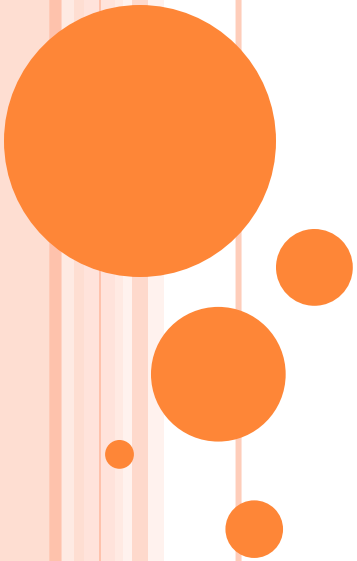


Работа с файлами

Лекция 6

Управляющие элементы

- **OpenFileDialog, *SaveFile*Dialog-** открывает диалоговое окно и предоставляет путь к файлу;
- ***Свойства:***
  - InitialDirectory имя папки, которая открывается при первом использовании окна;
  - Title - заголовок окна;
  - Filter – установка фильтра;
  - Filterlndex – № фильтра по умолчанию (если несколько фильтров).
- ***Методы:*** ShowDialog, FileName

Открытие файла для чтения

- FileOpen(номер_файла, путь, режим)
- **номер_файла** - это число от 1 до 255.
- **путь** - путь, по которому можно найти файл.
- **режим** - это ключевое слово, указывающее на то, как файл будет использоваться.
  - OpenMode.Input – чтение из файла и
  - OpenMode.Output- запись в файл)

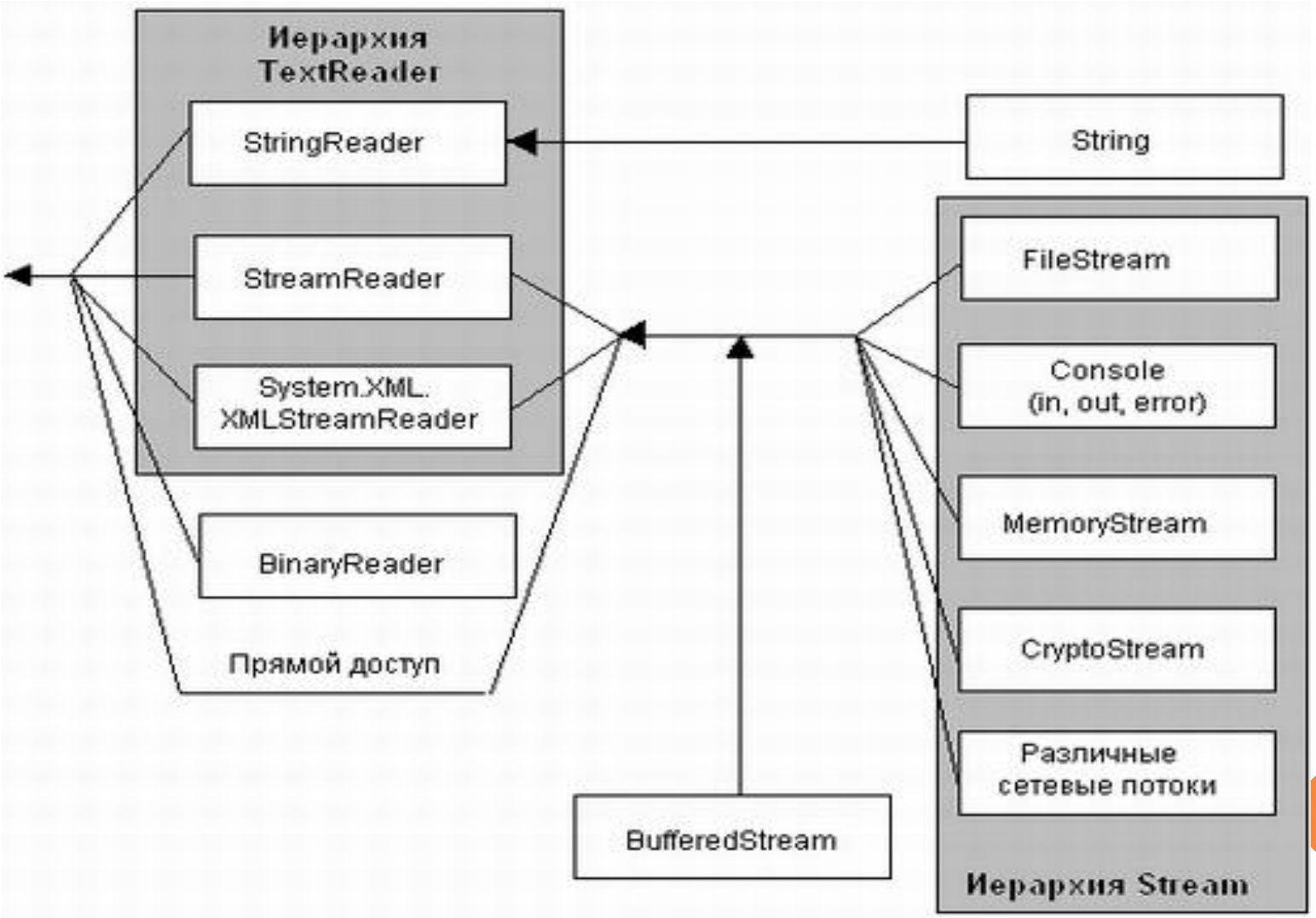
Пример

```
Dim ff, tt as string
OpenFileDialog1.Title = "Select a File"
OpenFileDialog1.Filter = "файлы (*.TXT) | *.TXT«
OpenFileDialog1.ShowDialog()
        If OpenFileDialog1.FileName <> "" Then
           Try
              FileOpen(1, OpenFileDialog1.FileName, OpenMode.Input)
              Do Until EOF(1)
         ff = LineInput(1)
                  tt = tt & ff & vbCrLf
              Loop
              tst.Text = OpenFileDialog1.FileName
              tst.Text = tt
          Catch
              MsgBox("Ошибка открытия файла.")
           Finally
              FileClose(1)    'закрываем файл
           End Try
        End If
```

Открытие файла для записи

- SaveFileDialog1.Filter = “(*.txt)|*.txt”
- SaveFileDialog1.ShowDialog()
- If SaveFileDialog1.FileName <> "" Then FileOpen(1, SaveFileDialog1.FileName,_ OpenMode.Output)
- PrintLine(1, tst.Text) FileClose(1)
- End If

Некоторые классы потоковых устройств



Класс StreamReader

Большинство классов, представляющих устройства ввода-вывода, являются производными от класса System. I0. Stream. Этот класс интерпретирует устройство как поток байтов (доступный для чтения или записи) и позволяет выполнять следующие операции:

- чтение одного или нескольких байтов данных;
- запись одного или нескольких байтов данных;
- асинхронное чтение или запись (с дополнительной возможностью оповещения о завершении операции);
- физическая запись данных из промежуточного буфера на устройство;
- переход к заданной позиции в потоке данных;
- закрытие потока (устройства) после завершения всех операций.

Методы класса TextReader

- Close- закрывает поток и освобождает системные ресурсы
- Peek- возвращает следующий символ в потоке без смещения указателя
- Read- читает один символ из потока
- ReadLine- читает строку
- ReadToEnd- читает файл целиком

Методы класса TextWrite

- Close – закрывает поток и освобождает ресурсы
- Write- записывает в поток любые базовые типы данных в текстовом формате
- Writine - записывает в поток любые базовые типы данных в текстовом формате, за которыми записывается CRLF

Класс StreamReader

- StreamReader класс из библиотеки .NET Framework, для его использования необходимо подключить библиотеку
- Imports System.IO
- Пример, вывод текстового файла в текстовое поле

```
Dim St As StreamReader
    St = New StreamReader("D:\r.txt")
    tst.Text = St.ReadToEnd
    St.Close()
    tst.Select(0, 0)
```

Запись в файл

```
Dim St As StreamWriter
    St = New StreamWriter("D:\r1.txt")
    St.Write(tst.Text)
    St.Close()
    tst.Select(0, 0)
```

Классы для работы с файлами

- **System.I0.Directory и System.IO.Directorylnfo**
- Классы предназначены для выполнения различных операций с каталогами, в том числе создания, удаления и перемещения каталогов. Кроме того, они позволяют получить или задать время создания и последней модификации каталога.

Классы для работы с файлами

- **System.IO.File и System.IO.FileInfo**
- Классы предназначены для выполнения различных операций с файлами, в том числе создания, удаления, копирования, перемещения и проверки существования файлов. Также с их помощью можно открывать файлы (функции открытия файлов возвращают объекты System. I0. FileStream, используемые при последующих операциях чтения и записи).

Классы для работы с файлами

- **System. IO.FileSystemInfo**
- Базовый класс для классов System.I0.DirectoryInfo и System. I0. FileInfo. Используется при перемещении в иерархии каталогов для получения информации о каталогах и файлах.

Классы для работы с файлами

- **System.IO.FileSystemWatcher**
- Класс предназначен для отслеживания событий файловой системы (создания, удаления и модификации файлов).

Проверка наличия файла

```
Private Function SourceFileExists() As Boolean
        If Not (System.IO.File.Exists(tst.Text)) Then
            MsgBox("The source fil e does not exist!", MsgBoxStyle.Exclamation)
        Else
            SourceFileExists = True
            Return (SourceFileExists)
        End If
    End Function
```

Копирование файлов

- Файлы копируются с помощью метода
- Copy () объекта System.IO. File
- **If Not {SourceFileExists()} Then Exit Sub**
- **System.I0.File.Copy(tst.Text, tstn.Text)**
- **MsgBox(“The file has been successfully copied.")**

Перемещение файлов

- При перемещении файла он удаляется из папки, в которой находится, и помещается в новую. При этом можно оставить ему прежнее имя, можно изменить. Перемещение файла выполняется методом Move () объекта System.IO. File.
- **If Not (SourceFileExists{)) Then Exit Sub**
- **System.I0.File.Move(tst.Text, tstn.Text}**
- **MsgBox("The file has been successfully moved.")**

Переименование файла

- Когда файл переименовывается, то с его содержимым ничего не происходит. Он остается в той же папке, изменяется только его имя. Для переименования файла используется метод Move( )
- Для этого надо указать имя файла и оставить его путь без изменений.

Удаление файлов

- Метод Delete () физически удаляет файлы
- **If Not SourceFileExists() Then Exit Sub**
- **If MsgBox (“Are you sure you want to delete the source file?", MsgBoxStyle.Question Or MsgBoxStyle.YesNo) = MsgBoxResult.Yes Then**
- **System.I0.FileDelete(tst.Text)**
- **MsgBox("The file has been successfully deleted.")**
- **End If**

Свойства файла

Сведения о файле доступные через объект File

Свойства	Описание
GetCreationTime	Возвращает дату и время создания файла
GetLastAccessTime	Возвращает дату и время последнего обращения к файлу
GetLastWriteTime	Возвращает дату и время последнего изменения файла

Работа с папками, используя объект Directory

- создать папку

System.IO.Directory.CreateDirectory("c:\my direct")

- существует ли папка

MsgBox(System.IO.Directory.Exists("c:\temp")

- переместить папку

System.IO.Directory.Move("c:\dir1","d:\dir2")

- Удалить папку
- **System.I0.Directory.Delete("c:\temp")**

Процедуры и функции

- Функция всегда возвращает некоторое значение или 0, если возвращаемое значение не указано. Процедуры никогда не возвращают значений
- x = F(Y) ' Результат используется
- F(Y) ' Результат не используется,
- F() ' Вызов функции без параметров

Процедуры и функции

- Синтаксис функции:

Function имя_функ (ByVal арг.1, арг2… as тип) as тип

Команды
……
Return выражение
End Function

Параметры передаются по значению

Хотя переменная передается по значению, функция получает копию адреса объекта, а не копию самого объекта. Иначе говоря, атрибуты ByVal и ByRef относятся к переменной, ссылающейся на объект, а не к самому объекту. При передаче по значению объекты не копируются. Передача по значению просто гарантирует, что после вызова исходная переменная будет ссылаться на прежний объект.

Передача массивов функциям

```
Function FM(BayVal a() as integer )as integer
Dim fn as integer = Ubound (a)
Dim max as integer=a(0)
For I =0 to fn
    IF a(I)>max then  max=a(i)
Next
Return max
End Function
```

Процедуры и функции с необязательным аргументом

- Для каждого необязательного параметра надо указать значение по умолчанию:

Sub PR (A as String, **Optional** ZZ as String=“01”)

Optional объявляет необязательный параметр

- Функции с произвольным количеством аргументов:

Function AD (BayVal **ParamArray** ST() AS Double) as Dbl

Вызов: X= PR(3,4,5,6)