Android provides many ways of storing data of an application.

- One of this way is called Shared Preferences.
- Shared Preferences allow you to save and retrieve data in the form of key,value pair.

Saving Key-Value Sets

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

This document (**downloaded from <u>android.suven.net</u>** , ***Compiled by Rocky sir***) shows you how to use the SharedPreferences APIs to store and retrieve simple values.

Note: The SharedPreferences APIs are only for reading and writing key-value pairs and you should not confuse them with the Preference APIs, which help you build a user interface for your app settings (although they use SharedPreferences as their implementation to save the app settings).

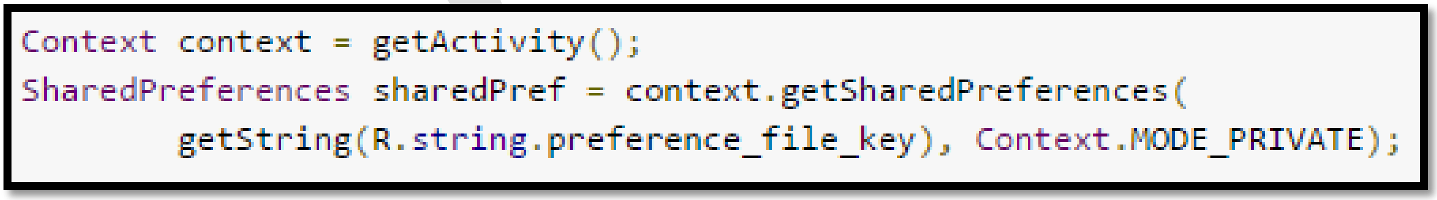
Get a Handle to a SharedPreferences

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.

For example, the following code accesses the shared preferences file that's identified by the resource string R.string.preference_file_key and opens it using the private mode so the file is accessible by only your app.



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

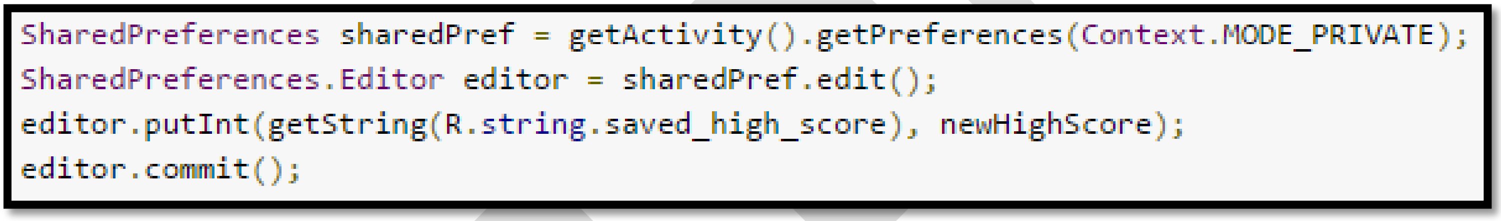
When naming your shared preference files, you should use a name that's uniquely identifiable to your app, such as "com.example.myapp.PREFERENCE_FILE_KEY".

Caution: If you create a shared preferences file with `MODE_WORLD_READABLE` or `MODE_WORLD_WRITEABLE`, then any other apps that know the file identifier can access your data.

Write to Shared Preferences

To write to a shared preferences file, create a **SharedPreferences.Editor** by calling **edit()** on your SharedPreferences.

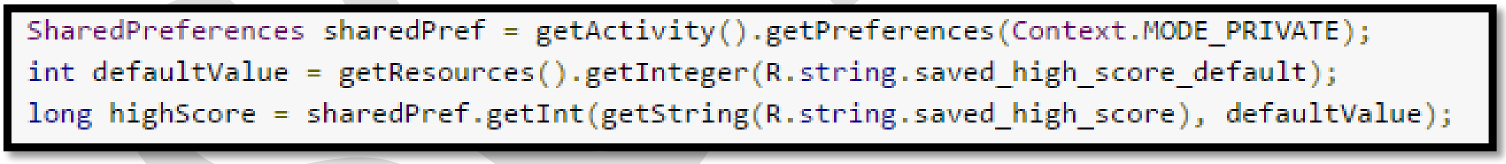
Pass the keys and values you want to write with methods such as **putInt()** and **putString()**. Then call **commit()** to save the changes. **For example:**



```
SharedPreferences sharedPref = getActivity().getPreferences(Context.MODE_PRIVATE);
SharedPreferences.Editor editor = sharedPref.edit();
editor.putInt(getString(R.string.saved_high_score), newHighScore);
editor.commit();
```

Read from Shared Preferences

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. For example:



```
SharedPreferences sharedPref = getActivity().getPreferences(Context.MODE_PRIVATE);
int defaultValue = getResources().getInteger(R.string.saved_high_score_default);
long highScore = sharedPref.getInt(getString(R.string.saved_high_score), defaultValue);
```

Summarizing all Points:

You can save and retrieve key, value pair data from Shared preferences.
SharedPreferences values will persist across user sessions.

I.e. ***Data in shared preferences will be persistent even though user closes the application.***

You can get values from Shared preferences using **getSharedPreferences()** method.
You also need an editor to edit and save the changes in shared preferences.
Use SharedPreferences to store **any primitive type data** like boolean, float, int, long and also **String**.

Available mode for shared preference:

1. ***MODE_WORLD_READABLE***
2. ***MODE_WORLD_WRITEABLE***
3. ***MODE_PRIVATE***

Complete Example: Store, Fetch, Remove, and Clear Data from SharedPreferences.

```
/******* Create SharedPreferences *******/
SharedPreferences pref =
getApplicationContext().getSharedPreferences("MyPref", MODE_PRIVATE);

Editor editor = pref.edit();

/**************** Storing data as KEY/VALUE pair *******************/
editor.putBoolean("key_name1", true);// Saving boolean - true/false
editor.putInt("key_name2", "int value");  // Saving integer
editor.putFloat("key_name3", "float value");  // Saving float
editor.putLong("key_name4", "long value");    // Saving long
editor.putString("key_name5", "string value");  // Saving string

// Save the changes in SharedPreferences
editor.commit(); // commit changes


/**************** Get SharedPreferences data *******************/
// If value for key, not exist then return second param value - In this
case null
Boolean b=pref.getBoolean("key_name1", null);   // getting boolean
int I = pref.getInt("key_name2", null);          // getting Integer
float f = pref.getFloat("key_name3", null);     // getting Float
double d = pref.getLong("key_name4", null);     // getting Long
String str = pref.getString("key_name5", null); // getting String



/************ Deleting Key value from SharedPreferences *****************/
    editor.remove("key_name3"); // will delete key key_name3
    editor.remove("key_name4"); // will delete key key_name4

    // Save the changes in SharedPreferences
    editor.commit(); // commit changes


/************ Clear all data from SharedPreferences *****************/
     editor.clear();
     editor.commit(); // commit changes
```

Project Work:

Create a simple android App to demonstrate the use of Shared Preference. This Project should be able to verify the following:

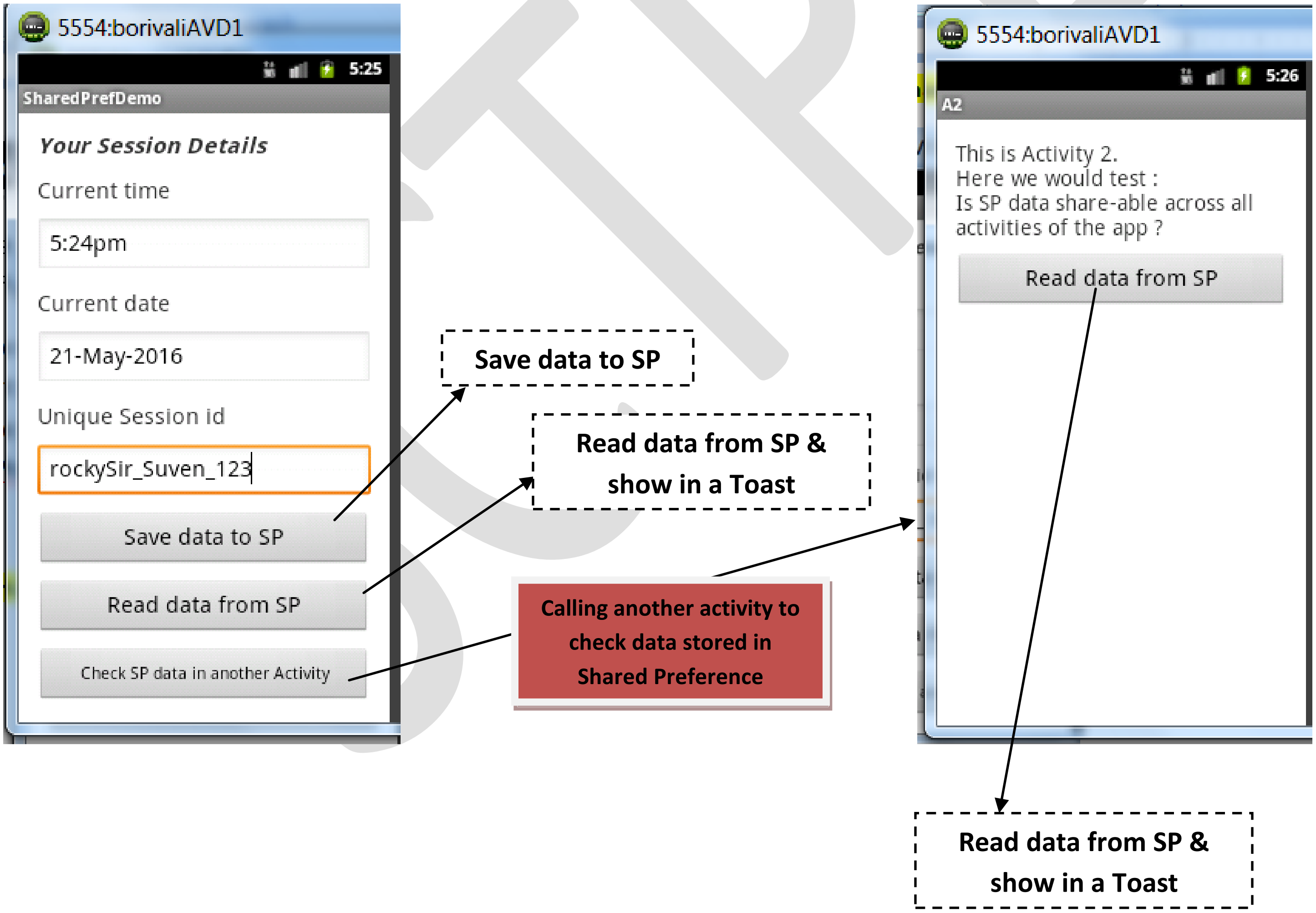
1> Data stored in ***SharedPreference*** file of an activity can be read in the same activity.

2> Same data can be read by any other activity of the same application, provided it knows the name of the ***SharedPreference*** file.

3> ***SharedPreference*** file persist across application calls.

Work Flow:

The App should have 2 activities. There Interfaces are :



Your Session Details

Current time

5:24pm

Current date

21-May-2016

Unique Session id

rockySir_Suven_123

Save data to SP

Read data from SP & show in a Toast

Calling another activity to check data stored in Shared Preference

This is Activity 2.
Here we would test :
Is SP data share-able across all activities of the app ?

Read data from SP & show in a Toast

Download the entire project from this link

https://drive.google.com/open?id=0B4rCFkKCsCeKU1N2Nmg2R3I2ZW8