1. Introducción a los estilos en CSS

En esta sesión, se comentarán los fundamentos de cómo aplicar estilos a los distintos componentes de nuestro proyecto en `React` mediante `CSS`. Para ello existen diversas técnicas y herramientas que podemos utilizar para mejorar su apariencia y presentación aunque, tal y como se verá más adelante, nos centraremos únicamente en una de ellas.

1.1. Los estilos en `React`

Los estilos presentan un papel fundamental en la experiencia del usuario en una aplicación web. La forma en que se muestran y se organizan la información y los elementos visuales puede influir en la usabilidad, accesibilidad y la percepción general de la calidad de una aplicación. Además, los estilos, si están correctamente diseñados pueden mejorar la legibilidad y la coherencia de la interfaz de usuario, lo que a su vez aumentará la satisfacción del usuario y la efectividad de dicha aplicación.

En `React`, existen varias formas de aplicar estilos a los componentes:

- `CSS` **en archivos separados**: los estilos se definen en archivos `CSS` externos para posteriormente ser importados en los distintos componentes de `React`.
  Esta es una forma tradicional de aplicar estilos y permite una separación muy clara entre lo que es la estructura del componente y su presentación visual.
- **Estilos en línea**: en este caso, los estilos se aplicarán directamente en el código `JSX` de los componentes utilizando el atributo `style`. Esta técnica puede ser conveniente para estilos muy simples y específicos para un único componente, pero puede volverse difícil de mantener en aplicaciones más grandes.
- **Módulos** `CSS`: se trata de una forma de encapsular estilos en archivos `CSS` locales a cada componente. Los estilos se importan y aplican de manera modular, lo que evita conflictos de nombres y hace que sean más fáciles de mantener y escalar en una aplicación `React`.
- `CSS` **en *JavaScript***: utiliza una combinación de *JavaScript* y `CSS` para definir estilos de componentes de una forma más dinámica y flexible.
- **Uso de preprocesadores**: `React` presenta compatibilidad con preprocesadores tales como `SASS`, `Less` o `Stylus` para poder trabajar de una forma más eficiente con `CSS`.
- **Uso de frameworks**: una manera para agilizar la creación de estilos es usando *frameworks* como *React Boostrap*, *MaterialUI* o *TailwindCSS* entre otros.

1.2. `CSS` en archivos separados

Se trata de la manera más conocida y fácil de agregar hojas de estilos a un proyecto en `React`. Este proceso también se conoce como `CSS Simple` o `Global` y lo que lo hace tan sencillo es que no se necesitan ningún tipo de configuración previa para su uso.

Sus ventajas principales son el hecho de mantener una vista limpia del componente, ya que tenemos bien separado el `CSS` y la posibilidad de utilizar todas las herramientas que pone `CSS` a nuestro alcance como pseudoselectores o variables.

1.3. Módulos CSS

Los módulos CSS permiten, al mismo tiempo, separar los estilos sólo para los componentes sin necesidad de utilizar hojas de estilo globales, aplicándose únicamente a los componentes que los usan. Por tanto, y comparándolos con el caso anterior, estaremos trabajando con CSS Simple pero evitando que la hoja de estilos afecte a todo el proyecto.

Para la creación de un módulo, lo primero que debemos hacer es agregar la palabra `module` al nombre de la hoja de estilo. Posteriormente, para usarlo en el componente, debemos importarlo. En este caso hemos creado un documento llamado `styles.module.css`.

```
.character {
  display: flex;
}

.character__header {
  padding: 10px;
}
```

De la misma forma, podemos disponer de un conjunto de archivos. En este caso tenemos el siguiente, que es `types.module.css`.

```
.big {
  font-size: 24px;
}

.small {
  font-size: 14px;
}
```

Y, por último, el archivo de React mencionamos la hoja de estilos seguida del selector:

```
const Character = () => (
  <div className={styles.character}>
    <header className={styles.character__header} >
      <h1 className={types.big} >
        Título
      </h1>
    </header>
  </div>
)
```

Muy importante es que en este último archivo debemos importar los dos anteriores:

```
import styles from './styles.module.css';
import types from './types.module.css';
```

1.4. Estilos en línea

Los estilos en línea no son una práctica recomendada a la hora de escribir CSS ya que, aunque pueden ser útiles en ciertas situaciones, como una manera rápida de proporcionar estilo a un

componente, presentan grandes limitaciones, especialmente en términos de mantenimiento, organización del código o reutilización.

En `React` podemos optar entre dos alternativas para escribir estos estilos:

- Estilos en línea de la forma tradicional.

```
const Character = () => (
  <div style={{ display: "block"; }}>
      ...
  </div>
)
```

- Declarando los estilos como un objeto de `React`.

```
const Character = () => {
  const character = {
    display: "flex"
  }

  const character__header = {
    padding: "10px"
  }

  const character__title = {
    font-size: "24px"
  }

  return (
    <div style={character}>
      <header style={character__header}>
        <h1 style={character__title}>Título</h1>
      </header>
    </div>
  )
}
```

Como ocurre con las hojas de estilos externas, para aplicar los estilos en línea tampoco necesitamos ninguna configuración adicional y, como ya se ha comentado pueden ser útiles en situaciones muy concretas, como a la hora de de realizar cálculos dinámicos. Aunque también debemos tener en cuenta que se limita mucho el uso de `CSS` al no poder utilizar algunas de sus características como animaciones o pseudoselectores.

Además, si la aplicación deja de ser pequeña ya no es una opción viable, porque no se trata de un código escalable y sería necesario migrar los estilos en línea y convertirlos a hojas de estilo.