DigitalHouse > Coding School

## Desarrollo Web Full Stack Node

**Ejercitación - M08C21**

# AJAX y local storage

## Práctica AJAX

Vamos a practicar un poco sobre AJAX.
Les recordamos que para usar un pedido por AJAX la estructura de código sería la siguiente:

```
fetch(url)
.then(function(response) {
  return response.json()
})
.then(function(information) {
  console.log(information);
})
.catch(function(error) {
  console.log("Error: " + error);
})
```

Dependiendo de la **url** se consultara un endpoint diferente. Siempre también es **fundamental** observar el contenido de **data** para entender la información que obtenemos como resultado y así operarla.

**Cuidado al copiar este código ya que no siempre se copia bien desde un pdf...**

## Parte 1 - Creación de app en GIPHY

1) Ingresar en https://developers.giphy.com/
2) Clickear en el botón de “Create an App” y obtener una **Api Key**. La API de GIPHY es semi pública. Si bien, todo el mundo puede tener acceso, es necesario registrarse y eso es lo que acabamos de hacer. La **Api Key** debería verse así:


## Parte 2 - Conociendo la API de GIPHY

La API de GIPHY cuenta con un fantástico explorador que nos permite conocer más de su API y los endpoints que tiene.

Incluimos a continuación una serie de consignas simplemente para que observes y entiendas un poco mejor cómo funciona la API de GIPHY.

1) Ingresa en https://developers.giphy.com/explorer/
2) Elegí como endpoint “Random”. Observa la URL generada y cómo se incluye directamente sobre la URL tu **Api Key**.
3) Clickea en “Send Request”. Lo que vemos a continuación es el resultado que nos va a proveer GIPHY. Es fundamental entender el formato con el que responde GIPHY para luego poder operar con esta información. Algunos factores a tener un cuenta de esta respuesta:
    a) GIPHY retorna un **gran** objeto literal que cuenta con:
        i) Propiedad **data** donde tendrá la información relevante de resultado
        ii) Propiedad **meta** donde indica el resultado de la operación.

    Nótese que esta distribución es **muy común** en las APIs
    b) Dentro de la propiedad **data** aparece toda la información del GIF random. Algunas propiedades interesantes que utilizaremos del GIF:
        i) **title**: el nombre del GIF
        ii) **image_url**: URL donde se encuentra alojado el GIF

iii) **username**: Nombre de usuario que subió el GIF. Nótese que este dato puede estar vacío.
iv) **source**: Fuente original del GIF. Nótese que este dato puede estar vacío.
v) **id**: Este es un código que permite identificar al GIF de forma **única**. Si luego quisiéramos buscar específicamente ese GIF lo haríamos mediante el ID.

Entonces, si yo tuviese mi resultado del pedido AJAX en la variable **resultado**, para acceder la URL del GIF tendría que escribir ***resultado.data.image_url***

4) Ahora elegí como endpoint “Trending”. Observa la URL que se genera. Observa que se modifica en la URL si cambias el límite de resultados…
5) Clickea en “Send Request”. Lo que vemos a continuación es el resultado que nos va a proveer GIPHY. Es fundamental entender el formato con el que responde GIPHY para luego poder operar con esta información. Algunos factores a tener un cuenta de esta respuesta:
    a) De nuevo GIPHY retorna un **gran** objeto literal que cuenta con:
        i) Propiedad **data** donde tendrá la información relevante de resultado
        ii) Propiedad **meta** donde indica el resultado de la operación.
    b) Sin embargo, en este caso encontramos en **data** un array con muchos gifs resultados! Cada uno de los elementos del array si tiene **title**, **image_url** y todos los datos respectivos de un GIF
6) Ahora elegí como endpoint “Trending”. Observa la URL que se genera y cómo se modifica según los parámetros que escribamos
7) Clickea en “Send Request” y observa el resultado…¿Queda más claro ahora?

## Parte 3 - Conectándonos al primer endpoint

1) Para empezar vamos a crear un archivo **HTML** y un archivo **JS** vinculados. No olvides preparar el evento **onload** para no tener conflictos luego.
2) Mediante **fetch** vamos a conectarnos a la API de GIPHY a través del endpoint **random**. Te pedimos que imprimas en consola la información resultante para confirmar que la conexión funcione bien.
3) Dentro del bloque de código que manejara el resultado te pedimos que aisles en 2 variables separadas el nombre del GIF y la URL del mismo.
4) Te pedimos que insertes dentro del HTML una etiqueta **<h1>** que indique el título del GIF. Nótese que al recargar el sitio el título debe cambiar ya que estamos pidiendo un GIF random.
5) Te pedimos que insertes dentro del HTML una etiqueta **<img>** cuyo atributo **src** permita mostrar el GIF obtenido mediante la API.
6) Crear un botón que al clickearlo traiga un nuevo gif sin recargar la página.

Si llegaste a este punto...FELICITACIONES. Te acabas de conectar a tu primer API para mostrar gifs random. Yay!

## Parte 4 - Listando GIFS

1) Una vez más vamos a crear un archivo **HTML** y un archivo **JS** vinculados. No olvides preparar el evento **onload** para no tener conflictos luego.
2) Mediante **fetch** vamos a conectarnos a la API de GIPHY a través del endpoint **trending** con 25 resultados. Te pedimos que imprimas en consola la información resultante para confirmar que la conexión funcione bien.
3) Igual que el ejercicio anterior te pedimos que insertes dentro del cuerpo del HTML el título y la imagen del GIF, sin embargo ahora el desafío está en que debes imprimir 25 GIFS. Para esto, deberás recorrer el array resultado mediante un **FOR**
   A tener en cuenta, cuando recorremos el array de resultados, la estructura de la data es distinta al ejercicio anterior. A la hora de imprimir los GIFS debemos prestar atención a la estructura que devuelve nuestro **fetch**.

   Obtendremos un array de gifs, cada uno tendrá varios atributos, buscaremos el que contiene la imagen y dentro de ella, deberán buscar la URL para poder imprimirlas (pista: la URL dentro de “original” es la que nos sirve).

4) Te pedimos que al cargar el sitio, se le pida al usuario mediante **prompt** cuántos resultados desea ver. Modificar el código en los lugares correspondientes para mostrar la cantidad de resultados ingresada por el usuario.

## Parte 5 - Buscando GIFS

Vamos a crear un nuevo archivo buscar.html en donde incluiremos un formulario para buscar gifs. Crearemos también un archivo buscar.js para colocar nuestro código javascript.

Tené en cuenta que el formulario será por **GET** ya que estamos “buscando” datos y no modificando info de la base de datos. También tené en cuenta que ahora el endpoint sera: “search”. Tendrás que investigarlo brevemente en la documentación y deberás identificar en qué parte de la url se insertan los parámetros de búsqueda que viene del formulario.

El formulario tendrá 2 campos: uno para el término de búsqueda y otro para la cantidad de gifs que devolverá la búsqueda.

Los gifs del resultado de búsqueda deben imprimirse en la misma página en donde se encuentra el formulario. Sugerimos que lo presentes en forma de una lista con el título de cada gif y su imagen.

Antes de los resultados debe aparece la leyenda “Resultados de búsqueda para: *término buscado.*” dentro de una etiqueta **<h1></h1>**

## Práctica Web Storage

**PRÁCTICA INICIAL**
Seguiremos experimentando con giphy.

Realizaremos las siguientes consignas en el archivo index.html:

- **Agregar** por cada GIF un ícono con una estrellita. Esta estrellita permitirá **guardar** GIFs favoritos mediante un click.
- Tendrán que capturar el **evento** click sobre **cada** estrellita (ojo que puede haber muchas estrellitas!!!).
- Al hacer click en la estrellita debe almacenar en el **storage** el id del GIF seleccionado.
    - **a.** Nótese que puede haber MUCHOS gifs favoritos. Dado esto…¿Qué tipo de dato debemos almacenar en storage?

Luego añadiremos una página nueva llamada **favoritos.html**:

- Esta página debe listar los GIFS que han sido marcados como favoritos por el usuario.
- Si aún no tuviese ningún GIF esta página debe indicar que aún no tiene nada.
- Agregar en el encabezado de todas las páginas de GIPHY un botón a “Mis GIFS favoritos”

**PRÁCTICA EXTRA (Opcional)**

Te invitamos a pensar en casa cómo resolver la siguiente petición:

Ahora intentaremos tomar el control de nuestro sitio y aplicaremos restricciones.

Para esto agregaremos una página de **login.html** que:

- Tenga un espacio donde el usuario aclare su nombre de usuario.
- Al enviar el formulario debe almacenar esta información en **sessionStorage**
- Sólo cuando el usuario está logueado puede crear favoritos (si no esta logueado se lo redirige a login.html).
- Agregar una validación en el formulario para que el campo no esté vacío y tenga un mínimo de 3 caracteres.