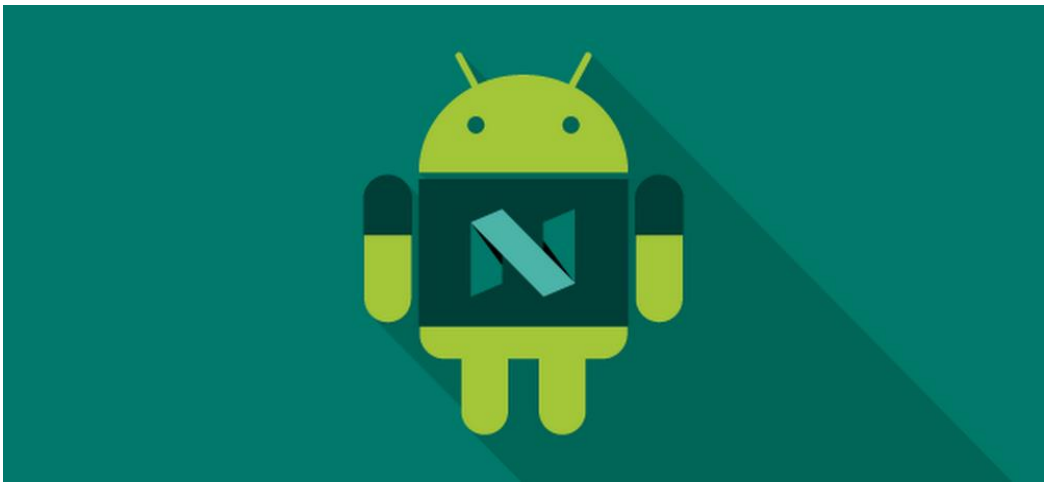
Infoweek Minicurso Android



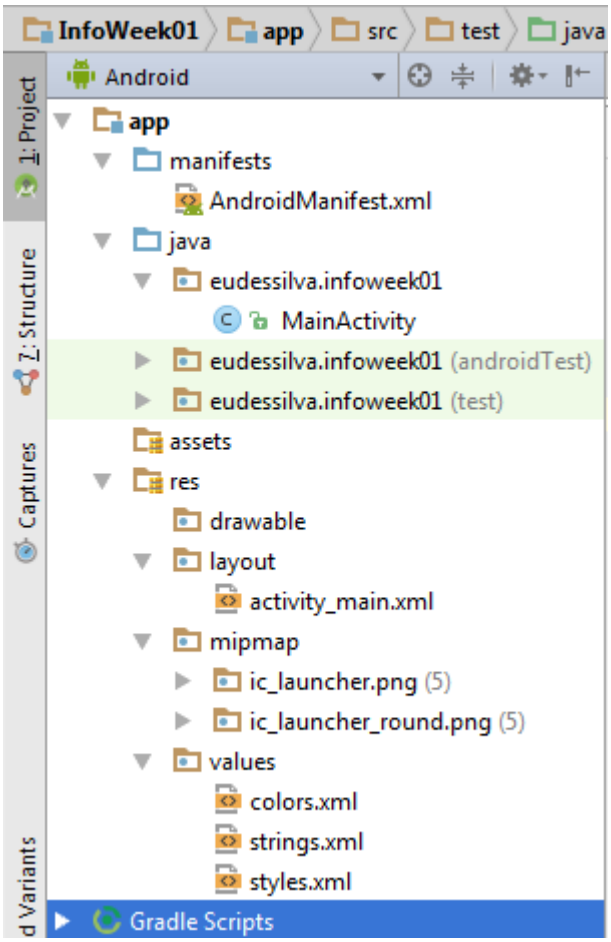
Quem eu sou :)

Eudes

Vida de proxy

Configurando proxy gradle

Estrutura padrão de projeto



app - Contém o núcleo do seu aplicativo, incluindo códigos de classes em Java, arquivos XML para design da interface gráfica, configurações, ícones e outros valores.

res - que fica localizada no caminho **app/src/main/res**. É nela que ficam todos os recursos do seu projeto, que não são necessariamente código Java. O projeto, no momento, deve ter algumas pastas, como **drawable** (vazia), **mipmap** e **values**.

manifests - aqui você encontra o manifesto do app. Este arquivo, AndroidManifest.xml define as propriedades do app, as telas registradas (Activities), as permissões exigidas pelo app, indicação quanto à versão mínima suportada, entre outras.

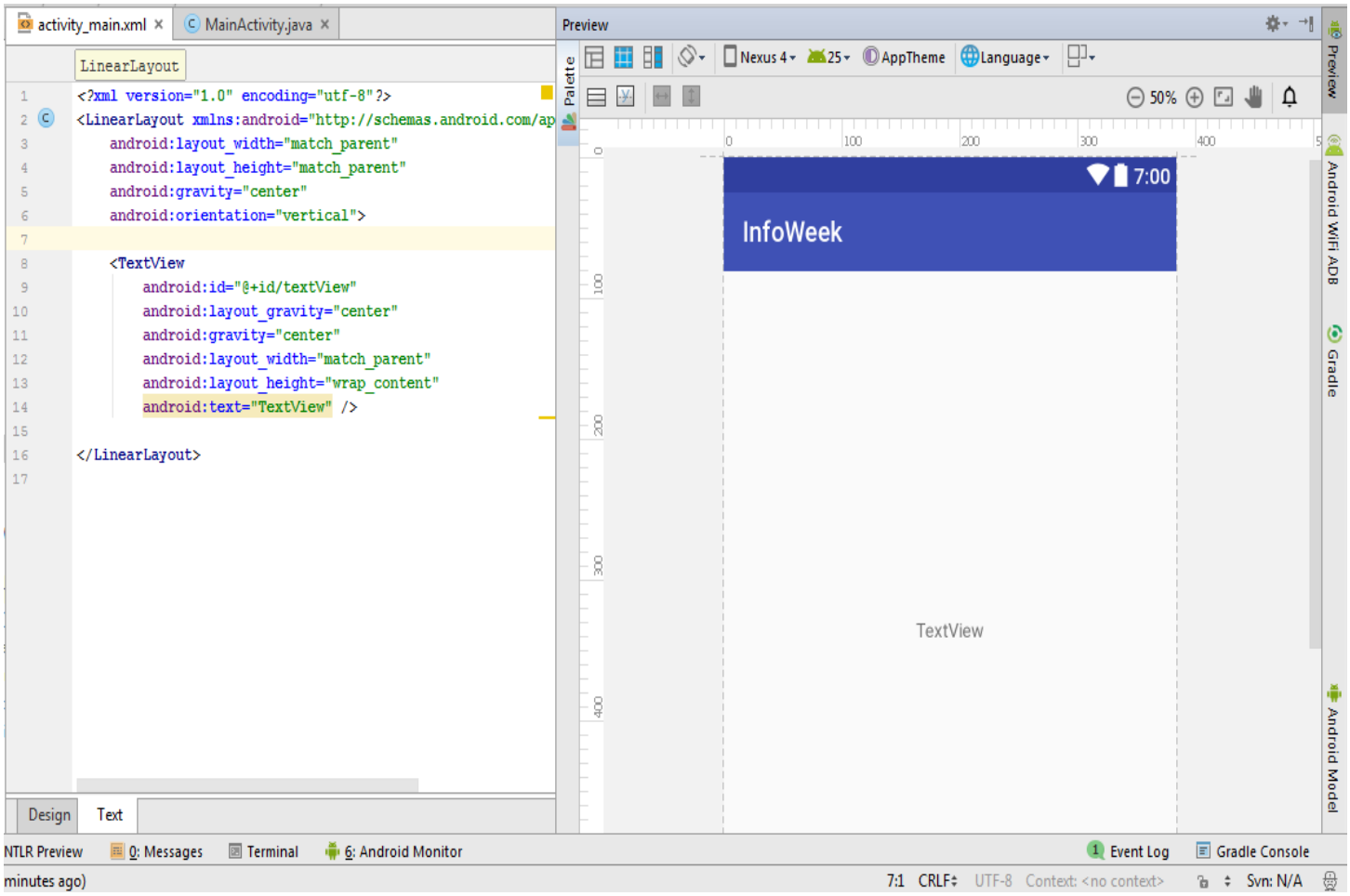
java - local para armazenar as classes Java que formam o núcleo das funcionalidades do aplicativo.

gradle - esse diretório contém um importante arquivo chamado gradle-wrapper, que será utilizado na construção do APK. Ele comunica-se com o sistema operacional e garante que o projeto será construído com a versão correta do Gradle.

Tabela com definição de Recursos

Diretório	Tipo de Resource
animator/	Arquivos XML que definem animações de propriedades.
anim/	Arquivos XML que definem tween animations. (Animações de propriedades podem ser salvas aqui também, mas o diretório *animator/* é preferível, por questão de organização)
color/	Arquivos XML que definem um estado de lista de cores. Veja Color State List Resource
drawable/	Arquivos Bitmap (.png, .jpg, .gif) ou arquivos XML que são compilados nos seguintes resources: • arquivos Bitmap • 9-Patch (bitmaps redimensionáveis) • listas de estado • Shapes (formas) • Animações (frame animations) • Outros tipos Veja Drawable Resources para mais informações.
mipmap/	Arquivos "drawable" especificamente para ícones de aplicações de diferentes tipos de densidade.
layout/	Arquivos XML que definem sua UI (user interface).
menu/	Arquivos XML que definem os menus da sua aplicação, como Menu de Opções, de Contexto e Sub-menus. Mais informações em: Menu Resource.
raw/	Pasta para salvar arquivos na sua forma natural (raw).
values/	Arquivos XML que contêm valores, como strings, inteiros, e cores. Dentro dessa pasta você deve nomear cada arquivo XML com o nome do seu resource e dessa forma irá acessar com uma subclasse de R. Por exemplo, o arquivo string.xml será acessado por R.string. Abaixo algumas convenções: • arrays.xml para Arrays. • colors.xml para valores de cores. • dimens.xml para dimensões. • strings.xml para todas as strings. • styles.xml para estilos.
xml/	Arquivos XML que podem ser lidos em tempo de execução chamando Resources.getXML(). Vários arquivos XML de configuração podem ser salvos aqui, por exemplo.

Layout e Preview



Primeiros passos

RecyclerView

RecyclerView é a versão mais avançada e flexível e eficiente do **ListView** com grande conjunto de dados de visualizações que podem ser rolados e reciclados.

Para usar o RecyclerView em seu projeto, você precisa adicionar a biblioteca de suporte de exibição de reciclador ao seu projeto.

Adicionando dependência para RecyclerView

```
1.  dependencies {
2.      compile 'com.android.support:recyclerview-v7:23.1.1'
3.  }
```

Criando o Layout para RecyclerView

activity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/an-
droid"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">


    <android.support.v7.widget.RecyclerView
        android:id="@+id/recycler_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="20dp" />


</LinearLayout>
```

Criando Layout para item row

Item_row.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:padding="20dp">

    <ImageView
        android:id="@+id/image"
        android:layout_width="match_parent"
        android:layout_height="100dp"
        android:background="#11000000"
        android:scaleType="centerCrop" />

    <TextView
        android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:textSize="18sp" />

</LinearLayout>
```

Criando a Activity

MainActivity.java

```
public class MainActivity extends AppCompatActivity {

    private List<String> itemList = new ArrayList<>();
    private AdapterProduct adapterProduct;

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

        RecyclerView recyclerView = (RecyclerView)
findViewById(R.id.recycler_view);
        recyclerView.setHasFixedSize(true);
        recyclerView.setLayoutManager(new LinearLayoutManager(this));
        fakeProductList();
        adapterProduct = new AdapterProduct(itemList, this);
        recyclerView.setAdapter(adapterProduct);

    }

    public void fakeProductList() {
        for (int i = 0; i < 20; i++) {
            itemList.add("This is product of number " + i);
        }
    }
}
```

Criando Adaptador para RecyclerView

```
public class AdapterProduct extends RecyclerView.Adapter<AdapterProduct.ViewHolderItem> {

    public Context context;
    ViewHolderItem viewHolder;
    private List<String> itemList;

    public AdapterProduct(List<String> itemList, Context context) {
        this.itemList = itemList;
        this.context = context;
    }

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

    public void onBindViewHolder(final ViewHolderItem viewHolder, final int position) {
        viewHolder.textView.setText(itemList.get(position));
        viewHolder.textView.setOnClickListener(new View.OnClickListener() {
           @Override
           public void onClick(View v) {
               Toast.makeText(v.getContext(), "OnClick :" + itemList.get(position),
Toast.LENGTH_SHORT).show();
           }
       });
    }

    @Override
    public ViewHolderItem onCreateViewHolder(ViewGroup parent, int viewType) {
        View itemLayoutView =
LayoutInflater.from(parent.getContext()).inflate(R.layout.item_row, null);
        viewHolder = new ViewHolderItem(itemLayoutView);
        return viewHolder;
    }

    // inicializar componentes para item da lista
    public static class ViewHolderItem extends RecyclerView.ViewHolder {

        public TextView textView;
        public ImageView imageView;

        public ViewHolderItem(View view) {
            super(view);
            textView = (TextView) view.findViewById(R.id.text);
            imageView = (ImageView) view.findViewById(R.id.image);
        }
    }
}
```

Criando métodos para remover e adicionar (Adapter)

```
public void addItem(String newItem){
    itemList.add(newItem);
    notifyDataSetChanged();
}

public void removeItem(int idItem){
    itemList.remove(idItem);
    notifyItemRemoved(idItem);
}
```

Chamando métodos de adicionar e remover (Adapter)

```
// chamando método de exclusão
viewHolder.btnExcluir.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        removeItem(position);
    }
});


// incluir button para excluir item
 public static class ViewHolderItem extends RecyclerView.ViewHolder {

    public Button btnExcluir;

    public ViewHolderItem(View view) {
        super(view);
        btnExcluir = (Button) view.findViewById(R.id.btn_excluir);
    }
}


// adicionar button no layout item_row.xml
<Button
  android:id="@+id/btn_excluir"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="Excluir"
  android:layout_alignBottom="@+id/image"
  android:layout_alignParentEnd="true" />
```

Chamando método de adicionar novo item (MainActivity)

//adicionar button em **activity_main.xml**

```
<Button
    android:id="@+id/btn_add"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Add new Item" />
```

// chamando método em **MainActivity.java**

```
Button btnAdd = (Button)findViewById(R.id.btn_add);
btnAdd.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        adapterProduct.addItem("Novo Item Added !!!");
    }
});
```

Banco de Dados

Bibliotecas para ORM

compile **'com.j256.ormlite:ormlite-android:4.48'**
compile **'com.j256.ormlite:ormlite-core:4.48'**

Criando as Entidades do Banco

```
import com.j256.ormlite.field.DatabaseField;
import com.j256.ormlite.table.DatabaseTable;


@DatabaseTable(tableName = "product")
public class Product {

    @DatabaseField(columnName = "id", generatedId = true)
    private int id;

    @DatabaseField(columnName = "name")
    private String name;

    @DatabaseField(columnName = "description")
    private String description;

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getDescription() {
        return description;
    }

    public void setDescription(String description) {
        this.description = description;
    }

    @Override
    public String toString() {
        return "Product{" +
                "id=" + id +
                ", name='" + name + '\'' +
                ", description='" + description + '\'' +
                '}';
    }
}
```

Gerenciando o Banco de Dados

```
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.util.Log;
import com.j256.ormlite.android.apptools.OrmLiteSqliteOpenHelper;
import com.j256.ormlite.dao.Dao;
import com.j256.ormlite.support.ConnectionSource;
import com.j256.ormlite.table.TableUtils;
import java.sql.SQLException;
import static android.content.ContentValues.TAG;

public class DatabaseHelper extends OrmLiteSqliteOpenHelper {

    public static final String DATABASE_NAME = "market.db";
    private static final int DATABASE_VERSION = 1;
    private static DatabaseHelper mInstance = null;

    public Dao<Product, Integer> productDAO = null;


    public DatabaseHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
        getWritableDatabase();
    }

    @Override
    public void onCreate(SQLiteDatabase db, ConnectionSource connec-
tionSource) {
        try {
            TableUtils.createTable(connectionSource, Product.class);
        } catch (SQLException e) {
            Log.e(TAG, "Error ao criar database", e);
        }
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, ConnectionSource connec-
tionSource, int oldVersion, int newVersion) {
        try {
            TableUtils.dropTable(connectionSource, Product.class,
true);
            onCreate(db, connectionSource);
        } catch (SQLException e) {
            Log.e(TAG, "Erro ao dropar tabelas", e);
        }
    }


    public static DatabaseHelper getInstance(Context context) {
        if (mInstance == null) {
            mInstance = new DatabaseHelper(context.getApplicationCon-
text());
        }
        return mInstance;
    }

    @Override
    public  void close() {
        super.close();
    }

    public Dao<Product, Integer> getProductDAO() throws SQLException {
        if (productDAO == null) {
            productDAO = getDao(Product.class);
        }
        return productDAO;
    }
}
```

Padrão DAO, Camada de Operações com o Banco

```
public class ProductDAO {

    private DatabaseHelper databaseHelper;
    private Dao<Product, Integer> productDAO;

    public ProductDAO(Context context) {
        try {
            databaseHelper = DatabaseHelper.getInstance(context);
            productDAO = databaseHelper.getProductDAO();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void closeDatabaseHelper() {
        databaseHelper.close();
    }

    public void create(Product product) {
        try {
            productDAO.create(product);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void update(Product product) {
        try {
            productDAO.update(product);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void delete(Product product) {
        try {
            productDAO.delete(product);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public Product readById(Integer idProduct) {
        try {
            return productDAO.queryForId(idProduct);
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return null;
    }

    public List<Product> listAll() {
        try {
            return productDAO.queryForAll();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

Testando Operações contra o Banco


```
private void exampleDB(){
    ProductDAO productDAO =  new ProductDAO(this);

    Product product = new Product();
    product.setName("Product 01");
    product.setDescription("Produto de Limpeza");

    productDAO.create(product);

    Log.e("INFO:", "Created!");
    Log.e("Product:", "Created: " +
productDAO.readById(product.getId()).getName() );

    product.setName("Produto 001");
    productDAO.update(product);
    Log.e("INFO:", "Edit!");
    Log.e("Product:", "Editado: " +
productDAO.readById(product.getId()).getName() );

    productDAO.delete(product);
    Log.e("INFO:", "Delete!");
    for (Product prod : productDAO.listAll()) {
        Log.e("Product:", "Deleted: " + prod.toString() );
    }
}
```

1° Day The End