



Departamento de Lenguajes y
Sistemas Informáticos



Universitat d'Alacant
Universidad de Alicante

ASP

Programación en Internet
Curso 2005-2006

Programación en Internet – Curso 2005-2006

Índice

- Introducción
- Versiones
- Qué necesito para programar
- Programación
- Directivas de procesamiento
- Objetos
- Acceso a bases de datos
- Mensajes de error
- Trucos

2

Introducción

- *Active Server Pages*
- Tecnología de Microsoft:
 - Personal Web Server
 - Internet Information Server
- Terceras partes:
 - Sun Chili!Soft ASP → Linux y Solaris
 - Instant ASP → Linux, Solaris, HP-UX, ...

3

Introducción

- Entorno de programación en el servidor
- Permite crear páginas web dinámicas
- Acceder a bases de datos



Aplicaciones web

4

Introducción

- Gratuito, sobre licencias de los sistemas operativos Microsoft Windows
- Podemos utilizar cualquier base de datos con el controlador ODBC de 32 bits apropiado:
 - Access
 - SQL Server
 - Oracle
 - Informix
 - ...

5

Versiones

- ASP 1.0: IIS 3.0 (Windows NT 4.0)
- ASP 2.0: IIS 4.0 (Windows NT 4.0 + Option Pack) y PWS (Windows 9x)
- ASP 3.0: IIS 5.0 (Windows 2000), IIS 5.1 (Windows XP)
- Sustituto: ASP+ → ASP.NET

6

Introducción

- Se puede programar en varios lenguajes:
 - VBScript
 - JavaScript (JScript)
 - ...
- Acceso a componentes ActiveX:
 - Amplia posibilidades y prestaciones
 - Suelen ser de pago

7

Qué necesito para programar (I)

- Editor ASCII estándar
- Servidor web que acepte ASP
- Navegador
- Y si accedemos a una base de datos:
 - Sistema Gestor de Bases de Datos
 - Controlador ODBC

8

Qué necesito para programar (y II)

- Entornos de programación que proporcionan soporte a la tecnología ASP:
 - Microsoft FrontPage 2000
 - Microsoft Visual Interdev
 - Macromedia Dreamweaver MX
 - Adobe GoLive

9

Programación

- El código ASP se mezcla con HTML
- Compatibilidad con todos los clientes:
 - El código ASP se interpreta en el servidor, lo único que ve el cliente es HTML
- .html → .asp
 - Si una página web contiene código ASP, tiene que tener la extensión .asp
 - En otro caso, si sólo contiene código de cliente (HTML y JavaScript), es preferible dejar la extensión .html (por razones de eficiencia)

10

Programación

- Delimitadores: `<% ... %>`
 - Para encerrar instrucciones de servidor
 - Utilizan el lenguaje principal
 - Predeterminado :”VBScript”

- Otra posibilidad:

```
<SCRIPT LANGUAGE="VBScript"  
  RUNAT="Server">
```

...

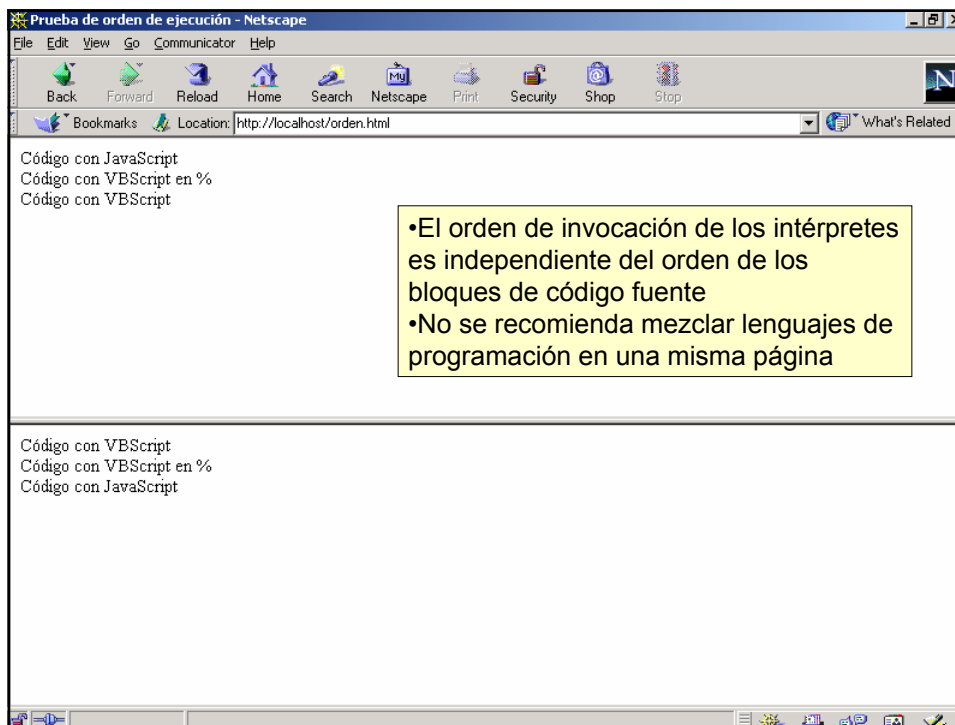
```
</SCRIPT>
```

11

Programación

```
<HTML>  
<BODY>  
<SCRIPT LANGUAGE="VBScript" RUNAT="Server">  
Response.Write "Código con VBScript<BR>"  
</SCRIPT>  
<% Response.Write "Código con VBScript en %<BR>"  
  %>  
<SCRIPT LANGUAGE="JavaScript" RUNAT="Server">  
Response.Write("Código con JavaScript<BR>");  
</SCRIPT>  
</BODY>  
</HTML>
```

12

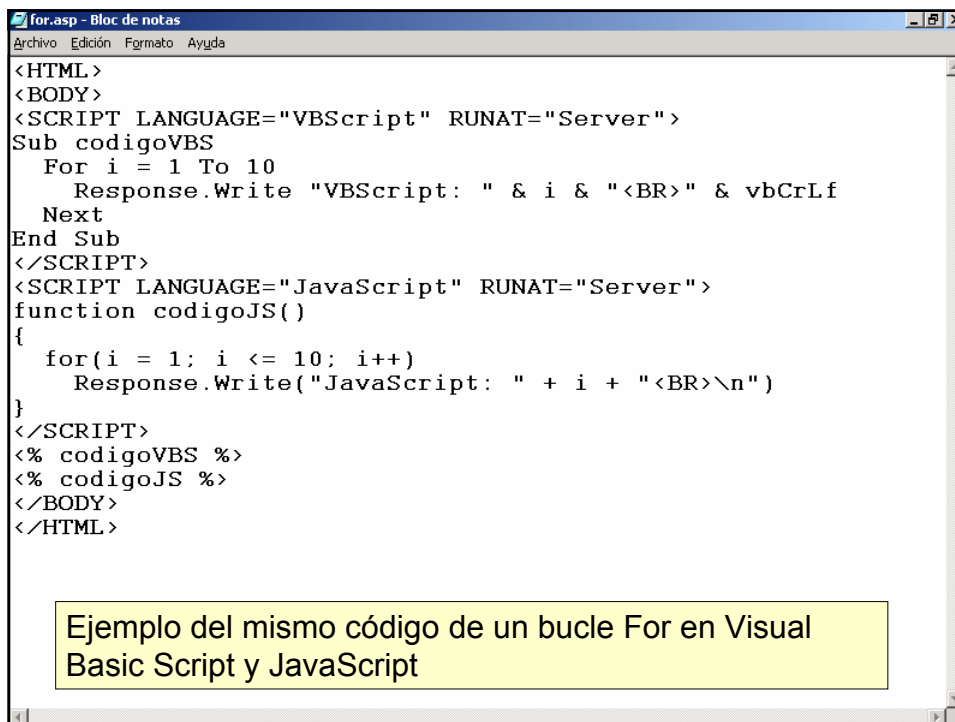


Programación en Internet – Curso 2005-2006

Programación

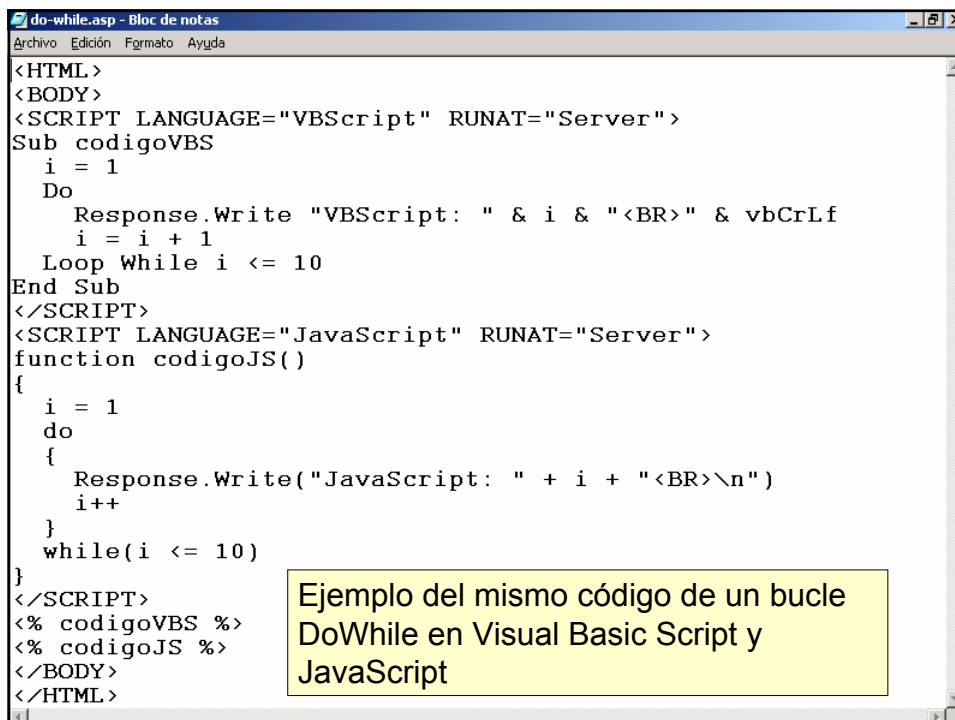
```
<HTML><BODY>  
<SCRIPT LANGUAGE="VBScript" RUNAT="Server">  
Sub codigoVBS  
  Response.Write "Código con VBScript<BR>"  
End Sub  
</SCRIPT>  
<SCRIPT LANGUAGE="JavaScript" RUNAT="Server">  
function codigoJS()  
{ Response.Write("Código con JavaScript<BR>"); }  
</SCRIPT>  
<% codigoVBS %>  
<% Response.Write "Código con VBScript en %<BR>" %>  
<% codigoJS %>  
</BODY></HTML>
```

14



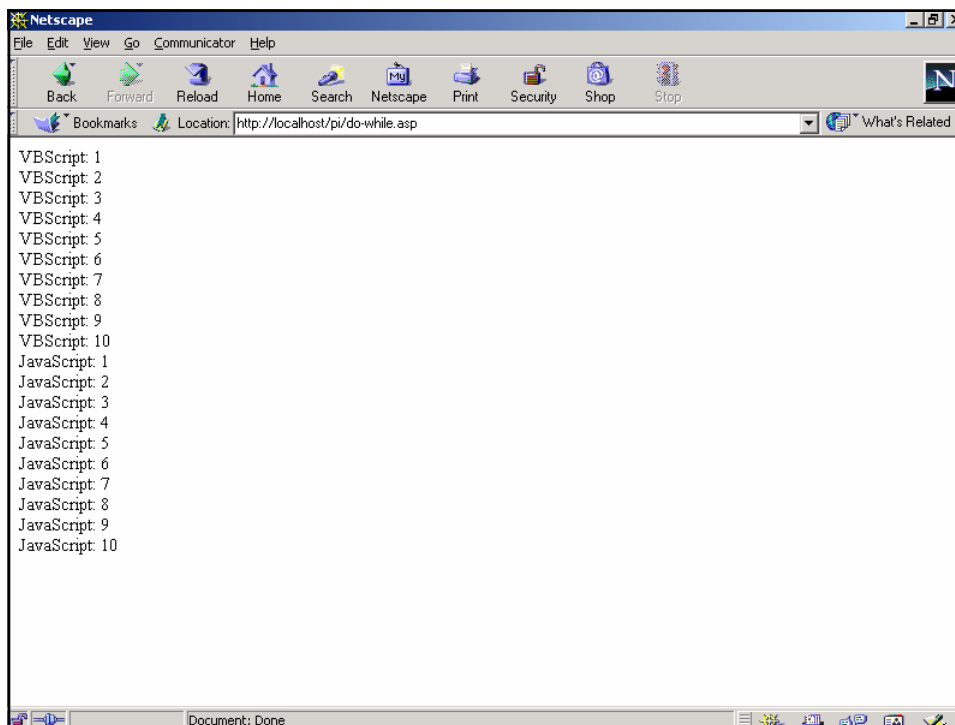
```
<HTML>
<BODY>
<SCRIPT LANGUAGE="VBScript" RUNAT="Server">
Sub codigoVBS
  For i = 1 To 10
    Response.Write "VBScript: " & i & "<BR>" & vbCrLf
  Next
End Sub
</SCRIPT>
<SCRIPT LANGUAGE="JavaScript" RUNAT="Server">
function codigoJS()
{
  for(i = 1; i <= 10; i++)
    Response.Write("JavaScript: " + i + "<BR>\n")
}
</SCRIPT>
<% codigoVBS %>
<% codigoJS %>
</BODY>
</HTML>
```

Ejemplo del mismo código de un bucle For en Visual Basic Script y JavaScript



```
<HTML>
<BODY>
<SCRIPT LANGUAGE="VBScript" RUNAT="Server">
Sub codigoVBS
  i = 1
  Do
    Response.Write "VBScript: " & i & "<BR>" & vbCrLf
    i = i + 1
  Loop While i <= 10
End Sub
</SCRIPT>
<SCRIPT LANGUAGE="JavaScript" RUNAT="Server">
function codigoJS()
{
  i = 1
  do
  {
    Response.Write("JavaScript: " + i + "<BR>\n")
    i++
  }
  while(i <= 10)
}
</SCRIPT>
<% codigoVBS %>
<% codigoJS %>
</BODY>
</HTML>
```

Ejemplo del mismo código de un bucle DoWhile en Visual Basic Script y JavaScript



Programación en Internet – Curso 2005-2006

Directivas de procesamiento (I)

- Permiten enviar información a IIS acerca de cómo procesar un archivo .asp
- Las directivas tienen que aparecer al principio de la página
- Sólo puede aparecer una de cada tipo
- Sintaxis:

```
<%@ directiva="valor" %>
```

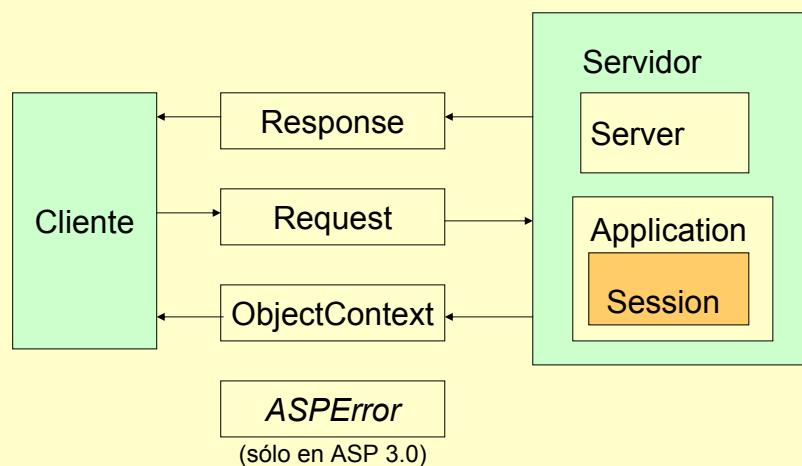
Directivas de procesamiento (y II)

- Directivas:
 - CODEPAGE: página de códigos de las cadenas literales
 - ENABLESESSIONSSTATE: seguimiento de sesiones
 - LANGUAGE: lenguaje de programación de servidor empleado
 - LCID: identificador de configuración regional (formato fechas, horas y moneda)
 - TRANSACTION: tratamiento de la ejecución de una página como una transacción
- En una misma línea se pueden escribir varias directivas:

```
<%@ LCID=1041 LANGUAGE="VBScript" %>
```

19

Objetos



20

Objeto Application

- Eventos:
 - Application_OnStart
 - Application_OnEnd
- Colecciones:
 - Contents
 - StaticObjects
- Métodos:
 - Lock
 - UnLock

21

Objeto Session (I)

- Eventos:
 - Session_OnStart
 - Session_OnEnd
- Colecciones:
 - Contents
 - StaticObjects
- Métodos:
 - Abandon

22

Objeto Session (y II)

- Propiedades:
 - CodePage
 - LCID
 - SessionId
 - Timeout

23

Objeto Server

- Propiedades:
 - ScriptTimeout
- Métodos:
 - CreateObject
 - HTML Encode
 - MapPath
 - URLEncode
 - *Execute, GetLastError, Transfer* → Nuevos en ASP 3.0

24

Objeto Request

- Colecciones:
 - ClientCertificate
 - Cookies
 - Form
 - QueryString
 - ServerVariables
- Propiedades:
 - TotalBytes
- Métodos:
 - BinaryRead

25

Objeto Response (I)

- Colecciones:
 - Cookies

26

Objeto Response (II)

- Propiedades:
 - Buffer
 - CacheControl
 - Charset
 - CodePage
 - ContentType
 - Expires
 - ExpiresAbsolute
 - IsClientConnected
 - LCID
 - PICS
 - Status

27

Objeto Response (y III)

- Métodos:
 - AddHeader
 - AppendToLog
 - BinaryWrite
 - Clear
 - End
 - Flush
 - Redirect
 - Write

28

Objeto ObjectContext

- Eventos:
 - OnTransactionCommint
 - OnTransactionAbort
- Métodos:
 - SetComplete
 - SetAbort

29

Objeto Application (I)

- Eventos:
 - Application_OnStart
 - Application_OnEnd
- Colecciones:
 - Contents
 - StaticObjects
- Métodos:
 - Lock
 - UnLock

30

Objeto Application (II)

- Compartir información entre todos los usuarios de una aplicación → Variables globales

```
<%  
Application("Email") = "a@b.es"  
%>  
...  
<a href="mailto:<% = Application("Email") %>">  
Webmaster</a>
```

31

Objeto Application (III)

- Bloquear el objeto (acceso exclusivo)

```
<%  
  Visitas = Application("Visitas")  
  Application("Visitas") = Visitas + 1  
%>  
...  
Eres el visitante número: <% = Visitas %>
```

Se pueden producir condiciones de carrera. Por ejemplo:

1. Llega el visitante A y obtiene Visitas = 5
2. Llega el visitante B y obtiene Visitas = 5
3. El visitante A realiza Application("Visitas") = 5 + 1 = 6
4. El visitante B realiza Application("Visitas") = 5 + 1 = 6
5. Al final, el número de visitantes es 6, cuando en realidad debería de ser 7

32

Objeto Application (IV)

- Bloquear el objeto (acceso exclusivo)

```
<%  
    Application.Lock  
    Visitas = Application("Visitas")  
    Application("Visitas") = Visitas + 1  
    Application.Unlock  
%>  
...  
Eres el visitante número: <% = Visitas %>
```

33

Objeto Application (V)

- Parametrizar la aplicación

```
Set conexion =  
    Server.CreateObject("ADODB.Connection")  
dsn = "DRIVER=Microsoft Access Driver (*.mdb);"  
dsn = dsn & "DefaultDir=E:\Web\almacen\bd;"  
dsn = dsn & "DBQ=E:\Web\almacen\bd\almacen.mdb"  
conexion.Open dsn  
...  
Set conexion =  
    Server.CreateObject("ADODB.Connection")  
conexion.Open Application("DSN")
```

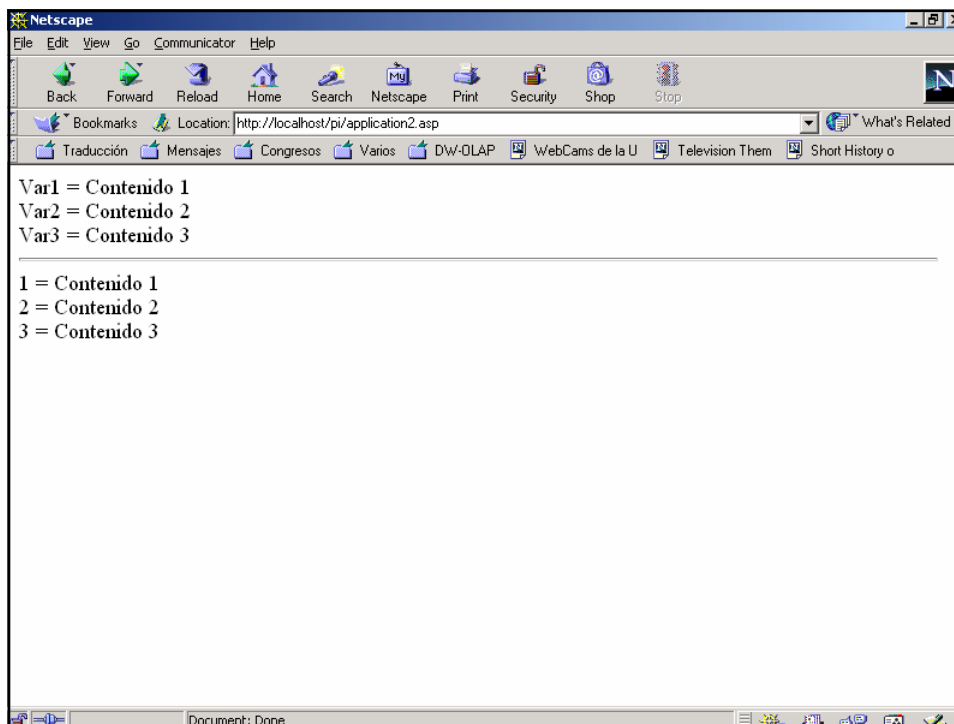
34

Objeto Application (y VI)

- Obtener todas las variables declaradas

```
Application("Var1") = "Contenido 1"  
Application("Var2") = "Contenido 2"  
Application("Var3") = "Contenido 3"  
For Each var in Application.Contents  
    Response.Write var & " = " &  
    Application.Contents(var)  
    Response.Write "<br>"  
Next  
Response.Write "<hr>"  
For i = 1 To Application.Contents.Count  
    Response.Write i & " = " &  
    Application.Contents(i)  
    Response.Write "<br>"  
Next
```

35



Objeto Session (I)

- Eventos:
 - Session_OnStart
 - Session_OnEnd
- Colecciones:
 - Contents
 - StaticObjects
- Métodos:
 - Abandon

37

Objeto Session (II)

- Propiedades:
 - CodePage
 - LCID
 - SessionID
 - Timeout

38

Objeto Session (III)

- Almacena información para un usuario concreto → Variables locales
- La información se mantiene durante la sesión de trabajo del usuario en el servidor y se mantiene al pasar de una página a otra
- La sesión finaliza cuando:
 - Automáticamente: caduca o expira (según la configuración establecida)
 - Manualmente: se abandona

39

Objeto Session (IV)

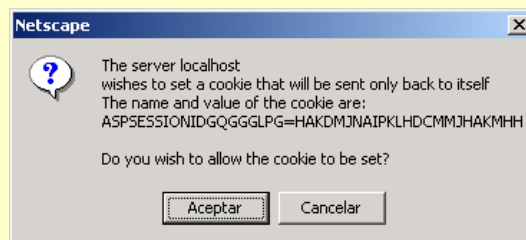
- Información individual de un usuario:

```
<%  
  ' Página 1  
  Session("nombre") = "Jose"  
  Session("idioma") = "es"  
%>  
...  
<%  
  ' Página 2  
  If Session("idioma") = "es" Then  
    Response.Write "Hola, " & Session("nombre")  
  Else  
    Response.Write "Hello, " & Session("nombre")  
  End If  
%>
```

40

Objeto Session (V)

- Requisito imprescindible: *cookies*
- `Session.SessionID`



41

Objeto Session (VI)

- Las variables se manejan como en el objeto `Application`

```
For Each var in Session.Contents
    Response.Write var & " = " &
    Session.Contents(var) & "<br>"
Next
```

42

Objeto Session (y VII)

- `Session.Timeout`: tiempo de espera, caducidad (minutos, valor por defecto 20) → Período de inactividad de una sesión, pasado el cual la sesión se destruye y se liberan los recursos
- `Session.Abandon`: destruye el objeto y libera sus recursos (cuando la página acaba su ejecución)

43

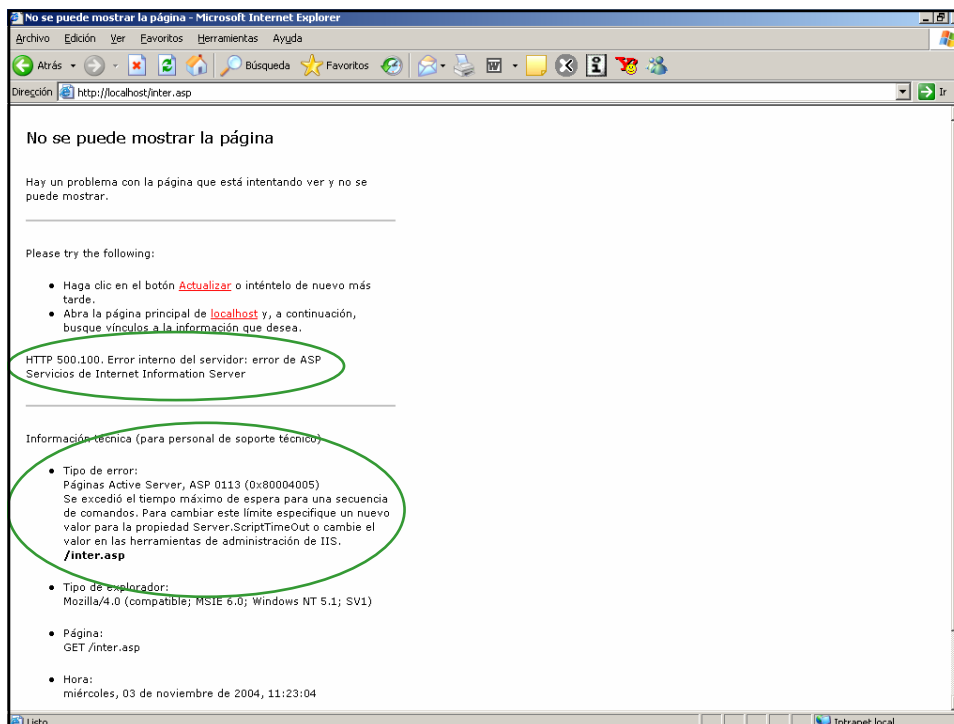
Objeto Server (I)

- Propiedades:
 - `ScriptTimeout`
- Métodos:
 - `CreateObject`
 - `HTMLEncode`
 - `MapPath`
 - `URLEncode`
 - *Execute, GetLastError, Transfer* → Nuevos en ASP 3.0

44

Objeto Server (II)

- Métodos y propiedades que proporciona el servidor
- `Server.ScriptTimeout`: tiempo máximo de ejecución (segundos, valor por defecto 90)



Objeto Server (III)

- Creación de objetos a partir de componentes

```
Server.CreateObject(idObjeto)
```

```
Server.CreateObject("ADODB.Connection")
```

```
Server.CreateObject("MSWC.AdRotator")
```

```
Server.CreateObject("MiDLL.MiObjeto")
```

47

Objeto Server (IV)

- Componente *Browser Capabilities*

```
Set bc = Server.CreateObject("MSWC.BrowserType")
```

```
Response.Write "Navegador: " & bc.browser & "<br>"
```

```
Response.Write "Versión: " & bc.version & "<br>"
```

```
Response.Write "Marcos: " & bc.frames & "<br>"
```

```
Response.Write "Tablas: " & bc.tables & "<br>"
```

```
Response.Write "VBScript: " & bc.vbscript & "<br>"
```

```
Response.Write "JavaScript: " & bc.javascript & "<br>"
```

48

Objeto Server (V)

- `Server.HTMLEncode`: **codificación HTML**

```
Server.HTMLEncode("España <b>""va bien""</b>")
```

```
Espa&#241;a &lt;b>&quot;va bien&quot;&lt;/b>;
```

- `Server.URLEncode`: **codificación URL**

```
Server.URLEncode("http://www.ua.es")
```

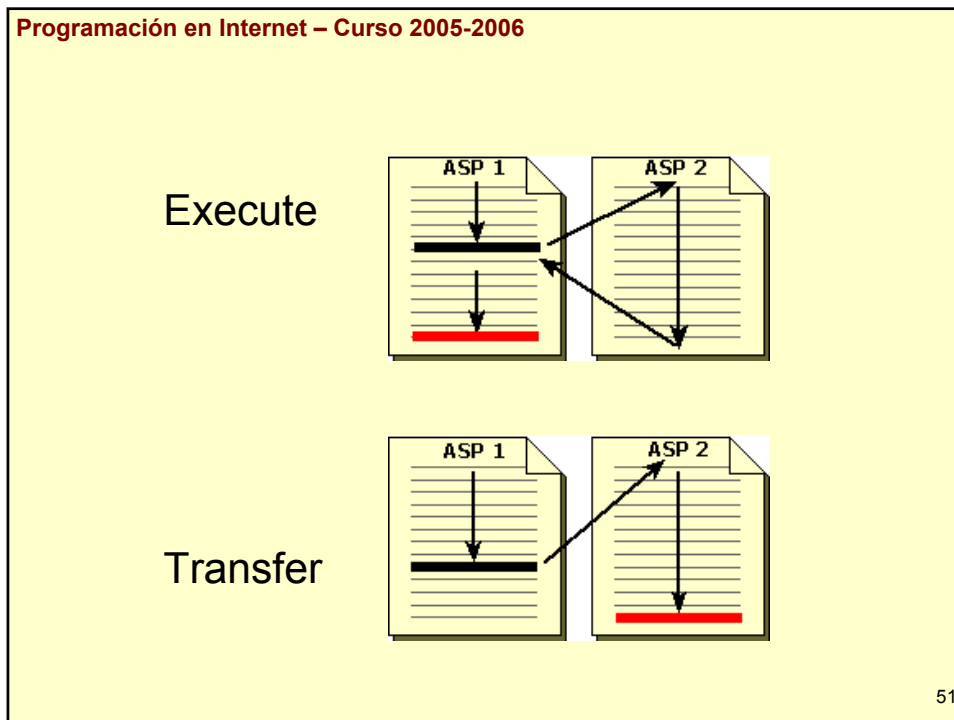
```
http%3A%2F%2Fwww%2Eua%2Ees
```

49

Objeto Server (y VI)

- `Server.Execute`
 - Similar a la ejecución de un procedimiento (retorna al finalizar)
- `Server.Transfer`
 - Similar a `Response.Redirect` (no retorna al finalizar), pero no inicia una nueva petición
 - Conserva los objetos integrados, incluidos los valores recibidos del cliente (formulario, etc.)

50



- Programación en Internet – Curso 2005-2006
- ## Objeto Request (I)
- Colecciones:
 - ClientCertificate
 - Cookies
 - Form
 - QueryString
 - ServerVariables
 - Propiedades:
 - TotalBytes
 - Métodos:
 - BinaryRead
- 52

Objeto Request (II)

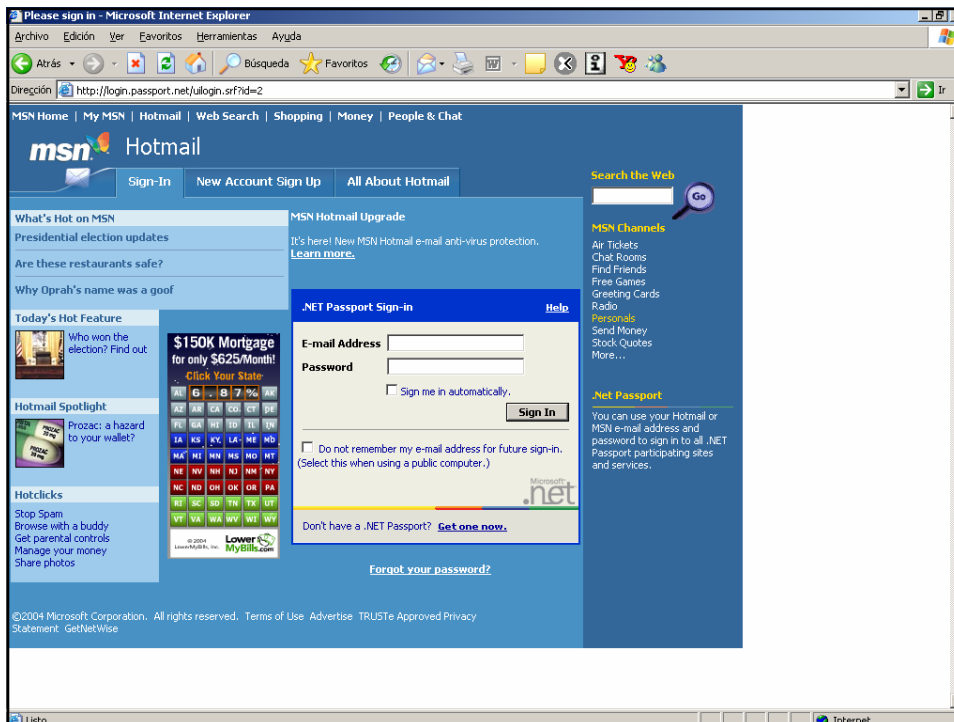
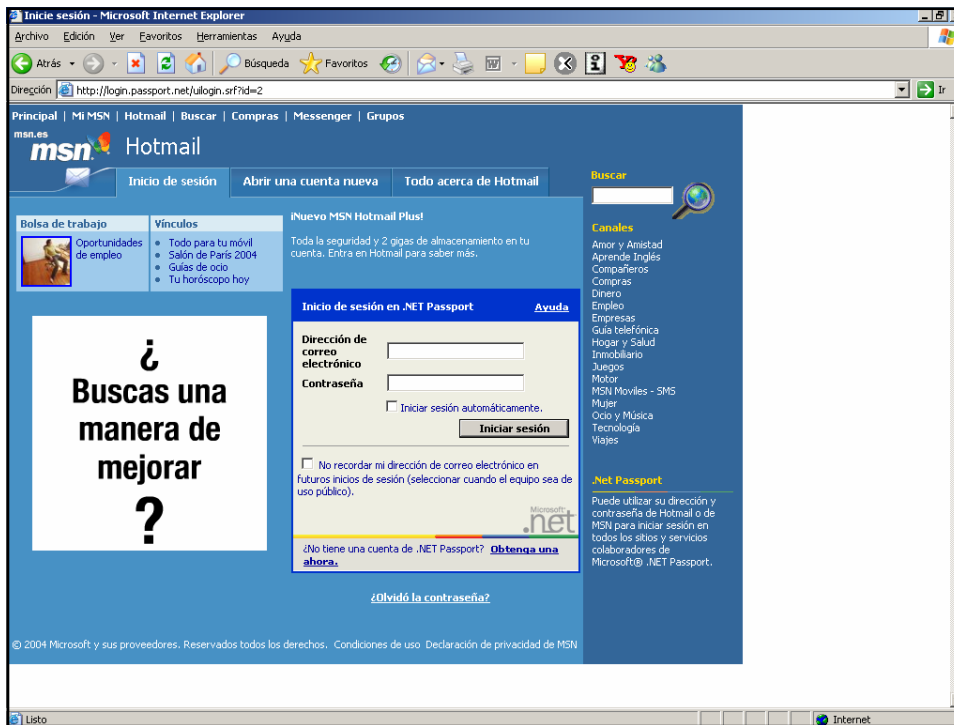
- Recupera los valores que envía el navegador al servidor en una petición HTTP
- `Request.ServerVariables`: variables de entorno (como en CGI):
 - `CONTENT_LENGTH`
 - `CONTENT_TYPE`
 - `PATH_INFO`
 - `QUERY_STRING`
 - ...

53

Objeto Request (III)

- Variable `HTTP_ACCEPT_LANGUAGE`: contenido de la cabecera `Accept-Language`: de la petición actual
→ Permite conocer las preferencias respecto al idioma del usuario y mostrar las páginas con el idioma correspondiente
- Estandarizado por ISO (639-1):
 - `es`: Español
 - `es-mx`: Español/Mexico
 - `es-ni`: Español/Nicaragua
 - `es-pr`: Español/Puerto Rico
 - `ca`: Catalán
 - `en`: Inglés
 - `fr`: Francés
 - ...
 - `zu`: Zulu
- <http://www.loc.gov/standards/iso639-2/englangn.html>

54





Programación en Internet – Curso 2005-2006

Preferencias de idioma

Algunos sitios Web ofrecen su contenido en múltiples idiomas. A continuación se le muestran los disponibles en orden de prioridad.

Idioma:

- Español - España (alfabetizaci [es]
- Español - España (alfabetizaci [es-es]
- Inglés (Estados Unidos) [en-us]

Subir

Bajar

Quitar

Agregar...

Los menús y cuadros de diálogo se muestran actualmente en Español - España (alfabetizaci.

Aceptar Cancelar

$es, es-es; q=0.7, en-us; q=0.3$

↙ ↘

Código idioma ISO

↙ ↘

Preferencia del lenguaje: 0 (menor), 1 (mayor)

58

Objeto Request (IV)

- Dos opciones:
 - Las páginas están disponibles en dos idiomas: español e inglés
 - “Si es está en la lista, el usuario sabe español y le mostramos la página en español” → Buscar un idioma en la lista (`InStr`)
 - “El usuario sólo sabe español si es el primer idioma de la lista” → Obtener los primeros caracteres en la lista (`Left`)

59

Objeto Request (V)

- `Request.Form`: valores de un formulario mediante POST

```
<input type="text" name="nombre">  
<input type="text" name="apellidos">
```

```
Request.Form("nombre")
```

```
Request.Form("apellidos")
```

```
` Devuelve los datos de envío sin procesar
```

```
Request.Form
```

60

Objeto Request (VI)

- `Request.QueryString`: valores de un formulario mediante GET o directamente en una URL

```
<input type="text" name="nombre">  
pagina.asp?id=3
```

```
Request.QueryString("nombre")  
Request.QueryString("id")  
` Devuelve los datos de envío sin procesar  
Request.QueryString
```

61

Objeto Request (VII)

- Campos escalares y vectoriales (colecciones) recibidos desde un formulario:

Escalar:

```
Response.Write Request.Form("lista") &  
"<br>"
```

Vectorial:

```
For i = 1 To Request.Form("lista").Count  
    Response.Write i & ": " &  
    Request.Form("lista")(i) & "<br>"  
Next
```

62

Programación en Internet – Curso 2005-2006

```
<html>
<body>
<%
Response.Write("el form:" & Request.Form & "<br>")
Response.Write("núm. checks:" &
Request.Form("ch").Count & "<br>")
For i=1 to Request.Form("ch").Count
Response.Write(i & " = " & Request.Form("ch")(i) &
"<br>")
Next
%>
<form name="form1" action="p.asp" method="post">
uno: <input type="checkbox" name="ch" value="1">
dos: <input type="checkbox" name="ch" value="2">
tres: <input type="checkbox" name="ch" value="3">
<br>
<input type="submit" name="s" value="Envío">
</form>
</body>
</html>
```

63

Programación en Internet – Curso 2005-2006

Objeto Request (y VIII)

- Acceso directo a todas las variables:
Request("variable")
- Orden de búsqueda en las colecciones:
 - QueryString
 - Form
 - Cookies
 - ClientCertificate
 - ServerVariables
- Si existe una variable con el mismo nombre en más de una colección, devuelve la primera instancia que encuentra
- Recomendable: utilizar el nombre completo
 - Evita problemas (la misma variable repetida)
 - Más rápido (no tiene que buscar en varias colecciones)

64

Objeto Response (I)

- Colecciones:
 - Cookies

65

Objeto Response (II)

- Propiedades:
 - Buffer
 - CacheControl
 - Charset
 - CodePage
 - ContentType
 - Expires
 - ExpiresAbsolute
 - IsClientConnected
 - LCID
 - PICS
 - Status

66

Objeto Response (III)

- Métodos:
 - AddHeader
 - AppendToLog
 - BinaryWrite
 - Clear
 - End
 - Flush
 - Redirect
 - Write

67

Objeto Response (IV)

- Envía la respuesta, el resultado del ASP al navegador
- `Response.Buffer`: indica si el resultado se almacena en un buffer
 - PWS 4.0 y IIS 4.0: False
 - \geq IIS 5.0: True
- `Response.Clear`: limpia el buffer
- `Response.End`: finaliza la ejecución
- `Response.Flush`: envía el buffer al cliente
 - `Response.Clear` y `Response.Flush` producen error si el buffer no está a True

68

Objeto Response (V)

- `Response.Write`: escribe en el resultado (`%> → %\>`)
 - `Response.Write exp → <% = exp %>`
- `Response.Redirect`: redirige el navegador a una URL (finaliza la ejecución y envía un mensaje al cliente para que se dirija a la URL) → Es un encabezado HTTP: ¡cuidado!
 - Mejor `Server.Execute` y `Server.Transfer`: minimizan la comunicación

69

Objeto Response (VI)

- `Response.CodePage`: Si no se indica, toma el valor de `Session.CodePage`; si no hay sesiones toma el valor de `@CODEPAGE`; si no toma el valor de la metabase de IIS
- `Response.LCID`: Si no se indica, toma el valor de `Session.LCID`; si no hay sesiones toma el valor de `@LCID`; si no toma el valor de la metabase de IIS
- `Response.PICS`: *Platform for Internet Content Selection* → Asesor de contenidos
 - Internet Content Rating Association (ICRA): <http://www.icra.org>

70

Objeto Response (VII)

- `Response.AppendToLog`: Añade información al registro del sitio web
- Se puede llamar varias veces en una misma página (petición), pero sólo se añade una cadena al registro
- Para que funcione:
 - Habilitar registro
 - Propiedades extendidas
 - Consulta (URI) solicitada
- Utilidad: registrar eventos especiales o errores de la aplicación

71

Objeto Response (y VIII)

```
<%
  Response.AppendToLog "Información del usuario:"
  Response.AppendToLog "Usuario: '" & Request("usu") & "'"
  Response.AppendToLog "Contraseña: '" & Request("con") & "'"
%>
-----
#Software: Microsoft Internet Information Services 5.1
#Version: 1.0
#Date: 2003-03-27 09:46:23
#Fields: time c-ip cs-method cs-uri-stem cs-uri-query sc-
status
09:49:11 127.0.0.1 GET /ej-appendtolog.asp
Información+del+usuario:Usuario:+ 'alumno'Contraseña:+ 'pim
emola' 200
09:51:35 127.0.0.1 GET /ej-appendtolog.asp
Información+del+usuario:Usuario:+ 'administrador'Contraseñ
a:+ 'apolo16' 200
```

72

Programación en Internet – Curso 2005-2006

Propiedades de Sitio Web predeterminado

Propiedades del registro extendido

Propiedades del registro extendido

Consulta (URI) solicitada

73

Programación en Internet – Curso 2005-2006

Global.asa (I)

- Archivo opcional
- No es visible para los usuarios
- Información de eventos y objetos globales
- Almacenado en el directorio raíz de la aplicación (directorios virtuales)

74

Programación en Internet – Curso 2005-2006

No se puede mostrar la página

Hay un problema con la página que está intentando abrir y no se puede mostrar.

Pruebe lo siguiente:

- Modifique la dirección de la página en la barra de direcciones para quitar `global.asa` y presione **ENTRAR**.
- Si llegó a esta página Web a través de un vínculo, póngase en contacto con el administrador de ese sitio Web.
- Abra la página principal `localhost` y busque vínculos a la información que desea.
- Si cree que debe ser capaz de ver este directorio o esta página, póngase en contacto con el administrador del sitio Web utilizando la dirección de correo electrónico o el número de teléfono que figura en la página principal `localhost`.

HTTP Error 500-15 - Requests for global.asa not allowed
Servicios de Internet Information Server

Información técnica (para personal de soporte)

- Más información:
[Soporte técnico de Microsoft](#)

El fichero 'global.asa' está protegido y el cliente no puede acceder a él

75

Programación en Internet – Curso 2005-2006

Global.asa (II)

- Estructura típica:

```
<SCRIPT LANGUAGE="VBScript" RUNAT="Server">  
Sub Session_OnStart  
...  
End Sub  
Sub Session_OnEnd  
...  
End Sub  
Sub Application_OnStart  
...  
End Sub  
Sub Application_OnEnd  
...  
End Sub  
</SCRIPT>
```

76

Global.asa (III)

```
<SCRIPT LANGUAGE="VBScript" RUNAT="Server">  
  
Sub Application_OnStart  
Application("DSN") = "DRIVER=Microsoft  
Access Driver (*.mdb);  
DefaultDir=E:\Web\almacen\bd;  
DBQ=E:\Web\almacen\bd\almacen.mdb"  
Application("Email") = "info@pi.es"  
Application("bgcolor") = "#99CCAA"  
Application("text") = "#DDDDDD"  
End Sub  
  
</SCRIPT>
```

77

Global.asa (y IV)

- Orden de ejecución:

```
Application_OnStart (1 vez)  
Session_OnStart (1 vez por cada usuario)  
Session_OnEnd (1 vez por cada usuario)  
Application_OnEnd (1 vez)
```

78

Acceso a una base de datos (I)

- Mediante ODBC u OLEDB
- DAO: Data Access Objects
- ADO: ActiveX Data Objects
- DAO → ADO

79

Acceso a una base de datos (II)

- Se tiene que crear un objeto en el servidor de tipo ADODB.Connection.
- Este objeto establece una conexión con la base de datos mediante una fuente de datos (*data source*)
- Un fuente de datos se identifica mediante un nombre (DSN, *Data Source Name*), que contiene información sobre la base de datos, el controlador a emplear y la ubicación del servidor de bases de datos

80

Acceso a una base de datos (III)

- Tipos de fuentes de datos:
 - Usuario: sólo es válida para el usuario creador de la fuente
 - Fichero: es general el DSN en un fichero y se puede compartir por diversos usuarios
 - Sistema: con ámbito global en el ordenador en que se crea
- Un DSN contiene como mínimo:
 - Tipo de controlador (depende del SGBD)
 - Nombre o dirección (ruta) del servidor
 - Nombre o dirección (ruta) de la base de datos
 - Usuario y contraseña de acceso
 - Parámetros de configuración: permisos, *timeouts*, etc.

81

Acceso a una base de datos (IV)

- Hay una forma más cómoda de establecer una conexión: usar una cadena de conexión (*string connection*) o sin DSN (*DSNLess*)
- Esta forma permite hacer aplicaciones más portables y flexibles
- Una cadena de conexión tiene la siguiente estructura:
 - `"Provider=Microsoft.Jet.OLEDB.4.0; Data Source=ruta\baseDeDatos;"`

82

Acceso a una base de datos (V)

```
DRIVER={Microsoft Access Driver (*.mdb)};  
UserCommitSync=Yes;  
Threads=3;  
SafeTransactions=0;  
PageTimeout=5;  
MaxScanRows=8;  
MaxBufferSize=512;  
ImplicitCommitSync=Yes;  
FIL=MS Access;  
DriverId=25;  
DefaultDir=rutaBD;  
DBQ=rutaBD\NomBD;
```

83

Acceso a una base de datos (VI)

- Código ASP necesario:

```
set cnC = server.createObject("ADODB.Connection")
```

```
→ cnC.open "DRIVER={Microsoft Access Driver  
(*.mdb)};UserCommitSync=Yes;Threads=3;SafeTrans  
actions=0;PageTimeout=5;MaxScanRows=8;MaxBuffer  
Size=512;ImplicitCommitSync=Yes;FIL=MS  
Access;DriverId=25;DefaultDir=c:\inetpub\wwwroo  
t\p;DBQ=c:\inetpub\wwwroot\p\p.mdb;"
```

```
→ 'cnC.Open "Provider=Microsoft.Jet.OLEDB.4.0; Data  
Source=c:\inetpub\wwwroot\p\p.mdb;"
```

```
→ 'cnC.Open "nomDSN"
```

```
sSql = "select id, nom from t2"  
set rsR = cnC.execute(sSql)
```

84

Acceso a una base de datos (VII)

- En cualquiera de los casos, almacenar los datos de conexión en el objeto Application:

```
Application("DSN") = "DRIVER={Microsoft Access  
Driver  
(* .mdb)};UserCommitSync=Yes;Threads=3;SafeTrans  
actions=0;PageTimeout=5;MaxScanRows=8;MaxBuffer  
Size=512;ImplicitCommitSync=Yes;FIL=MS  
Access;DriverId=25;DefaultDir=c:\inetpub\wwwroo  
t\p;DBQ=c:\inetpub\wwwroot\p\p.mdb; "
```

```
Application("DSN") =  
"Provider=Microsoft.Jet.OLEDB.4.0; Data  
Source=c:\inetpub\wwwroot\p\p.mdb; "
```

```
Application("DSN") = "nomDSN"
```

85

Acceso a una base de datos (VIII)

- Sentencias SQL y tratamiento:
 - Select: sólo lectura, devuelve un ResultSet.
 - Insert into: escritura, devuelve un entero
 - Update table: escritura, devuelve un entero
 - Delete: escritura, devuelve un entero
 - Alter: escritura, devuelven enteros
- Ejemplo de código:

```
set resultSet = connexion.execute("select ...")  
iResultado = connexion.execute("insert...")
```

86

Acceso a una base de datos (IX)

- ResultSets, tratamiento:
 - Un RS es una tabla asociativa con una columna para cada atributo de la consulta
 - `nomResultSet("nomAtributo")`: para acceder a una columna del registro actual
 - `nomResultSet.eof`: indica si está o no al final de los registros
 - `nomResultSet.close`: cierra el objeto y libera memoria
 - Otros: `open`, `requery`, `seek`, etc.

87

Acceso a una base de datos (y X)

- Resultset, movimiento por los registros:
 - `nomResultSet.Move n`: mueve el cursor a una posición concreta del RS
 - `nomResultSet.MoveFirst`
 - `nomResultSet.MoveLast`
 - `nomResultSet.MovePrevious`
 - `nomResultSet.MoveNext`

88

Mensajes de error (I)

- Cuando en un ASP se produce un error, el servidor web genera una página de error con su propio formato
- Problema:
 - Rompe la “identidad” de una aplicación web
 - Algunos mensajes de error son poco “amigables” de cara al usuario final (no informan adecuadamente)
- Solución: la aplicación genera sus propios mensajes de error con su propio formato

89

Mensajes de error (II)

- ¿Cómo?
 - Desactivar los mensajes de error:
`On Error Resume Next`
 - Consultar el estado de error después de alguna operación “peligrosa”
 - ASP 2.0: Objeto `Err` de VBScript
 - ASP 3.0: Objeto `Err` de VBScript y objeto `ASPError` de ASP
- Pendiente en ASP: un mejor tratamiento de los errores (como excepciones en Java)

90

Mensajes de error (III)

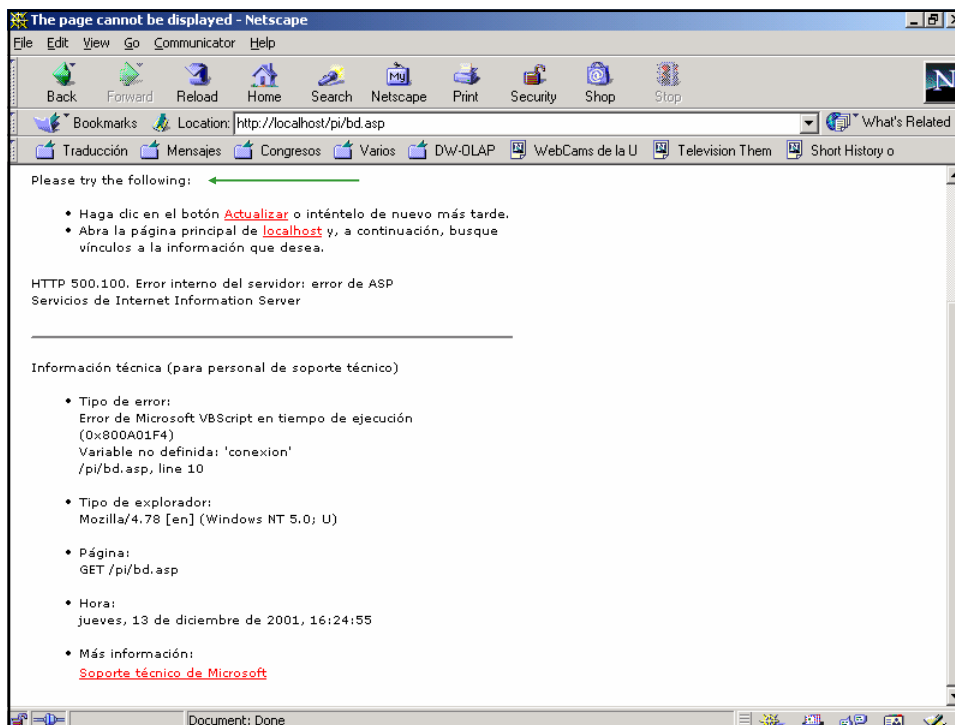
- Objeto Err:
 - Propiedades:
 - Description: descripción del error
 - HelpContext: identifica un tema en el HelpFile
 - HelpFile: fichero de ayuda
 - Number: entero que identifica el error
 - Source: origen del error
 - Métodos:
 - Clear: limpia el objeto (borrar el error)
 - Raise: genera un error en tiempo de ejecución

91

Mensajes de error (IV)

```
<%  
Option Explicit  
  
Set conexion = Server.CreateObject("ADODB.Connection")  
dsn = "almacen"  
sql = "SELECT * FROM Articulos ORDER BY Codigo"  
Set resultado = conexion.Execute(sql)  
...  
>
```

92



Programación en Internet – Curso 2005-2006

Mensajes de error (IV)

```

<%
Option Explicit
On Error Resume Next

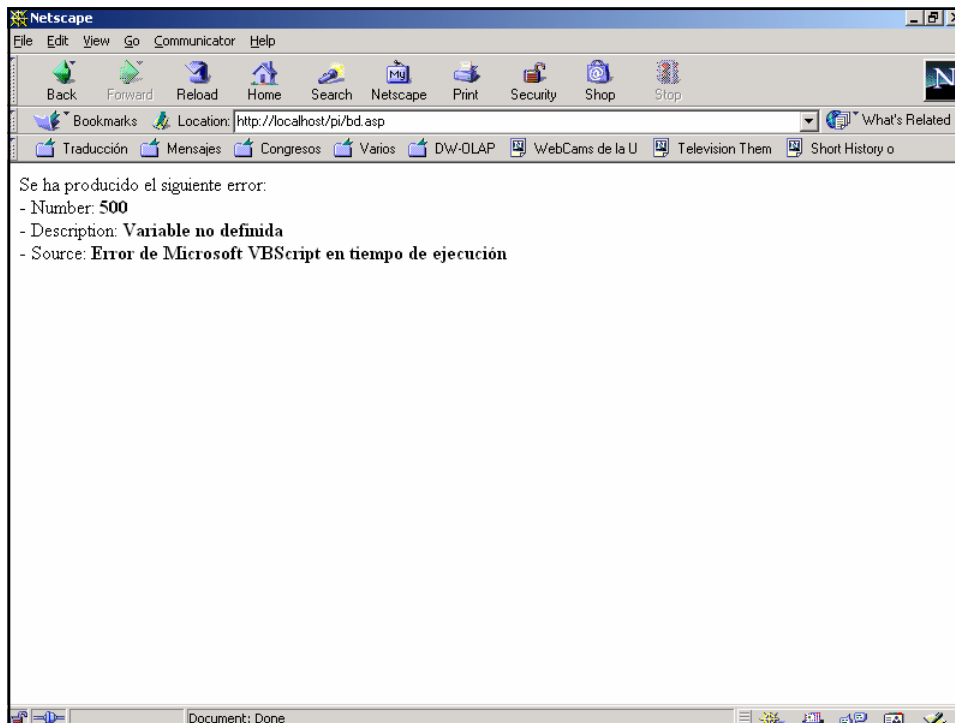
Set conexion = Server.CreateObject("ADODB.Connection")
If Err.Description <> "" Then
    Response.Write "<html><body>" & vbCrLf
    Response.Write "Se ha producido el siguiente error:<br>"
    Response.Write "- Number: <b>" & Err.Number & "</b><br>"
    Response.Write "- Description: <b>" & Err.Description & "</b><br>"
    Response.Write "- Source: <b>" & Err.Source & "</b><br>"
    Response.Write "</body></html>" & vbCrLf
    Response.End
End If
dsn = "almacen"
sql = "SELECT * FROM Articulos ORDER BY Codigo"
Set resultado = conexion.Execute(sql)

...
%>
    
```

```

Sub ChequeaError
...
End Sub
    
```

94

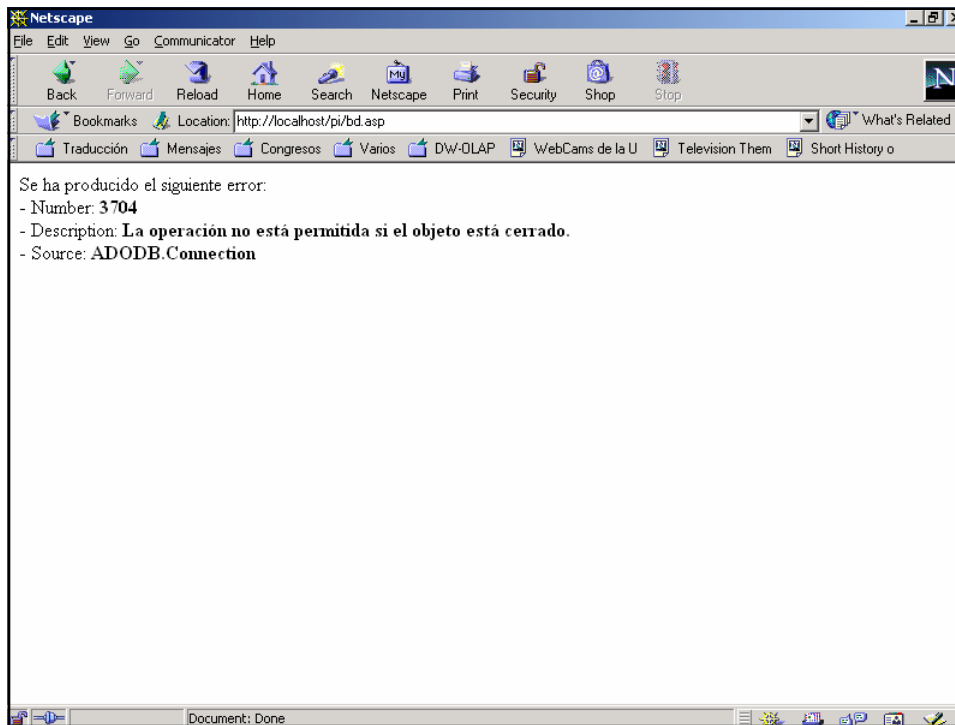


Programación en Internet – Curso 2005-2006

Mensajes de error (VI)

```
<%  
Option Explicit  
On Error Resume Next  
  
Dim conexion, sql, resultado, dsn, resultado2  
  
Set conexion = Server.CreateObject("ADODB.Connection")  
dsn = "almacen"  
sql = "SELECT * FROM Articulos ORDER BY Codigo"  
Set resultado = conexion.Execute(sql)  
  
ChequeaError  
...  
%>
```

96

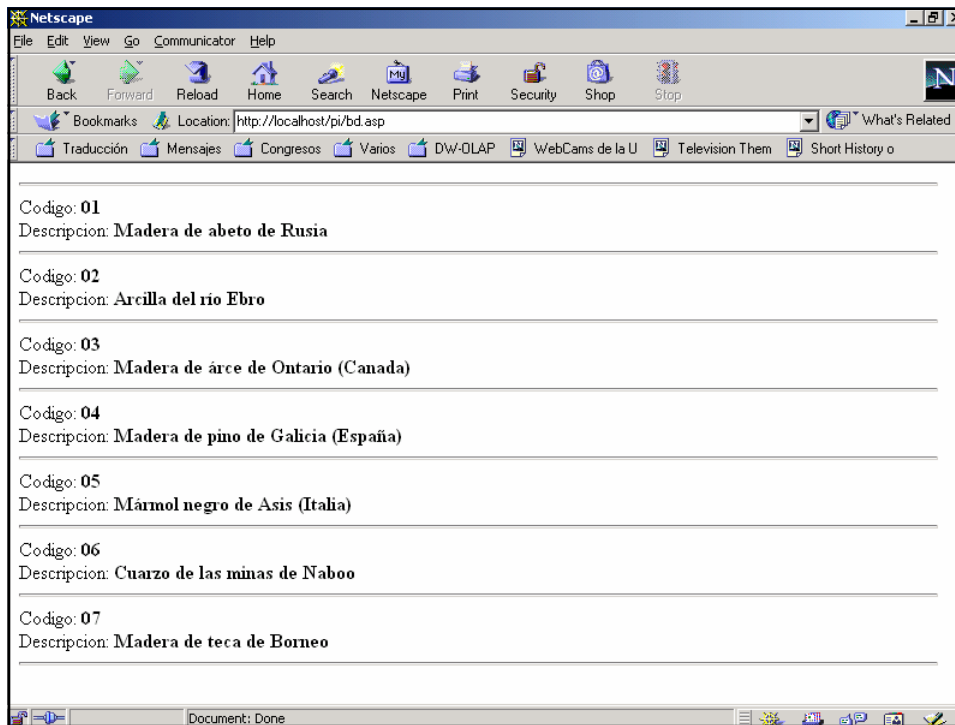


Programación en Internet – Curso 2005-2006

Mensajes de error (y VII)

```
<%  
Option Explicit  
On Error Resume Next  
  
Dim conexion, sql, resultado, dsn, resultado2  
  
Set conexion = Server.CreateObject("ADODB.Connection")  
dsn = "almacen"  
conexion.Open dsn  
sql = "SELECT * FROM Articulos ORDER BY Codigo"  
Set resultado = conexion.Execute(sql)  
  
ChequeaError  
...  
%>
```

98



Trucos

- Consultas SQL seguras
- Control de seguridad
- Gestión de memoria
- Reutilización de código
- Deshabilitar caché de páginas

Consultas SQL seguras

- Problemas con comillas ('...' y "...") en SQL → Inyección de SQL
- Distintas soluciones:

```
Replace(cadena, carácter, reemplazar)
```

```
Replace(nombre, "'", "`")
```

```
Replace(nombre, "\"", "`") o también
```

```
Replace(nombre, '\'', "`")
```

101

Control de seguridad

- Controlar en todas las páginas privadas que el usuario se ha validado utilizando variables de sesión

```
<!-- #INCLUDE VIRTUAL="control.inc" -->
```

```
If Session("Aceptado") <> "OK" Then
```

```
    Response.Redirect "paginaerror.html"
```

```
End If
```

102

Gestión de memoria

- De forma automática se tienen que destruir los objetos empleados en una página → Mejor destruirlos de forma manual:

```
Set a = Server.CreateObject("AAA.UnComponente")
Set b = Server.CreateObject("BBB.UnComponente")
...
...
' Cerrar objetos si se puede
' a.Close
' b.Close
Set a = Nothing
Set b = Nothing
```

103

Reutilización de código

- SSI permite reutilizar código → Incluir el mismo código en múltiples páginas
- El código se inserta en la página antes de que la página se interprete → SSI se procesa antes que ASP
- Permite reutilizar:
 - Funciones
 - Constantes globales

104

Deshabilitar caché de páginas

- Caché del servidor, proxy y cliente (navegador)

- Solución 1 (la más elegante):

```
Response.Expires = -100000  
Response.ExpiresAbsolute = #Jan 1, 1990 00:00:00#  
Response.AddHeader "Pragma", "no-cache"  
Response.AddHeader "cache-control", "no-store"
```

- Solución 2 (drástica, pero poco elegante): crear una URL distinta cada vez mediante el paso de un número aleatorio

```
<a href="pag.asp?a=6512344">Un enlace</a>
```

- Solución 3 (engañar al navegador): añadir al código un comentario HTML distinto en cada solicitud

```
<!-- Contra la caché: 988978787234 -->
```

105