

Laboratorio Práctico: Mostrar Información de Empleados utilizando Métodos de Array



Tiempo estimado necesario: 30 minutos

Lo que aprenderás

En este laboratorio, crearás un sistema de gestión de empleados. Utilizarás botones para activar funciones como mostrar todos los empleados, calcular salarios totales, filtrar y mostrar empleados del departamento de recursos humanos, y encontrar empleados por sus IDs. Usarás funciones de JavaScript para generar listas dinámicas de empleados utilizando métodos de array como `forEach`, `filter`, `reduce` y `find` para gestionar y presentar datos de manera interactiva.

Objetivos de aprendizaje

Después de completar este laboratorio, podrás:

- **Programación impulsada por eventos:** Aprender a activar funciones mediante clics en botones para la manipulación del DOM.
- **Dominio de métodos de array:** Adquirir experiencia en métodos de array de JavaScript (`forEach`, `filter`, `reduce`, `find`) para la manipulación de datos.
- **Manipulación dinámica:** Comprender cómo crear y actualizar elementos HTML dentro de una página web de manera dinámica.
- **Habilidades de desarrollo front-end:** Desarrollar habilidades fundamentales para crear interfaces interactivas y gestionar la presentación de datos en páginas web.

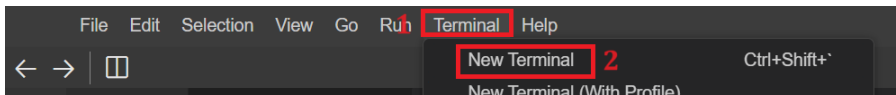
Requisitos previos

- Conocimientos básicos de HTML y comandos de Git.
- Comprensión básica de las variables de JavaScript y su alcance.
- Navegador web con consola (Chrome DevTools, Consola de Firefox, etc.).

Paso 1: Configuración del entorno

1. Primero, necesitas clonar tu repositorio principal en el **Entorno de Skills Network**. Sigue los pasos dados:

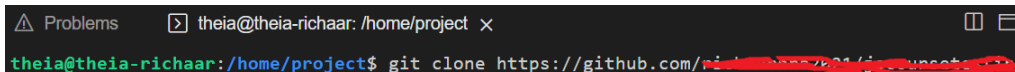
- Haz clic en la terminal en el panel superior derecho y luego selecciona **Nueva Terminal**.



- Realiza el comando `git clone` escribiendo el comando dado en la terminal.

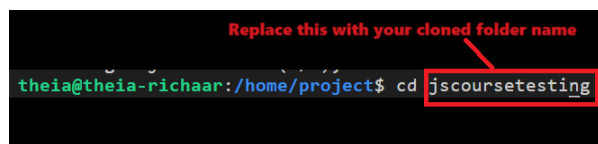
```
git clone <github-repository-url>
```

Nota: Pon tu propio enlace de repositorio de GitHub en lugar de `<github-repository-url>` y debería verse como se indica a continuación:



- El paso anterior clonará la carpeta de tu repositorio de GitHub en la carpeta del proyecto en el explorador. También verás múltiples carpetas dentro de la carpeta clonada.
- Ahora necesitas navegar dentro de la carpeta clonada. Para esto, escribe el comando dado en la terminal:

```
cd <repository-folder-name>
```



Nota: Escribe el nombre de tu carpeta clonada en lugar de `<repository-folder-name>`. Realiza `git clone` si has cerrado sesión en el **Entorno de Skills Network** y no puedes ver ningún archivo o carpeta después de iniciar sesión.

- Ahora, selecciona la carpeta **cloned Folder Name**, haz clic derecho sobre ella y elige **Nueva Carpeta**. Ingresas el nombre de la carpeta como **employeeDetails**. Esto creará la carpeta para ti. Luego, selecciona la carpeta **employeeDetails**, haz clic derecho y elige **Nuevo Archivo**. Ingresas el nombre del archivo como **employee_details.html** y haz clic en Aceptar. Esto creará tu archivo HTML.
- Ahora selecciona la carpeta **employeeDetails** nuevamente, haz clic derecho y selecciona **Nuevo Archivo**. Ingresas el nombre del archivo como **employee_details.js** y haz clic en Aceptar. Este proceso creará tu archivo JavaScript.
4. Crea una estructura básica de plantilla de un archivo HTML agregando el contenido proporcionado a continuación.

```
<!DOCTYPE html>
<html>
<head>
  <title>Sistema de Gestión de Empleados</title>
</head>
<body>
  <h1>Sistema de Gestión de Empleados</h1>
  <div>
    <button onclick="displayEmployees()">Mostrar Empleados</button>
    <button onclick="calculateTotalSalaries()">Calcular Sueldos Totales</button>
    <button onclick="displayHREmployees()">Mostrar Empleados de RRHH</button>
    <button onclick="findEmployeeById(2)">Buscar Empleado por ID 2</button>
  </div>
  <div id="employeesDetails"></div>
<script src="./employee_details.js"></script>
</body>
</html>
```

- Define botones dentro de un `div` para activar funciones de JavaScript para gestionar un sistema de gestión de empleados.
- Incluye cuatro botones. Al hacer clic en estos botones, se ejecutan funciones de JavaScript para mostrar empleados, calcular sueldos totales, filtrar empleados de RRHH y encontrar un empleado por ID.

- El código HTML incluye tres `<div>` para mostrar información de empleados según las acciones activadas por el usuario sin que la página se recargue dinámicamente al hacer clic en cualquiera de los botones.
- La etiqueta de script se utiliza en el archivo HTML anterior a la etiqueta `</body>` para incluir el archivo JS en **employee_details.html**.

Nota: Después de pegar el código, guarda tu archivo.

Paso 2: Definición de variables y funciones

1. En **employee_details.js**, inicializa el objeto array de empleados. Inicialízalo con pares de clave y valor de la siguiente manera:

```
const employees = [
  { id: 1, name: 'John Doe', age: 30, department: 'IT', salary: 50000 },
  { id: 2, name: 'Alice Smith', age: 28, department: 'HR', salary: 45000 },
  { id: 3, name: 'Bob Johnson', age: 35, department: 'Finance', salary: 60000 },
  //... Se pueden agregar más registros de empleados aquí
];
```

2. Crea la función **displayEmployees()** para mostrar los detalles de los empleados en el archivo **employee_details.js**.

```
// Función para mostrar todos los empleados
const totalEmployees = employees.map((employee, index) => `<p>${employee.id}: ${employee.name}: ${employee.name} - ${employee.department} - ${employee.salary}</p>`).join('');
document.getElementById('employeesDetails').innerHTML = totalEmployees;
}
```

- El método `map` itera a través de cada empleado en el array de empleados. Para cada empleado, construye una cadena de plantilla literal `<p>${employee.id}: ${employee.name}: ${employee.name} - ${employee.department} - ${employee.salary}</p>`, encapsulada dentro de etiquetas `<p>`, para representar los detalles del empleado.
- El array resultante de cadenas construidas se une en una sola cadena usando `join("")`. Esta concatenación fusiona todos los detalles de los empleados formateados en HTML en una cadena continua sin separadores.
- El método `map` almacena los detalles de los empleados en la variable **totalEmployees**, que muestra los detalles en el elemento `<div>` (con la ayuda de un ID) que muestra la información del empleado en la página web.

3. Crea la función **calculateTotalSalaries()** para calcular los salarios totales de los empleados. Incluye el código dado a continuación después del código JavaScript anterior.

```
function calculateTotalSalaries() {
  const totalSalaries = employees.reduce((acc, employee) => acc + employee.salary, 0);
  alert(`Total Salaries: ${totalSalaries}`);
}
```

- El método `reduce` itera a través de cada empleado y acumula sus salarios para calcular el total. El valor inicial del acumulador (`acc`) es 0.
- El método `reduce` acumula continuamente estos salarios sumando el salario de cada empleado al total anterior.
- El salario de cada empleado (`employee.salary`) se añade al acumulador (`acc`). Después de calcular la suma total de salarios, se activa un cuadro de diálogo de alerta usando `alert()`. Muestra los salarios totales calculados con el mensaje "Total Salaries: \$" seguido del valor de la variable `totalSalaries` calculada. Esta alerta muestra la suma total de todos los salarios de los empleados.

4. Crea la función **displayHREmployees()** para mostrar los detalles de los empleados según el departamento, como el departamento de **HR**. Incluye el código dado a continuación después del código JavaScript anterior.

```
function displayHREmployees() {
  const hrEmployees = employees.filter(employee => employee.department === 'HR');
  const hrEmployeesDisplay = hrEmployees.map((employee, index) => `<p>${employee.id}: ${employee.name}: ${employee.name} - ${employee.department} - ${employee.salary}</p>`).join('');
  document.getElementById('employeesDetails').innerHTML = hrEmployeesDisplay;
}
```

- El código anterior filtra el array de empleados usando el método **filter**, aislando y recolectando empleados cuyo propiedad de departamento coincide con 'HR' en un nuevo array llamado `hrEmployees`.
- Luego muestra los detalles coincidentes en la página principal como se muestra en la función **displayEmployees** usando el método `map` dentro del `<div>` usando su ID **employeesDetails**.

5. Crea la función **findEmployeeById()** para mostrar los detalles de los empleados según el ID. Incluye el código dado a continuación después del código JavaScript anterior.


```
function findEmployeeById(employeeId) {
  const foundEmployee = employees.find(employee => employee.id === employeeId);
  if (foundEmployee) {
    document.getElementById('employeesDetails').innerHTML = `<p>${foundEmployee.id}: ${foundEmployee.name}: ${foundEmployee.name} - ${foundEmployee.department} - ${foundEmployee.salary}</p>`;
  }
  else{
    document.getElementById('employeesDetails').innerHTML = 'no se ha encontrado ningún empleado con este ID';
  }
}
```

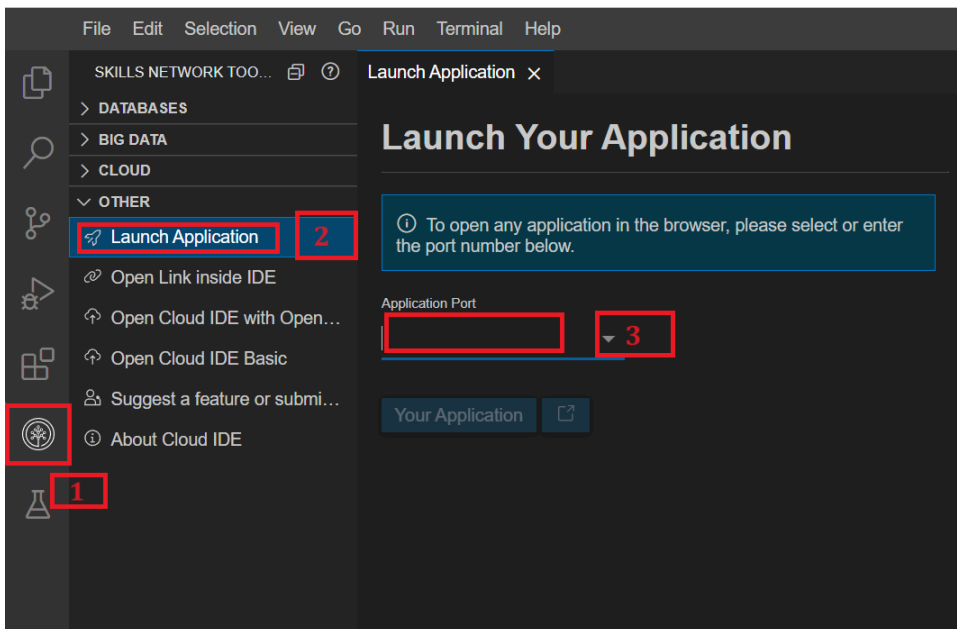
- Este código utiliza el método `find` para localizar un empleado en el array de empleados cuyo ID coincide con el ID del empleado proporcionado, almacenando el objeto del empleado encontrado en la variable `foundEmployee`.
- Se emplea una declaración `if` para verificar si se encuentran los detalles para ese ID en particular y mostrar los detalles en consecuencia.

Paso 3: Verifica la salida

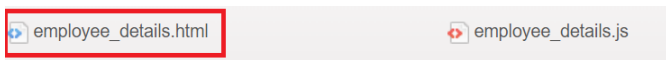
1. Para ver tu página HTML, haz clic derecho en el archivo **employee_details.html** después de seleccionarlo, luego selecciona "Abrir con Live Server".
2. El servidor debería iniciarse en el puerto 5500, indicado por una notificación en la esquina inferior derecha.



3. Haz clic en el botón de Skills Network a la izquierda (consulta el número 1) para abrir la "Caja de Herramientas de Skills Network". Luego, haz clic en **Lanzar Aplicación** (consulta el número 2). Desde allí, ingresa el número de puerto 5500 en el número 3 y haz clic en este botón .



4. Se abrirá tu navegador predeterminado, donde primero verás el nombre de tu carpeta clonada. Haz clic en esa carpeta, y dentro de ella, entre otras carpetas, encontrarás el nombre de la carpeta **employeeDetails**. Haz clic en la carpeta **employeeDetails**, y luego selecciona el archivo **employee_details.html**, como se muestra a continuación.



5. Se abrirá la página principal, y verás la página principal como se muestra a continuación.

Employee Management System

Display Employees Calculate Total Salaries Display HR Employees Find Employee by ID 2

6. Ahora, haz clic en todos los botones para ver la salida.

Nota: Después de pegar el código, guarda tu archivo. Puedes usar cualquier método de salida para esto. Si editas tu código, simplemente actualiza tu navegador ejecutándose a través del número de puerto 5500. De esta manera, no es necesario lanzar la aplicación una y otra vez.

Paso 4: Realizar comandos de Git

1. Realiza `git add` para agregar los últimos archivos y carpetas escribiendo el comando dado en la terminal en el entorno de git.

```
git add --a
```

Asegúrate de que la terminal tenga la ruta como sigue:



2. Luego realiza `git commit` en la terminal. Al realizar `git commit`, la terminal puede mostrar un mensaje para configurar tu `git config --global` para `user.name` y `user.email`. Si es así, necesitarás ejecutar el comando `git config` también para `user.name` y `user.email` como se indica.

```
git config --global user.email "you@example.com"
```

```
git config --global user.name "Your Name"
```

Nota: Reemplaza los datos entre comillas con tus propios detalles.

Luego realiza el comando de commit como se indica:

```
git commit -m "mensaje"
```

3. Luego realiza `git push` simplemente escribiendo el comando dado en la terminal.

```
git push origin
```

- Después del comando push, el sistema te pedirá que ingreses tu nombre de usuario y contraseña. Ingresa el nombre de usuario de tu cuenta de GitHub y la contraseña que creaste en el primer laboratorio. Después de ingresar las credenciales, todas tus últimas carpetas y archivos serán enviados a tu repositorio de GitHub.

Tarea de práctica

1. En esta tarea de práctica, necesitas realizar una funcionalidad en la que se pueda obtener información en base a la especialización del empleado.
2. Para esto, incluye un par clave-valor más en el array de objetos de empleados dentro de cada objeto como se muestra en la captura de pantalla proporcionada.

```
const employees = [  
  { id: 1, name: 'John Doe', age: 30, department: 'IT', salary: 50000, specialization: 'Javascript' },  
  { id: 2, name: 'Alice Smith', age: 28, department: 'HR', salary: 45000, specialization: 'Python' },  
  { id: 3, name: 'Bob Johnson', age: 35, department: 'Finance', salary: 60000, specialization: 'Java' },  
  //... More employee records can be added here  
];
```

3. Luego, crea un botón para buscar un empleado en función de la especialización como se muestra en la captura de pantalla proporcionada.

Employee Management System

Display Employees Calculate Total Salaries Display HR Employees Find Employee by ID 2 Find by Specialization JavaScript

4. A continuación, crea una función de JavaScript para mostrar los detalles de los empleados que tienen especialización en **JavaScript**. Puedes referirte a la función **findEmployeeById** en el código de JavaScript de este laboratorio para orientación.
5. La salida será como se muestra en la captura de pantalla proporcionada.

Employee Management System

Display Employees Calculate Total Salaries Display HR Employees Find Employee by ID 2 Find by Specialization JavaScript

1: John Doe - IT - JavaScript

Resumen

1. **Configurando el entorno:** El código HTML incluye tres `<div>` para mostrar la información de los empleados basada en acciones desencadenadas por el usuario sin que la página se recargue dinámicamente al hacer clic en cualquier botón.
2. **Definiendo variables y funciones:** El método `map` almacena los detalles de los empleados en la variable **totalEmployees**, que muestra los detalles en el elemento `<div>` (con la ayuda de un ID) y muestra la información de los empleados en la página web, mientras que el método `reduce` itera a través de cada empleado y acumula sus salarios para calcular el total.
3. **Verificando la salida:** Los comandos `git add`, `git commit` y `git push` actualizan los cambios en tu carpeta **Scope_Lab**; repositorio de GitHub para una gestión adecuada del código.

© IBM Corporation. Todos los derechos reservados.