UNIVERSITÀ DEGLI STUDI DE L'AQUILA

disim

Mobile Programming Laboratory

ANDROID
Preferences

Teachers

Ing. Tarquini Francesco, Ph.D

Ph.D in Computer Science Engineering

francesco.tarquini@univaq.it

Ing. D’Errico Leonardo

Ph.D Student in Computer Science Engineering

leonardo.derrico@graduate.univaq.it

Teaching Materials

Available on MOODLE platform

http://www.didattica.univaq.it

Google Drive Repository

https://drive.google.com/drive/folders/1ISqZfn0i9Ub3eWNXbvW00rd0hD9ya8OL?usp=sharing

Topics

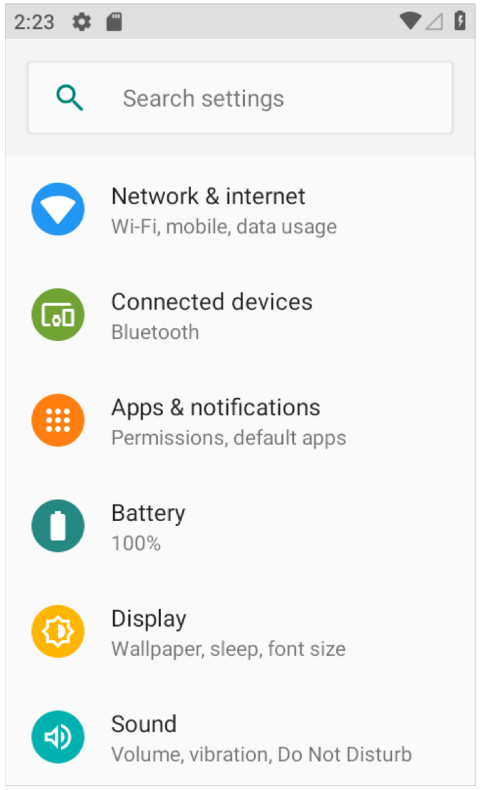
- Settings
- SharedPreferences

Settings

Apps often include settings that allow users to modify app features and behaviors.

For example, some apps allow users to specify whether notifications are enabled or specify how often the app syncs data with the cloud.

If you want to provide settings for your app, you should use Android's **Preference** APIs to build an interface that's consistent with the user experience in other Android apps (including the system settings).



2:23

Search settings

Network & internet
Wi-Fi, mobile, data usage

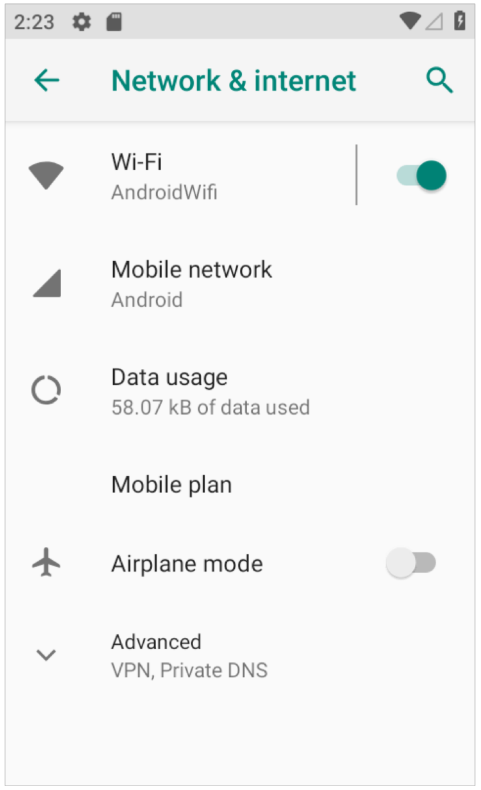
Connected devices
Bluetooth

Apps & notifications
Permissions, default apps

Battery
100%

Display
Wallpaper, sleep, font size

Sound
Volume, vibration, Do Not Disturb



2:23

Network & internet

Wi-Fi
AndroidWifi

Mobile network
Android

Data usage
58.07 kB of data used

Mobile plan

Airplane mode

Advanced
VPN, Private DNS

Settings

A Preference objects is the building block for a single setting.

Each Preference appears as an item in a list and provides the appropriate UI for users to modify the setting.

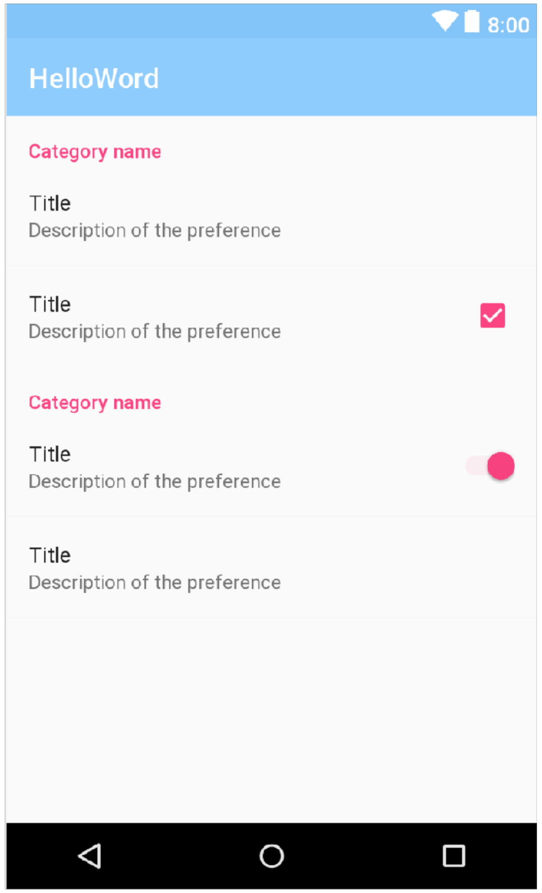
A few of the most common preferences are:

- CheckBoxPreference: Shows a checkbox for a setting that is either enabled or disabled
- ListPreference: Opens a dialog with a list of radio buttons
- EditTextPreference: Opens a dialog with an EditText
- SwitchPreference: Shows a switch for a setting that is either enabled or disabled

Settings

The developer has to create a XML file in res/xml/ folder.

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android">

  <PreferenceCategory android:title="Category name">

    <EditTextPreference
      android:key="option key 1"
      android:summary="Description of the preference"
      android:title="Title" />

    <CheckBoxPreference
      android:defaultValue="true"
      android:key="option key 2"
      android:summary="Description of the preference"
      android:title="Title" />
  </PreferenceCategory>

  <PreferenceCategory android:title="Category name">
    <SwitchPreference
      android:defaultValue="true"
      android:key="option key 1"
      android:summary="Description of the preference"
      android:title="Title" />

    <EditTextPreference
      android:key="option key 1"
      android:summary="Description of the preference"
      android:title="Title" />
  </PreferenceCategory>

</PreferenceScreen>
```


8:00

HelloWord

Category name

Title
Description of the preference

Title
Description of the preference

Category name

Title
Description of the preference

Title
Description of the preference

Settings

The second step is to create a Java class to handle the layout.

This class is a Screen, thus an Activity (deprecated)

```
public class Settings extends PreferenceActivity {

  @Override
  public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      addPreferencesFromResource(R.xml.settings);
  }
}
```

Settings

or a Fragment.

```
public class Settings extends AppCompatActivity {

  @Override
  public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);

      // Display the fragment as the main content.
      getFragmentManager().beginTransaction()
              .replace(android.R.id.content, new SettingsFragment())
              .commit();
  }

  public static class SettingsFragment extends PreferenceFragment {
      @Override
      public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);

          // Load the preferences from an XML resource
          addPreferencesFromResource(R.xml.settings);
      }
  }
}
```

Settings

Each Preference the developer adds has a corresponding **key-value pair** that the system uses to save the setting in a default **SharedPreferences** file for his app's settings.

When the user changes a setting, the system updates the corresponding value in the SharedPreferences file for you.

SharedPreferences

The SharedPreferences is an interface for accessing and modifying preference data returned.

For any particular set of preferences, there is a single instance of this class that all clients share.

Modifications to the preferences must go through an SharedPreferences.Editor object to ensure the preference values remain in a consistent state and control when they are committed to storage.

Objects that are returned from the various get methods must be treated as immutable by the application

The SharedPreferences stores the data in an XML file inside the private data of the application and every data are stored using a key-value pair

SharedPreferences

To write a data using Shared Preferences the developer must use its Editor.

```
SharedPreferences preferences = PreferenceManager.getDefaultSharedPreferences(getApplicationContext());
SharedPreferences.Editor editor = preferences.edit();
editor.putString("key", "value");
editor.apply();
```

To read a data:

```
SharedPreferences preferences = PreferenceManager.getDefaultSharedPreferences(getApplicationContext());
String value = preferences.getString("key", "fallbackValue");
```