



EBook Gratis

APRENDIZAJE asp-classic

Free unaffiliated eBook created from
Stack Overflow contributors.

#asp-classic

Tabla de contenido

Acerca de	1
Capítulo 1: Empezando con asp-classic	2
Observaciones.....	2
Versiones.....	2
Examples.....	2
Hola Mundo.....	2
Estructura de una página ASP simple.....	3
Capítulo 2: Bucle	4
Examples.....	4
En bucle.....	4
Hacer bucle.....	4
Capítulo 3: Conectando a una base de datos	6
Introducción.....	6
Examples.....	6
Poblando un desplegable desde la base de datos.....	6
Capítulo 4: Variables	7
Examples.....	7
Declarando.....	7
Tipos de variables.....	7
Creditos	9

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [asp-classic](#)

It is an unofficial and free asp-classic ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official asp-classic.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Empezando con asp-classic

Observaciones

Active Server Pages (ASP), también conocido como Classic ASP o ASP Classic, fue el primer motor de script del lado del servidor de Microsoft para páginas web generadas dinámicamente. La introducción de ASP.NET llevó al uso del término ASP clásico para la tecnología original.

El lenguaje de script del lado del servidor predeterminado para ASP es VBScript. Las páginas generadas están diseñadas para ser vistas en un navegador, por lo que usualmente utilizan el formato HTML y el estilo CSS.

¹ ASP no se instala de forma predeterminada en estas versiones de IIS. Debe ingresar a las funciones del administrador del servidor y agregar ASP.

Ver [ASP clásico no instalado de manera predeterminada en IIS 7.0 y superior](#)

Versiones

IIS	ÁSPID	Publicado
3.0	1.0	1996-12-01
4.0	2.0	1997-09-01
5.0	3.0	2000-11-01
6.0	3.0	2003-01-01
7.0	3.0 ¹	2008-01-01
7.5	3.0 ¹	2009-01-01
8.0	3.0 ¹	2012-01-01

Examples

Hola Mundo

```
<!doctype html>
<html>
  <head>
    <title>Example Page</title>
  </head>
  <body>
<%
  'This is where the ASP code begins
  'ASP will generate the HTML that is passed to the browser
```

```
'A single quote denotes a comment, so these lines are not executed
'Since this will be HTML, we included the html and body tags
'for Classic ASP we use Response.Write() to output our text
'like this

Response.Write ("Hello world")

'Now we will end the ASP block and close our body and html tags
%>
</body>
</html>
```

Cuando la respuesta se envía desde el servidor al navegador, la salida será así:

```
<!doctype html>
<html>
<head>
  <title>Example Page</title>
</head>
<body>
Hello world
</body>
</html>
```

Estructura de una página ASP simple

```
<%@ Language="VBScript" CodePage = 65001 %>
<%
Option Explicit
Response.Charset = "UTF-8"
Response.CodePage = 65001
%>
<!doctype html>
<html>
<head>
  <title>My First Classic ASP Page</title>
</head>

<body>
  <%= "Hello World"%>
</body>
</html>
```

Este es un ejemplo muy básico de una página ASP clásica que devuelve la frase "Hello World" al navegador junto con el resto del HTML estándar. Las partes de HTML son estáticas, es decir, el servidor las enviará al navegador tal como está. Las partes delimitadas por `<% %>` son lo que el servidor procesará realmente antes de enviarlo al cliente.

Tenga en cuenta que la sintaxis `<%= "stuff"%>` es una abreviatura para `<%Response.Write "stuff"%>`.

Lea Empezando con asp-classic en línea: <https://riptutorial.com/es/asp-classic/topic/1094/empezando-con-asp-classic>

Capítulo 2: Bucle

Examples

En bucle

En la ASP clásica podemos especificar un bucle for con la palabra clave *for*. Con la instrucción for necesitamos la *siguiente* instrucción que incrementará el contador.

```
For i = 0 To 10
    Response.Write("Index: " & i)
Next
```

La palabra clave *step* se puede usar para cambiar la forma en que la *siguiente* declaración modificará el contador.

```
For i = 10 To 1 Step -1
    'VBS Comment
Next
```

Para salir de un bucle for, use la instrucción *Exit For*

```
For i = 0 To 10
    Response.Write("Index: " & i)
    If i=7 Then Exit For 'Exit loop after we write index 7
Next
```

También podemos usar un `For...Each` bucle para realizar un bucle a través de una serie de elementos definidos en una colección. Por ejemplo:

```
Dim farm, animal
farm = New Array("Dog", "Cat", "Horse", "Cow")
Response.Write("Old MacDonald had a Farm, ")
For Each animal In farm
    Response.Write("and on that farm he had a " & animal & ".<br />")
Next
```

Hacer bucle

Do while es muy similar a for loop, sin embargo, esto generalmente se usa si nuestras repeticiones de bucles son desconocidas.

Hacer mientras

```
'Continues until i is greater than 10
Do While i <= 10
    i = i + 1
Loop
```

```
'Or we can write it so the first loop always executes unconditionally:  
'Ends after our first loop as we failed this condition on our previous loop  
Do  
    i = i + 1  
Loop While i <= 10
```

Hacer hasta:

```
'Ends once i equates to 10  
Do Until i = 10  
    i = i + 1  
Loop  
  
'Or we can write it so the first loop always executes unconditionally:  
'Ends after our first loop as we failed this condition on our previous loop  
Do  
    i = i + 1  
Loop Until i=10
```

Salir de un bucle Do es similar a un bucle for, pero solo se usa la instrucción *Exit Do*.

```
'Exits after i equates to 10  
Do Until i = 10  
    i = i + 1  
    If i = 7 Then Exit Do  
Loop
```

Lea Bucle en línea: <https://riptutorial.com/es/asp-classic/topic/5663/bucle>

Capítulo 3: Conectando a una base de datos

Introducción

ASP clásico utiliza una tecnología llamada [ActiveX Data Objects](#) cuando requiere acceso a fuentes de datos externas. La biblioteca ADODB proporciona tres objetos principales para este propósito, [ADODB.Connection](#) , [ADODB.Command](#) y el [ADODB.Recordset](#) .

Examples

Poblando un desplegable desde la base de datos

(Advertencia: hay muchos, muchos programadores que tienen connotaciones absolutas si cumplen con el código que usa conjuntos de registros en lugar de comandos y procedimientos almacenados).

```
<%
dim rs, sql
dim SelectedUser
SelectedUser = request.form("user")
if IsNumeric(SelectedUser) then
    SelectedUser = CLng(SelectedUser)
else
    SelectedUser = 0
end if
%>
...
<p>Select a user: <select name="user" size="1">
<%
sql = "SELECT id, displayname FROM users WHERE active = 1 ORDER BY displayname"
set rs = server.createobject("ADODB.Recordset")
rs.open sql, "[connection string stuff goes here]", 1, 2
do until rs.eof
    response.write "<option value='" & rs("id") & "'"
    if rs("id") = SelectedUser then response.write " selected"
    response.write ">" & rs("displayname") & "</option>" & vbCrLf
    rs.Movenext '←- VERY VERY IMPORTANT!
loop
rs.close
set rs = nothing
%>
</select></p>
...

```

Lea [Conectando a una base de datos en línea](https://riptutorial.com/es/asp-classic/topic/4991/conectando-a-una-base-de-datos): <https://riptutorial.com/es/asp-classic/topic/4991/conectando-a-una-base-de-datos>

Capítulo 4: Variables

Examples

Declarando

La creación de variables en VBScript se puede hacer usando la declaración Dim, Public o Private. Se recomienda colocar en la parte superior de la secuencia de comandos "Opción explícita", lo que obliga a definir explícitamente una variable.

Puedes declarar una variable como esta:

```
Option Explicit
Dim firstName
```

O puedes varias variables como esta:

```
Option Explicit
Dim firstName, middleName, lastName
```

Si no tiene la opción de la declaración explícita, puede crear variables como las siguientes:

```
firstName="Foo"
```

Esto **NO se** recomienda ya que pueden ocurrir resultados extraños durante la fase de tiempo de ejecución de su script. Esto sucede si se produce un error tipográfico más adelante al reutilizar la variable.

Para crear una matriz, simplemente declare con cuántos elementos en el parámetro:

```
Option Explicit
Dim nameList(2)
```

Esto crea una matriz con tres elementos.

Para establecer elementos de matriz, simplemente use la variable con el índice como parámetro, de esta forma:

```
nameList(0) = "John"
```

VBScript también soporta matrices multidimensionales:

```
Option Explicit
Dim gridFactors(2, 4)
```

Tipos de variables

VBScript es un lenguaje débilmente escrito; todas las variables son de tipo **variante** , aunque generalmente tienen un **subtipo** implícito que denota los datos que contienen.

Esto significa que su variable, no importa cómo la llame, puede contener cualquier valor:

```
Dim foo
foo = "Hello, World!"
foo = 123.45
foo = #01-Jan-2016 01:00:00#
foo = True
```

Tenga en cuenta que lo anterior es un código perfectamente válido, aunque mezclar sus variables de esta manera es una práctica increíblemente deficiente.

El subtipo de cadena siempre se asigna mediante marcas de voz " " . A diferencia de JavaScript y otros idiomas, el apóstrofe no proporciona la misma funcionalidad.

Los números en VBScript pueden incluir cualquier formato de número, pero tienen un subtipo en particular según su valor y si contienen un punto decimal o no.

Las fechas usan los # # especificadores. Tenga en cuenta que los formatos para un estilo de fecha numérico (por ejemplo, 01/01/2016) conservan un formato de fecha estadounidense, por lo que #05/06/2016# es el 6 de mayo, no el 5 de junio. Esto se puede hacer circular mediante el uso de un formato #dd-mmm-yyyy# , como en el ejemplo anterior.

Las variables booleanas contienen valores `True` o `False` .

Como se explicó anteriormente, las matrices se dimensionan utilizando un conjunto de paréntesis para definir el número de elementos y rangos (filas y columnas), por ejemplo:

```
Dim myArray(3, 4)
```

Todos los elementos de las matrices son de tipo variante, lo que permite que cada elemento sea de cualquier subtipo. Esto es muy importante cuando necesita realizar tareas como leer datos de un conjunto de registros u otro objeto. En estos casos, los datos se pueden asignar directamente a una variable, por ejemplo, cuando se devuelven desde un conjunto de registros ...

```
Dim myData
....
myData = rsMyRecordset.GetRows()
....
Response.Write(myData(3,2))
```

Un último tipo que requiere alguna explicación es el tipo de `Object` . Los objetos son básicamente punteros a la ubicación de memoria del objeto en sí. Los tipos de objetos deben ser `Set` ...

```
Dim myObj
Set myObj = Server.CreateObject("ADODB.ecordSet")
```

Lea Variables en línea: <https://riptutorial.com/es/asp-classic/topic/3195/variables>

Creditos

S. No	Capítulos	Contributors
1	Empezando con asp-classic	Community , David Starkey , Lankymart , Martha , RamenChef , SearchAndResQ
2	Bucle	Jake , Paul
3	Conectando a una base de datos	Lankymart , Martha
4	Variables	feetwet , Jake , John Odom , Lankymart , Paul