

Tratando clique na lista

Transcrição

Já conseguimos inserir e remover alunos, agora, vamos adicionar um meio para editá-los. Assim, caso acrescentemos uma informação errada podemos modificá-la quando quisermos.

Poderíamos inserir um novo `MenuContext` para abrir a opção de "Editar" junto a de "Deletar". Mas, vamos aprender algo novo! Vamos iniciar o caminho clicando no nome do aluno e logo em seguida sendo direcionados para o formulário que editaremos.

Para aplicar um comportamento no clique sobre o nome do aluno, voltamos na `ListaAlunosActivity.java`. Vamos na `onCreate` onde acabamos de recuperar a lista. Poderíamos inserir um `setOnClickListener`, como já fizemos anteriormente, mas ele devolveria uma `View`, que nesse caso é a lista inteira. Como queremos os subcomponentes da lista, isto é, apenas os nomes dos alunos que clicarmos, não usaremos o `setOnClickListener`. Usaremos um `Listener` diferente, o `setOnClickListener`, que manifesta que estamos interessados apenas no item clicado e não na lista inteira. Vamos completar com `new OnItemClickListener` e ficaremos com `listaAlunos.setOnClickListener(new OnItemClickListener)`. Agora, é só dar um "Enter" e criar a classe, ela será anônima. Teremos o seguinte:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_lista_alunos);

    listaAlunos = (ListView) findViewById(R.id.lista_alunos);

    listaAlunos.setOnClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        }
    });
}
```

Vamos implementar o método `onItemClick`. Aproveitando para explicar algumas coisas:

- o `parent` é justamente a lista na qual clicamos, vamos alterar o nome para `lista`;
- a `View` é o item que clicamos, então renomeamos para `View item`;
- o `int position` é a posição do item
- o `long id` é o `id` do item

Para pegar o aluno basta usar o método que já tínhamos utilizado no `onMenuItem`, o `getItemAtPosition`. Informamos para esse método uma posição e ele retorna o aluno.

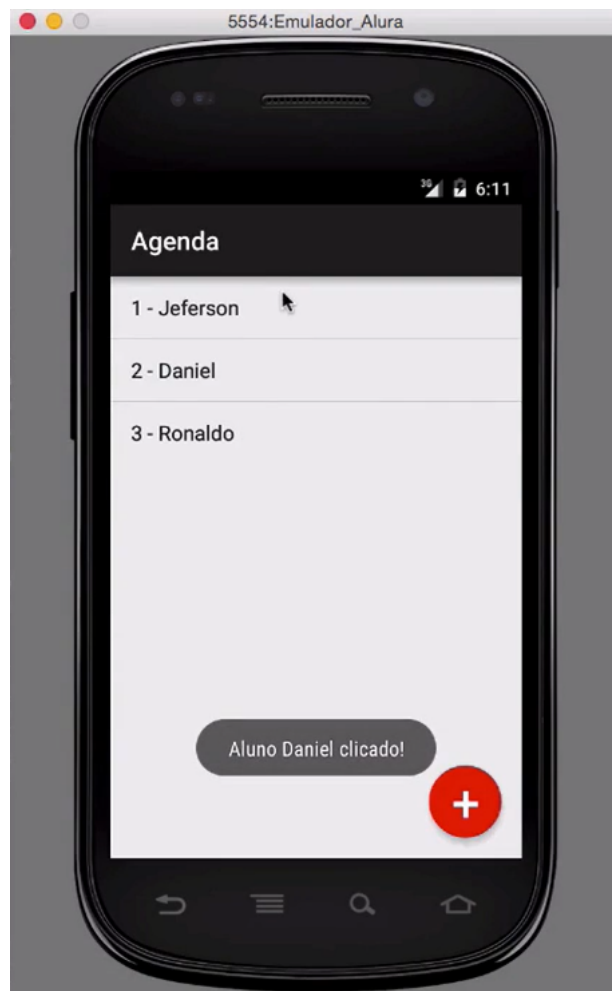
Vamos acrescentar no `setOnClickListener`, na linha de baixo do `onItemClick`, um `Aluno aluno = listaAlunos.getItemAtPosition(position)` e fazemos um `Cast`. Para imprimir o nome do aluno acrescentamos um `Toast` seguido do contexto e do nome do aluno clicado, a duração e o `show` para mostrar. Teremos:

```
Toast.makeText(ListaAlunosActivity.this, "Aluno " + aluno.getNome() + " clicado!", Toast.LENGTH_
```

Ficaremos com:

```
@Override
public void onItemClick(AdapterView<?> parent, View item, int position, long id) {
    Aluno aluno = (Aluno) listaAlunos.getItemAtPosition(position);
    Toast.makeText(ListaAlunosActivity.this, "Aluno " + aluno.getNome() + " clicado!", Toast.LEI
}
```

Vamos rodar a aplicação e verificar o que temos!



Agora o clique no aluno possui uma interação!

Observação: Podemos usar outro tipo de *Listener*, o `setOnItemLongClick`, que está relacionado ao clique longo na lista. Se quiser criar um `setOnItemLongClick` basta se guiar nos parâmetros que já fizemos. As únicas coisas que mudarão é o `public boolean` e o `return false`. Ao rodar a aplicação veremos que ele mostrará o `Toast` do clique longo e o `MenuContext`.