

PROGRAMACIÓN EN NUEVAS TECNOLOGÍAS 2

React: React Components, Introducción React Native

UNIDAD PREVIA

- React
- Programación Declarativa Vs Imperativa
- Props
- State
- todoApp.js

COMPONENTES

- Retorna un nodo (Elementos a ser renderizados en la app)
- Representa una parte de la Interfaz Gráfica (UI) de nuestra app.
- Los componentes en react native deben actuar como funciones en base a sus props.
- Existen dos tipos:
 - *Functional Component (FC)*: Componentes de función, tal como se vió en previas clases.
 - *Class Component. (React.Component)*: Componentes de clase extendiendo de React Component.

CICLO DE VIDA DE UN COMPONENTE

- MOUNT: Se **ejecuta automáticamente** al momento de inyectar el componente en mi UI.
- UPDATE: No se ejecuta automáticamente, sino que entre en juego cuando se actualiza un **props** o un **state**.
- UNMOUNT: Se ejecuta automáticamente, al momento de que el componente salga de la UI.

FUNCTIONAL COMPONENT (FC)

- Componentes simples: No es necesario trabajar con estados.
- Es una función simple que reciben props y retorna un nodo.
 - Debe ser “simple”: No debe tener ningún efecto en modelado de datos principal, actualizar estado, etc
 - Cualquier cambio en props va a causar que la función sea re-invocada.
- Para utilizar estados y ciclos de vida dentro de este componente es preciso el uso de Hooks

HOOKS

- Función que permite manipular estados y ciclos de vida de un componente
- Se usa en Componentes de Función.
- Debe ser llamado al inicio de un componente funcional
- La declaración no debe ser invocado condicionalmente!

USESTATE

- Es un Hook que nos permite manipular los estados de un componente funcional

```
const [count, setCount] = useState(0);
```

```
setCount(10)
```

USEEFFECT

- Es un Hook que nos permite acceder a los ciclos de vida de un componente funcional

```
useEffect(() => {  
  
  console.log("Initialized");  
  
  // clean up function runs before the component is unmounted  
  
  return () => {  
  
    console.log("Cleaned up");  
  
  };  
  
}, [count])
```


USECONTEXT

- Es un Hook que retorna la data dada por un contexto.
- Es una forma de manejar los estados globales en apps de reacts.
- Recomendados para aplicaciones chicas con bajo nivel de cambio de estados globales.

```
const data = useContext(dataContext);
```

REACT NATIVE

- Framework basado en React.
- Nos permite construir aplicaciones móviles usando solo código Javascript.
 - “Learn once, write anywhere”
- Soporta iOS y Android.

¿COMO FUNCIONA REACT NATIVE?

- Funciona a base de código Javascript.
 - Es Transpilado y minificado.
- Separa hilo de trabajo para UI, Layout y JavaScript.
- Comunicación Asíncrona mediante un “puente”.
 - Hilo de ejecución JS hace solicitud al UI para mostrar elementos.
 - JS puede ser bloqueado y el UI continuará su trabajo.

DIFERENCIAS DE REACT NATIVE VS WEB

- Basado en componentes.
- Estilos nativos según Plataforma.
- No utiliza API Browsers.
 - CSS animations, Canvas, SVG, etc.
 - Algunas APIs fueron traducidas: fetch, timers, console, etc.
- Navegación - .

COMPONENTES REACT NATIVE

- No existe en el scope global como los componentes React.
 - Para acceder es preciso usar la importación:
`import from 'react-native'`
- `div` -> `View`
- `span` -> `Text`
 - Todos los textos deben ser usados mediante la etiqueta: `<Text />`
- `button` -> `Button`
- `ScrollView`

<https://reactnative.dev/docs/components-and-apis>

MANEJO DE EVENTOS

- Al contrario de elementos WEB, no todos los componentes tienen todas las interacciones.
- Solo algunos componentes “Touchable”
 - Button
 - TouchableOpacity, TouchableHighlight, TouchableWithoutFeedback
 - TouchableNativeFeedback (Solo Android)
- Los eventos web reciben los eventos como argumentos. EN cambio en React Native se puede recibir distintos tipos de argumentos:
 - Ver documentacion: <https://reactnative.dev/docs/handling-touches>

STYLES

- React native usa Objetos para estilos en la app.
- Los atributos de esos objetos están basados en Propiedades CSS.
- Diseño Flexbox.
 - Por default el diseño por columnas.
- Las longitudes están en números sin unidades.
- Los props de estilos (style prop) pueden tomar un array de estilos.
- `StyleSheet.create()`
 - Funcionalmente lo mismo que crear objetos para estilos.
 - Optimización adicional: Solo envía IDs tokenizados para renderizar.
<https://reactnative.dev/docs/style>
<https://nativebase.io/> (Componentes extras mas desarrollados)
<https://reactnativeelements.com/>

Codificando en React Native

EXPO

- Es una plataforma que permite construir aplicaciones React, tanto web como react native.
- Es un set de herramientas de desarrollo que permite acelerar el proceso de desarrollo de React Native.
- Cuenta con los siguientes elementos:
 - Snack: Ejecuta aplicaciones React Native en la web.
 - XDE: Interfaz gráfica que permite ejecutar, compartir o publicar apps.
 - CLI: Lineas de comando que permite ejecutar, compartir o publicar apps.
 - Client (mobile): Permite ejecutar la app en el celular mientras se desarrolla.

- Requisitos:

- Nodejs: <https://nodejs.org/en/download/>
- NPM: Viene con la instalación de Nodejs.

- Instalación:

- En una terminal correr el siguiente comando

```
npx create-expo-app my-app
```

- Ejecutar mi primera app:

```
cd my-app  
npm start
```