Guía de Actividades Diagnósticas

Objetivo

El propósito de esta guía es realizar un diagnóstico inicial para conocer el nivel de conocimientos previos de los estudiantes en relación con JavaScript moderno (ES6), fundamentos de programación y primeros pasos con React. Los resultados permitirán ajustar la planificación del curso y ofrecer apoyos específicos según las necesidades del grupo.

Actividades

Actividad 1: Fundamentos de JavaScript

Objetivo: Evaluar el manejo de variables, tipos de datos, funciones y estructuras básicas.

Instrucciones:

1. Escribir un programa que:
   - Declare dos variables (`nombre` y `edad`)
   - Imprima un mensaje como: *"Hola, me llamo Ana y tengo 22 años."*
2. Dado el siguiente array:

```
const numeros = [3, 7, 12, 5, 2];
```

- Usar `map` para generar un nuevo array con los números al cuadrado.
- Usar `filter` para obtener los números mayores a 5.

1. Escribir una función flecha que reciba un número y devuelva si es par o impar.

Actividad 2: Desestructuración y objetos

Objetivo: Verificar la comprensión de objetos, acceso a propiedades y desestructuración.

Instrucciones:

Dado el siguiente objeto:

```
const persona = {
  nombre: "Lucía",
  edad: 28,
  profesion: "Diseñadora",
};
```

1. Mostrar en consola un mensaje que diga: *"Lucía tiene 28 años y trabaja como Diseñadora."* usando desestructuración.
2. Agregar una nueva propiedad al objeto llamada `ciudad` con el valor `"Rosario"`.

Actividad 3: Funciones y callbacks

Objetivo: Comprobar el uso de funciones como parámetros (callbacks).

Instrucciones:

1. Escribir una función que reciba un array y una función callback. La función debe aplicar el callback a cada elemento del array y retornar el nuevo array.

Ejemplo de uso:

```
procesar([1, 2, 3], x ⇒ x * 2); // [2, 4, 6]
```

Actividad 4: Primer componente en React

Objetivo: Verificar si los estudiantes pueden crear un componente básico y usar props.

Instrucciones:

1. Crear un componente llamado `Saludo` que reciba una prop `nombre` y muestre un saludo personalizado.

```
<Saludo nombre="Martín" />
```

Debe renderizar: **"Hola Martín"**

1. Crear otro componente llamado `Presentacion` que reciba `nombre`, `edad` y `profesion`, y renderice un párrafo con la información.

Actividad 5: Tailwind CSS

Objetivo: Diagnosticar si los estudiantes pueden integrar clases de Tailwind en JSX.

Instrucciones:

1. Crear un botón utilizando Tailwind con las siguientes características:
   - Fondo azul
   - Texto blanco
   - Bordes redondeados
   - Efecto hover que oscurezca el botón

Ejemplo:

```
<button className="bg-blue-500 text-white px-4 py-2 rounded hover:bg-blue-700">
  Hacer clic
</button>
```

Actividad 6: Profundización en métodos de arrays (map, filter, reduce)

Objetivo: Fortalecer el dominio de métodos funcionales modernos para transformar, filtrar y reducir datos.

Instrucciones:

Dado el siguiente array de objetos:

```
const productos = [
  { nombre: "Notebook", precio: 1200 },
  { nombre: "Mouse", precio: 20 },
  { nombre: "Teclado", precio: 50 },
  { nombre: "Monitor", precio: 300 },
```

```
  { nombre: "Auriculares", precio: 80 },
];
```

1. Usar `filter` para obtener solo los productos cuyo precio sea mayor a 100.
2. Usar `map` para obtener un nuevo array de strings con el siguiente formato: *"Notebook: $1200"*
3. Usar `reduce` para calcular el precio total de todos los productos.
4. Combinar `filter` y `map` para obtener los nombres de los productos que cuesten menos de 100, todo en minúsculas.

Actividad 7: Integración con Tailblocks.cc

Objetivo: Aplicar componentes preconstruidos de Tailblocks en un proyecto React y adaptarlos al formato JSX.

Instrucciones:

1. Ingresar a https://tailblocks.cc y seleccionar una sección (por ejemplo: *Hero*, *Contact*, *Features*, *CTA*, etc.).
2. Copiar el código HTML de un componente y pegarlo en un archivo `.jsx`.
3. Realizar las siguientes adaptaciones para que sea válido en React:
   - Reemplazar `class` por `className`
   - Cerrar correctamente las etiquetas (`<img />`, `<input />`, etc.)
   - Eliminar atributos inválidos como `autocomplete="off"` o `action="#"` si no están siendo usados.
   - En los `onClick`, asegurarse de usar funciones válidas de JS o crear funciones en el componente.
4. Personalizar el componente con tu propio contenido (texto, imágenes, íconos, etc.).
5. Insertarlo dentro de un componente principal (`App.jsx`) para visualizarlo.

Ejemplo de modificación básica:

```
<!-- Original en HTML -->
<button class="bg-blue-500 text-white">Enviar</button>
```

```
// Adaptado a JSX
<button className="bg-blue-500 text-white">Enviar</button>
```

1. (Opcional) Combiná más de un bloque de Tailblocks para crear una landing page sencilla con React + Tailwind.

Modo de entrega

Las actividades pueden realizarse en un entorno online como CodeSandbox o StackBlitz o bien crear un repositorio de GitHub que aloje las actividades solicitadas.

Compartir el enlace en el aula virtual.

Evaluación

Esta guía no se calificará numéricamente. Su objetivo es **diagnóstico** y servirá para:

- Identificar fortalezas y debilidades individuales y grupales
- Personalizar el acompañamiento durante el curso
- Ajustar contenidos si es necesario
- Ajustar la velocidad de las clases sincrónicas