



Android



UFOP

Universidade Federal
de Ouro Preto

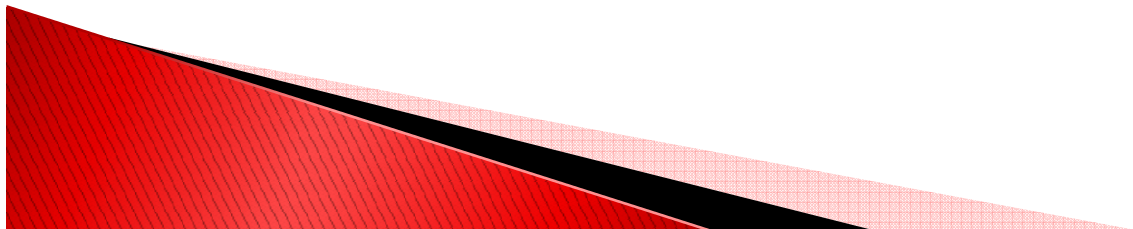


decom

departamento
de computação

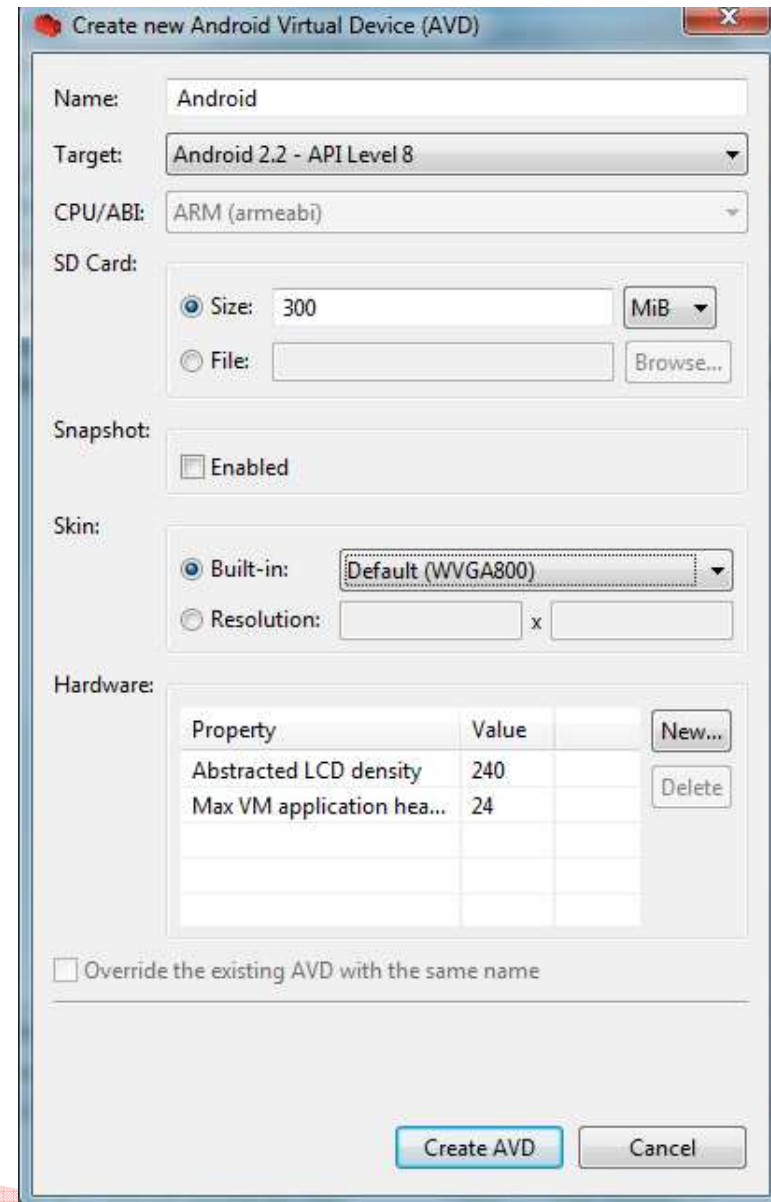
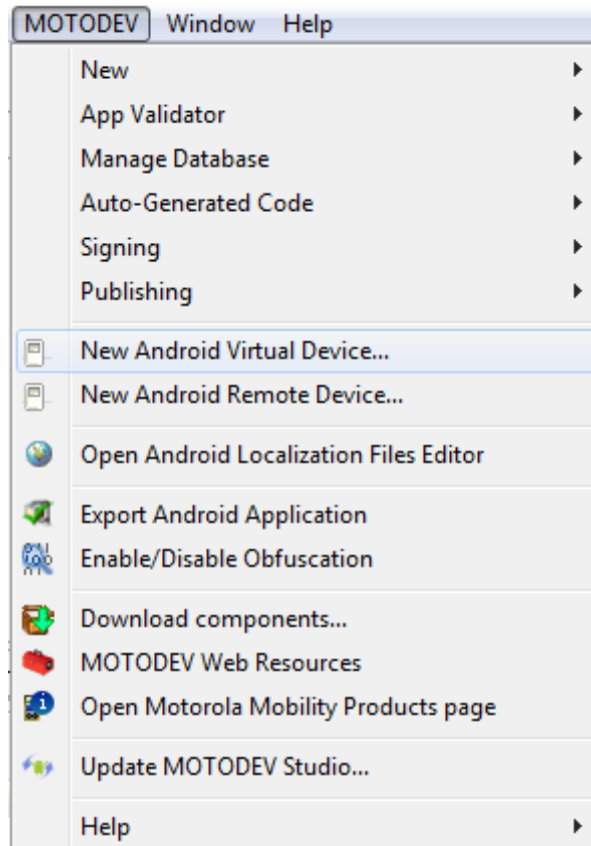
Configurando o ambiente

- ▶ Baixar e instalar o MOTODEV
 - <http://developer.motorola.com/tools/motodevstudio>
- ▶ Baixar e instalar o SDK Android
 - <http://developer.android.com/sdk>
- ▶ Se necessário, baixar e instalar o JDK
 - <http://www.oracle.com/technetwork/java/javase/>



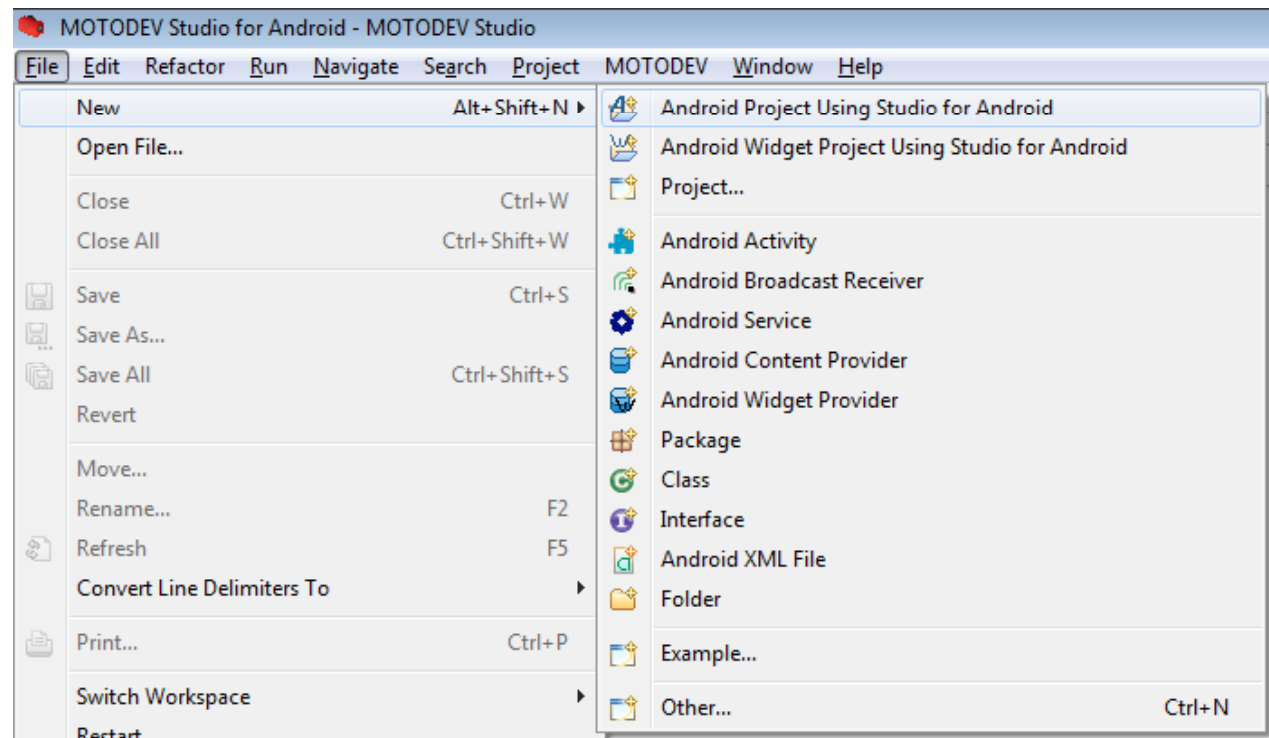
Configurando o ambiente

► Criar AVD



Novo projeto

- ▶ Criando um novo projeto
- ▶ File -> New -> Android Project Using Studio for Android



Novo projeto

New Android Project

Create an Android project using MOTODEV Studio for Android
Creates a new Android project resource.

Project name: Service

Contents

- ☒ Create new project
- ☐ Create new project using sample
- ☐ Create project from existing source
- ☒ Use default location

Location: C:\Users\Lucas Schmidt\workspace Browse...

Target

SDK Target	Vendor	API Level	Plat...
☐ Android 1.5	Android Open Source Project	3	1.5
☐ Android 1.6	Android Open Source Project	4	1.6
☐ Android 2.1	Android Open Source Project	7	2.1
☒ Android 2.2	Android Open Source Project	8	2.2

Standard Android platform 2.2

Application

Application name: User Application

Package name: com.service

Activity name: MainActivity

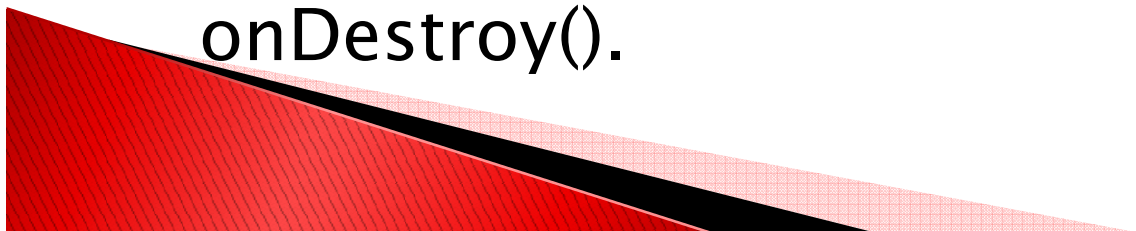
Min SDK version: 8

☐ Add native support

? < Back Next > Finish Cancel

Service

- ▶ Um serviço que executa um loop com um contador até 50 e imprime as mensagens no LogCat.
- ▶ A classe que representa o serviço deve ser uma subclasse de `android.app.Service` e deve obrigatoriamente implementar o método `IBinder onBind(intent)`, e se necessário métodos para controlar o ciclo de vida do Serviço, como `onCreate()`, `onStart()` e `onDestroy()`.

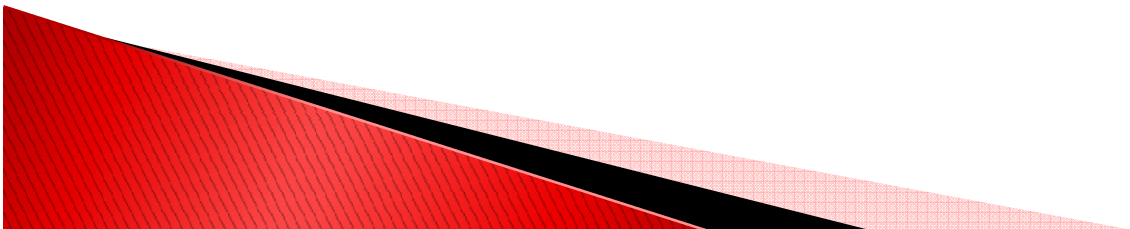


Service

- ▶ O método `IBinder onBind(intent)` serve para realizar conexões com outros componentes. Exemplo: conexões RPC

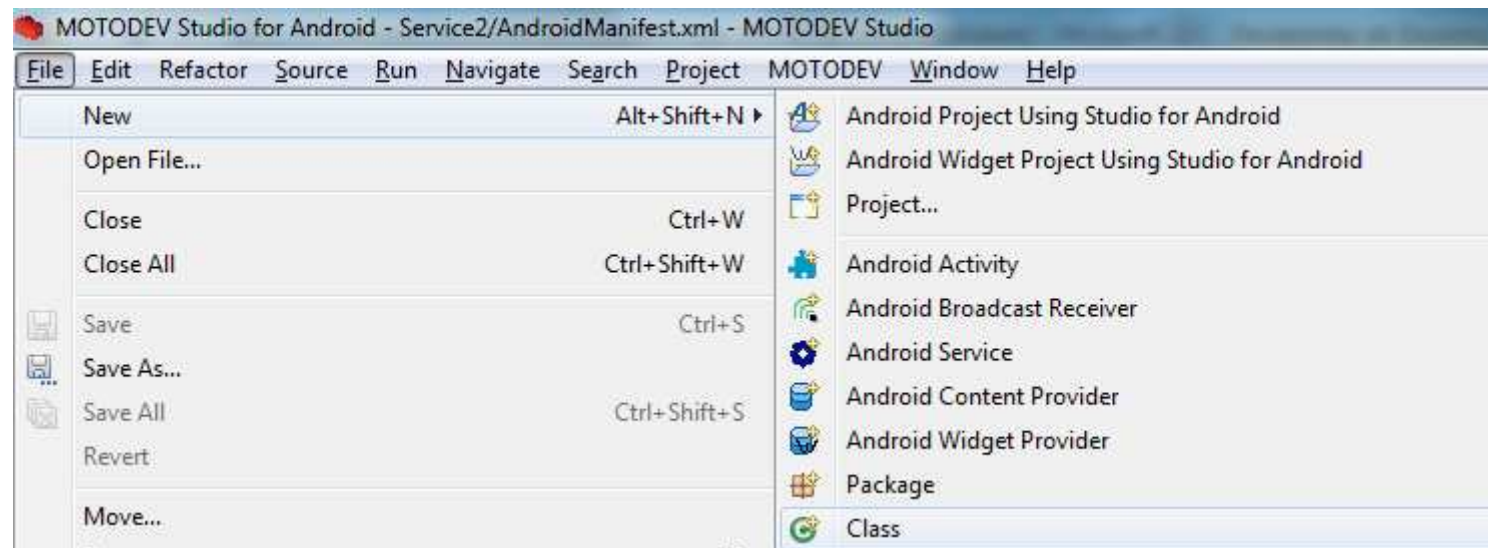


Vamos ver na prática!



Service

- ▶ Crie uma nova classe, chamada ExemploServico: Clique no pacote com.service e File -> New -> Class



Service

New Java Class

Java Class

⚠ The use of the default package is discouraged.

Source folder: Service/src Browse...

Package: (default) Browse...

☐ Enclosing type: Browse...

Name: ExemploService

Modifiers: ☒ public ☐ default ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass: java.lang.Object Browse...

Interfaces: Add...
Remove

Which method stubs would you like to create?

☐ public static void main(String[] args)
☐ Constructors from superclass
☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

? Finish Cancel

ExemploServico

```
package com.service;

import android.app.Service;
import android.content.Intent;
import android.os.IBinder;
import android.util.Log;

public class ExemploServico extends Service implements Runnable {

    private static final int MAX = 50;
    private static String CATEGORIA = "livro";
    protected int count;
    private boolean ativo;

    @Override
    public IBinder onBind(Intent arg0) {
        // TODO Auto-generated method stub
        return null;
    }
}
```

Limite do loop
Tag do LogCat

Método IBinder onBind(Intent)

ExemploServico

Métodos onCreate(),
onStart() e onDestroy()

```
@Override
public void onCreate() {
    Log.i(CATEGORIA, "ExemploServico.onCreate()");
    ativo = true;
    // Delega para uma thread
    new Thread(this).start();
}

@Override
public void onStart(Intent intent, int startId) {
    Log.i(CATEGORIA, "ExemploServico.onStart()");
}

@Override
public void onDestroy() {
    // Ao encerrar o serviço, altera o flag para a thread parar
    ativo = false;
    Log.i(CATEGORIA, "ExemploServico.onDestroy()");
}
```

ExemploServico

```
@Override
public void run() {
    // TODO Auto-generated method stub
    while (ativo && count < MAX) {
        fazAlgumaCoisa();
        Log.i(CATEGORIA, "ExemploServico executando..." + count);
        count++;
    }
    Log.i(CATEGORIA, "ExemploServico fim.");
    stopSelf();
}

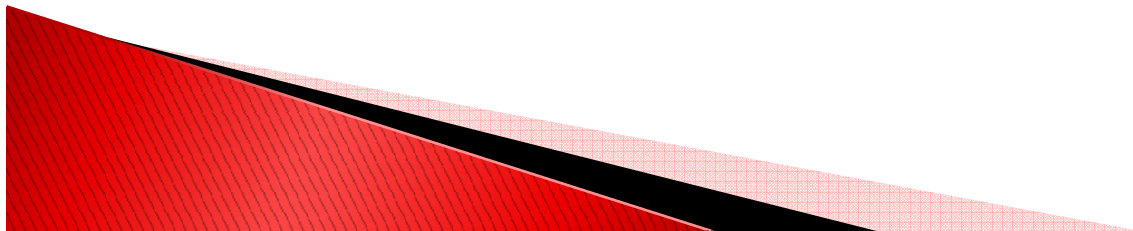
private void fazAlgumaCoisa() {
    try {
        // Simula algum processamento
        Thread.sleep(1000);
    } catch (InterruptedException e){
        e.printStackTrace();
    }
}
}
```

Método run() – padrão Runnable
Chama função fazAlgumaCoisa()

Para simular um processamento
demorado, a classe
fazAlgumaCoisa() faz a thread
dormir por 1 segundo

ExemploServico

- ▶ No método `run()`, quando o valor do contador chega a 50, o loop da thread termina e o método `stopSelf()` é chamado, o que encerra o ciclo de vida do serviço, fazendo com que o próprio Android chame o método `onDestroy`, encerrando o processo para liberar memória e recursos utilizados.



AndroidManifest.xml

- ▶ Dentro do projeto altere o arquivo AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.service2"
    android:versionCode="1"
    android:versionName="1.0">
    <uses-sdk android:minSdkVersion="8" />

    <application android:icon="@drawable/ic_launcher" android:label="@string/app_name">
        <activity android:name="MainActivity"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <service android:name="ExemploServico">
            <intent-filter>
                <action android:name="SERVICE_1" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </service>
    </application>
</manifest>
```

View

- ▶ Agora, vamos modificar nossa view para facilitar o Start da nossa aplicação:
- ▶ res/layout/main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
    <Button
        android:id="@+id/btIniciar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Iniciar"
        />
    <Button
        android:id="@+id/btParar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Parar"
        />
</LinearLayout>
```


MainActivity

- ▶ Modificar a Activity
(src/com.service/MainActivity.java)

```
package com.service;

+ import android.app.Activity;

public class MainActivity extends Activity {
    /** Called when the activity is first created. */

    private static final String CATEGORIA = "livro";

    - @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

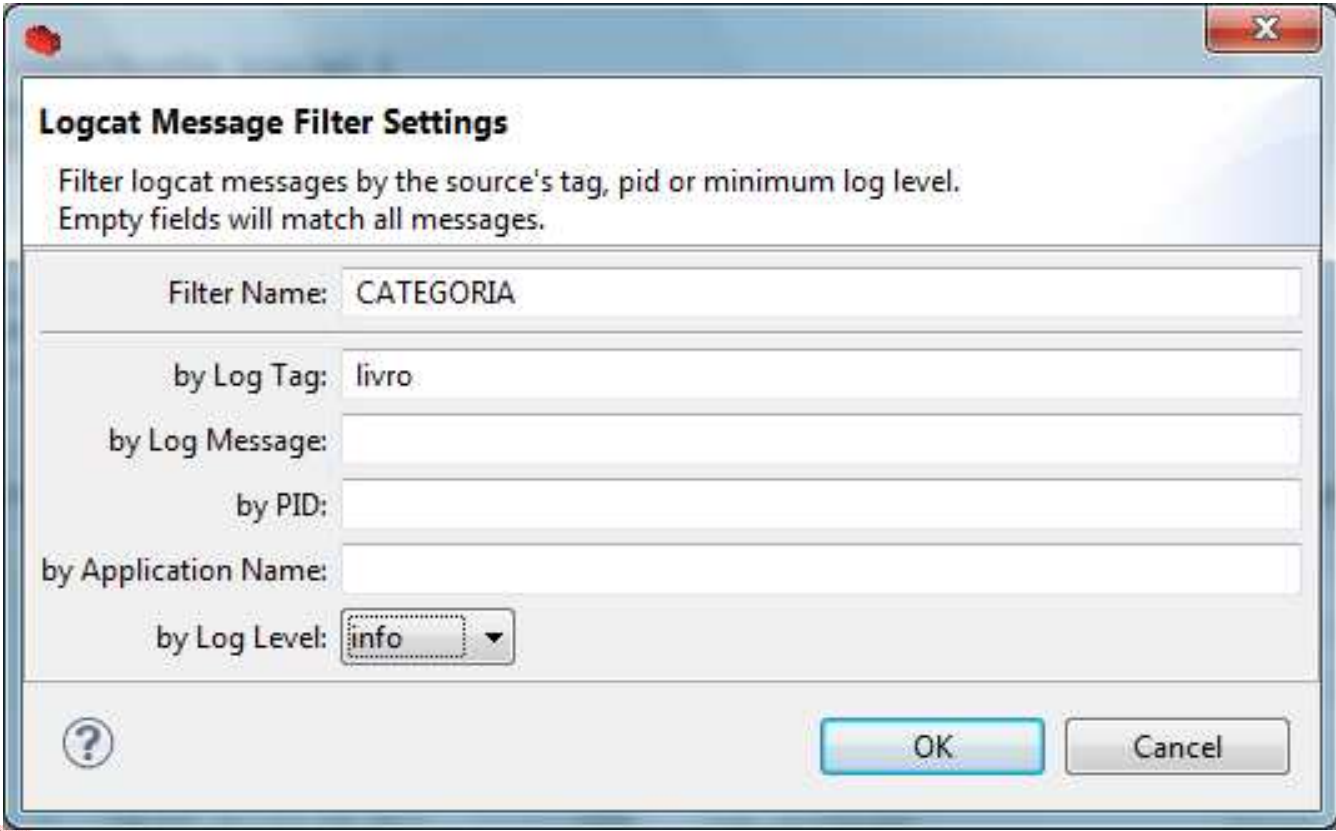
MainActivity

```
- // Mesma intent é utilizada para iniciar e parar
final Intent it = new Intent("SERVICE_1");
Button bIniciar = (Button) findViewById(R.id.btIniciar);
- bIniciar.setOnClickListener(new Button.OnClickListener() {
-     public void onClick(View v) {
        // TODO Auto-generated method stub
        // Iniciar o serviço
        startService(it);
    }
- });
Button bParar = (Button) findViewById(R.id.btParar);
- bParar.setOnClickListener(new Button.OnClickListener() {
-     public void onClick(View v) {
        //Parar o serviço
        stopService(it);
    }
- });
}

@Override
protected void onDestroy() {
    super.onDestroy();
    Log.i(CATEGORIA, "ExemploIniciarServico.onDestroy()");
}
}
```

Teste 1

- ▶ Crie um filtro do LogCat:



The screenshot shows a dialog box titled "Logcat Message Filter Settings". It contains instructions: "Filter logcat messages by the source's tag, pid or minimum log level. Empty fields will match all messages." Below this, there are several input fields: "Filter Name:" with the value "CATEGORIA", "by Log Tag:" with the value "livro", "by Log Message:" (empty), "by PID:" (empty), "by Application Name:" (empty), and "by Log Level:" with a dropdown menu set to "info". At the bottom, there is a help icon (question mark), an "OK" button, and a "Cancel" button.

Logcat Message Filter Settings

Filter logcat messages by the source's tag, pid or minimum log level.
Empty fields will match all messages.

Filter Name: CATEGORIA

by Log Tag: livro

by Log Message:

by PID:

by Application Name:

by Log Level: info

? OK Cancel

Teste 1

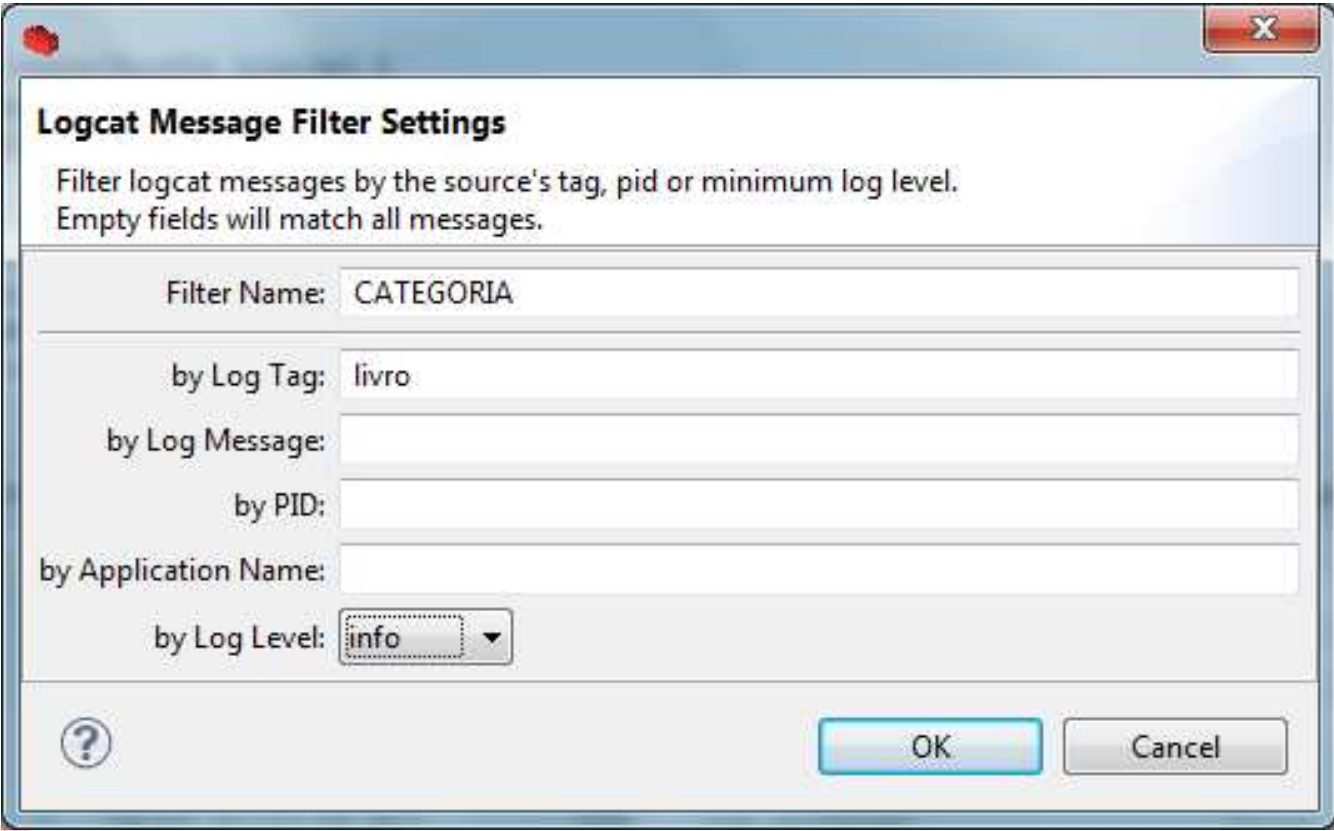
- ▶ Compile a aplicação;
- ▶ Clique no botão “Iniciar”;
- ▶ Podemos ver a execução da aplicação no LogCat:

I	04-17 21:01:14.539	279	com.service2	livro	ExemploServico.onCreate()
I	04-17 21:01:14.567	279	com.service2	livro	ExemploServico.onStart()
I	04-17 21:01:15.639	279	com.service2	livro	ExemploServico executando...0
I	04-17 21:01:16.645	279	com.service2	livro	ExemploServico executando...1
I	04-17 21:01:17.657	279	com.service2	livro	ExemploServico executando...2

- ▶ Clique em “Parar”.

Teste 2

- ▶ Crie um filtro do LogCat:



The screenshot shows a dialog box titled "Logcat Message Filter Settings". It contains instructions: "Filter logcat messages by the source's tag, pid or minimum log level. Empty fields will match all messages." Below this, there are several input fields: "Filter Name:" with the value "CATEGORIA", "by Log Tag:" with the value "livro", "by Log Message:" (empty), "by PID:" (empty), "by Application Name:" (empty), and "by Log Level:" with a dropdown menu set to "info". At the bottom, there is a help icon (question mark), an "OK" button, and a "Cancel" button.

Logcat Message Filter Settings

Filter logcat messages by the source's tag, pid or minimum log level.
Empty fields will match all messages.

Filter Name: CATEGORIA

by Log Tag: livro

by Log Message:

by PID:

by Application Name:

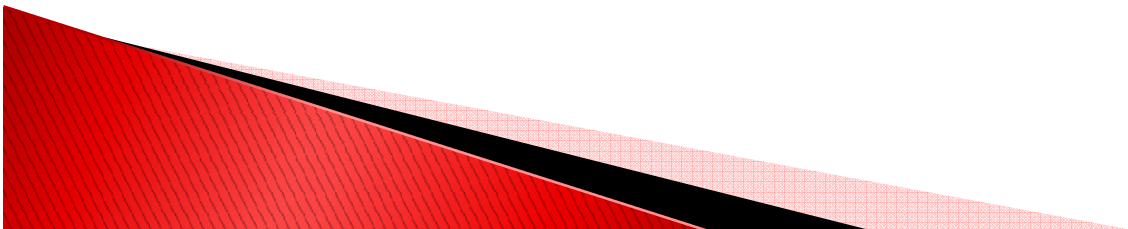
by Log Level: info

OK Cancel

Teste 2

- ▶ Compile a aplicação
- ▶ Clique no botão “Iniciar”;
- ▶ Podemos ver a execução da aplicação no LogCat

I	04-17 21:01:14.539	279	com.service2	livro	ExemploServico.onCreate()
I	04-17 21:01:14.567	279	com.service2	livro	ExemploServico.onStart()
I	04-17 21:01:15.639	279	com.service2	livro	ExemploServico executando...0
I	04-17 21:01:16.645	279	com.service2	livro	ExemploServico executando...1
I	04-17 21:01:17.657	279	com.service2	livro	ExemploServico executando...2



Teste 2

- ▶ Clique no botão “Sair” do emulador; 
- ▶ Confira o resultado no LogCat...
- ▶ Ele ainda está rodando, ok? Isto é o Service!



- ▶ A execução só será interrompida quando o loop chegar ao valor 50, ou se você entrar na aplicação e clicar no botão “Parar”.

