Introducción a ASP.NET e IIS

¿Qué es ASP.NET?

Se llama genéricamente **ASP.NET** al conjunto de tecnologías utilizadas para desarrollar aplicaciones Web disponibles en **Microsoft .Net Framework**, por ejemplo, **ASP.NET clásico ASP.NET MVC** , **ASP.NET Web** Pages, **ASP.NET Web API** , etc.
Para estas clases trabajaremos con proyectos asp.net clásico con .Net Framework; este tipo de proyecto no está disponible en .Net Core.
El link a la página oficial de esta tecnología es http://www.asp.net/.

Herramientas de desarrollo

Utilizaremos **Visual Studio Community 2017/9** y **Microsof t SQL Server Express 2016/7/9**. Ambas herramientas son gratuitas y están disponibles para su descarga. También podría utilizar **Visual Studio Community 2015** y **Microsoft SQL Server Express 2014.**

Arquitectura cliente-servidor

En el contexto Web, las aplicaciones residen en un servidor Web. En el caso concreto de las aplicaciones ASP.NET, estas requieren de un servidor Windows Server ejecutando IIS (Internet Information Server). También para entornos de desarrollo, puede usar el IIS Express incluido en Visual Studio.

Las aplicaciones Web reciben requerimientos de clientes. Típicamente el cliente es un navegador (Chrome, Internet Explorer, Firefox, etc.) pero también puede ser una aplicación de escritorio, móvil, etc.

Estos requerimientos se procesan en el ámbito del servidor Web y la información resultante se transmite como respuesta a los clientes.

Páginas Web estáticas vs. Páginas Web dinámicas

Como es sabido, los navegadores Web (browsers) son capaces de mostrar páginas Web. Estas pueden contener textos, gráficos, links, archivos de audio, etc.

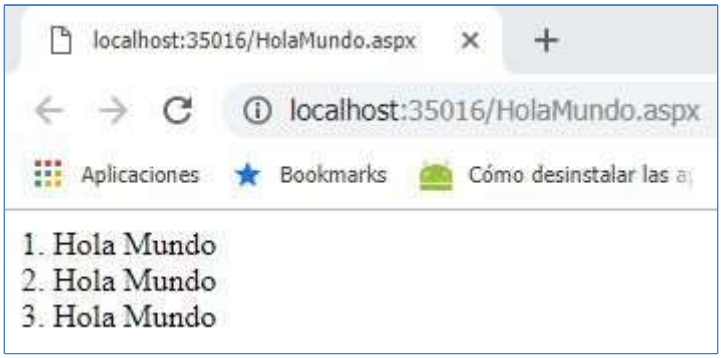
Una página estática es una página que ha sido requerida y devuelta por el servidor Web sin ningún procesamiento extra. Normalmente estás páginas tiene la extensión htm o html. En otras palabras, el contenido devuelto coincide con el que se encuentra en la página almacenada en el servidor.

Una página dinámica es una página producto de un procesamiento dentro del ámbito del servidor. Parte del contenido devuelto es entonces generado dentro del servidor. En el contexto de ASP.NET, estas páginas tienen la extensión aspx. ASP.NET es una librería del .Net Framework que permite procesar páginas web en el servidor para luego enviarlas al cliente: el navegador web. Se lo llama “motor” porque genera páginas web.

Este concepto se entiende mejor realizando el ejemplo HolaMundo, disponible para su descarga:

En la página HolaMundo.htm, el código fuente devuelto por el servidor es idéntico al que fue almacenado en la página.

En la página HolaMundo.aspx, el código fuente devuelto por el servidor difiere del almacenado en la página, ya que los textos "Hola Mundo" 2 y 3 fueron procesados y generados por el motor ASP.NET.



Código HTML y código .NET

Las páginas aspx se forman entonces combinando el HTML estático existente con el HTML dinámico producido por el código .NET. El motor combina ambos códigos produciendo la respuesta que es enviada al navegador que hizo el requerimiento

Cabe destacar que existen varios tipos de arquitecturas de aplicación dentro de ASP.NET: Web Forms (utilizada en este curso), MVC, Web API y Aplicaciones SPA (Single Page Application), algunas disponibles sólo en .Net Framework o otras disponibles en .Net Framework y en .Net Core.

Tipo de arquitectura de aplicación	.Net Framework	.Net Core
Web Forms	Sí	No
MVC	Sí	Sí

Web API	Sí	Sí
Single Page Application	Sí	Sí
Aplicación API de Azure	Sí	No
Azure Mobile App	Sí	No
API	No	Sí

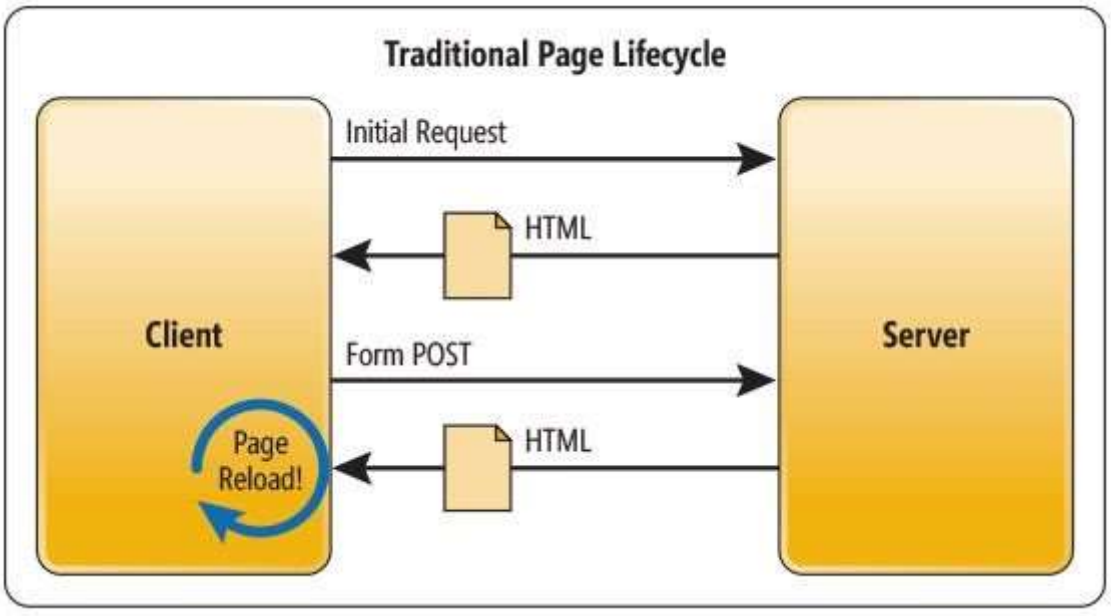
Sugerimos ver el sitio de Microsoft para más detalles sobre la versión de Visual Studio y los tipos de aplicaciones Web disponibles Documentación de .NET.

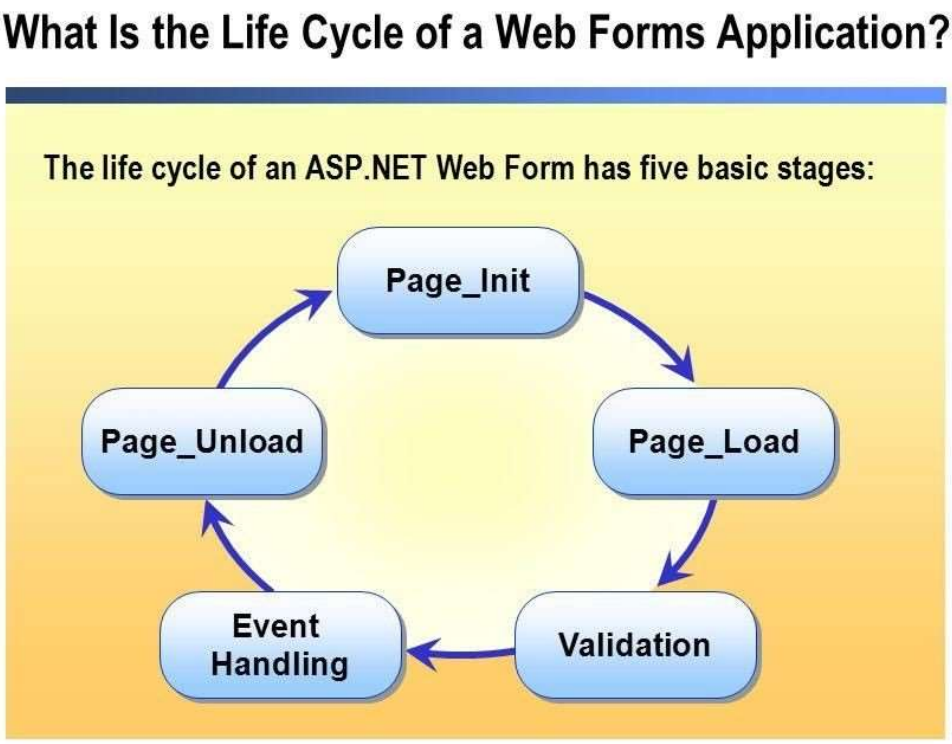
Diferencia entre ASP.NET Web Forms y MVC

En la arquitectura Web Form, las páginas se comportan en forma similar a un formulario Windows Form, es decir similar a las ventanas con carga de datos y/o visualización de resultados de aplicación de escritorio.

Esto significa que tienen propiedades, métodos y eventos, al igual que un formulario Windows (Windows Form). En el ejemplo HolaMundo, el código que cambia la propiedad Text del control lblHolaMundo se ejecuta en el evento Load del formulario.

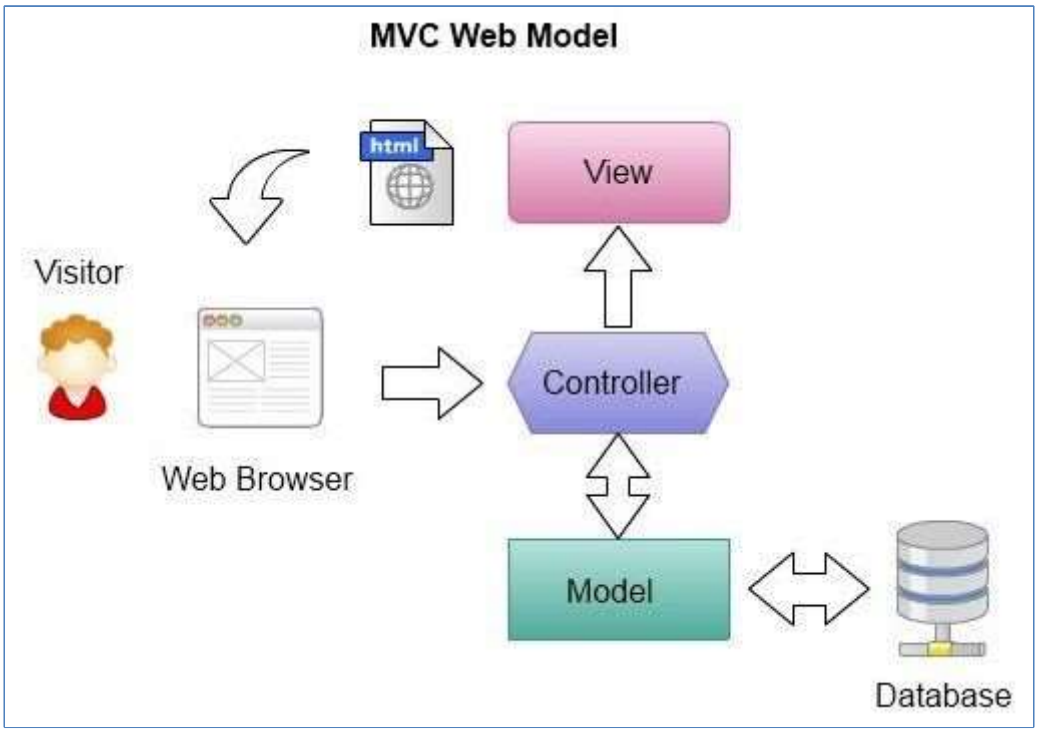
Si bien existe una separación entre el código HTML y el código .NET, es posible mezclar ambos, cómo puede apreciarse en la instrucción Response.Write() contenida el código HTML de la página.





En una aplicación **MVC** (sigla en inglés que significa Model View Controller = Modelo Vista Controlador), es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.
Se trata de un modelo muy maduro y que ha demostrado su validez a lo largo de los años en todo tipo de aplicaciones, y sobre multitud de lenguajes y plataformas de desarrollo.

- El **Modelo** que contiene una representación de los datos que maneja el sistema, su lógica/reglas de negocio, y sus mecanismos de persistencia de datos hacia un repositorio de datos.
- La **Vista**, o interfaz de usuario, que compone la información que se envía al cliente y los mecanismos de interacción con éste.
- El **Controlador**, que actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno.



El modelo es el responsable de:

- Acceder a la capa de almacenamiento de datos para recuperar/persistir los datos.
- Definir las reglas de negocio (la funcionalidad del sistema). Un ejemplo de regla puede ser: "Si los productos pedidos a un proveedor aún no llegaron al sector de abastecimientos, consultar el tiempo de entrega estándar del proveedor".
- Lleva un registro de las vistas y controladores del sistema.

El controlador es responsable de:

- Recibir los eventos de entrada (un click, un cambio en un campo de texto, la petición de una página web/vista, etc.).
- Contiene reglas de gestión de eventos, del tipo "SI Evento Z, entonces Acción W". Estas acciones pueden suponer peticiones al modelo o a las vistas. Una de estas peticiones a las vistas puede ser una llamada al método "Actualizar()". Una petición al modelo puede ser "Obtener_tiempo_de_entrega(nueva_orden_de_venta)".

Las vistas son responsables de:

- Recibir datos del modelo y para mostrarlos al usuario.

- Tener un registro de su controlador asociado (normalmente porque además ese controlador instancia la vista).

Código .NET

Las páginas ASP.NET pueden contener código .NET escrito en C#, VB.NET, etc. En este curso se utiliza C# que es uno de los lenguajes más extensibles y usados para aplicaciones web con tecnologías Microsoft.

HTML vs controles de servidor

En el ejemplo HTML-Control, puede verse que muchas veces es posible obtener el mismo resultado utilizando HTML o controles de servidor.

Por ejemplo, el título de la página HTML es simplemente texto HTML, en cambio en la página Control, se utiliza un control de servidor llamado Label.

Es importante tener en cuenta que cuando utilizamos un control de servidor, éste se instancia en la memoria del servidor, ejecuta una serie de eventos como cualquier objeto y finalmente el motor traduce la salida que este produce a HTML para combinarlo con el HTML estático de la página.

Esto evidentemente supone un costo en términos de recursos. Si por ejemplo, tenemos 100 clientes que simultáneamente acceden a dicha página, tendremos 100 instancias de dicho control instanciados en el servidor con el correspondiente gasto de recursos.

Por eso, siempre que sea posible, tiene sentido reemplazar un control de servidor por su equivalente en HTML estático, el más ágil su ejecución o como se dice en la jerga “ más performante”.

Por supuesto esto no es posible cuando el contenido que se desea generar necesariamente debe ser generado en el servidor. Es el caso de la hora y la fecha en el ejemplo mencionado.

Tecnologías involucradas en una página ASPX

- HTML
- JavaScript
- CSS
- NET (C#/VB.NET)

Html

Es un lenguaje con sintaxis similar a **XML** que permite representar visualmente una página Web. Se lo llama lenguaje de marcadores o marcas (markup language). Como todo documento XML, está basado en elementos. A su vez cada elemento posee atributos y éstos valores.
Existen diferentes versiones de HTML. Actualmente la última versión es **HTML5.**

Si bien no es indispensable para realizar este curso, un “desarrollador web”, en nuestro caso un desarrollador web ASP.NET debe tener o adquirir un nivel de conocimientos de intermedio a avanzado en HTML.

Durante el curso utilizaremos fundamentalmente los siguientes elementos:

- **head:** encabezado de la página
- **body:** cuerpo de la página
- **form**: formulario
- **div**: división
- **ul:** lista sin orden
- **li:** elemento de una lista

Para más información se sugiere ver este link: HTML.

JavaScript

Es uno de los lenguajes de programación más importantes hoy en día, y se usa para aplicaciones en distintas tecnologías y plataformas, capaz de ser ejecutado en el cliente (el código .NET sólo se ejecuta del lado del servidor).

JavaScript (JS) es un lenguaje liviano e interpretado, orientado a objetos con funciones de primera clase, más conocido como el lenguaje de script para páginas web.
Contrariamente a la falsa idea popular, JavaScript no es "Java interpretado"; es un lenguaje de programación dinámico que soporta construcción de objetos basado en prototipos. La sintaxis básica es similar a Java y C++ con la intención de reducir el número de nuevos conceptos necesarios para aprender el lenguaje.

Normalmente un desarrollador ASP.NET no necesitaba tener más que nociones del mismo, ya que los controles de servidor generan el código JavaScript del lado del cliente en forma automática. Esto es válido para este curso también.

Sin embargo, hoy día la arquitectura MVC si requiere mucho mayor por parte del desarrollador, que el necesario para la arquitectura Web Forms.

Para más información se sugiere ver estos links: JavaScript y www.javascript.com.

Css

Son las siglas de **Cascade Style Sheets** (hojas de estilo en cascada). Es un lenguaje que se utiliza para separar en una página Web la estructura formada por código HTML en sí de su apariencia visual.

Las aplicaciones ASP.NET basadas en Web Forms soportan el uso a CSS, aunque para la última versión (**CSS3**), el soporte puede decirse que es limitado.

El uso correcto de CSS puede ser bastante más complicado de lo que parece a simple vista. En ese curso se utilizará un framework que simplifica enormemente su uso, denominado **Bootstrap.**

De cualquier forma es necesario manejar al menos 3 conceptos básicos:

- **elementos:** el elemento HTML referenciado.
- **identificadores:** sólo pueden aparecer una única vez en una página.
- **clases:** pueden aparecer tantas veces como sea necesario.

A lo largo del desarrollo de la aplicación integradora se irán utilizando tanto elementos de HTML como de CSS y en algún ejemplo eventualmente JavaScript. Si bien no es el objetivo del curso convertirte en experto en ninguna de estas tecnologías, es importante que comprendas su uso.

Para más información se sugiere ver estos links: CSS y bootstrap.

Material de referencia

Además de los sitios oficiales antes mencionados, recomendamos el sitio W3Schools que contiene una muy buena referencia y tutoriales sobre HTML, CSS y JavaScript.

Instalar IIS en Windows

https://technet.microsoft.com/es-es/library/cc725762(v=ws.11).aspx
https://www.netveloper.com/instalar-iis-en-windows-10

Herramienta de Registro de ASP.NET para IIS

Correr una única vez luego de instalar IIS

Iniciar una línea de comandos, Inicio CMD botón derecho “Run As Administrator” (Ejecutar como administrador)

C:\> **CD C:\Windows\Microsoft.NET\Framework\v4.0.30319**

C:\Windows\Microsoft.NET\Framework\v4.0.30319> **Aspnet_regiis –i**

Distribuir (Deploy) de una Aplicación en IIS

Si lo desea en esta URL encontrara 12 Tutoriales completos sobre cómo distribuir aplicaciones Web. Aunque nuestra recomendación es que los efectúe al finalizar el curso, cuando sus conocimientos sean más amplios. https://www.asp.net/web-forms/overview/deployment/visual-studio-web-deployment/introduction

Recomendaciones:

Para profundizar y complementar los contenidos expuestos en esta clase, se sugiere abordar los siguientes cursos:

- HTML5: Fundamentos de una página Web
- Maquetación Web: HTML 5 y CSS
- Programming in HTML5 with JavaScript and CSS3 (20480)
- Introducción a la programación en Javascript
- Javascript Desarrollador Avanzado Front-End
- Developing ASP.NET MVC 5 Web Applications (20486)