I. Shared Preferences

Interface cho phép lưu trữ và đọc dữ liệu với bằng các cặp *key:value* và lưu dưới dạng một file xml. Dữ liệu của Shared Preferences sẽ được lưu trong ứng dụng android, nếu xoá ứng dụng đi hoặc là xoá dữ liệu app thì dữ liệu này sẽ mất.

Shared Preferences không lưu được Oject, tuy nhiên có thể chuyển sang String và sử dụng thư viện GSON để chuyển đổi. Thường để lưu dữ liệu nhỏ, ít và đơn lẻ ví dụ như High Score của một game hay phần Intro của một app.

1. Cách lưu dữ liệu xuống Shared Prenferences

Khởi tạo một đối tượng kiểu Shared Preferences:

```
SharedPreferences sharedPreferences = getSharedPreferences(SHARED_PREFERENCES_NAME,
Context.MODE_PRIVATE);
```

SHARED_PREFERENCES_NAME: là tên của file Shared Preferences, nơi lưu file này chính là thư mục: *data/data/[application package name]/shared_prefs/shared_preferences_name.xml*

Context.MODE_PRIVATE: đây là một chế độ bảo mật dữ liêu trong android, không cho ứng dụng khác có quyền truy cập vào được.

Tạo đối tượng Editor cho SharedPreferences, thêm dữ liệu lưu trữ:

```
SharedPreferences.Editor editor = sharedPreferences.edit();
editor.putX(String key,value)
editor.commit();//editor.apply();
```

- **X**: là kiểu dữ liệu (float, string, int, boolean, long).
- **key**: là tên dữ liệu.
- **value**: giá trị dữ liệu

Phương thức commit() : hoạt động theo cơ chế đồng bộ Synchronous, thông báo là true hay false nếu như thành công hoặc thất bại.

Phương thức apply() : hoạt động theo cơ chế không đồng bộ Asynchronous, không có kết quả trả về.

2. Lấy dữ liệu từ SharedPreferences

```
SharedPreferences sharedPreferences = getSharedPreferences(SHARED_PREFERENCES_NAME,
Context.MODE_PRIVATE);
```

```
sharedPreferences.getX(String key, defValue) :
```

X là kiểu dữ liệu, defValue là giá trị mặc định trong trường hợp chưa có value tương ứng.

Phương thức remove(String key): Xóa dữ liệu theo tên

Phương thức clear(): Xóa tất cả dữ liệu đang có

II. BỘ NHỚ TRONG (*INTERNAL STORAGE*)

Chỉ có ứng dụng mới có thể truy xuất được mà không có bất kì ứng dụng khác có quyền truy cập kể cả là người dùng, dữ liệu lưu trong Internal Storage sẽ được xoá bỏ khi ứng dụng bị xoá bỏ hoàn toàn hoặc khi người dùng xóa dữ liệu của ứng dụng đi.

Tương tự như SharedPreferences nhưng có thể lưu được Object

1. Ghi dữ liệu

```
public void saveDataDefault(){
        String fileName = "data_file_name";
        String content = "Data data data";
        FileOutputStream outputStream = null;
        try {
            outputStream = openFileOutput(fileName, Context.MODE_PRIVATE);
            outputStream.write(content.getBytes());
            outputStream.close();
            Toast.makeText(this, "Saved successfully", Toast.LENGTH_SHORT).show();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

Phương thức openFileOutput(String name, int mode): Lưu tập tin với **name** là tên tập tin, mode: chế độ lưu, ở trên mình để là Context.MODE_PRIVATE, đây là chế độ mặc định giúp ngăn chặn việc truy cập vào file từ các ứng dụng khác. Dữ liệu hiện tại đang được lưu vào thư mục: /data/data/application_package_name/files.

Để vào thư mục trên ở Android Studio bạn vào : Tools → Android → Android Device Monitor , rồi chọn thiết bị chạy, vào danh mục File Explorer.

```
public void saveDataOnCacher(){
        String content = "Data data";
        File file;
        FileOutputStream outputStream;
        try {
            file = new File(getCacheDir(), "data_name");

            outputStream = new FileOutputStream(file);
```

```
            outputStream.write(content.getBytes());
            outputStream.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

Có thể lưu dữ liệu trên Cache tuy nhiên lưu ở thư mục cache thì theo cơ chế của Android sẽ tự dọn dẹp thư mục Cache khi nó quá tải, nhiều khuyến cáo nên tự xoá nó đi khi dữ liệu vượt quá 1M.

2. Đọc dữ liệu

```
private void readData() {
        try {
            FileInputStream in = this.openFileInput("data_name");

            BufferedReader br= new BufferedReader(new InputStreamReader(in));

            StringBuffer buffer = new StringBuffer();
            String line = null;
            while((line= br.readLine())!= null)  {
                buffer.append(line).append("\n");
            }
            Log.d("read-data:",buffer.toString());

        } catch (Exception e) {
            Toast.makeText(this,"Error:"+ e.getMessage(),Toast.LENGTH_SHORT).show();
        }
    }
public void readData2(){
    BufferedReader input = null;
    File file = null;
    try {
        file = new File(getFilesDir(), "data_name");
        input = new BufferedReader(new InputStreamReader(new FileInputStream(file)));
        String line;
        StringBuffer buffer = new StringBuffer();
        while ((line = input.readLine()) != null) {
            buffer.append(line).append("\n");
        }
        Log.d("read-data:",buffer.toString());

    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

III. BỘ NHỚ NGOÀI (EXTERNAL STORAGE)

Điều này có nghĩa External Storage trong android là phần bộ nhớ mà có thể dùng để chia sẽ dữ liệu . nó có thể là bộ nhớ di động (SDCard), có thể là bộ nhớ trong.Những dữ liệu lưu trong bộ nhớ trong có thể được và chỉnh sửa được bởi người dùng thông quá kết nối USB với máy tính.

Bộ nhớ ngoài trong lập trình android có thể là Internal Storage, không phải chỉ là thẻ nhớ. Giúp cho việc lưu dữ liệu để chia sẽ ra bên ngoài, trao đổi giữa các ứng dụng với nhau.

3. Ghi dữ liệu

Cấu hình AndroidManifest.xml:

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
```

Nếu quyền android.permission.WRITE_EXTERNAL_STORAGE thì không cần phải thêm quyền android.permission.READ_EXTERNAL_STORAGE vì đã bao gồm quyền đọc file và ghi file rồi.

Kiểm tra, thêm quyền

```
private void checkAndRequestPermissions() {
        String[] permissions = new String[]{
                Manifest.permission.WRITE_EXTERNAL_STORAGE,
                Manifest.permission.READ_EXTERNAL_STORAGE
        };
        List<String> listPermissionsNeeded = new ArrayList<>();
        for (String permission : permissions) {
            if (ContextCompat.checkSelfPermission(this, permission) !=
PackageManager.PERMISSION_GRANTED) {
                listPermissionsNeeded.add(permission);
            }
        }
        if (!listPermissionsNeeded.isEmpty()) {
            ActivityCompat.requestPermissions(this, listPermissionsNeeded.toArray(new
String[listPermissionsNeeded.size()]), 1);
        }
    }
```

Kiểm tra thiết bị có bộ nhớ ngoài và có thể ghi file được không?

```
public boolean isExternalStorageReadable() {
    String state = Environment.getExternalStorageState();
    if (Environment.MEDIA_MOUNTED.equals(state) ||
        Environment.MEDIA_MOUNTED_READ_ONLY.equals(state)) {
        return true;
    }
    return false;
}
```

Lưu dữ liệu

```
public void saveData(){
        String content = "data data";
        File file;
        FileOutputStream outputStream;
        try {
            file = new File(Environment.getExternalStorageDirectory(), "data_name");
```

```
            Log.d("MainActivity",
Environment.getExternalStorageDirectory().getAbsolutePath());
            outputStream = new FileOutputStream(file);
            outputStream.write(content.getBytes());
            outputStream.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

Khi chạy trên máy ảo sẽ được lưu vào thư mục: storage/emulated/0, trên máy thật thì nó lưu vào thư mục: /storage/sdcard0/

4. Đọc dữ liệu

```
public void readData(){
        BufferedReader input = null;
        File file = null;
        try {
            file = new File(Environment.getExternalStorageDirectory(), "data_name");

            input = new BufferedReader(new InputStreamReader(new
FileInputStream(file)));
            String line;
            StringBuffer buffer = new StringBuffer();
            while ((line = input.readLine()) != null) {
                buffer.append(line);
            }

            Log.d("MainActivity", buffer.toString());
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```