

פרק 20 ב - טלפוניה

בפרק זה נלמד על בירור מצב הטלפון; שיחות חיוג, קבלת שיחה נכנסת, הודעות טקסט SMS, שליחה וקבלת הודעה.

אנדרואיד מספקת חבילה עם מחלקות שמטפלות בכל נושא הטלפוניה.

נעבור על נושאים כגון:

1. מנהל הטלפוניה TelephonyManager
2. קבלת תכונות ומצב הטלפון
3. זיהוי שיחה נכנסת
4. שליחת SMS ע"י Intent או ע"י מנהל המסרונים SmsManager
5. הרשאות הנחוצות לחיוג, קבלת שיחה, שליחת SMS וקבלת SMS


ועוד:

1. הדגמת השימוש במאזין מצב הטלפון PhoneStateListener ושימוש BroadcastReceiver
2. ההדגמה תכלול ביצוע שיחות זיהוי שיחה יוצאת
3. זיהוי שיחה נכנסת
4. שליחת SMS וקבלתו
5. תרגיל שימוש בשירות קיים במערכת שבה מתנהל הטלפון או תרגיל ל BroadcastReceiver ושימוש בהרשאות

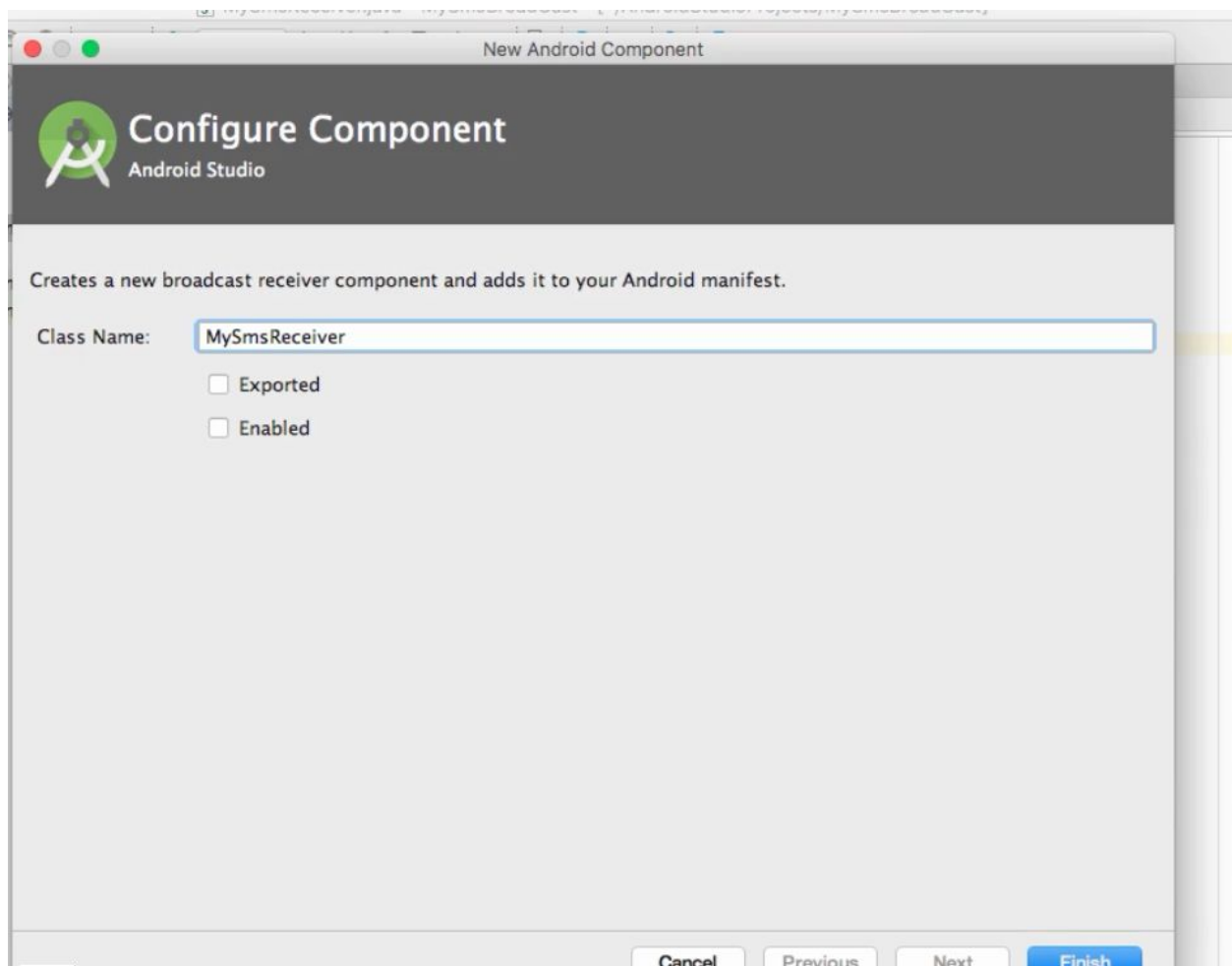
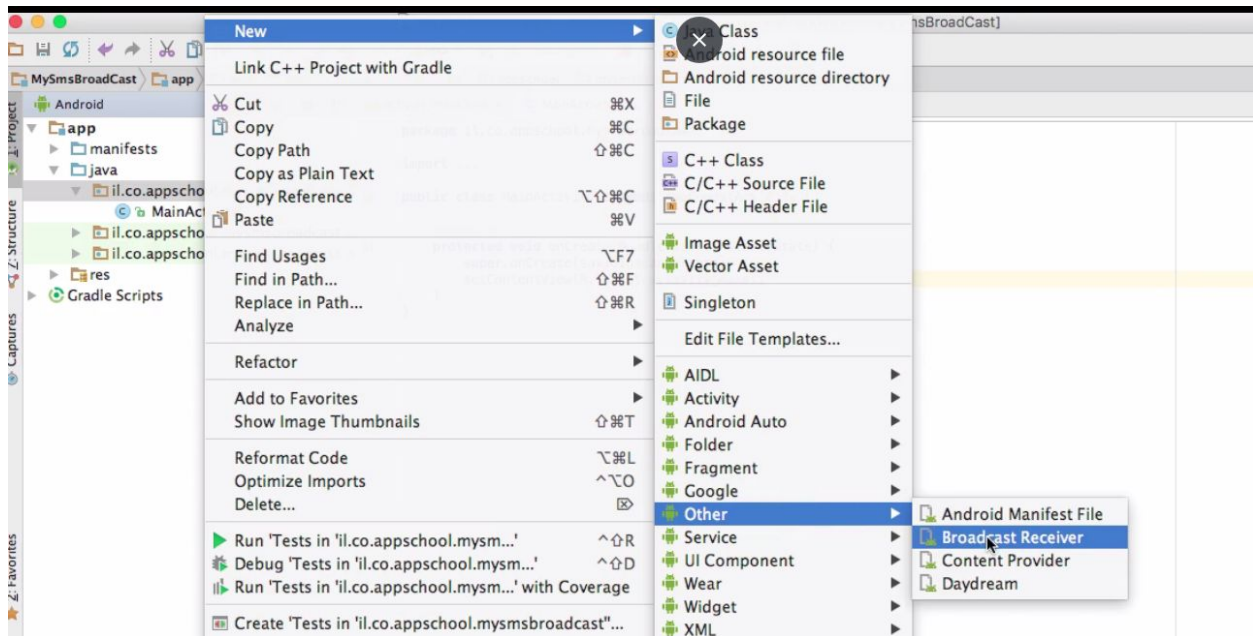
דוגמה 1 - BroadcastReceiver שמזהה קבלת SMS:

שלב 1 - מתן הרשאות במניפסט:

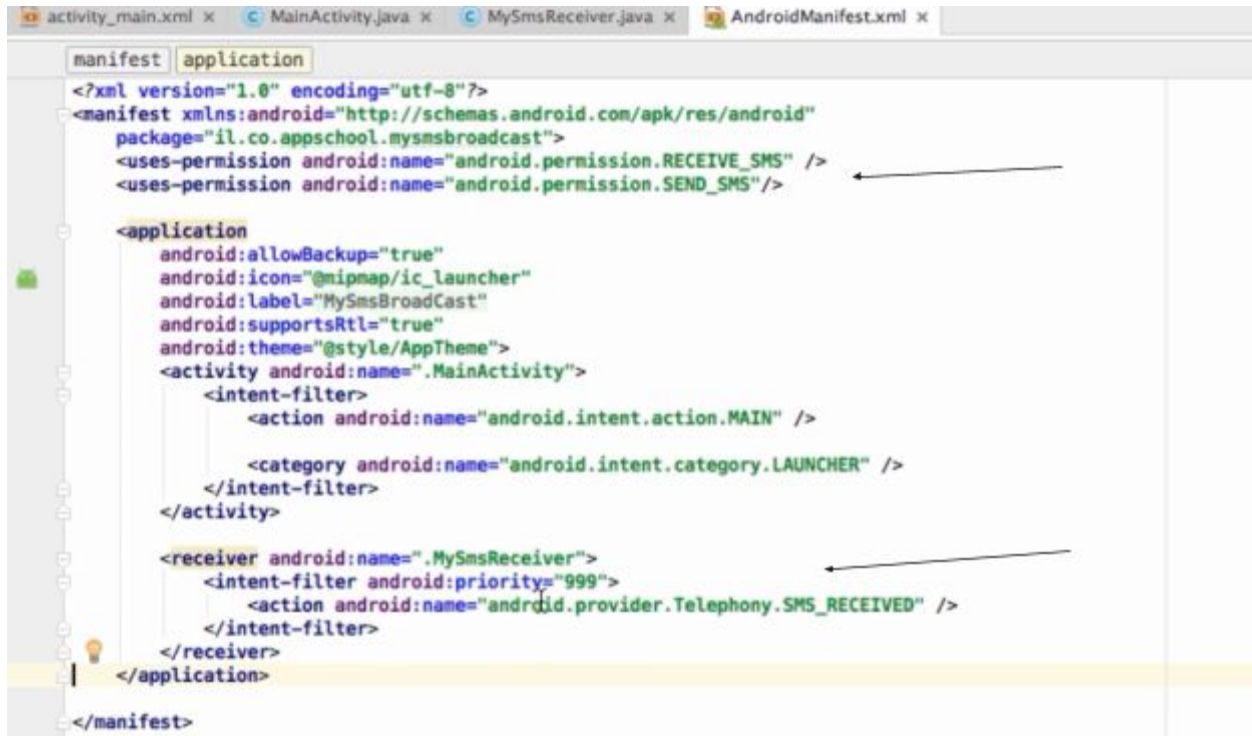
```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android" android:
3   <uses-sdk android:minSdkVersion="10" />
4
5   <uses-permission android:name="android.permission.RECEIVE_SMS" />
6   <uses-permission android:name="android.permission.SEND_SMS" />
7
8   <application android:allowBackup="true" android:icon="@mipmap/icon" andro
9   </application>
10 </manifest>
11
```



שלב 2 - יצירת קובץ מסוג BroadcastReceiver:



שלב 3 - מתן הרשאות במניפסט:



```
activity_main.xml x MainActivity.java x MySmsReceiver.java x AndroidManifest.xml x
manifest application
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="il.co.appschool.mysmsbroadcast">
    <uses-permission android:name="android.permission.RECEIVE_SMS" />
    <uses-permission android:name="android.permission.SEND_SMS" />
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="MySmsBroadCast"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <receiver android:name=".MySmsReceiver">
            <intent-filter android:priority="999">
                <action android:name="android.provider.Telephony.SMS_RECEIVED" />
            </intent-filter>
        </receiver>
    </application>
</manifest>
```

```
public class MySmsReceiver extends BroadcastReceiver {

    public MySmsReceiver() {

    }

}
```

@Override

```
public void onReceive(Context context, Intent intent) {

    // TODO: This method is called when the BroadcastReceiver is receiving
    // an Intent broadcast.

    if (intent.getAction().equals("android.provider.Telephony.SMS_RECEIVED")) {

        startBroadcast(intent, context);

    }

}
```

נבדוק שה action של ה intent
הוא של ה IntentFilter שלנו

נקרא לפונקציה
שנמש בהמשך

```
public void startBroadcast(Intent intent, Context context) {

    Bundle myBundle = intent.getExtras();
    SmsMessage[] messages = null;
    String strMessage = "";

    if (myBundle != null) {

        Object[] pdu = (Object[]) myBundle.get("pdu");
        messages = new SmsMessage[pdu.length];

        String number = "";
        for (int i = 0; i < messages.length; i++) {

            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {

                String format = myBundle.getString("format");
                messages[i] = SmsMessage.createFromPdu((byte[]) pdu[i], format);

            } else {

                messages[i] = SmsMessage.createFromPdu((byte[]) pdu[i]);

            }

            number = messages[i].getOriginatingAddress();
            strMessage += "SMS From: " + messages[i].getOriginatingAddress();
            strMessage += " : ";
            strMessage += messages[i].getMessageBody();
            strMessage += "in";

        }

    }

}
```

pdu הוא Extra של מערך
אובייקטים של כל הסמים

נקבל את המערך

ניצור מערך messages בגודל
המערך של ההודעות שקיבלנו

נבדוק מה הגרסה של ה SDK, כיוון
שהחל מגרסה מסוימת הקוד יהיה שונה

נקבל את הפורמט המתאים

ניצור את ההודעה ונכניס
אותה למערך messages

נשלף את מספר הטלפון שממנו נשלחה ההודעה

בהודעה יופיע מספר הטלפון
של השולח, ואת תוכן ההודעה

נ/ מבצע אנטר (ירידה בשורה)

```
Log.e("SMS", strMessage);
```

```
Toast.makeText(context, strMessage, Toast.LENGTH_SHORT).show();
```

```
SmsManager smsManager = SmsManager.getDefault();
```

```
smsManager.sendTextMessage(number, null, "hello world", null, null);
```

```
if (strMessage.length() > 0) {
```

```
Intent intent1 = new Intent(context, InfoActivity.class);
```

```
intent1.putExtra("info", strMessage);
```

```
intent1.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
```

```
context.startActivity(intent1);
```

```
smsManager.sendTextMessage(number, null, "hello uzi", null, null);
```

```
}  
//end of if
```

```
}
```

יוצג Toast של הודעה

נתחבר למנהל ההודעות

נשלח הודעה חזרה למספר שממנו נשלחה ההודעה

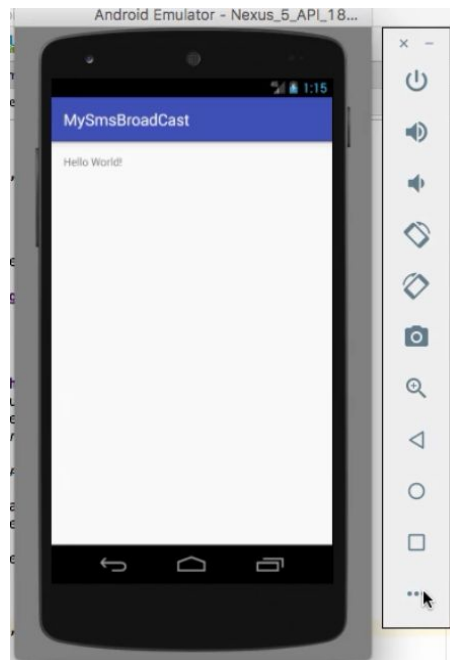
נעבור למסך InfoActivity (ניצור אותו בהמשך)

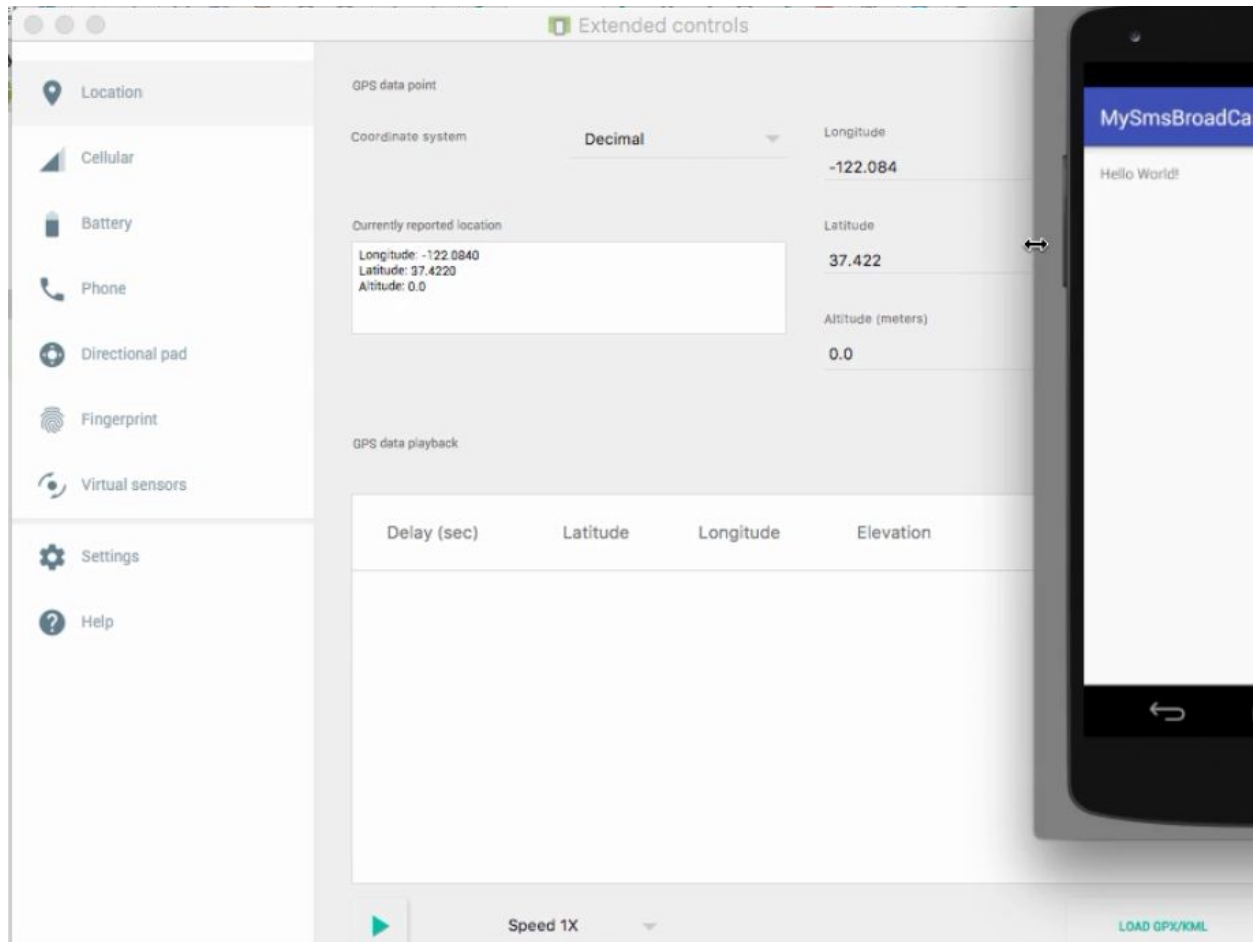
בעזרת הפלאג נוכל לפתוח את האפליקציה במידה והיא סגורה, בשביל להפעיל את האינטנט

נשלח בחזרה הודעה עם התוכן "hello uzi" שהשולח שלה הוא ה phone ששלפנו (לא חובה)

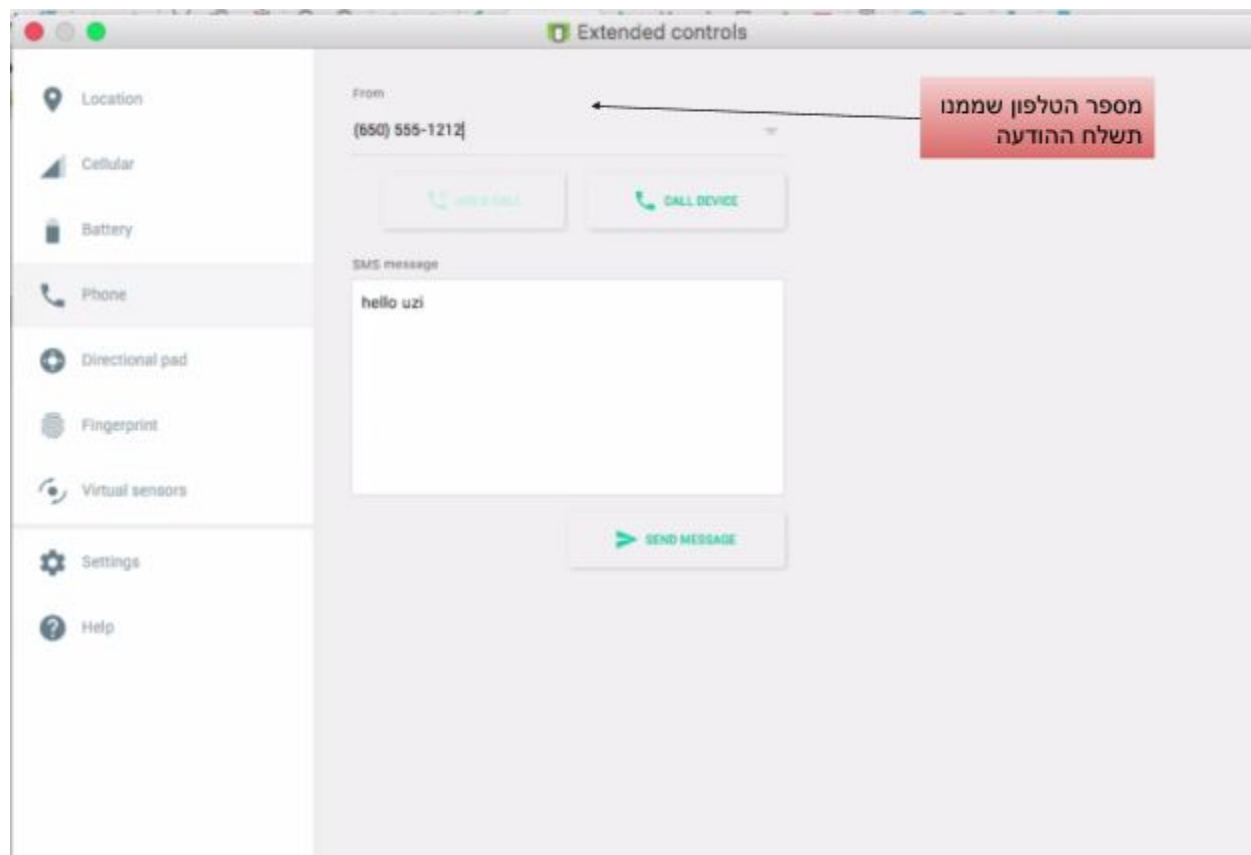
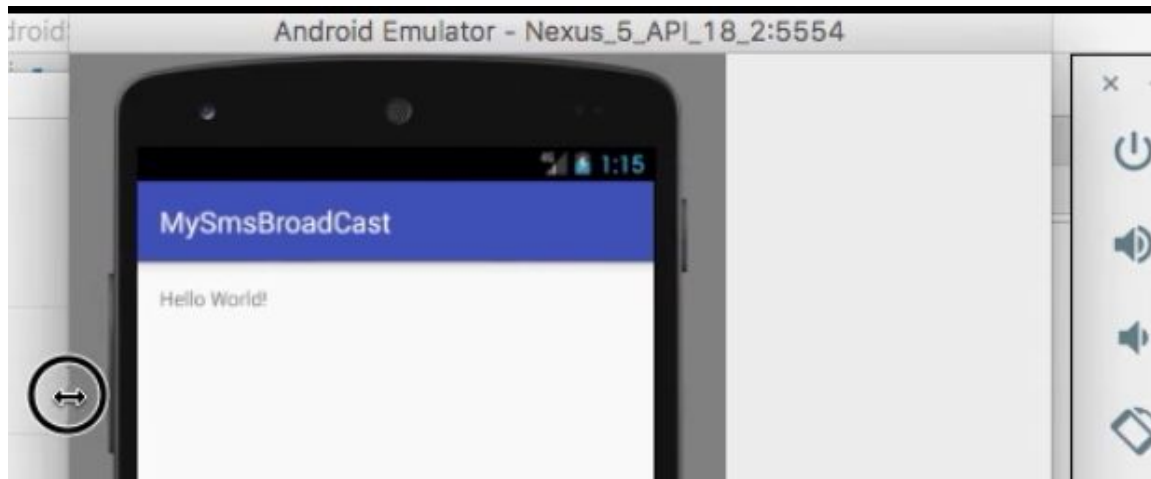
נוכל לשלוח הודעה גם מהאימולטור עצמו:

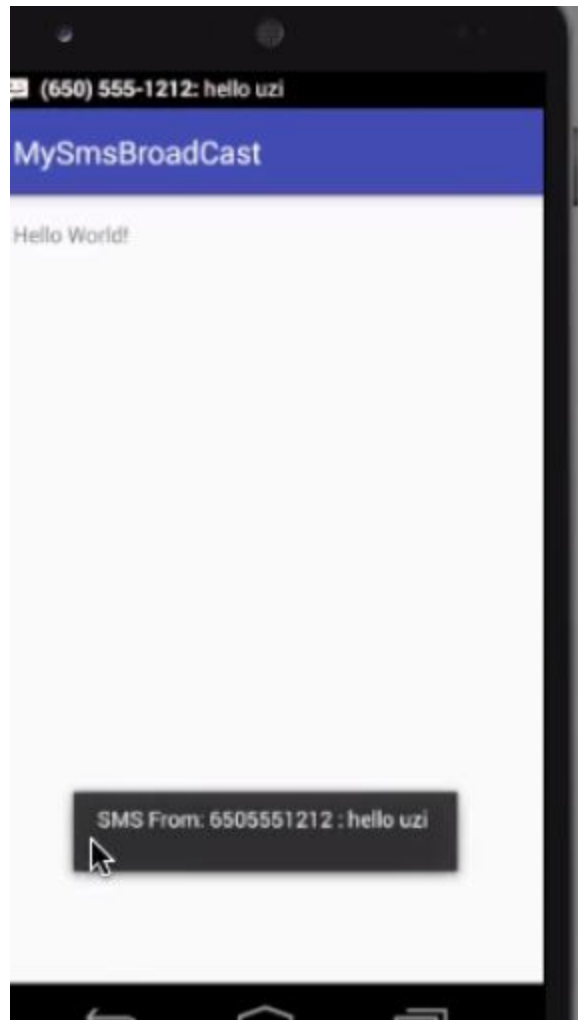
לחיצה על שלושת הנקודות תקפיץ לנו מסך:





בעזרת הרחבה של האימולטור תוכלו לראות את מספר הטלפון של
 האימולטור שלכם: (במקרה שלנו 5554)





שלב 3 - עיצוב InfoActivity:

```
RelativeLayout | TextView
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_info"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    tools:context="il.co.appschool.mysmsbroadcast.InfoActivity">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="20sp"
        android:id="@+id/tvInfo"
        |
    />
</RelativeLayout>
```

שלב 4 - עריכת InfoActivity:

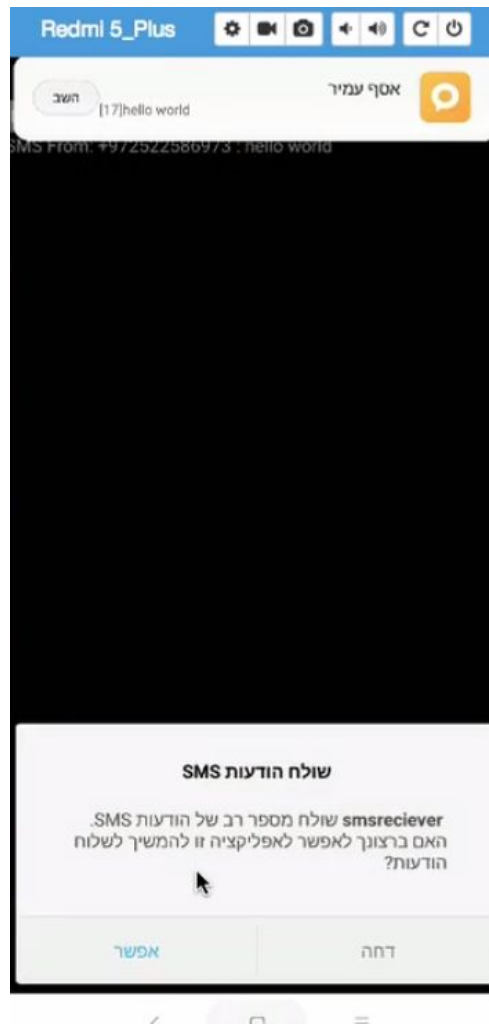
```
public class InfoActivity extends AppCompatActivity {  
  
    TextView tv;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_info);  
  
        tv = (TextView)findViewById(R.id.tvInfo);  
  
        Intent intent = getIntent();  
  
        if(intent!=null) {  
            Bundle bundle = intent.getExtras();  
  
            if (bundle != null) {  
                String info = bundle.getString("info");  
                Log.d("asaf", info);  
                tv.setText(info);  
            }  
        }  
    }  
}
```

תוכלו להריץ עכשיו את האפליקציה.

שימו לב שלא נגענו בכלל במסך MainActivity ובכל מקרה בכל פעם שנקבל הודעה נעבור למסך InfoActivity בזכות השימוש ב BroadcastReciever.

שימו לב! אם תשלחו הודעה מהטלפון של עצמכם תיכנסו לפעולה רקורסיבית, ולכן תיתקעו בלולאה.

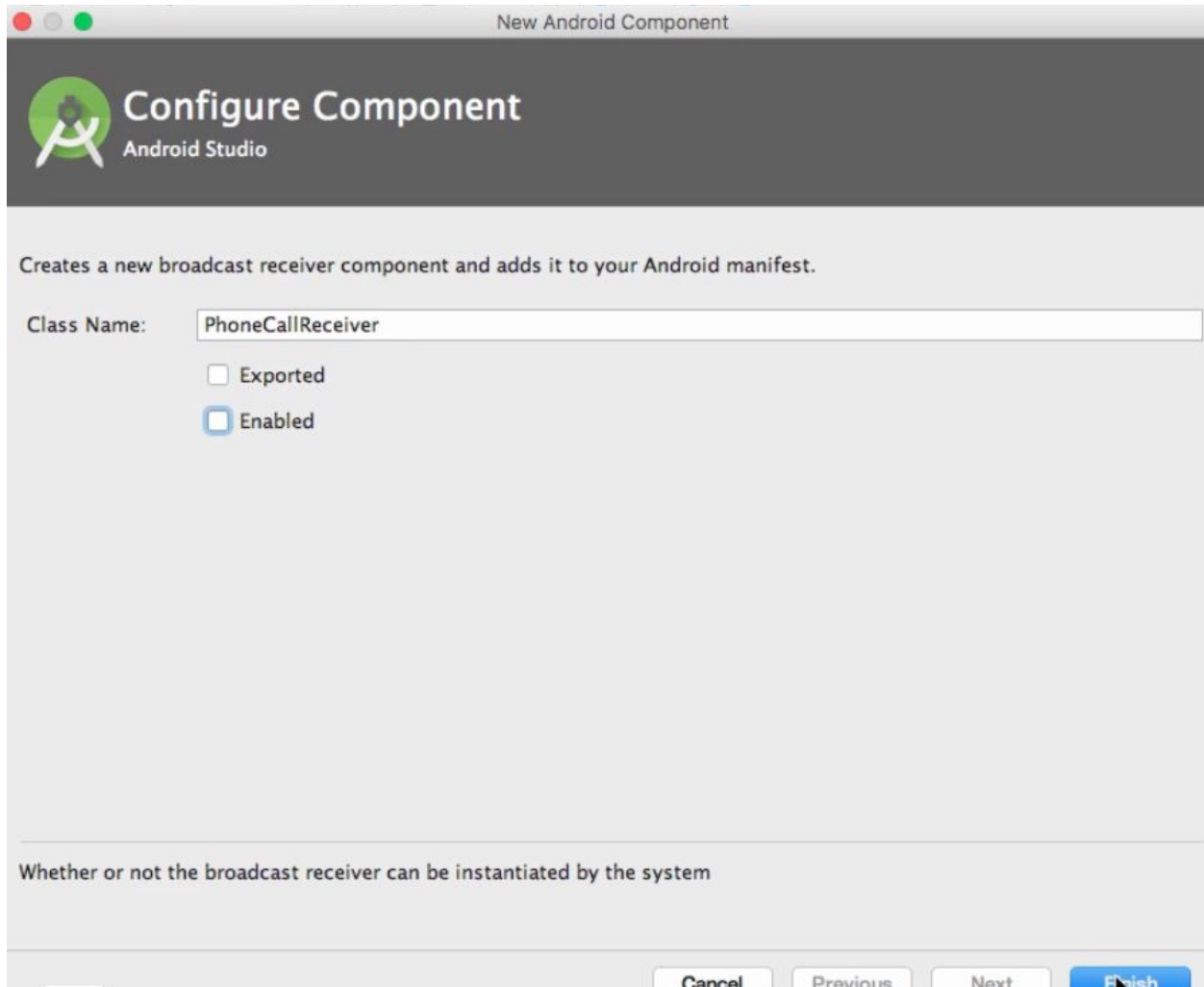
בשביל לבדוק את ההרצה כדאי לכם לשלוח את ההודעה ממספר טלפון אחר.



ברודקאסט לזיהוי שיחה נכנסת ויוצאת

Copyright © 2019 appSchool. Powered by appSchool.co.il

שלב 1 - ניצור ברודקאסט חדש:



New Android Component

Configure Component
Android Studio

Creates a new broadcast receiver component and adds it to your Android manifest.

Class Name:

☐ Exported

☒ Enabled

Whether or not the broadcast receiver can be instantiated by the system

Cancel Previous Next Finish

שלב 2 - עריכת הברודקאסט:

```
public class PhoneCallReceiver extends BroadcastReceiver {
```

```
@Override
```

```
public void onReceive(final Context context, Intent intent) {
```

```
    TelephonyManager manager = (TelephonyManager) context.getSystemService(Context.TELEPHONY_SERVICE);
```

```
    manager.listen(new PhoneStateListener() {
```

```
        @Override
```

```
        public void onCallStateChanged(int state, String incomingNumber) {
```

```
            super.onCallStateChanged(state, incomingNumber);
```

```
            if (state == TelephonyManager.CALL_STATE_RINGING) {
```

```
                Notification.Builder builder = new Notification.Builder(context);
```

```
                builder.setContentText(incomingNumber);
```

```
                builder.setSmallIcon(android.R.drawable.star_on);
```

```
                Notification notification = builder.build();
```

```
                NotificationManager notificationManager = (NotificationManager) context.getSystemService(Context.NOTIFICATION_SERVICE);
```

```
                notification.flags |= Notification.FLAG_AUTO_CANCEL;
```

```
                notification.defaults = Notification.DEFAULT_VIBRATE | Notification.DEFAULT_LIGHTS | Notification.DEFAULT_SOUND;
```

```
                notificationManager.notify(0, notification);
```

```
            }
```

```
        }
```

```
    }, PhoneStateListener.LISTEN_CALL_STATE);
```

```
}
```

```
}
```

במחלקה PhoneStateListener נחליט מה יעשה
ה manager בכל מצב (state) שבו הטלפון יהיה

נבדוק אם הטלפון מצלצל

נערוך את הטקסט להיות
המספר שמתקשר

האייקון של שיחה נכנסת יהיה תמונה
שהעלינו מהמחשב לתיקיה drawable

נתחבר ל notificationManager

המשתמש יוכל לבטל את ה notification

ניתן ל notification רטט, אורות וקול

ה manager יאזין ל PhoneStateListener.LISTEN_CALL_STATE

שלב 3 - עריכת הוספת הרשאה במניפסט:

```
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
```

```
<receiver android:name=".PhoneCallReceiver">
```

```
    <intent-filter>
```

```
        <action android:name="android.intent.action.PHONE_STATE" />
```

```
    </intent-filter>
```

```
</receiver>
```

```
</application>
```

```
</manifest>
```

נריץ את האפליקציה:

