


| | | |
|---|---|---|
|  | UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN | CICLO II |
| | DISEÑO Y PROGRAMACIÓN DE SOFTWARE MULTIPLATAFORMA CONFIGURACIÓN DE ENTORNO DE TRABAJO REACT NATIVE | GUIA DE LABORATORIO N° 4 |

I. RESULTADO DE APRENDIZAJE

Que el estudiante aprenda a:

- Configure el entorno de trabajo para realizar aplicaciones móviles.
- Utilice Visual Studio Code para la crear proyectos móviles.

II. CONFIGURANDO ENTORNO DE TRABAJO

Antes de comenzar con los primeros en el uso de React Native, debe verificar que posea el Software necesario para desarrollar y ejecutar esta tecnología. A continuación se le muestra las herramientas mínimas que debe poseer en su computadora:

1. **Software Developer Kit de Java** (SDK - Versión 12.0.2 o superior). El SDK reúne un grupo de herramientas que permiten la programación de aplicaciones móviles. Puede descargarlo en el siguiente enlace. Seleccione el tipo de instalador, según su la arquitectura y Sistema Operativo. Luego proceda a instalar el software.

<https://www.oracle.com/java/technologies/javase-downloads.html>

Java SE 16

Java SE 16.0.2 is the latest release for the Java SE Platform

- Documentation
- Installation Instructions
- Release Notes
- Oracle License
 - Binary License
 - Documentation License
- Java SE Licensing Information User Manual
 - Includes Third Party Licenses
- Certified System Configurations
- Readme

Oracle JDK

JDK Download

Documentation Download

| Java SE Development Kit 16.0.2 | | |
|--|-----------|---|
| This software is licensed under the Oracle Technology Network License Agreement for Oracle Java SE | | |
| Product / File Description | File Size | Download |
| Linux ARM 64 RPM Package | 144.87 MB | jdk-16.0.2_linux-aarch64_bin.rpm |
| Linux ARM 64 Compressed Archive | 160.75 MB | jdk-16.0.2_linux-aarch64_bin.tar.gz |
| Linux x64 Debian Package | 146.17 MB | jdk-16.0.2_linux-x64_bin.deb |
| Linux x64 RPM Package | 155.01 MB | jdk-16.0.2_linux-x64_bin.rpm |
| Linux x64 Compressed Archive | 170.04 MB | jdk-16.0.2_linux-x64_bin.tar.gz |
| macOS Installer | 166.6 MB | jdk-16.0.2_osx-x64_bin.dmg |
| macOS Compressed Archive | 167.21 MB | jdk-16.0.2_osx-x64_bin.tar.gz |
| Windows x64 Installer | 150.58 MB | jdk-16.0.2_windows-x64_bin.exe |
| Windows x64 Compressed Archive | 168.8 MB | jdk-16.0.2_windows-x64_bin.zip |

2. **Node.js**, ideado como entorno de ejecución de JavaScript orientado a eventos asíncronos, Node.js está diseñado para crear aplicaciones network escalables. Puede descargarlo e instalarlo en su computadora. <https://nodejs.org/es/>



[INICIO](#) | [ACERCA](#) | [DESCARGAS](#) | [DOCUMENTACIÓN](#) | [PARTICPE](#) | [SEGURIDAD](#) | [CERTIFICATION](#) | [NOTICIAS](#)

Node.js® es un entorno de ejecución para JavaScript construido con V8, motor de JavaScript de Chrome.

Descargar para Windows (x64)

18.17.1 LTS

Recomendado para la mayoría

20.5.1 Actual

Últimas características

Otras Descargas | Cambios | Documentación de la API | Otras Descargas | Cambios | Documentación de la API

O eche un vistazo al Programa de soporte a largo plazo (LTS).

3. **npm**, es un administrador de paquetes de JavaScript y gestor de dependencias. También se desempeña como administrador de proyectos.



Comando de instalación

`npm install [<package-spec> ...]`

Comando para verificar la instalación

`npm -v`

4. **Expo**, es una plataforma de código abierto para crear aplicaciones nativas para Android, iOS y web con JavaScript y React. Puede verificar los pasos de instalación en la siguiente página:



<https://docs.expo.dev/workflow/expo-cli/>


Comando de instalación

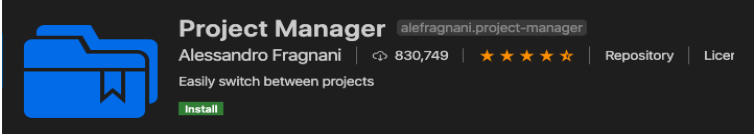
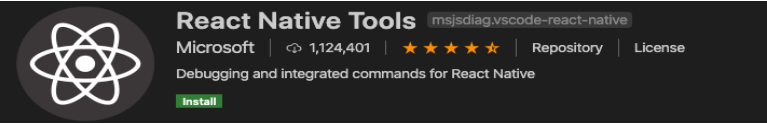
`yarn global add expo-cli`

5. **Visual Studio Code**, es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código. Puede descargarlo e instalarlo en su computadora: <https://code.visualstudio.com/download>



6. **Instalación de extensiones para Visual Studio Code**, puede ingresar a la siguiente página web para buscar las extensiones que se muestran en el cuadro de abajo <https://marketplace.visualstudio.com/VSCode> o puede utilizar su Visual Studio Code -> Menú "Ver" -> Extensiones (Ctrl + Mayús + X)

| # | Extensión | Descripción |
|-----|---|---|
| 6.1 |  Auto Close Tag <small>formulahendry.auto-close-tag</small> Jun Han 2,329,612 ★★★★★ Repository Automatically add HTML/XML close tag, same as Visual Studio IDE or Sublime Text Install | Permite agregar automáticamente la etiqueta de cierre HTML/XML. |
| 6.2 |  Bracket Pair Colorizer <small>coenraads.bracket-pair-colorizer</small> CoenraadS 2,304,900 ★★★★★ Repository License A customizable extension for colorizing matching brackets Install | Esta extensión permite identificar los corchetes con los colores. El usuario puede definir qué caracteres coincidir y qué colores usar. |
| 6.3 |  DotENV <small>mikestead.dotenv</small> mikestead 738,892 ★★★★★ Repository License Support for dotenv file syntax Install | Soporte para la sintaxis de archivos dotenv (variables de entorno) |
| 6.4 |  Firebase <small>toba.vsfire</small> toba 47,747 ★★★★★ Repository License Firestore Security Rules syntax highlighting Install | Resaltado de sintaxis de reglas de seguridad de Firebase. |
| 6.5 |  Prettier - Code formatter <small>esbenp.prettier-vscode</small> Esben Petersen 4,890,186 ★★★★★ ☆ Repository License Code formatter using prettier Install | Extensión que permite formatear (organizar) el código. |

| | | |
|-----|--|---|
| 6.6 |  <p>Project Manager alefragnani.project-manager Alessandro Fragnani 830,749 ★★★★★ Repository Licer Easily switch between projects Install</p> | Extensión que permite la organización y manejo fácil de proyectos. |
| 6.7 |  <p>React Native Tools msjsdiag.vscode-react-native Microsoft 1,124,401 ★★★★★ Repository License Debugging and integrated commands for React Native Install</p> | Extensión que permite la depuración de los comandos utilizados en React Native. |

7. **Android Studio**, proporciona las herramientas más rápidas para crear aplicaciones en todo dispositivo Android. Puede descargarlo e instalarlo en el siguiente enlace: <https://developer.android.com/studio>
8. **AVD Manager**, configurar un dispositivo virtual en Android Studio.

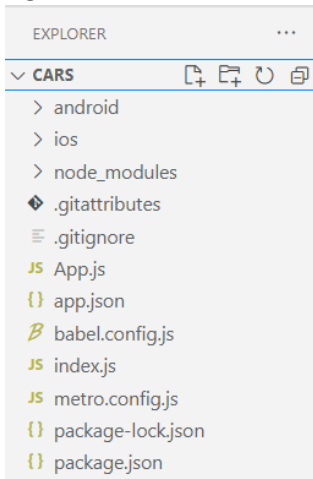
IV. DESARROLLO DE PRÁCTICA

PARTE 1

1. Abra la consola de comandos y ejecutar como administrador.
2. Ubicarse en consola en la dirección donde desea guardar su proyecto
3. En la consola de comandos ejecuta: **npm install -g create-react-native-app**
4. Ve a la carpeta donde deseas colocar tu proyecto desde tu consola de comandos y en esa carpeta ejecutar: **create-react-native-app init cars**
5. seleccionar la opción default new app:

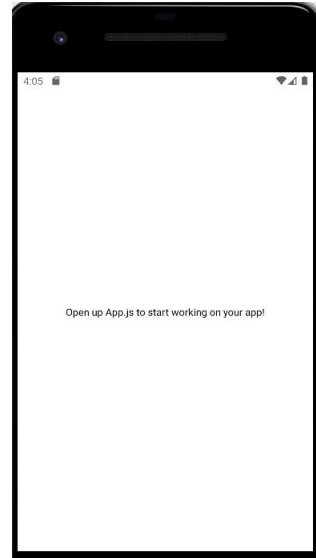
```
C:\Users\Karens_Medrano\Desktop>create-react-native-app cars
? How would you like to start » - Use arrow-keys. Return to submit.
> Default new app
Template from expo/examples: https://github.com/expo/examples
```

6. Después de generar la aplicación deberemos abrir este proyecto en nuestro editor y deberemos observar la siguiente estructura en nuestro proyecto.



7. Ejecutamos nuestra aplicación, escribiendo en consola el siguiente comando: **expo start / npm run android**

8. Deberíamos de visualizar lo siguiente en el emulador



9. Ahora cambiamos el archivo app.js de la aplicación para crear variables de objetos y mostrar en ella una lista de autos :

```
import React from 'react';
import { SafeAreaView, View, FlatList, StyleSheet, Text, StatusBar } from 'react-native';
```

```
const DATA = [
  {
    id: '1',
    title: 'Toyota',
  },
  {
    id: '2',
    title: 'Mazda',
  },
  {
    id: '3',
    title: 'Mitsubishi',
  },
];
```

```
const Item = ({ title }) => (
  <View style={styles.item}>
    <Text style={styles.title}>{title}</Text>
  </View>
);
```

```

export default function App() {

  const renderItem = ({ item }) => (
    <Item title={item.title} />
  );

  return (
    <SafeAreaView style={styles.container}>
      <FlatList
        data={DATA}
        renderItem={renderItem}
        keyExtractor={item => item.id}
      />
    </SafeAreaView>
  );
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    marginTop: StatusBar.currentHeight || 0,
  },
  item: {
    backgroundColor: '#f9c2ff',
    padding: 20,
    marginVertical: 8,
    marginHorizontal: 16,
  },
  title: {
    fontSize: 32,
  },
});

```

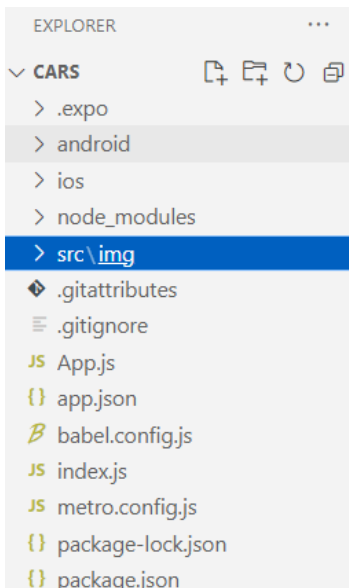
10. Debería de visualizar lo siguiente en el emulador:



11. Ahora procedemos a agregar imagen a esta lista de automóviles, primero importamos la librería siguiente:

```
import {AppRegistry} from 'react-native';  
import React from "react";  
import { SafeAreaView, View, FlatList, StyleSheet, Text, StatusBar, Image } from 'react-native';
```

12. Dentro de nuestro proyecto deberemos crear la carpeta src, y dentro de la carpeta src una carpeta imgs. Se deberá de ver la estructura de nuestro proyecto de la siguiente forma:



13. Dentro de la carpeta imgs, pondremos las imágenes proporcionadas en recursos de esta guía.

14. Ahora vamos a modificar nuestro código para agregar las imágenes.

- a. Primero en nuestro DATA vamos ir agregando la propiedad src, para cada uno de los elementos:

```
const DATA = [
  {
    id: '1',
    title: 'Toyota',
    src: require('./src/imgs/toyota.jpg'),
  },
  {
    id: '2',
    title: 'Mazda',
    src: require('./src/imgs/mazda.jpg'),
  },
  {
    id: '3',
    title: 'Mitsubishi',
    src: require('./src/imgs/mitsubishi.jpeg'),
  },
];
```

- b. vamos a modificar nuestra constante Item, y deberá quedarnos de la siguiente manera:

```
const Item = ({ title, img }) => (
  <View style={styles.item}>
    <Text style={styles.title}>{title}</Text>
    <Image style={styles.img} source={img} />
  </View>
);
```

- c. para poder apreciar los cambios debemos realizar un cambio dentro de nuestro render Item:

```
const App = () => {
  const renderItem = ({ item }) => (
    <Item title={item.title} img={item.src} />
  );
};
```

- d. Ahora solo nos queda modificar los estilos:

```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    marginTop: StatusBar.currentHeight || 0,
  },
  item: {
    backgroundColor: '#f9c2ff',
    padding: 20,
    marginVertical: 8,
    marginHorizontal: 16,
    alignItems: 'center',
  },
  title: {
    fontSize: 32,
  },
  img: {
    width: 200,
    height: 125,
    borderWidth: 2,
    borderColor: '#d35647',
    resizeMode: 'contain',
    margin: 8,
  },
});
```

15. Procedemos a verificar los cambios dentro de nuestro emulador el cual debería verse de la siguiente manera:



PARTE 2

1. Procederemos a crear nuestra versión apk.
2. En primer lugar, asegúrese de que su proyecto de Android esté libre de errores. Eso significa que se está compilando y ejecutándose correctamente en el emulador o en un dispositivo Android.
3. Necesitará una clave de firma generada por Java, que es un archivo de almacén de claves que se utiliza para generar un binario ejecutable React Native para Android.

Puede crear uno usando la herramienta de teclas en la terminal con el siguiente comando

```
"%JAVA_HOME%\bin\keytool" -genkey -v -keystore my-upload-key.keystore -alias my-key-alias -keyalg RSA -keysize 2048 -validity 10000
```

Puede cambiar **my-upload-key** con el nombre que desee, así como my-key-alias . Esta clave usa el tamaño de clave 2048, en lugar del predeterminado 1024 por razones de seguridad.

Por lo tanto, este comando le solicita la contraseña del almacén de claves, la clave real y los campos de nombre distinguido para su clave. Por lo tanto, todo debe ingresarse manualmente y con cuidado.

Ingrese su contraseña del almacén de claves: **Contrasenia123**

Vuelva a ingresar la nueva contraseña: **Contrasenia123**

¿Cuál es su nombre y apellido? [desconocido]: **Su Nombre**

¿Cuál es el nombre de tu unidad organizacional? [desconocido]: **udb**

¿Cuál es el nombre de su organizacion? [desconocido]: **udb**

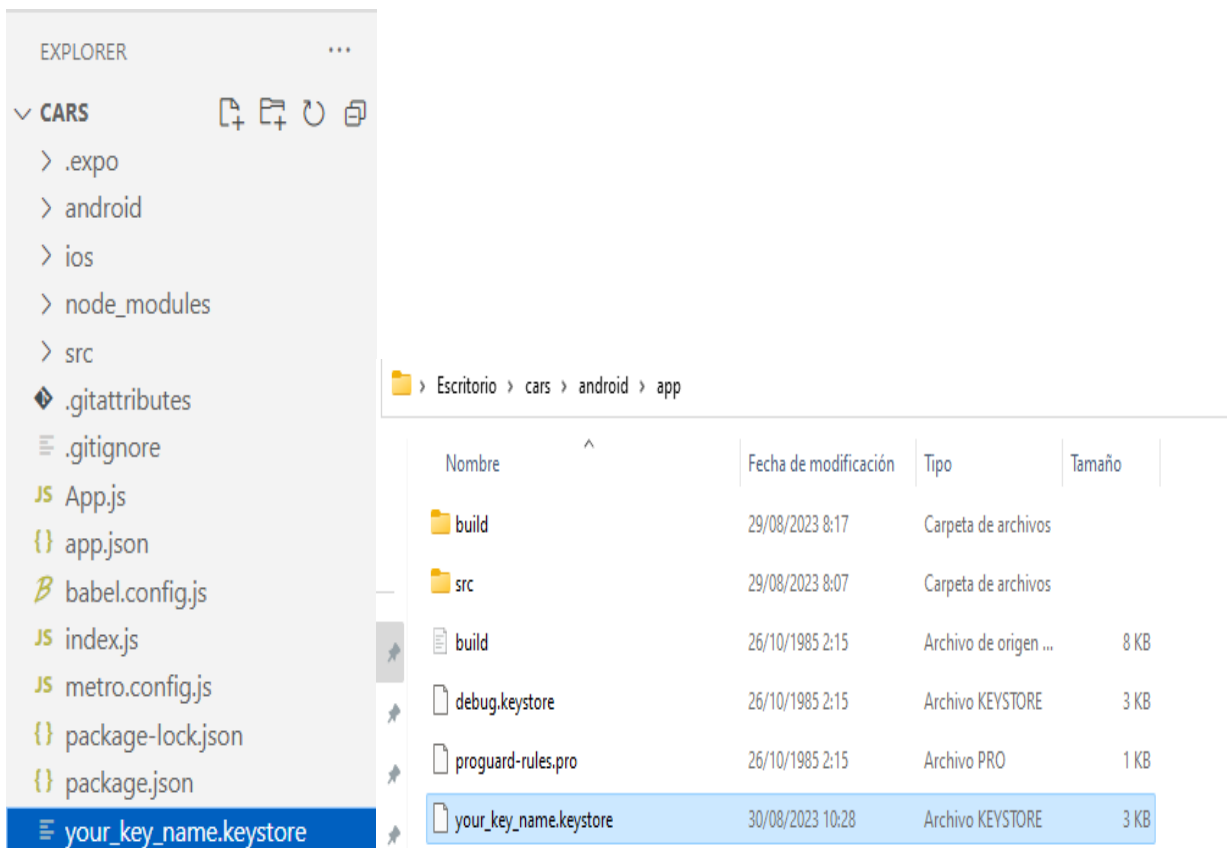
¿Cuál es el nombre de su ciudad o localidad? [desconocido]: **San Salvador**

¿Cuál es el nombre de su estado o provincia? [desconocido]: **San Salvador**

¿Cuál es el código de país de dos letras para esta unidad? [desconocido]: **sv**

NOTA: Como resultado, genera un archivo de almacén de claves en el directorio de su proyecto llamado my-upload-key.keystore válido por 10000 días. Lo más importante es hacer una copia de seguridad de este archivo de almacén de claves y sus credenciales (contraseña de almacenamiento, alias y contraseña de alias) que se solicitarán más adelante.

4. Ahora debe copiar el archivo my-upload-key.keystore y pegarlo en el directorio android / app en la carpeta de su proyecto React Native.



5. Debe abrir su archivo **android\app\build.gradle** y agregar la configuración del almacén de claves. Hay dos formas de configurar el proyecto con **keystore**.

```
android {  
  ....  
  signingConfigs {  
    release {  
      storeFile file('my-upload-key.keystore')  
      storePassword 'Contrasenia123'  
      keyAlias 'my-key-alias'  
      keyPassword 'Contrasenia123'  
    }  
  }  
  buildTypes {  
    release {  
      ....  
      signingConfig signingConfigs.release  
    }  
  }  
}
```

6. Ahora ejecutamos en la terminal
- ```
react-native bundle --platform android --dev false --entry-file index.js --bundle-output
android/app/src/main/assets/index.android.bundle
--assets-dest android/app/src/main/res/
```
7. Coloque su directorio de terminal Android usando
- ```
cd android
```
- Windows:
- ```
gradlew assembleRelease --no-daemon
```
- Linux y Mac OSX
- ```
./gradlew assembleRelease --no-daemon
```
8. Como resultado, **se realiza el proceso de creación de APK**. Puedes encontrar el **APK** generado en **android/app/build/outputs/apk/app-release.apk** Esta es la aplicación real, que puede enviar a su teléfono o subir en Google Play Store.

Felicitaciones, acaba de generar un APK de compilación de lanzamiento nativo de React para Android.

V. EJERCICIO COMPLEMENTARIO

Realizar una app de comidas típicas salvadoreñas donde se muestre, una fotografía de la comida típica, el nombre de la comida y la cantidad calórica de cada platillo típico, para realizar la aplicación deberá utilizar los elementos card que se encuentra en [react-native-elements](#):

