

## Trabalho 03

GIF do funcionamento: [https://github.com/jeffersoncarvalho/WEB\\_2020-1/blob/master/TRABALHOS/pokemon.gif](https://github.com/jeffersoncarvalho/WEB_2020-1/blob/master/TRABALHOS/pokemon.gif)

# Pokédex

## Parte 1 (Gonna catch'em all!)

O objetivo desta parte do trabalho, é acessar a **pokemon api**, listar os pokemons disponíveis em formato de paginação.

### 1 – O acesso a API

A **pokemon api** é um site que disponibiliza diversas informações sobre os mais variados Pokemons. Os dados são apresentados em formato JSON. Para acessar a **raiz** da API, basta usar o endereço:

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

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/"

Uma vez nesta página, note que é possível ter acesso a diversas informações sobre o universo Pokemon. No momento, o endereço que nos interessa é:

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

Nesta página, você terá acesso a todos os 964 (até o momento) pokemons da API. Note as seguintes propriedades:

- **count:** a quantidade de pokemons cadastrados até o momento;
- **next:** a próxima página com mais pokemons, de acordo com os parâmetro da URL offset e **limit** (no caso, fixado pra 20);
- **previous:** a página anterior de pokemons;
- **results:** uma lista (vetor) com uma quantidade fixa de pokemons. Cada elemento da lista é um objeto JSON formado por:

- **name:** o nome do pokemon
- **url:** mais informações sobre o pokemon.

Bem, agora que você entendeu mais ou menos o que é a API, eu sugiro ir clicando nos links e descobrindo por si próprio o que cada link mostra. Clique no next, previous e navegue pelos Pokemons.





## 2 – A primeira atividade: listar os pokemons.

A primeira tarefa é criar uma página chamada Pokedex.jsx onde você irá listar os pokemons disponíveis ao clicar em <https://pokeapi.co/api/v2/>. Minha sugestão é que você REUSE o mesmo código do projeto CRUD, visto em aulas anteriores. Veja como ficará a interface:

[Pokemon](#)
[Home](#)
[Pokédex](#)

### Pokédex

(964 Pokemons)

ID	Nome		
1	Bulbasaur		<a href="#">Informações</a> <a href="#">Capturar</a>
2	Ivysaur		<a href="#">Informações</a> <a href="#">Capturar</a>
3	Venusaur		<a href="#">Informações</a> <a href="#">Capturar</a>
4	Charmander		<a href="#">Informações</a> <a href="#">Capturar</a>

Sendo assim, você deve:

1. Criar uma tabela com quatro colunas (ID, Nome, Imagem e Ações).
2. Note que o **ID** de cada pokemon está no final de sua **URL**, dentro da propriedade **results**

```

count: 964
next: "https://pokeapi.co/api/v2/pokemon/?offset=20&limit=20"
previous: null
results:
  0:
    name: "bulbasaur"
    url: "https://pokeapi.co/api/v2/pokemon/1/"
  
```

3. O nome do pokemon deve estar **Capitalizado**, ou seja, a primeira letra deve estar em maiúsculo.
4. Você deve mostrar a imagem de frente do pokemon. Clicando na url de um pokemon, por exemplo, o Bulbasaur:

```

    order: 1
    species: {...}
    ▼ sprites:
      ▶ back_default: "https://raw.githubusercontent.com/ritesh/pokemon/back/1.png"
        back_female: null
      ▶ back_shiny: "https://raw.githubusercontent.com/pokemon/back/shiny/1.png"
        back_shiny_female: null
      ▶ front_default: "https://raw.githubusercontent.com/er/sprites/pokemon/1.png"
        front_female: null
      ▶ front_shiny: "https://raw.githubusercontent.com/ites/pokemon/shiny/1.png"
        front_shiny_female: null




```

Você irá notar a propriedade **sprites** e dentro dela a propriedade **front\_default**. É essa a imagem que eu quero que você mostre na tabela.

- Os botões **Informações** e **Capturar** ainda não terão NENHUMA funcionalidade. Apenas os coloque lá.

### 3 – A segunda atividade: paginação

Ao rolar a página do Pokedéx, no rodapé da tabela teremos o seguinte:

18	Pidgeot		<button>Informações</button> <button>Capturar</button>
19	Rattata		<button>Informações</button> <button>Capturar</button>
20	Raticate		<button>Informações</button> <button>Capturar</button>
		<button>Anterior</button> <button>Próximo</button>	

Os botões **Anterior** e **Próximo** navegam entre as páginas de pokemons, mostrando de 20 em 20 pokemons (ou mais ou menos, caso queira). Sendo assim:

- Implemente o botão Anterior fazendo uso da propriedade **previous**, ou seja, ao clicar em **Anterior**, a sua aplicação deve recarregar a página com os novos pokemons da página anterior.
- O mesmo raciocínio para o botão **Próximo**.
- Note que se não tem mais páginas anteriores e nem próximas, o botão deve ficar desabilitado.

## Parte 2

# (I choose you!)

O objetivo desta parte é mostrar informações sobre o Pokemon selecionado, em outra tela.

## 1 – Acessando mais detalhes de um Pokemon, através da api.

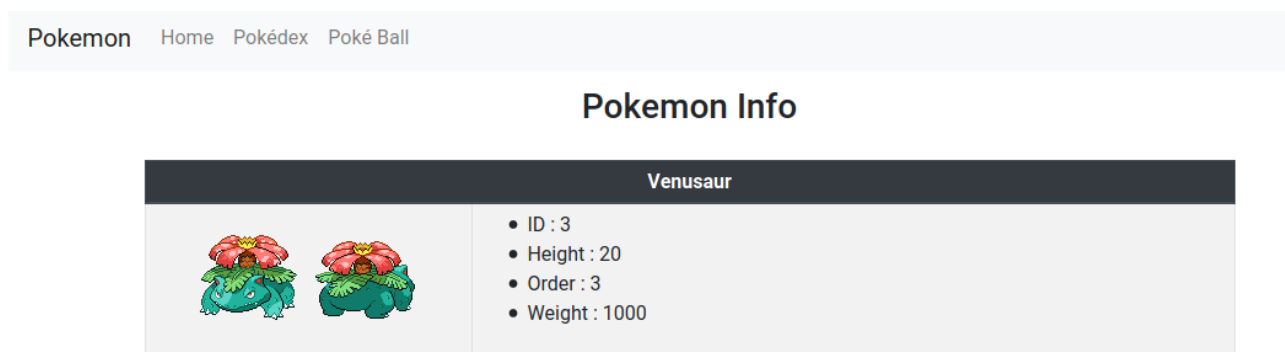
Cada Pokemon tem uma url associada. Se você clicar na url, verá o seguinte (para o Bulbasaur):

```
▶ abilities:      [...]
  base_experience: 64
▶ forms:          [...]
▶ game_indices:   [...]
  height:         7
  held_items:     []
  id:             1
  is_default:     true
  location_area_encounters: "https://pokeapi.co/api/v2/pokemon/1/encounters"
▶ moves:          [...]
  name:           "bulbasaur"
  order:         1
▶ species:        {...}
▶ sprites:        {...}
▶ stats:          [...]
▶ types:          [...]
  weight:         69
```

Note que são MUITAS informações para cada Pokemon. Caso você realmente esteja inspirado, poderia criar um jogo realmente GRANDE, com todas as informações, em especial, as de movimentos (propriedade **moves**).

## 2 – A terceira atividade: mostrar as informações de um Pokemon, em outra página.

Você deverá criar uma página chamada **PokemonInfo.jsx**, que exibirá as informações de um Pokemon selecionado na Pokédex. Veja:



As informações deverão ser exibidas em uma tabela (ou noutra forma que você achar melhor esticamente), da seguinte forma:

- Use um `axios.get` para acessar o objeto JSON que representa um Pokemon, assim como você fez para acessar a lista de Pokemons na Pokédex.
- Mostre as imagens de frente e costas do Pokemon (propriedade **images**)
- Mostre também: ID, Height, Order e Weight.

### 3 – A quarta atividade: mostrar movimentos (golpes) de um Pokemon.

Logo abaixo das informações principais, você também deve mostrar os movimentos de um Pokemon selecionado, veja:

Venusaur	
	<ul style="list-style-type: none"> <li>• ID : 3</li> <li>• Height : 20</li> <li>• Order : 3</li> <li>• Weight : 1000</li> </ul>
Movimentos	
swords-dance	<a href="https://pokeapi.co/api/v2/move/14/">https://pokeapi.co/api/v2/move/14/</a>
cut	<a href="https://pokeapi.co/api/v2/move/15/">https://pokeapi.co/api/v2/move/15/</a>
bind	<a href="https://pokeapi.co/api/v2/move/20/">https://pokeapi.co/api/v2/move/20/</a>
vine-whip	<a href="https://pokeapi.co/api/v2/move/22/">https://pokeapi.co/api/v2/move/22/</a>
headbutt	<a href="https://pokeapi.co/api/v2/move/29/">https://pokeapi.co/api/v2/move/29/</a>

Você deve mostrar o nome e o link para a página com MUITOS mais detalhes do movimento em questão. Não se preocupe que não iremos renderizar essas informações. Para acessar esses dados, você terá que usar a propriedade **moves** que é um vetor de objetos. Veja:

```

▼ moves:
  ▼ 0:
    ▼ move:
      name: "razor-wind"
      url: "https://pokeapi.co/api/v2/move/13/"
      ► version_group_details: [...]
      ► 1: {}

```

No caso do movimento **razor-wind**, a propriedade **move.name**, diz seu nome. E a propriedade **move.url**, a url com mais informações.


# Parte 3

## (POKEBALL, GO!)

O objetivo desta parte é capturar os Pokemons listados na Pokédex (máximo de 6).

### 1 – A quinta atividade: capturar um Pokemon.

Para capturar um Pokemon, você deve clicar no botão “Capturar”.

ID	Nome		
1	Bulbasaur		<button>Informações</button> <button>Capturar</button>

Ao capturar um Pokemon, armazene informações como ID e nome dentro de um objeto. Este objeto deverá ficar dentro de um Array (a pokebola ou **pokeball**), o qual será armazenado no SessionStorage. Veja um exemplo:

```
capturar(id,nome){  
  
    let pokeball = JSON.parse(sessionStorage.getItem('pokeball'))  
    if(!pokeball){  
        pokeball = []  
    }  
  
    if(pokeball.length===6){  
        alert('Capacidade máxima da Pokebola atingida.')  
        return  
    }  
  
    for (let index = 0; index < pokeball.length; index++) {  
        const element = pokeball[index];  
        if(id===element.id) {  
            alert('Pokemon já capturado! Escolha outro.')  
            return  
        }  
    }  
  
    pokeball.push({id:id,nome:nome})  
    sessionStorage.setItem('pokeball',JSON.stringify(pokeball))  
    alert('Pokemon capturado com sucesso!')  
}
```

Note que este código também testa se a pokebola (pokeball) já está cheia (6 Pokemons) ou se já tem o Pokemon que você quer capturar.

Veja mais sobre SessionStorage e LocalStorage em: <https://www.treinaweb.com.br/blog/quando-usar-sessionstorage-e-localstorage/>

O próximo passo é EXIBIR os Pokemons na Pokebola. Para isso, você irá criar uma nova página chamada **Pokeball.jsx**. Veja como ficará seu layout:




Pokemon

Home

Pokédex

Poké Ball

Poké Ball

ID	Nome	
1	Bulbasaur	
3	Venusaur	
6	Charizard	

Pokédex

Crie um link no Menu, para acessar a Pokebola. Mostre todos os Pokemons dentro da Pokebola e um botão para voltar para a Pokédex.

# Parte 4

## (A wild REACT Code appeared!)

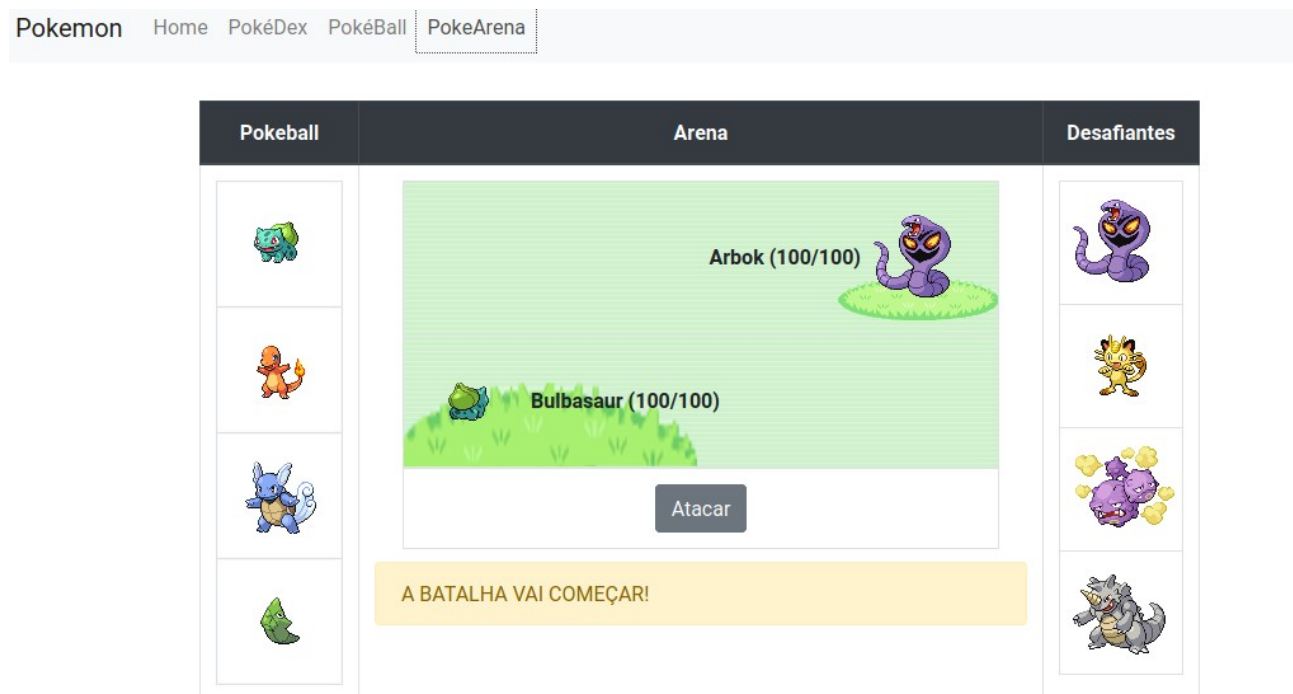
O objetivo desta parte criar uma arena simples para os Pokemons batalharem.

### 1 – A sexta atividade: criando a arena.

Crie o arquivo PokeArena.jsx. Em seu construtor, defina uma equipe inimiga. Por exemplo, a Equipe Rocket:

```
let equipeRocket = [  
  { id: 24, nome: 'arbok', life: 100 },  
  { id: 52, nome: 'meowth', life: 100 },  
  //{ id: 71, nome: 'victreebel', life: 100 },  
  //{ id: 108, nome: 'lickitung', life: 100 },  
  { id: 110, nome: 'weezing', life: 100 },  
  { id: 112, nome: 'rhydon', life: 100 }]
```

O layout deverá ficar mais ou menos assim:



Você poderá escolher os pokemons da sua Pokebola do lado esquerdo. Cada vez que ele ataca, o inimigo irá perder uma certa quantidade de vida. Ao desmaiar, passe para o próximo Pokemon inimigo.

Observação: essa parte do código é a mais **desafiadora**. Caso queiram pescar como eu fiz, além da imagem do fundo de batalha, basta acessar: [https://github.com/jeffersoncarvalho/WEB\\_2020-1/tree/master/IMPLEMENTACOES/aulaes/pokemon/src/components/pokearena](https://github.com/jeffersoncarvalho/WEB_2020-1/tree/master/IMPLEMENTACOES/aulaes/pokemon/src/components/pokearena)



**Observações:**

1. Este trabalho não deve ser entregue e também não vale nenhuma nota ou presença. Tente fazê-lo com o intuito de aprender e treinar.
2. Eventualmente eu irei disponibilizar a resolução como videoaula. Sugiro que você **TENTE** fazer antes de partir para a videoaula.
3. Este trabalho, pelo menos a Parte 01, não usará o MongoDB, apenas o AXIOS, para acessar a API dos pokemons.

© 2020 Pokémon. TM, ® Nintendo.