JavaScript 2022

Práctica 3

Objetivos

- JSON, AJAX, DOM.

Ejercicio 1

Verifique qué imprimen estas sentencias.

```
  const elements = [22, 44, 78, -43, 89];

  const point1 = { x: 0.0, y: 0.0 };
  const point2 = { x:3.0, y: 0.0 };
  const point3 = { x:0.0, y: 3.0 };

  let triangle = {
    corners : [point1, point2, point3]
  };

  console.log(JSON.stringify(elements));
  console.log(JSON.stringify(point1));
  console.log(JSON.stringify(triangle));
```

Ejercicio 2

Dada la siguiente asignación

```
var jsonStr = '['
  + '{"name":"Alice","dob": "2001-03-04T00:00:00.000Z","h": 165,"w": 68},'
  + '{"name":"Robert","dob": "1997-01-31T00:00:00.000Z","h": 170,"w": 88},'
  + '{"name":"Charles","dob": "1978-10-15T00:00:00.000Z","h": 188,"w":
102},'
  + '{"name":"Lucía","dob": "1955-08-07T00:00:00.000Z","h": 155,"w": 61},'
  + '{"name":"Peter","dob": "1988-03-09T00:00:00.000Z","h": 165,"w": 99},'
  + '{"name":"Lucas","dob": "1910-12-04T00:00:00.000Z","h": 172,"w": 75}]';
```

y la función `overweightNames` que sigue

```
function overweightNames(people){
    return people
```

```
        .filter(p => (p.w / Math.pow(p.h / 100, 2)) > 25)
        .map(p => p.name)
        .reduce((n1, n2) => n1 + ", " + n2);
}
```

a) Utilice el objeto JSON para invocar `overweightNames` e imprimir en la consola el resultado que debería ser el que sigue.

```
Robert, Charles, Lucía, Peter, Lucas
```

b) Imprima en la consola el nombre de la próxima persona en cumplir años.

Ejercicio 3

a) Use la función predefinida Fetch para hacer una página HTML que muestre una imagen de temática felina obtenida de `https://aws.random.cat/meow` o de `https://api.thecatapi.com/v1/images/search`.

b) Haga que la imagen cambie automáticamente cada 2 segundos.

Pista:: Puede usar setTimeout.

Ejercicio 4

Implemente una página HTML que liste los nombres y las fechas de estreno de los episodios de la serie `Rick and Morty` usando la API y, por ejemplo, colocando la información como elementos `li` dentro de un `ul`.

Pista: puede usar createElement y appendChild.

Ejercicio 5

Agregue a la página del ejercicio 4 un elemento `input` y un botón. Al presionar el botón, el listado de episodios debe mostrar únicamente aquellos cuyo nombre contiene el texto ingresado en el `input` o un mensaje si ninguno coincide.

Pista: puede usar removeChild y el filtro que provee la API.

Ejercicio 6

Basándose en los ejercicios 4 y 5, incorpore un elemento `select` que permita elegir el número de página que se quiere visualizar. Al cambiar la selección, el listado debe mostrar únicamente los resultados de la página seleccionada.

La información necesaria para implementar la paginación se obtiene del nodo `info` y el parámetro `page` de la URL permite solicitar una página específica.

```
 "info" : {
    "count": 51,
    "pages": 3,
    "next": "https://rickandmortyapi.com/api/episode?page=3",
    "prev": "https://rickandmortyapi.com/api/episode?page=1"
  },
```

Ejercicio 7

Implemente **junto a los demás integrantes de su grupo** una página que obtenga información de la PokéApi, liste los Pokémon y al seleccionarse uno de ellos muestre cierta información detallada.

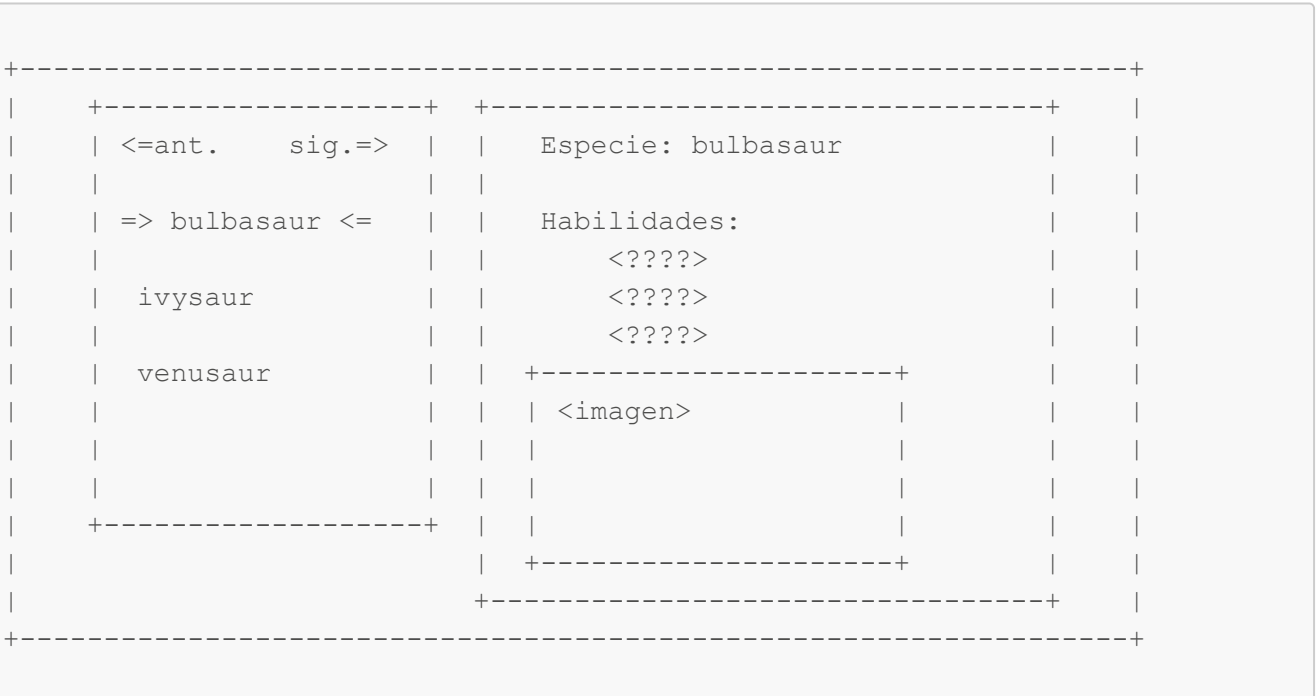
Tenga en cuenta que la lista de Pokémon es muy larga por lo que se debe presentar en forma paginada. Use los parámetros `offset` y `limit` de la api para ese fin.

La cantidad de elementos que se muestran por página no puede ser fija. Debe haber una constante definida que establezca esa cantidad y al modificarla y recargar la página se debe visualizar correctamente.

Para la información detallada puede elegir 4 o 5 características del Pokémon que considere importantes, **pero al menos debe mostrar una imagen del Pokémon**.

Recién al seleccionar un Pokémon de la lista, se debe obtener su información de la API.

La forma de visualización es a criterio del grupo. Por ejemplo, podría tener una distribución horizontal como la que sigue o una vertical.



Consideraciones sobre el ejercicio

- Se debe realizar en grupo.
- Se debe subir al repositorio git asignado al grupo en gitlab.

- Será evaluado.
- Cada integrante del grupo debe hacer sus *commits* con su aporte a la solución.