Computação II – Orientação a Objetos

Fabio Mascarenhas - 2014.1

http://www.dcc.ufrj.br/~fabiom/java

Layout de Tabela

- Nossa calculadora tem um display e dezesseis botões organizados em fileiras de quatro
- Uma maneira natural de organizar esses componentes é como uma tabela de quatro colunas e cinco linhas, onde o display ocupa toda a primeira linha
- O sistema Android tem um layout perfeito para isso: `TableLayout`
- Cada linha da tabela é um componente `TableRow`
- É mais fácil editar diretamente o XML do que usar o designer

Busca no Twitter

- Vamos fazer outra aplicação simples, para exercitar outras partes do Android
- Nossa aplicação vai fazer uma busca no Twitter, usando a biblioteca Twitter4J
- Ao invés de usar as próprias classes do Twitter4J diretamente como modelo, vamos escondê-las atrás de uma classe `Tweet`
- Temos apenas uma atividade: o usuário entra uma string para buscar, e isso dá uma lista de tweets

ListView

- Para mostrar a lista dos tweets, vamos usar um componente `ListView`
- Uma `ListView` é como um `LinearLayout` vertical que pode facilmente alterado em tempo de execução
- Se o número de componentes na lista é grande demais para exibir todos, a `ListView` pode ser deslizada
- Os componentes podem ser qualquer coisa: conectamos uma `ListView` ao que ela deve exibir com um adaptador derivado de `BaseAdapter`

WebView

- Para mostrar nossos tweets, vamos usar uma `WebView`
- Uma `WebView` é como um mini-browser, e pode mostrar HTML qualquer; é assim que vamos mostrar as fotos de perfil dos donos dos tweets
- Conectamos nossa `ListView` às `WebViews` no método `getView` do adaptador
- Podemos sempre criar uma nova `WebView`, mas é mais eficiente reutilizar a já existente, trocando apenas o conteúdo

Tarefas

- Há um problema na nossa busca: uma busca pode demorar para completer
- Enquanto a busca executa, nossa aplicação está travada
- Se demorar tempo demais, o sistema Android vai perguntar pro usuário se ele quer encerra a aplicação
- Tratadores de eventos nunca devem fazer nada que não seja efetivamente instantâneo; qualquer ação mais demorada deve ser delegada para uma *tarefa* que vai executar em segundo plano
- Tarefas são subclasses de `AsyncTask`

AsyncTask<E,I,S>

- Uma tarefa é parametrizada por três tipos:
  - `E` é o tipo dos parâmetros de entrada para a tarefa
  - `I` é o tipo dos resultados intermediários, caso ela reporte progresso
  - `S` é o tipo do resultado da tarefa
- O método `S doInBackGround(E... params)` executa a tarefa
- O método `void onPostExecute(S res)` atualiza a interface com o resultado

`Toast` e `ProgressDialog`

- Uma busca pode dar errado: pode não ter resultados, ou pode ter algum problema na rede
- O `Toast` é uma maneira de a aplicação informar ao usuário que alguma coisa aconteceu
- Ele faz aparecer uma mensagem que fica na tela por alguns segundos e depois some
- Para informar pro usuário que uma busca está acontecendo, podemos usar um `ProgressDialog` dentro de nossa `AsyncTask`