Content Providers

Tim Haspert
Make sure you Commit **and push** all of your work as you are developing your apps!
Content Providers

- Content providers manage access to a structured set of data.

- Use the ContentResolver object in your application’s Context to communicate with the provider as a client.

- Android includes content providers that manage data such as audio, video, images and personal contact information.
Content Providers

- You **don’t** need to develop your own if you don’t intend to share data with other apps

- You **do** need your own to provide custom search suggestions in your app

- You **do** need your own if you want to copy and paste complex data / files to other apps
A content provider manages access to a central repository of data. A provider is part of an Android application, which often provides its own UI for working with the data. However, content providers are primarily intended to be used by other applications, which access the provider using a provider client object. Together, providers and provider clients offer a consistent, standard interface to data that also handles inter-process communication and secure data access.

-developer.android.com
Content Provider Basics

Accessing a Provider -

- Apps accesses the data from a content provider with a ContentResolver object.
- The content resolver methods provide the basic “CRUD” functions of persistent storage.
Content Provider Basics

Accessing a Provider -
sample code from developer.android.com

```java
// Queries the user dictionary and returns results
mCursor = getContentResolver().query(
    UserDictionary.Words.CONTENT_URI, // The content URI of the words table
    mProjection, // The columns to return for each row
    mSelectionClause, // Selection criteria
    mSelectionArgs, // Selection criteria
    mSortOrder); // The sort order for the returned rows
```
Content Provider Basics

Accessing a Provider -

Query() compared to SQL query

<table>
<thead>
<tr>
<th>query() argument</th>
<th>SELECT keyword/parameter</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uri</td>
<td>FROM <code>table_name</code></td>
<td>Uri maps to the table in the provider named <code>table_name</code>.</td>
</tr>
<tr>
<td>projection</td>
<td><code>col, col, col, ...</code></td>
<td>projection is an array of columns that should be included for each row retrieved.</td>
</tr>
<tr>
<td>selection</td>
<td>WHERE <code>col = value</code></td>
<td>selection specifies the criteria for selecting rows.</td>
</tr>
<tr>
<td>selectionArgs</td>
<td>(No exact equivalent. Selection arguments replace <code>?</code> placeholders in the selection clause.)</td>
<td></td>
</tr>
<tr>
<td>sortOrder</td>
<td>ORDER BY <code>col, col, ...</code></td>
<td>sortOrder specifies the order in which rows appear in the returned Cursor.</td>
</tr>
</tbody>
</table>
Content Provider Basics

Content URI -

- Includes the symbolic name of the entire provider and a name that points to a table

  - scheme://authority/table

  - content://user_dictionary/words
Content Provider Basics

Requesting Read Access Permission -

- Can’t be done at run-time

- You have to specify that you need this permission in your manifest file

Read access to one or more tables

READ_CONTACTS, specified in AndroidManifest.xml with the <uses-permission> element as <uses-permission android:name="android.permission.READ_CONTACTS">.

Write access to one or more tables

WRITE_CONTACTS, specified in AndroidManifest.xml with the <uses-permission> element as <uses-permission android:name="android.permission.WRITE_CONTACTS">.
The Contacts Provider is a powerful and flexible Android component that manages the device's central repository of data about people. The Contacts Provider is the source of data you see in the device's contacts application, and you can also access its data in your own application and transfer data between the device and online services. The provider accommodates a wide range of data sources and tries to manage as much data as possible for each person, with the result that its organization is complex. Because of this, the provider's API includes an extensive set of contract classes and interfaces that facilitate both data retrieval and modification.
Contacts Provider

Contacts Provider Organization -

- `ContactsContract.Contacts`
  - Rows representing people

- `ContactsContract.RawContacts`
  - Summary of person’s data
  - Most data stored in .data

- `ContactsContract.Data`
  - Rows containing the details
  - ex - phone # or email
Contacts Provider Organization -

- **ContactsContract.Contacts**
  - Rows representing people

- **ContactsContract.RawContacts**
  - Summary of person’s data
  - Most data stored in .data

- **ContactsContract.Data**
  - Rows containing the details
  - ex - phone # or email
## Contacts Provider

### Raw Contacts -

<table>
<thead>
<tr>
<th>Column name</th>
<th>Use</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCOUNT_NAME</td>
<td>The account name for the account type that's the source of this raw contact. For example, the account name of a Google account is one of the device owner's Gmail addresses. See the next entry for ACCOUNT_TYPE for more information.</td>
<td>The format of this name is specific to its account type. It is not necessarily an email address.</td>
</tr>
<tr>
<td>ACCOUNT_TYPE</td>
<td>The account type that's the source of this raw contact. For example, the account type of a Google account is com.google. Always qualify your account type with a domain identifier for a domain you own or control. This will ensure that your account type is unique.</td>
<td>An account type that offers contacts data usually has an associated sync adapter that synchronizes with the Contacts Provider.</td>
</tr>
<tr>
<td>DELETED</td>
<td>The &quot;deleted&quot; flag for a raw contact.</td>
<td>This flag allows the Contacts Provider to maintain the row internally until sync adapters are able to delete the row from their servers and then finally delete the row from the repository.</td>
</tr>
</tbody>
</table>
Contacts Provider

Data from Sync Adapters -
Contacts Provider

Data from Sync Adapters -

- Contacts Provider is designed to sync.

- Android provides plug-in sync framework
  - Checks network availability
  - Schedule and execute sync, based on user prefs
  - Restart syncs that have stopped

- Implement sync adapter as subclass of AbstractThreadedSyncAdapter
Contacts Provider

Data -

- Descriptive Column Names
  
  `RAW_CONTACT_ID`
  The value of the _ID column of the raw contact for this data.

- Generic Column Names
  
  Data 1 - Data 15

- Type Specific Column Names
  
  Simply gives a constant name to help access data

**CAUTION:** Using a name that is also a “Descriptive Column Name” could result in data loss!
Contacts Provider

Contacts Provider Access -

- Querying Entities (already reviewed)
- Batch Modification
  Whenever possible, insert, update and delete data in “Batch Mode”;
  create an ArrayList and call applyBatch()
- Retrieval and Modification with Intents
  Allows user to do contacts-related work with selected UI
- Data Integrity
  Contact data is sensitive and needs to be accurate, handle with care
Contacts Provider

Social Stream Data -

- Use Social Stream Tables to access:
  - Social Stream Text
  - Social Stream Photos
  - Social Stream Interactions

- You must register your sync adapter!

- You can interact with social networks in app!
Contacts Provider

Additional Contacts Provider Features -

- Contact Groups
  - Label related contacts as a group

- Contact Photos
  - Accessed the same as other contact data
  - Separate from Social Stream Photos
Contacts Provider

Additional Contacts Provider Features -

● **Contact Groups**
  ○ Label related contacts as a group

● **Contact Photos**
  ○ Accessed the same as other contact data
  ○ Separate from Social Stream Photos
Content Providers

Questions?