# ¡Hazte con todos!

Este proyecto no tendrá mucho ficheros, vamos a intentar hacerlo de la manera más sencilla. por ello tendremos tres ficheros llamados **index.html**, **styles.css** y **pokedex.js.** Generamos una carpeta con el nombre `Pokédex` y dentro:

- **index.html**
- **styles.css**
- **pokedex.js**

## Fichero index.html

En este archivo generaremos un `div` con la clase `"container"` que contenga un `<h1>` con el texto Pokédex y una lista ordenada `<ol>` a la que le definiremos la `id` como `"pokedex"`. En esta lista pintaremos todos los Pokémon recuperados de la API.

No os podéis olvidar de invocar el archivo `.js` tal y como hemos visto en los ejemplos de las sesiones.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Pokedex</title>
    <link rel="stylesheet" href="styles/pokeapi.css" />
    <link href="https://fonts.googleapis.com/css?family=Rubik&display=swap" rel="stylesheet" />
</head>

<body>
    <div class="container">
        <h1>Pokedex</h1>
        <ol id="pokedex"></ol>
    </div>
    <script src="js/pokedex.js"></script>
</body>

</html>
```

## Fichero pokedex.js

En nuestro archivo JavaScript tendremos que seguir el siguiente flujo de funciones:

- Recuperar la lista con el id "podekex" y almacenarla en una variable.
- Ejecutar el fetch mediante una función recuperando los 150 primeros Pokemon a través de un bucle for e indicar el endpoint correcto de la API. En este caso los vamos a recuperar de la siguiente url: `https://pokeapi.co/api/v2/pokemon/`
- Hay que tener en cuenta que hay que especificarle a la url el valor que va a recuperar el bucle en cada iteración, ya que la información de cada Pokémon se almacenará en una url como estas:

  `https://pokeapi.co/api/v2/pokemon/1`

  `https://pokeapi.co/api/v2/pokemon/2`

  `https://pokeapi.co/api/v2/pokemon/3`
- Una vez recuperada la información tendremos que mapearla para imprimir los diferentes parámetros de los que compone. Para ello crearemos la constante `pokemon` dentro de la misma función en la que almacenaremos en diferentes valores la información recogida:

```
   const pokemon = results.map((result) => ({
              name: result.name,
              image: result.sprites['front_default'],
              type: result.types.map((type) => type.type.name).join(', '),
              id: result.id
```

En este caso hemos almacenado el nombre, la imagen, el tipo y el id (número). Si investigáis la API se pueden recuperar muchísima más información como los stats, los videojuegos en los que aparecen o diferentes generaciones de imágenes y displays.

- Una vez desglosada la información habrá que pintarla a través de otra función que nos recupere el resultado del `fetch` y nos pinte dentro de nuestro elemento `pokedex` una lista con dichos elementos. Esta función deberá ser ejecutada una vez termine la función del fetch (recordemos el flujo de funciones).
- Por último tenemos que llamar a la función fetch para que se ejecute al arrancar la aplicación y así nos recuperará la información y nos pintará nuestro listado.

Tened muy en cuenta la estructura del proyecto a la hora de llamar archivos para que todo funciones correctamente.

## Fichero styles.css

Aquí tenéis una guía de estilos para darle una mejor apariencia a la aplicación, pero recomendamos que le deis vuestro propio estilo para tener una Pokédex más personalizada.

```
body {
  background-color: orangered;
  margin: 0;
  font-family: rubik;
  color: white;
}
.container {
  padding: 40px;
  margin: 0 auto;
}
h1 {
```

```
  text-transform: uppercase;
  text-align: center;
  font-size: 54px;
}
.card {
  list-style: none;
  padding: 40px;
  background-color: #f4f4f4;
  color: #222;
  text-align: center;
}
.card-title {
  text-transform: uppercase;
  font-size: 32px;
  font-weight: normal;
  margin-bottom: 0;
}
.card-subtitle {
  font-weight: lighter;
  color: #666;
  margin-top: 5px;
}
.card-image {
  height: 180px;
}
#pokedex {
  padding-inline-start: 0;
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(320px, 1fr));
  grid-gap: 20px;
}
.card:hover {
  animation: bounce 0.5s linear;
}
@keyframes bounce {
  20% {
    transform: translateY(-6px);
  }
  40% {
    transform: translateY(0px);
  }
  60% {
    transform: translateY(-2px);
  }
  80% {
    transform: translateY(-0px);
  }
}
```