

# Aplicatii mobile pentru afaceri

Master SIA

Lect. Octavian Dospinescu  
2012

# Tematica generala

- Functionalitati GSM in Android
- Primire/interceptare mesaj SMS in mod programatic
- Clasa **BroadcastReceiver** – anatomie generala
- Conceptul de **Intent**
- Drepturi specifice in AndroidManifest.xml
- Directii de viitor

# Cadru de lucru - premise

- interceptarea mesajelor primite de către telefonul nostru de la un alt dispozitiv;
- inițiativa nu mai aparține utilizatorului nostru, ci unui terț care își folosește propriul dispozitiv mobil;
- nu avem un control cert asupra acestei operațiuni, ci trebuie mai degrabă „să fim pe fază” la eventuala primire a unui SMS.

# Notiuni de avut in vedere

- **Intent** = un eveniment care apare in viata aplicatiei ;
- un **BroadcastReceiver** este capabil să intercepteze diverse evenimente de tip **Intent** care apar din mediul extern sau din sistemul de operare;
- putem face asocierea cu notiunea de trigger;

# BroadcastReceiver – anatomie generală

- Un **BroadcastReceiver** implementează metoda abstractă **onReceive()** pentru a procesa **Intent**-urile care sosesc.
- Argumentele metodei sunt un **Context** și un **Intent**.
- Metoda returnează **void**, dar se poate folosi metoda **setResult** pentru a pasa înapoi un anumit rezultat.

# BroadcastReceiver – utilizare uzuala

- În practică, implementarea unui BroadcastReceiver se realizează prin **definirea unei clase derivate** din clasa de bază BroadcastReceiver, în care **definim metoda onReceive() în funcție de necesitățile aplicației**

# BroadcastReceiver – precizari

Spre deosebire de clasa Activity (care are drept **corespondent grafic** un formular afișat pe dispozitivul mobil), clasa **BroadcastReceiver** *nu are o reprezentare grafică vizibilă* pentru utilizator;

Ea funcționează „în fundalul” aplicației, **declanșându-se atunci când apare un Intent corespunzător.**

# BroadcastReceiver – model de implementare

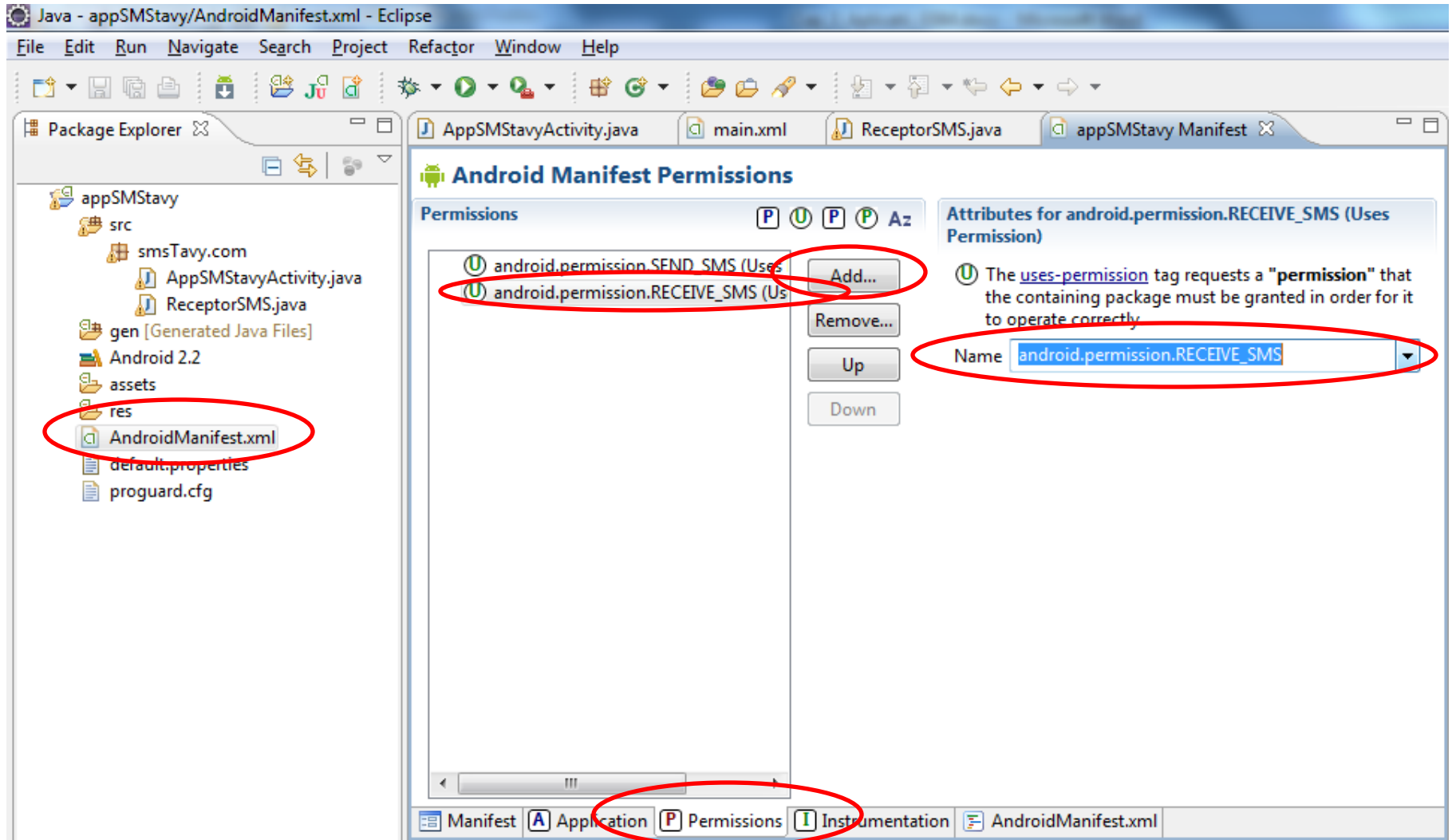
```
public class ReceptorSMS extends BroadcastReceiver {  
    @Override  
    public void onReceive(Context context, Intent intent) {  
        // TODO Auto-generated method stub  
        Log.i("din receiver", "s-a declansat metoda onReceive");  
        if(intent.getAction().equals("android.provider.Telephony.SMS_RECEIVED"))  
        {  
            Log.i("din receiver", "Am primit un SMS!!!");  
            //aici urmeaza sa completam cu cod de prelucrare  
        }  
    }  
}
```



# BroadcastReceiver – precizari importante!!!

- pentru ca aplicația noastră să poată **intercepta** mesaje SMS, este nevoie să **specificăm acest drept** în fișierul AndroidManifest.xml printr-o **clauză de tip <uses-permissions>** similară cu cea care ne-a dat dreptul să trimitem SMS-uri

# BroadcastReceiver – permisiuni!!!



# **BroadcastReceiver** – precizari importante!!!

- pentru ca **BroadcastReceiver-ul** nostru (ReceptorSMS) să funcționeze, el **trebuie să fie declarat de asemenea în fișierul AndroidManifest.xml**

# BroadcastReceiver – declarare in AndroidManifest

The screenshot displays the Eclipse IDE interface for an Android project. The Package Explorer on the left shows the project structure with `AndroidManifest.xml` highlighted. The main editor shows the `AndroidManifest.xml` file with the `Application` tab selected. The `Application Nodes` section lists `.AppSMStavyActivity` and `ReceptorSMS (Receiver)`. The `Attributes for ReceptorSMS (Receiver)` section shows the `Name` attribute set to `ReceptorSMS`. The `Manifest` tab is also highlighted at the bottom.

File Edit Run Navigate Search Project Refactor Window Help

Package Explorer

AppSMStavyActivity.java main.xml ReceptorSMS.java appSMStavy Manifest

### Android Manifest Application

**Application Toggle**

The `application` tag describes application-level components contained in the package, as well as general application attributes

Define an `<application>` tag in the `AndroidManifest.xml`

**Application Attributes**

**Application Nodes** [S] [P] [A] [A] [R] [M] [U] [Az]

- [A] .AppSMStavyActivity
- [R] ReceptorSMS (Receiver)

Add...  
Remove...  
Up  
Down

**Attributes for ReceptorSMS (Receiver)**

[R] The `receiver` tag declares an `android.content.BroadcastReceiver` class that is available as part of the package's application components, allowing the application to receive actions or data broadcast by other applications even if it is not currently running.

Name*	ReceptorSMS	Browse...
Label		Browse...
Description		Browse...
Icon		Browse...
Permission		
Process		Browse...

Manifest [A] Application [P] Permissions [I] Instrumentation [E] AndroidManifest.xml

# BroadcastReceiver – precizari importante!!!

- pentru acest receiver pe care tocmai l-am declarat în fișierul `AndroidManifest.xml` trebuie precizăm și *lista de Intent-uri pe care le „ascultă”*;
- un **BroadcastReceiver** poate intercepta mai multe tipuri de „mesaje” de tip `Intent`, cum ar fi: primirea unui apel telefonic, primirea unui mesaj SMS, primirea unui mail, descărcarea bateriei sub o anumită limită etc.

# **Intent**-uri si filtre (IntentFilter)

În Android este necesar să **mapăm un astfel de BroadcastReceiver cu o listă (un filtru) de Intent-uri.**

Această punere în corespondență se realizează prin intermediul conceptului de **IntentFilter**, care trebuie declarat tot în fișierul **AndroidManifest.xml**.

# Intent-uri si filtre (IntentFilter)

Operațiunea presupune poziționarea pe **ReceptorSMS(Receiver)** și acționarea butonului **Add...** urmată de selectarea opțiunii **IntentFilter**.

După această operațiune, observăm că ReceptorSMS a devenit rădăcina unui mic arbore în care avem o secțiune numită IntentFilter. Ne poziționăm pe această secțiune, acționăm butonul **Add...**, selectăm tipul **Action** și apoi alegem din listă intent-ul numit **android.provider.Telephony.SMS\_RECEIVED**.

# Intent-uri si filtre (IntentFilter)

## Android Manifest Application

### Application Toggle

The [application](#) tag describes application-level components contained in the package, as well as general application attributes

Define an <application> tag in the AndroidManifest.xml

### Application Attributes

#### Application Nodes

S P A A R M U Az

.AppSMStavyActivity (A)  
ReceptorSMS (Receiver) (R)  
Intent Filter (I)  
android.provider.Telephony.SMS\_RECEIVED (A)

Add...  
Remove...  
Up  
Down

#### Attributes for android.provider.Telephony.SMS\_RECEIVED (Action)

Attributes that can be supplied in an AndroidManifest.xml [action](#) tag, a child of the **intent-filter** tag.

Name android.provider.Telephony.SMS\_RECEIVED

Manifest Application Permissions Instrumentation AndroidManifest.xml



# Intent-uri si filtre (IntentFilter)

Prin această operațiune, am precizat de fapt mediului de lucru următorul fapt:

- BroadcastReceiver-ul cu numele ReceptorSMS va „reacționa” atunci când apare un „eveniment” (intent) de tip SMS\_RECEIVED, adică atunci când dispozitivul primește un mesaj de tip SMS.

# Structura fisierului AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="smsTavy.com"
    android:versionCode="1"
    android:versionName="1.0">
    <uses-sdk android:minSdkVersion="8" />
    <uses-permission android:name="android.permission.SEND_SMS"></uses-permission>
    <uses-permission android:name="android.permission.RECEIVE_SMS"></uses-permission>
    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".AppSMStavyActivity"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <receiver android:name="ReceptorSMS">
            <intent-filter>
                <action android:name="android.provider.Telephony.SMS_RECEIVED">
            </action>
            </intent-filter>
        </receiver>
    </application>
</manifest>
```

# Programarea aplicației

Pentru a definiți aplicația, trebuie să completăm metoda **onReceive(...)** astfel încât să obținem numărul de telefon al expeditorului, precum și mesajul text care alcătuiește mesajul primit. Metoda **onReceive(...)** are 2 parametri: un **Context** și un **Intent**.

Ceea ce ne interesează pe noi este conținut într-o formă „împachetată” în acel intent care este primit ca parametru de către metodă ☹.

# Strategia de lucru

- verificăm dacă intent-ul este de tip SMS\_RECEIVED;
- cu ajutorul metodei **getExtras()** obținem „pachetul” (bundle) din cadrul intent-ului primit ca parametru;
- din „pachet” obținem lista de obiecte conținute (Object[]), folosind metoda .get(“pdus”);

# Strategia de lucru - continuare

- definim un **mesaj** de tip **SmsMessage**;
- din clasa **SmsMessage** folosim metoda **createFromPdu** pentru a obține mesajul efectiv din primul obiect din pachet;
- pentru a obține numărul de telefon al expeditorului și mesajul primit, folosim metodele **getOriginatingAddress** și **getMessageBody**;
- afișăm pe ecran datele obținute, prin intermediul unui **Toast**.

# Implementare – model 😊

```
package smsTavy.com;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.telephony.SmsMessage;
import android.util.Log;
import android.widget.Toast;
public class ReceptorSMS extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        // TODO Auto-generated method stub
        Log.i("din receiver", "s-a declansata metoda onReceive");
        if(intent.getAction().equals("android.provider.Telephony.SMS_RECEIVED"))
        {
            Log.i("din receiver", "Am primit un SMS!!!");
            Bundle pachet;
            pachet = intent.getExtras();

            Object[] mesajSosit = (Object[]) pachet.get("pdu");

            SmsMessage mesaj;
            mesaj = SmsMessage.createFromPdu((byte[]) mesajSosit[0]);

            String numar;
            String text;

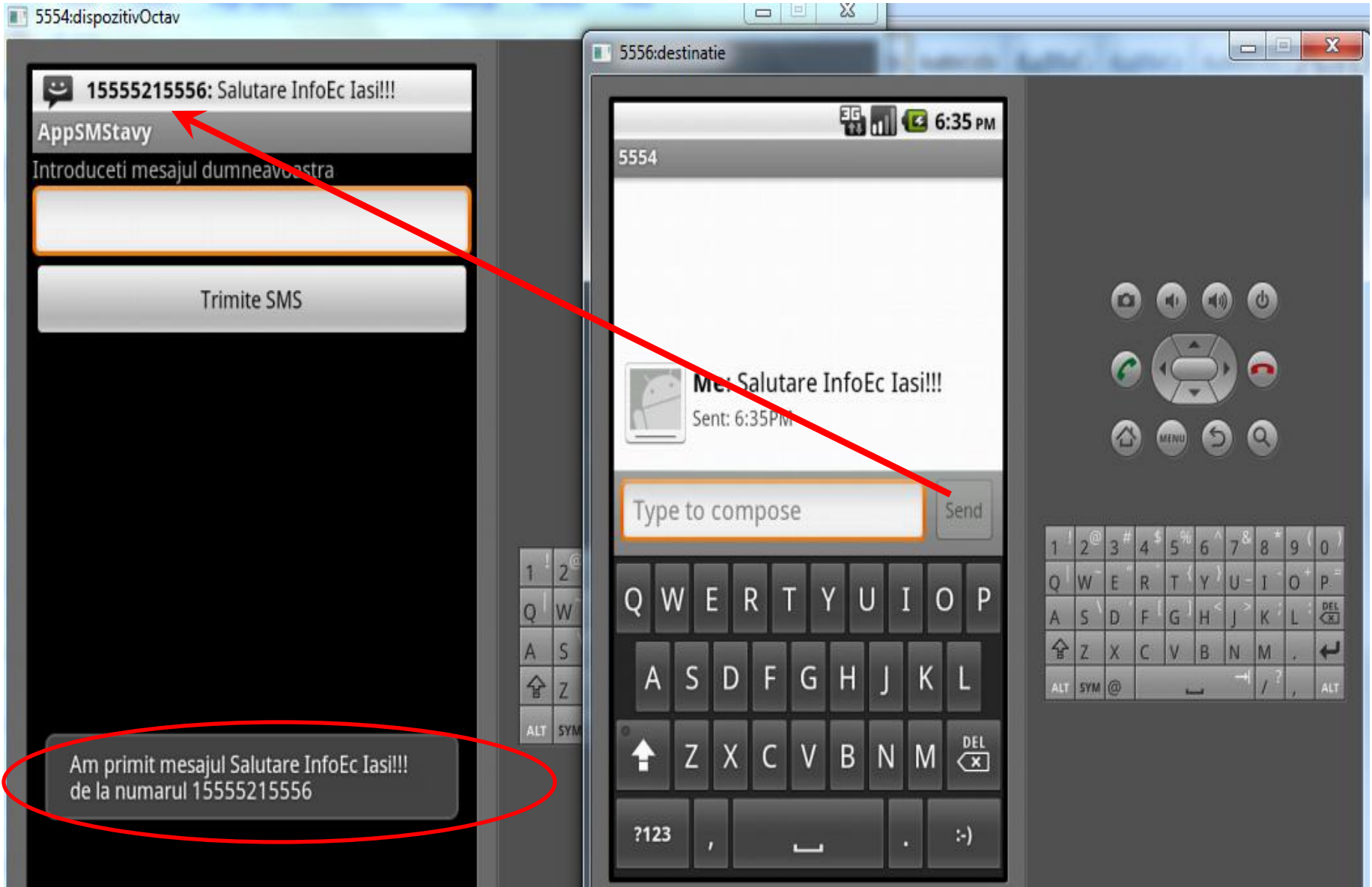
            numar = mesaj.getOriginatingAddress();
            text = mesaj.getMessageBody();

            Toast notificare;
            notificare = Toast.makeText(context, "Am primit mesajul " + text + " de la
numarul " + numar, Toast.LENGTH_LONG);

            notificare.show();

            Log.i("din receiver", numar + " " + text);
        }
    }
}
```

# Rezultat final



# Directii de viitor

- Monitorizarea copiilor de catre parinti 😊
  1. părintele trimite un SMS cu un mesaj special către telefonul copilului;
  2. aplicația instalată pe acel telefon trimite părintelui un răspuns SMS în care furnizează locația geografică (latitudine și longitudine) în care se află copilul;
  3. mai în glumă, mai în serios, ne putem imagina că aceste locații pot fi: școala, ~~discoteca~~ 😊, biblioteca etc.