

# JavaScript

## Pokedex

### Introducció

L'objectiu d'aquesta pràctica és arribar a construir una versió senzilla d'una Pokedex. A la pàgina oficial de Pokemon es pot veure un [exemple de Pokedex](#) , tot i que l'objectiu és construir una versió mínima d'això.

Una Pokedex és un llistat de Pokemons des d'on es pot consultar informació detallada de cadascun d'ells.

Bàsicament i, deixant de banda la temàtica de les dades, l'exercici tracta de consultar informació que ens arriba d'un recurs extern i un cop rebuda, maquetarla desde la nostra aplicació.

### Exercici

Crear una Pokedex amb HTML, CSS i JS.

S'han de tenir 2 pàgines, la pàgina inicial (index.html) i una pàgina de combat (combat.html) Cadascuna d'elles ha de contenir un menú de navegació que permeti anar d'una a l'altra.

### Pàgina inicial

#### **Consulta a la API de 10 Pokemon aleatoris**

Al carregar la pàgina inicial, s'ha de fer una consulta a la API de Pokemons (PokeAPI) i recuperar les dades de 10 Pokemons de forma aleatoria. Cadascun dels Pokemons s'ha de

mostrar maquetat com una targeta o carta i ha de contindre la següent informació: nom, atac, defensa i una imatge.

Aquesta informació es pot extreure de la API consultant l'ID del Pokemon. Per exemple, si es consulta el Pokemon amb ID = 1

<https://pokeapi.co/api/v2/pokemon/1/>

held_items:	[]
id:	1
is_default:	true
location_area_encounters:	"https://pokeapi.co/api/v2/pokemon/1/encounters"
▶ moves:	[...]
name:	"bulbasaur"
order:	1
past_types:	[]
▶ species:	{...}
▼ sprites:	
▶ back_default:	"https://raw.githubusercontent.com/PokeAPI/sprites/master/sprites/pokemon/back/1.png"
back_female:	null
▶ back_shiny:	"https://raw.githubusercontent.com/PokeAPI/sprites/master/sprites/pokemon/back/shiny/1.png"
back_shiny_female:	null
▼ front_default:	"https://raw.githubusercontent.com/PokeAPI/sprites/master/sprites/pokemon/1.png"
front_female:	null
▶ front_shiny:	"https://raw.githubusercontent.com/PokeAPI/sprites/master/sprites/pokemon/shiny/1.png"
front_shiny_female:	null
▶ other:	{...}
▶ versions:	{...}
▼ stats:	
▶ 0:	{...}
▼ 1:	
base_stat:	49
effort:	0
▼ stat:	
name:	"attack"
url:	"https://pokeapi.co/api/v2/stat/2/"
▼ 2:	
base_stat:	49
effort:	0
▼ stat:	
name:	"defense"
url:	"https://pokeapi.co/api/v2/stat/3/"
▶ 3:	{...}
▶ 4:	{...}
▶ 5:	{...}

Es pot obtenir la informació navegant per l'objecte fins a les rutes:

Nom = Objecte.name

Imatge = Objecte.sprites.front\_default

```
Atac = Objecte.stats[1].base_stat  
Defensa = Objecte.stats[2].base_stat
```

## **Pàgina amb informació detallada**

Les targetes de la pàgina principal també han de contenir un botó o un enllaç que permeti accedir a una pàgina amb informació més detallada del Pokemon. Aquesta informació es carregarà sobre la mateixa pàgina index.html, indicant a la mateixa url un paràmetre amb l'ID del Pokemon que es vol consultar. Per exemple: index.html?pokeID=5. Al carregar la pàgina s'ha de consultar si la url conté el paràmetre pokeID. Si el conté, només ha de mostrar la informació del Pokemon en qüestió i un botó per tornar enrere. En cas contrari, ha de mostrar el llistat inicial de Pokemons.

La informació d'aquesta pàgina ha de ser la mateixa que ja tenim a la targeta, més:

## **Filtre**

La pàgina inicial també ha de tenir un filtre de Pokemons. Ha de ser un input de text que, a mesura que es van escrivint lletres, només s'han de mostrar els Pokemon que en el seu nom continguin aquestes lletres. Aquesta comprovació s'ha de fer per cada lletra que s'escriu al filtre (event *input*)

Per exemple, si tenim les cartes: Charmander, Bulbasaur, Charizard i Blastoise, si s'escriu al filtre "cha", només han de ser visibles les cartes Charmandes i Charizard.

## **Pàgina combat**

La pàgina de combat de fer una consulta i retornar 10 Pokemons aleatoris. Aquests s'han de mostrar en format targeta com a l'apartat anterior, però inicialment la informació ha d'estar oculta. Les targetes han d'estar "girades" com si fóssin cartes boca avall i només es veu la part posterior de la targeta.

L'usuari haurà de seleccionar dos targetes fent clic. Les targetes seleccionades s'hauran de girar (efecte *flip card*) i mostrar la informació de la targeta: nom, imatge, atac i defensa.

Un cop seleccionades les cartes, aquestes entren en combat, seguint la següent lògica:

La primera carta seleccionada és la carta atacant. S'utilitzarà el seu valor "atac".  
La segona carta seleccionada és la carta que defensa. S'utilitzarà el seu valor "defensa".

Si el valor d'atac de la primera carta és superior al valor de defensa de la segona, la primera carta guanya el combat. S'ha de mostrar un missatge "{nom del Pokemon de la primera carta} ataca i guanya a {nom del Pokemon de la segona carta}".

Si, en canvi, el valor d'atac de la primera carta és igual o inferior al valor de defensa de la segona carta, guanya la segona carta. El missatge ha de ser: "{nom del Pokemon de la primera carta} ataca i perd contra {nom de Pokemon de la segona carta}".

A continuació s'ha de mostrar un botó ( o activar-lo si ja existeix i inicialment està desactivat) per tornar la pàgina al seu estat inicial i poder tornar a efectuar un combat.

Opcionalment es pot fer ús de *data-attributes* per emmagatzemar els valors d'atac, defensa i nom si resulta més còmode.

## A tenir en compte

No cal estar familiaritzat amb el món de Pokemon. Senzillament s'ha decidit utilitzar aquesta API perquè és molt complerta i gratuïta, i és molt utilitzada en exemples de programació de treball amb APIs.

## Objectius

El principal objectiu d'aquesta pràctica és treballar amb JavaScript i veure com es pot interactuar amb el DOM, creant i eliminant elements HTML i modificant i interactuant amb CSS.

També es treballarà amb una API externa (<https://pokeapi.co/>), realitzant operacions de consulta, i s'aprendrà a interpretar les dades rebudes en format JSON.

Treballar amb Figma per crear un wireframe des d'on partir a l'hora de maquetar l'aplicació.

## Recursos recomanats

Consultar exemples de Github del curs

Exemple pokedex

<https://www.pokemon.com/us/pokedex/>

PokeAPI

<https://pokeapi.co/>

Exemple CSS *Flip Card*

<https://codepen.io/Aleix/pen/ozWBoG>