

AJAX EN ASP.NET

Lenguaje de programación 2



16 de marzo de 2018

UNIVERSIDAD DON BOSCO

PILET

**Integrantes:**

Carlos Alexander Lemus Guardado LG171618

INDICE:

[Introducción: 2](#_Toc507875102)

[Ajax: 3](#_Toc507875103)

[Ventajas: 3](#_Toc507875104)

[Desventajas: 3](#_Toc507875105)

[Tecnología: 3](#_Toc507875106)

[AJAX con ASP.NET: 4](#_Toc507875107)

[Arquitectura del lado del cliente: 4](#_Toc507875108)

[Componentes: 4](#_Toc507875109)

[Compatibilidad de exploradores: 4](#_Toc507875110)

[Conexión de red: 5](#_Toc507875111)

[Servicios principales: 5](#_Toc507875112)

[Arquitectura del lado del servidor: 5](#_Toc507875113)

[Compatibilidad con los Script: 5](#_Toc507875114)

[Localización: 5](#_Toc507875115)

[Servicios web: 5](#_Toc507875116)

[Controles de servidor: 5](#_Toc507875117)

[Control Toolkit: 5](#_Toc507875118)

[ScriptManager: 6](#_Toc507875119)

[Indicaciones de su uso: 6](#_Toc507875120)

[Datos importantes: 6](#_Toc507875121)

[Administración de los errores: 6](#_Toc507875122)

[Registros de scripts: 6](#_Toc507875123)

[Registros de servicios web: 6](#_Toc507875124)

[ScriptManagerProxy: 7](#_Toc507875125)

[UpdatePanel: 7](#_Toc507875126)

[UpdateProgress: 8](#_Toc507875127)

[Datos importantes: 8](#_Toc507875128)

[Timer: 8](#_Toc507875129)

[Datos Importantes: 8](#_Toc507875130)

[Bibliografía: 9](#_Toc507875131)

# Introducción:

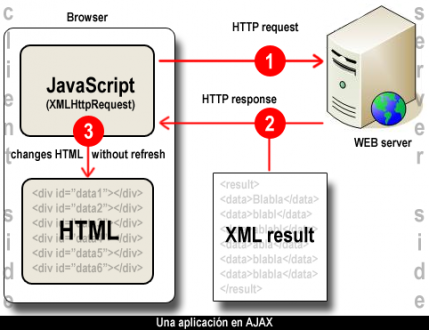
Una página web convencional maneja él envió de datos de forma síncrona, es decir cada vez que necesitamos tener la información más reciente del servidor necesitaremos recargar por completo la página, lo cual le quita dinamismo a esta, sin embargo existe el estándar del AJAX con el cual seremos más que capaces de cambiar esta situación, logrando que las paginas se acerquen más al comportamiento de las aplicaciones de escritorios.

Con AJAX lograremos que las paginas se manejen de forma asíncrona, utilizando lo que es JavaScript obteniendo los datos desde el servidor y almacenarlos en variables especiales para así decidir en qué momento, donde y como mostrar los datos dentro del DOM.

Aunque AJAX proporciona una gran manera de dar dinamismo a las páginas web, el usarlo puede ser algo complicado en especial a los nuevos desarrolladores e incluso logar grandes cosas con JavaScript puro puede ser complicado sin mencionar el tiempo de desarrollo a invertir, frente a esta problemática ASP.NET nos proporciona una serie de componentes que al juntarlos le dan una nueva potencia a la pagina web.

# Ajax:

AJAX es la función de JavaScript con XML la cual trabaja de manera asíncrona, utilizado para la creación de sitio web dinámicos, con la principal ventaja de que el cliente no necesita esperar a la repuesta del servidor para continuar con su trabajo.

Todas las páginas web pueden pedir información al servidor por medio de enlaces remplazando la vieja por los datos ya actualizados, teniendo en cuenta claro que para tener dicha información se recarga por completo el sitio web, sin embargo el uso de AJAX permite que la repuesta del servidor sea almacenada en una variable de JavaScript esto con el fin que podamos hacer lo que queramos con dicha información, uno de los ejemplos más sencillos seria el pedir la hora desde el servidor y en lugar de recargar por completo la página, se recargue solo la sección donde se vaya a mostrar la hora.

## Ventajas:

* El sitio web se asemeja al funcionamiento de las aplicaciones de escritorios, pues los cambios en la página se producen dinámicamente, logrando una mejor interacción con el usuario.
* AJAX posee una alta compatibilidad con la mayoría de aplicaciones web.
* Se consume menos ancho de banda y una menor carga para el servidor.
* Se logra una mayor rapidez al momento de hacer las transacciones pues son realizadas individualmente en lugar de hacerlas toda de una vez al recargar la página.

## Desventajas:

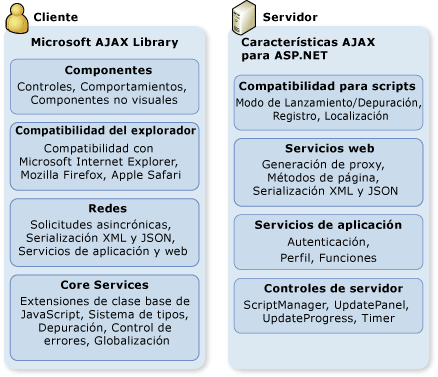
* El usuario se despista rápidamente con el uso de la página web, pues es un cambio algo brusco con la funcionalidad convencional de los sitios web, por ejemplo en la URL ya no se apreciaran los campos de filtros al momento de realizar una petición, el botón de retroceso no se manejara de la misma manera, será obligatorio mostrar que ha ocurrido un cambio en la página.
* Problema con el posicionamiento de la página web con los buscadores.
* La mayor desventaja se trata en el tiempo de desarrollo que lleva.
* No se muestran la información de las URL usadas.

## Tecnología:

Hay que resaltar que AJAX no fue la primera en implementar la característica de recargar la página dinámicamente, sin embargo fue AJAX aquella que se terminó estandarizando usando el XMLHttpRequest para el procesamiento de las peticiones, vale mencionar que no solo la tecnología XML y JavaScript intervienen en el proceso sino también los protocolos HTTP para el recibimiento de la información.

# AJAX con ASP.NET:

ASP.NET utiliza AJAX para no tener que recurrir a recargar toda la página y así lograr la creación de páginas web dinámicas, vale resaltar que utiliza AJAX en conjunto con los servidores IIS, para ello se nos presentan diversas herramientas para la facilitación de su incorpora miento como son las librerías Script.



Ventajas que nos presenta la integración AJAX en ASP.NET:

* Ventajas del uso del AJAX.
* Mayor eficiencia.
* Se nos presenta información de las herramientas al momento de usarla.
* Posee compatibilidad con la mayoría de navegadores.
* Las clases proxy generada automáticamente.
* Se nos permite la personalización de los controles.

## Arquitectura del lado del cliente:

Incorpora las bibliotecas para:

### Componentes:

Los componentes son aquellos que presentan un comportamiento enriquecido sin la devolución de datos, estos pueden ser tanto elementos del DOM como elementos personalizados, sin embargo es el usuario quien decide cual será la mejor opción a usar.

### Compatibilidad de exploradores:

El scripting de AJAX se puede escribir una única vez, pues este es adaptado a todos los tipos de navegadores.

### Conexión de red:

Para recargas parciales a la página se emplea el control UpdatePanel el cual realiza la conexiona la red de manera automática, quitándonos el peso de tener de programar el comportamiento necesario

### Servicios principales:

Las bibliotecas Script están compuestas por archivos JavaScript por lo que trae compatibilidad con la programación orientada a objetos, siendo un código extremadamente limpio cumpliendo con los pilares de la Programación Orientada a Objetos.

## Arquitectura del lado del servidor:

Compuesta por componentes web, manejo de la interfaz y el flujo de la aplicación contando con el manejo de la serializacion, validación, autenticación de formularios y el manejo de la sesiones.

### Compatibilidad con los Script:

Los script son enviados al cliente por el servidor, pudiendo enviar varios scripts y ejecutarlos en distintas zona de la página, contando con la opción de generar sus propios script personalizados, sin embargo una de sus mayores cualidades radica en la capacidad que puede poseer para el manejo de las excepciones como de la depuración.

### Localización:

AJAX se genera sobre la base del modelo de localización, dando de ese modo una cualidad adicional a los archivos JavaScript.

### Servicios web:

Se puede utilizar el script del cliente para ejecutar los servicios web como de la propia Microsoft y son las referencias agregadas automáticamente a la página al igual que los proxys.

### Controles de servidor:

Compuesto por código servidor y clientes con el fin de generar código enriquecido para el cliente, siendo la página quien envía automáticamente el script de cliente de soporte al explorador.

Controles de servidor de AJAX en ASP.NET:

* ScriptManager
* UpdatePanel
* UpdateProgress
* Timer

## Control Toolkit:

Es una colección de controles con la que se muestra la capacidad de los controles y extensiones AJAX, con la que se puede crear y modificar fácilmente los controles.

# ScriptManager:

El ScriptManager es el componente fundamental para el uso de AJAX, pues es el administrador del lado del cliente para el uso de AJAX con lo cual seremos capaces de manejar las actualizaciones parciales de la página, los servicios web y básicamente gestionar los recursos producidos por la pagina como son los scripts, estilos, campos ocultos y matrices, por medio del uso en conjunto del UpdatePanel.

El uso del ScriptManager nos permite:

1. Nos permite decidir si enviamos script de lanzamientos o depuración.
2. Nos facilita el acceso a los servicios web.
3. La propiedad ResourceUICultures nos da acceso a los recursos de localización para los archivos script.
4. Se puede representar en el ScriptManager aquellos servicios web que implementan de IExtenderControl o IScriptControl.

## Indicaciones de su uso:

Según el propio Microsoft se nos indica que se debe usar el ScriptManager en los siguientes casos:

* Cuando necesitemos enviar al explorador cualquier script para su manejo.
* Cuando se necesite recargar una sección específica de la página.
* El uso de proxy para cuando necesitemos usar los servicios web y en los métodos marcados en las páginas asp.net
* En necesitad de emplear la autenticación, perfil y funciones de ASP.NET.

## Datos importantes:

Dentro de la página solo puede existir un ScriptManager en su jerarquía sin embargo somos más que capaces de emplear el componente ScriptManagerProxy, el cual es verdaderamente útil al momento de ingresar componente anidados a la página maestra.

Aunque se encuentre un solo ScriptManager en toda la página con diversos UpdatePanel, es el propio ScriptManager quien gestiona todos los componentes dentro de los UpdatePanel, claro, esto será así durante toda la vida útil de la aplicación.

### Administración de los errores:

* Usando la propiedad AllowCustomErrorsRedirect para indicar por medio del web.config el cómo se manejara las excepciones al momento de recibir la información.
* El evento AsyncPostBackError es lanzado al momento que ocurre un error cuando se devuelve la información.
* La propiedad AsyncPostBackErrorMessage es aquella con la que se nos permite mostrar un mensaje en caso de cualquier error.

### Registros de scripts:

Contamos con el ScriptReference el cual posee un objeto de cada script disponible en el explorador, si los script registrados usan actualizaciones parciales debemos de usar los métodos de registros del ScriptManager.

### Registros de servicios web:

Debemos de agregar los servicios a la colección Services, para ello generamos un objeto proxy por cada servicio referenciado, los servicios pueden ser añadidos en tiempo de ejecución.

# ScriptManagerProxy:

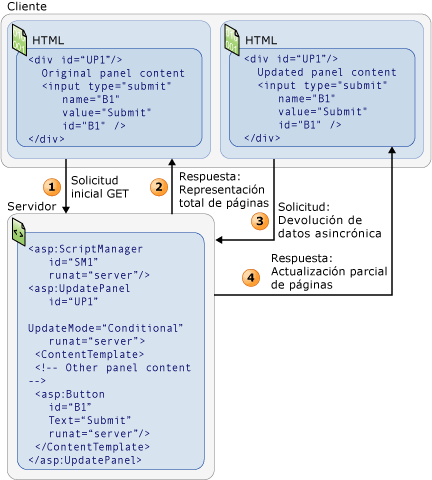
Las paginas ASP.NET solo aceptan un ScriptManager ya sea dentro de la propia página o por algún componente externo, por lo que si queremos que algún otro componente use las características del ScriptManager necesitaremos del componente ScriptManagerProxy es aquel que permite a los elementos anidados de una página puedan añadir lo que son secuencia de comandos y referencias a servicios definidos por el ScriptManager.

El ScriptManagerProxy requiere de una configuración específica por parte del web.config teniendo en cuenta que los errores de no tener una buena configuración se notara en la vista de diseño.

# UpdatePanel:

Se conoce como actualización parcial a la acción de actualizar una sección específica de la página web en lugar de que se recargue toda la página, para ello usamos el UpdatePanel que junto con el ScriptManager logramos la actualización parcial sin la necesidad de script personalizados.

La principal ventaja del uso del UpdatePanel radica en su simplicidad de uso, pues en páginas normales para lograr que no se recargue toda la página es necesario un alto conocimiento de JavaScript, pero en ASP.NET solo necesitaremos el arrastrar dos componentes sin la necesidad de un script personalizado pero con todas las ventajas del uso de AJAX.

Los UpdatePanel definen que regiones de la página son las que contaran con la capacidad, esto es gestionado por el ScriptManager y la clase PageRequestManager.

Los controles dentro del UpdatePanel realizan una llamada de datos de forma asíncrona lo cual nos permite que solo los elementos dentro del UpdatePanel se vean afectado por la llamada y es el PageRequestManager quien realiza una manipulación en el DOM.

Contamos con la propiedad UpdateMode la cual en caso que se establezca en Always el contenido que se encuentre adentro se actualizara por la llamada desde cualquier parte de la página ya sea que estén adentro o afuera de cualquier UpdatePanel, sin embargo en caso de que se establezca como conditional solo se actualizará en los siguientes casos:

* Al momento que un desencadenador realiza una devolución de datos.
* Usando el método update.
* Por medio UpdatePanel anidados.

# UpdateProgress:

Este control nos proporciona información sobre las actualizaciones parciales procedente del UpdatePanel, representando un elemento “<DIV>” el cual dependiendo del estado de la devolución de datos asíncrona puede aparecer o se puede ocultar, vale aclarar que solo en devoluciones de datos de tipo asíncrono será en la que aparezca.

Como hemos señalado el UpdateProgress depende de un UpdatePanel existente para poder funcionar, para enlazarlo con algún UpdatePanel debemos de usar el atributo AssociatedUpdatePanelID cuando en dicho UpdatePanel se produce algún cambien se ve reflejado dentro del UpdateProgress.

## Datos importantes:

* Al momento de que se inicia una devolución asíncrona y teniendo la propiedad ChildrenAsTriggers de un UpdatePanel en false, se mostraran los controles UpdateProgress que estén asociados.
* Se recomienda utilizar para mostrar el estado actual de la conexión cuando esta sea demasiado lenta.
* Un único UpdateProgress puede estar asociado a distintos controles UpdatePanel o en su defecto un único UpdateProgress por cada UpdatePanel existente.
* Cuenta con la propiedad DynamicLayout con la cual en lugar de ocupar un espacio definido en la página este se ajusta automáticamente según su contenido.
* Un control UpdateProgress puede estar dentro o fuera del control UpdatePanel al cual está asociado, esto sin ningún problema a la hora e visualizar.
* ASP.NET nos presentas los eventos beginRequest y endRequest con los cuales podemos definir en qué momento queremos que se muestre los controles en periodo de ejecución.

# Timer:

Como su nombre indica es el control encargado de la recarga de datos parciales e incluso de la recarga completa de la página en una serie de intervalos definidos, siendo un control de servidor que pone código JavaScript dentro de la página y es en esta parta donde indica en qué momento se dará la devolución de los datos por medio de la propiedad Interval, siendo obligatorio el uso del ScriptManager para logar esto.

El control Timer cuenta con el evento Tick del lado del servidor el cual es activado al momento de enviar una página al servidor.

## Datos Importantes:

* Se puede ejecutar código al momento que se envíen los datos.
* Actualización asíncrona de la página por medio de intervalos.
* Si el control Timer está fuera del UpdatePanel se debe definir este como un desencadenante en cambio si el control se encuentra dentro de dicho UpdatePanel este automáticamente se establece como un desencadenante.
* El “cronometro” del Timer empieza a correr una vez haya empezado la devolución de datos por lo que si el tiempo de espera es de unos 60 segundos y este tarda 5 segundo en mostrar los datos solo contara con 55 segundos para observar los cambios realizados.
* Si hay una devolución de datos e inicia una nueva, esta es cancelada.

# Bibliografía:

<https://www.digitallearning.es/blog/que-es-ajax/>

<http://www.4rsoluciones.com/blog/ventajas-y-desventajas-de-utilizar-ajax-2/>

<https://www.jose-aguilar.com/blog/ventajas-e-inconvenientes-de-utilizar-ajax/>

<https://www.adictosaltrabajo.com/tutoriales/begin-ajax/>

<https://msdn.microsoft.com/es-es/library/bb398874(v=vs.100).aspx>

<https://msdn.microsoft.com/es-es/library/bb398863(v=vs.100).aspx>

<https://msdn.microsoft.com/es-es/library/system.web.ui.scriptmanager(v=vs.90).aspx>

<https://msdn.microsoft.com/en-us/library/cc295370.aspx>

<https://msdn.microsoft.com/es-es/library/bb399001(v=vs.100).aspx>

<https://msdn.microsoft.com/es-es/library/bb386454(v=vs.100).aspx>

https://msdn.microsoft.com/es-es/library/bb398821(v=vs.100).aspx