Ambiente de desenvolvimento no React Native

Para testar nossas aplicações em dispositivos reais iremos utilizar **emuladores** que são apps que imitam o funcionamento de um dispositivo físico dentro do nosso sistema. Infelizmente no **Windows** e no **Linux** ainda não é possível configurar um emulador de **iOS** de forma nativa e, por isso, vamos focar no **Android** nessas duas plataformas.

Editor

O editor de código que utilizo para construção das minhas apps mobile é o **Visual Studio Code** da Microsoft. Com ele tenho uma interface muito semelhante aos editores **Sublime Text** e **Atom** enquanto mantenho recursos como autocomplete, debugging e outras features presentes apenas em IDE's.

O VSCode **não é obrigatório** e você pode utilizar o que preferir. Dentre os que eu recomendo estão: VSCode, Sublime Text, Atom e Vim.

Android

Para termos um ambiente de desenvolvimento para Android iremos instalar a **SDK** do mesmo com ferramentas adicionais para executarmos nossa app em um emulador. Da mesma forma, ao fim desse guia você também poderá emular sua aplicação em seu dispositivo físico utilizando **USB**.

O ambiente Android pode ser configurado no **Mac OSX, Linux e Windows**, siga o guia abaixo de acordo com seu sistema operacional.

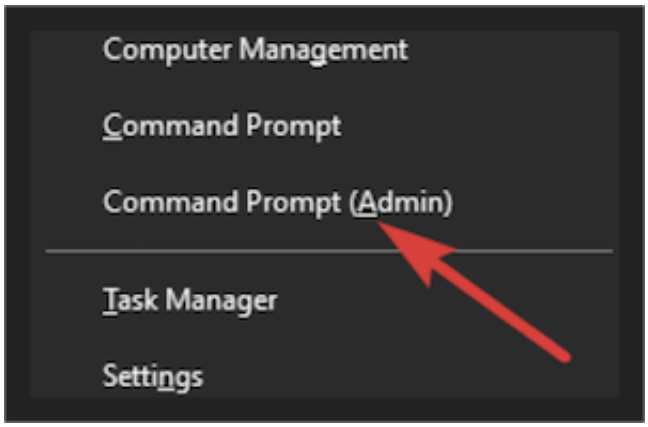
Windows

Para configurar o ambiente Android no Windows, vamos precisar instalar 4 dependências: **Node, Python2, JDK e Android Studio**.

Instalando chocolatey

Para instalar as libs no Windows, vamos utilizar um **gerenciador de pacotes** do Windows chamado Chocolatey. Essa ferramenta nos possibilita instalar **dependências** e ferramentas no sistema com poucos comandos e tudo pelo **terminal**.

Execute o prompt de comando ou powershell como administrador utilizando a tecla "**Windows + X**" ou clicando com o botão direito sobre o botão "Iniciar":



Agora, você deve executar os comandos abaixo de acordo com **a opção que você clicou**:

Command Prompt

Caso a opção que você tenha clicado tenha o nome de **Command Prompt**, execute o comando abaixo na janela aberta para instalar o Chocolatey:

```
@"%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile -InputFormat None -ExecutionPolicy Bypass -Command "iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))" && SET "PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin"
```

Powershell

Caso a opção tenha sido **Powershell**, execute o comando abaixo para verificar se você possui permissões para instalar dependências com o terminal:

```
Get-ExecutionPolicy
```

Se o retorno desse comando for "**Restricted**", execute o próximo comando em seu terminal, se não, prossiga para o **próximo passo**:

```
Set-ExecutionPolicy AllSigned
```

Agora, execute o seguinte comando para instalar o Chocolatey:

```
Set-ExecutionPolicy Bypass -Scope Process -Force; iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))
```

Agora, **teste** se a instalação ocorreu corretamente executando o seguinte comando no seu terminal (nada irá acontecer, mas não deve retornar erros). Nesse passo pode ser necessário **reiniciar seu terminal**.

```
choco
```

Agora vamos instalar o **Node**, **Python2** e a **JDK8** (Java Development Kit 8).

```
choco install -y nodejs.install python2 jdk8
```

*Se você tiver o **NodeJS** já instalado em sua máquina, certifique-se que sua versão é superior à 4 e caso esteja com o **JDK** instalado em sua máquina, certifique-se que sua versão é superior à 8.*

Agora com as dependências instaladas, vamos instalar o **CLI (Command Line Interface) do React Native** que nos ajudará na criação e teste de novos projetos. Nesse passo você provavelmente deve **reiniciar seu terminal** para o comando funcionar.

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

Se tudo ocorreu bem até aqui, você conseguirá executar o seguinte comando:

```
react-native -h
```

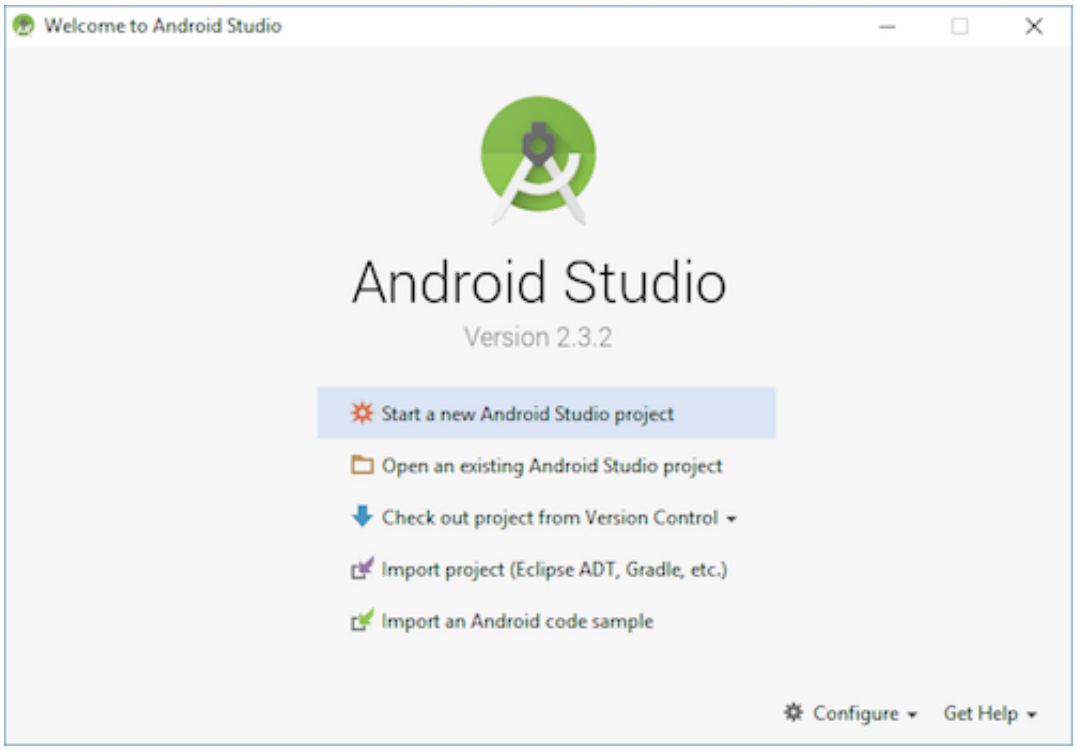
Esse comando deve exibir uma **lista de comandos** possíveis para serem executados com o React Native CLI.

Android Studio

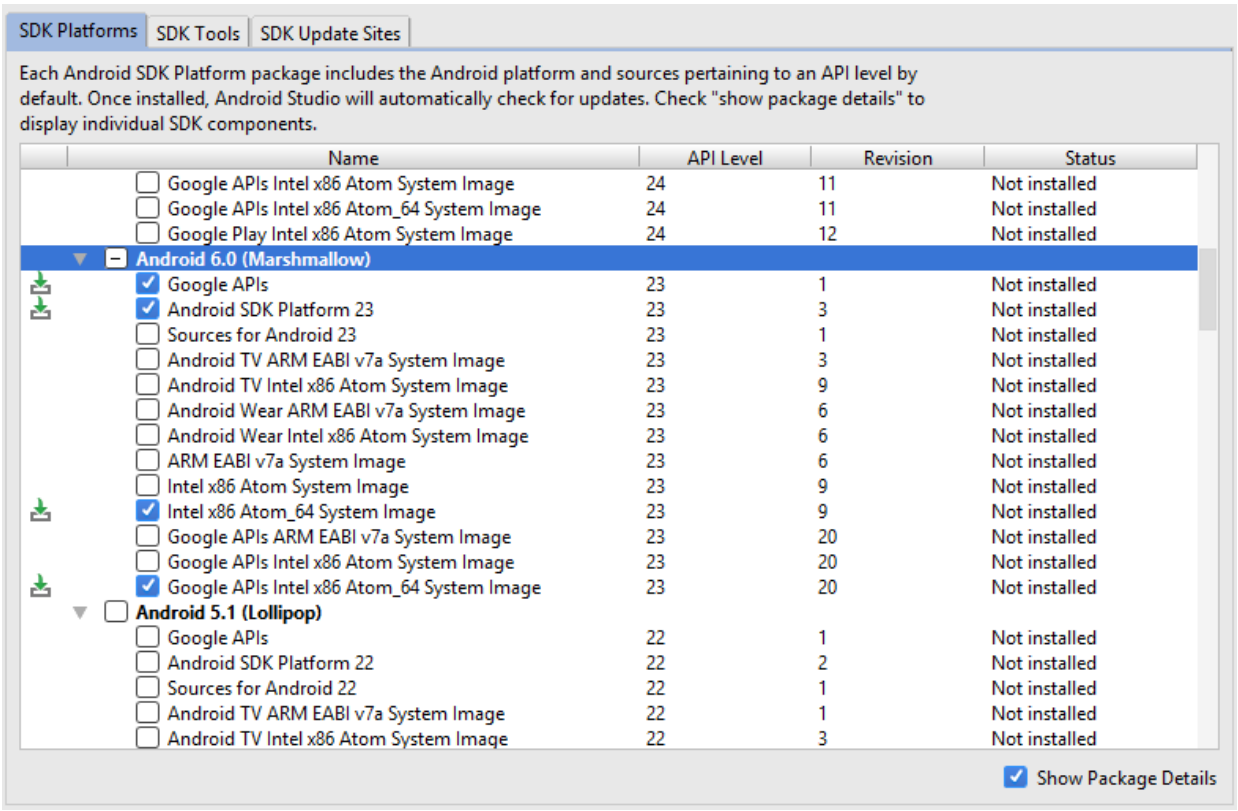
Além do Chocolatey vamos precisar instalar o Android Studio em nossa máquina. Não utilizaremos ele diretamente para programar mas sim para instalar a **SDK** e todas dependências de desenvolvimento do Android.

Comece instalando o Android Studio através do link: https://developer.android.com/studio/index.html e seguindo os passos descritos no arquivo executável.

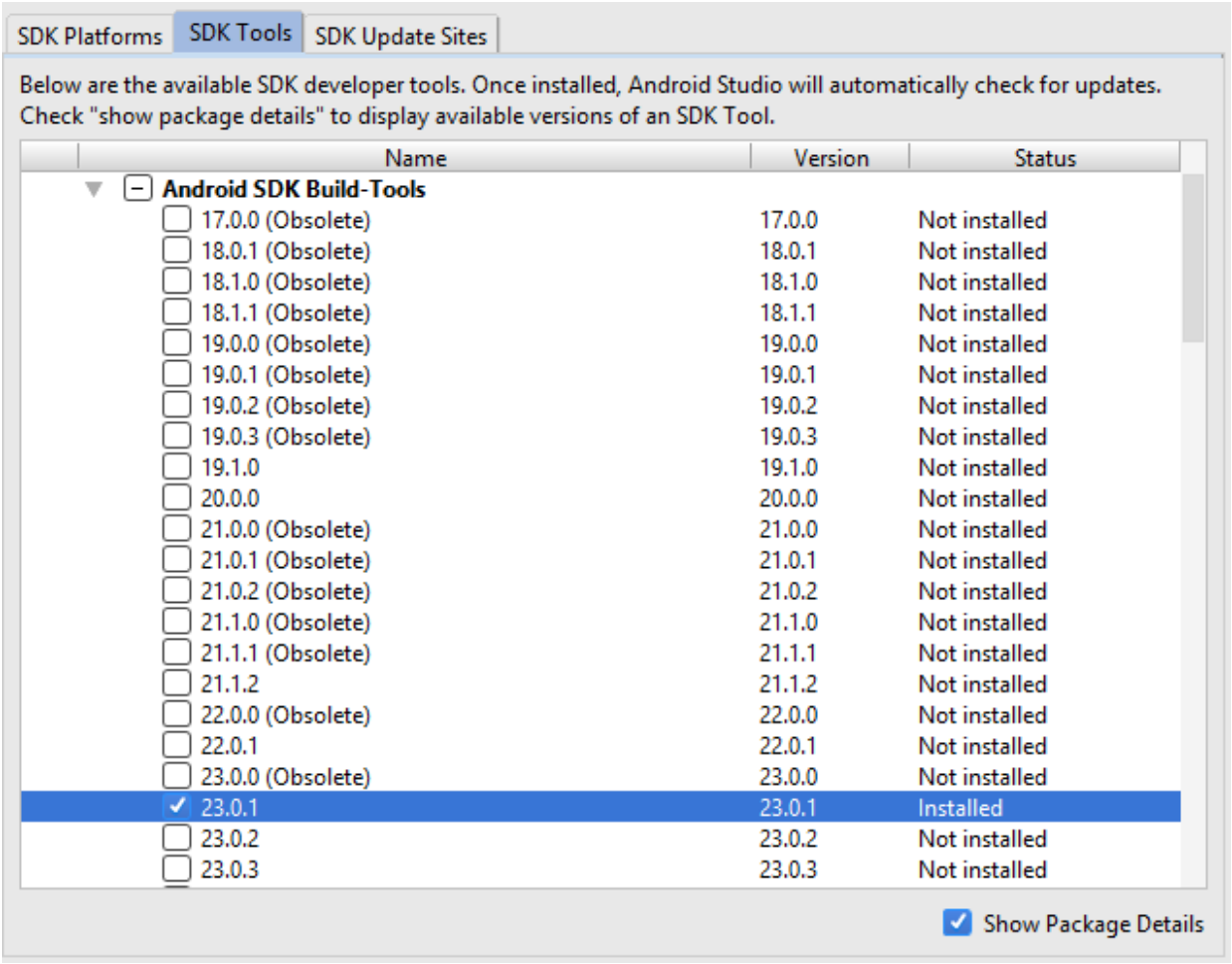
Se a instalação ocorrer bem, você receberá uma tela como a seguir:



Agora, nessa tela, clique no botão “Configure” e selecione a opção “SDK Manager”. Nessa tela selecione a opção "**Show Package Details**" no rodapé da janela e instale as dependências mostradas dentro da **versão 6.0 do Android**:

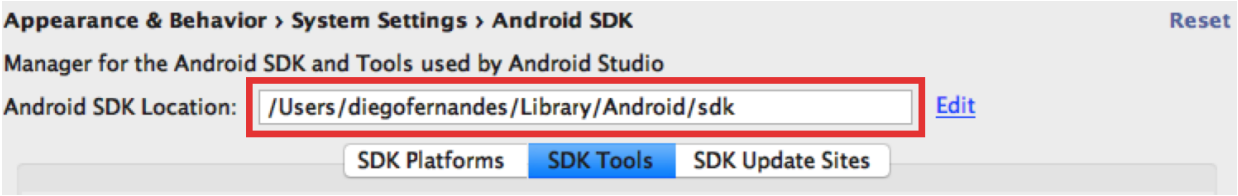


Agora na aba "**SDK Tools**" mantenha a opção "**Show Package Details**" selecionada e abrindo o item "**Android SDK Build-Tools**" selecione a opção "**23.0.1**":

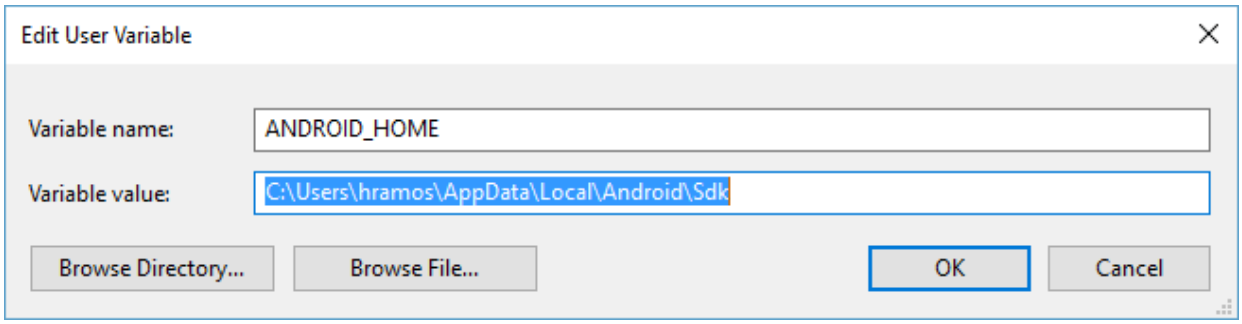


Configurando ANDROID_HOME

Agora que temos nossa SDK instalada com sucesso, precisamos configurar a variável ambiente ANDROID_HOME em nossosistema. Para isso, no SDK Manager do Android Studio, **copie o caminho da SDK do Android** como na imagem:



Agora, no **Painel de Controle do Windows**, abra o item “Sistema e Segurança” ou “Sistema”, clique em “Configurações avançadas do sistema”, selecione “Variáveis de ambiente” e clique no botão “**Nova variável de ambiente**”, indique o nome da variável como **ANDROID_HOME**, adicione o **caminho copiado do Android Studio** como segundo parâmetro e clique em OK.



Agora com a API configurada vamos **adicionar duas novas pastas** à nossa variável de ambiente **PATH** para conseguirmos chamar as ferramentas do Android diretamente pelo terminal.

Para isso, na mesma janela de "Variáveis de ambiente" no Windows, clique na variável PATH e então em "Editar". Haverá uma lista de caminhos e você deve adicionar esses dois novos caminhos no fim da lista:

```
%ANDROID_HOME%/platform-tools
%ANDROID_HOME%/tools
```

Agora que configuramos a SDK do Android será necessário configurar o **emulador do Android**. Para emular nossa aplicação vou utilizar uma aplicação chamada **Genymotion** já que o emulador que vem instalado com a SDK do Android é mais lento.

Para criar um projeto em React Native utilize o comando:

```
react-native init NomeDoProjeto
```

Para configurar o Genymotion siga para a seção "**Configurando Genymotion**".

Linux

Para configurar o ambiente Android no Linux, vamos precisar instalar 3 dependências: **Node, JDK e Android Studio**.

Certifique-se que você tenha o cURL instalado executando o seguinte comando no terminal:

```
sudo apt-get install curl
```

Agora com o cURL instalado, vamos instalar no **NodeJS** utilizando os seguintes comandos:

```
curl -sL https://deb.nodesource.com/setup_9.x | sudo -E bash -
sudo apt-get install -y nodejs
```

*Caso você não esteja em distribuições **Debian/Ubuntu**, siga os passos para instalação de acordo com seu sistema: https://nodejs.org/en/download/package-manager*

Com o NodeJS instalado, podemos instalar o CLI (Command Line Interface) do React Native:

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

Agora precisamos instalar a JDK (**Java Development Kit**) na versão 8 ou maior com o seguinte comando:

```
sudo add-apt-repository ppa:openjdk-r/ppa
sudo apt-get update
sudo apt-get install openjdk-8-jdk
```

Podemos testar a instalação do JDK com o seguinte comando:

```
java --version
```

Em grande parte das vezes precisamos instalar algumas bibliotecas da versão 32bits do Linux para conseguir emular nosso projeto e para isso vamos utilizar o seguinte comando:

```
sudo apt-get install gcc-multilib lib32z1 lib32stdc++6
```

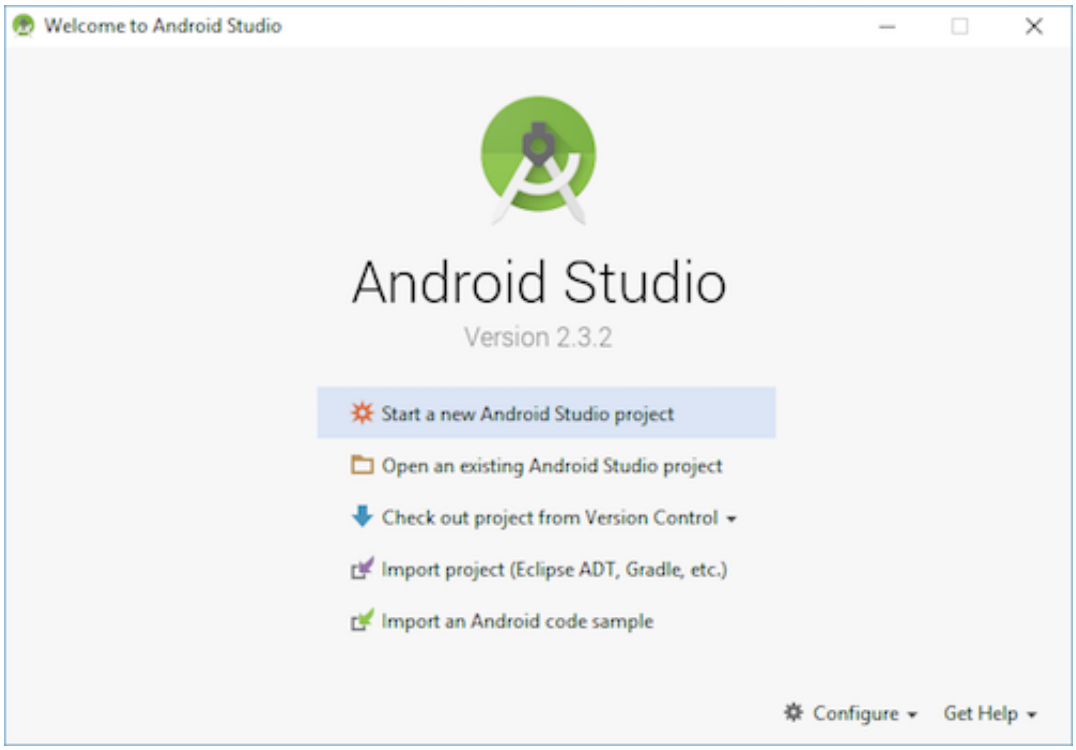
Android Studio

Para instalar a SDK do Android no Linux vamos utilizar o Android Studio. Não iremos utilizá-lo como editor de código mas apenas para **instalação das dependências** nativas para emulação do Android. Comece fazendo o **download** da ferramenta em: https://developer.android.com/studio/index.html

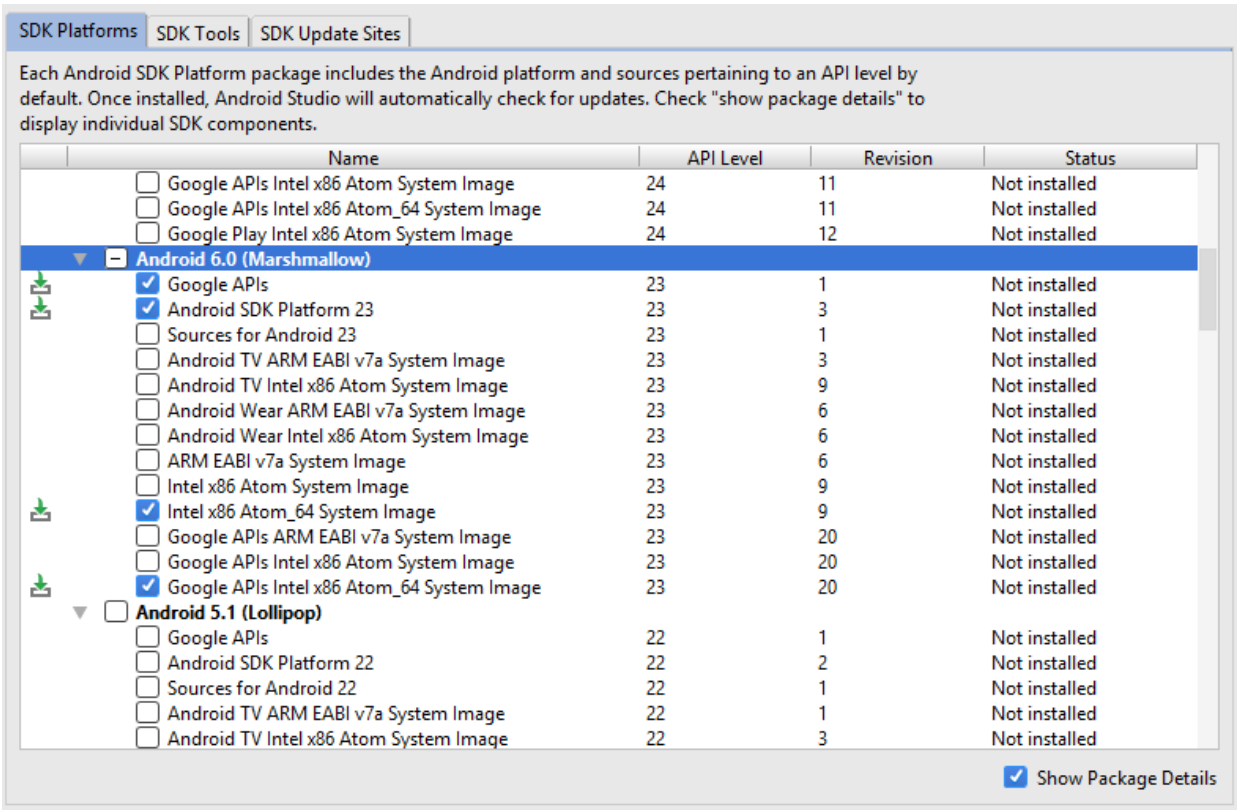
Extraia o pacote baixado em uma pasta de sua preferência e pelo terminal rode o seguinte comando **a partir da pasta extraída**:

```
./bin/studio.sh
```

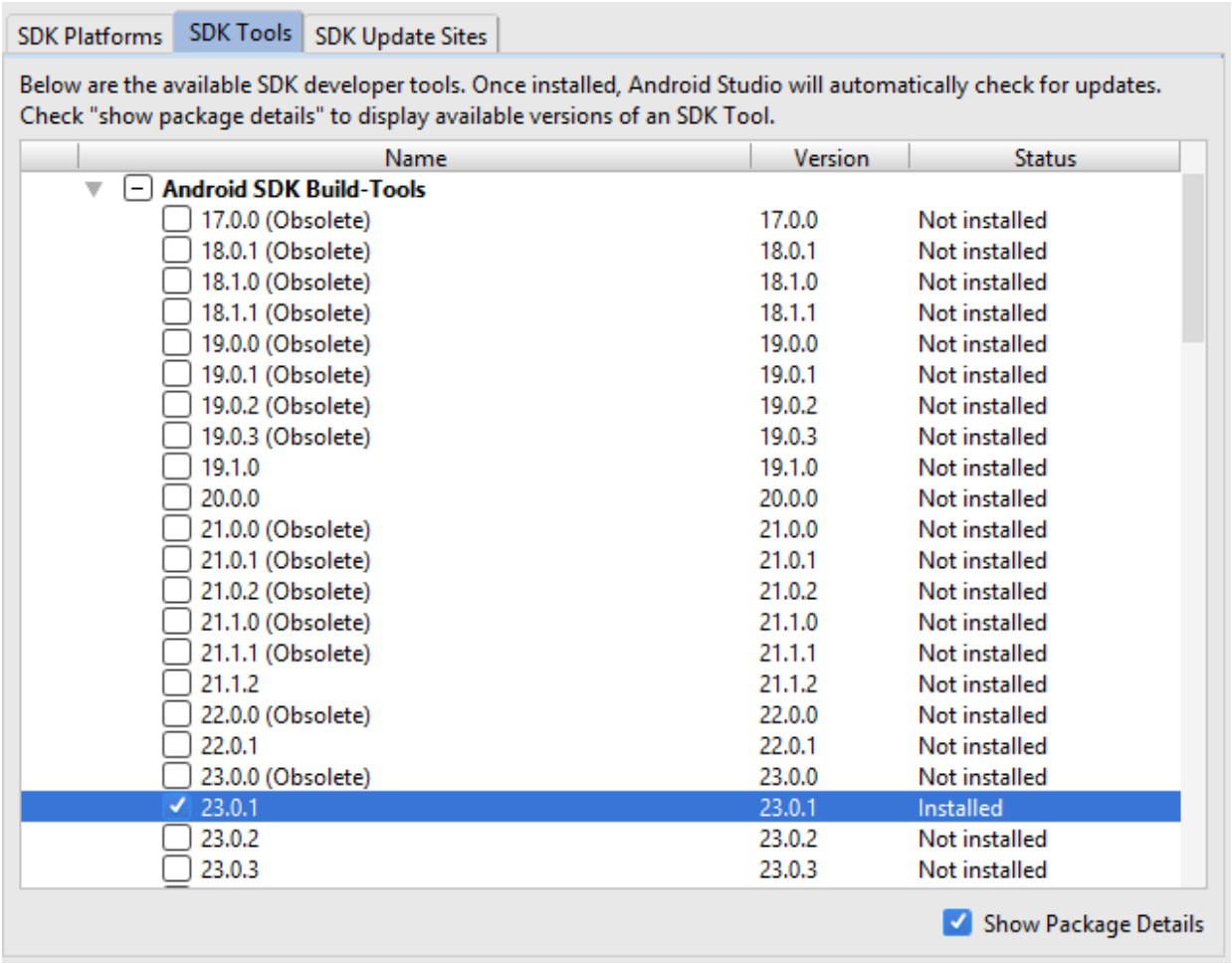
Se a instalação ocorrer bem, você receberá uma tela como a seguir:



Agora, nessa tela, clique no botão "**Configure**" e selecione a opção "**SDK Manager**". Nessa tela selecione a opção "**Show Package Details**" no rodapé da janela e instale as dependências mostradas dentro da **versão 6.0 do Android**:

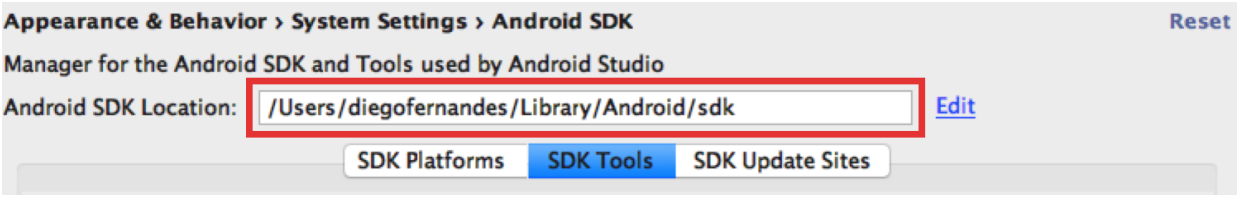


Agora na aba "**SDK Tools**" mantenha a opção "**Show Package Details**" selecionada e abrindo o item "**Android SDK Build-Tools**" selecione a opção "**23.0.1**":



Configurando ANDROID_HOME

Agora com isso configurado, **copie o endereço** presente no topo do SDK Manager com o caminho para a instalação do **SDK do Android**:



Com esse endereço precisamos configurar algumas variáveis ambiente em nosso sistema, procure pelo **primeiro** dos seguintes arquivos existentes no seu sistema: `~/.bash_profile`, `~/.profile` ou `~/.zshrc`, e adicione essas três linhas no arquivo (de preferência no início):

```
export ANDROID_HOME=CAMINHO_DA_SDK
export PATH=$PATH:$ANDROID_HOME/tools
export PATH=$PATH:$ANDROID_HOME/platform-tools
```

Adicione o caminho que você copiou do SDK Manager no lugar do texto "CAMINHO_DA_SDK".

Agora que configuramos a SDK do Android será necessário configurar o **emulador do Android**. Para emular nossa aplicação vou utilizar uma aplicação chamada **Genymotion** já que o emulador que vem instalado com a SDK do Android é mais lento.

Para criar um projeto em React Native utilize o comando:

```
react-native init NomeDoProjeto
```

Para configurar o Genymotion siga para a seção "**Configurando Genymotion**".

OS X (Mac)

Para configurar o ambiente Android no Linux, vamos precisar instalar 4 dependências: **Node, Watchman, JDK e Android Studio**. Para boa parte desses pacotes utilizaremos o **Homebrew** para instalação.

Instalando Homebrew

O Homebrew é um gerenciador de pacotes para OS X muito famoso e útil. Vamos instalá-lo em nosso sistema como seguinte comando:

```
/usr/bin/ruby -e "$(curl -fsSL https:
"raw.githubusercontent.com/Homebrew/install/master/install)"
```

O Ruby já vem instalado com o sistema OS X, se em algum caso não tiver instalado no seu sistema, procure seguir o tutorial de instalação através do site do Ruby.

Com o Homebrew instalado, vamos instalar o NodeJS e o Watchman:

```
brew install node
brew install watchman
```

Com o NodeJS instalado podemos seguir para a instalação do CLI (Command Line Interface) do React Native:

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

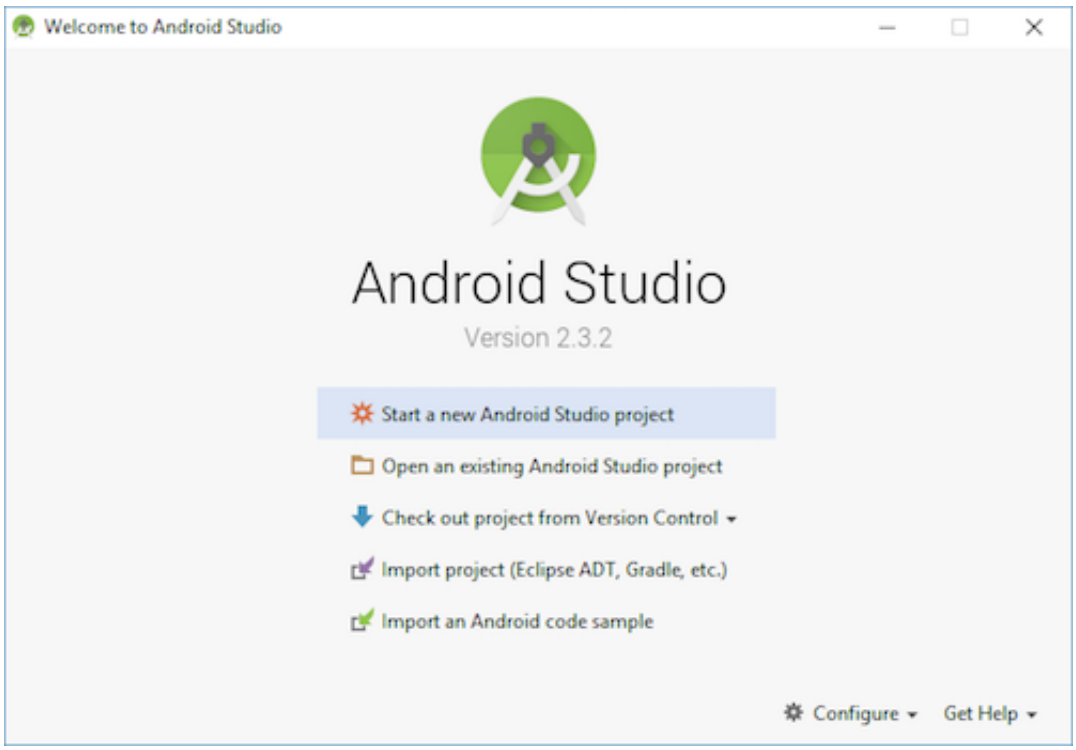
O último item a ser instalado é o JDK (Java Development Kit) do Java que pode ser baixado pelo link (Após instalado, execute o .dmg e instale seguindo os passos do instalador): http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.htm

Android Studio

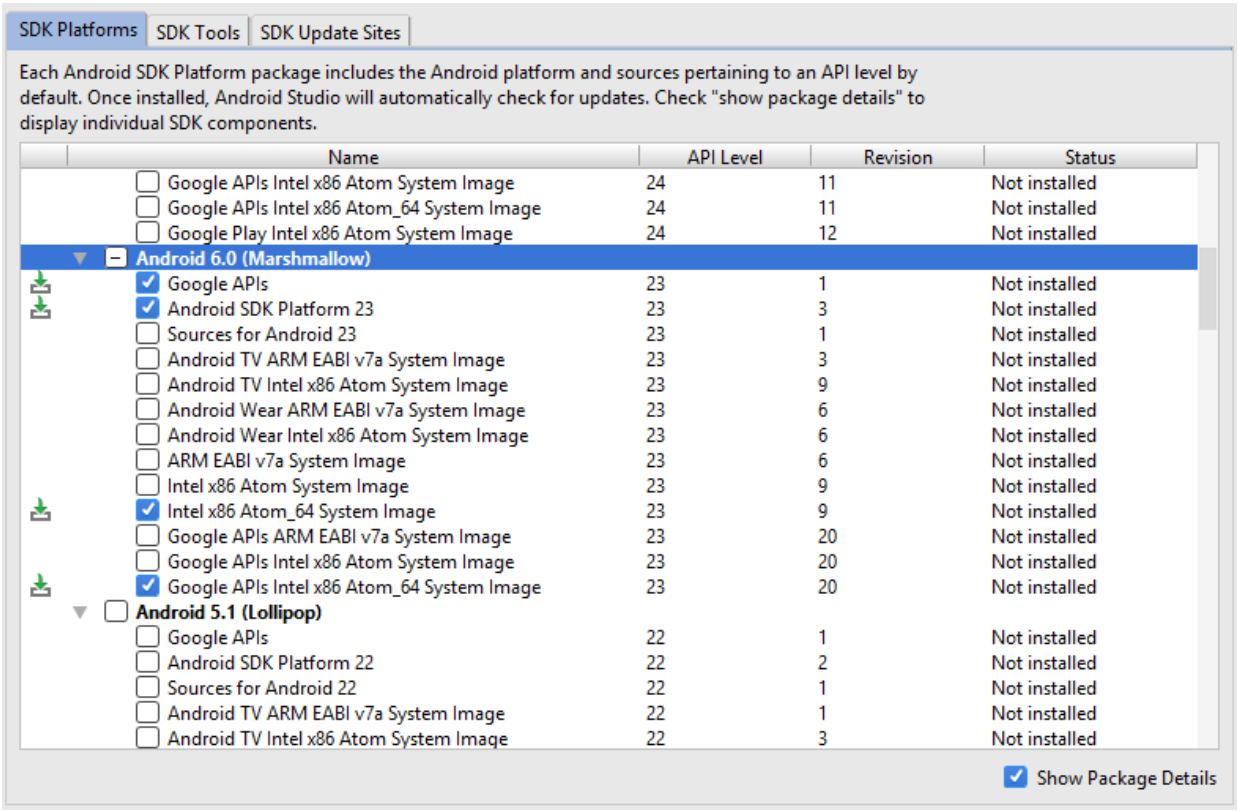
Agora vamos precisar instalar o Android Studio em nossa máquina. Não utilizaremos ele diretamente para programar mas sim para instalar a **SDK** e todas dependências de desenvolvimento do Android.

Comece instalando o Android Studio através do link: https://developer.android.com/studio/index.html e seguindo os passos descritos no arquivo executável.

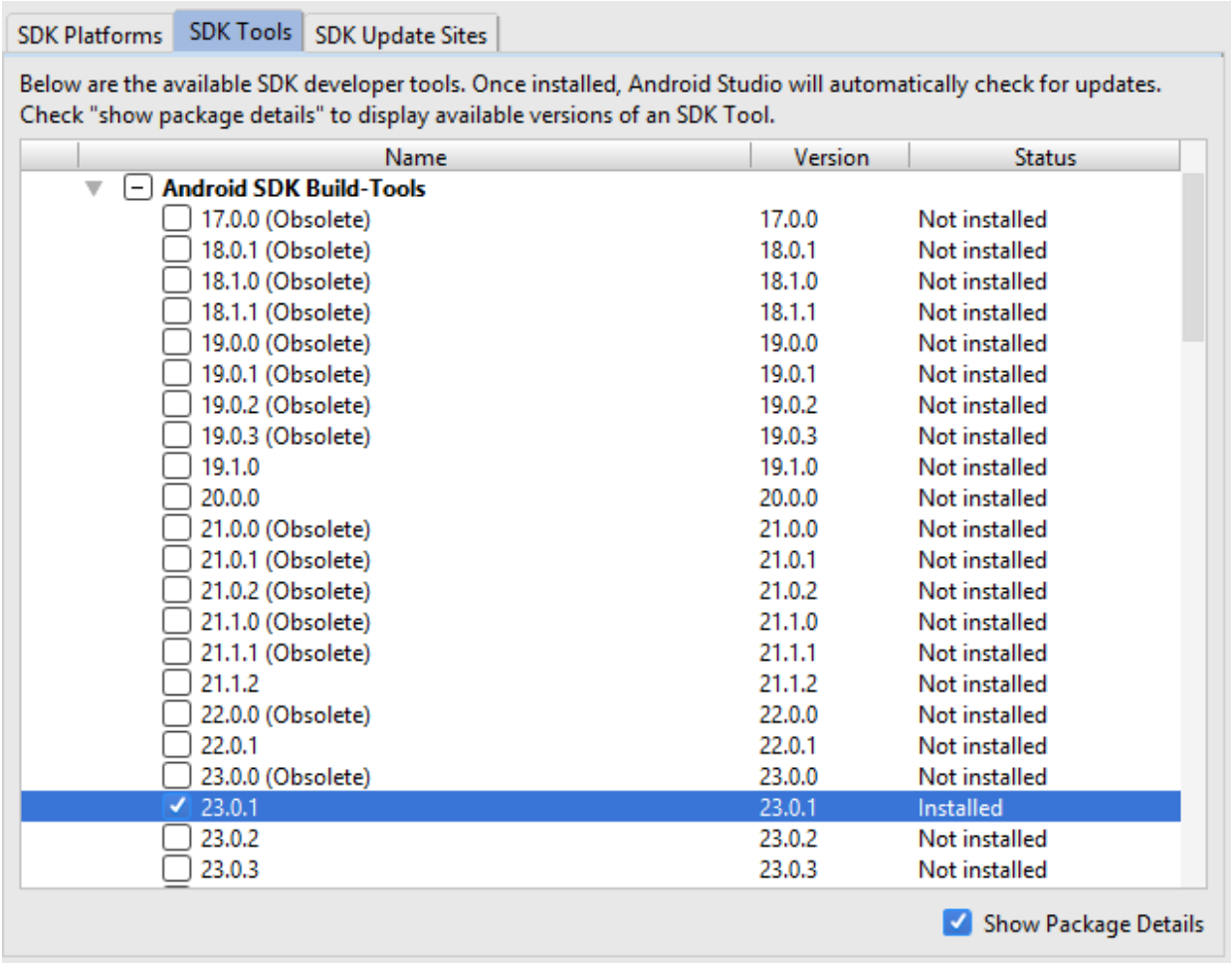
Se a instalação ocorrer bem, você receberá uma tela como a seguir:



Agora, nessa tela, clique no botão “Configure” e selecione a opção “SDK Manager”. Nessa tela selecione a opção "**Show Package Details**" no rodapé da janela e instale as dependências mostradas dentro da **versão 6.0 do Android**:

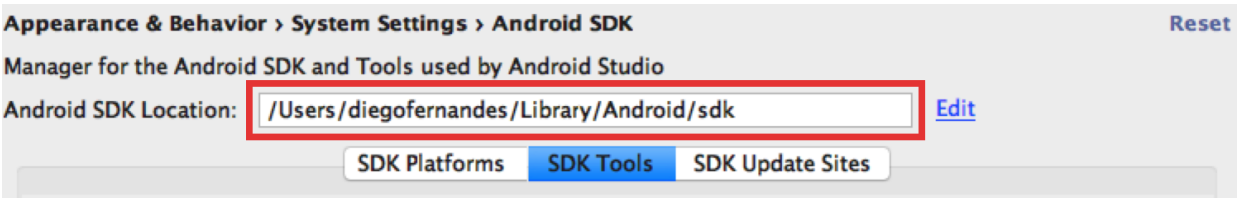


Agora na aba "**SDK Tools**" mantenha a opção "**Show Package Detail**s" selecionada e abrindo o item "**Android SDK Build-Tools**" selecione a opção "**23.0.1**":



Configurando ANDROID_HOME

Agora que temos nossa SDK instalada com sucesso, precisamos configurar a variável ambiente ANDROID_HOME em nossosistema. Para isso, no SDK Manager do Android Studio, **copie o caminho da SDK do Android** como na imagem:



Com esse endereço precisamos configurar algumas variáveis ambiente em nosso sistema, procure pelo **primeiro** dos seguintes arquivos existentes no seu sistema: `~/.bash_profile`, `~/.profile` ou `~/.zshrc`, e adicione essas três linhas no arquivo (de preferência no início):

```
export ANDROID_HOME=CAMINHO_DA_SDK
export PATH=$PATH:$ANDROID_HOME/tools
export PATH=$PATH:$ANDROID_HOME/platform-tools
```

Adicione o caminho que você copiou do SDK Manager no lugar do texto "CAMINHO_DA_SDK".

Agora que configuramos a SDK do Android será necessário configurar o **emulador do Android**. Para emular nossa aplicação vou utilizar uma aplicação chamada **Genymotion** já que o emulador que vem instalado com a SDK do Android é mais lento.

Para criar um projeto em React Native utilize o comando:

```
react-native init NomeDoProjeto
```

Para configurar o Genymotion siga para a seção "**Configurando Genymotion**".

iOS (Apenas Mac)

Para configurar o ambiente de iOS no OS X basta ter instalado o XCode no sistema. Caso você ainda não tenha instalado, você pode baixar o mesmo pelo link https://developer.apple.com/xcode/

Com o XCode instalado, basta executar o seguinte comando na pasta de um projeto React Native para rodar o React Native no simulador de iOS:

```
react-native init projeto
cd projeto
react-native run-ios
```

Configurando Genymotion

Utilizaremos o Genymotion como emulador do Android. Escolhi o Genymotion pelo fato de ser mais estável e rápido, mas você pode escolher utilizar o emulador do Android Studio.

Antes de instalar o Genymotion vamos instalar o VirtualBox. Para ambientes Mac OSX ou Windows acesse o link: https://www.virtualbox.org/wiki/Downloads

Para ambientes Linux basta executar o seguinte comando no terminal:

```
sudo apt-get install virtualbox
```

Agora vamos acessar o site https://www.genymotion.com/fun-zone/ e clicar no botão “Download Genymotion Personal Edition”. O Genymotion vai pedir que você crie uma conta na plataforma, realize esse processo e depois você poderá baixar o software

Agora, em ambientes Mac OSX ou Windows, basta instalar o software e abrí-lo através de seus executáveis.

Caso esteja no Linux precisamos realizar um processo a mais: extraia o arquivo .bin baixado do site do Genymotion em uma pasta de sua escolha e acesse-a via terminal. Na pasta do arquivo extraído execute o seguinte comando:

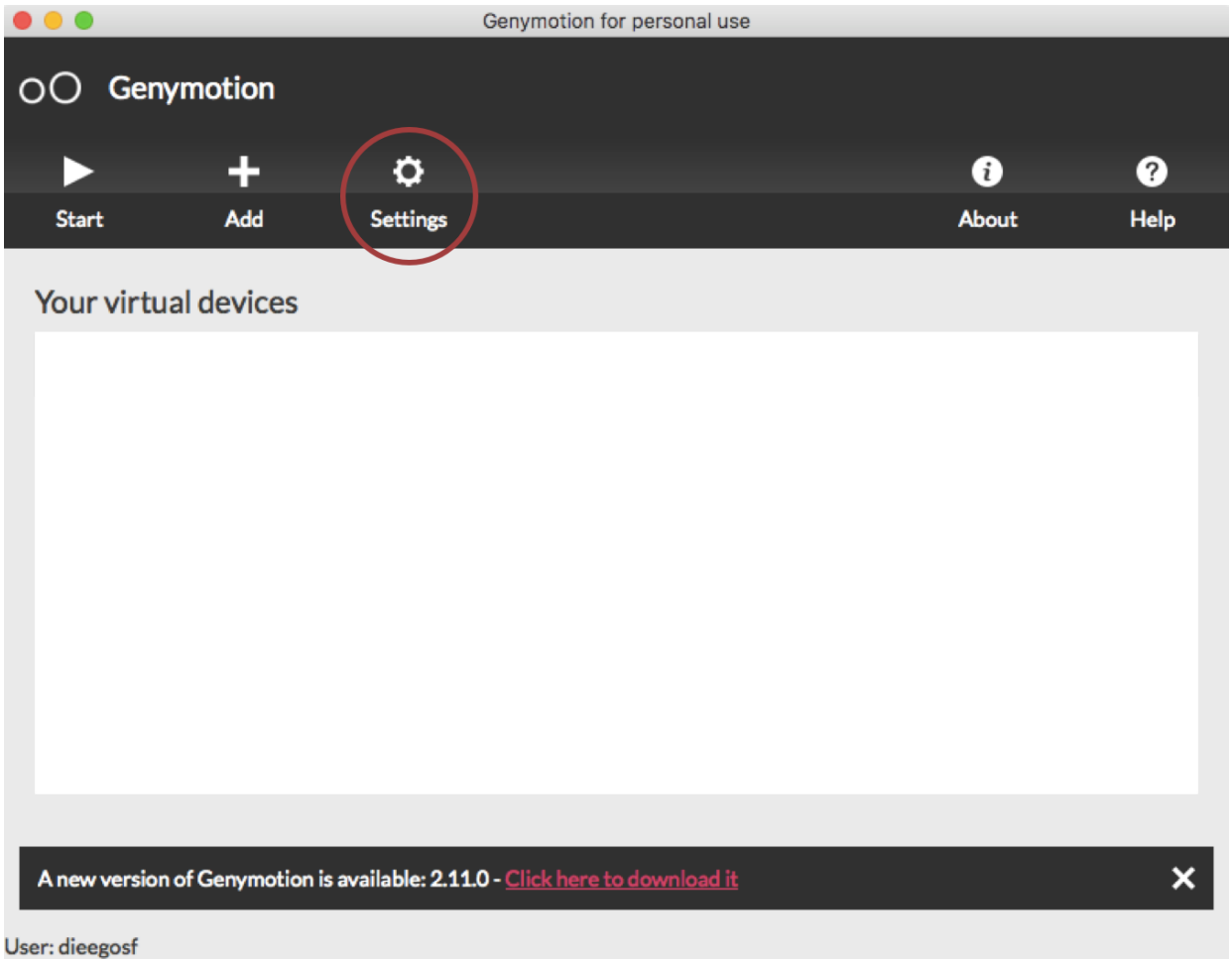
```
chmod +x genymotion-2.2.2_x64.bin
./genymotion-2.2.2_x64.bin
```

***ATENÇÃO**: Altere o nome do arquivo com a versão que você baixou (pode utilizar o TAB para completar o nome quando estiver digitando)*

Com isso o Genymotion será instalado em seu sistema e você já pode acessá-lo a partir da pasta de instalação. Por padrão a pasta que o Genymotion é instalado é "/home/[usuario]/genymotion/". Acesse essa pasta pelo terminal e execute o seguinte comando para abrir o Genymotion:

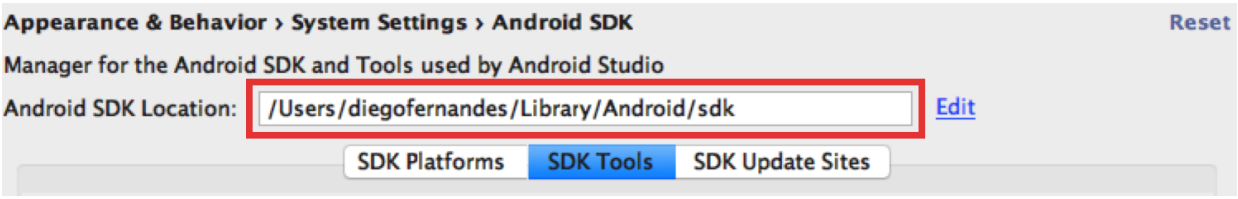
```
./genymotion
```

Se tudo ocorreu bem, você verá uma tela como a seguinte:



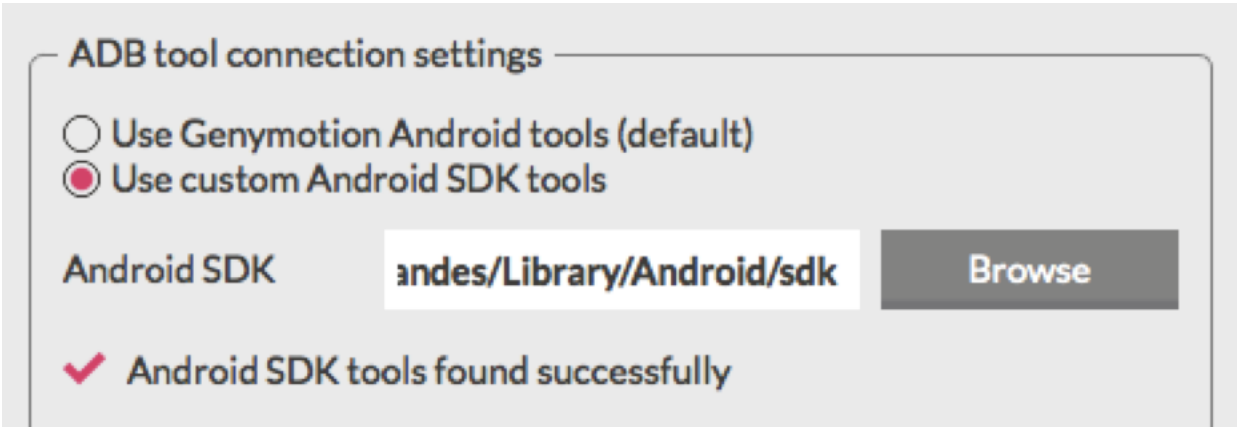
Clique no botão “Settings” e na aba “Account” faça login com sua conta criada no site do GenyMotion.

Após realizado login, ainda no menu “Settings”, na aba “ADB” precisamos informar o caminho da nossa `ANDROID_HOME` e pra isso podemos recuperar esse valor pelo Android Studio abrindo o SDK Manager:

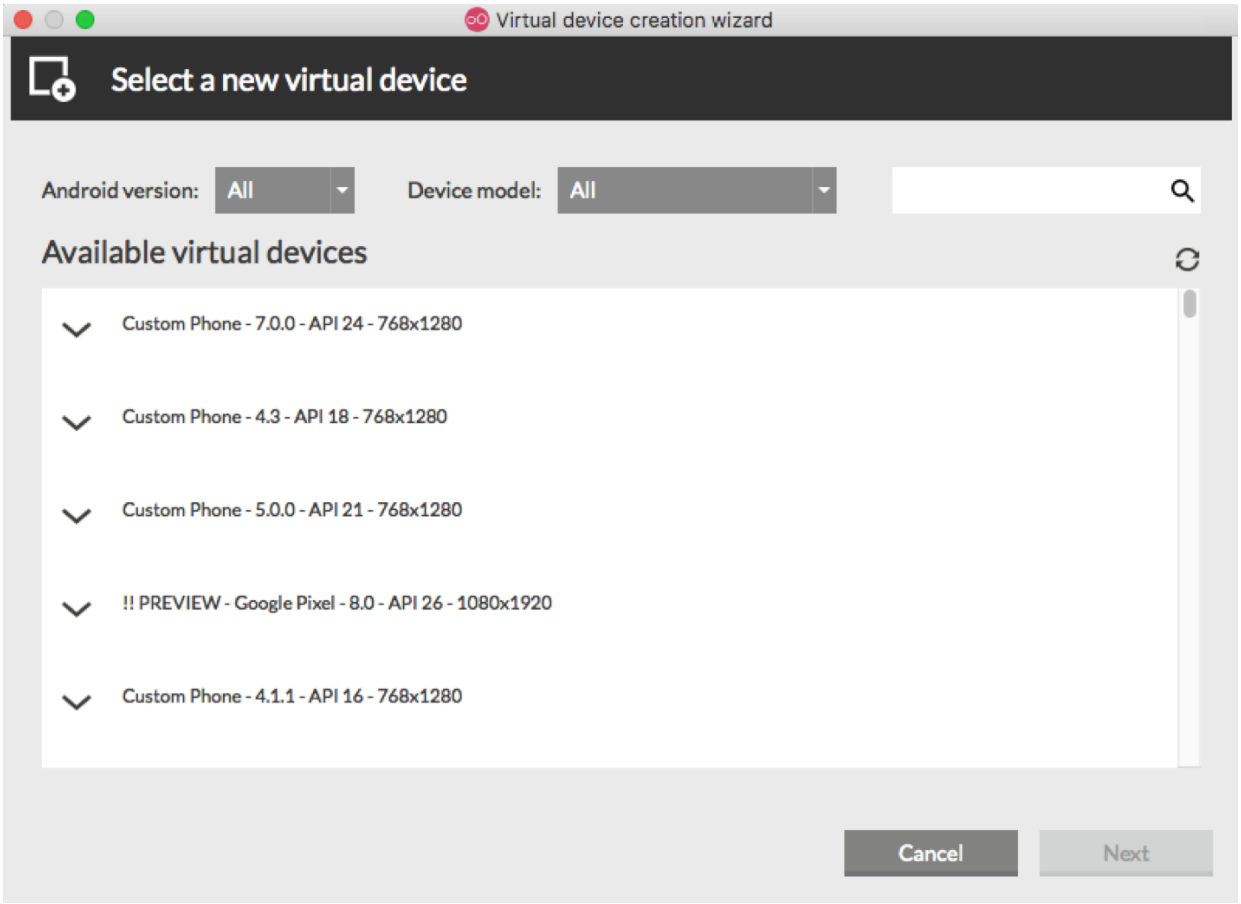


Selecione a opção “Use Custom Android SDK Tools” no Genymotion e copie a variável “Android SDK Location” do Android Studio para o campo “Android SDK”:

Sua configuração deve ficar parecida com essa:

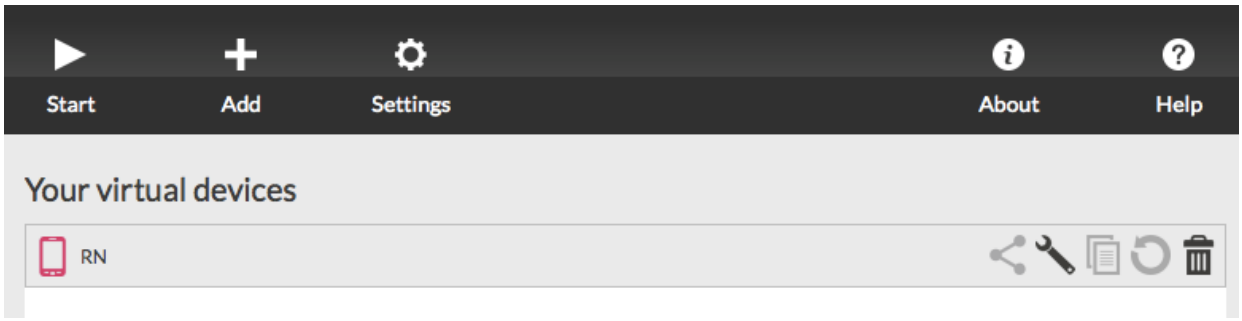


Agora fechando as configurações, voltando a tela principal do Genymotion podemos clicar na opção “Add” que abrirá a seguinte tela:



Vou utilizar um **Samsung Galaxy S8 - 7.0.0 API 24** mas aqui você pode selecionar a opção que mais lhe agradar, tente utilizar sempre versões mais recentes do Android. Selecione a opção e clique em Next.

Depois disso ele irá te pedir um nome para o emulador, coloque o que preferir ou deixe o padrão. Depois disso clique em Next novamente e aguarde o download das dependências. Ao acabar o processo você terá um novo emulador na lista inicial do Genymotion:



Com dois cliques em cima do Emulador agora será possível executar o ambiente Android. Caso você receba algum erro durante esse processo recomendo executar a máquina virtual do Android diretamente pelo VirtualBox pois dessa forma você terá o log de inicialização do ambiente que conterá qualquer possível erro.

Com o emulador aberto você pode realizar o run do React Native para Android através da pasta do seu projeto.

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

Rocketseat