Diseño Basado en Componentes

Excepciones en VB.NET

Ingeniería Informática
Universidad Carlos III de Madrid

Gestión de errores del .NET Framework
Errores y excepciones (I)

- **Error**: Evento que se produce durante la ejecución de un programa, provocando una interrupción en su flujo normal de ejecución. Genera un **objeto excepción**.
- **Excepción**: Objeto generado por un error, que contiene información sobre el error que se ha producido.
- La **gestión de errores** proporciona un mecanismo que permite controlar programas que tengan muchas características dinámicas durante su ejecución.

Gestión de errores del .NET Framework
Errores y excepciones (II)

- **Manipulador de excepciones**: Bloque de código que proporciona una respuesta al error que se ha producido.
- **Tipos de tratamiento de errores en VB.NET**:
  - Estructurado
  - No estructurado

Manipulación estructurada de errores
Estructura Try...End Try (I)

- Cada vez que se produce un error, se genera un objeto de la clase **Exception** o alguna de sus derivadas (hijas).
- La estructura **Try...End Try** proporciona el medio para definir un bloque de código sensible a errores, y los correspondientes manipuladores de excepciones.

```
Try
        ' Código que puede provocar errores
        ' ....
[Catch [Excepcion [As TipoExcepcionA]] [When Expresión]
        ' Respuesta a error de tipo A
        ' ....
        [Exit Try]
]
[Catch [Excepcion [As TipoExcepcionN]] [When Expresión]
        ' Respuesta a error de tipo N
        ' ....
        [Exit Try]
]
[Finally
        ' Código posterior al control de errores
        ' ....
]
End Try
```

Manipulación estructurada de errores
Estructura Try...End Try (II)

```
'Option Strict On
Dim sValor As String
Dim iNumero As Integer
Try
    ' Comienza el control de errores
    Console.WriteLine("Introducir un número:")
    sValor = Console.ReadLine()
    ' Si no hemos introducido un número...
    iNumero = CInt(sValor) ' ...aquí se producirá un error...
    ' ...y no llegaremos a esta parte del código
    iNumero = iNumero + 1000
Catch
    ' Si se produce un error, se genera una excepción
    ' que capturamos en este bloque de código
    ' manipulador de excepción, definido por Catch
    Console.WriteLine("Error al introducir el número." & _
    ControlChars.CrLf & "El valor '{0}' es incorrecto.", sValor)
```


Manipulación estructurada de errores
Estructura Try...End Try (III)

```
Finally
    ' Si se produce un error, después de Catch se ejecuta este bloque;
    ' Si no se produce error, después de Try también se ejecuta.
    Console.WriteLine("El controlador de errores ha finalizado.")
End Try
' Resto del código del procedimiento.
Dim dtFecha As Date
Console.WriteLine("Introducir una fecha: ")
sValor = Console.ReadLine()
' Si ahora se produce un error,
' al no disponer de una estructura para controlarlo
' se cancelará la ejecución.
dtFecha = CDate(sValor)
Console.WriteLine("La fecha es: {0}", dtFecha)
Console.ReadLine()
```


Manipulación estructurada de errores
Clase Exception (I)

- **Sentencia Catch**: se escribe el nombre de un identificador, definiéndolo como tipo **Exception** o alguno de los tipos de su jerarquía (*OverFlowException*, *IndexOutOfRangeException*...).

  Catch Excepcion As TipoExcepcionN

- **Message**: Descripción del error.
- **Source**: Nombre del objeto o aplicación que provocó el error.
- **StackTrace**: Ruta o traza del código en la que se produjo el error.
- **TargetSite**: Método que provocó el error.
- **ToString()**: Cadena con información detallada del error.

Manipulación estructurada de errores
Clase Exception (II)

```
' ....
Try
    ' ....
Catch oExcep As Exception
    ' Si se produce un error, se crea un objeto excepción que capturamos volcándolo
    ' a un identificador de tipo Exception
    Console.WriteLine("Se produjo un error. Información de la excepción")
    Console.WriteLine("================================================")
    Console.WriteLine("Message: {0}", oExcep.Message)
    Console.WriteLine("Source: {0}", oExcep.Source)
    Console.WriteLine("StackTrace: {0}", oExcep.StackTrace)
    Console.WriteLine(oExcep.ToString())
Finally
    ' ....
End Try
' ....
```

Manipulación estructurada de errores
Condición para un manipulador de excepciones

- **Sentencia Catch**: se añade cláusula **When** para situar una expresión que sirva como filtro o condición.

```
Dim byMiNum As Byte
Dim dtFHActual As Date
' Obtener la fecha actual
dtFHActual = System.DateTime.Today()
Try
        Console.WriteLine("Introducir un número:")
        ' Si introducimos un número no incluido en el rango de Byte...
        byMiNum = CByte(Console.ReadLine()) 'Considerando Strict On
Catch oExcep As OverflowException When (dtFHActual.Month = 3)
        ' ...saltará este manipulador de excepciones, pero sólo
        ' cuando las excepciones de desbordamiento
        ' se produzcan en el mes de marzo.
        Console.WriteLine("El número introducido " & _
        "no se encuentra en el rango adecuado (0-255).")
End Try
```


Manipulación estructurada de errores
Forzar la salida de un controlador de errores

- **Exit Try**: se sale del controlador de errores sin finalizarlo.
  - Si hay bloque Finally se ejecuta en cualquier caso.

```
Dim iMiNum As Integer
Dim aValores() As String = {"a", "b", "c"}
Try
    Console.WriteLine("Introducir un número: ")
    ' Si no es un número Byte se produce error.
    iMiNum = CInt(Console.ReadLine())
    ' Salimos de controlador de errores sin finalizarlo.
    Exit Try
    ' Esta línea produce error siempre, ya que no existe el índice 5 en el array.
    aValores(5) = "d"
Catch oExcep As OverflowException
    ' ....
Catch oExcep As Exception
    ' ....
End Try
' ....
```


Manipulación no estructurada de errores
Objeto Err

- Se crea automáticamente al iniciarse la aplicación.
- Ámbito público.
- Proporciona información sobre los errores ocurridos en el transcurso de la aplicación.
- Al producirse un error, la propiedad **Number** tendrá un valor mayor que cero.
- Al producirse un error, la propiedad **Description** dará información textual del error producido.

Manipulación no estructurada de errores
On Error (I)

- **On Error**: Instrucción que activa o desactiva una rutina de control de errores.
  - **On Error Goto** *Etiqueta*: activa una etiqueta de control de errores (**Bloque de código**), al que se desviará el flujo del programa cuando se produzca un error.

```
On Error Goto ControlErrores
  Dim dtFecha As Date
  dtFecha = "Valor incorrecto"
  '...
' Etiqueta de control de errores
ControlErrores:
  Console.WriteLine("Error: {0} - {1}", Err.Number, Err.Description)
  Console.ReadLine()
```

Manipulación no estructurada de errores
On Error (II)

- **On Error Resume Next**: hace que la ejecución continúe con la siguiente línea de código.
  - Para la captura de errores: consultar propiedades del objeto **Err**, y posteriormente, inicializar el **Err** llamando a su método Clear().
- **On Error Goto 0**: desactiva el controlador de errores que hubiera activado hasta el momento.

Gestión de errores del .NET Framework
Lanzamiento de errores (I)

- **Throw**: se utilizar para lanzar explícitamente un error. Genera un objeto del tipo excepción que se indique.

```
Public Sub AsignarCredito(ByVal ldbCredito As Double)
    If ldbCredito > 2500 Then
        Throw New CreditoException("Límite disponible: 2500 " & _
        "- Se intentó asignar " & CType(ldbCredito, String))
    Else
        mdbDisponible = ldbCredito
    End If
End Sub
```

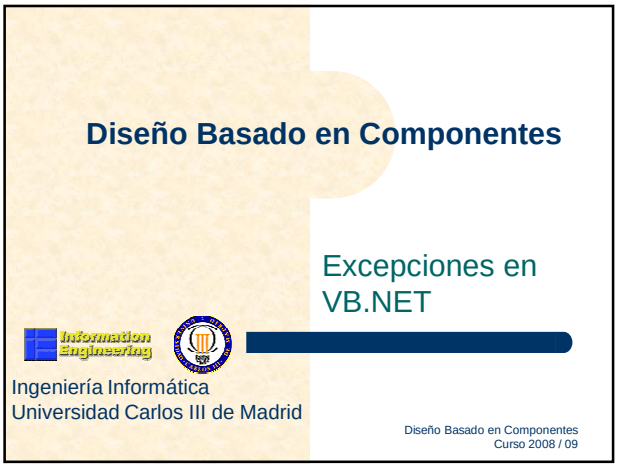

Gestión de errores del .NET Framework
Lanzamiento de errores (II)

```
Public Class CreditoException
Inherits Exception
    Private msDescripcion As String
    Public Sub New(ByVal lsDescripcion As String)
        msDescripcion = lsDescripcion
    End Sub
    Public ReadOnly Property Descripcion() As String
        Get
            Return msDescripcion
        End Get
    End Property
End Class
```


Gestión de errores del .NET Framework
Creación de errores con el objeto Err

- **Método Raise()**: permite generar nuestros propios errores.

```
    On Error Goto ControlErrores
    Dim iValor As Integer
    Console.WriteLine("Introducir un número")
    iValor = CInt(Console.ReadLine())
    If iValor > 500 Then
      Err.Raise(5100, , "El número debe ser menor de 500")
    End If
    Console.WriteLine("Esta línea se ejecuta después de un posible error")
    Console.ReadLine()
    ' ---
' etiqueta de control de errores
ControlErrores:
Console.WriteLine("Error: {0} - {1}", Err.Number, Err.Description)
Console.ReadLine()
Resume Next
```


Diseño Basado en Componentes
Excepciones en VB.NET
Information Engineering
Ingeniería Informática
Universidad Carlos III de Madrid
Diseño Basado en Componentes
Curso 2008 / 09