UNIVERSIDADE TUIUTI DO PARANÁ

LEVI PASSOS DO PINHO

RESUMO DA ESTRUTURA DO PROJETO ANDROID

CURITIBA

2023

1 MANIFESTO

Todo projeto de aplicativo precisa ter um arquivo *AndroidManifest.xml* (com esse exato nome) na raiz do conjunto de origem do projeto. O arquivo de manifesto descreve informações essenciais sobre o aplicativo para as ferramentas de compilação do Android, para o sistema operacional Android e para o Google Play.

- O nome do pacote do aplicativo, que normalmente corresponde ao namespace do seu código. Quando o projeto é criado, as ferramentas de compilação do Android usam esse dado para determinar o local das entidades do código. Ao empacotar o aplicativo, as ferramentas de compilação substituem esse valor pelo ID do aplicativo a partir dos arquivos de compilação do Gradle. Esse código é usado como o identificador exclusivo do aplicativo no sistema e no Google Play.
- Os componentes do aplicativo, que incluem todos os serviços, broadcast receivers, provedores de conteúdo e atividades. Cada componente precisa definir propriedades básicas como o nome do Kotlin ou da classe do Java. Além disso, pode declarar recursos como quais configurações de dispositivo podem ser processadas e os filtros de intents que descrevem a forma de inicialização do componente.
- As permissões que o aplicativo precisa ter para acessar partes protegidas do sistema ou de outros aplicativos. Também declara todas as permissões que outros aplicativos precisam ter para acessar conteúdo desse aplicativo.
- Os recursos de hardware e software exigidos pelo aplicativo, que afetam quais dispositivos podem instalar o aplicativo a partir do Google Play.

As seções a seguir descrevem como algumas das características mais importantes do aplicativo são refletidas no arquivo de manifesto.

2 CÓDIGO-FONTE

Os módulos oferecem um contêiner para o código-fonte, os arquivos de recursos e as configurações do app, como o arquivo de compilação do módulo e o arquivo de manifesto do Android. Cada módulo pode ser individualmente compilado, testado e depurado.

O Android Studio usa módulos para facilitar a adição de novos dispositivos ao projeto. Basta seguir algumas etapas simples no Android Studio para criar um módulo que contém código específico de um tipo de dispositivo, como Wear OS ou Android TV. O Android Studio cria automaticamente diretórios de módulos, como diretórios de origem e recursos, e um arquivo build.gradle padrão adequado para o tipo de dispositivo. Além disso, o Android Studio cria módulos de dispositivos com as configurações recomendadas de compilação, como o uso da biblioteca Leanback para módulos do Android TV.

Dentro da pasta Java serão encontrados os arquivos de código, podendo estar na extensão do Kotlin ou do Java. Também é possível criar novos arquivos de código-fonte

3 RECURSOS

Recursos são os arquivos complementares e o conteúdo estático usado pelo seu código, como bitmaps, definições de layout, strings da interface do usuário, instruções de animação, entre outros.

Sempre exteriorize os recursos do app, como imagens e strings do código, para que a manutenção deles possa ser feita de forma independente. Além disso, forneça recursos alternativos para configurações específicas do dispositivo agrupando-os em diretórios de recursos especialmente nomeados. Durante a execução, o Android usa o recurso adequado com base na configuração atual. Por exemplo, forneça um outro layout de IU de acordo com o tamanho da tela ou strings diferentes dependendo da configuração de idioma.

Ao exteriorizar os recursos do app, eles podem ser acessados com IDs de recurso gerados na classe R do projeto. Este documento mostra como agrupar os recursos no seu projeto Android. Ele também mostra como fornecer recursos alternativos para configurações específicas do dispositivo e, em seguida, acessá-los no código do app ou de outros arquivos XML.

- Animator: Arquivos XML que definem as animações de propriedade.
- Anim: Arquivos XML que definem as animações intermediárias. As animações de propriedade também podem ser salvas neste diretório, mas o diretório animator/ é o recomendado para que elas diferenciem os dois tipos.
- Color: Arquivos XML que definem uma lista de estado de cores. Para mais informações, consulte Recurso de lista de estados de cores.
- Drawable: Arquivos bitmap (PNG, .9.png, JPG ou GIF) ou arquivos XML que são compilados nestes subtipos de recursos drawable:
  - Arquivos Bitmap
  - Nine-patches
  - Listas de estado
  - Formas
  - Drawable de animação
  - Outros drawables
- Mipmap: São arquivos drawable para diferentes densidades do ícone na tela de início. Para saber mais sobre como gerenciar ícones de tela de início com pastas mipmap/, consulte Colocar ícones de apps em diretórios de mipmap.

- Layout: Arquivos XML que definem um layout de interface do usuário. Para mais informações, consulte Recurso de layout.
- Menu: São arquivos XML que definem os menus do app, como o menu "opções", o menu de contexto ou o submenu. Para mais informações, consulte Recurso de menu.
- Raw: São arquivos arbitrários para salvar na forma bruta.
- Values: São arquivos XML que contêm valores simples, como strings, números inteiros e cores. Enquanto os arquivos de recurso XML que estão em outros subdiretórios res/ definem um único recurso com base no nome do arquivo XML, os arquivos no diretório values/ descrevem vários. Para cada arquivo neste diretório, cada filho do elemento <resources> define um único recurso. Por exemplo: um elemento <string> cria um recurso R.string e um elemento <color> cria um recurso R.color. Confira algumas convenções de nome de arquivo para recursos que podem ser criados neste diretório:
  - arrays.xml para matrizes de recursos;
  - colors.xml para Valores de cor;
  - dimens.xml para Valores de dimensão;
  - strings.xml para Valores de string;
  - styles.xml para Estilos;

4 GRADLE

O 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. Os arquivos necessários para o build são:

- **Arquivo de configurações do Gradle**: 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.
- **Arquivo de build de nível superior**: O arquivo de nível superior build.gradle, no Groovy, ou build.gradle.kts, no script Kotlin, está localizado no diretório raiz do projeto. Ele define as dependências que se aplicam a todos os módulos do seu projeto. Por padrão, o arquivo de build de nível superior usa o bloco plugins para definir as dependências do Gradle que são comuns a todos os módulos do projeto. Além disso, ele contém o código usado para limpar o diretório do build.
- **Arquivo de build de módulo**: O arquivo build.gradle (para Groovy) ou build.gradle.kts (para script Kotlin) do módulo fica localizado em cada diretório project/module/. Esse arquivo permite definir configurações de build para o módulo específico em que ele se encontra. A definição dessas configurações permite disponibilizar opções de empacotamento personalizadas, como tipos de build e variações de produtos extras, além de substituir as configurações no manifesto do app main/ ou no arquivo build.gradle de nível superior ou no build.gradle.kts.

- **Arquivos de propriedades do Gradle**: O Gradle também contém dois arquivos de propriedades, localizados no diretório raiz do projeto, que são úteis para especificar configurações para o próprio kit de ferramentas de build do Gradle.