



ANDROID

Activity e MultiActivity



Activity em foco

Activity em modo Pause

- O método `onPause()` é chamado.
- Activity 2 entra na frente da Activity 1.
- Neste caso Activity 1 ainda é visível.

Activity em modo Stop

- `onStop()` é chamado.
- Activity 3 entra na frente da Activity 1.
- Neste caso Activity 1 não é mais visível.

Prof. Alessandro Brawerman

Activity

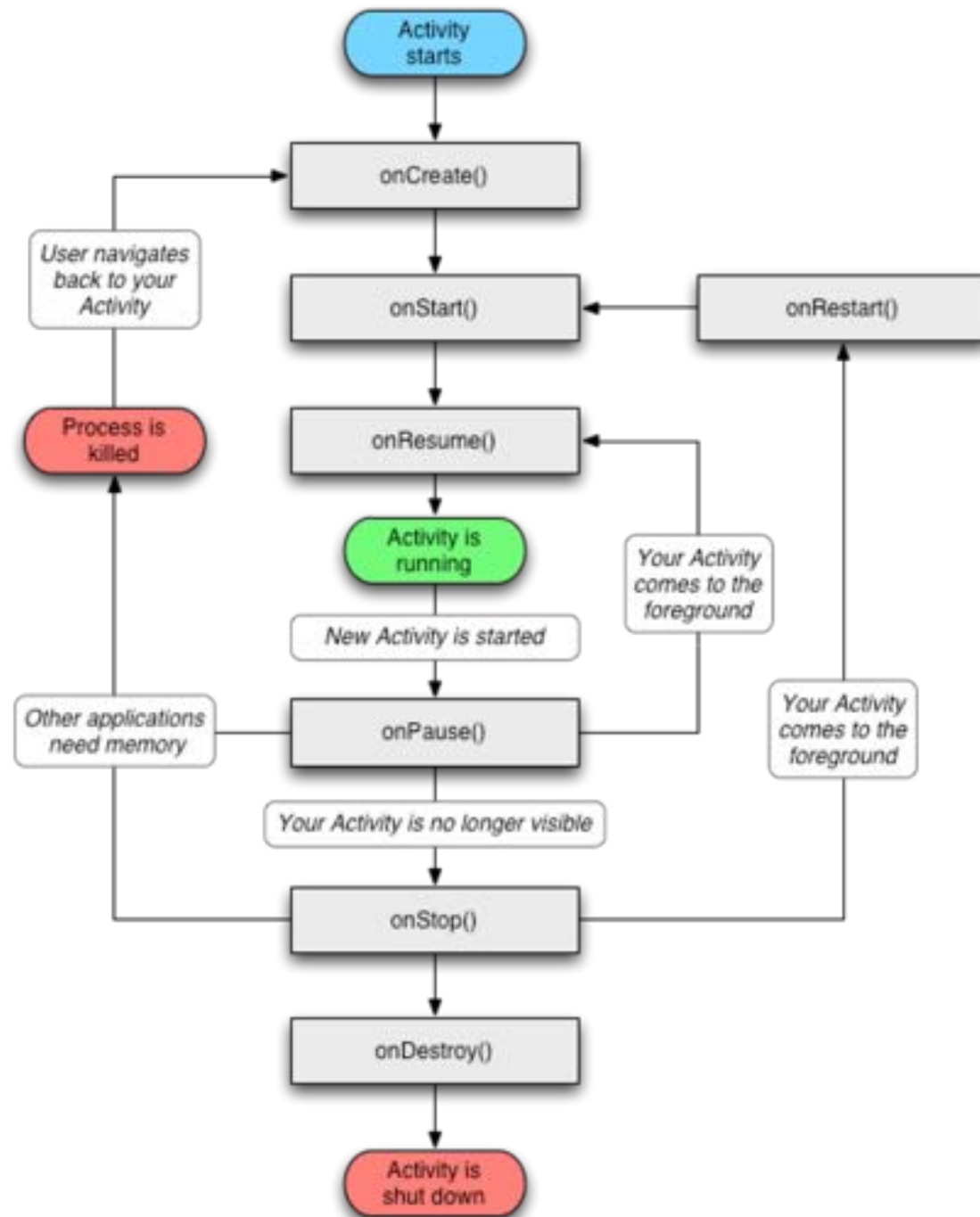
- É uma das classes mais importantes no Android.
- Geralmente, um activity representa uma view da app.
- Na prática activity = view = tela.
- Funções:
 - Controlar os eventos da view que representa
 - Definir qual view é responsável por desenhar a UI
- Herda da classe:
 - `android.app.Activity`
- Cada activity implementa o método `onCreate()`, obrigatório.
- As activities são declaradas no `AndroidManifest.xml`.
 - `<activity android:name=".MinhaClasseActivity"/>`

Activity – Ciclo de Vida

- Representa os possíveis estados que uma activity se encontra:
 - Executando
 - Interrompida (pause), pois está em background
 - Destruída
- O SO cuida do ciclo que é transparente ao usuário e desenvolvedor.
- No entanto, ao desenvolver uma app deve-se levar em conta cada estado para obter uma app mais robusta.
 - App rodando → Telefone toca → User atende
 - O que acontece com a app??
 - E quando a mesma voltar??

Activity – Ciclo de Vida

- Cada activity que é iniciada é inserida no topo de uma pilha, a activity stack.
 - App é iniciada, tela 1 é mostrada.
 - Tela 2 é carregada (Pilha = tela 2 | tela 1).
 - Tela 2 é encerrada, tela 1 reaparece.
- Activity no topo da pila é a que está executando.
- As outras podem estar em pause ou executando em segundo plano.
- Quando uma activity está pausada, o SO pode encerrar a mesma para liberar recursos.



Activity - Métodos

- onCreate()
 - Método de inicialização. Obrigatório.
 - Chamado uma única vez.
 - Aloca recursos necessários.
 - Cria uma view e chamar o método setContentView(view) para exibir esta view na tela.
 - Após este é chamado o método onStart() para mostrar a view na tela.
- onStart()
 - Inicia o ciclo de vida da activity.
 - Mostra a primeira view na tela.
 - É chamado após o onCreate() ou onResume().

Activity - Métodos

- `onRestart()`
 - Reiniciando ciclo de vida depois de um `onStop()`.
 - Usuário decidiu voltar a executar a app que estava em background.
 - Chama o `onStart()` de forma automática.
- `onResume()`
 - Representa o estado em que a activity está executando.
 - Ela está no topo da pilha.
 - Chamado sempre depois do `onStart()`.

Activity - Métodos

- `onDestroy()`
 - Encerra a execução da activity.
 - Libera recursos alocados.
 - Pode ser chamado pelo SO ou pela app a partir do método `finish()`.

Subciclos do Ciclo de Vida

- Há 3 subníveis do ciclo de vida principal.
- Estes ficam se repetindo durante a execução da app.
- Entire Lifetime
 - Ciclo de vida completo entre o início e destruição da activity.
 - Ocorre apenas uma vez, definindo o tempo de vida completo da activity.
 - Entre onCreate() e onDestroy().
 - No onCreate() a activity ainda não está visível e está alocando recursos.
 - No onDestroy() a activity deixa de ser visível e libera os recursos alocados.

Subciclos do Ciclo de Vida

- Visible Lifetime

- Activity está iniciada, pode estar sendo mostrada ou estar em segundo plano, pausada.
- Entre os métodos onStart() e onStop().
- Uma activity só ocupa o topo da pilha quando o onResume() é chamado.
- onStart() e onStop() podem ser chamados várias vezes.
- Toda vez que onStart() é chamado, o onResume() é chamado automaticamente.

- Foreground Lifetime

- Activity está no topo da pilha e interagindo com o usuário.
- Entre os métodos onResume() e onPause().

Apps Multi-View

- Dois métodos podem ser utilizados para iniciar outra activity:
 - `startActivity(intent)`
 - Inicia a próxima activity sem vínculo com a atual.
 - `startActivityForResult(intent, codigo)`
 - Recebe um parâmetro de quem a chamou para poder fornecer um retorno para a origem.

Exemplo App Multi-View

- Crie um novo projeto MultiView e uma activity Tela1.



```
public class Tela1 extends Activity {  
  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_tela1);  
    }  
  
    public void onClick(View view) {  
        Intent it = new Intent(this, Tela2.class);  
        startActivity(it);  
    }  
}
```

Content Description		...
On Click	onClick	...

Exemplo App Multi-View

- Crie uma nova activity Tela2.
 - New → Activity → Blank Activity



```
public class Tela2 extends Activity {  
  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_tela2);  
    }  
  
    public void onClick(View view) {  
        Intent it = new Intent(this, Tela1.class);  
        startActivity(it);  
    }  
}
```

Content Description		...
On Click	onClick	...

Exemplo App Multi-View: Manifesto

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.ale.multiview" >

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="MultiView"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".Tela1"
            android:label="MultiView" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".Tela2"
            android:label="Tela2" >
        </activity>
    </application>

</manifest>
```

Exemplo App Multi-View

- Note que as activities são empilhadas.
- Caso queira finalizar uma activity ao invés de empilhá-la, basta chamar o método finish().

```
public void onClick(View view) {  
    Intent it = new Intent(this, Tela2.class);  
    startActivity(it);  
    finish();  
}
```

- Execute o app, abra a tela 2 e tente usar o botão voltar.

Passando Dados entre Telas

- A classe Bundle permite passar parâmetros via uma intent.
- Basta inicializar um objeto e usar os métodos put de passagem de parâmetros.
 - putString, putInt, putDouble,
- O objeto é incluído na intent com o método putExtras.

```
public void onClick(View view) {  
    Intent it = new Intent(this, Tela2.class);  
    Bundle params = new Bundle();  
    params.putString("msg", "Olá");  
    it.putExtras(params);  
    startActivity(it);  
}
```

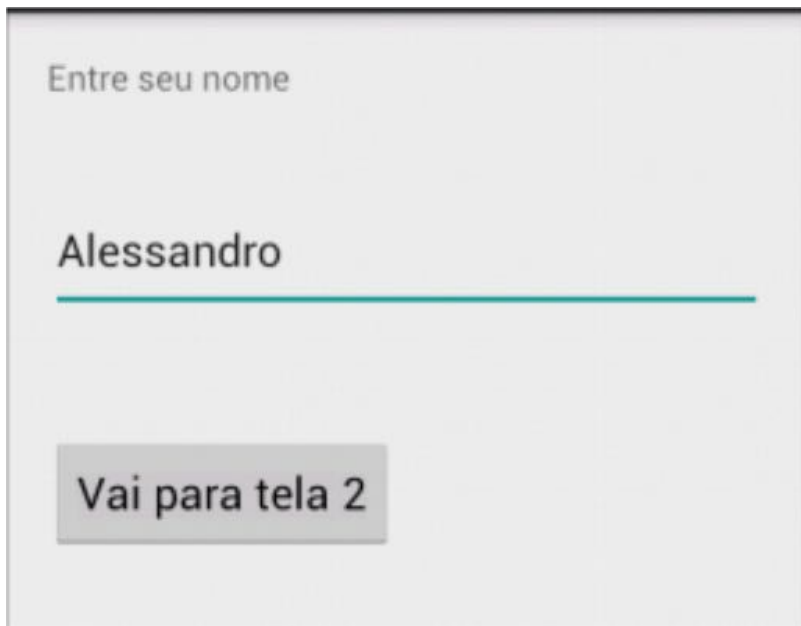

Passando Dados entre Telas

- A activity que recebe os parâmetros deve usar a intent para ter acesso aos mesmos.
- O método getIntent() acessa a intent.
- O método getExtas() recupera o objeto passado.

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_tela2);
    Intent it = getIntent();
    if(it != null) {
        Bundle params = it.getExtras();
        if (params != null) {
            String msg = params.getString("msg");
        }
    }
}
```

Exercício

- Faça com que a Tela 1 capture o nome do usuário e a Tela 2 exiba uma mensagem de boas vindas:
 - “Bem-vindo NOME_USER”

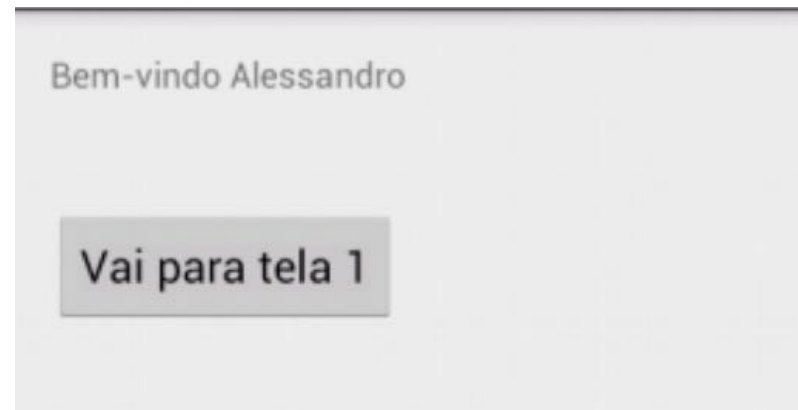


Entre seu nome

Alessandro

Vai para tela 2

This screenshot shows a mobile application screen with a light gray background. At the top, the text 'Entre seu nome' is displayed in a dark gray font. Below this, the name 'Alessandro' is entered into a text field, which is highlighted by a blue underline. At the bottom of the screen, there is a gray button with the text 'Vai para tela 2' in black.



Bem-vindo Alessandro

Vai para tela 1

This screenshot shows a mobile application screen with a light gray background. At the top, the text 'Bem-vindo Alessandro' is displayed in a dark gray font. Below this, there is a gray button with the text 'Vai para tela 1' in black.

Exercício

- Faça uma app que captura dados de um usuário em uma activity e mostra estes em uma segunda activity.
 - Nome
 - Idade
 - Telefone
 - Login
 - Senha
- Use as máscaras apropriadas dos EditTexts.
- Não faça um botão para voltar para a tela 1, use o botão padrão do Android.

Passando Dados entre Telas

```
public void onClick(View view) {  
    Intent it = new Intent(this, Tela2.class);  
    Bundle params = new Bundle();  
    params.putString("msg", "Olá");  
    it.putExtras(params);  
    startActivity(it);  
}
```

```
public void onClick(View view) {  
    Intent it = new Intent(this, Tela2.class);  
    it.putExtra("msg", "Olá");  
    startActivity(it);  
}
```

Passando Dados entre Telas

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_tela2);
    Intent it = getIntent();
    if(it != null) {
        Bundle params = it.getExtras();
        if (params != null) {
            String msg = params.getString("msg");
        }
    }
}
```

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_tela2);
    Intent it = getIntent();
    if(it != null) {
        String msg = it.getStringExtra("msg");
    }
}
```

Exercício

- Calculadora IMC
 - Faça um app que captura dados de um usuário em uma activity e mostra o resultado em uma segunda activity.
 - Nome
 - Peso
 - Altura
- Use as máscaras apropriadas dos EditTexts.
- Não faça um botão para voltar para a tela 1, use o botão padrão do Android.
- Use a tabela a seguir para o resultado do cálculo
 - $IMC = peso / altura^2$

Exercício

FAIXA DE IMC	PESO
Abaixo de 15	Extremamente abaixo do peso
Entre 15 e 16	Gravemente abaixo do peso
Entre 16 e 18,5	Abaixo o peso ideal
Entre 18,5 e 25	Faixa de peso ideal
Entre 25 e 30	Sobrepeso
Entre 30 e 35	Obesidade grau I
Entre 35 e 40	Obesidade grau II (grave)
Acima de 40	Obesidade grau III (mórbida)