Sumário

1 Atalhos de Lançadores de Aplicativos

A partir da versão do Android 7.1 - API 25, o desenvolvedor pode especificar atalhos com ações ou intenções específicas que aparecerá com o evento de touch *long-press* do usuário no lançador do aplicativo (que se encontra na home do dispositivo).

Neste tutorial, você aprenderá a criar esse atalho.

Há 3 maneiras de fazer isso: - A primeira é de forma estática com XML. Isto é, criar tarefas fixas para serem executadas. - A segunda é de forma dinâmica, em tempo de execução (*runtime*). - A terceira, pouco usada, é através de atalhos fixados caso o usuário dê permissões para isso.

Desta forma, além de deixar o seu aplicativo mais atrativo, essa é uma ótima oportunidade para aumentar o engajamento dos usuários com o app e deixá-lo executar tarefas rápidas sem a necessidade de abrir o próprio aplicativo. Por exemplo:

- Enviar uma nova mensagem
- Criar uma reserva
- Tocar um video
- Enviar um e-mail
- Checkpoint em jogos
- E muito mais

1.1 Perguntas Frequentes

1.1.1 Quais Activities pode ter atalhos?

Somente a Activity principal do seu aplicativo (aquela com `Intent.ACTION_MAIN` e `Intent.CATEGORY_LAUNCHER`) podem ter atalhos.

1.1.2 Quantos atalhos posso criar?

É permitido ter somente 5 atalhos por aplicativo (combinando atalhos dinâmicos e estáticos). Muitos lançadores podem exibir somente os primeiros 4.

Contudo, não há limite para o número de atalhos fixados no seu aplicativo que os usuários podem criar. Mesmo que o aplicativo não possa removê-los, o usuário ainda pode desativá-los.

1.1.3 Outros apps podem acessar os atalhos?

Não. **Outros aplicativos não podem acessar os metadados nos seus atalhos, mas o próprio launcher pode. Dito isto, oculte informações confidenciais do usuários nesse cado dos metadados.**

1.2 Criando Atalhos

Como dito anteriormente, os atalhos ajudam seus usuários acessar partes de conteúdos de forma mais rápida.

Essa entrega de conteúdo, depende do caso de uso do aplicativo. Ou seja, se for o usuário que decide o que o aplicativo vai entregar/devolver de conteúdo o mais indicado são atalhos dinâmicos ou fixados definidos por ele.

Segue alguns exemplos:

- Atalhos estáticos: Indicado quando o aplicativo possui uma estrutura já definida de entrega de conteúdo onde não depende da vida de interação do usuário como abrir um calendário ou email. Usar atalhos estáticos nesses casos, garante um tipo de "rotina" que o usuário faz.
- Atalhos dinâmicos: Neste caso, o atalho depende de uma interação do usuário e muda seu estado como em um jogo, onde há um atalho que permite o jogador começar do último checkpoint e cada vez que o mesmo passa de nível, o atalho dinâmico é atualizado.
- Atalhos fixados: Indicado para ações específicas orientadas pelo usuário como fixar um website. Isso facilita o usuário navegar no website por exemplo.

1.2.1 Criando atalhos estáticos

Para criar uma atalho estático você precisará:

1. Encontre no seu arquivo de manifesto `AndroidManifest.xml` a atividade responsável com as intenções `android.intent.action.MAIN` e `android.intent.category.LAUNCHER`.

2. Adicione o metadado `<meta-data>` que faz a referência ao arquivo de recurso onde o atalho está definido.

```
<meta-data android:name="android.app.shortcuts"
           android:resource="@xml/shortcuts" />
```

3. Crie um novo arquivo de recurso: `res/xml/shortcuts.xml`.

4. Agora o seu novo arquivo `shorcuts.xml` deve ter alguns atalhos semelhante a criação de menus (caso você já tenha feito uma vez). Cada atalho pode ter ícones, labels e informações das intenções que ele pretende fazer.

```
<shortcuts xmlns:android="http://schemas.android.com/apk/res/android">

    <shortcut
        android:enabled="true"
        android:icon="@drawable/ic_launcher_background"
        android:shortcutDisabledMessage="@string/disabled_message"
        android:shortcutId="compose"
        android:shortcutLongLabel="@string/long_label"
        android:shortcutShortLabel="@string/short_label">
        <intent
            android:action="android.intent.action.VIEW"
            android:targetClass="co.tiagoaguiar.appatalhos.MainActivity"
            android:targetPackage="co.tiagoaguiar.appatalhos" />
    </shortcut>

</shortcuts>
```

1.2.1.1 Campos obrigatórios:

- android:shortcutId: Id pro **ShortcutManager**

  - Nao pode ser @string, tem que ser uma string literal usada pelo **ShortcutManager**

- android:shortcutShortLabel: Exibida no Launcher
  - 10 caracteres
  - Tem que ser @string

Campos opcionais:

- android:shortcutLongLabel: Exibida no Launcher (quanto tiver espaço)
  - 25 caracteres
  - Tem que ser @string
- android:shortcutDisabledMessage: Mensagem em launcher que suporta exibir atalhos desativados pelo usuários. A mensagem explica ao usuário porque o atalho foi desativado. Não tem efeito se tiver o `android:enabled` como `true`
- android:enabled: ativado ou desativado
  - Deve especificar o android:shortcutDisabledMessage
- android:icon: Ícone

Todo `<shortchut>` deve ter um `intent`. Você pode fornecer múltiplas intenções para um mesmo atalho.

1.3 Testando o Atalho Estático

Use o long-press touch para inicializar o atalho e pegue a string literal de action passada no xml: `android:action="android.intent.action.VIEW"` (você pode definir qualquer string literal e comparar depois na execução da MainActivity com um `if`).

Ex.: Camera do WhatsApp