Asignatura	Datos del alumno	Fecha
Desarrollo de Aplicaciones en Red	Apellidos:	
	Nombre:	

Laboratorio. Tecnologías JavaScript y AJAX

Objetivos

En esta actividad se pretende poner en práctica los conocimientos adquiridos de las tecnologías JS y AJAX.

La actividad se subió a un repositorio de github, para luego ser desplegada con netlify Repositorio de github: https://github.com/andreidem18/actividad-desarrollo-en-red
Proyecto desplegado en netlify: https://voluble-caramel-754055.netlify.app/

JavaScript

1. Detectar si la cadena de entrada en un palíndromo.

Se creó un botón al cuál, al hacerle clic, ejecuta la función isPalindrome, la cuál recibe el valor del input de id palindrome-input por parámetros, y a través del método reverse() de javascript, invierte la cadena, de tal forma que se puede comparar si la cadena invertida es igual a la cadena sin invertir. Si ese es el caso, ejecuta un alert con el texto "La cadena es un palíndromo", de lo contrario, el alert dirá "La cadena NO es un palíndromo"



Asignatura	Datos del alumno	Fecha
Desarrollo de	Apellidos:	
Aplicaciones en Red	Nombre:	

2. Escribe un programa que pida dos números y escriba en la pantalla cual es el mayor.

Se creó un botón el cuál, al hacer clic, ejecuta la función isGreatherThan, la cuál recibe por parámetros los valores de los inputs greather-than-1-input y greather-than-2-input, los convierte a tipo de dato número con el operador "+", (ya que estos por defecto vienen como texto y no permitiría hacer bien la comparación) y luego compara si el primero es mayor que el segundo.



3. Escribe un programa que pida una frase y escriba las vocales que aparecen.

Se creó un botón el cuál, al hacer clic, ejecuta la función displayVowels, la cuál recibe por parámetros el contenido del input de id vowels-input. Luego, crea una estructura llamada Set, donde almacenaremos las vocales que se usan. Se optó por el Set ya que es una estructura que no permite duplicados, si una vocal se repite más de una vez, sólo se almacenara la primera vez que aparezca. Luego se itera la cadena letra por letra. En caso de que exista una vocal, la añade en el Set. Finalmente se ejecuta un alert, en el cuál se convierte el Set a un arreglo normal, para luego convertirlo a un string y así formatearlo y mostrarlo en el alert.

Asignatura	Datos del alumno	Fecha
Desarrollo de	Apellidos:	
Aplicaciones en Red	Nombre:	



4. Escribe un programa que pida una frase y escriba cuántas veces aparecen cada una de las vocales.

Se creó un botón el cuál, al hacer clic, ejecuta la función displayRepeatedVowels, la cuál recibe por parámetros el contenido del input de id vowels-repeated-input. Esta función tiene un comportamiento muy similar a la del ejercicio anterior. La diferencia es que en este caso no se optó por la estructura Set, sino por un objeto literal para almacenar las vocales. El objeto literal es una estructura que permite almacenar clave — valor. En este caso, en la clave se almacenarán las vocales, y en el valor se almacenará un número que representa las veces que se repiten dichas vocales. Por lo que, primero se crea el objeto con todas las vocales como keys, y como valores se establecieron todos en cero. Luego se iteró la cadena letra por letra, y cada vez que aparecía una vocal, en vez de añadirla al arreglo, se incrementaba su valor. Finalmente se formateó el objeto para poder mostrarlo en el alert



AJAX. A partir de la página web proporcionada, se pide añadir el código necesario para que:

Asignatura	Datos del alumno	Fecha
Desarrollo de	Apellidos:	
Aplicaciones en Red	Nombre:	

 Al cargar la página, el cuadro de texto debe mostrar por defecto la URL de la propia página.

Se creó un script al final, ya cuándo la página esta cargada. Allí se buscó el input de id url-page, y se le puso como valor la url de la página, la cuál se accede con window.location.href.

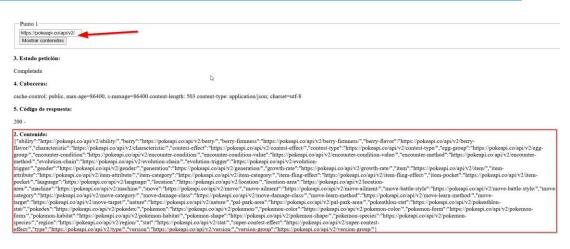
Ejercicio AJAX

Punto 1
https://voluble-caramel-754
Mostrar contenidos

Al pulsar el botón Mostrar Contenidos, se debe descargar mediante peticiones
 AJAX el contenido correspondiente a la URL introducida por el usuario. El
 contenido de la respuesta recibida del servidor en se debe mostrar en la zona
 de Contenidos del archivo.

Se creó un botón, que al darle clic, ejecuta la función getURL(), la cuál toma la url del input de id url-page, y hace una petición AJAX con la misma. Para esto, crea una instancia de XMLHttpRequest, y se usan los métodos "open()" y "send()" para ejecutar la petición. Para mostrar la respuesta, se usa el evento de AJAX llamado onreadystatechange. Allí se valida que la propiedad readyState sea 4, lo cuál quiere decir que la petición ha finalizado. Si ese es el caso, busca el input de id "contents" y le añade la respuesta a través del método innerHTTP.

Asignatura	Datos del alumno	Fecha
Desarrollo de	Apellidos:	
Aplicaciones en Red	Nombre:	



Aclaración:

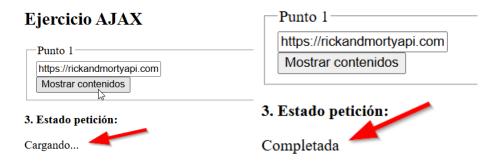
Es importante colocar una url de un servidor de acceso público para poder obtener una respuesta. En este caso, se colocó la url de una API Rest pública llamada "Pokeapi" https://pokeapi.co/api/v2/. Otros ejemplos de API Rest públicas son https://rickandmortyapi.com/api/character, o https://isonplaceholder.typicode.com/todos/1. También se puede utilizar la misma página web del proyecto, al ser del mismo dominio no habrá problemas. Pero al tratar de colocar otras páginas que retornen HTTP como https://www.google.com/, o https://www.google.com/, o https://www.google.com/, o https://www.unir.net/, lo más probable es que retorne error. Ya que normalmente las páginas web modernas no permiten ese tipo de peticiones desde aplicaciones del lado del servidor, ya que puede dar lugar a ciertas vulnerabilidades de seguridad.

3. En la zona Estados de la petición se debe mostrar en todo momento el estado en el que se encuentra la petición (no iniciada, cargando, completada, etc.).

Para esto, se modificó el método onreadystatechange, y se validó los demás valores de readyState. Previamente sólo se validó el número 4, que significa que ya la petición ha finalizado. Sin embargo, también tenemos el número 0 (No iniciada), número 1 (En progreso), número 2 (cabeceras recibidas) y

Asignatura	Datos del alumno	Fecha
Desarrollo de	Apellidos:	
Aplicaciones en Red	Nombre:	

número 3 (Descargando datos). En base al número obtenido, se tomó el div de id "status", y se le agregó el texto correspondiente al valor de readyState.



4. Mostrar el contenido de todas las cabeceras de la respuesta del servidor en la zona Cabeceras HTTP de la respuesta del servidor.

Para este caso nuevamente se modificó el método onreadystatechange, y se obtuvieron las cabeceras con el método getAllResponseHeaders(). El resultado obtenido se colocó en el div de id 'headers'

Ejercicio AJAX



5. Mostrar el código y texto de la respuesta del servidor en la zona Código de estado.

Una vez más se modificó el método onreadystatechange, el código y texto de la respuesta se obtuvieron con las propiedades status y statusText respectivamente, y se colocaron en el div de id status-code

Asignatura	Datos del alumno	Fecha
Desarrollo de Aplicaciones en Red	Apellidos:	
	Nombre:	

Ejercicio AJAX

Punto 1		
runio 1		
http://127.0.0.1:5500/index.		
Mostrar contenidos		

3. Estado petición:

Completada

4. Cabeceras:

accept-ranges: bytes access-control-allow-credentials: true cache-control: public, max-age=0 connection: keep-alive content-length: 7001 content-type: text/html; charset=UTF-8 date: Sat, 16 Nov 2024 23:51:14 GMT etag: W/"1584-193374ee83e" keep-alive: timeout=5 last-modified: Sat, 16 Nov 2024 23:28:58 GMT vary: Origin

5. Código de respuesta:

200 - OK