Atividade 2

Considere a estrutura de um projeto Android:

- Manifesto
- Código-fonte
- Recursos
- Gradle

Escreva um resumo destacando os tipos de arquivos que devem ser armazenados em cada uma das seções do projeto. Destaque a finalidade de cada tipo de arquivo e suas particularidades.

Manifesto

Entre muitas outras coisas, o arquivo de manifesto precisa declarar os itens a seguir:

- O nome do pacote do aplicativo, que normalmente corresponde ao namespace do seu código.
- Os componentes do aplicativo, que incluem todos os serviços, broadcast receivers, provedores de conteúdo e atividades.
- As permissões que o aplicativo precisa ter para acessar partes protegidas do sistema ou de outros aplicativos.
- Os recursos de hardware e software exigidos pelo aplicativo, que afetam quais dispositivos podem instalar o aplicativo a partir do Google Play.

Exemplo de manifesto:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.myapp"
    android:versionCode="1"
    android:versionName="1.0" >
    ...
</manifest>
```

O arquivo de manifesto também é onde você pode declarar quais são os tipos de recursos de hardware ou software exigidos pelo aplicativo e com quais tipos de dispositivo ele é compatível. O Google Play Store não permite que seu aplicativo seja instalado em dispositivos que não fornecem os recursos ou a versão do sistema exigidos.

Código fonte

Um agrupamento de elementos, que são instruções escritas para informar ao computador o que ele precisa fazer. Isso é o código fonte, que é decifrado tanto pelas máquinas quanto pelos humanos.

Se um programador tem o código fonte em mãos, ele consegue modificar a forma de funcionamento, acrescentar ou remover recursos e fazer diversas outras alterações.

Na janela Create New Module, o Android Studio oferece estes tipos de módulo de biblioteca:

- Biblioteca do Android: contém todos os tipos de arquivo com suporte em um projeto do Android, exceto código C++ nativo, incluindo código-fonte em Java e Kotlin, recursos e arquivos de manifesto. O build gera um arquivo ARchive do Android (AAR), que pode ser adicionado como dependência dos módulos do seu app Android.
- Biblioteca Java ou Kotlin: contém apenas arquivos de origem Kotlin ou Java. O resultado do build é um arquivo Java (JAR), que você pode adicionar como dependência de módulos do seu app Android ou outros projetos Kotlin ou Java.

Recursos

Cada página nesta seção descreve o uso, o formato e a sintaxe de um determinado tipo de recurso de app que você pode fornecer no diretório de recursos do projeto (res/).

Recursos de animação

- Definem animações predeterminadas.
- As animações de interpolação são salvas em res/anim/ e acessadas pela classe R.anim.
- As animações de frame são salvas em res/drawable/ e acessadas pela classe R.drawable.

Recurso da lista de estados de cor

- Define um recurso de cor que muda com base no estado View.
- Salvo em res/color/ e acessado pela classe R.color.

Recursos drawables

- Definem vários gráficos com bitmaps ou XML.
- Salvos em res/drawable/ e acessados pela classe R.drawable.

Recurso de layout

- Define o layout da interface do aplicativo.
- Salvo em res/layout/ e acessado pela classe R.layout.

Recurso de menu

- Define o conteúdo dos menus do aplicativo.
- Salvo em res/menu/ e acessado pela classe R.menu.

Recursos de string

- Definem strings, matrizes de strings e plurais. Além disso, incluem formatação e estilo de strings.
- Salvos em res/values/ e acessados pelas classes R.string, R.array e R.plurals.

Recurso de estilo

- Define a aparência e o formato dos elementos da interface.
- Salvo em res/values/ e acessado pela classe R.style.

Recursos de fonte

- Definem famílias de fontes e incluem fontes personalizadas em XML.
- Salvos em res/font/ e acessados pela classe R.font.

Mais tipos de recursos

- Definem outros valores primitivos, como recursos estáticos, incluindo:

Booleano

- Recurso XML que carrega um valor booleano.

Cor

- Recurso XML que carrega um valor de cor hexadecimal.

Dimensão

- Recurso XML que carrega um valor de dimensão com uma unidade de medida.

ID

- Recurso em XML que fornece um identificador único para recursos e componentes do aplicativo.

Número inteiro

- Recurso XML que carrega um valor inteiro.

Matriz de números inteiros

- Recurso XML que fornece uma matriz de números inteiros.

Matriz tipada

- Recurso XML que fornece uma TypedArray, que você pode usar para uma matriz de drawables.

Gradle

Sistema de build do Android compila os recursos e o código-fonte do app e os coloca em APKs ou Android App Bundles que podem ser testados, implantados, assinados e distribuídos.

O Android Studio usa o Gradle (link em inglês), um kit de ferramentas de build avançado, para automatizar e gerenciar o processo, permitindo que você defina configurações de build personalizadas e flexíveis. Cada configuração de build pode definir o próprio conjunto de códigos e recursos, reutilizando as partes comuns a todas as versões do app. O Plug-in do Android para Gradle trabalha com o kit de ferramentas de build para fornecer processos e configurações ajustáveis que são específicos para criar e testar apps Android.

O Gradle e o Plug-in do Android para Gradle são executados de forma independente do Android Studio. Isso significa que você pode criar seus apps Android no Android Studio, na linha de comando da sua máquina ou usando máquinas em que o Android Studio não está instalado, como servidores de integração contínua.

O arquivo settings.gradle (para Groovy) ou settings.gradle.kts (para script Kotlin) fica localizado no diretório raiz do projeto. Ele define as configurações de repositório do projeto e informa ao Gradle os módulos que vão ser incluídos ao criar seu app. Projetos com vários módulos precisam especificar cada um dos que serão incluídos no build final.

Por padrão, o arquivo fica assim na maior parte dos projetos:

```
pluginManagement {
    /**
     * The pluginManagement {repositories {...}} block configures the
     * repositories Gradle uses to search or download the Gradle plugins and
     * their transitive dependencies. Gradle pre-configures support for remote
     * repositories such as JCenter, Maven Central, and Ivy. You can also use
     * local repositories or define your own remote repositories. The code below
     * defines the Gradle Plugin Portal, Google's Maven repository,
     * and the Maven Central Repository as the repositories Gradle should use to look for its
     * dependencies.
     */

    repositories {
```

```
        gradlePluginPortal()
        google()
        mavenCentral()
    }
}
dependencyResolutionManagement {

    /**
     * The dependencyResolutionManagement {repositories {...}}
     * block is where you configure the repositories and dependencies used by
     * all modules in your project, such as libraries that you are using to
     * create your application. However, you should configure module-specific
     * dependencies in each module-level build.gradle file. For new projects,
     * Android Studio includes Google's Maven repository and the Maven Central
     * Repository by default, but it does not configure any dependencies (unless
     * you select a template that requires some).
     */

    repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS)
    repositories {
        google()
        mavenCentral()
    }
}
rootProject.name = "My Application"
include ':app'
```