Licenciatura en Gestión Tecnológica

Programación Avanzada 2

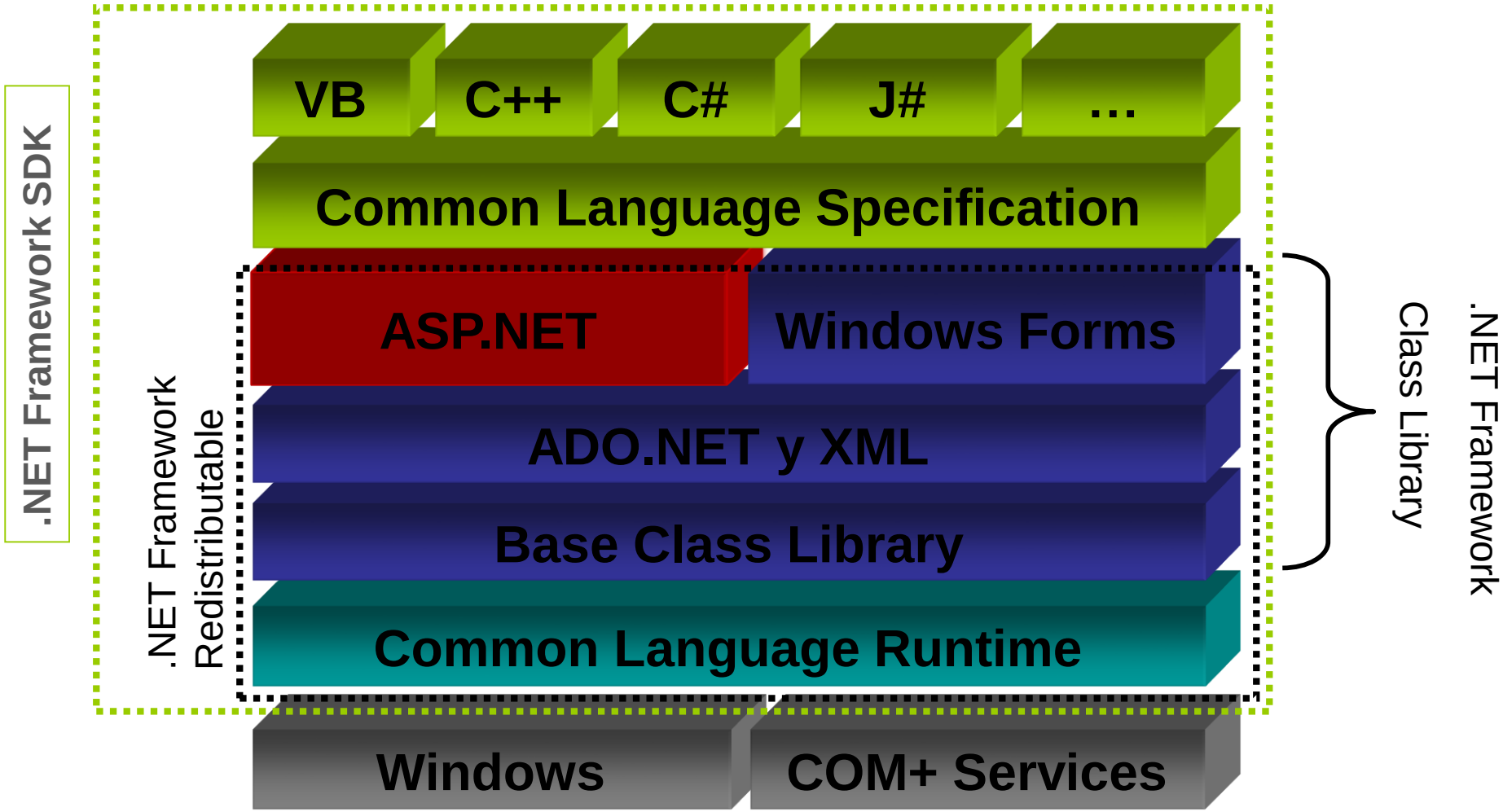
Formación Continua

Introducción a ASP.NET

Ing. Mariano Juiz

Introducción

Arquitectura del .NET Framework



Introducción

Páginas web

Una aplicación o sitio web es un conjunto de páginas que se transmiten por medio del protocolo HTTP de un servidor al cliente y viceversa

HTTP define como los navegadores y los servidores Web se comunican uno con otro. Esta basado en texto y es transmitido sobre conexiones TCP

El cliente solicita la página mediante un Request y el servidor web responde mediante un Response

Request

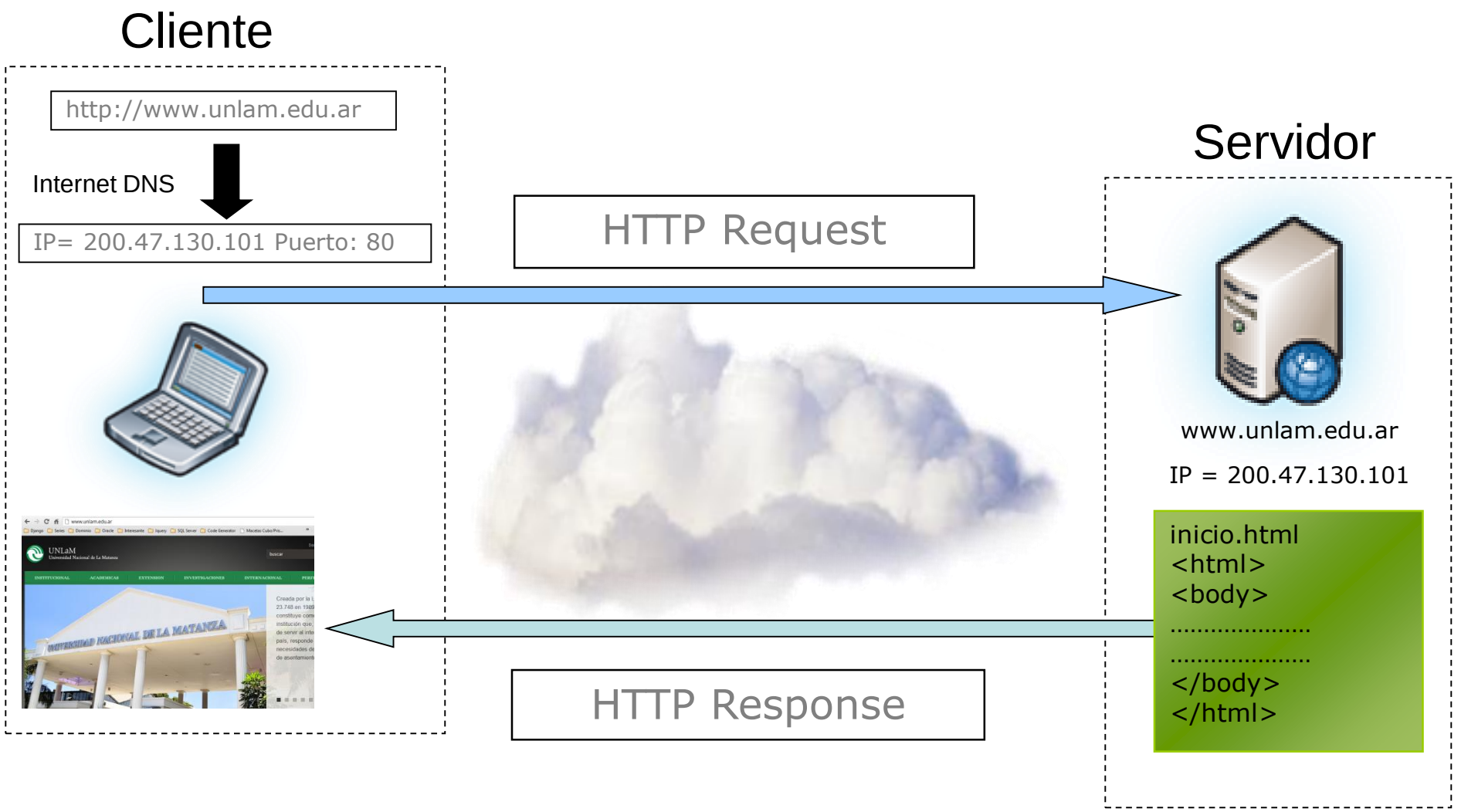
```
GET /inicio.html HTTP/1.1
Accept: */*
Accept-Language:...
Accept-Encoding:...
If-Modified-Since:...
If-None-Match:...
User-Agent: Mozilla/4.0...
Host: www.unlam.edu.ar
Connection: Keep-Alive
[blank line]
```

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/7.0
Date: ...
Content-Type: text/html
Accept-Ranges: bytes
Last-Modified: ...
ETag: ...
Content-Length: 46
[blank line]
<html>
<body>
…….
</body>
</html>
```

Introducción

Páginas web



Introducción

ASP.NET

ASP.NET es un “Marco” (framework) para programar aplicaciones web

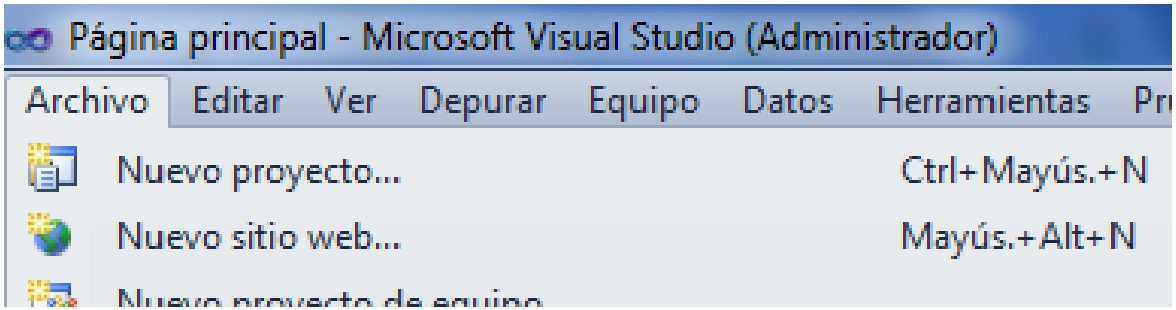
Internet Information Server (IIS), es el servidor Web de Microsoft que corre sobre plataformas Windows. Los servicios que ofrece son: FTP, SMTP, NNTP y HTTP/HTTPS

El componente fundamental de ASP.NET es el WebForm

Permite utilizar cualquier lenguaje .NET

Permite crear sitios web, aplicaciones web, servicios web, controles web, entre otras cosas.

¿Sitio Web o Aplicación Web?



¿Sitio Web o Aplicación Web?

Comparaciones

	Sitio Web	Aplicación Web
Alojamiento de Archivos	1 **Carpeta** para el sitio web dentro de C:\Users\<Usuario>\Documents\Visual Studio 2010\WebSites. 1 **archivo .sln** dentro de C:\Users\<Usuario>\Documents\Visual Studio 2010\Projects.	1 **Carpeta** que contiene tanto la aplicación como 1 **fichero .sln** y **1 archivo .vbproj o .csproj**, dentro de C:\Users\<Usuario>\Documents\Visual Studio 2010\Projects.
Archivo de proyecto	No hay. Todos los archivos dentro de la carpeta del sitio web se incluyen automáticamente en el sitio.	Existe un archivo vbproj o csproj que almacena la lista de ficheros incluidos.
Excluir archivos del proyecto	Agrega la extensión *.exclude*	No agrega ninguna extensión al archivo, simplemente actualiza el archivo de proyecto para no incluirlo.
Modelo de código	CodeFile Inherits="Carpeta1_Default“ (La clase Carpeta1_Default es **parcial)**	CodeBehind Inherits="**EspacioDeNombresRaiz**._Default“ (la clase EsppacioDeNombres.Raiz._Default**no es parcial)**
Espacios de Nombres (namespaces)	No lo crea (se pueden crear manualmente)	Los crea automáticamente

¿Sitio Web o Aplicación Web?

Comparaciones

	Sitio Web	Aplicación Web
App_Code	Es en la única carpeta (también en sus subcarpetas) donde se pueden guardar archivos *.vb o *.cs	Se puede guardar un archivo *.vb o *.cs en cualquier sitio del proyecto y compilará sin problemas
Compilación	Genera 1 assembly (*.dll) por cada directorio y 1 sòlo para todo el App_Code Facilita el Mantenimiento, si cambia una página sólo se actualiza el assembly de dicha página	Se genera 1 sólo assembly (*.dll) con toda la aplicación (páginas, controles de usuario, código del proyecto de la aplicación web, etc.) Ante un cambio se actualiza el único assembly
Performance en Navegación	No carga todas las páginas (sólo se carga lo necesario). Hay demora al cargar por primera vez cada página Si se cargan POCOS asseblies, ocupa menos memoria.	Carga todas las páginas por única vez., demora inicial al acceder a la aplicación web. No hay demora cuando se accede a cada página. 1 assembly ocupa menos memoria que TODOS los assemblies cargadas para la misma aplicación
Debug / Release	Al no haber un fichero de proyecto no se manejan estos conceptos. Se hace referencia únicamente en el archivo web.config (debug=”true|false”)	Archivos de Debug y Archivos de Release Tiene la configuración es debug=”true|false” del archivo web.config.

¿Sitio Web o Aplicación Web?

Comparaciones

	Sitio Web	Aplicación Web
Assemblies Referenciados	Aparecen Identificados en el Web.config	Aparecen Identificados en el *.vbproj o *.csproj y en la carpeta Referencias (donde se agregan)
Referencias a otros proyectos web	No	Sí
Atributos del ensamblado (nombre, versión, etc.)	No	Sí
Publicaciòn	Opción 1: xcopy (copiar TODOS los archivos al servidor destino como están) Opción 2: Precompilar Sitio y luego xcopy (más óptimo que el primero) incluyendo los archivos aspx	Opción 1: Generar ensamblado de la aplicación y copiarlo en el servidor destino junto con los archivos aspx sin precompilar Opción 2: Generar ensamblado de la aplicación e instalarlos mediante un proyecto de instalación (archivos aspx precompilados)

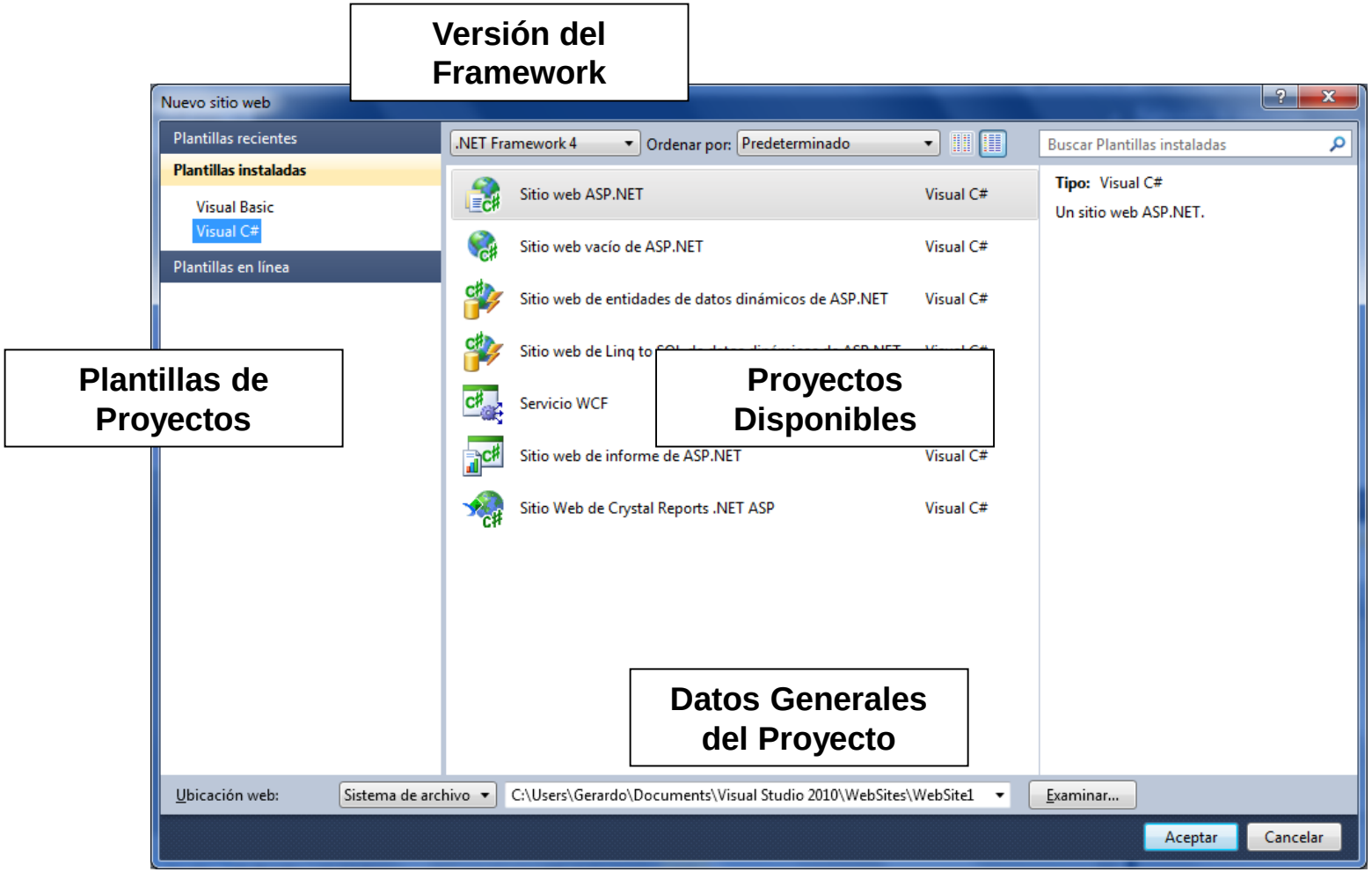
¿Sitio Web o Aplicación Web?

“No hay ninguna diferencia de rendimiento entre un proyecto de sitio web y un proyecto de aplicación web.”

http://msdn.microsoft.com/es-es/library/dd547590.aspx

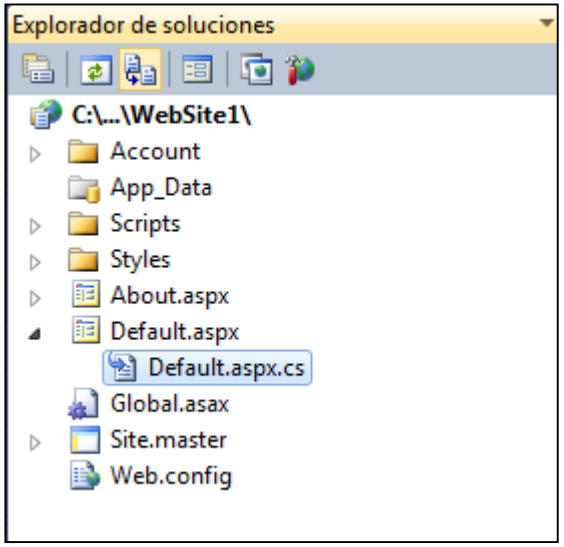
Aplicación ASP.NET

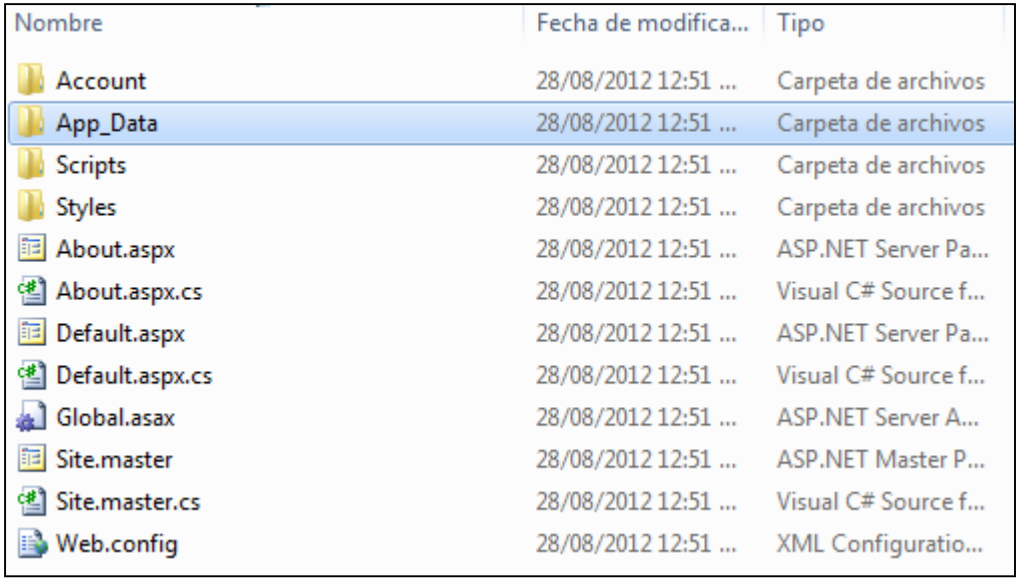
Sitio Web



Aplicación ASP.NET

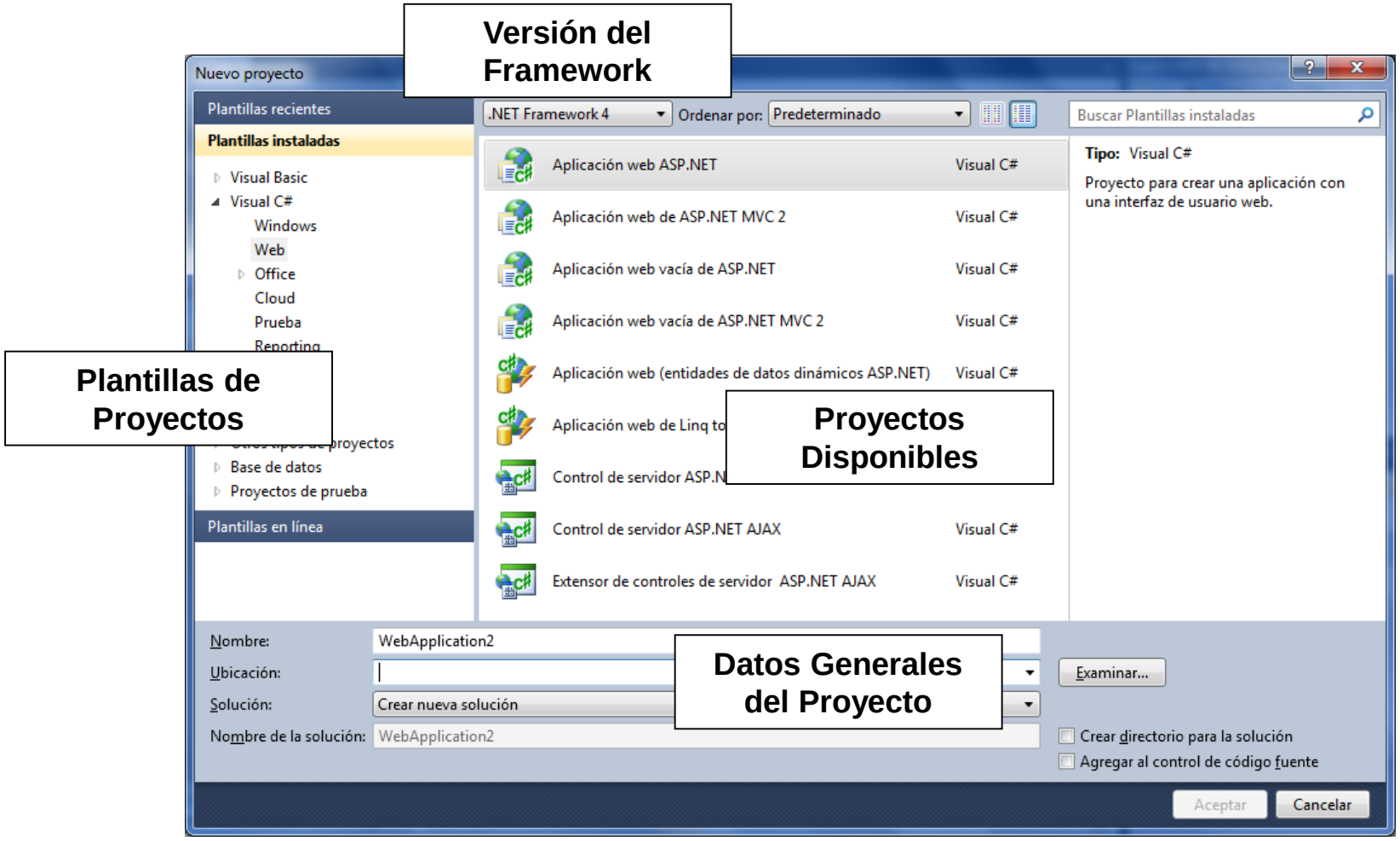
Sitio Web





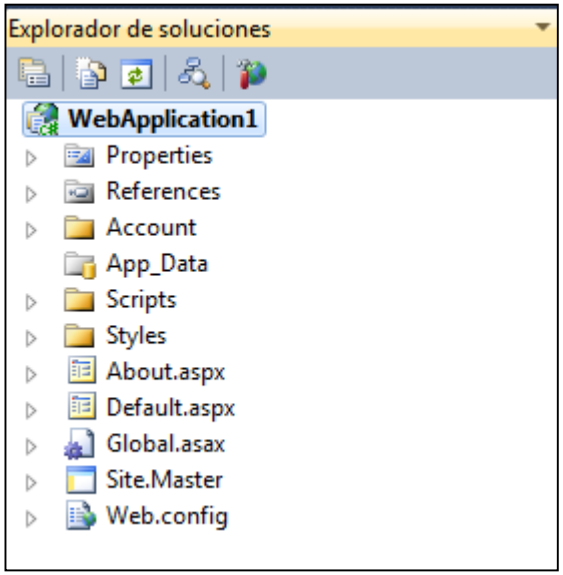
Aplicación ASP.NET

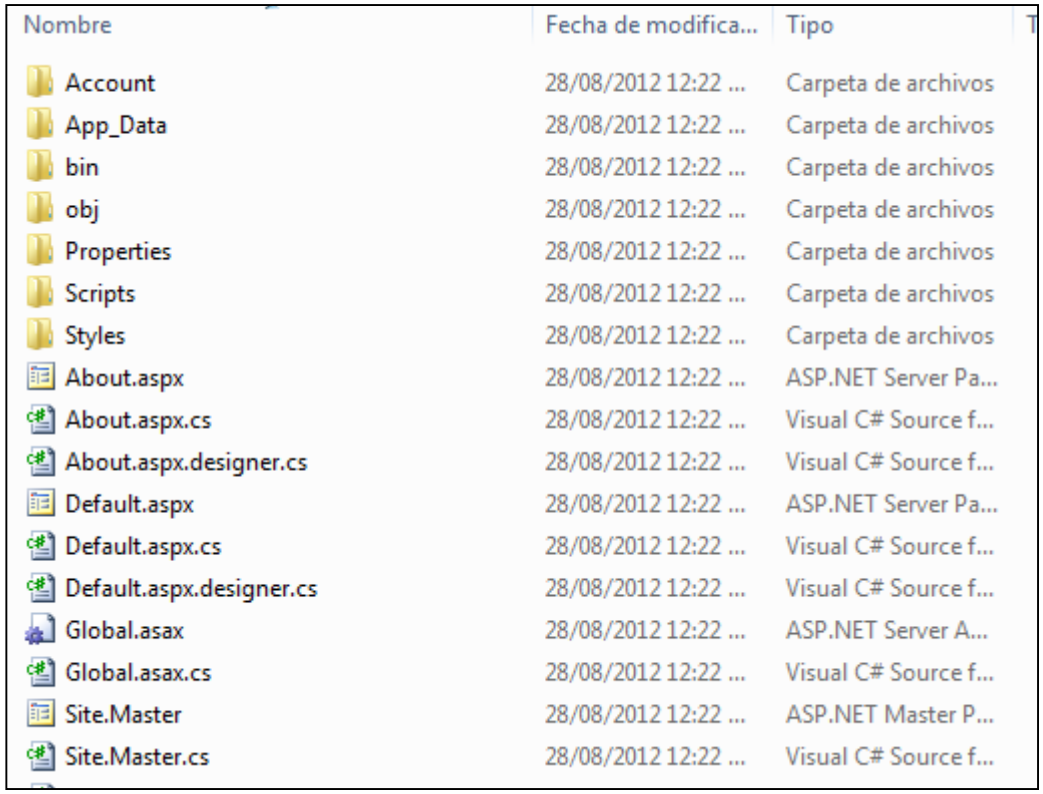
Aplicación Web



Aplicación ASP.NET

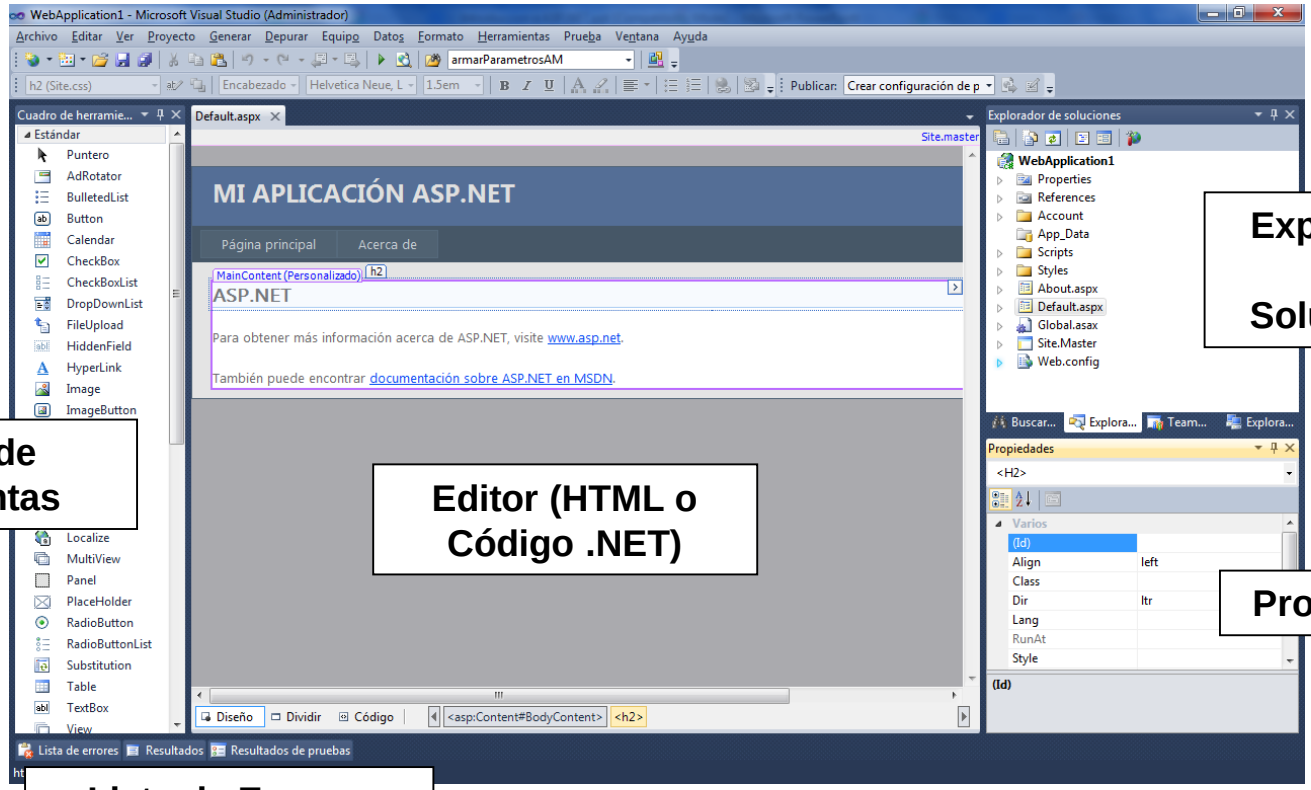
Aplicación Web





Aplicación ASP.NET

Visual Studio (IDE)



Cuadro de herramientas

Editor (HTML o Código .NET)

Explorador de Soluciones

Propiedades

Lista de Errores
Lista de Resultados
Lista de Tareas

Aplicación ASP.NET

Componentes

WebForms (Formularios Web)

- Uno o más archivos con extensión **.aspx**

Archivos Code-Behind / Code-File

- Archivos asociados a WebForms que contienen código del lado del servidor Ej.: C# (*.cs), VB.NET (*.vb), etc.
- Al usar Code-Behind, se utilizan clases parciales para vincularse con su aspx.
- Las páginas heredan de System.Web.UI.Page

Archivos de configuración con formato XML

- Un archivo Web.config por c/aplicación
- Un único archivo Machine.config por servidor

Directorio BIN

- Contiene el assembly de la aplicación (Ej.: MiAplic.dll)
- Cero o más assemblies (Componentes externos)

Aplicación ASP.NET

Componentes

Global.asax

Eventos a nivel de aplicación

Enlaces a Servicios Web XML

Permiten a la aplicación ASP.NET enviar y recibir datos desde Servicios Web

Archivo de Proyecto de Aplicación Web

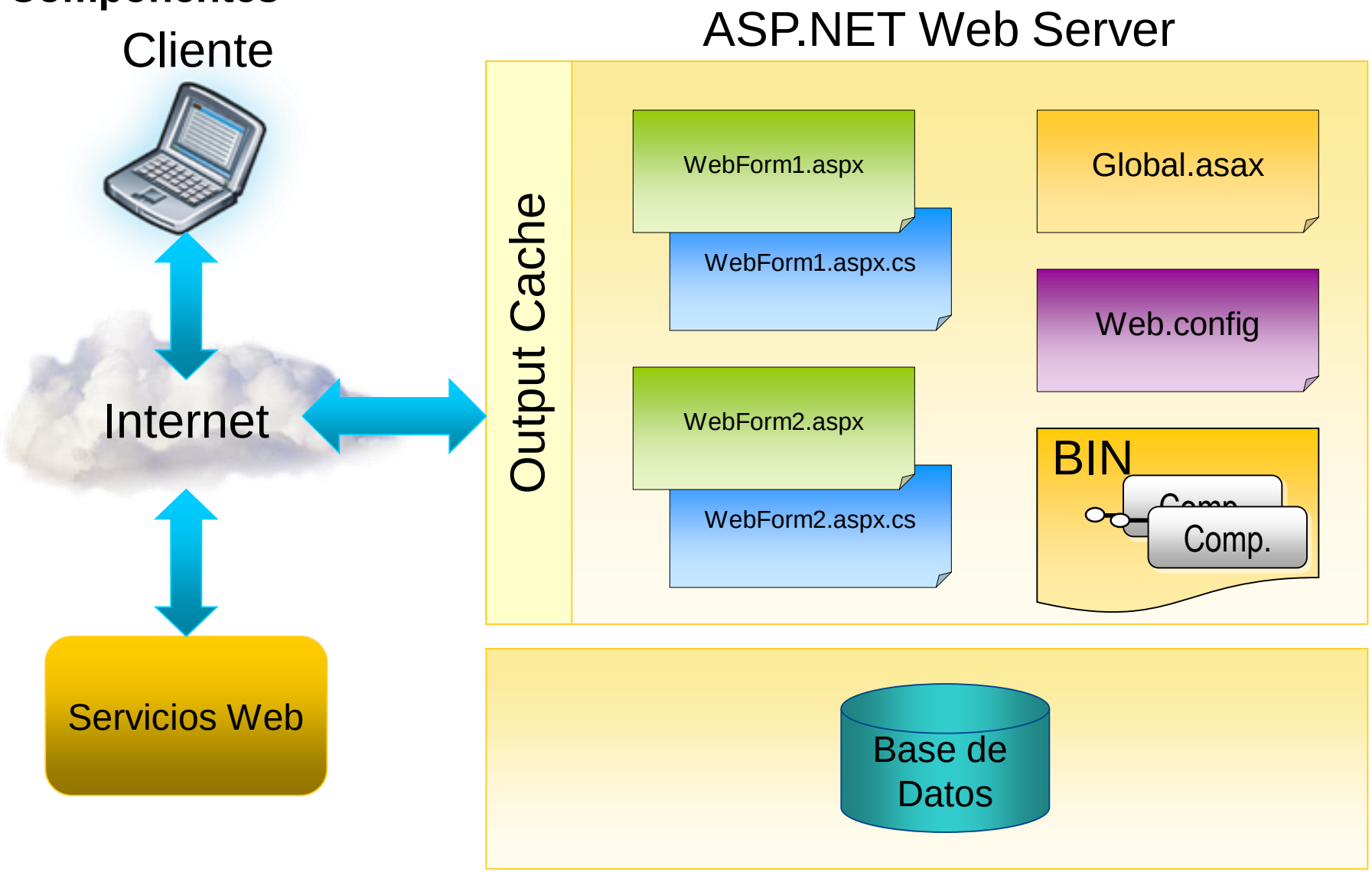
Guardan información en formato XML de los archivos utilizados por el proyecto. Utilizan extensión *.csproj o *.vbproj

Archivo de Solución de Aplicación ASP.NET

Guardan información en formato XML de los proyectos asp.net. Utilizan extensión *.sln

Aplicación ASP.NET

Componentes



Página web ASP.NET

Introducción

- Extensión .aspx
- Atributos de página
    - Directiva @ Page

```
<%@ Page Language=“c#" Codebehind="WebForm1.aspx.cs" %>
```

- Atributos de cuerpo

- Atributos de formulario

Página web ASP.NET

Código “En línea”

Código y contenido en el mismo archivo

Distintas secciones en el archivo para el código y HTML

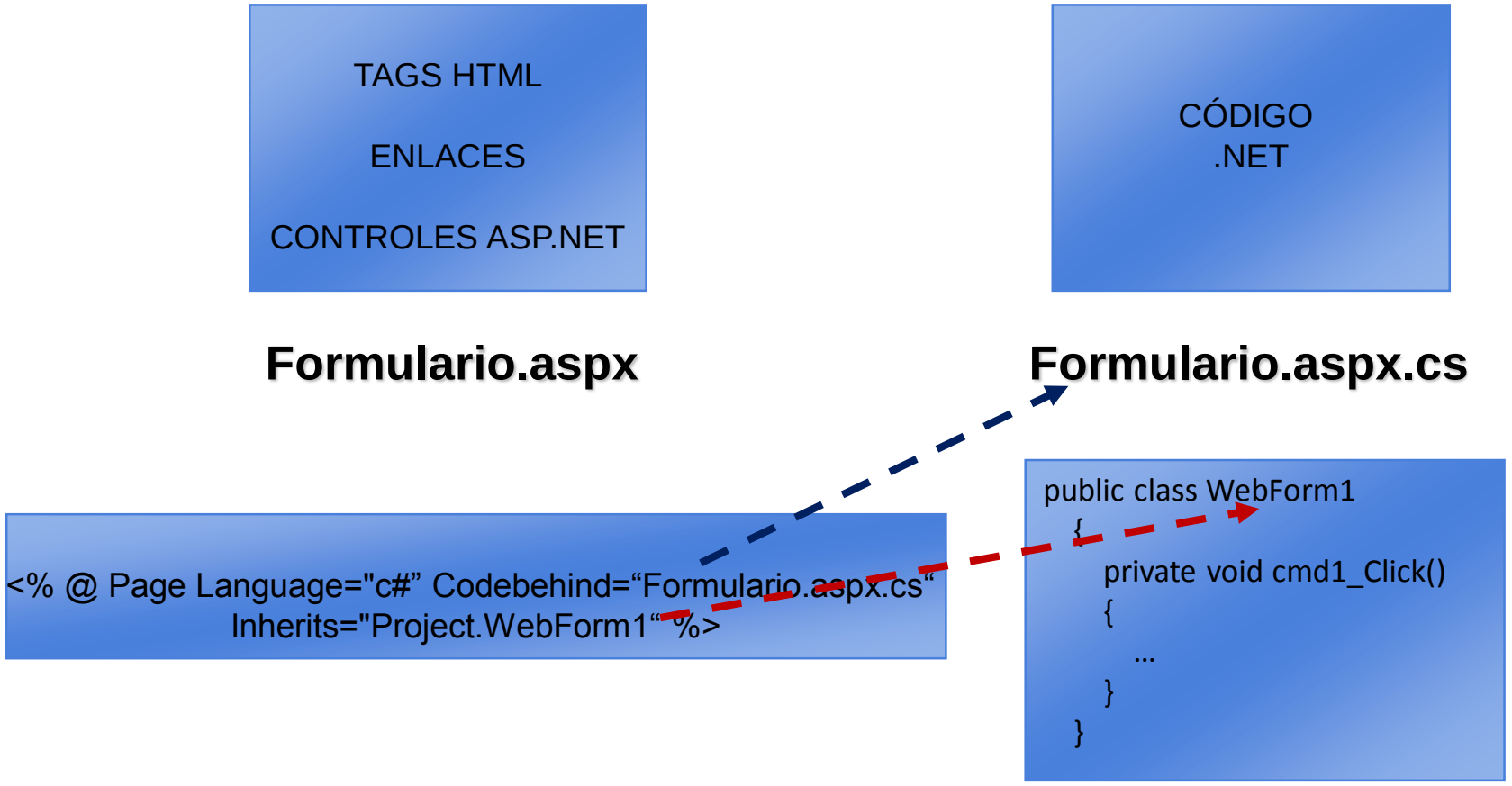
```
<HTML>
<HEAD>
<SCRIPT Language="c#" runat="server">
  private void btnAceptar_Click(object sender, System.EventArgs e)
  {
    . . .
  }
</SCRIPT>
</HEAD>
<BODY>
<form id=“form1” runat=“server”>
        <asp:Button id="btnAceptar" runat="server"/>
</form>

</BODY>
</HTML>
```

Formulario.aspx

Página web ASP.NET

Código “Subyacente” (Code-Behind)

Separación de código y contenido



Página web ASP.NET

Código “Mixto”

- Código embebido dentro del .aspx a la forma asp tradicional.
- Todo lo que esté entre “<% %>” se ejecutará del lado del servidor.

```
<%@ Page  Language="C#“%>
<HTML>
<HEAD>
</HEAD>
<BODY>
<form id=“form1” runat=“server”>
<div>
    <%
        if (DateTime.Now.Year == 2013)
        {
            Response.Write(“Es nuestro año”);
        }
        else
        {
            Response.Write(“Estas en el pasado o en el futuro”);
        }
    %>
</div>
</form>
</BODY>
</HTML>
```

Formulario.aspx

Controles de Servidor

Introducción

Componentes que se ejecutan en el lado del servidor

Poseen el atributo **runat=“server”**

Poseen un modelo de objetos común. Ej.: todos tienen las propiedades Id y Text

Mantienen su “estado” entre postbacks al servidor (ViewState)

“Generan HTML especifico según el browser cliente”

Controles de Servidor

Tipos de Controles de Servidor

Controles de Servidor HTML

Por defecto, los elementos HTML no son accesibles desde código del lado del servidor

Agregando runat="server" y el atributo id, se convierten en Controles de Servidor HTML

Controles de Servidor Web

Conocidos como WebControls

Solo accesibles del lado del servidor

Muchos tipos

- Intrínsecos
- Validación
- "Ricos"
- Del tipo lista de datos

No tienen una relación 1:1 con elementos HTML

Controles de Servidor

Tipos de Controles de Servidor

Botón HTML “clásico” (No es de Servidor)

```
<INPUT type="button" value="Buscar">
```

Control de Servidor HTML

```
<INPUT type="button" value="Buscar" id="cmdBuscar" runat="server">
```

Control de Servidor Web

```
<asp:Button id="cmdBuscar" runat="server" Text="Buscar"/>
```

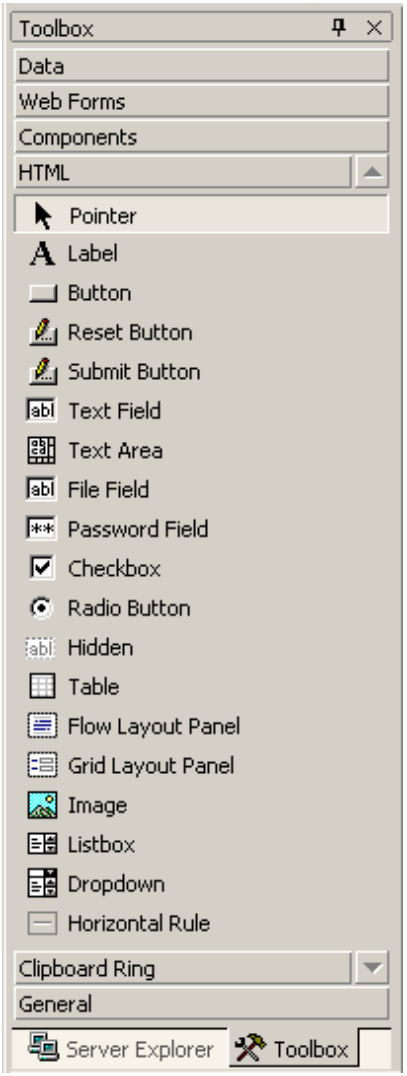
Controles de Servidor

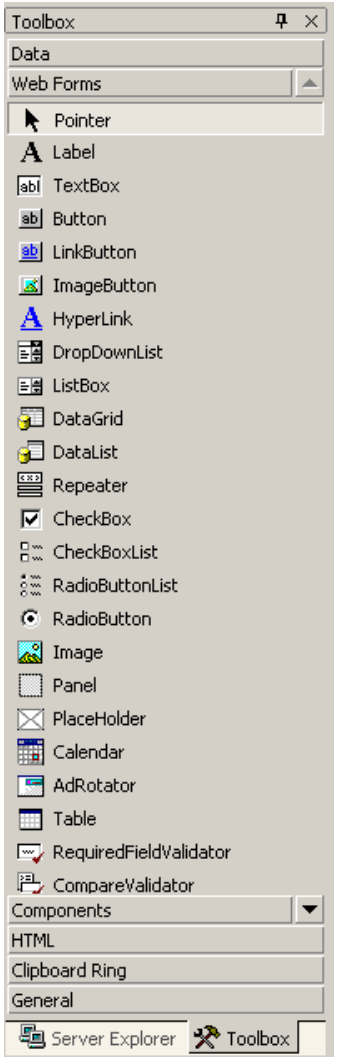
Tipos de Controles de Servidor

WebControl	HTML equivalente
<asp:button>	<input type=submit>
<asp:checkbox>	<input type=checkbox>
<asp:hyperlink>	
<asp:image>	<img src="…">
<asp:imagebutton>	<input type=image>
<asp linkButton>	
<asp:label>	
<asp:panel>	
<asp:radiobutton>	<input type=radiobutton>
<asp:table>	<table> </table>
<asp:textbox>	<input type=text>
<asp:listbox>	<select size="5"> </select>

Controles de Servidor

Tipos de Controles de Servidor





Controles de Servidor

Controles HTML Servidor vs Controles ASP.NET

Utilizar controles de servidor HTML si:	Utilizar controles de servidor Web si:
Se prefiere un modelo de objetos como HTML	Se prefiere un modelo de programación (C# o VB.NET)
Se está trabajando con páginas HTML existentes y se desea agregar funcionalidades de página ASP.NET Web	"Se está escribiendo una página que puede ser utilizada por varios navegadores"
El control interactuará con scripts cliente y servidor	Se necesitan funcionalidades específicas como un calendario o rotación de publicidad
El ancho de banda es limitado	El ancho de banda no es un problema

Controles de Servidor

Controles HTML Servidor vs Controles ASP.NET

Ambos heredan de System.Web.UI.Control que se encuentra contenida en System.Object

Ambos guardan estados

Los controles HTML :

- Son mucho más sencillos que los controles ASP.NET
- Tienen menos propiedades y eventos (no es consistente con los controles ASP.NET)
- Son más adecuados cuando no requerimos una gran flexibilidad y queremos cargar la página lo mínimo posible
- Existen en el espacio de nombres System.Web.UI.HtmlControls

Los controles ASP.NET:

- Son más complejos (calendario, grilla, validación, login, etc.)
- “Se adaptan al navegador”
- Al navegador se renderizar como controles HTML
- Existen en el espacio de nombres System.Web.UI.WebControls

Controles de Servidor

Controles ASP.NET

Sintaxis del control

```
<asp:TextBox id="TextBox1” runat="server“ Text=“Texto”/>
```

HTML generado por el control

```
<input name="TextBox1" type="text” value=“Texto“ Id="TextBox1"/>
```

Controles de Servidor

Scripts de eventos del lado del cliente

- Normalmente, se utilizan únicamente con controles HTML
- Interpretado por el navegador y se ejecuta en el cliente
- No tiene acceso a los recursos del servidor
- Utiliza <SCRIPT language="lenguaje">

Scripts de eventos del lado del servidor

- Utilizados tanto con controles de servidor Web como HTML
- El código se compila y ejecuta en el servidor
- Tienen acceso a recursos del servidor
- Utilizan <SCRIPT language="vb" runat="server"> o <SCRIPT language=“cs" runat="server">
  O bien ejecutando el evento del CodeBehind

•Se genera un POST desde el cliente hacia el servidor y una respuesta. Este ida y vuelta dentro de un mismo formulario web se llama **Postback**.

Controles de Servidor

Scripts de eventos del lado del servidor

Declaración de eventos en un control:

```
<asp:Button ID="btnEjemplo" runat="server" Text="Aceptar" onclick="btnEjemploClick" />
```

Atención del evento en el servidor (code behind)

```
protected void btnEjemploClick(object sender, EventArgs e)
{
    if (!string.IsNullOrEmpty(txtEjemplo.Text))
    {
        lblEjemplo.Text = txtEjemplo.Text;
    }
}
```

Controles de Servidor

Postback

Mecanismo de recarga de una página a partir de una ejecución del lado del cliente

Los estados de los controles podrán persistir si se encuentra activo el ViewState.

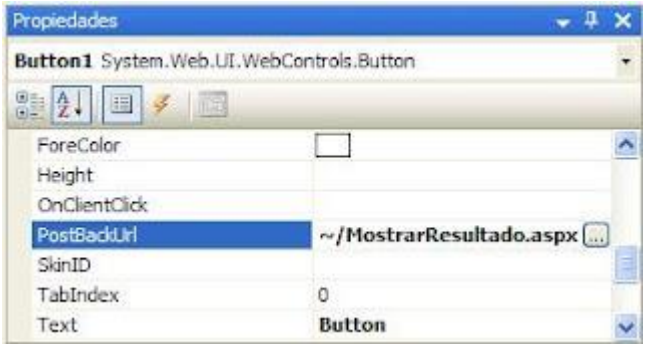
La propiedad IsPostBack indica si la página se está mostrando por primera vez o se está cargando como respuesta a un Postback

Para que algunos controles envíen la petición al servidor, deben setear la propiedad AutoPostBack en true

Controles de Servidor

Postback

A partir de la versión 2.0 del framework es posible realizar "Cross Page PostBacks“ (PostBacks hacia otras páginas)




```
if (PreviousPage != null)
{
    TextBox txt = (TextBox)PreviousPage.FindControl("TextBox1");

    Label1.Text = txt.Text; // Por ejemplo
}
```

Controles de Servidor de Usuario

• Los **User controls** simplifican la reutilización de código e Interfaz de Usuario dentro de una aplicación web.

• Un **User Control** es un **Web Server Control** definido por el usuario con extensión “.ascx”

• Contienen HTML, pero no los tags <HTML>, <BODY>, o <FORM>

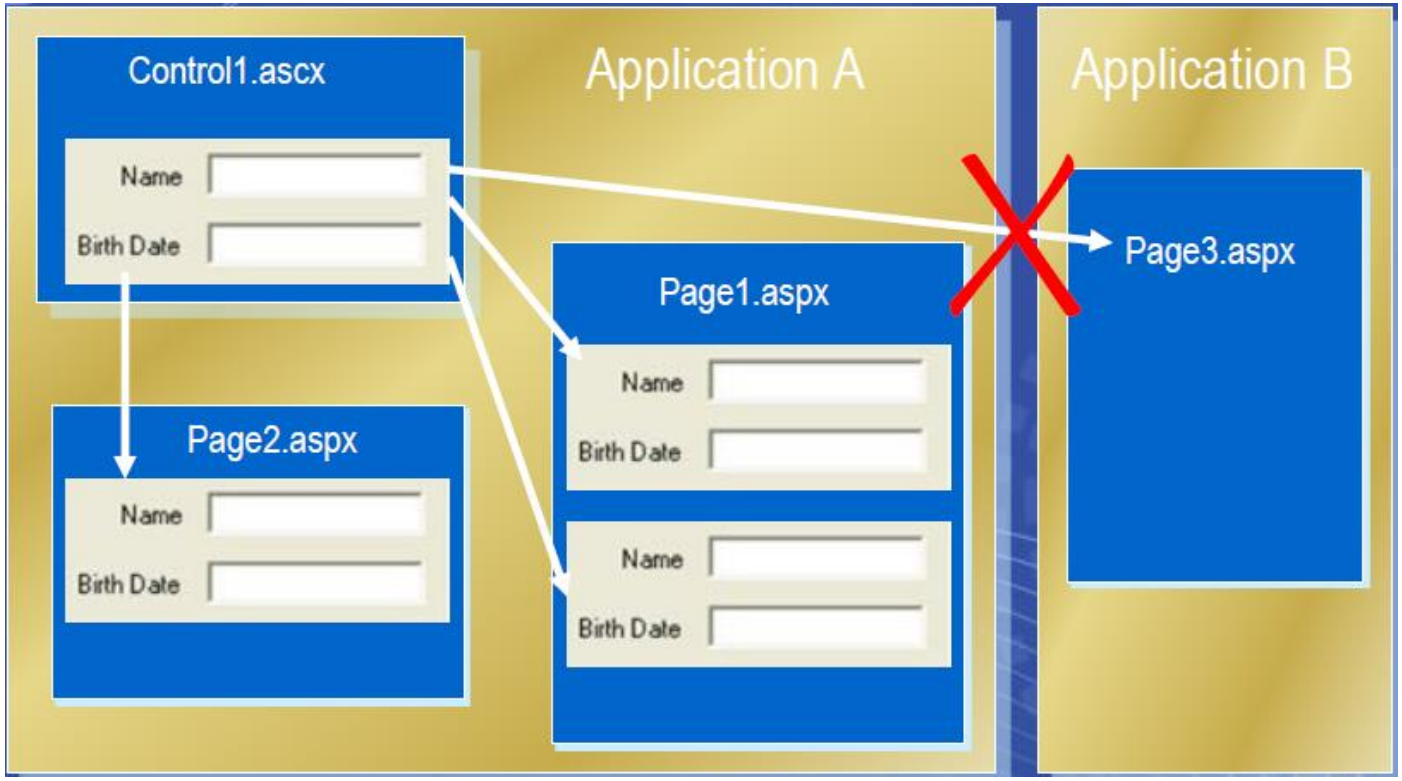
• Pueden tener también codigo del lado del servidor, ej c#

```
<%@ Control Language="C#" %>
```

Controles de Servidor de Usuario

¿Por que utilizarlos?

- Para reutilizar interfaz de usuario y código
- Pueden estar escritos en diferentes lenguajes



Controles de Servidor de Usuario

Pasos

- Usar la directiva @Register para incluir un user control en una pagina ASP.NET:

```
<%@ Register TagPrefix="uc" TagName="controlTxt" Src="miTextBox.ascx" %>
```

- Insertar el user control en un Formulario Web:

```
<uc:controlTxt id="ucDireccion" runat="server"/>
```

- Debemos codificar propiedades en el user control para que pueda ser accedido:

```
public string Texto
{
        get{ return txt.Text;}
        set{ txt.Text = value;}
}
```

- Accedemos a la/s propiedad/es desde la página que contiene el user control:

```
string direccion =ucDireccion.Texto;
```

Ciclo de Vida de una Página ASP.NET

Sucesos mas importantes del ciclo de vida de una página

- Solicitud de Página
- Inicio
  - Se instancian los objetos de los contextos **request** y **response**
  - Se crean el árbol de controles declarado en la página .aspx
  - Se determina si la página fue llamada en modo postback o no
- Inicialización de los objetos
  - Se inicializan los objetos y la configuración de sus estados
  - Se dispara del evento **Init** en cada uno de ellos, y luego el evento Init de la página contenedora
  - Se aplican la Master Page y los Themes que afecten a la página
- Carga del View State
  - ASP.NET carga en los estados de los controles y los valores del view state
  - Se procesan los datos del post
  - Se cargan los valores en cada uno de los objetos
- Carga de la página
  - Se dispara el evento **Page_Load**

Ciclo de Vida de una Página ASP.NET

Sucesos mas importantes del ciclo de vida de una página

- Eventos Post Back
  - Se ejecutan los manejadores de eventos programados por el desarrollador.
  - Se ejecuta la lógica del negocio y acceso a datos.

- Graba View State
  - Se guardan el estado de los controles

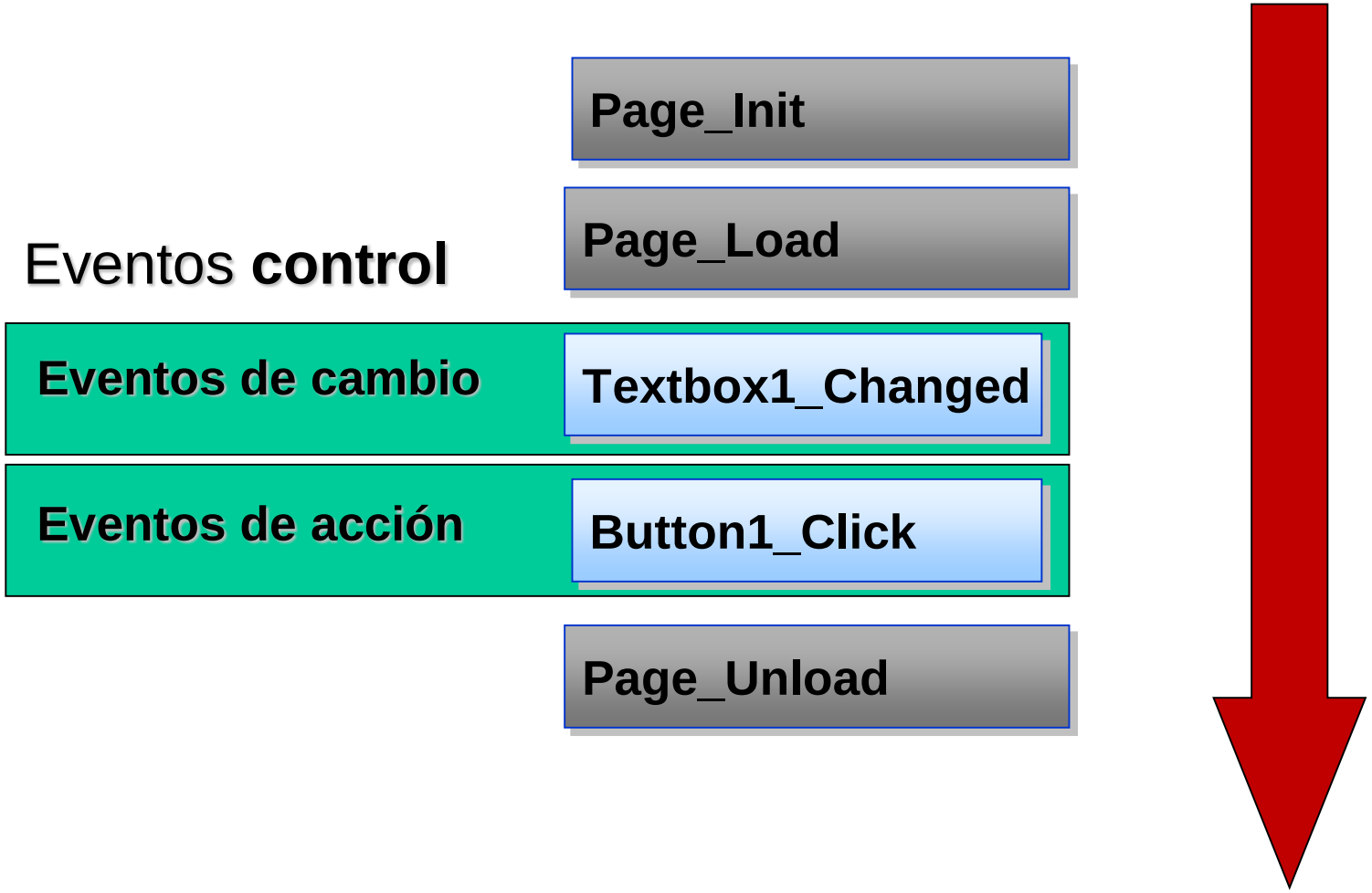
- Render
  - El lenguaje de marcas es generado (html, xml, etc).

- UnLoad
  - Los controles hijos son descargados, momento en el cual se dispara el evento **UnLoad** de cada uno de ellos y el de la página que los contiene.

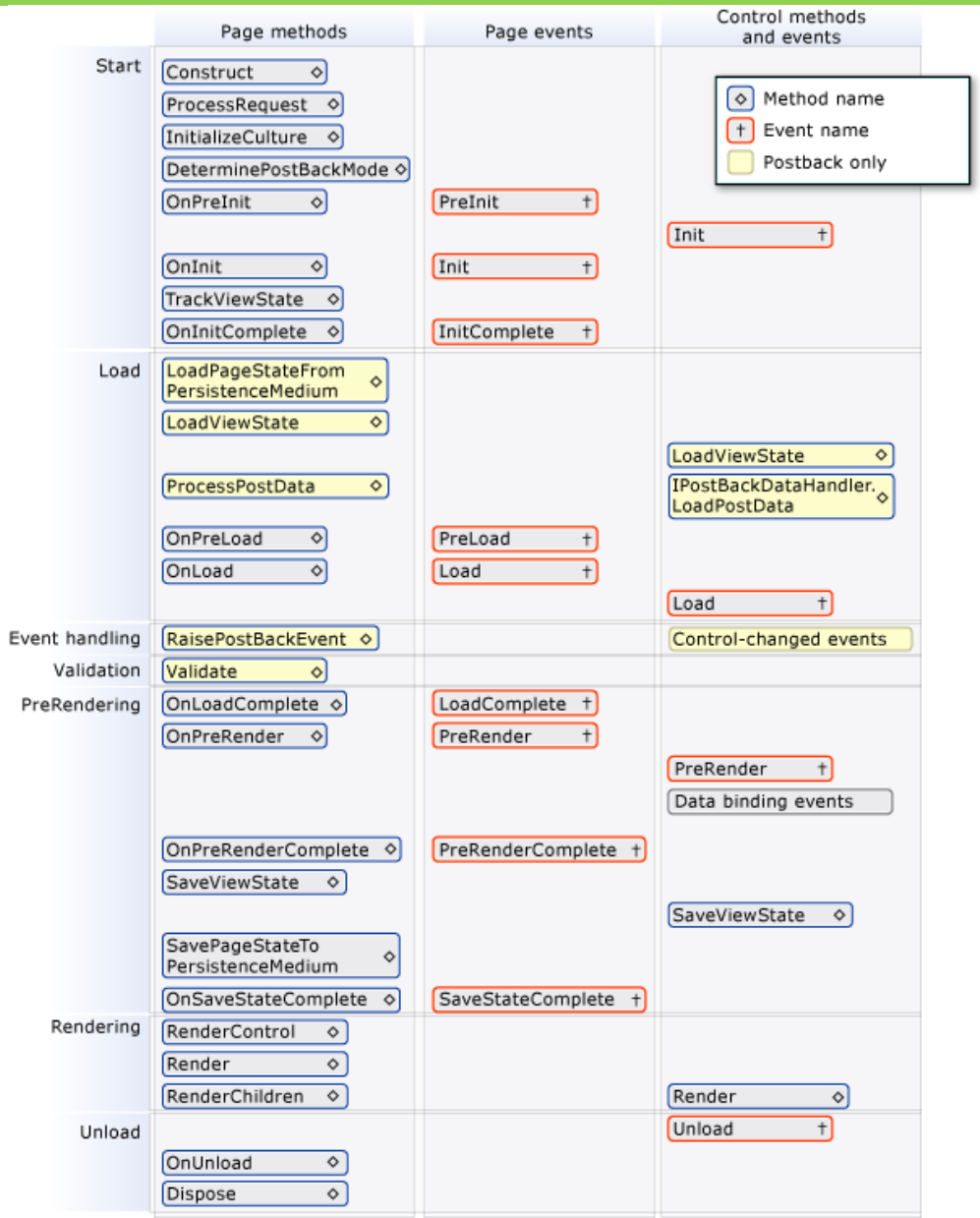
  - Es responsabilidad del recolector de basura (Garbage Collector) destruir (**Dispose**) todos los objetos incluidos en el ciclo de vida de la página.

Ciclo de Vida de una Página ASP.NET

Evento de una Página ASP.NET



Ciclo de Vida de una Página ASP.NET



Propiedad Page AutoEventWireUp

```
<%@ Page AutoEventWireup="true|false" %>
```

True (por defecto): La infraestructura de la página se encargará automáticamente de enlazar los eventos de la página con aquellos métodos definidos en ésta que tengan los nombres y las firmas apropiados.
Ej: Page_Load(…)

False: Debemos enlazar los eventos a los métodos de manera MANUAL:

- **Por código:**

```
override protected void OnInit(EventArgs e)
 {
        this.Load += new System.EventHandler(this.Page_Load);
 }
```

- **O en el form:**

```
<form id="form1" runat="server" onload="Page_Load">
```

Licenciatura en Gestión Tecnológica

Programación Avanzada 2

Formación Continua

Muchas Gracias

Ing. Mariano Juiz