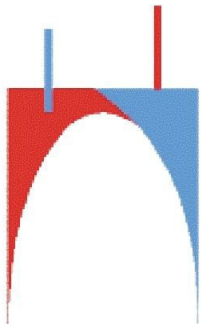


Instituto de Enseñanza Secundaria

Santiago Hernández

DESARROLLO WEB EN ENTORNO CLIENTE

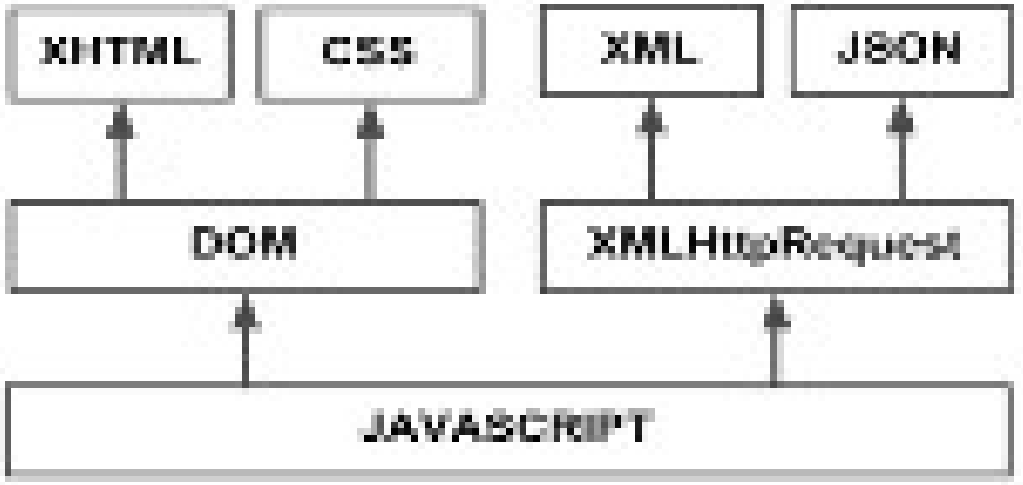
(Curso 2021 – 2022)

UD4. JAVASCRIPT ASÍNCRONO. AJAX.

AJAX

- Asynchronous Javascript And XML.

Intercambiar información entre el servidor y el cliente (un navegador web) de forma asíncrona. **Realiza cambios sobre una web sin necesidad de actualizarla por completo**.



Tecnologías

Extensible Hypertext Markup Language.

- XHTML y CSS: presentación de la página.
- DOM: manipulación dinámica de elementos de la página
- JSON y XML: formatos de intercambio de información.
- **Objeto XMLHttpRequest: intercambio asíncrono de información (sin recargar la página)**
- JavaScript: permite aplicar las anteriores tecnologías.

FUNCIONAMIENTO

Todo se basa en el objeto XMLHttpRequest

forma parte de AJAX

Pasos

- Creamos el objeto XMLHttpRequest. PASO1
- Preparamos la comunicación con el servidor. PASO 2
- Enviamos la petición (del cliente al servidor). PASO3
- Cuando llegue el resultado (respuesta) **lo procesamos mediante JavaScript** para modificar dinámicamente la página e incluir la nueva información.

Aplicación web clásica

Pasos

- El cliente hace una petición al servidor.
- El servidor recibe la petición.
- El servidor procesa la petición y **genera una nueva página con la petición procesada**.
- El cliente recibe la nueva página **completa** y la muestra.

AJAX

Ventajas

- Menor tráfico de internet
- Mayor dinamismo
- Aproxima el funcionamiento de una web a una aplicación de escritorio
- No es necesario actualizar al completo la página web.

Inconvenientes

- Diseño de aplicaciones web más complejo.
- Algunas funciones a las que estamos acostumbrados en la navegación web pueden no funcionar como esperamos. Por ejemplo, el botón de atrás, guardar marcador o actualizar la página web en cualquier momento.

AJAX

Implementación según el navegador

- Si el navegador implementa AJAX de forma nativa:
  - httpRequest = new XMLHttpRequest();
- Si no implementa AJAX de forma nativa pero permite crear objetos ActiveX:
  - httpRequest = new ActiveXObject("Microsoft.XMLHTTP");

httpRequest es el nombre de variable (puede ser cualquiera)

Observaciones

- El objeto XMLHttpRequest se soporta de forma estándar a partir de la versión 7 de Internet Explorer, antes se implementaba como un objeto ActiveX.
- Solo tiene un evento que se dispara cuando cambia su propiedad readyState y llama a la función de procesamiento.

https://cursosgs.es/dwec/2017/12/04/4-2-primeros-pasos/

OBJETO XMLHTTPREQUEST

Implementación

```
//Crear el objeto
if(window.XMLHttpRequest) {
    peticion_http = new XMLHttpRequest();
} else if(window.ActiveXObject) {
    peticion_http = new ActiveXObject("Microsoft.XMLHTTP");
}else{
    alert("Su navegador no soporta AJAX");
}
//Asignar la función de procesamiento
peticion_http.onreadystatechange=procesaRespuesta;
//preparar la petición
peticion_http.open("GET","noticias.txt",true);    //asíncrono

//enviar la petición
peticion_http.send(null); //no son necesarios parámetros
```

OBJETO XMLHTTPREQUEST

Método para preparar la petición

- open(metodo,url,async,usuario,password): Declara la petición. Parámetros:
  - método: GET o POST. Con "GET" los parámetros de la petición se codifican en la URL, con "POST" en las cabeceras de HTTP.
  - url: ruta de acceso al fichero (completa o relativa)
  - async: true (asíncrono) o false (síncrono). *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.
  - usuario: (opcional) usuario
  - psw: (opcional) password

OBJETO XMLHTTPREQUEST

Métodos para enviar la petición

- send(): para peticiones GET, envía la petición al servidor (sin parámetros). Envía la información a la URL especificada en open.
- send(string): para peticiones POST, envía la petición al servidor.
- setRequestHeader(): añade un par key/value a la cabecera de la petición a enviar.

Enviar parámetros

- Para enviar parámetros hemos de construir la querystring que es una cadena con parejas clave valor con el siguiente formato:
  - param1=valor1&param2=valor2&param3=valor3…
- No todos los caracteres se pueden enviar a través de http(ñ, espacio, & o /).
- encodeURIComponent(valor): codifica un texto (por ejemplo de un input type text)
- encodeURI(uri): para la URI completa ya que no codifica /, @ y otros caracteres propios de la URL.

OBJETO XMLHTTPREQUEST

Enviando

- Enviar por GET:
  - url = "http://servidor/applogin?" + querystring;
  - ObjetoXMLHttpRequest.open(“GET“, url , true);
  - ObjetoXMLHttpRequest.send(null);
- Enviar por POST:
  - ObjetoXMLHttpRequest.open(“POST“, url, true);
  - ObjetoXMLHttpRequest.setRequestHeader ("Content-Type", “application /x-www-form-urlencoded");
  - ObjetoXMLHttpRequest.send(querystring);
    - Ejemplo de querystring: ?usuario=yo&passwd=y0 (ejemplo, guardar una consulta en favoritos)

para peticiones POST se añade este par key/valor a la cabecera HTTP

OBJETO XMLHTTPREQUEST

Otros métodos

- abort(): Cancela la petición actual.
- getAllResponseHeaders(): Devuelve el conjunto de cabeceras HTTP como una cadena.
- getResponseHeader(nombreCabecera): Devuelve el valor de la cabecera HTTP especificada.

OBJETO XMLHTTPREQUEST

Función de procesamiento

```
function procesaRespuesta(){
        if(peticion_http.readyState== 4){        //datos listos
                if(peticion_http.status== 200){  //respuesta correcta
                        //aquí el código que procesa la respuesta
                        alert("Datos recibidos correctamente");
                }
        }
}
```

OBJETO XMLHTTPREQUEST

Atributos

- readyState: La función de procesamiento es llamada cada vez que cambia el valor de readyState. Valores:
  - 0 = No inicializado: el objeto está disponible para usar: o bien se ha creado pero no se ha invocado el método open o bien la última transacción ha finalizado.
  - 1 = Cargando: no se ha invocado el método send todavía.
  - 2 = Cargado: se ha invocado el método send , pero el servidor aún no ha respondido.
  - 3 = Interactivo: se han recibido algunos datos pero no todos. Estos datos se pueden recibir por bloques, por lo que el navegador puede invocar el método de procesamiento con este valor varias veces.
  - 4 = Completo: se han recibido todos los datos de la respuesta del servidor.

OBJETO XMLHTTPREQUEST

Atributos

- **status** y **statusText**: aunque la respuesta esté completa el servidor podría enviar un mensaje de error que se refleja en estos atributos. Valores:
  - status=200. statusTex=«OK»
  - status=403. statusText=«Forbidden»
  - status=404. statusText=«Not Found».
- responseText: devuelve la respuesta como una cadena.
- responseXML: devuelve la respuesta como un *XML Data*.

RESPUESTA

Formatos de respuesta

- Xml : encontraremos los datos almancenados en responseXML que es un objeto de tipo document.
- texto simple: usaremos la variable responseText.
- Json : es de tipo texto pero se codifica de una forma en concreto que es fácil de interpretar por Javascript.

- Servidor y cliente deben acordar previamente el tipo de respuesta que se va a utilizar.

RESPUESTA TEXTO

Ejemplo

- Noticias.txt tiene la siguiente forma
  - Noticia 1:este es el cuerpo de la noticia1
  - Noticia 2:este es el cuerpo de la noticia2
- código que procesa la respuesta
  - var noticias= ObjetoXMLHttpRequest.responseText.split("\n");
  - for (i=0; noticias.length;i++){

```
var estanoticia =noticias[i].split(":");
alert ("Noticia " + i + "Titulo: " + estanoticia [0]);
alert ("Noticia " + i + "Cuerpo: " + estanoticia [1]);
}
```

Para indicar que cada linea las separe con un salto de linea (\n)

RESPUESTA XML

Ejemplo

Noticias.xml

- ~~Noticias.txt~~ tiene la siguiente forma

(espacio)

```
<?xmlversion="1.0" encoding="ISO-8859-1"?>
<noticias>
    <noticia>
        <titulo>Noticia 1</titulo>
        <cuerpo>este es el cuerpo de la noticia1</cuerpo>
    </noticia>
    <noticia>
        <titulo>Noticia 2</titulo>
        <cuerpo>este es el cuerpo de la noticia2</cuerpo>
    </noticia>
</noticias>
```

RESPUESTA XML

Ejemplo

- código que procesa la respuesta
  - var noticias= ObjetoXMLHttpRequest.responseObjetoXMLHttpRequest.responseXML.getElementsByTagName("noticia");
  - alert(noticias[0].childNodes[1].childNodes[0].nodeValue); // el título de la primera noticia
  - alert(noticias[0].childNodes[3].childNodes[0].nodeValue); // el cuerpo de la primera noticia