ADS
DR2 Des. Android

Desenvolvimento de Interfaces Android

Análise e Desenvolvimento de Software - 2020 / 4

Menus

Menus

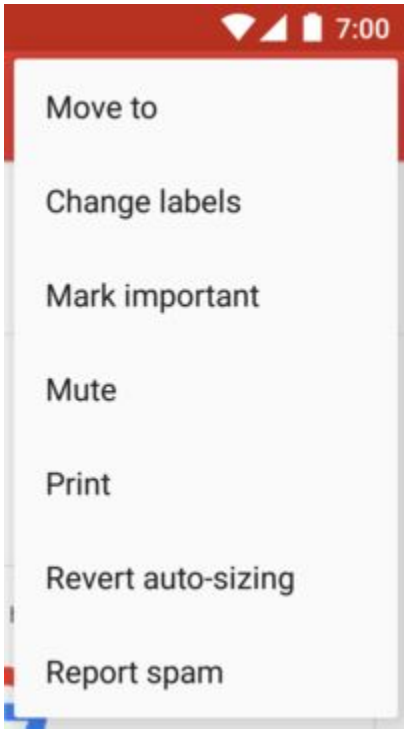
Menus são componentes comuns da interface do usuário em diversos tipos de aplicativos. Para fornecer uma experiência familiar e consistente ao usuário, você precisa usar APIs de Menu para apresentar ações de usuário e outras opções nas suas atividades.

Desde o Android 3.0 (API de nível 11), dispositivos Android não precisam mais fornecer um botão de Menu dedicado. Com essa alteração, os aplicativos Android migrarão de uma dependência do painel de menu de seis itens tradicional para fornecer uma barra de aplicativos para apresentar as ações comuns de usuário.

Menus

Apesar de o design e a experiência do usuário para alguns dos itens do menu terem passado por mudanças, a semântica para definir um conjunto de ações e opções ainda se baseia em APIs de Menu.

1. **Menu de opções e barra de aplicativos.**
2. Modo de ação contextual e menu de contexto.
3. Menu pop-up.



Menus

Para todos os tipos de menu, o Android fornece um formato XML padrão para definir os itens de menu.

Em vez de criar um menu no código da atividade, você precisa definir um menu e todos os seus itens em um recurso de menu XML.

Usar um recurso de menu é uma boa prática por alguns motivos:

- É mais fácil para visualizar a estrutura do menu em XML.
- Ele separa o conteúdo do menu do código comportamental do aplicativo.
- Ele permite criar configurações alternativas de menu para versões diferentes de plataforma, de tamanhos de tela e de outras configurações aproveitando a biblioteca de recursos do aplicativo.

Menus

Para definir o menu, crie um arquivo XML dentro do diretório res/menu/ do projeto e crie o menu com os seguintes elementos:

<menu> → Define um Menu, que é um contêiner para os itens de menu. Um elemento <menu> precisará ser o nó raiz para o arquivo e pode reter um ou mais elementos <item> e <group>.

<item> → Cria um MenuItem, que representa um único item em um menu. Esse elemento pode conter um elemento <menu> aninhado para criar um submenu.

<group> → Um contêiner invisível e opcional para os elementos <item>. Ele permite que você categorize itens de menu para que eles compartilhem propriedades como estado ativo e visibilidade. Para mais informações, consulte a seção Criação de grupos de menu.

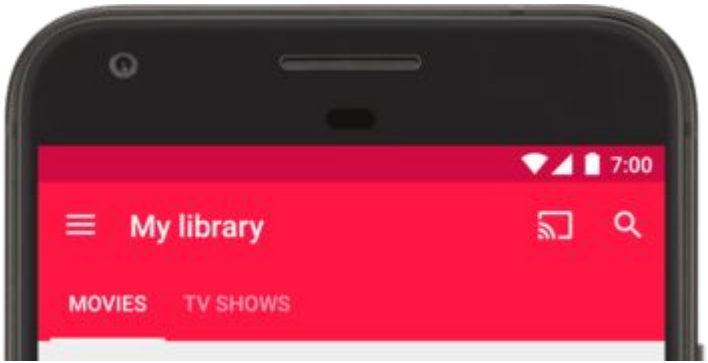
Menus

Caso tenha desenvolvido o aplicativo para Android 2.3.x (API de nível 10) ou inferior, os conteúdos do menu de opções aparecerão na parte inferior da tela, quando o usuário pressionar o botão Menu, como mostrado na figura.

Menus

Se você desenvolveu o aplicativo para Android 3.0 (API de nível 11) ou superior, os itens do menu de opções estão disponíveis na barra de aplicativos.

Por padrão, o sistema posiciona todos os itens nas ações adicionais, que o usuário pode revelar com o ícone de ações adicionais no lado direito da barra de aplicativos (ou pressionando o botão Menu no dispositivo, se disponível).



Menus

```xml
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/new_game"
          android:icon="@drawable/ic_new_game"
          android:title="@string/new_game"
          android:showAsAction="ifRoom"/>
    <item android:id="@+id/help"
          android:icon="@drawable/ic_help"
          android:title="@string/help" />
</menu>
```

```kotlin
override fun onCreateOptionsMenu(menu: Menu): Boolean {
    val inflater: MenuInflater = menuInflater
    inflater.inflate(R.menu.game_menu, menu)
    return true
}
```

Menus

```
override fun onOptionsItemSelected(item: MenuItem): Boolean {
    // Handle item selection
    return when (item.itemId) {
        R.id.new_game -> {
            newGame()
            true
        }
        R.id.help -> {
            showHelp()
            true
        }
        else -> super.onOptionsItemSelected(item)
    }
}
```

WebView

WebView

Se você quer entregar um aplicativo da Web (ou apenas uma página da Web) como parte de um aplicativo cliente, é possível fazer isso usando WebView.

A classe WebView é uma extensão da classe View do Android, que permite exibir páginas da Web como parte do layout de atividades.

Ela não inclui recursos de um navegador da Web completamente desenvolvido, como controles de navegação ou uma barra de endereço.

Por padrão, tudo o que WebView faz é mostrar uma página da Web.

WebView

Usar WebView pode ser útil quando você quer fornecer informações que talvez precisem ser atualizadas, por exemplo, como um contrato de usuário final ou um guia do usuário.

No app para Android, você pode criar uma Activity que contenha uma WebView e usá-la para exibir o documento hospedado on-line.

WebView

1

```
<WebView
    android:id="@+id/webview"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
/>
```

2

```
val myWebView: WebView = findViewById(R.id.webview)
myWebView.loadUrl("http://www.example.com")
```

3

```
<manifest ... >
    <uses-permission android:name="android.permission.INTERNET" />
    ...
</manifest>
```

WebView

Por padrão, o JavaScript fica desativado em uma WebView.

Você pode ativá-lo por meio das WebSettings anexadas à WebView.

Você pode recuperar WebSettings com getSettings() e ativar o JavaScript com setJavaScriptEnabled().

```
val myWebView: WebView = findViewById(R.id.webview)
myWebView.settings.javaScriptEnabled = true
```

WebView

Quando o usuário clica em um link de uma página da Web na sua WebView, o comportamento padrão é que o Android abra um app que gerencia URLs.

Geralmente, o navegador da Web padrão é aberto e carrega o URL de destino.

No entanto, você pode modificar esse comportamento para sua WebView e fazer com que os links sejam abertos na WebView.

Você pode permitir que o usuário volte e avance no histórico de páginas da Web mantido pela WebView.

```
val myWebView: WebView = findViewById(R.id.webview)
myWebView.webViewClient = WebViewClient()
```

Exercício

Implementar uma aplicação de teste com Menu e WebView apresentada na aula.

Implementar o menu contextual e pop-up.