Content Providers

Dr Tadhg O'Sullivan

Outline

- Content Provider Basics
- Using Content Provider
- Creating Content Provider

Content Providers

- Manage access to a structured set of data, encapsulate the data, and provide mechanisms for defining data security
- Is the standard interface that connects data in one process with code running in another
- Is the only practical way for applications to exchange data (except for 3rd party services and external SD Card)

Native Android Content Providers

- Lots of Android services are available to your app as content providers (considering you got correct permissions)
  - Browser
  - CallLog
  - Contacts
  - MediaStore
  - Settings
  - UserDictionary

Content Providers: Basics

- Android is fully responsible for the lifecycle of Content Providers
- Internal implementation of a content provider, *i.e. how it actually stores data is up to its* software developer
  - *Remember, there are four different ways to CRUD data in Android!*
- All content providers implement a common interface for CRUDing data

Content Providers: Basics

- Content Providers allow two types of access:

- SQL-like – using the same methods as SQLite
- File-like – OutputStream and InputStream (preferable instead of quering BLOB)

Using Content Providers

- Make sure you have permissions *<usespermission android:name=”android.permission.READ_ USER_DICTIONARY”>*
- Get Reference to a Content Provider with ContentResolver:

```
ContentResolver cr =getContentResolver();
```

- Send your CRUD query...

Using Content Providers: Query

- Query parameters:
  - Uri (from table)
  - Projection (columns)
  - Selection (criteria)
  - SelectionArgs
  - SortOrder

- SQL Comparison:
  - FROM table_name
  - col,col,col
  - WHERE col=value
  - ORDER BY

Content Providers: URI

- Content URI syntax
  - content://authority/path/id

- URI examples
  - content://constants
  - content://contacts/people
  - content://ie.ucd.info/course/30480

Examples:

- Query

  Cursor mycursor =

  getContentResolver().query(MyProvider.

  CONTENT_URI, columns, selection, args,sortOrder);
- Query example: return all rows (select * from)

  Cursor allRows =

  getContentResolver().query(MyProvider.

  CONTENT_URI, null, null, null, null);

Content Provider File Access Example

```
Uri uri =
getContentResolver().insert(MyProvider.
                                        CONTENT_URI, newValues);
try
{
    OutputStream outStream =
    getContentResolver().openOutputStream(uri);
    sourceBitmap.compress(Bitmap.
    CompressFormat.JPEG, 50, outStream);
}
catch (FileNotFoundException e) { }
```

Do You Need a Content Provider?

- Content Providers are meant to share your data with other applications, but you can use it within your application as well
- You want to offer complex data or files to other applications or allow users to copy complex data into other apps
- However, you don't need a content provider if all you need is to use SQLite database within your project

Creating a Content Provider

- Design the raw storage:

  Files vs “Table-like” data
- Define the authority string and content URI
- Implement ContentProvider class and its methods
- Add sample data, server synchronisation

Deciding the Raw Storage

- If you need to store binary objects (BLOBs), choose data storage options:
  - Internal file system
  - SD card
  - Network
- If you need to store table-like data, *a la* structured, data, use SQLite DB (or network)

Declaring Content Provider

```
<provider
android:name="ie.ucd.CourseInfoProvider"
android:authorities="ie.ucd.courseinfoprovider" />
...
</provider>
```

Implementing Content Provider

- Extend ContentProvider
- Use **onCreate()** to initialise your storage
- Override **insert()**, **delete()**, **update()**, and **query()** methods

Questions

- Please ask in the Student Forum