



# Ajax



- AJAX
  - Asynchronus JavaScript + XML
  - También a sido descrito como “Dynamic HTML and remote scripting”
  - Interfaces para clientes enriquecidos
    - Es más complicado que diseñar una página web
  - “Enriquecido” se refiere al modelo de interacción con el cliente
    - Puede soportar una amplia variedad de métodos de entrada
    - Puede responder intuitivamente y oportunamente
  - Web browser
    - Son clientes contactando a servidores web
    - Tiene algunas funcionalidades: botón “back”, history, tabs...



- Páginas web
  - Se consideran como una aplicación
- 4 principios que definen AJAX
- (1) El browser recibe y mantiene una aplicación, no contenido
  - Aplicaciones clásicas
    - En las aplicaciones web clásicas el browser sólo es un “terminal tonto”
    - El servidor mantiene toda la información: sesión de usuario
    - El browser recibe los documentos y los despliega, cada vez que recibe uno nuevo hace lo mismo
  - Aplicaciones AJAX mueven lógica de la aplicación hacia el browser



- (1) El browser recibe y mantiene una aplicación, no contenido
  - El browser recibe un documento más complejo, con bastante código Javascript
- (2) El servidor entrega datos, no contenido
  - El servidor solo envía datos relevantes, no repetitivos
  - Aplicación Ajax puede hacerlo por medio de trozos de código javascript, stream de texto o XML
  - Se reduce el uso de ancho de banda comparado con las aplicaciones tradicionales



- (3) Interacción del usuario con la aplicación puede ser fluida y continua
  - Web browser provee dos mecanismos de entrada de datos: hyperlinks y formularios HTML
  - Mientras la página esta siendo enviada al servidor, el usuario está en el limbo
    - La antigua página se mantiene visible por mientras
    - El browser permite que el usuario siga haciendo click en los controles que se mantienen a la vista
  - Con Ajax se pueden conectar eventos para acciones del usuario
    - “Drag and drop” similar a las aplicaciones de escritorio
    - El servidor trabaja en conjunto con el usuario, ante cada evento que realiza en la interfaz



- (4) Este es código real, necesita disciplina
  - Aplicaciones web clásicas limitan el uso de Javascript para validaciones y controles muy simples
    - Javascript ganó una reputación de trivial
  - Una aplicación Ajax es completamente diferente
    - El código generado es utilizado desde que el usuario lanza la aplicación hasta que la cierra
      - No debe bajar la performance ni memory-leaks
  - Se debe escribir código con alta performance y mantenible
    - Se debe usar la misma disciplina que en el desarrollo en el lado del servidor
  - Aplicación Ajax es una aplicación funcional compleja
    - Se comunica eficientemente con el servidor



- Implementación
  - La clave es el objeto XMLHttpRequest
  - Actualmente todos los browser utilizan XMLHttpRequest, IE5 e IE6 utilizan ActiveXObject
  - Se debe averiguar que tipo de objeto soporta el browser
  - Este objeto realiza la comunicación con el servidor
  - Antes de enviar datos al servidor, se debe trabajar con las propiedades de XMLHttpRequest



- Propiedades de XMLHttpRequest

- onreadystatechange

- Permite definir la función que recibirá y procesará los datos desde el servidor
    - Esta función es almacenada en esta propiedad para ser invocada automáticamente

```
xmlhttp.onreadystatechange=function()  
{  
    // We are going to write some code here  
}
```

- readyState

- Mantiene el estado de la respuesta del servidor
    - Cada vez que cambia el estado de readystate, la función onreadystatechange es ejecutada





## – readyState: Posibles estados

State	Description
0	The request is not initialized
1	The request has been set up
2	The request has been sent
3	The request is in process
4	The request is complete

## – Se necesita un if en onreadystatechange para analizar si la respuesta está completa:

```
xmlhttp.onreadystatechange=function()  
{  
  if(xmlhttp.readyState==4)  
  {  
    // Get data from the server's response  
  }  
}
```



## –.responseText

- Los datos enviados desde el servidor se pueden recuperar desde el.responseText:

```
xmlhttp.onreadystatechange=function()  
{  
if(xmlhttp.readyState==4)  
    {  
        document.myForm.time.value=xmlhttp.responseText;  
    }  
}
```



- Enviando solicitudes al servidor
  - Se utiliza los métodos “send()” y “open()”
  - Método open recibe 3 argumentos:
    - Método para enviar la solicitud (GET o POST)
    - URL del programa que se ejecutará en el servidor
    - Indicar si el método se debe manejar de manera asíncrona

```
xmlhttp.open("GET","time.php",true);  
xmlhttp.send(null);
```

- Finalmente se debe decidir cuando la funcionalidad Ajax se debe ejecutar desde el HTML
  - Lo típico es cuando se gatilla algun evento

# Ajax



- Ejemplos
  - W3schools
    - [http://www.w3schools.com/Ajax/ajax\\_database.asp](http://www.w3schools.com/Ajax/ajax_database.asp)
  - Google Maps
    - <http://maps.google.com>
- Herramientas
  - GWT
    - <http://code.google.com/intl/es-CL/webtoolkit/overview>
    - <http://gwt.google.com/samples/Showcase/Showcas>
  - Closure
    - <http://code.google.com/intl/es-CL/closure/library/>