ĐẠI HỌC CÔNG NGHIỆP TP.HCM
Khoa Công nghệ Thông tin

Chương 4 :

XỬ LÝ FILE, LƯU TRẠNG THÁI ỨNG DỤNG

TS. Huỳnh Hữu Nghĩa
luckerhuynhvn@gmail.com

Nội dung:

- Các lựa chọn lưu trữ dữ liệu (*Shared Preferences, Internal Storage, External Storage, ...*).
- Files
- Shared Preferences

Giới thiệu

- Android cung cấp một số lựa chọn để lưu trữ dữ liệu ứng dụng lâu dài.

- Việc lựa chọn phụ thuộc vào nhu cầu cụ thể, như dữ liệu có nên riêng tư cho ứng dụng của bạn hay cho phép truy cập từ ứng dụng khác và dữ liệu của bạn cần bao nhiêu dung lượng.

- Android cung cấp cách để bạn có thể chia sẽ dữ liệu riêng tư của mình đến các ứng dụng khác thông qua content provider. Content provider là thành phần tùy chọn thể hiện quyền Read/Write đến ứng dụng của bạn, tuân theo những hạn chế mà bạn thiết lập.

Các lựa chọn lưu trữ

- Shared Preferences
  - *Lưu trữ dữ liệu riêng tư theo các cặp key-value.*
- Internal Storage
  - *Lưu trữ dữ liệu riêng tư trên bộ nhớ trong của thiết bị.*
- External Storage
  - *Lưu trữ dữ liệu công khai trên bộ nhớ ngoài của thiết bị.*
- SQLite Databases
  - *Lưu trữ dữ liệu có cấu trúc trong cơ sở dữ liệu riêng.*
- Network Connection
  - *Lưu trữ dữ liệu trên Web với máy chủ mạng của riêng bạn.*

Shared Preferences

- SharedPreferences cung cấp một khung làm việc chung cho phép bạn lưu và truy hồi các cặp **key-value** cho các kiểu dữ liệu cơ bản.

- Bạn có thể sử dụng SharedPreferences để lưu bất kỳ dữ liệu cơ bản: *booleans*, *floats*, *ints*, *longs*, và *strings*.

- Dữ liệu sẽ tồn tại thông qua phiên người dùng *(ngay cả ứng dụng của bạn bị tắt)*

Shared Preferences

❖ Tạo SharedPreferences

Để lấy đối tượng SharedPreferences cho ứng dụng, bạn sử dụng một trong hai phương thức:

- **getSharedPreferences**(String filename, int mode) – *Sử dụng khi bạn cần tham chiếu đến nhiều file. Bạn cần cung cấp tên*
- **getPreferences()** – *Sử dụng khi bạn chỉ cần tham chiếu đến một file cho Activity. Do chỉ dùng có một file nên bạn không cần cung cấp tên.*

Shared Preferences

- **Ghi các giá trị:**
  - *Gọi* **edit()** *để lấy SharedPreferences.Editor.*
  - *Thêm các giá trị với các phương thức* **putXXX()** *– XXX là kiểu dữ liệu, như:* ***putBoolean()*** *và* ***putString().***
  - *Hoàn tất thêm các giá trị mới với* **commit()**

- **Truy hồi các giá trị:**
  - *Sử dụng các phương thức* **getXXX()**, *như:* ***getBoolean()*** *và* ***getString().***

Các MODE

- **MODE_PRIVATE:** *tạo hay thay thế file có cùng tên đang tồn tại với chế độ dùng riêng cho ứng dụng (mặc định).*
- **MODE_APPEND:** *tạo file mới hoặc mở để ghi thêm vào file đang tồn tại và có dữ liệu.*
- **MODE_WORLD_READABLE:** *cho phép các ứng dụng khác có thể đọc dữ liệu từ file.*
- **MODE_WORLD_WRITEABLE:** *cho phép các ứng dụng khác có thể ghi dữ liệu lên file.*
- *...*

Example of Shared Preferences

❖ ***Khởi tạo:***

```
SharedPreferences pref =
getApplicationContext().getSharedPreferences(“myfile", 0);
        // 0 - for private mode
Editor editor = pref.edit();
```

Example of Shared Preferences

❖ ***Ghi dữ liệu:***

```
editor.putBoolean("key_name", true); // Storing boolean - true/false
editor.putString("key_name", "string value"); // Storing string
editor.putInt("key_name", "int value"); // Storing integer
editor.putFloat("key_name", "float value"); // Storing float
editor.putLong("key_name", "long value"); // Storing long


editor.commit(); // commit changes
```

Example of Shared Preferences

❖ ***Truy hồi dữ liệu:***

```
pref.getBoolean("key_name", null); // Getting boolean
pref.getString("key_name", null); // Getting  string
pref.getInt("key_name", -1); // Getting integer
pref.getFloat("key_name", null); // Getting float
pref.getLong("key_name", null); // Getting long
```

Example of Shared Preferences

❖ ***Xóa dữ liệu:***

```
editor.remove("name"); // will delete key name
editor.remove("email"); // will delete key email
editor.commit(); // commit changes
```

```
editor.clear(); // will delete all values
editor.commit(); // commit changes
```

Internal Storage

- Bạn có thể lưu files trực tiếp vào bộ nhớ trong của thiết bị.
- Mặc định, files được lưu trữ ở bộ nhớ trong là riêng tư với ứng dụng của bạn và ứng dụng khác không thể truy cập được.
- Khi người dùng gỡ cài đặt ứng dụng của bạn, files này sẽ bị xóa.
- Nếu bạn muốn lưu file tĩnh trong ứng dụng tại thời điểm biên dịch thì lưu file trong thư mục **res/raw/directory** của project. Bạn có thể mở nó với openRawResource(), thông qua ID tài nguyên R.raw.<filename>. Phương thức này trả về một InputStream mà có thể sử dụng để đọc file *(nhưng không thể ghi vào file gốc)*

Internal Storage

- **Ghi dữ liệu:**
  - Gọi openFileOutput() với tên file và mode, trả về một FileOutputStream.
  - Ghi dữ liệu với write() và Đóng stream với close().

```
String filename = “hello.txt”;
String data = “Welcome to our class”;
FileOutputStream fout = openFileOutput(filename, Context.MODE_PRIVATE);
fout.write(data.getBytes());
fout.close();
```

Internal Storage

- **Truy hồi dữ liệu:**
  - Gọi openFileInput() với tên file trả về một FileInputStream.
  - Đọc dữ liệu với read() và Đóng stream với close().

```
String filename = “hello.txt”;
String data = “”; int c;
FileInputStream fin = openFileInput(filename);
while((c = fin.read()) != -1){
        data += Character.toString((char)c));
}
fin.close();
```

Internal Storage

❖ **Tham khảo thêm một số hàm:**

- **getFileDir():** *trả về đường dẫn tuyệt đối các file đã lưu.*
- **getDir():** *tạo (hay mở) một thư mục bên trong bộ nhớ trong.*
- **deleteFile():** *xóa file được lưu trong bộ nhớ trong.*
- **fileList():** *trả về mảng các file được lưu bởi ứng dụng hiện tại.*

External Storage

- Mỗi thiết bị tương thích Android đều hỗ trợ “bộ nhớ ngoài” (external storage) mà bạn có thể sử dụng để lưu files.

- Có thể là phương tiện lưu trữ di động (như: SD card) hoặc bộ nhớ trong cố định.

- File được lưu trữ trong bộ nhớ ngoài có thể được đọc và sửa đổi bởi người dùng, dữ liệu có thể chuyển files sang máy tính.

External Storage

- Để đọc và ghi files lên bộ nhớ ngoài, ứng dụng của bạn phải có được các quyền READ_EXTERNAL_STORAGE hoặc WRITE_EXTERNAL_STORAGE.

- Quyền này được khai báo trong *Manifest.xml*:

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

External Storage

❖ Do không biết bộ nhớ ngoài đang ở tình trạng thế nào?

```
String state = Enviroment.getExternalStorageState();
if(state.equals(Enviroment.MEDIA_MOUNTED)){
        // Đọc được, ghi được
} else if(state.equals(Enviroment.MEDIA_MOUNTED_READ_ONLY)){
        // Đọc được, không ghi được
} else {
        // Không đọc được và không ghi được
}
```

External Storage

❖ **Truy cập File:**

- ✓ Dùng API Level 8 trở lên dùng **getExternalFilesDir()** để mở file. Có thể truyền thêm đối số chỉ đến loại thư mục con bạn cần. Ví dụ: DIRECTORY_MUSIC, DIRECTORY_RINGTONES, ... (khi để null là truy xuất đến root).

- ✓ Dùng API Level 7 trở xuống dùng **getExternalStorageDirectory()** để mở file.

External Storage

❖ **Lưu File và chia sẽ:**

- ✓ Muốn lưu file có thể dùng chung, không bị mất khi ứng dụng bị bỏ cài đặt thì phải lưu vào các thư mục dùng chung. Những thư mục này nằm ở root.

- ✓ Dùng API Level 8 trở lên sử dụng **getExternalStoragePublicDirectory()** để mở file và truyền vào loại thư mục dùng chung như DIRECTORY_MUSIC, DIRECTORY_PICTURES, DIRECTORY_RINGTONES, ...

External Storage

❖ Lưu File và chia sẽ:

- ✓ Dùng API Level 7 trở xuống dùng **getExternalStorageDirectory()** để mở file và lưu vào các thư mục:
  - **Music**/ – *các file như music*
  - **Podcasts**/ – *các file như podcasts*
  - **Ringtones**/ – *các file như ringtone*
  - **Alarms**/ – *các file như alarm sound*
  - **Notifications**/ – *các file như notification sound.*
  - **Pictures**/ – *các file hình ảnh chụp từ camera.*
  - **Movies**/ – *các đoạn phim quay từ camcorder.*
  - **Download**/ – *các download.*

External Storage

❖ **Ví dụ ghi File:**

```
String filename = “hello.txt";
String data = “Welcome to our class”;
File folder = Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DOWNLOADS);
File myFile = new File(folder, filename);
FileOutputStream fstream = new FileOutputStream(myFile);
fstream.write(data.getBytes());
fstream.close();
```

External Storage

❖ **Ví dụ đọc File:**

```
String filename = “hello.txt";
File folder = Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DOWNLOADS);
File myFile = new File(folder, filename);
FileInputStream fstream = new FileInputStream(myFile);
StringBuffer sbuffer = new StringBuffer();
int i;
while ((i = fstream.read()) != -1)
        sbuffer.append((char)i);
fstream.close();
```

Files Input/Output

- Dữ liệu input/output cần thông qua Stream.

- Stream như đường dẫn:
  - Dẫn dữ liệu từ nguồn đến chương trình để đọc dữ liệu vào (input)
  - Hoặc từ chương trình đến nguồn dữ liệu để lưu (output)

- Stream có 2 loại: InputStream và OutputStream

InputStream

- Đọc dữ liệu byte mảng byte từ nguồn dữ liệu.
- Nguồn dữ liệu có thể là file, chuỗi, hay bộ nhớ.
- Các phương thức phổ biến:
  - **read**(): *đọc dữ liệu*
  - **available**(): *tổng số byte của stream.*
  - **close**(): *đóng stream.*

OutputStream

- Ghi dữ liệu byte mảng byte từ stream đến nguồn dữ liệu.
- Các phương thức phổ biến:
  - **write**(): *ghi dữ liệu*
  - **flush**(): *ép dữ liệu từ stream ghi ra nguồn.*
  - **close**(): *đóng stream.*

Đọc/Ghi file vào Internal Storage

- Là thư mục chỉ đọc, files được chép vào thư mục này trước.

Thư mục Assets

- Là thư mục chỉ đọc, files được chép vào thư mục này trước.

```
String filename = “hello.txt";
InputStream in = getAssets().open(myFile);
int size = in.available();
byte[] buffer = new byte[size];
in.read(buffer);
String st = new String(buffer);
while ((i = fstream.read()) != -1)
        sbuffer.append((char)i);
fstream.close();
```


Q&A