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
`abort()`	Cancela la petición en curso
`getAllResponseHeaders()`	Devuelve el conjunto de cabeceras HTTP como una cadena.
`getResponseHeader(cabecera)`	Devuelve el valor de la cabecera HTTP especificada.
`open( método,URL[,asíncrono)`	Especifica el método, URL y otros atributos opcionales de una petición. El parámetro de **método** puede tomar los valores "GET“ o "POST". El parámetro **URL** puede ser una URL relativa o absoluta. El parámetro **asíncrono** especifica si la petición será gestionada asíncronamente o no. El valor por defecto es **true**

Métodos del objeto XMLHttpRequest

Métodos	Descripción
`setRequestHeader(cabecera,valor)`	Añade la cabecera HTTP para especificar el tipo de petición que se envía a través del método send().
`send([datos])`	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.

Conversión de un texto JSON a JavaScript

- Uno de los usos más comunes es obtener datos JSON desde un servidor web (como un archivo o como HttpRequest), convertirlos en un objeto JavaScript, y luego utilizarlos en una página web.
- Crear una cadena de JavaScript que contiene la sintaxis de JSON:

```
var txt = '{ "employees" : [' +
          '{ "firstName":"John" , "lastName":"Doe" },' +
          '{ "firstName":"Anna" , "lastName":"Smith" },' +
          '{ "firstName":"Peter" , "lastName":"Jones" } ]}';
```

- La función **JSON.parse(txt)** de JavaScript se puede utilizar para convertir un texto JSON en un objeto de JavaScript.

- Esta función utiliza el compilador de JavaScript que analiza el texto JSON y produce un objeto de JavaScript.

```
var obj = JSON.parse(txt);
```

Tipos de cabeceras – cliente (js)

Envío de parámetros al servidor (POST)

- Parámetros clave=valor

```
peticion_http.setRequestHeader("Content-Type",
                               "application/x-www-form-urlencoded");
```

- Parámetros en formato XML

```
peticion_http.setRequestHeader("Content-Type", "application/xml");
```

- Parámetros en formato JSON

```
http_request.setRequestHeader("Content-Type", "application/json");
```

Tipos de cabeceras – servidor (php)

- Resultados en formato XML

```
header('Content-Type: text/xml; charset=utf-8');
```

- Resultados en formato JSON

```
header('Content-type: application/json; charset=utf-8');
```

$.ajax(), $.post(), $.get

- A partir de la versión 3. 0 de jQuery han sido sustituidos algunos métodos:
  - Success → done
  - Error → fail
  - Complete → always