Mobile App Dev

Lec5: Saving data

Ekarat Rattagan, PhD

Outline

- Saving data as Key-Value Sets
- Saving data in Files
- Saving data in SQL Databases

Outline

- Saving data as Key-Value Sets
- Saving data in Files
- Saving data in SQL Databases

Saving Key-Value Sets

If you have a relatively small collection of key-values that you'd like to save.

Scenarios why we need to save data, e.g.,

- To check the first time use
- To check the latest version
- To save geographical location
- To save session
- To save app’s setting

Saving Key-Value Sets

You should use the SharedPreferences APIs.

A SharedPreferences object points to a file containing key-value pairs and provides simple methods to read and write them. Each SharedPreferences file is managed by the framework and can be private or shared.

Get a Handle to a SharedPreferences

- You can create a new shared preference file or access an existing one by calling one of two methods:
  - getSharedPreferences() — Use this if you need multiple shared preference files identified by name, which you specify with the first parameter. You can call this from any Context in your app.
  - getPreferences() — Use this from an Activity if you need to use only one shared preference file for the activity. Because this retrieves a default shared preference file that belongs to the activity, you don't need to supply a name.

Create SharedPreferences' objects

For example:

```
Context context = getActivity();
SharedPreferences sharedPref = context.getSharedPreferences(
        getString(R.string.preference_file_key),
        Context.MODE_PRIVATE);
```

If you need just one shared preference file for your activity

```
SharedPreferences sharedPref =
getActivity().getPreferences(Context.MODE_PRIVATE);
```

Write to Shared Preferences(1/2)

To write to a shared preferences file,

- create a SharedPreferences.Editor by calling edit() on your SharedPreferences.
- Pass the keys and values you want to write with methods such as putInt() and putString().
- Then call commit() to save the changes.

Write to Shared Preferences(2/2)

Example:

```
int newHighScore;
SharedPreferences  sharedPref = getActivity().getPreferences
(Context.MODE_PRIVATE);

SharedPreferences.Editor  editor = sharedPref.edit();

editor.putInt(“highScore”, newHighScore);

editor.commit();
```

Read from Shared Preferences (1/2)

To retrieve values from a shared preferences file

- call methods such as getInt() and getString(),
- providing the key for the value you want, and optionally a default value to return if the key isn't present.

Read from Shared Preferences (2/2)

Example,

```
SharedPreferences sharedPref = getActivity().
getPreferences(Context.MODE_PRIVATE);

int defaultValue = 0;

long highScore = sharedPref.getInt(“highScore”, defaultValue);
```

Delete Shared Preferences

1. **Remove one key**

```
Editor editor = shared.edit();
editor.remove(getString(R.string.saved_high_score));
editor.commit();
```

2. **Remove all data**

```
Editor editor = shared.edit();
editor.clear();
editor.commit();
```

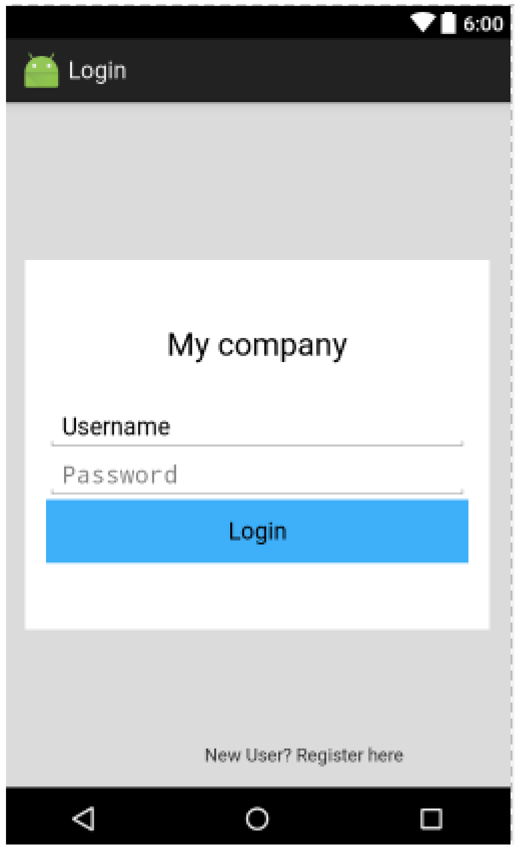
Case study 1 (Simple)

```
public class SharedPreferencesDemo extends Activity
{
    private static final String MY_PREFS = "my_prefs";
    @Override
    public void onCreate(Bundle savedInstanceState)
    {

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

        SharedPreferences shared = getApplicationContext().getSharedPreferences
        (MY_PREFS,  Context.MODE_PRIVATE);
```

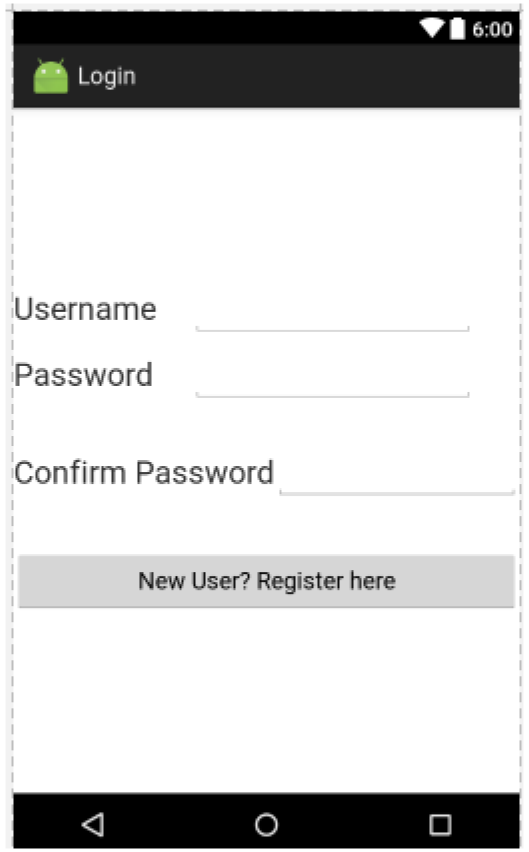
Case study 1

```
    // Write
    Editor editor = shared.edit();
    editor.putString("stringKey", “Ekarat");
    editor.putBoolean("booleanKey", true);
    editor.commit();

    //Read
    String stringValue = shared.getString("stringKey", "not found!");
    boolean booleanValue = shared.getBoolean("booleanKey, false);
    Toast.makeText(this.getApplicationContext(), "name = "+value1+",boolean="+value2,
    Toast.LENGTH_LONG).show();
} }
```

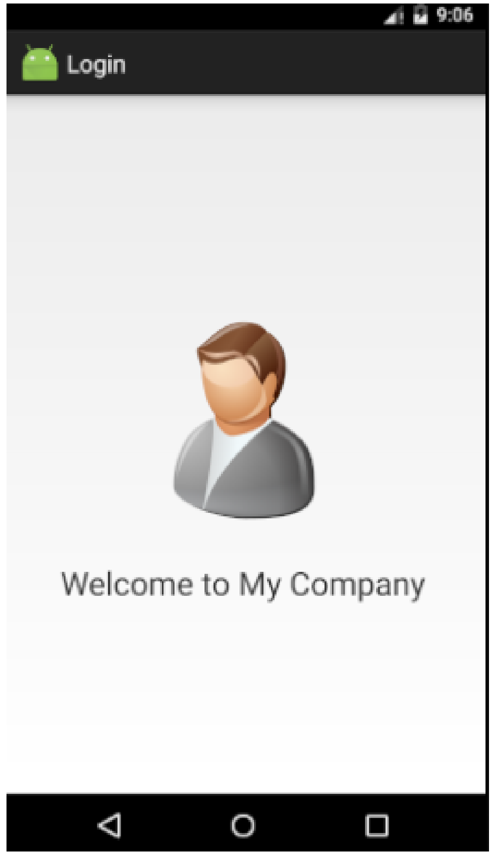
Case study 2: (Login process)



Login UI



Register UI



Main UI

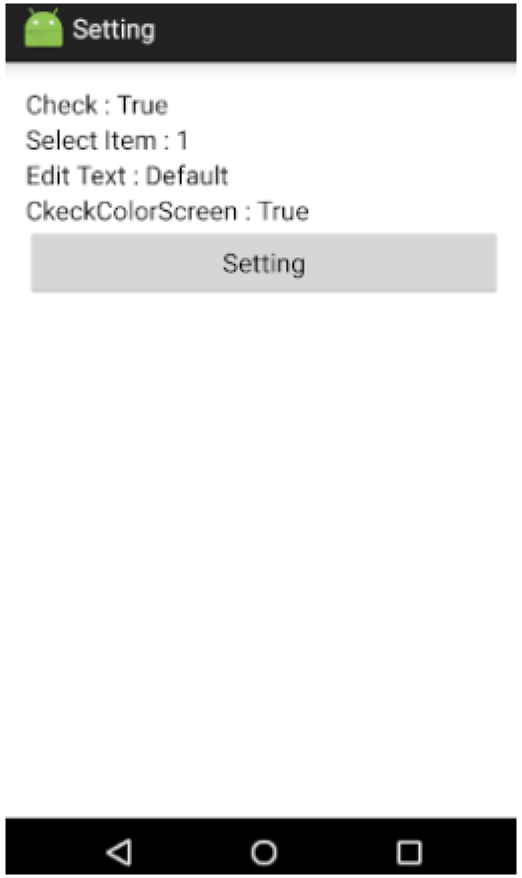
Column 1
Column 2
Column 3

Preference (Setting)

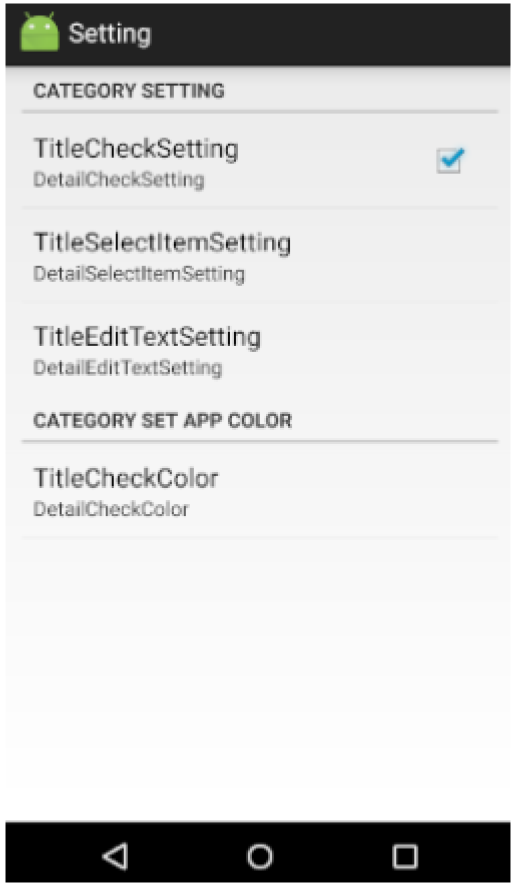
Represents the basic Preference UI building block displayed by a PreferenceActivity in the form of a ListView. Associates with a SharedPreferences to store/retrieve the preference data.

Specifying a preference hierarchy in XML, each element can point to a subclass of Preference, similar to the view hierarchy and layouts.

Case study 3: (Setting)


Setting

Check : True
Select Item : 1
Edit Text : Default
CkeckColorScreen : True


Setting

CATEGORY SETTING

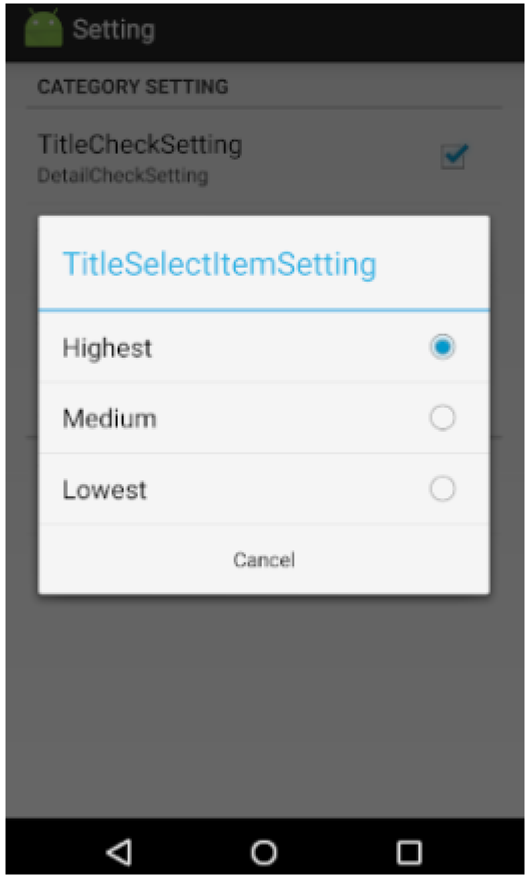
TitleCheckSetting ✓
DetailCheckSetting

TitleSelectItemSetting
DetailSelectItemSetting

TitleEditTextSetting
DetailEditTextSetting

CATEGORY SET APP COLOR

TitleCheckColor
DetailCheckColor


Setting

CATEGORY SETTING

TitleCheckSetting ✓
DetailCheckSetting

TitleSelectItemSetting

[✓ Highest]
Medium
Lowest

Exercise

1. Create a setting to allow users to change the background colors of your apps.

Resource

- https://developer.android.com/training/basics/data-storage/shared-preferences.html
- https://devahoy.com/posts/android-login-activity-with-sharedpreferences/
- http://www.androidcode.in.th/2012/?p=228