What are files on Android system?

Under the Android file system, there are usually six main partitions found on every device. Some devices may come with a couple of additional partitions, which differ from model to model, but six primary partitions are found on every Android device. These include **/boot, /system, /recovery, /data, /cache, and /misc**.

The misc. refers to miscellaneous files, which are all the temporary files and files which cannot be classified as any of the above, such as zip, rar, apk files, application database, thumbnail files, font files, advertisement files, metadata files or any kind of unknown file formats.

Is it safe to delete Misc files on Android?

The Misc category can contain many types of files and **you cannot be sure deleting misc files will not cause any issues on your Android device**. It may or may not cause problems on your phone. Sometimes, deleting misc files can actually delete some of your WhatsApp files.

Where is file on Android?

On your phone, you can usually find your files in the Files app . If you can't find the Files app, your device manufacturer might have a different app.  
...  
**Find & open files**

1. Open your phone's Files app . Learn where to find your apps.
2. Your downloaded files will show. To find other files, tap Menu . ...
3. To open a file, tap it.

<https://www.geeksforgeeks.org/shared-preferences-in-android-with-examples/>

One of the most Interesting Data Storage options **Android** provides its users is **Shared Preferences**. **Shared Preferences** is the way in which one can store and retrieve small amounts of primitive data as key/value pairs to a file on the device storage such as String, int, float, Boolean that make up your preferences in an XML file inside the app on the device storage. **Shared Preferences** can be thought of as a dictionary or a key/value pair. For example, you might have a key being “username” and for the value, you might store the user’s username. And then you could retrieve that by its key (here username). You can have a simple shared preference API that you can use to store preferences and pull them back as and when needed. The shared Preferences class provides APIs for reading, writing, and managing this data. A sample GIF is given below to get an idea about what we are going to do in this article. The code for that has been given in both**Java and Kotlin Programming Language for Android.**

**Shared Preferences** are suitable for different situations. For example, when the user’s settings need to be saved or to store data that can be used in different activities within the app. As you know, onPause() will always be called before your activity is placed in the background or destroyed, So for the data to be saved persistently, it’s preferred to save it in onPause(), which could be restored in onCreate() of the activity. The data stored using shared preferences are kept private within the scope of the application. However, shared preferences are different from that activity’s instance state.

### ****How to Create Shared Preferences?****

The first thing we need to do is to create one shared preferences file per app. So name it with the package name of your app- unique and easy to associate with the app. When you want to get the values, call the **getSharedPreferences()** method. Shared Preferences provide modes of storing the data (private mode and public mode). It is for backward compatibility- use only **MODE\_PRIVATE**to be secure.

***public abstract SharedPreferences getSharedPreferences (String name, int mode)***

*This method takes two arguments, the first being the name of the****SharedPreference(SP) file****and the other is the****context mode****that we want to store our file in.*

***MODE\_PUBLIC****will make the file public which could be accessible by other applications on the device*

***MODE\_PRIVATE****keeps the files private and secures the user’s data.*

***MODE\_APPEND****is used while reading the data from the SP file.*

### ****Nested classes of Shared Preferences****

1. **SharedPreferences.Editor**: Interface used to write(edit) data in the SP file. Once editing has been done, one must **commit()** or **apply()** the changes made to the file.
2. **SharedPreferences.OnSharedPreferenceChangeListener()**: Called when a shared preference is changed, added, or removed. This may be called even if a preference is set to its existing value. This callback will be run on your main thread.

### ****Following are the methods of Shared Preferences****

1. **contains(String key)**: This method is used to check whether the preferences contain a preference.
2. **edit()**: This method is used to 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.
3. **getAll()**: This method is used to retrieve all values from the preferences.
4. **getBoolean(String key, boolean defValue)**: This method is used to retrieve a boolean value from the preferences.
5. **getFloat(String key, float defValue)**: This method is used to retrieve a float value from the preferences.
6. **getInt(String key, int defValue)**: This method is used to retrieve an int value from the preferences.
7. **getLong(String key, long defValue)**: This method is used to retrieve a long value from the preferences.
8. **getString(String key, String defValue)**: This method is used to retrieve a String value from the preferences.
9. **getStringSet(String key, Set defValues)**: This method is used to retrieve a set of String values from the preferences.
10. **registerOnSharedPreferencechangeListener(SharedPreferences.OnSharedPreferencechangeListener listener)**: This method is used to register a callback to be invoked when a change happens to a preference.
11. **unregisterOnSharedPreferencechangeListener(SharedPreferences.OnSharedPreferencechangeListener listener)**: This method is used to unregister a previous callback.

#### ****Following is a sample byte code on how to write Data in Shared Preferences:****

// Storing data into SharedPreferences

SharedPreferences sharedPreferences = getSharedPreferences("MySharedPref",MODE\_PRIVATE);

// Creating an Editor object to edit(write to the file)

SharedPreferences.Editor myEdit = sharedPreferences.edit();

// Storing the key and its value as the data fetched from edittext

myEdit.putString("name", name.getText().toString());

myEdit.putInt("age", Integer.parseInt(age.getText().toString()));

// Once the changes have been made, we need to commit to apply those changes made,

// otherwise, it will throw an error

myEdit.commit();

#### ****Following is the sample byte code on how to read Data in Shared Preferences:****

// Retrieving the value using its keys the file name

must be same in both saving and retrieving the data

SharedPreferences sh = getSharedPreferences("MySharedPref", MODE\_APPEND);

// The value will be default as empty string because for

the very

// first time when the app is opened, there is nothing to show

String s1 = sh.getString("name", "");

int a = sh.getInt("age", 0);

// We can then use the data

name.setText(s1);

age.setText(String.valueOf(a));

### ****Example to Demonstrate the use of Shared Preferences in Android****

Below is the small demo for Shared Preferences. In this particular demo, there are two [EditTexts](https://www.geeksforgeeks.org/edittext-widget-in-android-using-java-with-examples/), which save and retain the data entered earlier in them. This type of feature can be seen in applications with forms. Using Shared Preferences, the user will not have to fill in details again and again. Invoke the following code inside the **activity\_main.xml**file to implement the UI:

**<?xml version="1.0" encoding="utf-8"?>**

**<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"**

**xmlns:tools="http://schemas.android.com/tools"**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent"**

**tools:context=".MainActivity"**

**tools:ignore="HardcodedText">**

**<TextView**

**android:id="@+id/textview"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_centerHorizontal="true"**

**android:layout\_marginTop="32dp"**

**android:text="Shared Preferences Demo"**

**android:textColor="@android:color/black"**

**android:textSize="24sp" />**

**<!--EditText to take the data from the user and save the data in SharedPreferences-->**

**<EditText**

**android:id="@+id/edit1"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:layout\_below="@+id/textview"**

**android:layout\_marginStart="16dp"**

**android:layout\_marginTop="8dp"**

**android:layout\_marginEnd="16dp"**

**android:hint="Enter your Name"**

**android:padding="10dp" />**

**<!--EditText to take the data from the user and save the data in SharedPreferences-->**

**<EditText**

**android:id="@+id/edit2"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:layout\_below="@+id/edit1"**

**android:layout\_marginStart="16dp"**

**android:layout\_marginTop="8dp"**

**android:layout\_marginEnd="16dp"**

**android:hint="Enter your Age"**

**android:inputType="number"**

**android:padding="10dp" />**

**</RelativeLayout>**

#### Working with the MainActivity file to handle the two of the EditText to save the data entered by the user inside the SharedPreferences.

Below is the code for the **MainActivity** file. Comments are added inside the code to understand the code in more detail.

**import androidx.appcompat.app.AppCompatActivity;**

**import android.content.SharedPreferences;**

**import android.os.Bundle;**

**import android.widget.EditText;**

**public class MainActivity extends AppCompatActivity {**

**private EditText name, age;**

**@Override**

**protected void onCreate(Bundle savedInstanceState) {**

**super.onCreate(savedInstanceState);**

**setContentView(R.layout.activity\_main);**

**name = findViewById(R.id.edit1);**

**age = findViewById(R.id.edit2);**

**}**

**// Fetch the stored data in onResume() Because this is what will be called when the app opens again**

**@Override**

**protected void onResume() {**

**super.onResume();**

**// Fetching the stored data from the SharedPreference**

**SharedPreferences sh = getSharedPreferences("MySharedPref", MODE\_PRIVATE);**

**String s1 = sh.getString("name", "");**

**int a = sh.getInt("age", 0);**

**// Setting the fetched data in the EditTexts**

**name.setText(s1);**

**age.setText(String.valueOf(a));**

**}**

**// Store the data in the SharedPreference in the onPause() method**

**// When the user closes the application onPause() will be called and data will be stored**

**@Override**

**protected void onPause() {**

**super.onPause();**

**// Creating a shared pref object with a file name "MySharedPref" in private mode**

**SharedPreferences sharedPreferences = getSharedPreferences("MySharedPref", MODE\_PRIVATE);**

**SharedPreferences.Editor myEdit = sharedPreferences.edit();**

**// write all the data entered by the user in SharedPreference and apply**

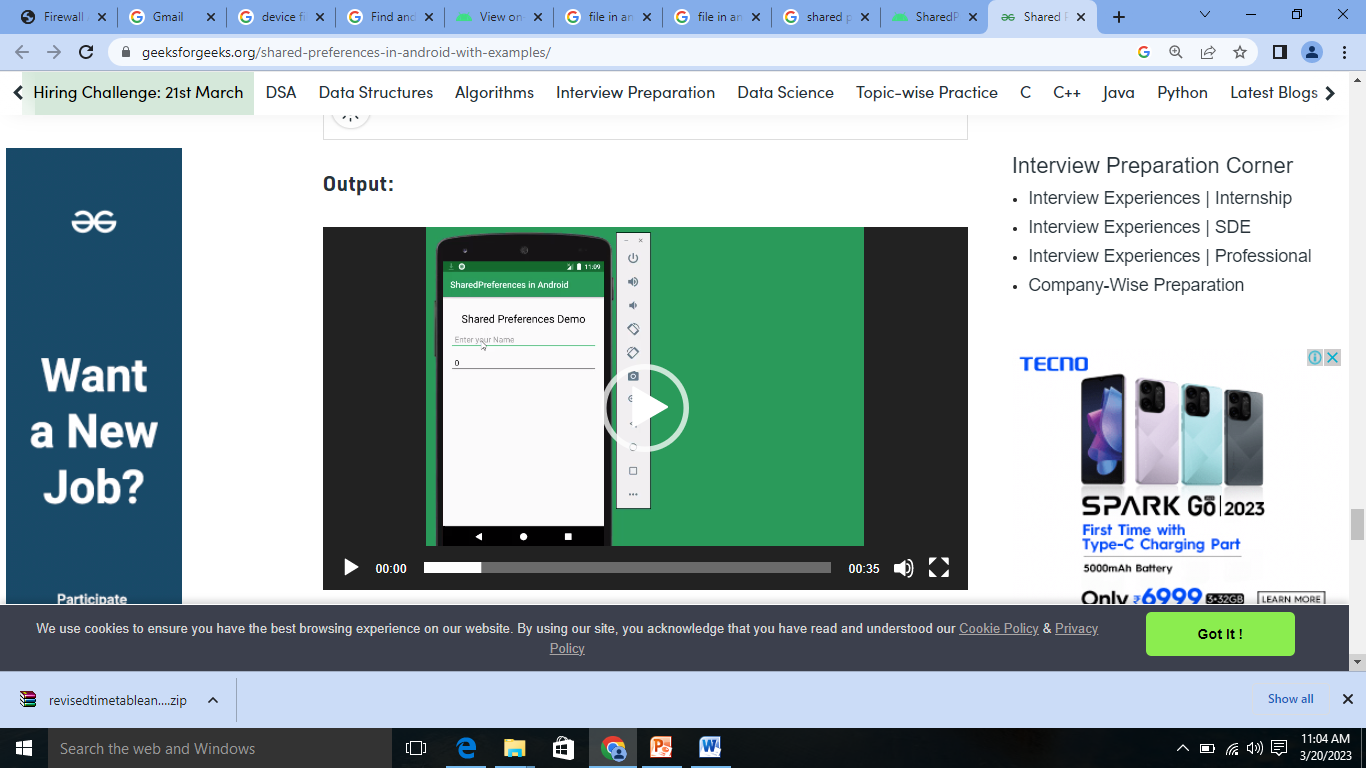
**myEdit.putString("name", name.getText().toString());**

**myEdit.putInt("age", Integer.parseInt(age.getText().toString()));**

**myEdit.apply();**

**}**

**}**



Most of the time in Android uses the SQLite as a mobile database technology. SQLite is a relational database that is used in various embedded on iOS and android platforms. it has a public license so that anybody can use it under the public domain.

What is mobile database in Android?

A Mobile database is **a database that can be connected to a mobile computing device over a mobile network (or wireless network)**. Here the client and the server have wireless connections.

What are mobile based databases?

Definition.

A mobile database is **a database that resides on a mobile device such as a PDA, a smart phone, or a laptop**. Such devices are often limited in resources such as memory, computing power, and battery power.

What is PDA in Android?

**Personal digital assistant** is a term for a small, mobile, handheld device that provides computing and information storage and retrieval capabilities for personal or business use, often for keeping schedules, calendars and address book information handy.

<https://www.tutorialspoint.com/android/android_sqlite_database.htm>