FUNDAÇÃO INSTITUTO DE EDUCAÇÃO DE BARUERI

Jardim Belval

Programação de Aplicativos 3

Tutorial Listas com RecyclerView

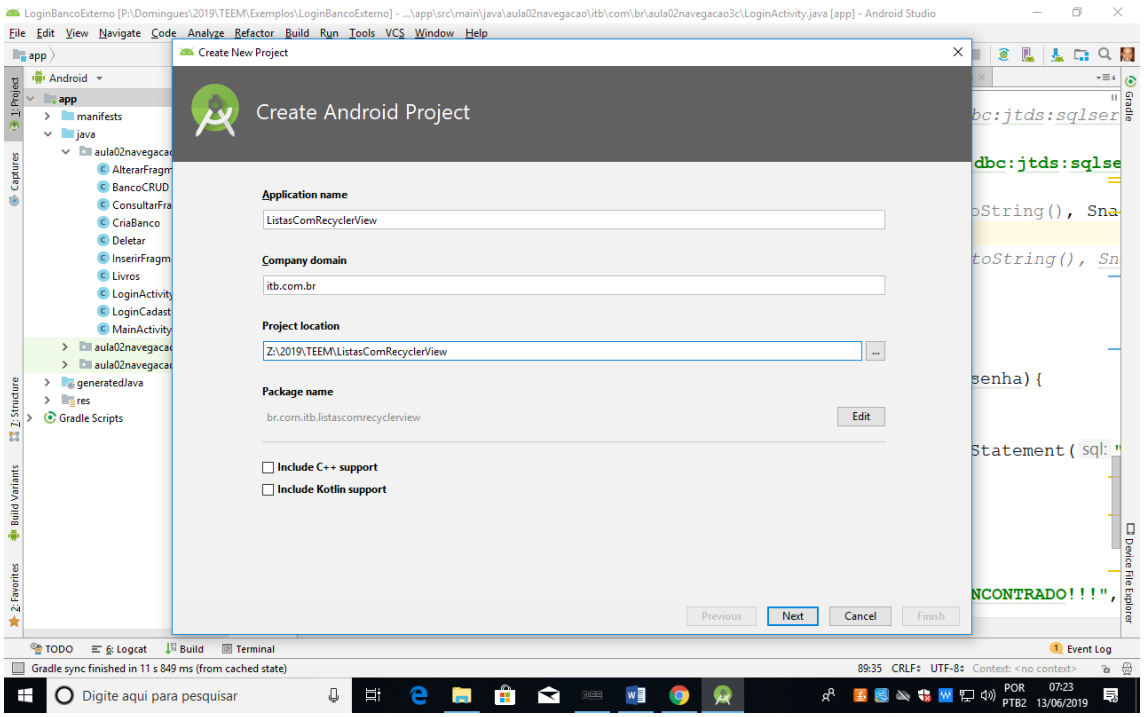
versão 3.0

Prof. Adriano Domingues

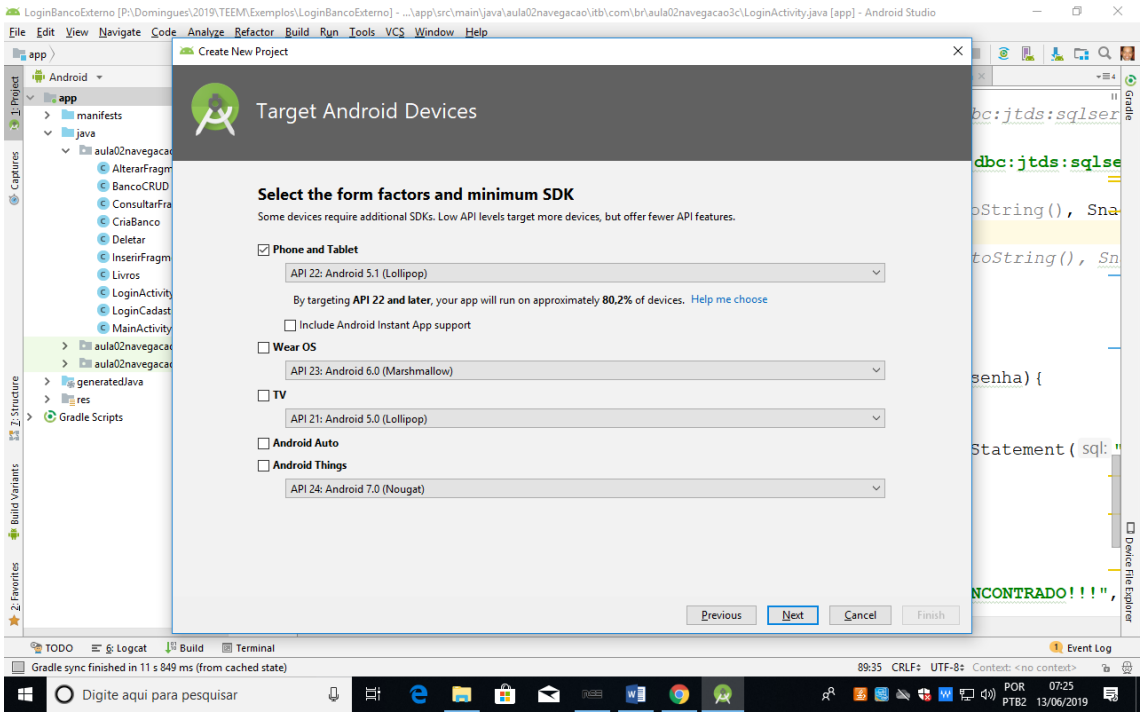
2020

AULA 12 – RecyclerView

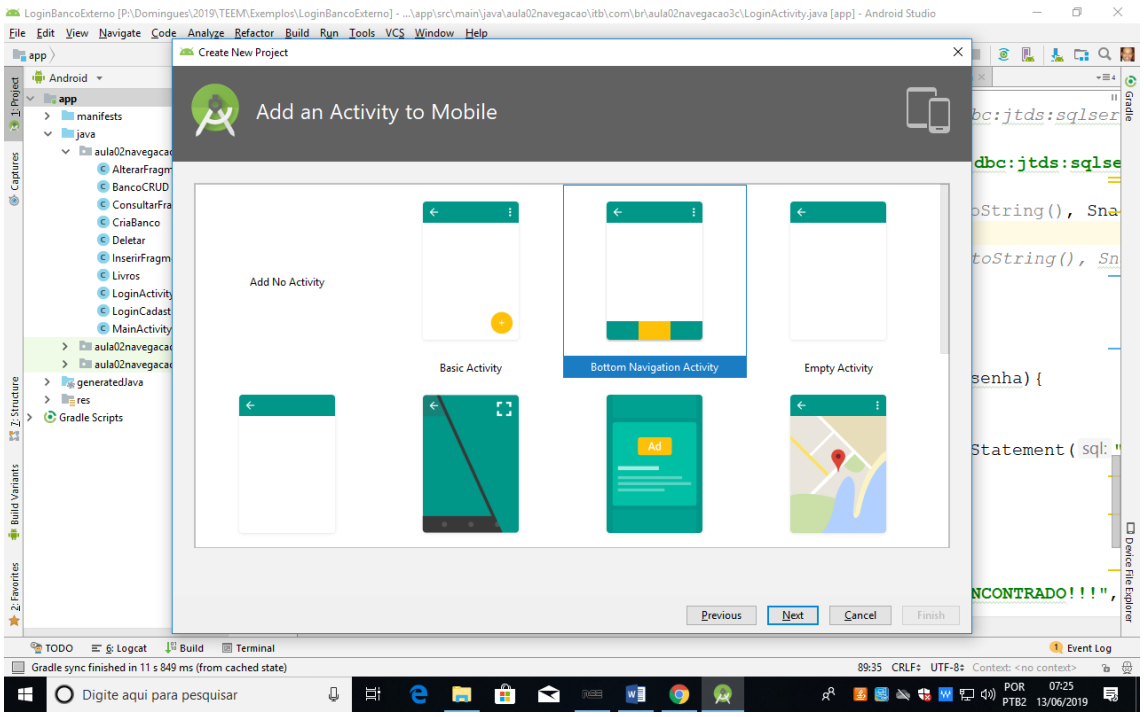
1. CRIAR PROJETO:
    a. FILE > NEW > NEW PROJECT...



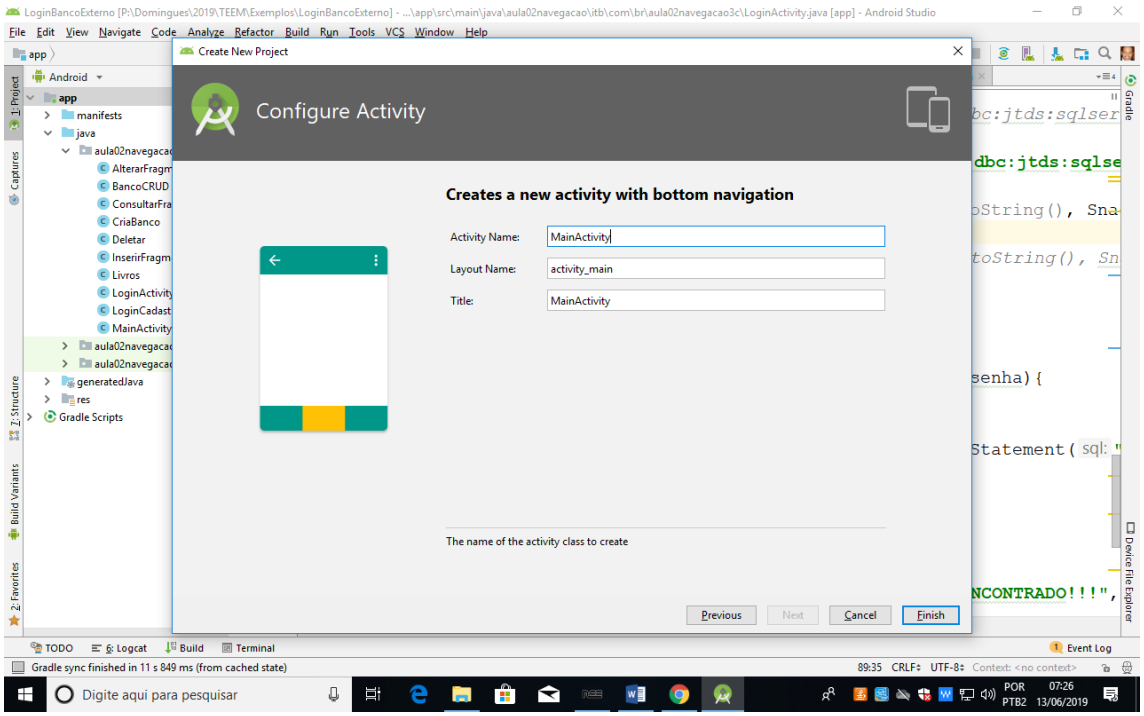
    a. Altere as opções da primeira janela e clique em Next:
        i. Aplication Name: Aula12ListasComRecyclerView
        ii. Company Domain: itb.com.br
        iii. Location: Z:\... (sua pasta na Z:)



    b. API 22

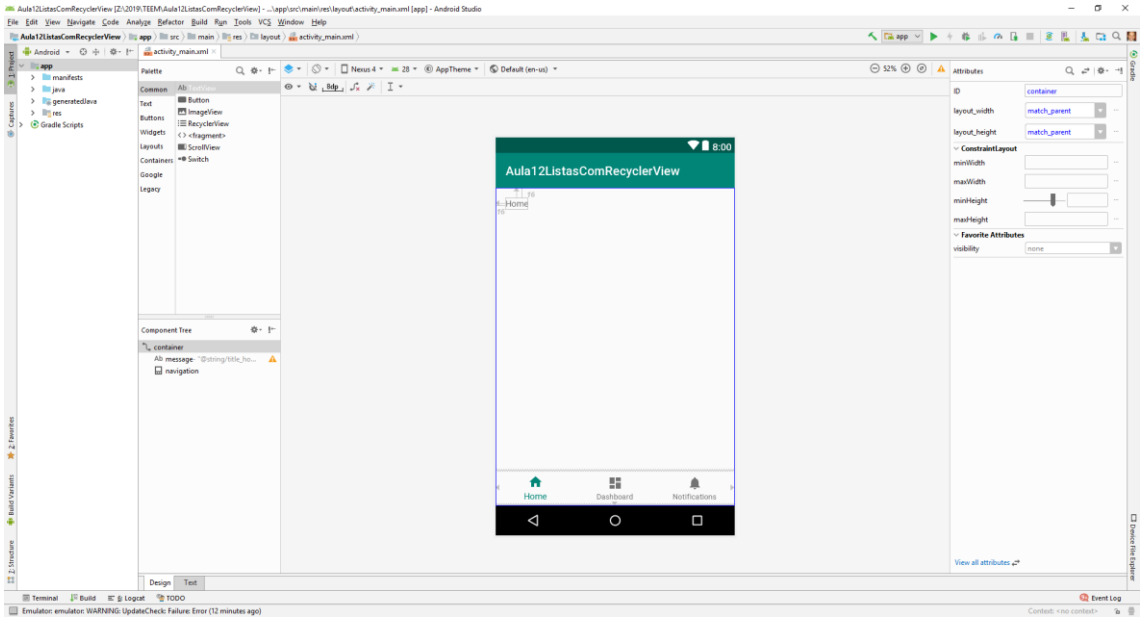


c. ESCOLHA A ATIVIDADE DE EXEMPLO: Bottom Navigation Activity



iv. Não altere nada na última janela e clique em Finish

d. Aguarde o carregamento do projeto:

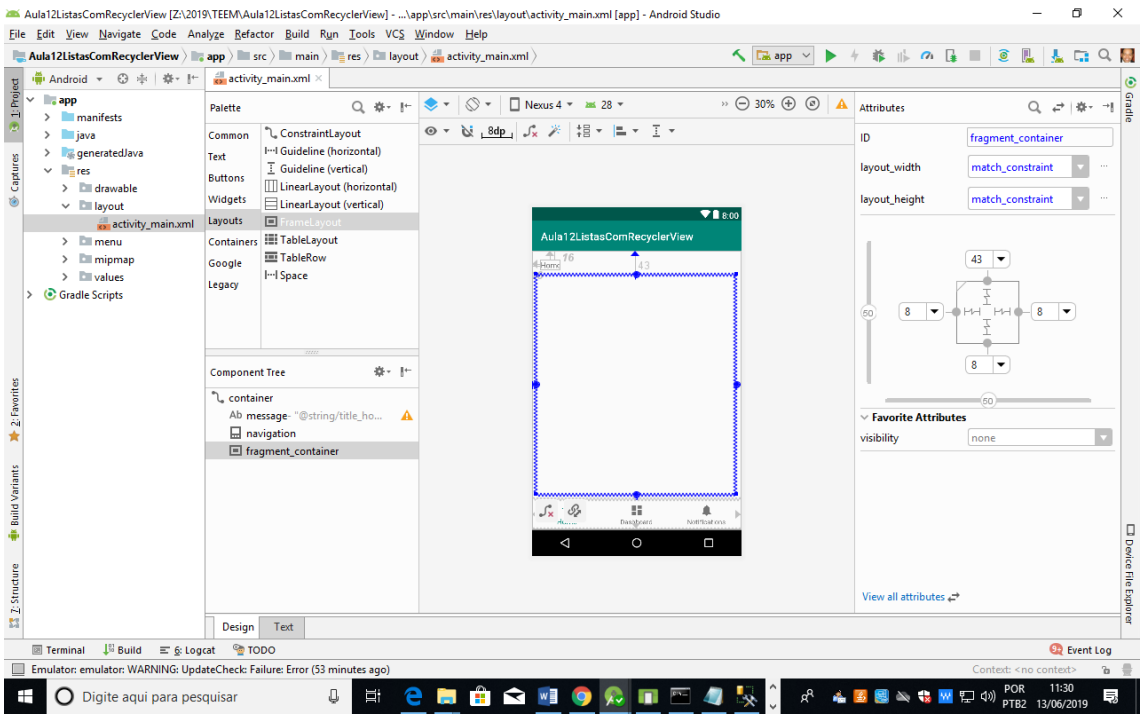


e. Vamos agora preparar o menu de navegação para carregarmos a lista de produtos que serão pesquisados no SQL Server.

a. Segue abaixo script do banco com a tabela produto:

```
use master
go
drop DATABASE TEEM_ANDROID
go
CREATE DATABASE TEEM_ANDROID
GO
USE TEEM_ANDROID
GO

CREATE TABLE produto(
codigo varchar(50) PRIMARY KEY,
descricao VARCHAR(100) NOT NULL unique,
qtde int not null,
valor_unit decimal(15,2) not null,
status int NOT NULL)   --0 INATIVO / 1 ATIVO
GO

INSERT INTO PRODUTO VALUES (1, 'Coca-cola', 10, 4.99, 1);
GO
INSERT INTO PRODUTO VALUES (2, 'Bolacha', 30, 2.99, 1);
GO
INSERT INTO PRODUTO VALUES (3, 'Chocolate', 100, 3.99, 1);
GO
INSERT INTO PRODUTO VALUES (4, 'Salgado', 20, 1.99, 1);
GO
INSERT INTO PRODUTO VALUES (5, 'Suco', 30, 4.99, 1);
GO

select * from produto;
GO
```

f. Insira uma FrameLayout na activity_main.xml, em Android > app > res > layout

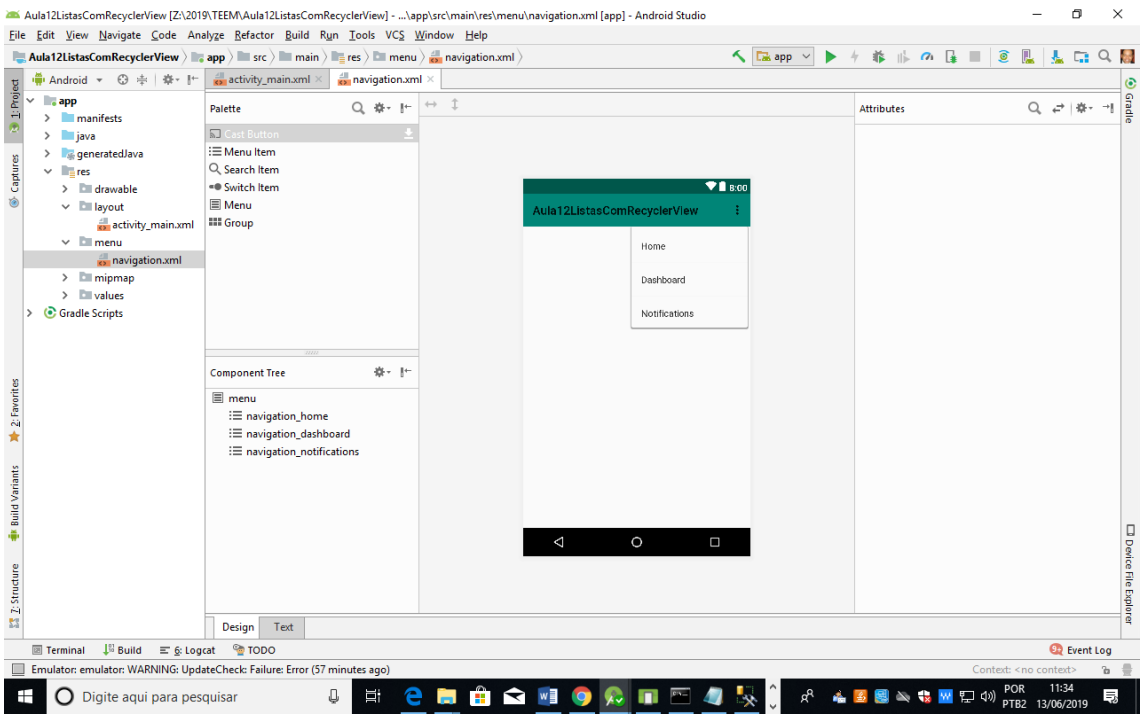


Atenção para os atributos que foram alterados:

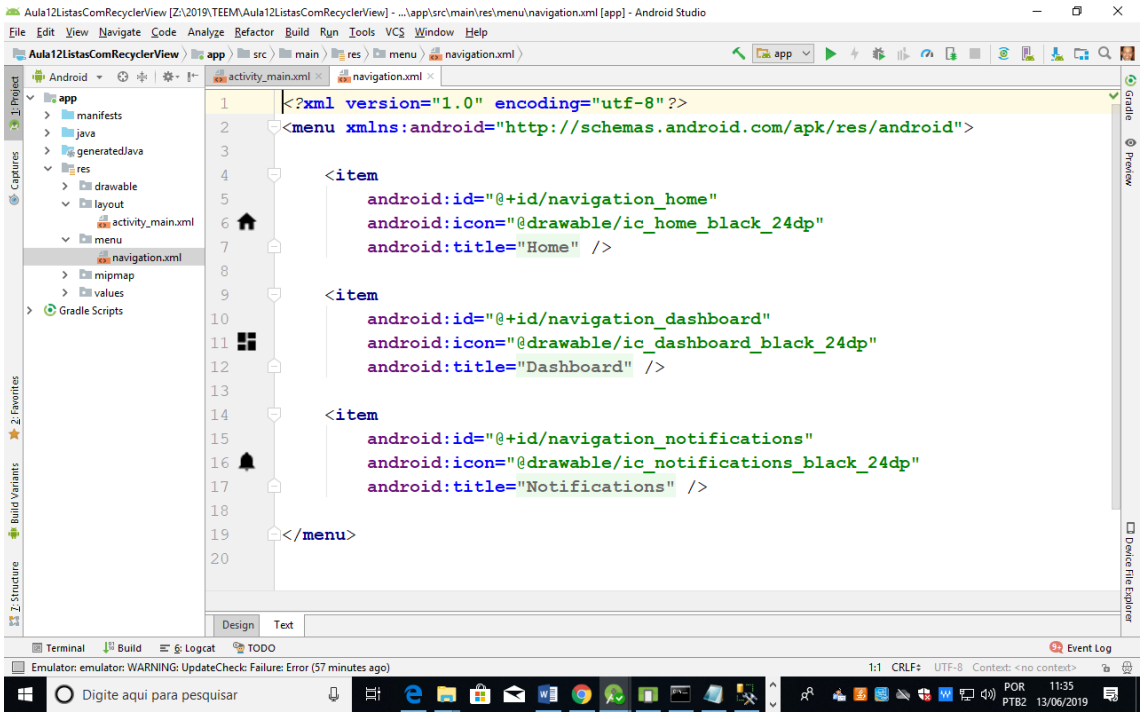
id: fragment_container
layout_width: match_constraint
layout_height: match_constraint

Vamos usar o elemento acima para trocar as janelas internas ao clicar nos itens do menu do rodapé.

Agora acesse o arquivo de definição do menu em: Android > app > res > menu > navigation.xml



Clique na guia abaixo chamada Text:

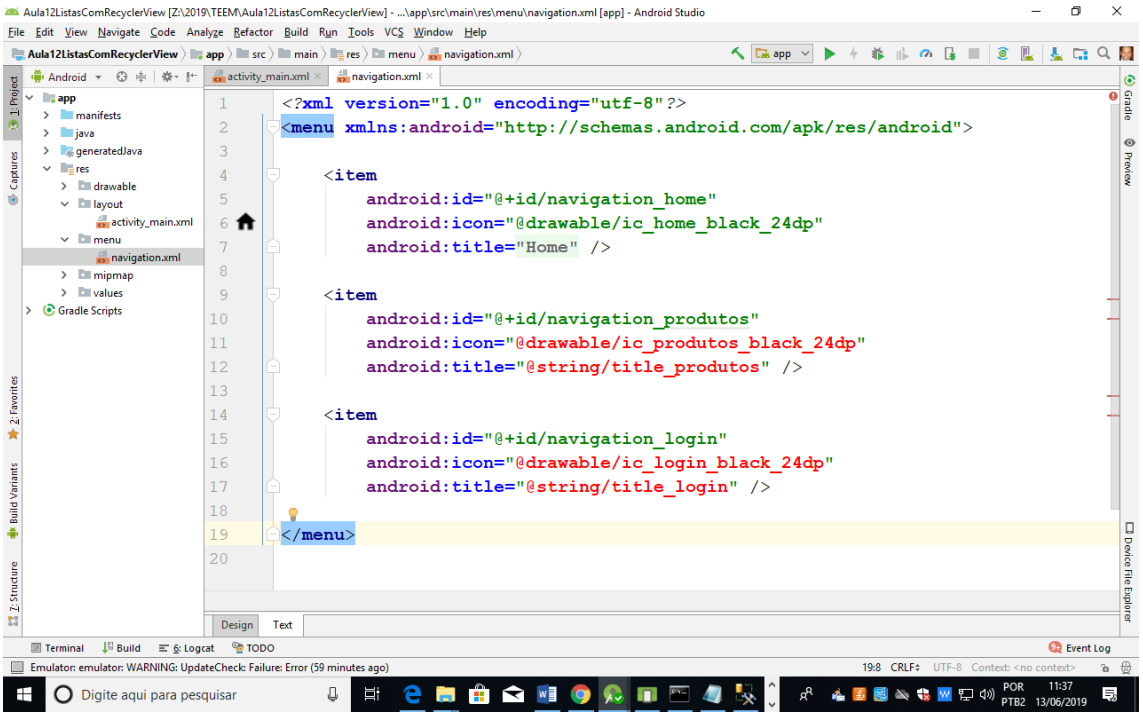


Altere com o código abaixo:

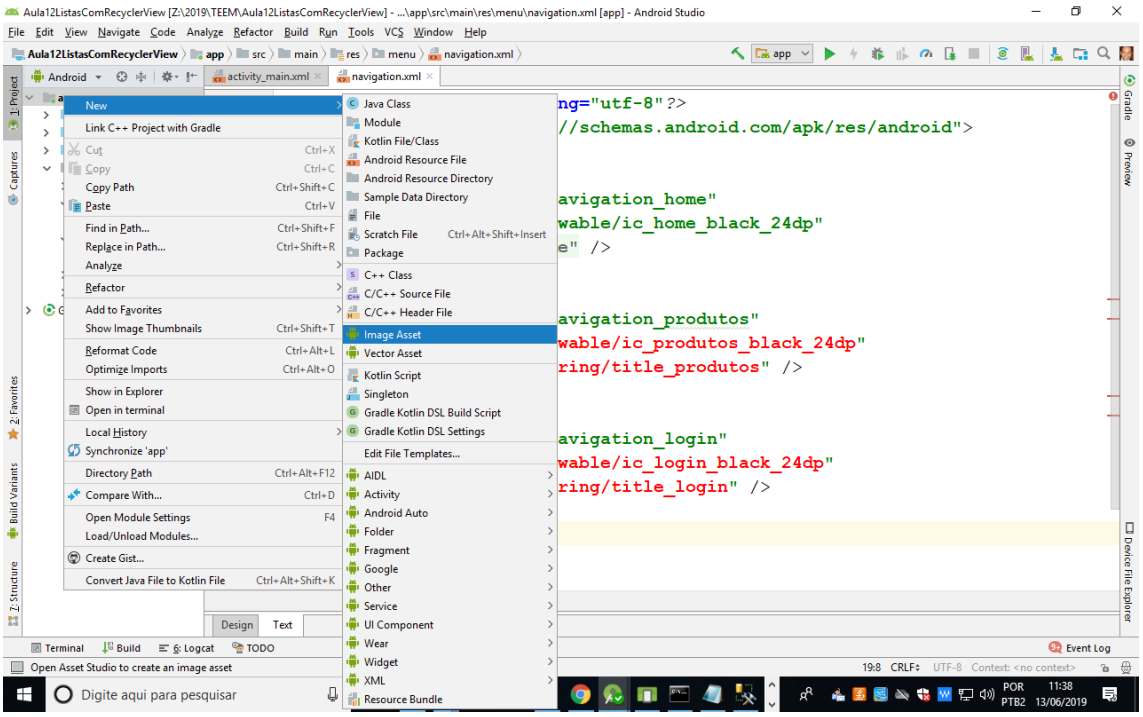
```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@+id/navigation_home"
        android:icon="@drawable/ic_home_black_24dp"
        android:title="@string/title_home" />
```

```
    <item
        android:id="@+id/navigation_produtos"
        android:icon="@drawable/ic_produtos_black_24dp"
        android:title="@string/title_produtos" />

    <item
        android:id="@+id/navigation_login"
        android:icon="@drawable/ic_login_black_24dp"
        android:title="@string/title_login" />

</menu>
```

Os ícones e títulos de produtos e login ficarão vermelhos, devemos criar estes recursos:



Clique com o botão direito em **app**: Em seguida no menu New > Image Asset



Escolha **Action Bar and Tab Icons** em *Icon Type*.

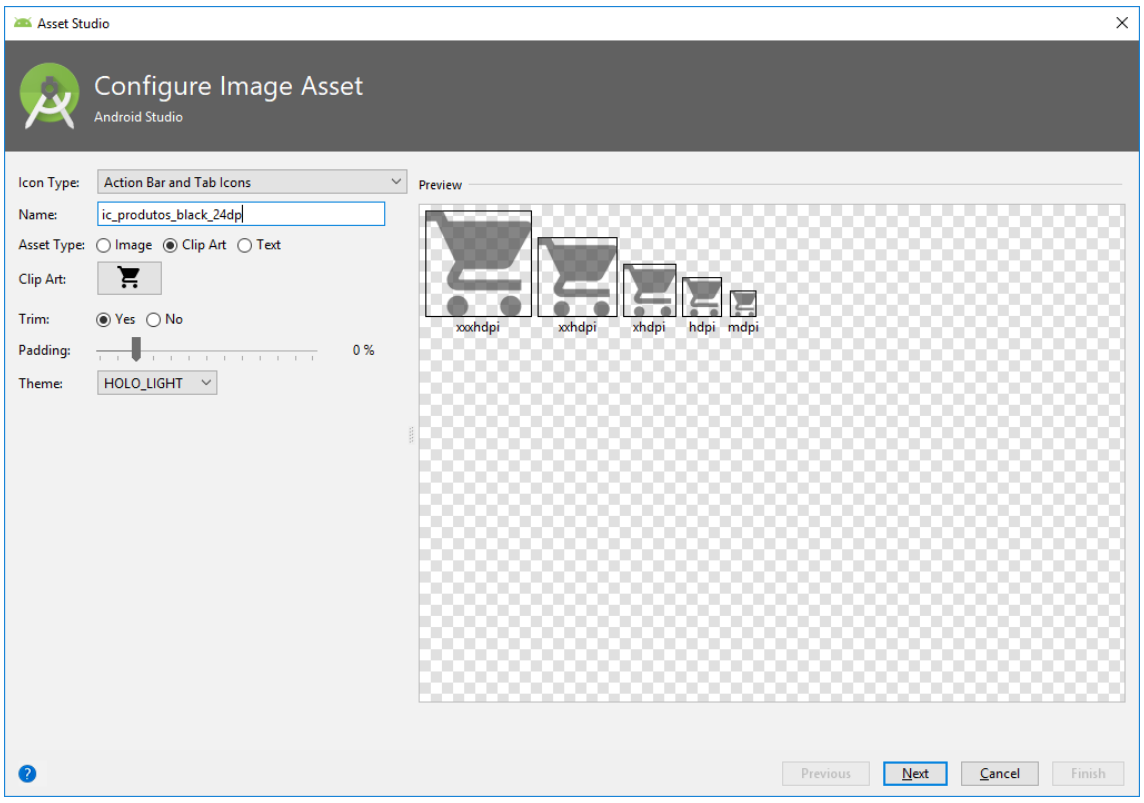
Altere e crie o ícone para o menu **produtos**:

Altere os atributos para:

Name: **ic_produtos_black_24dp**

Clip Art: shopping cart

Trim: Yes



Repita o processo para abrir o Image Asset e:

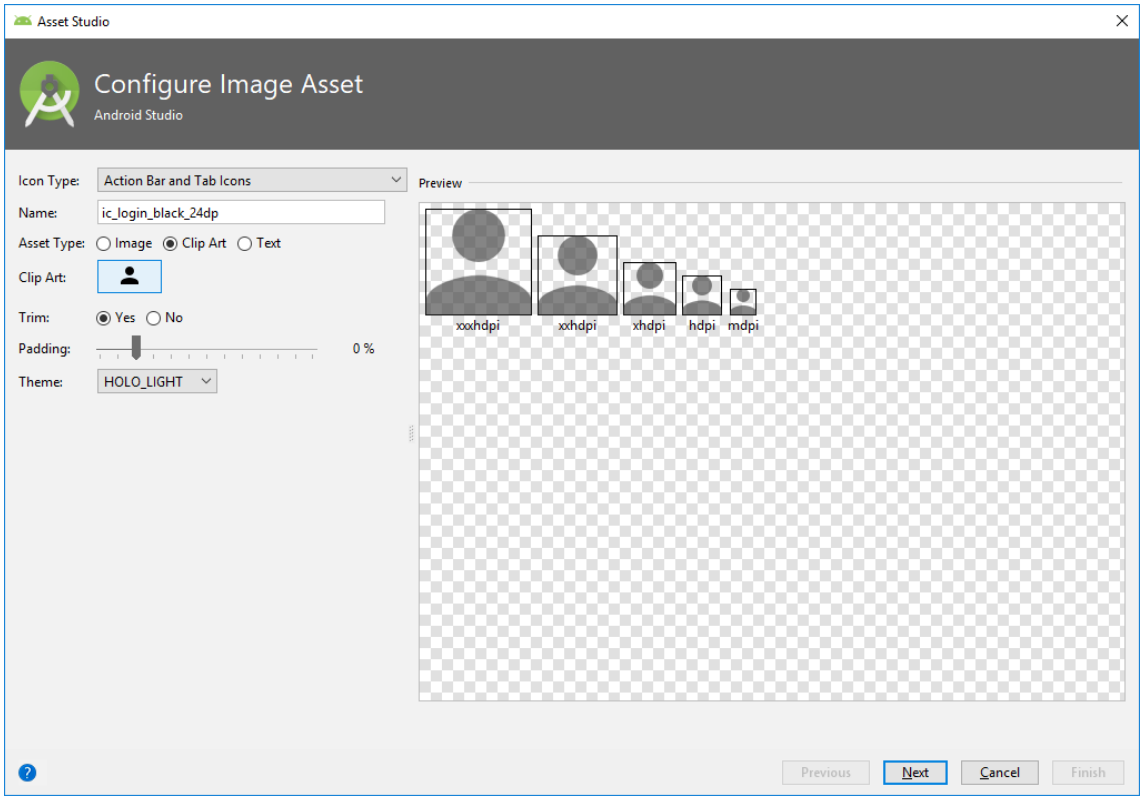
Altere e crie o ícone para o menu **login**:

Altere os atributos para:

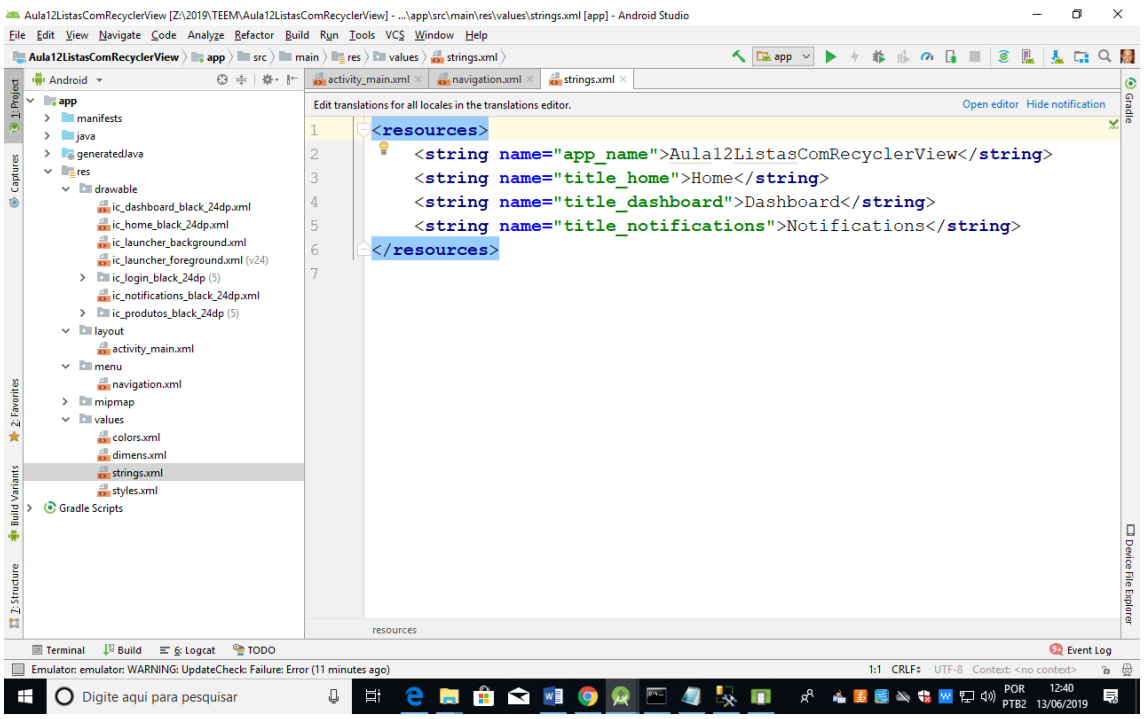
Name: **ic_login_black_24dp**

Clip Art: person

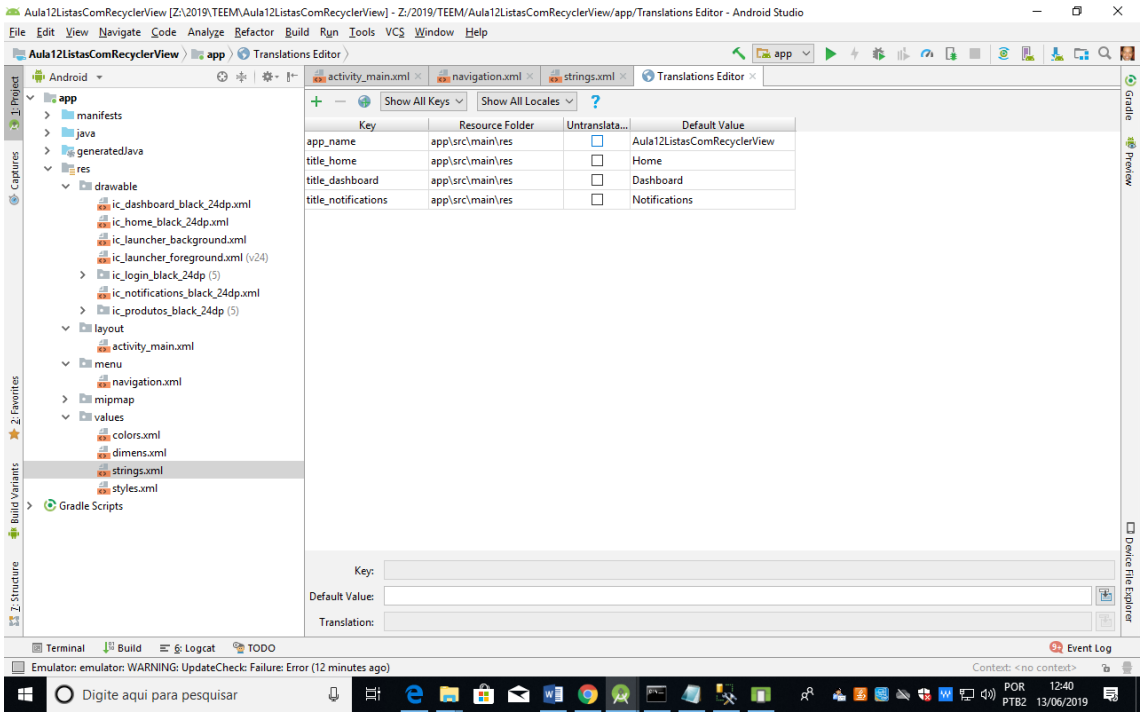
Trim: Yes



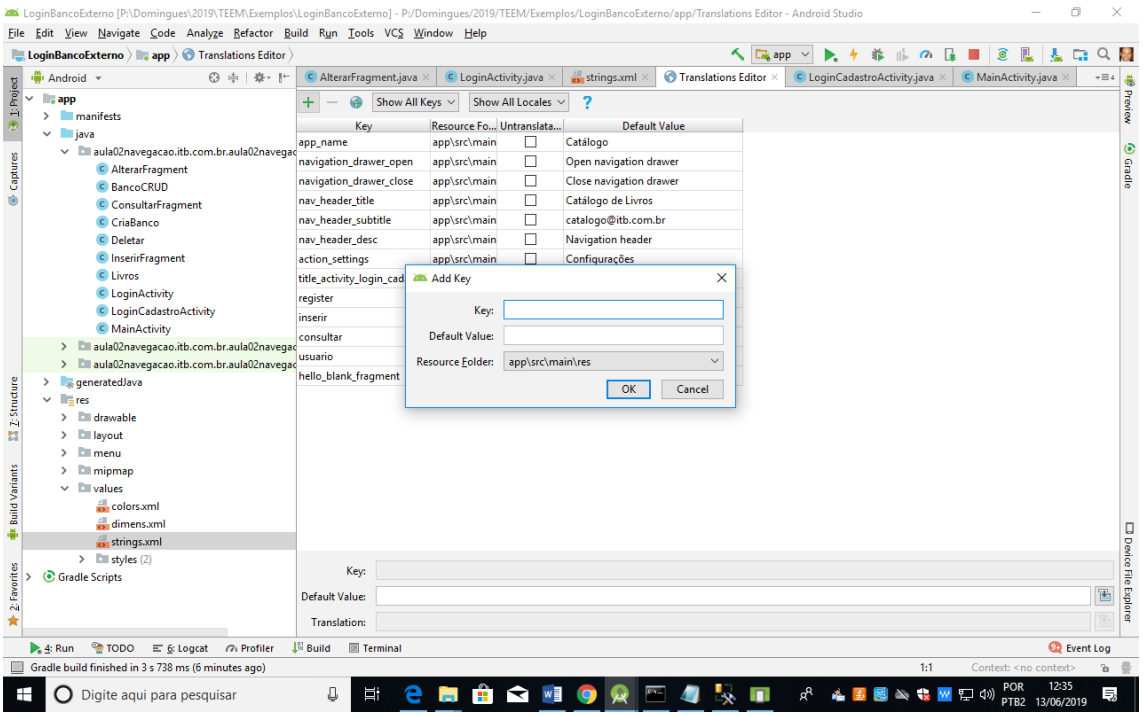
Para criar os títulos abra o arquivo string.xml:
Clique em Android > app > values > strings.xml

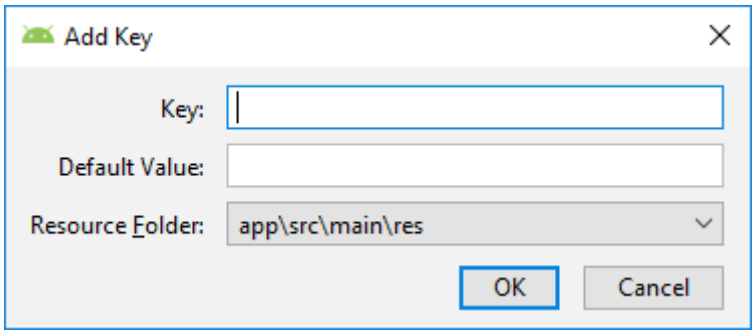


Clique em Open Editor, lado direito superior:



Clique no sinal de + verde para adicionar uma nova string:



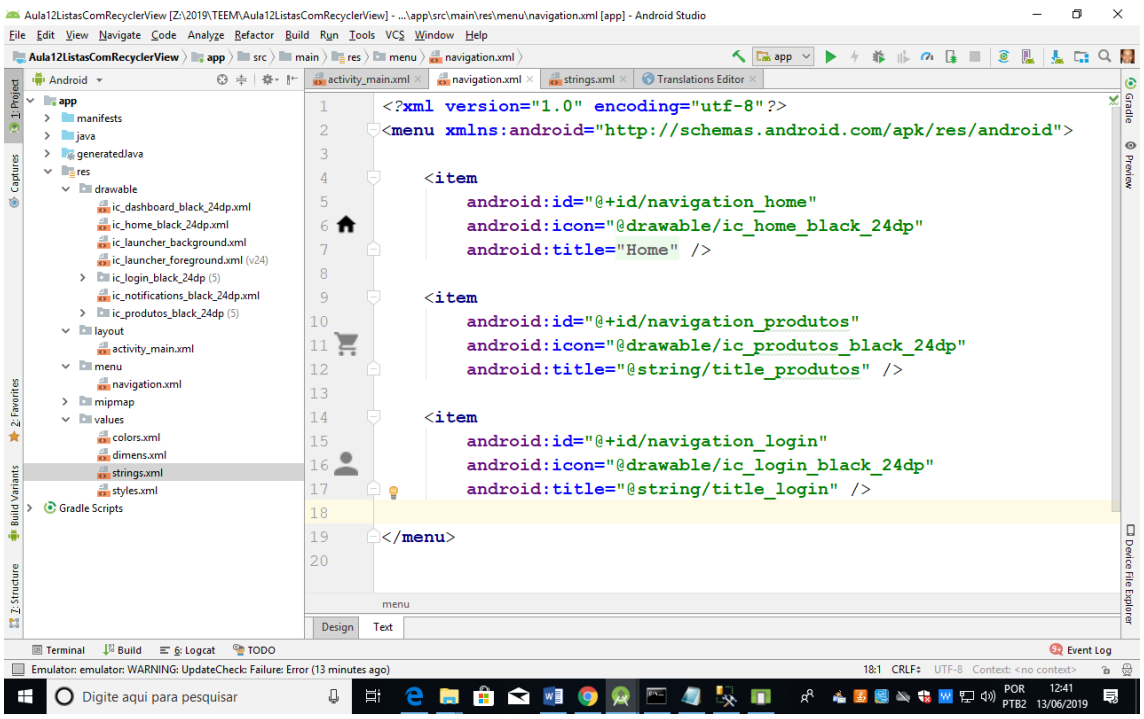


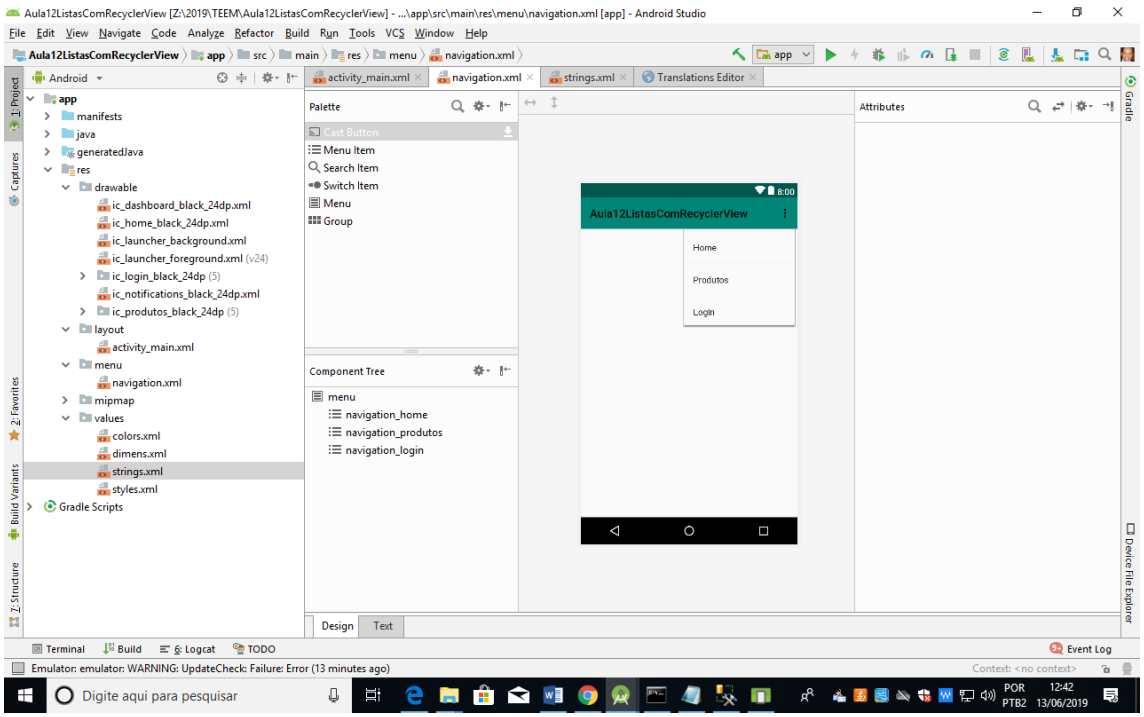
Em **Key** digite **title_produtos**
Em **Default Value** digite Produtos

Repita o processo para adicionar outra string e digite:

Em **Key** digite **title_login**
Em **Default Value** digite Login

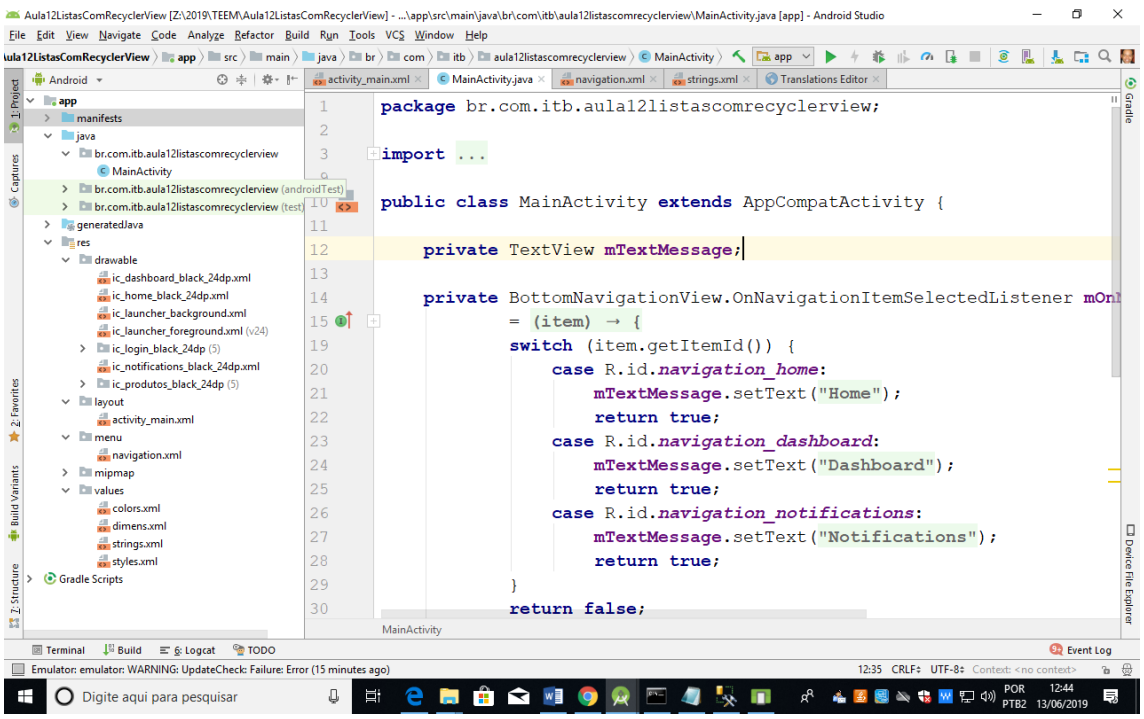
Observe que todos os erros desaparecem, e tudo fica verde:



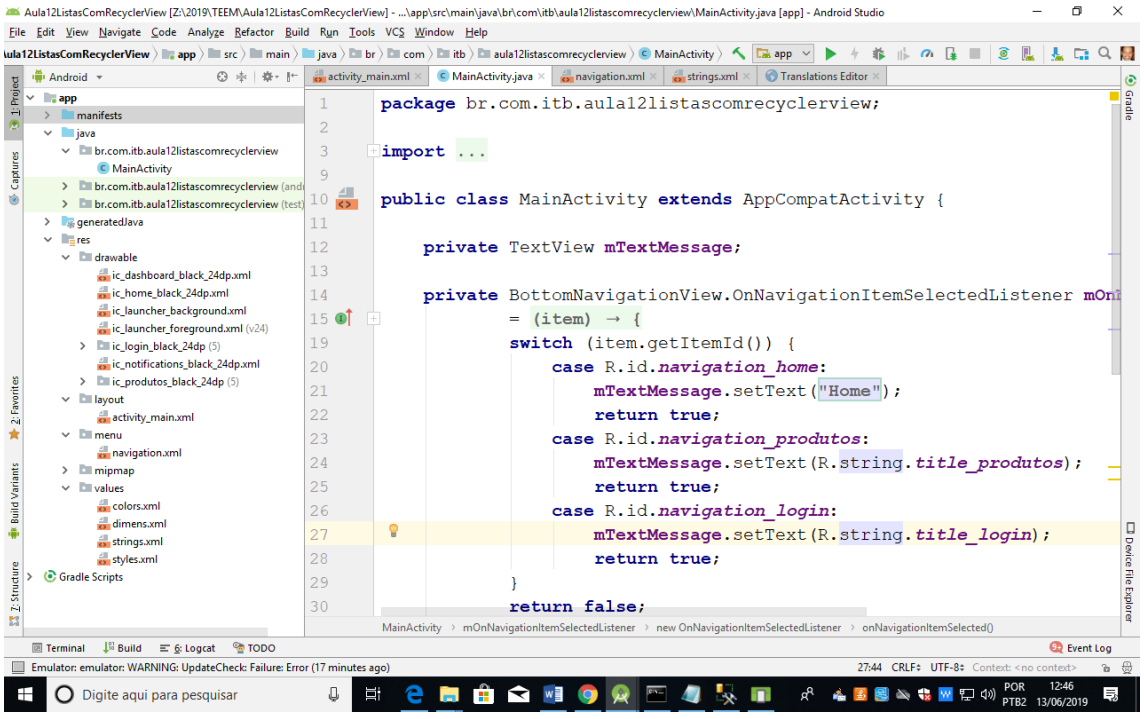


Agora vamos abrir a classe MainActivity.java em :

Android > app > java > <pacote> > MainActivity.java



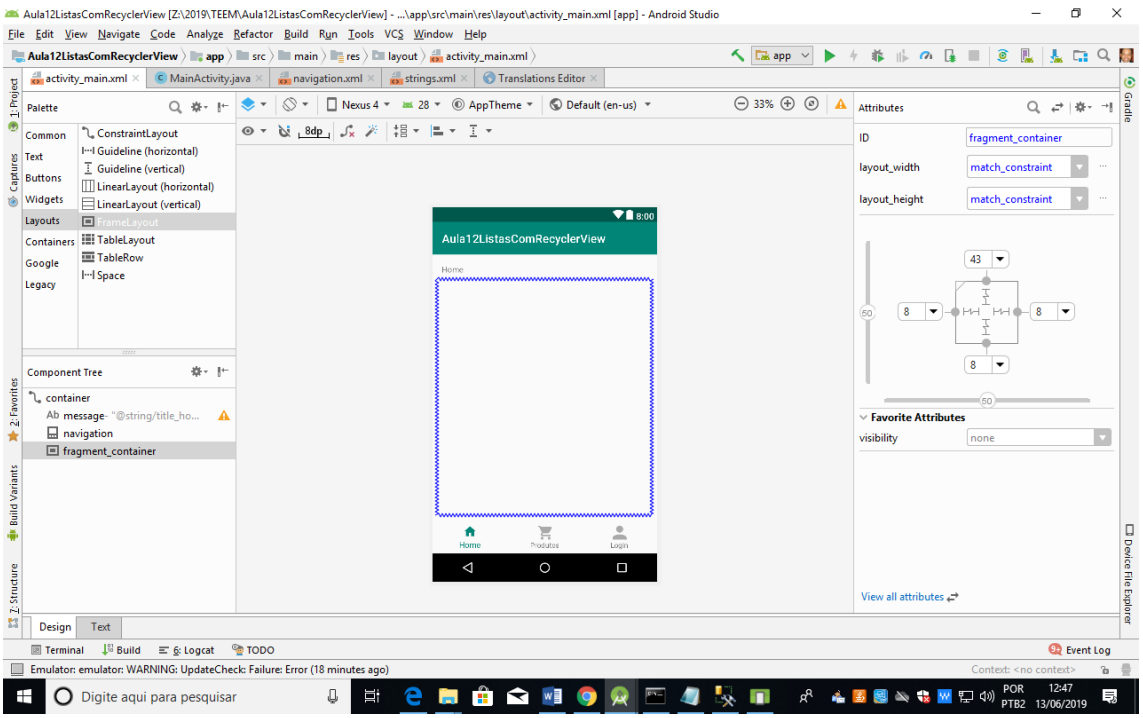
Altere os menus Dashboard e Notifications para Produtos e Login:




```
private BottomNavigationView.OnNavigationItemSelectedListener
mOnNavigationItemSelectedListener
        = new BottomNavigationView.OnNavigationItemSelectedListener()
{

    @Override
    public boolean onNavigationItemSelected(@NonNull MenuItem item) {
        switch (item.getItemId()) {
            case R.id.navigation_home:
                mTextMessage.setText(R.string.title_home);
```

```
                return true;
            case R.id.navigation_produtos:
                mTextMessage.setText(R.string.title_produtos);
                return true;
            case R.id.navigation_login:
                mTextMessage.setText(R.string.title_login);
                return true;
        }
        return false;
    }
};
```

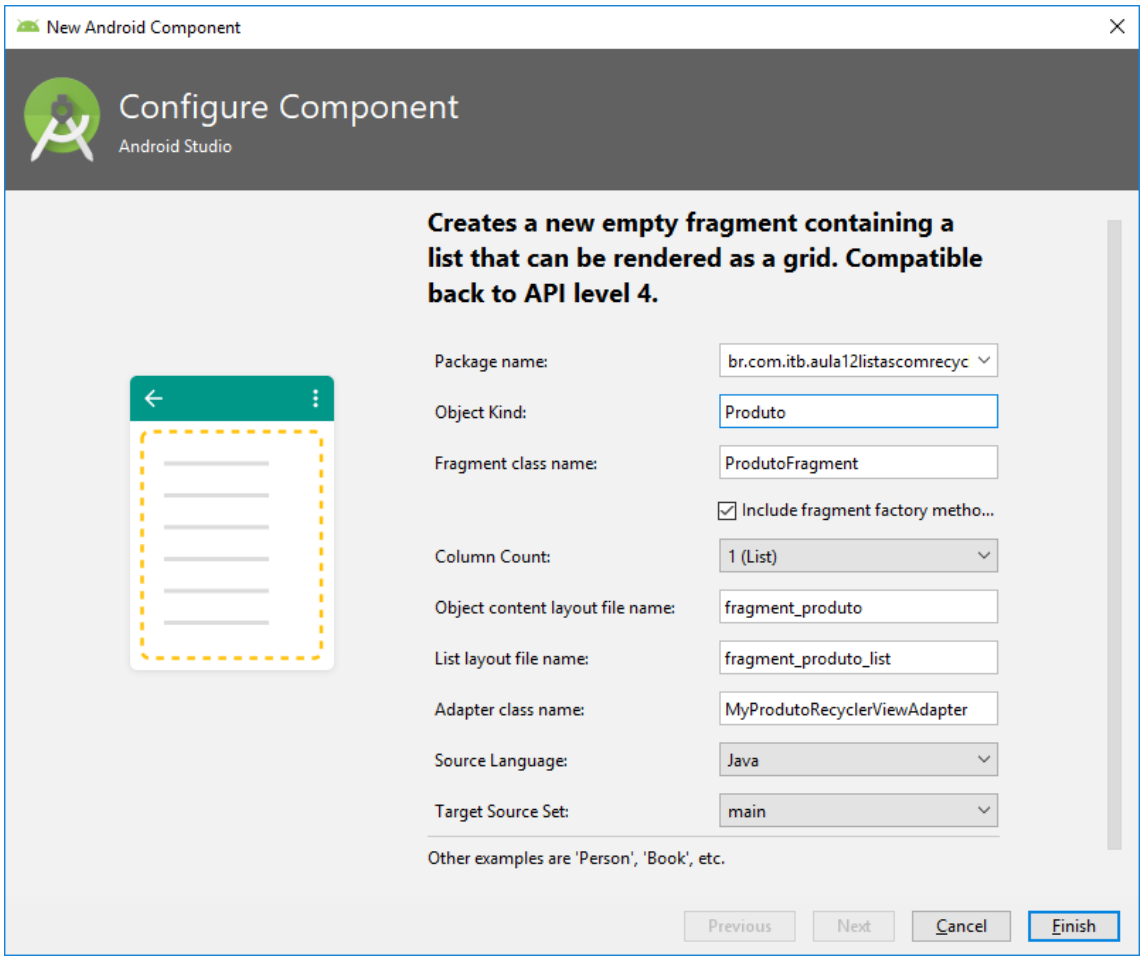
Nossa tela ficará como abaixo:



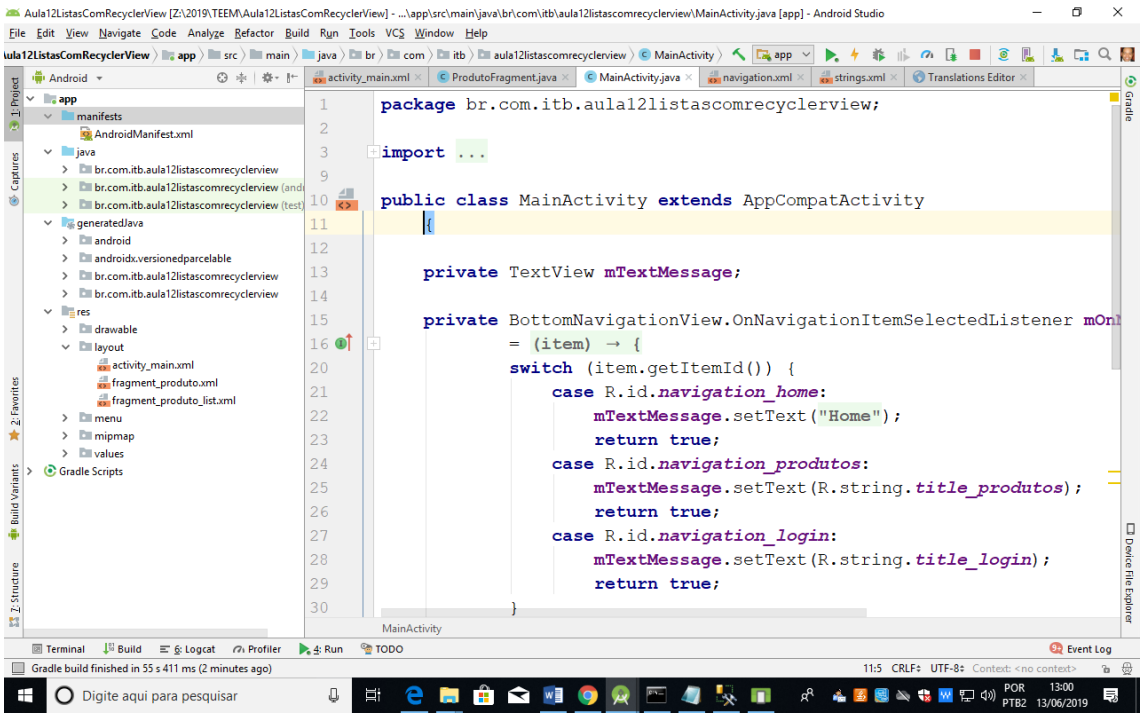
Neste momento vamos adicionar o fragmento para buscar produtos na base de dados SQL Server:

Clique em File > New > Fragment > Fragment (List)

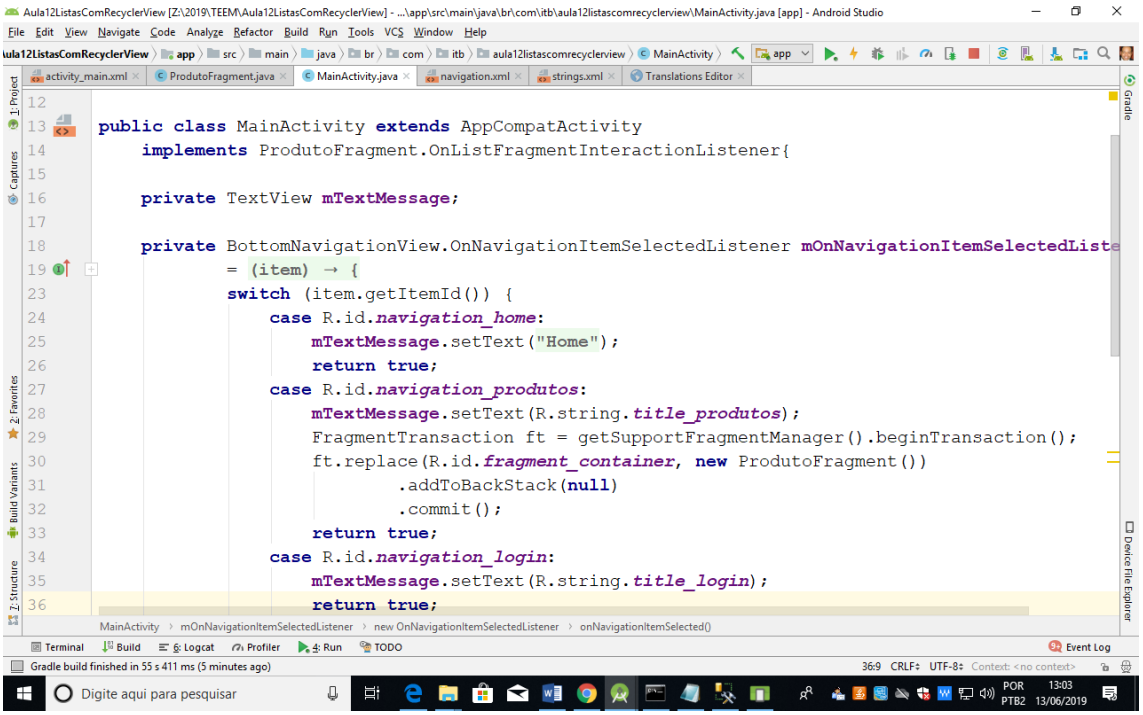
Altere o Object Kind para **Produto**



Clique em FINISH e aguarde o carregamento:



Altere o MainActivity com o código abaixo:



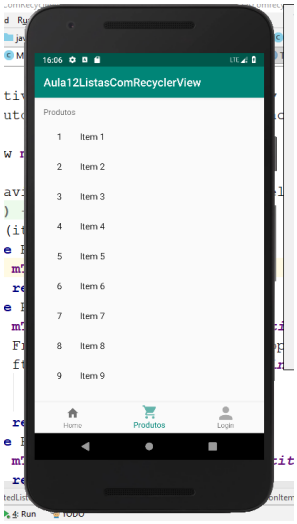
Assinatura da classe MainActivity:

```
public class MainActivity extends AppCompatActivity
    implements ProdutoFragment.OnListFragmentInteractionListener{
...
```

Alteração no case do menu Produtos, para acionar o fragmento de produtos:

```
case R.id.navigation_produtos:
    mTextMessage.setText(R.string.title_produtos);
    FragmentTransaction ft =
getSupportFragmentManager().beginTransaction();
    ft.replace(R.id.fragment_container, new ProdutoFragment())
            .addToBackStack(null)
            .commit();
    return true;
```

Teste o aplicativo para verificar se o carregamento do ProdutoFragment foi bem realizado:



Agora vamos modificar o fragment_produto.xml

Neste xml vamos adicionar os campos que serão utilizados para receberem os dados da tabela produto.

Antes vamos criar os recursos na strings.xml para utilizarmos na tela de listagem do produto:

```
<resources>
    <string name="app_name">Aula12ListasComRecyclerView</string>
    <string name="title_home">Home</string>
    <string name="title_dashboard">Dashboard</string>
    <string name="title_notifications">Notifications</string>
    <string name="title_produtos">Produtos</string>
    <string name="title_login">Login</string>
    <string name="codigo">Código</string>
    <string name="descricao">Descrição</string>
    <string name="quantidade">Qtde</string>
    <string name="valor_unitario">Valor Unit</string>
    <string name="status">Situação</string>
    <string name="ativo">Ativo</string>
</resources>
```

Adicione os objetos, conforme o xml abaixo:

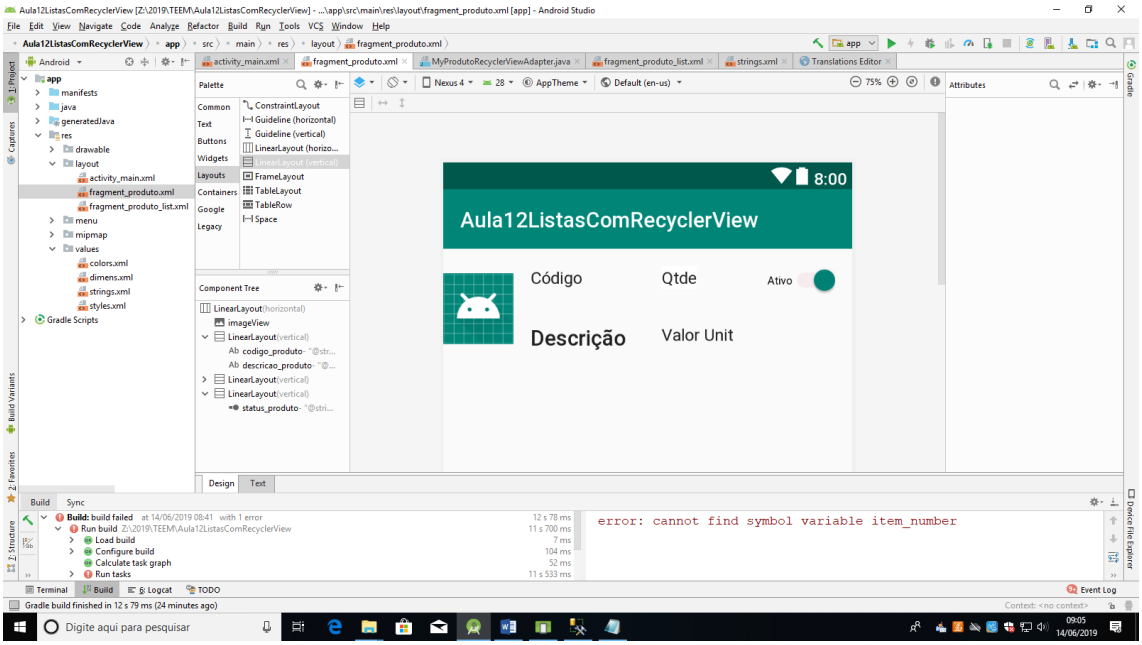
Basta copiar o código abaixo e sobrescrever o código do fragment_produto.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
```

```
    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        app:srcCompat="@mipmap/ic_launcher" />

    <LinearLayout
        android:layout_width="123dp"
        android:layout_height="wrap_content"
        android:orientation="vertical">

        <TextView
            android:id="@+id/codigo_produto"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="@dimen/text_margin"
            android:text="@string/codigo"

android:textAppearance="@android:style/TextAppearance.DeviceDefault.Se
archResult.Subtitle" />

        <TextView
            android:id="@+id/descricao_produto"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="@dimen/text_margin"
            android:text="@string/descricao"

android:textAppearance="@android:style/TextAppearance.DeviceDefault.Wi
dget.ActionBar.Title" />

    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

        <TextView
            android:id="@+id/quantidade_produto"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="@dimen/text_margin"
            android:background="@android:color/background_light"
            android:text="@string/quantidade"

android:textAppearance="@android:style/TextAppearance.DeviceDefault.Se
archResult.Subtitle" />

        <TextView
            android:id="@+id/valor_produto"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="@dimen/text_margin"
            android:text="@string/valor_unitario"

android:textAppearance="@android:style/TextAppearance.DeviceDefault.Se
archResult.Subtitle" />

    </LinearLayout>
```

```
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

        <Switch
            android:id="@+id/status_produto"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_margin="@dimen/text_margin"
            android:checked="true"

android:switchTextAppearance="@android:style/TextAppearance.Holo.Searc
hResult.Subtitle"
            android:text="@string/ativo"
            android:textSize="10sp"
            android:thumbTint="@color/colorPrimary" />
    </LinearLayout>

</LinearLayout>
```

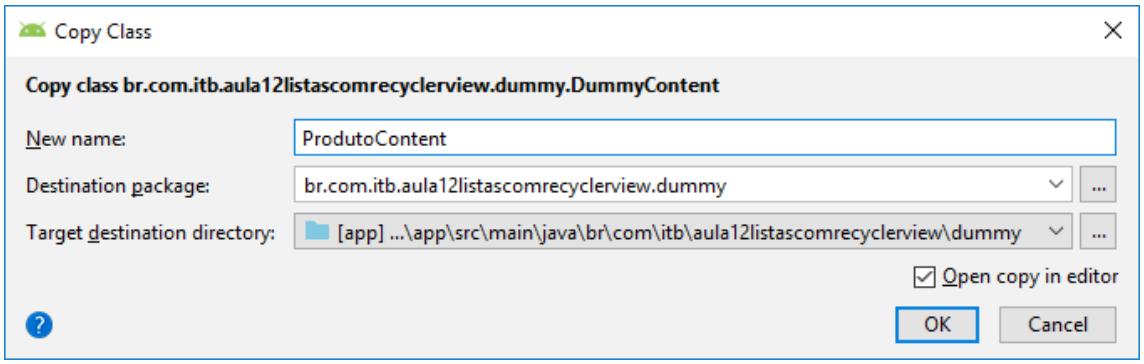
O layout desta tela deve ficar com esta aparência:



Agora vamos programar a classe `MyProdutoRecyclerViewAdapter`
Para ajustar com os atributos do produto:

Observe na linha 21 que os dados do produto serão listados a partir de um objeto chamado mValues de uma List<DummyItem>, temos que substituir em todo código DummyItem pela classe ProdutoItem, que será apresentada agora e deve ser inserida no projeto.

Crie uma classe Java chamada ProdutoContent, cópia de DummyContent:




```
package br.com.itb.aula12listascomrecyclerview.dummy;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

/**
 * Helper class for providing sample content for user interfaces
created by
 * Android template wizards.
 * <p>
 * TODO: Replace all uses of this class before publishing your app.
 */
public class ProdutoContent {

    /**
     * An array of sample (dummy) items.
     */
    public static final List<ProdutoItem> ITEMS = new
ArrayList<ProdutoItem>();

    /**
     * A map of sample (dummy) items, by ID.
     */
    public static final Map<String, ProdutoItem> ITEM_MAP = new
HashMap<String, ProdutoItem>();

    private static final int COUNT = 25;

    static {
        // Add some sample items.
        for (int i = 1; i <= COUNT; i++) {
            addItem(createProdutoItem(i));
        }
    }

    private static void addItem(ProdutoItem item) {
        ITEMS.add(item);
        ITEM_MAP.put(item.codigo, item);
    }

    private static ProdutoItem createProdutoItem(int position) {
        return new ProdutoItem(String.valueOf(position), "Produto " +
position, makeDetails(position), null, null);
    }

    private static String makeDetails(int position) {
```

```
        StringBuilder builder = new StringBuilder();
        builder.append("Detalhes sobre o produto: ").append(position);
        for (int i = 0; i < position; i++) {
            builder.append("\nMais informações sobre o produto.");
        }
        return builder.toString();
    }

    /**
     * A dummy item representing a piece of content.
     */
    public static class ProdutoItem {
        public final String codigo;
        public final String descricao;
        public final String qtde;
        public final String valor_unit;
        public final String status;

        public ProdutoItem(String codigo, String descricao, String
qtde, String valor_unit, String status) {
            this.codigo = codigo;
            this.descricao = descricao;
            this.qtde = qtde;
            this.valor_unit = valor_unit;
            this.status = status;
        }

        @Override
        public String toString() {
            return descricao;
        }
    }
}
```

Classe `MyProdutoRecyclerViewAdapter`:

```
package br.com.itb.aula12listascomrecyclerview;

import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import android.widget.Switch;

import
br.com.itb.aula12listascomrecyclerview.ProdutoFragment.OnListFragmentI
nteractionListener;
import
br.com.itb.aula12listascomrecyclerview.dummy.ProdutoContent.ProdutoIte
m;

import java.util.List;

/**
 * {@link RecyclerView.Adapter} that can display a {@link DummyItem}
and makes a call to the
 * specified {@link OnListFragmentInteractionListener}.
 * TODO: Replace the implementation with code for your data type.
```

```
 */
public class MyProdutoRecyclerViewAdapter extends
RecyclerView.Adapter<MyProdutoRecyclerViewAdapter.ViewHolder> {

    private final List<ProdutoItem> mValues;
    private final OnListFragmentInteractionListener mListener;

    public MyProdutoRecyclerViewAdapter(List<ProdutoItem> items,
OnListFragmentInteractionListener listener) {
        mValues = items;
        mListener = listener;
    }

    @Override
    public ViewHolder onCreateViewHolder(ViewGroup parent, int
viewType) {
        View view = LayoutInflater.from(parent.getContext())
                .inflate(R.layout.fragment_produto, parent, false);
        return new ViewHolder(view);
    }

    @Override
    public void onBindViewHolder(final ViewHolder holder, int
position) {
        holder.mItem = mValues.get(position);
        holder.mCodigoView.setText(mValues.get(position).codigo);

holder.mDescricaoView.setText(mValues.get(position).descricao);
        holder.mQtdeView.setText(mValues.get(position).qtde);
        holder.mValorView.setText(mValues.get(position).valor_unit);
        holder.mStatusView.setChecked(true);

        holder.mView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (null != mListener) {
                    // Notify the active callbacks interface (the
activity, if the
                    // fragment is attached to one) that an item has
been selected.
                    mListener.onListFragmentInteraction(holder.mItem);
                }
            }
        });
    }

    @Override
    public int getItemCount() {
        return mValues.size();
    }

    public class ViewHolder extends RecyclerView.ViewHolder {
        public final View mView;
        public final TextView mCodigoView;
        public final TextView mDescricaoView;
        public final TextView mQtdeView;
        public final TextView mValorView;
        public final Switch mStatusView;
        public ProdutoItem mItem;

        public ViewHolder(View view) {
```

```
            super(view);
            mView = view;
            mCodigoView = (TextView)
view.findViewById(R.id.codigo_produto);
            mDescricaoView = (TextView)
view.findViewById(R.id.descricao_produto);
            mQtdeView = (TextView)
view.findViewById(R.id.quantidade_produto);
            mValorView = (TextView)
view.findViewById(R.id.valor_produto);
            mStatusView = (Switch)
view.findViewById(R.id.status_produto);
        }

        @Override
        public String toString() {
            return super.toString() + " '" + mDescricaoView.getText()
+ "'";
        }
    }
}
```

Classe MainActivity:

```
package br.com.itb.aula12listascomrecyclerview;

import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.design.widget.BottomNavigationView;
import android.support.v4.app.FragmentTransaction;
import android.support.v7.app.AppCompatActivity;
import android.view.MenuItem;
import android.widget.TextView;

import br.com.itb.aula12listascomrecyclerview.dummy.ProdutoContent;

public class MainActivity extends AppCompatActivity
    implements ProdutoFragment.OnListFragmentInteractionListener{

    private TextView mTextMessage;

    private BottomNavigationView.OnNavigationItemSelectedListener
mOnNavigationItemSelectedListener
            = new
BottomNavigationView.OnNavigationItemSelectedListener() {

        @Override
        public boolean onNavigationItemSelected(@NonNull MenuItem
item) {
            switch (item.getItemId()) {
                case R.id.navigation_home:
                    mTextMessage.setText(R.string.title_home);
                    return true;
                case R.id.navigation_produtos:
                    mTextMessage.setText(R.string.title_produtos);
                    FragmentTransaction ft =
getSupportFragmentManager().beginTransaction();
                    ft.replace(R.id.fragment_container, new
ProdutoFragment())
                            .addToBackStack(null)
                            .commit();
```

```
                    return true;
                case R.id.navigation_login:
                    mTextMessage.setText(R.string.title_login);
                    return true;
            }
            return false;
        }
    };

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

        mTextMessage = (TextView) findViewById(R.id.message);
        BottomNavigationView navigation = (BottomNavigationView)
findViewById(R.id.navigation);

navigation.setOnNavigationItemSelectedListener(mOnNavigationItemSelect
edListener);
    }

    @Override
    public void onListFragmentInteraction(ProdutoContent.ProdutoItem
item) {

    }
}
```

Classe ProdutoFragment:

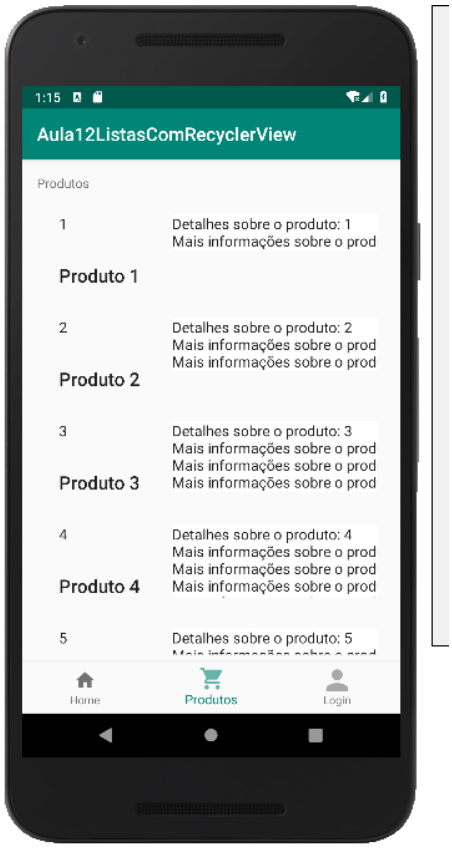
```
package br.com.itb.aula12listascomrecyclerview;

import android.content.Context;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.support.v7.widget.GridLayoutManager;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import br.com.itb.aula12listascomrecyclerview.dummy.ProdutoContent;
import
br.com.itb.aula12listascomrecyclerview.dummy.ProdutoContent.ProdutoIte
m;

import java.util.List;

/**
 * A fragment representing a list of Items.
 * <p/>
 * Activities containing this fragment MUST implement the {@link
OnListFragmentInteractionListener}
 * interface.
 */
public class ProdutoFragment extends Fragment {

    // TODO: Customize parameter argument names
```

```
    private static final String ARG_COLUMN_COUNT = "column-count";
    // TODO: Customize parameters
    private int mColumnCount = 1;
    private OnListFragmentInteractionListener mListener;

    /**
     * Mandatory empty constructor for the fragment manager to
instantiate the
     * fragment (e.g. upon screen orientation changes).
     */
    public ProdutoFragment() {
    }

    // TODO: Customize parameter initialization
    @SuppressWarnings("unused")
    public static ProdutoFragment newInstance(int columnCount) {
        ProdutoFragment fragment = new ProdutoFragment();
        Bundle args = new Bundle();
        args.putInt(ARG_COLUMN_COUNT, columnCount);
        fragment.setArguments(args);
        return fragment;
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        if (getArguments() != null) {
            mColumnCount = getArguments().getInt(ARG_COLUMN_COUNT);
        }
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container,
                             Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.fragment_produto_list,
container, false);

        // Set the adapter
        if (view instanceof RecyclerView) {
            Context context = view.getContext();
            RecyclerView recyclerView = (RecyclerView) view;
            if (mColumnCount <= 1) {
                recyclerView.setLayoutManager(new
LinearLayoutManager(context));
            } else {
                recyclerView.setLayoutManager(new
GridLayoutManager(context, mColumnCount));
            }
            recyclerView.setAdapter(new
MyProdutoRecyclerViewAdapter(ProdutoContent.ITEMS, mListener));
        }
        return view;
    }


    @Override
    public void onAttach(Context context) {
        super.onAttach(context);
        if (context instanceof OnListFragmentInteractionListener) {
```

```
            mListener = (OnListFragmentInteractionListener) context;
        } else {
            throw new RuntimeException(context.toString()
                    + " must implement
OnListFragmentInteractionListener");
        }
    }

    @Override
    public void onDetach() {
        super.onDetach();
        mListener = null;
    }

    /**
     * This interface must be implemented by activities that contain
this
     * fragment to allow an interaction in this fragment to be
communicated
     * to the activity and potentially other fragments contained in
that
     * activity.
     * <p/>
     * See the Android Training lesson <a href=
     *
"http://developer.android.com/training/basics/fragments/communicating.
html"
     * >Communicating with Other Fragments</a> for more information.
     */
    public interface OnListFragmentInteractionListener {
        // TODO: Update argument type and name
        void onListFragmentInteraction(ProdutoItem item);
    }
}
```

Até este ponto da aula basta trocar toda palavra Dummy por Produto, e se analisar os códigos, adicionamos os campos criados na tabela do script apresentado, quantidade, valor e status.

Teste o aplicativo novamente neste ponto, uma lista como apresentada abaixo deve aparecer, quando clicar no menu Produtos:



O desafio agora é trazer os dados do banco de dados SQL Server, para isso acontecer vamos utilizar a classe Conexao.java abaixo descrita:

```
package br.com.itb.aula12listascomrecyclerview;

import android.os.StrictMode;
import android.support.design.widget.Snackbar;
import android.util.Log;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import br.com.itb.aula12listascomrecyclerview.dummy.ProdutoContent;

public class Conexao {
    public static Connection conexaoBD(){
        Connection conexao = null;
```

```
        try {
            StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy
                    .Builder().permitAll().build();
            StrictMode.setThreadPolicy(policy);

Class.forName("net.sourceforge.jtds.jdbc.Driver").newInstance();
            //NÃO ESQUECER DE DESCOBRIR O IP DA MÁQUINA ONDE ESTA O
SQL SERVER
            String IP = "172.19.0.73";
            conexao =
DriverManager.getConnection("jdbc:jtds:sqlserver://" + IP + ";" +

"databaseName=TEEM_ANDROID;user=sa;password=123456;");
        }catch (Exception e){
            e.getMessage().toString();
        }
        return conexao;
    }

    //Método de pesquisa de produtos na base SQL Server
    public static ArrayList<ProdutoContent.ProdutoItem>
pesquisarProdutos(){
        ArrayList<ProdutoContent.ProdutoItem> lista = new
ArrayList<ProdutoContent.ProdutoItem>();
        try{
            //Pesquisa no banco de dados
            PreparedStatement pst =
Conexao.conexaoBD().prepareStatement("select * from produto");
            //Resultado da pesquisa realizada
            ResultSet res = pst.executeQuery();
            //Se existir linhas no objeto res, a lista será carregada
com produtos
            while(res.next()){
                ProdutoContent.ProdutoItem produto = new
ProdutoContent.ProdutoItem(
                        res.getString(1),
                        res.getString(2),
                        res.getInt(3),
                        res.getDouble(4),
                        res.getInt(5)
                );
                lista.add(produto);
            }
        }catch(SQLException e){
            e.getMessage().toString();
        }
        return lista;
    }
}
```

Agora insira os recursos no arquivo strings.xml

```
<string name="disponivel">Disponível</string>
<string name="indisponivel">Indisponível</string>
```

Nova alteração no arquivo ProdutoContent.java, para fazer o objeto estático procurar os produtos na base de dados e carrega-los no objeto estático ITEMS, que é a listagem que será devolvida para a tela.

```
package br.com.itb.aula12listascomrecyclerview.dummy;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import br.com.itb.aula12listascomrecyclerview.Conexao;

public class ProdutoContent {

    /**
     * Os produtos serão carregados e devolvidos na tela neste objeto
ITEMS.
     */
    public static final ArrayList<ProdutoItem> ITEMS = new
ArrayList<ProdutoItem>();

    /**
     * Esse é o mapeamento dos objetos listados para o adaptador do
RecycleView.
     */
    public static final Map<String, ProdutoItem> ITEM_MAP = new
HashMap<String, ProdutoItem>();

    // Esta estrutura estática vai realizar a pesquida no banco de
dados
    static {
        // Pesquisa realizada e devolvida na lista
        ArrayList<ProdutoItem> lista = Conexao.pesquisarProdutos();
        // Aqui a lista é percorrida (lida) e adiciona item a item no
ITEMS acima
        for(ProdutoItem produtoItem: lista){
            addItem(createProdutoItem(produtoItem));
        }
    }

    // Método que adiciona items quando chamado
    private static void addItem(ProdutoItem item) {
        ITEMS.add(item);
        ITEM_MAP.put(item.codigo, item);
    }

    // Método que cria item a item no objeto ITEMS
    private static ProdutoItem createProdutoItem(ProdutoItem produto)
{
        return new ProdutoItem(produto.codigo, produto.descricao,
produto.qtde, produto.valor_unit, produto.status);
    }

    // Exemplo de criação de detalhe do produto na listagem
    // Não estou utilizando este método, mas o deixei ai como exemplo
para os alunos
    private static String makeDetails(int position) {
        StringBuilder builder = new StringBuilder();
        builder.append("Detalhes sobre o produto: ").append(position);
```

```
        for (int i = 0; i < position; i++) {
            builder.append("\nMais informações sobre o produto.");
        }
        return builder.toString();
    }

    /**
     * Classe estática do ProdutoItem (produto)
     * Observe que os tipos de dados são os mesmos do banco de dados
     */
    public static class ProdutoItem {
        public final String codigo;
        public final String descricao;
        public final int qtde;
        public final double valor_unit;
        public final int status;

        public ProdutoItem(String codigo, String descricao, int qtde,
double valor_unit, int status) {
            this.codigo = codigo;
            this.descricao = descricao;
            this.qtde = qtde;
            this.valor_unit = valor_unit;
            this.status = status;
        }

        @Override
        public String toString() {
            return descricao;
        }
    }
}
```

Vamos apresentar agora as alterações no adaptador do Recycler View, responsável por efetuar o de-para, isto é, capturar o resultado da pesquisa realizada e enviar este resultado para a tela da listagem.

```
package br.com.itb.aula12listascomrecyclerview;

import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.TextView;
import android.widget.Switch;

import
br.com.itb.aula12listascomrecyclerview.ProdutoFragment.OnListFragmentI
nteractionListener;
import
br.com.itb.aula12listascomrecyclerview.dummy.ProdutoContent.ProdutoIte
m;

import java.util.ArrayList;
import java.util.List;
```

```
/**
 * {@link RecyclerView.Adapter} that can display a {@link ProdutoItem}
and makes a call to the
 * specified {@link OnListFragmentInteractionListener}.
 * TODO: Replace the implementation with code for your data type.
 */
public class MyProdutoRecyclerViewAdapter extends
RecyclerView.Adapter<MyProdutoRecyclerViewAdapter.ViewHolder> {

    private final ArrayList<ProdutoItem> mValues;
    private final OnListFragmentInteractionListener mListener;

    public MyProdutoRecyclerViewAdapter(ArrayList<ProdutoItem> items,
OnListFragmentInteractionListener listener) {
        mValues = items;
        mListener = listener;
        //Adicionei este método para informar alterações no objeto de
pesquisa
        notifyDataSetChanged();
    }

    @Override
    public ViewHolder onCreateViewHolder(ViewGroup parent, int
viewType) {
        View view = LayoutInflater.from(parent.getContext())
                .inflate(R.layout.fragment_produto, parent, false);
        return new ViewHolder(view);
    }

    /*
    Aqui nesta método cada valor capturado é adicionado na tela,
            no seu objeto correspondente
    */
    @Override
    public void onBindViewHolder(final ViewHolder holder, int
position) {

        holder.mItem = mValues.get(position);
        // O código será apresentado no objeto da tela
        holder.mCodigoView.setText(mValues.get(position).codigo);
        // A descrição será apresentada no objeto correspondente da
tela

holder.mDescricaoView.setText(mValues.get(position).descricao);
        // a quantidade será apresentada no objeto correspondente da
tela

holder.mQtdeView.setText(String.valueOf(mValues.get(position).qtde));
        // O valor unitário será apresentado no objeto correspondente
da tela

holder.mValorView.setText(String.valueOf(mValues.get(position).valor_u
nit));

        // O status capturado é apresentado, de acordo com o valor
        // Se o valor for 1 o produto mostra que está disponível
        // Senão o produto será apresentado como indisponível
        if(mValues.get(position).status == 1){
            holder.mStatusView.setChecked(true);
            holder.mStatusView.setTextOn("Disponível");
        }else{
```

```
            holder.mStatusView.setChecked(false);
            holder.mStatusView.setTextOff("Indisponível");
        }
        // Este método aguarda um clique no item selecionado da
listagem
        // Aqui faremos o link para o detalhamento do produto
selecionado
        holder.mView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (null != mListener) {
                    // Notify the active callbacks interface (the
activity, if the
                    // fragment is attached to one) that an item has
been selected.
                    mListener.onListFragmentInteraction(holder.mItem);
                }
            }
        });
    }
    @Override
    public int getItemCount() {
        return mValues.size();
    }

    // Método para estabelecer o vínculo entre os objetos da tela XML
    // e os objetos criados no Java
    public class ViewHolder extends RecyclerView.ViewHolder {
        // Objetos Java
        public final View mView;
        public final TextView mCodigoView;
        public final TextView mDescricaoView;
        public final TextView mQtdeView;
        public final TextView mValorView;
        public final Switch mStatusView;
        public ProdutoItem mItem;

        // Vínculo dos objetos Java acima e os objetos da tela.
        //  Exemplo - R.id.codigo_produto é o mCodigoView
        public ViewHolder(View view) {
            super(view);
            mView = view;
            mCodigoView = (TextView)
view.findViewById(R.id.codigo_produto);
            mDescricaoView = (TextView)
view.findViewById(R.id.descricao_produto);
            mQtdeView = (TextView)
view.findViewById(R.id.quantidade_produto);
            mValorView = (TextView)
view.findViewById(R.id.valor_produto);
            mStatusView = (Switch)
view.findViewById(R.id.status_produto);
        }

        @Override
        public String toString() {
            return super.toString() + " '" + mDescricaoView.getText()
+ "'";
        }
    }
}
```

Nenhuma alteração foi realizada no ProdutoFragment.java.

A próxima etapa deste tutorial é a realização da pesquisa no banco de dados, somente com a informação fornecida no filtro:

Adicione uma cor no arquivo colors.xml em Android > app > res > values

```
<color name="bege">#fee1a1</color>
```

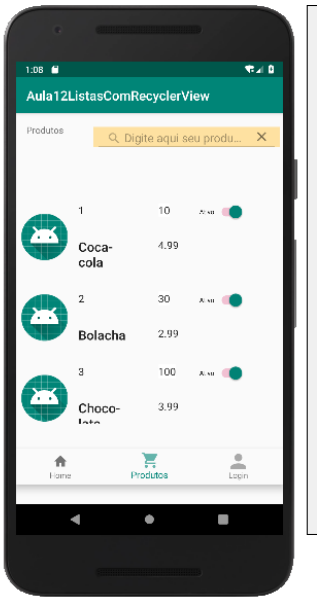
Adicione mais um recurso de linguagem no arquivo strings.xml

```
<string name="pesquisar_produto">Digite aqui seu produto...</string>
```

Agora vamos alterar o layout da tela da listagem, e adicionar um objeto SearchView na tela do activity_main.xml, altere os valores dos atributos:

- **id** : svPesquisar
- **background**: @color/bege
- **queryHint**: @string/pesquisar_produto
- **style**: @android:style/Widget.Material.Light.SearchView

Ao executar o app e chamar o item Produtos do menu, ficará com esta aparência:



Agora vamos alterar o código da classe ProdutoFragment.java para buscar apenas o que for digitado no filtro. Antes devemos alterar a classe Conexao para inserir mais um método de pesquisa por filtro:

```
package br.com.itb.aula12listascomrecyclerview;
```

```
import android.os.StrictMode;
import android.support.design.widget.Snackbar;
import android.util.Log;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import br.com.itb.aula12listascomrecyclerview.dummy.ProdutoContent;

public class Conexao {
    public static Connection conexaoBD(){
        Connection conexao = null;
        try {
            StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy
                    .Builder().permitAll().build();
            StrictMode.setThreadPolicy(policy);

Class.forName("net.sourceforge.jtds.jdbc.Driver").newInstance();
            //NÃO ESQUECER DE DESCOBRIR O IP DA MÁQUINA ONDE ESTA O
SQL SERVER
            String IP = "172.19.0.73";
            conexao =
DriverManager.getConnection("jdbc:jtds:sqlserver://" + IP + ";" +

"databaseName=TEEM_ANDROID;user=sa;password=123456;");
        }catch (Exception e){
            e.getMessage().toString();
        }
        return conexao;
    }

    //Método de pesquisa de produtos na base SQL Server
    public static ArrayList<ProdutoContent.ProdutoItem>
pesquisarProdutos(){
        ArrayList<ProdutoContent.ProdutoItem> lista = new
ArrayList<ProdutoContent.ProdutoItem>();
        try{
            //Pesquisa no banco de dados
            PreparedStatement pst =
Conexao.conexaoBD().prepareStatement("select * from produto");
            //Resultado da pesquisa realizada
            ResultSet res = pst.executeQuery();
            //Se existir linhas no objeto res, a lista será carregada
com produtos
            while(res.next()){
                ProdutoContent.ProdutoItem produto = new
ProdutoContent.ProdutoItem(
                        res.getString(1),
                        res.getString(2),
                        res.getInt(3),
                        res.getDouble(4),
                        res.getInt(5)
                );
                lista.add(produto);
            }
```

```
        }catch(SQLException e){
            e.getMessage().toString();
        }
        return lista;
    }

    //Método de pesquisa de produtos na base SQL Server
    public static ArrayList<ProdutoContent.ProdutoItem>
pesquisarProdutosFiltro(String texto){
        ArrayList<ProdutoContent.ProdutoItem> lista = new
ArrayList<ProdutoContent.ProdutoItem>();
        try{
            //Pesquisa no banco de dados
            PreparedStatement pst =
Conexao.conexaoBD().prepareStatement("select * from produto " +
                    "where codigo like '%" + texto.toString() + "%' "
+
                    "or descricao like '%" + texto.toString() + "%'");
            //Resultado da pesquisa realizada
            ResultSet res = pst.executeQuery();
            //Se existir linhas no objeto res, a lista será carregada
com produtos
            while(res.next()){
                ProdutoContent.ProdutoItem produto = new
ProdutoContent.ProdutoItem(
                        res.getString(1),
                        res.getString(2),
                        res.getInt(3),
                        res.getDouble(4),
                        res.getInt(5)
                );
                lista.add(produto);
            }
        }catch(SQLException e){
            e.getMessage().toString();
        }
        return lista;
    }
}
```

Agora o trecho de código do ProdutoFragment.java que deve ser alterado:

Referência Bibliográfica

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

[2] https://pt.stackoverflow.com/questions/246212/salvar-na-galeria-imagem-tirada-pela-c%C3%A2mera-e-pegar-caminho-dessa-imagem-andr. Acessado em 26/05/2019.

[3] https://cursos.alura.com.br/forum/topico-abrir-galeria-de-foto-no-android-e-colocar-foto-no-imageview-28141#554095. Acessado em 27/05/2019.