

Preparando o ambiente: macOS

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.

Preparando ambiente no macOS para desenvolvimento IOS com React Native

Vamos começar a preparação do ambiente de desenvolvimento instalando os recursos necessários para rodar apps IOS usando o React Native. Todos os passos dessa configuração estão disponíveis na documentação do React Native, que pode ser acessada através [este link \(https://Reactnative.dev/docs/environment-setup\)](https://Reactnative.dev/docs/environment-setup).

Node.js

Para começar precisaremos contar com o runtime do [Node.js \(https://nodejs.org/en/\)](https://nodejs.org/en/) para executar nosso código JavaScript. Podemos instalá-lo facilmente usando o [Homebrew \(https://brew.sh/index_pt-br.html\)](https://brew.sh/index_pt-br.html) no mac:

```
brew install node
```

[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.

Watchman

Além do node, precisaremos também do [watchman](#) (<https://facebook.github.io/watchman/>).

```
brew install watchman
```

[COPIAR CÓDIGO](#)

Watchman também é uma ferramenta desenvolvida pelo Facebook para observar alterações em arquivos no seu filesystem, e é fundamental para alguns recursos interessantes que utilizaremos no decorrer do curso no ambiente de desenvolvimento com React Native.

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](#)

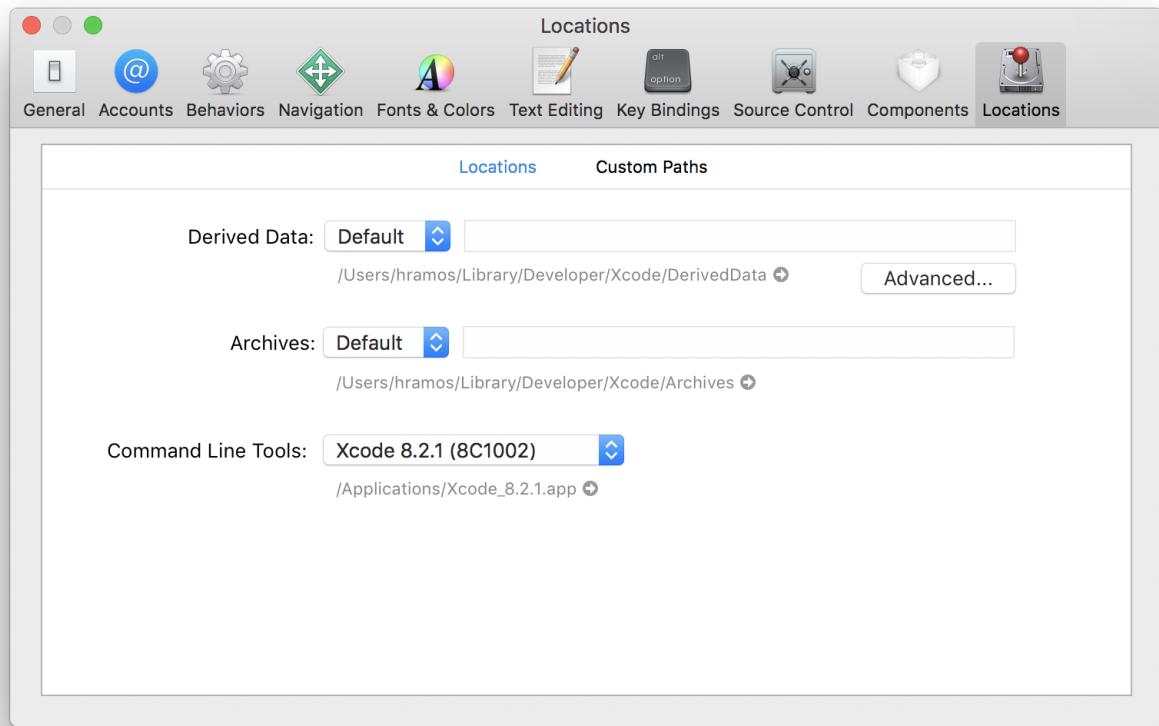
Caso você queira instalar o CLI do React-native você pode faze-lo com o seguinte comando `npm install -g React-native-cli`. Nesse caso, basta não colocar `npx` em frente aos comandos que quer executar.

Recomenda-se a utilização do Node Package Manager na versão 4. Você também pode (e é recomendável) utilizar a ferramenta Yarn. 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`

XCode

Por último, mas não menos importante, precisaremos do Xcode para contar com os recursos de desenvolvimento focadas na plataforma IOS e ter nosso simulador IOS disponível. Podemos baixar e instalar o Xcode diretamente na [Mac App Store](#) (<https://itunes.apple.com/us/app/xcode/id497799835?mt=12>). Como previsto pela própria documentação do React Native, é necessário a instalação da versão 8 ou superior do Xcode.

Após a instalação, precisaremos apenas contar com as Command Line Tools do Xcode. Para isso, abra o Xcode, selecione Preferences no menu Xcode. Vá até a aba Locations e instale as ferramentas apenas selecionando a versão mais recente no campo de seleção Command Line Tools.



Preparando ambiente no macOS para desenvolvimento Android com React Native

Algumas etapas podem ser consideradas já concluídas caso tenha configurado o ambiente para desenvolvimento na plataforma IOS

Java

Para podermos rodar nossa app também 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).

Android Studio

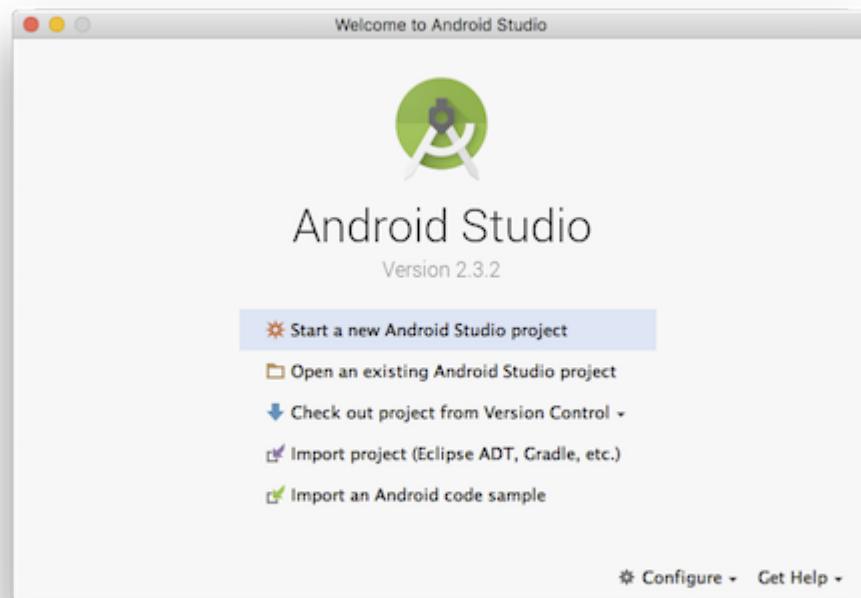
Seguindo em frente precisaremos também do ambiente de desenvolvimento Android configurado, portanto, vamos baixar e instalar também a ferramenta

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
- Performance (Intel® HAXM)
- Android Virtual Device

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



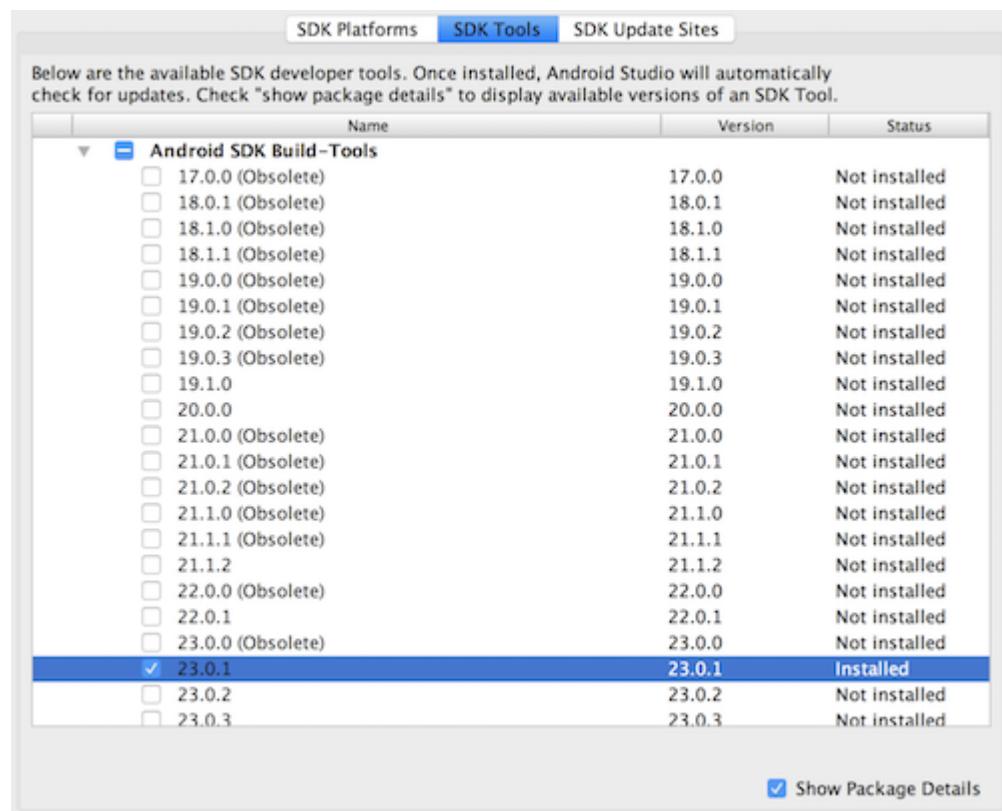
O Android Studio já instala por padrão a última versão do SDK do Android, porém para desenvolver apps nativas para o Android com React Native, precisaremos instalar adicionalmente o SDK na versão Android 6.0 (Marshmallow). Podemos adicionar SDKs adicionais no SDK Manager do Android Studio. Para acessá-lo, 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

Name	API Level	Revision	Status
Google APIs ARM 64 v8a System Image	24	10	Not installed
Google APIs ARM EABI v7a System Image	24	10	Not installed
Google APIs Intel x86 Atom System Image	24	10	Not installed
Google APIs Intel x86 Atom_64 System Image	24	10	Not installed
Android 6.0 (Marshmallow)			
Google APIs	23	1	Installed
Android SDK Platform 23	23	3	Installed
Sources for Android 23	23	1	Installed
Android TV ARM EABI v7a System Image	23	3	Not installed
Android TV Intel x86 Atom System Image	23	8	Not installed
Intel x86 Atom System Image	23	9	Not installed
Intel x86 Atom_64 System Image	23	9	Installed
Google APIs ARM EABI v7a System Image	23	19	Not installed
Google APIs Intel x86 Atom System Image	23	19	Not installed
Google APIs Intel x86 Atom_64 System Image	23	19	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".



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=$HOME/Library/Android/sdk
export PATH=$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

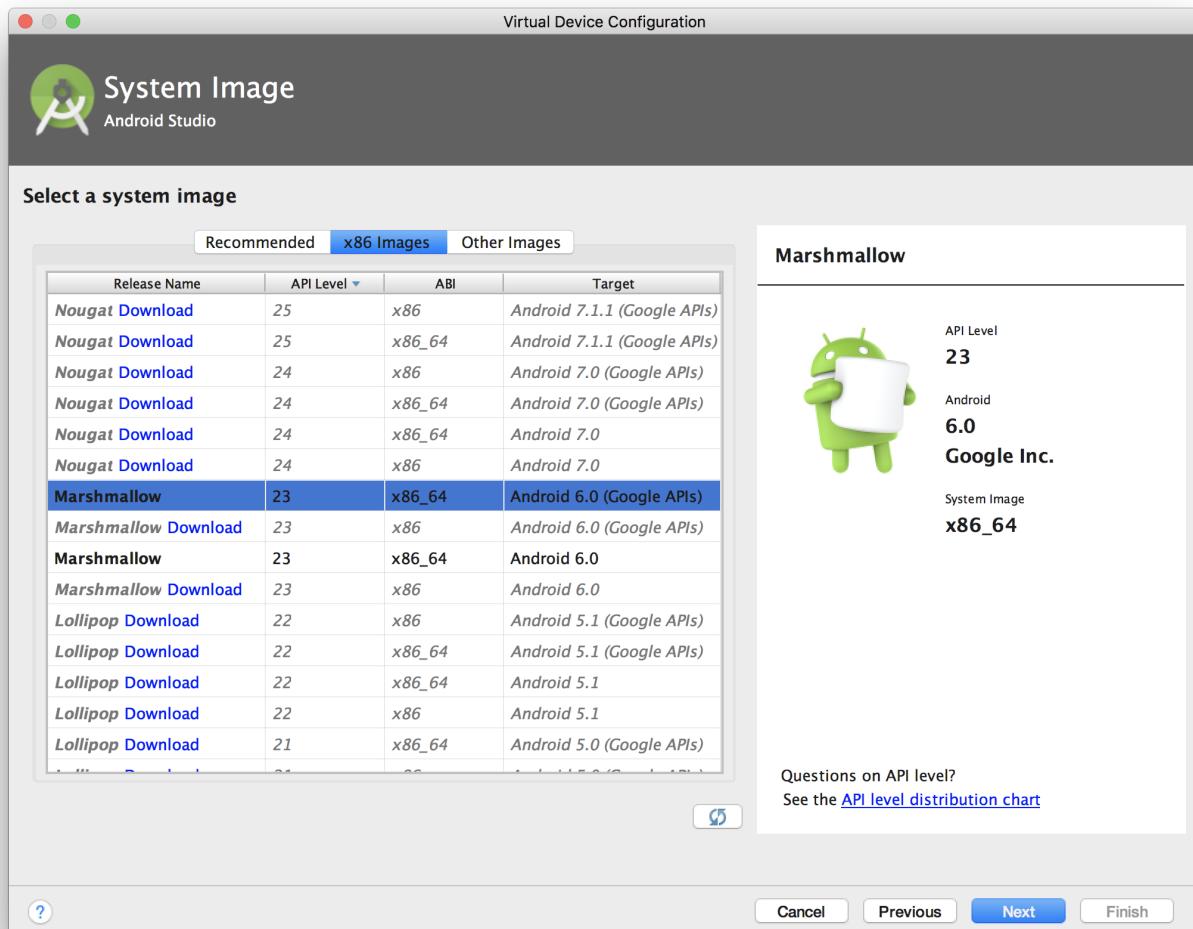
ANDROID_HOME foi adicionado corretamente à variável de ambiente PATH executando: echo \$PATH.

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)".



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 =)