App development für Android

Localization

Prof. Dr. Paul Rawiel

Geocoordinates

- New project with empty Activity
- Project Name: Localization
- Package Name: de.cursoufpr.localization
- Select Minimum SDK API

→ create project

Project structure as shown before

Important nodes

- java for code
- res for layout

Prepare the Layout

- No textfield needed
- What we need?
  - a button for the start of the tracking
  - A button to stop the tracking
- now Permissions are needed

Permissions

- Geokoordinates are considered sensible personal data
- → Permissions needed

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

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

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

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

Geocoordinates

- A system service provides Geocoordinates
- → LocationManager
  - Create an attribute for a LocaionManager

```
private lm LocationManager;
```

  - Create an object in the onCreate method

```
lm = (LocationManager) getSystemService(LOCATION_SERVICE);
```

- Listener
  - Analog to sensors
  - LocationListener – an Interface, that has to be implemented
  - Method onLocationChanged(Location location) { … }
    - storage of the new location in einer Arraylist of locations

Request of Permissions

- In Android 6 or higher the user must agree to Permissions
- Start button only enabled, if the necessary permissions are granted
- Request Permissions

```
if(ContextCompat.checkSelfPermission(this,
Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
PackageManager.PERMISSION_GRANTED)  {
findViewById(R.id.start).setEnabled(false);
ActivityCompat.requestPermissions(this, new String[]{
Manifest.permission.WRITE_EXTERNAL_STORAGE,
Manifest.permission.ACCESS_COARSE_LOCATION,
Manifest.permission.ACCESS_FINE_LOCATION,
Manifest.permission.ACCESS_NETWORK_STATE}, REQUESTCODE_PERMISSIONS);
}
```

Request Permissions

- The class ActivityCompat implements Interfaces, so that it is mandatory to handle the result of the method call requestPermissions
- A corresponding function has to be implemented:

```
@Override
public void onRequestPermissionsResult(int requestCode, String[]
permissions, int[] grantResults) {
if(requestCode==REQUESTCODE_PERMISSIONS && permissions.length>0 &&
grantResults[0]==PackageManager.PERMISSION_GRANTED) {
findViewById(R.id.start).setEnabled(true);        }    }
```

Permissions

- To protect private data
- Data relevant to security
  - System state
  - User contacts
- Actions relevant to security
  - Connection to another device
  - Audio records

Permissions

- Does the App need Permissions?
- If yes, they have to be announced
- In runtime, they need to be requested from the user
- If the user declines, a corresponding text maybe shown
- The App has to react to the users input

Permissions Workflow

- Declaration in the Manifest

```
<manifest ...>
    <uses-permission android:name="android.permission.CAMERA"/>
    <application ...>
        ...
    </application>
</manifest>
```

- Check, if the user already granted the Permission

```
if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.CAMERA)
                == PackageManager.PERMISSION_GRANTED) {
            // Permission granted, Camera can be used
```

Permission Workflow

```
ActivityCompat.requestPermissions(this,
String[] permissions, REQUESTCODE_PERMISSIONS)
```

- A requestPermissions call has to be handled

```
@Override
public void onRequestPermissionsResult(int
requestCode, String[] permissions, int[]
grantResults) {
…
}
```