TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE TLAXIACO

Reporte de Investigación: Integración de Mapas en React Native

Presenta:

Darío Sánchez Linton

Docente: Ing. Eloy Sánchez Salmorán

Materia: Desarrollo Móvil

Carrera: Ingeniería en Sistemas Computacionales

Grupo: 7 U

Especialidad: ISCDS / Desarrollo De Software Isie-Des-2019-01

29 de noviembre de 2023

INTRODUCCION

React Native es un framework JavaScript para crear aplicaciones reales nativas para iOS y Android, basado en la librearía de JavaScript React para la creación de componentes visuales, cambiando el propósito de los mismos para, en lugar de ser ejecutados en navegador, correr directamente sobre las plataformas móviles nativas, en este caso iOS y Andorid. Es decir, en lugar de desarrollar una aplicación web híbrida o en HTML5, lo que obtienes al final como resultado es una aplicación real nativa, indistinguible de la que podrías desarrollar con tu código en Objective-C o Java.

La red de GPS tiene un total de 27 satélites orbitando la tierra, de los cuales se utilizan 24 de forma activa y otros tres funcionan como refuerzo en el caso de que alguno de los activos falle. Esta red está repartida de tal manera que en cualquier sitio al que vayas puedas tener conexión directa con varios de estos satélites.

Y aquí es donde entra en acción el receptor GPS de tu móvil, que se conecta con al menos tres de los satélites que hay cerca de ti.

Los mapas se han convertido en una de las interfaces más populares de muchas de las aplicaciones que tenemos instaladas en nuestros teléfonos. Aprender a trabajar sobre mapas, representar la información de forma adecuada y crear una buena interfaz de navegación resulta cada vez más importante.

DESARROLLO

INTEGRACIÓN DE GOOGLE MAPS EN UNA APLICACIÓN DE REACT NATIVE, USANDO LA LIBRERÍA REACT-NATIVE-MAPS

Instalaciones previas:

- Node.js
- **React Native**: Para configurar el entorno de desarrollo seguir los pasos de la Documentación Oficial, teniendo en cuenta el sistema operativo en el que se desarrolla e ingresando a la sección *React Native CLI Quickstart.*

  - Si se va a correr en dispositivo físico se debe configurar.
- Un editor de código, recomendado Visual Studio Code.

Entorno de desarrollo con el que se realizó este tutorial:

- React: 16.11.0
- React-Native: 0.62.2
- Node: 12.14.0
- react-native-maps: 0.27.1

Crear el proyecto

1. Desde una terminal ejecutar el siguiente comando:

   npx react-native init projectName

2. Inicializar *Metro Bundler* en la terminal:

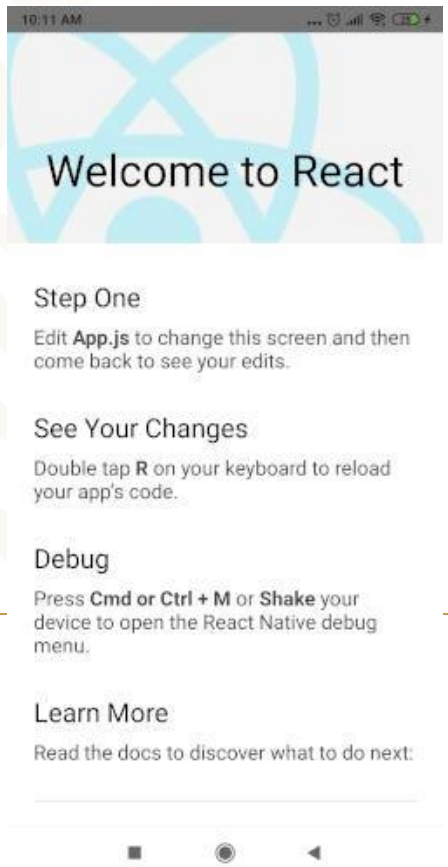
   cd projectName

   npx react-native start

3. Correr el proyecto en otra terminal.

   cd projectName

   npx react-native run-android



Instalar librería

Para iniciar corre el siguiente comando en la carpeta raíz del proyecto.

```
npm install react-native-maps --save
```

Esto agregará todas las dependencias necesarias en la carpeta `react-native-maps` dentro de `node_module`.

Ahora simplemente enlace la librería. Para ello corre el suiguiente comando:

```
react-native link react-native-maps
```

Debido a ciertos fallos en la librería hay que hacer unos arreglos tanto en Android como en iOS.

Arreglos en Android

Primero de todo se debe localizar la carpeta android. Dicha carpeta debería estar en la raíz del proyecto.

Mover los archivos en el compilador nativo, en este caso Android Studio.

Realiza los siguientes pasos:

1. Abrir Android Studio
2. Dar click en “Open an existing Android Studio project”
3. Localizar la carpeta “android” del proyecto
4. Dar en “OK” y listo

Localiza el archivo **app > build.gradle** y ábrelo.
Desplázate hacia abajo en la sección de “dependencies” y busca una línea parecida a la siguiente, si no la encuentras agrégala.

```
implementation project(':react-native-maps')
```

Si en lugar de implementation dice compile igual esta bien pero estarías usando librerías de Gradle obsoletas.

Localiza el archivo **settings.gradle** y ábrelo.

De igual manera debe de incluir las siguiente líneas:

```
include ':react-native-maps'
project(':react-native-maps').projectDir = new
File(rootProject.projectDir,
    '../node_modules/react-native-maps/lib/android')
```

En la carpeta **app > src > main > java** localiza el archivo **MainApplication.java** y ábrelo.

Debería incluir algo parecido a lo siguiente:

```
@Override
protected List<ReactPackage> getPackages() {
    return Arrays.asList(
            new MainReactPackage(),
            new MapsPackage()
    );
}
```

Una vez hecho eso agrega la clave al proyecto.

Para ello abre el archivo **app > src > main > AndroidManifest.xml** y agrega lo siguiente:

```
<meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="Cambiar_por_tu_Clave_API" />
```

Y Listo.

Arreglos en iOS

OJO los siguientes arreglos se deben realizar en una computadora mac.

Primero de todo se debe localizar la carpeta ios y abre la terminal en la misma.

Si la carpeta no incluye un archivo Podfile corre el siguiente comando:

```
pod init
```

Abre el archivo para edición y cámbialo a lo siguiente:

```
# Uncomment the next line to define a global platform for
your project
# platform :ios, '9.0'

target '_YOUR_PROJECT_TARGET_' do
  rn_path = '../node_modules/react-native'
  rn_maps_path = '../node_modules/react-native-maps'

  pod 'yoga', path:
"#{rn_path}/ReactCommon/yoga/yoga.podspec"
  pod 'React', path: rn_path, subspecs: [
    'Core',
    'CxxBridge',
    'DevSupport',
    'RCTActionSheet',
    'RCTAnimation',
    'RCTGeolocation',
    'RCTImage',
    'RCTLinkingIOS',
    'RCTNetwork',
    'RCTSettings',
    'RCTText',
    'RCTVibration',
    'RCTWebSocket',
  ]

  # React Native third party dependencies podspecs
  pod 'DoubleConversion', :podspec => "#{rn_path}/third-party
      podspecs/DoubleConversion.podspec"
```

```
  pod 'glog', :podspec => "#{rn_path}/third-party-
podspecs/glog.podspec"
  # If you are using React Native <0.54, you will get the
following error:
  # "The name of the given podspec `GLog` doesn't match the
expected one `glog`"
  # Use the following line instead:
  #pod 'GLog', :podspec => "#{rn_path}/third-party-
podspecs/GLog.podspec"
  pod 'Folly', :podspec => "#{rn_path}/third-party-
podspecs/Folly.podspec"

  # react-native-maps dependencies
  pod 'react-native-maps', path: rn_maps_path
  pod 'react-native-google-maps', path: rn_maps_path
  pod 'GoogleMaps'
  pod 'Google-Maps-iOS-Utils'
end

post_install do |installer|
  installer.pods_project.targets.each do |target|
    if target.name == 'react-native-google-maps'
      target.build_configurations.each do |config|
        config.build_settings['CLANG_ENABLE_MODULES'] = 'No'
      end
    end
    if target.name == "React"
      target.remove_from_project
    end
  end
end
```

Ahora en la terminal corre el siguiente comando:

```
pod install
```

Abre el proyecto en Xcode, pare ello sigue los siguientes pasos:

1. Abre Xcode
2. Da click en “Open another project”
3. Localiza la carpeta “ios” del proyecto
4. Dale en “open” y listo

Procediendo obtén la clave de la API. Para ello sigue el tutorial oficial.

En la subcarpeta principal (mismo nombre que la principal) abre el archivo **AppDelegate.m**.

Agrega la siguiente línea en la sección de librerías:

```
#import <GoogleMaps/GoogleMaps.h>
```

Posteriormente en el mismo archivo agrega:

```
...

- (BOOL)application:(UIApplication *)application
didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
+  [GMSServices provideAPIKey:@"Cambiar_por_tu_Clave_API"];
```

Compatibilidad

react-native-mapsrequiere react-native >= 0.64.3.

API de componentes

<MapView />API de componentes

<Marker />API de componentes

<Callout />API de componentes

<Polygon />API de componentes

<Polyline />API de componentes

<Circle />API de componentes

<Overlay />API de componentes

<Heatmap />API de componentes

<Geojson />API de componentes

Uso general

```
import MapView from 'react-native-maps';
```

o

```
var MapView = require('react-native-maps');
```

Este componente MapView está diseñado para que las características del mapa (como marcadores, polígonos, etc.) se especifiquen como elementos secundarios del propio MapView. Esto proporciona una API intuitiva y similar a una reacción para controlar de forma declarativa funciones en el mapa.

Representar un mapa con una región inicial

Vista del mapa

```
<MapView
 initialRegion={{
   latitude: 37.78825,
   longitude: -122.4324,
   latitudeDelta: 0.0922,
   longitudeDelta: 0.0421,
 }}
/>
```

Representar una lista de marcadores en un mapa

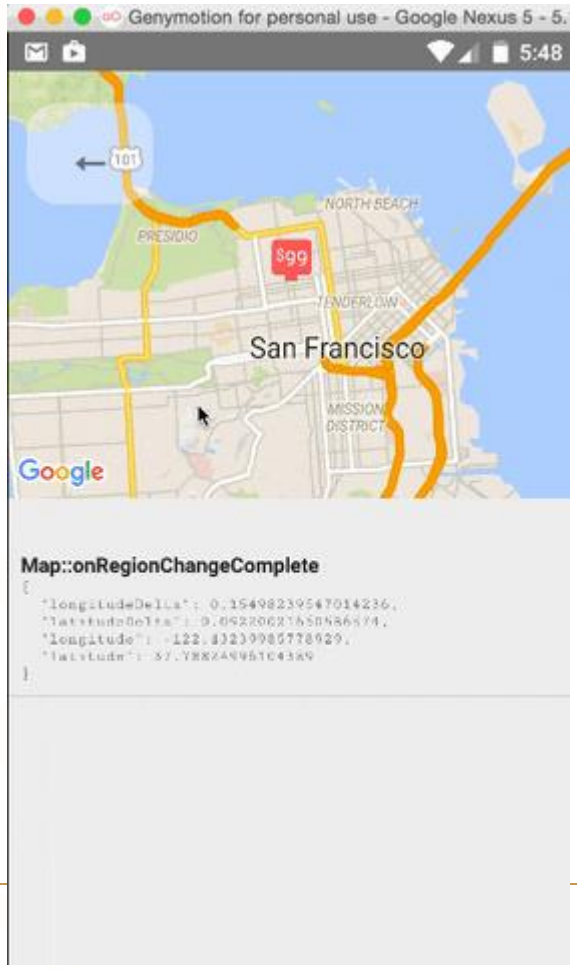
```
import {Marker} from 'react-native-maps';


<MapView region={this.state.region} onRegionChange={this.onRegionChange}>
 {this.state.markers.map((marker, index) => (
   <Marker
    key={index}
    coordinate={marker.latlng}
    title={marker.title}
    description={marker.description}
```

```
  />
 )}}
</MapView>;
```

Eventos de vista de mapa

El <MapView />componente y sus componentes secundarios tienen varios eventos a los que puede suscribirse. Este ejemplo muestra algunos de ellos en un registro a modo de demostración.

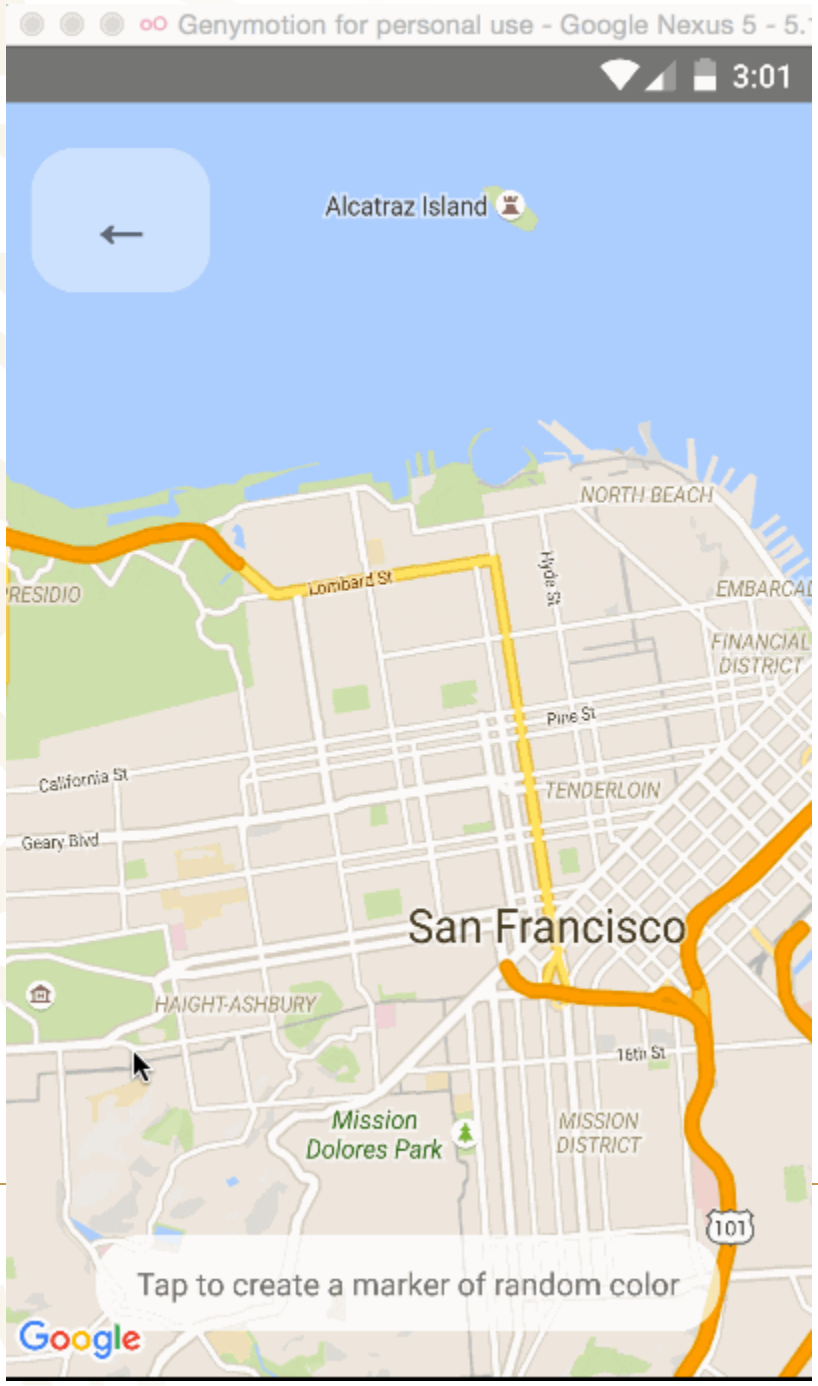


Región cambiante programáticamente

Se puede cambiar la posición de la vista del mapa usando referencias y métodos de componentes, o pasando un regionaccesorio actualizado. Los métodos del componente permitirán animar a una posición determinada como lo haría la API nativa.

Marcadores predeterminados

Los marcadores predeterminados se representarán a menos que se especifique un marcador personalizado. Opcionalmente, se puede ajustar el color del marcador predeterminado utilizando el pinColoraccesorio



Región animada

MapView puede aceptar un AnimatedRegionvalor como regionaccesorio. Esto le permite utilizar la API animada para controlar el centro y el zoom del mapa.

```
import MapView, { AnimatedRegion, Animated } from 'react-native-maps';


getInitialState() {
 return {
   region: new AnimatedRegion({
    latitude: LATITUDE,
    longitude: LONGITUDE,
    latitudeDelta: LATITUDE_DELTA,
    longitudeDelta: LONGITUDE_DELTA,
   }),
 };
}

onRegionChange(region) {
 this.state.region.setValue(region);
}

render() {
 return (
   <Animated
    region={this.state.region}
    onRegionChange={this.onRegionChange}
   />
 );
}
```

Posición del marcador animado

Los marcadores también pueden aceptar un AnimatedRegionvalor como coordenada.

```
import Mapview, { AnimatedRegion, MarkerAnimated } from 'react-native-maps';


getInitialState() {
 return {
  coordinate: new AnimatedRegion({
    latitude: LATITUDE,
    longitude: LONGITUDE,
  }),
 };
}


componentWillReceiveProps(nextProps) {
 const duration = 500


 if (this.props.coordinate !== nextProps.coordinate) {
  if (Platform.OS === 'android') {
   if (this.marker) {
    this.marker.animateMarkerToCoordinate(
     nextProps.coordinate,
     duration
    );
   }
  } else {
   this.state.coordinate.timing({
    ...nextProps.coordinate,
    useNativeDriver: true, // defaults to false if not passed explicitly
```

```
      duration
     }).start();
    }
   }
}

render() {
 return (
   <MapView initialRegion={...}>
    <MarkerAnimated
     ref={marker => { this.marker = marker }}
     coordinate={this.state.coordinate}
    />
   </MapView>
 );
}
```

Solución de problemas

El mapa esta en blanco

Asegúrese de haber instalado correctamente reaccionar-native-maps.

Consulte los registros si hay más información sobre el problema.

Intente configurar el estilo de MapView en una posición absoluta con los valores superior, izquierdo, derecho e inferior establecidos.

Asegúrese de haber habilitado la API de Google Maps en la consola de desarrollador de Google

```
const styles = StyleSheet.create({
 map: {
  ...StyleSheet.absoluteFillObject,
 },
});
```

```
<MapView
 style={styles.map}
 // other props
/>
```

Las entradas no se enfocan

Cuando las entradas no se enfocan o los elementos no responden al toque, observe el orden de la jerarquía de vistas; a veces el problema podría deberse al orden de los componentes renderizados; prefiera poner MapView como el primer componente.

Malo:

```
<View>
 <TextInput />
 <MapView />
</View>
```

Bien:

```
<View>
 <MapView />
 <TextInput />
</View>
```

Los componentes secundarios no se vuelven a renderizar

Los componentes que no están declarados por esta biblioteca (por ejemplo, marcadores, polilínea) no deben ser hijos del componente MapView debido a la metodología de representación única de MapView. Coloque sus componentes/vistas personalizados fuera del componente MapView y colóquelos de forma absoluta para garantizar que solo se vuelvan a representar según sea necesario. Ejemplo: Malo:

```
<View style={StyleSheet.absoluteFillObject}>
 <MapView style={StyleSheet.absoluteFillObject}>
```

```
  <View style={{position: 'absolute', top: 100, left: 50}} />
 </MapView>
</View>
```

Bien:

```
<View style={StyleSheet.absoluteFillObject}>
 <MapView style={StyleSheet.absoluteFillObject} />
 <View style={{position: 'absolute', top: 100, left: 50}} />
</View>
```

Fallo con EXC_BAD_ACCESS en iOS al cambiar de aplicación

<MapView>El uso de Apple Maps a mapType: "standard"veces falla cuando pones la aplicación en segundo plano o cambias a otra aplicación. Esto es solo un problema en XCode que usa Metal API Validation y no sucederá en producción. Para eliminar este problema incluso durante la depuración en XCode, vaya a Edit Scheme... -> Run (Debug) -> Diagnosticsy desmarque Metal -> API Validation. (h/t @Simon-TechForm).

La devolución de llamada onRegionChangeComplete() se llama infinitamente

Si el cambio de estado onRegionChangeCompletese realiza infinitamente, agregue una condición para limitar que estas llamadas ocurran solo cuando el cambio de región se realizó como resultado de la acción de un usuario.

```
onRegionChangeComplete={ (region, gesture) => {
        // This fix only works on Google Maps because isGesture is NOT available on Apple Maps
        if (!gesture.isGesture) {
  return;
 }

}}
```

CONCLUSIÓN

La integración de mapas en React Native constituye un herramienta potente y útil en diversas áreas de la vida laboral y cotidiana del usuario; tanto las facilidades personales que podrían considerarse de naturaleza simple como la incorporación en actividades más técnicas y precisas del ámbito científico o laboral. Aunque las librerías y métodos son muchos y variados, al igual que las técnicas de cada programador para implementarlas, conocer la incorporación de una tecnología como los mapas representa una cualidad ponderativa para el desarrollo de aplicaciones móviles acorde al avance tecnológico por la demanda de esta herramienta en las múltiples áreas en la vida humana.