

 04

Preparando o ambiente: Linux

Para conseguir criar projetos, desenvolver nossas apps e testar seus comportamentos precisamos primeiramente entender quais softwares e ferramentas são necessários para configurar corretamente nosso ambiente de desenvolvimento.

Mãos à obra

A preparação do ambiente de desenvolvimento para o React Native além de levar em conta as diferenças entre os sistemas operacionais da máquina de desenvolvimento, também precisa se ater a detalhes de cada uma das plataformas do mundo mobile onde queremos testar e implantar nossas apps (Android/IOS).

Por esse motivo, nossa etapa de preparação do ambiente está dividida primeiramente por plataforma da máquina de desenvolvimento, seguindo para o ambiente das plataformas mobile.

Ubuntu Linux

IMPORTANTE: Usando sistemas Linux não é possível desenvolver apps com código nativo para a plataforma IOS.

Preparando ambiente para desenvolvimento Android com React Native

Vamos começar instalando todos os recursos necessários para rodar apps Android utilizando o React Native. Todos os passos dessa configuração estão disponíveis na documentação do React Native, que pode ser acessada através [deste link \(https://facebook.github.io/React-native/docs/getting-started.html\)](https://facebook.github.io/React-native/docs/getting-started.html).

Node.js

Para começar precisaremos contar com o runtime do [Node.js](#) (<https://nodejs.org/en/>) para executar nosso código JavaScript. Podemos instalá-lo facilmente usando o apt-get no Ubuntu:

```
curl -sL https://deb.nodesource.com/setup_6.x | sudo -E bash -  
sudo apt-get install -y nodejs
```

[COPIAR CÓDIGO](#)

Caso já tenha o node instalado em sua máquina certifique-se de que a versão instalada é a 4 ou mais recente, como recomendado na própria documentação. Recomendamos o uso da versão 6.x.x, para já podermos contar com algumas das features do ES6, por exemplo.

Caso utilize outra distribuição Linux, siga as [instruções de instalação do Node Js](#) (<https://nodejs.org/en/download/package-manager/>) mais específicas para o seu ambiente

React Native

Atualmente precisamos fazer a instalação de nenhum CLI global para executarmos comandos React-native, isso porque a equipe do React criou uma lib executável através do npx, que vem instalado com o Node. Nesse caso só precisamos chamar os comandos do React Native com o npx.

```
npx React-native init  
npx React-native start  
npx React-native run-ios  
npx React-native run-android
```

[COPIAR CÓDIGO](#)

Caso você queira instalar o CLI do React-native você pode faze-lo com o seguinte comando `npm install -g React-native-cli`. Nesse caso, basta não colocar `npx` em frente aos comandos que quer executar.

Recomenda-se a utilização do Node Package Manager na versão 4. Você também pode (e é recomendável) utilizar a ferramenta [Yarn](https://yarnpkg.com/) (<https://yarnpkg.com/>). Yarn é um gerenciador de pacotes criado também pelo Facebook que já conta com uma série de otimizações para facilitar o gerenciamento das dependências nos seus projetos que usam ferramentas da própria empresa como React, React Native, Jest, Watchman, etc. Para instalar o Yarn utilizando o Homebrew digite o seguinte comando no terminal: `brew install yarn`

Java

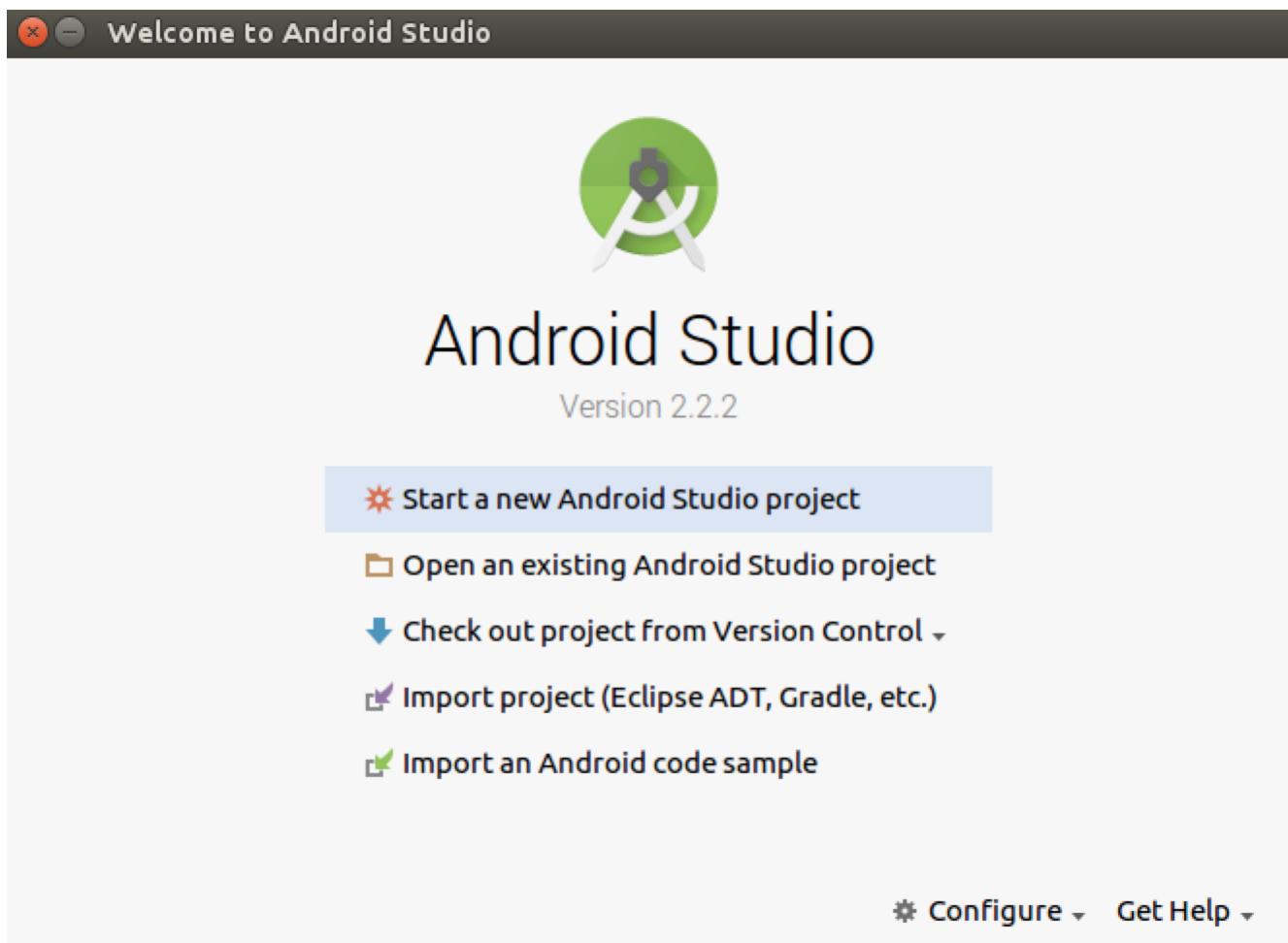
Para podermos rodar nossa app no Android precisaremos do Java Development Kit (JDK) na versão 8 ou superior. Você pode baixar o JDK [aqui](http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html) (<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>).

Android Studio

Seguindo em frente precisaremos também do ambiente de desenvolvimento Android configurado, portanto, vamos baixar e instalar também a ferramenta Android Studio e as SDK Tools. [Baixe e instale o Android Studio](https://developer.android.com/studio/index.html) (<https://developer.android.com/studio/index.html>), selecione "Custom" quando perguntado sobre o tipo de instalação desejado e certifique-se de marcar as seguintes opções no instalador da ferramenta antes de clicar em "Next" e instalar efetivamente os componentes:

- Android SDK
- Android SDK Platform
- Android Virtual Device

Após a instalação, será exibida uma tela de boas vindas do Android Studio.



O Android Studio já instala por padrão a última versão do SDK do Android, porém para desenvolver apps nativas para o Android com React Native, precisaremos instalar adicionalmente o SDK na versão Android 6.0 (Marshmallow). Podemos adicionar SDKs adicionais no SDK Manager do Android Studio. Para acessá-lo, clique em "Configure" na tela de boas vindas, e então selecione "SDK Manager".

Selecione a aba "SDK Platforms" e marque o checkbox "Show Package Details" no canto inferior direito. Abra a seção "Android 6.0 (Marshmallow)", e certifique-se de selecionar os seguintes items:

- Google APIs
- Android SDK Platform 23
- Intel x86 Atom_64 System Image
- Google APIs Intel x86 Atom_64 System Image

The screenshot shows the 'SDK Platforms' tab selected in the Android SDK Manager. It lists various packages for different API levels. The packages are categorized by platform version. The 'Show Package Details' checkbox is checked at the bottom right.

Name	API L...	Revis...	Status
Google APIs Intel x86 Atom_64 System Image	24	20	Not installed
Google Play Intel x86 Atom System Image	24	19	Not installed
Android 6.0 (Marshmallow)			
Google APIs	23	1	Not installed
Android SDK Platform 23	23	3	Not installed
Sources for Android 23	23	1	Not installed
Android TV ARM EABI v7a System Image	23	12	Not installed
Android TV Intel x86 Atom System Image	23	14	Not installed
Android Wear ARM EABI v7a System Image	23	6	Not installed
Android Wear Intel x86 Atom System Image	23	6	Not installed
Intel x86 Atom System Image	23	10	Not installed
Intel x86 Atom_64 System Image	23	10	Not installed
Google APIs ARM EABI v7a System Image	23	26	Not installed
Google APIs Intel x86 Atom System Image	23	17	Update Available: ...
Google APIs Intel x86 Atom_64 System Image	23	26	Not installed
Android 5.1 (Lollipop)			
Google APIs	22	1	Not installed
Android SDK Platform 22	22	2	Not installed

Agora selecione a aba "SDK Tools" e marque o checkbox "Show Package Details" no canto inferior direito. Abra a seção "Android SDK Build-Tools" e selecione a opção "23.0.1".

Appearance & Behavior > System Settings > Android SDK

Manager for the Android SDK and Tools used by Android Studio

Android SDK Location: [Edit](#)

[SDK Platforms](#) [SDK Tools](#) [SDK Update Sites](#)

Below are the available SDK developer tools. Once installed, Android Studio will automatically check for updates. Check "show package details" to display available versions of an SDK Tool.

Name	Version	Status
Android SDK Build-Tools		
17.0.0 (Obsolete)	17.0.0	Not installed
18.0.1 (Obsolete)	18.0.1	Not installed
18.1.0 (Obsolete)	18.1.0	Not installed
18.1.1 (Obsolete)	18.1.1	Not installed
19.0.0 (Obsolete)	19.0.0	Not installed
19.0.1 (Obsolete)	19.0.1	Not installed
19.0.2 (Obsolete)	19.0.2	Not installed
19.0.3 (Obsolete)	19.0.3	Not installed
19.1.0	19.1.0	Not installed
20.0.0	20.0.0	Not installed
21.0.0 (Obsolete)	21.0.0	Not installed
21.0.1 (Obsolete)	21.0.1	Not installed
21.0.2 (Obsolete)	21.0.2	Not installed
21.1.0 (Obsolete)	21.1.0	Not installed
21.1.1 (Obsolete)	21.1.1	Not installed
21.1.2	21.1.2	Not installed
22.0.0 (Obsolete)	22.0.0	Not installed
22.0.1	22.0.1	Not installed
23.0.0 (Obsolete)	23.0.0	Not installed
23.0.1	23.0.1	Not installed

Show Package Details

Por fim, clique em "Apply" para baixar e instalar o SDK e as Build Tools.

Agora precisaremos configurar a variável de ambiente ANDROID_HOME, para que o ambiente do React Native consiga enxergar o SDK do Android no momento de instalar e rodar nossas apps no Android.

Adicione as seguintes linhas ao seu arquivo de configuração bash

\$HOME/.bash_profile:

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

[COPIAR CÓDIGO](#)

Adicionalmente, rode o seguinte comando para recarregar as configurações no terminal: source \$HOME/.bash_profile. Você pode também verificar se o valor de ANDROID_HOME foi adicionado corretamente à variável de ambiente PATH executando: echo \$PATH.

Preparando um emulador Android

Um último passo importante é que precisamos preparar um Android Virtual Device (AVD) para podermos testar nossas aplicações. Você pode ver a lista com os emuladores configurados acessando o "AVD Manager" do Android Studio. Procure por um ícone como o que segue na barra de ferramentas do Android Studio:



Selecione "Create Virtual Device", escolha um modelo de dispositivo disponível (Nexus 5X, por exemplo) e clique em "Next". Selecione a aba "x86 Images", e então procure por "Marshmallow API Level 23, x86_64 ABI image" - com "Android 6.0 (Google APIs)".

Release Name	API Level	ABI	Target
Nougat Download	25	x86	Android 7.1.1 (Google APIs)
Nougat Download	25	x86_64	Android 7.1.1 (Google APIs)
Nougat Download	24	x86	Android 7.0 (Google APIs)
Nougat Download	24	x86_64	Android 7.0 (Google APIs)
Nougat Download	24	x86_64	Android 7.0
Nougat Download	24	x86	Android 7.0
Marshmallow	23	x86_64	Android 6.0 (Google APIs)
Marshmallow Download	23	x86	Android 6.0 (Google APIs)
Marshmallow	23	x86_64	Android 6.0
Marshmallow Download	23	x86	Android 6.0
Lollipop Download	22	x86	Android 5.1 (Google APIs)
Lollipop Download	22	x86_64	Android 5.1 (Google APIs)
Lollipop Download	22	x86_64	Android 5.1
Lollipop Download	22	x86	Android 5.1
Lollipop Download	21	x86_64	Android 5.0 (Google APIs)
...

Marshmallow

API Level
23

Android
6.0

Google Inc.

System Image
x86_64

Questions on API level?
See the [API level distribution chart](#)

Cancel Previous Next Finish

Clique em "Next" e "Finish" para criar seu AVD. Nesse ponto já é possível verificar o novo dispositivo adicionado à lista e inclusive selecionar o botão de play verde para emular o dispositivo.

Pronto =)