В.А.Капустин

Введение в применение объекта FileSystemObject и справочное руководство по библиотеке времени исполнения Scripting.

Санкт-Петербург
2003

Содержание

ИНФОРМАЦИЯ О ПРИМЕНЕНИИ **5**

ИСПОЛЬЗОВАНИЕ ОБЪЕКТНОЙ МОДЕЛИ FILESYSTEMOBJECT В ПРИЛОЖЕНИЯХ MICROSOFT OFFICE **6**

ОБЪЕКТЫ FILESYSTEMOBJECT **6**

ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ ОБЪЕКТА FILESYSTEMOBJECT **8**

Создание объекта FileSystemObject 8
Использование подходящего метода 8
Доступ к существующим дискам, файлам и каталогам 8
Доступ к свойствам объектов 9
Работа с дисками и каталогами 9
Получение информации о дисках *9*
Пример использования объекта Drive 10
Работа с каталогами *10*
Работа с файлами *11*
Создание файлов 11
Добавление данных в файл. 12
Чтение файлов 13
Перемещение, копирование и удаление файлов 14
Пример кода, использующего FileSystemObject *15*

СПРАВОЧНАЯ ИНФОРМАЦИЯ ОБ ОБЪЕКТНОЙ МОДЕЛИ FILESYSTEMOBJECT **30**

Классы, входящие в объектную модель FileSystemObject 31
Объект FileSystemObject 32
Свойства и методы объекта FileSystemObject *33*
Свойства объекта FileSystemObject 33
Свойство Drives 33
Методы объекта FileSystemObject 34
Метод BuildPath 36
Метод CopyFile 37
Метод CopyFolder 39
Метод CreateFolder 41
Метод CreateTextFile 42
Метод DeleteFile 43
Метод DeleteFolder 44
Метод DriveExists 45
Метод FileExists 45
Метод FolderExists 46
Метод GetAbsolutePathName 47
Метод GetBaseName 48
Метод GetDrive 49
Метод GetDriveName 50
Метод GetExtensionName 50
Метод GetFile 51
Метод GetFileName 51
Метод GetFileVersion 52

Метод GetFolder 52
Метод GetParentFolderName 53
Метод GetSpecialFolder 53
Метод GetStandardStream 54
Метод GetTempName 55
Метод MoveFile 56
Метод MoveFolder 57
Метод OpenTextFile 58
Объект Drive 60
Свойства и методы объекта Drive *61*
Свойства объекта Drive 61
Свойство AvailableSpace 62
Свойство DriveLetter 62
Свойство DriveType 63
Свойство FileSystem 64
Свойство FreeSpace 64
Свойство IsReady 65
Свойство Path 66
Свойство RootFolder 66
Свойство SerialNumber 67
Свойство ShareName 68
Свойство TotalSize 68
Свойство VolumeName 69
Коллекция Drives 70
Свойства и методы коллекции Drives *70*
Объекты Folder и File 70
Объект Folder *70*
Объект File *71*
Свойства и методы объектов Folder и File *71*
Одноименные свойства объектов Folder и File 71
Свойства объекта Folder 72
Описания одноименных свойств объектов Folder и File 73
Свойство Attributes 73
Свойство DateCreated 74
Свойство DateLastAccessed 75
Свойство DateLastModified 75
Свойство Drive 76
Свойство Name 76
Свойство ParentFolder 77
Свойство ShortName 78
Свойство ShortPath 79
Свойство Size 79
Свойство Type 80
Описания свойств объекта Folder 80
Свойство Files 80
Свойство IsRootFolder 81
Свойство SubFolders 81
Одноименные методы объектов Folder и File 82

Описания одноименных методов объектов Folder и File 82
Метод Copy 82
Метод Delete 83
Метод Move 84
Метод CreateTextFile объекта Folder 85
Метод OpenAsTextStream объекта File 86
КОЛЛЕКЦИЯ FOLDERS 88
Свойства и методы коллекции Folders *88*
Метод Add 88
КОЛЛЕКЦИЯ FILES 89
Свойства и методы коллекции Files *89*
ОБЪЕКТ TEXTSTREAM 90
Свойства и методы объекта TextStream *90*
Свойства объекта TextStream 90
Свойство AtEndOfLine 91
Свойство AtEndOfStream 91
Свойство Column 92
Свойство Line 92
Методы объекта TextStream 93
Метод Close 93
Метод Read 94
Метод ReadAll 94
Метод ReadLine 95
Метод Skip 95
Метод SkipLine 96
Метод Write 97
Метод WriteBlankLines 98
Метод WriteLine 98

Информация о применении

Модель объекта **FileSystemObject** (FSO) позволяет использовать для работы с файлами и каталогами знакомый многим синтаксис object.method для богатого набора свойств и методов. Этот объектно-ориентированный инструмент может быть использован Всюду, где применим Visual Basic или Visual Basic for Application:

- HTML для создания веб-страниц;
- Windows Scripting Host для создания командных файлов для Microsoft Windows;
- Элементом управления Script Control для поддержки скриптов в различных приложениях, разрабатываемых на различных языках программирования.
- В приложениях Microsoft Office.

Использование FSO в программах-клиентах (веб-страницах) создает серьезные проблемы, связанные с безопасностью, обеспечивая нежелательный доступ к локальной файловой системе на компьютере пользователя. Поэтому настоящая документация предполагает, что для создания веб-страниц объектная модель FSO будет использоваться на стороне сервера. Такой подход означает, что установки безопасности Internet Explorer, принятые по умолчанию, не позволяют использовать объект **FileSystemObject** на стороне пользователя. Изменение этих установок безопасности может подвергнуть компьютер пользователя угрозе нежелательного доступа к файловой системе, которая может привести к полному разрушению целостности файловой системы, вызвав тем самым потери данных или даже что-нибудь похуже.

Модель объекта FSO позволяет серверным приложениям и приложениям Microsoft Office создавать, изменять, перемещать и удалять каталоги; определять, существует ли тот или иной каталог, а если существует, то где. Можно также получать такую информацию о каталогах, как их имена, даты создания или последней модификации и т.д.

Модель объекта FSO также упрощает обработку файлов. Главная задача обработки файлов – хранение данных в таком формате, который был бы компактен, не требовал бы значительных системных ресурсов и обеспечивал бы удобный доступ к данным. Файлы необходимо создавать, сохранять в них данные и изменять эти данные, а также читать их. Поскольку хранение данных в базе данных, например, в Access или SQL-сервере, требует значительных накладных расходов, хранение данных в двоичном или текстовом файле может оказаться более эффективным решением. Возможно, вы предпочтете обойтись без дополнительных накладных расходов, а возможно, требования по доступу к вашим данным не требуют применения тех возможностей, которые обеспечивает развитая система управления базами данных.

Модель объекта FSO, которая содержится в библиотеке типов Scripting (Scrrun.dll), поддерживает создание текстовых файлов и манипуляцию ими с помощью объекта TextStream. Хотя пока модель объекта FSO не поддерживает создание или манипуляции двоичными файлами, такая поддержка планируется компанией Microsoft.

Использование объектной модели FileSystemObject в приложениях Microsoft Office

Для использования объекта FileSystemObject в приложениях Microsoft Office рекомендуется включить в проект ссылку на библиотеку Scrrun.dll (меню Tools|References) – см. стр. 30.

Включение этой библиотеки в проект позволяет использовать любой из стилей программирования с применением FileSystemObject:

Небрежный

Типы описываемых переменных не указываются:

```
Dim f, d, s
```

В результате все описываемые переменные приобретают тип Variant, подтипом которого может стать Object. Переменные этого типа могут содержать ссылки на объект любого типа, в частности, на объекты, входящие в модель FileSystemObject.

Квазистрогий

Все переменные, хранящие ссылки на объекты, описываются как имеющие тип Object:

```
Dim f As Object, d As Object, s As String
```

Строгий

Все переменные, хранящие ссылки на объекты, описываются как имеющие тип того объекта, на который указывает ссылка:

```
Dim f As FileSystemObject, d As Drive, s As String
```

Первые два стиля возможны и без включения в проект ссылки на библиотеку Scrrun.dll. Эти стили программирования позволяют писать программы быстро, но увеличивают вероятность появления ошибок ("небрежный" стиль программирования приводит к значительному количеству ошибок, поскольку при таком программировании VBA/VB вообще не в состоянии выполнять статический контроль типов).

Строгий стиль программирования возможен только при включении в проект ссылки на библиотеку Scrrun.dll. Этот стиль программирования позволяет VBA/VB выполнять полный контроль типов переменных в процессе компиляции программы и поддерживать сам процесс программирования, демонстрируя программисту перечни свойств и методов объектов и параметры методов.

Документация компании Microsoft, поставляемая в составе MS Office Developer Edition, в примерах кода использует небрежный стиль программирования и содержит ошибки.

Большой пример кода, приводимый в настоящем документе (см. стр. 15–29), представляет собой переработанный (из небрежного) в квазистрогий стиль пример из документации MS Office Developer Edition.

Небольшие фрагменты кода, сопровождающие приводимые в настоящем документе описания отдельных свойств и методов модели FileSystemObject, выполнены в строгом стиле программирования.

Объекты FileSystemObject

Модель объекта FileSystemObject (FSO) содержит следующие объекты и коллекции:

FileSystemObject

Главный объект. Содержит методы и свойства, которые позволяют создавать, уничтожать, получать информацию о, и выполнять произвольные манипуляции с логическими дисками, каталогами и файлами. Многие из этих методов, связанные с объектом FSO, дублируются в других (под-)объектах FSO – для удобства.

Формальное описание FileSystemObject приведено на стр. 32.

Drive

Объект. Содержит методы и свойства, которые позволяют получать информацию о доступных логических дисках, такую, как, например, имя разделяемого каталога на сервере и объем свободного места. Заметьте, что "диск" – это не обязательно жесткий диск; это может быть привод CD-ROM, виртуальный диск (в оперативной памяти) и т.д. Диск может не быть физически подключен к данному компьютеру, он может быть присоединен логически – через сеть.

Формальное описание приведено на стр. 60.

Drives

Коллекция. Обеспечивает доступ к списку дисков, доступных в системе физически или логически. Коллекция Drives включает все диски, независимо от их типа. Дисководы со съемными носителями представлены в этой коллекции даже в том случае, если в них не вставлены носители.

Формальное описание приведено на стр. 70.

File

Объект. Содержит методы и свойства, которые позволяют создавать, удалять или перемещать файл. С помощью методов этого объекта можно также запросить у системы имя файла, путь к нему и информацию о других свойствах файла.

Формальное описание приведено на стр. 71.

Files

Коллекция. Список всех файлов в некотором каталоге.

Формальное описание приведено на стр. 89.

Folder

Объект. Содержит методы и свойства, которые позволяют создавать, удалять или перемещать каталоги. С помощью методов этого объекта можно также запросить у системы имя каталога, путь к нему и информацию о других свойствах каталога.

Формальное описание приведено на стр. 70.

Folders

Коллекция. Список всех каталогов в некотором каталоге.

Формальное описание приведено на стр. 88.

TextStream

Объект. Позволяет читать и писать текстовые файлы.

Формальное описание приведено на стр. 90.

Программирование с использованием объекта FileSystemObject

Чтобы писать программы с применением объектной модели FileSystemObject (FSO):

- Используйте метод CreateObject для создания объекта FileSystemObject.
- Используйте подходящие для вашей задачи методы вновь созданного объекта.
- Используйте свойства этого объекта.

Объектная модель FSO содержится в библиотеке типов Scripting (которая расположена в файле Scrrun.dll). Таким образом, для использования FSO вам необходимо иметь этот файл в надлежащем месте на диске (%SystemRoot%\SYSTEM32).

Создание объекта FileSystemObject

Прежде всего, создайте FileSystemObject, используя метод CreateObject:

```
Dim fs As Object
Set fs = CreateObject("Scripting.FileSystemObject")

Dim fso As FileSystemObject
Set fso = CreateObject("Scripting.FileSystemObject")
```

В приведенном примере Scripting – это имя библиотеки типа, а FileSystemObject – имя создаваемого объекта. Объект FileSystemObject всегда создается в единственном экземпляре, независимо от числа ваших попыток создать другие экземпляры этого объекта.

Использование подходящего метода

Создав FileSystemObject, используйте его подходящий метод. Например, для создания нового файла используйте CreateTextFile (стр. 42), а для создания каталога – CreateFolder (стр. 41). (Объектная модель FSO не поддерживает создание или уничтожение логических дисков).

Для удаления объектов используйте методы DeleteFile (для удаления файла – стр. 43) или DeleteFolder (для удаления каталога – стр. 44) объекта FileSystemObject или метод Delete (стр. 83) соответствующих объектов (File или Folder). С помощью подходящих методов можно также копировать файлы и каталоги.

> **Замечание**. Некоторая функциональность в объектной модели **FileSystemObject** избыточна. Например, можно скопировать файл, используя или метод **CopyFile** объекта **FileSystemObject**, или метод **Copy** объекта **File**. Оба метода работают абсолютно одинаково, оба они существуют для гибкости программирования.

Доступ к существующим дискам, файлам и каталогам

Чтобы получить доступ к существующему диску, файлу или каталогу, используйте соответствующий метод "get" объекта FileSystemObject:

- GetDrive (стр. 49)
- GetFolder (стр. 52)
- GetFile (стр. 51)

Например, для получения доступа к существующему файлу служит следующий код:

```
Dim fso As FileSystemObject, f1 As File
Set fso = CreateObject("Scripting.FileSystemObject")
Set f1 = fso.GetFile("c:\test.txt")
```

Не используйте методы "get" для доступа к только что созданным объектам – методы "create" возвращают дескрипторы соответствующих объектов. Например, если вы создали новый каталог, используя метод CreateFolder, не используйте метод GetFolder для того, чтобы получить доступ к свойствам этого каталога (имени, пути и пр.). Вместо этого просто присвойте соответствующей переменной значение, возвращаемое методом CreateFolder, а затем используйте эту переменную для доступа к свойствам каталога:

```
Sub CreateFolder
  Dim fso As FileSystemObject, fldr As Folder
  Set fso = CreateObject("Scripting.FileSystemObject")
  Set fldr = fso.CreateFolder("C:\MyTest")
  Debug.Print "Created folder: " & fldr.Name
End Sub
```

Доступ к свойствам объектов

Как только вы получили дескриптор объекта, у вас появляется возможность доступа к свойствам этого объекта. Например, для получения имени соответствующего каталога, сначала создайте экземпляр объекта FSO, затем получите дескриптор с помощью подходящего метода (в случае каталога, GetFolder, поскольку каталог существует):

```
Set fldr = fso.GetFolder("c:\")
```

Теперь, когда у вас уже есть дескриптор объекта Folder, вы можете проверить его свойство Name:

```
Debug.Print "Каталог называется: " & fldr.Name
```

Для того, например, чтобы выяснить время последней модификации файла:

```
Dim fso As FileSystemObject, f1 As File
Set fso = CreateObject("Scripting.FileSystemObject")
' Получим объект File для дальнейшего исследования.
Set f1 = fso.GetFile("c:\detlog.txt")
' Выведем информацию.
Debug.Print "Время последнего измененеия файла: " & f1.DateLastModified
```

Работа с дисками и каталогами

Используя объектную модель FileSystemObject (FSO), можно программировать работу с дисками и каталогами в том же стиле, как если бы вы интерактивно выполняли те же действия в Проводнике. Можно копировать и перемещать каталоги, получать информацию о дисках и каталогах и т.д.

Получение информации о дисках

Объект Drive (стр. 60) позволяет получать информацию как о физических, так и о сетевых дисках, имеющихся в системе. Свойства этого объекта позволяют получать информацию о:

- Общем объеме диска в байтах (свойство TotalSize)
- Сколько места (в байтах) доступно на диске (свойства AvailableSpace или FreeSpace)
- Какая буква приписана диску (свойство DriveLetter)
- Каков тип диска (съемный, жесткий, CD-ROM или виртуальный – свойство DriveType)
- Серийный номер (свойство SerialNumber)
- Тип файловой системы – FAT, FAT32, NTFS и т.п. (свойство FileSystem)
- Готов ли диск к работе (свойство IsReady)
- Имя разделяемого каталога и/или тома (свойства ShareName и VolumeName)
- Путь или корневой каталог диска (свойства Path или RootFolder)

Пример использования объекта Drive

Используем объект Drive для сбора информации о диске. В приведенном ниже коде объект Drive в явном виде не присутствует; вместо этого использован метод GetDrive для получения ссылки на существующий объект Drive (в приведенном случае, drv):

```
Sub ShowDriveInfo(drvPath)
  Dim fso As FileSystemObject, drv As Drive, s As String
  Set fso = CreateObject("Scripting.FileSystemObject")
  Set drv = fso.GetDrive(fso.GetDriveName(drvPath))
  s = "Диск " & UCase(drvPath) & " - "
  s = s & drv.VolumeName & vbCr
  s = s & "Общий объем: " & FormatNumber(drv.TotalSize / 1024, 0)
  s = s & " килобайт" & vbCr
  s = s & "Свободно: " & FormatNumber(drv.FreeSpace / 1024, 0)
  s = s & " килобайт"
  Debug.Print s
End Sub
```

Замечание. Для вставки в длинную строку кода перевода строки (chr(13)) использована константа VBA: **vbCr**.

Работа с каталогами

В приводимой таблице – сводка типичных задач работы с каталогами и применяемых для этого методов.

Задача	Метод
Создать каталог.	FileSystemObject.CreateFolder (стр. 41)
Удалить каталог.	Folder.Delete (стр. 83) или FileSystemObject.DeleteFolder (стр. 44)
Переместить/переименовать каталог.	Folder.Move (стр. 84) или FileSystemObject.MoveFolder (стр. 57)
Скопировать каталог.	Folder.Copy (стр. 82) или FileSystemObject.CopyFolder (стр. 39)
Получить имя каталога.	Folder.Name (стр. 76)
Проверить, существует ли задан-	FileSystemObject.FolderExists (стр. 46)

ный каталог на указанном диске.	
Получить экземпляр существующего объекта Folder.	FileSystemObject.GetFolder (стр. 52)
Получить имя родительского каталога.	FileSystemObject.GetParentFolderName (стр. 53)
Получить путь к системным каталогам.	FileSystemObject.GetSpecialFolder (стр. 53)

Приведенный ниже пример демонстрирует, как можно использовать объекты Folder и FileSystemObject для манипуляций с каталогами и получения информации о них.

```
Sub ShowFolderInfo()
  Dim fso As FileSystemObject, fldr As Folder, s As String
' Получить экземпляр FileSystemObject.
  Set fso = CreateObject("Scripting.FileSystemObject")
' Получить объект Drive (диск).
  Set fldr = fso.GetFolder("c:")
' Вывести имя родительского каталога.
  Debug.Print "Имя родительского каталога: " & fldr
' Вывести имя (букву) диска.
  Debug.Print "Contained on drive " & fldr.Drive & vbCr
' Вывести результат проверки, не является ли каталог корневым.
  If fldr.IsRootFolder = True Then
    Debug.Print "Это корневой каталог."
  Else
    Debug.Print "Это НЕ корневой каталог." & vbCr
  End If
' Создать новый каталог с помощью объекта FileSystemObject.
  fso.CreateFolder ("C:\Bogus.dir")
  Debug.Print "Создан каталог C:\Bogus.dir"
' Вывести имя каталога (без расширения, если таковое есть).
  Debug.Print "Собственно имя каталога: " _
    & fso.GetBaseName("c:\Bogus.dir")
' Удалить только что созданный каталог.
  fso.DeleteFolder ("C:\Bogus.dir")
  Debug.Print "Каталог C:\Bogus.dir удален"
End Sub
```

Работа с файлами

Существует два основных категории манипуляций с файлами:

- Создание файлов, добавление данных в них или удаление данных из них и чтение файлов
- Перемещение/переименование, копирование и удаление файлов

Создание файлов

Есть три способа создания пустого текстового файла (вместо слов "текстовый файл" мы будем использовать термин "текстовый поток" – text stream).

Первый способ заключается в использовании метода CreateTextFile (стр. 42):

```
Dim fso As FileSystemObject, f1 As File
Set fso = CreateObject("Scripting.FileSystemObject")
Set f1 = fso.CreateTextFile("c:\testfile.txt", True)
```

Пример использования метода CreateTextFile объекта FileSystemObject смотри в примере кода на стр. 26.

Второй способ создания текстового файла – использовать метод OpenTextFile (стр. 58) объекта FileSystemObject с установленным флагом ForWriting. Например:

```
Dim fso As FileSystemObject, ts As TextStream
' Const ForWriting = 2
Set fso = CreateObject("Scripting.FileSystemObject")
Set ts = fso.OpenTextFile("c:\test.txt", IOMode.ForWriting, True)
```

Замечание. Символическая константа ForWriting для использования в качестве флага становится доступной только при подключении к проекту библиотеки scrrun.dll. Поскольку эта символическая константа не определена в стандартных библиотеках VBA, то без подключенной библиотеки scrrun.dll приходится определять ее значение непосредственно в программе. Приведем здесь значения всех флагов, которые можно использовать в методе CreateTextFile объекта FileSystemObject.

- 1 – для чтения (ForReading)
- 2 – для записи (ForWriting)
- 8 – для дописывания в конец файла (ForAppending)

Третий способ создания текстового файла – использовать метод OpenAsTextStream (стр. 86) объекта File с установленным флагом ForWriting. Например:

```
Dim fso As FileSystemObject, fl As File, ts As TextStream
' Const ForWriting = 2
Set fso = CreateObject("Scripting.FileSystemObject")
Set f1 = fso.CreateTextFile ("c:\test1.txt")
Set ts = f1.OpenAsTextStream(IOMode.ForWriting, True)
```

Добавление данных в файл.

Как только вы создали текстовый файл, можно добавить в него данные, выполнив следующие три шага:

1. Открыть текстовый файл
2. Записать в него данные
3. Закрыть файл

Для открытия существующего файла используйте метод OpenTextFile объекта FileSystemObject или метод OpenAsTextStream объекта File.

Для записи данных в открытый текстовый файл, используйте методы Write, WriteLine, или WriteBlankLines объекта TextStream (стр. 90), в зависимости от требуемой задачи:

Задача	Метод
Вывести данные в открытый текстовый файл без символа перевода строки после выводимых данных	Write (стр. 97)
Вывести данные в открытый текстовый файл с символом перевода строки после выводимых данных	WriteLine (стр. 98)
Вывести одну или несколько пустых строк в открытый текстовый файл	WriteBlankLines (стр. 98)

Примеры того, как методы Write, WriteLine, и WriteBlankLines используются с объектом FileSystemObject смотри в примере кода на стр. 26.

Для закрытия открытого файла используется метод Close (стр. 93) объекта TextStream (см. также пример кода на стр. 26).

> **Замечание**. Символ перевода строки может соответствовать одному или двум символам (в зависимости от операционной системы), которые переводят курсор к началу следующей строки (возврат каретки/перевод строки). Имейте в виду, что в конце некоторых строк эти невидимые символы уже могут присутствовать.

Следующий пример демонстрирует, как открыть файл, использовать три метода для добавления данных в файл и затем закрыть файл:

```
Sub CreateFile()
  Dim fso As FileSystemObject, tf As TextStream
  Set fso = CreateObject("Scripting.FileSystemObject")
  Set tf = fso.CreateTextFile("c:\testfile.txt", True)
' Вывести строку с переводом строки в ее конце.
  tf.WriteLine("Testing 1, 2, 3.")
' Вывести в файл три пустые строки.
  tf.WriteBlankLines(3)
' Вывести строку.
  tf.Write ("This is a test.")
  tf.Close
End Sub
```

Чтение файлов

Для чтения данных из файла используйте методы Read, ReadLine, или ReadAll объекта TextStream (стр. 90). Нижеследующая таблица описывает, какой метод следует использовать для решения каких задач.

Задача	Метод
Прочесть заданное количество символов из файла	Read (стр. 94)
Прочесть целую строку (до, но не включая символ перехода на новую строку)	ReadLine (стр. 95)
Прочесть все содержимое текстового файла	ReadAll (стр. 94)

Примеры того, как методы ReadAll, и ReadLine используются с объектом FileSystemObject смотри в примере кода на стр. 27.

Если вы используете метод Read или метод ReadLine и хотите пропустить какую-то порцию данных, используйте один из методов Skip (стр. 95) или SkipLine (стр. 96). Текст, который возвращают методы Read или ReadLine, хранится в строке, которую можно использовать для вывода в элемент управления, для обработки с использованием функций (например, Left, Right и Mid), конкатенации и т.д.

Приводимый пример демонстрирует, как открыть файл, записать в него данные, а затем прочитать данные из этого файла:

```
Sub ReadFiles
  Dim fso As FileSystemObject, f1 As TextStream, _
      ts As TextStream, s As String
  ' Const ForReading = 1
  Set fso = CreateObject("Scripting.FileSystemObject")
  Set f1 = fso.CreateTextFile("c:\testfile.txt", True)
' Прочесть строку.
```

```
  Debug.Print "Пишу в файл"
  f1.WriteLine "Hello World"
  f1.WriteBlankLines(1)
  f1.Close
' Прочесть все содержимое файла.
  Debug.Print "Читаю из файла"
  Set ts = fso.OpenTextFile("c:\testfile.txt", IOMode.ForReading)
  s = ts.ReadLine
  Debug.Print "Файл содержит: '" & s & "'"
  ts.Close
End Sub
```

Перемещение, копирование и удаление файлов

Объектная модель FSO обладает двумя методами для перемещения, копирования и удаления файлов:

Задача	Метод
Переместить файл	File.Move (стр. 84) или FileSystemObject.MoveFile (стр. 56)
Скопировать файл	File.Copy (стр. 82) или FileSystemObject.CopyFile (стр. 37)
Удалить файл	File.Delete (стр. 83) или FileSystemObject.DeleteFile (стр. 43)

Следующий пример программы создает текстовый файл в корневом каталоге диска C, пишет в него некоторую информацию, перемещает этот файл в каталог с именем \tmp, делает копию этого файла в каталоге \temp, а затем удаляет обе копии из обоих каталогов. Чтобы эта программа работала, необходимо предварительно создать нужные каталоги (\tmp и \temp) в корне диска C:.

```
Sub ManipFiles
  Dim fso As FileSystemObject, f1 As TextStream, _
      f2 As File, s As String
  Set fso = CreateObject("Scripting.FileSystemObject")
  Set f1 = fso.CreateTextFile("c:\testfile.txt", True)
  Debug.Print "Начинаю писать в файл"
' Запишем строку.
  f1.Write ("Это проба.")
' Закрываем файл на запись.
  f1.Close
  Debug.Print "Перемещаю файл в c:\tmp"
' Получить дескриптор файла в корне C:.
  Set f2 = fso.GetFile("c:\testfile.txt")
' Переместить его в каталог \tmp.
  f2.Move ("c:\tmp\testfile.txt")
  Debug.Print "Копирую файл в c:\temp"
' Копировать файл в \temp.
  f2.Copy ("c:\temp\testfile.txt")
  Debug.Print "Удаляю файл..."
' Получить дескрипторы файлов по их текущему положению.
  Set f2 = fso.GetFile("c:\tmp\testfile.txt")
  Set f3 = fso.GetFile("c:\temp\testfile.txt")
' Удалить файлы.
  f2.Delete
  f3.Delete
  Debug.Print "Вот и все!"
End Sub
```

Пример кода, использующего FileSystemObject

Описанный в настоящем разделе фрагмент кода дает реальный пример, который демонстрирует множество возможностей, предоставляемых объектной моделью FileSystemObject. Этот код показывает, как все эти возможности модели работают вместе, и как их можно эффективно использовать в вашем собственном коде.

Обратите внимание на то, что, поскольку приводимый код является довольно общим, то для того, чтобы заставить его работать на вашей машине, придется дописать немного вашего собственного кода и слегка модифицировать приведенный пример. Эти изменения необходимы ...

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' FileSystemObject Sample Code
'
' Copyright 1998 Microsoft Corporation.  All Rights Reserved.
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Option Explicit

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' Замечания по поводу качества кода:
'
' 1) Нижеприведенный код выполняет много операций со строками,
'    конкатенируя короткие строки с помощью оператора "&".
'    Поскольку конкатенация строк ресурсоемка, такой способ написания
'    кода крайне неэффективен. Однако, это очень удобный способ
'    написания кода с точки зрения его дальнейшего сопровождения,
'    и он использован здесь потому, что эта программа все равно
'    выполняет много дисковых операций, а диск много медленнее, чем
'    операции в оперативной памяти, необходимые для конкатенации
'    строк. Имейте в виду, что приводимый здесь код – демонстранционный,
'    а не промышленный.
'
' 2) Использована опция "Option Explicit", поскольку доступ к
'    объявленным переменным выполняется несколько быстрее, чем к
'    не объявленным. Эта опция также предохраняет ваш код от появления
'    некоторых ошибок (в оригинале: "bugs from creeping into your code"
'    – "от заползания жучков в ваш код"), таких, например, как ошибочная
'    замена DriveTypeCDROM на DriveTypeCDORM.
'
' 3) Приведенный код совершенно не содержит обработок ошибок – просто
'    для того, чтобы его было удобнее читать. Хотя были предприняты
'    определенные предосторожности для того, чтобы при выполнении этого
'    кода в большинстве случаев не возникали ошибки, файловые системы
'    таят много непредсказуемого. В промышленном коде используйте
'    конструкцию On Error Resume Next и объект Err для перехвата
'    возможных ошибок.
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' Некоторые полезные глобальные переменные (и константы)
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Dim TabStop As String
Dim NewLine As String

Const TestDrive As String = "C"
Const TestFilePath As String = "C:\Test"

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' Константы, возвращаемые Drive.DriveType
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Const DriveTypeRemovable As Long = 1
Const DriveTypeFixed As Long = 2
Const DriveTypeNetwork As Long = 3
Const DriveTypeCDROM As Long = 4
Const DriveTypeRAMDisk As Long = 5

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' Константы, возвращаемые File.Attributes
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Const FileAttrNormal As Long = 0
Const FileAttrReadOnly As Long = 1
Const FileAttrHidden As Long = 2
Const FileAttrSystem As Long = 4
Const FileAttrVolume As Long = 8
Const FileAttrDirectory As Long = 16
Const FileAttrArchive As Long = 32
Const FileAttrAlias As Long = 64
Const FileAttrCompressed As Long = 128

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' Константы для открытия файлов
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Const OpenFileForReading As Long = 1
Const OpenFileForWriting As Long = 2
Const OpenFileForAppending As Long = 8
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' ShowDriveType
'
' Цель:
'
' Создает строку, описывающую тип диска для заданного объекта Drive
'
' Что иллюстрирует:
'
'   Drive.DriveType
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Function ShowDriveType(Drive As Object) As String

  Dim S As String

  Select Case Drive.DriveType
  Case DriveTypeRemovable
    S = "Съемный"
  Case DriveTypeFixed
    S = "Жесткий"
  Case DriveTypeNetwork
    S = "Сетевой"
  Case DriveTypeCDROM
    S = "CD-ROM"
  Case DriveTypeRAMDisk
    S = "Виртуальный"
  Case Else
    S = "Неизвестный"
  End Select

  ShowDriveType = S

End Function
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' ShowFileAttr
'
' Цель:
'
' Создает строку, описывающую атрибуты файла или каталога.
'
' Что иллюстрирует:
'
'   File.Attributes
'   Folder.Attributes
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Function ShowFileAttr(File As Object) As String
                     ' File - может быть или файл, или каталог

  Dim S As String
    Dim Attr As Long

  Attr = File.Attributes

  If Attr = 0 Then
    ShowFileAttr = "Обычный"
    Exit Function
  End If

  If Attr And FileAttrDirectory  Then S = S & "Каталог "
  If Attr And FileAttrReadOnly   Then S = S & "Только для чтения "
  If Attr And FileAttrHidden     Then S = S & "Скрытый "
  If Attr And FileAttrSystem     Then S = S & "Системный "
  If Attr And FileAttrVolume     Then S = S & "Том "
  If Attr And FileAttrArchive    Then S = S & "Архивный "
  If Attr And FileAttrAlias      Then S = S & "Синоним "
  If Attr And FileAttrCompressed Then S = S & "Сжатый "

  ShowFileAttr = S

End Function
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' GenerateDriveInformation
'
' Цель:
'
' Создает строку, описывающую текущее состояние доступных дисков
'
' Что иллюстрирует:
'
'   FileSystemObject.Drives
'   Перебор коллекции Drives
'   Drives.Count
'   Drive.AvailableSpace
'   Drive.DriveLetter
'   Drive.DriveType
'   Drive.FileSystem
'   Drive.FreeSpace
'   Drive.IsReady
'   Drive.Path
'   Drive.SerialNumber
'   Drive.ShareName
'   Drive.TotalSize
'   Drive.VolumeName
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Function GenerateDriveInformation(FSO As Object) As String

  Dim Drives As Object
  Dim Drive As Object
  Dim S As String

  Set Drives = FSO.Drives

  S = "Всего дисков:" & TabStop & Drives.Count & NewLine & NewLine

  ' Создадим 1-ю строку отчета.
  S = S & String(16, "-") & " Диск "
  S = S & String(16, "-") & Space(8) & " Файловая"
  S = S & " Всего"
  S = S & Space(7) & "Свободно"
  S = S & Space(4) & "Доступно"
  S = S & Space(4) & "Серийный" & NewLine

  ' Создадим 2-ю строку отчета.
  S = S & "Буква"
  S = S & " Путь"
  S = S & " Тип"
  S = S & Space(7) & "Готов?"
  S = S & Space(2) & "Метка"
  S = S & Space(5) & Space(8) & "система"
  S = S & Space(2) & "места"
  S = S & Space(7) & "места"
  S = S & Space(7) & "места"
  S = S & Space(7) & "номер" & NewLine

  ' Строка-разделитель.
  S = S & String(105, "-") & NewLine
```

```
    For Each Drive In Drives

      S = S & Drive.DriveLetter
      S = S & Space(5) & Drive.Path
      S = S & Space(3) & ShowDriveType(Drive)
      S = S & Space(10 - Len(ShowDriveType(Drive))) & Drive.IsReady

      If Drive.IsReady Then
        Dim nm As String
        S = S & Space(8 - Len(Drive.IsReady))
        If DriveTypeNetwork = Drive.DriveType Then
          nm = Drive.ShareName
        Else
          nm = Drive.VolumeName
        End If
        S = S & nm

        S = S & Space(18 - Len(nm)) & Drive.FileSystem
        S = S & Space(9 - Len(Drive.FileSystem)) & Drive.TotalSize
        S = S & Space(12 - Len(Drive.TotalSize)) & Drive.FreeSpace
        S = S & Space(12 - Len(Drive.FreeSpace)) & Drive.AvailableSpace
        S = S & Space(12 - Len(Drive.AvailableSpace)) & _
              Hex(Drive.SerialNumber)

      End If

      S = S & NewLine

    Next

    GenerateDriveInformation = S

  End Function
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' GenerateFileInformation
'
' Цель:
'
' Создает строку, описывающую текущее состояние файла
'
' Что иллюстрирует:
'
'    File.Path
'    File.Name
'    File.Type
'    File.DateCreated
'    File.DateLastAccessed
'    File.DateLastModified
'    File.Size
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Function GenerateFileInformation(File As Object) As String

  Dim S As String

  S = NewLine & "Путь:" & TabStop & File.Path
  S = S & NewLine & "Имя:" & TabStop & File.Name
  S = S & NewLine & "Тип:" & TabStop & File.Type
  S = S & NewLine & "Атрибуты:" & TabStop & ShowFileAttr(File)
  S = S & NewLine & "Создан:" & TabStop & File.DateCreated
  S = S & NewLine & "Доступ был:" & TabStop & File.DateLastAccessed
  S = S & NewLine & "Изменен:" & TabStop & File.DateLastModified
  S = S & NewLine & "Размер:" & TabStop & File.Size & NewLine

  GenerateFileInformation = S

End Function
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' GenerateFolderInformation
'
' Цель:
'
' Создает строку, описывающую текущее состояние каталога
'
' Что иллюстрирует:
'
'   Folder.Path
'   Folder.Name
'   Folder.DateCreated
'   Folder.DateLastAccessed
'   Folder.DateLastModified
'   Folder.Size
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Function GenerateFolderInformation(Folder As Object) As String

  Dim S As String

  S = NewLine & "Путь:" & TabStop & Folder.Path
  S = S & NewLine & "Имя:" & TabStop & Folder.Name
  S = S & NewLine & "Атрибуты:" & TabStop & ShowFolderAttr(Folder)
  S = S & NewLine & "Создан:" & TabStop & Folder.DateCreated
  S = S & NewLine & "Доступ был:" & TabStop & Folder.DateLastAccessed
  S = S & NewLine & "Изменен:" & TabStop & Folder.DateLastModified
  S = S & NewLine & "Размер:" & TabStop & Folder.Size & NewLine

  GenerateFolderInformation = S

End Function




''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' GenerateAllFolderInformation
'
' Цель:
'
' Создает строку, описывающую текущее состояние каталога
' и всех файлов и подкаталогов в нем.
'
' Что иллюстрирует:
'
'   Folder.Path
'   Folder.SubFolders
'   Folders.Count
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Function GenerateAllFolderInformation(Folder As Object) As String

  Dim S As String
  Dim SubFolders As Collection
  Dim SubFolder As Object
  Dim Files As Collection
  Dim File As Object
```

```
    S = "Каталог:" & TabStop & Folder.Path & NewLine & NewLine

    Set Files = Folder.Files

    If 1 = Files.Count Then
      S = S & "Присутствует 1 файл" & NewLine
    Else if 1 < Files.Count And Files.Count < 5 Then
      S = S & "Присутствуют " & Files.Count & " файла" & NewLine
    Else
      S = S & "Присутствуют " & Files.Count & " файлов" & NewLine
    End If

    If Files.Count <> 0 Then

      For Each File In Files
        S = S & GenerateFileInformation(File)
      Next

    End If

    Set SubFolders = Folder.SubFolders

    If 1 = SubFolders.Count Then
      S = S & NewLine & " Присутствует 1 подкаталог" & NewLine & NewLine
    Else if 1 < SubFolders.Count And SubFolders.Count < 5 Then
      S = S & NewLine & "Присутствуют " & SubFolders.Count & _
          " подкаталога" & NewLine & NewLine
    Else
      S = S & NewLine & "Присутствуют " & SubFolders.Count & _
          " подкаталогов" & NewLine & NewLine
    End If

    If SubFolders.Count <> 0 Then

      For Each SubFolder In SubFolders
        S = S & GenerateFolderInformation(SubFolder)
      Next

      S = S & NewLine

      For Each SubFolder In SubFolders
        S = S & GenerateAllFolderInformation(SubFolder)
      Next

    End If

    GenerateAllFolderInformation = S

  End Function
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' GenerateTestInformation
'
' Цель:
'
' Создает строку, описывающую текущее состояние каталога C:\Test
' и всех файлов и подкаталогов в нем.
'
' Что иллюстрирует:
'
'   FileSystemObject.DriveExists
'   FileSystemObject.FolderExists
'   FileSystemObject.GetFolder
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Function GenerateTestInformation(FSO As Object) As String

  Dim TestFolder As Object
  Dim S As String

  If Not FSO.DriveExists(TestDrive) Then Exit Function
  If Not FSO.FolderExists(TestFilePath) Then Exit Function

  Set TestFolder = FSO.GetFolder(TestFilePath)

  GenerateTestInformation = GenerateAllFolderInformation(TestFolder)

End Function
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' DeleteTestDirectory
'
' Цель:
'
' Очищает рабочий каталог.
'
' Что иллюстрирует:
'
'   FileSystemObject.GetFolder
'   FileSystemObject.DeleteFile
'   FileSystemObject.DeleteFolder
'   Folder.Delete
'   File.Delete
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Sub DeleteTestDirectory(FSO As Object)

  Dim TestFolder As Object
  Dim SubFolder As Object
  Dim File As Object

  ' Два способа удалить файл:

  FSO.DeleteFile(TestFilePath & "\Beatles\OctopusGarden.txt")

  Set File = FSO.GetFile(TestFilePath & "\Beatles\BathroomWindow.txt")
  File.Delete


  ' Два способа удалить каталог:

  FSO.DeleteFolder(TestFilePath & "\Beatles")

  FSO.DeleteFile(TestFilePath & "\ReadMe.txt")

  Set TestFolder = FSO.GetFolder(TestFilePath)
  TestFolder.Delete

End Sub
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' CreateLyrics
'
' Цель:
'
' Создает пару тестовых файлов в каталоге.
'
'
' Что иллюстрирует:
'
'   FileSystemObject.CreateTextFile
'   TextStream.WriteLine
'   TextStream.Write
'   TextStream.WriteBlankLines
'   TextStream.Close
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Sub CreateLyrics(Folder As Object)

  Dim TextStream As Object

  Set TextStream = Folder.CreateTextFile("OctopusGarden.txt")

  TextStream.Write("Octopus' Garden ")
     ' Заметим, что предыдущий оператор не добавил перевод строки в файл
  TextStream.WriteLine("(by Ringo Starr)")
  TextStream.WriteBlankLines(1)
  TextStream.WriteLine( _
     "I'd like to be under the sea in an octopus' garden in the shade,")
  TextStream.WriteLine( _
     "He'd let us in, knows where we've been -- " & _
     "in his octopus' garden in the shade.")
  TextStream.WriteBlankLines(2)

  TextStream.Close

  Set TextStream = Folder.CreateTextFile("BathroomWindow.txt")
  TextStream.WriteLine( _
     "She Came In Through The Bathroom Window (by Lennon/McCartney)")
  TextStream.WriteLine("")
  TextStream.WriteLine( _
     "She came in through the bathroom window " & _
     "protected by a silver spoon")
  TextStream.WriteLine( _
     "But now she sucks her thumb and wanders " & _
     "by the banks of her own lagoon")
  TextStream.WriteBlankLines(2)
  TextStream.Close

End Sub
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' GetLyrics
'
' Цель:
'
' Вывести содержимое файла со стихами.
'
'
' Что иллюстрирует:
'
'   FileSystemObject.OpenTextFile
'   FileSystemObject.GetFile
'   TextStream.ReadAll
'   TextStream.Close
'   File.OpenAsTextStream
'   TextStream.AtEndOfStream
'   TextStream.ReadLine
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Function GetLyrics(FSO As Object) As String

  Dim TextStream As Object
  Dim S As String
  Dim File As Object

  ' Есть несколько способов открыть текстовый файл,
  ' и несколько способов прочесть данные из файла.
  ' Вот по два из них на каждый случай:

  Set TextStream = FSO.OpenTextFile(TestFilePath & _
                   "\Beatles\OctopusGarden.txt", OpenFileForReading)

  S = TextStream.ReadAll & NewLine & NewLine
  TextStream.Close

  Set File = FSO.GetFile(TestFilePath & "\Beatles\BathroomWindow.txt")
  Set TextStream = File.OpenAsTextStream(OpenFileForReading)
  Do   While Not TextStream.AtEndOfStream
    S = S & TextStream.ReadLine & NewLine
  Loop
  TextStream.Close

  GetLyrics = S

End Function
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' BuildTestDirectory
'
' Цель:
'
' Создает иерархию рабочих каталогов для демонстрации FileSystemObject.
'
' Мы будем строить иерархию в следующем порядке:
'
' C:\Test
' C:\Test\ReadMe.txt
' C:\Test\Beatles
' C:\Test\Beatles\OctopusGarden.txt
' C:\Test\Beatles\BathroomWindow.txt
'
' Что иллюстрирует:
'
'   FileSystemObject.DriveExists
'   FileSystemObject.FolderExists
'   FileSystemObject.CreateFolder
'   FileSystemObject.CreateTextFile
'   Folders.Add
'   Folder.CreateTextFile
'   TextStream.WriteLine
'   TextStream.Close
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Function BuildTestDirectory(FSO As Object) As String

  Dim TestFolder As Object
  Dim SubFolders As Collection
  Dim SubFolder As Object
  Dim TextStream As Object

  ' Выпутываемся из ситуаций, когда (а) требуемый диск не существует
  ' (недоступен) или (б) каталог, который пытаемся построить,
  ' уже есть.

  If Not FSO.DriveExists(TestDrive) Then
    BuildTestDirectory = False
    Exit Function
  End If

  If FSO.FolderExists(TestFilePath) Then
    BuildTestDirectory = False
    Exit Function
  End If

  Set TestFolder = FSO.CreateFolder(TestFilePath)

  Set TextStream = FSO.CreateTextFile(TestFilePath & "\ReadMe.txt")
  TextStream.WriteLine("My song lyrics collection")
  TextStream.Close

  Set SubFolders = TestFolder.SubFolders

  Set SubFolder = SubFolders.Add("Beatles")

  CreateLyrics SubFolder

  BuildTestDirectory = True

End Function
```

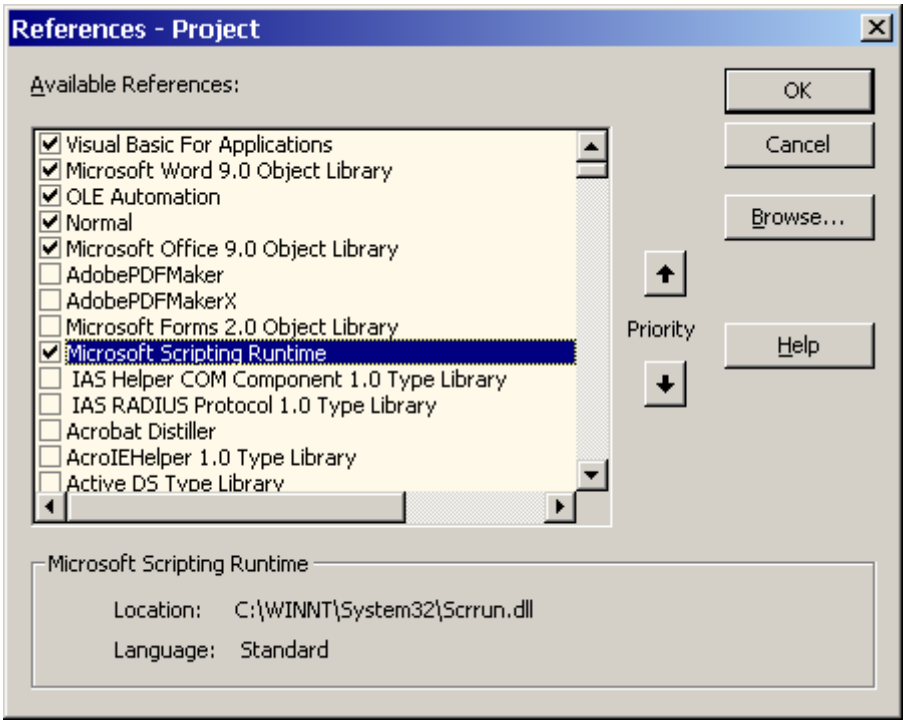
```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
' Главная программа
'
' Первым делом она создает рабочий каталог вместе с некоторыми
'   подкаталогами и файлами.
' Затем программа выводит (dumps ;-) некоторую информацию о
'   существующих дисках и о рабочем каталоге,
' а затем все почищает за собой.
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Sub Main

  Dim FSO As Object

  ' Установим значения глобальных переменных.
  TabStop = Chr(9)   ' Табулятор
  NewLine = vbCr

  Set FSO = CreateObject("Scripting.FileSystemObject")

  If Not BuildTestDirectory(FSO) Then
    MsgBox("Рабочий каталог уже существует или не может быть создан." _
        & "  Продолжить работу невозможно."
    Exit Sub
  End If

  Print GenerateDriveInformation(FSO) & NewLine & NewLine

  Print GenerateTestInformation(FSO) & NewLine & NewLine

  Print GetLyrics(FSO) & NewLine & NewLine

  DeleteTestDirectory(FSO)

End Sub
```

Справочная информация об объектной модели FileSystemObject

Объектная модель FileSystemObject находится в библиотеке Scrrun.dll и включает значительное количество классов (объектов), а также ряд перечислимых типов, использование которых обеспечивает удобство программирования с применением объектной модели FileSystemObject.

Замечания

1. Для того, чтобы объектная модель FileSystemObject была полностью (не только объекты, их свойства и методы, но и константы, определенные в перечислимых типах этой модели) доступна приложению, рекомендуется включить в проект ссылку на соответствующую библиотеку (меню Tools|References):



Включение библиотеки Scrrun.dll в проект позволяет описывать объектные переменные, используя предельно точный тип вместо обобщенного типа Object:

```
Dim fso As FileSystemObject
Dim fldr As Folder
Dim f As File
Dim ts As TextStream
```

2. Все составляющие объектной модели FileSystemObject – объекты, так что не следует забывать инициализировать их, а также не следует забывать о необходимости использования оператора Set при присваивании:

```
Dim fso As FileSystemObject
Set fs = CreateObject("Scripting.FileSystemObject")
```

3. Всюду в описаниях объектной модели термин "строка" означает строковое выражение.

Классы, входящие в объектную модель FileSystemObject

Объектная модель FileSystemObject включает следующие классы:

Dictionary
: Объект, предназначенный для хранения пар "ключ–значение". Строго говоря, не относится к объектной модели FileSystemObject
В данном руководстве не рассматривается

Drive
: Интерфейсный объект для доступа к информации о диске или разделяемом каталоге
(см. стр. 60)

Drives
: Коллекция интерфейсных объектов дисков
(см. стр. 70)

Encoder
: Назначение не описано
В данном руководстве не рассматривается.

File
: Интерфейсный объект для доступа к файлу
(см. стр. 71)

Files
: Коллекция интерфейсных объектов файлов
(см. стр. 89)

FileSystemObject
: Интерфейсный объект для доступа к файловой системе локального компьютера (включая файлы и каталоги других компьютеров, доступные на данном компьютере через сеть)
(см. стр. 32)

Folder
: Интерфейсный объект для доступа к каталогу
(см. стр. 70)

Folder
: Коллекция интерфейсных объектов каталогов
(см. стр. 70)

TextStream
: Интерфейсный объект для доступа к текстовым потокам (текстовым файлам и стандартным потокам ввода, вывода и сообщений об ошибках)
(см. стр. 90)

Объектная модель FileSystemObject включает также следующие перечислимые типы:

CompareMethod
: Способ сравнения строк (используется объектом Dictionary)
В данном руководстве не рассматривается

DriveTypeConst
: Типы дисков (CD-ROM, Removable и т.п.) – см. стр. 63

FileAttribute
: Атрибуты файлов (см. стр. 73)

IOMode

Режим ввода/вывода (запись, дозапись, чтение) – см. стр. 59

SpecialFolderConst

Виды специальных каталогов (системный, для временных файлов и т.п.) – см. стр. 53

StandardStreamTypes

Виды стандартных текстовых потоков (ввод, вывод, сообщения об ошибках) – см. стр. 55

Tristate

Формат кодировки текста (Unicode/ASCII) – см. стр. 59

Объект FileSystemObject

Описание

Обеспечивает доступ к файловой системе компьютера.

Синтаксис

*Scripting.**FileSystemObject***

Замечания

Нижеприведенный код иллюстрирует то, каким образом объект FileSystemObject используется для получения объекта TextStream, который, в свою очередь, может быть использован для чтения данных из файла или записи данных в файл (см. стр. 90):

```
Set fs = CreateObject("Scripting.FileSystemObject")
Set a = fs.CreateTextFile("c:\testfile.txt", True)
a.WriteLine("This is a test.")
a.Close
```

В приведенном выше коде функция CreateObject возвращает объект FileSystemObject (в переменную fs). Метод CreateTextFile затем создает файл как объект TextStream (переменная a), а метод WriteLine записывает строку текста в созданный текстовый файл. Метод Close сбрасывает файловый буфер на диск и закрывает файл.

Свойства и методы объекта FileSystemObject

Приведем краткую сводку свойств и методов объекта FileSystemObject. Подробные описания свойств и методов объекта будут приведены ниже.

Свойства объекта FileSystemObject

Объект FileSystemObject имеет единственное свойство:

Drives As DriveCollection

Коллекция дисков, доступных на данной машине. Только для чтения.

Свойство Drives

Описание

Возвращает коллекцию, состоящую из интерфейсных объектов Drive для всех дисков, доступных на локальной машине. Эту коллекцию можно только читать.

Синтаксис

object.***Drives***

object – всегда объект класса FileSystemObject

Замечания

Для того, чтобы диски для съемных носителей были включены в коллекцию Drives, необязательно, чтобы в них присутствовали носители.

Можно последовательно получать доступ ко всем членам коллекции Drives, применяя конструкцию For Each … Next – примерно так, как показано в нижеприведенном коде:

```
Sub ShowDriveList
  Dim fs As FileSystemObject, d As Drive, dc As Collection, _
    s As String, n As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set dc = fs.Drives
  For Each d in dc
    s = s & d.DriveLetter & " - "
    If d.DriveType = 3 Then
      n = d.ShareName
    Else
      n = d.VolumeName
    End If
    s = s & n & vbCrLf
  Next
  MsgBox s
End Sub
```

Методы объекта FileSystemObject

Объект FileSystemObject имеет 26 методов:

Function **BuildPath**(Path As String, Name As String) As String

Возвращает строку, содержащую путь, сконструированный из заданного префикса пути (Path) и последнего звена пути (Name).

Sub **CopyFile**(Source As String, Destination As String, _
Optional OverWriteFiles As Boolean = True)

Копирует файл.

Sub **CopyFolder**(Source As String, Destination As String, _
Optional OverWriteFiles As Boolean = True)

Копирует каталог.

Function **CreateFolder**(Path As String) As Folder

Создает каталог.

Function **CreateTextFile**(FileName As String, _
Optional Overwrite As Boolean = True, _
Optional Unicode As Boolean = False) As TextStream

Создает текстовый файл и открывает доступ к нему через объект текстового потока.

Sub **DeleteFile**(FileSpec As String, Optional Force As Boolean = False)

Удаляет файл.

Sub **DeleteFolder**(FolderSpec As String, _
Optional Force As Boolean = False])

Удаляет каталог

Function **DriveExists**(DriveSpec As String) As Boolean

Проверяет, существует ли диск или разделяемый каталог

Function **FileExists**(FileSpec As String) As Boolean

Проверяет, существует ли файл

Function **FolderExists**(FolderSpec As String) As Boolean

Проверяет, существует ли каталог

Function **GetAbsolutePathName**(Path As String) As String

Возвращает каноническое представление полного пути

Function **GetBaseName**(Path As String) As String

Возвращает базовое (без расширения) имя файла

Function **GetDrive**(DriveSpec As String) As Drive

Возвращает объект Drive – интерфейсный объект диска или разделяемого каталога

Function **GetDriveName**(Path As String) As String

Выделяет букву диска из строки пути

Function **GetExtensionName**(Path As String) As String

Возвращает "расширение" имени файла

Function **GetFile**(FilePath As String) As File

Возвращает File – интерфейсный объект файла

Function **GetFileName**(Path As String) As String

Возвращает имя файла (базовое, точку и расширение – если последнее есть)

Function **GetFileVersion**(FileName As String) As String

Возвращает версию файла (если атрибут Version файла пуст, то возвращается пустая строка)

Function **GetFolder**(FolderPath As String) As Folder

Возвращает Folder – интерфейсный объект каталога

Function **GetParentFolderName**(Path As String) As String

Возвращает строку – имя родительского каталога

Function **GetSpecialFolder**(SpecialFolder As SpecialFolderConst) _
As Folder

Возвращает Folder – интерфейсный объект запрошенного специального каталога

Function **GetStandardStream**(_
StandardStreamType As StandardStreamTypes, _
Optional Unicode As Boolean = False) As TextStream

Возвращает интерфейсный объект – TextStream – текстового потока, связанного с одним из стандартных потоков (ввода, вывода или сообщений об ошибках); в VBA неприменим

Function **GetTempName**() As String

Порождает случайную текстовую строку, которую удобно использовать в качестве имени временного файла

Sub **MoveFile**(Source As String, Destination As String)

Перемещает или переименовывает файл(ы)

Sub **MoveFolder**(Source As String, Destination As String)

Перемещает или переименовывает каталог(и)

Function **OpenTextFile**(FileName As String, _
Optional IO_Mode As IO_Mode = IOMode.ForReading, _
Optional Create As Boolean = False, _
Optional Format As Tristate = Tristate.TristateFalse]) As TextStream

Открывает файл как текстовый поток

Метод BuildPath

Описание

Добавляет имя в конец пути.

Синтаксис

*object.**BuildPath**(path, name)*

Формат вызова метода BuildPath состоит из трех частей ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
path	String	Да	Строка, содержащая путь, к которому должно быть добавлено значение, содержащееся в строке name. Путь, содержащийся в строке path, может быть абсолютным или относительным, и не обязательно должен указывать на существующий каталог.
name	String	Да	Строка, добавляемая как имя файла в конец пути, содержащегося в строке path.

Тип возвращаемого значения

String

Замечания

Путь, используемый в качестве первого параметра метода BuildPath, может оканчиваться обратной косой чертой, а может и не оканчиваться ею. При необходимости метод BuildPath вставляет обратную косую черту перед строкой name.

Метод CopyFile

Описание

Копирует один или несколько файлов из одного места файловой системы в другое.

Синтаксис

*object.**CopyFile** source, destination*[*, overwrite*]

Формат вызова метода BuildPath состоит из трех (или четырех) частей ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
source	String	Да	Строка, содержащая спецификацию одного или нескольких файлов, которые должны быть скопированы. Может содержать метасимволы ("?" и/или "*").
destination	String	Да	Строка, содержащая спецификацию места (каталога и/или файла/-файлов), в которое должны быть скопированы файлы, описанные параметром source. Метасимволы в этом параметре не допускаются.
overwrite	Boolean	Нет	Значение по умолчанию – True. Булевское (Boolean) значение, указывающее на возможность записи копируемых файлов поверх уже существующих файлов с такими же именами ("перезапись"). Если этот параметр равен True, то перезапись возможна, если False – запрещена. **Замечание.** Вне зависимости от значения параметра overwrite метод CopyFile не выполняется, если у существующего файла, указанного как destination, установлен атрибут read-only ("только для чтения").

Тип возвращаемого значения

Нет.

Замечания

Метасимволы разрешается использовать только в последнем звене аргумента source. Например, можно написать

```
FileSystemObject.CopyFile "c:\mydocuments\letters\*.doc", "c:\tempfolder\"
```

Но нельзя

```
FileSystemObject.CopyFile "c:\mydocuments\*\R1???97.xls", "c:\tempfolder"
```

Если источник данных (параметр source) содержит метасимволы или параметр-приемник (destination) заканчивается разделителем звеньев пути (\), предполагается, что параметр destination означает существующий каталог, в который необходимо скопировать файлы с именами, соответствующими параметру source. В противном случае второй параметр (destination) интерпретируется как имя файла, который необходимо создать. В любом случае при копировании единственного файла возможно возникновение одной из трех ситуаций:

- Если файл, задаваемый параметром destination, не существует, то файл-источник копируется. Так обычно и происходит.
- Если файл, задаваемый параметром destination – существующий файл, а параметр overwrite имеет значение False, то возникает ошибка. В противном случае (overwrite имеет значение True) делается попытка скопировать файл-источник поверх существующего файла (предварительно его удалив).
- Если destination – каталог, то возникает ошибка.

Ошибка также возникает, если параметр source содержит метасимволы, и значению этого параметра не соответствует ни один файл.

Метод CopyFile останавливается при первой же обнаруженной ошибке. При этом не делается никаких попыток откатить или отменить те изменения в файловой системе, которые уже были сделаны при выполнении этого метода до момента возникновения ошибки.

Метод CopyFolder

Описание

Рекурсивно (со всеми вложенными) копирует каталоги из одного места в другое.

Синтаксис

*object.**CopyFolder** source, destination*[*, overwrite*]

Формат вызова метода CopyFolder состоит из трех (или четырех) частей ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
source	String	Да	Строка, содержащая спецификацию одного или нескольких каталогов, которые должны быть скопированы. Может содержать метасимволы ("?" и/или "*").
destination	String	Да	Строка, содержащая спецификацию места, в которое должны быть скопированы каталоги и подкаталоги, описанные параметром source. Метасимволы в этом параметре не допускаются.
overwrite	Boolean	Нет	Значение по умолчанию – True. Булевское (Boolean) значение, указывающее на возможность записи копируемых каталогов поверх уже существующих каталогов с такими же именами ("перезапись"). Если этот параметр равен True, то перезапись возможна, если False – запрещена.

Тип возвращаемого значения

Нет

Замечания

Метасимволы разрешается использовать только в последнем звене аргумента source. Например, можно написать

```
FileSystemObject.CopyFolder "c:\mydocuments\letters\*", "c:\tempfolder\"
```

Но нельзя

```
FileSystemObject.CopyFolder "c:\mydocuments\*\*", "c:\tempfolder\"
```

Если источник данных (параметр source) содержит метасимволы или параметр-приемник (destination) заканчивается разделителем звеньев пути (\), предполагает-

ся, что параметр destination означает существующий каталог, в который необходимо скопировать каталоги с именами, соответствующими параметру source, и их подкаталоги. В противном случае второй параметр (destination) интерпретируется как имя каталога, который необходимо создать. В любом случае при копировании единственного каталога возможно возникновение одной из четырех ситуаций:

- Если каталог, задаваемый параметром destination, не существует, то каталог-источник со всем его содержимым копируется. Так обычно и происходит.
- Если параметр destination задает существующий файл, то возникает ошибка.
- Если параметр destination задает существующий каталог, то делается попытка скопировать каталог, указанный параметром source, вместе со всем его содержимым. Если каталог-источник содержит файл с именем, совпадающим с именем одного из файлов в каталоге-приемнике, и параметр overwrite имеет значение False, то возникает ошибка. В противном случае (overwrite имеет значение True) делается попытка скопировать файл-источник поверх существующего файла (предварительно его удалив).
- Если параметр destination задает каталог, имеющий атрибут read-only, и при этом параметр overwrite имеет значение False, ошибка возникает при попытке скопировать существующий файл с атрибутом read-only.

Ошибка также возникает, если параметр source содержит метасимволы, и значению этого параметра не соответствует ни один файл.

Метод CopyFolder останавливается при первой же обнаруженной ошибке. При этом не делается никаких попыток откатить или отменить те изменения в файловой системе, которые уже были сделаны при выполнении этого метода до момента возникновения ошибки.

Метод CreateFolder

Описание

Создает каталог.

Синтаксис

*object.**CreateFolder**(foldername)*

Формат вызова метода CreateFolder состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
foldername	String	Да	Строка, идентифицирующая каталог, который требуется создать.

Тип возвращаемого значения

Объект Folder

Замечания

Если каталог, указанный в качестве значения параметра foldername, уже существует, то возникает ошибка.

Метод CreateTextFile

Описание

Создает текстовый файл с указанным именем и возвращает объект класса TextStream (см. стр. 90), который может быть использован для записи в созданный файл и/или чтения из него.

Синтаксис

*object.**CreateTextFile**(filename[, overwrite[, unicode]])*

Формат вызова метода CreateTextFile состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
filename	String	Да	Строка, идентифицирующая путь к файлу, который требуется создать.
overwrite	Boolean	Нет	Значение по умолчанию – True. Булевское (Boolean) значение, указывающее на возможность записи создаваемого файла поверх уже существующего файла с таким же именем ("перезапись"). Если этот параметр равен True, то перезапись возможна, если False – запрещена.
unicode	Boolean	Нет	Значение по умолчанию – False. Если значение этого параметра равно True, то текстовый файл создается в кодировке Unicode, если False, – то в текущей кодировке Windows.

Тип возвращаемого значения

Объект TextStream

Замечания

Нижеприведенный код иллюстрирует, как использовать метод CreateTextFile для того, чтобы создать и открыть текстовый файл:

```
Sub CreateAfile
  Dim fs As FileSystemObject, ts As TextStream
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set ts = fs.CreateTextFile("c:\testfile.txt", True)
  ts.WriteLine("This is a test.")
  ts.Close
End Sub
```

Если значение параметра overwrite – False (или этот параметр отсутствует), а параметр filename указывает на уже существующий файл, то возникает ошибка.

Метод DeleteFile

Описание

Удаляет указанный файл (указанные файлы).

Синтаксис

*object.**DeleteFile** filespec*[*, force*]

Формат вызова метода DeleteFile состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
filespec	String	Да	Строка, содержащая спецификацию одного или нескольких файлов, которые должны быть удалены. В последнем звене пути может содержать метасимволы ("?" и/или "*").
force	Boolean	Нет	Значение по умолчанию – False. Это значение следует установить в True, если среди удаляемых файлов есть файлы с установленным атрибутом read-only ("только для чтения") и они должны быть удалены.

Тип возвращаемого значения

Нет

Замечания

Если значению параметра filespec не соответствует ни один файл, возникает ошибка.

Метод DeleteFile останавливается при первой же обнаруженной ошибке. При этом не делается никаких попыток откатить или отменить те изменения в файловой системе, которые уже были сделаны при выполнении этого метода до момента возникновения ошибки.

Метод DeleteFolder

Описание

Удаляет указанный каталог (указанные каталоги) вместе со всем содержимым.

Синтаксис

*object.**DeleteFolder** folderSpec*[*, force*]

Формат вызова метода DeleteFolder состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
folderSpec	String	Да	Строка, содержащая спецификацию одного или нескольких каталогов, которые должны быть удалены. В последнем звене пути может содержать метасимволы ("?" и/или "*").
force	Boolean	Нет	Значение по умолчанию – False. Это значение следует установить в True, если среди удаляемых каталогов или файлов есть файлы с установленным атрибутом read-only ("только для чтения") и они должны быть удалены.

Тип возвращаемого значения

Нет

Замечания

Метод DeleteFolder не делает различия между каталогами, в которых что-то есть и пустыми каталогами. Указанный каталог удаляется вне зависимости от того, есть ли в нем другие каталоги и/или файлы.

Метод DeleteFolder останавливается при первой же обнаруженной ошибке. При этом не делается никаких попыток откатить или отменить те изменения в файловой системе, которые уже были сделаны при выполнении этого метода до момента возникновения ошибки.

Метод DriveExists

Описание

Возвращает True, если указанный диск существует, а если указанного диска не существует, то возвращает False.

Синтаксис

*object.**DriveExists**(drivespec)*

Формат вызова метода DriveExists состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
drivespec	String	Да	Строка, содержащая букву диска или полный путь.

Тип возвращаемого значения

Boolean

Замечания

Для дисководов со съемными носителями метод DriveExists возвращает значение True, даже если носитель в приводе отсутствует. Для определения готовности дисковода к работе следует использовать свойство IsReady объекта Drive.

Метод FileExists

Описание

Возвращает True, если указанный файл существует, а если указанного файла не существует, то возвращает False.

Синтаксис

*object.**FileExists**(filespec)*

Формат вызова метода FileExists состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
filespec	String	Да	Строка, содержащая спецификацию файла, существование которого проверяется. Если проверяется существование файла в каталоге, отличном от текущего, спецификация должна содержать путь к файлу (абсолютный или относительный).

Тип возвращаемого значения

Boolean

Метод FolderExists

Описание

Возвращает True, если указанный каталог существует, а если указанного каталога не существует, то возвращает False.

Синтаксис

*object.**FolderExists**(folderspec)*

Формат вызова метода FolderExists состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
folderspec	String	Да	Строка, содержащая спецификацию каталога, существование которого проверяется. Если проверяется существование каталога в каталоге, отличном от текущего, спецификация должна содержать путь к проверяемому каталогу (абсолютный или относительный).

Тип возвращаемого значения

Boolean

Замечания

Метод GetAbsolutePathName

Описание

Возвращает полный путь, который однозначно соответствует (неполной) спецификации пути, указанной в качестве параметра вызова метода.

Синтаксис

*object.**GetAbsolutePathName**(pathspec)*

Формат вызова метода GetAbsolutePathName состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	*Да*	Всегда имя объекта класса FileSystemObject.
pathspec	String	*Да*	Строка, содержащая (неполную) спецификацию пути, которую надлежит превратить в полную спецификацию.

Тип возвращаемого значения

String

Замечания

Спецификация пути полна и однозначна, если она указывает полный путь от корня соответствующего диска. Полный путь может заканчиваться разделителем звеньев (обратной косой чертой – "\"), только если этот путь специфицирует корневой каталог на т.н. "отображаемом" (mapped) диске.

Пример:

Предположим, что текущий каталог – это c:\mydocuments\reports. Нижеприведенная таблица иллюстрирует поведение метода GetAbsolutePathName в этом случае.

Значение параметра **pathspec**	Строка, возвращаемая методом **GetAbsolutePathName**
"c:"	"c:\mydocuments\reports"
"c:.."	"c:\mydocuments"
"c:\\\"	"c:\"
"c:*.*\may97"	"c:\mydocuments\reports*.*\may97"
"region1"	"c:\mydocuments\reports\region1"
"c:\..\..\mydocuments"	"c:\mydocuments"

Метод GetBaseName

Описание

Возвращает строку, содержащую базовое имя последнего звена пути (без последней точки и расширения, если они есть).

Синтаксис

*object.**GetBaseName**(path)*

Формат вызова метода GetBaseName состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	*Да*	Всегда имя объекта класса FileSystemObject.
path	String	*Да*	Строка, содержащая спецификацию пути, из которой надлежит извлечь базовое имя последнего звена.

Тип возвращаемого значения

String

Замечания

Метод GetBaseName возвращает пустую строку (""), если значение параметра path не соответствует синтаксису задания пути.

Метод GetBaseName работает только со строкой, заданной ему в качестве параметра. Он не пытается проверить существование указанного пути.

Метод GetDrive

Описание

Возвращает объект класса Drive, соответствующий диску в указанной спецификации пути.

Синтаксис

*object.**GetDrive** drivespec*

Формат вызова метода GetDrive состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
drivespec	String	Да	Строка, содержащая спецификацию пути, по которой следует идентифицировать диск. Параметр drivespec может иметь одну из следующих форм: • буква диска, например, c • буква диска с последующим двоеточием, например, c: • буква диска с последующими двоеточием и разделителем звеньев пути, например, c:\ или • спецификация сетевого разделяемого каталога, например, \\computer2\share1

Тип возвращаемого значения

Объект Drive

Замечания

Для сетевых разделяемых каталогов выполняется проверка их существования.

Если спецификация пути, идентифицирующего диск (параметр drivespec) не соответствует ни одной из приведенных форм, или указанный в ней диск не существует, то возникает ошибка.

Для того, чтобы вызвать метод GetDrive для строки Path, содержащей обычный путь, используйте следующую конструкцию для получения строки DriveSpec, пригодной в качестве спецификации пути к диску – параметра метода GetDrive:

```
DriveSpec = GetDriveName(GetAbsolutePathName(Path))
```

Метод GetDriveName

Описание

Возвращает строку, содержащую имя диска (букву или последовательность \\<имя компьютера>\<имя разделяемого каталога>), соответствующего указанному пути.

Синтаксис

*object.**GetDriveName**(path)*

Формат вызова метода GetDriveName состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
path	String	Да	Строка, содержащая спецификацию пути, из которой надлежит извлечь имя диска.

Тип возвращаемого значения

String

Замечания

Метод GetDriveName возвращает пустую строку (""), если заданная строка path не позволяет определить имя диска .

Метод GetDriveName работает только со строкой, заданной ему в качестве параметра. Он не пытается проверить существование указанного пути.

Метод GetExtensionName

Описание

Возвращает строку, содержащую имя расширения для последнего звена пути.

Синтаксис

*object.**GetExtensionName**(path)*

Формат вызова метода GetExtensionName состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
path	String	Да	Строка, содержащая спецификацию пути, из которой надлежит извлечь имя расширения.

Тип возвращаемого значения

String

Замечания

Для путей к сетевым дискам обозначение корневого каталога ("\") считается звеном.

Если параметр path метода GetExtensionName не может быть интерпретирован как путь, метод возвращает пустую строку.

Метод GetFile

Описание

Возвращает интерфейсный объект файла, соответствующего указанному пути.

Синтаксис

*object.**GetFile**(filespec)*

Формат вызова метода GetFile состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
filespec	String	Да	Строка, содержащая путь (абсолютный или относительный) к требуемому файлу.

Тип возвращаемого значения

Объект File

Замечания

Если указанный файл не существует, возникает ошибка.

Метод GetFileName

Описание

Возвращает последнее звено указанного пути, которое входит в часть пути, не являющуюся спецификацией диска.

Синтаксис

*object.**GetFileName**(pathspec)*

Формат вызова метода GetFileName состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
pathspec	String	Да	Строка, содержащая путь (абсолютный или относительный).

Тип возвращаемого значения

String

Замечания

Если строка pathspec заканчивается звеном, которое не может быть интерпретировано как имя файла, метод GetFileName возвращает пустую строку.

Метод GetFileName работает только со строкой, заданной ему в качестве параметра. Он не пытается проверить существование указанного пути.

Метод GetFileVersion

Описание

Возвращает строку, содержащую описание версии файла (если таковая есть, обычно – последовательность чисел, разделенных точками).

Синтаксис

*object.**GetFileVersion**(filespec)*

Формат вызова метода GetFileVersion состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
filespec	String	Да	Строка, содержащая путь (абсолютный или относительный) к файлу, версию которого следует определить.

Тип возвращаемого значения

String

Замечания

Если указанный файл не существует, возвращается пустая строка.

Метод GetFolder

Описание

Возвращает интерфейсный объект каталога.

Синтаксис

*object.**GetFolder**(folderspec)*

Формат вызова метода GetFolder состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
folderspec	String	Да	Строка, содержащая путь (абсолютный или относительный) к требуемому каталогу.

Тип возвращаемого значения

Объект Folder

Замечания

Если указанный каталог не существует, возникает ошибка.

Метод GetParentFolderName

Описание

Возвращает строку, содержащую имя родительского (предпоследнего) звена в пути к последнему звену указанного пути.

Синтаксис

*object.**GetParentFolderName**(path)*

Формат вызова метода GetParentFolderName состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	*Да*	Всегда имя объекта класса FileSystemObject.
path	String	*Да*	Строка, содержащая путь (абсолютный или относительный) к объекту, имя родительского каталога которого надлежит определить.

Тип возвращаемого значения

String

Замечания

Если родительский каталог в указанном пути выделить невозможно, возвращается пустая строка.

Метод GetParentFolderName работает только со строкой, заданной ему в качестве параметра. Он не пытается проверить существование указанного пути.

Метод GetSpecialFolder

Описание

Возвращает интерфейсный объект запрошенного специального каталога

Синтаксис

*object.**GetSpecialFolder**(folderspec)*

Формат вызова метода GetSpecialFolder состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	*Да*	Всегда имя объекта класса FileSystemObject.
folderspec	SpecialFolderConst	*Да*	Константа, указывающая, интерфейсный объект к какому именно специальному каталогу следует вернуть (см. замечания ниже).

Тип возвращаемого значения

Объект Folder

Замечания

Тип SpecialFolderConst – перечислимый тип со следующими тремя значениями:

Символьная константа	Целое значение	Описание
WindowsFolder	0	Каталог, который содержит файлы текущей установки Windows
SystemFolder	1	Системный каталог, содержащий библиотеки, шрифты, драйверы и т.п.
TemporaryFolder	2	Каталог для хранения временных файлов. Путь к этому каталогу соответствует значению переменной окружения TMP.

Метод GetStandardStream

Описание

Возвращает интерфейсный объект текстового потока, связанного с одним из стандартных потоков (ввода, вывода или сообщений об ошибках)

Синтаксис

*object.**GetStandardStream**(StdStrType,* [*Unicode As Boolean*]*)*

Формат вызова метода GetStandardStream состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	*Да*	Всегда имя объекта класса FileSystemObject.
StdStrType	StandardStreamTypes	*Да*	Константа, указывающая, интерфейсный объект к какому из стандартных потоков следует вернуть (см. замечания ниже).
unicode	Boolean	Нет	Значение по умолчанию – False. Если значение этого параметра равно True, то текстовый файл создается в кодировке Unicode, если False, – то в текущей кодировке Windows.

Тип возвращаемого значения

Объект TextStream

Замечания

В VBA неприменим. Применяется в скриптах Windows Scripting Host для управления Windows.

Тип StandardStreamTypes – перечислимый тип со следующими тремя значениями:

Символьная константа	Целое значение	Описание
StdIn	0	Поток стандартного ввода.
StdOut	1	Поток стандартного вывода.
StdErr	2	Поток сообщений об ошибках.

Метод GetTempName

Описание

Возвращает случайное имя, пригодное для использования в качестве имени временного файла или каталога.

Синтаксис

*object.**GetTempName***

Формат вызова метода GetTempName состоит из единственного обязательного компонента:

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.

Тип возвращаемого значения

String

Замечания

Метод не создает файла, а лишь генерирует подходящее имя. Используйте метод CreateTextFile для создания файла или CreateFolder для создания каталога.

Метод MoveFile

Описание

Перемещает или переименовывает файл(ы).

Синтаксис

*object.**MoveFile** source, destination*

Формат вызова метода MoveFile состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
source	String	Да	Строка, содержащая спецификацию одного или нескольких файлов, которые должны быть перемещены/переименованы. Может содержать метасимволы ("?" и/или "*"), но только в последнем звене пути.
destination	String	Да	Путь, указывающий, куда должны быть перемещены файлы (переименован файл). Метасимволы не допускаются.

Тип возвращаемого значения

Нет

Замечания

Если источник данных (параметр source) содержит метасимволы или параметр-приемник (destination) заканчивается разделителем звеньев пути (\), предполагается, что параметр destination означает существующий каталог, в который необходимо переместить файлы с именами, соответствующими параметру source. В противном случае второй параметр (destination) интерпретируется как имя файла, который необходимо создать. В любом случае при перемещении единственного файла возможно возникновение одной из трех ситуаций:

- Если файл, задаваемый параметром destination, не существует, то файл-источник перемещается/переименовывается. Так обычно и происходит.
- Если файл, задаваемый параметром destination – существующий файл, то возникает ошибка.
- Если destination – каталог, то возникает ошибка.

Ошибка также возникает, если параметр source содержит метасимволы, и значению этого параметра не соответствует ни один файл.

Метод MoveFile останавливается при первой же обнаруженной ошибке. При этом не делается никаких попыток откатить или отменить те изменения в файловой системе, которые уже были сделаны при выполнении этого метода до момента возникновения ошибки.

Перемещение файлов между томами (дисками) выполняется только в том случае, если это поддерживается операционной системой.

Метод MoveFolder

Описание

Рекурсивно (со всеми вложенными) перемещает каталоги из одного места в другое или переименовывает каталог.

Синтаксис

*object.**MoveFolder** source, destination*

Формат вызова метода MoveFolder состоит из трех частей ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	*Да*	Всегда имя объекта класса FileSystemObject.
source	String	*Да*	Строка, содержащая спецификацию одного или нескольких каталогов, которые должны быть перемещены/переименованы. Может содержать метасимволы ("?" и/или "*"), но только в последнем звене пути.
destination	String	*Да*	Путь, указывающий, куда должны быть перемещены каталоги (переименован каталог) и подкаталоги, описанные параметром source. Метасимволы в этом параметре не допускаются.

Тип возвращаемого значения

Нет

Замечания

Если источник данных (параметр source) содержит метасимволы или параметр-приемник (destination) заканчивается разделителем звеньев пути (\), предполагается, что параметр destination означает существующий каталог, в который необходимо переместить каталоги с именами, соответствующими параметру source, и их подкаталоги. В противном случае второй параметр (destination) интерпретируется как имя каталога, который необходимо создать. В любом случае при перемещении/переименовании единственного каталога возможно возникновение одной из трех ситуаций:

- Если каталог, задаваемый параметром destination, не существует, то каталог-источник со всем его содержимым перемещается или переименовывается. Так обычно и происходит.
- Если параметр destination задает существующий файл, то возникает ошибка.
- Если параметр destination задает существующий каталог, то возникает ошибка.

Ошибка также возникает, если параметр source содержит метасимволы, и значению этого параметра не соответствует ни один файл.

Метод MoveFolder останавливается при первой же обнаруженной ошибке. При этом не делается никаких попыток откатить или отменить те изменения в файловой системе, которые уже были сделаны при выполнении этого метода до момента возникновения ошибки.

Метод OpenTextFile

Описание

Открывает указанный текстовый файл и возвращает интерфейсный объект – TextStream (см. стр. 90) – текстового потока, который можно использовать для чтения данных из этого файла и/или записи данных в него.

Синтаксис

*object.**OpenTextFile**(filename[, iomode[, create[, format]]])*

Формат вызова метода OpenTextFile состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса FileSystemObject.
filename	String	Да	Строка, содержащая путь (абсолютный или относительный) к файлу, который требуется открыть.
iomode	IOMode	Нет	Указывает режим ввода/вывода. Может принимать одно из значений, допустимых в перечислимом типе IOMode: ForReading, ForWriting и ForAppending (см. замечания ниже).
create	Boolean	Нет	Указывает, следует ли создавать текстовый файл, если файл с указанным параметром filename именем не существует. Если значение параметра create равно True, то файл будет создан, а если False, то не будет. Значение по умолчанию – False (не создавать).
format	Tristate	Нет	Одно из трех значений типа Tristate, задающих формат кодировки открываемого файла (принятый в операционной системе, ASCII или Unicode).

Тип возвращаемого значения

Объект TextStream

Замечания

Перечислимый тип данных IOMode задает три значения:

Символьная константа	Целое значение	Описание
ForReading	1	Только для чтения. Запись в файл, открытый в этом режиме, невозможна.
ForWriting	2	Файл будет открыт на запись. Если файл с указанным именем существует, то его содержимое в момент открытия файла будет уничтожено.
ForAppending	8	Файл будет открыт на запись. Запись данных будет осуществляться в конец открытого файла.

В документации на Office 2000 имеется ошибка. В ней указано, что для задания режима ввода/вывода для открываемого файла допустимо использование только двух из упомянутых трех констант перечислимого типа IOMode (ForReading, ForWriting и ForAppending), в то время как поставляемая библиотека Scripting на самом деле правильно реализует и запись в текстовый файл, открытый с использованием значения ForWriting.

Перечислимый тип данных Tristate задает четыре символьные константы, использующие только три значения:

Символьная константа	Альтернативная символьная константа	Целое значение	Описание
TristateFalse	vbFalse	0	Открывает файл, считая, что формат кодировки текста в файле – ASCII.
TristateTrue	vbTrue	-1	Открывает файл, считая, что формат кодировки текста в файле – Unicode.
TristateMixed	vbUseDefault	-2	Открывает файл, считая, что формат кодировки открываемого файла совпадает с форматом, принятым в операционной системе.
TristateUseDefault	vbUseDefault	-2	То же, что и TristateMixed.

Использование символьных констант перечислимых типов IOMode и Tristate возможно, только если библиотека типов Scripting подключена к проекту. В против-

ном случае необходимо использовать целые значения, соответствующие необходимым константам.

Приведенный ниже код иллюстрирует применение метода OpenTextFile для открытия файла с целью дописывания текста в конец файла:

```
Sub OpenTextFileTest
  Dim fs as FileSystemObject, f as TextStream
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.OpenTextFile("c:\testfile.txt", _
                          IOMode.ForAppending, _
                          Tristate.TristateFalse)
  f.Write "Hello world!"
  f.Close
End Sub
```

Объект Drive

Описание

Обеспечивает доступ к информации о свойствах конкретного диска или сетевого разделяемого каталога.

Замечания

Приведенный ниже код иллюстрирует использование объекта Drive для доступа к информации о свойствах диска.

```
Sub ShowFreeSpace(drvPath)
  Dim fs As FileSystemObject, d As Drive, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set d = fs.GetDrive(fs.GetDriveName(drvPath))
  s = "Диск " & UCase(drvPath) & " - "
  s = s & d.VolumeName  & vbCrLf
  s = s & "Свободно: " & FormatNumber(d.FreeSpace/1024, 0)
  s = s & " килобайт"
  MsgBox s
End Sub
```

Свойства и методы объекта Drive

Объект Drive не имеет методов.

Приведем краткую сводку свойств объекта Drive. Подробные описания свойств объекта Drive будут приведены ниже.

Свойства *объекта Drive*

Объект Drive имеет 12 свойств. Все они, за исключением свойства VolumeName, являются свойствами "***только для чтения***".

AvailableSpace As Long
: Размер доступного ***для данного пользователя*** пространства на указанном диске или в сетевом разделяемом каталоге

DriveLetter As String
: Буква диска для физического диска или сетевого разделяемого каталога

DriveType As DriveTypeConst
: Целое число, означающее тип указанного диска

FileSystem As String
: Строка, обозначающая тип файловой системы

FreeSpace As Long
: Размер свободного пространства на указанном диске или в сетевом разделяемом каталоге

IsReady As Boolean
: True, если указанный диск находится в состоянии готовности к работе, и False – в противном случае

Path As String
: Путь для указанного диска

RootFolder As Folder
: Объект Folder, представляющий корневой каталог указанного диска

SerialNumber As Long
: Десятичное число – уникальный идентификатор дискового тома ("серийный номер")

ShareName As String
: Строка, содержащая имя разделяемого сетевого каталога для указанного диска

TotalSize As Long
: Полный объем (в байтах) указанного диска или разделяемого сетевого каталога

Property **VolumeName**(newname As String) As String
: Позволяет прочесть и/или изменить метку тома указанного диска

Свойство AvailableSpace

Описание

Размер доступного **для данного пользователя** пространства на указанном диске или в сетевом разделяемом каталоге (в байтах).

Синтаксис

*object.**AvailableSpace***

object – всегда объект Drive

Тип возвращаемого значения

Long

Замечания

Значение, возвращаемое свойством AvailableSpace, обычно совпадает с величиной, возвращаемой свойством FreeSpace. Разница может возникать в системах, в которых установлены квоты дискового пространства.

Приведенный ниже код иллюстрирует использование свойства AvailableSpace:

```
Sub ShowAvailableSpace(drvPath)
  Dim fs As FileSystemObject, d As Drive, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set d = fs.GetDrive(fs.GetDriveName(drvPath))
  s = "Диск " & UCase(drvPath) & " - "
  s = s & d.VolumeName  & vbCrLf
  s = s & "Доступно: " & FormatNumber(d.AvailableSpace/1024, 0)
  s = s & " килобайт"
  MsgBox s
End Sub
```

Свойство DriveLetter

Описание

Буква диска для физического диска или сетевого разделяемого каталога

Синтаксис

*object.**DriveLetter***

object – всегда объект Drive

Тип возвращаемого значения

String

Замечания

Свойство DriveLetter возвращает пустую строку (""), если указанный диск не ассоциирован с буквой, например, для сетевого разделяемого каталога, не отображенного на букву диска.

Приведенный ниже код иллюстрирует использование свойства DriveLetter:

```
Sub ShowDriveLetter(drvPath)
  Dim fs As FileSystemObject, d As Drive, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set d = fs.GetDrive(fs.GetDriveName(drvPath))
  s = "Диск " & d.DriveLetter & ": - "
  s = s & d.VolumeName  & vbCrLf
  s = s & "Свободно: " & FormatNumber(d.FreeSpace/1024, 0)
  s = s & " килобайт"
  MsgBox s
End Sub
```

Свойство DriveType

Описание

Целое число, означающее тип указанного диска

Синтаксис

*object.**DriveType***

object – всегда объект Drive

Тип возвращаемого значения

DriveTypeConst

Замечания

Свойство возвращает целое число (от 0 до 5), принадлежащее перечислимому типу DriveTypeConst, определенному в объектной модели FileSystemObject.

Перечислимый тип данных DriveTypeConst задает шесть констант, соответствующих различным типам дисков:

Символьная константа	Целое значение	Описание
UnknownType	0	Неизвестный тип диска
Removable	1	Привод со съемным носителем (для гибких дисков и т.п.)
Fixed	2	Несъемный ("жесткий") диск
Remote	3	Сетевой ("удаленный")
CDRom	4	Привод для оптических носителей
RamDisk	5	Виртуальный диск (в ОЗУ)

Приведенный ниже код иллюстрирует использование свойства DriveType:

```
Sub ShowDriveType(drvpath)
  Dim fs As FileSystemObject, d As Drive, s As String, t As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set d = fs.GetDrive(drvpath)
  Select Case d.DriveType
     Case DriveTypeConst.UnknownType
       t = "Неизвестный (Unknown)"
     Case DriveTypeConst.Removable
       t = "Съемный (Removable)"
     Case DriveTypeConst.Fixed
       t = "Несъемный (Fixed)"
     Case DriveTypeConst.Remote
       t = "Сетевой (Network)"
     Case DriveTypeConst.CDRom
       t = "CD-ROM"
     Case DriveTypeConst.RamDisk
       t = "Виртуальный (RAM Disk)"
  End Select
  s = "Диск " & d.DriveLetter & ": - " & t
  MsgBox s
End Sub
```

Свойство FileSystem

Описание

Строка, обозначающая тип файловой системы

Синтаксис

*object.**FileSystem***

object – всегда объект Drive

Тип возвращаемого значения

String

Замечания

В настоящее время свойство FileSystem возвращает одну из трех строк: "FAT", "NTFS" или "CDFS".

Приведенный ниже код иллюстрирует использование свойства FileSystem:

```
Sub ShowFileSystemType
  Dim fs As FileSystemObject, d As Drive
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set d = fs.GetDrive("e:")
  MsgBox d.FileSystem
End Sub
```

Свойство FreeSpace

Описание

Размер свободного пространства на указанном диске или в сетевом разделяемом каталоге.

Синтаксис

*object.**FreeSpace***

object – всегда объект Drive

Тип возвращаемого значения

Long

Замечания

Значение, возвращаемое свойством FreeSpace, обычно совпадает с величиной, возвращаемой свойством AvailableSpace. Разница может возникать в системах, в которых установлены квоты дискового пространства.

Приведенный ниже код иллюстрирует использование свойства FreeSpace:

```
Sub ShowAvailableSpace(drvPath)
  Dim fs As FileSystemObject, d As Drive, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set d = fs.GetDrive(fs.GetDriveName(drvPath))
  s = "Диск " & UCase(drvPath) & " - "
  s = s & d.VolumeName  & vbCrLf
  s = s & "Свободно: " & FormatNumber(d.AvailableSpace/1024, 0)
  s = s & " килобайт"
  MsgBox s
End Sub
```

Свойство IsReady

Описание

Готовность диска: True, если указанный диск находится в состоянии готовности к работе, и False – в противном случае

Синтаксис

*object.**IsReady***

object – всегда объект Drive

Тип возвращаемого значения

Boolean

Замечания

Для дисководов, предназначенных для съемных носителей и для приводов CD-ROM свойство IsReady возвращает значение True только в том случае, когда соответствующий носитель вставлен в привод и готов к доступу.

Приведенный ниже код иллюстрирует использование свойства IsReady:

```
Sub ShowDriveInfo(drvpath)
  Dim fs As FileSystemObject, d As Drive, s As String, t As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set d = fs.GetDrive(drvpath)
  Select Case d.DriveType
     Case DriveTypeConst.UnknownType
       t = "Неизвестный (Unknown)"
     Case DriveTypeConst.Removable
       t = "Съемный (Removable)"
     Case DriveTypeConst.Fixed
       t = "Несъемный (Fixed)"
     Case DriveTypeConst.Remote
       t = "Сетевой (Network)"
     Case DriveTypeConst.CDRom
       t = "CD-ROM"
     Case DriveTypeConst.RamDisk
       t = "Виртуальный (RAM Disk)"
  End Select
  s = "Drive " & d.DriveLetter & ": - " & t
  If d.IsReady Then
     s = s & vbCrLf & "Диск готов.
  Else
     s = s & vbCrLf & "Диск не готов.
  End If
  MsgBox s
End Sub
```

Свойство Path

Описание

Путь для указанного диска, файла или каталога. Только для чтения.

Синтаксис

*object.**Path***

object может быть одним из следующих объектов: File, Folder или Drive

Тип возвращаемого значения

String

Замечания

Для букв дисков корневой каталог в путь не включается. Например, для диска C будет возвращен путь C:, но не C:\.

Приведенный ниже код иллюстрирует использование свойства Path с объектом File:

```
Sub ShowFileAccessInfo(filespec)
  Dim fs As FileSystemObject, f As File, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFile(filespec)
  s = UCase(f.Path) & vbCrLf
  s = s & "Created: " & f.DateCreated & vbCrLf
  s = s & "Last Accessed: " & f.DateLastAccessed & vbCrLf
  s = s & "Last Modified: " & f.DateLastModified
  MsgBox s, 0, "File Access Info"
End Sub
```

Свойство RootFolder

Описание

Объект Folder, представляющий корневой каталог указанного диска

Синтаксис

*object.**RootFolder***

object – всегда объект Drive

Тип возвращаемого значения

Folder

Замечания

Возвращаемый объект Folder можно использовать для доступа ко всем файлам и каталогам диска.

Свойство SerialNumber

Описание

Десятичное число – уникальный идентификатор дискового тома ("серийный номер")

Синтаксис

*object.**SerialNumber***

object – всегда объект Drive

Тип возвращаемого значения

Long

Замечания

Свойство SerialNumber можно использовать для того, чтобы проверить, что в дисковод, предназначенный для съемных носителей, вставлен правильный диск.

Приведенный ниже код иллюстрирует использование свойства SerialNumber:

```
Sub ShowDriveInfo(drvpath)
  Dim fs As FileSystemObject, d As Drive, s As String, t As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set d = fs.GetDrive(fs.GetDriveName(fs.GetAbsolutePathName(drvpath)))
  Select Case d.DriveType
     Case DriveTypeConst.UnknownType
       t = "Неизвестный (Unknown)"
     Case DriveTypeConst.Removable
       t = "Съемный (Removable)"
     Case DriveTypeConst.Fixed
       t = "Несъемный (Fixed)"
     Case DriveTypeConst.Remote
       t = "Сетевой (Network)"
     Case DriveTypeConst.CDRom
       t = "CD-ROM"
     Case DriveTypeConst.RamDisk
       t = "Виртуальный (RAM Disk)"
  End Select
  s = "Диск " & d.DriveLetter & ": - " & t
  s = s & vbCrLf & "Серийный номер: " & d.SerialNumber
  MsgBox s
End Sub
```

Свойство ShareName

Описание

Строка, содержащая имя разделяемого сетевого каталога для указанного диска

Синтаксис

*object.**ShareName***

object – всегда объект Drive

Тип возвращаемого значения

String

Замечания

Если объект – не сетевой диск, то свойство ShareName возвращает пустую строку ("").

Приведенный ниже код иллюстрирует использование свойства ShareName:

```
Sub ShowDriveInfo(drvpath)
  Dim fs As FileSystemObject, d As Drive, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set d = fs.GetDrive(fs.GetDriveName(fs.GetAbsolutePathName(drvpath)))
  s = "Диск " & d.DriveLetter & ": - " & d.ShareName
  MsgBox s
End Sub
```

Свойство TotalSize

Описание

Полный объем (в байтах) указанного диска или разделяемого сетевого каталога

Синтаксис

*object.**TotalSize***

object – всегда объект Drive

Тип возвращаемого значения

Long

Замечания

Приведенный ниже код иллюстрирует использование свойства TotalSize:

```
Sub ShowSpaceInfo(drvpath)
  Dim fs As FileSystemObject, d As Drive, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set d = fs.GetDrive(fs.GetDriveName(fs.GetAbsolutePathName(drvpath)))
  s = "Диск " & d.DriveLetter & ":"
  s = s & vbCrLf
  s = s & "Полный объем: " & _
        FormatNumber(d.TotalSize/1024, 0) & " Килобайт"
  s = s & vbCrLf
  s = s & "Доступно: " & _
        FormatNumber(d.AvailableSpace/1024, 0) & " Килобайт"
  MsgBox s
End Sub
```

Свойство VolumeName

Описание

Позволяет прочесть и/или изменить метку тома указанного диска. Для чтения или записи.

Синтаксис

*object.**VolumeName*** [*= newname*]

Свойство VolumeName имеет следующие два компонента:

Часть	Тип	Об.	Описание
object	Drive	Да	Всегда имя объекта типа Drive.
newname	String	Нет	Если параметр newname задан, то он задает новое имя объекта

Тип возвращаемого или присваиваемого значения

String

Замечания

Приведенный ниже код иллюстрирует использование свойства VolumeName:

```
Sub ShowDriveInfo(drvpath)
  Dim fs As FileSystemObject, d As Drive, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set d = fs.GetDrive(fs.GetDriveName(fs.GetAbsolutePathName(drvpath)))
  s = "Диск " & d.DriveLetter & ": - " & d.VolumeName
  MsgBox s
End Sub
```

Коллекция Drives

Описание

Коллекция, содержащая интерфейсные объекты всех дисков, доступных на машине. Только для чтения.

Замечания

Для того, чтобы диски для съемных носителей были включены в коллекцию Drives, необязательно, чтобы в них присутствовали носители.

Можно последовательно получать доступ ко всем членам коллекции Drives, применяя конструкцию For Each ... Next – примерно так, как показано в нижеприведенном коде:

```
Sub ShowDriveList
  Dim fs As FileSystemObject, d As Drive, dc As Collection, _
    s As String, n As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set dc = fs.Drives
  For Each d in dc
    s = s & d.DriveLetter & " - "
    If d.DriveType = DriveTypeConst.Remote Then
      n = d.ShareName
    Else
      n = d.VolumeName
    End If
    s = s & n & vbCrLf
  Next
  MsgBox s
End Sub
```

Свойства и методы коллекции Drives

Коллекция Drives не имеет методов.

Коллекция Drives имеет только стандартные свойства коллекций: Count и Item.

Объекты Folder и File

Объекты Folder и File имеют много схожих свойств и методов.

Объект Folder

Описание

Интерфейсный объект, обеспечивающий доступ ко всем свойствам каталога ("папки").

Замечания

Приведенный ниже код иллюстрирует использование объекта Folder и как вернуть значение одного из его свойств:

```
Sub ShowFolderInfo(folderspec)
  Dim fs As FileSystemObject, f As Folder, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFolder(folderspec)
  s = f.DateCreated
  MsgBox s
End Sub
```

Объект File

Описание

Интерфейсный объект, обеспечивающий доступ ко всем свойствам файла.

Замечания

Приведенный ниже код показывает, как получить объект File и как вывести значение одного из его свойств:

```
Sub ShowFileInfo(filespec)
  Dim fs As FileSystemObject, f As File, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFile(filespec)
  s = f.DateCreated
  MsgBox s
End Sub
```

Свойства и методы объектов Folder и File

Одноименные свойства объектов Folder и File

Объекты Folder и File имеют 12 свойств, имена которых совпадают для обоих объектов. Смысл этих свойств, как правило, один и тот же для объектов Folder и File.

Property **Attributes**(newattr As Long) As Long

Возвращает и устанавливает новые значения атрибутов файла или каталога. Не все атрибуты можно изменять – некоторые атрибуты можно только читать

DateCreated As Date

Возвращает дату и время создания указанного файла или каталога. Только для чтения

DateLastAccessed As Date

Возвращает дату и время последнего доступа (на запись или на чтение) к указанному файлу или каталогу. Только для чтения

DateLastModified As Date

Возвращает дату и время последнего изменения указанного файла или каталога. Только для чтения

Drive As String

Возвращает букву диска, на котором расположен указанный файл или каталог. Только для чтения

Property **Name**(newname As String) As String

Возвращает или изменяет имя указанного файла или каталога

ParentFolder As Folder

Возвращает интерфейсный объект для доступа к родительскому каталогу указанного файла или каталога. Только для чтения

Path As String

Путь для указанного файла или каталога. Только для чтения

ShortName As String

Возвращает так называемое "короткое имя" объекта, которое требуется для совместимости с устаревшим соглашением "8.3"

ShortPath As String

Возвращает так называемый "короткий путь" для объекта, который требуется для совместимости с устаревшим соглашением "8.3". Каждое звено короткого пути удовлетворяет соглашению "8.3"

Size As Long

Возвращает размер файла или суммарный объем (в байтах) всех файлов и подкаталогов указанного каталога

Type As String

Возвращает информацию о типе файла, ассоциированном с расширением имени файла. Для каталога возвращается строка "File Folder"

Свойства объекта Folder

Объект Folder имеет 3 свойства, которые отсутствуют у объекта File.

Files As Files

Возвращает коллекцию класса Files, содержащую все интерфейсные объекты для доступа ко всем файлам указанного каталога, включая и те, у которых установлены атрибуты Hidden (скрытый) и/или System (системный)

IsRootFolder As Boolean

Возвращает True, если указанный каталог – корневой, – и False – в противном случае

SubFolders As Folders

Возвращает коллекцию Folders, содержащую все подкаталоги указанного каталога, включая те из них, для которых установлены атрибуты Hidden (скрытый) и/или System (системный)

Описания одноименных свойств объектов Folder и File

Свойство Attributes

Описание

Возвращает и устанавливает новые значения атрибутов файлов и каталогов. Не все атрибуты можно изменять – некоторые атрибуты можно только читать.

Синтаксис

*object.**Attributes*** [*= newattributes*]

Свойство Attributes имеет следующие два компонента:

Часть	Тип	Об.	Описание
object	File или Folder	Да	object принадлежит одному из двух классов: File или Folder.
newattributes	Long	Нет	Если параметр newattributes задан, то он задает новые значения атрибутов

Тип возвращаемого значения

Long

Замечания

Свойством Attributes обладают объекты двух классов: File и Folder.

Значение, возвращаемое свойством Attributes, является суммой (которую иногда удобно рассматривать как логическое ИЛИ – Or) констант, принадлежащих перечислимому типу FileAttribute, определенному в scrrun.dll.

Перечислимый тип данных FileAttribute задает девять констант, соответствующих различным атрибутам файла/каталога ("Изм." – атрибут можно изменять):

Символьная константа	Целое значение	Изм.	Описание
Normal	0		Ни один атрибут не установлен
ReadOnly	1	Да	Только для чтения
Hidden	2	Да	Скрытый
System	4	Да	Системный
Volume	8	Нет	Метка тома
Directory	16	Нет	Объект является каталогом
Archive	32	Да	"Подлежит архивированию" – файл был изменен с момента выполнения его последней резервной копии
Alias	64	Нет	Ссылка или ярлык
Compressed	128	Нет	Сжатый файл (на томе с файловой системой NTFS)

Приведенный ниже код иллюстрирует использование свойства Attributes с объектом File:

```
Sub SetClearArchiveBit(filespec)
  Dim fs As FileSystemObject, f As File, b As String, t As String
  b = " Архивный бит "
  t = "Установка/сброс архивного бита"
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFile(fs.GetFileName(filespec))
  If f.attributes And 32 Then
    If MsgBox(b & "установлен, сбросить?", vbYesNo, t) = vbYes Then
      f.attributes = f.attributes - 32
      MsgBox b & "сброшен."
    Else
      MsgBox b & "остался установленным."
    End If
  Else
    If MsgBox(b & "сброшен, установить?", vbYesNo, t) = vbYes Then
      f.attributes = f.attributes + 32
      MsgBox b & "установлен."
    Else
      MsgBox b & "остался сброшенным."
    End If
  End If
End Sub
```

Свойство DateCreated

Описание

Возвращает дату и время создания указанного каталога или файла. Только для чтения.

Синтаксис

*object.**DateCreated***

object принадлежит одному из двух классов: File или Folder.

Тип возвращаемого значения

Date

Замечания

Приведенный ниже код иллюстрирует использование свойства DateCreated с объектом File:

```
Sub ShowFileInfo(filespec)
  Dim fs As FileSystemObject, f As File
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFile(filespec)
  MsgBox "Создан: " & f.DateCreated
End Sub
```

Свойство DateLastAccessed

Описание

Возвращает дату и время последнего доступа (на запись или на чтение) к указанному каталогу или файлу. Только для чтения.

Синтаксис

*object.**DateLastAccessed***

object принадлежит одному из двух классов: File или Folder.

Тип возвращаемого значения

Date

Замечания

Приведенный ниже код иллюстрирует использование свойства DateLastAccessed с объектом File:

```
Sub ShowFileAccessInfo(filespec)
  Dim fs As FileSystemObject, f As File, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFile(filespec)
  s = UCase(filespec) & vbCrLf
  s = s & "Создан: " & f.DateCreated & vbCrLf
  s = s & "Последний доступ: " & f.DateLastAccessed & vbCrLf
  s = s & "Последняя модификация: " & f.DateLastModified
  MsgBox s, vbOKOnly, "Информация об использовании файла"
End Sub
```

Важное замечание. Функциональность этого свойства зависит от операционной системы, на которой выполняется приложение. Если операционная система не поддерживает информацию о времени доступа к файлу (например, Windows 98), то свойство не возвращает ничего.

Свойство DateLastModified

Описание

Возвращает дату и время последнего изменения указанного каталога или файла. Только для чтения.

Синтаксис

*object.**DateLastModified***

object принадлежит одному из двух классов: File или Folder.

Тип возвращаемого значения

Date

Замечания

Приведенный ниже код иллюстрирует использование свойства DateLastModified с объектом File:

```
Sub ShowFileAccessInfo(filespec)
  Dim fs As FileSystemObject, f As File, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFile(filespec)
  s = UCase(filespec) & vbCrLf
  s = s & "Создан: " & f.DateCreated & vbCrLf
  s = s & "Последний доступ: " & f.DateLastAccessed & vbCrLf
  s = s & "Последняя модификация: " & f.DateLastModified
  MsgBox s, vbOKOnly, "Информация об использовании файла"
End Sub
```

Свойство Drive

Описание

Возвращает букву диска, на котором расположен указанный файл или каталог. Только для чтения.

Синтаксис

*object.**Drive***

object принадлежит одному из двух классов: File или Folder.

Тип возвращаемого значения

String

Замечания

Приведенный ниже код иллюстрирует использование свойства Drive с объектом File:

```
Sub ShowFileAccessInfo(filespec)
  Dim fs As FileSystemObject, f As File, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFile(filespec)
  s = f.Name & " on Drive " & UCase(f.Drive) & vbCrLf
  s = s & "Создан: " & f.DateCreated & vbCrLf
  s = s & "Последний доступ: " & f.DateLastAccessed & vbCrLf
  s = s & "Последняя модификация: " & f.DateLastModified
  MsgBox s, vbOKOnly, "Информация об использовании файла"
End Sub
```

Свойство Name

Описание

Возвращает или изменяет имя указанного файла или каталога.

Синтаксис

*object.**Name*** [*= newname*]

Свойство Name имеет следующие два компонента:

Часть	Тип	Об.	Описание
object	File или Folder	Да	object принадлежит одному из двух классов: File или Folder.
newname	Long	Нет	Если параметр newname задан, то он задает новое имя указанного объекта

Тип возвращаемого значения

String

Замечания

Приведенный ниже код иллюстрирует использование свойства Name с объектом File:

```
Sub ShowFileAccessInfo(filespec)
  Dim fs As FileSystemObject, f As File, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFile(filespec)
  s = "Файл " & f.Name & " на диске " & UCase(f.Drive) & vbCrLf
  s = s & "Создан: " & f.DateCreated & vbCrLf
  s = s & "Последний доступ: " & f.DateLastAccessed & vbCrLf
  s = s & "Последняя модификация: " & f.DateLastModified
  MsgBox s, vbOKOnly, "Информация об использовании файла"
End Sub
```

Свойство ParentFolder

Описание

Возвращает интерфейсный объект для доступа к родительскому каталогу указанного файла или каталога. Только для чтения.

Синтаксис

*object.**ParentFolder***

object принадлежит одному из двух классов: File или Folder.

Тип возвращаемого значения

Объект Folder

Замечания

Приведенный ниже код иллюстрирует использование свойства ParentFolder с объектом File:

```
Sub ShowFileAccessInfo(filespec)
  Dim fs As FileSystemObject, f As File, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFile(filespec)
  s = "Файл " & f.Name & " В каталоге " & UCase(f.ParentFolder) & vbCrLf
  s = s & "Создан: " & f.DateCreated & vbCrLf
  s = s & "Последний доступ: " & f.DateLastAccessed & vbCrLf
  s = s & "Последняя модификация: " & f.DateLastModified
  MsgBox s, vbOKOnly, "Информация об использовании файла"
End Sub
```

Описание

Путь для указанного каталога, файла или диска. Только для чтения.

Синтаксис

*object.**Path***

object принадлежит одному из трех классов: File, Folder или Drive

Тип возвращаемого значения

String

Замечания

Для букв дисков корневой каталог в путь не включается. Например, для диска C будет возвращен путь C:, но не C:\.

Приведенный ниже код иллюстрирует использование свойства Path с объектом File:

```
Sub ShowFileAccessInfo(filespec)
  Dim fs As FileSystemObject, f As File, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFile(filespec)
  s = UCase(f.Path) & vbCrLf
  s = s & "Created: " & f.DateCreated & vbCrLf
  s = s & "Last Accessed: " & f.DateLastAccessed & vbCrLf
  s = s & "Last Modified: " & f.DateLastModified
  MsgBox s, 0, "File Access Info"
End Sub
```

Свойство ShortName

Описание

Возвращает так называемое "короткое имя" объекта, которое требуется для совместимости с устаревшим соглашением "8.3".

Синтаксис

*object.**ShortName***

object принадлежит одному из двух классов: File или Folder.

Тип возвращаемого значения

String

Замечания

Приведенный ниже код иллюстрирует использование свойства ShortName с объектом File:

```
Sub ShowShortName(filespec)
  Dim fs As FileSystemObject, f As File, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFile(filespec)
  s = "Короткое имя для файла " & "" & UCase(f.Name) & vbCrLf
  s = s & ": " & "" & f.ShortName & ""
  MsgBox s, 0, "Short Name Info"
End Sub
```

Свойство ShortPath

Описание

Возвращает так называемый "короткий путь" для объекта, который требуется для совместимости с устаревшим соглашением "8.3". Каждое звено короткого пути удовлетворяет соглашению "8.3".

Синтаксис

*object.**ShortPath***

object принадлежит одному из двух классов: File или Folder.

Тип возвращаемого значения

String

Замечания

Приведенный ниже код иллюстрирует использование свойства ShortPath с объектом File:

```
Sub ShowShortPath(filespec)
  Dim fs As FileSystemObject, f As File, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFile(filespec)
  s = "Короткое путь для файла " & "" & UCase(f.Name) & vbCrLf
  s = s & ": " & "" & f.ShortPath & ""
  MsgBox s, 0, "Short Path Info"
End Sub
```

Свойство Size

Описание

Для файлов возвращает размер файла в байтах.

Для каталогов возвращает суммарный объем (в байтах) всех файлов и подкаталогов указанного каталога.

Синтаксис

*object.**Size***

object принадлежит одному из двух классов: File или Folder.

Тип возвращаемого значения

Long

Замечания

Приведенный ниже код иллюстрирует использование свойства Size с объектом Folder:

```
Sub ShowSize(folderspec)
  Dim fs As FileSystemObject, f As Folder, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFolder(folderspec)
  MsgBox _
    "Каталог " & UCase(f.Name) & " использует " & f.Size & " байтов.", _
    0, "Folder Size Info"
End Sub
```

Свойство Type

Описание

Возвращает информацию о типе файла, ассоциированном с расширением имени файла. Например, для файла с расширением .txt будет возвращено "Text Document". Для каталога возвращается строка "File Folder".

Синтаксис

*object.**Type***

object принадлежит одному из двух классов: File или Folder.

Тип возвращаемого значения

String

Замечания

Приведенный ниже код иллюстрирует использование свойства Type с объектом Folder:

```
Sub ShowFolderType(folderspec)
  Dim fs As FileSystemObject, f As Folder, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFolder(folderspec)
  MsgBox UCase(f.Name) & " is a " & f.Type, 0, "Folder Type Info"
End Sub
```

Описания свойств объекта Folder

Свойство Files

Описание

Возвращает коллекцию класса Files, содержащую все интерфейсные объекты для доступа ко всем файлам указанного каталога, включая и те, у которых установлены атрибуты Hidden (скрытый) и/или System (системный).

Синтаксис

*object.**Files***

object – имя объекта класса Folder.

Тип возвращаемого значения

Коллекция Files

Замечания

Приведенный ниже код иллюстрирует использование свойства Files:

```
Sub ShowFileList(folderspec)
  Dim fs As FileSystemObject, f As Folder, _
      f1 As File, fc As Files, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFolder(folderspec)
  Set fc = f.Files
  For Each f1 in fc
    s = s & f1.name & vbCrLf
  Next
  MsgBox s
End Sub
```

Свойство IsRootFolder

Описание

Возвращает True, если указанный каталог – корневой, – и False – в противном случае.

Синтаксис

*object.**IsRootFolder***

object – имя объекта класса Folder.

Тип возвращаемого значения

Boolean

Замечания

Приведенный ниже код иллюстрирует использование свойства IsRootFolder:

```
Sub DisplayLevelDepth(pathspec)
  Dim fs As FileSystemObject, f As Folder, n As Long, s As String
  s = " Указанный каталог "
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFolder(folderspec)
  If f.IsRootFolder Then
    MsgBox s & "- корневой."
  Else
    n = 0
    Do Until f.IsRootFolder
      Set f = f.ParentFolder
      n = n + 1
    Loop
    MsgBox s & " находится на " & n & "-м уровне вложенности."
  End If
End Sub
```

Свойство SubFolders

Описание

Возвращает коллекцию Folders, содержащую все подкаталоги указанного каталога, включая те из них, для которых установлены атрибуты Hidden (скрытый) и/или System (системный).

Синтаксис

*object.**SubFolders***

object – имя объекта класса Folder.

Тип возвращаемого значения

Коллекция Folders

Замечания

Приведенный ниже код иллюстрирует использование свойства SubFolders:

```
Sub ShowFolderList(folderspec)
  Dim fs As FileSystemObject, f As Folder, f1 As Folder, _
      sf As Folders, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFolder(folderspec)
  Set sf = f.SubFolders
  For Each f1 in sf
    s = s & f1.name & vbCrLf
  Next
  MsgBox s
End Sub
```

Одноименные методы объектов Folder и File

Объекты Folder и File имеют 3 метода, имена которых совпадают для обоих объектов. Смысл этих методов, как правило, один и тот же для объектов Folder и File.

Sub **Copy**(destination As String, _
Optional overwrite As Boolean = True)
Копирует указанный файл или каталог из одного места в другое

Sub **Delete**(Optional force As Boolean = False)
Удаляет указанный файл или каталог

Sub **Move**(destination As String)
Перемещает из одного места в другое или переименовывает указанный файл или каталог

Описания одноименных методов объектов Folder и File

Метод Copy

Описание

Копирует указанный файл или каталог из одного места в другое.

Синтаксис

*object.**Copy** destination*[*, overwrite*]

Метод Copy имеет следующие компоненты:

Часть	Тип	Об.	Описание
object	File или Folder	Да	object принадлежит одному из двух классов: File или Folder.
destination	String	Да	Строка, задающая абсолютный или относительный путь к тому объекту – файлу или каталогу, – который получиться в результате копирования. Метасимволы (* или ?) не допускаются.
overwrite	Boolean	Нет	Если параметр overwrite установлен в True, то разрешена замена существующего файла или каталога, если False – нет. Значение по умолчанию – True.

Тип возвращаемого значения

Нет

Замечания

Результат выполнения метода Copy объектов File или Folder идентичен результату операции, выполняемой с применением методов FileSystemObject.CopyFile (стр. 37) или FileSystemObject.CopyFolder (стр. 39), соответственно, в том случае, если имя файла или каталога, переданное этим методам, указывает на тот же объект (файл или каталог), метод Copy которого используется. Следует, однако, от-

метить, что методы объекта допускают применение метасимволов (т.е. могут выполнить копирование сразу нескольких файлов и/или каталогов), в то время как метод Copy объектов File или Folder не допускает применения метасимволов и выполняет копирование единственного объекта.

Метод Delete

Описание

Удаляет указанный файл или каталог.

Синтаксис

*object.**Delete*** [*force*]

Метод Delete имеет следующие компоненты:

Часть	Тип	Об.	Описание
object	File или Folder	Да	object принадлежит одному из двух классов: File или Folder.
force	Boolean	Нет	Если значение этого параметра – True, – то указанный файл или каталог будет удален, даже если для него установлен атрибут Read-Only ("только для чтения"). Если значение этого параметра – False, – то файлы и каталоги с установленным атрибутом Read-Only при применении метода Delete не удаляются. Значение по умолчанию – False.

Тип возвращаемого значения

Нет

Замечания

Если указанный файл или каталог не существует, то возникает ошибка.

Результат выполнения метода Delete с объектом класса File или Folder идентичен результату применению методов, соответственно, DeleteFile или DeleteFolder к объекту FileSystemObject (FileSystemObject.DeleteFile – стр. 43 – или FileSystemObject.DeleteFolder – стр. 44 – с надлежащим именем файла или каталога).

Метод Delete в применении к каталогу не делает различия между каталогами, в которых что-то есть и пустыми каталогами. Указанный каталог удаляется вне зависимости от того, есть ли в нем другие каталоги и/или файлы.

Метод Move

Описание

Перемещает из одного места в другое или переименовывает указанный файл или каталог (каталог – рекурсивно (со всеми вложенными)).

Синтаксис

*object.**Move** destination*

Метод Move имеет следующие компоненты:

Часть	Тип	Об.	Описание
object	File или Folder	Да	object принадлежит одному из двух классов: File или Folder.
destination	String	Да	Путь, указывающий, куда должен быть перемещен (переименован с каким новым именем) файл или каталог. Метасимволы (* или ?) не допускаются.

Тип возвращаемого значения

Нет

Замечания

Результат выполнения метода Move с объектом класса File или Folder идентичен результату применению методов, соответственно, MoveFile или MoveFolder к объекту FileSystemObject (FileSystemObject.MoveFile – стр. 56 – или FileSystemObject.MoveFolder – стр. 57 – с надлежащим именем файла или каталога).

Методы объекта FileSystemObject, однако, могут переместить (переименовать) сразу несколько файлов или каталогов (за счет возможности использования метасимволов).

Метод CreateTextFile объекта Folder

Описание

Создает текстовый файл с указанным именем в указанном каталоге и возвращает интерфейсный объект текстового потока (класса TextStream – см. стр. 90), который можно использовать для чтения из или записи в файл.

Синтаксис

*object.**CreateTextFile**(filename[, overwrite[, unicode]])*

Метод CreateTextFile имеет следующие компоненты:

Часть	Тип	Об.	Описание
object	Folder	Да	object принадлежит классу Folder.
filename	String	Да	Строка, идентифицирующая путь к файлу, который требуется создать.
overwrite	Boolean	Нет	Значение по умолчанию – True. Булевское (Boolean) значение, указывающее на возможность записи создаваемого файла поверх уже существующего файла с таким же именем ("перезапись"). Если этот параметр равен True, то перезапись возможна, если False – запрещена.
unicode	Boolean	Нет	Значение по умолчанию – False. Если значение этого параметра равно True, то текстовый файл создается в кодировке Unicode, если False, – то в текущей кодировке Windows.

Тип возвращаемого значения

Объект TextStream

Замечания

Нижеприведенный код иллюстрирует, как использовать метод CreateTextFile для того, чтобы создать и открыть текстовый файл:

```
Sub CreateAfile
  Dim fs As FileSystemObject, f As Folder, ts As TextStream
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFolder(folderspec)
  Set ts = f.CreateTextFile("testfile.txt", True)
  ts.WriteLine("This is a test.")
  ts.Close
End Sub
```

Если значение параметра overwrite – False (или этот параметр отсутствует), а параметр filename указывает на уже существующий файл, то возникает ошибка.

Параметр filename может содержать относительный путь (в том числе и ведущий в вышележащие – относительно каталога, связанного с объектом object – каталоги, например, приведенный ниже код создает файл в корневом каталоге диска C:

```
Sub CreateAfileInParentFolder
  Dim fs As FileSystemObject, f As Folder, ts As TextStream
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFolder("C:\temp")
  Set ts = f.CreateTextFile("..\testfile.txt", True)
  ts.WriteLine("This is a test.")
  ts.Close
End Sub
```

Метод OpenAsTextStream объекта File

Описание

Открывает указанный текстовый файл и возвращает интерфейсный объект – TextStream (см. стр. 90) – текстового потока, который можно использовать для чтения данных из этого файла и/или записи данных в него.

Синтаксис

*object.**OpenAsTextStream**([iomode,* [*format*]]*)*

Формат вызова метода OpenAsTextStream состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	FileSystemObject	Да	Всегда имя объекта класса File.
iomode	IOMode	Нет	Указывает режим ввода/вывода. Может принимать одно из значений, допустимых в перечислимом типе IOMode: ForReading, ForWriting и ForAppending.
format	Tristate	Нет	Одно из трех значений типа Tristate, задающих формат кодировки открываемого файла (принятый в операционной системе, ASCII или Unicode).

Тип возвращаемого значения

Объект TextStream

Замечания

Метод OpenAsTextStream объекта File обеспечивает ту же функциональность, что и метод OpenTextFile (см. стр. 58) объекта FileSystemObject.

Метод OpenAsTextStream объекта File чаще всего используется для организации записи в файл.

Нижеприведенный код иллюстрирует использование метода OpenAsTextStream:

```
Sub TextStreamTest()
  Dim fs As FileSystemObject, f As File, ts As TextStream, s As String
  Set fs = CreateObject("Scripting.FileSystemObject")
```

```
    fs.CreateTextFile "test1.txt"            'Создадим файл
    Set f = fs.GetFile("test1.txt")
    Set ts = f.OpenAsTextStream(IOMode.ForWriting, Tristate.UseDefault)
    ts.Write "Всем привет!"
    ts.Close
    Set ts = f.OpenAsTextStream(IOMode.ForReading, Tristate.UseDefault)
    s = ts.ReadLine
    MsgBox s
    ts.Close
End Sub
```

Коллекция Folders

Описание

Коллекция Folders включает все интерфейсные объекты, обеспечивающие доступ ко всем каталогам, содержащимся в каталоге, доступ к которому был получен через родительский объект (Folder) этой коллекции.

Замечания

Приведенный ниже код показывает, как получить коллекцию Folders и как последовательно получать доступ ко всем членам этой коллекции, применяя конструкцию For Each ... Next:

```
Sub ShowFolderList(folderspec)
  Dim fs As FileSystemObject, f As Folder, fc As Collection, _
    s As String, f1 As Folder
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFolder(folderspec)
  Set fc = f.SubFolders
  For Each f1 in fc
    s = s & f1.name
    s = s &  vbCrLf
  Next
  MsgBox s
End Sub
```

Свойства и методы коллекции Folders

Коллекция Folders имеет только стандартные свойства коллекций: Count и Item.

Коллекция Folders имеет единственный метод **Add**.

Метод Add

Описание

Создает новый каталог в родительском каталоге коллекции Folders.

Синтаксис

*object.**Add** folderName*

Формат вызова метода Add состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	Folders	Да	Всегда имя коллекции Folders.
folderName	String	Да	Имя создаваемого каталога.

Тип возвращаемого значения

Объект Folder

Замечания

Если каталог с именем folderName уже существует в родительском каталоге коллекции object, то возникает ошибка.

Коллекция Files

Описание

Коллекция интерфейсных объектов класса File, обеспечивающих доступ ко всем файлам в данном каталоге.

Замечания

Приведенный ниже код показывает, как получить коллекцию Files и как последовательно получать доступ ко всем членам этой коллекции, применяя конструкцию For Each ... Next:

```
Sub ShowFileList(folderspec)
  Dim fs As FileSystemObject, f As Folder, fc As Collection, _
    s As String, f1 As File
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set f = fs.GetFolder(folderspec)
  Set fc = f.Files
  For Each f1 in fc
    s = s & f1.name
    s = s &  vbCrLf
  Next
  MsgBox s
End Sub
```

Свойства и методы коллекции Files

Коллекция Files не имеет методов.

Коллекция Files имеет только стандартные свойства коллекций: Count и Item.

Объект TextStream

Описание

Обеспечивает последовательный доступ к текстовому файлу для чтения и записи. Текстовый файл, открытый для последовательного чтения или записи, называют **потоком** (или **текстовым потоком**).

Замечания

Объект класса TextStream должен быть получен надлежащим методом объектов

FileSystemObject (методы CreateTextFile или OpenTextFile)

или

File (метод OpenAsTextStream)

например:

```
Dim fs As FileSystemObject, a As TextStream
Set fs = CreateObject("Scripting.FileSystemObject")
Set a = fs.CreateTextFile("c:\testfile.txt", True)
```

Объект класса TextStream использует понятие "**текущая позиция** в файле".

Непосредственно после открытия файла **для чтения** текущая позиция находится перед первым символом файла, и при чтении файла продвигается далее по файлу, пока не достигнет его конца.

Непосредственно после открытия файла **для записи** текущая позиция находится перед концом файла и при записи продвигается вместе с концом файла.

Свойства и методы объекта TextStream

Объект TextStream имеет 4 свойства и 9 методов.

Свойства объекта TextStream

AtEndOfLine As Boolean

Возвращает True, если текущая позиция в файле находится непосредственно перед маркером конца строки, и False в противном случае. Только для чтения

AtEndOfStream As Boolean

Возвращает True, если текущая позиция в файле находится непосредственно в конце файла, и False в противном случае. Только для чтения

Column As Long

Возвращает номер текущей символьной позиции в текущей строке файла, связанного с текстовым потоком. Только для чтения

Line As Long

Возвращает порядковый номер текущей строки в файле, связанном с текстовым потоком. Только для чтения

Свойство AtEndOfLine

Описание

Возвращает True, если текущая позиция в файле находится непосредственно перед маркером конца строки, и False в противном случае. Только для чтения.

Синтаксис

*object.**AtEndOfLine***

object – всегда имя объекта класса TextStream.

Тип возвращаемого значения

Boolean

Замечания

Свойство AtEndOfLine применимо только к потокам, открытым для чтения. Использование этого свойства с потоком, открытым для записи, приводит к ошибке.

Приведенный ниже код иллюстрирует использование свойства AtEndOfLine:

```
Dim fs As FileSystemObject, ts As TextStream, s As String
Set fs = CreateObject("Scripting.FileSystemObject")
Set ts = fs.OpenTextFile("c:\testfile.txt", IOMode.ForReading, False)
Do While ts.AtEndOfLine <> True
  s = ts.Read(1)
Loop
ts.Close
```

Свойство AtEndOfStream

Описание

Возвращает True, если текущая позиция в файле находится непосредственно в конце файла, и False в противном случае. Только для чтения.

Синтаксис

*object.**AtEndOfStream***

object – всегда имя объекта класса TextStream.

Тип возвращаемого значения

Boolean

Замечания

Свойство AtEndOfStream применимо только к потокам, открытым для чтения. Использование этого свойства с потоком, открытым для записи, приводит к ошибке.

Приведенный ниже код иллюстрирует использование свойства AtEndOfStream:

```
Dim fs As FileSystemObject, ts As TextStream, s As String
Set fs = CreateObject("Scripting.FileSystemObject")
Set ts = fs.OpenTextFile("c:\testfile.txt", IOMode.ForReading, False)
Do While ts.AtEndOfStream <> True
  s = ts.Read(1)
Loop
ts.Close
```

Свойство Column

Описание

Возвращает номер текущей символьной позиции в текущей строке файла, связанного с текстовым потоком. Только для чтения.

Синтаксис

*object.**Column***

object – всегда имя объекта класса TextStream.

Тип возвращаемого значения

Long

Замечания

После записи в файл символа перехода на новую строку, но до момента записи первого символа в новую строку свойство Column возвращает 1.

Свойство Line

Описание

Возвращает порядковый номер текущей строки в файле, связанном с текстовым потоком. Только для чтения.

Синтаксис

*object.**Line***

object – всегда имя объекта класса TextStream.

Тип возвращаемого значения

Long

Замечания

Непосредственно после открытия файла на запись и до того, как в файл было что-либо записано, Line возвращает 1.

Методы объекта TextStream

Sub **Close**

Закрывает файл, связанный с текстовым потоком

Function **Read**(characters As Long) As String

Читает указанное число символов из текстового потока и возвращает полученную таким образом строку

Function **ReadAll**() As String

Читает весь файл, связанный с текстовым потоком как одну (большую) строку

Function **ReadLine**() As String

Читает и возвращает в качестве результата остаток текущей строки файла, связанного с текстовым потоком, от текущего положения в строке до конца строки, не включая символ перехода на новую строку

Sub **Skip**(characters As Long)

Пропускает требуемое число символов при чтении текстового потока

Sub **SkipLine**()

Пропускает текущую строку при чтении текстового потока

Sub **Write**(text As String)

Выводит указанную строку в файл, связанный с текстовым потоком

Sub **WriteBlankLines**(lines As Long)

Выводит в файл, связанный с текстовым потоком, указанное количество символов перехода на новую строку

Sub **WriteLine**(text As String)

Выводит указанную строку в файл, связанный с текстовым потоком, и добавляет символ перехода на новую строку

Метод Close

Описание

Закрывает файл, связанный с текстовым потоком.

Синтаксис

*object.**Close***

object – всегда имя объекта класса TextStream.

Тип возвращаемого значения

Нет

Замечания

Попытка закрыть неоткрытый (уже закрытый) поток приводит к ошибке (91).

Метод Read

Описание

Читает указанное число символов из текстового потока и возвращает полученную таким образом строку.

Синтаксис

*object.**Read**(characters)*

Формат вызова метода Read состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	TextStream	Да	Всегда имя объекта класса TextStream.
characters	Long	Да	Количество символов, которые хочется прочесть из файла.

Тип возвращаемого значения

String

Замечания

Вызов метода Read продвигает текущую позицию на прочитанное количество символов.

Метод Read рассматривает символ перехода на новую строку как один символ (vbCr)

Попытка прочесть больше символов, чем осталось до конца файла, связанного с потоком, приводит к ошибке (62).

Метод ReadAll

Описание

Читает весь файл, связанный с текстовым потоком как одну (большую) строку.

Синтаксис

*object.**ReadAll***

object – всегда имя объекта класса TextStream.

Тип возвращаемого значения

String

Замечания

Вызов метода ReadAll продвигает текущую позицию в конец файла, связанного с текстовым потоком, и последующее чтение из файла приводит к ошибке (62).

Если читаемый файл велик, то потребность в оперативной памяти для применения метода ReadAll может быть велика. В таких случаях рекомендуется применять альтернативные методы, например, читать файл построчно.

Метод ReadLine

Описание

Читает и возвращает в качестве результата остаток текущей строки файла, связанного с текстовым потоком, от текущего положения в строке до конца строки, не включая символ перехода на новую строку (если он есть).

Синтаксис

*object.**ReadLine***

object – всегда имя объекта класса TextStream.

Тип возвращаемого значения

String

Замечания

Вызов метода ReadLine продвигает текущую позицию в начало строки, следующей за прочитанной строкой.

При последовательном применении метода ReadLine символ перехода на новую строку (если он есть – его может не быть в последней строке файла) не включается ни в одну строку, возвращаемую методом ReadLine.

Метод Skip

Описание

Пропускает требуемое число символов при чтении текстового потока.

Синтаксис

object.***Skip*** characters

Формат вызова метода Skip состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	TextStream	Да	Всегда имя объекта класса TextStream.
characters	Long	Да	Количество символов, которые хочется пропустить.

Тип возвращаемого значения

Нет

Замечания

Вызов метода Skip продвигает текущую позицию на пропущенное количество символов.

Метод Skip рассматривает символ перехода на новую строку как один символ независимо от операционной системы.

Метод Skip позволяет указать количество пропускаемых символов большее, чем остается до конца файла. При этом пропускаются все символы файла, и последующее чтение из файла приводит к ошибке (62).

Метод SkipLine

Описание

Пропускает текущую строку при чтении текстового потока.

Синтаксис

*object.**SkipLine***

object – всегда имя объекта класса TextStream.

Тип возвращаемого значения

Нет

Замечания

В документации **ошибочно написано** "Пропускает **следующую** строку".

Действия методов Skip и SkipLine не согласованы: если текущая позиция находится за символом перехода на новую строку, метод SkipLine **не пропускает строку**:

```
Sub TestSkipLine()
  Dim fs As FileSystemObject, ts As TextStream
  Const testPath As String = "c:\temp\test.txt"
  Set fs = CreateObject("Scripting.FileSystemObject")
  Set ts = fs.CreateTextFile(testPath, True)
  ts.WriteLine "12345"
  ts.WriteLine "67890"
  ts.Close
  Set ts = fs.GetFile(testPath).OpenAsTextStream(IOMode.ForReading)
  ts.Skip 6 ' Пропустить первую строку посимвольно
  ts.SkipLine
  MsgBox ts.Read(2) ' Выводит "67" - пропущена первая строка
  ts.Close
  Set ts = fs.GetFile(testPath).OpenAsTextStream(IOMode.ForReading)
  ts.SkipLine ' Пропустить первую строку целиком
  ts.SkipLine
  MsgBox ts.Read(2) ' Приводит к ошибке - пропущены обе строки
  ts.Close
End Sub
```

Метод Write

Описание

Выводит указанную строку в файл, связанный с текстовым потоком, открытым на запись (в режимах IOMode.ForWriting или IOMode.ForAppending – см. стр. 59).

Синтаксис

*object.**Write** string*

Формат вызова метода Write состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	TextStream	Да	Всегда имя объекта класса TextStream.
string	String	Да	Строковое выражение, содержащее или порождающее выводимые символы.

Тип возвращаемого значения

Нет

Замечания

Метод Write выводит строку, полученную в результате вычисления строкового выражения string, в файл, связанный с текстовым потоком object, без каких-либо разделителей, которые могли бы отделять эту строку от других, записанных этим же или иными методами в тот же файл до или после вызова метода Write. Для вывода строк, оканчивающихся переходом на новую строку, следует использовать метод WriteLine (см. стр. 98).

Попытка вывода в текстовый поток, открытый для чтения, приводит к ошибке (54).

Метод WriteBlankLines

Описание

Выводит в файл, связанный с текстовым потоком, открытым на запись (в режимах IOMode.ForWriting или IOMode.ForAppending – см. стр. 59), указанное количество символов перехода на новую строку.

Синтаксис

*object.**WriteBlankLines** lines*

Формат вызова метода WriteBlankLines состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	TextStream	Да	Всегда имя объекта класса TextStream.
lines	Long	Да	Количество символов перехода на новую строку, которое необходимо записать в файл. Неотрицательное число

Тип возвращаемого значения

Нет

Метод WriteLine

Описание

Выводит указанную строку в файл, связанный с текстовым потоком, открытым на запись (в режимах IOMode.ForWriting или IOMode.ForAppending – см. стр. 59), и добавляет символ перехода на новую строку.

Синтаксис

*object.**WriteLine*** [*string*]

Формат вызова метода WriteLine состоит из следующих компонентов ("Об." – обязательность значения):

Часть	Тип	Об.	Описание
object	TextStream	Да	Всегда имя объекта класса TextStream.
string	String	Нет	Строковое выражение, содержащее или порождающее выводимые символы.

Тип возвращаемого значения

Нет

Замечания

Символ перехода на новую строку, добавляемый методом к выводимой строке, зависит от операционной системы. Для Microsoft Windows это двухбайтовая строка vbCrLf.