

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 es 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”

Ejercicios JavaScript

Punto 1

Punto 2

voluble-caramel-754055.netlify.app says
La cadena NO es un palíndromo

OK

Ejercicios JavaScript

Punto 1

Punto 2

voluble-caramel-754055.netlify.app says
La cadena es un palíndromo

OK

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

Ejercicios JavaScript

Punto 1

Punto 2

voluble-caramel-754055.netlify.app says
el número 10 es mayor

OK

Ejercicios JavaScript

Punto 1

Punto 2

voluble-caramel-754055.netlify.app says
el número 123 es mayor

OK

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 Aplicaciones en Red	Apellidos:	
	Nombre:	

Ejercicios JavaScript

Punto 1

Punto 2

Punto 3

voluble-caramel-754055.netlify.app says

Las vocales en la frase son: u, i, e, a, o

OK

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

Ejercicios JavaScript

Punto 1

Punto 2

Punto 3

Punto 4

voluble-caramel-754055.netlify.app says

Las vocales en la frase son:

a: 1
e: 3
i: 0
o: 0
u: 0

OK

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

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

1. 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

2. 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 `innerHTML`.

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

Punto 1

<https://pokeapi.co/api/v2/> 

Mostrar contenidos

3. Estado petición:

Completada

4. Cabeceras:

cache-control: public, max-age=86400, s-maxage=86400 content-length: 503 content-type: application/json; charset=utf-8

5. Código de respuesta:

200 -

2. Contenido:

```
{
  "ability": "https://pokeapi.co/api/v2/ability/",
  "berry": "https://pokeapi.co/api/v2/berry/",
  "berry-firmness": "https://pokeapi.co/api/v2/berry-firmness/",
  "berry-flavor": "https://pokeapi.co/api/v2/berry-flavor/",
  "characteristic": "https://pokeapi.co/api/v2/characteristic/",
  "contest-effect": "https://pokeapi.co/api/v2/contest-effect/",
  "contest-type": "https://pokeapi.co/api/v2/contest-type/",
  "egg-group": "https://pokeapi.co/api/v2/egg-group/",
  "encounter-condition": "https://pokeapi.co/api/v2/encounter-condition/",
  "encounter-condition-value": "https://pokeapi.co/api/v2/encounter-condition-value/",
  "encounter-method": "https://pokeapi.co/api/v2/encounter-method/",
  "evolution-chain": "https://pokeapi.co/api/v2/evolution-chain/",
  "evolution-trigger": "https://pokeapi.co/api/v2/evolution-trigger/",
  "gender": "https://pokeapi.co/api/v2/gender/",
  "generation": "https://pokeapi.co/api/v2/generation/",
  "growth-rate": "https://pokeapi.co/api/v2/growth-rate/",
  "item": "https://pokeapi.co/api/v2/item/",
  "item-attribute": "https://pokeapi.co/api/v2/item-attribute/",
  "item-category": "https://pokeapi.co/api/v2/item-category/",
  "item-fling-effect": "https://pokeapi.co/api/v2/item-fling-effect/",
  "item-fling-effect": "https://pokeapi.co/api/v2/item-fling-effect/",
  "item-pocket": "https://pokeapi.co/api/v2/item-pocket/",
  "language": "https://pokeapi.co/api/v2/language/",
  "location": "https://pokeapi.co/api/v2/location/",
  "location-area": "https://pokeapi.co/api/v2/location-area/",
  "machine": "https://pokeapi.co/api/v2/machine/",
  "move": "https://pokeapi.co/api/v2/move/",
  "move-ailment": "https://pokeapi.co/api/v2/move-ailment/",
  "move-battle-style": "https://pokeapi.co/api/v2/move-battle-style/",
  "move-category": "https://pokeapi.co/api/v2/move-category/",
  "move-damage-class": "https://pokeapi.co/api/v2/move-damage-class/",
  "move-learn-method": "https://pokeapi.co/api/v2/move-learn-method/",
  "move-target": "https://pokeapi.co/api/v2/move-target/",
  "nature": "https://pokeapi.co/api/v2/nature/",
  "pal-park-area": "https://pokeapi.co/api/v2/pal-park-area/",
  "pokeathlon-stat": "https://pokeapi.co/api/v2/pokeathlon-stat/",
  "pokedex": "https://pokeapi.co/api/v2/pokedex/",
  "pokemon": "https://pokeapi.co/api/v2/pokemon/",
  "pokemon-color": "https://pokeapi.co/api/v2/pokemon-color/",
  "pokemon-form": "https://pokeapi.co/api/v2/pokemon-form/",
  "pokemon-habitat": "https://pokeapi.co/api/v2/pokemon-habitat/",
  "pokemon-shape": "https://pokeapi.co/api/v2/pokemon-shape/",
  "pokemon-species": "https://pokeapi.co/api/v2/pokemon-species/",
  "region": "https://pokeapi.co/api/v2/region/",
  "stat": "https://pokeapi.co/api/v2/stat/",
  "super-contest-effect": "https://pokeapi.co/api/v2/super-contest-effect/",
  "type": "https://pokeapi.co/api/v2/type/",
  "version": "https://pokeapi.co/api/v2/version/",
  "version-group": "https://pokeapi.co/api/v2/version-group/"
}
```

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/)” <https://pokeapi.co/api/v2/>. Otros ejemplos de API Rest públicas son <https://rickandmortyapi.com/api/character/>, o <https://jsonplaceholder.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.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 Aplicaciones en Red	Apellidos:	
	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.

Ejercicio AJAX

Punto 1

3. Estado petición:

Cargando...

Punto 1

3. Estado petición:

Completada

- 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

Punto 1

3. Estado petición:

Completada

4. Cabeceras:

cache-control: max-age=259200 content-length: 1209 content-type: application/json; charset=utf-8 expires: Tue, 19 Nov 2024 18:20:17 GMT

- 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 textStatus 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

<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/"1584-193374ee83c" keep-alive: timeout=5 last-modified: Sat, 16 Nov 2024 23:28:58 GMT vary: Origin

5. Código de respuesta:

200 - OK