

Começando com Android Studio

O GUIA PASSO A PASSO



Fillipe Cordeiro



Começando com Android Studio

O guia passo a passo

Fillipe Cordeiro | AndroidPro

1^a edição

Copyright © 2016, AndroidPro

Todos os direitos reservados e protegidos pela Lei 9.610 de 19/02/1998

Nenhuma parte deste eBook, sem autorização prévia por escrito do autor, poderá ser reproduzida ou transmitida sejam quais forem os meios empregados: eletrônicos, mecânicos, fotográficos, gravação ou quaisquer outros.

Começando com Android Studio - O guia passo a passo

AUTOR Fillipe Cordeiro

SITE www.androidpro.com.br

E-MAIL fillipe@androidpro.com.br

O autor não possui nenhum vínculo com as instituições e produtos citados, utilizando-os apenas para ilustrações.

Sumário

Introdução

1 Configurando o Ambiente de Desenvolvimento do Android Studio

1.1 Requisitos do Sistema

1.2 Como Instalar o Java Development Kit (JDK)

1.2.1 Instalação do JDK no Windows

1.2.2 Instalação do JDK no Mac OS X

1.2.3 Instalação do JDK no

1.3 Download do Pacote Android Studio

1.4 Instalando o Android Studio

1.4.1 Instalação no Windows

1.4.2 Instalação no Mac OS X

1.4.3 Instalação no Linux

1.5 O Assistente de Configuração Android Studio

1.6 Instalando os Pacotes mais Recentes do Android SDK

1.7 Atualizando o Android Studio e o SDK

1.8 Resumo

2 Criação de um Aplicativo Exemplo

2.1 Criando um novo projeto Android

2.2 Definindo as Configurações do Projeto e SDK

2.3 Criando uma Activity

2.4 Modificando o Aplicativo de Exemplo

2.5 Visualização de Layout

2.6 Resumo

3 Um Tour pela Interface do Android Studio

3.1 A Tela de Boas Vindas

3.2 Painel Principal

3.3 As Janelas

3.4 Switcher e Navegação por Arquivos Recentes

3.5 Mudando o Tema do Android Studio

3.6 Resumo

4 Criando um Dispositivo Virtual Android (AVD)

4.1 Sobre Dispositivos Virtuais Android (AVDs)

4.2 Criando um novo AVD

4.3 Iniciando o Emulador

4.4 Executando o aplicativo no AVD

4.5 Configurações de Run / Debug

4.6 Parando um Aplicativo em Execução

4.7 Resumo

5 Testando Apps em Dispositivos Físicos

5.1 Uma visão geral do Debug Android Bridge (ADB)

5.2 Ativando o ADB em Dispositivos Android

5.2.1 Configuração de ADB no Mac OS X

5.2.2 Configuração do ADB no Windows

5.2.3 Configuração do ADB no Linux

5.3 Resumo

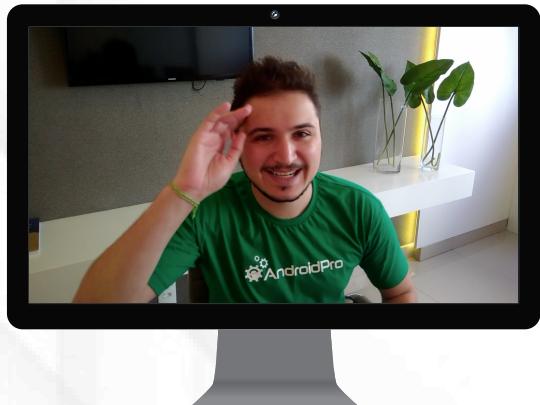
..... AULA ONLINE GRATUITA!

OS 5 PASSOS OBRIGATÓRIOS

PARA VOCÊ SER UM **DESENVOLVEDOR ANDROID**
PROFISSIONAL E INDEPENDENTE!

Porque você precisa participar?

- 1 Para entender os **erros** que fazem você desistir e se frustrar
- 2 Para descobrir um **método COMPROVADO** e **ÚNICO** para desenvolver praticamente qualquer tipo de aplicativo Android
- 3 Para saber quais as **habilidades exatas**, além da programação, que vão te transformar em um profissional
- 4 Para aprender a **criar oportunidades de trabalho** e ganhar dinheiro com desenvolvimento Android desde já!



VAGAS *LIMITADAS!*

INSCREVA-SE AQUI!

Introdução

O objetivo deste livro é ensinar os conceitos básicos da ferramenta oficial do Google para desenvolvimento Android, o Android Studio.

O Android Studio é baseado no IntelliJ IDEA, uma das ferramentas mais poderosas de desenvolvimento Java, oferecendo muitos recursos que melhoram a sua produtividade na criação de aplicativos para Android, tais como:

- Um sistema flexível de compilação baseado no Gradle
- Compilação de variações de aplicativos e geração múltipla do arquivo APK
- Modelos de código para ajudar a desenvolver aplicativos
- Um editor de layout rico com suporte para edição de arrastar e soltar
- Ferramentas de análise para verificar desempenho, usabilidade, compatibilidade de versão, e outros problemas
- Proteção de código com ProGuard e Gradle
- Suporte para o Google Cloud Platform, tornando mais fácil a integração com o Google Cloud Messaging e App Engine

Considerando que você já possui alguma experiência em programação Java, você está pronto para baixar o Android Studio e começar seu aprendizado.

Boa leitura!

1. Configurando o Ambiente de Desenvolvimento do Android Studio

Para começar devemos configurar o ambiente de desenvolvimento Android. Isto envolve uma série de etapas que consistem em instalar o Java Development Kit (JDK) e o Android Studio (IDE - Ambiente de Desenvolvimento Integrado), que também inclui o Kit Android de Desenvolvimento de Software (mais conhecido como SDK - Software Development Kit).

Neste capítulo, vamos ver as etapas necessárias para instalar os componentes que precisamos para o desenvolvimento de aplicativos Android nos sistemas Windows, Mac OS X e Linux.

1.1 Requisitos do Sistema

Podemos desenvolver aplicativos Android em qualquer um dos seguintes tipos de sistemas a seguir:

- Microsoft® Windows® 8/7/Vista/2003 (32 ou 64 bits)
- Mac® OS X® 10.8.5 ou posterior, até o 10.9 (Mavericks)
- Sistemas Linux com a versão 2.11 ou posterior do GNU C Library (glibc)
- Mínimo de 2 GB de RAM, 4 GB de RAM recomendado
- Pelo menos 1 GB para o Android SDK, imagens do sistema de emulador e caches
- Kit de desenvolvimento Java (JDK) 7

1.2 Como Instalar o Java Development Kit (JDK)

O SDK do Android foi desenvolvido utilizando a linguagem de programação Java. Da mesma forma, os aplicativos Android também são desenvolvidos usando essa linguagem. Como resultado, o Kit de Desenvolvimento Java (JDK) é o primeiro componente que deve ser instalado.

Para desenvolver para Android, precisamos instalar qualquer versão 7 ou 8 da Edição Standard do Java Platform Development Kit.

1.2.1 Instalação do JDK no Windows

Para sistemas Windows, o JDK pode ser obtido no website da Oracle Corporation (<http://www.oracle.com/technetwork/java/javase/downloads/index.html>).

Caso o JDK apropriado não esteja instalado em seu sistema, baixe o pacote mais recente que coincide com o sistema da sua máquina, ou da máquina que irá usar. Uma vez baixado, inicie o executável de instalação e siga as instruções na tela para completar o processo de instalação.

1.2.2 Instalação do JDK no Mac OS X

Por padrão, o Java não vem instalado nas versões recentes do Mac OS X. Para confirmar se o Java está ou não presente no seu computador, abra o terminal e digite o seguinte comando:

```
java -version
```

Assumindo que o Java esteja instalado, você verá na janela do terminal a seguinte saída:

```
java version "1.7.0_71"  
Java(TM) SE Runtime Environment (build 1.7.0_71-b14)  
Java HotSpot(TM) 64-Bit Server VM (build 24.71-b01, mixed mode)
```

Caso o Java não esteja instalado, ao executar o comando "java" na janela do terminal, aparecerá uma mensagem em conjunto com um alerta e um botão de Mais Informações que, quando clicado, vai exibir a página web da Oracle Java:

```
No Java runtime present, requesting install
```

Ou seja, o Java não está instalado na sua máquina. Sendo assim, na página da Oracle Java, localize e baixe o pacote de instalação do Java SE 7 JDK para Mac OS X.

Abra o executável baixado (arquivo .dmg) e clique duas vezes no ícone para instalar o pacote Java:



Figura 1 - Instalando o Java

O Java vai aparecer na a janela de instalação do OS X, e vai te guiar através das etapas envolvidas na instalação do JDK. Uma vez que a instalação for completada, volte para a janela do Terminal e execute o seguinte comando:

```
java -version
```

1.2.3 Instalação do JDK no Linux

Em primeiro lugar, se o sistema de desenvolvimento escolhido estiver executando a versão do Ubuntu de 64 bits, então é essencial que o pacote de suporte da biblioteca 32 bits seja instalado:

```
sudo apt-get install lib32stdc++6
```

Para fazer a instalação mais facilmente, vamos utilizar um repositório do WebUpd8 que já faz a instalação automática do Java.

Abra o terminal do Linux e execute os seguintes comandos, separadamente:

```
sudo add-apt-repository ppa:webupd8team/java  
sudo apt-get update
```

Java 7:

```
sudo apt-get install oracle-java7-installer
```

Ou Java 8:

```
sudo apt-get install oracle-java8-installer
```

Após o término da instalação, verifique se correu de forma correta, executando o seguinte comando:

```
java -version
```

1.3 Download do Pacote Android Studio

A maioria do trabalho envolvido no desenvolvimento de aplicativos Android será executada utilizando o Android Studio, que pode ser

baixado em: <http://developer.android.com/sdk/index.html>

Nessa página, clique no botão de download, se ele listar a plataforma correta (em um navegador web baseado no Windows, por exemplo, o botão será **Download Android Studio for Windows**), ou selecione o link **Outras Opções de Download** para selecionar manualmente o pacote certo para a sua plataforma e sistema operacional.

Na tela seguinte, aceite os termos e condições para iniciar o download.

1.4 Instalando o Android Studio

Uma vez baixado, as etapas exatas para a instalação do Android Studio diferem dependendo do sistema operacional em que a instalação será realizada.

1.4.1 Instalação no Windows

Localize o arquivo executável do Android Studio que você baixou (chamado android-studio-bundle-<versão>.exe) e clique duas vezes sobre ele para iniciar o processo de instalação. Clique no botão Sim na janela do Controle de Contas do Usuário, se ele aparecer.

Quando o assistente de configuração do Android Studio aparecer,

percorra as várias telas para configurar a instalação de acordo com os seus requisitos: de localização da pasta na qual o Android Studio deve ser instalado, e se deve ou não ser disponibilizado para outros usuários do sistema. Uma vez que as opções tenham sido configuradas, clique no botão Instalar para iniciar o processo de instalação.

Em versões do Windows com o menu Iniciar, o Android Studio recém-instalado pode ser iniciado a partir da entrada adicionada a esse menu durante a instalação. No Windows 8, o executável pode ser colocado na barra de tarefas, para facilitar o acesso ao navegar para o diretório **android-studio/bin**, basta clicar com o botão direito sobre o arquivo executável e selecionar a opção de menu **Fixar na Barra de tarefas**. Note que o arquivo executável é fornecido nas versões executáveis em **32 bits (studio)** e **64 bits (studio64)**. Se você estiver executando um sistema de **32 bits** certifique-se de usar o executável certo.

1.4.2 Instalação no Mac OS X

O Android Studio para Mac OS X é baixado na forma de um arquivo de imagem de disco (.dmg). Uma vez que o arquivo **android-studio-ide-<versão>.dmg** tenha sido baixado, localize-o em uma janela do **Finder** e clique duas vezes sobre ele para abri-lo, como mostrado na figura abaixo:



Figura 2 - Instalando o Android Studio em um Mac OS X

Para instalar o pacote, basta arrastar o ícone do Android Studio e soltá-lo na pasta **Applications** ou **Aplicativos** (dependendo da configuração de idioma da sua máquina). Dessa forma, o pacote será instalado na pasta de aplicativos do sistema, um processo que pode demorar alguns minutos.

Para iniciar o Android Studio, localize o arquivo executável na pasta **Applications** ou **Aplicativos**, usando o **Finder**, e clique duas vezes sobre ele. Se, ao tentar executar o Android Studio, aparecer uma mensagem de erro indicando que o JVM não pôde ser encontrado, você precisará baixar e instalar o pacote Mac OS X Java 6 JRE no sistema. Isso pode ser feito pelo site da Apple usando o seguinte link: <http://support.apple.com/kb/DL1572>

Uma vez que o pacote Java para OS X tenha sido instalado, o Android Studio pode ser iniciado livre de qualquer problema.

1.4.3 Instalação no Linux

Depois de baixar o pacote do Android Studio para Linux, extraia os arquivos do ZIP.

Perceba que o pacote do Android Studio será extraído de um sub-diretório de mesmo nome. Supondo que o passo acima tenha sido executado em **/home/android**, os arquivos serão descompactados em **/home/android/android-studio**. Para iniciar o Android Studio, abra uma janela de terminal, vá para o diretório **android-studio/bin** e execute o seguinte comando:

```
./studio.sh
```

1.5 O Assistente de Configuração Android Studio

Na primeira vez que iniciarmos o Android Studio após a instalação, veremos uma janela que oferece a opção de importar as configurações a partir de uma versão anterior do Android Studio. Se você tiver configurações de uma versão anterior e desejar importá-las para a instalação mais recente, selecione a opção e o local apropriado. Caso contrário, indique que você não precisa importar nenhuma configuração anterior e clique em OK para prosseguir.

Depois que o Android Studio terminar de carregar, teremos o assistente de configuração:

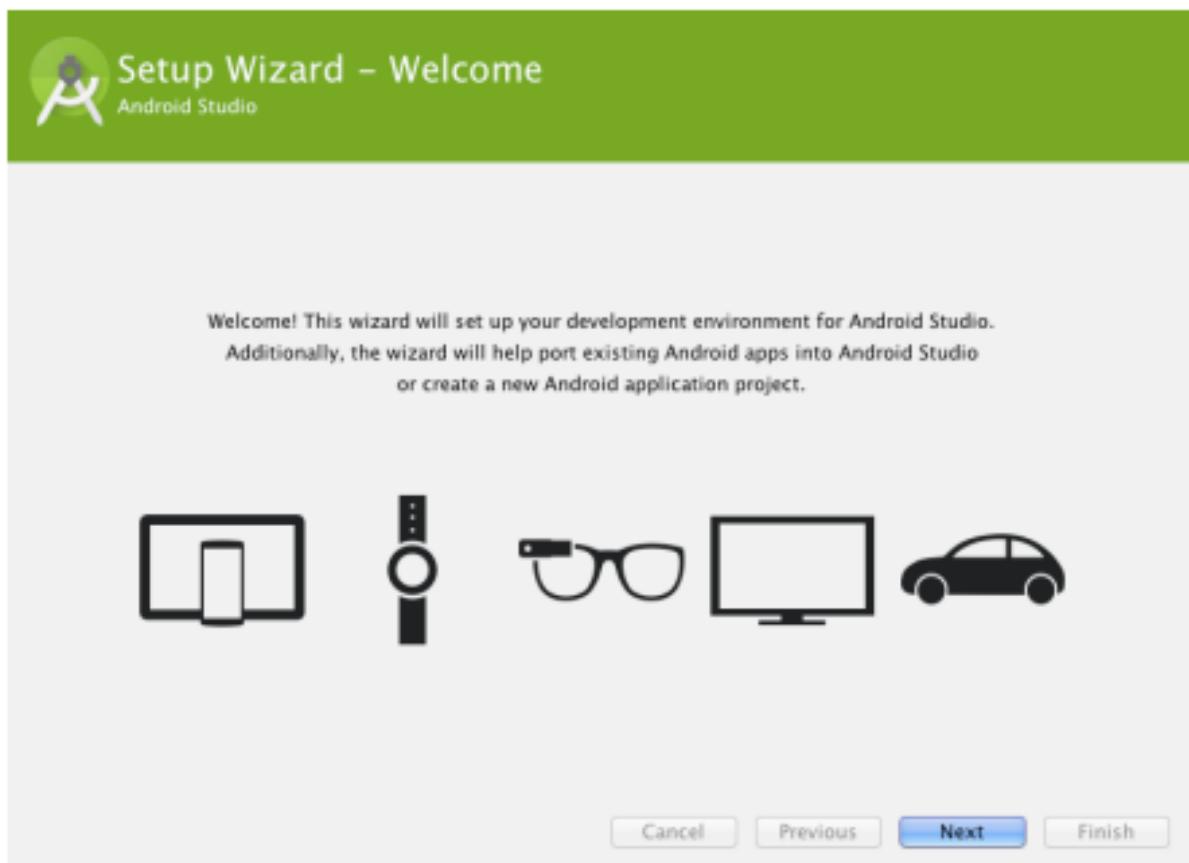


Figura 3 - Assistente de Configuração do Android Studio

Clique no botão **Next** para avançar, na opção de instalação padrão e em **Next** novamente.

Na tela do contrato de licença, selecione e aceite cada uma das licenças listadas antes de clicar em **Finish**, para concluir o processo de configuração. Após a conclusão, teremos a tela de boas vindas à ferramenta mostrada na Figura 4.

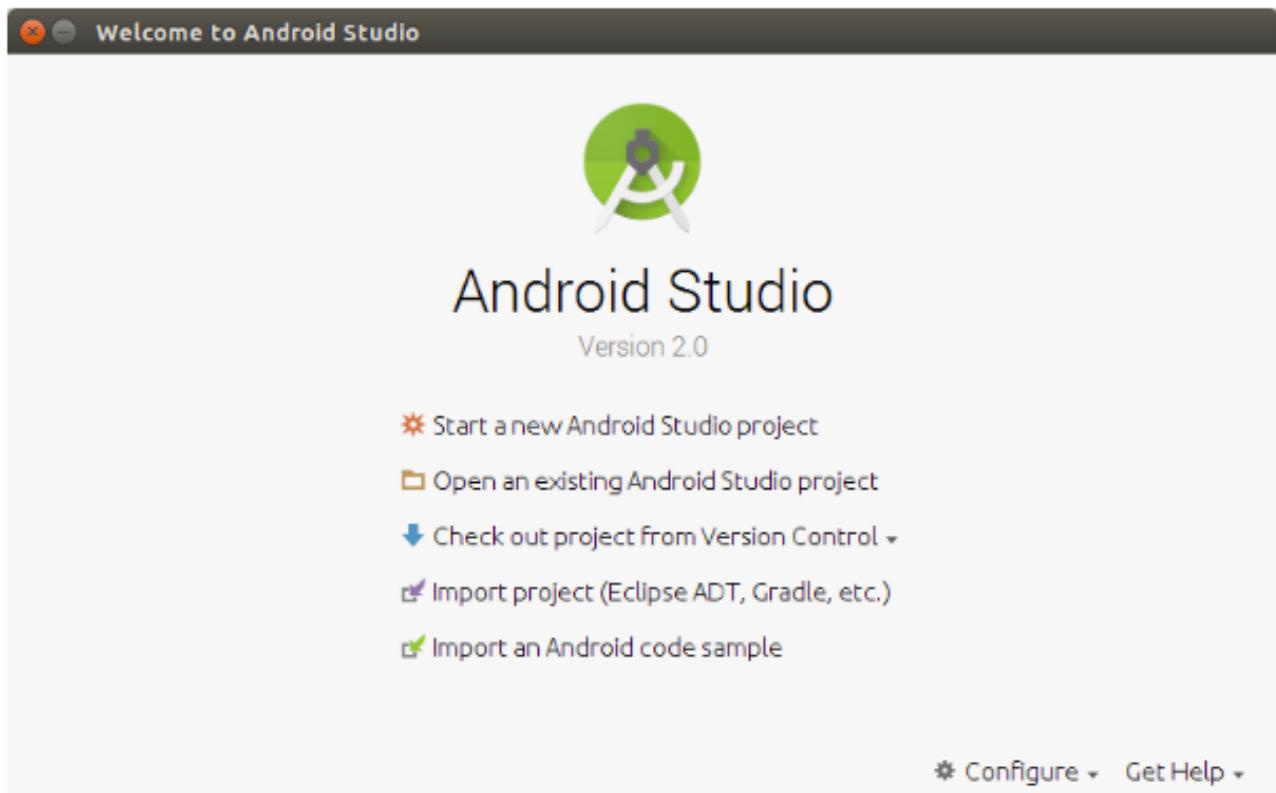


Figura 4 - Tela de Boas Vindas do Android Studio

1.6 Instalando os Pacotes mais Recentes do Android SDK

Através dos passos realizados até agora, instalamos o Java, a IDE Android Studio e o atual conjunto de pacotes SDK padrão do Android. Antes de prosseguirmos, vale a pena usar alguns minutos para verificar quais pacotes estão instalados e para instalar todos os pacotes faltantes.

Esta tarefa pode ser realizada utilizando o Android SDK Manager, iniciado a partir do Android Studio através do caminho Configure > SDK Manager de dentro da tela de boas-vindas.

A ferramenta SDK Manager aparece como o ilustrado abaixo:

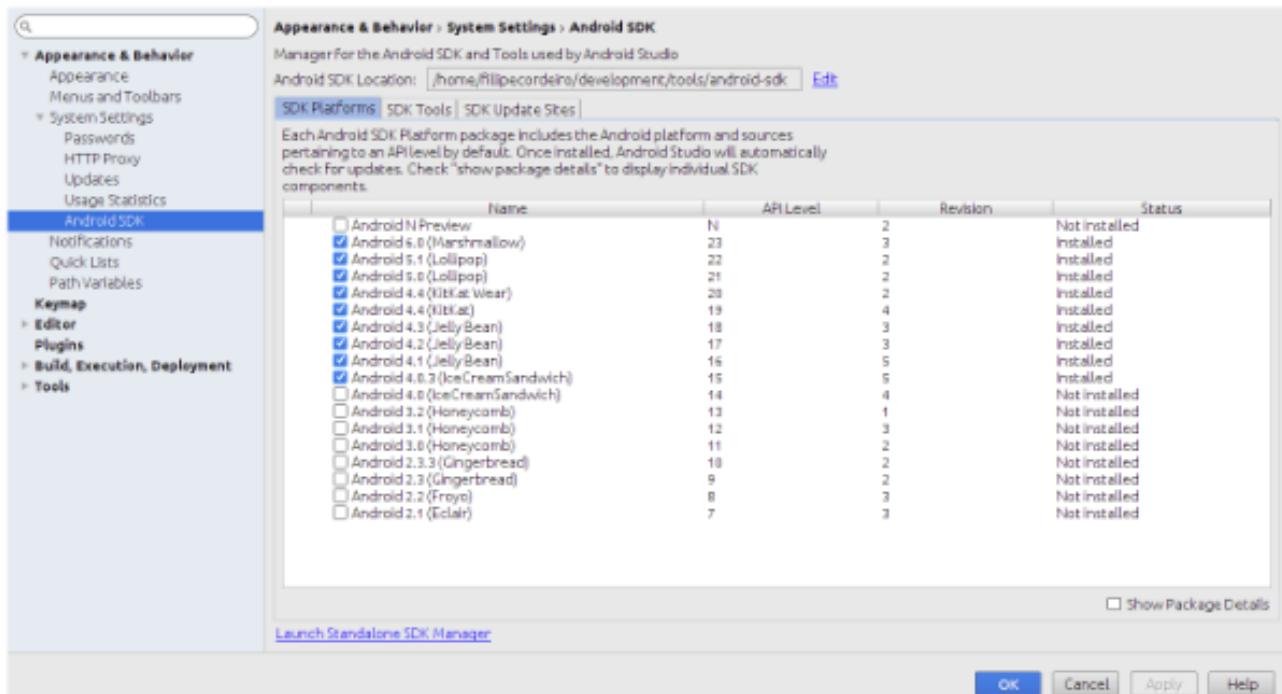


Figura 5 - SDK Manager

Dentro do Android SDK Manager, certifique-se de que os seguintes pacotes estejam listados como Instalados na coluna Status:

- Tools > Android SDK Tools
- Tools > Android SDK Platform-tools
- Tools > Android SDK Build-tools
- SDK Platform (versão mais recente) > SDK Platform
- SDK Platform (versão mais recente) > ARM EABI v7a System Image
- Extras > Android Support Repository
- Extras > Android Support Library
- Extras > Google Repository
- Extras > Google USB Driver (necessário apenas em sistemas Windows)

No caso de qualquer um dos pacotes acima estar listado como não instalado, simplesmente selecione as caixas ao lado dos pacotes e clique no botão Apply para iniciar o processo de instalação. Aceite os termos de licença antes confirmar a instalação. Feito isso, o SDK Manager começa a baixar e instalar os pacotes selecionados. À medida que a instalação prossegue, uma barra de progresso aparecerá na parte inferior da janela do gerenciador indicando o status do processo.

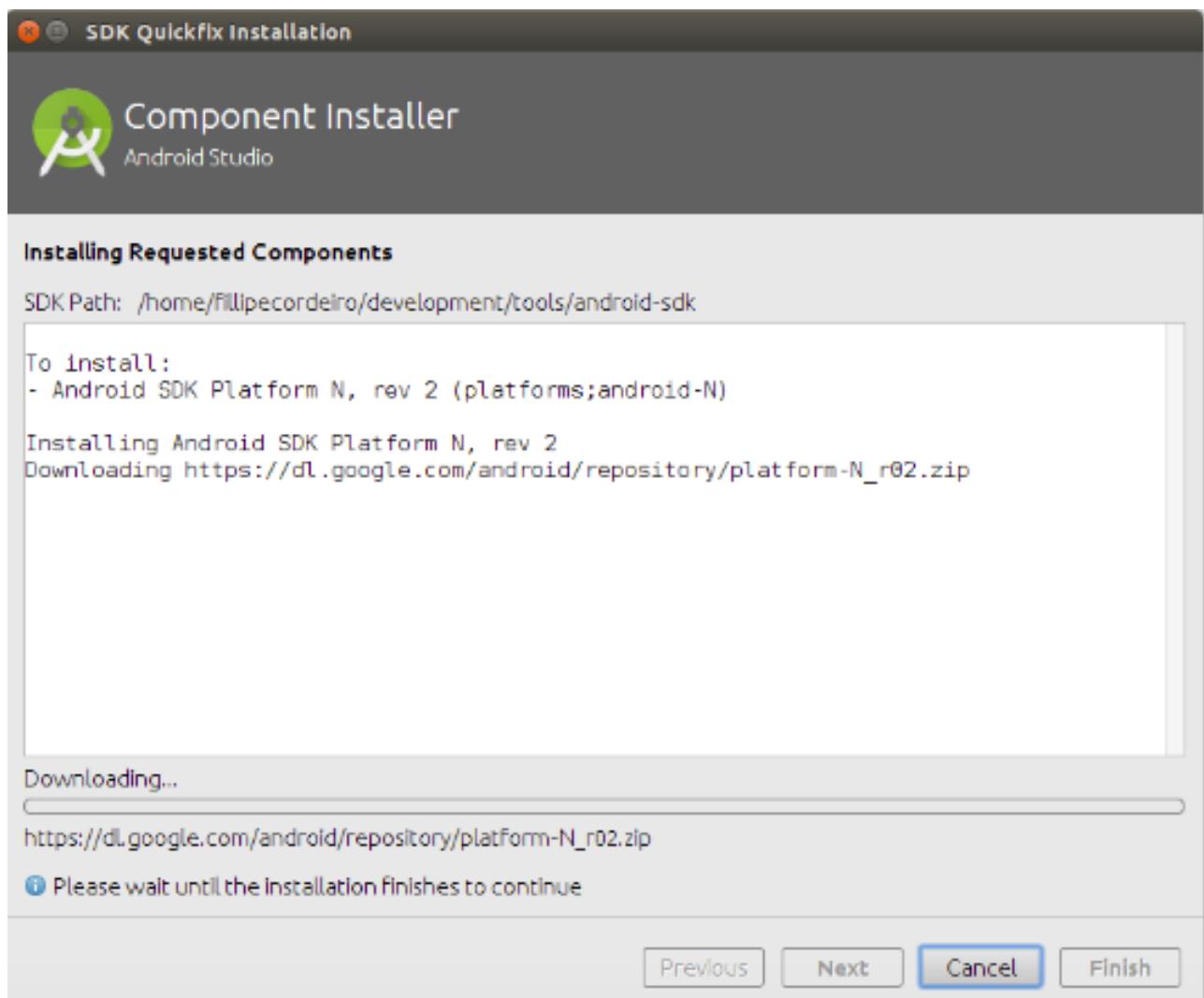


Figura 6 - Instalando Pacotes Faltantes no SDK Manager

Quando a instalação for concluída, reveja a lista de pacotes e certifique-se de que todos pacotes selecionados estejam listados como Instalados na coluna Status. Se algum acusar "não instalado", verifique se eles estão selecionados e inicie o processo descrito novamente.

1.7 Atualizando o Android Studio e o SDK

De vez em quando são liberadas novas versões do Android Studio e do Android SDK. Novas versões do SDK são instaladas usando o Android SDK Manager. Já o Android Studio tem um sistema de notificação para quando uma atualização estiver pronta para ser instalada.

1.8 Resumo

O primeiro passo para começar a desenvolver aplicativos Android é a criação de um ambiente de desenvolvimento adequado. Esse ambiente é composto pelo Java Development Kit (JDK), SDKs Android e o Android Studio. Neste capítulo, cobrimos as etapas necessárias para instalar esses pacotes nos sistemas Windows, Mac OS X e Linux.

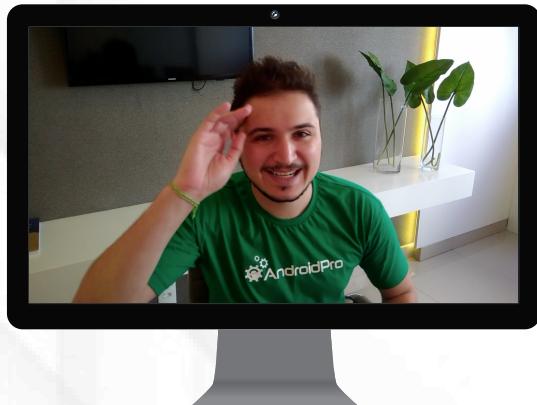
..... AULA ONLINE GRATUITA!

OS 5 PASSOS OBRIGATÓRIOS

PARA VOCÊ SER UM **DESENVOLVEDOR ANDROID**
PROFISSIONAL E INDEPENDENTE!

Porque você precisa participar?

- 1 Para entender os **erros** que fazem você desistir e se frustrar
- 2 Para descobrir um **método COMPROVADO** e **ÚNICO** para desenvolver praticamente qualquer tipo de aplicativo Android
- 3 Para saber quais as **habilidades exatas**, além da programação, que vão te transformar em um profissional
- 4 Para aprender a **criar oportunidades de trabalho** e ganhar dinheiro com desenvolvimento Android desde já!



VAGAS *LIMITADAS!*

INSCREVA-SE AQUI!

2. Criação de um Aplicativo Exemplo

Agora que passamos pelos primeiros passos com o Android Studio, antes de seguirmos para temas um pouco mais avançados, é preciso ter certeza de que todos os pacotes de desenvolvimento que precisamos estejam instalados e funcionando corretamente. A melhor maneira de fazer isso é criar um aplicativo Android, compilá-lo e executá-lo. E é isso que vamos ver agora. Vamos criar um projeto Android simples, usando o Android Studio. Uma vez com esse projeto criado, poderemos explorar o uso de emuladores para executar um teste do app.

2.1 Criando um Novo Projeto Android

Para iniciar o processo de desenvolvimento de um aplicativo, criamos um novo projeto dentro do Android Studio. Vamos começar, portanto, iniciando o Android Studio para que a tela de boas-vindas apareça (conforme a Figura 4).

Quando essa janela aparece, quer dizer que o Android Studio já está pronto para a criação de um novo projeto. Para criar o novo projeto, basta clicar em **Start a new Android Studio project**, isso exibirá a primeira tela do assistente **New Project**.

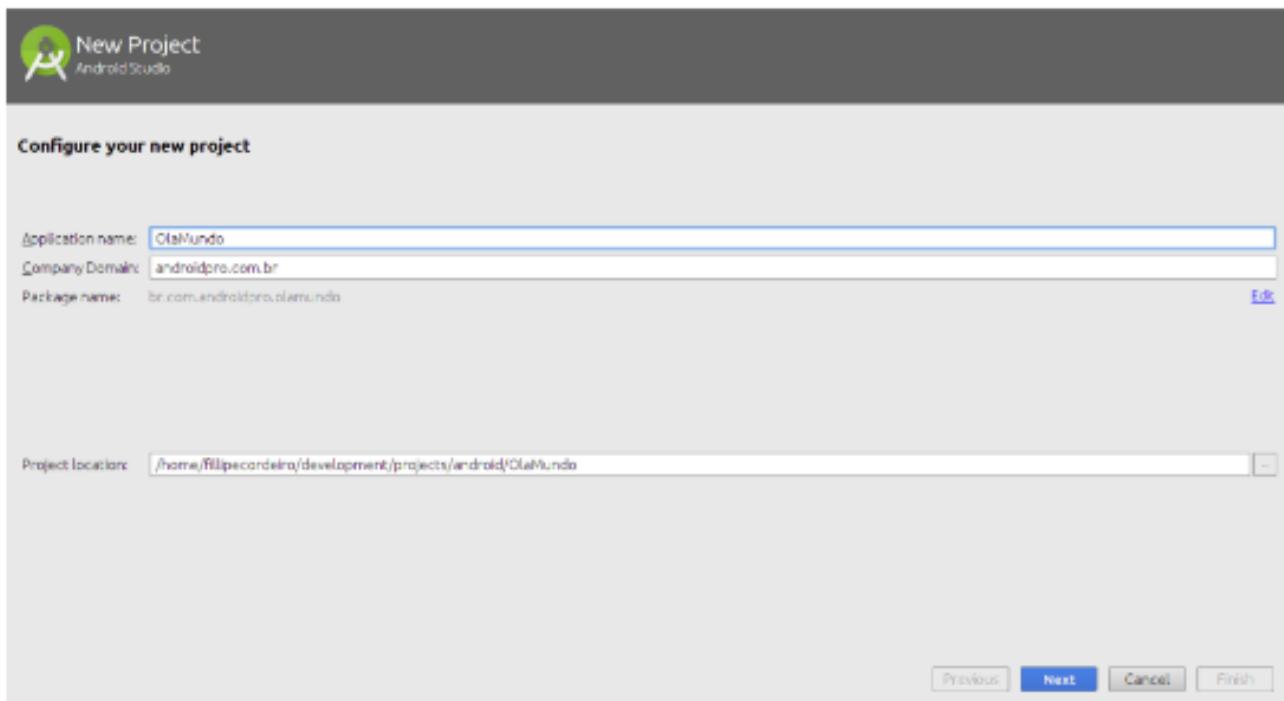


Figura 7 - Novo Projeto

2.2 Definindo as Configurações do Projeto e SDK

Na janela **New Project**, defina o campo **Application name** para OlaMundo. O nome do aplicativo é o nome pelo qual ele será referenciado e identificado dentro do Android Studio. É, também, o nome que será utilizado quando o aplicativo for disponibilizado para download na Google Play Store.

O **Package name** é usado exclusivamente para identificar o aplicativo dentro do ecossistema de aplicativos Android. Deve ser baseado na URL invertida do nome de domínio seguido do nome do aplicativo. Por exemplo o seu domínio é www.meusite.com.br, e a aplicação foi nomeada como OlaMundo, o nome do pacote pode ser especificado como segue: **br.com.meusite.olamundo**

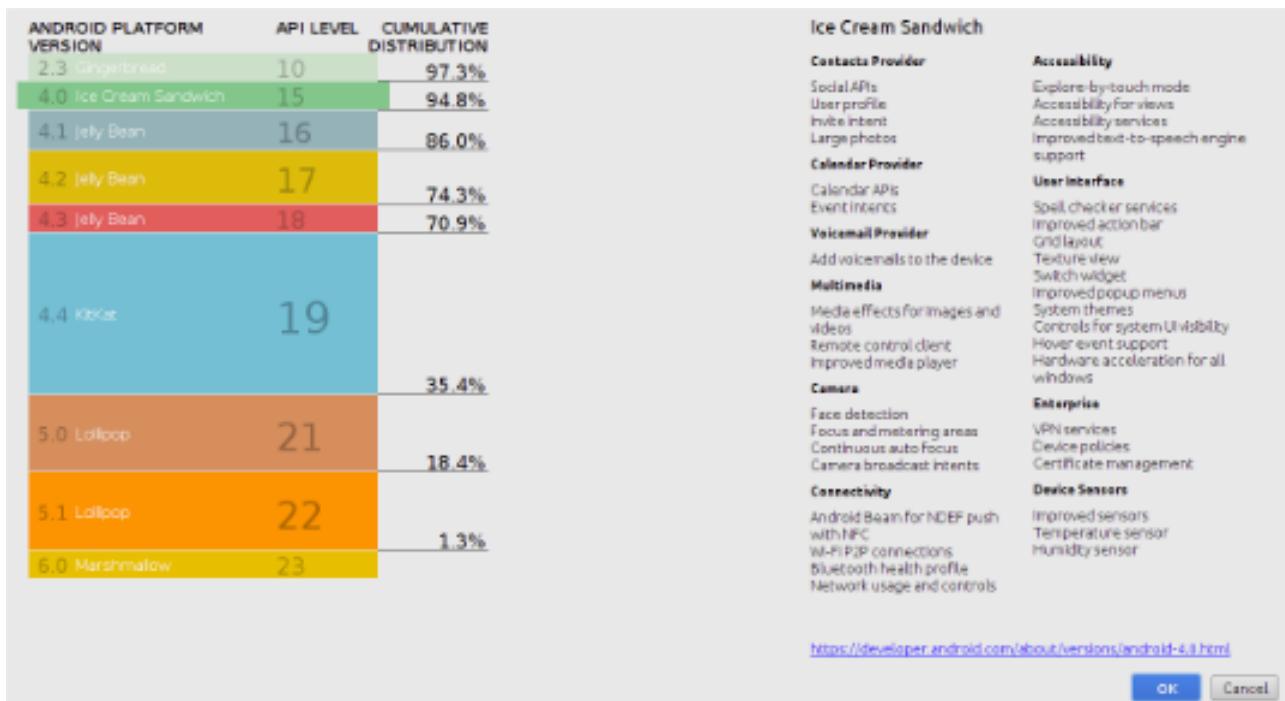


Figura 8 - Versões de Plataforma Android

Desde que o projeto não seja destinado ao Android TV, Google Glass ou a dispositivos portáteis, deixe as opções restantes desativadas antes de clicar em **Next**.

2.3 Criando uma Activity

O próximo passo é definir o tipo de **Activity** inicial que será criada para o aplicativo. Existem várias opções de **Activities** para usarmos quando criamos os projetos Android no Android Studio. Nesse exemplo, no entanto, basta selecionar a opção de criar uma **Basic Activity**.

Se você não tem um domínio, use androidpro.com.br para fins de teste, embora isso deva ser mudado antes da publicação:
br.com.androidpro.olamundo

A configuração do **Project location** será padrão para um local na pasta chamada `AndroidStudioProjects`, localizada no seu diretório home, e pode ser alterada clicando no botão à direita do campo de texto que contém a definição do caminho atual.

Clique em **Next** para continuar. Na próxima tela, habilite a opção de **Phone and Tablet** e definia a configuração do **SDK mínimo** para **API 15: Android 4.0.3**. A razão para a seleção de uma versão SDK mais antiga é que, dessa forma, asseguramos que o aplicativo será capaz de ser executado no maior número possível de dispositivos Android.

Quanto maior for a seleção do **SDK mínimo**, maior será a restrição do aplicativo a dispositivos Android mais recentes. Um gráfico útil pode ser visualizado clicando no link **Help me choose**. Ele descreve as várias versões do SDK e níveis de API disponíveis para uso, e o percentual de dispositivos Android no mercado onde o aplicativo será executado se o SDK for usado como nível mínimo. Em geral, só deve ser necessário selecionar um SDK mais recente quando essa versão tiver uma característica específica e necessária para o seu aplicativo. Para ajudar no processo de decisão, selecione um nível de API e o gráfico mostrará os recursos que são suportados nesse nível.

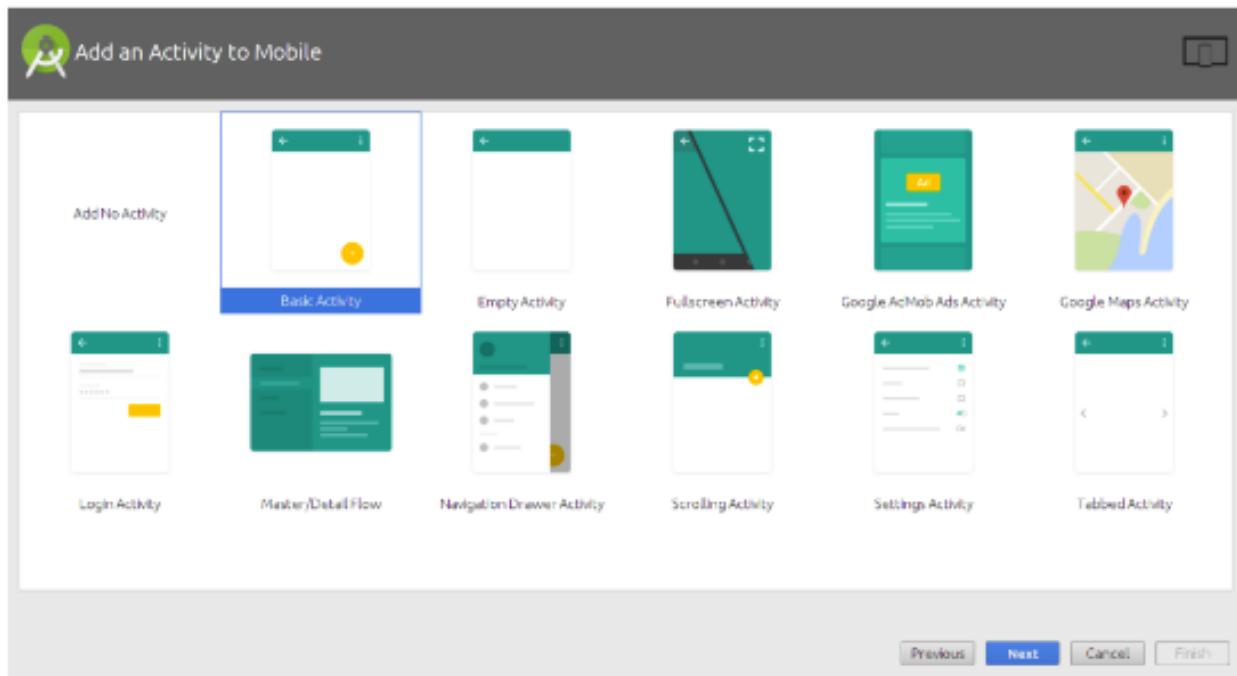


Figura 9 - Criando uma activity

Com a opção de Basic Activity selecionada, clique em Next. A Activity consistirá em um layout de tela de interface, que vamos chamar de `activity_main`, conforme apresentado na Figura 11, e com um recurso de menu chamado `menu_main`.

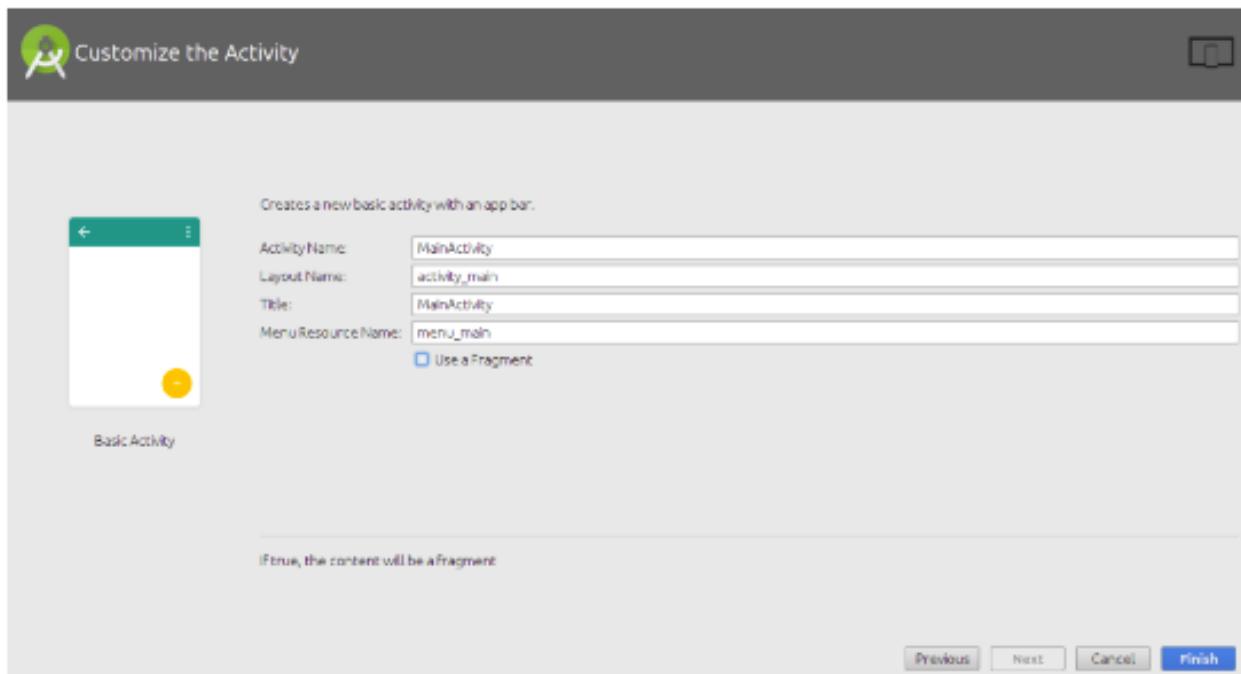


Figura 10 - Customizando a Activity

Por fim, clique em Finish para iniciar o processo de criação do projeto.

2.4 Modificando o Aplicativo de Exemplo

Até o momento, o Android Studio criou um projeto de exemplo de aplicativo mínimo e abriu a janela principal.

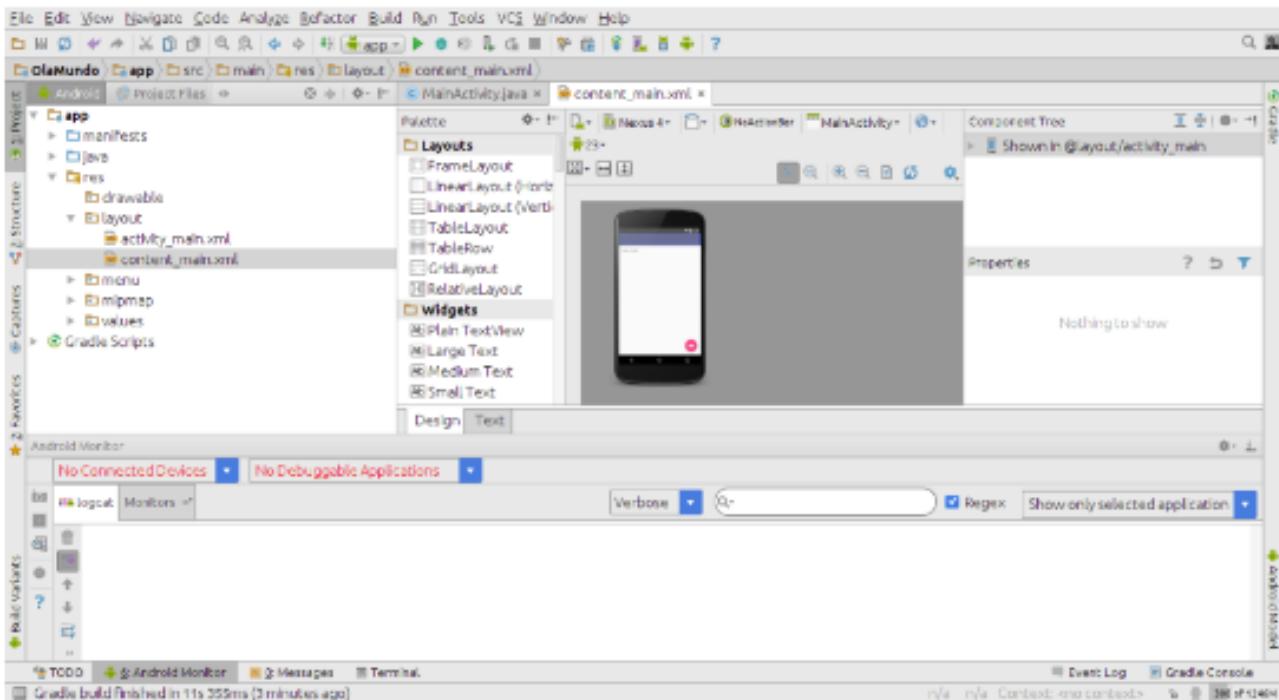


Figura 11 - Painel Principal do Android Studio

O projeto recém-criado e as referências aos arquivos associados estão listados no lado esquerdo da janela principal do projeto. Esse painel tem vários modos de visualizações diferentes e, por padrão, está no modo Android. Essa configuração é controlada pelo menu drop-down no topo do painel como destacado na figura abaixo.

Se o painel não estiver no modo Android, clique no menu indicado e mude para o modo Android.

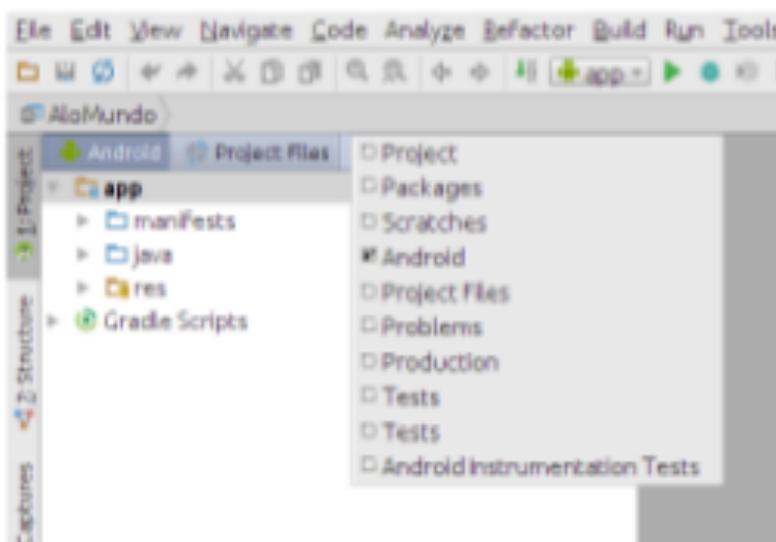


Figura 12 - Selecionando o Modo Android

O projeto de exemplo criado consiste em uma interface de usuário que contém uma **View** que diz "**Olá Mundo**" quando o aplicativo é executado.

O próximo passo que vou ensinar é a modificar a interface do usuário do nosso aplicativo para que ele mostre uma **View** de texto maior, com uma mensagem diferente da que é normalmente fornecida pelo Android Studio.

O design de interface da nossa **Activity** é armazenado em dois arquivos chamados **activity_main.xml** e **content_main.xml** que, por sua vez, encontram-se em **app > res > layout** na hierarquia de arquivo de projeto.

Nas novas versões do Android Studio, ele cria o layout inicial já com alguns princípios do **Material Design**. No arquivo **activity_main.xml**, encontramos a **ToolBar** e o **FloatingButton** e um include para o outro arquivo de layout **content_main.xml**. Nesse arquivo, fica a **View** que mostra a mensagem “Olá Mundo!”

Localize o arquivo **content_main.xml** como ilustrado na seguinte imagem:

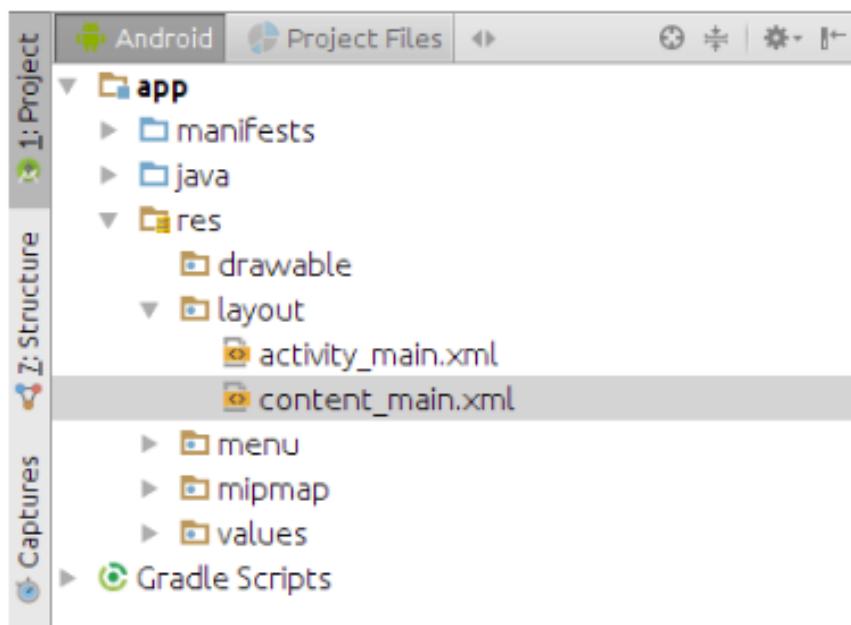


Figura 13 - Selezionando o Arquivo content_main.xml

Uma vez localizado, clique duas vezes no arquivo para abri-lo no editor do Android Studio.

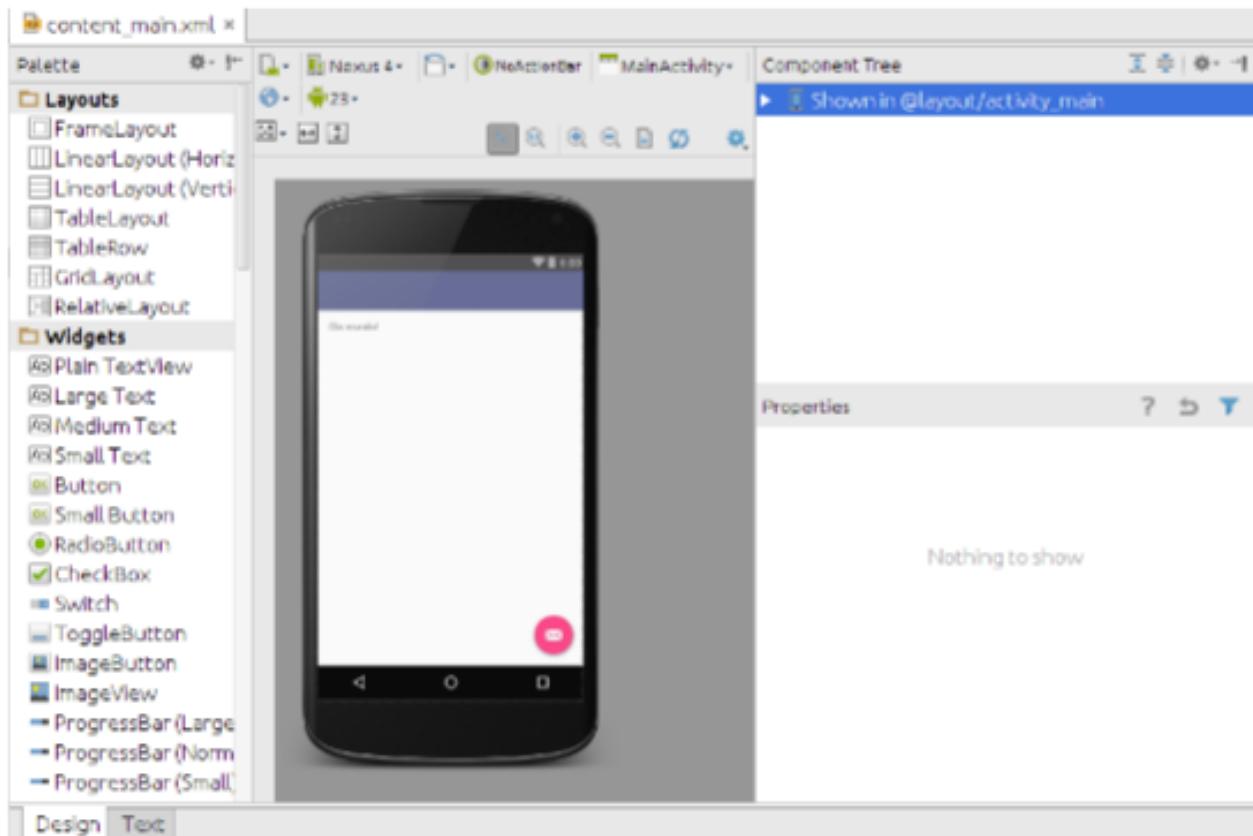


Figura 14 - Arquivo content_main.xml

Na barra de ferramentas na parte superior da janela **Design**, temos um menu definido como **Nexus 4**, que reflete a representação visual do dispositivo dentro do painel **Design**. Muitas outras opções de dispositivos estão disponíveis para seleção clicando neste menu.

Para alterar a orientação da representação do dispositivo entre retrato (portrait) e paisagem (landscape), basta usar o menu drop-down, à direita do menu de seleção do dispositivo.

Como podemos ver na tela do dispositivo, o layout já inclui uma view que exibe a mensagem “**Olá Mundo!**”, olhando para o lado esquerdo do painel vemos uma paleta contendo diferentes categorias de componentes de interface, que podem ser usados, como botões, campos de texto e etc.

Devemos lembrar, no entanto, que nem todos os componentes de interface são, obviamente, visíveis para o usuário. Uma dessas categorias consiste em **Layouts**. O Android suporta uma variedade de layouts que proporcionam diferentes níveis de controle sobre como os componentes visuais de interface são posicionados e gerenciados na tela. Apesar de ser difícil dizer apenas olhando para a representação visual da interface, o design atual foi criado usando um `RelativeLayout`. Isso pode ser confirmado olhando no painel **Component Tree** que, por padrão, está localizado no canto superior direito do painel de **Design** conforme abaixo.

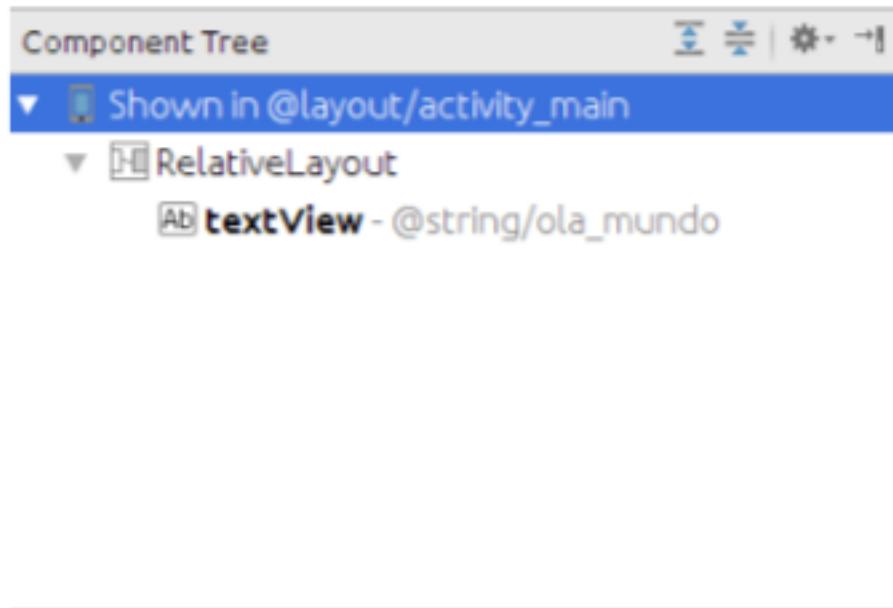


Figura 15 - Component Tree

Como podemos ver a partir da hierarquia da **Component Tree**, a interface consiste de um pai **RelativeLayout**, com uma única filha na forma de um **TextView**.

O primeiro passo para modificar o aplicativo é excluir o componente **TextView** do design. Comece clicando no **TextView**, dentro da interface de usuário, para que ele apareça com uma borda azul em torno dele. Uma vez selecionado, pressione a tecla Excluir no seu teclado para remover o objeto do layout.

No painel **Palette**, localize a categoria **Widgets**. Clique e arraste o **Large Text** e solte-o no centro da interface, quando as linhas do marcador verde aparecerem para indicar o centro da tela:

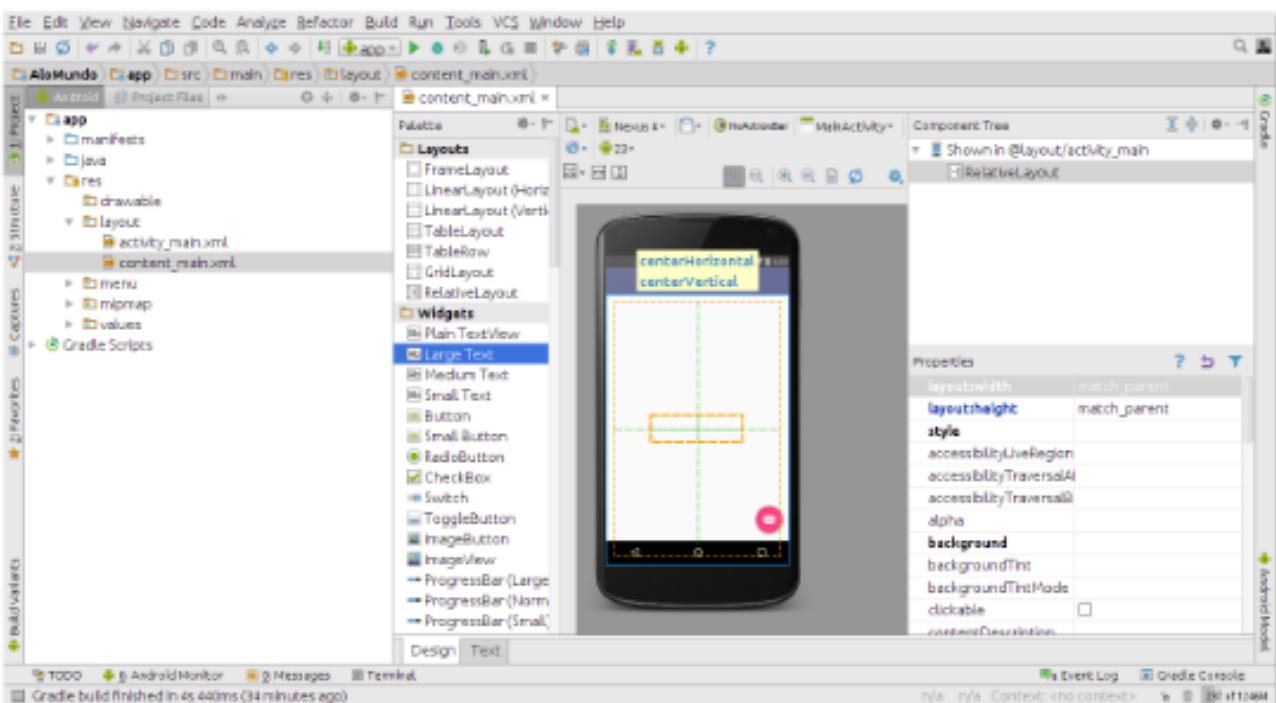


Figura 16 - Posicionando o Large Text

O próximo passo é mudar o texto exibido atualmente pelo componente **TextView**. Clique duas vezes sobre o objeto no layout do projeto para exibir o painel de edição de texto e id conforme ilustrado abaixo. Dentro do painel, altere a propriedade de texto a partir de “Large Text” para “Bem-vindo ao Android Studio”.

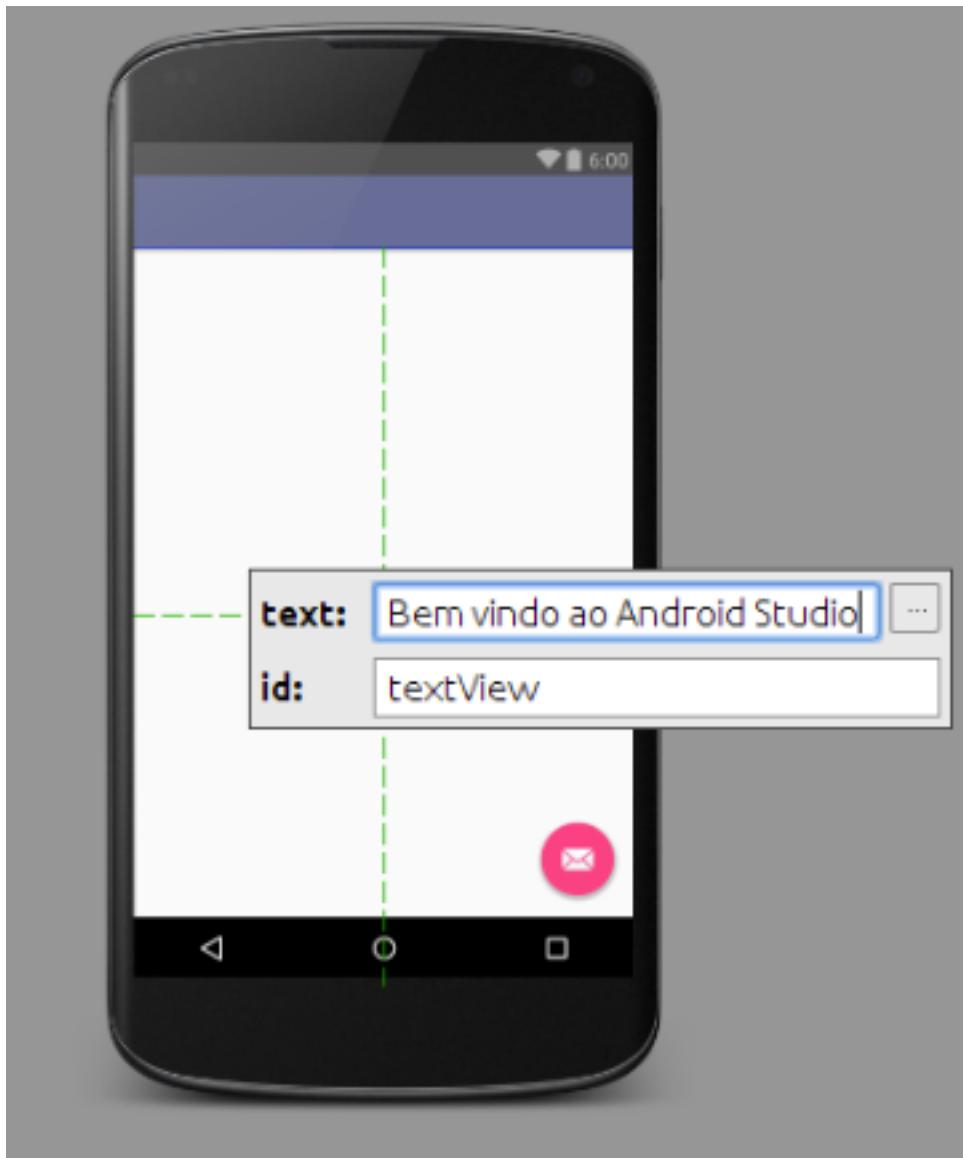


Figura 17 - Alterando o TextView

É importante explicar a lâmpada ao lado do objeto **TextView** no layout (que pode ser vista na Figura 19). Ela indica um possível problema, e recomenda algumas soluções. Ao clicar no ícone, neste caso, ele nos informa que o problema é o seguinte:

[I18N] Hardcoded string “Bem vindo ao Android Studio”, should use @string resource

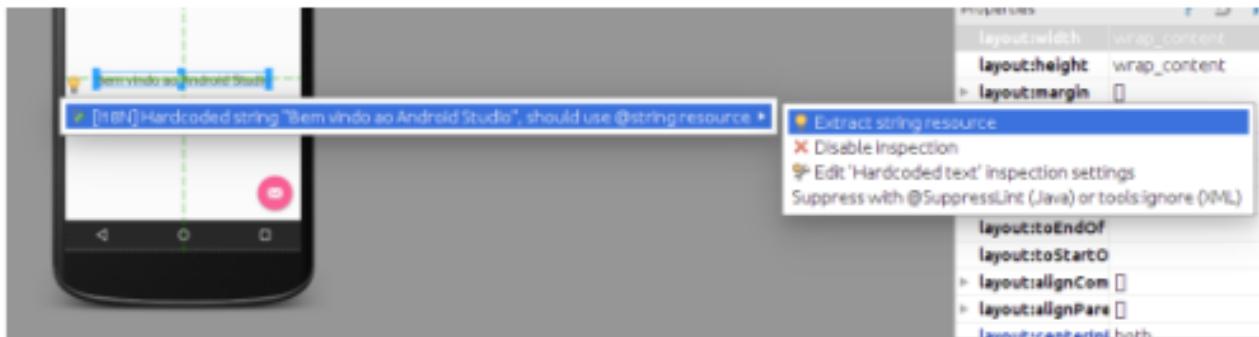


Figura 18 - Analisando um Possível Problema

Essa mensagem de **I18N** está nos dizendo que existe um potencial problema no que diz respeito à internacionalização do projecto (o nome "**I18N**" vem do fato de que a palavra "internationalização" começa com um "I", termina com um "N" e, na tradução em inglês, tem 18 letras entre uma e outra).

O alerta está nos lembrando que quando estamos desenvolvendo aplicativos Android, atributos e valores como strings devem ser armazenados sob a forma de recursos sempre que possível. Fazer isso permite que alterações na aparência do aplicativo possam ser feitas através da modificação dos arquivos de recursos, em vez de alterar o código-fonte da aplicação. Isso pode ser muito valioso ao traduzirmos uma interface de usuário para uma língua diferente. Se todo o texto de uma interface estiver contido em um único arquivo de recurso, por exemplo, esse arquivo pode ser enviado a um tradutor, esse irá executar o trabalho de tradução e retornar o arquivo traduzido para a inclusão no aplicativo. Isso permite que vários idiomas sejam selecionados, sem a necessidade de qualquer alteração de código-fonte. Neste exemplo, vamos criar um novo recurso chamado **benvindo** e atribuir a ele a **String** "**Bem-vindo ao Android Studio**".

A partir do menu, selecione a opção **Extract string resource** para exibir o alerta **Extract Resource**. Nesse alerta, insira “**bemvindo**” no campo Resource name antes de clicar em **OK**. A String está agora armazenada como um recurso no arquivo **app > res > values > strings.xml**.

2.5 Visualização de Layout

Até agora, o layout só foi visualizado em uma representação do dispositivo Nexus 4. Como discutido anteriormente, o layout pode ser testado para outros dispositivos, através de seleções de menu do dispositivo na barra de ferramentas do outro lado da margem superior do painel Designer.

Outra opção útil fornecida por este menu é a **Visualizar Todos os Tamanhos de Tela** que, quando selecionada, mostra o layout em todas as configurações de dispositivos atuais.

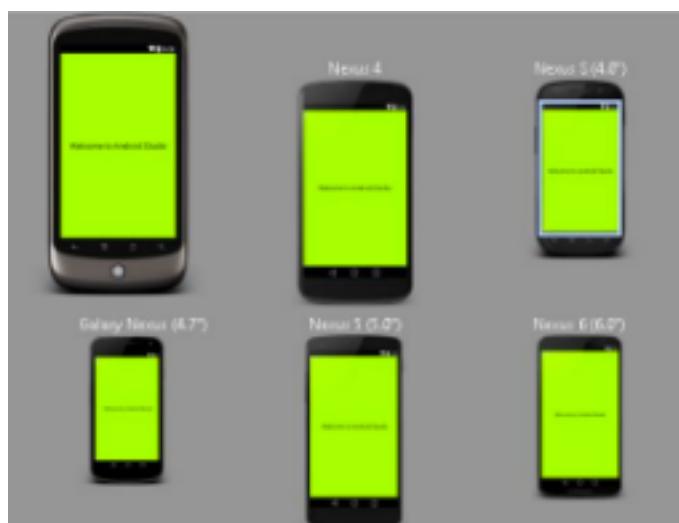


Figura 19 - Visualizar Todos os Tamanhos de Tela

Para reverter para um layout de pré-visualização único, selecione o menu do dispositivo, mais uma vez, desta vez escolhendo a opção Remover Previews.

2.6 Resumo

Embora não seja nada excessivamente complexo, uma série de passos estão envolvidos na criação de um ambiente de desenvolvimento Android. Ao executar essas etapas, vale a pena trabalhar em um exemplo simples, para garantir que o ambiente esteja corretamente instalado e configurado.

Nesse breve capítulo, nós criamos um app simples e, em seguida, usamos a ferramenta de design do Android Studio para modificar o layout da interface. Assim, exploramos a importância do uso de recursos, em particular no caso de valores de String, e brevemente sobre o tema layouts. Finalmente, olhamos para o XML que é usado para armazenar os designs de interface de aplicativos Android.

Embora seja útil para visualizar um layout a partir da ferramenta de design do Android Studio, não há substituto ao teste de um aplicativo através da compilação e execução nos emuladores. Logo mais, vamos criar um emulador para fins de teste e poderemos falar melhor sobre o assunto.

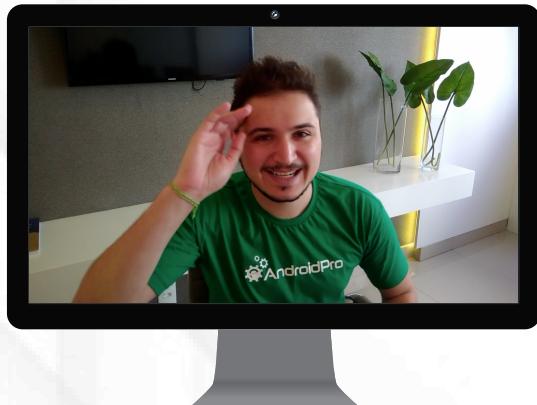
..... AULA ONLINE GRATUITA!

OS 5 PASSOS OBRIGATÓRIOS

PARA VOCÊ SER UM **DESENVOLVEDOR ANDROID**
PROFISSIONAL E INDEPENDENTE!

Porque você precisa participar?

- 1 Para entender os **erros** que fazem você desistir e se frustrar
- 2 Para descobrir um **método COMPROVADO** e **ÚNICO** para desenvolver praticamente qualquer tipo de aplicativo Android
- 3 Para saber quais as **habilidades exatas**, além da programação, que vão te transformar em um profissional
- 4 Para aprender a **criar oportunidades de trabalho** e ganhar dinheiro com desenvolvimento Android desde já!



VAGAS *LIMITADAS!*

INSCREVA-SE AQUI!

3. Um Tour pela Interface do Android Studio

Antes de executarmos o aplicativo exemplo que criamos no capítulo anterior, precisamos entender uma série de aspectos da interface de usuário do Android Studio, para que possamos aproveitar melhor a IDE.

O Android Studio é um ambiente de desenvolvimento rico e poderoso. Quanto mais você investir o tempo necessário para se familiarizar com o layout e organização da interface da IDE, mais fácil e rápido será entender outros recursos que ela oferece.

Sendo assim, vamos conhecer o ambiente Android Studio.

3.1 A Tela de Boas Vindas

A tela de boas-vindas é exibida a qualquer momento que o Android Studio for executado sem projetos abertos (projetos abertos podem ser fechados a qualquer momento, selecionando as opções de menu File > Close Project). Se o Android Studio for previamente fechado enquanto um projeto ainda estiver aberto, a ferramenta irá "pular" a tela de boas-vindas da próxima vez que for iniciada, abrindo automaticamente o projeto previamente ativo.

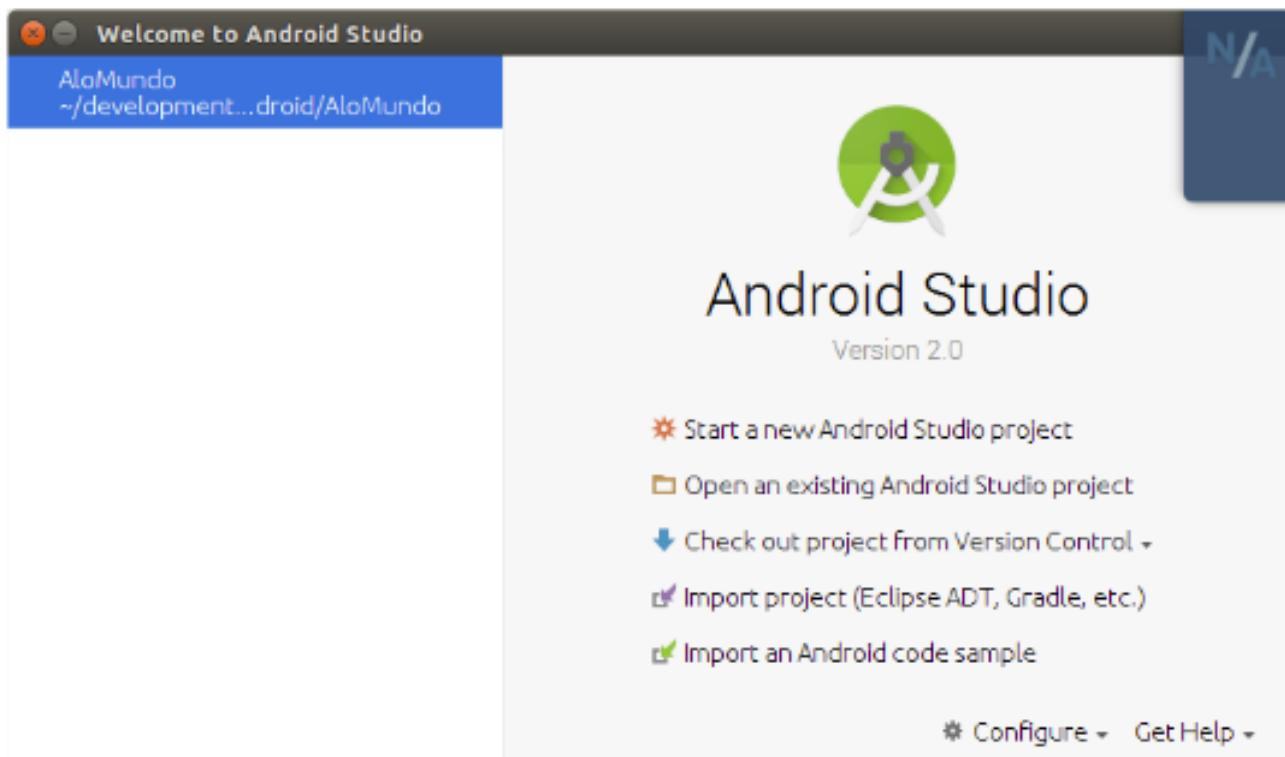


Figura 20 - Tela de boas-vindas exibindo projetos recentes

Além de uma lista de projetos recentes, o menu a esquerda oferece várias opções para a execução de tarefas como abrir, criar e importar projetos, juntamente com o acesso a projetos atualmente sob controle de versão. A opção Configure fornece acesso ao SDK Manager e a várias opções configurações, além disso, também inclui uma opção para verificar se há atualizações para o Android Studio disponíveis para download.

3.2 Painel Principal

Quando um novo projeto é criado, ou um já existe um aberto, o painel principal do Android Studio vai aparecer. Quando vários projetos estão abertos simultaneamente, cada um será atribuído a

sua própria janela principal. A configuração exata do painel varia de acordo com quais ferramentas e painéis foram exibidos na última vez que o projeto foi aberto, mas será tipicamente semelhante a seguinte imagem:

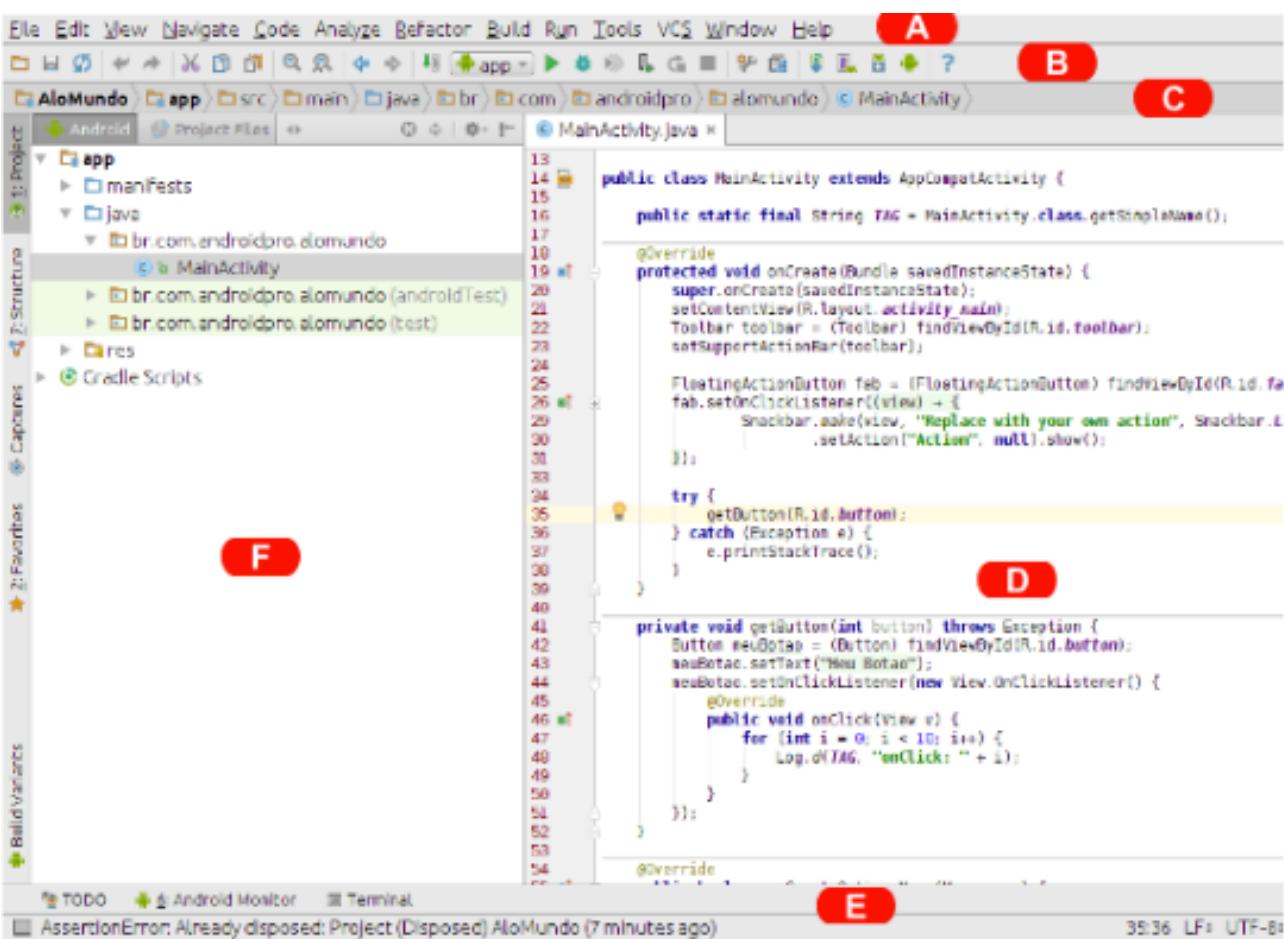


Figura 21 - Painel Principal

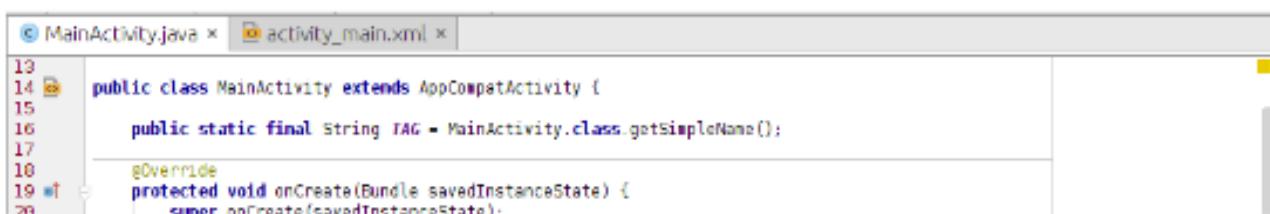
Os vários elementos da janela principal podem ser resumidos da seguinte forma:

A - Barra de Menu: Contém uma série de menus para a execução de tarefas dentro do ambiente Android Studio.

B - Barra de Ferramentas: Uma seleção de atalhos para ações executadas com frequência. Os botões da barra de ferramentas fornecem acesso mais rápido a um seletivo conjunto de ações da barra de menu. A barra de ferramentas pode ser personalizada clicando com o botão direito na barra e selecionando Customize Menus and Toolbars.

C - Barra de Navegação: A barra de navegação fornece uma maneira conveniente para se mover entre os arquivos e pastas que compõem o projeto. Clicando sobre um elemento na barra de navegação, aparece um submenu listando as subpastas e arquivos naquele local, prontos para seleção. Isso fornece uma alternativa à janela da ferramenta Project.

D - Janela do Editor: A janela do editor exibe o conteúdo do arquivo no qual o desenvolvedor está trabalhando atualmente. Ao editar o código, por exemplo, o editor de código aparecerá. Quando estiver trabalhando em um arquivo de layout de interface de usuário, por outro lado, a ferramenta de Design de interface é que vai aparecer. Quando vários arquivos são abertos, cada arquivo é representado por um separador localizado ao longo da margem superior do editor tal como mostrado logo abaixo:



The screenshot shows the Android Studio code editor with two tabs open: 'MainActivity.java' and 'activity_main.xml'. The 'MainActivity.java' tab is active, displaying Java code for an AppCompatActivity. The code includes imports for AppCompatActivity and AppCompatViewGroup, and defines a static final string TAG. It also contains an onCreate method override. The 'activity_main.xml' tab is visible but inactive. The editor interface includes a top navigation bar with icons for file operations like New, Open, Save, and Run, along with tabs for recent projects and tools like the Device Manager and Logcat.

Figura 22 - Janela do Editor

E - Barra de Status: A barra de status exibe mensagens informativas sobre o projeto e as activities do Android Studio junto ao botão de menu Ferramentas, localizado no canto esquerdo. Ao

posicionar o cursor sobre os itens na barra de status você terá uma descrição do campo. Muitos campos são interativos, permitindo que o usuário clique para executar tarefas ou obtenha informações de status mais detalhadas.

F - Janela Ferramentas de Projeto: A janela de ferramentas de projeto fornece uma visão hierárquica da estrutura do arquivo de projeto, permitindo a navegação para arquivos e pastas específicos para serem abertos. O menu drop-down na barra de ferramentas pode ser usado para exibir o projeto de maneiras diferentes. A configuração padrão é a visualização Android que é o modo utilizado principalmente no restante deste livro.

A janela de ferramentas de projeto é apenas uma de uma série de janelas de ferramentas disponíveis no ambiente do Android Studio.

3.3 As Janelas

Além de janela de ferramentas do projeto, o Android Studio inclui, também, uma série de outras janelas que, quando ativadas, são exibidas ao longo do rodapé e nas laterais da janela principal. O menu de acesso rápido da janela de ferramentas pode ser acessado ao posicionar o ponteiro do mouse sobre o botão localizado no canto inferior esquerdo da barra de status (abaixo) sem clicar o botão do mouse.

Selecionar um item do menu de acesso rápido fará com que a janela da ferramenta correspondente apareça dentro da janela principal.

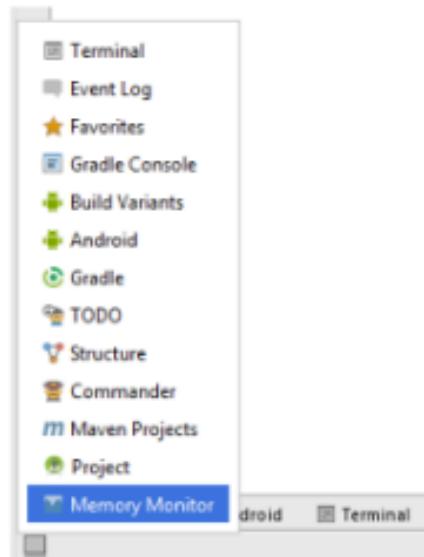


Figura 23 - Menu de Acesso Rápido

Selecionar um item do menu de acesso rápido fará com que a janela da ferramenta correspondente apareça dentro da janela principal.

De outra forma, um conjunto de ferramentas pode ser exibido clicando no ícone de menu de acesso rápido na barra de status. Essas barras aparecem ao longo das margens esquerda, direita e inferior da janela principal (como indicado pelas setas na figura a seguir), e contêm botões para mostrar e ocultar cada uma das janelas de ferramentas. Quando as barras da janela de ferramentas são exibidas, um segundo clique no botão na barra de status irá escondê-las.

Ao clicar em um botão, veremos a janela de ferramenta correspondente, enquanto um segundo clique irá esconder a janela. Botões prefixados com um número (por exemplo, 1: Project) indicam que a janela da ferramenta também pode ser exibida pressionando a tecla **Alt** no teclado (ou a tecla **Command** para Mac OS X) junto ao número correspondente.

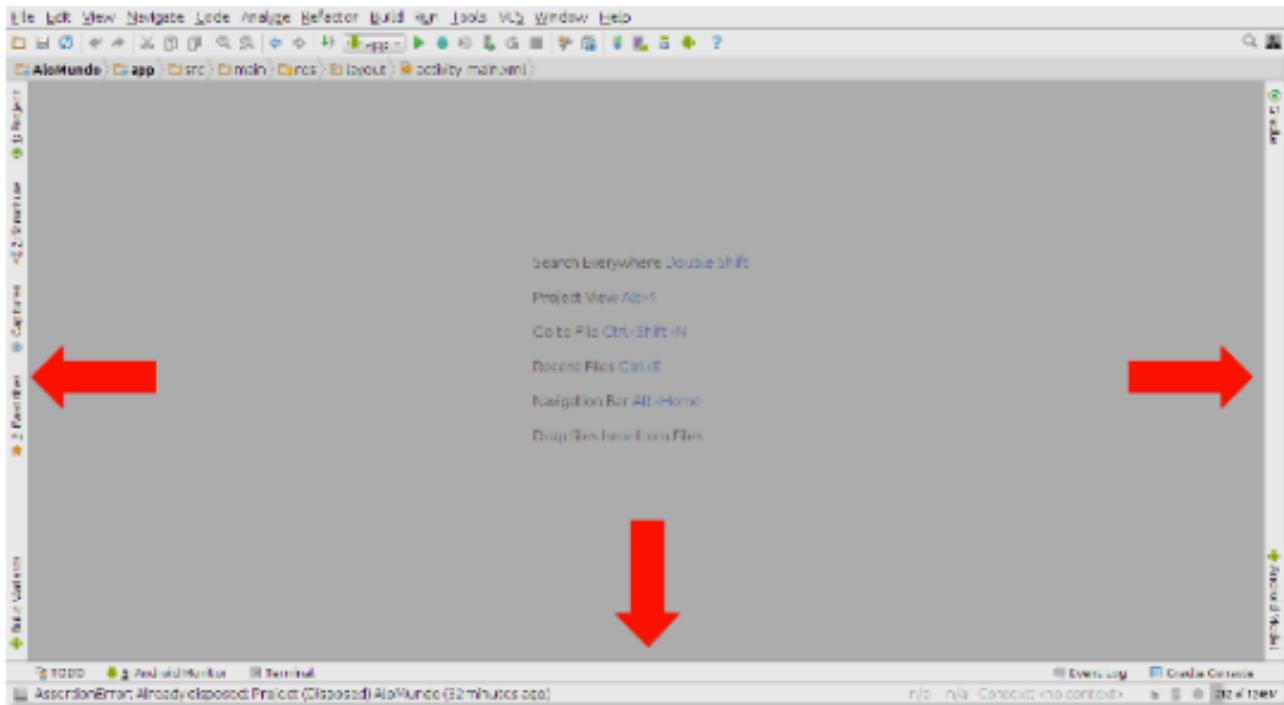


Figura 24 - Localização das barras de status

Cada janela de ferramenta tem a sua própria barra de ferramentas na margem superior. Os botões dentro dessas barras podem variar de uma ferramenta para a outra, mesmo que todas as janelas de ferramentas contenham a opção de configurações, representada pelo ícone de engrenagem, que permite que vários aspectos da janela sejam alterados. A figura abaixo mostra o menu de configurações para a janela de ferramenta do projeto. As opções estão disponíveis para, por exemplo, desencaixar uma janela para permitir que esta flutue fora dos limites da janela principal do Android Studio.

Todas as janelas também incluem um botão mais à direita na barra de ferramentas fornecendo uma maneira adicional de ocultar a janela da ferramenta de visualização. Podemos executar uma busca dos itens dentro de uma janela de ferramenta simplesmente

focando nessa janela, clicando nela e digitando o termo de pesquisa (por exemplo, o nome de um arquivo na janela da ferramenta de projeto). Uma caixa de pesquisa aparecerá na barra de ferramentas da janela e os itens que corresponderem à pesquisa serão destacados.

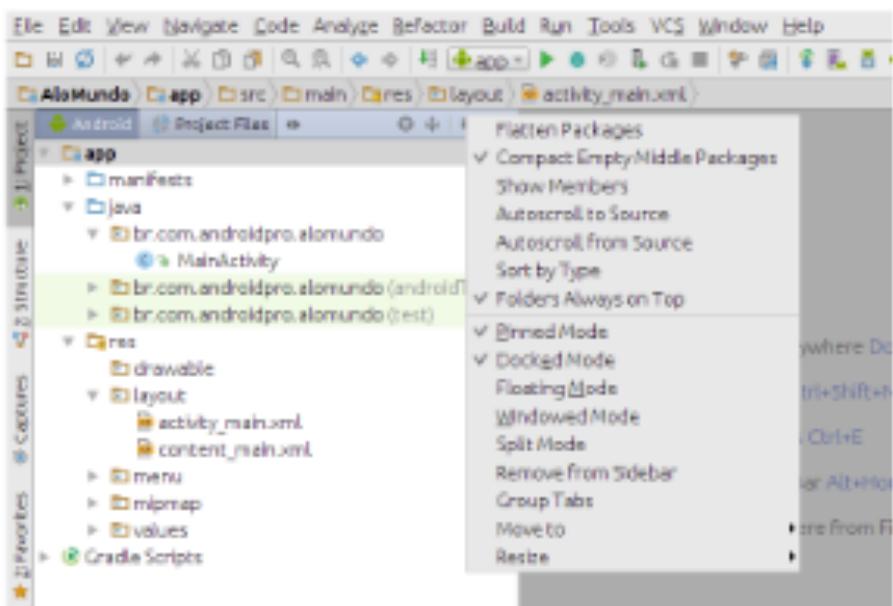


Figura 25 - Barra de Ferramentas

O Android Studio oferece um grande volume de janelas de ferramenta, as mais utilizadas são as seguintes:

Project: Uma visualização que fornece uma visão geral da estrutura de arquivos que compõe o projeto, permitindo uma navegação rápida entre os arquivos. Geralmente, um duplo clique sobre um arquivo fará com que esse arquivo seja carregado para a ferramenta de edição adequada.

Structure: A ferramenta Structure fornece uma visão de alto nível da estrutura das classes atualmente exibidas no editor. Essa informação inclui uma lista de itens, tais como classes, métodos e

variáveis no arquivo. A seleção de um item da lista vai te levar para esse local no arquivo na janela do editor.

Captures: A janela da ferramenta Captures fornece acesso a arquivos de dados de desempenho gerados pelas ferramentas de monitoramento da janela Android Monitor.

Favorites: Uma variedade de itens do projeto podem ser adicionada à lista de favoritos. Ao clicar com o botão direito em um arquivo na visualização de projeto, por exemplo, fornece acesso a opção Add to Favorites. Do mesmo modo, um método em um arquivo de recursos pode ser adicionado como um favorito ao clicar com o botão direito do mouse sobre ele na janela de ferramentas Structure. Qualquer coisa adicionada a uma lista de favoritos pode ser acessada através desta janela de ferramenta Favorites.

Build Variants: A janela Build Variants fornece uma maneira rápida de configurar diferentes tipos de compilação para o projeto atual (por exemplo, diferentes versões para debug e liberação de versões do aplicativo, ou várias compilações para atingir diferentes categorias de dispositivos).

TODO: Como o nome sugere (to do), esta ferramenta fornece um lugar para os itens que ainda têm de ser concluídos no projeto. O Android Studio compila essa lista verificando os arquivos que compõem o projeto olhando comentários que correspondem a padrões especificados como TODO. Esses padrões podem ser revisados e alterados, selecionando File > Settings e navegando pela página TODO listada no editor.

Messages: Os dados da janela de Messages são gerados a partir

do sistema de compilação Gradle, e podem ser úteis para identificar as causas de problemas na compilação de projetos.

Android Monitor: Esse recurso possibilita o acesso ao sistema de debug do Android. Dentro desta janela, podemos executar tarefas básicas de debug como monitorar a saída de log de um aplicativo em execução, tirar screenshots e vídeos do aplicativo, e parar um processo.

Android Model: A janela Android Model exibe uma lista completa das configurações do seu projeto. Estes podem variar de configurações mais óbvias, tais como a versão do SDK alvo, para coisas mais escondidas tais como regras da configuração de compilação.

Terminal: É uma janela de terminal do sistema na qual o Android Studio está em execução. No Windows essa interface é o prompt de comando, enquanto que no Linux e Mac OS X é um acesso ao terminal de comandos.

Run: Essa ferramenta torna-se disponível quando um aplicativo está em execução e fornece uma visualização dos resultados da execução, junto a opções para parar ou reiniciar um processo em execução. Se um aplicativo falha na instalação e execução em um dispositivo ou emulador, essa janela, geralmente, vai fornecer informações de diagnóstico relativas ao problema.

Event Log: A janela Evento Log exibe mensagens relacionadas a eventos e atividades realizadas no Android Studio. A compilação bem-sucedida de um projeto, por exemplo, ou o fato de que um app está sendo executado serão mostrados dentro dessa janela.

Gradle Console: O Gradle Console é usado para exibir todas as saídas do sistema Gradle enquanto os projetos são compilados dentro do Android Studio. Isso inclui informações sobre o sucesso ou insucesso do processo de desenvolvimento, assim como avisos ou detalhes de qualquer erro.

Gradle: A janela do Gradle mostra as tarefas do Gradle que compõem a configuração do projeto. Essa janela lista as tarefas que estão envolvidas na compilação dos vários elementos do projeto em um aplicativo Android. Clique com o botão direito do mouse em uma tarefa e selecione a opção **Open Gradle Config** para abrir o arquivo no editor.

3.4 Switcher e Navegação por Arquivos Recentes

Um mecanismo útil para navegar dentro da janela principal do Android Studio envolve o uso do Switcher. Acessado via atalho de teclado **Ctrl + Tab**, o switcher aparece como um painel, listando ambas as janelas de ferramentas e arquivos abertos no momento.

Uma vez aberto, o switcher permanecerá visível durante o tempo que a tecla Ctrl permanecer pressionada. Ao pressionar a tecla Tab repetidamente enquanto a tecla Ctrl estiver pressionada, passaremos pelas várias opções de seleção, enquanto soltar a tecla Ctrl faz com que o item destacado naquele momento seja selecionado e exibido dentro da janela principal.

Além do switcher, a navegação de arquivos abertos recentemente é fornecida pelo painel Recent Files. Que pode ser acessado através do atalho de teclado **Ctrl + E** (**Cmd + E** no Mac OS X).

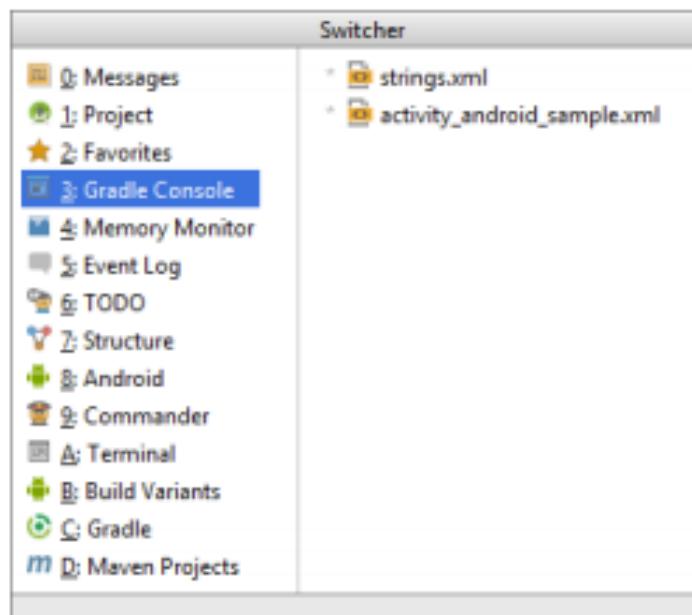


Figura 26 - Switcher

Uma vez exibido, podemos usar o cursor do mouse para selecionar uma opção ou podemos usar as setas do teclado para percorrermos as opções de arquivos e janelas de ferramentas.

Para selecionar o item destacado basta pressionarmos o **Enter**.

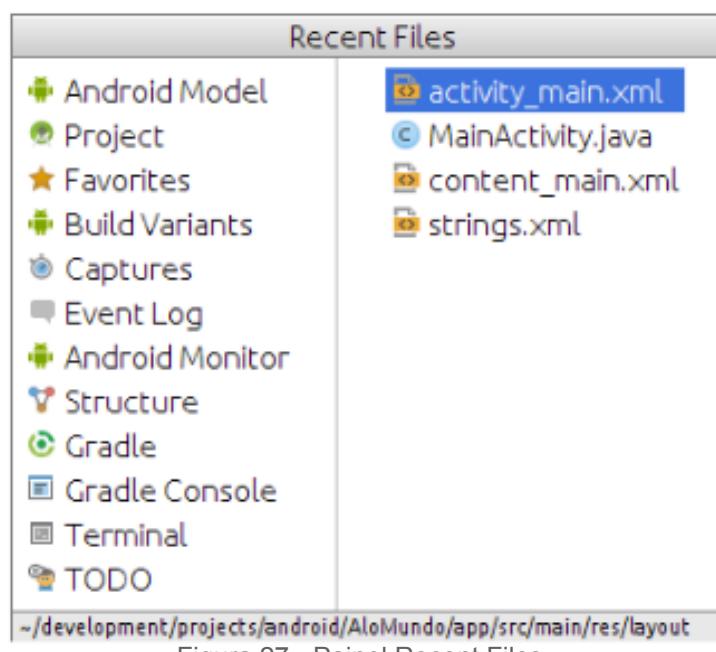


Figura 27 - Painel Recent Files

3.5 Mudando o Tema do Android Studio

O tema do ambiente do Android Studio pode ser alterado tanto a partir da tela de boas vindas usando o **Configure > Settings**, quanto através do **File > Settings** no menu da janela principal.

Uma vez que a janela de configuração for exibida, selecione a opção **Appearance** no painel do lado esquerdo e, em seguida, altere a configuração do menu **Theme** antes de clicar no botão **Apply**. A figura seguir mostra um exemplo do Android Studio com o tema Darcula selecionado:

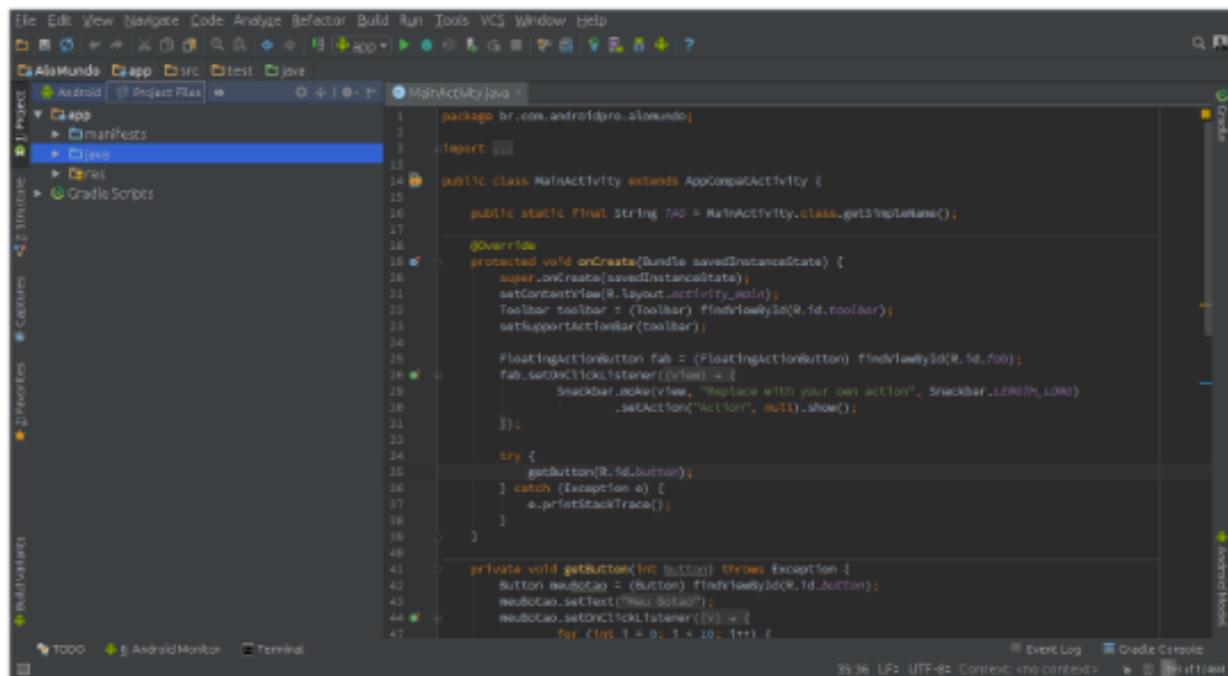


Figura 28 - Usando o Tema Darcula no Android Studio

3.6 Resumo

Os principais elementos de ambiente do Android Studio consistem

na tela de boas vindas e o painel principal. Para cada projeto aberto é atribuído o seu próprio painel principal, que, por sua vez, consiste em uma barra de menu, barra de ferramentas de edição e área de design, barra de status e uma coleção de janelas de ferramentas. As janelas de ferramentas aparecem nas laterais e margem inferiores da janela principal, e pode ser acessadas usando o menu de acesso rápido localizado na barra de status, ou através das barras das janelas de ferramentas.

No Android Studio há poucas ações que não podem ser acionadas através de um atalho de teclado. Um mapa de teclado de atalhos de teclado padrão pode ser acessado a qualquer momento a partir da janela principal da ferramenta.

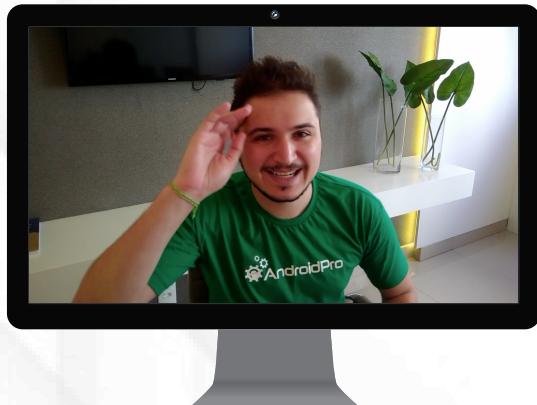
..... AULA ONLINE GRATUITA!

OS 5 PASSOS OBRIGATÓRIOS

PARA VOCÊ SER UM **DESENVOLVEDOR ANDROID**
PROFISSIONAL E INDEPENDENTE!

Porque você precisa participar?

- 1 Para entender os **erros** que fazem você desistir e se frustrar
- 2 Para descobrir um **método COMPROVADO** e **ÚNICO** para desenvolver praticamente qualquer tipo de aplicativo Android
- 3 Para saber quais as **habilidades exatas**, além da programação, que vão te transformar em um profissional
- 4 Para aprender a **criar oportunidades de trabalho** e ganhar dinheiro com desenvolvimento Android desde já!



VAGAS **LIMITADAS!**

INSCREVA-SE AQUI!

4. Criando um Dispositivo Virtual Android (AVD)

No decorrer do desenvolvimento de aplicativos Android no Android Studio, precisaremos compilar e executar um aplicativo várias vezes. Um aplicativo Android pode ser testado através da instalação e execução em um dispositivo físico ou em um emulador AVD. Antes de usar um AVD, ele deve primeiro ser criado e configurado para coincidir com a especificação de um modelo de dispositivo específico. O objetivo deste capítulo, é trabalhar com as etapas envolvidas na criação de um dispositivo virtual usando o celular Nexus 5 como referência.

4.1 Sobre Dispositivos Virtuais Android (AVDs)

Os AVDs são basicamente emuladores que permitem que os aplicativos Android sejam testados sem a necessidade de instalar o aplicativo em um dispositivo físico baseado em Android. Um AVD pode ser configurado para emular uma variedade de recursos de hardware, incluindo opções como tamanho da tela, capacidade de memória e a presença ou não de recursos como câmera, suporte de navegação GPS ou um acelerômetro. Como parte da instalação padrão do Android Studio, vários templates de emuladores são instalados, permitindo que os AVDs sejam configurados em um grande número de dispositivos diferentes. Os templates adicionais podem ser utilizados para criar qualquer dispositivo Android físico, por propriedades específicas como o tipo

de processador, capacidade de memória e tamanho e densidade da tela. Verifique a documentação do desenvolvedor on-line para o seu dispositivo para saber se as definições do emulador estão disponíveis para download e instalação no ambiente AVD.

Quando iniciado, um AVD aparecerá como uma janela que contém um dispositivo Android emulado. A figura abaixo, por exemplo, mostra um AVD configurado para emular o dispositivo Google Nexus 5.

Novos AVDs podem ser criados e gerenciados através do **Android Virtual Device Manager**, que pode ser usado tanto no modo linha de comando, como com uma interface gráfica mais fácil de ser usada.

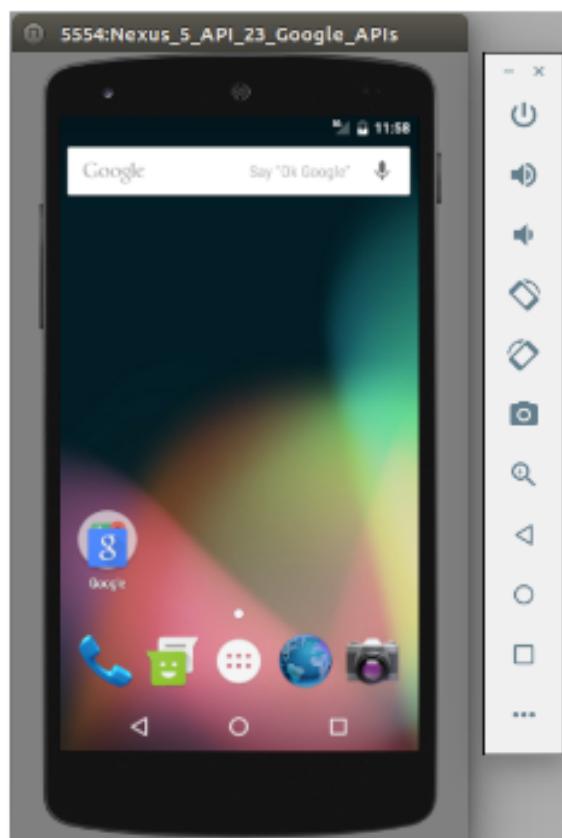


Figura 29 - AVD emulando o Google Nexus 5

4.2 Criando um novo AVD

Para testar o comportamento de um aplicativo, será necessário criar um AVD para uma configuração de dispositivo Android específica.

Para criar um novo AVD, o primeiro passo é iniciar o AVD Manager. Isso pode ser feito de dentro do ambiente do Android Studio ao selecionarmos **Tools > Android > AVD Manager**, a partir da janela principal.

Uma vez iniciada, a ferramenta se parece com a imagem abaixo. Considerando uma nova instalação do Android SDK, não haverá AVDs listados atualmente:

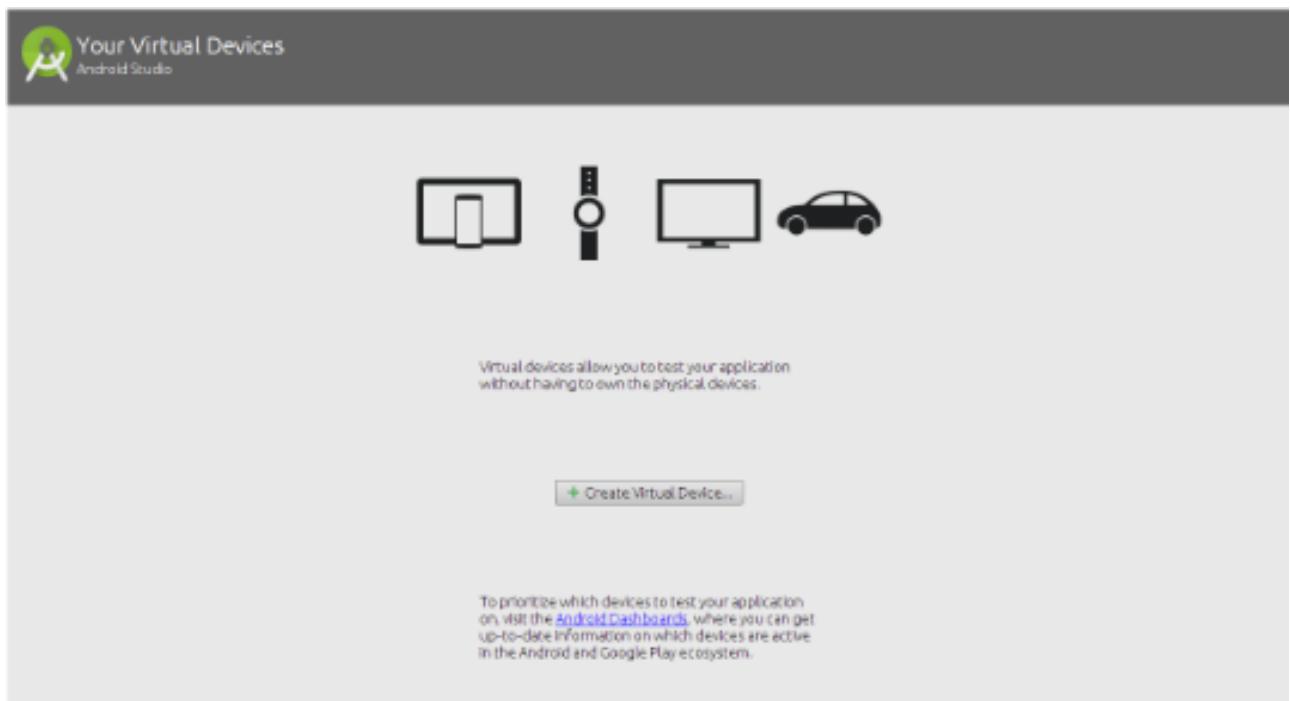


Figura 30 - Iniciando o AVD Manager

Comece o processo de criação do AVD clicando no botão **Create Virtual Device**, para abrir a janela **Virtual Device Configuration**.

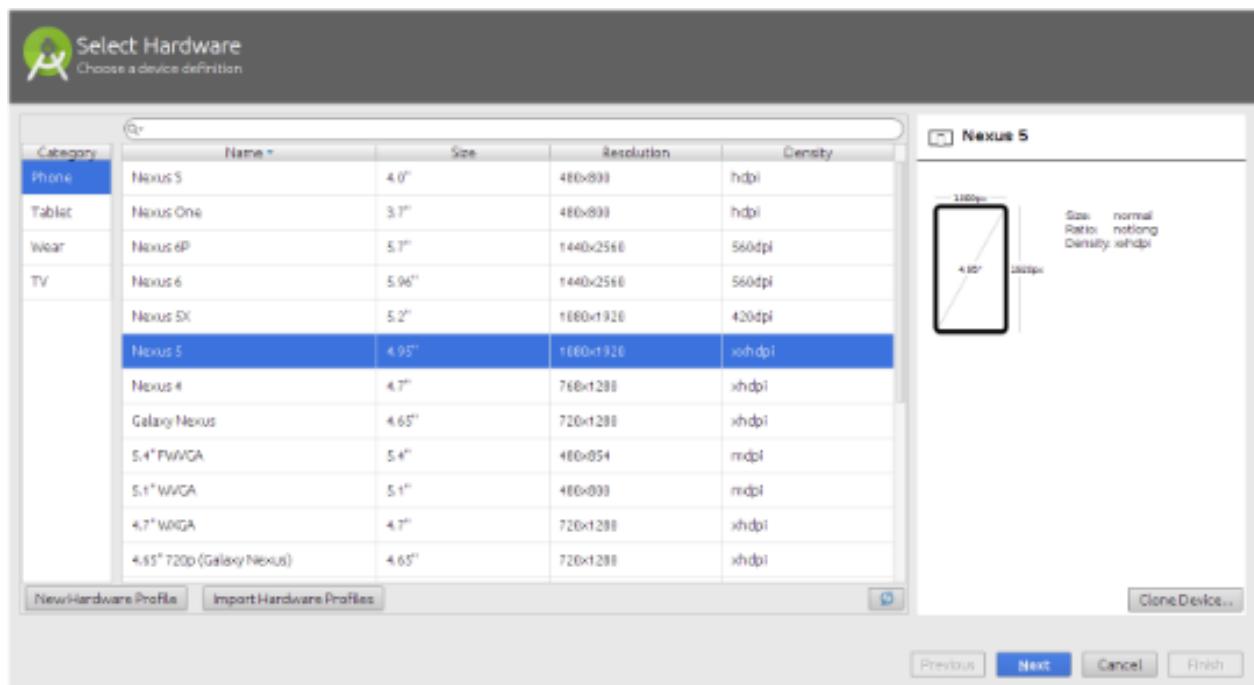


Figura 31 - Painel de seleção dos dispositivos

Dentro da janela, execute os seguintes passos para criar um emulador compatível ao Nexus 5:

1. No painel de categorias, selecione a opção **Phone** para obtermos a lista de modelos de celulares Android disponíveis.
2. Selecione a opção **Nexus 5** e clique em **Next**.
3. Na tela **System Image**, selecione a versão mais recente do Android (no momento é o Android 6, Marshmallow nível API 23) para o x86 ABI. Clique em **Next** para continuar.
4. Digite um nome (por exemplo **Nexus 5 API 23**) no campo **AVD Name**.
5. Clique em **Finish** para criar o AVD.

Com o AVD criado, podemos fechar o AVD Manager. Se futuras modificações forem necessárias, simplesmente abra o AVD Manager, selecione o AVD da lista e clique no ícone de lápis, na coluna Actions da linha dispositivo no AVD Manager.

4.3 Iniciando o Emulador

Para executar o emulador AVD recém-criado, basta selecionar o emulador no AVD Manager e clicar no botão de iniciar (o triângulo verde na coluna Actions). O emulador irá aparecer em uma nova janela e, depois de um curto período de tempo, o logotipo "Android" aparecerá no centro da tela. A quantidade de tempo que leva para o emulador iniciar dependerá da configuração tanto do AVD quanto do sistema no qual ele está sendo executado.

Se o tempo de início do emulador no seu sistema for longo, deixe o emulador em execução. O sistema detecta que já está em execução e vai mostrá-lo quando os aplicativos forem executados, economizando mais tempo de inicialização.

O emulador padrão provavelmente vai aparecer no tamanho completo e na orientação retrato (portrait). Você pode alterar essas opções dentro do AVD Manager, basta selecionar o AVD criado e clicar no ícone de lápis na coluna Actions da linha dispositivo. Na tela de configuração, localize a seção **Startup size and orientation** e altere o menu Scale para **2DP on device = 1px on screen** para reduzir o tamanho do emulador na tela. Reinicie o emulador para ver a alteração. O tamanho do emulador também pode ser alterado dinamicamente, redimensionando a janela.

Para economizar tempo, na próxima seção deste capítulo deixe o emulador em execução antes de prosseguir.

4.4 Executando o aplicativo no AVD

Com o emulador AVD configurado, o exemplo de aplicativo **OlaMundo**, criado anteriormente, pode ser compilado e executado. Com o projeto **OlaMundo** carregado no Android Studio, basta clicar no botão de execução, representado por um triângulo verde localizado na barra de ferramentas do Android Studio, como mostrado na figura abaixo. Selecione **Run > Run 'app'**, ou use o atalho **Shift + F10**.

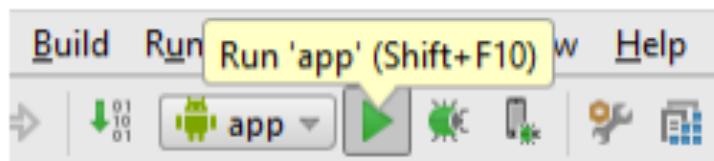


Figura 32 - Botão de Execução

Por padrão, o Android Studio exibe a janela para escolha do dispositivo. Isto nos dá a opção de executar o aplicativo em uma instância do AVD que já está em execução, ou de iniciar um novo AVD especificamente para esse aplicativo. A figura a seguir mostra o AVD Nexus 5, criado anteriormente, como um dispositivo em execução. Com este dispositivo selecionado na caixa de diálogo, clique em **OK** para instalar e executar o aplicativo no emulador.

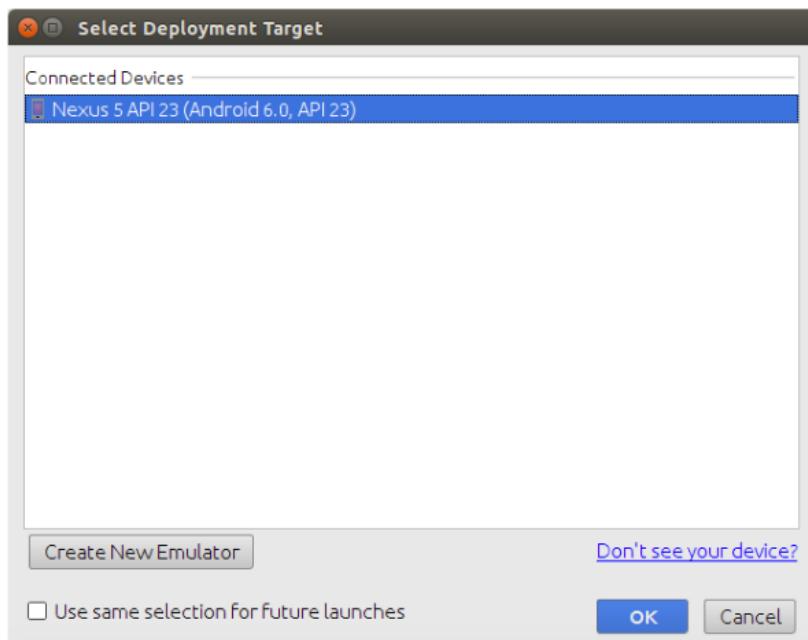


Figura 33 - Janela para escolha do dispositivo

Uma vez que o aplicativo esteja instalado e funcionando, a interface do usuário para a classe MainActivity vai aparecer dentro do emulador:

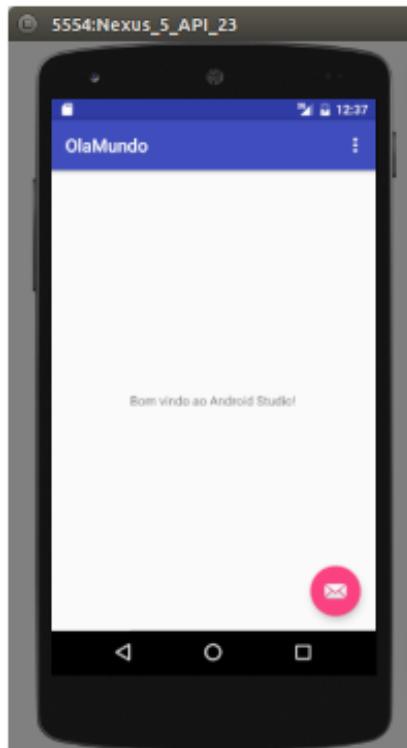


Figura 34 - Interface de usuário

No caso em que a Activity não iniciar automaticamente, verifique se o ícone do aplicativo apareceu entre os aplicativos no emulador. Se tiver, basta clicar sobre ele para iniciar o aplicativo. Uma vez que o processo de execução é iniciado, as janelas de ferramentas Run e Android vão ficar disponíveis. A janela da ferramenta Run irá exibir informações de diagnóstico sobre a instalação e execução do aplicativo.



Figura 35 - Janela da ferramenta Run

Se problemas forem encontrados durante o processo de inicialização, a ferramenta **Run** vai fornecer informações que poderão ajudar a encontrar a causa do problema.

Considerando que o aplicativo seja carregado para o emulador e executado conforme o esperado, verificamos que o ambiente de desenvolvimento Android está corretamente instalado e configurado.

4.5 Configurações de Run / Debug

Um projeto em particular pode ser configurado de tal forma que um dispositivo ou emulador específico seja utilizado automaticamente, cada vez que é executado no Android Studio.

Isso evita a necessidade de se fazer uma seleção a partir do seletor de dispositivos a cada vez que o aplicativo for executado. Para modificar a configuração **Run/Debug**, clique no botão à esquerda do botão de execução na barra de ferramentas e selecione **Edit Configurations**:

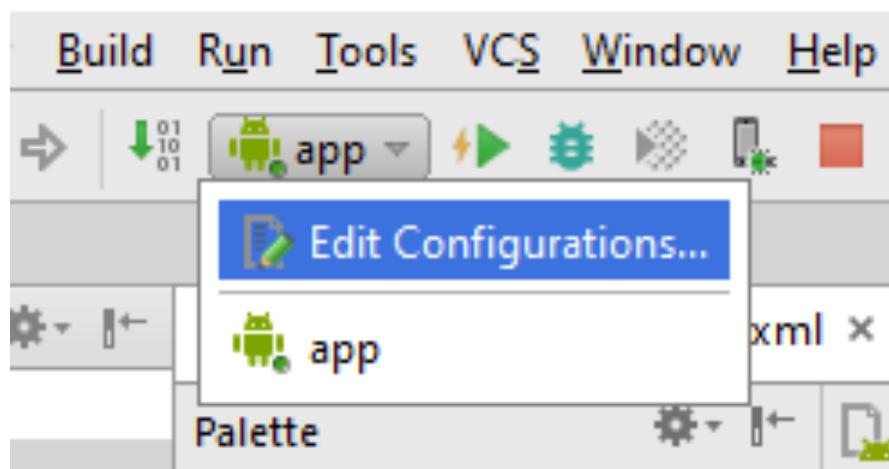


Figura 36 - Selecionando Edit Configurations a partir da barra de ferramentas

Na janela das configurações do **Run/Debug**, o aplicativo deve ser configurado para usar um emulador de preferência ao selecionarmos a opção **Emulator** em **Target**, na seção **Deployment Target Options** e selecionar o emulador do menu drop down. Abaixo, a imagem mostra o aplicativo OlaMundo configurado para ser executado no emulador Nexus 5 que criamos anteriormente.

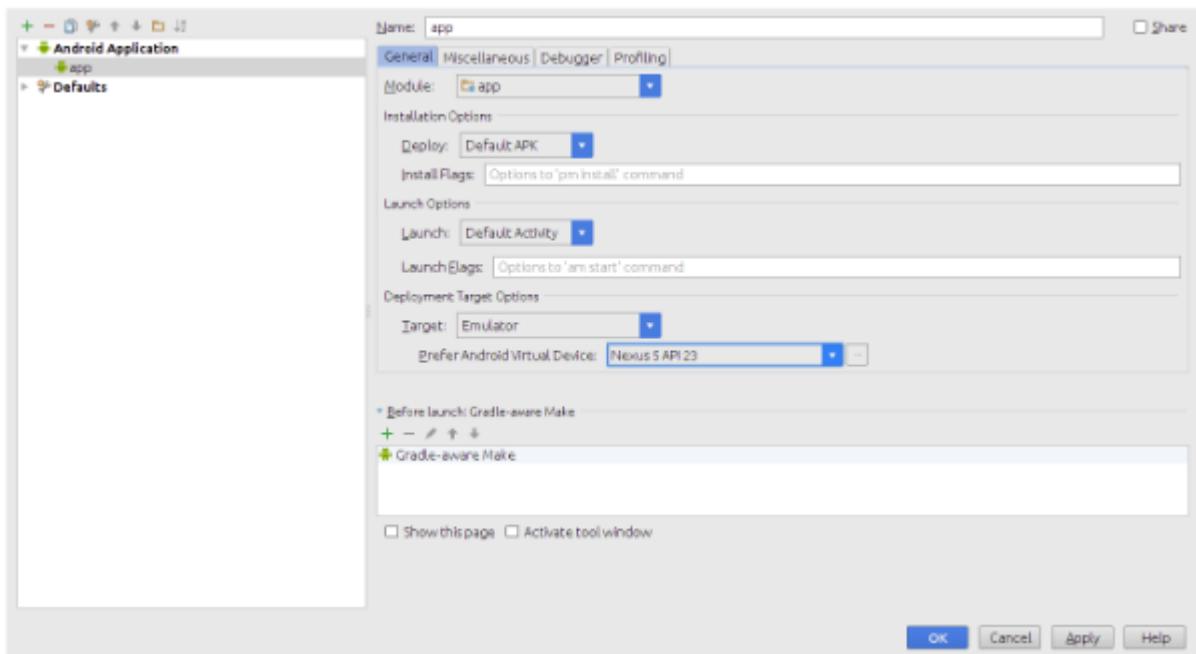


Figura 37 - Configuração do app OlaMundo

4.6 Parando um Aplicativo em Execução

Ao compilar e executar um aplicativo para fins de teste, cada vez que uma nova versão do aplicativo for compilada e executada, a instância anterior do aplicativo em execução no dispositivo ou emulador será automaticamente substituída pela nova versão. Também é possível, no entanto, interromper manualmente um aplicativo em execução a partir do Android Studio.

Para parar uma aplicação em execução, basta clicar no botão de Stop localizado na barra de ferramentas principal como mostrado na abaixo.



Figura 38 - Botão Stop na barra de ferramentas

Um aplicativo também pode ser finalizado usando o Android Monitor. Abra o Android Monitor pelo painel de ferramentas ou usando o botão da barra do painel, ou através do menu de acesso rápido (movendo o ponteiro do mouse sobre o botão no canto esquerdo da barra de status como mostrado abaixo).

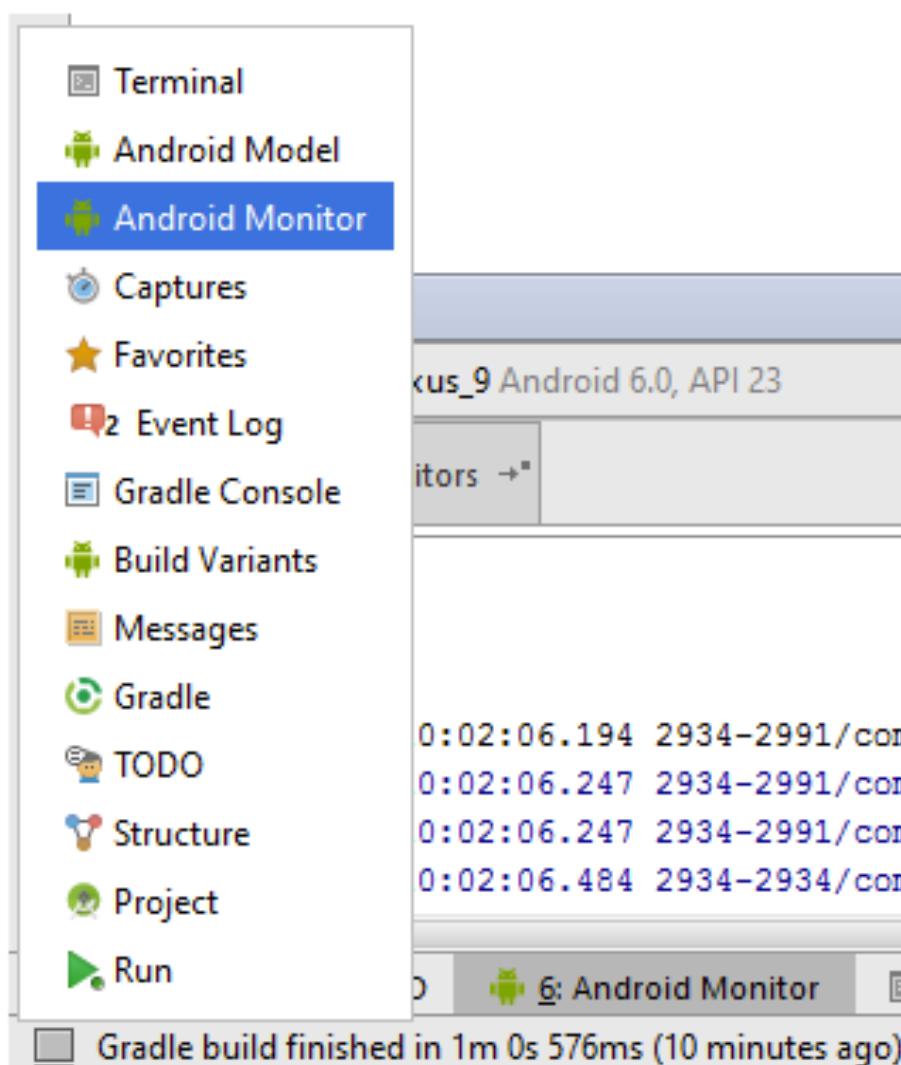


Figura 39 - Chamando o Android Monitor através do menu de acesso rápido

Uma vez que a janela do Android Monitor é aberta, selecione aplicativo no menu conforme abaixo:

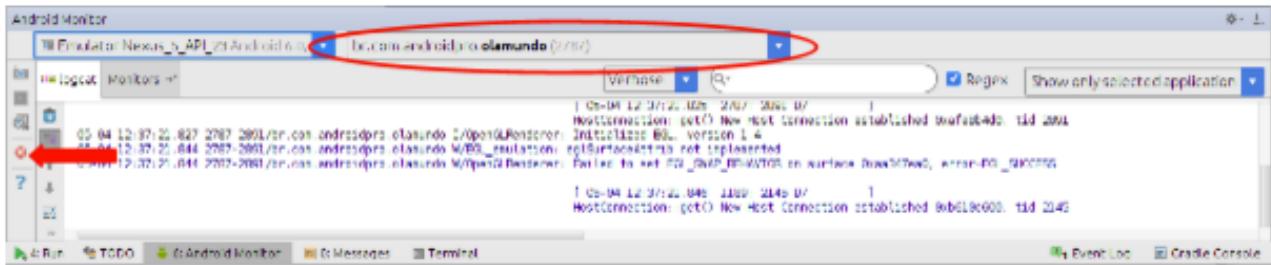


Figura 40 - Selecionando aplicativo

Com o processo selecionado, clique no botão vermelho **Terminate Application** na barra de ferramentas a esquerda da lista do processo indicado pela seta na figura acima.

4.7 Resumo

O processo típico de desenvolvimento de aplicativos segue o ciclo: codificar, compilar e executar em um ambiente de testes. Os aplicativos Android podem ser testados em qualquer dispositivo Android físico ou usando um emulador Android Virtual Device (AVD). Os AVDs são criados e gerenciados usando a ferramenta AVD Manager. Ao criar um AVD para simular um modelo de dispositivo, é importante que o dispositivo virtual seja configurado com uma especificação de hardware que coincide com o do dispositivo físico.

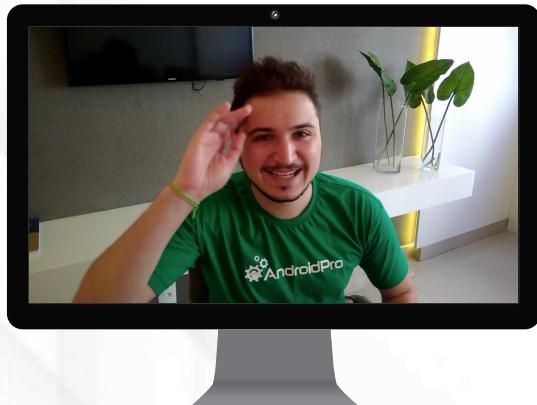
..... AULA ONLINE GRATUITA!

OS 5 PASSOS OBRIGATÓRIOS

PARA VOCÊ SER UM **DESENVOLVEDOR ANDROID**
PROFISSIONAL E INDEPENDENTE!

Porque você precisa participar?

- 1 Para entender os **erros** que fazem você desistir e se frustrar
- 2 Para descobrir um **método COMPROVADO** e **ÚNICO** para desenvolver praticamente qualquer tipo de aplicativo Android
- 3 Para saber quais as **habilidades exatas**, além da programação, que vão te transformar em um profissional
- 4 Para aprender a **criar oportunidades de trabalho** e ganhar dinheiro com desenvolvimento Android desde já!



VAGAS *LIMITADAS!*

INSCREVA-SE AQUI!

5. Testando Apps em Dispositivos Físicos

Muitos testes podem ser realizados usando um AVD, mas não há nenhum substituto para um dispositivo físico na hora da realização de testes reais, e há uma série de características do Android que só estão disponíveis em dispositivos físicos.

A comunicação com ambas as instâncias (AVD e dispositivos Android físicos conectados) é tratada pelo **Debug Android Bridge** (ADB). Vamos ver as etapas para configurar o ambiente ADB para permitir testes de aplicativos em um dispositivo físico com Mac OS X, Windows e Linux.

5.1 Uma visão geral do Debug Android Bridge (ADB)

O objetivo principal do ADB é facilitar a interação entre um sistema de desenvolvimento, neste caso o Android Studio, e ambos os emuladores AVD e dispositivos físicos, para a execução e debugging dos aplicativos.

O ADB é composto por um cliente, um servidor rodando em segundo plano no sistema de desenvolvimento e um processo em background rodando tanto em AVDs quanto em dispositivos Android reais, tais como telefones e tablets.

O cliente ADB pode ser utilizado em uma variedade de formas. Por exemplo, um cliente é fornecido sob a forma de uma ferramenta de linha de comando chamado adb, localizada no subdiretório platform-tools do SDK Android. Da mesma forma, o Android Studio também tem um cliente interno.

Uma variedade de tarefas podem ser realizadas utilizando a ferramenta de linha de comando adb. Por exemplo, uma listagem de dispositivos virtuais ou físicos atualmente ativos pode ser obtida usando o comando devices. A seguir, a saída do comando indica a presença de um AVD no sistema, mas não há dispositivos físicos:

```
$ adb devices
List of devices attached
emulator-5554 device
```

5.2 Ativando o ADB em Dispositivos Android

Antes de o ADB poder se conectar a um dispositivo Android, esse dispositivo deve primeiro ser configurado para permitir essa conexão. Em telefones e tablets que executam o Android 5.0 ou posterior, os passos para conseguir isso são os seguintes:

1. Abra as configurações do Android no dispositivo e selecione **Sobre o Tablet ou Sobre o Telefone**.
2. Na tela **Sobre**, vá para o campo **Build Number** (conforme figura abaixo) e clique nele **sete vezes**, até aparecer uma mensagem indicando que o modo de desenvolvedor foi ativado.

Kernel version
3.1.10-g6ff7a51
android-build@vpbs1.mtv.corp.google.com #1
Wed Dec 10 19:55:59 UTC 2014

Build number
LRX22G

Figura 41 - Build number

3. Retorne à tela principal de configurações e observe o aparecimento de uma nova opção chamada **Opções do Desenvolvedor**. Selecione esta opção e localize a configuração de tela do desenvolvedor chamada **USB debugging**. Ative a caixa de seleção ao lado deste item como ilustrado a seguir, para permitir a conexão do ADB.

Debugging

USB debugging

Debug mode when USB connected



Figura 42 - Caixa de seleção USB debugging

4. A partir do topo da tela, abra o painel de notificações e observe que o dispositivo está atualmente conectado como um "dispositivo de mídia".

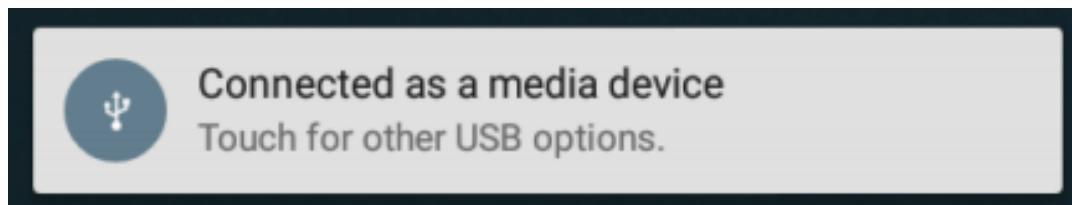


Figura 43 - Notificação de dispositivo conectado

Agora, o dispositivo está configurado para aceitar conexões do ADB no sistema de desenvolvimento. Agora só falta configurar o sistema de desenvolvimento para detectar o dispositivo, quando ele estiver conectado. Ainda que seja um processo relativamente simples, os passos envolvidos são diferentes dependendo do sistema de desenvolvimento que está sendo executado (Windows, Mac OS X ou Linux). Note que os seguintes passos consideram que o diretório platform-tools está incluído na variável de ambiente do sistema **PATH**.

5.2.1 Configuração de ADB no Mac OS X

Para configurar o ADB em um sistema Mac OS X, conecte o dispositivo ao computador através de um cabo USB, abra uma janela de terminal e execute o seguinte comando:

```
android update adb
```

Em seguida, reinicie o servidor ADB usando os seguintes comandos na janela do terminal:

```
$ adb kill-server  
$ adb start-server  
* daemon not running. starting it now on port 5037 *  
*daemon started successfully *
```

Uma vez que o servidor esteja rodando com sucesso, execute o seguinte comando para verificar se o dispositivo foi detectado:

```
$ adb devices  
List of devices attached  
74CE000600000001 offline
```

Se o dispositivo estiver listado como offline, vá para o dispositivo Android e verifique a presença da mensagem mostrada abaixo, solicitando a permissão **Allow USB debugging**. Ative a caixa de seleção ao lado da opção que diz **Always allow from this computer**, antes de clicar em **OK**. Repetindo o comando **adb devices** agora deveríamos ver o dispositivo como disponível:

```
List of devices attached  
015d41d4454bf80c device
```

No caso de o dispositivo não estar listado, tente fazer o logout, e em seguida, volte para a área de trabalho do Mac OS X e, se o problema persistir, reinicie o sistema.

5.2.2 Configuração do ADB no Windows

O primeiro passo para configurar um sistema de desenvolvimento baseado em Windows para se conectar a um dispositivo Android usando o ADB é instalar os drivers USB apropriados no sistema. No caso de alguns dispositivos, o Google USB Driver deve ser instalado (a lista completa de dispositivos suportados pelo Google USB Driver esta em <http://developer.android.com/sdk/win-usb.html>).

Para dispositivos Android não suportados pelo Google USB Driver,

Com os drivers instalados, e o dispositivo reconhecido corretamente, abra uma janela de Prompt de comando e execute o seguinte comando:

```
adb devices
```

Este comando deve mostrar as informações sobre o dispositivo conectado semelhante ao seguinte:

```
List of devices attached  
HT4CTJT01906  
offline
```

Se o dispositivo está listado como offline ou não autorizado, vá até a tela do dispositivo e verifique se a caixa de diálogo mostrada na figura abaixo apareceu pedindo a permissão para a depuração USB.

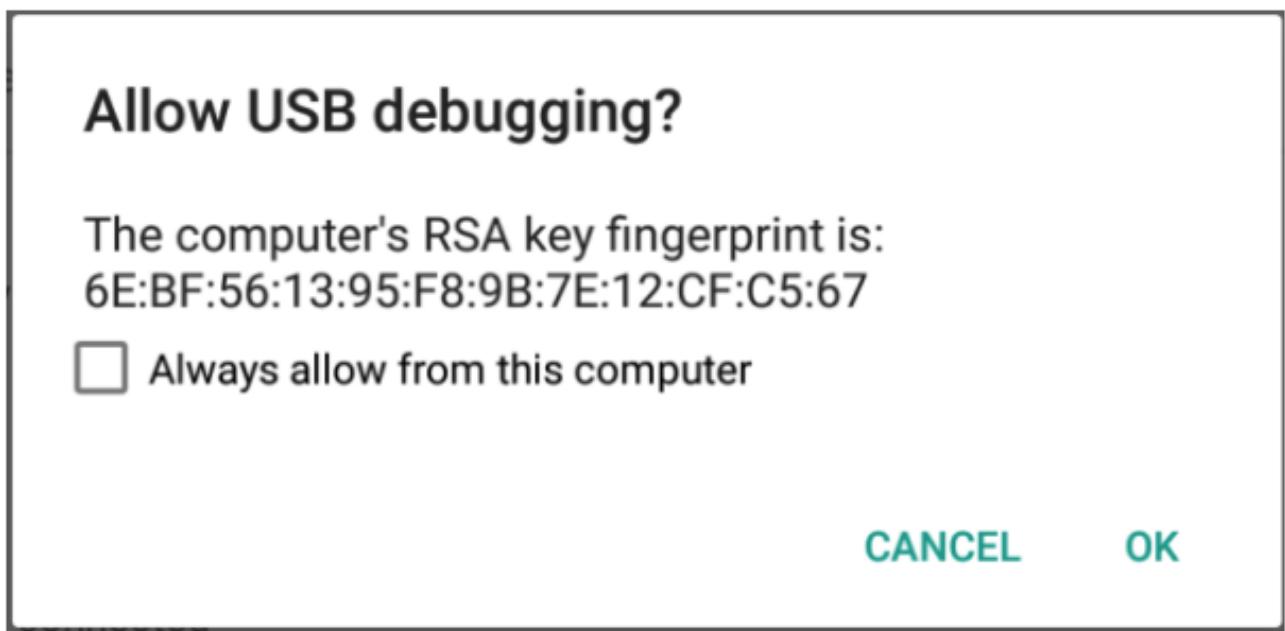


Figura 44 - Caixa de Diálogo

Ative a caixa de seleção ao lado da opção que diz Always allow from this computer, antes de clicar em OK. Repita o comando adb devices, que agora deve listar o dispositivo como pronto:

```
List of devices attached  
HT4CTJT01906  
device
```

No caso de o dispositivo não estar na lista, execute os seguintes comandos para reiniciar o servidor ADB:

```
adb kill-server  
adb start-server
```

Se o dispositivo ainda não estiver na lista, tente executar o seguinte comando:

```
android update adb
```

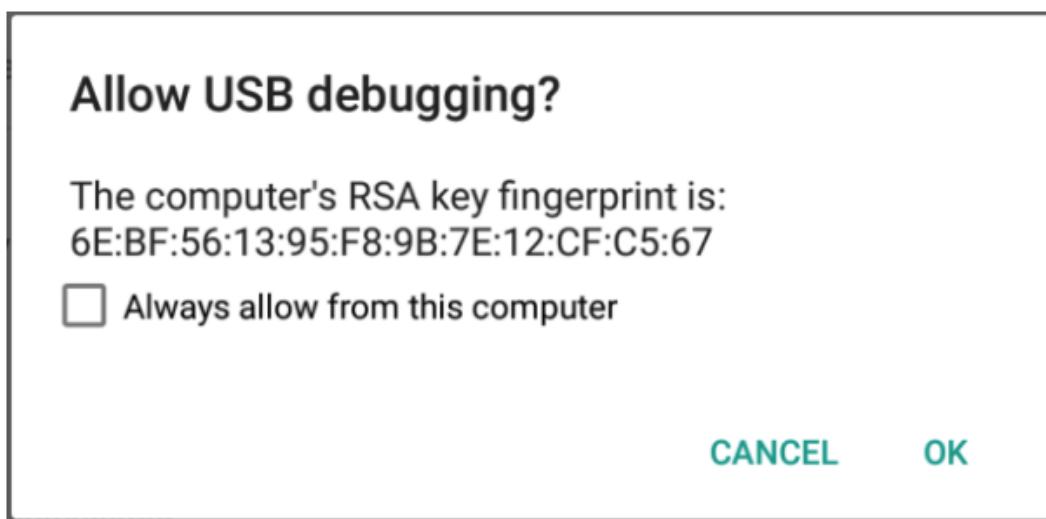


Figura 44 - Caixa de Diálogo

Pode ser necessário reiniciar o sistema.

5.2.3 Configuração de ADB no Linux

Agora, vamos mais uma vez usar o Ubuntu como um exemplo de referência em termos de configuração do ADB no Linux para se conectar a um dispositivo Android físico.

Comece por conectar o dispositivo Android em uma porta USB no Ubuntu. Uma vez conectado, abra uma janela do terminal e execute o comando **lsusb** para visualizar os dispositivos USB disponíveis atualmente:

```
~$ lsusb
```

```
Bus 001 Device 003: ID 18d1:4e44 asus Nexus 7 [9999]
```

```
Bus 001 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

Cada dispositivo USB detectado no sistema será listado junto com um ID do fornecedor e identificação do produto. Uma lista de IDs de fornecedor pode ser encontrada acessando
<http://developer.android.com/tools/device.html#VendorIds>.

A saída acima mostra que um dispositivo Nexus 7 foi detectado. Anote o fornecedor e os números de identificação do produtos do seu dispositivo (no caso acima estes são 18D1 e 4E44 respectivamente).

Use o comando **sudo** para editar o arquivo **51-android.rules** localizado no diretório /etc/udev/rules.d. Por exemplo:

```
sudo gedit /etc/udev/rules.d/51-android.rules
```

Dentro do editor, adicione a entrada para o dispositivo Android, substituindo <vendor_id> e <Product_id> com as identificações de fornecedores e produtos vistos anteriormente pelo comando lsusb:

```
SUBSYSTEM=="usb", ATTR{idVendor}=="<vendor_id>",  
ATTRS{idProduct}=="<product_id>", MODE="0666", GROUP="plugdev"
```

Uma vez que a entrada tenha sido adicionada, salve o arquivo e saia do editor.

Em seguida, execute o seguinte comando para dar as permissões necessárias ao arquivo criado:

```
chmod a+r /etc/udev/rules.d/51-android.rules
```

Uma vez que as alterações acima forem feitas, reinicie o Ubuntu. Quando o sistema for reiniciado, abra uma janela do terminal, inicie o servidor ADB e verifique a lista de dispositivos conectados:

```
$ adb start-server  
* daemon not running. starting it now on port 5037 *  
* daemon started successfully *  
$ adb devices  
List of devices attached  
015d41d4454bf80c  
offline
```

Se o dispositivo estiver como offline ou não autorizado, vá para o dispositivo Android e verifique se a caixa de diálogo que pede permissão para a depuração USB está aparecendo.

5.3 Resumo

Enquanto o emulador Android fornece um excelente ambiente de teste, é importante ter em mente que não há nenhum substituto ao dispositivo real, para ter certeza de que um aplicativo esteja sendo executado corretamente, sem erros.

O Android Studio não está configurado para detectar os dispositivos Android como ambiente de testes. Por isso, é necessário realizar alguns passos, a fim de ser capaz de executar as aplicações diretamente no dispositivo Android, a partir do Android Studio. As etapas para alcançar este objetivo diferem, dependendo da plataforma de desenvolvimento que for usada. Neste capítulo final, nós vimos esses passos para Linux, Mac OS X e Windows.

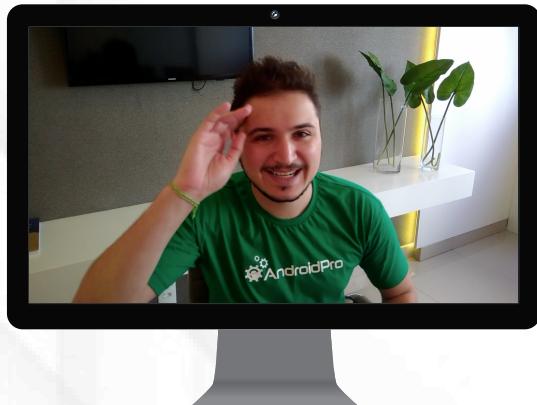
..... AULA ONLINE GRATUITA!

OS 5 PASSOS OBRIGATÓRIOS

PARA VOCÊ SER UM **DESENVOLVEDOR ANDROID**
PROFISSIONAL E INDEPENDENTE!

Porque você precisa participar?

- 1 Para entender os **erros** que fazem você desistir e se frustrar
- 2 Para descobrir um **método COMPROVADO** e **ÚNICO** para desenvolver praticamente qualquer tipo de aplicativo Android
- 3 Para saber quais as **habilidades exatas**, além da programação, que vão te transformar em um profissional
- 4 Para aprender a **criar oportunidades de trabalho** e ganhar dinheiro com desenvolvimento Android desde já!



VAGAS *LIMITADAS!*

INSCREVA-SE AQUI!