

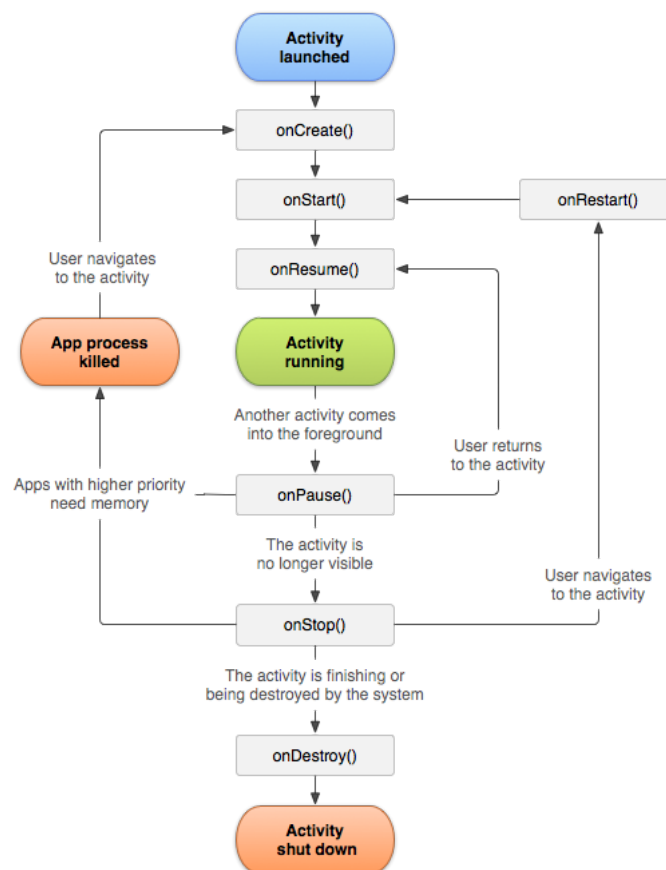
Ciclo de vida das Activities

Transcrição

Lembrando que conseguimos fazer com que um novo aluno aparecesse em nossa lista, mas ele só é adicionado quando saímos da aplicação. Vamos mudar isso!

Para fazer essa mudança temos que entender um pouco sobre o ciclo de vida das *activities*. Vamos observar o seguinte desenho disponível no site do *Android* (d.android.com). No link fazemos uma busca por *activity* e na página dos resultados clicamos em *Reference*, no menu do canto esquerdo.

Selecionamos o *Activity/Android Developers* que aparece logo no topo e vai abrir uma página na qual teremos o ciclo de vida das *activities* :



Para entender o Ciclo de vida das Activities

A *activity*, como vemos na imagem, evoca o método `onCreate`, que chama o `onStart` e o `onResume`, cujas funções são, respectivamente, iniciar a *activity* e continuá-la. Só depois de fazer isso ela começa a ser executada.

Dependendo do celular, se algum outro método entra na frente da *Activity*, ele chama o método `onPause`. Isso ocorre no caso dos "pop ups" que não tapam completamente a *activity*, ou seja, a *activity* é pausada temporariamente e apenas depois do "pop up" ser fechado é chamado o `onResume` para retornar a atividade.

Outro caso é o das chamadas telefônicas, a aplicação nos avisa o recebimento da chamada eletrônica tomando a tela inteira. Nesse caso o *Android* chama o `onPause` e como a aplicação deixa de estar visível ele também chama o `onStop`, parando a *activity*. Esse processo pode matar nossa aplicação, isto é, seguindo o caminho de "App process killed". O que

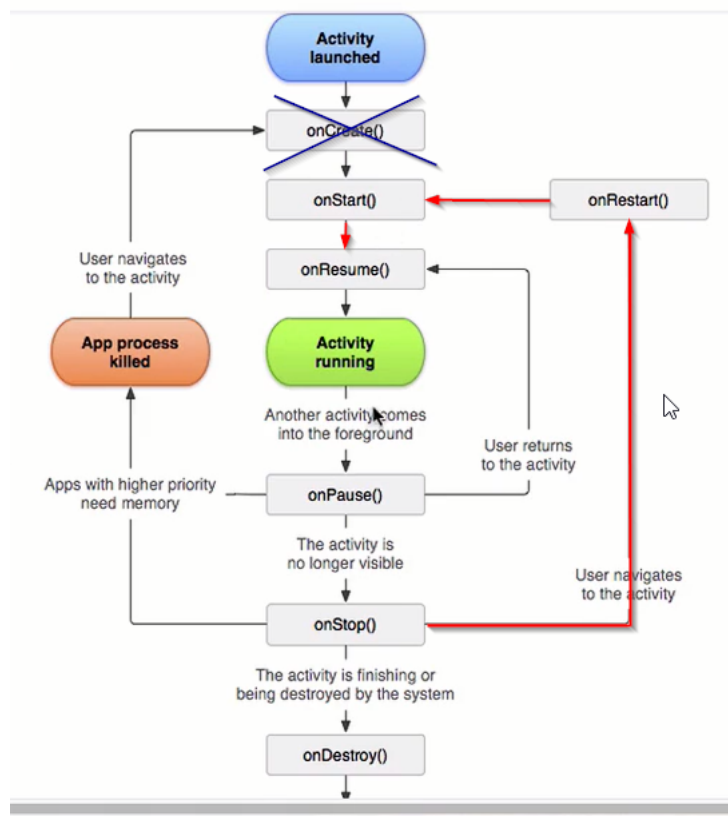
acontece é que o processo pode reiniciar sem salvar nada. Isso ocorre quando o *Android* precisa da memória, caso contrário ele retorna a *activity* através do `onRestart` e reiniciamos a aplicação a partir do `onStart`.

Agora que compreendemos um pouco a respeito da vida da *activity* vamos ver como isso se aplica ao que estamos fazendo!

Quando estamos no celular e aparece a tela da Lista de Alunos, já foram executados o `onCreate`, o `onStart` e o `onResume`. Quando passamos para o formulário estamos em uma *activity* que toma a tela inteira, assim, o *Android* chama o `onStop`. Quando apertarmos o botão de *check mark* a *activity* vai seguir o caminho da *Activity shut down*, como demonstra a imagem. Mas, quando acaba a *activity* do formulário voltamos para a lista de alunos que estará no `onStop` que seguirá o caminho `onRestart` e seguirá no `onStart` normalmente.

Vamos voltar um pouquinho no *Android Studio* e verificar onde é feito o carregamento da lista, na `ListaAlunosActivity.java`.

O carregamento é feito no método `onCreate` e aí temos o problema! Pois o ciclo da *activity* depois do `onStop` dirige-se para o `onRestart` e o `onResume` e não passa pelo `onCreate`. É por isso que não conseguimos mostrar o aluno que acrescentamos na lista.



Para resolver isso podemos acrescentar um `onResume`, pois todos os caminhos passam por ele. Se colocássemos um método `onStart` teríamos o problemas de que, caso houvesse uma pausa, não voltaríamos para o `onStart` e a lista não carregaria.

Para fazer isso, vamos na `ListaAlunosActivity.java` e escrevemos abaixo do `onCreate` um `onResume`, damos um "Enter". Teremos:

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

```

```

setContentView(R.layout.activity_lista_alunos);

AlunoDAO dao = new AlunoDAO(this);
List<Aluno> alunos = dao.buscaAlunos(); dao.close();

ListView listaAlunos = (ListView) findViewById(R.id.lista_alunos);
ArrayAdapter<Aluno> adapter = new ArrayAdapter<Aluno>(this, android.R.layout.simple_li:
listaAlunos.setAdapter(adapter);

Button novoAluno = (Button) findViewById(R.id.novo_aluno);
novoAluno.setOnClickListener((v) -> {
    Intent intentVaiProFormulario = new Intent(ListaAlunosActivity.this, Formulario
});
}

@Override
protected void onResume() {
    super.onResume();
}
}

```

Lembre-se de chamar o `super` para manter o comportamento da `AppCompatActivity` !

O código que faz o carregamento vai do `AlunoDAO` até o fim do `ListView` . Para facilitar, vamos extrair o código que faz o carregamento da lista em um método. Seleccionamos com o mouse o seguinte:

```

AlunoDAO dao = new AlunoDAO(this);
List<Aluno> alunos = dao.buscaAlunos(); dao.close();

ListView listaAlunos = (ListView) findViewById(R.id.lista_alunos);
ArrayAdapter<Aluno> adapter = new ArrayAdapter<Aluno>(this, android.R.layout.simple_li:
listaAlunos.setAdapter(adapter);

```

E com o botão direito vamos em "Refactor> Extract> Method". Clicamos no *Method* e uma janela é aberta pedindo para dar um nome ao método. Chamaremos de "carregalista" e ele será *private*. É só dar um "ok" e teremos o seguinte:

```

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

    carregalista();

    Button novoAluno = (Button) findViewById(R.id.novo_aluno);
    novoAluno.setOnClickListener((v) -> {
        Intent intentVaiProFormulario = new Intent(ListaAlunosActivity.this, Formulario
    });
}

private void carregalista() {
    AlunoDAO dao = new AlunoDAO(this);
    List<Aluno> alunos = dao.buscaAlunos(); dao.close();
}

```

```
ListView listaAlunos = (ListView) findViewById(R.id.lista_alunos);
ArrayAdapter<Aluno> adapter = new ArrayAdapter<Aluno>(this, android.R.layout.simple_li:
listaAlunos.setAdapter(adapter);
}

@Override
protected void onResume() {
    super.onResume();
}
}
```

Mas, repare que o `carregaLista` está no `oncreate`. Não queremos carregá-lo a partir desse método, senão o mesmo erro de antes ocorrerá. Vamos selecionar o `carregaLista()`; e dar um "Comand +X" ou "Ctrl+X" e no `onResume`, depois do `super`, damos um "Command+V" ou "Ctrl+V" e teremos:

```
@Override
protected void onResume() {
    super.onResume();
    carregaLista();
}
```

Agora sim, carregamos a lista a partir do `onResume`. Vamos ver se deu certo? Vamos rodar o emulador e quando adicionarmos mais uma pessoa, salvando esse formulário, teremos o novo nome na lista!

