Universidad de Costa Rica
Escuela de Ciencias de la Computación e Informática

Ingeniería de Software

CI-0126

Laboratorio 2 A y B

HTML, CSS y Javascript

Repositorio del laboratorio
`https://github.com/MarioCordero/my-labs-inge-soft-CI-0126`

Profesora: Msc. Rebeca Obando Vásquez

Estudiante: Mario Cordero

Carnet: C22306

26 de agosto de 2025

1. Laboratorio 2: HTML, CSS y Javascript

1.1. Parte A: HTML y CSS

1.1.1. ¿Qué son forms en HTML?

Estructuras que permiten al usuario ingresar datos que luego pueden ser enviados a un servidor para su procesamiento. Se escriben usando la etiqueta `<form>` y se usan así:

Ejemplo básico de form

```
<form action="procesar.php" method="post">
    <label for="nombre">Nombre:</label>
    <input type="text" id="nombre" name="nombre" required>

    <label for="email">Correo:</label>
    <input type="email" id="email" name="email" required>

    <label for="comentario">Comentario:</label>
    <textarea id="comentario" name="comentario"></textarea>

    <input type="submit" value="Enviar">
</form>
```

1.1.2. ¿Qué es Bootstrap y cómo usarlo para tabs?

Framework CSS y JS que facilita el diseño de páginas web responsivas y modernas, ofreciendo componentes listos, como botones, menús, tablas y tabs.

Ejemplo de tabs con Bootstrap

```
<ul class="nav nav-tabs" id="myTab" role="tablist">
  <li class="nav-item">
    <a class="nav-link active" id="sobre-tab" data-bs-toggle="tab" href="#sobre" role="tab">Sobre mí</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" id="trabajos-tab" data-bs-toggle="tab" href="#trabajos" role="tab">Trabajos</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" id="pasatiempos-tab" data-bs-toggle="tab" href="#pasatiempos" role="tab">Pasatiempos</a>
  </li>
</ul>
```

1.1.3. ¿Qué es LESS y cómo usarlo?

Preprocesador CSS que permite usar variables, funciones, anidamiento y otras herramientas que no están disponibles en CSS puro.

Ejemplo de variables en LESS

```
@colorPrincipal: 3498db;
body  font-family: Arial, sans-serif; color: @colorPrincipal;
h1  font-size: 2em;
```

2. Parte B: Preguntas sobre JavaScript

2.0.1. 1. Data types soportados por JavaScript

- **Primitivos:** `string`, `number`, `boolean`, `null`, `undefined`, `symbol`, `bigint`.
- **Complejos:** `object`, `array`, `function`, `date`, etc.

2.0.2. 2. Crear un objeto en JavaScript

Ejemplo

```
const persona = {
    nombre: "Mario",
    edad: 21,
    saludar: function() {
        console.log(‘Hola, soy ${this.nombre}‘);
    }
};
```

2.0.3. 3. Alcances de variables

- **Global:** accesible desde cualquier parte del código.
- **Funcional:** declaradas con `var` dentro de funciones.
- **Bloque:** declaradas con `let` y `const` dentro de bloques.

2.0.4. 4. Diferencia entre undefined y null

- `undefined`: valor por defecto de variables no inicializadas.
- `null`: valor asignado intencionalmente para indicar ausencia.

2.0.5. 5. ¿Qué es el DOM?

El **DOM** (Document Object Model) es una representación en forma de árbol de los elementos HTML de una página web, que puede ser manipulada mediante JavaScript.

2.0.6. 6. Acceso a elementos del DOM

- `getElementById`: retorna un elemento por su ID.
- `querySelector`: selecciona el primer elemento que coincide con un selector CSS.

Ejemplo

```
const titulo = document.getElementById("titulo");
const primerParrafo = document.querySelector(".parrafo");
```

2.0.7. 7. Crear nuevos elementos en el DOM

Ejemplo

```
const nuevoDiv = document.createElement("div");
nuevoDiv.textContent = "Hola mundo";
document.body.appendChild(nuevoDiv);
```

2.0.8. 8. Propósito del operador this

Hace referencia al contexto actual del objeto o función en el que se está utilizando.

2.0.9. 9. Promises en JavaScript

Las **promesas** representan operaciones asíncronas.

Ejemplo

```
const promesa = new Promise((resolve, reject) => {
    setTimeout(() => resolve("Éxito"), 1000);
});
promesa.then(res => console.log(res));
```

2.0.10. 10. Fetch en JavaScript

Permite hacer solicitudes HTTP de forma sencilla.

Ejemplo

```
fetch('https://api.example.com/data')
  .then(response => response.json())
  .then(data => console.log(data));
```

2.0.11. 11. Async/Await

Simplifica el manejo de promesas.

Ejemplo

```
async function obtenerDatos() {
    const res = await fetch('https://api.example.com/data');
    const data = await res.json();
    console.log(data);
}
```

2.0.12. 12. Callback

Una función que se pasa como argumento a otra función.

Ejemplo

```
function saludar(nombre, callback) {
    console.log("Hola " + nombre);
    callback();
}
saludar("Mario", () => console.log("Bienvenido"));
```

2.0.13. 13. Closure

Un **closure** es una función que recuerda el entorno en el que fue creada.

Ejemplo

```
function crearContador() {
    let contador = 0;
    return function() {
        contador++;
        return contador;
    }
}
const contar = crearContador();
console.log(contar()); // 1
console.log(contar()); // 2
```

2.0.14. 14. Crear cookies en JavaScript

Ejemplo

```
document.cookie = "usuario=Mario; expires=Fri, 31 Dec 2025 23:59:59 GMT; path=/";
```

Referencias

- GetBootstrap. (2025). *Navs and tabs – Bootstrap v5*. Recuperado el 25 de agosto de 2025, de https://getbootstrap.com/docs/5.0/components/navs-tabs/
- Mozilla Contributors. (2025). *async function*. MDN Web Docs. Recuperado el 25 de agosto de 2025, de https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/async_function
- Mozilla Contributors. (2025). *Callbacks*. MDN Web Docs. Recuperado el 25 de agosto de 2025, de https://developer.mozilla.org/en-US/docs/Glossary/Callback_function
- Mozilla Contributors. (2025). *Closures*. MDN Web Docs. Recuperado el 25 de agosto de 2025, de https://developer.mozilla.org/en-US/docs/Web/JavaScript/Closures
- Mozilla Contributors. (2025). *Document: getElementById, querySelector*. MDN Web Docs. Recuperado el 25 de agosto de 2025, de https://developer.mozilla.org/en-US/docs/Web/API/Document
- Mozilla Contributors. (2025). *Document.cookie*. MDN Web Docs. Recuperado el 25 de agosto de 2025, de https://developer.mozilla.org/en-US/docs/Web/API/Document/cookie
- Mozilla Contributors. (2025). *Document Object Model (DOM)*. MDN Web Docs. Recuperado el 25 de agosto de 2025, de https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model
- Mozilla Contributors. (2025). *Fetch API*. MDN Web Docs. Recuperado el 25 de agosto de 2025, de https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API
- Mozilla Contributors. (2025). *Null and undefined*. MDN Web Docs. Recuperado el 25 de agosto de 2025, de https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/null
- Mozilla Contributors. (2025). *Promises*. MDN Web Docs. Recuperado el 25 de agosto de 2025, de https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise
- Mozilla Contributors. (2025). *Scopes and closures*. MDN Web Docs. Recuperado el 25 de agosto de 2025, de https://developer.mozilla.org/en-US/docs/Glossary/Scope
- Mozilla Contributors. (2025). *this*. MDN Web Docs. Recuperado el 25 de agosto de 2025, de https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/this
- Mozilla Contributors. (2025). *Working with objects*. MDN Web Docs. Recuperado el 25 de agosto de 2025, de https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Working_with_Objects
- The Core Less Team. (s.f.). *Getting started | Less.js*. Recuperado el 25 de agosto de 2025, de https://lesscss.org/

- Tutorial Republic. (2025). *Creating Forms with Bootstrap*. Recuperado el 25 de agosto de 2025, de https://www.tutorialrepublic.com/twitter-bootstrap-tutorial/bootstrap-forms.php
- W3Schools. (2025). *Bootstrap Forms*. Recuperado el 25 de agosto de 2025, de https://www.w3schools.com/bootstrap/bootstrap_forms.asp