React Native

Setup development environment

Estas instrucciones son para **instalar y configurar** el entorno de desarrollo en **linux**.
En linux no es posible construir la aplicación para iOS, **sólo para android.**
Cualquier editor te texto sirve para desarrollar con RN, no es necesario Android Studio.

Pre-requisitos

- Node.js 4 o superor

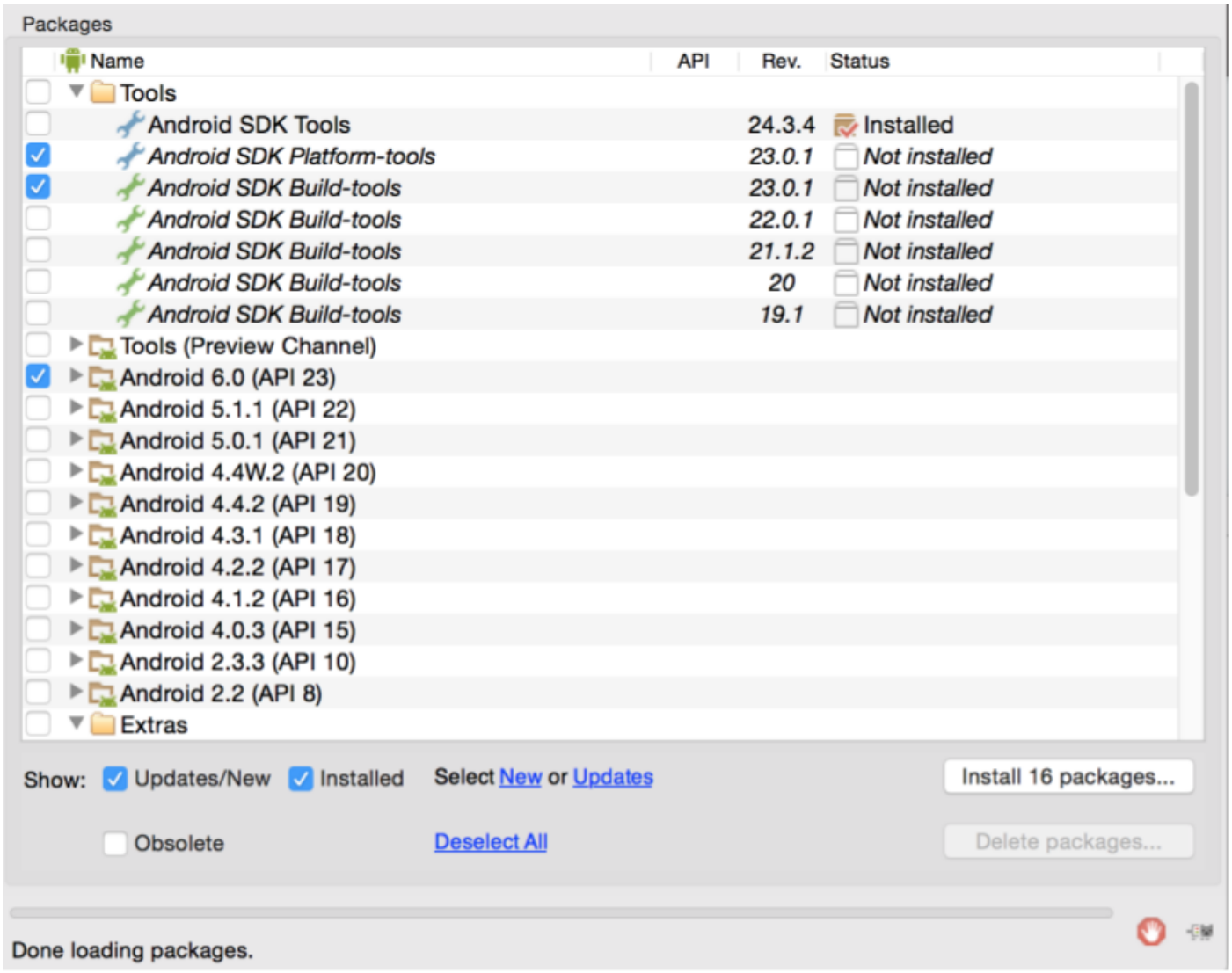
Android SDK

Estos son los paquetes **necesarios** a instalar en el sdk manager:

- Android SDK Platform-tools Rev. 23.0.1
- Android SDK Build-tools Rev. 23.0.1
- Android 6.0 (API 23)
    - SDK Platform

    - Google APIs
- Extras
    - Android Support Repository
    - Google Repository*

* Opcional, pero requerido por algunos modulos extras de RN.(Recomendado)



La guía oficial de RN instala **librerías extra** como las imágenes para la emulación de android en el **emulador de android studio**, pero sólo son necesarias si se hará uso de éste.

Variables de entorno ANDROID_HOME

La variable de entorno ANDROID_HOME es requerida por RN.
También es necesario agregar las herramientas de android sdk al PATH.

```
export ANDROID_HOME=${HOME}/Android/Sdk
export PATH=${PATH}:${ANDROID_HOME}/tools
export PATH=${PATH}:${ANDROID_HOME}/platform-tools
```

Watchman

Opcional pero recomendado, acelera la construcción de la aplicación.

> You can use these steps below to get watchman built. You will need autoconf and automake. You may optionally build watchman without pcre and python support.

```
$ git clone https://github.com/facebook/watchman.git
$ cd watchman
$ git checkout v4.7.0  # the latest stable release
$ ./autogen.sh
$ ./configure
$ make
$ sudo make install
```

Guía completa de instalación Watchman

Inotify Watchers

Será necesario **aumentar la cantidad de *inotify watchers***

Para Debian, Redhat o similar:

`$ echo fs.inotify.max_user_watches=524288 | sudo tee -a /etc/sysctl.conf && sudo sysctl -p`

Archlinux:

`$ echo fs.inotify.max_user_watches=524288 | sudo tee /etc/sysctl.d/40-max-user-watches.conf && sudo sysctl --system`

Emulador o configuración de dispositivo

Dispositivo android

La mejor opción, por comodidad y eficiencia, es definitivamente un **dispositvo con android**.

En el dispositivo android el **modo desarrollador** y la **depuración USB** deben estar activados.

Emulador

El emulador de android studio tiende a ser muy lento, se recomienda la instalación de **Genymotion**.

Instalación de RN CLI

Basta con instalarla desde NPM

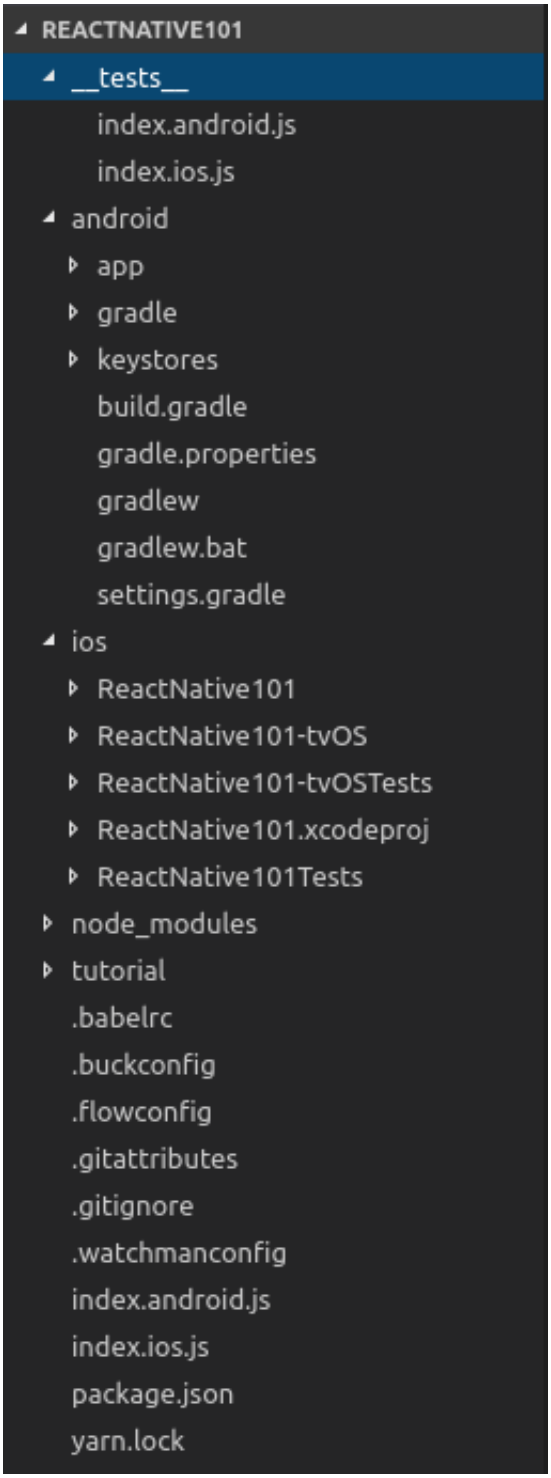
```
$ npm install -g react-native-cli
```

Para iniciar un nuevo proyecto:

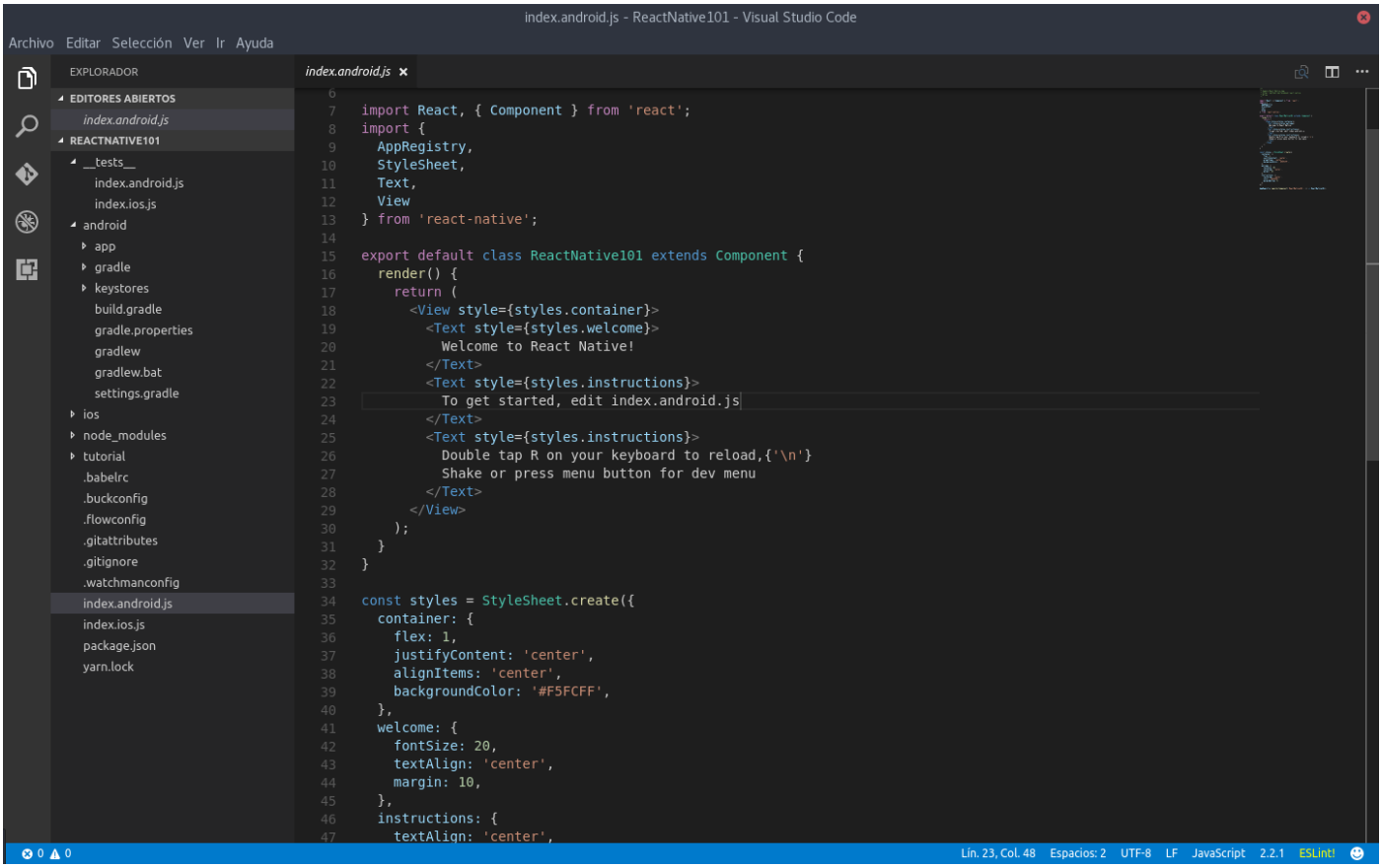
```
$ react-native init NuevoProyecto
```

A partir de react-native-cli v1.2.0 instalará las dependencias mediante yarn.

Al iniciar un nuevo proyecto se creará la siguiente estructura de directorios y archivos ademas de instalar las dependencias necesarias:



Los archivos **index.android.js** y **index.ios.js** son los *main* que se utilizarán para construir la aplicación para cada plataforma:



Package.json:

package.json

```
{
  "name": "ReactNative101",
  "version": "0.0.1",
  "private": true,
  "scripts": {
    "start": "node node_modules/react-native/local-cli/cli.js start",
    "test": "jest"
  },
  "dependencies": {
    "react": "~15.4.1",
    "react-native": "0.42.0"
  },
  "devDependencies": {
    "babel-jest": "19.0.0",
    "babel-preset-react-native": "1.9.1",
    "jest": "19.0.2",
    "react-test-renderer": "~15.4.1"
  },
  "jest": {
    "preset": "react-native"
  }
}
```

Probar la instalación en el emulador o dispositivo

El emulador debe estar en ejecución, y en el caso del dispositivo debe estar conectado mediante USB.

Para verificar que hay dispositivos disponibles, se puede utilizar la herramienta `$ adb devices`

```
$ adb devices
List of devices attached
emulator-5554 offline   # Google emulator
14ed2fcc device         # Physical device
```

Para construir la aplicación e instalarla en el dispositivo:

```
$ react-native run-android
```

Si todo va bien, el dispositivo debe ejecutar la aplicación base de react native:

En la terminal debe estar ejecutándose el packager:

```
================================================================
|  Running packager on port 8081.
|  Keep this packager running while developing on any JS
|  projects. Feel free to close this tab and run your own
|  packager instance if you prefer.
|
|     https://github.com/facebook/react-native
|
================================================================
```

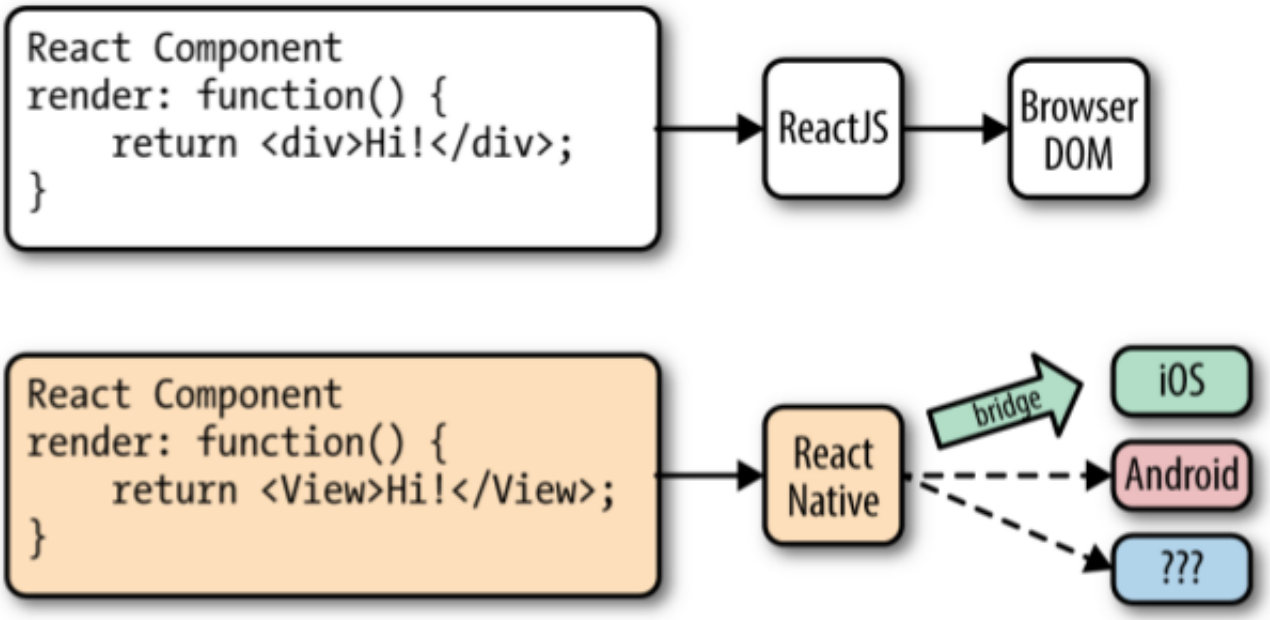
Algunas veces el packager crashea y es necesario iniciarlo nuevamente con `$ npm start`

Desarrollando con React Native

Funcionamiento de React Native

En React, el Virtual DOM actua como una capa entre la **descripción del desarrollador de como deberían verse las cosas**, y el procesamiento para renderizar la aplicación.

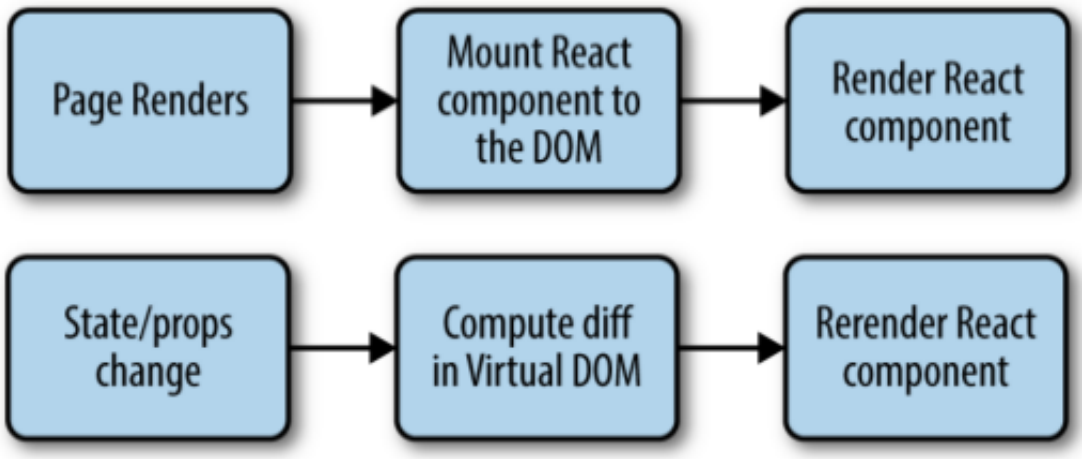
En el contexto Web, el Virtual Dom se ve primeramente como una optimización y sí que lo es; pero dado que **React "entiende" como debe verse la aplicación**, en lugar de renderizar al DOM del navegador, **React Native invoca APIs de Objective-C/Java para renderear componentes nativos** en iOS/Android.



Los componentes de React retornan *markup* de su función render, que describe como deben verse los componentes. En React Web esto se traduce directamente al DOM del navegador, y **en React Native se traduce para ajustarse a la plataforma adecuada, así un se convierte en UIView de iOS**.

Lifecyle

Es ciclo de vida de los componentes es **básicamente el mismo que React para Web**.



El procesamiento es distinto ya que **React Native depende del *bridge***, que se encarga de traducir javascript y hacer las llamadas correspondientes a la plataforma host para renderear los componentes.

Creando componentes

Los componentes de React Native son mayormente como los de React, pero **existen algunas diferencias importantes**

para el renderizado y estilo.

Views

En React para Web se renderizan componentes normales de HTML (, <p>, ,<a>, etc.) estos elementos son **remplazados por componentes de React específicos para cada plataforma**. El componente **<View> es el componente mas básico multiplataforma** en React Native, es el análogo a .

React Native provee de elementos básicos multiplataforma similares a los de Web:

React	React Native
	<View>
	<Text>
,	<ListView>
<img>	<Image>

Y también provee de elementos que son específicos para cada plataforma:

Componentes Nativos iOS
Componentes Nativos Android

Dado que los elementos de la UI son componentes de React, es necesario importarlos explícitamente de React Native.

```
import React, { Component } from 'react';
import {
  StyleSheet,
  Text,
  View
} from 'react-native';
```

Los componentes varian de plataforma a plataforma, por esto, **la manera en que se estrucutran los componentes es muy importante en RN**.

Si se planea reutilizar código en RN, mantener una separación entre este tipo de componentes es algo crítico. Por eso **es importante que los componentes de React encapsulen la lógica**, para que sea fácil asociar un componente con una vista de acuerdo a la plataforma.

Estilos

Al trabajar con React Native se utiliza un estándar para el estilo que es **un subconjunto de CSS**, basado

principalmente en flexbox para el diseño, y **se centra en la simplicidad en lugar de implementar la gama completa de reglas CSS**.

RN insiste en el uso de estilos *inline* trabajando con objetos en lugar de *stylesheets*:

```
// Definimos un estilo...
var style = {
backgroundColor: 'white',
fontSize: '16px'
};
// ...y lo aplicamos.
var texto = (
<Text style={style}>
Texto con estilos
</Text>);
```

Esta aplicación ejemplo de React Native permite explorar los componentes y estilos soportador por RN:
React Native UIExplorer

APIs para cada plataforma

Las APIs para cada plataforma permiten que las aplicaciones tengan **una experiencia de usuario mucho mas natural**. Incluyen todo desde **almacenamiento de datos, servicios de ubicación, acceder al hardware como la cámara, etc.** conforme mas avanza el desarrollo de RN se crean mas y mas APIs.

APIS

ActionSheetIOS
AdSupportIOS
Alert
AlertIOS
Animated
AppRegistry
AppState
AsyncStorage
BackAndroid
CameraRoll
Clipboard
DatePickerAndroid
Dimensions
Easing
Geolocation
ImageEditor
ImagePickerIOS
ImageStore
InteractionManager
Keyboard
LayoutAnimation
Linking
NativeMethodsMixin
NetInfo
PanResponder
PermissionsAndroid
PixelRatio
PushNotificationIOS