



PADO
Labs

INTRODUÇÃO AO DESENVOLVIMENTO MOBILE



Desenvolvimento Mobile - Aula I



Contato



Professor



David Krepsky



david@hausenn.com.br



DKrepsky

O QUE É O ANDROID



- O Android é um sistema operacional baseado no Linux projetado inicialmente para smartphones.
- Hoje o ecossistema android engloba uma série de produtos como tablets, notebooks, tvs e smartwatches.
- É um projeto 100% Open Source.
- Desenvolvido pela Google.
- Em 2022, o Android está presente em mais de 2.5 bilhões de dispositivos pelo mundo.
- 90% dos celulares utilizam o sistema operacional Android.

VERSÕES DO ANDROID



- Android Astro - 1.0
- Android CupCake - 1.5
- Android Donut - 1.6
- Android Éclair - 2.0
- Android Froyo - 2.2
- Android Gingerbread - 2.3
- Android Honeycomb - 3.0
- Android Kitkat - 4.4
- Android Lollipop - 5.0
- Android Marshmallow - 6.0
- Android Nougat- 7.0
- Android Oreo - 8.0
- Android Pie - 9.0
- Android Q - 10.0
- Android Red Velvet - 11.0
- Android Snow Cone - 12.0

ARQUITETURA ANDROID



APLICAÇÃO

FRAMEWORKS

BIBLIOTECAS

MÁQUINA VIRTUAL JAVA (Dalvik)

LINUX

FERRAMENTAS PARA O DESENVOLVIMENTO



- Java SE 11.0
- Android Studio
- Android SDK
- Android Platform Tools
- Linux KVM



CONFIGURAÇÃO DO AMBIENTE DE DESENVOLVIMENTO

INSTALAÇÃO DO JAVA



No terminal, digite:

```
sudo apt install openjdk-11-jdk
```

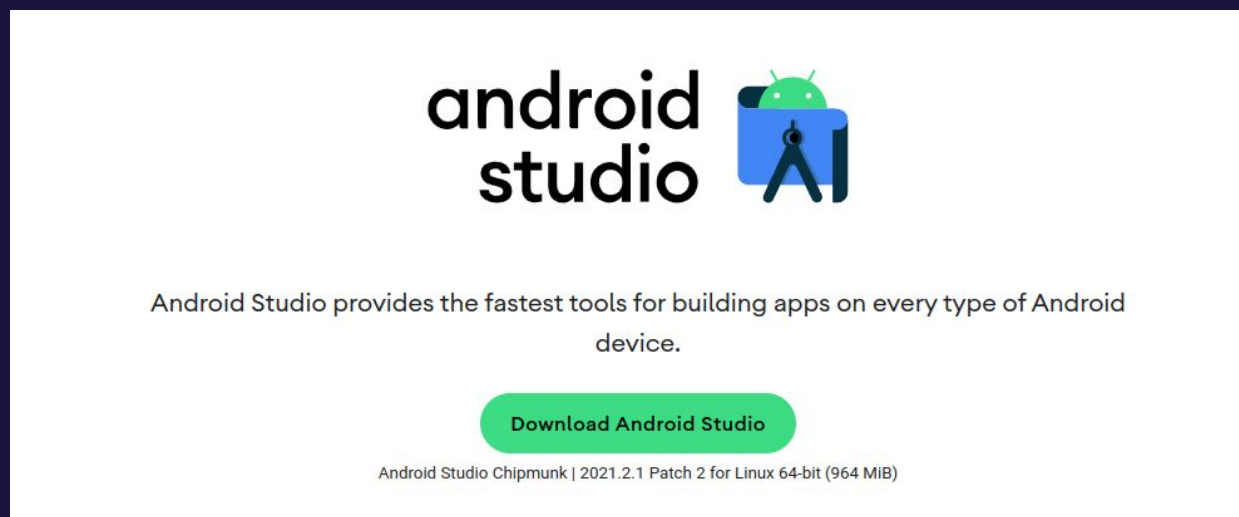
INSTALAÇÃO DO ANDROID STUDIO



Acesse:

- <https://developer.android.com/studio>

Faça o download clicando no botão *Download Android Studio*:



INSTALAÇÃO DO ANDROID STUDIO



Aceite os termos de uso e prossiga com o download:

13.1 Google may make changes to the License Agreement as it distributes new versions of the SDK. When these changes are made, Google will make a new version of the License Agreement available on the website where the SDK is made available.

14. General Legal Terms

14.1 The License Agreement constitutes the whole legal agreement between you and Google and governs your use of the SDK (excluding any services which Google may provide to you under a separate written agreement), and completely replaces any prior agreements between you and Google in relation to the SDK. 14.2 You agree that if Google does not exercise or enforce any legal right or remedy which is contained in the License Agreement (or which Google has the benefit of under any applicable law), this will not be taken to be a formal waiver of Google's rights and that those rights or remedies will still be available to Google. 14.3 If any court of law, having the jurisdiction to decide on this matter, rules that any provision of the License Agreement is invalid, then that provision will be removed from the License Agreement without affecting the rest of the License Agreement. The remaining provisions of the License Agreement will continue to be valid and enforceable. 14.4 You acknowledge and agree that each member of the group of companies of which Google is the parent shall be third party beneficiaries to the License Agreement and that such other companies shall be entitled to directly enforce, and rely upon, any provision of the License Agreement that confers a benefit on (or rights in favor of) them. Other than this, no other person or company shall be third party beneficiaries to the License Agreement. 14.5 EXPORT RESTRICTIONS. THE SDK IS SUBJECT TO UNITED STATES EXPORT LAWS AND REGULATIONS. YOU MUST COMPLY WITH ALL DOMESTIC AND INTERNATIONAL EXPORT LAWS AND REGULATIONS THAT APPLY TO THE SDK. THESE LAWS INCLUDE RESTRICTIONS ON DESTINATIONS, END USERS AND END USE. 14.6 The rights granted in the License Agreement may not be assigned or transferred by either you or Google without the prior written approval of the other party. Neither you nor Google shall be permitted to delegate their responsibilities or obligations under the License Agreement without the prior written approval of the other party. 14.7 The License Agreement, and your relationship with Google under the License Agreement, shall be governed by the laws of the State of California without regard to its conflict of laws provisions. You and Google agree to submit to the exclusive jurisdiction of the courts located within the county of Santa Clara, California to resolve any legal matter arising from the License Agreement. Notwithstanding this, you agree that Google shall still be allowed to apply for injunctive remedies (or an equivalent type of urgent legal relief) in any jurisdiction. July 27, 2021

☒ I have read and agree with the above terms and conditions

[Download Android Studio Chipmunk 2021.2.1 Patch 2 for Linux](#)

`android-studio-2021.2.1.16-linux.tar.gz`

INSTALAÇÃO DO ANDROID STUDIO

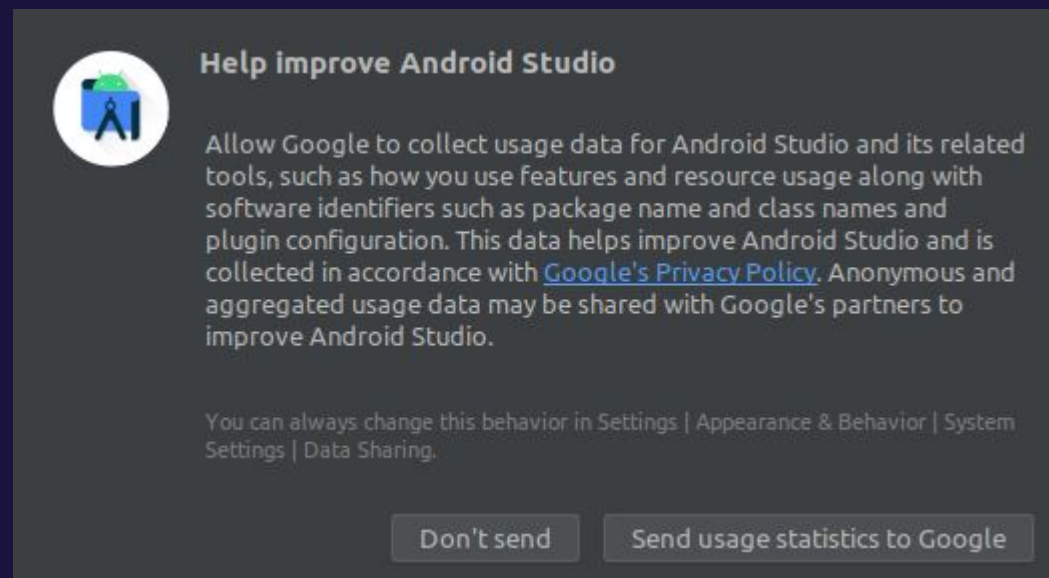


- Extraia o arquivo na pasta Home do seu usuário.
- Abra o Android Studio executando o arquivo no caminho `<home>/android studio/bin/studio.sh`

INSTALAÇÃO DO ANDROID STUDIO



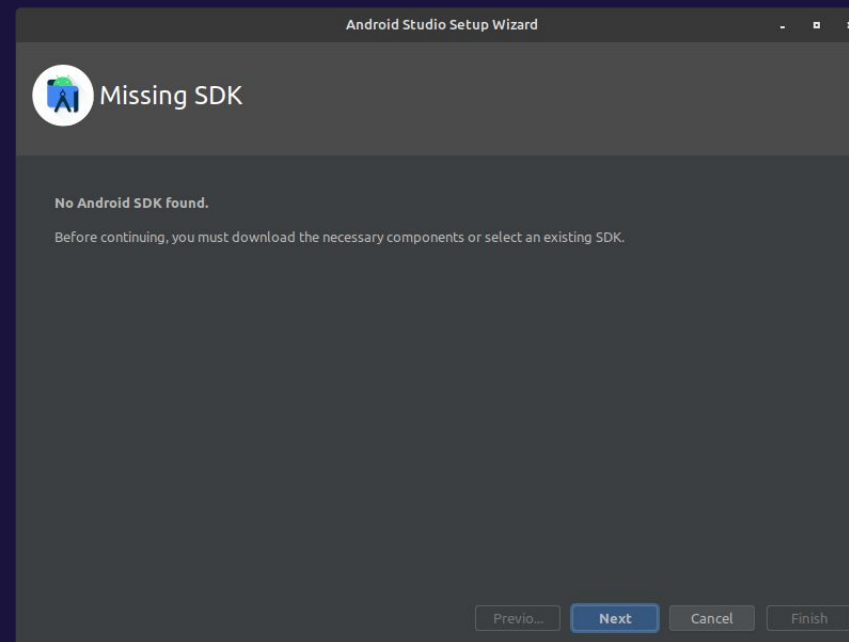
- Selecione a opção “Não Enviar” (*Don't Send*) na tela de envio de informações ao Google.



INSTALAÇÃO DO ANDROID STUDIO



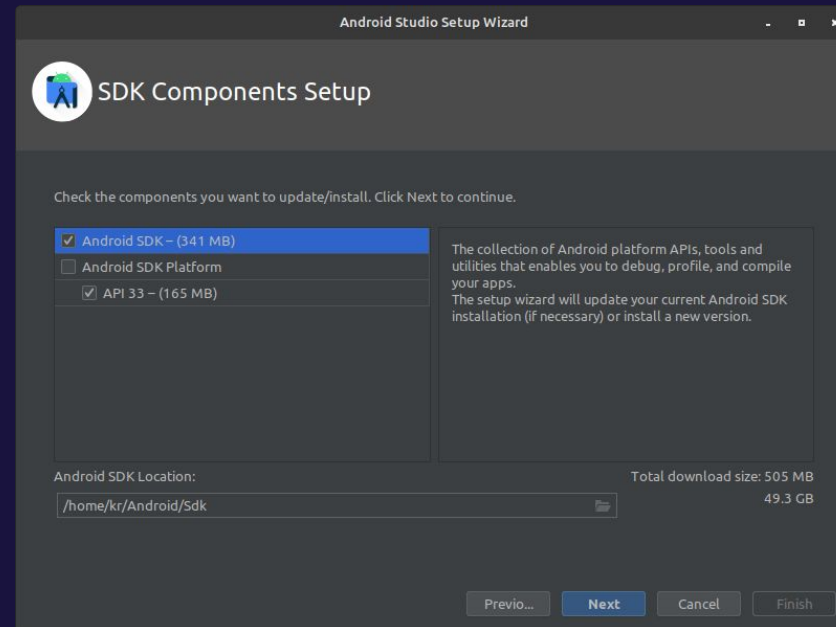
- Pressione *Próximo (Next)*



INSTALAÇÃO DO ANDROID STUDIO



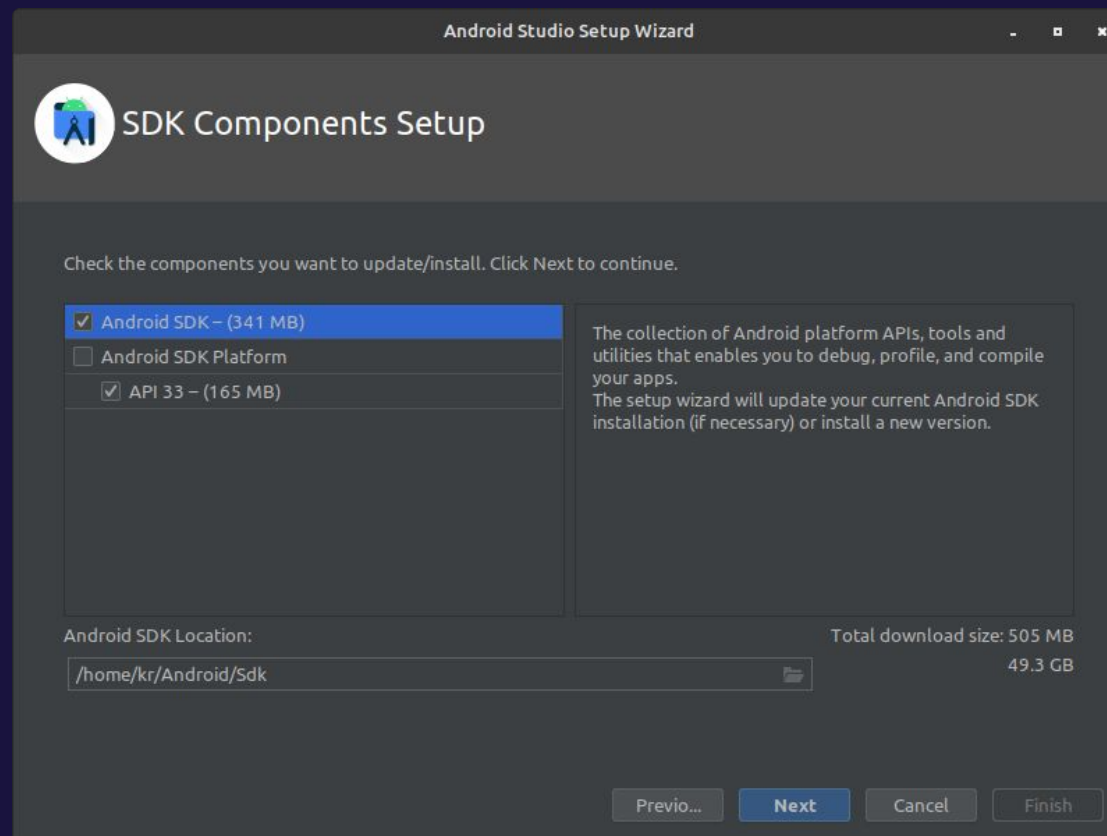
- Pressione *Próximo (Next)*



INSTALAÇÃO DO ANDROID STUDIO



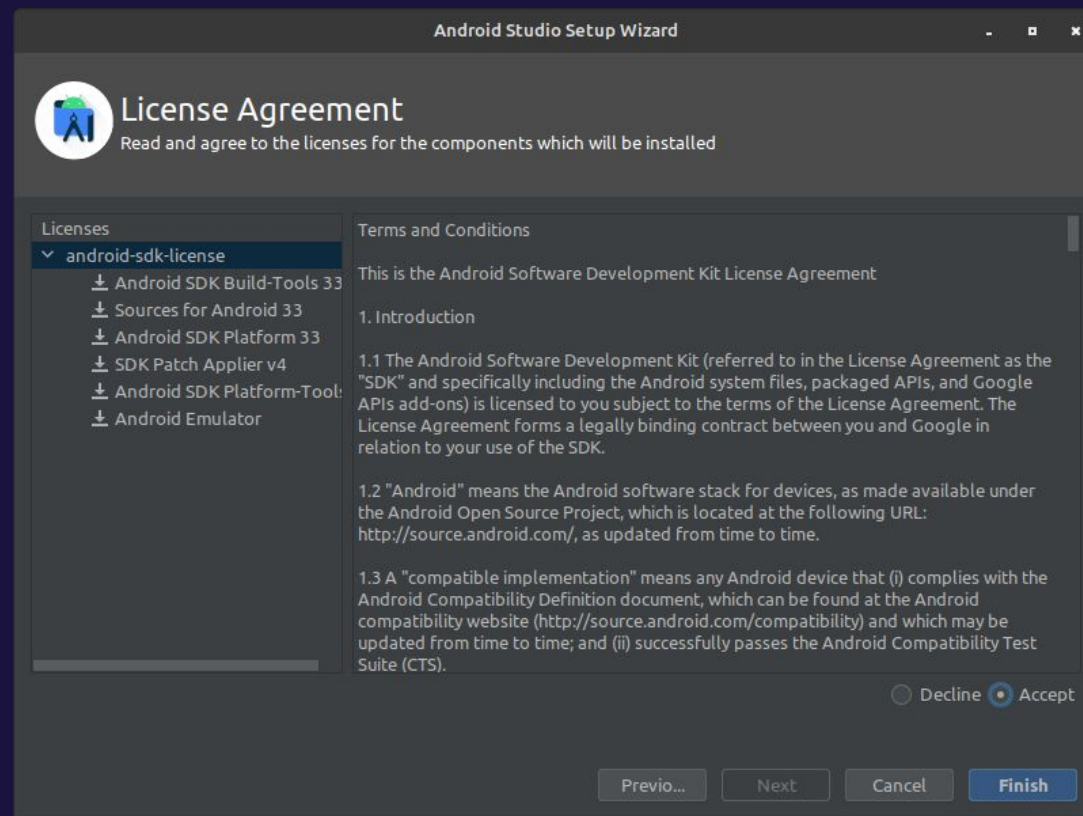
- Pressione *Próximo (Next)*



INSTALAÇÃO DO ANDROID STUDIO



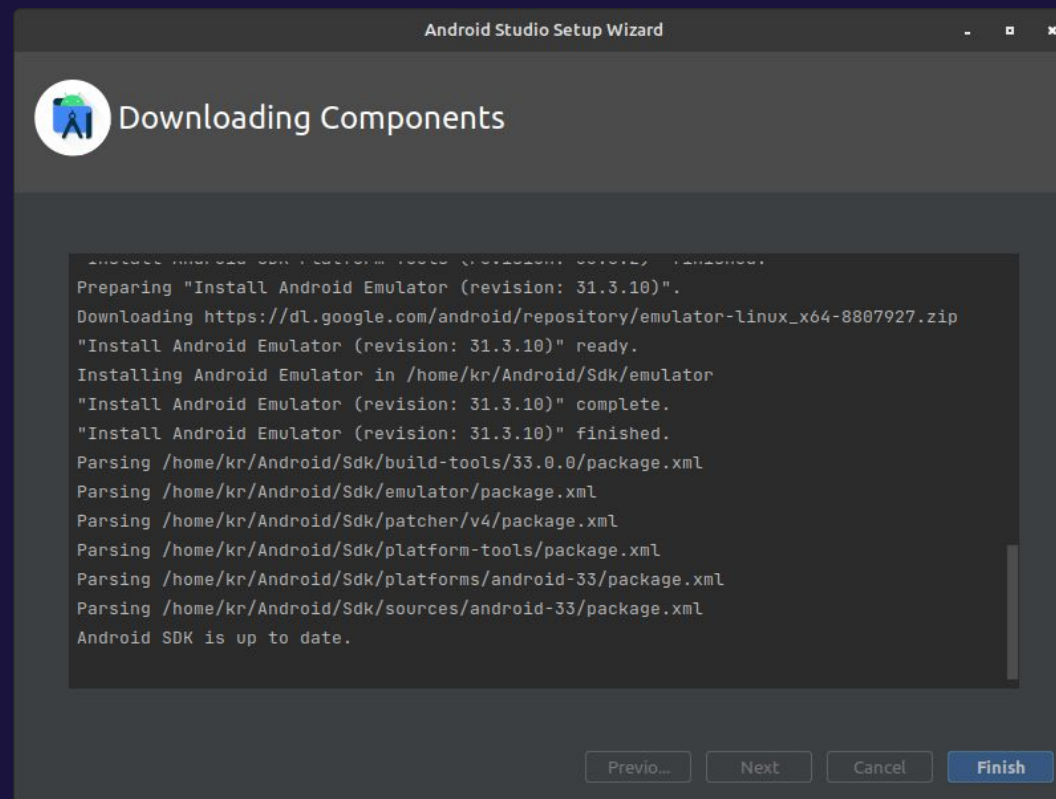
- Aceite as licenças e pressione Finalizar (*Finish*)



INSTALAÇÃO DO ANDROID STUDIO



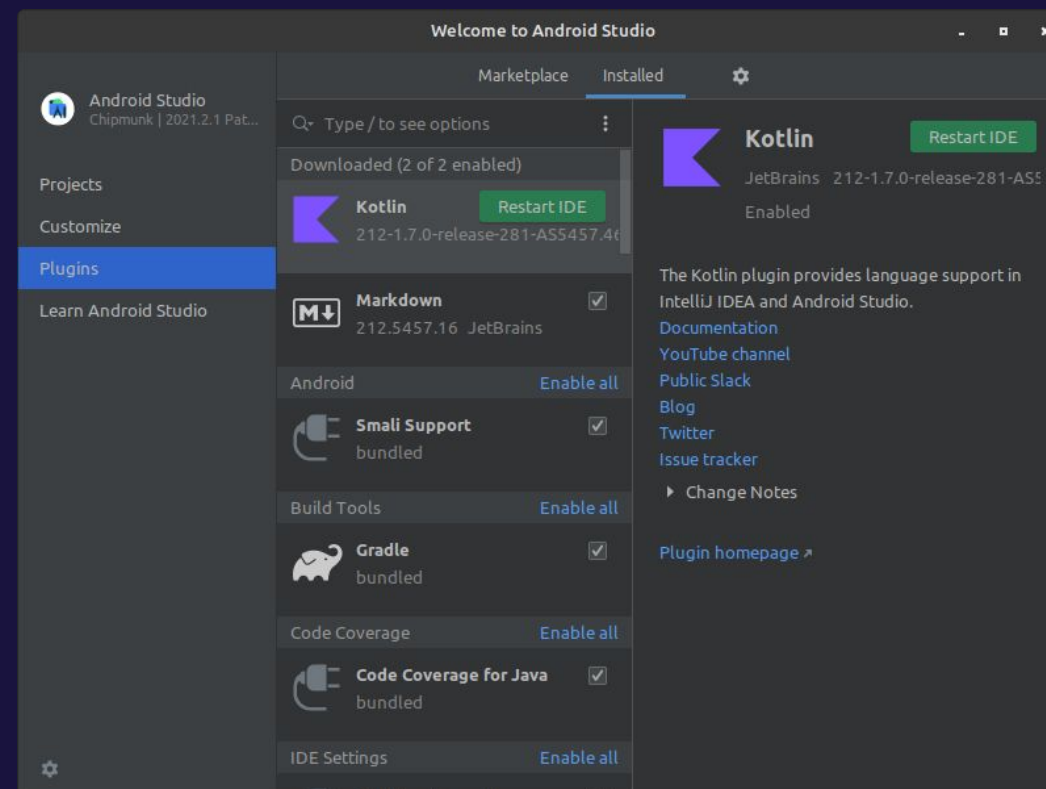
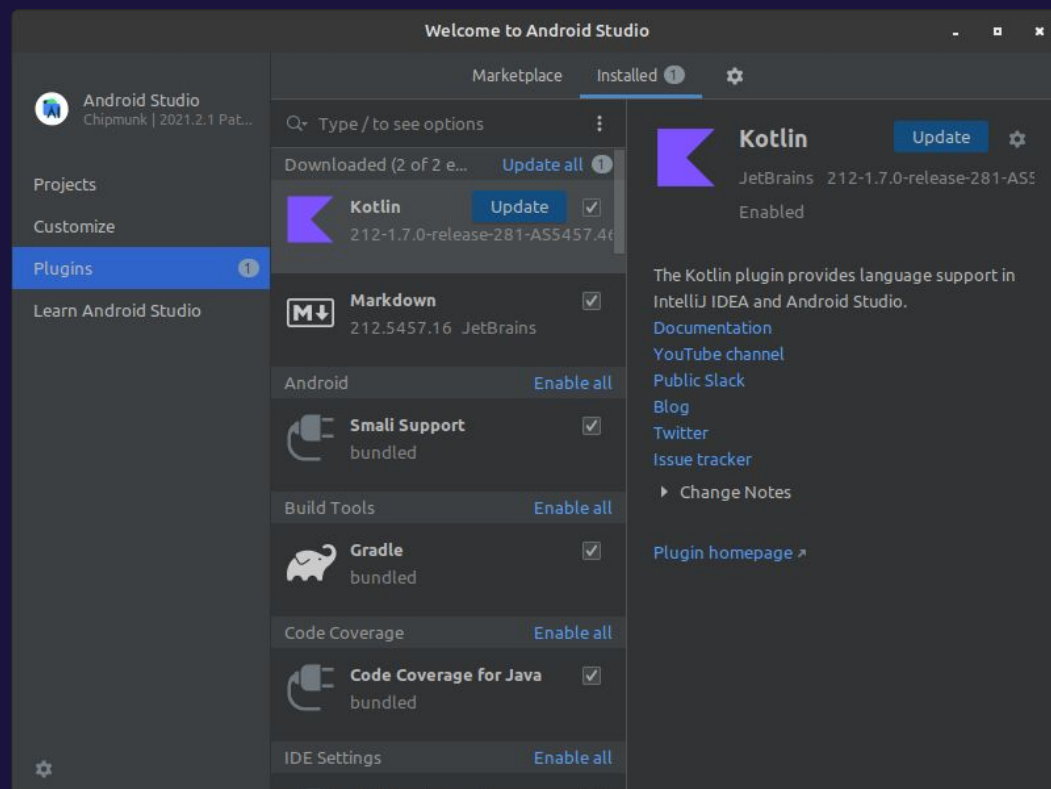
- Após o download do SDK, pressione Finalizar (*Finish*)



INSTALAÇÃO DO ANDROID STUDIO



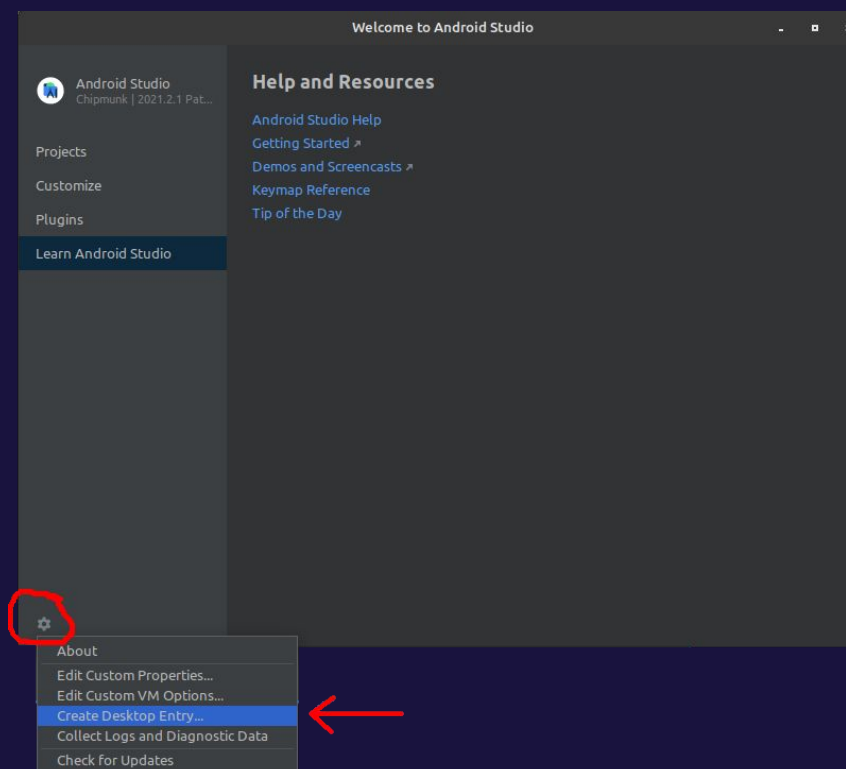
- Na Aba de Plugins, atualize os plugins necessários e reinicie o Android Studio



INSTALAÇÃO DO ANDROID STUDIO



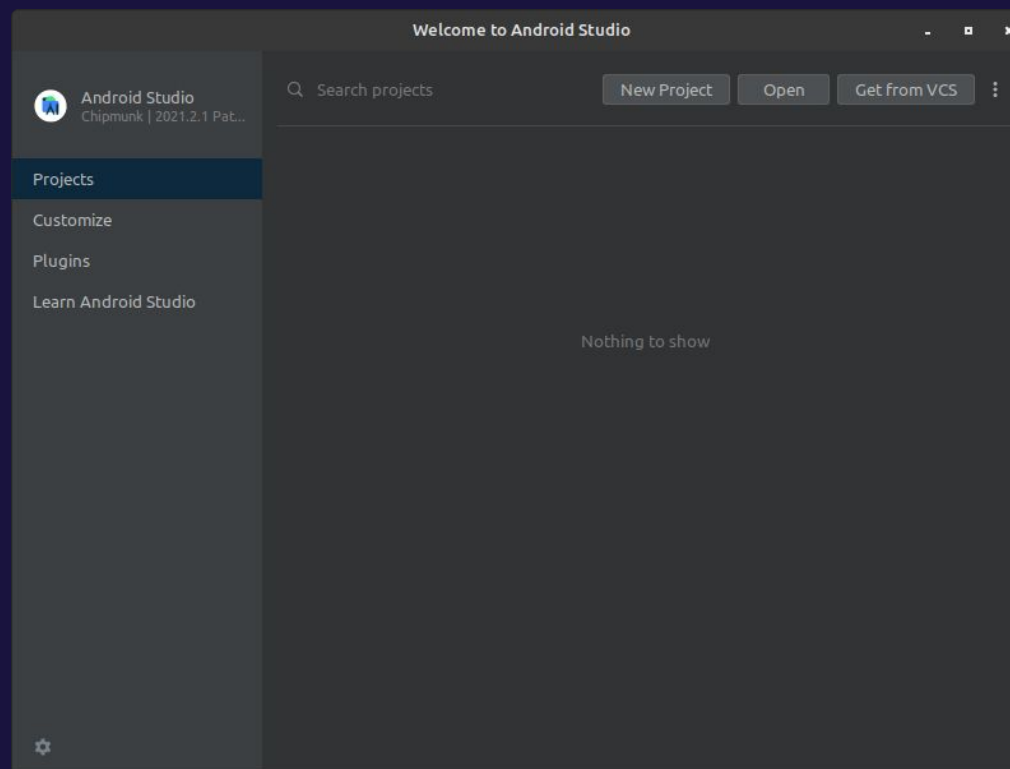
- (Opcional) Crie um link para a área de trabalho clicando na opção conforme a imagem abaixo:



INSTALAÇÃO DO ANDROID STUDIO



- Caso a instalação tenha ocorrido com sucesso, a tela inicial do Android Studio será apresentada.



INSTALAÇÃO DO KVM

- No terminal, digite:

```
sudo apt install qemu-kvm libvirt-daemon-system libvirt-clients  
bridge-utils
```

- Adicione seu usuário ao grupo libvirt:

```
sudo adduser 'username' libvirt
```

- Adicione seu usuário ao grupo kvm:

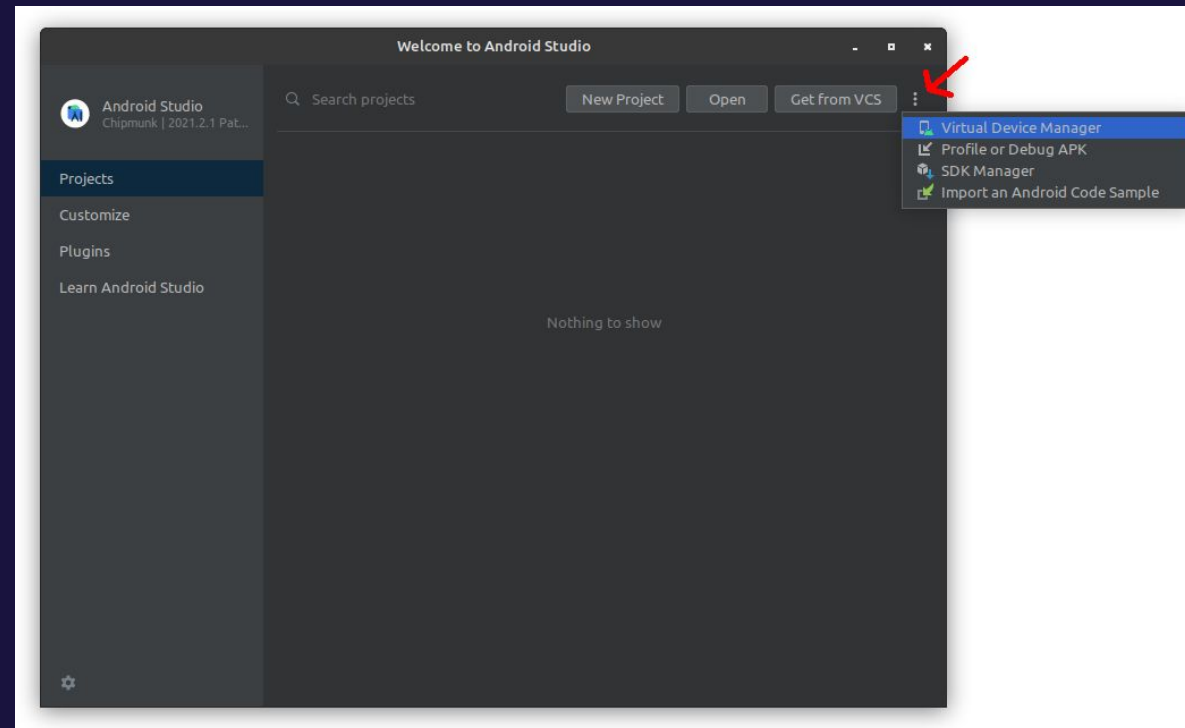
```
sudo adduser 'username' kvm
```

- Reinicie o computador

CONFIGURAÇÃO DO EMULADOR



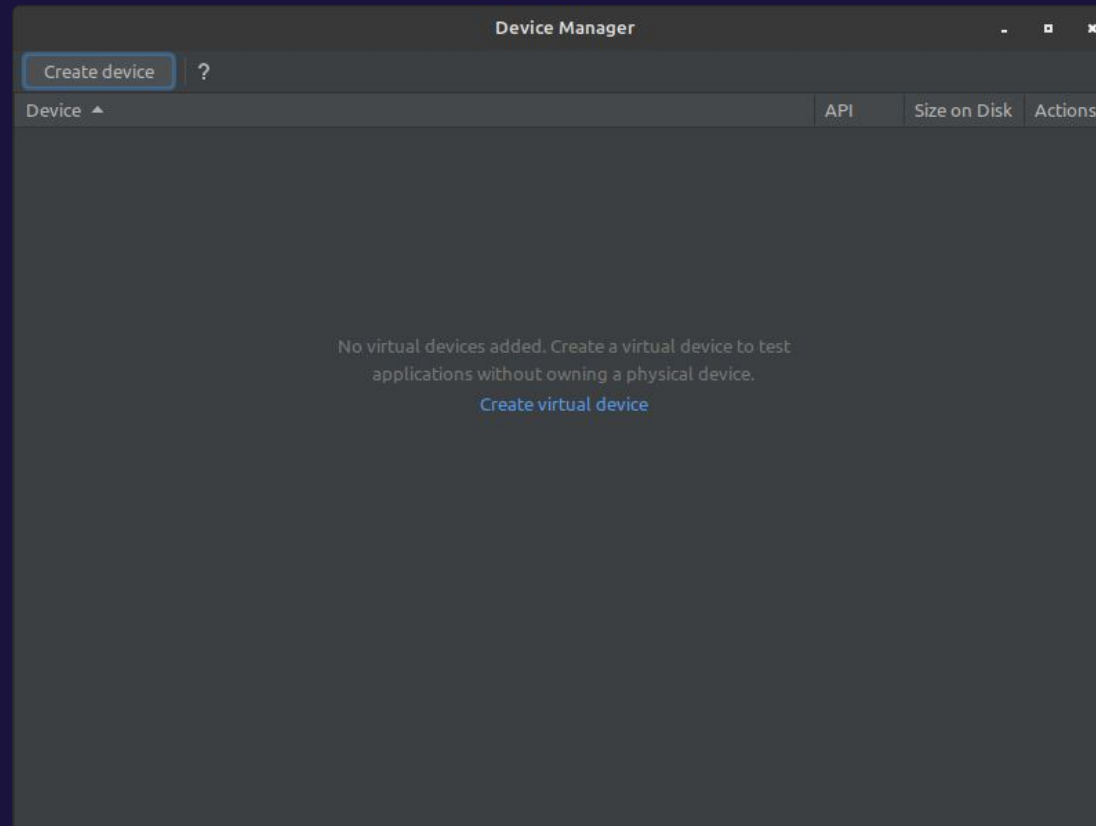
Abra o gerenciador de dispositivos virtuais



CONFIGURAÇÃO DO EMULADOR



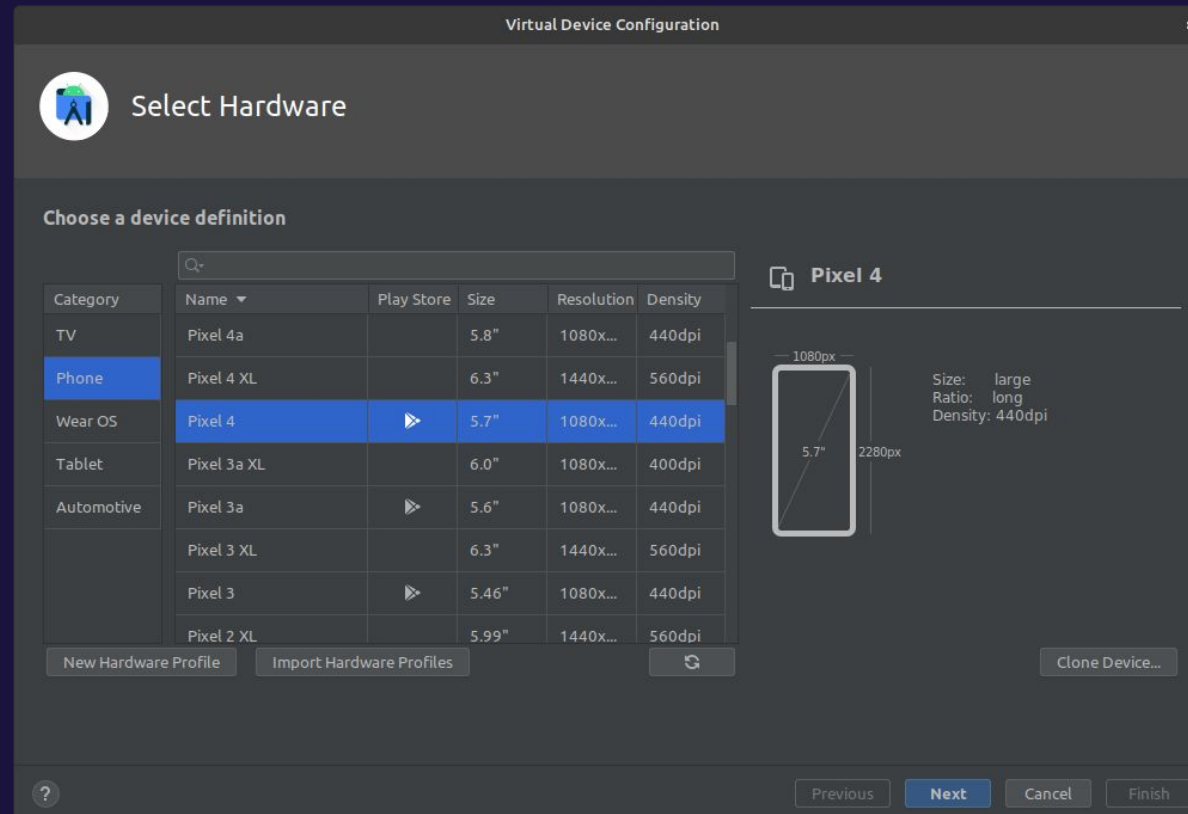
Clique no botão “Criar dispositivo” (Create device)



CONFIGURAÇÃO DO EMULADOR



Selecione o telefone Pixel 4 e clique em Próximo (Next)



CONFIGURAÇÃO DO EMULADOR



Clique em download na opção R (API 30, Android 11.0)

Virtual Device Configuration

System Image

Select a system image

Recommended x86 Images Other Images

Release Name	API Level	ABI	Target
Tiramisu Privacy Sandbox Download	33	x86_64	Android API Tiramisu Privacy
API 33 Download	33	x86_64	Android API 33 (Google Play)
Sv2 Download	32	x86_64	Android API 32 (Google Play)
S Download	31	x86_64	Android 12.0 (Google Play)
R Download	30	x86	Android 11.0 (Google Play)
Q Download	29	x86	Android 10.0 (Google Play)
Pie Download	28	x86	Android 9.0 (Google Play)
Oreo Download	27	x86	Android 8.1 (Google Play)
Oreo Download	26	x86	Android 8.0 (Google Play)
Nougat Download	25	x86	Android 7.1.1 (Google Play)

R

API Level
30

Android
11.0

Google Inc.

System Image
x86

We recommend these Google Play images because this device is compatible with Google Play.

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

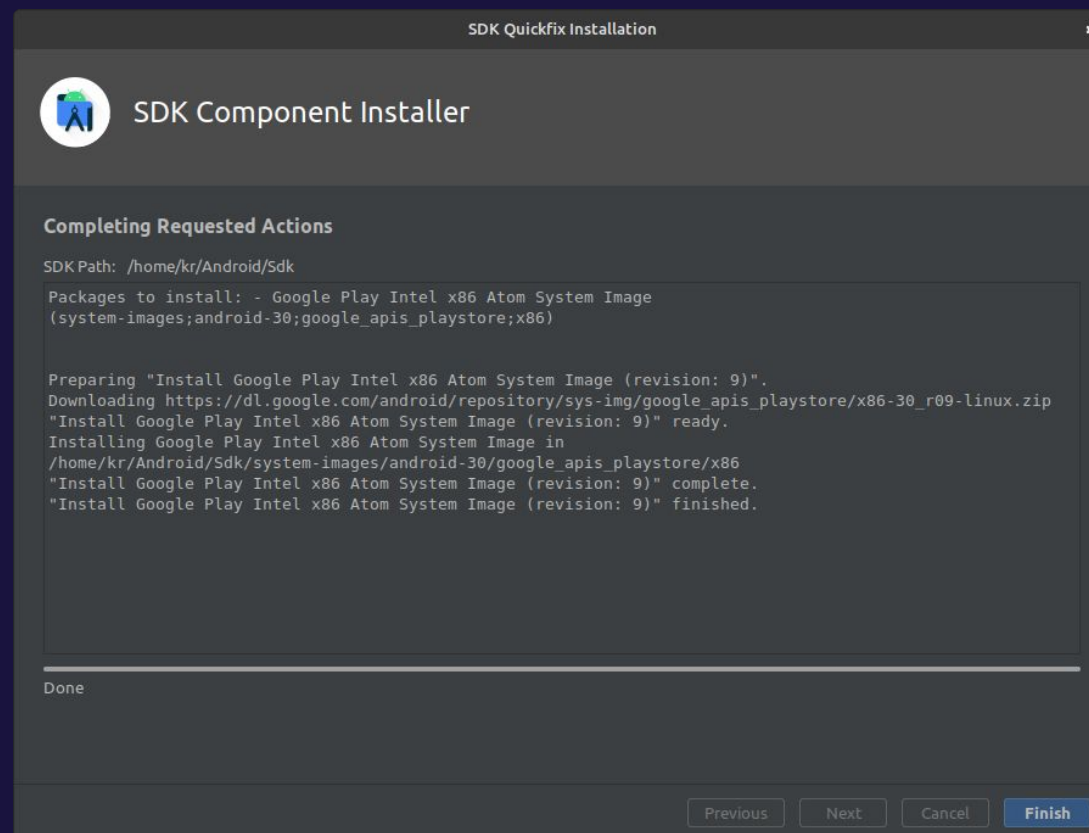
A system image must be selected to continue.

Previous Next Cancel Finish

CONFIGURAÇÃO DO EMULADOR



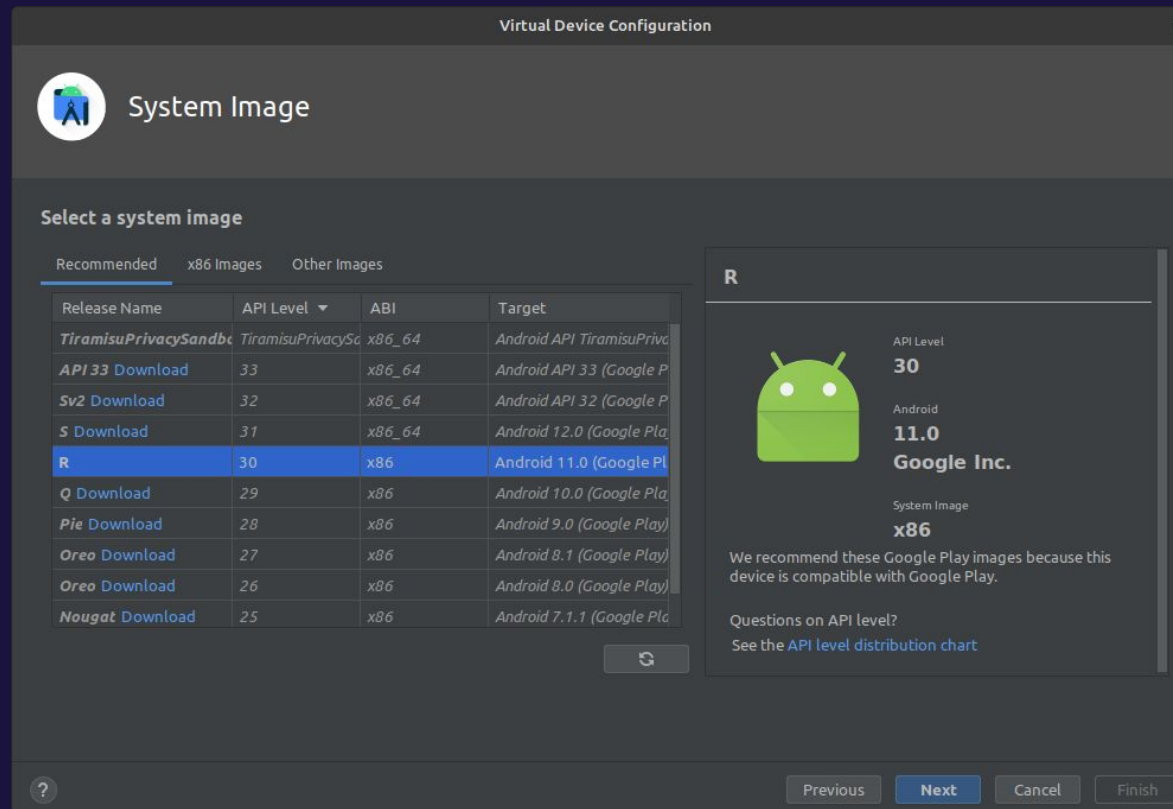
Após o download da imagem, clique em Finalizar (Finish)



CONFIGURAÇÃO DO EMULADOR



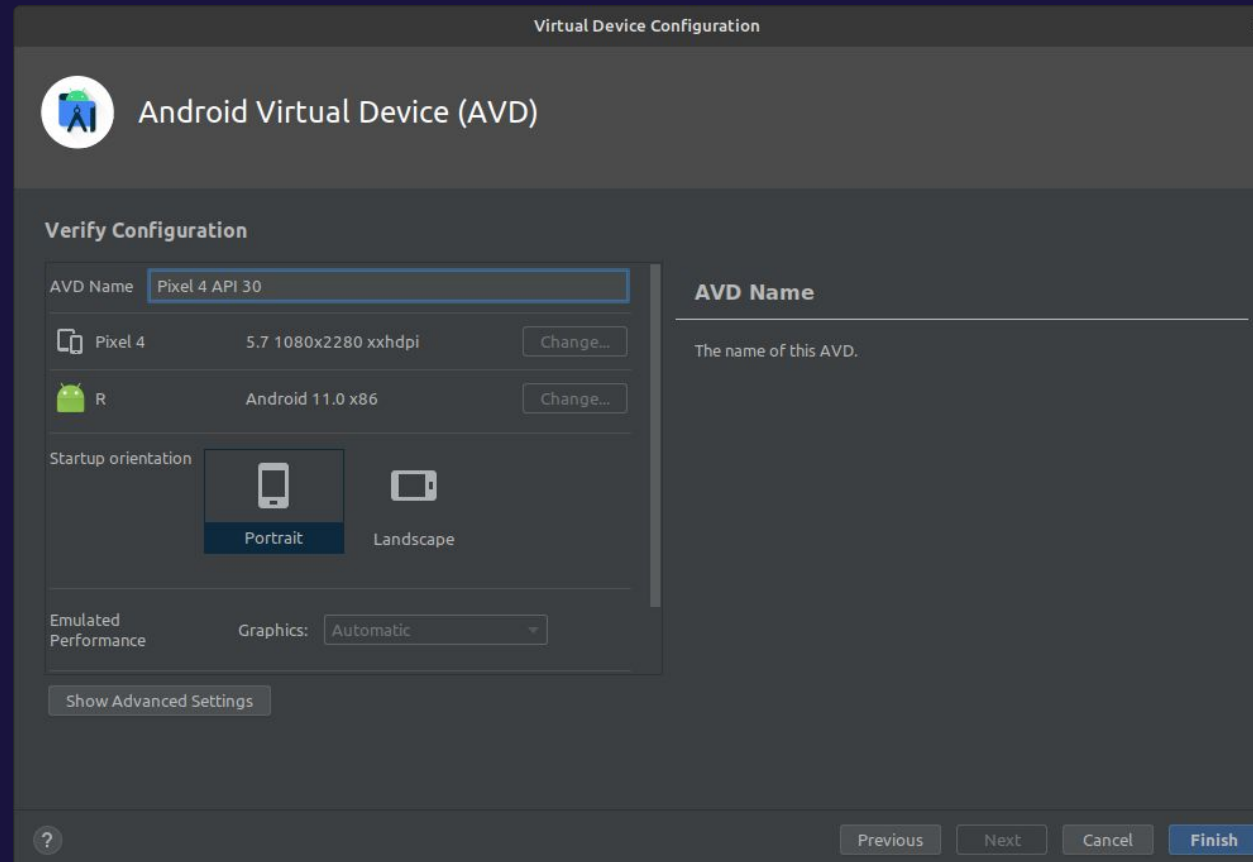
Com a opção R selecionada, clique em Próximo (Next)



CONFIGURAÇÃO DO EMULADOR



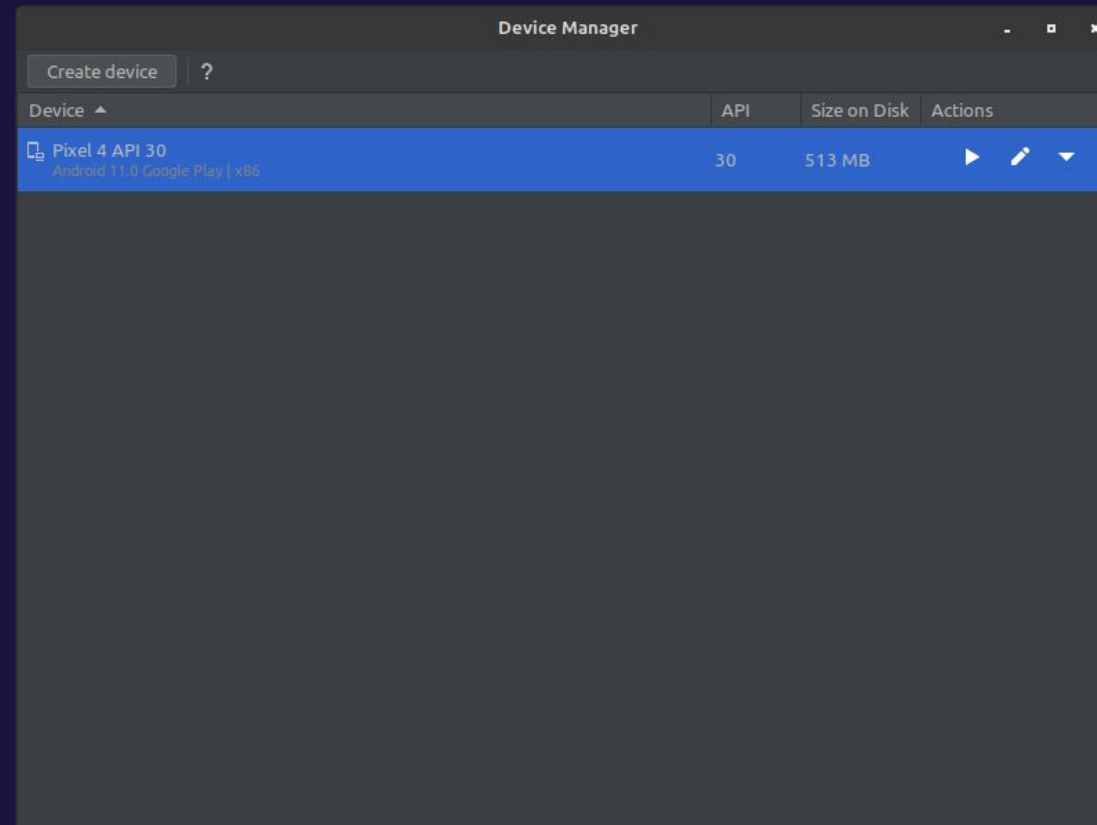
Confira as configurações e clique em Finalizar (Finish)



CONFIGURAÇÃO DO EMULADOR



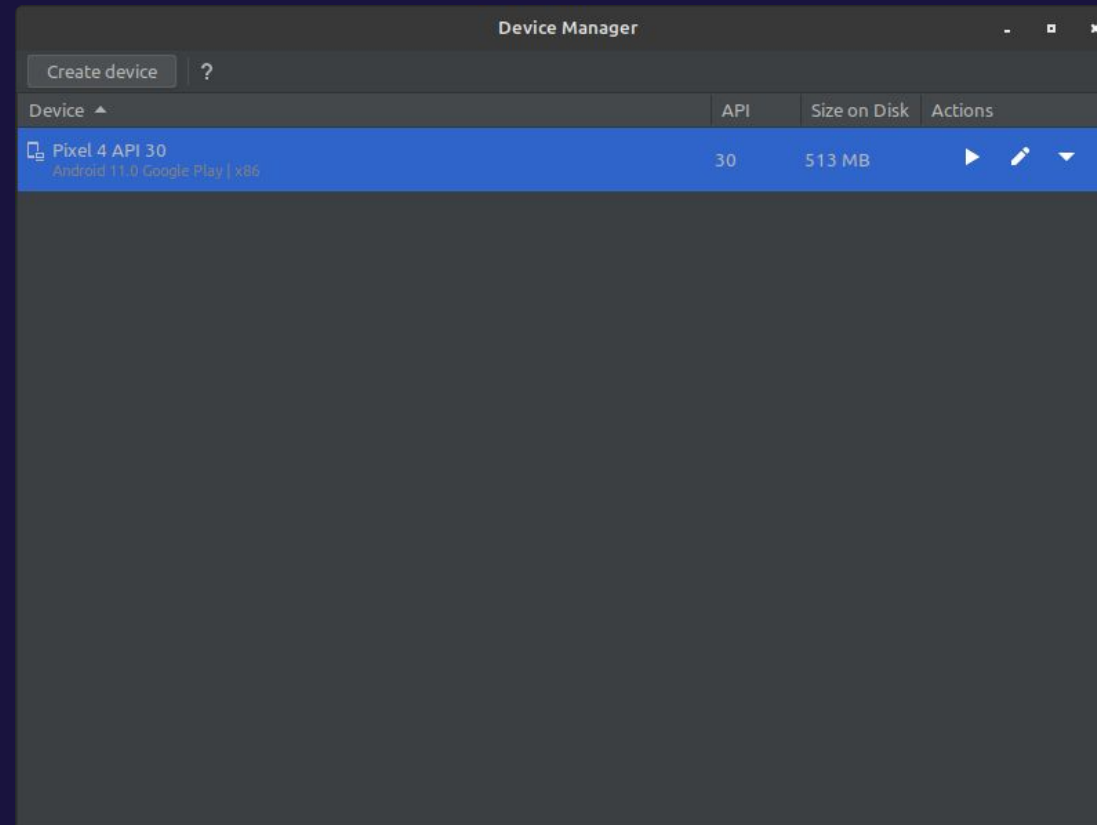
Com o Emulador criado, aperte o botão de Play para ligar o telefone virtual



CONFIGURAÇÃO DO EMULADOR



Com o Emulador criado, aperte o botão de Play para ligar o telefone virtual



CONFIGURAÇÃO DO EMULADOR



Aguarde a inicialização do sistema e o emulador estará pronto





CONFIGURAÇÃO DO GITHUB

- Crie um novo repositório público no seu github com o nome: *aulas-mobile-padolabs*
- Adicione um arquivo README.md com no mínimo o seguinte texto: *Aluno: <Seu Nome>*
- Envie o link do repositório para o e-mail: david@hausenn.com.br com o título “[PADOLABS] Repositório Aulas Mobile <Seu Nome>”

CONFIGURAÇÃO DO GITHUB



- Para a aula de hoje, crie uma pasta no seu repositório com o nome *aula1*.
- Os projetos do Android devem ser criados dentro da pasta da aula.

CONFIGURAÇÃO DO GITHUB



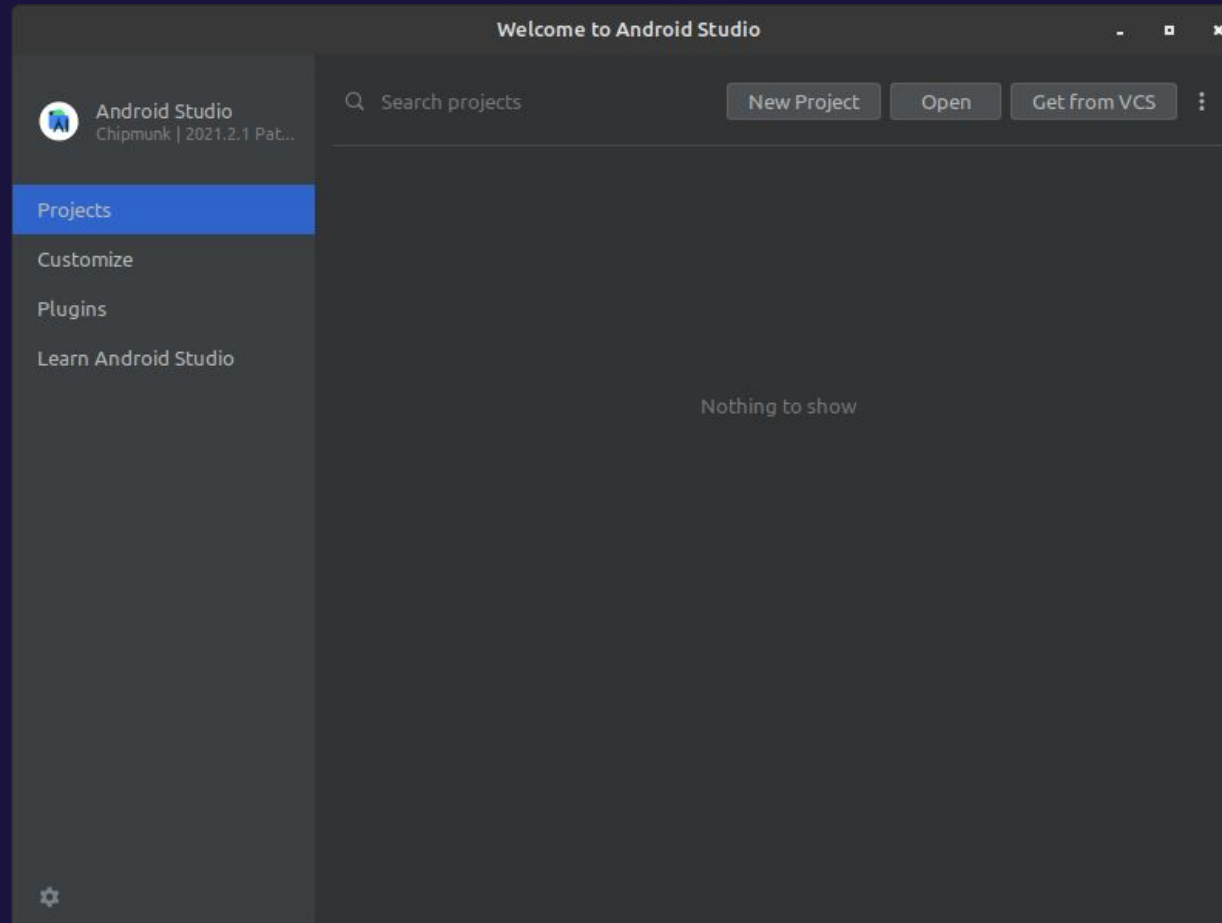
LEMBRE-SE DE SEMPRE
COMMITAR E REALIZAR O
PUSH AO FINAL DA AULA.



ATIVIDADE I

Hello World

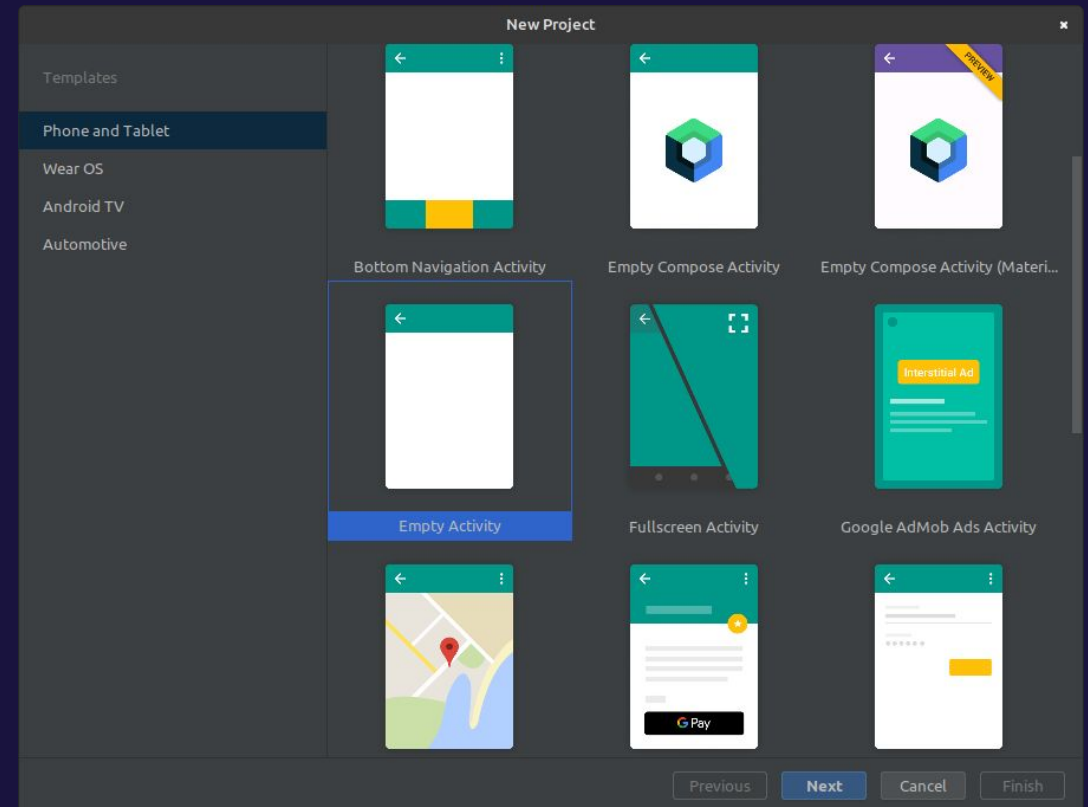
TELA DE BOAS VINDAS



Novo Projeto



- Clique no botão “Novo Projeto” e na categoria de telefones selecione a activity vazia (Empty Activity) depois Próximo (Next).



Novo Projeto

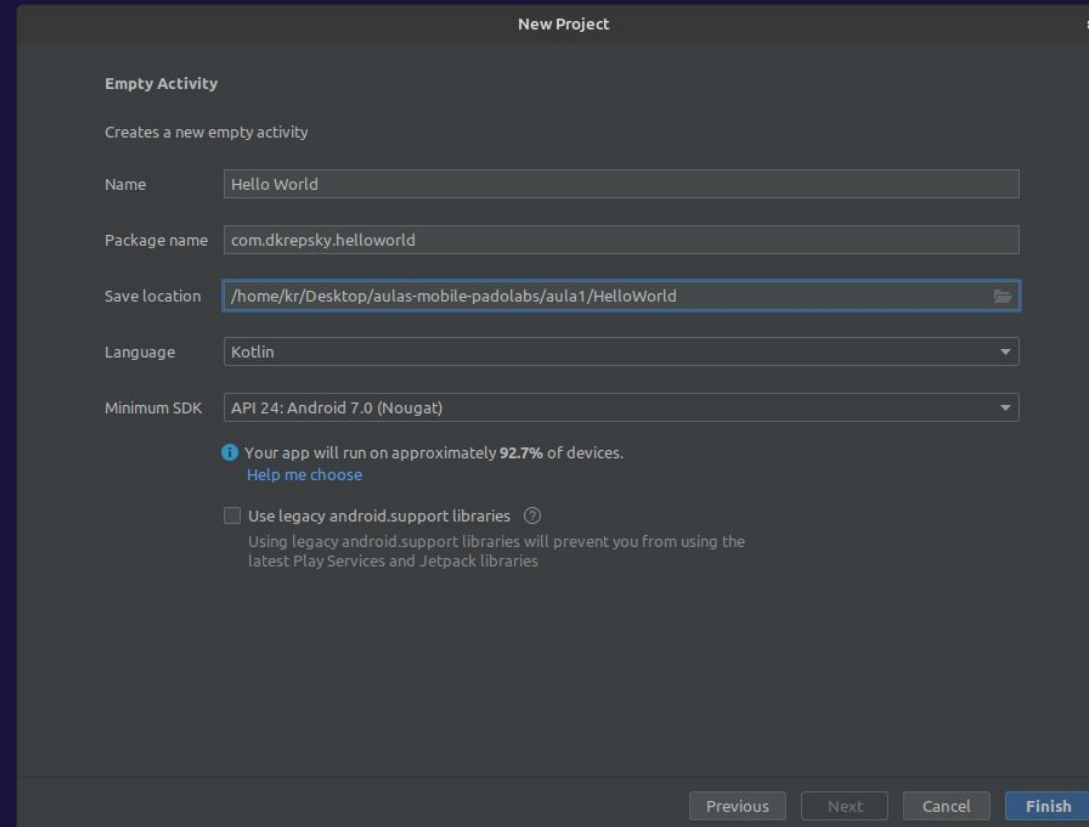


- Mude o nome do projeto para *Hello World* e selecione como local de destino a pasta da aula 1 dentro do seu repositório.

A screenshot of the 'New Project' dialog box in Android Studio. The dialog is titled 'New Project' and has a close button (X) in the top right corner. It contains several input fields and dropdown menus. The 'Name' field is set to 'Hello World'. The 'Package name' field is set to 'com.dkrepky.helloworld'. The 'Save location' field is set to '/home/kr/Desktop/aulas-mobile-padolabs/aula1/HelloWorld'. The 'Language' dropdown is set to 'Kotlin'. The 'Minimum SDK' dropdown is set to 'API 24: Android 7.0 (Nougat)'. Below these fields, there is a blue information icon followed by the text 'Your app will run on approximately 92.7% of devices.' and a link 'Help me choose'. There is also a checkbox labeled 'Use legacy android.support libraries' with a question mark icon, and a note below it stating 'Using legacy android.support libraries will prevent you from using the latest Play Services and Jetpack libraries'. At the bottom of the dialog, there are four buttons: 'Previous', 'Next', 'Cancel', and 'Finish'.

Novo Projeto

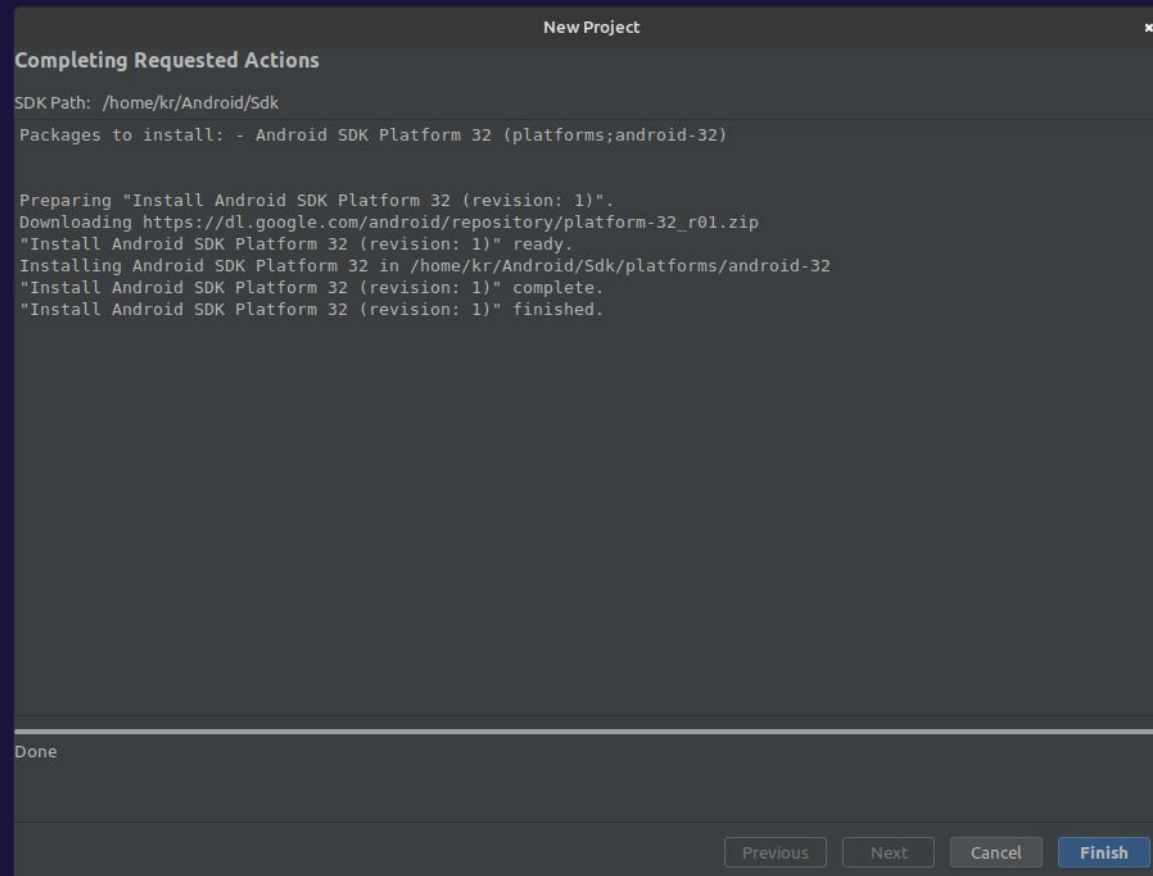
- Confira as opções selecionadas e clique em Finalizar (Finish).

A screenshot of the "New Project" dialog box in an IDE. The dialog has a dark gray background and a title bar that says "New Project" with a close button. The main content area is titled "Empty Activity" and includes a subtitle "Creates a new empty activity". Below this, there are several input fields and dropdown menus: "Name" with the value "Hello World", "Package name" with the value "com.dkrepky.helloworld", "Save location" with the value "/home/kr/Desktop/aulas-mobile-padolabs/aula1/HelloWorld", "Language" with a dropdown menu showing "Kotlin", and "Minimum SDK" with a dropdown menu showing "API 24: Android 7.0 (Nougat)". Below these fields, there is a blue information icon followed by the text "Your app will run on approximately 92.7% of devices." and a link "Help me choose". There is also a checkbox labeled "Use legacy android.support libraries" which is currently unchecked, with a help icon and a note below it stating "Using legacy android.support libraries will prevent you from using the latest Play Services and Jetpack libraries". At the bottom of the dialog, there are four buttons: "Previous", "Next", "Cancel", and "Finish". The "Finish" button is highlighted in blue.

Novo Projeto



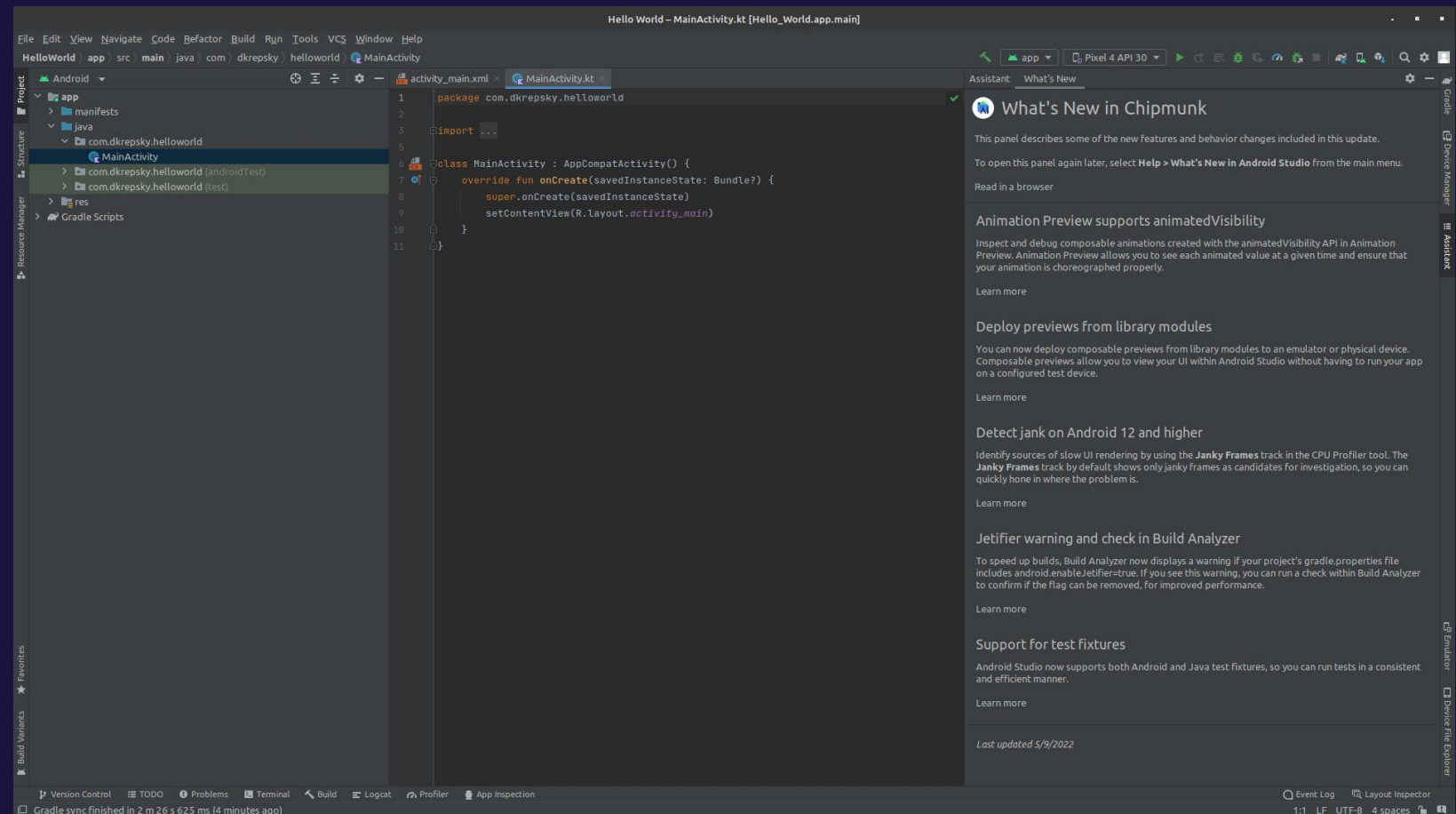
- Após a instalação da API, clique em Finalizar (Finish)



Editor do Android Studio

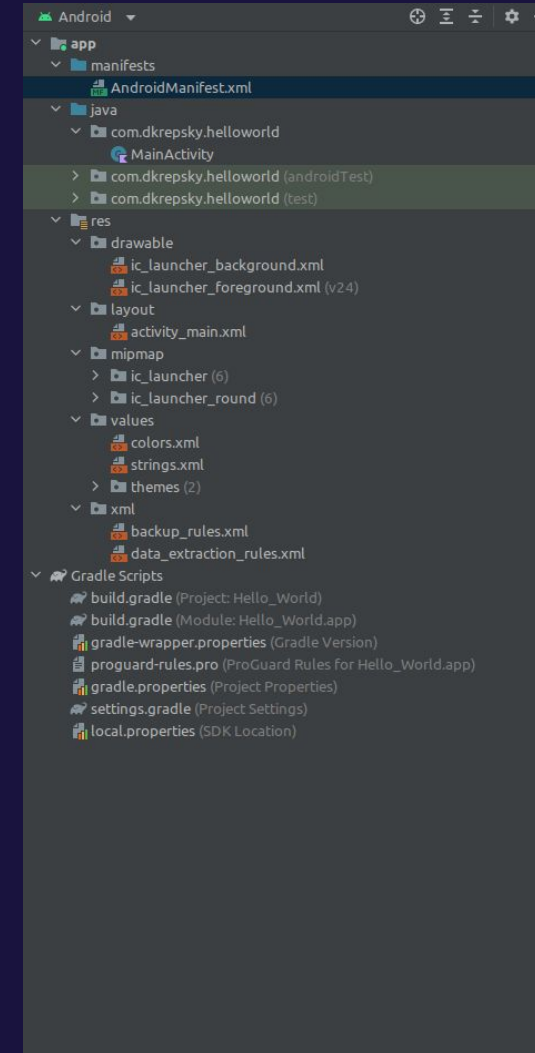


- Se tudo deu certo até aqui, o projeto foi criado com sucesso!



Arquivos do projeto

O Visualizador de arquivos do projeto encontra-se na lateral esquerda do Android Studio.



Arquivo AndroidManifest.xml



O arquivo
AndroidManifest.xml
contém diversas opções
e configurações do
aplicativo, como por
exemplo o nome do App
e o id do pacote.

```
AndroidManifest.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3    xmlns:tools="http://schemas.android.com/tools"
4    package="com.dkrepsky.helloworld">
5
6    <application
7        android:allowBackup="true"
8        android:dataExtractionRules="@xml/data_extraction_rules"
9        android:fullBackupContent="@xml/backup_rules"
10       android:icon="@mipmap/ic_launcher"
11       android:label="Hello World"
12       android:roundIcon="@mipmap/ic_launcher_round"
13       android:supportsRtl="true"
14       android:theme="@style/Theme.HelloWorld"
15       tools:targetApi="31">
16        <activity
17            android:name=".MainActivity"
18            android:exported="true">
19            <intent-filter>
20                <action android:name="android.intent.action.MAIN" />
21
22                <category android:name="android.intent.category.LAUNCHER" />
23            </intent-filter>
24        </activity>
25    </application>
26
27 </manifest>
```

Arquivos .kt

Os arquivos .kt são o código fonte do aplicativo, na linguagem kotlin.

A MainActivity.kt é o ponto de entrada do programa.

```
MainActivity.kt x
1  package com.dkrepsky.helloworld
2
3  import androidx.appcompat.app.AppCompatActivity
4  import android.os.Bundle
5
6  class MainActivity : AppCompatActivity() {
7      override fun onCreate(savedInstanceState: Bundle?) {
8          super.onCreate(savedInstanceState)
9          setContentView(R.layout.activity_main)
10     }
11 }
```

Arquivos .kt

Os arquivos .kt são o código fonte do aplicativo, na linguagem kotlin.

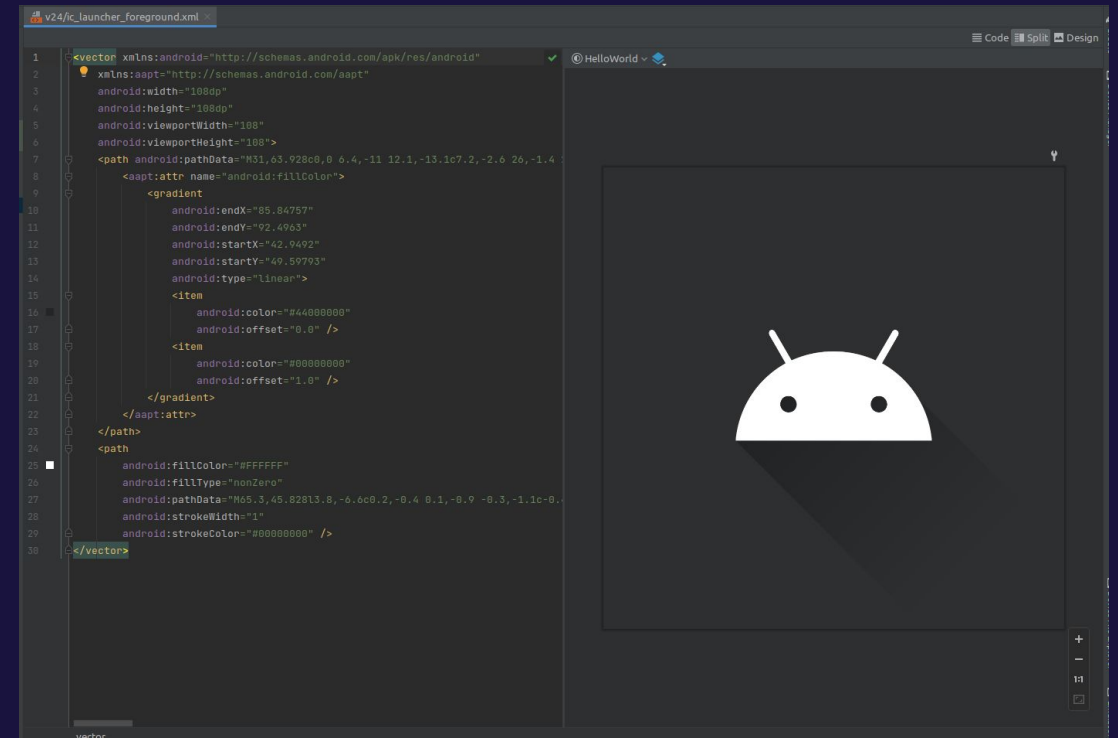
A MainActivity.kt é o ponto de entrada do programa.

```
MainActivity.kt x
1  package com.dkrepsky.helloworld
2
3  import androidx.appcompat.app.AppCompatActivity
4  import android.os.Bundle
5
6  class MainActivity : AppCompatActivity() {
7      override fun onCreate(savedInstanceState: Bundle?) {
8          super.onCreate(savedInstanceState)
9          setContentView(R.layout.activity_main)
10     }
11 }
```

Pasta res > drawable

Contém os arquivos de mídia como por exemplo ícones e imagens.

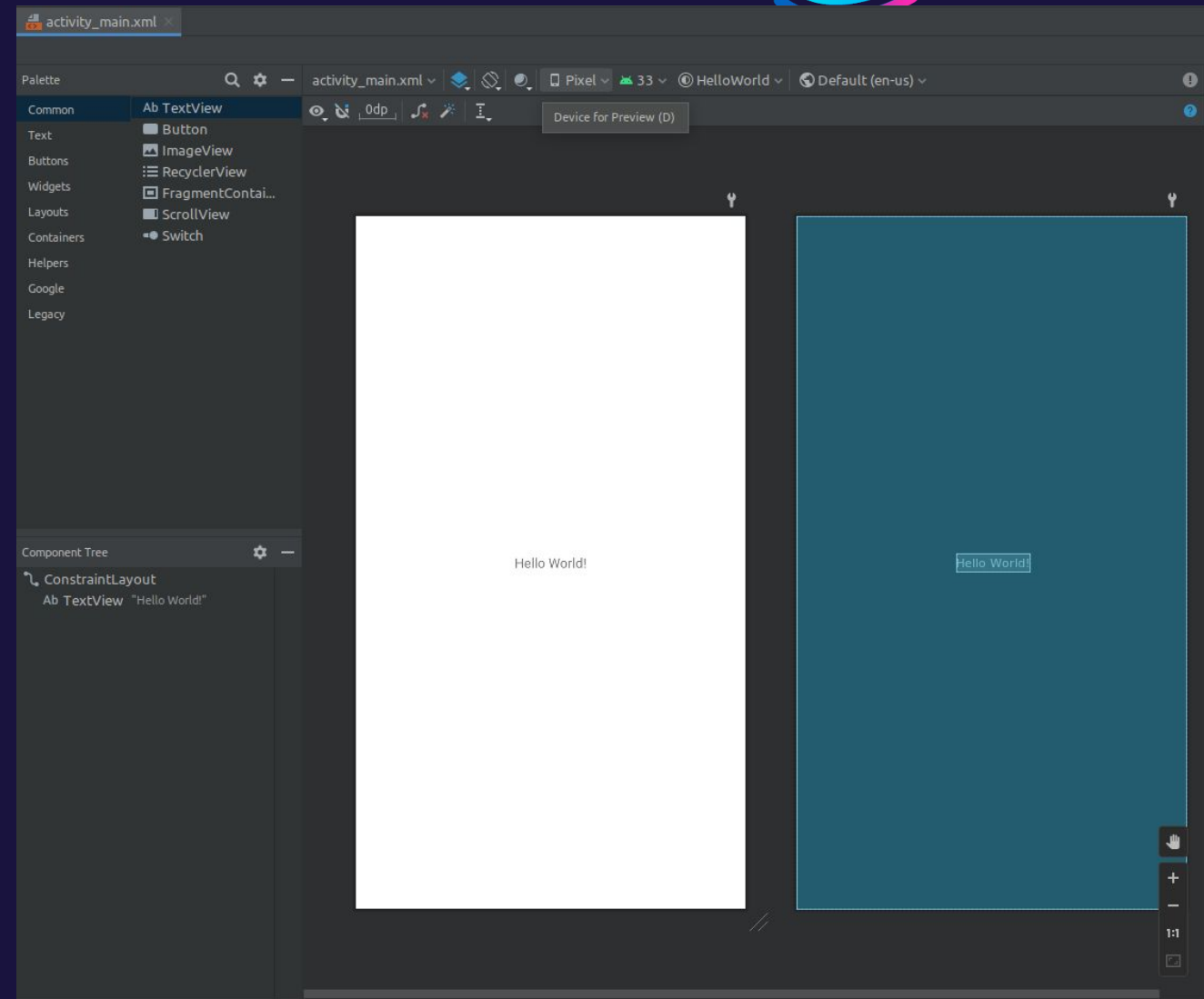
O android aceita diversos formatos de arquivos, incluindo jpg, png e svg.



Pasta res > layout

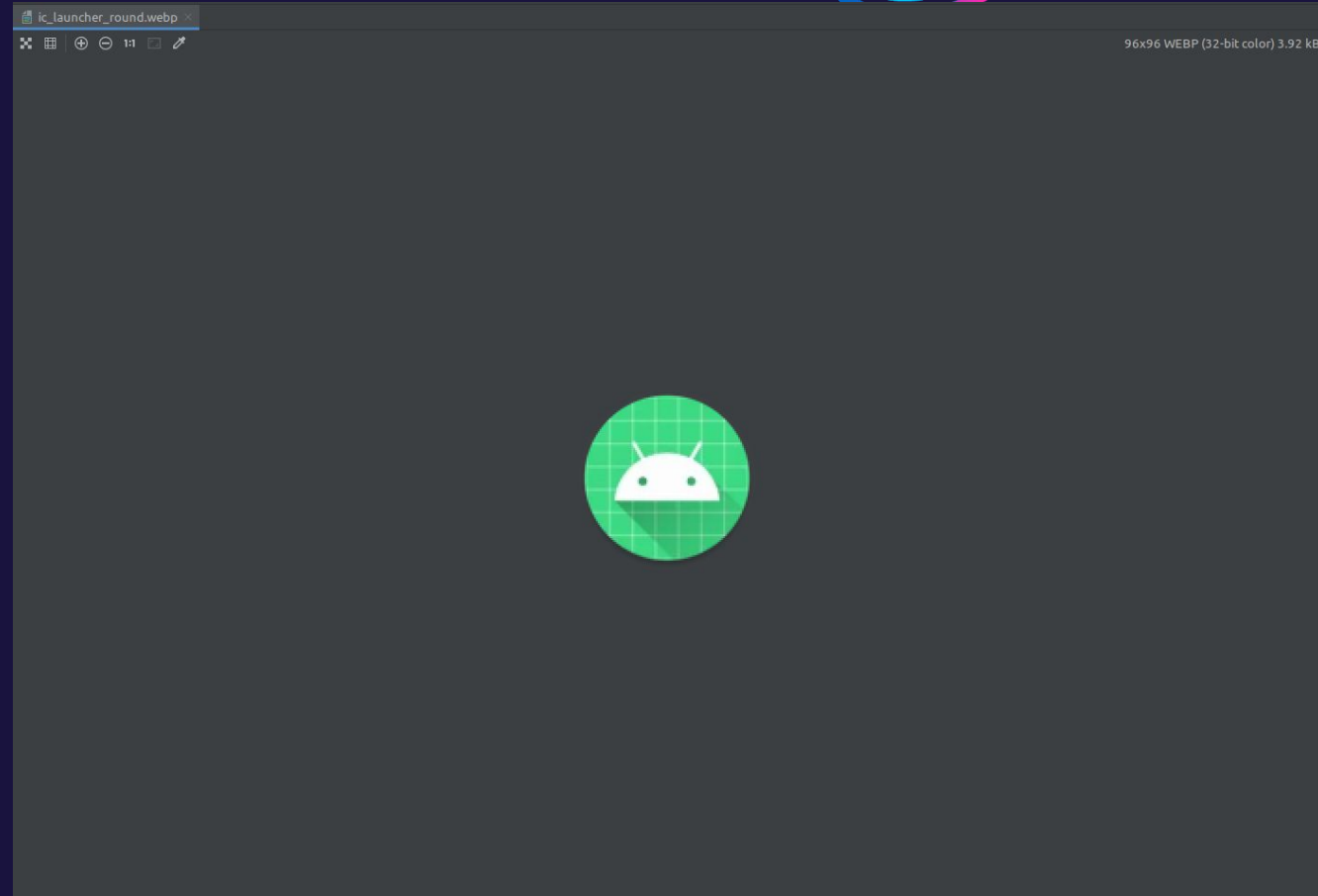
Contém os arquivos de estrutura das páginas.

Os arquivos de estrutura são utilizados no editor de tela para facilitar a construção das interfaces do aplicativo.



Pasta res > mipmap

Contém os arquivos de imagens em resoluções diferentes, pré-renderizados.



Pasta res > values > colors.xml

No arquivo colors.xml, são criadas as definições de cores.

O Android recomenda concentrar toda a definição de cores nesse arquivo, para facilitar a criação de temas.

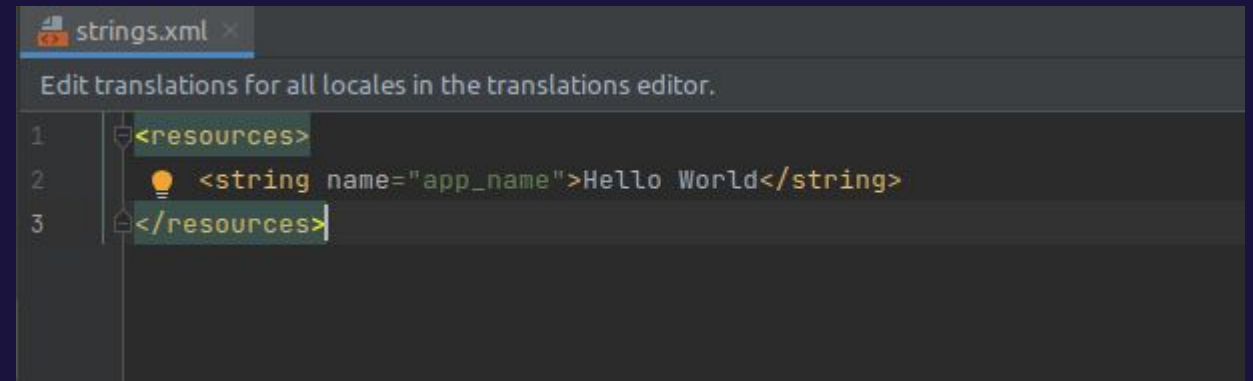
A screenshot of a code editor showing the contents of a file named "colors.xml". The editor has a dark background with light-colored text. On the left side, there is a vertical list of line numbers from 1 to 10. To the right of the line numbers, the XML code is displayed. The code starts with an XML declaration on line 1, followed by an opening <resources> tag on line 2. Lines 3 through 9 contain individual <color> tags with names and hex values. Line 10 contains the closing </resources> tag. Each line of code is preceded by a small colored square icon: a purple square for line 3, a blue square for line 4, a teal square for line 5, a black square for line 8, and a white square for line 9. The icons for lines 6 and 7 are not visible.

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3     <color name="purple_200">#FFBB86FC</color>
4     <color name="purple_500">#FF6200EE</color>
5     <color name="purple_700">#FF3700B3</color>
6     <color name="teal_200">#FF03DAC5</color>
7     <color name="teal_700">#FF018786</color>
8     <color name="black">#FF000000</color>
9     <color name="white">#FFFFFFFF</color>
10 </resources>
```

Pasta res > values > strings.xml

Todos os textos do aplicativo são colocados no arquivo strings.xml.

O Android Studio utiliza esse arquivo para facilitar a construção de apps multi-linguagem com o uso de uma ferramenta chamada Translations Editor.

A screenshot of the Android Studio interface showing the 'strings.xml' file. The file is open in the 'Translations Editor' tab, which has a title bar that says 'strings.xml' and a close button. Below the title bar, there is a message: 'Edit translations for all locales in the translations editor.' The main area shows the XML code for the strings file. Line 1 contains the opening tag '<resources>'. Line 2 contains a string resource: '<string name="app_name">Hello World</string>'. Line 3 contains the closing tag '</resources>'. The code is syntax-highlighted, with tags in green and text in white. A lightbulb icon is visible next to the string resource, indicating a suggestion or tip.

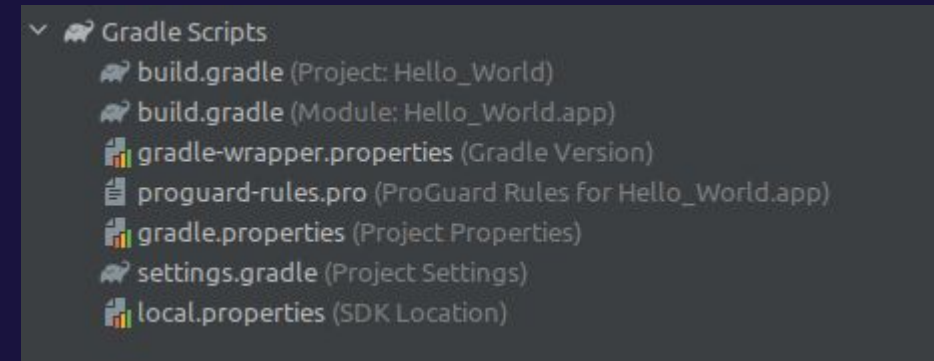
Pasta res > themes

Os arquivos dessa pasta permitem a criação de temas para o aplicativo, como por exemplo o tema Modo Escuro.

```
night/themes.xml
1 <resources xmlns:tools="http://schemas.android.com/tools">
2   <!-- Base application theme. -->
3   <style name="Theme.HelloWorld" parent="Theme.MaterialComponents.DayNight.DarkActionBar">
4     <!-- Primary brand color. -->
5     <item name="colorPrimary">@color/purple_200</item>
6     <item name="colorPrimaryVariant">@color/purple_700</item>
7     <item name="colorOnPrimary">@color/black</item>
8     <!-- Secondary brand color. -->
9     <item name="colorSecondary">@color/teal_200</item>
10    <item name="colorSecondaryVariant">@color/teal_200</item>
11    <item name="colorOnSecondary">@color/black</item>
12    <!-- Status bar color. -->
13    <item name="android:statusBarColor" tools:targetApi="l">?attr/colorPrimaryVariant</item>
14    <!-- Customize your theme here. -->
15  </style>
16 </resources>
```

Gradle

O Gradle é a ferramenta utilizada para compilar o aplicativo e gerenciar as dependências do projeto.



Executando nosso aplicativo Hello World no emulador



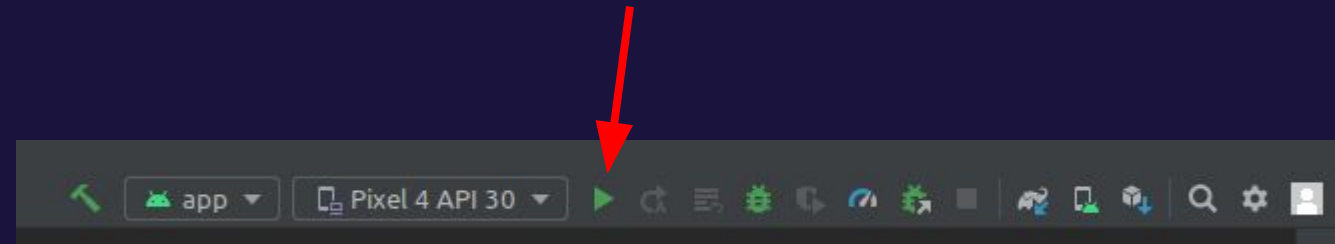
Na barra de ferramentas, verifique se o dispositivo selecionado é o emulador configurado anteriormente.



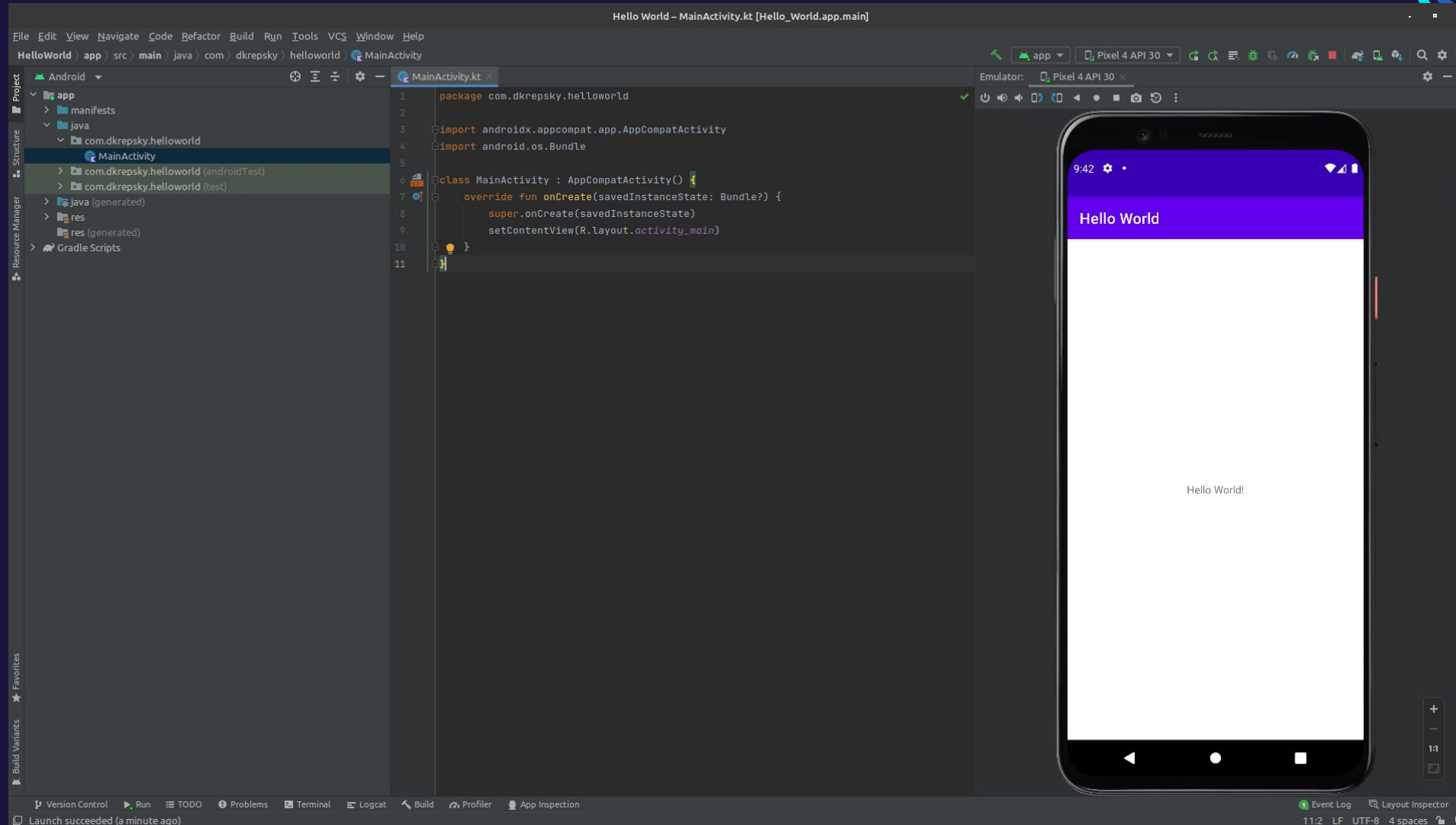
Executando nosso aplicativo Hello World no emulador

Pressione o botão de Play para rodar o app no emulador.

Caso o emulador esteja desligado, o Android Studio irá iniciar o emulador para você.



Parabéns! Seu primeiro App foi Criado :)



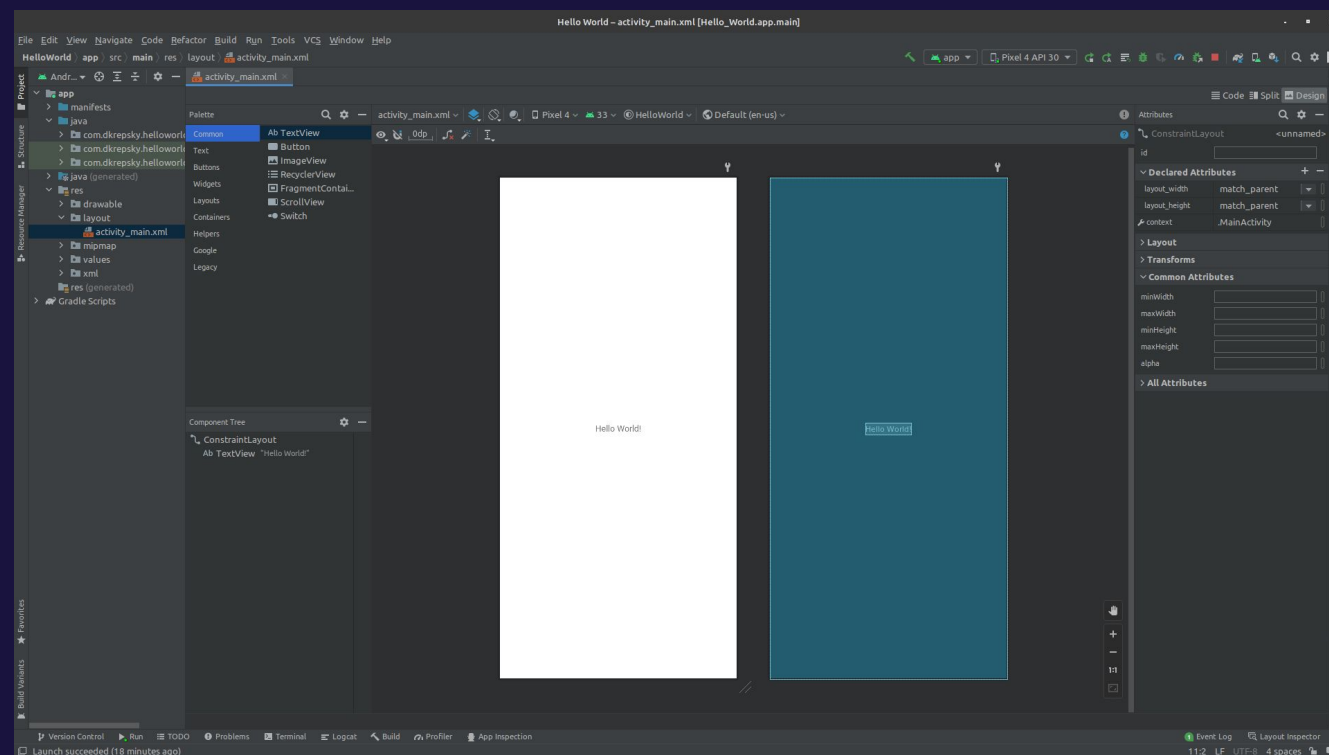
Próximo Passo

Nosso próximo passo é adicionar um Input para obter o nome do usuário e um botão, que quando pressionado, mostra a frase: “Hello *Nome*”.

O Editor de telas



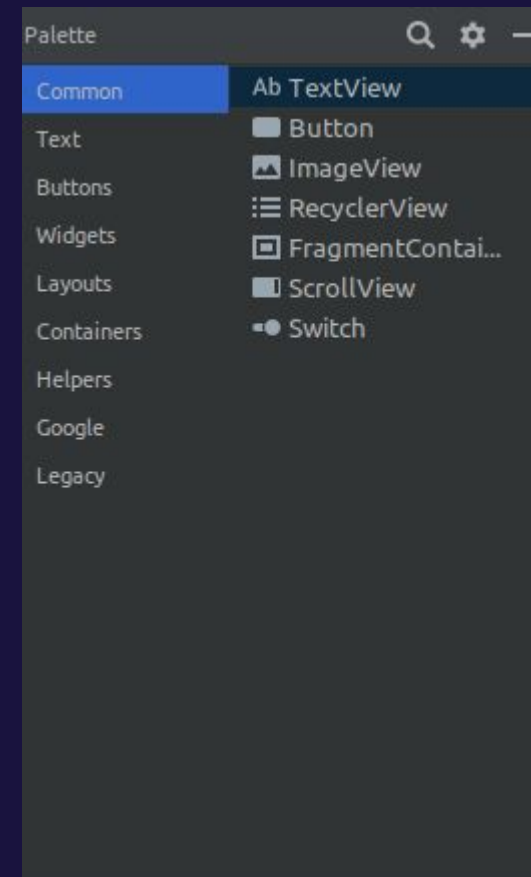
O Android Studio possui uma Ferramenta bastante útil que é o Editor de Telas. Nele conseguimos criar e editar as telas do aplicativo de modo visual.



O Editor de telas

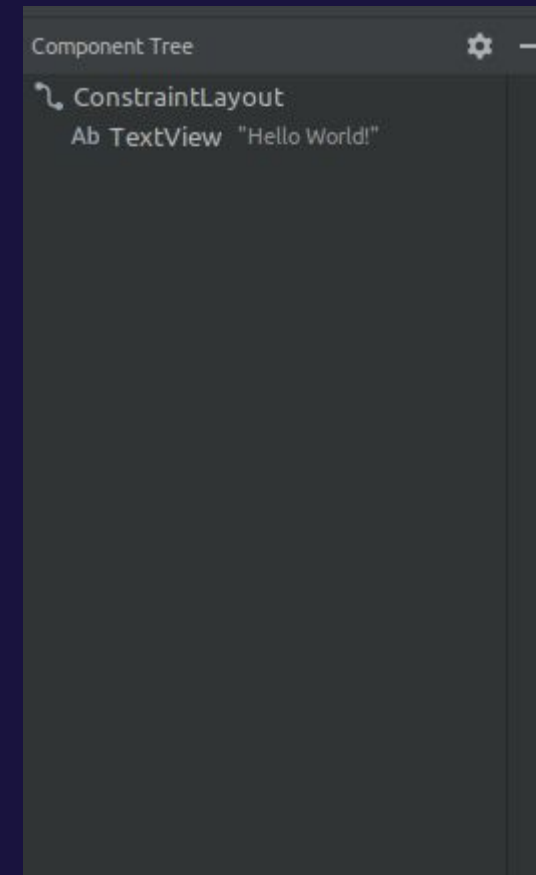
No menu Palette, encontramos diversos tipos de componentes que são amplamente utilizados, tais como botões e imagens.

Também estão disponíveis componentes estruturais de layout.



O Editor de telas

No menu Component Tree, encontramos a estrutura do layout.

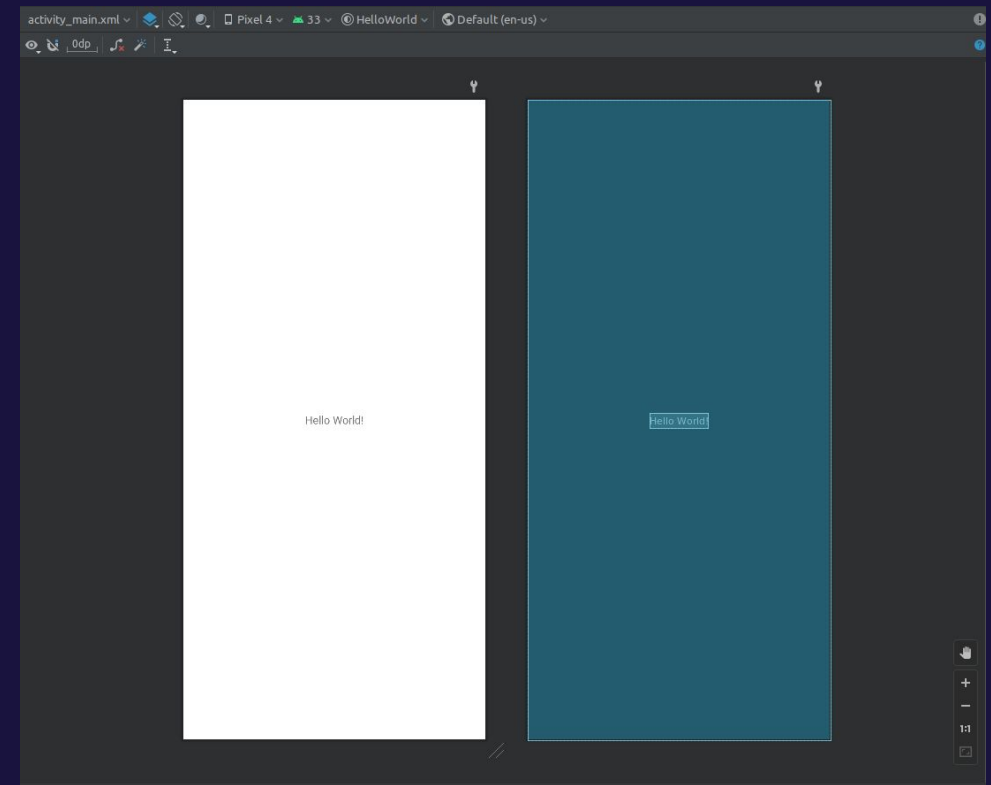


O Editor de telas

Na parte central, encontramos o visualizador de páginas.

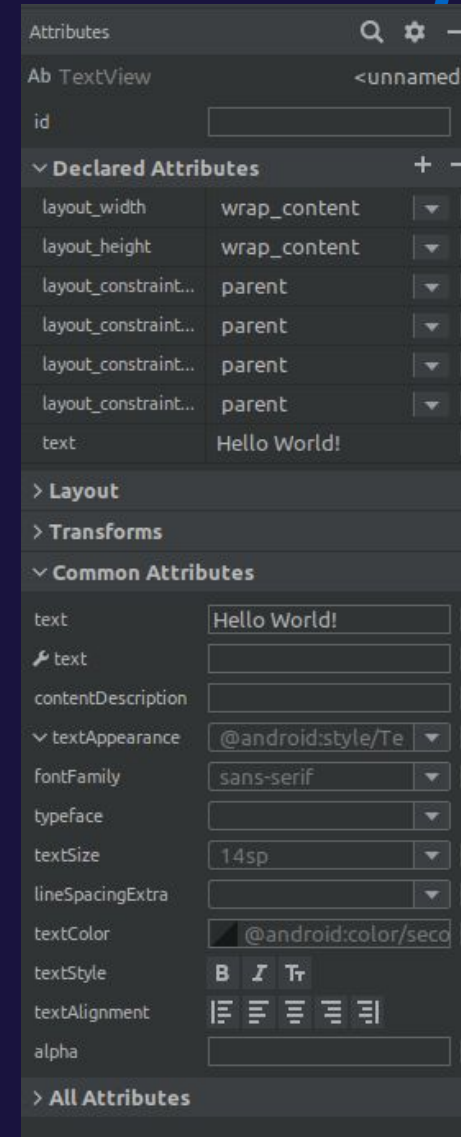
A tela em branco é a visualização de design, como o usuário vê.

A tela em azul é o Blueprint que mostra como a tela está estruturada e os *constraints* utilizados.



O Editor de telas

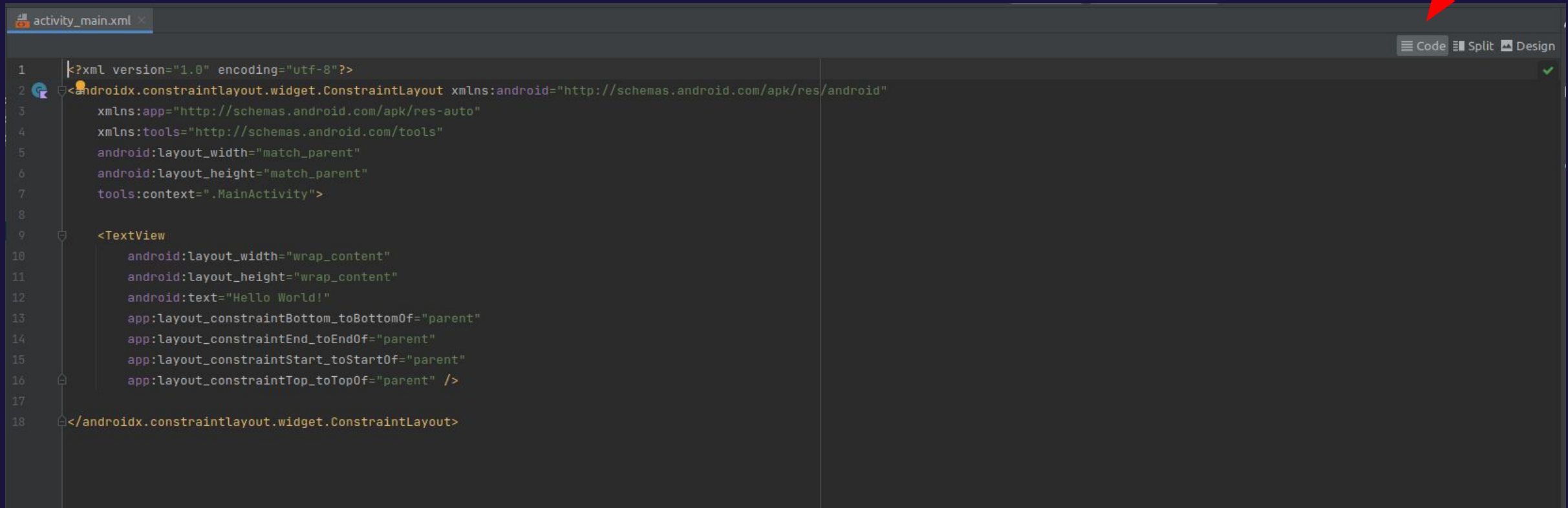
No menu Attributes, estão localizados todos os atributos do componente selecionado na visualização.



O Editor de telas



Todas as telas no android são arquivos XML, com TAGs e Atributos, similar ao HTML.

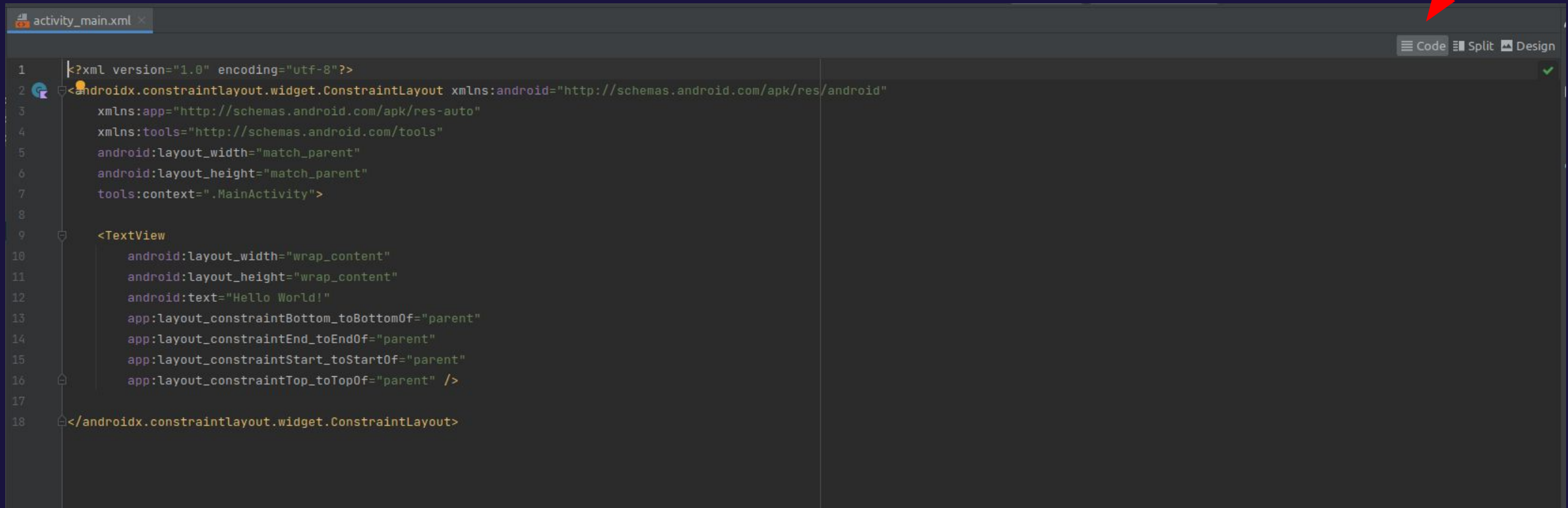
A screenshot of the Android Studio interface showing the XML editor for 'activity_main.xml'. The editor is in 'Code' mode, as indicated by the selected tab in the top right corner. A red arrow points to this tab. The XML code is displayed with line numbers on the left. The code defines a ConstraintLayout containing a TextView with the text 'Hello World!'.

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     tools:context=".MainActivity">
8
9     <TextView
10         android:layout_width="wrap_content"
11         android:layout_height="wrap_content"
12         android:text="Hello World!"
13         app:layout_constraintBottom_toBottomOf="parent"
14         app:layout_constraintEnd_toEndOf="parent"
15         app:layout_constraintStart_toStartOf="parent"
16         app:layout_constraintTop_toTopOf="parent" />
17
18 </androidx.constraintlayout.widget.ConstraintLayout>
```

O Editor de telas



Todas as telas no android são arquivos XML, com TAGs e Atributos, similar ao HTML.

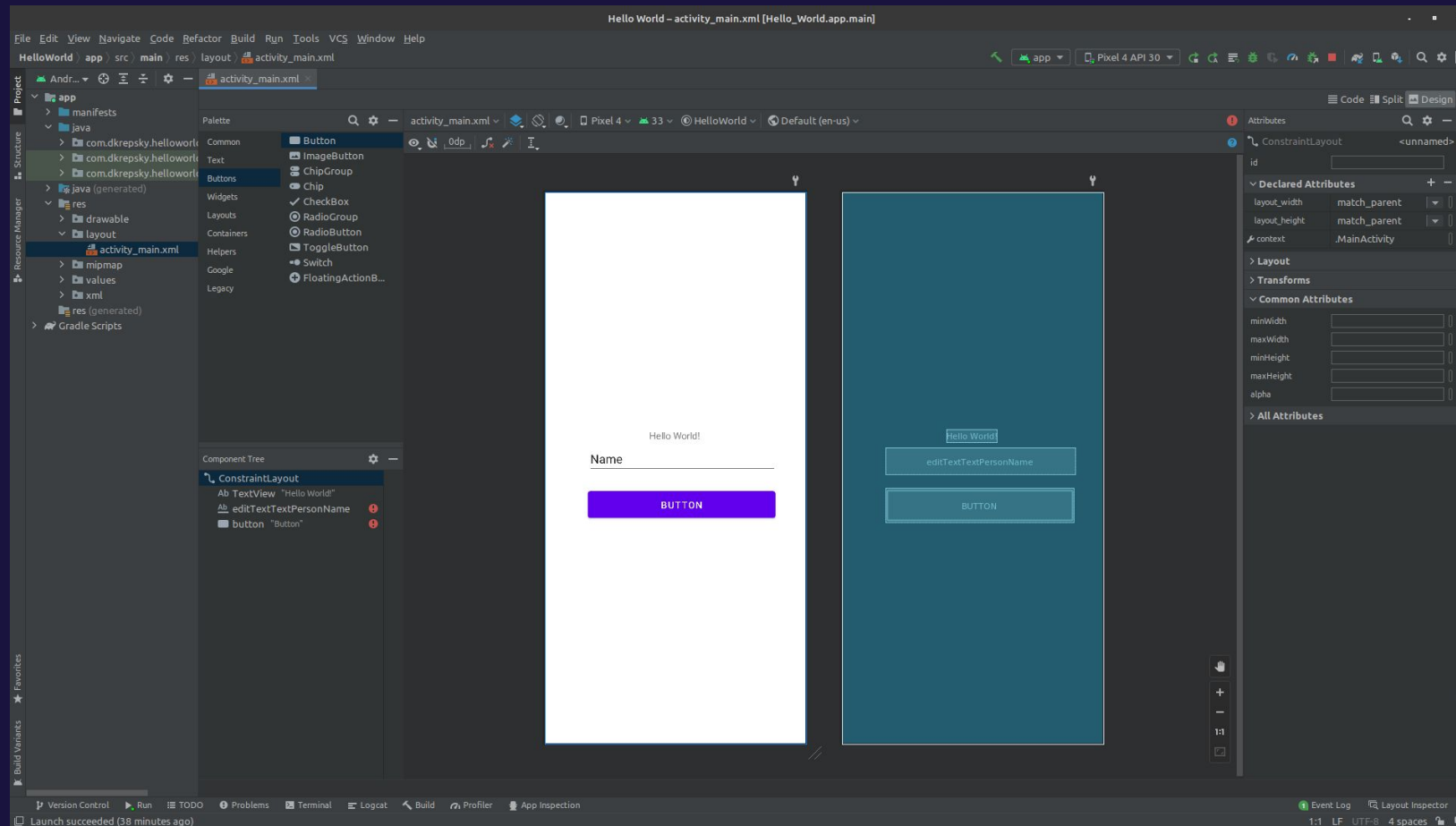
A screenshot of the Android Studio interface. The top bar shows the file name "activity_main.xml" and tabs for "Code", "Split", and "Design". A red arrow points to the "Design" tab. The main editor area displays XML code for a ConstraintLayout. The code includes a root layout tag with various attributes and a nested TextView tag with its own attributes and constraints.

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3   xmlns:app="http://schemas.android.com/apk/res-auto"
4   xmlns:tools="http://schemas.android.com/tools"
5   android:layout_width="match_parent"
6   android:layout_height="match_parent"
7   tools:context=".MainActivity">
8
9   <TextView
10     android:layout_width="wrap_content"
11     android:layout_height="wrap_content"
12     android:text="Hello World!"
13     app:layout_constraintBottom_toBottomOf="parent"
14     app:layout_constraintEnd_toEndOf="parent"
15     app:layout_constraintStart_toStartOf="parent"
16     app:layout_constraintTop_toTopOf="parent" />
17
18 </androidx.constraintlayout.widget.ConstraintLayout>
```


Adicionando Componentes



Adicione um componente Plain Text e um Button a tela

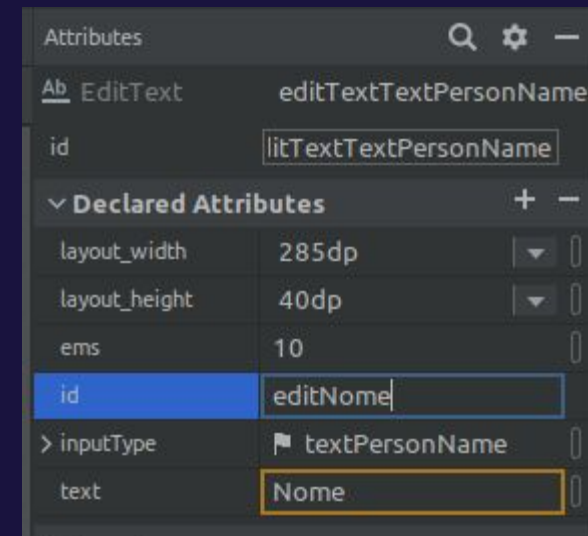


Adicionando Componentes

Selecione o Input e na aba Attributes, mude os campos:

- id: editNome
- text: Nome

Confirme a alteração clicando em *Refactor*.



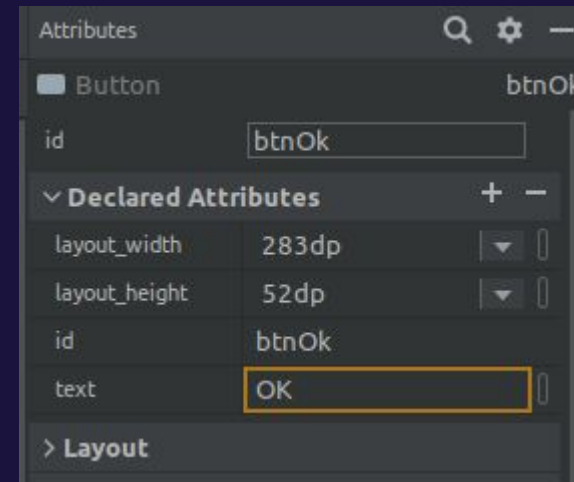
Adicionando Componentes



Selecione o Botão e na aba Attributes, mude os campos:

- id: btnOk
- text: OK

Confirme a refatoração.

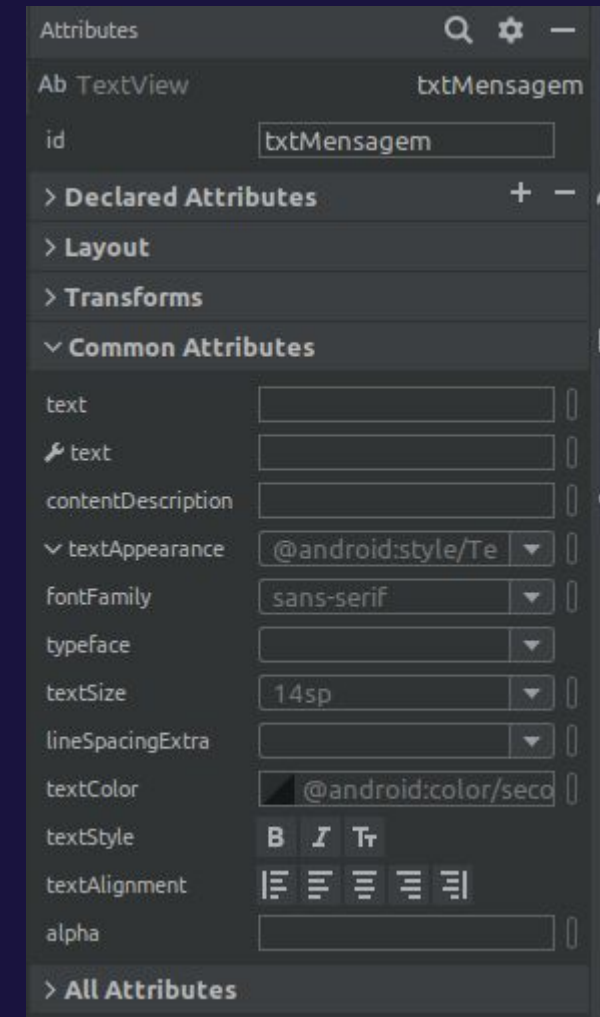


Adicionando Componentes

Selecione o texto Hello World e na aba Attributes, mude os campos:

- id: txtMensagem
- text:

Confirme a refatoração.



Adicionando Componentes



Ao final, sua tela deve estar da seguinte forma:

A vertical rectangular mockup of a mobile application screen with a white background and a dark grey border. Near the bottom, there is a text input field with the label "Nome" above it. Below the input field is a solid blue rectangular button with the text "OK" centered on it.

Adicionando Funcionalidade

No desenvolvimento Web, as páginas são estruturadas em HTML e as funcionalidades são adicionadas em JavaScript.

No Android, as páginas são estruturadas em XML e a funcionalidade é adicionada na linguagem Kotlin.

Anatomia de uma aplicação Android



```
MainActivity.kt x
1  package com.dkrepky.helloworld
2
3  import androidx.appcompat.app.AppCompatActivity
4  import android.os.Bundle
5
6  class MainActivity : AppCompatActivity() {
7      override fun onCreate(savedInstanceState: Bundle?) {
8          super.onCreate(savedInstanceState)
9          setContentView(R.layout.activity_main)
10     }
11 }
```

Package

Packages ou “pacotes” é um conjunto de arquivos de código fonte que possuem funcionalidade similar.

Packages são a base da estrutura de modularização do Kotlin.

Import

O import permite a inclusão de código de outros pacotes. Similar ao `#include` do C.

Activity e MainActivity

No android, toda tela apresentada ao usuário é uma Activity.

Um app deve possuir no mínimo uma Activity, que é a primeira tela do app.

Em nosso exemplo, a tela principal é a MainActivity.

Activity e MainActivity



Uma Activity é uma classe em Kotlin e podemos adicionar métodos e propriedades a essa classe.

Activity onCreate

- Quando o app é aberto pelo usuário, a função onCreate é chamada.
- Nessa função devemos definir qual o layout de tela será apresentado ao usuário e realizar o setup inicial da nossa aplicação.

Resources

Para manipularmos os elementos criados dentro da pasta `res` no nosso código em Kotlin, precisamos de uma referência para o elemento a ser manipulado.

O Android provê o objeto *R*, onde conseguimos obter um identificador para cada um dos chamados “*resources*” da nossa aplicação.

Resources

Com R, temos uma referência para nossos arquivos de layout, os componentes que adicionamos no Layout, imagens, cores, strings, temas e muito mais.

Resources

- Obtendo uma referência ao nosso arquivo de layout: *R.layout.activity_main*
- Referência ao botão que criamos: *R.id.btnOk*
- Referência a uma cor do arquivo colors.xml:
R.color.purple_200

Resources

- Obtendo uma referência ao nosso arquivo de layout: *R.layout.activity_main*
- Referência ao botão que criamos: *R.id.btnOk*
- Referência a uma cor do arquivo colors.xml:
R.color.purple_200

Definindo o layout da Activity

- O Layout da nossa activity é definido com o uso da função *setContentView*.
- *setContentView* recebe a referência do arquivo de layout a ser utilizado.

Acesso ao botão, input e mensagem

Para acessar os métodos e propriedades dos elementos que adicionamos à tela, de forma similar ao HTML & JS, precisamos criar objetos Kotlin atrelados aos elementos da tela.

Para isso utilizamos a função *findViewById*, passando como parâmetro a referência do elemento que estamos buscando.

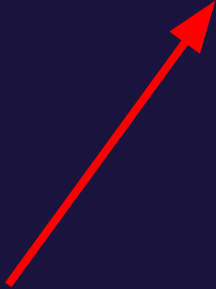
Acesso ao botão, input e mensagem

Exemplo:

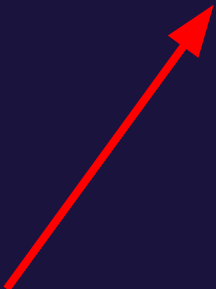
// Obter elemento input

```
val editNome = findViewById<EditText>(R.id.editNome)
```

Cria Variável
Kotlin



Função que
procura o
elemento



Classe do
Elemento



ID do
elemento



Acesso ao botão, input e mensagem

Após obter o elemento, é possível acessar seus métodos e propriedades, exemplo:

```
editNome.setText("Digite o seu nome")
```

Acesso ao botão, input e mensagem

Com o elemento, podemos também atribuir funções de tratamento de eventos. Exemplo:

```
btnOk.setOnClickListener {  
    // Código de tratamento do evento  
}
```

Acesso ao botão, input e mensagem

Com o elemento, podemos também atribuir funções de tratamento de eventos. Exemplo:

```
btnOk.setOnClickListener {  
    // Código de tratamento do evento  
}
```

Construção da lógica do aplicativo

Com os conhecimentos adquiridos, podemos construir a lógica do nosso aplicativo.

- Quando o usuário clicar no botão OK, se o input estiver vazio, aparecerá a mensagem: *“Por favor, informe seu nome”*.
- Quando o usuário clicar no botão OK, se o input não estiver vazio, aparecerá a mensagem: *“Olá <nome>, seja bem vindo ao mundo mobile”*.



```
package com.dkrepsky.helloworld

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Button
import android.widget.EditText
import android.widget.TextView

// Classe Main do nosso app.
class MainActivity : AppCompatActivity() {

    // Função executada quando o app é aberto pelo usuário.
    override fun onCreate(savedInstanceState: Bundle?) {

        // Executa o método onCreate da classe Pai
        // para que o Android inicialize outras coisas.
        super.onCreate(savedInstanceState)

        // Define o layout de tela a ser utilizado.
        setContentView(R.layout.activity_main)

        // Obtém os elementos criados no layout.
        val txtMensagem = findViewById<TextView>(R.id.txtMensagem)
        val btnOk = findViewById<Button>(R.id.btnOk)
        val editNome = findViewById<EditText>(R.id.editNome)

        // Adiciona um "listener" ao evento de onClick do botão.
        btnOk.setOnClickListener { it: View?

            // Define a mensagem de acordo com o conteúdo do input de texto.
            if(!editNome.text.isEmpty()){
                txtMensagem.setText("Olá " + editNome.text.trim() + " seja bem vindo ao mundo mobile")
            } else {
                txtMensagem.setText("Por favor, informe seu nome")
            }
        }
    }
}
```


Funções extra

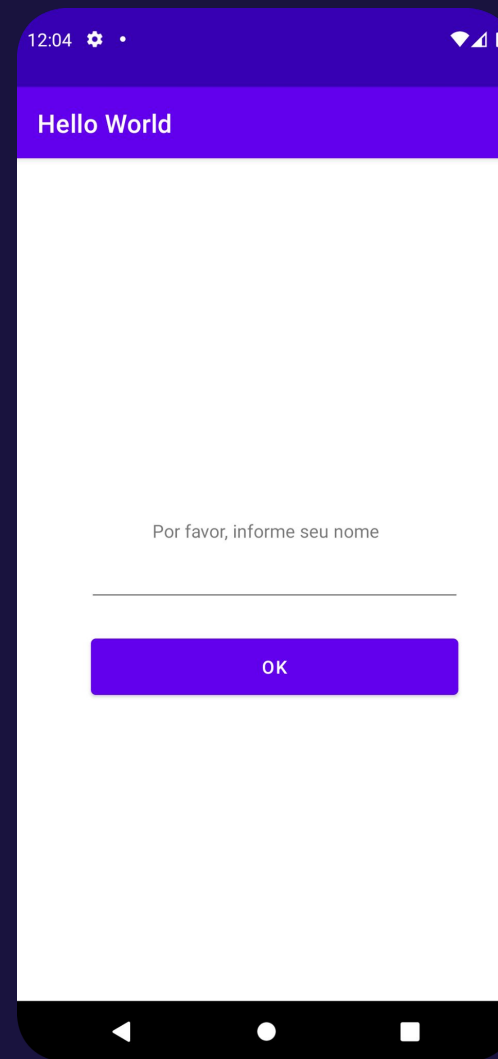
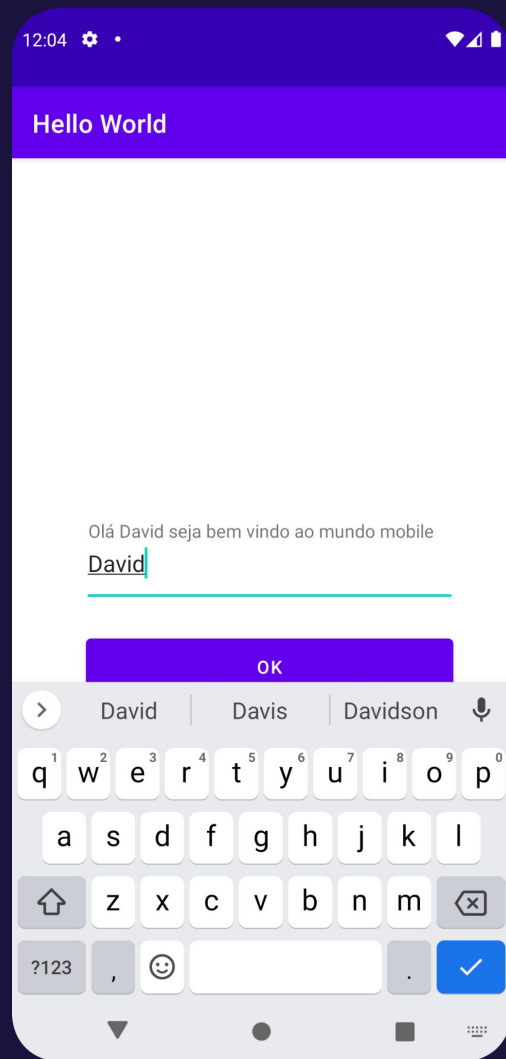
- isEmpty: verifica se uma string está vazia, retorna true ou false.
- trim: remove os espaços do começo e final de uma string.



Rodando nosso aplicativo

Pressione o botão de *Play* novamente para ver o aplicativo em funcionamento.

Rodando nosso aplicativo



Parabéns! Seu primeiro aplicativo foi um sucesso!



Dê o Commit no seu código e realize o push para o GitHub.



ATIVIDADE II

Calculadora de IMC

Atividade II



Crie um aplicativo que calcula o Índice de Massa Corpórea de uma pessoa.

<https://www.tuasaude.com/calculadora/imc/>

<https://www.programasaudefacil.com.br/calculadora-de-imc>

Subir no GitHub até a próxima Aula



PADO
Labs