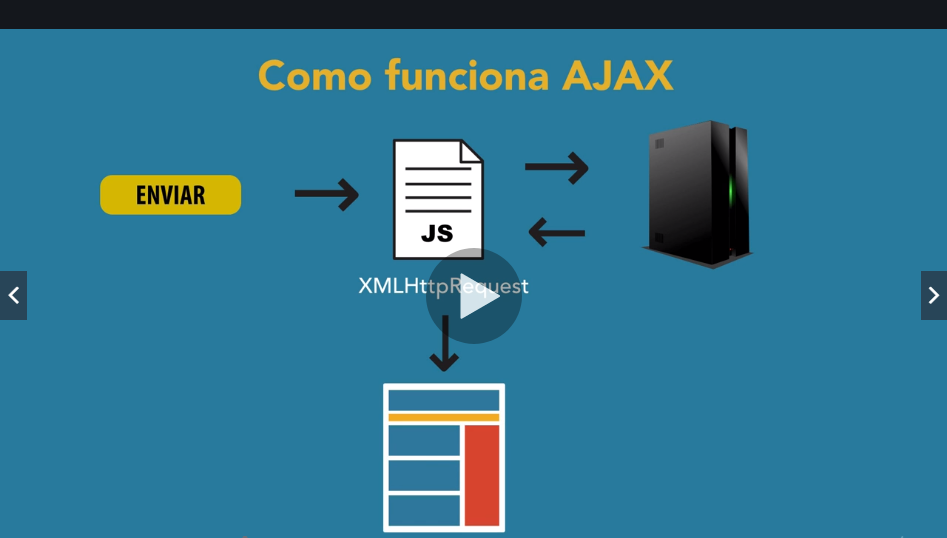
**Ajax**

Ajax es un conjunto de tecnologías que nos permiten realizar consultas al servidor en el background de la página, sin la necesidad de que esta tenga que actualizarse. Se maneja con archivos xml, jason, etc.



Como vemos en la imagen, en un ejemplo de enviar un formulario, el contacto con el servidor es intervenido por el archivo xml, que el cual recibe y envía información. De esta forma no se necesita intervenir la página y de esta forma actualizarse.

XMLHttpRequest

Este es el objeto con el que vamos a intercambiar datos detrás de escena. Se suele crean en el lenguaje JavaScript con el código:

**Var xhr = new XMLHttpRequest();**

**Métodos de XHR**

**new XMLHttpRequest();:** Crea un nuevo request.

**new FormData();:** Crea un nuevo request, pero este esta direccionado solo a formularios.

**.abort():** Cancela el request.

**.open():** Se abre la conexión y le pasas tus parámetros.

**.send():** Envia el request al servidor.

**.getAllResponseHeaders():** Retorna información del header.

**.setRequestHeader:** Agrega valores al header que se desea enviar.

Ejemplo de envio Ajax en un formulario:

**Var xhr = new XMLHttpRequest();**

**xhr.open(“POST** o **GET”, “archivo.php”, “true”);**

**xhr.setRequestHeader(“Content-type”, “application/x-www-form-urlencoded”)**

**xhr.send(“**parametros del form**”);**

**\***Dado que PHP no posee objetos de por sí, sino arreglos asociativos, pero en cambio en javascript estos si están, lo que se hace es utilizar un lenguaje de transporte como lo es **JSON**. Esto nos permite, en un ejemplo, recibir datos de un formulario en PHP por POST, pasarlos a **JSON** y luego leerlos con Ajax en Javascript.

**Post o Get**

**POST** es utilizado para enviar o intercambiar información.

**GET** es utilizado para recibir información.

**Propiedades del XHR**

**readyState:** Estado del xhr request, nos regresa un valor numérico (0- request sin iniciar, 1 – conexión establecida, 2 - request recibida, 3 – procesando el request, 4 – request finalizado y respuesta lista).

**responseText:** Retorna la respuesta com un string.

**responseXML:** Retorna la respuesta como un xml.

**status:** en los response aparece y define el estado del request. Sus números suelen ser (200 – Correcto, 403 – prohibido, 404 – no ecnontrado)

**onreadyStateChange:** Define una función que deberá ser llamada cuando cambien el estado del request, es decir cuando **readyState** cambie. Ejemplo:

**xhr.onreadyStateChange = function() {**

**if(xhr.readyState = 4 && xhr.status == 200) {**

**var divUsuario = document.getElementById(“registro”);**

**divUsuario.innerHTML = xhr.responseText;**

**}**

**}**

Se realiza un cambio en el html si se realiza la request de forma correcta.

**Ejemplos**

Archivo HTML de los dos:

<!DOCTYPE html>

<html>

    <head>

        <meta charset="utf-8" />

        <title>Ajax</title>

        <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/normalize/5.0.0/normalize.min.css" />

        <link rel="stylesheet" href="css/estilos.css" />

    </head>

    <body>

        <div class="contenedor">

            <h1>Aprendiendo AJAX</h1>

            <button id="cargar" value="cargar">Cargar</button>

            <div id="contenido"></div>

        </div>

        <!-- <script src="js/05.js"></script> -->

        <script src="js/06.js"></script>

    </body>

</html>

\*05.js es el text y 06.js es el php. Se comentara cada uno dependiendo el que usamos

**Ejemplo Con archivo Text**

Archivo Javascript:

// pasos para la creación de un request

var btnCargar = document.getElementById("cargar");

function cargarContenido() {

    // crearlo

    var xhr = new XMLHttpRequest();

    // abrirlo

    xhr.open("GET", "texto\_prueba.txt", true);

    // revisar que cambie

    xhr.onreadystatechange = function () {

        //console.log(xhr.readyState);

        if (xhr.readyState == 4 && xhr.status == 200) {

            //console.log("Se cargo correctamente");

            var contenido = document.getElementById("contenido");

            contenido.innerHTML = xhr.responseText;

        }

    };

    xhr.send();

}

btnCargar.addEventListener("click", cargarContenido);

* El archivo text solo tiene simple texto.

**Ejemplo Con archivo PHP**

Archivo PHP:

<?php

    $tecnologias = array(

        "frontend" => "html5, css3, javascript",

        "backend" => "php, mysql, apache",

        "fullstack" => "html5, css3, javascript, php, mysql, apache"

    );

    echo json\_encode($tecnologias);

?>

Archivo Javascript:

// pasos para la creación de un request

var btnCargar = document.getElementById("cargar");

function cargarContenido() {

    // crearlo

    var xhr = new XMLHttpRequest();

    // abrirlo

    xhr.open("GET", "servidor.php", true);

    // revisar que cambie

    xhr.onreadystatechange = function () {

        //console.log(xhr.readyState);

        if (xhr.readyState == 4 && xhr.status == 200) {

            //console.log("Se cargo correctamente");

            var json = JSON.parse(xhr.responseText);

            var contenido = document.getElementById("contenido");

            contenido.innerHTML = json.fullstack;

        }

    };

    xhr.send();

}

btnCargar.addEventListener("click", cargarContenido);

**Ejemplo: Enviar los datos de un Formulario a una base de datos**

**O editarlos**

Archivo Formulario (HTML o PHP):

<div class="campos">

    <div class="campo">

        <label for="nombre">Nombre:</label>

        <input type="text" placeholder="Nombre Contacto" id="nombre">

    </div>

    <div class="campo">

        <label for="empresa">Empresa:</label>

        <input type="text" placeholder="Nombre Empresa" id="empresa">

    </div>

    <div class="campo">

        <label for="nombre">Teléfono:</label>

        <input type="tel" placeholder="Teléfono Contacto" id="telefono">

    </div>

</div>

<div class="campo enviar">

    <input type="hidden" id="accion" value="crear">

    <input type="submit" value="Añadir">

</div>

Archivo Javascript:

const formularioContactos = document.querySelector("#contacto");

eventListeners();

function eventListeners() {

    // Cuando el formulario de crear o editar se ejecuta

    formularioContactos.addEventListener("submit", leerFormulario);

}

function leerFormulario(e) {

    e.preventDefault();

    // Leer los datos de los inputs

    const nombre = document.querySelector("#nombre").value,

        empresa = document.querySelector("#empresa").value,

        telefono = document.querySelector("#telefono").value,

        accion = document.querySelector("#accion").value;

    if (nombre === "" || empresa === "" || telefono === "") {

        mostrarNotificacion("Todos los campos son obligatorios", "error");

        mostrarNotificacion("Contacto creado Correctamente", "exito");

    } else {

        //Pasa la validacion, crear llamado a Ajax

        const infoContacto = new FormData();

        infoContacto.append("nombre", nombre);

        infoContacto.append("empresa", empresa);

        infoContacto.append("empresa", empresa);

        infoContacto.append("accion", accion);

        if (accion === "crear") {

            //craremos un nuevo elemento

            insertarBD(infoContacto);

        } else {

            //editar el contacto

        }

    }

}

// Insertar los datos en la base de datos via Ajax

function insertarBD(datos) {

    //llamado a Ajax

    //Crear objeto

    var xhr = new XMLHttpRequest();

    //Abrir la conexion

    xhr.open("POST", "inc/modelos/modelo-contactos.php", true);

    //Pasar los datos

    xhr.onload = function () {

        if (this.status === 200) {

            const respuesa = JSON.parse(xhr.responseText);

        }

    };

    //Enviar los datos

    xhr.send(datos);

}

* Se utiliza un FormData antes de llamar a Ajax dado de que este es mas común y mejor aplicable cuando se trata de un formulario.
* Dado a que el FormData es un objeto, debemos colocar los valores del formulario en el con una llave y un valor. Siendo el valor el **.value** de cada input.
* Para acceder a los valores dentro del objeto en un **console.log()** debemos hacer: **console.log(…varN)**

Archivo PHP JSON encode

<?php

 echo json\_encode($\_POST)

?>

Archivo PHP Conexión Base de Datos

<?php

//Credenciales de las Bases de Datos

define("DB\_USUARIO", "root");

define("DB\_PASSWORD", "root");

define("DB\_HOST", "localhost");

define("DB\_HOST", "agendaphp");

$conn = new mysqli(DB\_HOST, DB\_USUARIO, DB\_PASSWORD, DB\_NOMBRE);

?>

Explicación Resumida:

Lo que estamos haciendo es primero en el HTML recibir los datos del usuario, luego javascript los capta y los guarda en su memoria. En tercer lugar un archivo PHP recibe los datos del form como POST y los hace **encode** a JSON.