Android Developers

Manifesto do aplicativo

Neste documento

Todo aplicativo tem que ter um arquivo `AndroidManifest.xml` (precisamente com esse nome) no diretório raiz. O arquivo de manifesto apresenta informações essenciais sobre o aplicativo ao sistema Android, necessárias para o sistema antes que ele possa executar o código do aplicativo.

Entre outras coisas, o arquivo do manifesto serve para:

- Nomear o pacote Java para o aplicativo. O nome do pacote serve como identificador exclusivo para o aplicativo.
- Descrever os componentes do aplicativo, que abrangem atividades, serviços, receptores de transmissão e provedores de conteúdo que compõem o aplicativo. Ele também nomeia a classe que implementa cada um dos componentes e publica suas capacidades, como as mensagens `Intent` `(https://developer.android.com/reference/android/content/Intent.html?hl=pt-br)` que podem lidar. Essas declarações informam ao sistema Android os componentes e as condições em que eles podem ser inicializados.
- Determinar os processos que hospedam os componentes de aplicativo.
- Declarar as permissões que o aplicativo deve ter para acessar partes protegidas da API e interagir com outros aplicativos. Ele também declara as permissões que outros devem ter para interagir com os componentes do aplicativo.
- Listar as classes `Instrumentation` `(https://developer.android.com/reference/android/app/Instrumentation.html?hl=pt-br)` que fornecem geração de perfil e outras informações durante a execução do aplicativo. Essas declarações estão presentes no manifesto somente enquanto o aplicativo está em desenvolvimento e são removidas antes da publicação do aplicativo.
- Declarar o nível mínimo da Android API que o aplicativo exige.

- Listar as bibliotecas às quais o aplicativo deve se vincular.

> **Observação**: À medida que você prepara o seu aplicativo Android para executar em Chromebooks, há algumas limitações de recursos importantes de hardware e software a considerar. Consulte o documento Compatibilidade do manifesto do aplicativo para Chromebooks (https://developer.android.com/topic/arc/manifest.html?hl=pt-br) para obter mais informações.

Estrutura do arquivo do manifesto

O snippet de código abaixo ilustra a estrutura geral do arquivo de manifesto e cada elemento que ele pode conter. Cada elemento e seus atributos estão documentados completamente em um arquivo separado.

> **Dica**: Para ver informações detalhadas de qualquer elemento mencionado no texto deste documento, basta clicar no nome do elemento.

Eis um exemplo do arquivo de manifesto:

```
<?xml version="1.0" encoding="utf-8"?>

<manifest> (https://developer.android.com/guide/topics/manifest/manifest-element.html?hl=pt-br)

    <uses-permission /> (https://developer.android.com/guide/topics/manifest/uses-permission-element.html?h
    <permission /> (https://developer.android.com/guide/topics/manifest/permission-element.html?hl=pt-br)
    <permission-tree /> (https://developer.android.com/guide/topics/manifest/permission-tree-element.html?h
    <permission-group /> (https://developer.android.com/guide/topics/manifest/permission-group-element.html
    <instrumentation /> (https://developer.android.com/guide/topics/manifest/instrumentation-element.html?h
    <uses-sdk /> (https://developer.android.com/guide/topics/manifest/uses-sdk-element.html?hl=pt-br)
    <uses-configuration /> (https://developer.android.com/guide/topics/manifest/uses-configuration-element
    <uses-feature /> (https://developer.android.com/guide/topics/manifest/uses-feature-element.html?hl=pt-br
    <supports-screens /> (https://developer.android.com/guide/topics/manifest/supports-screens-element.html
    <compatible-screens /> (https://developer.android.com/guide/topics/manifest/compatible-screens-element
    <supports-gl-texture /> (https://developer.android.com/guide/topics/manifest/supports-gl-texture-eleme

    <application> (https://developer.android.com/guide/topics/manifest/application-element.html?hl=pt-br)

        <activity> (https://developer.android.com/guide/topics/manifest/activity-element.html?hl=pt-br)
            <intent-filter> (https://developer.android.com/guide/topics/manifest/intent-filter-element.htm
                <action /> (https://developer.android.com/guide/topics/manifest/action-element.html?hl=pt-
                <category /> (https://developer.android.com/guide/topics/manifest/category-element.html?hl
                <data /> (https://developer.android.com/guide/topics/manifest/data-element.html?hl=pt-br)
            </intent-filter> (https://developer.android.com/guide/topics/manifest/intent-filter-element.ht
            <meta-data /> (https://developer.android.com/guide/topics/manifest/meta-data-element.html?hl=pt
        </activity> (https://developer.android.com/guide/topics/manifest/activity-element.html?hl=pt-br)

        <activity-alias> (https://developer.android.com/guide/topics/manifest/activity-alias-element.html?h
            <intent-filter> (https://developer.android.com/guide/topics/manifest/intent-filter-element.htm
            <meta-data /> (https://developer.android.com/guide/topics/manifest/meta-data-element.html?hl=pt
        </activity-alias> (https://developer.android.com/guide/topics/manifest/activity-alias-element.html

        <service> (https://developer.android.com/guide/topics/manifest/service-element.html?hl=pt-br)
            <intent-filter> (https://developer.android.com/guide/topics/manifest/intent-filter-element.htm
```

```
            <meta-data/> (https://developer.android.com/guide/topics/manifest/meta-data-element.html?hl=pt-
        </service> (https://developer.android.com/guide/topics/manifest/service-element.html?hl=pt-br)

        <receiver> (https://developer.android.com/guide/topics/manifest/receiver-element.html?hl=pt-br)
            <intent-filter> (https://developer.android.com/guide/topics/manifest/intent-filter-element.htm
            <meta-data /> (https://developer.android.com/guide/topics/manifest/meta-data-element.html?hl=pt
        </receiver> (https://developer.android.com/guide/topics/manifest/receiver-element.html?hl=pt-br)

        <provider> (https://developer.android.com/guide/topics/manifest/provider-element.html?hl=pt-br)
            <grant-uri-permission /> (https://developer.android.com/guide/topics/manifest/grant-uri-perm
            <meta-data /> (https://developer.android.com/guide/topics/manifest/meta-data-element.html?hl=pt
            <path-permission /> (https://developer.android.com/guide/topics/manifest/path-permission-elem
        </provider> (https://developer.android.com/guide/topics/manifest/provider-element.html?hl=pt-br)

        <uses-library /> (https://developer.android.com/guide/topics/manifest/uses-library-element.html?hl=

    </application> (https://developer.android.com/guide/topics/manifest/application-element.html?hl=pt-br)

</manifest> (https://developer.android.com/guide/topics/manifest/manifest-element.html?hl=pt-br)
```

A lista a seguir contém todos os elementos que podem aparecer no arquivo de manifesto em ordem alfabética:

- <action> (https://developer.android.com/guide/topics/manifest/action-element.html?hl=pt-br)
- <activity> (https://developer.android.com/guide/topics/manifest/activity-element.html?hl=pt-br)
- <activity-alias> (https://developer.android.com/guide/topics/manifest/activity-alias-element.html?hl=pt-br)
- <application> (https://developer.android.com/guide/topics/manifest/application-element.html?hl=pt-br)
- <category> (https://developer.android.com/guide/topics/manifest/category-element.html?hl=pt-br)
- <data> (https://developer.android.com/guide/topics/manifest/data-element.html?hl=pt-br)
- <grant-uri-permission> (https://developer.android.com/guide/topics/manifest/grant-uri-permission-element.html?hl=pt-br)
- <instrumentation> (https://developer.android.com/guide/topics/manifest/instrumentation-element.html?hl=pt-br)
- <intent-filter> (https://developer.android.com/guide/topics/manifest/intent-filter-element.html?hl=pt-br)
- <manifest> (https://developer.android.com/guide/topics/manifest/manifest-element.html?hl=pt-br)
- <meta-data> (https://developer.android.com/guide/topics/manifest/meta-data-element.html?hl=pt-br)
- <permission> (https://developer.android.com/guide/topics/manifest/permission-element.html?hl=pt-br)
- <permission-group> (https://developer.android.com/guide/topics/manifest/permission-group-element.html?hl=pt-br)
- <permission-tree> (https://developer.android.com/guide/topics/manifest/permission-tree-element.html?hl=pt-br)
- <provider> (https://developer.android.com/guide/topics/manifest/provider-element.html?hl=pt-br)

- <receiver> (https://developer.android.com/guide/topics/manifest/receiver-element.html?hl=pt-br)
- <service> (https://developer.android.com/guide/topics/manifest/service-element.html?hl=pt-br)
- <supports-screens> (https://developer.android.com/guide/topics/manifest/supports-screens-element.html?hl=pt-br)
- <uses-configuration> (https://developer.android.com/guide/topics/manifest/uses-configuration-element.html?hl=pt-br)
- <uses-feature> (https://developer.android.com/guide/topics/manifest/uses-feature-element.html?hl=pt-br)
- <uses-library> (https://developer.android.com/guide/topics/manifest/uses-library-element.html?hl=pt-br)
- <uses-permission> (https://developer.android.com/guide/topics/manifest/uses-permission-element.html?hl=pt-br)
- <uses-sdk> (https://developer.android.com/guide/topics/manifest/uses-sdk-element.html?hl=pt-br)

> **Observação**: Estes são os únicos elementos legais – não é possível adicionar elementos ou atributos próprios.

Convenções de arquivo

Esta seção descreve as convenções e as regras que se aplicam geralmente a todos elementos e atributos no arquivo de manifesto.

Elementos

Só são necessários os elementos <manifest> (https://developer.android.com/guide/topics/manifest/manifest-element.html?hl=pt-br) e <application> (https://developer.android.com/guide/topics/manifest/application-element.html?hl=pt-br). Eles devem estar presentes e ocorrer somente uma vez. A maioria dos outros elementos podem ocorrer várias vezes ou nenhuma. No entanto, ao menos alguns deles devem estar presentes antes de o arquivo de manifesto se tornar útil.

Se um elemento contiver qualquer coisa, ele conterá outros elementos. Todos os valores são definidos por meio de atributos, e não como dados de caracteres dentro de um elemento.

Elementos de mesmo nível geralmente não são ordenados. Por exemplo: os elementos <activity> (https://developer.android.com/guide/topics/manifest/activity-element.html?hl=pt-br), <provider> (https://developer.android.com/guide/topics/manifest/provider-element.html?hl=pt-br) e <service> (https://developer.android.com/guide/topics/manifest/service-element.html?hl=pt-br) podem se combinar em qualquer sequência. Há duas exceções principais a essa regra:

- O elemento <activity-alias> (https://developer.android.com/guide/topics/manifest/activity-alias-element.html?hl=pt-br) deve seguir o <activity> (https://developer.android.com/guide/topics/manifest/activity-element.html?hl=pt-br) para o qual ele é um alias.

- O elemento `<application>` (https://developer.android.com/guide/topics/manifest/application-element.html?hl=pt-br) deve ser o último elemento dentro do elemento `<manifest>` (https://developer.android.com/guide/topics/manifest/manifest-element.html?hl=pt-br). Em outras palavras, a tag de fechamento `</application>` deve aparecer imediatamente antes da tag de fechamento `</manifest>`.

Atributos

Formalmente, todos os atributos são opcionais. No entanto, é preciso especificar alguns atributos para que um elemento realize o seu propósito. Use a documentação como guia. Para atributos verdadeiramente opcionais, ele menciona um valor padrão ou declara o que acontece na ausência de uma especificação.

Exceto por alguns atributos do elemento `<manifest>` (https://developer.android.com/guide/topics/manifest/manifest-element.html?hl=pt-br), todos os nomes de atributo começam com o prefixo `android:`. Por exemplo: `android:alwaysRetainTaskState`. Como o prefixo é universal, a documentação geralmente o omite ao referir-se a atributos pelo nome.

Declaração de nomes de classe

Vários elementos correspondem a objetos Java, inclusive elementos para o próprio aplicativo (o elemento `<application>` (https://developer.android.com/guide/topics/manifest/application-element.html?hl=pt-br)) e seus principais componentes: atividades (`<activity>` (https://developer.android.com/guide/topics/manifest/activity-element.html?hl=pt-br)), serviços (`<service>` (https://developer.android.com/guide/topics/manifest/service-element.html?hl=pt-br)), receptores de transmissão (`<receiver>` (https://developer.android.com/guide/topics/manifest/receiver-element.html?hl=pt-br)) e provedores de conteúdo (`<provider>` (https://developer.android.com/guide/topics/manifest/provider-element.html?hl=pt-br)).

Se você definir uma subclasse, como quase sempre acontece para classes de componentes (`Activity` (https://developer.android.com/reference/android/app/Activity.html?hl=pt-br), `Service` (https://developer.android.com/reference/android/app/Service.html?hl=pt-br), `BroadcastReceiver` (https://developer.android.com/reference/android/content/BroadcastReceiver.html?hl=pt-br) e `ContentProvider` (https://developer.android.com/reference/android/content/ContentProvider.html?hl=pt-br)), ela será declarada por meio de um atributo `name`. O nome deve conter toda a designação do pacote. Por exemplo, pode-se declarar uma subclasse `Service` (https://developer.android.com/reference/android/app/Service.html?hl=pt-br) assim:

```
<manifest . . . >
    <application . . . >
        <service android:name="com.example.project.SecretService" . . . >
            . . .
        </service>
        . . .
    </application>
</manifest>
```

No entanto, se o primeiro caractere da string for um ponto, a string será acrescentada ao nome do pacote do aplicativo (conforme especificado pelo atributo `package` (https://developer.android.com/guide/topics/manifest/manifest-element.html?hl=pt-br#package) do elemento `<manifest>` (https://developer.android.com/guide/topics/manifest/manifest-element.html?hl=pt-br)). A seguinte atribuição é a mesma mostrada acima:

```
<manifest package="com.example.project" . . . >
    <application . . . >
        <service android:name=".SecretService" . . . >
            . . .
        </service>
        . . .

    </application>
</manifest>
```

Ao iniciar um componente, o sistema Android cria uma instância da subclasse nomeada. Se nenhuma subclasse for especificada, ele criará uma instância da classe base.

Vários valores

Se for especificado mais de um valor, o elemento sempre será repetido em vez de listar os vários valores dentro de um único elemento. Por exemplo, um filtro de intent pode listar algumas ações:

```
<intent-filter . . . >
    <action android:name="android.intent.action.EDIT" />
    <action android:name="android.intent.action.INSERT" />
    <action android:name="android.intent.action.DELETE" />
    . . .
</intent-filter>
```

Valores de recurso

Alguns atributos têm valores que é possível exibir aos usuários, como um rótulo e um ícone de uma atividade. Os valores desses atributos devem ser localizados e definidos a partir de um recurso ou tema. Os valores de recurso são expressos no formato a seguir:

```
@[package:]type/name
```

É possível omitir o nome do *pacote* se o recurso estiver no mesmo pacote que o aplicativo. O *type* é um tipo de recurso, como uma *string* ou um *drawable* (desenhável), e *name* é o nome que identifica o recurso específico. Vejamos um exemplo:

```
<activity android:icon="@drawable/smallPic" . . . >
```

Os valores de um tema são expressados similarmente, mas com um `?` inicial em vez de `@`:

```
?[package:]type/name
```

Valores de string

> Onde um valor de atributo é uma string, você deve usar barras invertidas duplas (`\\`) para caracteres de escape, como `\\n` para uma nova linha ou `\\uxxxx` para um caractere Unicode.

Recursos de arquivo

As seções a seguir descrevem como alguns recursos do Android se refletem no arquivo de manifesto.

Filtros de intent

Os componentes fundamentais de um aplicativo, como atividades, serviços e receptores de transmissão, são ativados por *intents*. Intents são pacotes de informações (objetos `Intent` (https://developer.android.com/reference/android/content/Intent.html?hl=pt-br)) que descrevem uma ação desejada, inclusive dados usados em ações, a categoria do componente que deve executar a ação e outras instruções pertinentes. O sistema Android localiza um componente adequado para responder à intent, inicia uma nova instância do componente se necessário e passa-o ao objeto `Intent` (https://developer.android.com/reference/android/content/Intent.html?hl=pt-br).

Os componentes aconselham os tipos de intents aos quais podem responder pelos *filtro de intent*. Como o sistema Android precisa saber que intents um componente pode processar antes de iniciá-lo, os filtros de intent são especificados no manifesto como elementos `<intent-filter>` (https://developer.android.com/guide/topics/manifest/intent-filter-element.html?hl=pt-br). Os componentes podem ter qualquer quantidade de filtros, em que cada um descreve um recurso diferente.

A intent que nomeia explicitamente um componente alvo ativará aquele componente para que o filtro não assuma nenhuma função. Uma intent que não especifica nenhum alvo pelo nome pode ativar um componente somente se ele puder passar por um dos filtros do componente.

Para obter informações sobre como objetos `Intent` (https://developer.android.com/reference/android/content/Intent.html?hl=pt-br) são testados contra filtros de intent, confira o documento Intents e filtros de intent (https://developer.android.com/guide/components/intents-filters.html?hl=pt-br).

Ícones e rótulos

Alguns elementos têm atributos `icon` e `label` de um pequeno ícone e um rótulo de texto que pode ficar visível para os usuários. Alguns deles também têm um atributo `description` para um texto explicativo mais longo que também pode ser exibido na tela. Por exemplo: o elemento `<permission>` (https://developer.android.com/guide/topics/manifest/permission-element.html?hl=pt-br) tem todos os três atributos; assim, quando o usuário é consultado para dar permissão a um aplicativo que a solicitou, serão apresentados ao usuário um ícone que representa a permissão, o nome da permissão e uma descrição de tudo o que está envolvido.

Em todos os casos, o ícone e o rótulo definidos em um elemento recipiente se tornam as configurações `icon` e `label` padrão de todos os subelementos do contêiner. Assim, o ícone e o rótulo definidos no elemento `<application>` (https://developer.android.com/guide/topics/manifest/application-element.html?hl=pt-br) são o

ícone e o rótulo padrão para cada um dos componentes do aplicativo. Da mesma forma, o ícone e o rótulo definidos para um componente, como um elemento `<activity>` (https://developer.android.com/guide/topics/manifest/activity-element.html?hl=pt-br), são as configurações padrão para cada um dos elementos `<intent-filter>` (https://developer.android.com/guide/topics/manifest/intent-filter-element.html?hl=pt-br) do componente. Se um elemento `<application>` (https://developer.android.com/guide/topics/manifest/application-element.html?hl=pt-br) define um rótulo, mas uma atividade e seu filtro de intent não definem, o rótulo do aplicativo é tratado como o rótulo da atividade e do filtro de intent.

O ícone e o rótulo definidos para um filtro de intent são usados para representar um componente apresentado para o usuário para preencher a função anunciada pelo filtro. Por exemplo: um filtro com as configurações `android.intent.action.MAIN` e `android.intent.category.LAUNCHER` anuncia uma atividade como uma que inicia um aplicativo. Ou seja, que deve ser exibida no inicializador do aplicativo. O ícone e o rótulo definidos no filtro são exibidos no inicializador.

Permissões

Permissão é uma restrição que limita o acesso a parte do código ou aos dados de um dispositivo. A limitação é imposta para proteger dados essenciais que podem sofrer mau uso e distorções ou prejudicar a experiência do usuário.

Cada permissão é identificada por um rótulo exclusivo. Geralmente o rótulo indica a ação que foi restringida. Eis algumas permissões que são definidas pelo Android:

- `android.permission.CALL_EMERGENCY_NUMBERS`
- `android.permission.READ_OWNER_DATA`
- `android.permission.SET_WALLPAPER`
- `android.permission.DEVICE_POWER`

Um recurso pode ser protegido somente por uma permissão.

Se um aplicativo precisar de acesso a um recurso protegido por uma permissão, ele deverá declarar que precisa da permissão com um elemento `<uses-permission>` (https://developer.android.com/guide/topics/manifest/uses-permission-element.html?hl=pt-br) no manifesto. Assim, quando o aplicativo é instalado no dispositivo, o instalador determina se concederá ou não a permissão solicitada, marcando as autoridades que assinaram os certificados do aplicativo e, em alguns casos, perguntando ao usuário. Se a permissão for concedida, o aplicativo será capaz de usar os recursos protegidos. Caso contrário, sua tentativa de acessar esses recursos falhará sem nenhuma notificação ao usuário.

Um aplicativo também pode proteger os próprios componentes com permissões. Ele pode empregar qualquer uma das permissões definidas pelo Android, conforme listadas em `android.Manifest.permission` (https://developer.android.com/reference/android/Manifest.permission.html?hl=pt-br), ou declaradas por outros aplicativos. Ele também pode definir uma permissão própria. As novas permissões são declaradas com o elemento `<permission>` (https://developer.android.com/guide/topics/manifest/permission-element.html?hl=pt-br). Por exemplo, uma atividade pode ser protegida da seguinte forma:

```
<manifest . . . >
    <permission android:name="com.example.project.DEBIT_ACCT" . . . />
    <uses-permission android:name="com.example.project.DEBIT_ACCT" />
    . . .
    <application . . .>

        <activity android:name="com.example.project.FreneticActivity"
                  android:permission="com.example.project.DEBIT_ACCT"
                  . . . >
            . . .
        </activity>
    </application>
</manifest>
```

Observe que, nesse exemplo, a permissão `DEBIT_ACCT`, além de declarada com o elemento `<permission>` (https://developer.android.com/guide/topics/manifest/permission-element.html?hl=pt-br), tem seu uso solicitado com o elemento `<uses-permission>` (https://developer.android.com/guide/topics/manifest/uses-permission-element.html?hl=pt-br). É preciso solicitá-la para que outros componentes do aplicativo iniciem a atividade protegida, mesmo que a proteção seja imposta pelo próprio aplicativo.

Se, no mesmo exemplo mostrado acima, o atributo `permission` fosse definido como uma permissão declarada em outro lugar, como `android.permission.CALL_EMERGENCY_NUMBERS`, não seria necessário declará-la novamente com um elemento `<permission>` (https://developer.android.com/guide/topics/manifest/permission-element.html?hl=pt-br). No entanto, ainda seria necessário solicitar seu uso com `<uses-permission>` (https://developer.android.com/guide/topics/manifest/uses-permission-element.html?hl=pt-br).

O elemento `<permission-tree>` (https://developer.android.com/guide/topics/manifest/permission-tree-element.html?hl=pt-br) declara um namespace para um grupo de permissões que é definido no código, e `<permission-group>` (https://developer.android.com/guide/topics/manifest/permission-group-element.html?hl=pt-br) define um rótulo para um conjunto de permissões, ambos declarados no manifesto com os elementos `<permission>` (https://developer.android.com/guide/topics/manifest/permission-element.html?hl=pt-br) e aqueles declarados em outros lugares. Isto afeta somente a forma com que as permissões estão agrupadas quando apresentadas ao usuário. O elemento `<permission-group>` (https://developer.android.com/guide/topics/manifest/permission-group-element.html?hl=pt-br) não especifica as permissões que pertencem ao grupo, mas fornece um nome ao grupo. É possível incluir uma permissão no grupo atribuindo o nome do grupo ao atributo `permissionGroup` (https://developer.android.com/guide/topics/manifest/permission-element.html?hl=pt-br#pgroup) do elemento `<permission>` (https://developer.android.com/guide/topics/manifest/permission-element.html?hl=pt-br).

Bibliotecas

Todo aplicativo está vinculado à biblioteca Android padrão, que contém os pacotes básicos para programar aplicativos (com classes comuns como Activity, Service, Intent, View, Button, Application e ContentProvider).

No entanto, alguns pacotes residem em bibliotecas próprias. Se o aplicativo usar código de algum desses pacotes, ele deverá receber solicitação explícita para ser vinculado a eles. O manifesto deve conter um elemento `<uses-library>` (https://developer.android.com/guide/topics/manifest/uses-library-element.html?hl=pt-br) separado para nomear cada uma das bibliotecas. É possível encontrar o nome da biblioteca na documentação do pacote.