**Aplicações Móveis**

Lab-1

**Coordenação de Engenharia Informática**

Departamento de Engenharias e Tecnologias

Instituto Superior Politécnico de Tecnologias e Ciências

**Nome :** Marcelo Rocha - 20210032

# Respostas Lab 1

**Descreva a função dos seguintes ficheiros :**

**app/src/main/res/layout/activity\_main.xml**

**app/src/main/java/ ao.co.isptec.aplm.exercise\_1/****MainActivity.java**

**app/src/res/AndroidManifest.xml**

**app/build.gradle**

**R :** O **MainActivity.java** é o ponto de entrada principal do aplicativo Android, onde a primeira interface ou tela do aplicativo é criada. Atua como o controlador principal que interage com os elementos da interface, controla a navegação do aplicativo e gerencia o comportamento do ciclo de vida da tela inicial do app.

**O activity\_main.xml** é o layout que define a interface gráfica da atividade associada, no caso, o MainActivity.java. Ele especifica os elementos visuais que aparecem na tela, como botões, caixas de texto, imagens e outros componentes de UI.

**O AndroidManifest.xml**  é um componente essencial de qualquer projeto Android. Ele serve como uma espécie de "mapa" que informa ao sistema Android informações importantes sobre o aplicativo. Ele declara as atividades, permissões, serviços, recebedores de broadcast, configurações de componentes de navegação além de outras configurações essenciais para o funcionamento do app. Resumindo e concluindo actua como uma "declaração formal" do aplicativo Android, informando ao sistema quais componentes estão presentes (atividades, serviços, etc.), quais permissões são necessárias e informações globais importantes sobre o app, como nome, ícone e versão.

O **build.gradle** é parte crucial de projetos Android (e Java em geral) que utilizam o sistema de build Gradle. Ele define como o projeto será compilado, quais dependências serão incluídas, como o aplicativo será empacotado, entre outras configurações importantes.Existem dois tipos de arquivos **build.gradle** em um projeto Android:

1. **build.gradle do projeto (raiz)**: Configurações globais do projeto inteiro (Este arquivo contém configurações aplicáveis a todo o projeto, incluindo plugins globais e repositórios para download de dependências);
2. **build.gradle do módulo (app)**: Configurações específicas para cada módulo (como o aplicativo Android em si). Este arquivo contém as configurações específicas para o aplicativo Android, como as dependências do projeto, versão do SDK, configurações de assinatura do app, entre outras.

**Esboce a árvore da UI: desenhe uma árvore da UI que reflita o layout da actividade mostrada no lado esquerdo da Figura 2. Dica: utilize as views Button, EditText e LinearLayout.**

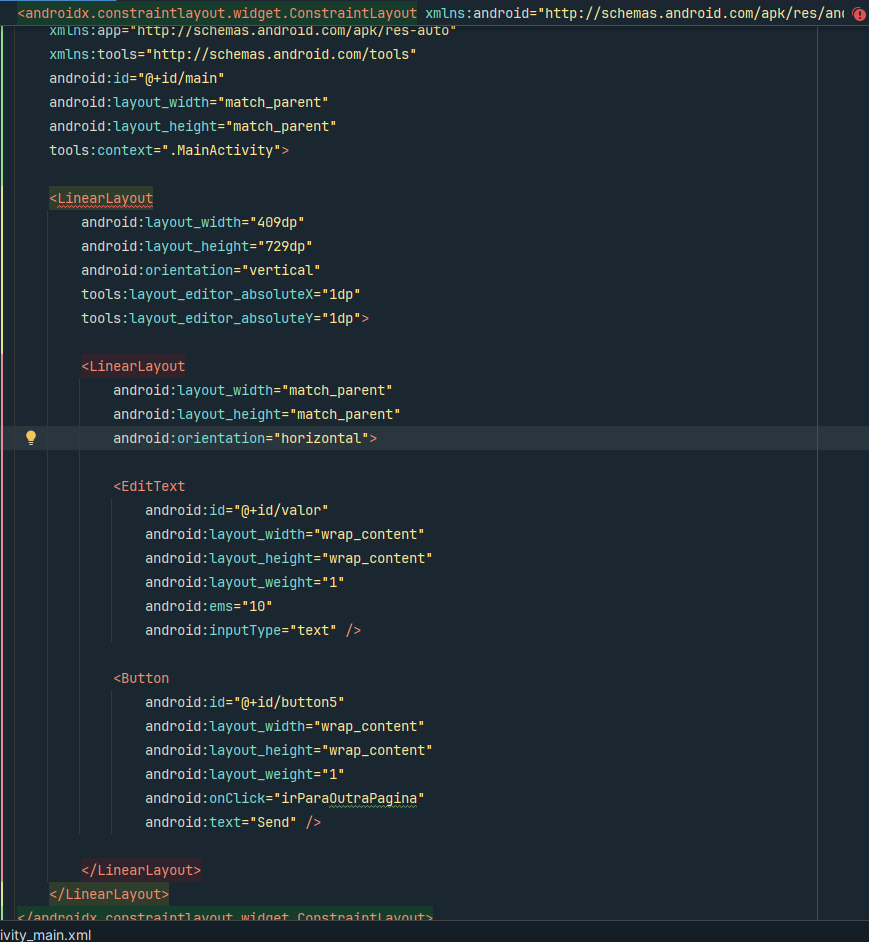
Linear Layout

Button

EditText

Linear Layout

**Escreva o código XML correspondente: Dica: use o conteúdo do arquivo activity\_main.xml (em anexo) e veja como ele reflete sua árvore de UI.**



**O quê que alterou no ficheiro AndroidManifest.xml?**

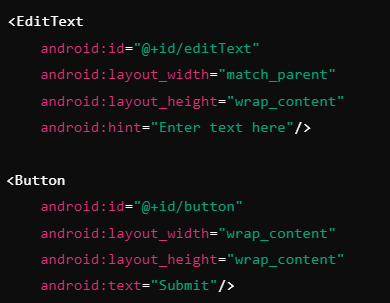
O arquivo AndroidManifest.xml foi atualizado automaticamente para registrar a nova atividade que foi criada (DisplayMessageActivity) . O arquivo AndroidManifest.xml é essencial, pois informa ao sistema Android sobre as atividades, serviços e outras componentes da aplicação.

**Como o evento clique é tratado?**

O evento de clique é tratado de forma a pegar o texto digitado no EditText criado e após isso criar-se uma intent com o objectivo de iniciar a actividade DisplayMessageActivity , sem esquecer de passar o valor do EditText através de “putExtra()” e só assim chamar “startActivity()”.

**Como identificar e obter ou definir componentes de interface do código Java?**

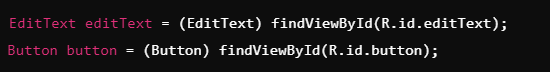
No arquivo de layout XML, cada componente pode ter um ID único, definido pelo atributo android:id. Por exemplo:

****

Aqui, temos um EditText com o ID editText e um Button com o ID button.

Podemos usar o método findViewById() para obter a referência a esses componentes. Para cada tipo de componente (por exemplo, EditText, Button), você precisa fazer um casting apropriado (não obrigatório).

Exemplo de como obter esses componentes:



Aqui, usamos o ID (R.id.editText e R.id.button) definido no arquivo XML para obter a referência ao componente correspondente no código.

Depois de obter a referência do componente, você pode modificar suas propriedades ou adicionar eventos. Exemplo de modificação de um texto :



**Qual é a finalidade da intent da variável criada na classe DisplayMessageActivity?**

A finalidade da Intent criada na classe DisplayMessageActivity é permitir que essa atividade receba e utilize informações enviadas de outra atividade (neste caso, o que foi escrito no editText da MainActivity). Uma Intent é uma abstração que permite a comunicação entre atividades no Android.

**Qual é o propósito da declaração apply plugin: "java" em build.gradle?**

A declaração apply plugin: "java" no arquivo build.gradle é usada para aplicar o plugin Java ao seu projeto. Este plugin é necessário para que o Gradle entenda que você está construindo um projeto Java. O propósito principal desta declaração é permitir que o Gradle gerencie as tarefas e dependências necessárias para compilar, testar e empacotar projetos Java.

**Execute gradle tasks --all e explore as tarefas definidas pelo plugin Gradle Java. Em particular, execute gradle build e explique o resultado desta tarefa.**

o Gradle realizou uma série de etapas para compilar e preparar o projeto para execução :

* Compilação do Código-Fonte: O Gradle compilou o código-fonte Java e gerou os arquivos .class correspondentes, que ficam no diretório build/classes/.
* Criação de Artefatos: O Gradle empacotou os arquivos .class e outros recursos em um arquivo JAR. O arquivo gerado foi colocado no diretório build/libs/.

**Por que o Quota.java é colocado nesse directório específico?**

O arquivo Quota.java é colocado no diretório src/main/java/ao/uan/fc/dam/quota para seguir a estrutura padrão de diretórios de projetos Java, que organiza o código em pacotes de acordo com as convenções do Maven e do Java.

**Explique o significado das dependências e repositórios de blocos no ficheiro build.gradle.**

Os repositórios referem-se a um bloco que especifica de onde o Gradle deve buscar as dependências do seu projeto.

As dependências referem-se a um bloco onde se declara as bibliotecas e frameworks que seu projeto usa.

**Em quais subprojectos este projecto está dividido? Determine como executar o Gradle para construir (i) todo o projecto e (ii) cada subprojecto individualmente.**

Este projecto está dividido em 3 subprojectos (api , common e app).

Para construir todo o projeto Gradle, incluindo todos os subprojectos utiliza-se : **gradle build.** Se for apenas para construir um subprojeto específico **gradle :<nomeSubprojecto>:build.**

**Identifique as dependências de cada subprojecto e desenhe o gráfico de dependência do projeto.**

common

app

api

**Identifique dentro da estrutura interna do projecto, quais ficheiros são relevantes para o Gradle e qual é a sua função.**

**build.gradle (Raiz do Projeto)**

Função: Este arquivo contém a configuração geral do projeto e pode incluir dependências, plugins e outras configurações de build. Pode ter um build.gradle para o projeto raiz e um ou mais para subprojetos.

**settings.gradle**

Função: Este arquivo é utilizado para incluir subprojetos no build do Gradle. É onde você define quais módulos (subprojetos) estão presentes no seu projeto, como :app, :api, etc.

**build.gradle (Módulo/app)**

Função: Este arquivo é específico para o módulo (por exemplo, app) e contém configurações específicas desse módulo, como dependências, compilação de SDK, configurações de variante e outras opções de build.

**gradle.properties**

Função: Este arquivo é usado para definir propriedades do Gradle, como configurações que podem ser reutilizadas em outros arquivos, incluindo a configuração de opções de build.

**gradlew e gradlew.bat**

Função: Esses arquivos são scripts para executar o Gradle Wrapper. O gradlew é utilizado em sistemas Unix, enquanto o gradlew.bat é para Windows. O Gradle Wrapper permite que você execute o Gradle sem precisar instalá-lo manualmente.

**local.properties**

Função: Este arquivo contém configurações locais específicas do usuário, como o caminho para o SDK do Android. Não deve ser versionado, pois é específico para cada desenvolvedor.