

Disciplina: Programação de dispositivos móveis

Professor: Aristofânio Garcia

Aluno: Fatima Ferreira de Sousa

Para o desenvolvimento dessa atividade será utilizado os exemplos disponibilizados no documento proposto pelo professor. O mesmo consiste em um pequeno tutorial para o desenvolvimento de duas aplicações simples feitas em Android. Abaixo temos a descrição de cada uma das atividades:

Atividade PDM – Descrição atividade 1

Passos que devem ser seguidos para realizar atividade 01, que consiste na criação de uma aplicação que realizara o download de um arquivo.

Inicialmente foi criado um projeto Android com o nome Services. O nome do pacote deve ser br.com.k19.android.cap08, e o nome da activity MainActivity.

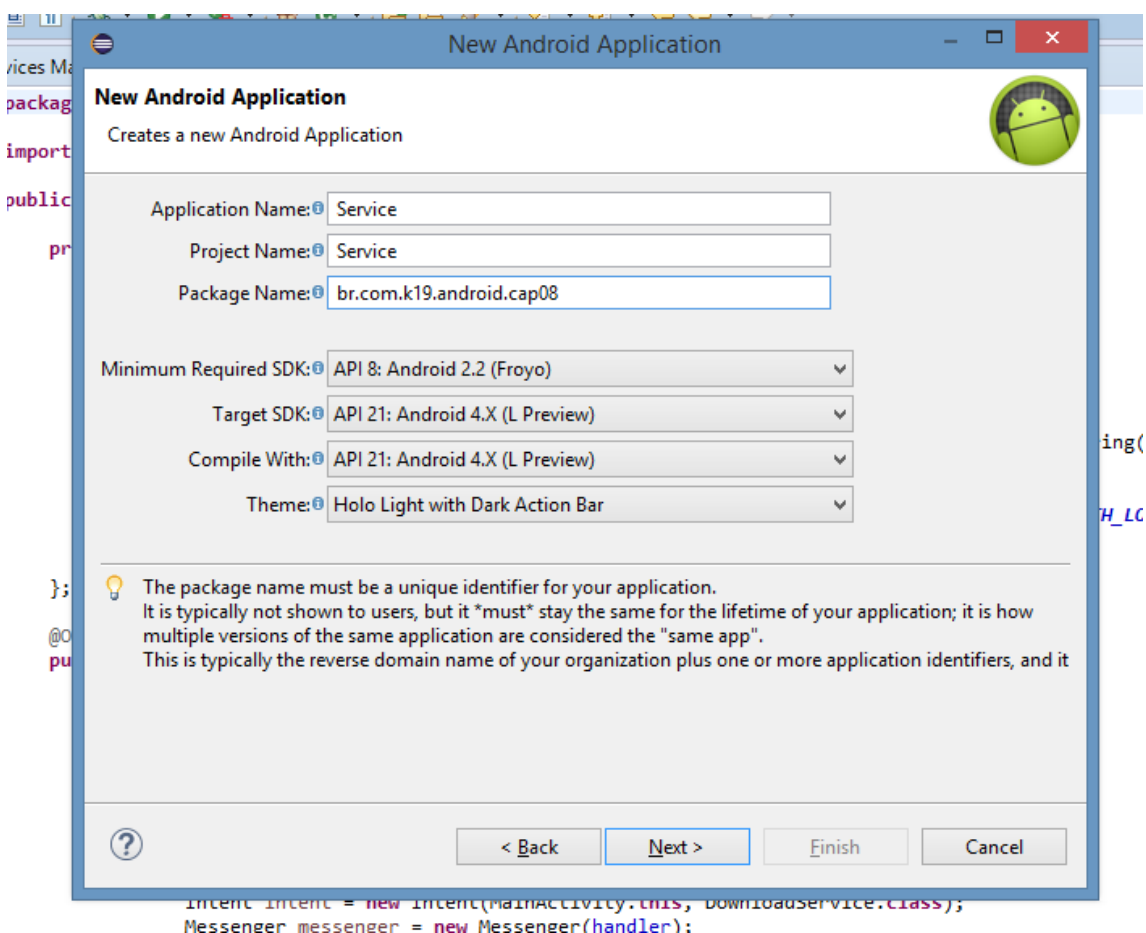


Figura 1 - Criando Projeto

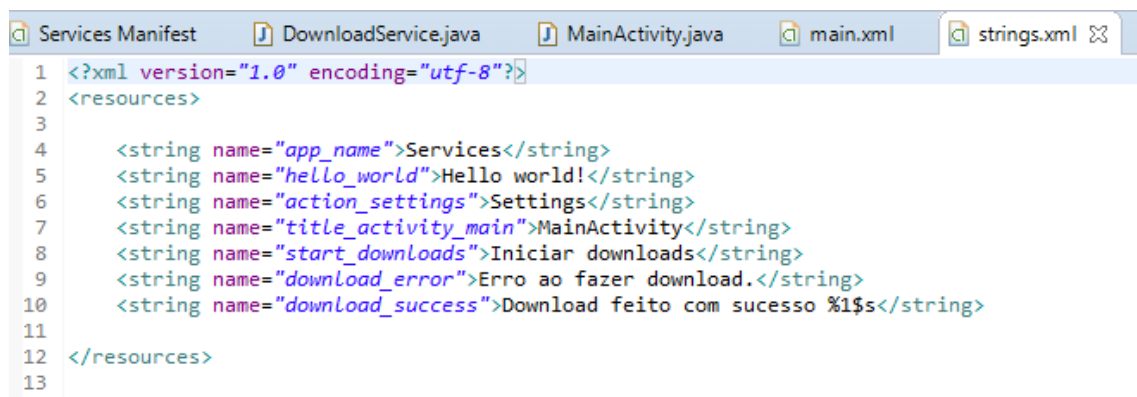
Em seguida, na pasta res/layouts crie um arquivo chamado main.xml. Similar o arquivo mostrado na figura abaixo:



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:layout_width="match_parent"
4     android:layout_height="match_parent"
5     android:orientation="vertical" >
6
7     <Button
8         android:id="@+id/start_button"
9         android:layout_width="match_parent"
10        android:layout_height="wrap_content"
11        android:text="@string/start_downloads"
12        android:layout_gravity="center" />
13
14 </LinearLayout>
15
```

Figura 2 - main.xml

Logo depois você deve alterar o arquivo strings.xml, o mesmo deve ficar igual ao da figura abaixo:



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3
4     <string name="app_name">Services</string>
5     <string name="hello_world">Hello world!</string>
6     <string name="action_settings">Settings</string>
7     <string name="title_activity_main">MainActivity</string>
8     <string name="start_downloads">Iniciar downloads</string>
9     <string name="download_error">Erro ao fazer download.</string>
10    <string name="download_success">Download feito com sucesso %1$s</string>
11
12 </resources>
13
```

Figura 3 - strings.xml

Feito isso, crie um arquivo DownloadService.java semelhante aos das figuras abaixo:

```
Services Manifest | DownloadService.java | MainActivity.java
import java.io.*;

18
19 public class DownloadService extends IntentService {
20
21     private int result = Activity.RESULT_CANCELED;
22
23     public DownloadService() {
24         super("DownloadService");
25     }
26
27     @Override
28     protected void onHandleIntent(Intent intent) {
29         Uri data = intent.getData();
30         String urlPath = intent.getStringExtra("urlPath");
31         String fileName = data.getPath();
32         File output = new File(Environment.getExternalStorageDirectory(),
33             fileName);
34
35         if (output.exists()) {
36             output.delete();
37         }
38
39         InputStream stream = null;
40         FileOutputStream fos = null;
41
42         try {
43             URL url = new URL(urlPath);
44             stream = url.openConnection().getInputStream();
45             InputStreamReader reader = new InputStreamReader(stream);
46             fos = new FileOutputStream(output.getPath());
47             int next = -1;
48             while ((next = reader.read()) != -1) {
49                 fos.write(next);
50             }
51
52             result = Activity.RESULT_OK;
53
54         } catch (Exception e) {
55             e.printStackTrace();
56         }
57     }
58 }
```

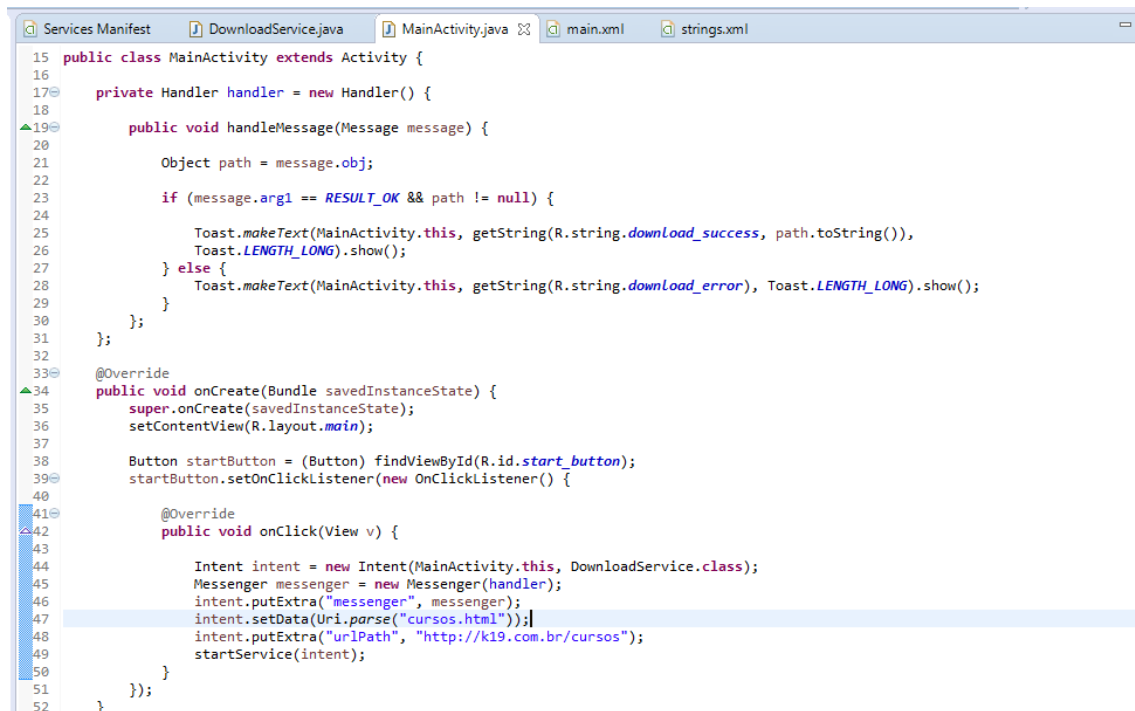
```

53
54     } catch (Exception e) {
55         e.printStackTrace();
56     }
57     finally {
58         if (stream != null) {
59             try {
60                 stream.close();
61             } catch (IOException e) {
62                 e.printStackTrace();
63             }
64         }
65
66         if (fos != null) {
67             try {
68                 fos.close();
69             } catch (IOException e) {
70                 e.printStackTrace();
71             }
72         }
73     }
74
75     Bundle extras = intent.getExtras();
76
77     if (extras != null) {
78         Messenger messenger = (Messenger) extras.get("messenger");
79         Message msg = Message.obtain();
80         msg.arg1 = result;
81         msg.obj = output.getAbsolutePath();
82         try {
83             messenger.send(msg);
84         } catch (android.os.RemoteException e1) {
85             Log.e("DownloadService", "Erro ao enviar mensagem", e1);
86         }
87     }
88 }
89
90 }

```

Figura 4 - DownloadService.java

Para finalizar, edite a classe MainActivity.java, deixando a mesma igual à da figura abaixo:



```
15 public class MainActivity extends Activity {
16
17     private Handler handler = new Handler() {
18
19         public void handleMessage(Message message) {
20
21             Object path = message.obj;
22
23             if (message.arg1 == RESULT_OK && path != null) {
24
25                 Toast.makeText(MainActivity.this, getString(R.string.download_success, path.toString()),
26                     Toast.LENGTH_LONG).show();
27             } else {
28                 Toast.makeText(MainActivity.this, getString(R.string.download_error), Toast.LENGTH_LONG).show();
29             }
30         };
31     };
32
33     @Override
34     public void onCreate(Bundle savedInstanceState) {
35         super.onCreate(savedInstanceState);
36         setContentView(R.layout.main);
37
38         Button startButton = (Button) findViewById(R.id.start_button);
39         startButton.setOnClickListener(new OnClickListener() {
40
41             @Override
42             public void onClick(View v) {
43
44                 Intent intent = new Intent(MainActivity.this, DownloadService.class);
45                 Messenger messenger = new Messenger(handler);
46                 intent.putExtra("messenger", messenger);
47                 intent.setData(Uri.parse("cursos.html"));
48                 intent.putExtra("urlPath", "http://k19.com.br/cursos");
49                 startService(intent);
50             }
51         });
52     }
53 }
```

Figura 5 - MainActivity.java

Para execução é importante que o emulador esteja configurado com size de 200 mib e também é necessário que o arquivo ServiceManifest.xml esteja configurado de acordo com a figura abaixo:

```
Services Manifest
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     package="br.com.k19.android.cap08"
4     android:versionCode="1"
5     android:versionName="1.0" >
6
7     <uses-sdk
8         android:minSdkVersion="8"
9         android:targetSdkVersion="21" />
10
11     <uses-permission android:name="android.permission.INTERNET" />
12     <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
13     <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
14
15 <application
16     android:allowBackup="true"
17     android:icon="@drawable/ic_launcher"
18     android:label="@string/app_name"
19     android:theme="@style/AppTheme" >
20     <activity
21         android:name=".MainActivity"
22         android:label="@string/app_name" >
23         <intent-filter>
24             <action android:name="android.intent.action.MAIN" />
25
26             <category android:name="android.intent.category.LAUNCHER" />
27         </intent-filter>
28     </activity>
29 </application>
30
31 </manifest>
32
```

Figura 6 - ServiceManifest

Depois de feitas essas configurações, execute a aplicação e o resultado deve ser semelhante ao da figura abaixo:



Figura 7 - Aplicação Executando

Atividade PDM – Descrição atividade 2

Passos que devem ser seguidos para realizar atividade 02, que consiste em uma aplicação que faz uso do BroadcastReceiver. O receiver geralmente é usado para receber alguma notificação do sistema e a partir disto executar alguma tarefa dependendo do tipo de notificação recebido. Abaixo um exemplo simples de uma aplicação que faz uso do receiver para notificar a chegada de uma ligação telefônica.

Inicialmente deve-se criar um novo projeto Android no eclipse com o seguinte nome: Reveiller. O nome do pacote deve ser br.com.k19.android.cap08_02, e o nome da activity deve ser MainActivity.

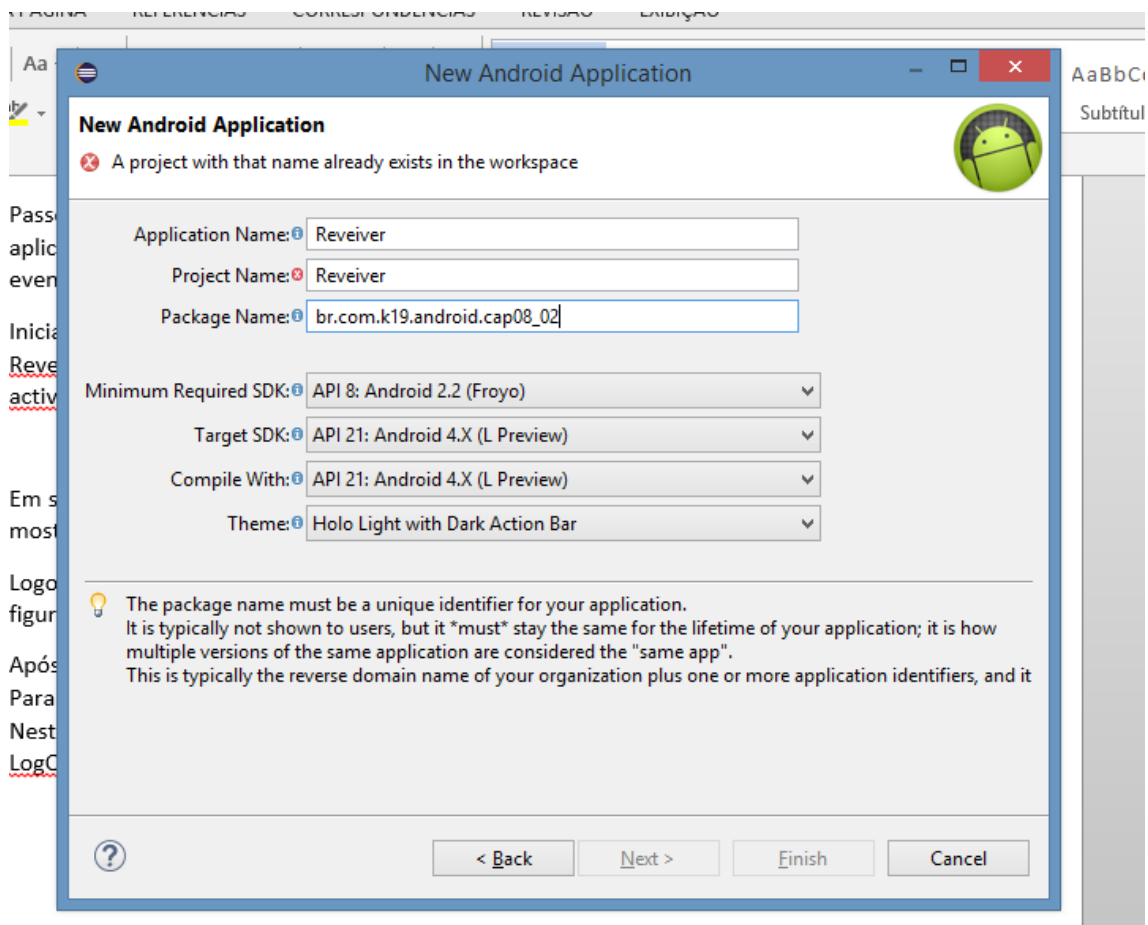
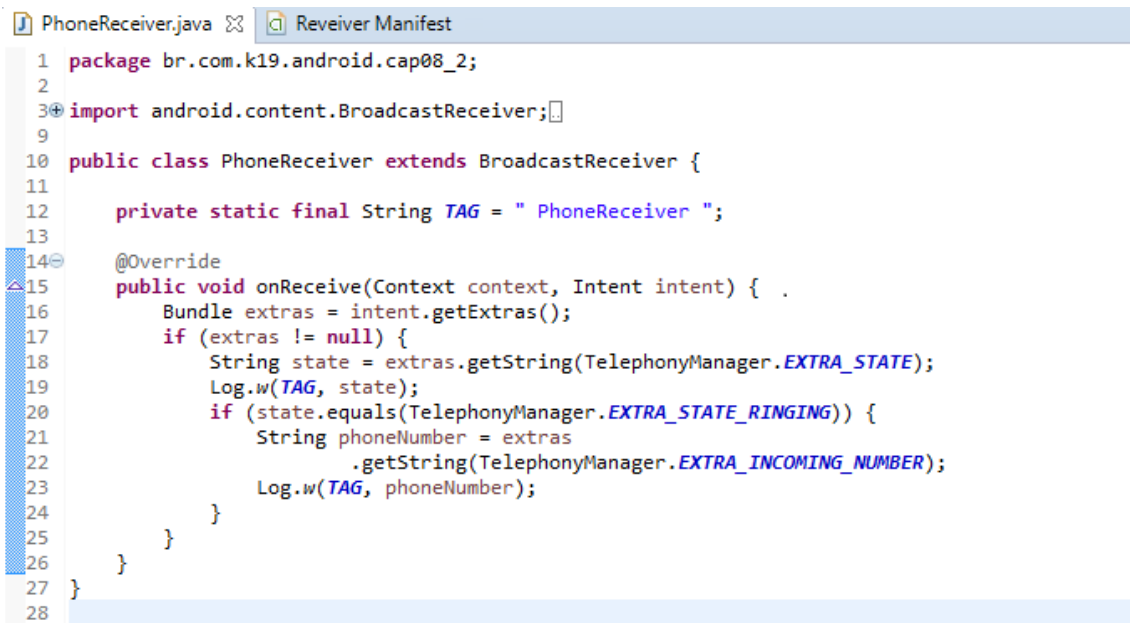


Figura 8 - Criando Projeto

Em seguida deve ser criado um arquivo chamado PhoneReceiver.java com o conteúdo mostrado na imagem abaixo:



```

1 package br.com.k19.android.cap08_2;
2
3 import android.content.BroadcastReceiver;
4
5
6
7
8
9
10 public class PhoneReceiver extends BroadcastReceiver {
11
12     private static final String TAG = " PhoneReceiver ";
13
14     @Override
15     public void onReceive(Context context, Intent intent) {
16         Bundle extras = intent.getExtras();
17         if (extras != null) {
18             String state = extras.getString(TelephonyManager.EXTRA_STATE);
19             Log.w(TAG, state);
20             if (state.equals(TelephonyManager.EXTRA_STATE_RINGING)) {
21                 String phoneNumber = extras
22                     .getString(TelephonyManager.EXTRA_INCOMING_NUMBER);
23                 Log.w(TAG, phoneNumber);
24             }
25         }
26     }
27 }
28

```

Figura 9 - PhoneReceiver.java

Logo após deve-se alterar o arquivo ReceiverManifest.xml, para que ficar similar o da figura abaixo:



```

1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     package="br.com.k19.android.cap08_2"
4     android:versionCode="1"
5     android:versionName="1.0" >
6
7     <uses-sdk
8         android:minSdkVersion="8"
9         android:targetSdkVersion="21" />
10
11     <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
12
13     <application
14         android:icon="@drawable/ic_launcher"
15         android:label="@string/app_name"
16         android:theme="@style/AppTheme" >
17         <activity
18             android:name=".MainActivity"
19             android:label="@string/app_name" >
20             <intent-filter>
21                 <action android:name="android.intent.action.MAIN" />
22                 <category android:name="android.intent.category.LAUNCHER" />
23             </intent-filter>
24         </activity>
25
26         <receiver android:name=".PhoneReceiver">
27             <intent-filter>
28                 <action android:name="android.intent.action.PHONE_STATE"></action >
29             </intent-filter >
30         </receiver >
31
32     </application>
33 </manifest>

```

Figura 10 - ReceiverManifest.xml

Quando a aplicação for executada teremos uma tela igual à mostrada na figura abaixo:

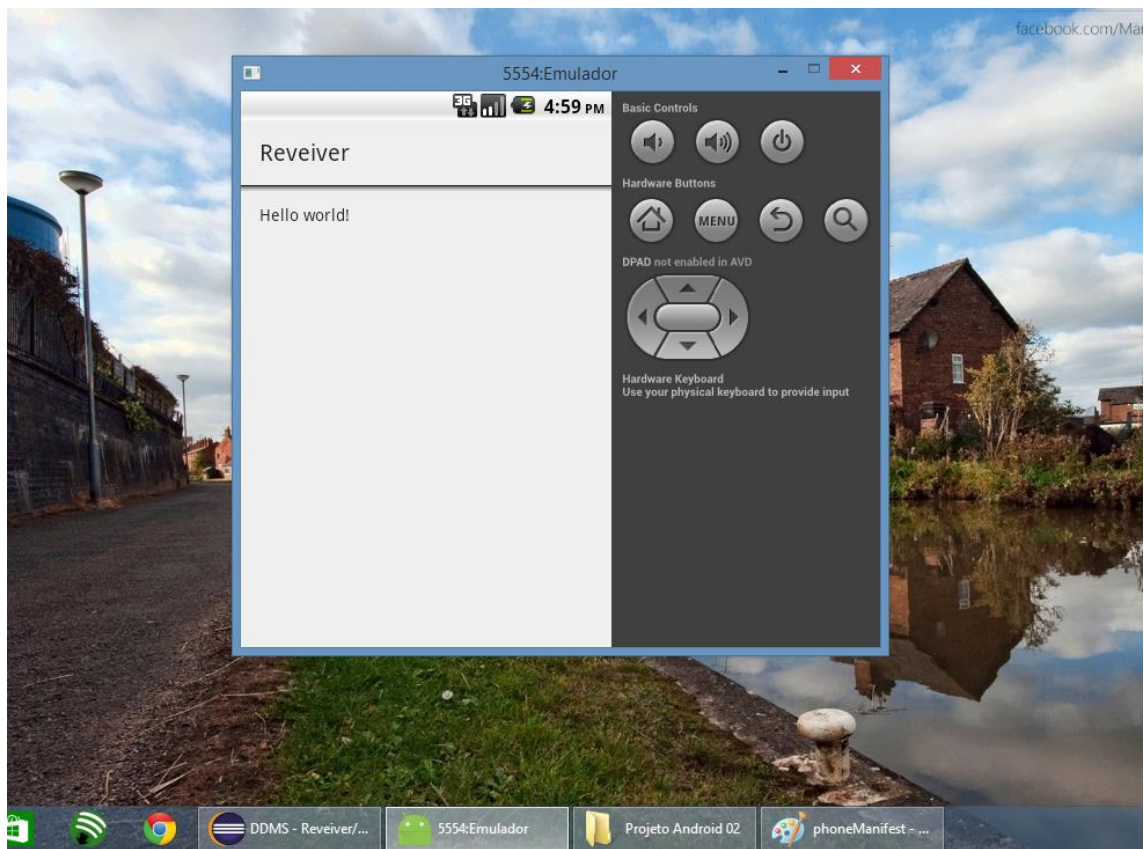


Figura 11 - Aplicação Executando 1

Em seguida mude a perspectiva para DDMS, e encontre a aba Emulator Control. Nesta aba, basta preencher um número e pressionar o botão Call, como mostrado na imagem abaixo:

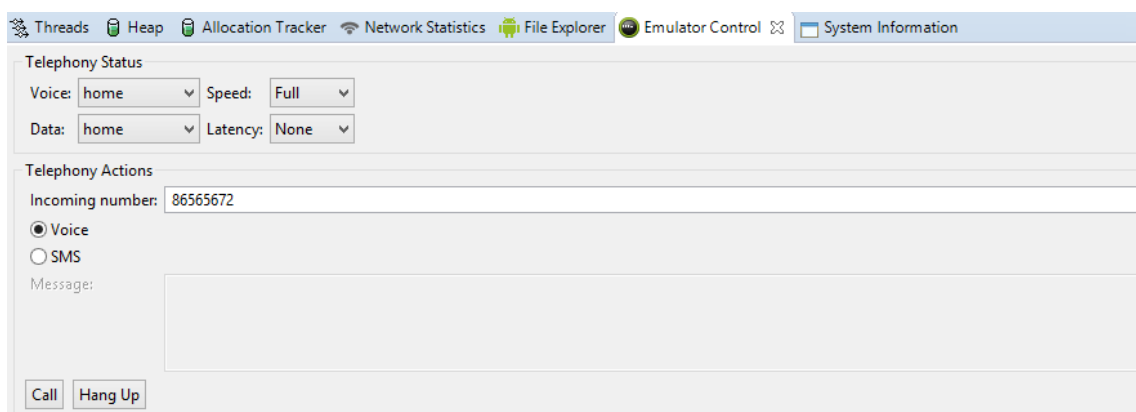


Figura 12 - Configuração de Execução DDML

Após isso, você deve ter um resultado similar ao da figura abaixo:

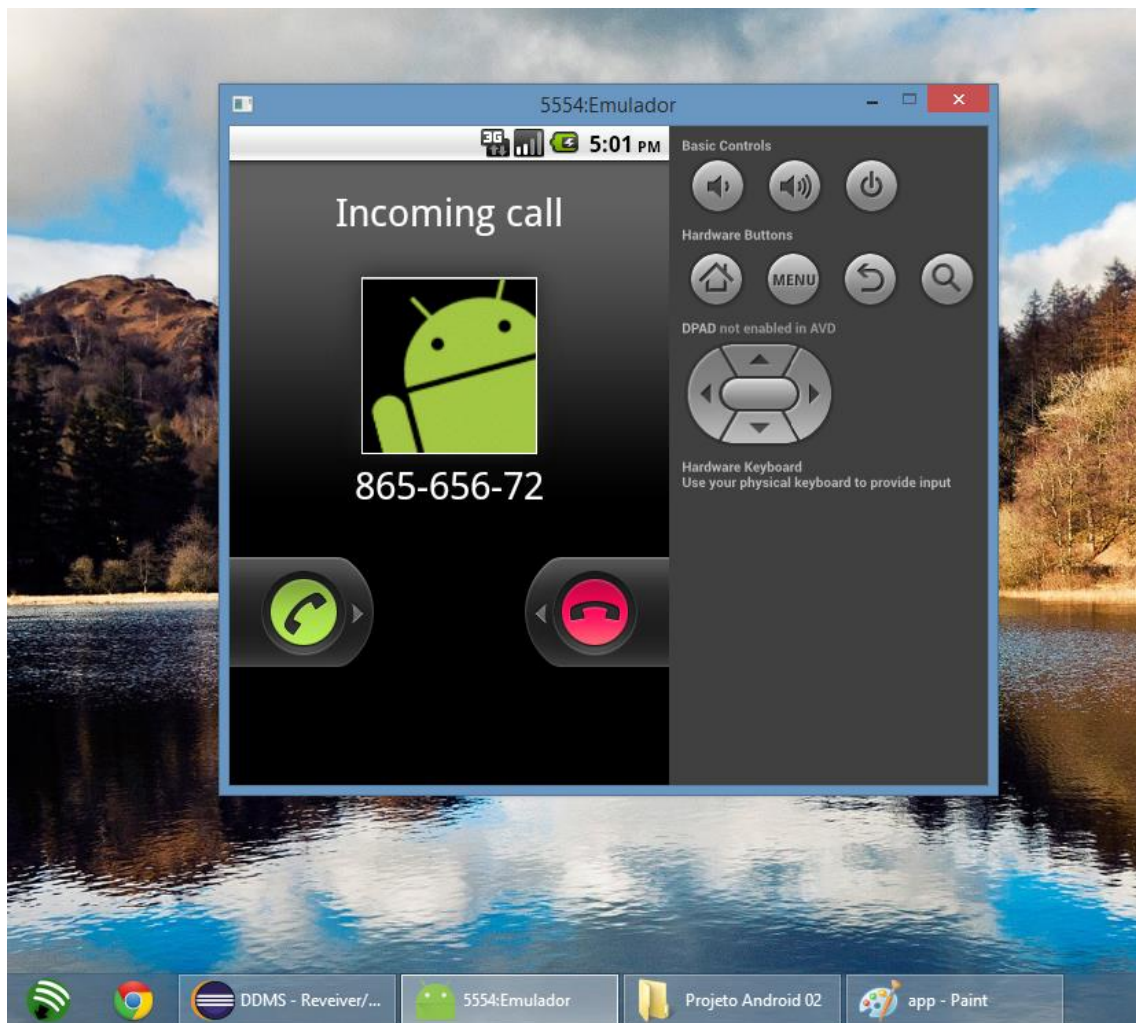


Figura 13 - Aplicação Executando 2