ITU-FAO-DOA-TRCSL

Training on

"Innovation & Application Development for E-Agriculture"

Runtime Permissions

11th-15th December 2017

Peradeniya, Sri Lanka

Shahryar Khan & Imran Tanveer, ITU Experts

Runtime Permissions

Runtime permissions is a very useful feature of Android that was introduced in Android 6.0 (MarshMallow). According to this feature you grant the permission to the app while they are running on device not during app installation. So user can cancel any of the permission at any time after the application has been installed, while allowing other permissions to the application. They have divided the permissions in 2 groups

- Normal Permissions
- Dangerous Permissions

Normal Permissions:

These permissions are granted automatically if you ask them through the manifest of the Android applications. They do not harm or affect the privacy of the user.

Dangerous Permissions:

These permissions can harm the privacy of the user. They allow the app to access your confidential data. So to take these permissions you need to ask user explicitly to grant these for your application. Some of these permissions are CAMERA, READ_CONTACTS, CALL_PHONE etc.

In the next section you will use CALL_PHONE permission to make a phone call. This is one of the dangerous permission so you need to take its permission at runtime.

Strings.xml

```
<resources>
   <string name="app_name">Runtime Permissions</string>
</resources>
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
>
```

```
<TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Runtime Permissions"
      android:textStyle="bold"
      android:textSize="30sp"
      android:layout_centerHorizontal="true"
      app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintLeft_toLeftOf="parent"
      app:layout_constraintRight_toRightOf="parent"
      app:layout_constraintTop_toTopOf="parent" />

   <Button
      android:id="@+id/buttonCall"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="call +923336407423"
      android:layout_centerInParent="true"
      />


</RelativeLayout>
```

MainActivity.java

```
package mixare.org.callintentwithpermissions;

import android.Manifest;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.net.Uri;
import android.os.Bundle;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.List;
```

```
public class MainActivity extends AppCompatActivity {

   public static final int MULTIPLE_PERMISSIONS = 10; // code you want.

   String[] permissions = new String[]{
         Manifest.permission.CALL_PHONE
   };

   public void onCreate(Bundle savedInstanceState) {

      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

      Button button = (Button) findViewById(R.id.buttonCall);

      checkPermissions();

      // add button listener
      button.setOnClickListener(new View.OnClickListener() {

         @Override
         public void onClick(View arg0) {


               Intent callIntent = new Intent(Intent.ACTION_CALL);
               callIntent.setData(Uri.parse("tel:03336407423"));
               if (ActivityCompat.checkSelfPermission(arg0.getContext(),
Manifest.permission.CALL_PHONE) != PackageManager.PERMISSION_GRANTED) {

                  return;
               }
               startActivity(callIntent);
            }

      });

   }

   private  boolean checkPermissions() {
      int result;
      List<String> listPermissionsNeeded = new ArrayList<>();
```

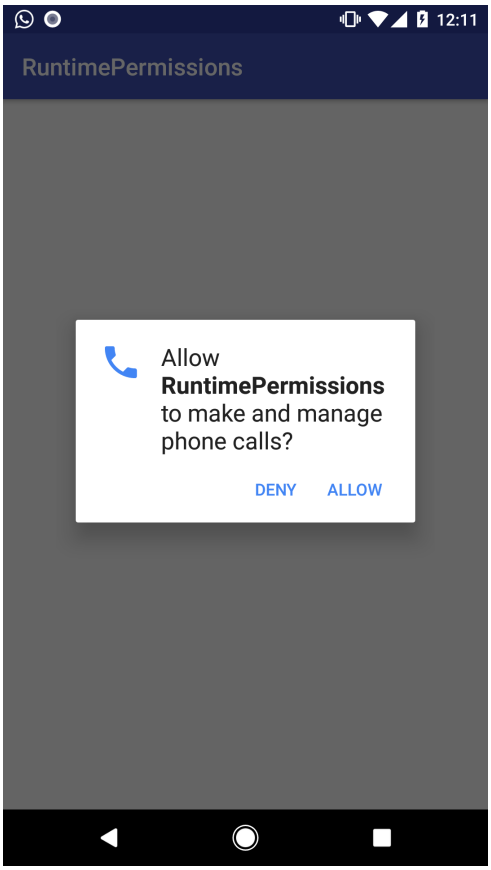
```
for (String p:permissions) {

result = ContextCompat.checkSelfPermission(this,p);
        if (result != PackageManager.PERMISSION_GRANTED) {
          listPermissionsNeeded.add(p);
        }
      }
      if (!listPermissionsNeeded.isEmpty()) {
        ActivityCompat.requestPermissions(this, listPermissionsNeeded.toArray(new
String[listPermissionsNeeded.size()]),MULTIPLE_PERMISSIONS );
        return false;
      }
      return true;
   }


   @Override
   public void onRequestPermissionsResult(int requestCode, String permissions[], int[]
grantResults) {
      switch (requestCode) {
        case MULTIPLE_PERMISSIONS:{
          if(grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED){
            // permissions granted.
            Toast.makeText(this, "Permission granted now you can make a call",
Toast.LENGTH_LONG).show();
          } else {
            Toast.makeText(this, "Oops you just denied the permission",
Toast.LENGTH_LONG).show();
            // no permissions granted.
          }
          return;
        }
      }
   }

}
```


7:00
Runtime Permissions
Runtime Permissions
CALL +923336407423


12:11
RuntimePermissions
Allow
RuntimePermissions
to make and manage
phone calls?
DENY
ALLOW