Birzeit University
Faculty of Engineering and Technology
Electrical and Computer Engineering Department
Advance Computer Systems Engineering Lab ENCS515

EXP. No. 6. Singleton and Shared Preferences

1. Pre-Lab

- In this experiment we will use Constraint layout as the main layout. So, you need to read about how to add widgets to a Constraint layout. Press Link 1 and Link 2 to see a tutorial about constraint layouts.

2. Objectives

- Provide a simple knowledge of shared preferences in android.
- Introduce you for a new java concept about singleton classes.
- To provide the ability to save strings, integers, long, Boolean and other variable types that are commonly used in the application.

3. Introduction

Shared preferences is commonly used in any android application with will give the user the ability to save the most used values in the application and any other setting locally on the device. To use the shared preferences, we only need one object to write and reed the values locally, so a singleton class is the perfect use for this.

3.1. Singleton Class in Java

In object-oriented programming, a singleton class is a class that can have only one object (an instance of the class) at a time.

After first time, if we try to instantiate the Singleton class, the new variable also points to the first instance created. So whatever modifications we do to any variable inside the class through any instance, it affects the variable of the single instance created and is visible if we access that variable through any variable of that class type defined. To design a singleton class:

- Make constructor as private.
- Write a static method that has return type object of this singleton class. Here, the concept of Lazy initialization in used to write this static method.

Normal class and Singleton class. Difference in normal and singleton class in terms of instantiation is that, For normal class we use constructor, whereas for singleton class we use getInstance() method. In general, to avoid confusion we may also use the class name as method name while defining this method.

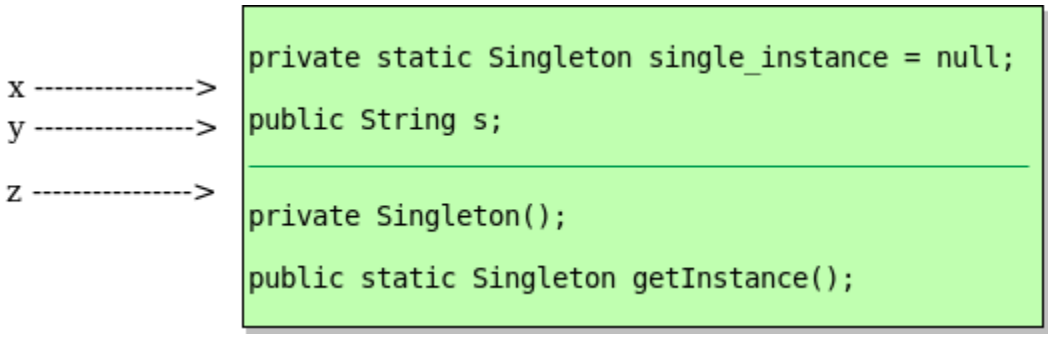

Figure 6.1 singleton class attributes

In the Singleton class, when we first time call getInstance() method, it creates an object of the class with name single_instance and return it to the variable. Since single_instance is static, it is changed from null to some object. Next time, if we try to call getInstance() method, since single_instance is not null, it is returned to the variable, instead of instantiating the Singleton class again. This part is done by if condition.

3.2. Shared preferences

It is used by the application to save data in key-value pairs like Bundle. Data is stored in XML file in the directory "data/data/<package name>/shared-prefs folder. Shared preferences only allows you to save primitive data types (int, byte, short, long, float, double, boolean, and char) In addition to Strings.

- There are two methods to access shared preferences:

  - getPreferences(int mode) : it is used if you have only 1 preference file.
  - getSharedPreferences(String name , int mode): it is used if you have several files.

- The mode parameter can take several values:

  - MODE_PRIVATE: Only your app can access the file.
  - MODE_WORLD_READABLE: All apps can read the file.
  - MODE_WORLD_WRITABLE: All apps can write to the file.

- You can use the following steps to store the data to a shared preference file

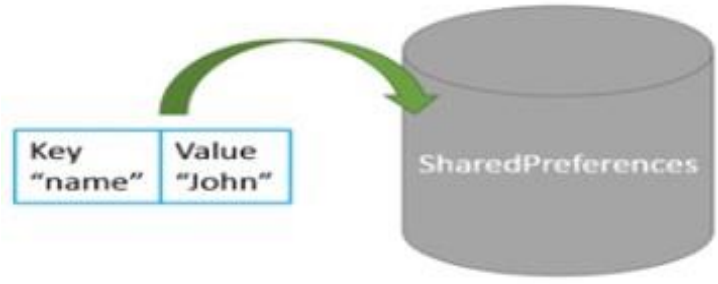

Figure 6.2 writing to the shared preferences

- Get a reference to the SharedPreferences object.

```
sharedPreferences = getSharedPreferences(SHARED_PREF_NAME,
Context.MODE_PRIVATE);
```

Where SHARED_PREF_NAME is a string constant of the name of the shared preferences file.

- Call the editor by using SharedPreferences object.

```
SharedPreferences.Editor editor = sharedPreferences.edit();
```

- Use the editor to add the data with a key.

```
String key = "name";
String value = "Rajaie";
editor.putString(key,value);
```

- Commit editor changes.

```
editor.commit();
```

➢ You can use the following steps to read the data from a shared preference file

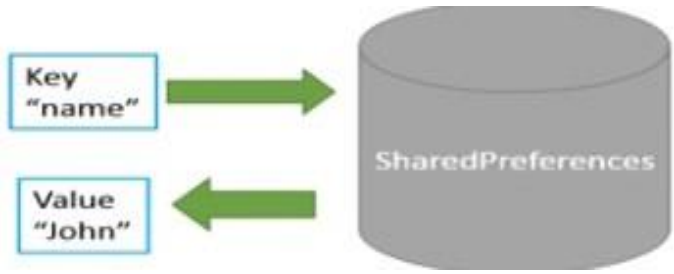

Figure 6.3 Reading From the Shared Preferences

- Get a reference to the SharedPreferences object.

```
sharedPreferences = getSharedPreferences(SHARED_PREF_NAME,
Context.MODE_PRIVATE);
```

- Use the key provided earlier to get data

```
String name =  sharedPreferences.getString("name","noValue");
```

Note: getString method will return noValue if the data is not found.

4. Procedure

You will build an application that has two activities as shown in Figure 6.4, the first will save a username and password in shared preferences and the second will load these values from the shared preferences. You will access the shared preferences using singleton class called SharedPrefManager.

Create a new project and call the project “Shared Preferences Application”, use empty activity to the project, then create a new Activity call this one SecondActivity.

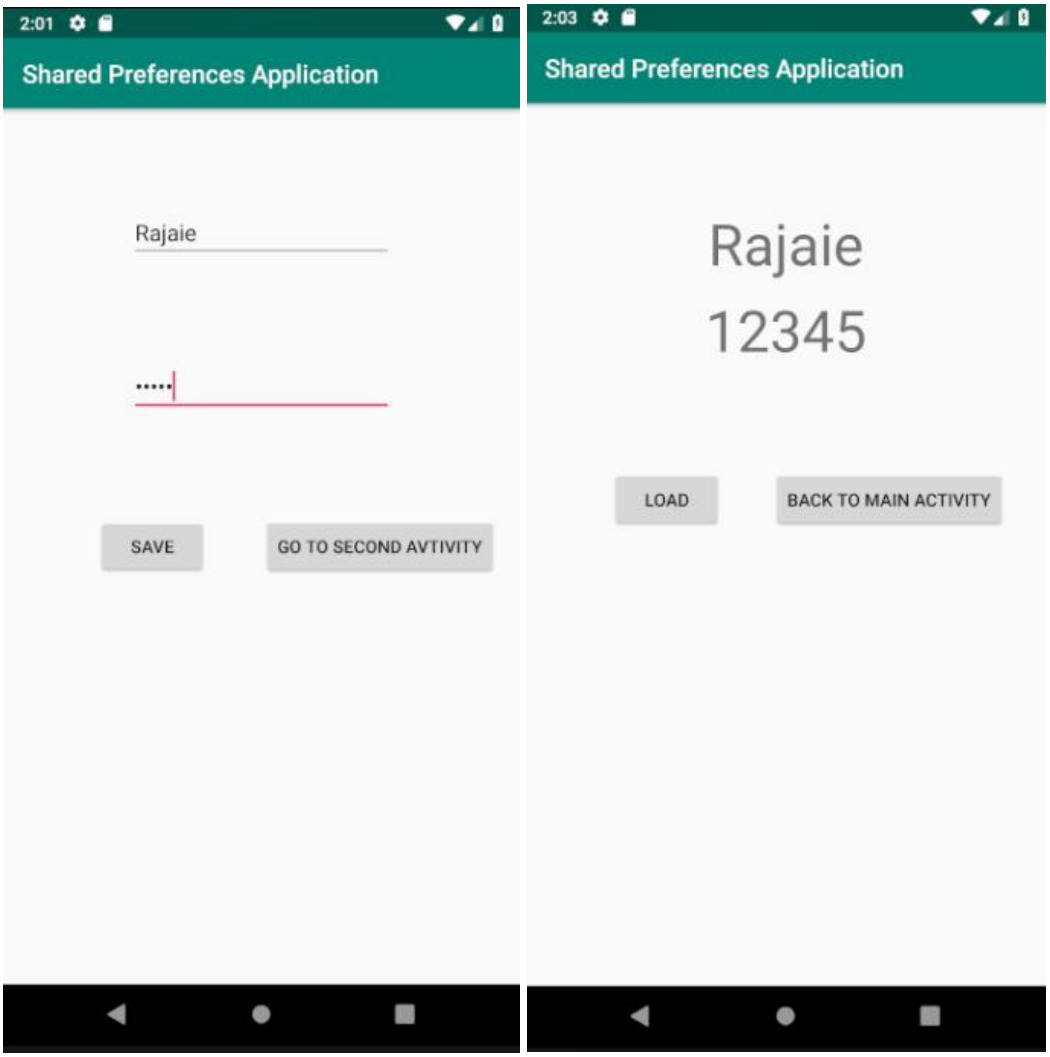

Figure 6.4 SharedPreferences Application Layouts

4.1. Creation of SharedPrefManager Singleton Class

Create a new java class in the main package of the project as shown in and call it SharedPrefManager, this class will access the shared preferences for reading and writing. This class will have the following

- Attribute that is called ourInstance that has a type of SharedPrefManager and initialed to null and private.
- Attribute that is called sharedPreference that has a type of SharedPreferences and initiated to null and private.
- Attribute that is called editor that is used for writing to the shared preferences has a type of SharedPreferences. Editor and initiated to null and private.
- A private constructor which will initiate the sharedPreference and the editor.
- A static method called getInstance which will return ourInstance if its not null and if its null (first time to initiate the object) it will create a new object of SharedPrefManager using the private constructor and return it.
- Public method for writing to the sharedPreferences
- Public method for reading from the sharedPreferences
- Other constant values as the sharedPreference name, shared Preference access Modes etc…

The following code shows the singleton SharedPrefManager class with two method to write and read string from the sharedPreferences

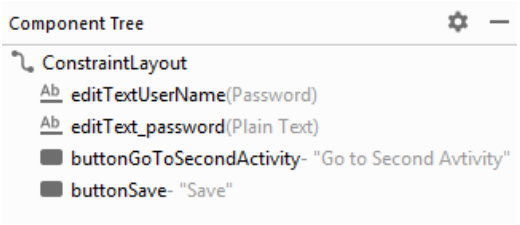

Figure 6.5 MainActivity Component Tree

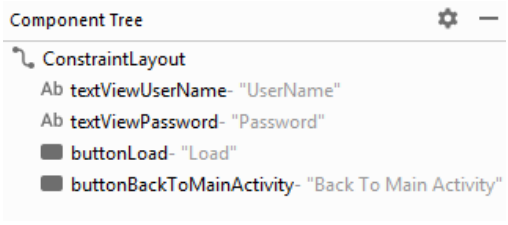

Figure 6.6 SecondActivity Component Tree

```
import android.content.Context;
import android.content.SharedPreferences;

public class SharedPrefManager {


    private static final String SHARED_PREF_NAME = "My Shared Preference";
    private static final int SHARED_PREF_PRIVATE = Context.MODE_PRIVATE;

    private static SharedPrefManager ourInstance = null;
    private static SharedPreferences sharedPreferences = null;
    private SharedPreferences.Editor editor = null;

    static SharedPrefManager getInstance(Context context) {

        if (ourInstance != null) {
            return ourInstance;
        }
        ourInstance=new SharedPrefManager(context);
        return ourInstance;
    }

    private SharedPrefManager(Context context) {
        sharedPreferences = context.getSharedPreferences(SHARED_PREF_NAME,
SHARED_PREF_PRIVATE);
        editor = sharedPreferences.edit();
    }

    public boolean writeString(String key, String value) {

        editor.putString(key, value);
        return editor.commit();
    }

    public String readString(String key, String defaultValue) {
        return sharedPreferences.getString(key, defaultValue);
    }

}
```

4.2. Building the MainActivity Layout and the SecondActivity Layout

Build the layout activity_main as shown inFigure 6.4 (the component tree is shown in Figure 6.5) and the activity_second layout as shown in Figure 6.4 (the component tree is shown in Figure 6.6)

4.3. Building the MainActivity Java Class

Get reference to the UserName and Password editTexts and the Save and GoToSecondActivity Buttons you added in the activity_main layout.

Initate a SharedPrefManager by calling the static getInstance method. Make a listener to the Save button and save the values from the editTexts to the sharedPreferences using the writeString from the SharedPrefManager by passing the key (e.g. “username”) and the value from the edit text.

Make a listener to the GoToSecondActivity and start the SecondActivity. The code below is for the main Activity.

```
EditText editTextUserName;
EditText editTextPassword;
Button buttonSave;
Button buttonGoToSecondActivity;
SharedPrefManager sharedPrefManager;
Intent intent;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    editTextUserName = (EditText) findViewById(R.id.editTextUserName);
    editTextPassword = (EditText) findViewById(R.id.editText_password);
    buttonSave = (Button) findViewById(R.id.buttonSave);
    buttonGoToSecondActivity = (Button) findViewById(R.id.buttonGoToSecondActivity);
    sharedPrefManager =SharedPrefManager.getInstance(this);
    intent = new Intent(MainActivity.this,SecondActivity.class);

    buttonSave.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {

sharedPrefManager.writeString("userName",editTextUserName.getText().toString());

sharedPrefManager.writeString("password",editTextPassword.getText().toString());
         Toast.makeText(MainActivity.this, "Values written to shared Preferences",
Toast.LENGTH_SHORT).show();
        }
    });

    buttonGoToSecondActivity.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(intent);
            finish();
        }
    });
}
```

4.4. Building the SecondActivity Java Class

Get reference to the UserName and Password TextViews and the Load and BackToMainActivity Buttons you added in the activity_Second layout.Initate a SharedPrefManager by calling the static getInstance method. Make a listener to the Load button and Load the values from the sharedPreferences to the editTexts using the readString from the SharedPrefManager by passing the key (e.g. “username”) and the defalt value. Make a listener to the BackToMainActivity and start the MainActivity. The code below is for the Second Activity.

```
TextView textViewUserName;
TextView textViewPassword;
Button buttonLoad;
Button buttonBackToMainActivity;
SharedPrefManager sharedPrefManager;
Intent intent;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_second);
    textViewUserName = (TextView) findViewById(R.id.textViewUserName);
    textViewPassword = (TextView) findViewById(R.id.textViewPassword);
    buttonLoad = (Button) findViewById(R.id.buttonLoad);
    buttonBackToMainActivity = (Button)
findViewById(R.id.buttonBackToMainActivity);
    sharedPrefManager=SharedPrefManager.getInstance(this);
    intent = new Intent(SecondActivity.this,MainActivity.class);
    buttonLoad.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {

textViewUserName.setText(sharedPrefManager.readString("userName","noValue"));

textViewPassword.setText(sharedPrefManager.readString("password","noValue"));
        }
    });
    buttonBackToMainActivity.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(intent);
            finish();
        }
    });
}
```

5. Todo

This part will be given to you by the teacher assistant in the lab time.