Curso de React y React Native

Clase 12 y 13

Agenda de la clase

Agenda

- Repaso.
- Navigation en React Native.
- Ejercicios.

Navegación en React Native

Navegación

Al igual que para React en la Web, para la navegación en React Native contamos con varias opciones.

- React Router.
- React Navigation.

React Router transmite los mismos conceptos que vimos para Routing en web, lo que lo hace muy simple de aprender y utilizar; **será la que vamosa utilizar en el curso**.

React Navigation es la otra solución que suele usarse y es muy popular. Está orientada más a experiencias 100% nativas, pero a su vez utiliza un paradigma de navegación completamente distinto al que ya aprendimos en el curso.

React Router

En React Router, la mayoría de los elementos de navegación son Componentes de React, tratando de mantener el modo de pensar que se utiliza al desarrollar aplicaciones en React.

Para poder utilizarlo, se necesita instalarlo en el proyecto proyecto por `npm` o `yarn`:

- `$ npm install --save react-router-native`
- `$ yarn add react-router-native`

React Router

En vez de utilizar `BrowserRouter`, se utiliza `NativeRouter` para envolver la jerarquía de componentes, de forma que funcionen las Rutas en cualquiera de ellos. Para las rutas se sigue utilizando `Route`, pero importado desde la nueva librería.

```
import React from "react";
import { View } from "react-native";
import { NativeRouter, Route, Link } from "react-router-native";
const App = () =>
   <NativeRouter>
       <View style={styles.container}>
           <Route exact path="/" component={Home} />
           <Route path="/about" component={About} />
           <Route path="/topics" component={Topics} />
       </View>
   </NativeRouter>;
```

React Router

`Link` tambien es muy similar, pero también puede recibir una prop `component` donde le podemos indicar que componente utilizar para dibujarse. Por defecto utiliza `TouchableHighlight` y todas las props que recibe, se las pasa también a ese componente, por ejemplo, `underlayColor`.

```
<View style={styles.nav}>
      <Link to="/" underlayColor="#f0f4f7" style={styles.navItem}>
            <Text>Home</Text>
      </Link>
      <Link to="/about" underlayColor="#f0f4f7" style={styles.navItem}>
            <Text>About</Text>
      </Link>
      <Link to="/topics" component={TouchableOpacity} activeOpacity={0.1} style={styles.navItem}>
            <Text>Topics</Text>
      </Link>
</View>
```

React Router

Todos los componentes que son dibujados por `Route` van a recibir en sus Props la propiedad llamada `match`, que tiene acceso a varias cosas de la ruta como la URL (que en realidad es interna, esto no es un navegador web y no maneja URLs). Aquí también llegan en `params` los parámetros definidos en la ruta.

```
<Route path={`${match.url}/:topicId`} component={Topic} />
```

```
const Topic = ({ match }) =>
<View>
   <Text style={styles.topic}>
       Url is: {match.url}
   </Text>
   <Text style={styles.topic}>
       Param topicId is: {match.params.topicId}
   </Text>
</View>
```

React Router

Al igual que en la web, en el caso de tener rutas **ambiguas**, en la dos pueden matchear, se puede utilizar el `Switch` para envolver las rutas y va a devolver la primera que coincida.

```
import { NativeRouter, Route, Switch } from "react-router-native";
<NativeRouter>
   <View style={styles.container}>
     <Switch>
       <Route path="/about" component={About}/>
       <Route path="/company" component={Company}/>
       <Route path="/:user" component={User}/>
     </Switch>
   </View>
</NativeRouter>
```

React Router

Como puede apreciarse, las similitudes con React Router Web son grandes, motivo por el cual podemos reutilizar el conocimiento adquirido previamente en el curso para tener navegación en entornos nativos.

La documentación se encuentra en éste link.

Ejercicios

Ejercicio

1. Crear proyecto de React Native
2. Utilizando la API de libraries.io, listar las *platforms* disponibles.
3. Cada *platform* constituirá un Link a la página de la plataforma.
4. Al visitar la página de una plataforma, mostrar una barra de búsqueda para listar los resultados para dicha plataforma.
5. Cada resultado será un Link a la página de detalle de dicho proyecto, mostrando la información que nos provee la API.

No hace falta utilizar Redux. No obstante, quienes deseen re-utilizar la store del ejercicio de la clase 8, siéntanse libres de hacerlo.