Shared Preferences

TOPICS

- Introduction
- Shared Preferences
- Saving data to Shared preferences
- Reading data from shared preferences
- Examples

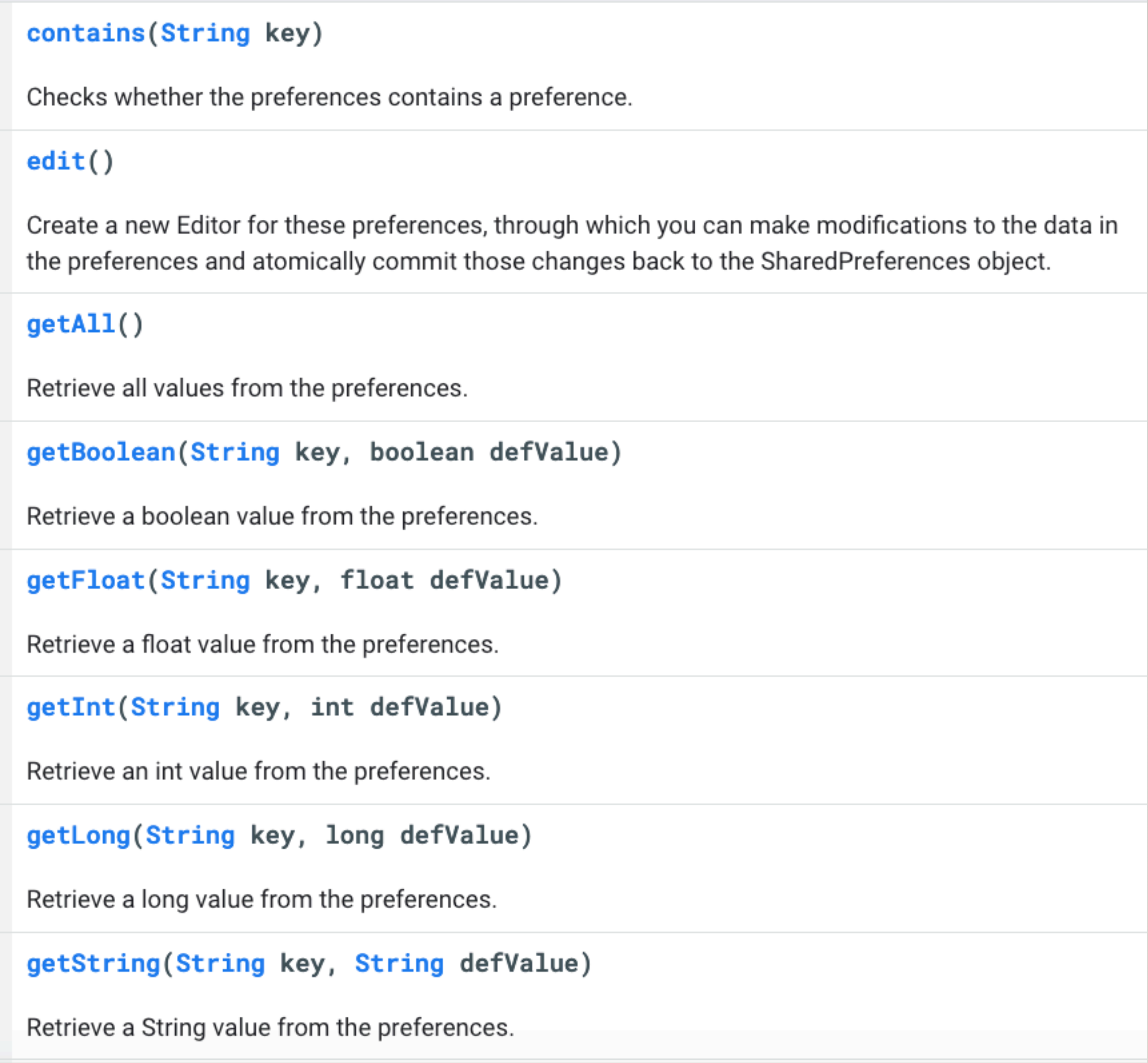
DATA PERSISTENCE IN ANDROID

- Android provides several options for you to save your app data
- **1- Files I/O**
  - Internal file storage: Store app-private files on the device
  - External file storage: Store files on the shared external file system. This is usually for shared user files, such as photos.
- 2- Shared preferences: Store private primitive data in key-value pairs.
- 3- Databases: Store structured data in a private database.
- 4- Networking: store data on remote server

Shared Preferences

- Allow us to store and retrieve data in key/value pair.
- Each value has its own key for storage and retrieval of that data.
- You can get Shared Preference instance using the
  - getSharedPreferences (String fileName, int mode) // object points to a file containing //key-value pairs
  - getPreferences(MODE_PRIVATE); // activity default shared preference

Shared Preferences Methods



`contains(String key)`

Checks whether the preferences contains a preference.

`edit()`

Create a new Editor for these preferences, through which you can make modifications to the data in the preferences and atomically commit those changes back to the SharedPreferences object.

`getAll()`

Retrieve all values from the preferences.

`getBoolean(String key, boolean defValue)`

Retrieve a boolean value from the preferences.

`getFloat(String key, float defValue)`

Retrieve a float value from the preferences.

`getInt(String key, int defValue)`

Retrieve an int value from the preferences.

`getLong(String key, long defValue)`

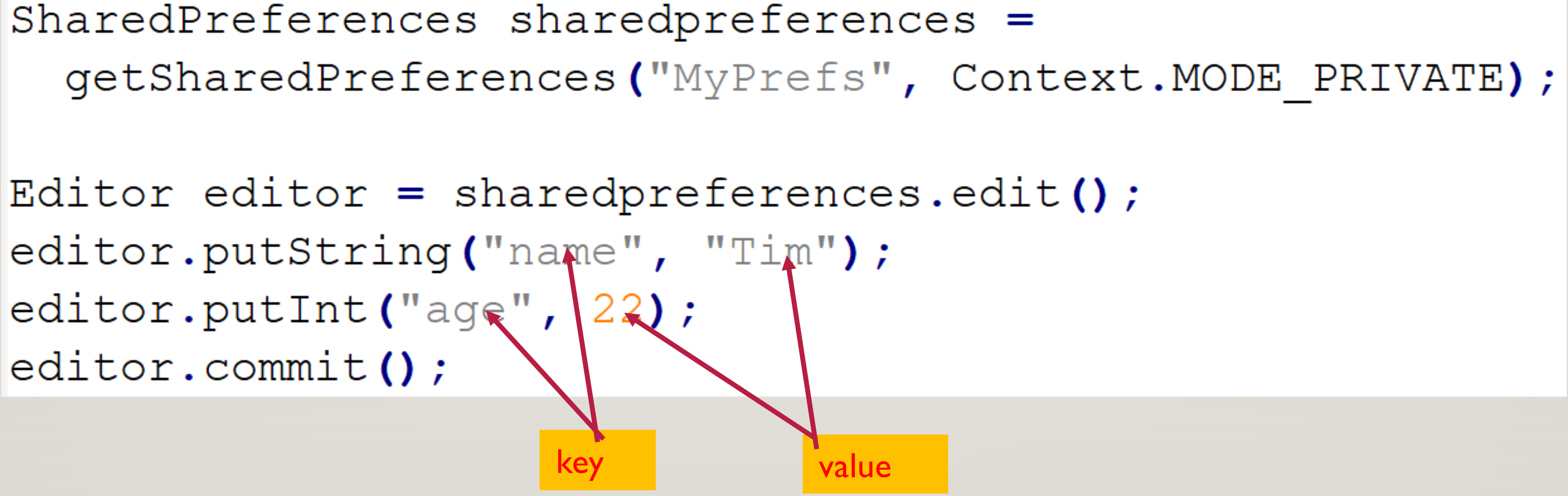
Retrieve a long value from the preferences.

`getString(String key, String defValue)`

Retrieve a String value from the preferences.

SAVING DATA TO SHARED PREFERENCES

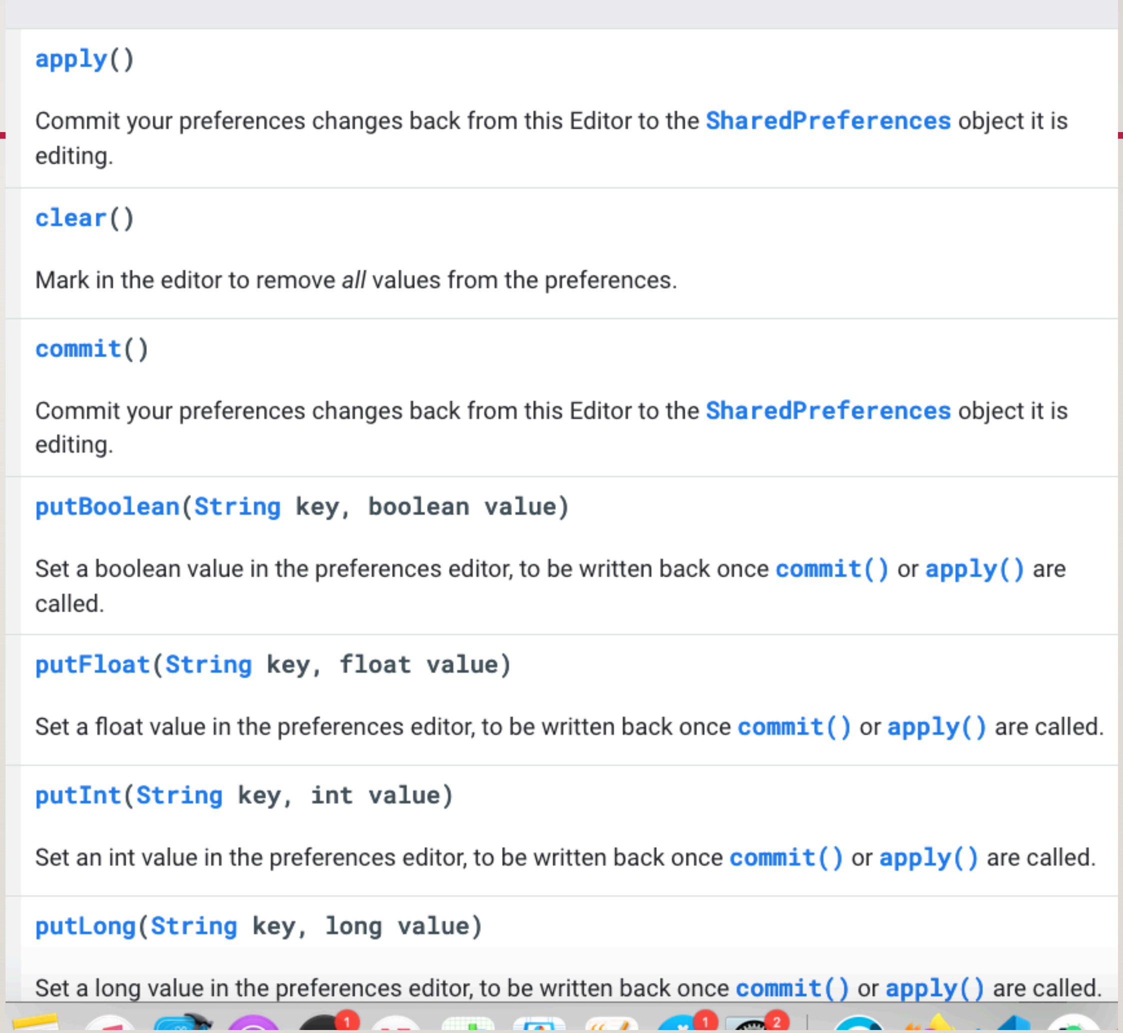
- We can save data to shared preferences using **SharedPreferences.Editor** class.
- .



```
SharedPreferences sharedpreferences =
  getSharedPreferences("MyPrefs", Context.MODE_PRIVATE);

Editor editor = sharedpreferences.edit();
editor.putString("name", "Tim");
editor.putInt("age", 22);
editor.commit();
```

SHAREDPREFERENCES.EDITOR CLASS.



`apply()`

Commit your preferences changes back from this Editor to the `SharedPreferences` object it is editing.

`clear()`

Mark in the editor to remove *all* values from the preferences.

`commit()`

Commit your preferences changes back from this Editor to the `SharedPreferences` object it is editing.

`putBoolean(String key, boolean value)`

Set a boolean value in the preferences editor, to be written back once `commit()` or `apply()` are called.

`putFloat(String key, float value)`

Set a float value in the preferences editor, to be written back once `commit()` or `apply()` are called.

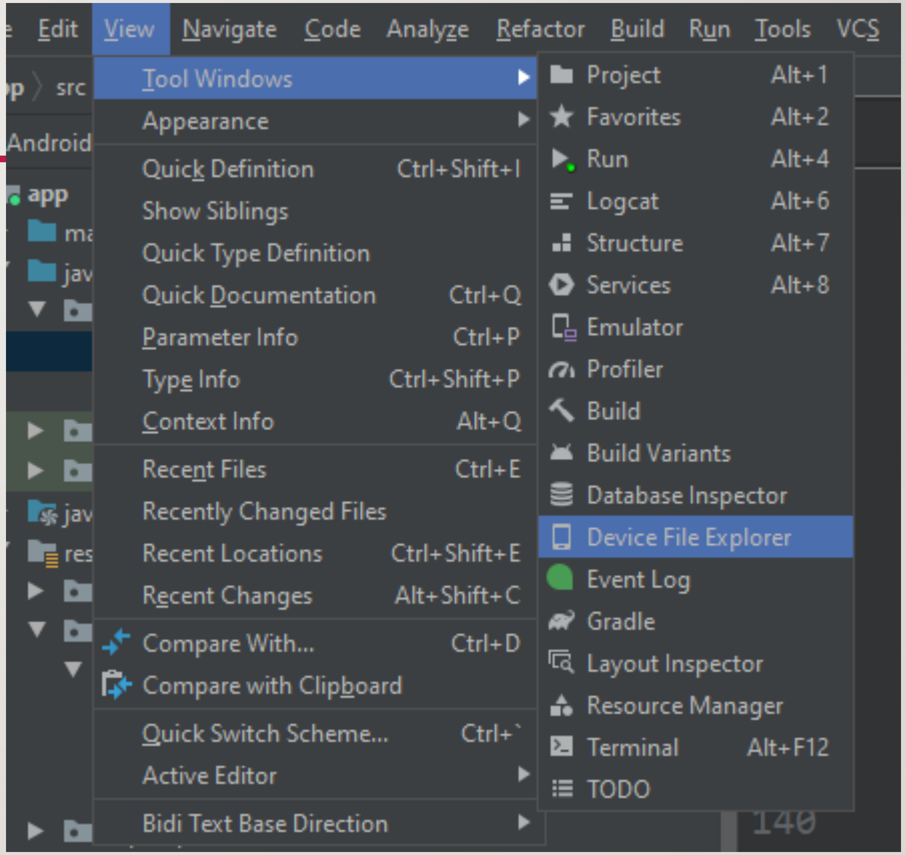
`putInt(String key, int value)`

Set an int value in the preferences editor, to be written back once `commit()` or `apply()` are called.

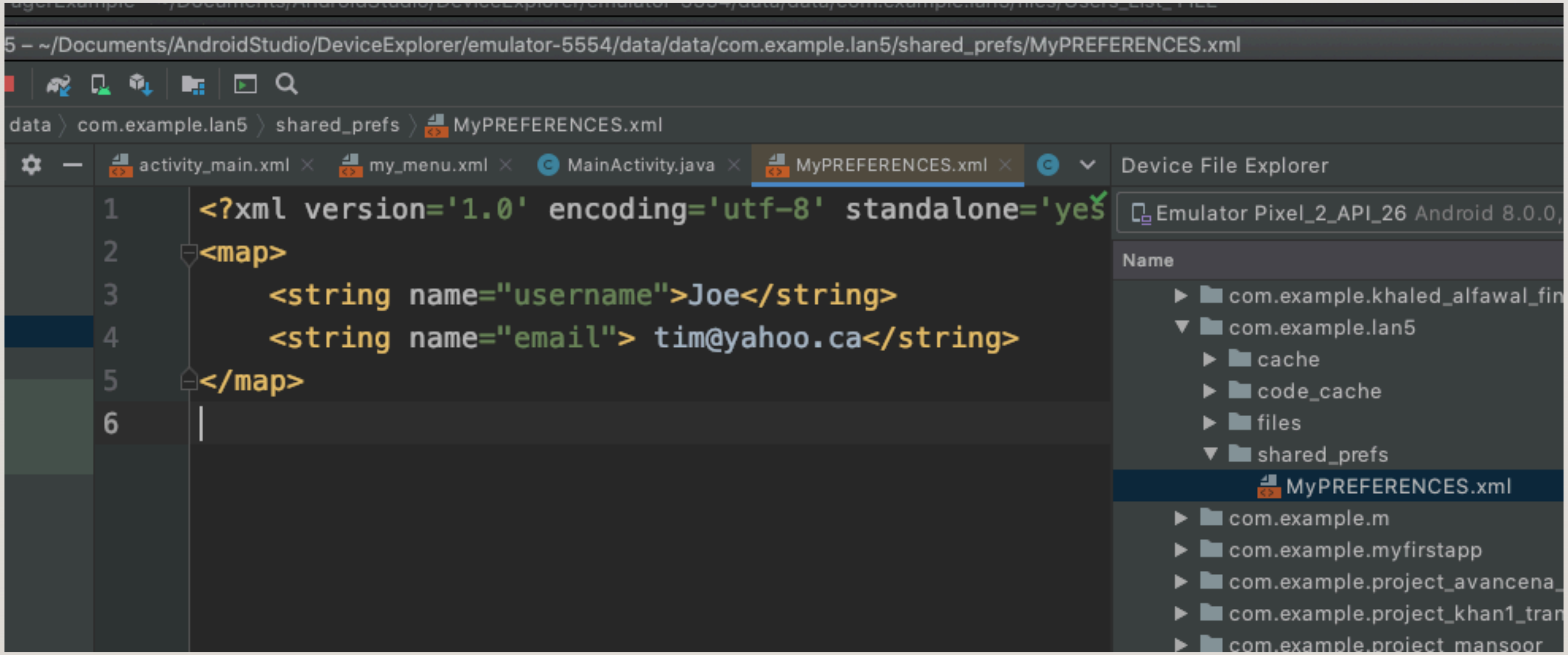
`putLong(String key, long value)`

Set a long value in the preferences editor, to be written back once `commit()` or `apply()` are called.

Files are stored in your apps sandbox in the /data/data folder



https://developer.android.com/studio/debug/device-file-explorer


5 – ~/Documents/AndroidStudio/DeviceExplorer/emulator-5554/data/data/com.example.lan5/shared_prefs/MyPREFERENCES.xml
data 〉 com.example.lan5 〉 shared_prefs 〉 MyPREFERENCES.xml
activity_main.xml
my_menu.xml
MainActivity.java
MyPREFERENCES.xml
Device File Explorer
Emulator Pixel_2_API_26 Android 8.0.0,
Name
com.example.khaled_alfawal_fin
com.example.lan5
cache
code_cache
files
shared_prefs
MyPREFERENCES.xml
com.example.m
com.example.myfirstapp
com.example.project_avancena_
com.example.project_khan1_tran
com.example.project_mansoor

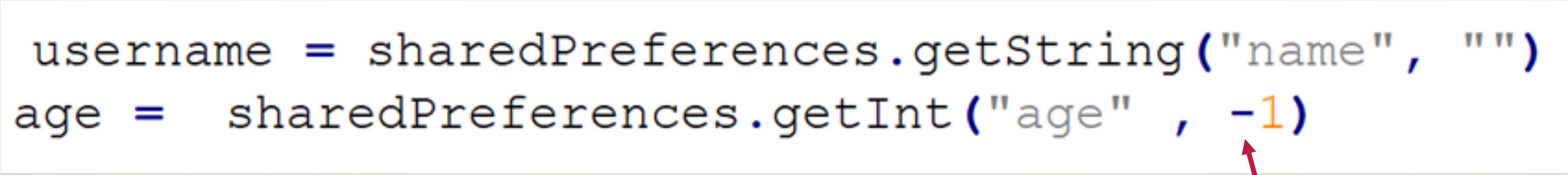
```
<?xml version='1.0' encoding='utf-8' standalone='yes
<map>
    <string name="username">Joe</string>
    <string name="email"> tim@yahoo.ca</string>
</map>
```

`APPLY()` VS `COMMIT()`.

When using the Editor to write values to the SharedPreferences, the changes are only applied when you call `apply()` or `commit()`.

- `commit()`
  - The method will block until the changes are reflected both in memory as well as the underlying XML file.
  - Returns true on success, false on failure
- `apply()`
  - The method call updates the in-memory representation of the SharedPreferences but returns before the underlying XML file is updated "on disk"
- `clear()`
  - Mark the editor to clear all values. Must call `apply()` or `commit()` after

Reading data from shared preferences

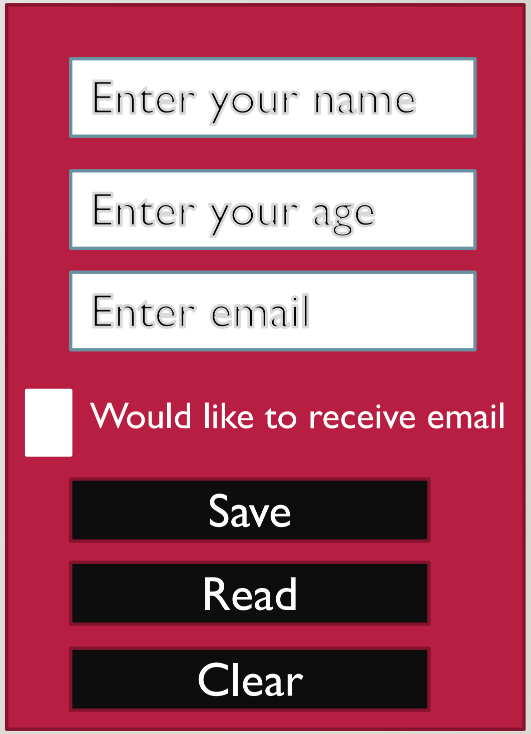


```
 username = sharedPreferences.getString("name", "")
age =  sharedPreferences.getInt("age" , -1)
```

if the key is not found in SharedPref return -1

EXAMPLE

- Create an app that will ask for
  - name , age and email
- Save Button saves the data to a SharedPreferences file called “Prefs”
- Read Button read the data from a SharedPreferences and display on logcat
- Clear button clears the SharedPreferences values



REFERENCES

- https://developer.android.com/training/data-storage/shared-preferences
- https://developer.android.com/reference/android/content/SharedPreferences.Editor
- https://developer.android.com/reference/android/content/SharedPreferences