

Universidad Católica Andrés Bello

Facultad de Ingeniería

Escuela de Ingeniería Informática

Cátedra: Programación Orientada a la Web

**Práctica Nro. 7 – Consumiendo APIs**

Profesora: Estudiante:

Mora, Noraimar Patiño, Ángel

Puerto Ordaz, junio del 2024

**Introducción**

Mediante el planteamiento y posterior resolución de un problema en el cual se requiere crear una pagina web donde se muestre información de diferentes personajes de la serie Rick & Morty, se podrá reforzar los conocimientos adquiridos a lo largo de clases en referente al tema del consumo de APIs.

El principal objetivo será implementar un código JavaScript que permita consumir la información de los personajes mediante la API de Rick and Morty, con lo cual se procederá a desarrollar tarjetas de personajes en donde se detallara dicha información como el nombre del personaje, su estado, raza, debut en la serie, ultima localización y una imagen que haga referencia al personaje en determinado momento de la serie.

Sera crucial el uso de async/await y fetch para lograr el cometido de la práctica y además, se deberá desarrollar un carrusel de imágenes para el banner, investigar e implementar una BD del lado del cliente mediante IndexedDB y finalmente considerar un diseño responsive para los usuarios de dispositivos móviles.

**Contenido – Resolución del Problema Planteado**

Desarrolle un sitio web como el que se muestra a continuación, considerando lo siguiente:

• Replicar diseño presentado (5 ptos)

• Cargar la información desde el API <https://rickandmortyapi.com/api> (Documentación: https://rickandmortyapi.com/documentation) (3 ptos)

• Almacenar la información del lado del cliente usando IdexedDB (2 ptos)

• Adaptar el diseño para su visualización en móviles (3 ptos)

• En la sección del banner, presentar un carousel de imágenes (2 ptos)

• Usar como base el archivo rick\_and\_morty\_base.zip

**Código Fuente - Solución del Enunciado**

**HTML**

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8">

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

    <meta name="author" content="Ángel Patiño">

    <meta name="description" content="Consuming The Rick and Morty API">

    <title>Consuming The Rick and Morty API</title>

    <link rel="stylesheet" href="css/styles.css">

    <link rel="shortcut icon" href="img/favicon-32x32.png" type="image/x-icon">

    <script src="https://kit.fontawesome.com/8fc56b0b18.js" crossorigin="anonymous"></script>

  </head>

  <body>

    <header>

      <a href="https://rickandmortyapi.com/documentation"><img src="img/rick-and-morty-logo.png" alt="Rick and Morty Logo"></a>

      <nav id="navbar">

        <ul>

          <li><a href="">Characters</a></li>

          <li><a id="documentation" href="https://rickandmortyapi.com/documentation">API DOCS</a></li>

        </ul>

      </nav>

    </header>

    <main>

      <section id="hero">

        <ul>

          <li><img src="img/rick-and-morty-desktop-1.png" alt="First slider image"></li>

          <li><img src="img/rick-and-morty-desktop-2.png" alt="Second slider image"></li>

          <li><img src="img/rick-and-morty-desktop-3.png" alt="Third slider image"></li>

        </ul>

        <h1>Consuming The Rick and Morty API</h1>

      </section>

      <div id="characters-container">

        <div id="characters">

          <!-- Characters' List -->

        </div>

        <!-- Load Button -->

        <div id="load">

          <button type="button" id="load-btn">LOAD MORE...</button>

        </div>

      </div>

    </main>

    <footer id="copyright">

      &lt;&gt;&nbsp;<p>by <a href="#hero">Ángel Patiño</a> 2024</p>

    </footer>

    <script src="js/functions.js"></script>

  </body>

</html>

**CSS**

@import url("https://fonts.googleapis.com/css?family=Open+Sans|Roboto:400,700&display=swap");

:root {

    --clr-dark-1: #272B33;

    --clr-dark-2: #383B42;

    --clr-white: #ffffff;

    --clr-light-grey: #f5f5f5;

    --clr-grey: #9E9E9E;

    --clr-green: #55CC44;

    --clr-red: #D63D2E;

    --clr-orange: #FF9800;

    --clr-black: #222222;

    --ff-primary: "Roboto", sans-serif;

    --ff-secondary: "Open Sans", sans-serif;

}

\*,

::after,

::before {

    margin: 0;

    padding: 0;

    box-sizing: border-box;

    font-size: 16px;

    /\*background-color: var(--clr-dark-1);\*/

}

body {

    font-family: var(--ff-primary);

}

header {

    background-color: var(--clr-black);

    display: flex;

    justify-content: space-between;

    align-items: center;

    padding: 1em 2em;

}

#navbar ul {

    display: flex;

    list-style: none;

    gap: 2em;

}

#navbar a {

    color: var(--clr-white);

    text-decoration: none;

    font-weight: bold;

}

#navbar #documentation {

    border: 1px solid var(--clr-orange);

    padding: 0.5em 1em;

    border-radius: 0.6em;

    transition: background-color 0.5s;

}

#navbar #documentation:hover {

    background-color: var(--clr-orange);

}

#hero {

    position: relative;

    background-color: var(--clr-dark-1);

    overflow: hidden;

}

#hero ul {

    display: flex;

    width: 300%;  /\* 3 times the width of the container \*/

    animation: slide 15s infinite alternate ease-in-out; /\* 15 seconds / 3 img = 5 seconds per img \*/

}

#hero ul li {

    list-style: none;

    width: 100%;

}

#hero img {

    object-fit: cover;

    position: relative;

    width: 100%;

    height: auto;

}

#hero h1 {

    position: absolute;

    font-size: 140px;

    top: 1.6em;

    bottom: 0;

    right: 0;

    left: 0;

    text-align: center;

    color: var(--clr-black);

}

@keyframes slide {

    0% {

        margin-left: 0%;

    }

    30% {

        margin-left: 0%;

    }

    35% {

        margin-left: -100%;

    }

    65% {

        margin-left: -100%;

    }

    70% {

        margin-left: -200%;

    }

    100% {

        margin-left: -200%;

    }

}

#characters-container {

    display: flex;

    flex-direction: column;

    justify-content: center;

    align-items: center;

    gap: 3.8em;

    padding: 3.5em 0em;

    background-color: var(--clr-dark-1);

}

#characters-container #characters {

    display: grid;

    grid-template-columns: repeat(2, 1fr);

    gap: 2em;

}

#characters-container #load-btn {

    border: 1px solid var(--clr-orange);

    background-color: var(--clr-dark-1);

    color: var(--clr-white);

    font-weight: bold;

    padding: 0.8em 1em;

    border-radius: 0.6em;

    cursor: pointer;

    transition: background-color 0.5s;

}

#characters-container #load-btn:hover {

    background-color: var(--clr-orange);

}

.character {

    display: flex;

    justify-content: space-between;

    align-items: center;

    gap: 1em;

    background-color: var(--clr-dark-2);

    border-radius: 0.6em;

    width: 35em;

}

.character img {

    object-fit: cover;

    width: 16em;

    height: 15em;

    border-top-left-radius: 0.6em;

    border-bottom-left-radius: 0.6em;

}

.character .character-info {

    display: flex;

    flex-direction: column;

    gap: 0.5em;

    text-align: justify;

    padding-right: 8.5em;

}

.character .character-info h3 {

    font-size: 19px;

}

.character .character-info .status {

    padding-bottom: 2em;

    font-size: 15px;

}

.character .character-info .name, .character .character-info .status, .character .character-info .desc, .character .character-info i {

    color: var(--clr-white);

}

.character .character-info .apparition {

    color: var(--clr-grey);

}

.character .character-info .icon i {

    color: #9E9E9E;

}

footer {

    display: flex;

    justify-content: center;

    align-items: center;

    background-color: var(--clr-black);

    color: var(--clr-grey);

    padding: 4em 2em 2em 2em;

    text-align: center;

}

footer a {

    color: var(--clr-white);

    text-decoration-color: var(--clr-orange);

    font-weight: bold;

}

@media screen and (max-width: 768px) {

    #characters-container #characters {

        grid-template-columns: repeat(1, 1fr);

    }

    #hero h1 {

        font-size: 1.6em;

    }

    #characters-container {

        gap: 1em;

    }

    .character {

        width: 20em;

    }

    .character img {

        width: 10em;

        height: 22.5em;

    }

    .character .character-info {

        padding-right: 1.5em;

    }

    .character .character-info .status {

        padding-bottom: 2.5em;

    }

}

**JavaScript**

// Character card creation

function createCard(img, name, status, race, location, debut) {

    let statusColor = '';

    let characterCard = document.createElement('div');

    characterCard.className = 'character';

    if (status === 'Alive') {

        statusColor = '#55CC44';

    } else if (status === 'Dead') {

        statusColor = "#D63D2E";

    } else {

        statusColor = "#9E9E9E";

    }

    characterCard.innerHTML = `

        <img src="${img}" alt="Character">

        <div class="character-info">

            <h3 class="name">${name}</h3>

            <p class="status"><span class="icon"><i class="fa-solid fa-circle" style="color: ${statusColor}"></i></span>&nbsp;${status} - ${race}</p>

            <p class="apparition">Last Known location:</p>

            <p class="desc">${location}</p>

            <p class="apparition">First seen in:</p>

            <p class="desc">${debut}</p>

        </div>

    `;

    charactersContainer.appendChild(characterCard);

}

// Add character cards to the page

function addCharacterCard() {

    while (index < 6) {

        let random = randomNumber();

        if (!characterIDs.includes(random)) {

            characterIDs.push(random);

            fetchData(random);

            index++;

        }

    }

    index = 0;

}

// Get a random number

function randomNumber () {

    return Math.floor(Math.random() \* 827) + 1;

}

// Fetch data from the API

async function fetchData(i) {

    let characterResponse = await fetch(`https://rickandmortyapi.com/api/character/${i}`);

    let characterData = await characterResponse.json();

    let episodeResponse = await fetch(characterData.episode[0]);

    let episodeData = await episodeResponse.json();

    createCard(characterData.image, characterData.name, characterData.status, characterData.species, characterData.location.name, episodeData.name);

}

// Elements from the HTML file

const charactersContainer = document.getElementById('characters');

const loadBtn = document.getElementById('load-btn');

var index = 0;

var characterIDs = [];

window.onload = addCharacterCard;

loadBtn.addEventListener('click', addCharacterCard);

**Conclusiones**

Para la realización de la actividad se utilizaron los conocimientos adquiridos en todas las practicas anteriores para lo que fue el desarrollo del HTML, CSS y JavaScript de la página web, sin embargo, esta vez se requirió del uso de async/await y fetch para poder consumir la API de Rick y Morty.

Como en la página principal de la API se muestra un diseño aleatorio (en donde cada vez que se ingresa a la página se muestran tarjetas de personajes diferentes), se decidió referenciar dicho diseño para la página realizada. Para poder generar más tarjetas se implementó un botón de LOAD MORE… el cual generara 6 personajes adicionales en la página.

Lamentablemente, el uso de IndexedDB no fue posible, pero idealmente se debía almacenar la información consumida de la API (solo los elementos importantes para el desarrollo de las tarjetas) en la BD de manera que, si ya se encontraba almacenada en el lado del cliente, no se requería del llamado continuo de la API.

Finalmente, se implementó un diseño responsive para las versiones de dispositivos móviles, en donde se redujo el tamaño de las tarjetas para que entraran y se vieran ordenadas y agradables para la vista humana.

**Referencias**

W3Schools. (s.f.). *JavaScript Random.* W3Schools.

<https://www.w3schools.com/js/js_random.asp>

Axel Fuhrmann. (s.f.). *The Rick and Morty API.* The Rick and Morty API.

<https://rickandmortyapi.com/>

W3Schools. (s.f.). Async/Await. W3Schools.

<https://www.w3schools.com/js/js_async.asp>