POKEMON API

HTML

The given code is an HTML file that contains the structure and styling of a web page. It is used to display the list of Pokemon on a web page using the Poke API. Below is a detailed explanation of each section of the code.

HTML Structure:

The HTML file starts with a DOCTYPE declaration that specifies the version of HTML used in the document.

The HTML file has a head section that contains metadata information about the document such as character encoding, viewport size, and links to external files.

The body section contains the main content of the web page.

Metadata:

The character encoding for the HTML file is set to UTF-8.

The viewport is set to the width of the device and initial-scale of 1.0.

The title of the web page is set to "pokemon".

The external stylesheet "style.css" is linked to the HTML file.

The Google font "Oswald" is linked to the HTML file.

Pokémon

#001

Bulbasaur

Weight: 69 lbs

#002

Ivysaur

Weight: 130 lbs

#003

Venusaur

Weight: 1000 lbs

#004

Charmander

Weight: 85 lbs

#005

Charmeleon

Weight: 190 lbs

#006

Charizard

Weight: 905 lbs

The Bootstrap CSS and JavaScript files are linked to the HTML file to add styling and functionality.
Body Section:
The body section starts with an image of the Pokemon logo, with the "alt" attribute set to "Your Image".
A container div with the class name "pokemon-container" is created, which will contain the list of Pokemon.
A navigation bar is created with two list items, "previous" and "next".
The spinner div with the id "spinner" is created and set to display a loading animation.
A script tag with the source file "script.js" is included at the end of the body section to provide functionality to the web page.
JavaScript:
The script.js file contains the code that fetches the data from the PokeAPI, populates the "pokemon-container" div with the list of Pokemon, and adds functionality to the "previous" and "next" buttons.
The fetch API is used to get the list of Pokemon from the PokeAPI.
The data is converted to JSON format and processed to display the list of Pokemon in the "pokemon-container" div.
The "previous" and "next" buttons are assigned click event listeners that fetch the previous and next pages of Pokemon respectively, and update the content of the "pokemon-container" div.
Overall, this code is a basic template for displaying a list of Pokemon using the PokeAPI and Bootstrap styling.

CSS

This is a CSS code that styles a web page related to Pokemon . Here's a breakdown of what each part of the code does:

The first block of code sets the overall style for the page. The text is centered, and the font used is Arial, Helvetica, sans-serif, with a backup font of "Oswald", sans-serif. The background image is set to a URL.

```
body {
    text-align: center;
    font-family: Arial, Helvetica, sans-serif;
    font-family: "Oswald", sans-serif;
    background-image: url (https://wallpaperaccess.com/full/174950.png);
}
```

The following block of code styles the h1 tag, setting its text color to white.

```
h1 {
    color: white;
}
```

The .pokemon-container class sets up a grid layout with three columns, and a gap of 30 pixels. It also center it with a margin, and sets its width to 80% of the page.

```
.pokemon-container {
    display: grid;
    grid-template-columns: 1fr 1fr 1fr;
    gap: 30px;
    width: 80%;
    margin-left: auto;
    margin-right: auto;
    margin-bottom: 40px;
    margin-top: 20px;
}
```

The .pokemon-block and .pokemon-block-back classes apply to the Pokemon cards displayed on the page. They set the border radius to 10 pixels, add some padding and a white background color, and apply a box shadow.

```
.pokemon-block,
.pokemon-block-back {
    border-radius: 10px;
    padding: 10px;
    background-color: white;
    box-shadow: 0 3px 15px rgba(100, 100, 100, 0.5);
}
```

The .img-container class sets up the background of the Pokemon image containers to not repeat and be centered.

```
.img-container {
    background-repeat: no-repeat;
    background-position: center;
}
```

The .name class sets the Pokemon name text to be capitalized, bold, and with a font size of 1.2 rem.

```
.name {
    text-transform: capitalize;
    font-weight: bold;
    font-size: 1.2rem;
}
```

The #spinner id hides the loading spinner element from the page.

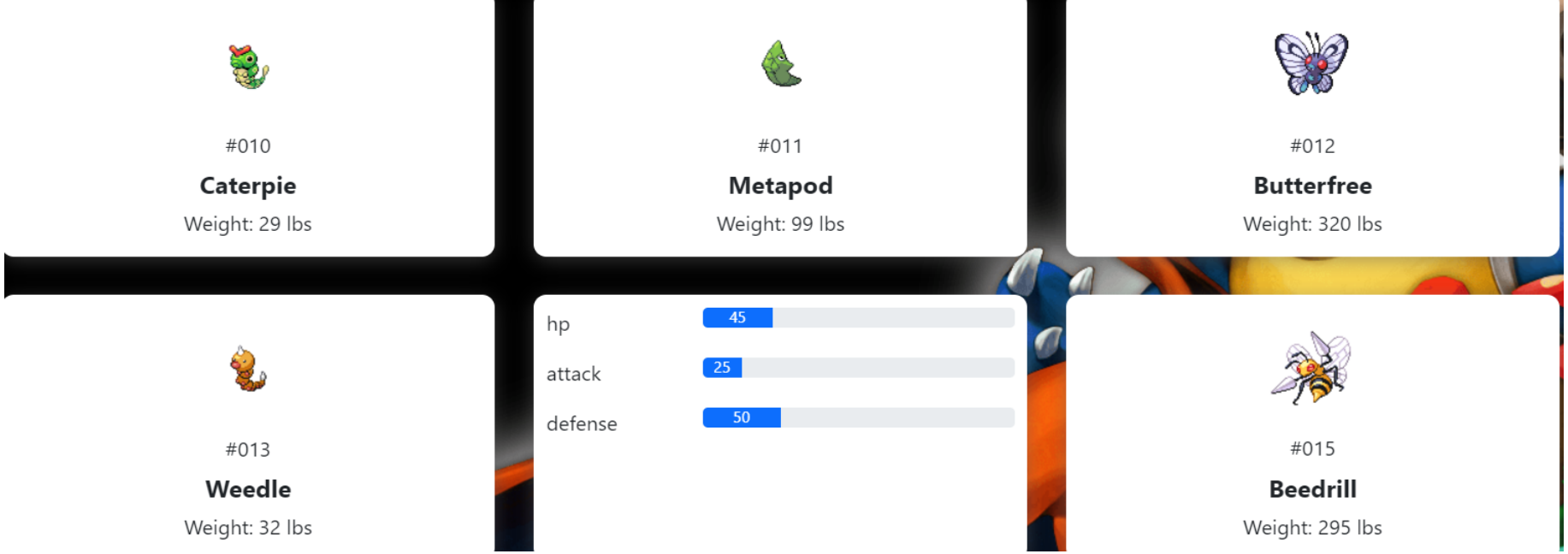
```
#spinner {
    display: none;
    position: absolute;
    top: 50%;
    left: 50%;
}
```

The .pagination class styles the navigation links at the bottom of the page, setting their width, centering them with margins, and adding some margin to the bottom.

```
.pagination {
    width: 90%;
    margin-left: auto;
    margin-right: auto;
    margin-bottom: 50px;
}
```

The .card-container class applies a transform animation when the Pokemon card is hovered over, flipping the card.

```
.card-container {
    position: relative;
    width: 100%;
    height: 100%;
    text-align: center;
    transition: transform 0.8s;
    transform-style: preserve-3d;
}
```




```
.flip-card:hover .card-container {
    transform: rotateY(180deg);
}
```

The .stat-container class sets up the display of the Pokemon's stats, with a grid layout of two columns.

```
.stat-container {
    display: grid;
    grid-template-columns: 1fr 2fr;
    text-align: left;
}
```

JSS

This is a JavaScript code that fetches Pokemon data from the PokeAPI and creates a dynamic webpage to display the Pokemon. The code is divided into four main parts:

Event listeners for the previous and next buttons: The code listens for clicks on the previous and next buttons and fetches a new set of Pokemon based on the current offset and limit.

Functions to fetch Pokemon data: The code fetches Pokemon data from the PokeAPI using the fetch() method. It then converts the response to JSON using the .json() method and passes the resulting data to another function to create the Pokemon on the webpage.

Functions to create Pokemon on the webpage: The code creates a new div element to represent each Pokemon and sets its contents based on the Pokemon data fetched from the API. The createPokemon() function also creates a flip-card container to allow the user to see the back of the card with more Pokemon information.

Functions to remove child nodes: The code includes a function that removes all child nodes of a given parent element. This is used to clear the Pokemon container before fetching a new set of Pokemon.

Overall, this code fetches data from the PokeAPI, creates Pokemon cards with front and back faces, and allows the user to navigate through the list of Pokemon by clicking on previous and next buttons