Vimos o projeto do "Garconapp" rodar no navegador. Queremos agora testá-lo no Android. Esta é uma plataforma que podemos rodar no Mac, no Windows e no Linux, então, é mais simples para ser acessado pelos desenvolvedores.

Primeiramente, precisaremos do Java. Para isto, iremos acessar o [site da *Oracle*](http://www.oracle.com/technetwork/pt/indexes/downloads/index.html), na seção de Downloads, e iremos descarregar a última versão do *Java*. Depois, faremos a instalação.

Para termos certeza de que o *Java* foi instalado, iremos no terminal e usaremos o comando java - version. Ele irá indicar a versão instalada do Java:

garconapp $ java -version

java version "1.8.0\_45"

Java(TM) SE Runtime Environment (build 1.8.0\_45-b14)

Java HotSpot(TM) 64-Bit Server VM (build 25.45-b02, mixed mode)

garconapp $

É importante nos assegurarmos, porque o Android é baseado em Java.

O segundo passo é instalar o SDK do Android, no endereço [developer.android.com](https://developer.android.com/studio/index.html?hl=pt-br#downloads), onde você poderá baixar diversas ferramentas como *Android Studio*. Para trabalhar com o Cordova é suficiente baixar as *Ferramentas de Linha de Comando*. Se você utilizar Mac ou Windows, você irá baixar um arquivo .zip. No Linux, será um arquivo .tgz. Depois que você instalar o SDK, poderá rodar android sdk, no terminal.

garconapp $ android sdk

Depois de fazer a instalação, você precisará adicionar alguns componentes que são importantes para a execução do Cordova. Em seguida, irei indicar alguns deles:

Android SDK Tools Android SDK Platform-tools Android SDK Build-tools (última versão) SDK Platform (mais recente ou versões antigas)

Então, pedimos para instalar.

Após a instalação, iremos trabalhar com Cordova no Android. Nós já vimos como fazer isto: voltaremos para o projeto "Garçonapp" e iremos adicionar cordova platform add android

garconapp $ cordova platform add android

Com isto iremos adicionar a plataforma Android dentro do nosso projeto.

garconapp $ android sdk

garconapp $ cordova platform add android

Adding android project...

Running command: / Users/alura/.cordova/lib/npm\_cache/cordova-android/4.1.1/package/bin/create / Users/alura/garconapp/platforms/android org.sergiolopes.garconapp "Garçonete So de Cenoura" --cli

Creating Cordova project for the Android Pplatform:

Path: platforms/android

Package: org.sergiolopes.garconapp

Name: Garçonete So de Cenoura

Activity: MainActivity

Android target: android-22

Copying template files...

Android project created with cordova-android@4.1.1

Installing "cordova-plugin-whitelist" for android

This plugin is only applicabe for versions of cordova-android greater than 4.0. If you have a precious platform version, you do \*not\* need this plugin since the whishlist will be built in.

garconapp $

Agora aonde iremos rodar o nosso aplicativo? Temos duas opções: iremos rodar em um emulador ou em um dispositivo próprio. Vamos fazer a instalação em um dispositivo Android para ver como funciona.

Primeiro, vamos até as configurações do aparelho, e iremos habilitar no menu Programador (ou Desenvolvedor), a "Depuração USB". Vamos conectar com o cabo USB, o celular ao computador. Irá aparecer a mensagem: "Permitir depuração USB?" Damos "OK" e autorizamos o processo.

Estou com o meu celular funcionando. Queremos rodar agora o nosso projeto de teste dentro do aparelho. No terminal, vamos adicionar "cordova run android".

garconapp $ cordova run android

Se queremos rodar no dispositivo, adicionamos também --device

garconapp $ cordova run android --device

Ele irá fazer o build da minha app no device, ele irá fazer o build da minha app no Android - se não tivermos com o ambiente configurado corretamente, com SDK e outros fatores, o processo será interrompido. Mas se estiver correto, após alguns instantes, ele irá começar o processo de instalação no sistema de operação do meu celular.

Estou com celular aberto, conectado no cabo USB, e estou esperando a instalação. Veremos a mensagem "BUILD SUCESSFUL" e depois ele comecará a instalar o aplicativo. Após a instalação, ele irá abrir para mim o aplicativo. Veremos o ícone do aplicativo, com a mensagem "OI CORDOVA" e ele irá rodar o app no nosso celular. O nosso teste funcionou, sem problemas.

E se não tivermos um celular Android. Precisaremos de emulador de Android instalado no seu computador. Através do *Android SDK* podemos fazer a instalação de um emulador, porém, alguns usuários consideram que os emuladores do SDK são lentos, porque são muito pesados para rodar no computador. Uma solução é utilizar o **Genimotion**. Trata-se de emuladores de Android, que podem instalar tanto no Windows, como no Mac, e que rodam dentro do*VirtualBox* por emulação e são bastante rápidos. Você deve acessar o [site do Genymotion](https://www.genymotion.com/), criar um usuário, fazer o Download e instalação, após a realização deste processo, você poderá executá-lo da mesma forma como fizemos anteriormente. A diferença é que não precisaremos plugar um cabo USB, apenas utilizaremos o comando cordova run android no terminal e ele já irá executá-lo direto no *Genymotion*.

Vamos preparar o ambiente para suportar o desenvolvimento Android. Precisamos:

* Java JDK
* Android SDK
* VirtualBox
* Emuladores Genymotion

**Java SDK**

Precisamos do Java para usar o Android SDK. Se você não tem certeza se possui o Java instalado, abra um terminal e rode o comando:

javac -version

Precisamos do Java na versão 1.8.x ou superior. Para instalá-lo, acesse:

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>.

Localize o pacote **Java Platform JDK** na versão 8 ou superior (não precisa baixar a versão Netbeans). Baixe e siga a instalação padrão com *Next*, *Next*, *Finish*.

Ao final, abra um novo terminal e teste novamente: javac -version.

**Android SDK**

Apesar de ser um projeto híbrido, o Cordova exige as ferramentas nativas instaladas para montar a App final e testá-la. Por isso, vamos precisar do **Android SDK** que funciona no Windows, Mac e Linux.

Entre no site do Android e baixe o SDK para sua plataforma (está no finalzinho da página, depois do Studio):

<https://developer.android.com/studio/index.html#downloads>

**No Windows**

Siga a instalação padrão com *Next*, *Next* e *Finish*. No final, tem a opção *Start SDK Manager*, deixe-a marcada. Deverá abrir o *SDK Manager* do Android. Senão, vá no menu *Iniciar* e ache o *SDK Manager*.

**No Mac ou Linux**

O pacote para Mac e Linux nada mais é que um ZIP que você pode descompactar onde quiser, como a home do seu usuário ou alguma outra pasta. Faça essa descompactação e *anote o caminho completo onde colocou*.

Você vai precisar adicionar o caminho no PATH. Para isso, abra o arquivo ~/.bashrc e adicione a seguinte linha, colocando o caminho correto na sua máquina:

export PATH=$PATH:<caminho\_android\_sdk>/tools:<caminho\_android\_sdk>/platform-tools:<caminho\_android\_sdk>/build-tools

E no terminal recarregue essas novas configurações com:

source ~/.bashrc

Por fim, você precisa abrir o *SDK Manager* rodando no terminal:

android sdk

**Configuração do Android SDK**

Instalado o Android SDK, precisamos então adicionar certos componentes a ele. Abra o **Android SDK Manager** como no passo anterior. Vamos instalar:

* SDK Platform
* Android SDK Platform-tools
* Android SDK Build-tools

Marque as últimas versões do *Build-tools*, *Platform-tools* e do *SDK Platform*. Clique em *Install*. Ele baixará bastante coisa e é bem demorado para instalar.

**Instalação do Genymotion**

O último passo é instalar um emulador de Android. O SDK traz emuladores oficiais, mas eles são *muito lentos*. Uma ferramenta que cresceu bastante no mercado é o **Genymotion**, que permite rodar o Android direto virtualizado no VirtualBox. E ele é **muito mais rápido**.

O Genymotion é gratuito para uso pessoal, mas o site tenta vender a versão paga a todo custo. Você vai precisar criar uma conta, mas é rápido. Acesse <https://www.genymotion.com/download/> e crie sua conta. Então faça o download para seu sistema operacional.

Na versão Windows, você tem a opção de baixar o Genymotion já com VirtualBox embutido (ou só Genymotion caso já tenha o VirtualBox instalado). Baixe e siga a instalação padrão.

No Mac e Linux, é necessário instalar o VirtualBox separadamente. Entre em [http://virtualbox.org](http://virtualbox.org/), baixe o arquivo e siga a instalação padrão. Você talvez precise reiniciar a máquina após a instalação do VirtualBox para que o Genymotion o encontre.

**Baixando emulador no Genymotion**

Instalado o Genymotion e o VirtualBox, precisamos baixar emuladores de versões específicas do Android que nos interessam.

Abra o **Genymotion**. Ele pedirá seu usuário e senha, que você criou no Site. Depois, você pode adicionar novos emuladores. Teste instalar um **tablet com Android 5**, por exemplo, ou outra versão.

Para testar, clique em *Start* e veja se o emulador abre e funciona.

Você pode instalar outros emuladores também, em mais versões e tamanhos de tela.

\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Se o ambiente estiver propriamente configurado para Android, executar nesta plataforma é tão simples quanto no *browser*. Primeiro, adicionamos a plataforma:

cordova platform add android

E, para rodar:

cordova run android

Isso provavelmente deu um erro para você. Ele reclama que não há dispositivo Android nenhum. Logo, precisamos de um dispositivo Android para rodar, seja real ou emulado. Vamos usar os **emuladores Genymotion** que instalamos antes.

Abra o *Genymotion* e inicie algum emulador. Depois de iniciado, tente novamente:

cordova run android

Pronto, agora a App deve executar no emulador.

Se você tiver um aparelho com Android, podemos rodar a App com a mesma facilidade. O aparelho precisa estar preparado para duas coisas:

* Instalação de aplicativos fora da loja;
* Aberto para depuração USB.

**Configuração de fontes desconhecidas**

Se ainda não fez, vá nas *Configurações* do Android e entre na seção *Segurança*. Lá, **habilite a opção *Fontes Desconhecidas***, que permite a instalação de aplicativos fora da Play Store. O nome do menu pode variar um pouco de acordo com o fabricante, mas todos têm essa opção.

**Depuração USB**

Primeiro, você precisa que o menu *Programador* ou *Desenvolvedor* esteja habilitado nas *Configurações* do Android. Ele é o penúltimo menu, logo antes de *Sobre*, e não vem habilitado por padrão. Para habilitá-lo, vá no menu *Sobre* e localize o *Número da versão*. Toque 7 vezes no menu e ele mostrará uma mensagem de que abriu as opções de desenvolvimento.

Volte nas *Configurações* do Android. Agora você deve ver a opção *Programador*. Entre nessa opção e habilite o menu **Depuração USB**. Conecte o cabo USB no Desktop e ele deve pedir para confirmar a chave única do computador (marque a opção de *sempre permitir*).

Configure o USB para transferência de fotos para que o aplicativo consiga ser executado no celular.

**Finalmente executando no aparelho**

Toda essa configuração é necessária apenas a primeira vez em cada aparelho. Feito tudo, basta plugar o cabo USB e rodar no terminal:

cordova run android --device

Você deve ver a App demo do Cordova abrindo no aparelho.

Você vai precisar rodar o OSX, seja em um Mac real, um Hackintosh ou mesmo virtualizado de alguma maneira. Vamos instalar:

* Xcode
* ios-sim
* ios-deploy

## Xcode

As ferramentas de desenvolvimento da Apple estão todas no guarda-chuva do Xcode. Então, precisamos instalá-lo. Baixe o Xcode na Mac App Store, ou [no site](https://developer.apple.com/xcode/downloads/).

## ios-sim e ios-deploy

O cordova-ios usa duas ferramentas Node.js. O **ios-sim** ajuda a lidar com o emulador, e o **ios-deploy** permite rodar a App em um dispositivo real. Vamos instalar ambas.

No terminal, use o npm para instalar globalmente:

sudo npm -g install ios-sim ios-deploy

No Mac já preparado conforme o exercício anterior, podemos agora rodar no emulador do iOS que vem junto com o Xcode. Basta adicionar a plataforma como fizemos antes:

cd garconapp

cordova platform add ios

Feito este passo, conseguimos rodar no emulador usando:

cordova emulate ios

Se tudo correu bem, ele deve abrir nossa App no emulador padrão.

Para ver quais emuladores há disponíveis, execute:

cordova emulate ios --list

Você pode escolher um deles na hora de rodar, como por exemplo:

cordova emulate ios --target iPad-Air

odar num dispositivo real é mais complicado. A App precisa estar assinada digitalmente com chaves válidas da Apple. No Xcode 7 podemos fazer isso diretamente por ele de forma gratuita. (a outra forma seria ter a conta paga da Apple, criar as chaves e importá-las no Cordova para exportação pelo terminal)

Abra o projeto no Xcode. Ele fica em garconapp/platforms/ios/, e é um arquivo com o nome da App e extensão .xcodeproj.

Repare no botão de *Play* em cima. Você pode escolher algum dos emuladores e rodar imediatamente. Mas isso não é o que queremos; queremos rodar no dispositivo real. Para isso, precisamos adicionar nossa conta no Xcode e criar um *Provisioning Profile*.

Nesta tela mesmo do projeto, repare que no meio há opções do projeto - como *Bundle identifier*, *Version* etc. Nos interessa agora a configuração **Team**. Se você nunca usou o Xcode antes, ele deve estar vazio. Clique na opção e em **Add an Account**. Siga os passos para fazer login com seu Apple ID.

Adicionada sua conta, ela deve aparecer agora no dropdown de *Team* do projeto. Selecione-o. Ele deve reclamar que falta um *provisioning profile* para essa App. Clique no botão *Fix* que ele gerará os certificados para você.

Agora, conecte o dispositivo via USB e desbloqueie a tela. Na aba superior do botão *Play*, deve aparecer seu aparelho, além dos emuladores. Selecione-o e clique em *Play* para executar.