приложениями по определенной технологии.
- **Windows Installer** — подсистема Microsoft Windows, обеспечивающая установку программ. Начиная с Windows 2000 эта технология является составной частью Windows. Вся необходимая для установки информация содержится в установочных пакетах (Installation Packages), известных как MSI-файлы.

## Пакетная установка программного обеспечения

Для установки нескольких программ в автоматическом режиме обычно используется пакетная установка на основе командных файлов (*.BAT или *.CMD). Поскольку Windows Installer не поддерживает одновременную работу нескольких экземпляров, то необходимо обеспечить последовательное выполнение процессов установки. Для этого используется команда `start /wait`:

```
start /wait %systemdrive%\install\setup.exe /s
```

# Windows Installer

Windows Installer — это сервис установки и конфигурирования программных продуктов, который входит в состав операционной системы. Также он может устанавливаться как пакет обновления или отдельный дистрибутив. В настоящее время используется Windows Installer версии 4.5 (рис. 18.1), который можно бесплатно загрузить с сайта Microsoft в виде быстрого исправления (hotfix) KB942288: **http://download.microsoft.com/download/2/6/1/261fca42-22c0-4f91-9451-0e0f2e08356d/WindowsXP-KB942288-v3-x86.exe**.

Windows Installer поддерживает ряд ключей командной строки, обеспечивающих управление автоматической установкой пакета:

- `/norestart` инструктирует не перезагружать компьютер после установки обновления;
- `/quiet` заставляет выполнить установку в "тихом" режиме;
- `/help` приводит к выводу на экран доступных ключей и их краткого описания.

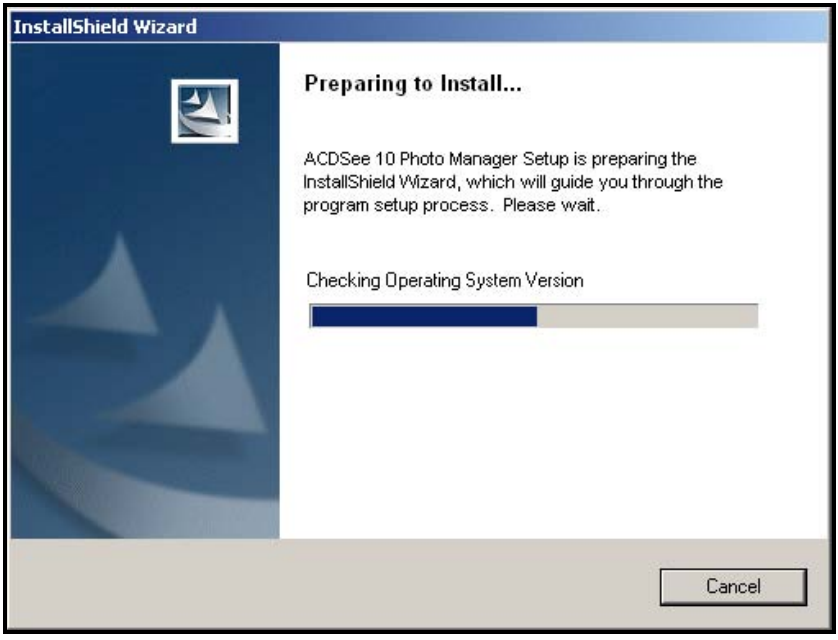



**Рис. 18.1.** Windows Installer

На каждом компьютере, использующем Windows Installer, хранится единая база данных с информацией о каждом установленном с помощью этой технологии приложении. Она включает файлы, записи в реестре и компоненты. При удалении приложения с машины Windows Installer проверяет базу данных, чтобы удостовериться в том, что не будут удалены файлы, ключи реестра и компоненты, нужные другим приложениям. Таким образом, удаление приложения не повлияет на работоспособность других программ, установленных на компьютере.

Использование технологии Windows Installer дает пользователям следующие преимущества:

- более простая и быстрая установка программного обеспечения;
- возможность установки по требованию — информация о неустановленных компонентах приложения хранится в одном месте, и при обновлении конфигурации программы нет необходимости переустанавливать все компоненты;
- самовосстановление программ. Хранение инсталляционной информации в одном месте позволяет приложению самостоятельно восстанавливаться. Неправильно работающее приложение может проверить инсталляционные данные, определить, какие файлы повреждены или отсутствуют, а затем восстановить их;
- возможности отката. Windows Installer позволяет отменять любые изменения в конфигурации как устанавливаемого продукта, так и операционной системы. Это делает установку программ, поддерживающих технологию Windows Installer, относительно безопасным и предсказуемым занятием.

Благодаря перечисленным преимуществам пользователи теряют меньше времени на удаление и переустановку приложений, а также избавляются от необходимости исправлять непонятные и трудно диагностируемые ошибки в конфигурации приложения.

## Параметры командной строки для Msiexec

Файл Msiexec.exe представляет собой оболочку, которая управляет основной библиотекой msi.dll. Эта графическая оболочка позволяет:

- разбирать параметры командной строки и выполнять соответствующие задачи;

- устанавливать при выходе уровень ошибки, соответствующий системным кодам ошибок (**http://msdn.microsoft.com/en-us/library/aa368542.aspx**);
- связывать MSI-файлы с Windows Installer командой

```
"%SystemRoot%\System32\msiexec.exe" /i "your_filename.msi
```

Параметры командной строки, которые понимает Msiexec.exe (**http://technet.microsoft.com/en-us/library/cc759262.aspx**), приведены в табл. 18.1. Полный список параметров приведен на сайте Microsoft.

***Таблица 18.1.*** *Параметры командной строки msiexec.exe*

| Параметр | Описание |
|---|---|
| `/i` | Установка приложения в обычном режиме |
| `/f [p\|o\|e\|d\|c\|a\|u\|m\|s\|v]` | Переустановка приложения. Если задан только ключ `/f`, то имеется в виду команда `/fpecsm`. В общем случае можно задавать следующие параметры:<br>• `p` — восстановление удаленных файлов;<br>• `o` — обновление до более новой версии;<br>• `e` — полная переустановка существующей версии приложения;<br>• `d` — восстановление удаленных файлов и перезапись файлов более старой версии;<br>• `c` — восстановление удаленных файлов или перезапись файлов, если CRC (контрольная сумма) установленных файлов не совпадает с CRC файлов в дистрибутиве;<br>• `a` — переустановка всех файлов;<br>• `u` — перезапись всех ключей реестра, касающихся данного приложения, в ветвях `HCKU`, `HCU`;<br>• `m` — перезапись всех ключей реестра, касающихся данного приложения, в ветви `HCLM`;<br>• `s` — перезапись всех ярлыков;<br>• `v` — запустить с исходного носителя и заново кэшировать локальный каталог |

***Таблица 18.1*** *(окончание)*

| Параметр | Описание |
|---|---|
| `/a` | Выполнение административной установки приложения |
| `/x` | Деинсталляция продукта |
| `/j[u\|m[/g]]` | Управление публикацией программного продукта:<br>• `u` — публикация для текущего пользователя;<br>• `m` — публикация для всех пользователей;<br>• `g` — идентификатор языка (управление установкой языкового интерфейса) |
| `/q[n\|b\|r\|f]` | Определение уровня пользовательского интерфейса во время установки:<br>• `n` — отсутствие интерфейса;<br>• `b` — базовый интерфейс (по умолчанию);<br>• `r` — сокращенный интерфейс;<br>• `f` — полный пользовательский интерфейс |

# Windows Installer SDK

Пакет утилит Windows Installer SDK, который можно загрузить бесплатно с сайта компании Microsoft по адресу **http://www.microsoft.com/downloads/details.aspx?displaylang=ru&FamilyID=6a35ac14-2626-4846-bb51-ddce49d6ffb6**, используется для различного рода манипуляций с MSI-файлами. По умолчанию каталог установки Windows Installer SDK — %ProgramFiles%\Windows Installer 4.5 SDK.

В его комплект входят несколько утилит, в том числе дистрибутив программы Orca.exe, расположенный в папке Tools.

# Доступ к MSI-файлам с помощью графической оболочки Orca.exe

Программа Orca.exe, дистрибутив которой находится в папке %ProgramFiles%\Windows Installer 4.5 SDK\Tools, предназначена для моди-

фикации MSI-файлов. С помощью этой утилиты можно не только управлять устанавливаемыми компонентами, но и указать серийный номер продукта, изменить выводимые сообщения и др. Установка Orca.exe в папку %ProgramFiles%\Orca осуществляется запуском файла Orca.msi.

Запуск программы осуществляется через **Пуск | Программы | Orca**. После успешного запуска приложения нужно открыть MSI-файл, который необходимо модифицировать. В данном случае в качестве опытного образца используется дистрибутив ACDSee 10 — файл ACDSee 10 Photo Manager.msi, находящийся в каталоге C:\ACDSee.

Дистрибутив ACDSee 10 — это EXE-файл, представляющий собой самораспаковывающийся архив, который при запуске распаковывает MSI-файл в папке C:\Documents and Settings\%username%\Local Settings\Application Data\Downloaded Installations\{0051E50E-CECD-4E04-A71A-B8C49733542A}\ ACDSee 10 Photo Manager.msi. Именно этот файл необходимо открыть утилитой Orca.exe.

Любой MSI-файл содержит в себе базу данных. Графической оболочкой, с помощью которой можно получить к ней доступ, является утилита Orca.exe (рис. 18.2). В левой колонке приведен список доступных таблиц, в правой части — колонки, заголовки которых соответствуют полям выбранной таблицы, а в строках расположены значения этих полей.

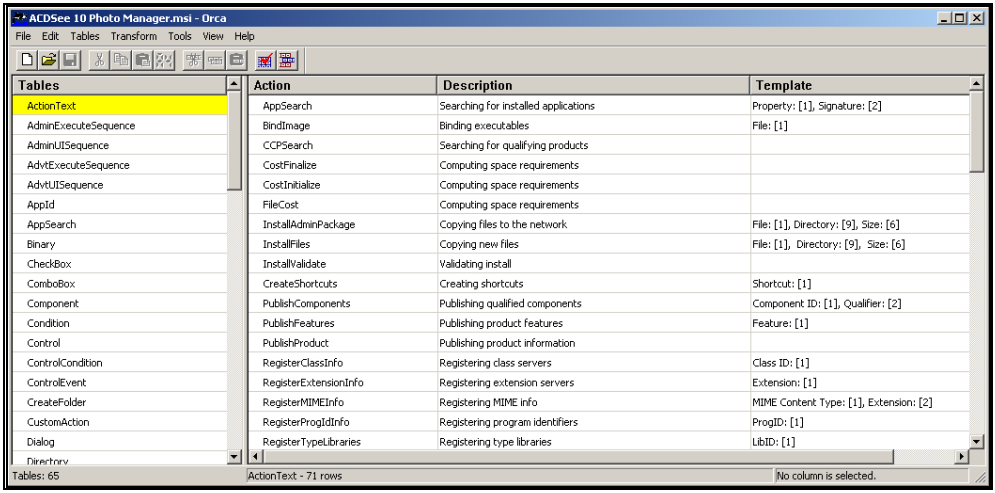



**Рис. 18.2.** Внешний вид утилиты Orca.exe

# Структура MSI-файлов

Множество таблиц, присутствующих в базе данных, можно объединить в группы. Выделим самые важные из них:

- базовые таблицы;
- файловые таблицы;
- таблицы информации о программе.

## Базовые таблицы

К базовым таблицам относят таблицы, описывающие параметры работы инсталлятора и устанавливаемые компоненты. Группа состоит из 11 связанных таблиц, краткое описание которых приведено в табл. 18.2.

***Таблица 18.2.*** *Базовые таблицы MSI Installer*

| Имя таблицы | Краткое описание |
|---|---|
| Feature | Содержит список всех функций программного продукта |
| Condition | Содержит описание условий, определяющих порядок установки каждой функции, описанной в таблице |
| FeatureComponents | Связывает функции с компонентами |
| Component | Содержит список всех компонентов приложения |
| Directory | Содержит список всех каталогов, необходимых для инсталляции |
| PublishComponent | Содержит список функций и компонентов, публикуемых для использования в других приложениях |
| Assembly | Задает установки для сборок .NET Framework CLR и Win32 |
| AssemblyName | Задает схему для именования сборок .NET Framework CLR и Win32 |
| Complus | Содержит информацию, необходимую для установки приложений COM+ |

***Таблица 18.2*** *(окончание)*

| Имя таблицы | Краткое описание |
|---|---|
| IsolatedComponent | Связывает компонент, заданный в столбце Component_Application (обычно это файл с расширением EXE), с компонентом, заданным в столбце Component_Shared (обычно это динамическая библиотека DLL) |
| Upgrade | Содержит информацию для значительных обновлений программного продукта |

## Файловые таблицы

Эта группа таблиц содержит информацию обо всех файлах, входящих в комплект поставки программного продукта. Бóльшая часть этих файлов перечислена в таблице File. Хотя таблица Directory не входит в эту группу, она тесно связана с ней, так как отражает структуру каталогов приложения. В табл. 18.3 приведен список файловых таблиц и их краткое описание.

***Таблица 18.3.*** *Файловые таблицы MSI Installer*

| Имя таблицы | Краткое описание |
|---|---|
| File | Представляет собой список файлов, которые будут распакованы в ходе инсталляции. Эта таблица связана с таблицей Directory, в которой зафиксирована структура каталогов устанавливаемого приложения, и с таблицей Component, с помощью которой обеспечивается связь с компонентами |
| RemoveFile | Содержит список файлов, которые необходимо удалить при выполнении операции RemoveFiles |
| Font | Включает в себя список шрифтов, которые необходимо зарегистрировать в операционной системе |
| SelfReg | Содержит список саморегистрирующихся модулей. Windows Installer не регистрирует EXE-файлы |
| Media | Описывает набор инсталляционных дисков |
| BindImage | Содержит информацию о привязках исполняемых файлов или DLL |

***Таблица 18.3*** *(окончание)*

| Имя таблицы | Краткое описание |
|---|---|
| MoveFile | Содержит список файлов, которые необходимо перенести во время инсталляции из исходного каталога в заданный каталог |
| DuplicateFile | Содержит список файлов, дублируемых либо в другой каталог с тем же именем, что и исходный файл, либо в тот же каталог, но с другим именем |
| Environment | Используется для задания переменных окружения |
| Icon | Хранит файлы иконок. Каждая иконка из этой таблицы во время инсталляции копируется в отдельный файл на диске |
| MsiFileHash | Эта таблица хранит 128-разрядное хэш-значение для исходных файлов в пакете инсталляции |

## Таблицы информации о программе

Таблицы этой группы содержат важную информацию о пакете инсталляции, используемую на протяжении всего процесса инсталляции. В табл. 18.4 приведена информация о назначении таблиц.

***Таблица 18.4.*** *Файловые таблицы MSI Installer*

| Имя таблицы | Краткое описание |
|---|---|
| Property | Хранит все свойства пакета инсталляции |
| Binary | Содержит двоичные данные для иконок, растров и т. п. Также здесь хранятся данные для пользовательских операций |
| Error | Используется для поиска шаблонов форматирования при обработке ошибок. Windows Installer имеет свой собственный механизм обработки ошибок |
| Shortcut | Здесь хранится вся информация, необходимая для создания файловых ярлыков |
| ReserveCost | Эта таблица содержит информацию о необходимом дисковом пространстве для каждого компонента приложения |

# Доступ к MSI-файлам программным способом

Для обеспечения программного управления базой данных, хранящейся в MSI-файле, используется шаблон, приведенный в листинге 18.1. Необходимо отметить, что шаблон чтения данных отличается от шаблона коррекции данных. О нем будет рассказано отдельно.

**Листинг 18.1. Шаблон доступа к базе в MSI-файле**

```
Set obj = CreateObject("WindowsInstaller.Installer")
Set base = obj.OpenDatabase(MSI, Mode)
base.OpenView(SQL-query).Execute
base.Commit
```

В приведенном шаблоне в первой строке осуществляется подключение к COM-объекту, с помощью которого осуществляется доступ к базе данных.

В следующей строке осуществляется подключение к базе данных с помощью метода `OpenDatabase()`, который имеет два аргумента. Первый из них — имя базы, которым является полный путь к MSI-файлу. Второй — режим доступа к базе данных:

- 0 — только чтение;
- 1 — чтение/запись.

В третьей строке реализовано выполнение задаваемого запроса. SQL-запрос задается с помощью метода `OpenView()`, а выполняется с помощью метода `Execute()`. Напомним, что в SQL-запросе регистр имен полей и таблиц имеет значение.

В последней строке осуществляется запись сделанных изменений в базе с помощью команды `Commit`.

## Работа с таблицами в Orca.exe

Рассмотрим подробнее структуру таблиц, содержащихся в MSI-файле. Orca.exe представляет собой простой редактор таблиц, хранящихся в дистрибутиве.

С полями таблиц можно осуществлять следующие действия:

- читать существующие поля;
- создавать новые поля;
- удалять поля;
- изменять поля.

## Чтение существующих полей

Для просмотра всех записей таблицы используется шаблон (листинг 18.2), отличающийся от приведенного в листинге 18.1. Это объясняется тем, что для этой задачи нужно перебрать все строки таблицы.

Отличия от листинга 18.1 начинаются с третьей строки. Здесь методы `OpenView()` и `Execute()` разносятся на две строки.

Далее осуществляется последовательное чтение строк в цикле. Обратите внимание на то, что в запросе `SELECT` поля могут быть перечислены только по именам. Значение `*` в данном случае не работает. Данные считываются в память методом `Fetch()`, и для дальнейшего обращения к ним надо использовать свойство `StringData`, передавая ему номер поля. Нумерация полей начинается с единицы.

В листинге 18.2 приведен пример чтения полей Property и Value таблицы Property файла ACDSee 10 Photo Manager.msi.

**Листинг 18.2. Чтение полей в таблице**

```
Set obj = CreateObject("WindowsInstaller.Installer")
Set ob = obj.OpenDatabase ("C:\ACDSee\ACDSee 10 Photo Manager.msi", 0)
Set View=ob.OpenView("SELECT Property, Value FROM Property")
View.Execute
Do
    Set Record = View.Fetch
    If Record Is Nothing Then Exit Do
    Wscript.Echo Record.StringData(1) + vbTab + Record.StringData(2)
Loop
Set View = Nothing
```

## Создание новых строк в таблице

Новые строки требуется создавать достаточно часто, например, чтобы записать в дистрибутив программы ее серийный номер. Для этого необходимо добавить в таблицу Property строку, содержащую PIDKEY в поле Property и с серийный номер продукта в поле Value. Чтобы добавить новую строку, необходимо установить курсор на нужную таблицу (рис. 18.3) и, нажав в произвольной ее части правую кнопку мыши, выбрать пункт **Add Row** или нажать комбинацию клавиш <Ctrl>+<R>.

В появившемся диалоговом окне будет приведен список доступных полей. Необходимые значения можно присвоить в этом же окне.

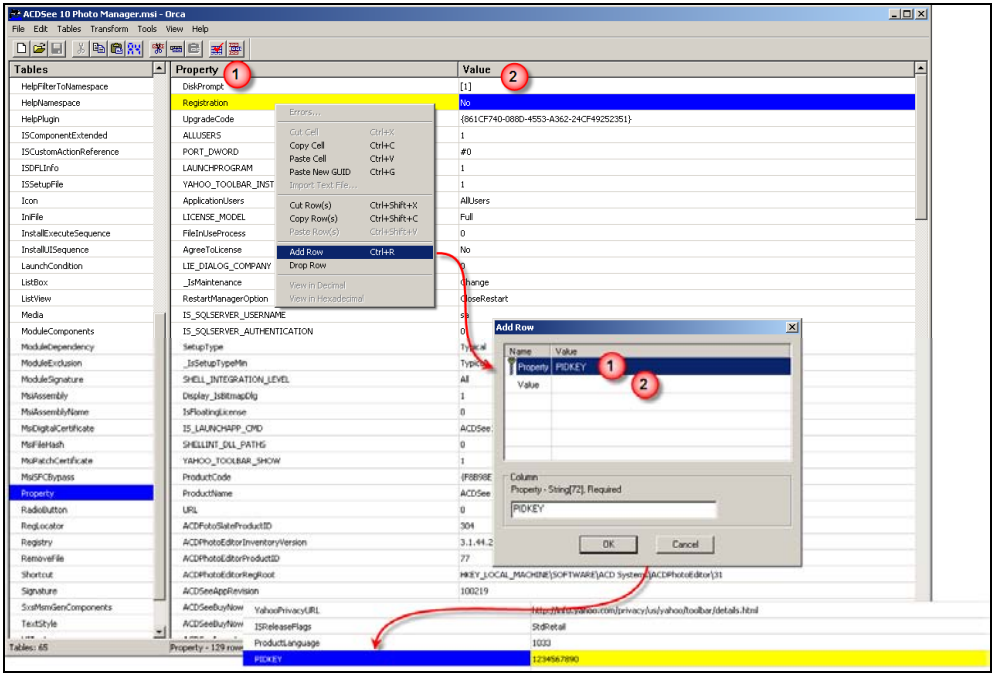



**Рис. 18.3.** Создание новых полей в таблице

Аналогичную операцию можно выполнить программным способом. Для этого необходимо создать объект `WindowsInstaller.Installer`. На основе шаблона, приведенного в листинге 18.1, сформируем программу, приведенную в листинге 18.3. В качестве второго параметра метода `OpenDatabase()` необходимо указать 1, поскольку выполняется операция записи данных.

Самым сложным в данной задаче будет правильно сформировать запрос. Особенность запроса `INSERT INFO` заключается в том, что внутри первых по счету скобок, где указываются названия полей, все поля должны быть заданы с именем таблицы в качестве префикса. Все имена параметров в запросе указываются без кавычек; все присваиваемые значения — в кавычках (апострофах) и, наконец, названия полей и таблиц чувствительны к регистру.

**Листинг 18.3. Создание новой строки в таблице**

```
Set a = CreateObject("WindowsInstaller.Installer")
Set b = a.OpenDatabase ("C:\acdsee\ACDSee 10 Photo Manager.msi", 1)
b.OpenView("INSERT INTO Property (Property.Property, Property.Value)" & _
           " VALUES ('PIDKEY', '1234567890')").Execute
b.Commit
```

## Удаление строк из таблицы

Удаление какой-либо строки осуществляется в соответствии со следующим алгоритмом:

1. Установить курсор на таблицу, в которой необходимо удалить строку.
2. Установить курсор на удаляемую строку в таблице.
3. Нажать кнопку <Del> на клавиатуре или открыть правой кнопкой мыши контекстное меню и выбрать в нем пункт **Delete**.
4. Подтвердить намерение удалить параметр.

Эту же задачу можно решить программным способом (листинг 18.4), используя шаблон листинга 18.1. Для обеспечения возможности записи в таблице необходимо установить второй параметр метода `OpenDatabase` равным единице (1).

**Листинг 18.4. Удаление существующего атрибута в таблице**

```
Set a = CreateObject("WindowsInstaller.Installer")
Set b = a.OpenDatabase ("C:\acdsee\ACDSee 10 Photo Manager.msi", 1)
b.OpenView("DELETE FROM Property WHERE 'PIDKEY'='1234567890'").Execute
b.Commit
```

## Изменение значений существующих полей

Для изменения значения нужно сделать двойной щелчок левой кнопкой мыши по необходимой ячейке (рис. 18.4).

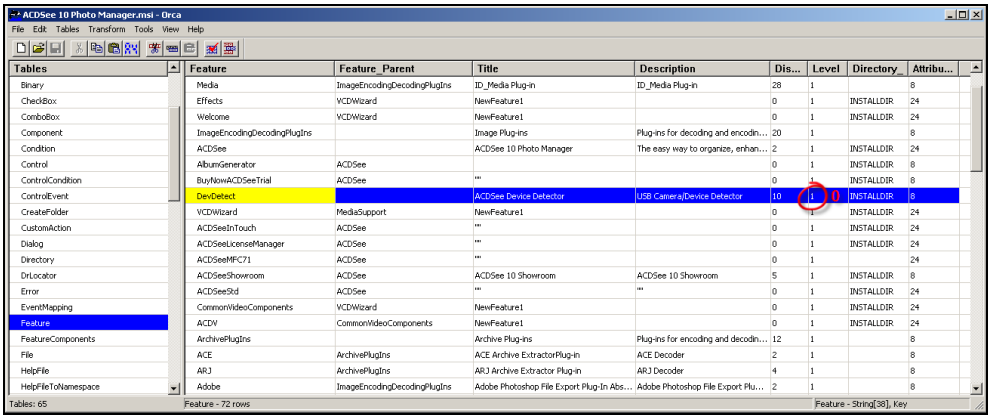



**Рис. 18.4.** Изменение параметра

Программное изменение свойства осуществляется по тем же правилам. По сравнению с предыдущими двумя случаями изменяется только запрос. В листинге 18.5 осуществляется отключение компонента Device Detector.

**Листинг 18.5. Изменение значений существующих полей таблицы**

```
Set a = CreateObject("WindowsInstaller.Installer")
Set b = a.OpenDatabase ("C:\acdsee\ACDSee 10 Photo Manager.msi", 1)
b.OpenView("UPDATE Feature SET Level='0' " & _
           "WHERE Feature='DevDetect'").Execute
b.Commit
```



# Приложения

## Приложение 1



# Таблица ASCII

Набор символов ASCII (American Standard Code for Information Interchange) — стандартная кодировка первых 128 символов. Последующие 128 символов содержат символы национальных алфавитов, и их кодировка зависит от кодовой страницы; кроме того, сейчас все большее распространение приобретает кодировка Unicode, в которой возможно одновременное кодирование символов различных алфавитов.

В табл. П1.1 приведены символы (Char) и их коды в десятичной (Dec) и шестнадцатеричной (Hex) записи.

***Таблица П1.1.*** *Кодировка символов ASCII*

| Dec | Hex | Char | Dec | Hex | Char | Dec | Hex | Char | Dec | Hex | Char |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | NUL | 32 | 20 | пробел | 64 | 40 | @ | 96 | 60 | ` |
| 1 | 1 | SOH | 33 | 21 | ! | 65 | 41 | A | 97 | 61 | a |
| 2 | 2 | STX | 34 | 22 | " | 66 | 42 | B | 98 | 62 | b |
| 3 | 3 | ETX | 35 | 23 | # | 67 | 43 | C | 99 | 63 | c |
| 4 | 4 | EOT | 36 | 24 | $ | 68 | 44 | D | 100 | 64 | d |
| 5 | 5 | ENQ | 37 | 25 | % | 69 | 45 | E | 101 | 65 | e |
| 6 | 6 | ACK | 38 | 26 | & | 70 | 46 | F | 102 | 66 | f |
| 7 | 7 | BEL | 39 | 27 | ' | 71 | 47 | G | 103 | 67 | g |
| 8 | 8 | BS | 40 | 28 | ( | 72 | 48 | H | 104 | 68 | h |
| 9 | 9 | TAB | 41 | 29 | ) | 73 | 49 | I | 105 | 69 | i |
| 10 | A | LF | 42 | 2A | * | 74 | 4A | J | 106 | 6A | j |
| 11 | B | VT | 43 | 2B | + | 75 | 4B | K | 107 | 6B | k |

***Таблица П1.1*** *(окончание)*

| Dec | Hex | Char | Dec | Hex | Char | Dec | Hex | Char | Dec | Hex | Char |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 12 | C | FF | 44 | 2C | , | 76 | 4C | L | 108 | 6C | l |
| 13 | D | CR | 45 | 2D | - | 77 | 4D | M | 109 | 6D | m |
| 14 | E | SO | 46 | 2E | . | 78 | 4E | N | 110 | 6E | n |
| 15 | F | SI | 47 | 2F | / | 79 | 4F | O | 111 | 6F | o |
| 16 | 10 | DLE | 48 | 30 | 0 | 80 | 50 | P | 112 | 70 | p |
| 17 | 11 | DC1 | 49 | 31 | 1 | 81 | 51 | Q | 113 | 71 | q |
| 18 | 12 | DC2 | 50 | 32 | 2 | 82 | 52 | R | 114 | 72 | r |
| 19 | 13 | DC3 | 51 | 33 | 3 | 83 | 53 | S | 115 | 73 | s |
| 20 | 14 | DC4 | 52 | 34 | 4 | 84 | 54 | T | 116 | 74 | t |
| 21 | 15 | NAK | 53 | 35 | 5 | 85 | 55 | U | 117 | 75 | u |
| 22 | 16 | SYN | 54 | 36 | 6 | 86 | 56 | V | 118 | 76 | v |
| 23 | 17 | ETB | 55 | 37 | 7 | 87 | 57 | W | 119 | 77 | w |
| 24 | 18 | CAN | 56 | 38 | 8 | 88 | 58 | X | 120 | 78 | x |
| 25 | 19 | EM | 57 | 39 | 9 | 89 | 59 | Y | 121 | 79 | y |
| 26 | 1A | SUB | 58 | 3A | : | 90 | 5A | Z | 122 | 7A | z |
| 27 | 1B | ESC | 59 | 3B | ; | 91 | 5B | [ | 123 | 7B | { |
| 28 | 1C | FS | 60 | 3C | < | 92 | 5C | \ | 124 | 7C | \| |
| 29 | 1D | GS | 61 | 3D | = | 93 | 5D | ] | 125 | 7D | } |
| 30 | 1E | RS | 62 | 3E | > | 94 | 5E | ^ | 126 | 7E | ~ |
| 31 | 1F | US | 63 | 3F | ? | 95 | 5F | _ | 127 | 7F | DEL |

# Назначение служебных символов

Как видно из табл. П1.1, некоторые символы не являются печатными знаками, а представлены двух- или трехбуквенными кодами. Это так называемые служебные символы. Их можно разбить на группы по назначению.

## Форматирование

Ряд символов предназначен для управления форматированием выводимого текста:

- BS или Backspace (возврат на один символ) — указывает на движение механизма печати или курсора дисплея назад на одну позицию;
- HT или Horizontal Tabulation (горизонтальное табулирование) — указывает на движение механизма печати или курсора дисплея до следующей предписанной позиции табуляции;
- LF или Line Feed (перевод строки) — указывает на движение механизма печати или курсора дисплея к началу следующей строки (на одну строку вниз);
- VT или Vertical Tabulation (вертикальное табулирование) — указывает на движение механизма печати или курсора дисплея к следующей группе строк;
- FF или Form Feed (перевод страницы) — указывает на движение механизма печати или курсора дисплея к исходной позиции следующей страницы, формы или экрана;
- CR или Carriage Return (перевод каретки) — указывает на движение механизма печати или курсора дисплея к исходной (крайней левой) позиции текущей строки.

## Передача данных

Некоторые служебные символы служат для управления передачей данных:

- SOH или Start of Heading (начало заголовка) — используется для указания начала заголовка, который может содержать информацию о маршрутизации или адрес;
- STX или Start of Text (начало текста) — указывает на начало текста и одновременно на конец заголовка;
- ETX или End of Text (конец текста) — используется при завершении текста, который был начат с символа STX;
- ENQ или Enquiry (запрос) — отмечает запрос идентификационных данных от удаленной станции;

- ACK или Acknowledge (подтверждение) — передается приемным устройством отправителю в качестве подтверждения успешного приема данных;
- NAK или Negative Acknowledgement (неподтверждение) — передается приемным устройством отправителю в случае отрицания (неудачи) приема данных;
- SYN или Synchronous/Idle (синхронизация) — используется в синхронизированных системах передачи. В моменты отсутствия передачи данных система непрерывно посылает символы SYN для обеспечения синхронизации;
- ETB или End of Transmission Block (конец блока передачи) — указывает на конец блока данных для коммуникационных целей. Используется для разбиения на отдельные блоки больших объемов данных.

## Разделительные знаки при передаче информации

Четыре символа используются как разделители:

- FS или File Separator (разделитель файлов);
- GS или Group Separator (разделитель групп);
- RS или Record Separtator (разделитель записей);
- US или Unit Separator (разделитель элементов).

## Другие символы

Остальные символы трудно отнести к какой-то группе. Опишем их назначение:

- NUL или Null (No character, нет данных) — используется для передачи в случае отсутствия данных; кроме того, часто обозначает конец строки;
- BEL или Bell (звонок) — используется для управления устройствами сигнализации;
- SO или Shift Out — указывает, что все последующие кодовые комбинации должны интерпретироваться согласно внешнему набору символов до прихода символа SI;
- SI или Shift In — указывает, что последующие кодовые комбинации должны интерпретироваться согласно стандартному набору символов;

- `DLE` или Data Link Escape (переключение) — служит для изменения значения идущих следом символов. Используется для дополнительного контроля или для передачи произвольной комбинации бит;
- `DC1`, `DC2`, `DC3`, `DC4` или Device Controls (контроль устройства) — предназначены для управления вспомогательными устройствами (специальными функциями);
- `CAN` или Cancel (отмена) — указывает, что данные, которые предшествовали этому символу в сообщении или блоке, должны игнорироваться (обычно в случае обнаружения ошибки);
- `EM` или End of Medium (конец носителя) — указывает на физический конец ленты или другого носителя информации;
- `SUB` или Substitute (заместитель) — используется для подмены ошибочного или недопустимого символа;
- `ESC` или Escape (расширение) — используется для расширения кода, указывая на то, что следующий символ имеет альтернативное значение;
- пробел — непечатаемый символ для разделения слов или перемещения механизма печати или курсора дисплея вперед на одну позицию;
- `DEL` или Delete (удаление) — используется для удаления (стирания) предыдущего знака в сообщении.

## ПРИЛОЖЕНИЕ 2



# Измерения в байтах

В табл. П2.1 приведены приставки, используемые для определения больших размеров информации. Использование двоичных приставок обусловлено двумя фактами: во-первых, в компьютерах используется двоичная система счисления, а во-вторых, $2^{10}$=1024, что очень близко к тысячи.

Следует заметить, что двоичные приставки, хотя и являются правильными, в обыденной речи часто заменяются соответствующими десятичными.

***Таблица П2.1.*** *Приставки, используемые при измерениях в байтах*

| **Десятичная приставка** | | | **Двоичная приставка** | | | |
|---|---|---|---|---|---|---|
| **Название** | **Символ** | **Степень** | **Название** | **Символ** | | **Степень** |
| | | | | **МЭК** | **ГОСТ** | |
| килобайт | KB | $10^3$ | кибибайт | KiB | Килобит | $2^{10}$ |
| мегабайт | MB | $10^6$ | мебибайт | MiB | Мегабит | $2^{20}$ |
| гигабайт | GB | $10^9$ | гибибайт | GiB | Гигабит | $2^{30}$ |
| терабайт | TB | $10^{12}$ | тебибайт | TiB | Терабит | $2^{40}$ |
| петабайт | PB | $10^{15}$ | пебибайт | PiB | | $2^{50}$ |
| эксабайт | EB | $10^{18}$ | эксбибайт | EiB | | $2^{60}$ |
| зеттабайт | ZB | $10^{21}$ | зебибайт | ZiB | | $2^{70}$ |
| йоттабайт | YB | $10^{24}$ | йобибайт | YiB | | $2^{80}$ |

## Приложение 3



# Системы счисления

## Десятичная система счисления

В десятичной системе счисления основанием является число 10. Для задания значений используют цифры от 0 до 9. Чтобы получить из десятичной записи числа его значение, нужно умножить эти цифры справа налево на увеличивающуюся степень десяти $10^0, 10^1, 10^2, ..., 10^n$, и сложить полученные числа. Так, число 856 представлено как $8 \cdot 10^2 + 5 \cdot 10^1 + 6 \cdot 10^0 = 8 \cdot 100 + 5 \cdot 10 + 6 \cdot 1$.

Десятичная система счисления является удобной для использования в повседневной жизни, однако она малоприменима в компьютерных устройствах, в которых используются только цифры 0 (нет сигнала) и 1 (есть сигнал). В компьютерном мире обычно применяется двоичная система счисления.

## Двоичная система счисления

Основанием двоичной системы является число 2. Значение формируется с помощью двух цифр — нуля и единицы. Соответственно, запись 100110 в двоичной системе расшифровывается следующим образом:

$$1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 32 + 0 + 0 + 4 + 2 + 0 = 38$$

По такому же алгоритму расшифровываются и другие числа, записанные в двоичной системе.

Переведем число 50 из десятичной в двоичную систему счисления с помощью табл. П3.1. Найдем в ней наибольшее из чисел, меньших 50, — это число 32, которому соответствует пятый разряд. Сразу можно сказать, что двоичное число будет иметь шесть цифр, первая из которых 1. Определим остаток разности сравниваемого числа с найденным: 50 − 32 = 18. Поступаем

аналогично с этим остатком: ищем наибольшее из меньших его чисел — это 16. Оно соответствует четвертому разряду, значит, пятым символом двоичного числа также будет единица. Разность 18 — 16 = 2.

***Таблица П3.1.*** *Пересчет чисел из десятичной в двоичную систему счисления*

| **Разряд** | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Значение** | 1024 | 512 | 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| | $2^{10}$ | $2^{9}$ | $2^{8}$ | $2^{7}$ | $2^{6}$ | $2^{5}$ | $2^{4}$ | $2^{3}$ | $2^{1}$ | $2^{1}$ | $2^{0}$ |

Повторяя описанную последовательность действий, получаем, что ближайшее число — 2, соответствует первому разряду. Знак равенства сигнализирует об окончании процедуры перевода. В те разряды, которые не равны единице, записываются нули. Таким образом, числу 50 в десятичной системе соответствует число 110010 в двоичной системе счисления.

# Шестнадцатеричная система счисления

Преобразовывать числа в двоичном формате в десятичную систему и обратно — задача довольно сложная, поскольку даже относительно небольшие числа имеют длинную запись в двоичной системе счисления. Гораздо удобнее использовать шестнадцатеричную систему, которая имеет знаменатель 16. Для обозначения числа в этой системе счисления используются цифры от 0 до 9 и буквы от A до F (также служащие "цифрами"). Иногда используют строчные буквы, которые имеют те же значения, что и соответствующие прописные. В табл. П3.2 приведены эквиваленты в шестнадцатеричной, десятичной и двоичной системах.

***Таблица П3.2.*** *Соотношения между системами счисления*

| Двоичная система | Шестнадцатеричная система | Десятичная система |
|---|---|---|
| 0000 | 0 | 0 |
| 0001 | 1 | 1 |
| 0010 | 2 | 2 |
| 0011 | 3 | 3 |

***Таблица П3.2*** *(окончание)*

| Двоичная система | Шестнадцатеричная система | Десятичная система |
|---|---|---|
| 0100 | 4 | 4 |
| 0101 | 5 | 5 |
| 0110 | 6 | 6 |
| 0111 | 7 | 7 |
| 1000 | 8 | 8 |
| 1001 | 9 | 9 |
| 1010 | A | 10 |
| 1011 | B | 11 |
| 1100 | C | 12 |
| 1101 | D | 13 |
| 1110 | E | 14 |
| 1111 | F | 15 |

## Приложение 4



# Переменные среды

При программном управлении компьютером большое значение имеют переменные среды, которые позволяют узнать ряд текущих настроек. Список доступных переменных среды приведен в табл. П4.1.

***Таблица П4.1.*** *Переменные среды*

| Переменная | Тип | Описание |
|---|---|---|
| %ALLUSERSPROFILE% | Локальная | Размещение профиля All Users |
| %APPDATA% | Локальная | Используемое по умолчанию размещение, которое применяется пользовательскими приложениями данных |
| %CD% | Локальная | Путь к текущей папке |
| %CMDCMDLINE% | Локальная | Командная строка, с помощью которой был запущен данный экземпляр Cmd.exe |
| %CMDEXTVERSION% | Системная | Номер версии текущих расширений обработчика команд |
| %COMPUTERNAME% | Системная | Имя компьютера |
| %COMSPEC% | Системная | Путь к исполняемой командной оболочке |
| %DATE% | Системная | Текущая дата. Использует тот же формат, что и команда `date /t`. Создается командным интерпретатором Cmd.exe |
| %ERRORLEVEL% | Системная | Код ошибки последней выполненной команды. Значение, не равное нулю, обычно указывает на наличие ошибки |

***Таблица П4.1*** *(продолжение)*

| Переменная | Тип | Описание |
|---|---|---|
| %HOMEDRIVE% | Системная | Имя диска локальной рабочей станции, связанного с основным каталогом пользователя. Задается на основании расположения основного каталога. Основной каталог пользователя указывается в оснастке **Локальные пользователи и группы** |
| %HOMEPATH% | Системная | Полный путь к основному каталогу пользователя. Задается на основании расположения основного каталога. Основной каталог пользователя указывается в оснастке **Локальные пользователи и группы** |
| %HOMESHARE% | Системная | Сетевой путь к общему основному каталогу пользователя. Задается на основании расположения основного каталога. Основной каталог пользователя указывается в оснастке **Локальные пользователи и группы** |
| %LOGONSEVER% | Локальная | Имя контроллера домена, который проверял подлинность текущей сессии |
| %NUMBER_OF_ PROCESSORS% | Системная | Количество процессоров, установленных на компьютере |
| %OS% | Системная | Имя операционной системы. При использовании Windows 2000 и XP имя операционной системы отображается как Windows_NT |
| %PATH% | Системная | Путь поиска для исполняемых файлов |
| %PATHEXT% | Системная | Список расширений файлов, которые рассматриваются операционной системой как исполняемые |
| %PROCESSOR_ARCHITECT URE% | Системная | Архитектура процессора. Возможные значения: x86, IA64 |

***Таблица П4.1*** *(окончание)*

| Переменная | Тип | Описание |
|---|---|---|
| %PROCESSOR_IDENTFIER% | Системная | Описание процессора |
| %PROCESSOR_LEVEL% | Системная | Номер модели процессора, установленного на компьютере |
| %PROCESSOR_REVISION% | Системная | Номер модификации процессора |
| %PROMPT% | Локальная | Параметры командной строки для текущего интерпретатора. Создается командным интерпретатором Cmd.exe |
| %RANDOM% | Системная | Произвольное десятичное число от 0 до 32 767. Создается командным интерпретатором Cmd.exe |
| %SYSTEMDRIVE% | Системная | Имя диска, содержащего корневой каталог Windows XP (то есть системный каталог) |
| %SYSTEMROOT% | Системная | Размещение системного каталога Windows XP |
| %TEMP%<br>%TMP% | Системная и пользовательская | Временные папки, по умолчанию используемые приложениями, которые доступны пользователям, выполнившим вход в систему. Некоторые приложения используют переменную TEMP, другие — переменную TMP |
| %TIME% | Системная | Текущее время. Использует тот же формат, что и команда `time /t`. Создается командным интерпретатором Cmd.exe |
| %USERDOMAIN% | Локальная | Имя домена, содержащего список учетных записей пользователей |
| %USERNAME% | Локальная | Имя пользователя, выполнившего вход в систему |
| %USERPROFILE% | Локальная | Размещение профиля для текущего пользователя |
| %WINDIR% | Системная | Размещение каталога операционной системы |

## ПРИЛОЖЕНИЕ 5



# Буквенные сокращения стран

В табл. П5.1 приведены буквенные сокращения стран, используемые в Active Directory.

***Таблица П5.1.*** *Сокращенные и полные названия стран*

| Буквенное сокращение | Международное название | Русское название |
|---|---|---|
| AC | Ascension Island | остров Вознесения |
| AD | Andorra | Андорра |
| AE | United Arab Emirates | Объединенные Арабские Эмираты |
| AF | Afghanistan | Афганистан |
| AG | Antigua and Barbuda | Антигуа и Барбуда |
| AI | Anguilla | Ангилья |
| AL | Albania | Албания |
| AM | Armenia | Армения |
| AN | Netherlands Antilles | Голландские Антильские острова |
| AO | Angola | Ангола |
| AQ | Antarctica | Антарктика |
| AR | Argentina | Аргентина |
| AS | American Samoa | Американское Самоа |
| AT | Austria | Австрия |

***Таблица П5.1*** *(продолжение)*

| Буквенное сокращение | Международное название | Русское название |
|---|---|---|
| AU | Australia | Австралия |
| AW | Aruba | Аруба |
| AX | Aland Islands | Аландские острова |
| AZ | Azerbaijan | Азербайджан |
| BA | Bosnia and Herzegovina | Босния и Герцеговина |
| BB | Barbados | Барбадос |
| BD | Bangladesh | Бангладеш |
| BE | Belgium | Бельгия |
| BF | Burkina Faso | Буркина-Фасо |
| BG | Bulgaria | Болгария |
| BH | Bahrain | Бахрейн |
| BI | Burundi | Бурунди |
| BJ | Benin | Бенин |
| BM | Bermuda | Бермудские острова |
| BN | Brunei Darussalam | Бруней |
| BO | Bolivia | Боливия |
| BR | Brazil | Бразилия |
| BS | Bahamas | Багамские острова |
| BT | Bhutan | Бутан |
| BV | Bouvet Island | остров Буве |
| BW | Botswana | Ботсвана |
| BY | Belarus | Беларусь |
| BZ | Belize | Белиз |
| CA | Canada | Канада |

***Таблица П5.1*** *(продолжение)*

| Буквенное сокращение | Международное название | Русское название |
|---|---|---|
| CC | Cocos Keeling Islands | Кокосовые острова |
| CD | Congo | Конго |
| CF | Central African Republic | Центральноафриканская Республика |
| CG | Congo | Конго |
| CH | Switzerland | Швейцария |
| CI | Côte d'Ivoire | Кот-д'Ивуар |
| CK | Cook Islands | острова Кука |
| CL | Chile | Чили |
| CM | Cameroon | Камерун |
| CN | China | Китай |
| CO | Colombia | Колумбия |
| CR | Costa Rica | Коста-Рика |
| CS | Serbia and Montenegro | Сербия и Черногория |
| CU | Cuba | Куба |
| CV | Cape Verde | Кабо-Верде |
| CX | Christmas Island | остров Рождества |
| CY | Cyprus | Кипр |
| CZ | Czech Republic | Чехия |
| DE | Germany | Германия |
| DJ | Djibouti | Джибути |
| DK | Denmark | Дания |
| DM | Dominica | Доминика |
| DO | Dominican Republic | Доминиканская Республика |
| DZ | Algeria | Алжир |

***Таблица П5.1*** *(продолжение)*

| Буквенное сокращение | Международное название | Русское название |
|---|---|---|
| EC | Ecuador | Эквадор |
| EE | Estonia | Эстония |
| EG | Egypt | Египет |
| EH | Western Sahara | Западная Сахара |
| ER | Eritrea | Эритрея |
| ES | Spain | Испания |
| ET | Ethiopia | Эфиопия |
| FI | Finland | Финляндия |
| FJ | Fiji | Фиджи |
| FK | Falkland Islands | Фолклендские острова |
| FM | Micronesia | Микронезия |
| FO | Faroe Islands | Фарерские острова |
| FR | France | Франция |
| GA | Gabon | Габон |
| GB | United Kingdom | Соединенное Королевство Великобритании и Северной Ирландии |
| GD | Grenada | Гренада |
| GE | Georgia | Грузия |
| GF | French Guiana | Французская Гвиана |
| GG | Guernsey | остров Гернси |
| GH | Ghana | Гана |
| GI | Gibraltar | Гибралтар |
| GL | Greenland | Гренландия |
| GM | Gambia | Гамбия |

***Таблица П5.1*** *(продолжение)*

| Буквенное сокращение | Международное название | Русское название |
|---|---|---|
| GN | Guinea | Гвинея |
| GP | Guadeloupe | Гваделупа |
| GQ | Equatorial Guinea | Экваториальная Гвинея |
| GR | Greece | Греция |
| GS | South Georgia and the South Sandwich Islands | Южная Джорджия и Южные Сандвичевы острова |
| GT | Guatemala | Гватемала |
| GU | Guam | Гуам |
| GW | Guinea-Bissau | Гвинея-Бисау |
| GY | Guyana | Гайана |
| HK | Hong Kong | Гонконг |
| HM | Heard and McDonald Islands | Остров Херд и Острова Макдоналд |
| HN | Honduras | Гондурас |
| HR | Croatia / Hrvatska | Хорватия |
| HT | Haiti | Гаити |
| HU | Hungary | Венгрия |
| ID | Indonesia | Индонезия |
| IE | Ireland | Ирландия |
| IL | Israel | Израиль |
| IM | Isle of Man | остров Мэн |
| IN | India | Индия |
| IO | British Indian Ocean Territory | Британская территория в Индийском океане |
| IQ | Iraq | Ирак |
| IR | Iran | Иран |

***Таблица П5.1*** *(продолжение)*

| Буквенное сокращение | Международное название | Русское название |
|---|---|---|
| IS | Iceland | Исландия |
| IT | Italy | Италия |
| JE | Jersey | остров Джерси |
| JM | Jamaica | Ямайка |
| JO | Jordan | Иордания |
| JP | Japan | Япония |
| KE | Kenya | Кения |
| KG | Kyrgyzstan | Кыргызстан |
| KH | Cambodia | Камбоджа |
| KI | Kiribati | Кирибати |
| KM | Comoros | Коморские острова |
| KN | Saint Kitts and Nevis | Сент-Китс и Невис |
| KP | Korea | Северная Корея |
| KR | Korea | Южная Корея |
| KW | Kuwait | Кувейт |
| KY | Cayman Islands | Каймановы острова |
| KZ | Kazakhstan | Казахстан |
| LA | Lao | Лаос |
| LB | Lebanon | Ливан |
| LC | Saint Lucia | Сент-Люсия |
| LI | Liechtenstein | Лихтенштейн |
| LK | Sri Lanka | Шри-Ланка |
| LR | Liberia | Либерия |
| LS | Lesotho | Лесото |

***Таблица П5.1*** *(продолжение)*

| **Буквенное сокращение** | **Международное название** | **Русское название** |
|---|---|---|
| LT | Lithuania | Литва |
| LU | Luxembourg | Люксембург |
| LV | Latvia | Латвия |
| LY | Libya | Ливия |
| MA | Morocco | Марокко |
| MC | Monaco | Монако |
| MD | Moldova | Молдова |
| MG | Madagascar | Мадагаскар |
| MH | Marshall Islands | Маршалловы острова |
| MK | Macedonia | Македония |
| ML | Mali | Мали |
| MM | Myanmar | Мьянма |
| MN | Mongolia | Монголия |
| MO | Macau | Макао |
| MP | Northern Mariana Islands | Северные Марианские острова |
| MQ | Martinique | Мартиника |
| MR | Mauritania | Мавритания |
| MS | Montserrat | Монтсеррат |
| MT | Malta | Мальта |
| MU | Mauritius | Маврикий |
| MV | Maldives | Мальдивы |
| MW | Malawi | Малави |
| MX | Mexico | Мексика |
| MY | Malaysia | Малайзия |

***Таблица П5.1*** *(продолжение)*

| Буквенное сокращение | Международное название | Русское название |
|---|---|---|
| MZ | Mozambique | Мозамбик |
| NA | Namibia | Намибия |
| NC | New Caledonia | Новая Каледония |
| NE | Niger | Нигер |
| NF | Norfolk Island | Норфолк |
| NG | Nigeria | Нигерия |
| NI | Nicaragua | Никарагуа |
| NL | Netherlands | Нидерланды |
| NO | Norway | Норвегия |
| NP | Nepal | Непал |
| NR | Nauru | Науру |
| NU | Niue | Ниуэ |
| NZ | New Zealand | Новая Зеландия |
| OM | Oman | Оман |
| PA | Panama | Панама |
| PE | Peru | Перу |
| PF | French Polynesia | Французская Полинезия |
| PG | Papua New Guinea | Папуа-Новая Гвинея |
| PH | Philippines | Филиппины |
| PK | Pakistan | Пакистан |
| PL | Poland | Польша |
| PM | Saint Pierre and Miquelon | Сен-Пьер и Микелон |
| PN | Pitcairn Island | остров Питкэрн |
| PR | Puerto Rico | Пуэрто-Рико |

***Таблица П5.1*** *(продолжение)*

| Буквенное сокращение | Международное название | Русское название |
|---|---|---|
| PS | Palestine | Палестина |
| PT | Portugal | Португалия |
| PW | Palau | Палау |
| PY | Paraguay | Парагвай |
| QA | Qatar | Катар |
| RE | Reunion Island | остров Реюньон |
| RO | Romania | Румыния |
| RU | Russian Federation | Россия |
| RW | Rwanda | Руанда |
| SA | Saudi Arabia | Саудовская Аравия |
| SB | Solomon Islands | Соломоновы острова |
| SC | Seychelles | Сейшельские острова |
| SD | Sudan | Судан |
| SE | Sweden | Швеция |
| SG | Singapore | Сингапур |
| SH | Saint Helena | остров Святой Елены |
| SI | Slovenia | Словения |
| SJ | Svalbard and Jan Mayen Islands | острова Свалбард и Ян Мейен |
| SK | Slovakia | Словакия |
| SL | Sierra Leone | Сьерра-Леоне |
| SM | San Marino | Сан-Марино |
| SN | Senegal | Сенегал |
| SO | Somalia | Сомали |
| SR | Suriname | Суринам |

***Таблица П5.1*** *(продолжение)*

| Буквенное сокращение | Международное название | Русское название |
|---|---|---|
| ST | Sao Tome and Principe | Сан-Томе и Принсипи |
| SU | exUSSR | СССР |
| SV | El Salvador | Сальвадор |
| SY | Syria | Сирия |
| SZ | Swaziland | Свазиленд |
| TC | Turks and Caicos Islands | острова Теркс и Кайкос |
| TD | Chad | Чад |
| TF | French Southern Territories | Французские Южные Территории |
| TG | Togo | Того |
| TH | Thailand | Таиланд |
| TJ | Tajikistan | Таджикистан |
| TK | Tokelau | Токелау |
| TL | Timor-Leste | Тимор Лесте |
| TM | Turkmenistan | Туркменистан |
| TN | Tunisia | Тунис |
| TO | Tonga | Тонга |
| TP | East Timor | Восточный Тимор |
| TR | Turkey | Турция |
| TT | Trinidad and Tobago | Тринидад и Тобаго |
| TV | Tuvalu | Тувалу |
| TW | Taiwan | Тайвань |
| TZ | Tanzania | Танзания |
| UA | Ukraine | Украина |
| UG | Uganda | Уганда |

***Таблица П5.1*** *(окончание)*

| Буквенное сокращение | Международное название | Русское название |
|---|---|---|
| UK | United Kingdom | Соединенное Королевство Великобритании и Северной Ирландии |
| UM | United States Minor Outlying Islands | Малые тихоокеанские отдаленные острова |
| US | United States | США |
| UY | Uruguay | Уругвай |
| UZ | Uzbekistan | Узбекистан |
| VA | Holy See, Vatican | Ватикан |
| VC | Saint Vincent and the Grenadines | Сент-Винсент и Гренадины |
| VE | Venezuela | Венесуэла |
| VG | Virgin Islands, British | Виргинские острова, Британские |
| VI | Virgin Islands, US | Виргинские острова, США |
| VN | Vietnam | Вьетнам |
| VU | Vanuatu | Вануату |
| WF | Wallis and Futuna Islands | Острова Уоллис и Футуна |
| WS | Western Samoa | Западное Самоа |
| YE | Yemen | Йемен |
| YT | Mayotte | Майотта |
| YU | Yugoslavia | Югославия |
| ZA | South Africa | ЮАР |
| ZM | Zambia | Замбия |
| ZW | Zimbabwe | Зимбабве |

## Приложение 6



# Интегрирование собственных шаблонов в PrimalScript

Одной из важнейших особенностей PrimalScript является поддержка собственной библиотеки шаблонов, которую читатель может пополнить самостоятельно. Эту же библиотеку можно использовать и для хранения собственных сценариев.

Создадим шаблон на основе сценария, приведенного в листинге П6.1. В нем производится чтение всех учетных записей пользователей из Active Directory.

**Листинг П6.1. Чтение данных из Active Directory с помощью ADODB-соединения**

```
domain = "LDAP://" + _
         GetObject("LDAP://RootDSE").Get("defaultNamingContext")
Set objConnection = CreateObject("ADODB.Connection")
Set objCommand = CreateObject("ADODB.Command")
objConnection.CommandTimeout = 120
objConnection.Provider = "ADsDSOObject"
objConnection.Properties("ADSI Flag") = 1+2
objConnection.Properties("User ID") = "msk\adminstrator"
objConnection.Properties("Password") = "password"
objConnection.Properties("Encrypt Password") = True
objConnection.Open "Active Directory Provider"
Set objCommand.ActiveConnection = objConnection
objCommand.Properties("Page size") = 10000
objCommand.Properties("Timeout") = 300
objCommand.Properties("Cache Results") = False
```

```
Set st = objConnection.Execute( _
          "SELECT sAMAccountName, description FROM ' " & Domain & _
          " ' WHERE objectClass='person')

st.MoveFirst
Temp = ""
Do Until st.EOF
    SamAccountName = ""
    SamAccountName = St.Fields("sAMAccountName").Value

    Description = ""
    A_Description = St.Fields("description").Value
    For Each AA in A_Description
        Description = Description + AA
    Next

    Temp = Temp + "Имя: "+ SamAccountName + _
           "Описание: " + Description + chr(13) + chr(10)
    st.MoveNext
Loop

Wscript.Echo Temp
```

Несмотря на то, что, изучая приведенный сценарий, можно увидеть различные шаблоны, например, шаблон определения длинного имени домена, соединения с Active Directory, SQL-запрос для обращения к объекту и т. д., ограничимся созданием одного шаблона — определения имени текущего домена. Для определения этого имени используется виртуальный объект RootDSE (листинг П6.2).

**Листинг П6.2. Шаблон соединения с Active Directory**

```
Set obj = GetObject("LDAP://RootDSE")
Domain = obj.Get("defaultNamingContext")
```

Шаблон представляет собой текстовый файл с расширением snippet, расположенный в папке C:\Program Files\SAPIEN\PrimalScript Professional\Snippets,

если программа установлена в директорию по умолчанию. Иерархическую структуру расположения шаблонов и описания функций различных языков определяет структура каталогов и файлы с расширением snippet (рис. П6.1).

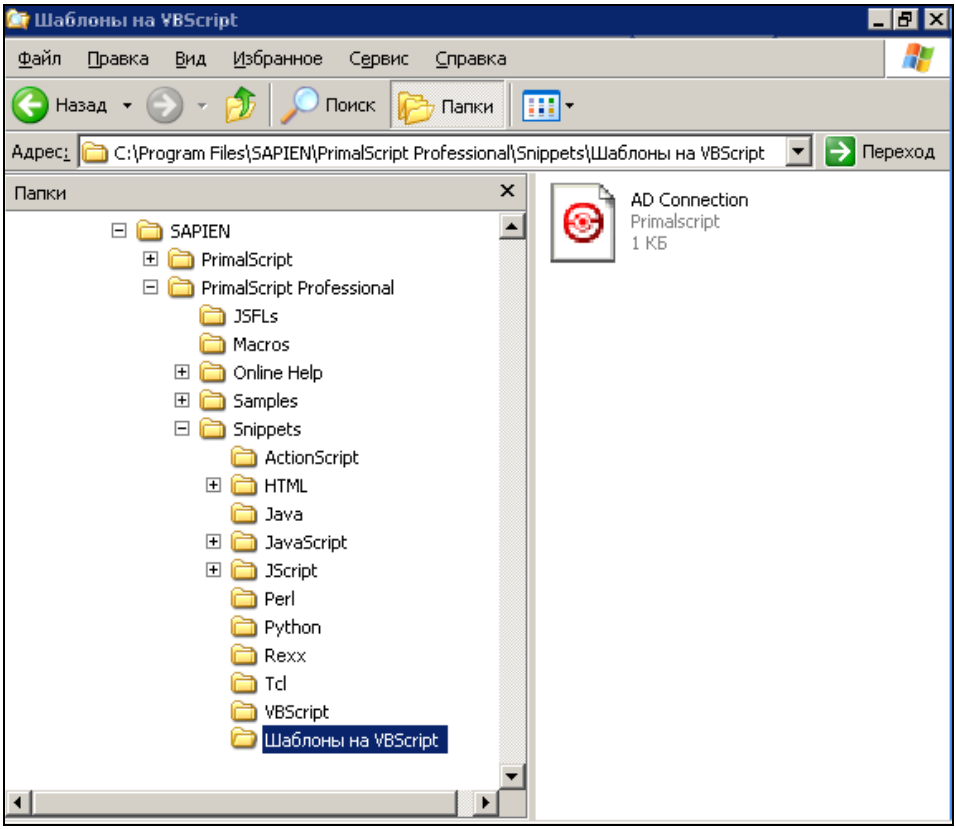



**Рис. П6.1.** Расположение шаблонов в файловой системе

Создадим в каталоге Snippets подкаталог Шаблоны на VBScript, а в нем текстовый файл AD Connection.snippet. Все его содержимое представляет собой шаблон. После создания файла и заполнения его содержимым необходимо перезапустить PrimalScript, чтобы внесенные в файловую структуру изменения вступили в силу.

В редакторе вызовите окно **Nexus Windows**, в котором находится список шаблонов.

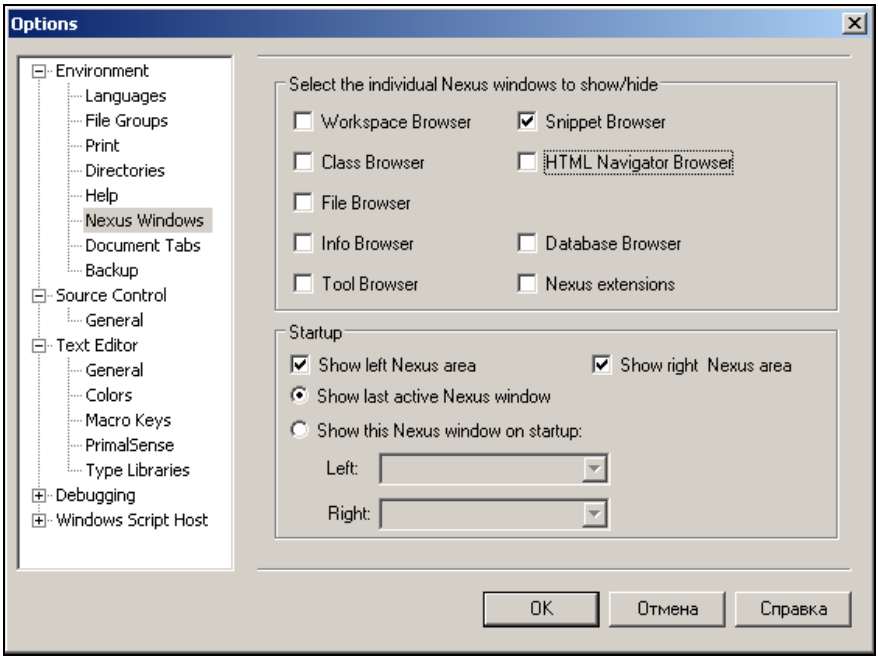



**Рис. П6.2.** Настройка свойств Primal Script 4.0

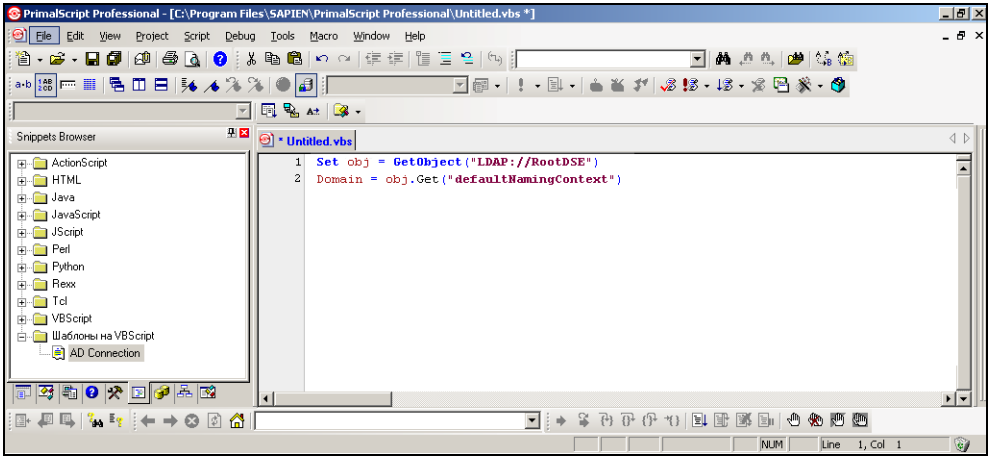



**Рис. П6.3.** Пример использования созданного шаблона

Для этого необходимо в меню **Tools** выбрать пункт **Options**, а в появившемся окне войти в **Environment | Nexus Windows** и сделать активным меню

**Snippet Browser**, как это показано на рис. П6.2. Чтобы сделанные изменения вступили в силу, необходимо перезапустить редактор.

В навигаторе редактора **Snippets Browser** (рис. П6.3) помимо 10 встроенных подразделов появится еще один — **Шаблоны на VBScript**, в котором будет единственный шаблон — **AD Connection**. Для его использования следует создать файл с расширением VBS и два раза щелкнуть левой кнопкой мыши по имени шаблона. Шаблон из файла AD Connection.snippet будет вставлен в сценарий.

# Ответственность

В книге описан очень мощный инструментарий по управлению серверами и настройке программного обеспечения. Все скрипты перед введением в эксплуатацию настоятельно рекомендуется многократно и всесторонне протестировать в отдельной зоне, не влияющей на работу основных серверов. Перед внедрением скриптов в эксплуатацию следует делать резервные копии. В неумелых руках сценарии могут привести к необратимым последствиям.

# Литература


1. Аллен Р. Active Directory. Сборник рецептов для Windows Server 2003 и Windows 2000. Для профессионалов. — СПб.: Питер, 2004.
2. Аллен Р. Рецепты администрирования Windows Server 2000/2003. — СПб.: Питер, 2006.
3. Борн Г. Руководство разработчика Microsoft Windows на Script Host 2.0. — СПб.: Питер, 2001.
4. Джонс Д. Средства автоматизации в Microsoft Windows. — М.: Эком, М.: Бином, 2006.
5. Коробко И. Администрирование сетей Windows с помощью сценариев. — СПб.: БХВ-Петербург, 2007.
6. Попов А. Командные файлы и сценарии Windows Script Host. — СПб.: БХВ-Петербург, 2002.
7. Торрес Д. Скрипты для администратора Windows. — СПб.: Питер, 2002.
8. Уилсон Э. Сценарии Microsoft Windows. — М.: Эком; М.: Бином, 2006.
9. Хонейкатт Д. Реестр Windows XP. Справочник профессионала. — М.: Эком, 2006.
10. Экк Т. Сценарии ADSI для системного администрирования Windows NT/2000. — М.: Вильямс, 2000.

# Предметный указатель


## A

ABE 355, 388
Access-based Enumerator 388
ACE 355
ACL 355
Active Directory 19
  архитектура 24
  атрибуты, общие для всех объектов 193
  доступ к объектам 59
  идентификаторы 56
  компоненты 26
  объекты 32
  основные объекты 194
  схема 20
  типы данных 33, 75
  типы объектов 57
  шаблон доступа с помощью ADO 136
Active Directory Explorer 182
Active Directory Viewer 179
ActiveX Data Object 135
ADM-файл 405, 420
ADO 55, 135
ADODB 31
  Command 146
  Connection 137
  Recordset 155
ADSI 26, 55
  доступ к службам 28
  объектная модель 31
  провайдеры 29
ARC-путь 465, 474
Autorun.inf 489

## B

BAT-файл *См. Пакетный файл*
Boot.ini 472
  параметры запуска ОС 475
BSoD 464

## C

Cmd.exe 493
CMD-файл *См. Пакетный файл*
CN 23, 57
Cscript.exe 422

## D

DC 23, 57
Dcpromo.exe 36, 42
DEP 464
DNS 25, 35
DSA 26

## E

ESE 26

# F

FAT 356
FQDN 25, 171
Fully Qualified Domain Name 25, 171

# G

GC 26
Generalized Time 79
GetSID.exe 346
Globally Unique Identifier *См. GUID*
GPO 289
Group Policy Object 289
Group Policy Template 289
Group scope 271
GT 79
GUID 191, 337
  версия 339
  создание 341
  структура 338
  формирование 340

# H

HAL 464

# I

IADs 93
  методы 98
  свойства 94
IADs*, семейство интерфейсов 31, 55, 90
  классификация 93
  объекты 127
  свойства объектов 113
  типы данных 91
IADsAccessControlEntry 381
IADsAccessControlList 373
  методы 374
  свойства 380
IADsComputer 128
IADsContainer 105
  методы 106
  свойства 106
IADsDomain 128
IADsGroup 128
IADsNamespaces 111
IADsOpenDSObject 112
IADsOu 128
IADsPropertyEntry 120
IADsPropertyList 114
IADsPropertyValue 124
IADsPropertyValue2 124
IADsSecurityDescriptor 370
IADsSecurityUtility 365
  методы 366
  свойства 369
IADsUser 128

# L

LDAP 26, 27, 55, 62, 90, 171
LDAP Browser 183
Ldp.exe 178
LNK 436
Login 214
Logon Script 214

# M

MAPI 27
Master File Table 359
MBR 464

MFT 359
Msiexec.exe 504
MSI-файл 502
  добавление данных в таблицы 513
  изменение 515
  программный доступ 511
  структура 508
  удаление строк из таблиц 514
  чтение таблиц 512

## N

NetBIOS 25
Ntds.dit 23, 40
Ntdsutil.exe 24
NTFS 356
Ntldr 464, 466

## O

OID 171
Orca.exe 506
Organizational Unit 289
OU 23, 57

## P

Profile 214

## R

RDN 171
Regobj.dll 403
REG-файл 400
Relative Distinguished Name 171
REPL 27
RFC 20
RootDSE 35, 171
  атрибуты 172
  получение сведений 178
  программное управление 188

## S

Safe Mode *См. Безопасный режим*
SAM 25, 27
Security group 271
SID 191, 337
  преобразование 344
  структура 342
  чтение из AD 343
  широко известные 348
SQL 55, 66, 135

## T

TCP/IP 35

## U

UNC 23, 56, 171
UPN 23
URL 436
Uuidgen.exe 341

## V

VBScript:
  CreateObject() 65

GetObject() 61
VarType() 75

## W

Well-known SID 348
Windows Installer 502
Windows Installer SDK 506
WScript 426
  методы 430
  свойства 426
WSH 424
  ошибки сценариев 461
WshArguments 433
WshNetwork 455
WshShell 402, 433

## A

Атрибут Active Directory:
  accountExpires 240
  altServer 175
  c 235, 298
  cn 231, 274, 282, 318
  co 235, 298
  codePage 193
  company 249
  configurationNamingContext 173
  countryCode 193
  createTimeStamp 193
  currentTime 173
  defaultNamingContext 172
  department 249
  description 193, 232, 298, 319
  directReports 251
  displayName 231
  distinguishedName 193, 227, 273
  dnsHostName 175
  domainControllerFunctionality 175
  domainFunctionality 174
  dsServiceName 175
  facsimileTelephoneNumber 246
  forestFunctionality 174
  givenName 231
  gPLink 302
  gPOptions 303
  groupType 274, 283
  homeDirectory 245
  homeDrive 245
  homePhone 247
  info 247, 284
  instanceType 193
  ipPhone 246
  isCriticalSystemName 319
  isGlobalCatalogReady 175
  isSynchronized 175
  l 235, 298
  ldapServiceName 175
  location 325
  logonHours 238
  mail 232
  managedBy 326
  manager 249, 287, 299
  member 285
  memberOf 252, 286, 322
  mobile 245
  modifyTimeStamp 193
  msNPAllowDialin 255, 328
  msNPCallingStationID 256, 329
  msNPSavesCallingStationID 258, 331
  msRADIUSCallbackNum 256, 329

msRADIUSFramedIP 257, 330
msRADIUSFramedRoute 257, 330
msRADIUSServiceType 256, 329
msRASSavesCallbackNum 258, 331
msRASSavesFramedIP 258, 331
msRASSavesFramedRoute 258, 331
msTSAllowLogon 270
msTSBrokenConnectionAction 264
msTSConnectClientDrives 261
msTSConnectClientPrinterDrives 261
msTSDefaultToMainPrinter 261
msTSHomeDirectory 269
msTSHomeDrive 269
msTSInitialProgram 260
msTSMaxConnectionTime 264
msTSMaxDisconnectionTime 263
msTSMaxIdleTime 264
msTSProfilePath 269
msTSReconnectionAction 264
msTSRemoteControl 265
msTSWorkDirectory 260
name 193, 231, 274, 282, 298, 318
namingContexts 172
objectCategory 193
objectClass 194
objectGUID 194, 338
objectSID 337, 342
operatingSystem 320
operatingSystemServicePack 321
operatingSystemVersion 320
otherFacsimileTelephoneNumber 246
otherHomePhone 247
otherIpPhone 246
otherMobile 245
otherPager 245
otherTelephone 232
ou 293, 298
pager 245
physicalDeliveryOfficeName 232
postalCode 235, 298
postOfficeBox 234
primaryGroupID 253, 322
profilePath 244
pwdLastSet 219, 240
rootDomainNamingContext 173
sAMAccountName 217, 238, 274, 282, 318
schemaNamingContext 173
scriptPath 244
serverName 175
setPassword 218
sn 232
st 235, 298
streetAddress 234, 298
subSchemaSubEntry 194
subSchemaSubEntry 173
supportedCapabilities 174
supportedControl 174
supportedExtension 174
supportedLDAPPolicies 174
supportedLDAPVersion 174
supportedSASLMechanisms 174
telephoneNumber 232
title 249
url 233
userAccountControl 219, 235, 240, 323
userPrincipalName 217, 227, 238
userWorkstations 239
uSNChanged 194
uSNCreated 194
whenChanged 194
whenCreated 194
wWWHomePage 232

## Б

База данных:
иерархическая 22
реляционная 22
Безопасный режим 465, 469

## В

Вывод на экран 442

# Г

Главная файловая таблица 359
Глобальный каталог 19, 27
Группа безопасности *См. Учетная запись группы*
Группа рассылки 271
Групповая политика 404
  CATEGORY 407
  CLASS 407
  POLICY 409

# Д

Дерево 19
Диск, сетевой 460
Домен 19, 35
Доступ к объектам AD 59

# З

Запрос к службе каталогов 62
  LDAP 62
  SQL 66
Запуск компьютера 465
Запуск приложений из сценария 444
Значок 490

# И

Именование 23
Интерфейсы доступа к Active Directory 93
Интерфейсы IADs*

# К

Компьютер:
  варианты загрузки 468
  запуск 465
Контейнер *См. Учетная запись контейнера*
Контроллер домена 20, 35
Куст 397

# Л

Лес 19

# М

Мастер установки Active Directory 36
Метафайлы NTFS 359

# О

Объект:
  создание 431
  удаление 433
Объектная модель 90
Относительное составное имя 171
Отношения, доверительные 19

# П

Пакетный файл 493
  call 495
  echo 496

endlocal 496
for 496
goto 497
if 498
pause 500
rem 500
setlocal 499
shift 501
Папка *См. Учетная запись контейнера*
Пароль 218, 222, 240
Переменные окружения 436
Подключение сетевых дисков 460
Пользователь *См. Учетная запись пользователя*
Прилинкованный SQL-сервер 69
Приложение:
активация 435
запуск 444
Принтер 456
Провайдеры ADSI 29
Пространство имен 20
Протоколирование 441


## Р


Реестр 397
кусты 398
перечисление ключей 403
программное управление 402
редактор 401
типы данных 399
Репликация 26


## С


Сеанс подключения к источнику данных 137
Сетевой диск 460
Сетевые ресурсы, управление 455
Синий экран 464
Скрипт 11
Специальная папка 421, 448
Список управления доступом 355
Сценарий 421
аргументы 427


## Т


Тип группы 271
Тип данных 75
булев 87
дата-время 79
длинное целое 77
массив 88
объект 85
строка 82


## У


Управляющий объект 175
Установка Active Directory 36
Учетная запись группы 204, 272
свойства 281
создание, программное 277
создание, с помощью мастера 272
удаление 279
Учетная запись компьютера 208, 304
свойства 316
создание, программное 312
создание, с помощью мастера 304
удаление 315
Учетная запись контейнера 207, 289
свойства 296
создание, программное 293
создание, с помощью мастера 289
удаление 294

Учетная запись пользователя 195, 214
настройки доступа к сети 235
свойства 227
создание, программное 224
создание, с помощью мастера 215
удаление 227
удаленный доступ 253
членство в группах 251

## Ф

Файл подкачки 355, 486
Файловая система 36, 355
FAT 356
FAT32 357
HPFS 357
NTFS 358
VFAT 357
права доступа 364
Формат имен 23

## Э

Эмуляция нажатия клавиш 446

## Я

Ядро 467
Ярлык 421
к сетевому ресурсу 454
к файлу 450
свойства 453
создание 451
удаление 453