// Select the root element in the HTML where we will append our content

const app = document.getElementById('root');

const PNG = document.getElementById('image')

// Bool variable for the toggle function

let isDataVisible = false;

let isDataVisible2 = false;

// Create an element segment for the F1 logo

const logo = document.createElement('img');

logo.src = 'FormulaOne.png';

// Creates a div for the toggle buttons

const buttons = document.createElement('div');

buttons.setAttribute('class', 'container');

// Container 1 - 4 are all created here and use the same container class CSS

const container = document.createElement('div');

container.setAttribute('class', 'container');

const container2 = document.createElement('div');

container2.setAttribute('class', 'container');

const container3 = document.createElement('div');

container3.setAttribute('class', 'container');

const container4 = document.createElement('div');

container4.setAttribute('class', 'container');

// Appends all the containers to the root div

PNG.appendChild(logo);

app.appendChild(buttons)

app.appendChild(container);

app.appendChild(container2);

app.appendChild(container3);

app.appendChild(container4)

//The Json API used does not have images, the following 3 arrays of images are used in specific container.

//Drivers Images

let images= ['1alealb01.png','2feralo01.png','3olibea01.png','4valbot01.png',

  '5fracol01.png','6piegas01.png','7lewham01.png','8nichul01.png','9lialaw01.png',

  '10chalec01.png','11kevmag01.png','12lannor01.png','13estoco01.png','14serper01.png',

  '15oscpia01.png','16danric01.png','17georus01.png','18carsai01.png','19logsar01.png',

  '20lanstr01.png','21yuktsu01.png','22maxver01.png','23guazho01.png'

]

//Constructors Images

let ConImages = ['alpine.png','aston-martin.png','ferrari.png','haas.png','mclaren.png',

  'mercedes.png','rb.png','red-bull-racing.png','sauber.png','williams.png'

]

//Track Images

let TrackImages = ['1Bahrain\_Circuit.png','2Saudi\_Arabia\_Circuit.png','3Australia\_Circuit.png',

  '4Japan\_Circuit.png','5China\_Circuit.png','6Miami\_Circuit.png','7Emilia\_Romagna\_Circuit.png',

  '8Monaco\_Circuit.png','9Canada\_Circuit.png','10Spain\_Circuit.png','11Austria\_Circuit.png',

  '12Great\_Britain\_Circuit.png','13Hungary\_Circuit.png','14Belgium\_Circuit.png','15Netherlands\_Circuit.png',

  '16Italy\_Circuit.png','17Baku\_Circuit.png','18Singapore\_Circuit.png','19USA\_Circuit.png',

  '20Mexico\_Circuit.png','21Brazil\_Circuit.png','22Las\_Vegas\_Circuit.png','23Qatar\_Circuit.png',

  '24Abu\_Dhabi\_Circuit.png'

]

//FetchDrivers function fetches the data from the URL.json

async function fetchDrivers() {

  const DriversTitle = document.createElement('img'); //the image element for Drivers Title

  DriversTitle.src = 'F1 Drivers 2024.png';

  container.appendChild(DriversTitle);

  //counter for the images

  let counter = 0

  try {

    const response = await fetch('https://api.jolpi.ca/ergast/f1/2024/drivers/');

    const data = await response.json(); // Parses the response as JSON

    const drivers = data.MRData.DriverTable.Drivers;

    drivers.forEach(driver => {

      console.log(`Name: ${driver.givenName} ${driver.familyName}`); //writes all the data to the console for testing purposes

      console.log(`Number: ${driver.permanentNumber}`);

      console.log(`Nationality: ${driver.nationality}`);

      console.log(`More Info: ${driver.url}`);

      console.log('---');

      createDriverCards(driver, counter) //calls the function to sort and print the Data

      counter++

    });

  } catch (error) {

    // Display an error message if the fetch fails

    const errorMessage = document.createElement('div');

    errorMessage.textContent = `Gah, it's not working! Error: ${error.message}`;

    app.appendChild(errorMessage);

  }

}

//Function to sort and print the Data

function createDriverCards(driver, counter)

{

    const card = document.createElement('div');

    card.setAttribute('class', 'card'); // Styling class for each card

    const driverName = document.createElement('h1');

    driverName.textContent = `${driver.givenName} ${driver.familyName}`; // Set the title for each card

    const driverimage = document.createElement('img') //image for the cards

    driverimage.src = "photos/"+images[counter];

    //Data form the json to be sorted and printed

    const driverNumber = document.createElement('h2')

    driverNumber.textContent = `Driver Number : ${driver.permanentNumber}`;

    const driverCode = document.createElement('h3')

    driverCode.textContent = `Driver initials : ${driver.code}`

    const driverNationality = document.createElement('h3')

    driverNationality.textContent = `Nationality : ${driver.nationality}`;

    const driverDOB = document.createElement('h3')

    driverDOB.textContent = `Date of Birth : ${driver.dateOfBirth}`;

    container.appendChild(card); // Append card to container

    card.appendChild(driverName); //Appends the data to each card

    card.appendChild(driverimage)

    card.appendChild(driverNumber);

    card.appendChild(driverCode);

    card.appendChild(driverNationality);

    card.appendChild(driverDOB);

};

//FetchConstructors function fetches the data from the URL.json

async function fetchConstructors() {

  let counter2 = 0

  const constructorTitle = document.createElement('img'); //the image element for Constructors Title

  constructorTitle.src = 'F1 Teams 2024.png';

  container2.appendChild(constructorTitle);

    try {

      const response = await fetch('https://api.jolpi.ca/ergast/f1/2024/constructors/');

      const data = await response.json(); // Parses the response as JSON

        const constructors = data.MRData.ConstructorTable.Constructors; //access the specific data

        constructors.forEach(teams => {

        console.log(`Name: ${teams.name}`);

        Constructor(teams, counter2)

        counter2++

      });

    } catch (error) {

      // Display an error message if the fetch fails

      const errorMessage = document.createElement('div');

      errorMessage.textContent = `Gah, it's not working! Error: ${error.message}`;

      app.appendChild(errorMessage);

    }

};

function Constructor(teams, counter2)

{

    const card = document.createElement('div');

    card.setAttribute('class', 'card');

    const constructorsName = document.createElement('h1');

    constructorsName.textContent = `${teams.name}`;

    const constructorsimage = document.createElement('img')

    constructorsimage.src = "teamphotos/"+ConImages[counter2];

    const teamNationality = document.createElement('h2')

    teamNationality.textContent = `Nationality: ${teams.nationality}`;

    container2.appendChild(card); // Append card to container

    card.appendChild(constructorsName); //Append data to the card

    card.appendChild(constructorsimage);

    card.appendChild(teamNationality);

}

//FetchTrackData function fetches the data from the URL.json

async function fetchTracks() {

  let counter3 = 0

  const ScheduleTitle = document.createElement('img');

  ScheduleTitle.src = 'F1 Schedule 2024.png';

  container3.appendChild(ScheduleTitle);

  try {

    const response = await fetch('https://api.jolpi.ca/ergast/f1/2024/');

    const data = await response.json();

      const Track = data.MRData.RaceTable.Races;

      Track.forEach(tracks => {

      console.log(`Name: ${tracks.raceName}`);

      console.log(`Circuit: ${tracks.Circuit.circuitName}`)

      console.log(`Race Date & Time: ${tracks.date} -- ${tracks.time}`)

      console.log(`FP1 Date & Time: ${tracks.FirstPractice.date} -- ${tracks.FirstPractice.time}`)

      Tracks(tracks, counter3)

      counter3++

    });

  } catch (error) {

    // Display an error message if the fetch fails

    const errorMessage = document.createElement('div');

    errorMessage.textContent = `Gah, it's not working! Error: ${error.message}`;

    app.appendChild(errorMessage);

  }

};

function Tracks(tracks, counter3)

{

  let FP2 = ('')

  let FP3 = ('')

  let Quli = ('')

  let Sprint =('')

  let SQ1 = ('')

  const card = document.createElement('div');

  card.setAttribute('class', 'card'); // Styling class for each movie card

  const GPName = document.createElement('h1');

  GPName.textContent = `${tracks.raceName}`;

  const trackimage = document.createElement('img')

  trackimage.src = "trackphotos/"+TrackImages[counter3];

  const TrackName = document.createElement('h2')

  TrackName.textContent = `Circuit: ${tracks.Circuit.circuitName}`;

  const GPRace = document.createElement('h3')

  GPRace.textContent = `Race Date & Time: ${tracks.date} -- ${tracks.time}`;

  const FP1 = document.createElement('h3')

  FP1.textContent = `FP1 Date & Time: ${tracks.FirstPractice.date} -- ${tracks.FirstPractice.time}`;

  console.log(tracks.Qualifying)

  container3.appendChild(card); // Append card to container

  card.appendChild(GPName);

  card.appendChild(trackimage)

  card.appendChild(TrackName);

  card.appendChild(GPRace);

  card.appendChild(FP1)

  if (tracks.Sprint) //If statement to check the different race formats between weekends.. Sprint race weekend and a typical Race Weekend

  {

    Quli = document.createElement('h3')

    Quli.textContent = `Qulifying Date & Time: ${tracks.Qualifying.date} -- ${tracks.Qualifying.time}`;

    Sprint = document.createElement('h3')

    Sprint.textContent = `Sprint Date & Time: ${tracks.Sprint.date} -- ${tracks.Sprint.time}`;

    SQ1 = document.createElement('h3')

    SQ1.textContent = `Sprint Date & Time: ${tracks.SprintQualifying.date} -- ${tracks.SprintQualifying.time}`;

    card.appendChild(SQ1);

    card.appendChild(Sprint); //and appends the correct race format

    card.appendChild(Quli);

  }

  else

  {

    FP2 = document.createElement('h3')

    FP2.textContent = `FP2 Date & Time: ${tracks.SecondPractice.date} -- ${tracks.SecondPractice.time}`;

    FP3 = document.createElement('h3')

    FP3.textContent = `FP3 Date & Time: ${tracks.ThirdPractice.date} -- ${tracks.ThirdPractice.time}`;

    Quli = document.createElement('h3')

    Quli.textContent = `Qulifying Date & Time: ${tracks.Qualifying.date} -- ${tracks.Qualifying.time}`;

    card.appendChild(FP2);

    card.appendChild(FP3); //appends the correct race format

    card.appendChild(Quli);

  }

}

//FetchStandings function fetches the data from the URL.json

async function fetchStandings() {

  const StandingsTitle = document.createElement('img');

  StandingsTitle.src = '2024 Drivers’ Standings.png';

  container4.appendChild(StandingsTitle);

  let count = 1

  try {

    const response = await fetch('https://api.jolpi.ca/ergast/f1/2024/driverstandings/');

    const data = await response.json(); // Parses the response as JSON

    console.log(data)

    const standings = data.MRData.StandingsTable.StandingsLists[0].DriverStandings // access the correct data for sorting and display

    console.log(standings)

    standings.forEach(driver => {

      console.log(`Name: ${driver.Driver.givenName} ${driver.Driver.familyName}`); //writes the data to the console

      console.log(`Ponits: ${driver.points}`);

      console.log(`Wins: ${driver.wins}`);

      console.log('---');

      createStandings(driver, count)

      count++

    });

  } catch (error) {

    // Display an error message if the fetch fails

    const errorMessage = document.createElement('div');

    errorMessage.textContent = `Gah, it's not working! Error: ${error.message}`;

    app.appendChild(errorMessage);

  }

}

function createStandings(driver, count)

{

    const card = document.createElement('div');

    card.setAttribute('class', 'card');

    const driverName = document.createElement('h2');

    driverName.textContent = `${count} : ${driver.Driver.givenName} ${driver.Driver.familyName}` // Set the title for each card

    const Points = document.createElement('h3')

    Points.textContent = `Points : ${driver.points}`;

    const Wins = document.createElement('h3')

    Wins.textContent = `Wins : ${driver.wins}`;

    container4.appendChild(card); // Append card to container

    card.appendChild(driverName);

    card.appendChild(Points);

    card.appendChild(Wins);

};

//Display function for the toggle data function

async function Display()

{

  fetchDrivers();

  fetchConstructors();

  fetchTracks();

}

//same as previous but this is for the drivers standing

async function Display2()

{

  fetchStandings();

}

//Toggle between hiding the data from the user and showing the data

async function toggleData() {

  const button = document.getElementById("toggleButton");

  if (isDataVisible) {

      // Hides the data

      container.innerHTML = "";

      container2.innerHTML="";

      container3.innerHTML="";

      button.textContent = "Show F1 Data";

  } else {

      // calls the Display function where the called fetch functions are waiting to display the data

      await Display();

      button.textContent = "Hide F1 Data";

  }

  // switches between true and false bool.

  isDataVisible = !isDataVisible;

}

//A Copy of the toggle function for the drivers standings

async function toggleData2() {

  const button = document.getElementById("toggleButton2");

  if (isDataVisible2) {

      // Hides the data

      container4.innerHTML="";

      button.textContent = "Show F1 Standings Data";

  } else {

      // calls the Display function where the called fetch functions are waiting to display the data

      await Display2();

      button.textContent = "Hide F1 Standings Data";

  }

  isDataVisible2 = !isDataVisible2;

}

// Add event listener to the button

document.getElementById("toggleButton").addEventListener("click", toggleData);

document.getElementById("toggleButton2").addEventListener("click", toggleData2);

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Formula One</title>

  <link href="https://fonts.googleapis.com/css?family=Dosis:400,700" rel="stylesheet">

  <link href="style.css" rel="stylesheet">

</head>

<body>

  <!-- Div for the Title Image -->

  <div id="image"></div>

  <!-- Buttons for User Input -->

  <button id="toggleButton"> Formula One Data </button>

  <button id="toggleButton2"> Drivers Standings Data </button>

  <!-- Root div for the append the containers and cards -->

  <div id="root"></div>

  <!-- script.js  -->

  <script src="scripts.js"></script>

</body>

</html>

\* {

  box-sizing: border-box

}

html {

  -webkit-font-smoothing: antialiased;

  -moz-osx-font-smoothing: grayscale;

  font-family: 'Dosis', sans-serif;

  line-height: 1.6;

  color: #666;

  background: #F6F6F6;

}

#root {

  max-width: 1200px;

  margin: 0 auto;

}

#image {

  max-width: 1200px;

  margin: 0 auto;

}

#DriversButton {

  display: block;

  margin: 4rem auto;

  max-width: 100%;

  padding: 0 2.5rem 2.5rem

}

#ConstructorsButton{

  display: block;

  margin: 4rem auto;

  max-width: 100%;

  padding: 0 2.5rem 2.5rem;

}

#TracksButton{

  display: block;

  margin: 4rem auto;

  max-width: 100%;

  padding: 0 2.5rem 2.5rem;

}

#toggleButton {

  display: block;

  margin: 4rem auto;

  max-width: 100%;

  padding: 0 2.5rem 2.5rem

}

#toggleButton2 {

  display: block;

  margin: 4rem auto;

  max-width: 100%;

  padding: 0 2.5rem 2.5rem

}

h1 {

  text-align: center;

  padding: 1.5rem 2.5rem;

  background-image: radial-gradient(#00a2ff 0%, #00087a 100%);

  margin: 0 0 2rem 0;

  font-size: 1.5rem;

  color: white;

}

img {

  display: block;

  margin: 4rem auto;

  max-width: 100%;

}

container2.img {

  display: block;

  margin: 4rem auto;

  max-width: 100%;

  padding: 0 2.5rem 2.5rem

}

container1.img {

  display: block;

  margin: 4rem auto;

  max-width: 100%;

  padding: 0 2.5rem 2.5rem

}

h2 {

  padding: 0 2.5rem 0;

  margin: 0;

}

h3 {

    padding: 0 2.5rem 0;

    margin: 0;

  }

p {

  padding: 0 2.5rem 2.5rem;

  margin: 0;

}

.container {

  display: flex;

  flex-wrap: wrap;

}

.card {

  margin: 1rem;

  background: white;

  box-shadow: 2px 4px 25px rgba(0, 0, 0, .1);

  border-radius: 12px;

  overflow: hidden;

  transition: all .2s linear;

}

.card:hover {

  box-shadow: 2px 8px 45px rgba(0, 0, 0, .15);

  transform: translate3D(0, -2px, 0);

}

@media screen and (min-width: 600px) {

  .card {

    flex: 1 1 calc(50% - 2rem);

  }

}

@media screen and (min-width: 900px) {

  .card {

    flex: 1 1 calc(33% - 2rem);

  }

}