EP – 01
Extração de Pokémons

BCC - 6° Semestre, 2024
Lucas da Mata Guimarães

Problema

- Coletar os dados referentes a todos os Pokémons, até o ano de 2024;
- Tratar variações de quantidade de dados, assim como valores repetidos que estouram em mais subi formações;
- Retornar esses dados como um arquivo JSON.

Exemplo de dado


```
{
  "url_page":"https:\/\/pokemondb.net\/pokedex\/bulbasaur",
  "pokemon_name":"Bulbasaur",
  "peso":"6.9\u00a0kg ",
  "altura":"0.7\u00a0m ",
  "tipo 1":"Grass",
  "tipo 2":"Poison",
  "evolucao 1":"#0003-Venusaur-
    https:\/\/pokemondb.net\/pokedex\/venusaur",
  "evolucao 2":"#0002-Ivysaur-
    https:\/\/pokemondb.net\/pokedex\/ivysaur",
  "abilidade 1":"Chlorophyll",
  "abilidade 2":"Overgrow",
  "url":"https:\/\/pokemondb.net\/ability\/chlorophyll",
  "nome":"Chlorophyll",
  "descricao":"Boosts the Pok\u00e9mon's Speed in sunshine."
}
```


- Aqui temos os dados já coletados e tratados do Bulbasaur;
- Mostrando suass habilidade, evoluções e dados.

Seletores

- A maioria dos seletores CSS utilizados, vai diretamente no elemento HTML e retorna o dados;
- Apenas os seletores de abilidades e evoluções que operam de maneira diferente;

Seletor de Evoluções

- Primeiro achamos o elemento com da classe .infocard-list-evo, e passamos ele para uma função:

```
'evolutions': self.get_evoluion(response.css('.infocard-list-evo'))
```

- A função por sua vez acha todos os elementos dentro do card passado, e retorna uma string contendo todos os dados relevantes:

```
def get_evoluion(self, resposnse) -> str:
    resp = ""
    list_evolution = resposnse.css('div')
    for evolution in list_evolution:
        info = evolution.css('span.text-muted')
        num = info.css('small::text').get()
        nome = info.css('a::text').get()
        link = info.css('a::attr(href)').get()
        resp += f"{num}-{nome}-{self.domain}{link}|"
    return resp
```

Seletor de Evoluções

- Aqui achamos em qual parte do HTML estão as abilidades e passamos para uma função:

```
'abilities': self.get_abilities(response.css('.vitals-table > tbody > tr:nth-child(6)')),
```

- Dentro da função achamos todas as abilidades que o Pokemon pode ter e retornamos uma string com seus nomes:

```
def get_abilities(self, response) -> str:
    abiliities = response.css('td > .text-muted')
    resp = ""
    for abilitie in abiliities:
        name = abilitie.css('a::text').get()
        resp = resp + name + "|"
    return resp
```

Spider de Abilidades

- Temos um segundo scrper que coleta os dados relevantes da abilidades;
- Esses seram intregados ao Pokemon quando formos criar o nosso JSON.