Lưu trữ dữ liệu

Nội dung

- **Giới thiệu**
- Lưu trữ dữ liệu
  - Shared Preferences
  - Android File System
  - Bộ nhớ trong
  - Bộ nhớ ngoài
  - SQLite
  - Network
- Q & A

Giới thiệu

- Android cung cấp cho lập trình viên một số lựa chọn để lưu trữ dữ liệu của ứng dụng. Việc lựa chọn phụ thuộc vào các nhu cầu cụ thể như: dữ liệu có cần được bảo mật, quyền truy cập dữ liệu cho các ứng dụng khác, kích thước dữ liệu, v.v.
- Android cung cấp một cơ chế để truy cập dữ liệu private đó là Trình cung cấp nội dung (Content provider). Đây là một thành phần tùy chọn, cho phép cung cấp quyền đọc/ghi dữ liệu ứng dụng với các giới hạn được áp đặt bởi lập trình viên.

Nội dung

- Giới thiệu
- **Lưu trữ dữ liệu**
  - Shared Preferences
  - Android File System
  - Bộ nhớ trong
  - Bộ nhớ ngoài
  - SQLite
  - Network
- Q & A

Các giải pháp lưu trữ dữ liệu

- Shared Preferences
- Bộ nhớ trong
- Bộ nhớ ngoài
- SQLite
- Network

Các tiêu chí lựa chọn giải pháp lưu trữ dữ liệu

- Khả năng truy cập dữ liệu
  - Từ:
    - Chính ứng dụng đó
    - Ứng dụng khác/Người dùng
  - Khi ứng dụng:
    - Đã gỡ cài đặt hoặc chưa
    - Đã kết nối mạng hoặc chưa
- Dung lượng lưu trữ

Nội dung

- Giới thiệu
- Lưu trữ dữ liệu
  - **Shared Preferences**
  - Android File System
  - Bộ nhớ trong
  - Bộ nhớ ngoài
  - SQLite
  - Network
- Q & A

Shared Preferences

Shared Preferences

- Chứa dữ liệu private nguyên thủy theo dạng key-value
  - booleans, floats, ints, longs, and strings
- Shared preferences không chỉ để lưu trữ thiết lập của người dùng
  - *PreferenceActivity được sử dụng để tạo thiết lập người dùng*
- Shared preferences được lưu trữ trong tệp XML trên thiết bị có đường dẫn như sau:
  /data/data/<application's package name>/shared_prefs
- Sẽ bị xóa bỏ khi gỡ cài đặt ứng dụng

Shared Preferences

- Shared preferences có thể được liên kết với ứng dụng hoặc một hành động cụ thể.
- Sử dụng một trong hai cách sau để lấy đối tượng SharedPreferences
  - getSharedPreferences(String name, int mode) - Sử dụng khi cần nhiều tệp preferences được xác định bởi tên.
  - getPreferences() - Sử dụng khi chỉ cần một tệp preferences cho hành động.
    - Cơ chế bên trong: sử dụng tên lớp của hành động như tên của preferences

Shared Preferences - Đọc

- Đọc preferences sử dụng các phương thức sau:
  - contains(String key) - Kiểm tra xem preferences có chứa một preference cụ thể không
  - getAll() - Lấy toàn bộ values từ preferences
  - getBoolean(String key, boolean defValue) - Lấy một giá trị kiểu boolean từ preferences.
  - getFloat(String key, float defValue) - Lấy một giá trị kiểu float từ preferences.
  - getInt(String key, int defValue) - Lấy một giá trị kiểu int từ preferences.
  - getLong(String key, long defValue) - Lấy một giá trị kiểu long từ preferences.
  - getString(String key, String defValue) - Lấy một giá trị kiểu String từ preferences.
  - getStringSet(String key, Set<String> defValues) - Lấy một tập giá trị kiểu String từ preferences.

Shared Preferences - Ghi

- Các bước ghi giá trị
  - Gọi hàm edit() để nhận SharedPreferences.Editor
  - Thêm giá trị bằng các hàm putBoolean() và putString()
  - Ghi nhận giá trị mới với hàm commit()

```
// We need an Editor object to make preference changes.
// All objects are from android.context.Context
SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
SharedPreferences.Editor editor = settings.edit();
editor.putBoolean("silentMode", mSilentMode);

// Commit the edits!
editor.commit();
```

Nội dung

- Giới thiệu
- Lưu trữ dữ liệu
  - Shared Preferences
  - **Android File System**
  - Bộ nhớ trong
  - Bộ nhớ ngoài
  - SQLite
  - Network
- Q & A

Android File System

Android File System

- Android sử dung Linux kernel tuân thủ VFS (Virtual FileSystem) - một lớp trừu tượng phía trên của các tệp tin hệ thống cụ thể
  - Dễ dàng thêm nhiều kiểu tệp tin hệ thống vào Android
  - Thông thường hỗ trợ EXT2/3/4, F2FS, VFAT, exFAT
  - Pseudo file systems: cgroup, procfs, sysfs, tmpfs
  - Đa số lưu trữ trong được định dạng theo EXT4
- Có cấu trúc tương tự đối với các bản phân phối Linux chuẩn khác.
  - Do đều tuân theo Filesystem Hierarchy Standard
  - Kiểm tra: chạy "adb shell ls -l" trên máy tính kết nối Android với USB debugging được bật.

Android File System

- Cấu trúc thư mục thông thường:
  - etc - Một liên kết tượng trưng tới /system/etc
  - proc - mount point cho tệp tin hệ thống procfs, cho phép truy cập tới cấu trúc dữ liệu kernel. Ứng dụng như ps, lsof, và vmstat sử dụng /proc như nguồn dữ liệu
  - root - Thư mục gốc cho tài khoản root
  - sdcard - Một liên kết tượng trưng tới một thư mục trong /storage/*
  - sys - mount point cho tệp tin hệ thống sysfs pseudo, là sự ánh xạ của cấu trúc đối tượng thiết bị của kernel.
  - system - mount point cho /dev/block/mtdblock0. Các thư mục trong đây thường được nhìn thấy trong thư mục root của các bản phân phối tiêu chuẩn Linux bao gồm các thư mục bin, etc, lib, usr, và xbin.
  - vendor - một liên kết tượng trưng tới /system/vendor chứa các đoạn mã thuộc về nhà sản xuất.

Nội dung

- Giới thiệu
- Lưu trữ dữ liệu
  - Shared Preferences
  - Android File System
  - **Bộ nhớ trong**
  - Bộ nhớ ngoài
  - SQLite
  - Network
- Q & A

Bộ nhớ trong

Bộ nhớ trong

- Có thể lưu trực tiếp các tệp tin vào bộ nhớ trong của thiết bị.
- Thông thường, các tệp tin lưu vào bộ nhớ trong sẽ là private đối với ứng dụng của bạn, các ứng dụng khác không thể truy cập.
- Các tệp tin này được liên kết tới ứng dụng của bạn và bị xóa khi ứng dụng bị gỡ bỏ cài đặt.

Bộ nhớ trong - Đọc

- Để đọc một tệp tin từ bộ nhớ trong:
  - Gọi openFileInput() và truyền vào tên của tệp tin cần đọc. Hàm sẽ trả lại một FileInputStream.
  - Đọc bytes từ tệp tin với hàm read()
  - Đóng luồng với hàm close()

```
FileInputStream fis = context.openFileInput("hello.txt");
InputStreamReader isr = new InputStreamReader(fis);
BufferedReader bufferedReader = new BufferedReader(isr);
StringBuilder sb = new StringBuilder();
String line;
while ((line = bufferedReader.readLine()) != null) {
  sb.append(line);
}
fis.close();
```

Bộ nhớ trong - Ghi

- Đọc và ghi một tệp private trong bộ nhớ trong:
  - Gọi hàm openFileOutput() với tên của tệp tin và chế độ họa động. Hàm trả về một FileOutputStream.
  - Ghi tệp sử dụng hàm write()
  - Đóng tệp sử dụng hàm close()

```
String FILENAME = "hello_file";
String string = "hello world!";

FileOutputStream fos = openFileOutput(FILENAME, Context.MODE_PRIVATE);
fos.write(string.getBytes());
fos.close();
```

Bộ nhớ trong - Ghi

- Chế độ openFileOutputI).
  - MODE_PRIVATE - chế độ mặc định, tệp tin được tạo ra chỉ có thể truy cập bởi ứng dụng gọi nó (hoặc các ứng dụng có chung user ID)
  - MODE_APPEND - nếu tệp tin đã tồn tại, tiếp tục ghi dữ liệu cho tới cuối tệp tin thay vì xóa bỏ tệp tin.
  - ~~MODE_WORLD_WRITEABLE~~ (deprecated from API level 17) - Tạo ra các tệp tin world-writable là rất nguy hiểm, dễ gây ra các lỗ hổng bảo mật. Ứng dụng nên sử dụng các cơ chế khác formal hơn cho tương tác như ContentProvider, BroadcastReceiver, and Service.
  - ~~MODE_WORLD_READABLE~~ (deprecated from API level 17)

Bộ nhớ trong - Tệp tin cache

- Sử dụng phương thức getCacheDir() để mở một File đại diện cho thư mục lưu trữ tạm thời.
- Các tệp cache sẽ tự động được xóa bởi hệ thống nếu cần chỗ trống trên đĩa.
  - Hệ thống sẽ luôn luôn xóa các tệp cũ trước bằng cách sử dụng phương thức lastModified().
  - Một cách có kiểm soát hơn là sử dụng setCacheBehaviorGroup(File, boolean) và setCacheBehaviorTombstone(File, boolean).
- Khuyến khích giữ dung lượng cache sử dụng dưới định mức quy đinh bởi getCacheQuotaBytes(java.util.UUID). Định mức thay đổi theo thời gian phụ thuộc vào:
  - Tần suất người dùng tương tác với ứng dụng
  - Dung lượng đĩa system-wide disk sử dụng

Bộ nhớ trong - Các phương thức hữu ích khác

- getFilesDir() - Lấy đường dẫn tuyệt đối tới thư mục filesystem, nơi mà các tệp internal được lưu trữ.
- getDir() - Tạo (hoặc mở nếu đã tồn tại) thư mục bên trong bộ nhớ trong
- deleteFile() - Xóa tệp tin trong bộ nhớ trong
- fileList() - Trả về mảng tệp tin hiện tại được lưu trữ bởi ứng dụng.

Nội dung

- Giới thiệu
- Lưu trữ dữ liệu
  - Shared Preferences
  - Android File System
  - Bộ nhớ trong
  - **Bộ nhớ ngoài**
  - SQLite
  - Network
- Q & A

Bộ nhớ ngoài

Bộ nhớ ngoài

- Các thiết bị Android cung cấp "bộ nhớ ngoài" được chia sẻ để lưu trữ các tệp tin.
- Có thể là các thiết bị lưu trữ có thể tháo bỏ (ví dụ SD card) hoặc bộ nhớ trong (không thể tháo bỏ).
- Tệp tin được lưu trữ trong bộ nhớ ngoài là world-readable, có thể bị sử đổi khi người dùng bật USB mass storage để chuyển tệp tin trên máy tính.

Bộ nhớ ngoài - Yêu cầu quyền truy cập

- Yêu cầu quyền hệ thống READ_EXTERNAL_STORAGE hoặc WRITE_EXTERNAL_STORAGE để đọc và ghi các tệp tin.
  - Quyền đọc được ngầm yêu cầu bởi quyền ghi.

```
<manifest ...>

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

    ...

</manifest>
```

Bộ nhớ ngoài - Kiểm tra tính khả dụng

- Do bộ nhớ lưu trữ ngoài có thể bị tháo bỏ, nên sử dụng phương thức getExternalStorageState() để kiểm tra trạng thái sẵn sàng của thiết bị trước khi thực hiện các hành động khác.

```
/* Checks if external storage is available for read and write */
public boolean isExternalStorageWritable() {
  String state = Environment.getExternalStorageState();
  return Environment.MEDIA_MOUNTED.equals(state));
}

/* Checks if external storage is available to at least read */
public boolean isExternalStorageReadable() {
  String state = Environment.getExternalStorageState();
  return Environment.MEDIA_MOUNTED.equals(state) ||
         Environment.MEDIA_MOUNTED_READ_ONLY.equals(state);
}
```

Bộ nhớ ngoài - Tệp tin công khai

- Luôn sẵn sàng sử dụng bởi các ứng dụng khác và người dùng
- Khi người dùng gỡ cài đặt ứng dụng, các tệp tin này nên tiếp tục sẵn sàng cho người dùng sử dụng.
- Gọi getExternalStoragePublicDirectory(String type) để nhận thư mục lưu trữ các tệp tin công khai.
  - ***Tham số type*** thuộc vào các giá trị sau: DIRECTORY_MUSIC, DIRECTORY_PODCASTS, DIRECTORY_RINGTONES, DIRECTORY_ALARMS, DIRECTORY_NOTIFICATIONS, DIRECTORY_PICTURES, DIRECTORY_MOVIES, DIRECTORY_DOWNLOADS, DIRECTORY_DCIM, hoặc DIRECTORY_DOCUMENTS
  - Đường dẫn trả về bởi phương thức này có thể không tồn tại, nên sử dụng path.mkdirs() để đảm bảo điều này.

Bộ nhớ ngoài - Tệp tin công khai

- Path = “<external storage path>/” + DIRECTORY_*
  - DIRECTORY_MUSIC = "Music";
  - DIRECTORY_PODCASTS = "Podcasts";
  - DIRECTORY_RINGTONES = "Ringtones";
  - DIRECTORY_ALARMS = "Alarms";
  - DIRECTORY_NOTIFICATIONS = "Notifications";
  - DIRECTORY_PICTURES = "Pictures";
  - DIRECTORY_MOVIES = "Movies";
  - DIRECTORY_DOWNLOADS = "Download";
  - DIRECTORY_DCIM = "DCIM";

Bộ nhớ ngoài - Tệp tin công khai

```
public File getAlbumStorageDir(String albumName) {
    // Get the directory for the user's public pictures directory.
    File file = new File(Environment.getExternalStoragePublicDirectory(
            Environment.DIRECTORY_PICTURES), albumName);
    if (!file.mkdirs()) {
        Log.e(LOG_TAG, "Directory not created");
    }
    return file;
}
```

Bộ nhớ ngoài - Tệp tin private

- Vẫn có thể truy cập bởi người dùng và ứng dụng khác, nhưng không cung cấp giá trị cho người dùng bên ngoài ứng dụng.
- Khi người dùng gỡ bỏ cài đặt, hệ thống xóa toàn bộ tệp tin trong thư mục external private
- Gọi phương thức getExternalFilesDir(String type) để nhận thư mục lưu trữ các tệp tin private.

```
void deleteExternalStoragePrivateFile() {
    // Get path for the file on external storage.  If external
    // storage is not currently mounted this will fail.
    File file = new File(getExternalFilesDir(null), "DemoFile.jpg");
    if (file != null) {
        file.delete();
    }
}
```

Nội dung

- Giới thiệu
- Lưu trữ dữ liệu
  - Shared Preferences
  - Android File System
  - Bộ nhớ trong
  - Bộ nhớ ngoài
  - **SQLite**
  - Network
- Q & A

SQLite

SQLite

- Android có sẵn hỗ trợ cho cơ sở dữ liệu SQLite, có trong gói android.database.sqlite
- Phù hợp với các dữ liệu lặp hay có cấu trúc, ví dụ dữ liệu liên lạc.
- Android không áp dụng bất kì giới hạn nào ngoài các khái niệm chuẩn của SQLite
  - Khuyến khích có sử dụng một giá trị khóa tăng dần như một ID duy nhất hỗ trợ cho việc tăng tốc tìm kiếm.
- Android lưu dữ liệu trên một vùng đĩa private được liên kết với ứng dụng.

SQLite - Định nghĩa lược đồ và Contract

- Lớp Contract chỉ định rõ khung của lược đồ dữ liệu theo một cách có hệ thống và tự tài liệu hóa.
  - Chứa các hằng định nghĩa tên cho URIs, bảng, và cột.
  - Cho phép sử dụng chung hằng trong toàn bộ các lớp trong cùng package.
- Cách tổ chức lớp Contract được khuyến khích:
  - Đặt các định nghĩa có tính tổng quát đối với cơ sở dữ liệu vào mức root của lớp.
  - Sau đó tạo inner class cho mỗi bảng liệt kê các cột của nó

SQLite - Định nghĩa lược đồ và Contract

```
public final class FeedReaderContract {
    // To prevent someone from accidentally instantiating the contract class,
    // make the constructor private.
    private FeedReaderContract() {}

    /* Inner class that defines the table contents */
    public static class FeedEntry implements BaseColumns {
        public static final String TABLE_NAME = "entry";
        public static final String COLUMN_NAME_TITLE = "title";
        public static final String COLUMN_NAME_SUBTITLE = "subtitle";
    }
}
```

SQLite - Tạo cơ sở dữ liệu

```
public class FeedReaderDbHelper extends SQLiteOpenHelper {
    private static final String SQL_CREATE_ENTRIES =
        "CREATE TABLE " + FeedEntry.TABLE_NAME + " (" +
        FeedEntry._ID + " INTEGER PRIMARY KEY," +
        FeedEntry.COLUMN_NAME_TITLE + " TEXT," +
        FeedEntry.COLUMN_NAME_SUBTITLE + " TEXT)";

    // If you change the database schema, you must increment the database version.
    // ...
    // ...
    public FeedReaderDbHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(SQL_CREATE_ENTRIES);
    }
}
```

SQLite - Đọc từ cơ sở dữ liệu

```
SQLiteDatabase db = mDbHelper.getReadableDatabase();

// Define a projection that specifies which columns from the database
// you will actually use after this query.
String[] projection = {FeedEntry._ID, FeedEntry.COLUMN_NAME_TITLE,
FeedEntry.COLUMN_NAME_SUBTITLE};
// Filter results WHERE "title" = 'My Title'
String selection = FeedEntry.COLUMN_NAME_TITLE + " = ?";
String[] selectionArgs = { "My Title" };
// How you want the results sorted in the resulting Cursor
String sortOrder = FeedEntry.COLUMN_NAME_SUBTITLE + " DESC";

Cursor cursor = db.query(
    FeedEntry.TABLE_NAME,                     // The table to query
    projection,                               // The columns to return
    selection,                                // The columns for the WHERE clause
    selectionArgs,                            // The values for the WHERE clause
    null,                                     // don't group the rows
    null,                                     // don't filter by row groups
    sortOrder                                 // The sort order
    );
```

SQLite - Ghi vào cơ sở dữ liệu

- Chèn dữ liệu vào cơ sở dữ liệu bằng cách truyền một đối tượng ContentValues vào phương thức insert().

```
// Gets the data repository in write mode
SQLiteDatabase db = mDbHelper.getWritableDatabase();

// Create a new map of values, where column names are the keys
ContentValues values = new ContentValues();
values.put(FeedEntry.COLUMN_NAME_TITLE, title);
values.put(FeedEntry.COLUMN_NAME_SUBTITLE, subtitle);

// Insert the new row, returning the primary key value of the new row
long newRowId = db.insert(FeedEntry.TABLE_NAME, null, values);
```

SQLite - Xóa từ cơ sở dữ liệu

- Để xóa các hàng khỏi bảng, cần cung cấp điều kiện chọn để xác định các hàng đó.

```
// Define 'where' part of query.
String selection = FeedEntry.COLUMN_NAME_TITLE + " LIKE ?";
// Specify arguments in placeholder order.
String[] selectionArgs = { "MyTitle" };
// Issue SQL statement.
db.delete(FeedEntry.TABLE_NAME, selection, selectionArgs);
```

Realm database cho Android

- Giải pháp thay thế SQlite
- Realm nhanh hơn khoảng 2-10 lần trong các tác vụ đọc, ghi so với SQLitethuần và một số thư viện ORM phổ biến hiện nay.

Nội dung

- Giới thiệu
- Lưu trữ dữ liệu
  - Shared Preferences
  - Android File System
  - Bộ nhớ trong
  - Bộ nhớ ngoài
  - SQLite
  - **Network**
- Q & A

Network

Network

- Lưu dữ liệu trên máy tính khác/đám mây, truy cập qua mạng
- Để thực hiện các thao tác liên quan tới mạng, sử dụng các gói sau:
  - java.net.*
  - android.net.*