Programación Web Avanzada

Ajax- Ejemplo Práctico

Universidad Nacional del Comahue
Facultad de Informática

Ajax

- Asynchronous JavaScript And XML.
- Técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications).
- Estas aplicaciones se ejecutan en el cliente, mientras se mantiene la comunicación asíncrona con el servidor en segundo plano.

Asíncrono o en Segundo Plano

- Se actualiza la página sin la necesidad de recargar toda la página.
- Se ahorra ancho de banda porque el tamaño de la información que se transfiere es menor.
- Aplicaciones mas interactivas

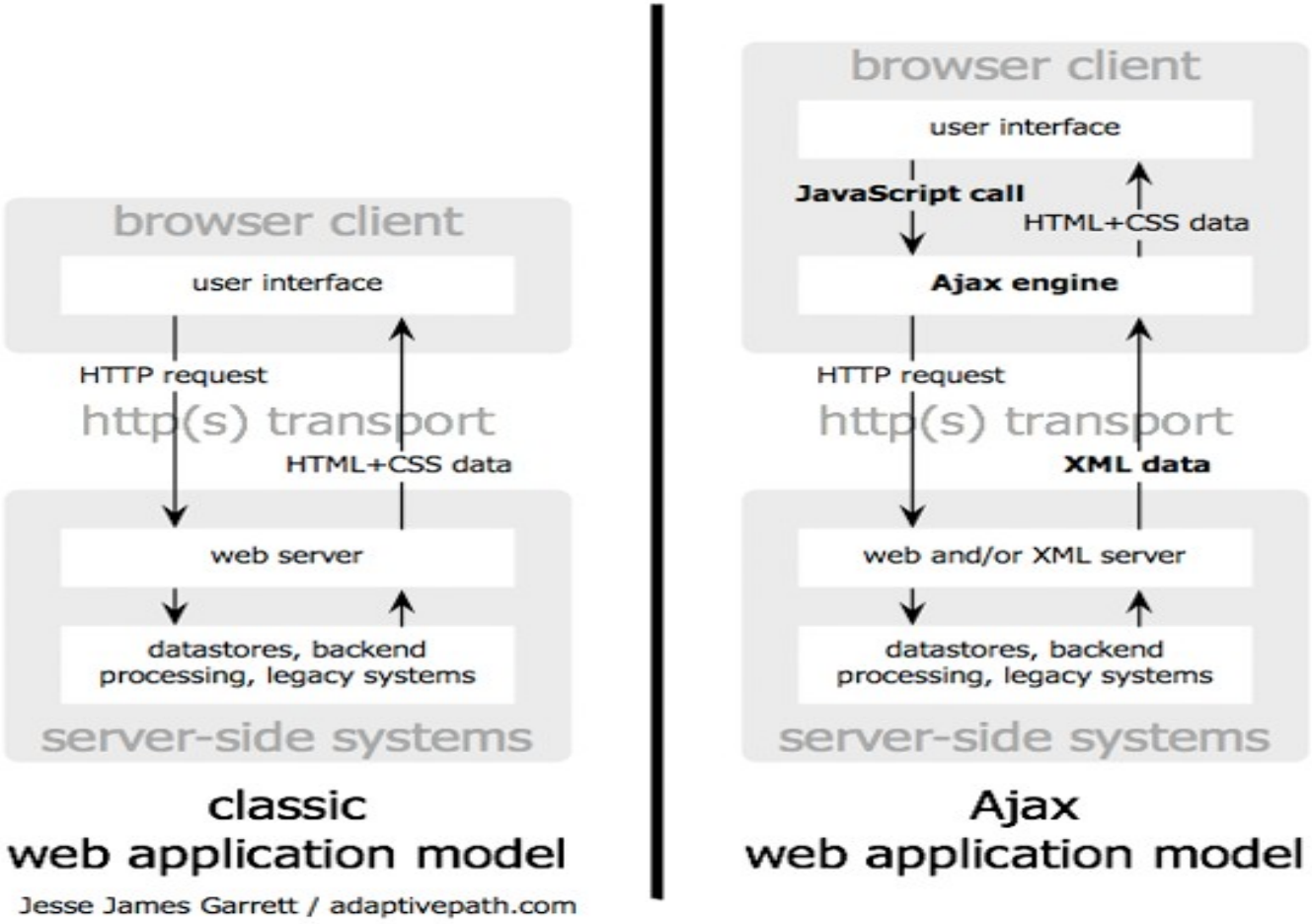
Que es Ajax? Ejemplos

- Combo asociado. País – Ciudad
- Validación de formularios en linea.
- Chat en linea.
- Tablas paginadas.
- Tabs.
- Etc.....

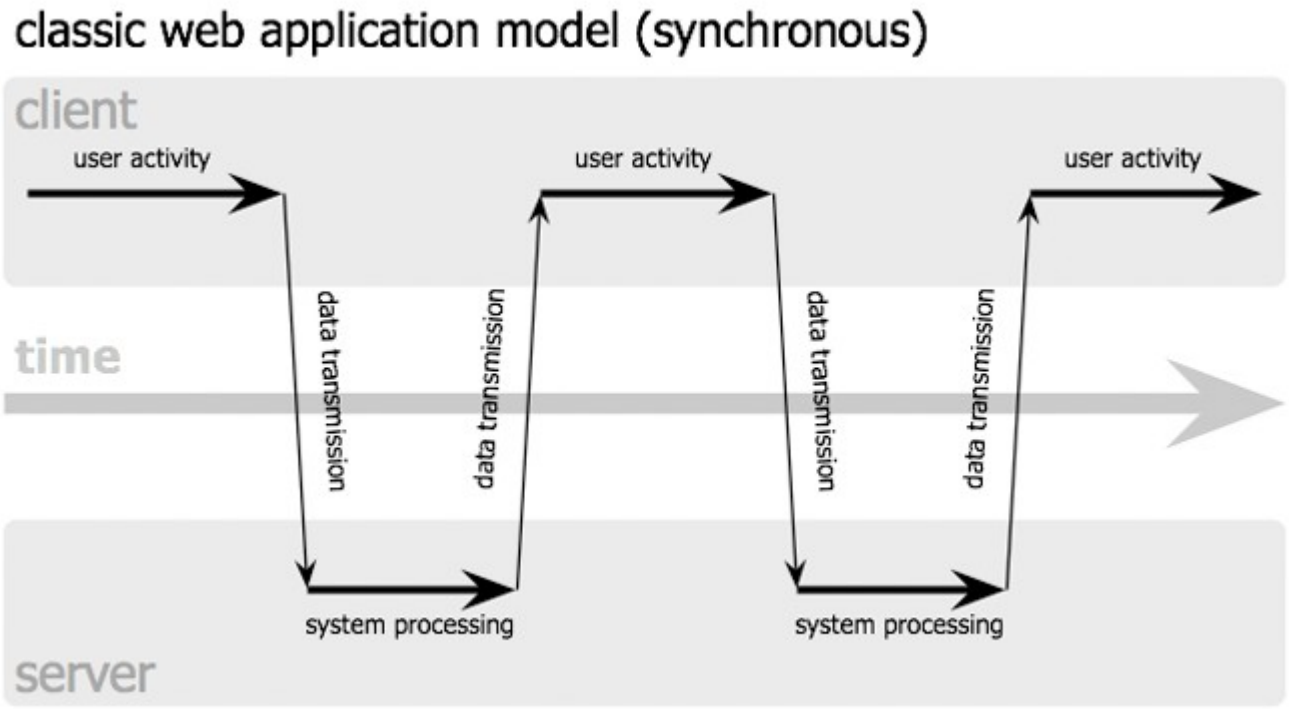
Historia

- 1996. Recarga del elemento iframe. introducido en Internet Explorer 3. (esta clase parte 1)
- 2000. Objeto XMLHttpRequest, Internet Explorer 5. (esta clase parte 2)
- 2005. Creación del termino “Ajax” por Jesse James Garrett.
- 2006..... Librerias de implementación de Ajax. (esta clase parte 3)
- 2011 ej Qooxdoo todo es Ajax http://demo.qooxdoo.org/current/playground/#

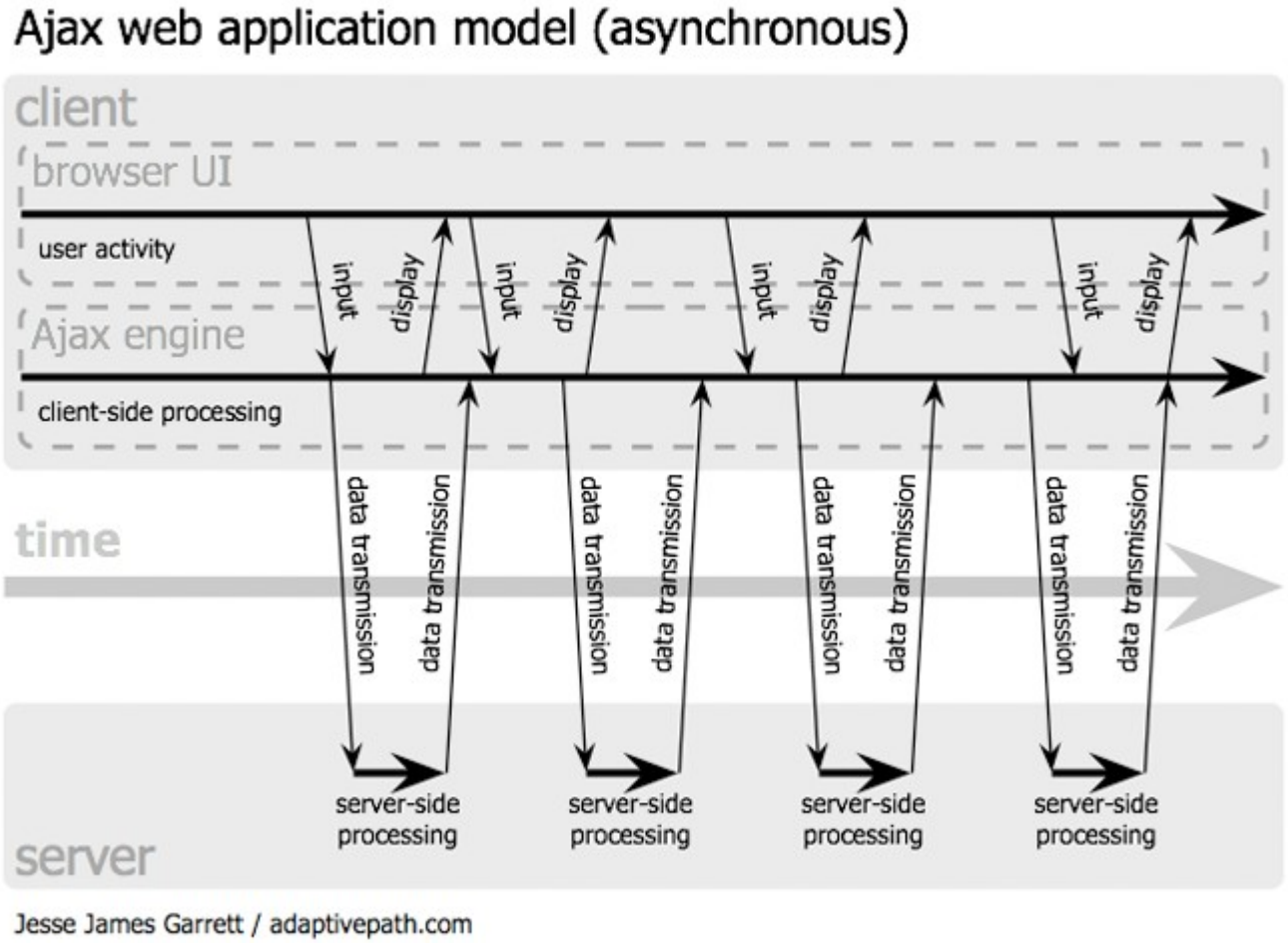
Arquitectura



Aplicaciones Web Clásicas

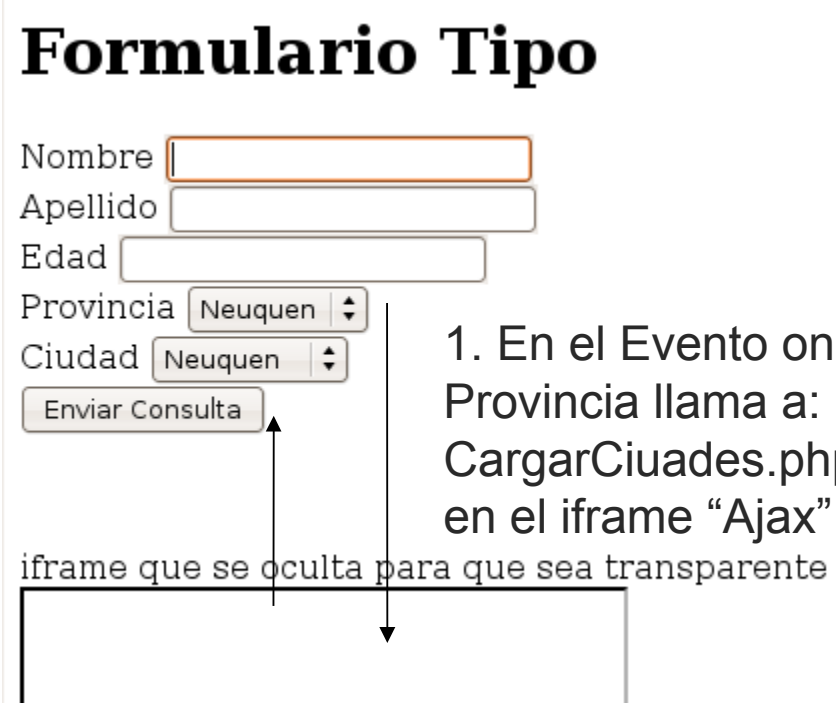


Aplicaciones Web con el concepto de Ajax



Ejemplo combo asociado – Iframe

- Pagina inicial



1. En el Evento onchange del combo Provincia llama a: CargarCiuades.php?Provincia=[Provincia] en el iframe “Ajax”

2. El script php imprime código js para actualizar el combo Ciudad según $_Get['Provincia']

XMLHttpRequest

JavaScript es el lenguaje en el que normalmente se efectúan las funciones de llamada de Ajax mientras que el acceso a los datos se realiza mediante el objeto XMLHttpRequest

XMLHttpRequest

- XMLHttpRequest es una clase empleada para realizar peticiones HTTP y HTTPS a servidores WEB
- Para los datos transferidos se usa cualquier codificación basada en texto, texto plano, XML, HTML
- Se pueden generar tantas instancias como necesite para manejar el diálogo con el servidor.

XMLHttpRequest

- Un poco de Historia:
  - La primera versión fue desarrollada por Microsoft que la introdujo en la versión 5.0 de Internet Explorer como un objeto ActiveX. A partir de Internet Explorer 7 se ofrece de manera integrada
  - Mozilla incorporó la primera implementación integrada en la versión 1.0 de la Suite Mozilla en 2002.
  - Luego seguiría Apple a partir de Safari 1.2, Konqueror, Opera Software a partir del Opera 8.0 e iCab desde la versión 3.0b352.
  - El 25 de febrero de 2008 se publicó la primera versión de la especificación estándar XMLHttpRequest Level 2

XMLHttpRequest

- Constructor:

var req = new XMLHttpRequest();

En caso de versiones de IE anteriores a IE7:

var req = new ActiveXObject("Microsoft.XMLHTTP");

XMLHttpRequest

- En capsulando el comportamiento del constructor:

```
//función para crear un objeto XMLHttpRequest según el navegador que corresponda
function objetoAjax(){
   var xmlhttp=false;
   if(window.XMLHttpRequest)
   { // Navegadores que siguen los estándares Firefox, Safari, Opera, Internet Explorer 7 y 8
      xmlhttp = new XMLHttpRequest();
   }
   else
      if(window.ActiveXObject)
      { // Navegadores obsoletos Internet Explorer 6 y anteriores
         xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
      }
   return xmlhttp;
}
```

XMLHttpRequest

- Atributos:

Atributo	Descripción
`readyState`	Devuelve el estado del objeto como sigue: 0 = sin inicializar, 1 = abierto, 2 = cabeceras recibidas, 3 = cargando y 4 = completado.
`responseBody`	*(Level 2)* Devuelve la respuesta como un array de bytes
`responseText`	Devuelve la respuesta como una cadena
`responseXML`	Devuelve la respuesta como XML. Esta propiedad devuelve un objeto documento XML, que puede ser examinado usando las propiedades y métodos del árbol del DOM.
`status`	Devuelve el estado como un número (p. ej. 404 para "Not Found" y 200 para "OK").
`statusText`	Devuelve el estado como una cadena (p. ej. "Not Found" o "OK").

XMLHttpRequest

- Métodos:

Método	Descripción
`abort()`	Cancela la petición en curso
`getAllResponseHeaders()`	Devuelve el conjunto de cabeceras HTTP como una cadena.
`getResponseHeader( nombreCabecera )`	Devuelve el valor de la cabecera HTTP especificada.
`open` `( método, URL [, asíncrono [, nombreUsuario [, clave]]] )`	Especifica el método, URL y otros atributos opcionales de una petición. El parámetro de método puede tomar los valores "GET", "POST", ("GET" y "POST" son dos formas para solicitar datos, con "GET" los parámetros de la petición se codifican en la URL y con "POST" en las cabeceras de HTTP). El parámetro *URL* puede ser una URL relativa o completa. El parámetro *asíncrono* especifica si la petición será gestionada asíncronamente o no. Un valor *true* indica que el proceso del script continúa después del método send(), sin esperar a la respuesta, y *false* indica que el script se detiene hasta que se complete la operación, tras lo cual se reanuda la ejecución. En el caso asíncrono se especifican manejadores de eventos, que se ejecutan ante cada cambio de estado y permiten tratar los resultados de la consulta una vez que se reciben, o bien gestionar eventuales errores.
`send([datos])`	Envía la petición
`setRequestHeader( etiqueta, valor )`	Añade un par etiqueta/valor a la cabecera HTTP a enviar.

XMLHttpRequest - Ejemplo

```
function actualizarComboCiudad(provincia){
var divTarget = document.getElementById("Actualizar");

//instanciamos el objetoAjax
var ajax = objetoAjax();
//uso del medotod GET
ajax.open("GET","cargarCiudades.php?Provincia="+provincia);

 //-- Ahora estamos esperando a recibir la respuesta para eso tenemos la siguiente función
  //-- que llamamos cada vez que ocurre que el estado cambia
ajax.onreadystatechange=function() {
        //-- Funcion anonima a ejecutar cada vez que el estado de la peticion cambia. Cuando el estado es 4,
        se completo
        if (ajax.readyState==4) {
                //mostrar resultados en esta capa
                divTarget.innerHTML = ajax.responseText;
        }
  }
//como hacemos uso del metodo GET colocamos null
//Con Post en lugar de null deberia mandarle la cadena con los parametros y sus valores Seria
“Provincia=”+Provincia
ajax.send(null);
}
```

XMLHttpRequest - Ejemplo

```
function actualizarComboCiudad(provincia){
var divTarget = document.getElementById("Actualizar");

//instanciamos el objetoAjax
var ajax = objetoAjax();
//uso del medotod GET
ajax.open("POST","cargarCiudades.php");

 //-- Ahora estamos esperando a recibir la respuesta para eso tenemos la siguiente función
  //-- que llamamos cada vez que ocurre que el estado cambia
ajax.onreadystatechange=function() {
        //-- Funcion anonima a ejecutar cada vez que el estado de la peticion cambia. Cuando el estado es 4,
        se completo
        if (ajax.readyState==4) {
                //mostrar resultados en esta capa
                divTarget.innerHTML = ajax.responseText;
        }
  }

ajax.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
//Con Post en lugar de null deberia mandarle la cadena con los parametros y sus valores Seria
"Provincia="+Provincia
ajax.send("Provincia="+provincia);
}
```

Referencias

Tutorial

http://www.w3schools.com/ajax

Articulo creador del concepto

http://www.adaptivepath.com/ideas/essays/archives/000385.php

XMLHttpRequest Level 2

http://www.w3.org/TR/XMLHttpRequest2/