

DESARROLLO WEB EN ENTORNO CLIENTE

CAPÍTULO 7: **Utilización de Mecanismos de Comunicación** **Asíncrona**

Juan Manuel Vara Mesa
Marcos López Sanz
David Granada
Emanuel Irrazábal
Jesús Javier Jiménez Hernández
Jenifer Verde Marín



Objetivos

- Mecanismos de comunicación asíncrona en las aplicaciones Web.
- Tecnologías asociadas con la técnica AJAX.
- Formatos de envío y recepción de información asíncrona.
- Llamadas asíncronas.
- Librerías de actualización dinámicas actuales.

Mecanismos de comunicación síncrona

- En un proceso habitual el cliente es el que inicia el intercambio de información solicitando datos al servidor que responde enviando un flujo de datos al cliente.

Mecanismos de comunicación asíncrona

- El mecanismo de comunicación **asíncrona** recarga en **segundo plano** una **parte** de la página Web, dejando **desbloqueado** el resto.
- El cliente que envía una petición no permanece bloqueado esperando la respuesta del servidor.
- Esto ayuda a que las aplicaciones Web tengan una interactividad similar a las aplicaciones de escritorio.
- Es en parte lo que hace algunos años se denomina Web 2.0.

AJAX

- La necesidad = aplicaciones Web interactivas.
- Solución = nuevo uso a tecnologías como XML, CSS o DOM.
- En 2005 J.J. Garrett habla por primera vez sobre AJAX (*Asynchronous JavaScript And XML*) = **JavaScript asíncrono y XML**.
- Se suprimen los efectos secundarios de las recargas, como la pérdida del contexto, la ubicación del *scroll* o las respuestas más lentas.

Tecnologías en AJAX

- XHTML y CSS para una presentación basada en estándares.
- DOM para la interacción y la visualización dinámica de datos.
- XML y XSLT para el intercambio y transformación de datos.
- XMLHttpRequest para la recuperación asíncrona de los datos.
- y JavaScript como elemento de unión.

Elección de AJAX

- Uso de nuevas tecnologías y mayor complejidad.
- El comportamiento considerado lógico por el usuario al utilizar la funcionalidad de “volver a la página anterior” no es reproducido de la misma manera.
- Las aplicaciones Web o sitios Web con AJAX utilizan más recursos del servidor.
- El uso de las tecnologías asociadas con AJAX no están presentes por defecto en cualquier tipo de dispositivos.
- Aunque cada vez menos, todavía existen incompatibilidades entre navegadores.

Tecnologías involucradas: XHTML y CSS

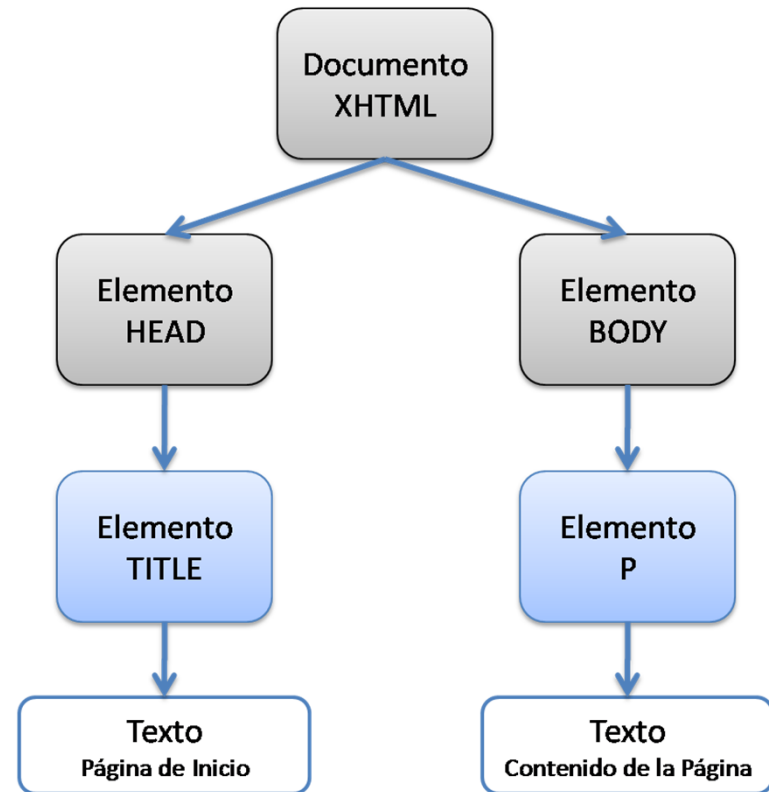
- XHTML es un HTML estándar especificado mediante un documento XML. XHTML es más riguroso con su estructura:
 - Los valores de los atributos, siempre entre comillas:
 - Incorrecto: `<td colspan=2>`.
 - Correcto: `<td colspan="2">`.
 - Los nombres de elementos y atributos deben ir en minúsculas:
 - Incorrecto: ``.
 - Correcto: ``.
 - No está permitida la minimización de atributos (se usa el nombre del atributo como valor):
 - Incorrecto: `<textarea readonly>`.
 - Correcto: `<textarea readonly="readonly">`.

Tecnologías involucradas: XHTML y CSS

- Un cambio en el CSS es reflejado instantáneamente en todas las páginas que utilizan algún elemento de presentación identificado en dicha hoja de estilo.

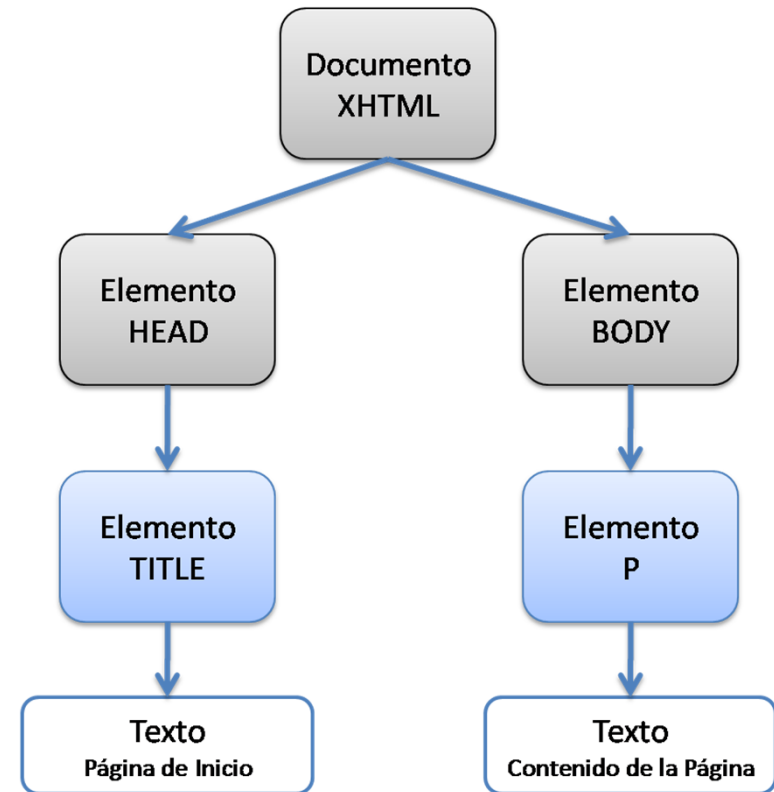
DOM

- El *Document Object Model*: (DOM) es una representación de la página Web en una estructura de jerarquía de árbol.
- Es una API para representar documentos XHTML.
- Todas las partes de la página están accesibles desde el cliente y no es necesario utilizar tecnologías del lado del servidor.

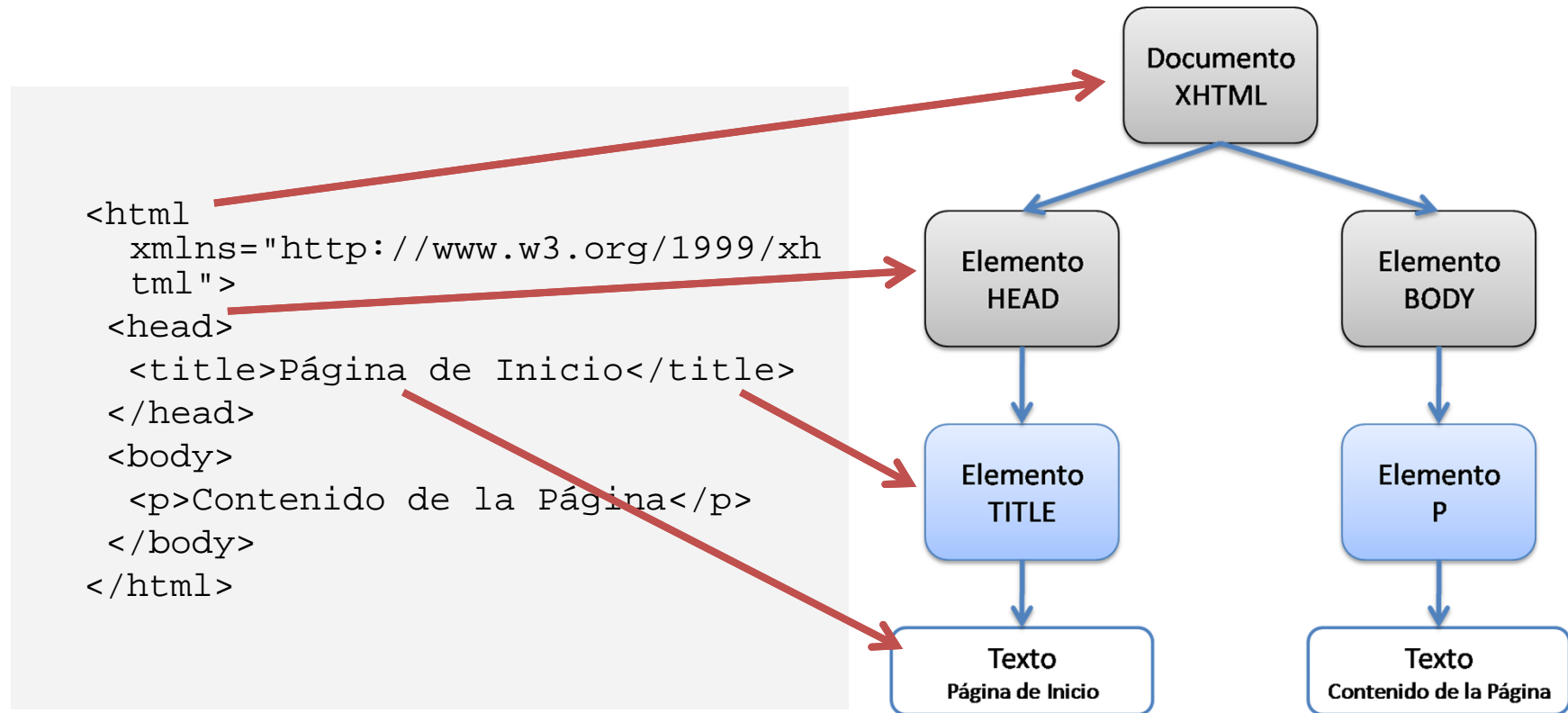


DOM

```
<html
  xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Página de Inicio</title>
  </head>
  <body>
    <p>Contenido de la Página</p>
  </body>
</html>
```



DOM



JavaScript

- El lenguaje de scripting JavaScript es el más utilizado actualmente en los navegadores.
- Es un lenguaje orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

JavaScript

- Utilización principal es en el lado del cliente.
- Permitiendo mejoras en la interfaz de usuario, páginas web dinámicas y accediendo al árbol DOM para realizar modificaciones instantáneas del código fuente.
- El código JavaScript se ejecuta en el propio navegador y puede ir formando parte del propio código HTML de dicha página.

Ejemplo de JavaScript

```
<head>
  <script type="text/javascript">
    function cambiarAltura() {
      if (novedades.style.height == "150px") {
        novedades.style.height = "25px";
      } else {
        novedades.style.height = "150px";
      }
    }
  </script>
</head>
<body>
  <div id="novedades" style="height:150px;
  border:1px;">
    <a href="#" onClick="cambiarAltura()">X</a>
    <p>Alerta: La línea de bus 25 cambia su
    recorrido</p>
  </div>
</body>
```

Ejemplo de JavaScript

```
<head>
  <script type="text/javascript">
    function cambiarAltura() {
      if (novedades.style.height == "150px") {
        novedades.style.height = "25px";
      }
      else{
        novedades.style.height = "150px";
      }
    }
  </script>
</head>
<body>
  <div id="novedades" style="height:150px;
  border:1px;">
    <a href="#" onClick="cambiarAltura()">X</a>
    <p>Alerta: La línea de bus 25 cambia su
    recorrido</p>
  </div>
</body>
```

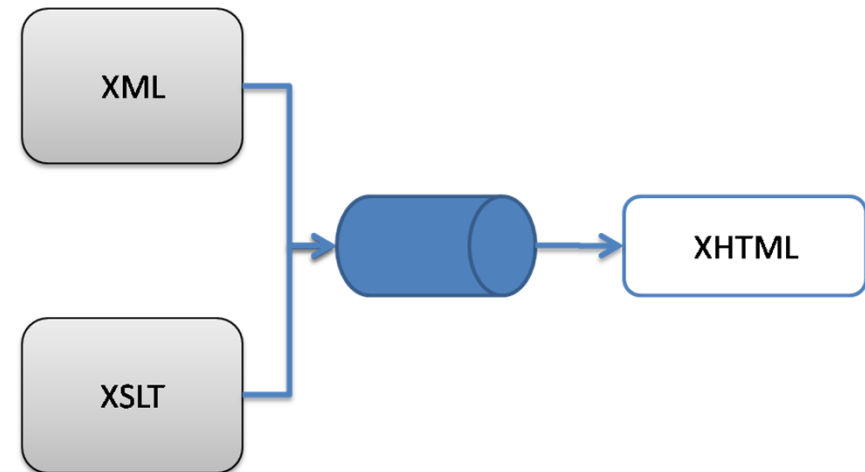


XML

- El lenguaje XML es utilizado para describir y estructurar datos.
- Alrededor de XML encontramos otras tecnologías: XQuery, XSLT.
- Los navegadores contienen funcionalidades internas para trabajar con documentos XML.

XSLT

- Lenguaje para transformar un documento XML en otro documento XML, texto plano, etc.
- Con XSLT podrías recibir un documento XML representando una tabla y transformarlo en una tabla XHTML.
- XSLT utiliza otro lenguaje, el XPath, para realizar consultas en el documento XML cuando se aplican las transformaciones.
- En este caso, XPath busca elementos dentro del XML de entrada y XSLT los procesa.



El objeto XMLHttpRequest

- Aparece a partir de Internet Explorer 5 en la forma de un control ActiveX llamado XMLHttpRequest.
- Se fueron transformando en un estándar de facto en navegadores como Firefox, Safari y Opera (pero todavía existen diferencias).
- Actualmente el objeto XMLHttpRequest se encuentra descrito por el World Wide Web Consortium y sirve como una interfaz con la que se realizan peticiones a servidores Web.

Atributos del objeto XMLHttpRequest

Atributo	Descripción
readyState	Devuelve el estado del objeto como sigue: 0 = sin inicializar, 1 = abierto, 2 = cabeceras recibidas, 3 = cargando y 4 = completado.
responseBody	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 DOM.
status	Devuelve el estado como un número (p. ej. 404 para "Not Found").
statusText	Devuelve el estado como una cadena (p. ej. "Not Found").

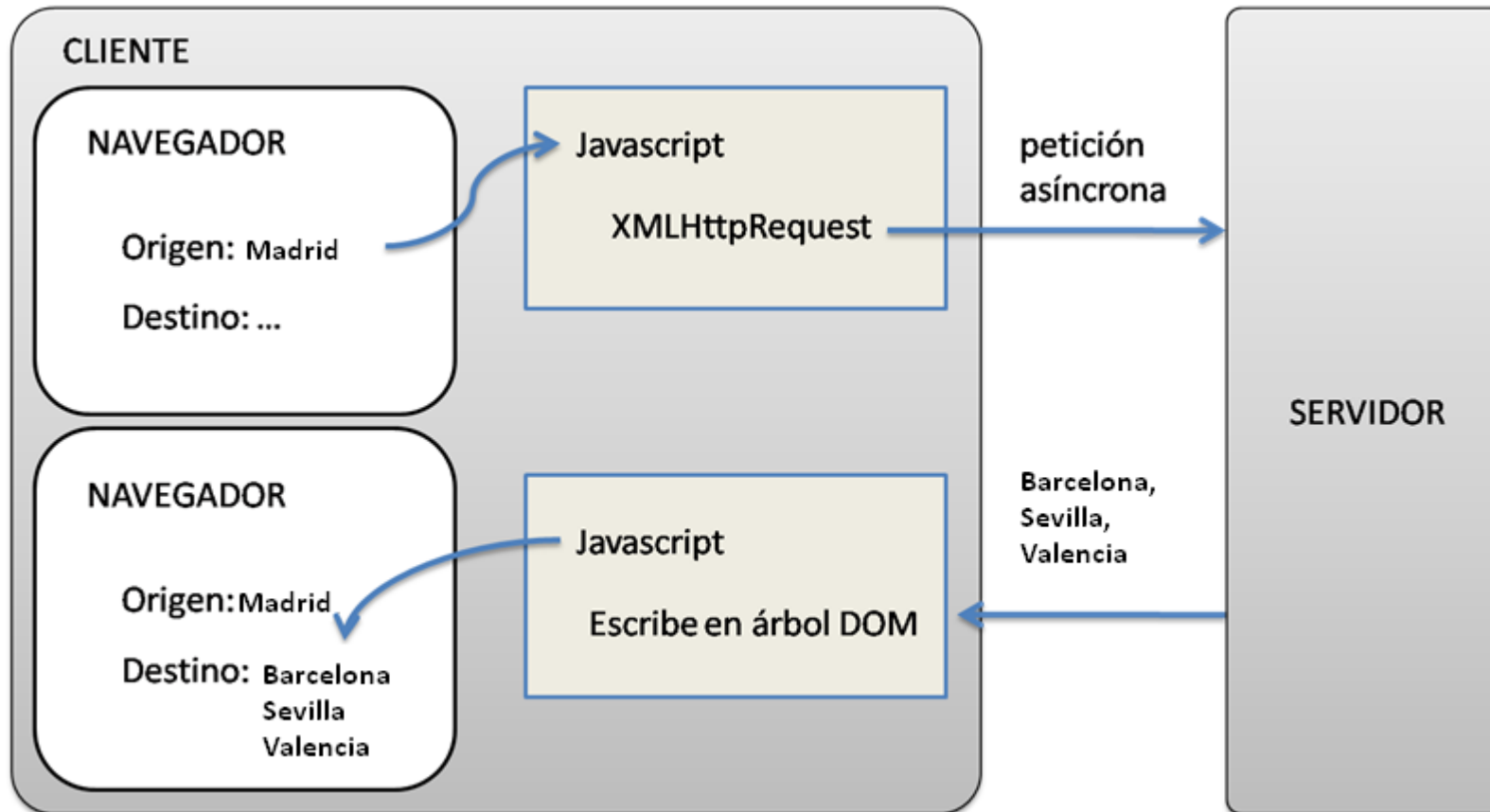
Métodos del objeto XMLHttpRequest

Métodos	Descripción
<code>abort()</code>	Cancela la petición en curso
<code>getAllResponseHeaders()</code>	Devuelve el conjunto de cabeceras HTTP como una cadena.
<code>getResponseHeader(cabecera)</code>	Devuelve el valor de la cabecera HTTP especificada.
<code>open(método, URL [, asíncrono [, nombreUsuario [, clave]]])</code>	<p>Especifica el método, URL y otros atributos opcionales de una petición.</p> <p>El parámetro de método puede tomar los valores "GET", "POST", o "PUT".</p> <p>El parámetro URL puede ser una URL relativa o completa.</p> <p>El parámetro asíncrono especifica si la petición será gestionada asíncronamente o no.</p>
<code>send([datos])</code>	Envía la petición.

Propiedades del objeto XMLHttpRequest

Propiedades	Descripción
onreadystatechange	Evento que se dispara con cada cambio de estado.
onabort	Evento que se dispara al abortar la operación.
onload	Evento que se dispara al completar la carga.
onloadstart	Evento que se dispara al iniciar la carga.
onprogress	Evento que se dispara periódicamente con información de estado.

Perspectiva Global con AJAX



JSON

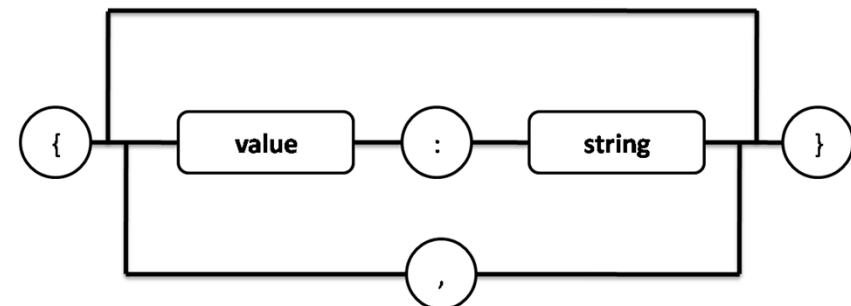
- JSON (JavaScript Object Notation).
- Formato ligero para el intercambio de datos.
- Es una manera de almacenar información.
- Fue pensado en un primer momento como una alternativa a XML.

JSON

- XML tienen una gran cantidad de información extra, asociada a su estructura.
- JSON está constituido por dos estructuras:
 - Una colección de pares de nombre/valor.
 - Una lista ordenada de valores.

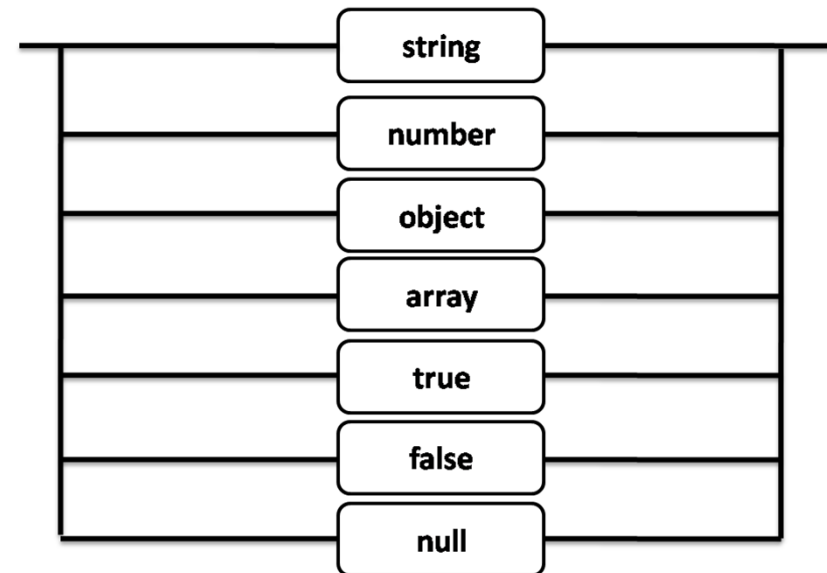
Object

- El elemento base de la sintaxis es el *object*.
- Está conformado por un conjunto desordenado de pares nombre/valor.
- Un objeto comienza con una llave de apertura y finaliza con una llave de cierre.
- Cada nombre es seguido por dos puntos, estando los pares nombre/valor separados por una coma.



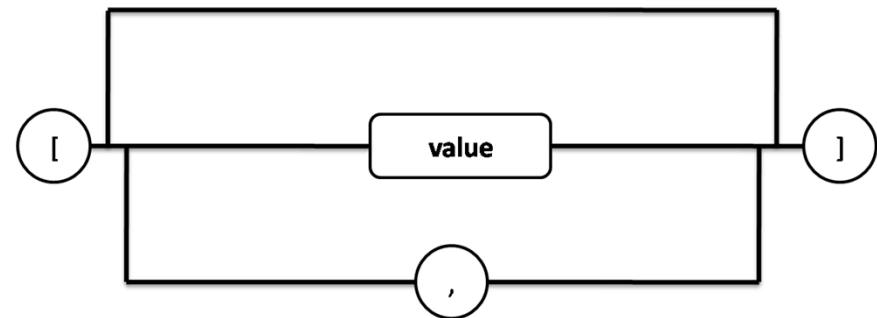
Array

- Un *array* es una colección de elementos *values*.
- Comienza por un corchete izquierdo y termina con un corchete derecho.
- Los elementos *value* se separan por una coma.



Value

- A su vez un elemento *value* puede ser una cadena de caracteres o *string* con comillas dobles, un *number*, los valores booleanos *true* o *false*, *null*, un *object* o un *array*.
- Estas estructuras pueden anidarse.



Ejemplo JSON y XML

<pre>{ 'nombre': 'pepe', 'edad': 34, 'domicilio': 'calle alcalá 1', 'estudios': ['primario', 'secundario', 'universitario'] }</pre>	<pre><ciudadano> <nombre>pepe</nombre> <edad>34</edad> <domicilio> calle alcalá 1 </domicilio> <estudios> <estudio>primario</estudio> <estudio>secundario</estudio> <estudio>universitario</estudio> </estudios> </ciudadano></pre>
---	---

Ejemplo de AJAX

- La página Web muestra un botón el cual, cuando hacemos click sobre él, muestra un mensaje en un elemento div cambiando el texto que se encontraba anteriormente.

```
<form>
  <input type = "button" value = "Buscar información"
  onclick = "obtenerDatosServidor('http://web/datos.txt',
  'elemento_destino')">
</form>
<div id="elemento_destino">
  <p>La información aparecerá aquí</p>
</div>
```

Ejemplo de AJAX

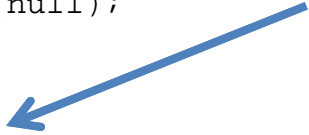
1. Función “obtenerDatosServidor” contiene dos parámetros.
2. Se elige el elemento HTML a ser modificado.
3. Se configura una conexión asíncrona con una URL.
4. Se indica la función a ser llamada una vez el estado del objeto cambie.
5. Se abre la conexión.

```
<script language = "javascript">

    var objetoXHR = new XMLHttpRequest();

    1 function obtenerDatosServidor(origen, elemento)
    {
    2     var objeto_destino = document.getElementById(elemento);
    3     objetoXHR.open("GET", origen);
    4     objetoXHR.onreadystatechange = respuesta();
    5     objetoXHR.send(null);
    }

    function respuesta(){
        if (objetoXHR.readyState == 4 &&
            objetoXHR.status == 200) {
            objeto_destino.innerHTML = objetoXHR.responseText;
        }
    }
</script>
```



Librerías de Actualización Dinámica

- Junto con la tecnología AJAX están las librerías que implementan una gran cantidad de funciones y controles a ser utilizados por los desarrolladores:
 - Independiente de la tecnología del servidor (por ejemplo PHP, JSP, ASP, etc.).
 - Manejar de manera transparente las incompatibilidades de los diferentes navegadores.
 - Manejar la comunicación asíncrona, sin necesidad de realizar la gestión de las operaciones de bajo nivel, como por ejemplo el manejo de estados y de tipos de errores.
 - Acceso sencillo al árbol DOM.
 - Información de errores para facilitar su utilización al desarrollador.
 - Proporcionar controles y objetos gráficos configurables, como por ejemplo: botones, calendarios, campos de texto.