FUNDAÇÃO INSTITUTO DE EDUCAÇÃO DE BARUERI

Instituto Técnico de Barueri “Brasílio Flores de Azevedo”

Tecnologias Emergentes

Tutorial Android para Gerar QRCode

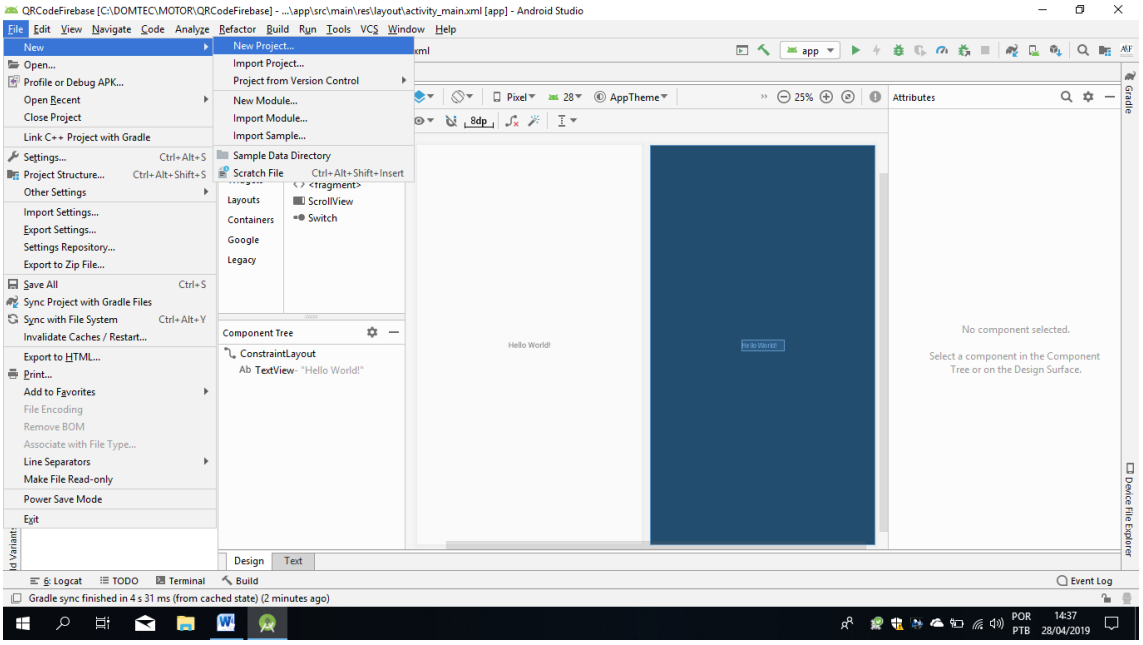
versão 1.0

Prof. Adriano Domingues

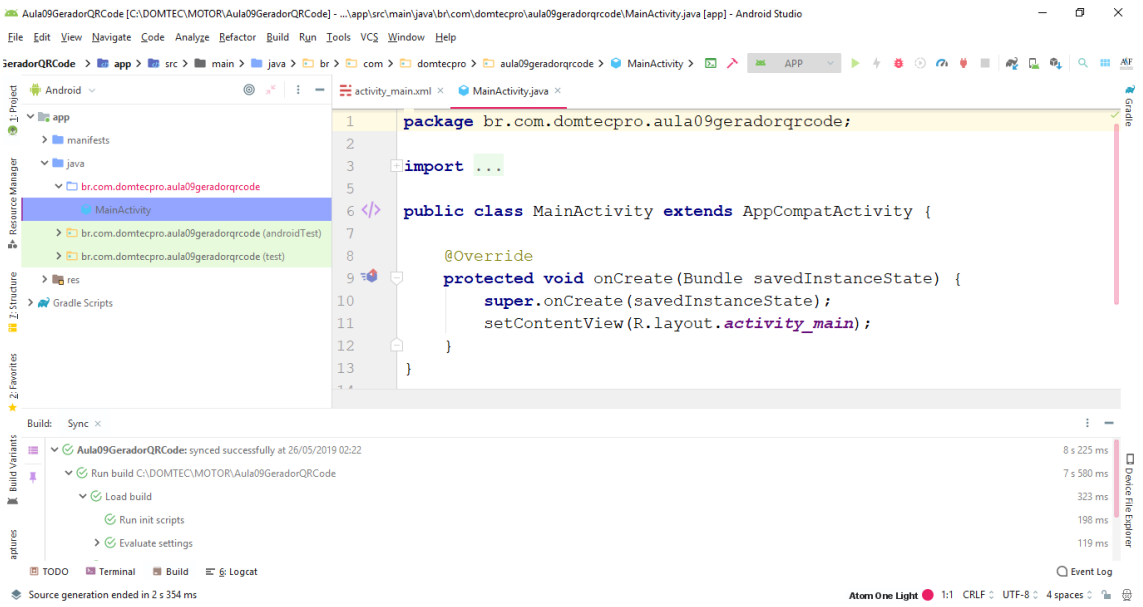
2019

AULA 09 – Gerar QRCode

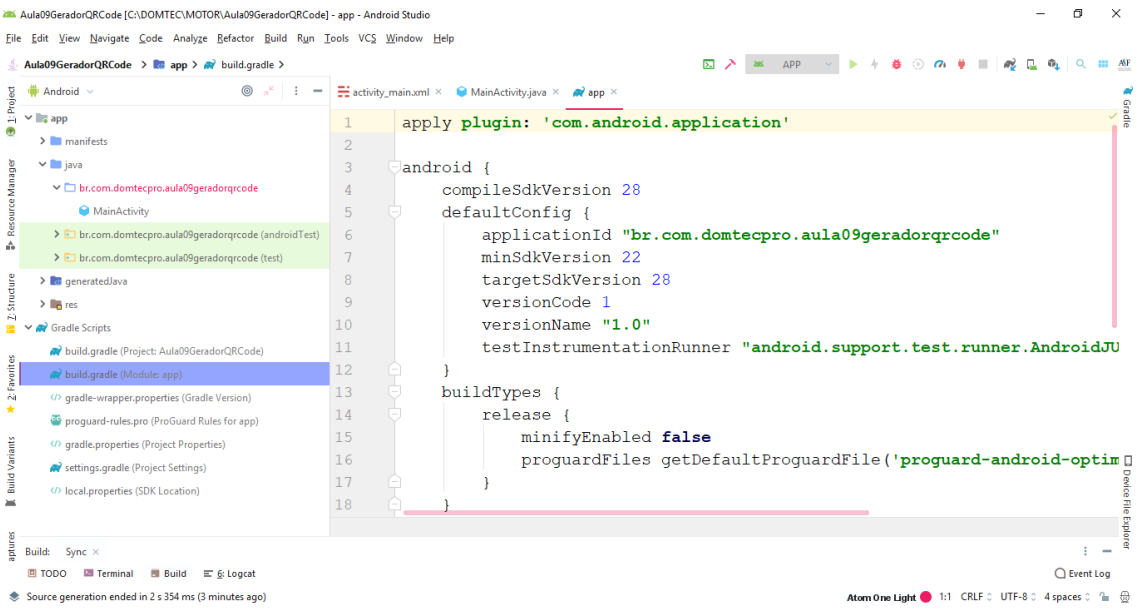
1. CRIAR PROJETO:
    a. FILE > NEW > NEW PROJECT...



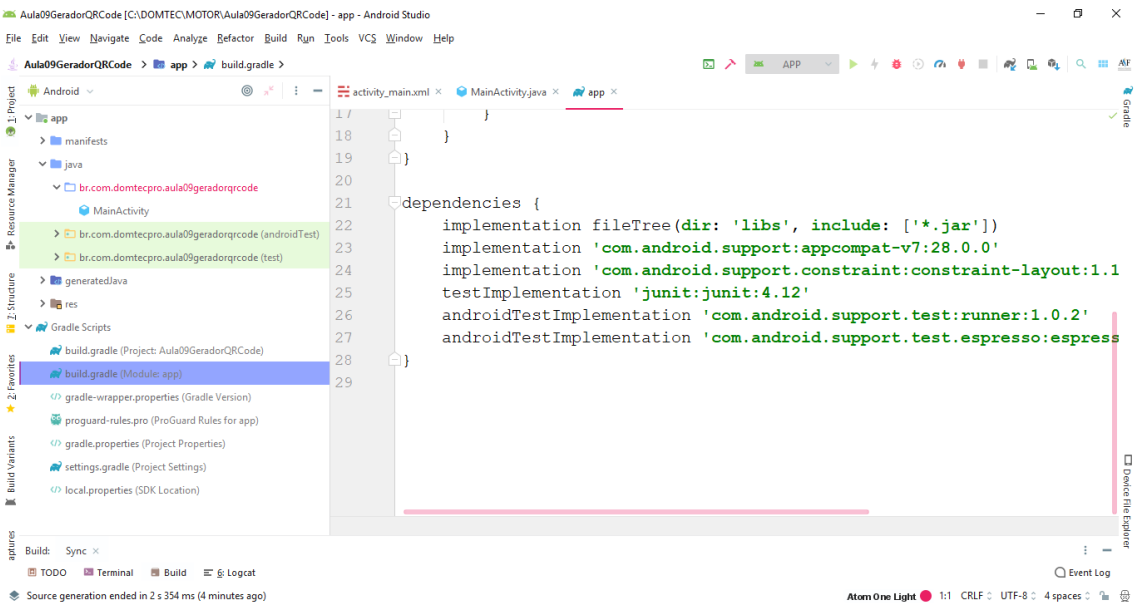
    a. Altere as opções da primeira janela e clique no Finish:
        i. Aplication Name: Aula09GeradorQRCode
        ii. Company Domain: aula09geradorqrcode.itb.com.br
        iii. Location: Z:\... (sua pasta na Z:)
    b. API 22
    c. ESCOLHA A ATIVIDADE DE EXEMPLO: Empty Activity
        iv. Não altere nada na última janela e clique em Finish
    d. Aguarde o carregamento do projeto:



e. Vamos adicionar uma dependência, isto é, uma biblioteca externa ou também chamada de API, traz classes prontas e desenvolvidas para um determinado fim, neste caso para gerar QRCodes.
    a. Clique em Android > Gradle Scripts > build.gradle (Module: app)



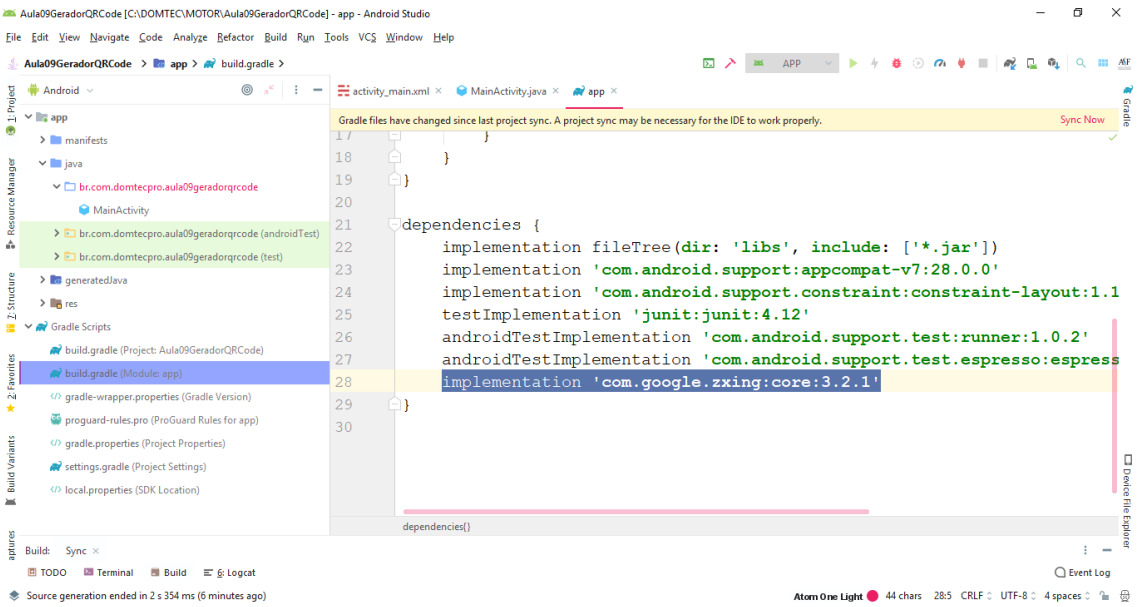
b. Role o arquivo até chegar em dependencies:



Digite abaixo da linha 27:

```
implementation 'com.google.zxing:core:3.2.1'
```

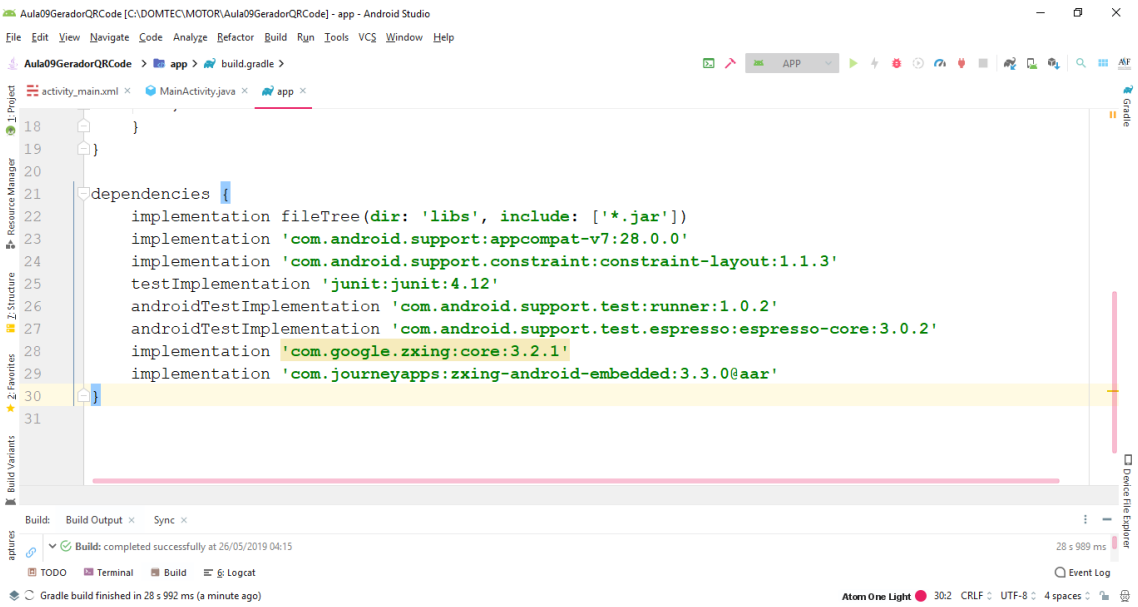
O código deve ficar como abaixo, clique em Sync Now, no canto direito superior:



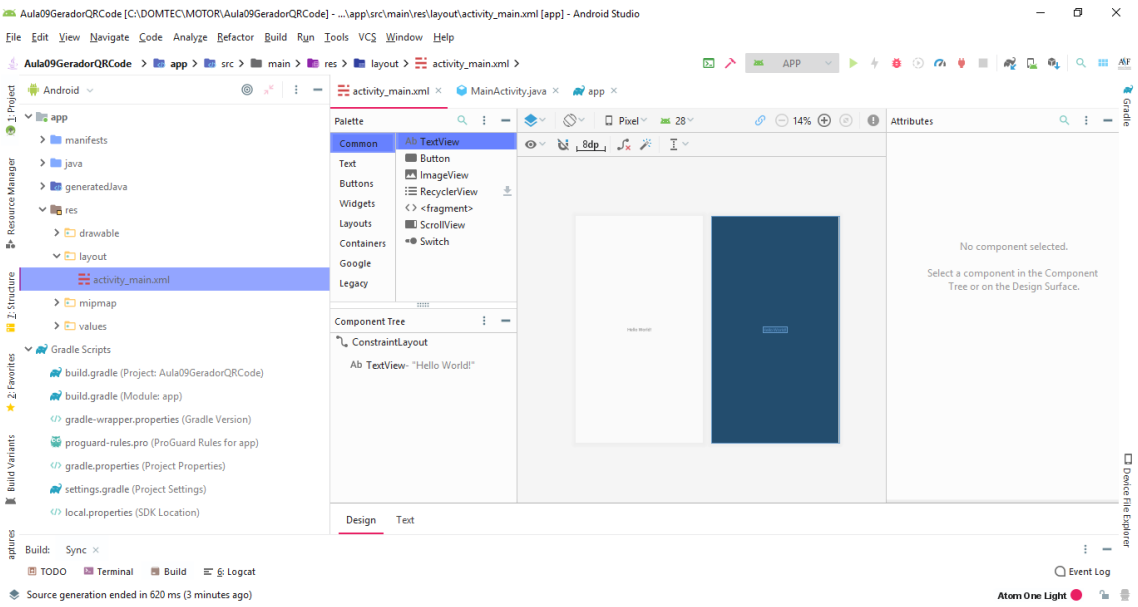
Aguarde a sincronização e adicione mais uma linha

```
implementation 'com.journeyapps:zxing-android-embedded:3.3.0@aar'
```

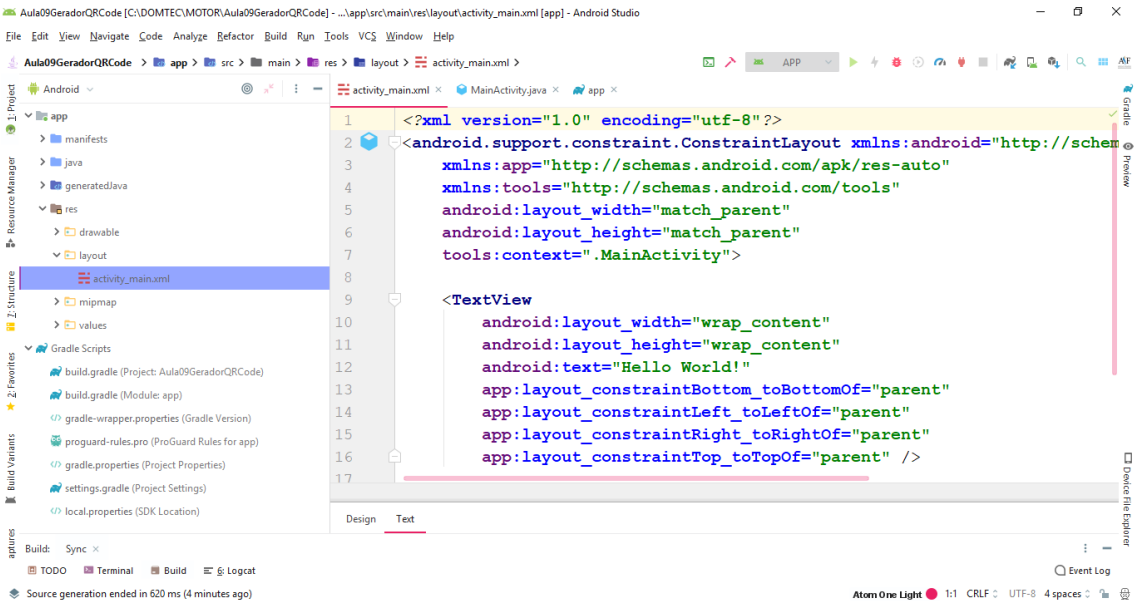
Novamente após adicionar a linha clique em Sync Now:



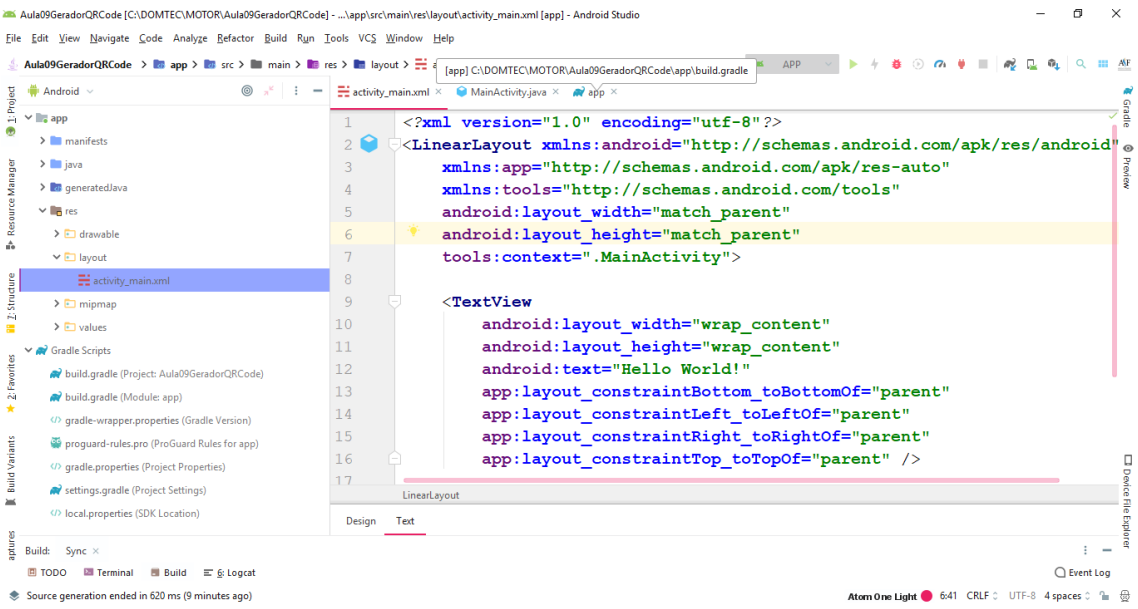
f. Abra a tela principal do projeto em Android > app > res > layout > activity_main.xml



a. Abra o código XML da janela activity_main.xml

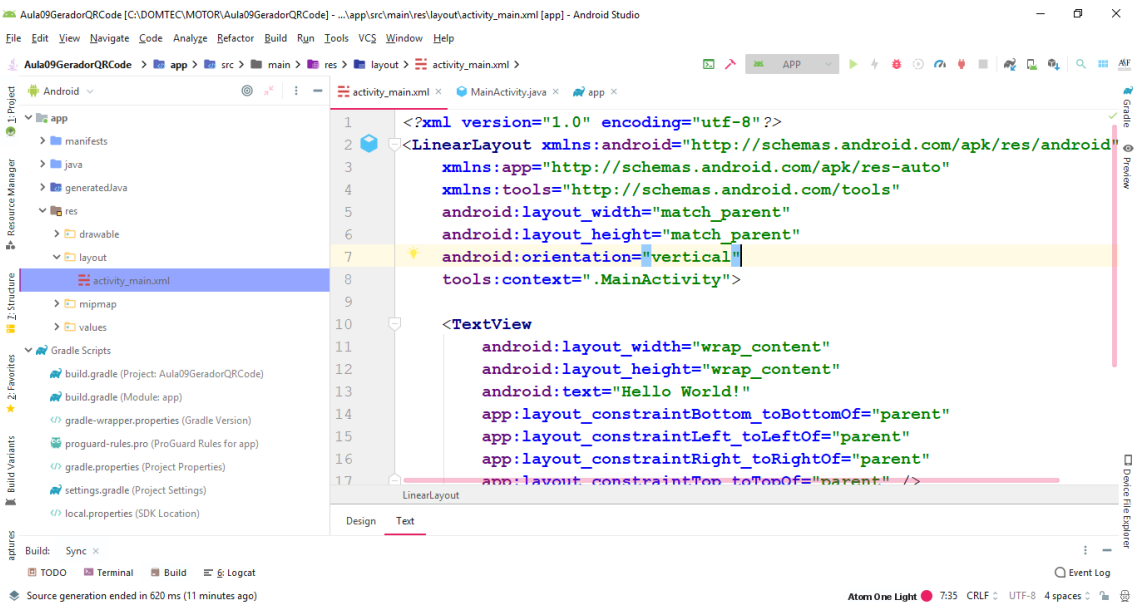


Na linha 2, troque o layout para LinearLayout:

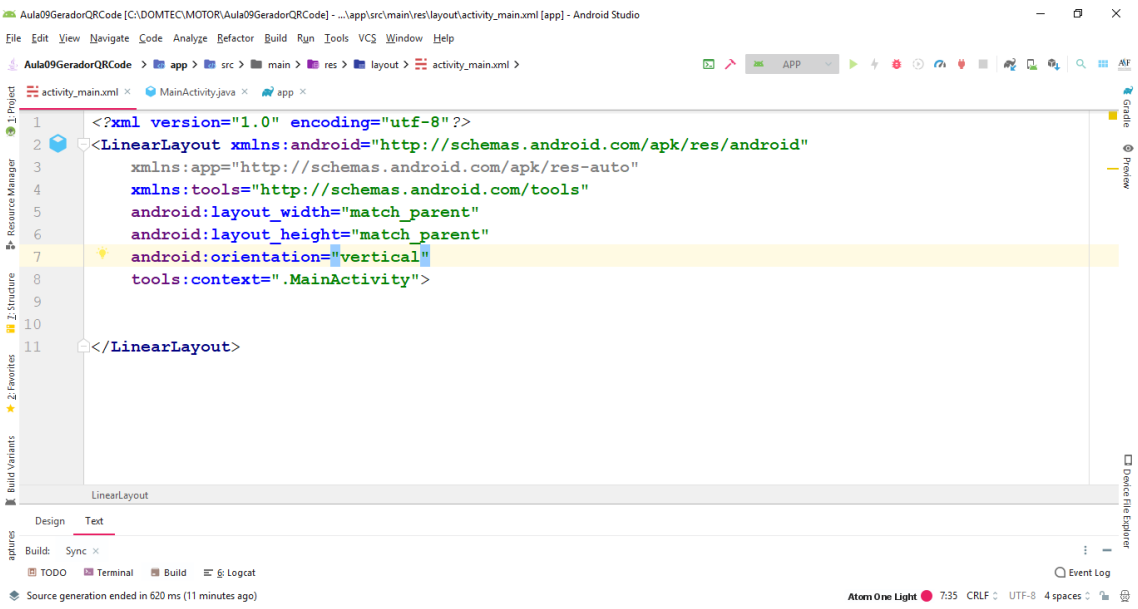


Clique no final da linha 6 e abra uma linha em branco na linha 7, para adicionar o código abaixo:

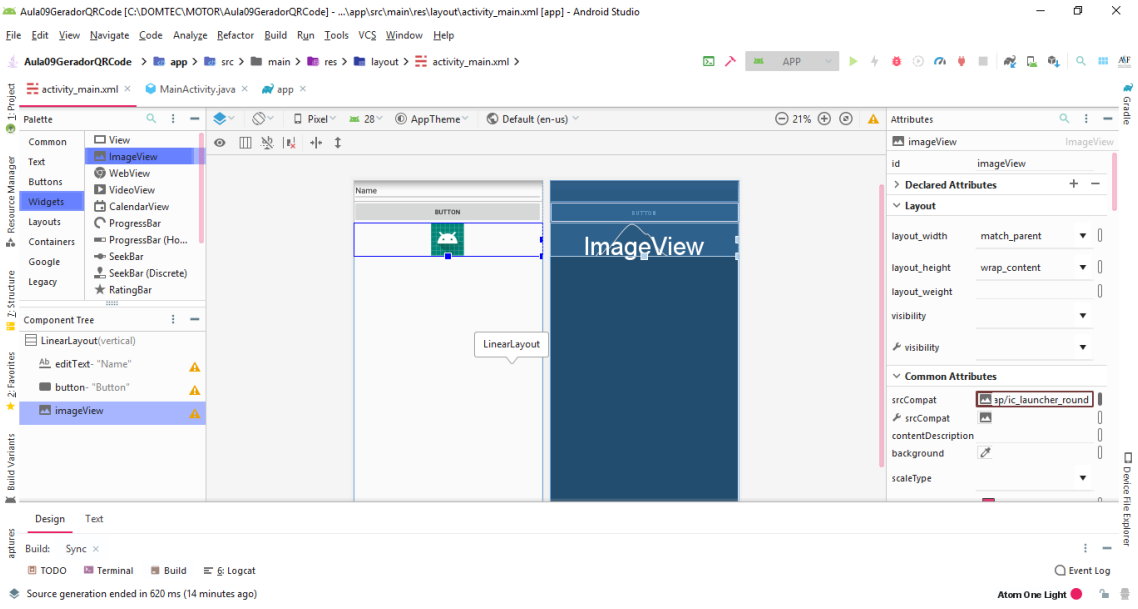
```
android:orientation="vertical"
```



Apague o TextView do “Hello World”



Vamos agora adicionar os elementos da tela para geração do QRCode:



Adicione com acima:

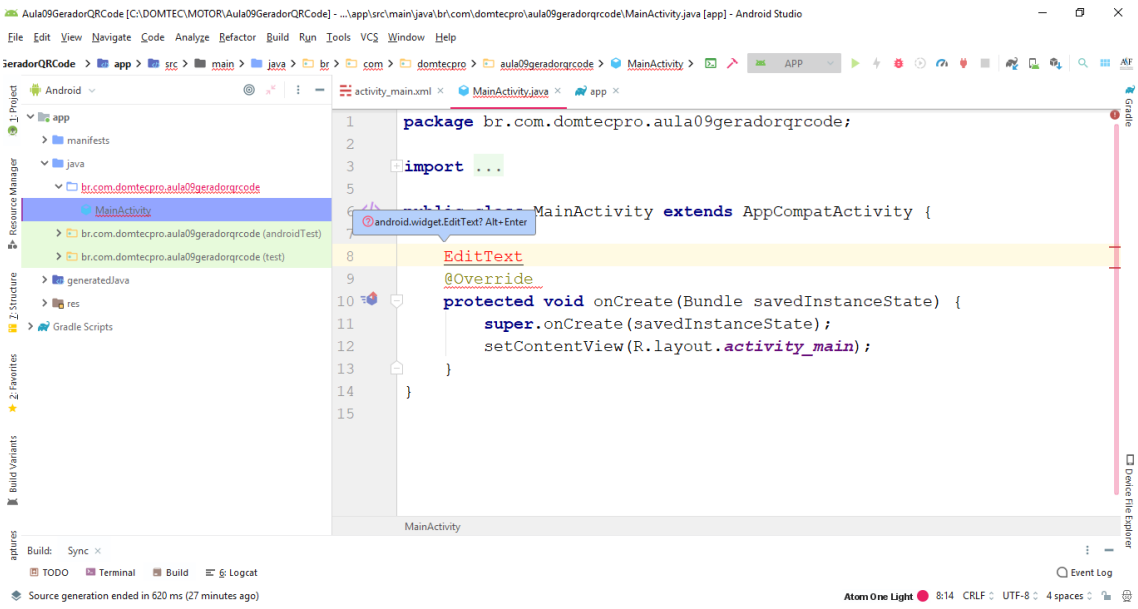
- PlainText
- Button
- ImageView

Voltando para o código faremos as alterações de cada objeto:

```
<EditText
    android:id="@+id/edtTexto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textSize="30dp"
    android:inputType="textPersonName"
    android:hint="Informe a palavra para codificação" />
<Button
    android:id="@+id/btnGerar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Gerar QR Code" />


<ImageView
    android:id="@+id/ivQRCode"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
```

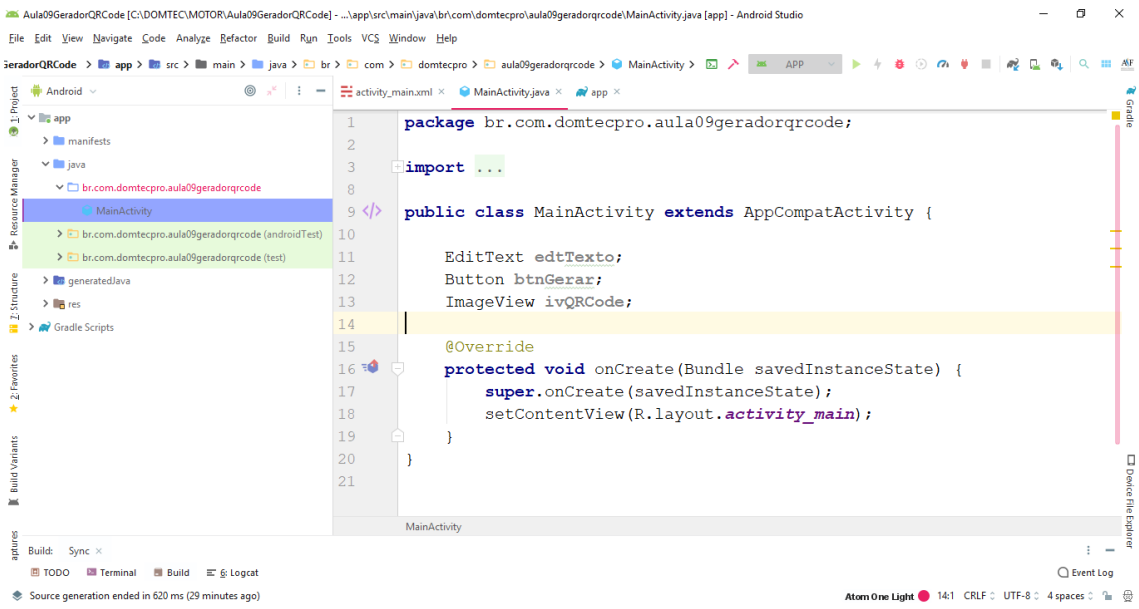
Agora vamos abrir o código Java em Android > app > java > br.com.itb.aula09geradorqrcode > MainActivity.java

Na linha 7 dê um ENTER e inicie a declaração dos objetos:

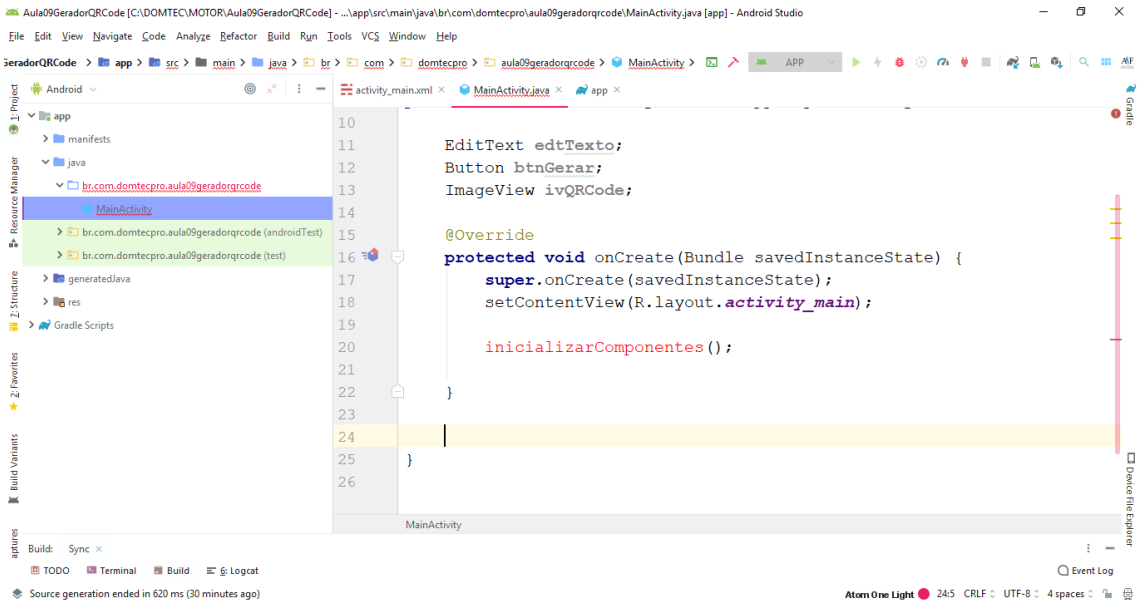


Obs.: Quando a classe ficar vermelha, solte o mouse e tecle ALT+ENTER para importar a classe EditText, por exemplo, como mostrado acima.

```
EditText edtTexto;
Button btnGerar;
ImageView ivQRCode;
```



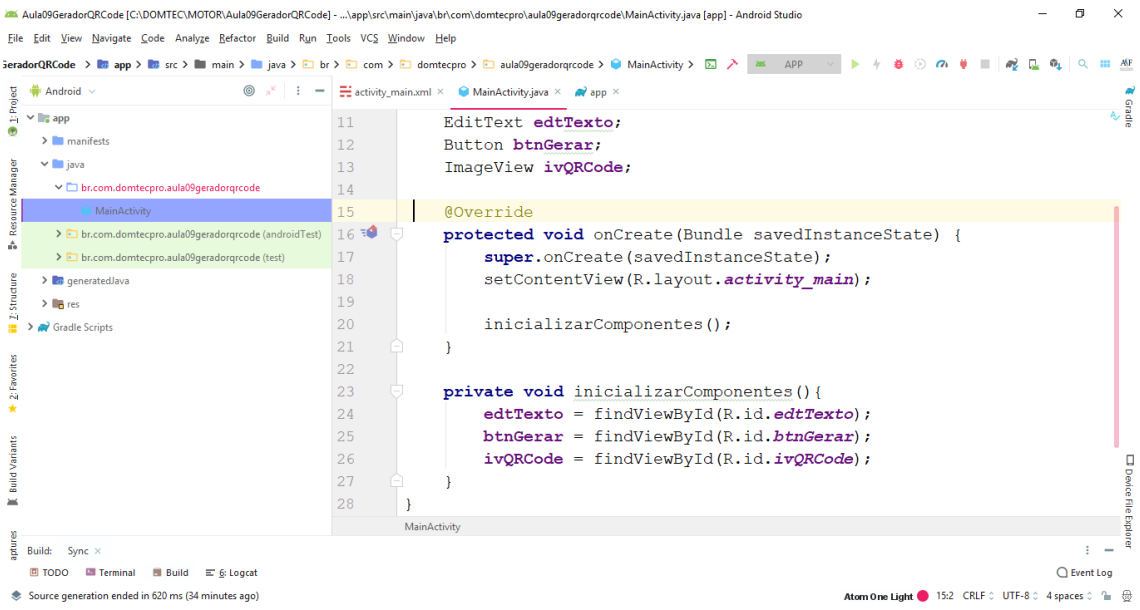
Dentro do método onCreate faça a chamada de um método e inicialização que iremos criar agora:




```
inicializarComponentes();
```

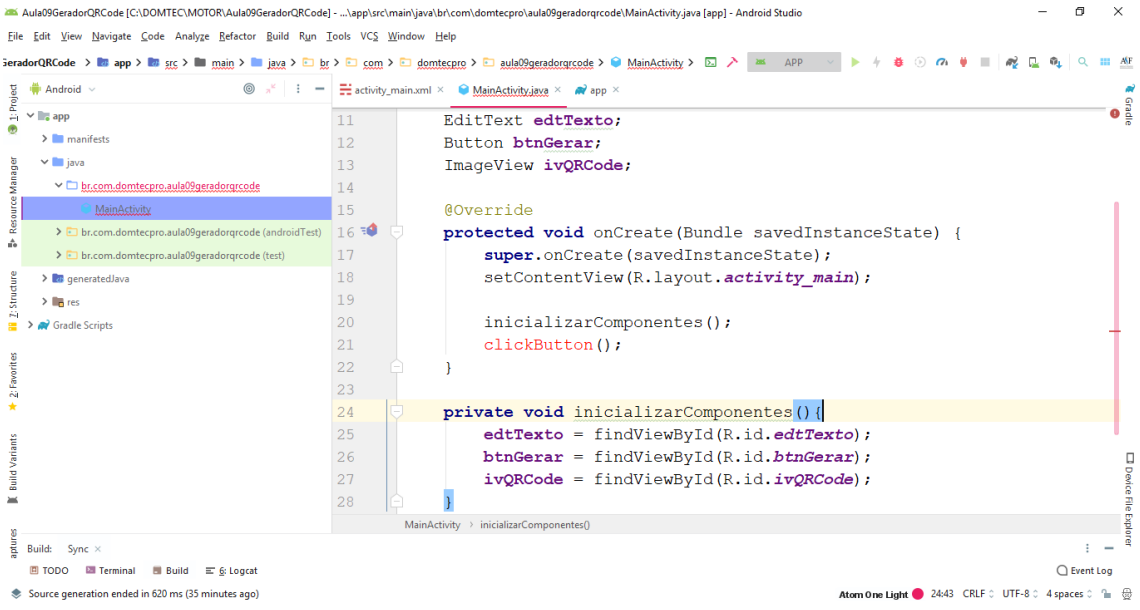
Fora do método onCreate, abaixo, crie o método que esta sendo chamado:

```
private void inicializarComponentes(){
    edtTexto = findViewById(R.id.edtTexto);
    btnGerar = findViewById(R.id.btnGerar);
    ivQRCode = findViewById(R.id.ivQRCode);
}
```



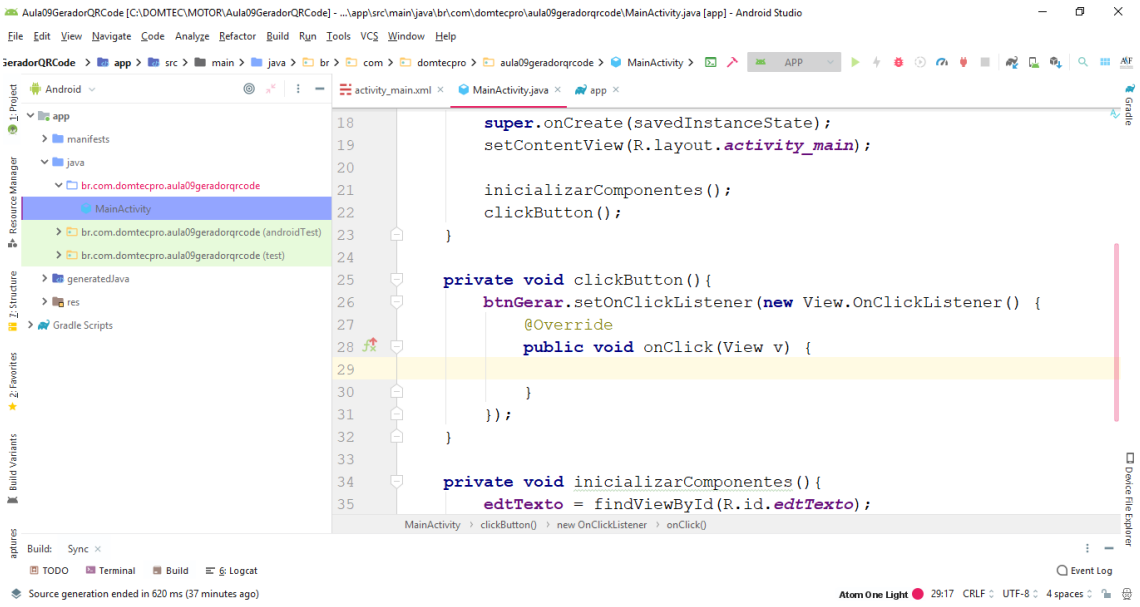
Abaixo da chamada do método inicializarComponentes(), faremos outra chamada:

```
clickButton();
```



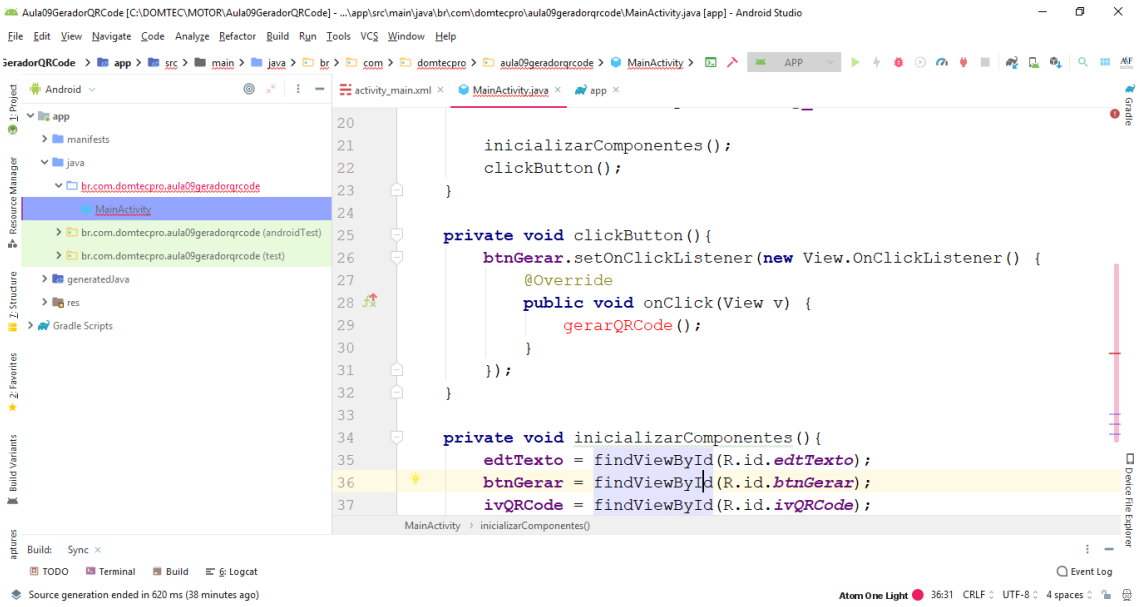
E vamos criar o método em questão, para concentrar todos os cliques possíveis em um só método:

No caso aqui, vamos programar o clique do botão:



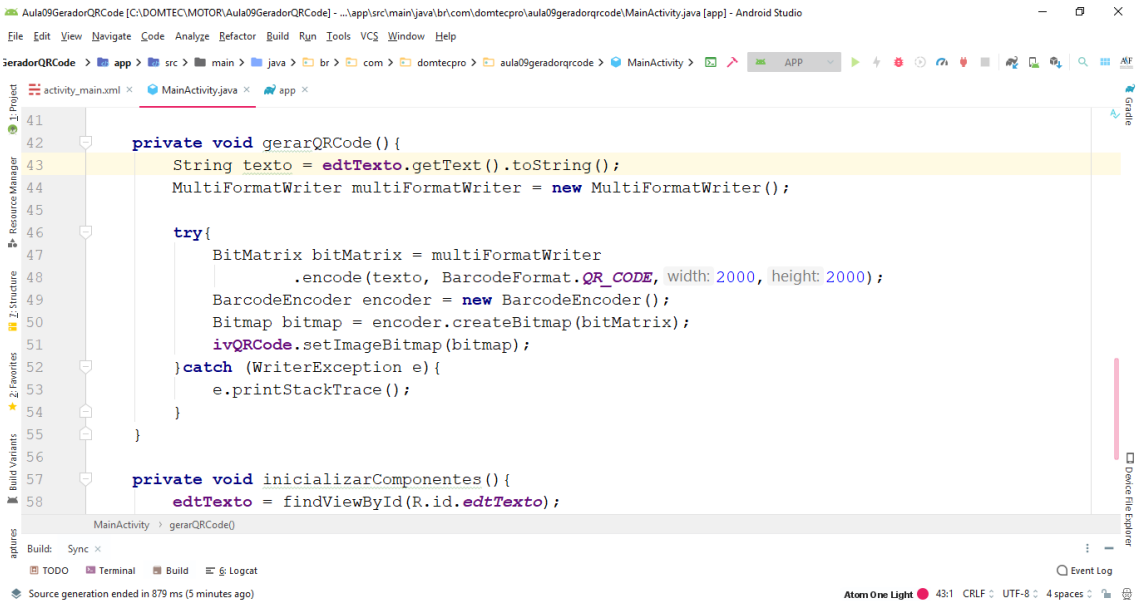
Novamente vamos criar uma nova chamada, de um novo método que será criado, dentro do método onClick():

```
gerarQRCode();
```



Digite o código abaixo:

```
private void gerarQRCode(){
    String texto = edtTexto.getText().toString();
    MultiFormatWriter multiFormatWriter = new MultiFormatWriter();

    try{
        BitMatrix bitMatrix = multiFormatWriter
                .encode(texto, BarcodeFormat.QR_CODE,2000,2000);
        BarcodeEncoder encoder = new BarcodeEncoder();
        Bitmap bitmap = encoder.createBitmap(bitMatrix);
        ivQRCode.setImageBitmap(bitmap);
    }catch (WriterException e){
        e.printStackTrace();
    }
}
```

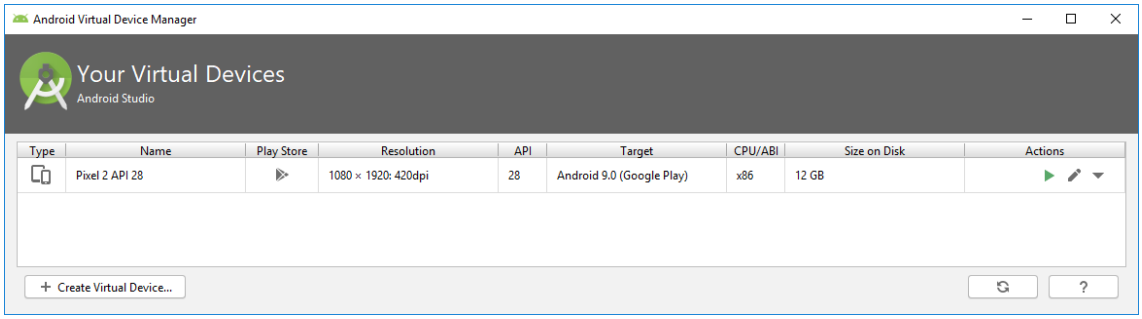


O código completo do MainActivity.java ficará como abaixo:

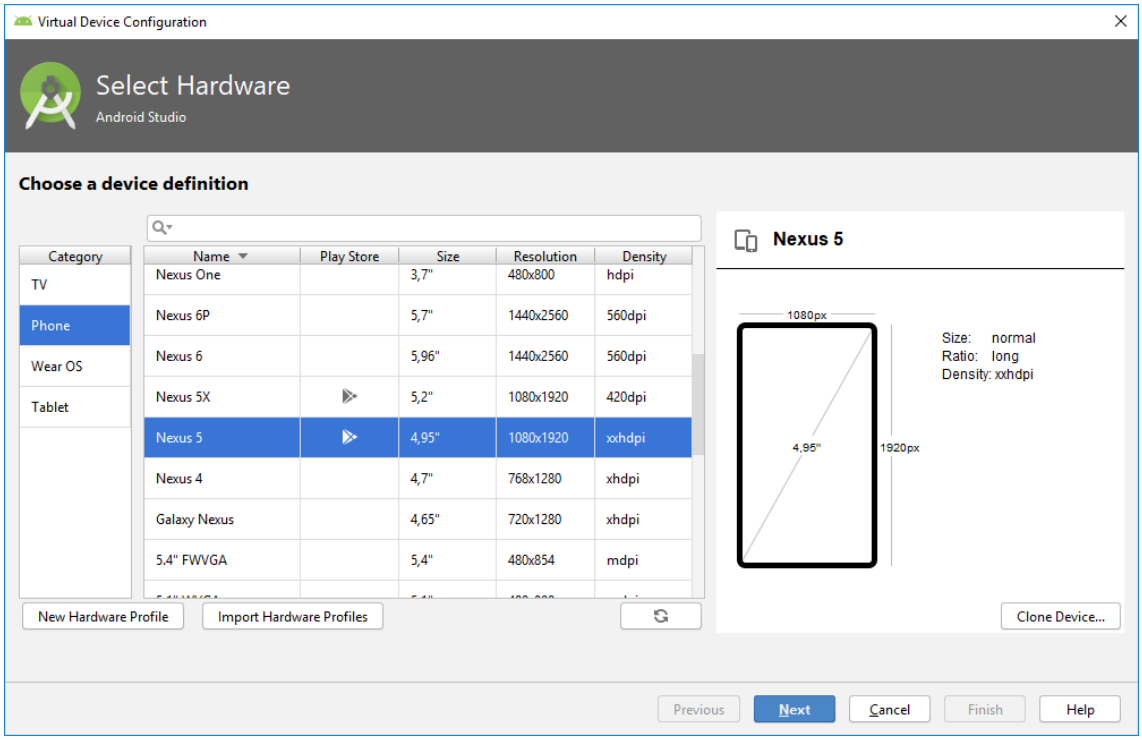
```
package br.com.domtecpro.aula09geradorqrcode;

import android.graphics.Bitmap;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;

import com.google.zxing.BarcodeFormat;
import com.google.zxing.MultiFormatWriter;
import com.google.zxing.WriterException;
import com.google.zxing.common.BitMatrix;
import com.journeyapps.barcodescanner.BarcodeEncoder;


public class MainActivity extends AppCompatActivity {
```

```
    EditText edtTexto;
    Button btnGerar;
    ImageView ivQRCode;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        inicializarComponentes();
        clickButton();
    }

    private void clickButton(){
        btnGerar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                gerarQRCode();
            }
        });
    }

    private void gerarQRCode(){
        String texto = edtTexto.getText().toString();
        MultiFormatWriter multiFormatWriter = new MultiFormatWriter();

        try{
            BitMatrix bitMatrix = multiFormatWriter
                    .encode(texto, BarcodeFormat.QR_CODE,2000,2000);
            BarcodeEncoder encoder = new BarcodeEncoder();
            Bitmap bitmap = encoder.createBitmap(bitMatrix);
            ivQRCode.setImageBitmap(bitmap);
        }catch (WriterException e){
            e.printStackTrace();
        }
    }

    private void inicializarComponentes(){
        edtTexto = findViewById(R.id.edtTexto);
        btnGerar = findViewById(R.id.btnGerar);
        ivQRCode = findViewById(R.id.ivQRCode);
    }
}
```

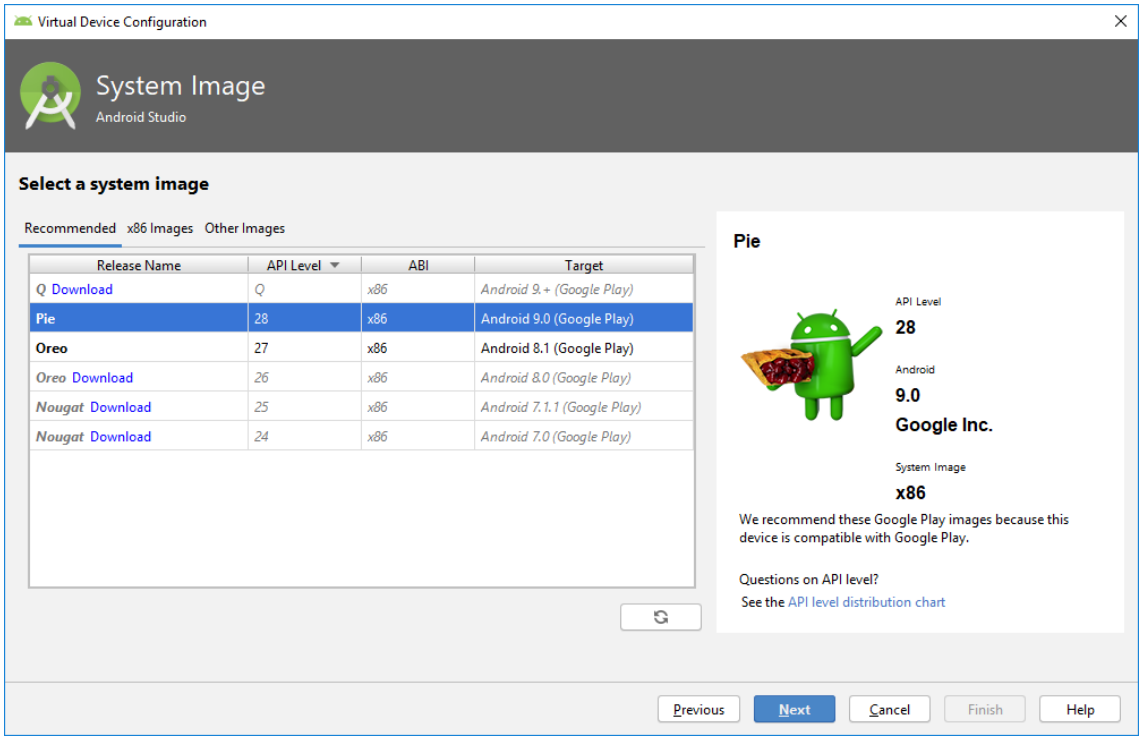
Clique em Tools > AVD Manager e crie sua máquina virtual clicando em Create Virtual Device:



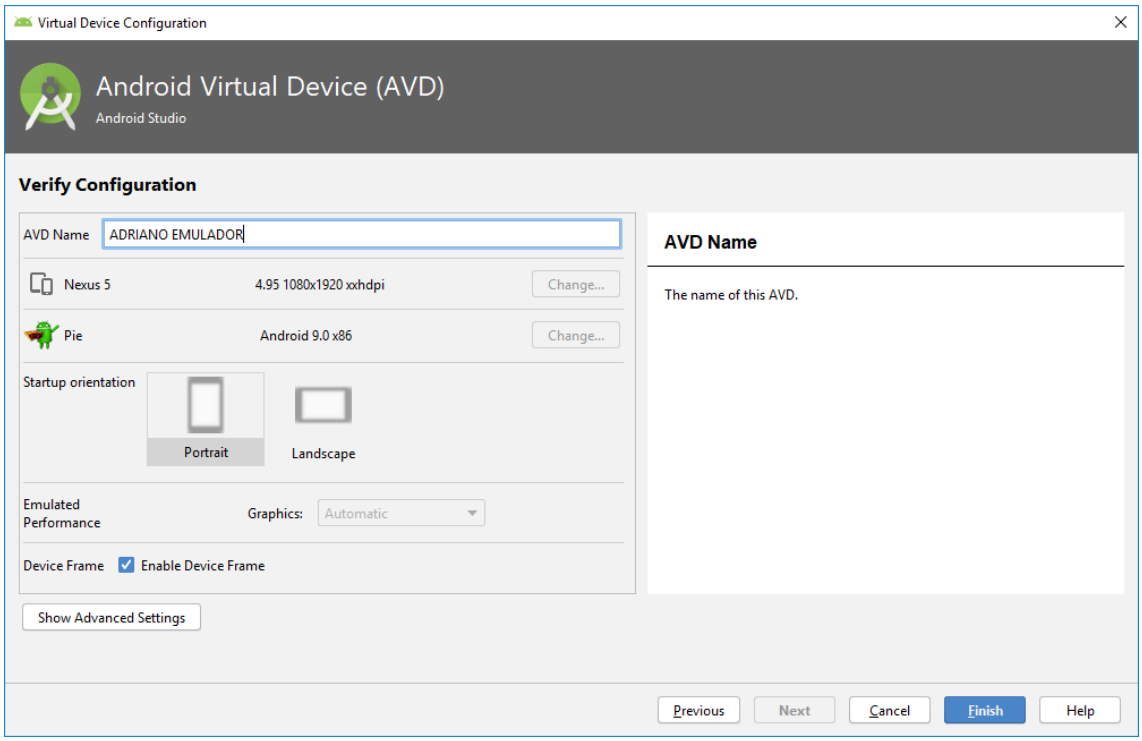
Selecione o aparelho, isto é a dimensão da tela desejada e clique em Next:



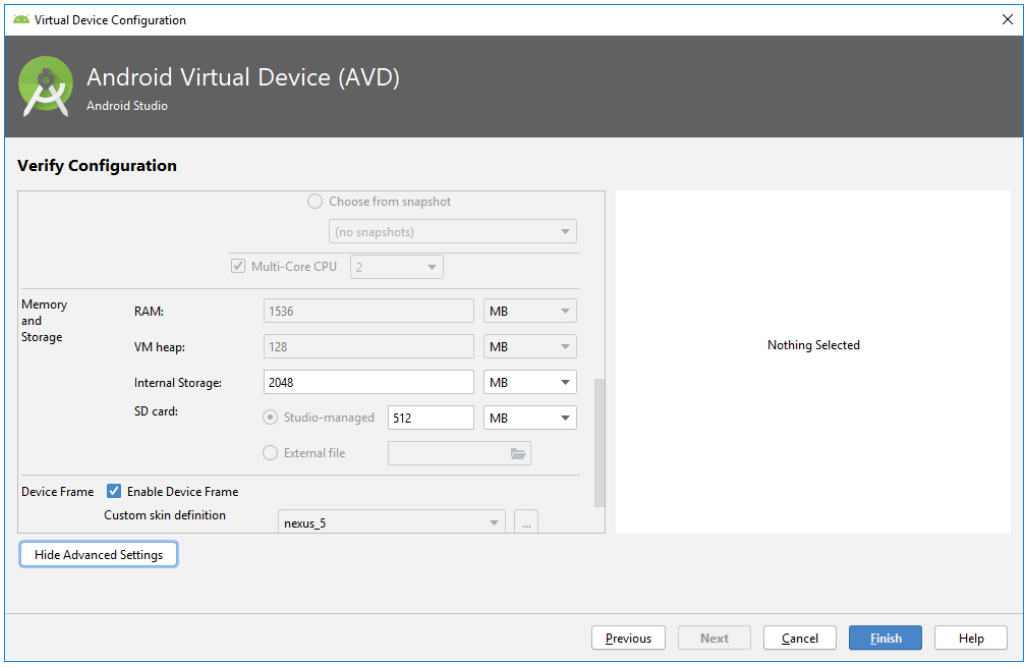
Selecione a imagem de sistema operacional da máquina virtual e clique em Next:



Altere o AVD Name para <SEU NOME> EMULADOR:
Se preferir remova o clique na opção Enable Device Frame se preferir obter apenas a emulação da tela do celular.



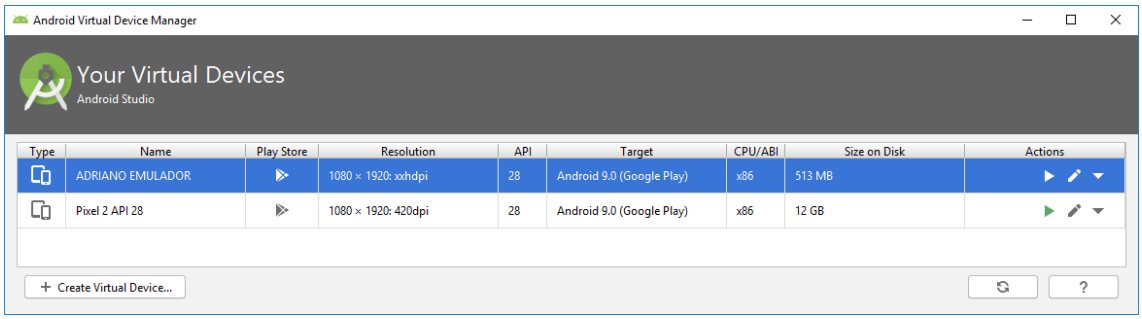
Clique no botão chamado Show Advanced Settings para ajustar a memória do emulador:



Altere a propriedade RAM para 200Mb e a propriedade VM heap para 32Mb.

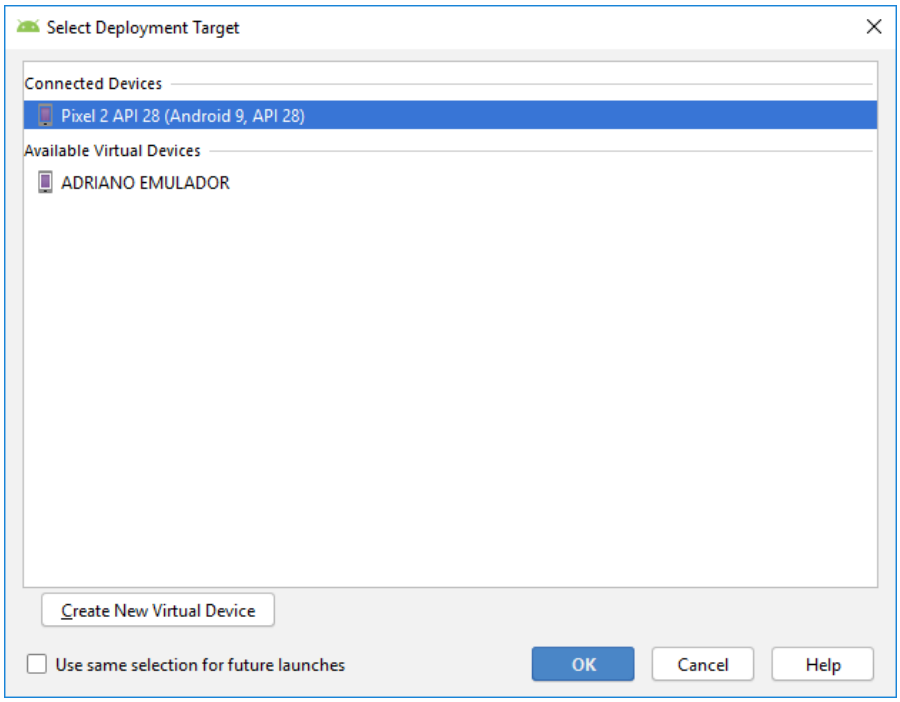
Clique em Finish ao terminar os ajustes de memória.

Execute o emulador criado, clicando no ícone de execução (play), para editar a máquina virtual basta clicar na caneta ao lado do play:


Android Virtual Device Manager
Your Virtual Devices
Android Studio
Type
Name
Play Store
Resolution
API
Target
CPU/ABI
Size on Disk
Actions
ADRIANO EMULADOR
1080 × 1920: xxhdpi
28
Android 9.0 (Google Play)
x86
513 MB
Pixel 2 API 28
1080 × 1920: 420dpi
28
Android 9.0 (Google Play)
x86
12 GB
Create Virtual Device...


19:01
Domingo, 28 de abr
Arquivos

Clique no menu Run > Run app… (Shift+F10)



Escolha o emulador e clique em OK (dê preferência ao aparelho ou emulador em execução em Connected Devices)

Referência Bibliográfica

[1] http://developer.android.com. Acessado em 28/04/2019.

[2] https://www.youtube.com/watch?v=KdQ9vKmvSEg. Acessado em 26/05/2019.

[3] https://mvnrepository.com/artifact/com.journeyapps/zxing-android-embedded/3.3.0. Acessado em 26/05/2019.

[4] https://stackoverflow.com/questions/18543668/integrate-zxing-in-android-studio. Acessado em 26/05/2019.