Interacting with Content Provider Content

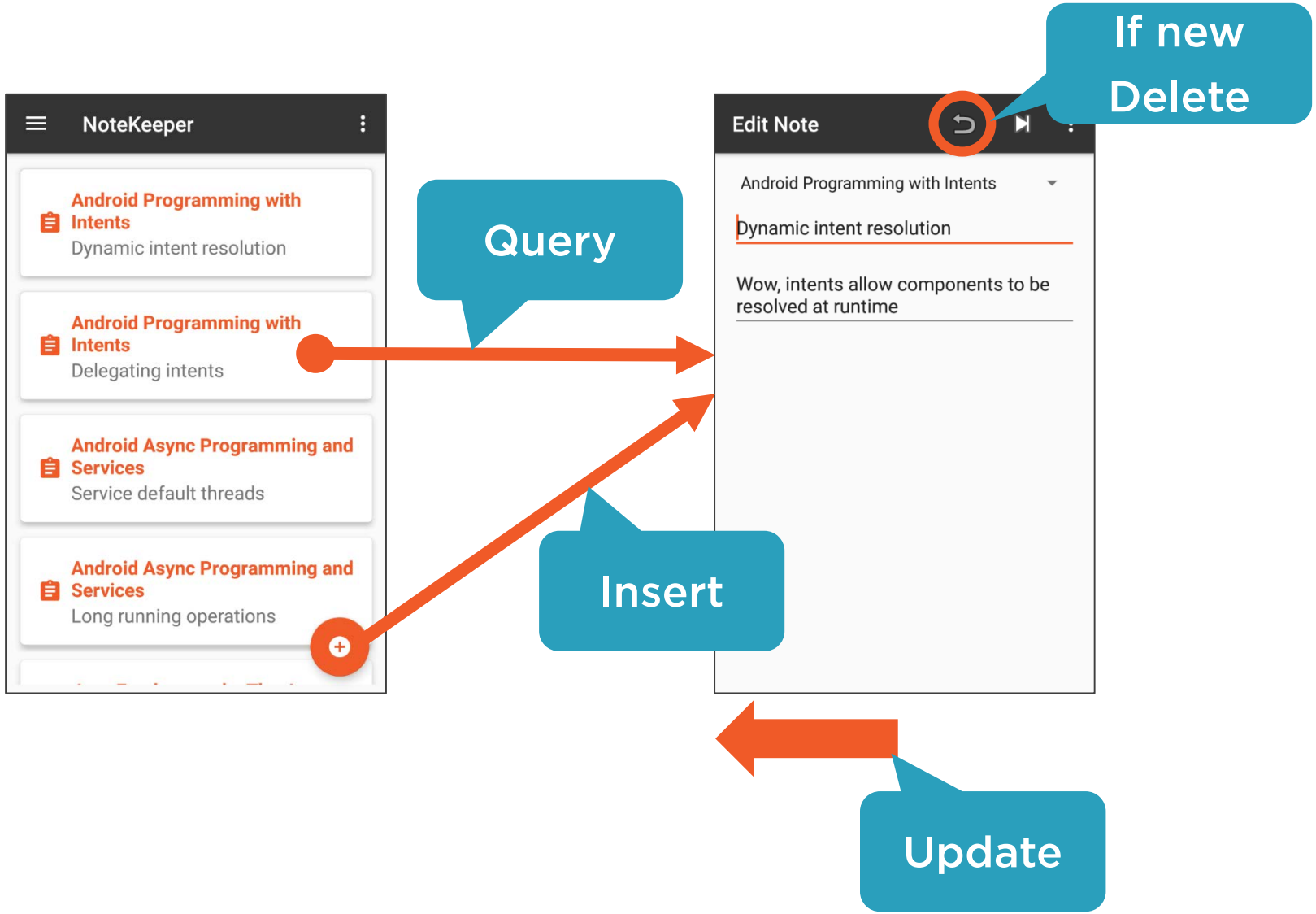
Jim Wilson
MOBILE SOLUTIONS DEVELOPER & ARCHITECT

@hedgehogjim blog.jwhh.com

What to Expect from This Module

- ContentResolver
- Inserting ContentProvider Data
- Row URIs
- Interacting with Other Applications
- Returning the Appropriate Mime Type

Note Operations in Our App



Content Provider Operations

Support range of data operations

- Query
- Insert
- Update
- Delete

Accessing a content provider

- CursorLoader commonly used for query
- Need a way to handle non-query

Accessing a Content Provider

Accessing a Content Provider

Content Resolver

Available from current context

- Use getContentResolver method
- Returns ContentResolver reference

Serves as operation intermediary

- Exposes methods for each operation
- Methods accept a URI
- Locates content provider
- Delegates operation to content provider

Inserting a Row into a Table

Use ContentResolver insert method

- Pass URI of table
- Pass values as ContentValues
- Returns URI of new row

Inserting a Row into a Table

Handled by ContentProvider insert method

- Receives URI and ContentValues
- Determine target with UriMatcher.match
- Insert into SQLite database
- Return row URI

Row URI

- Based on table URI
- Append row ID value as path to end
- Use ContentUris.withAppendedId

Data Interaction and Row URIs

Row URI

- Based on table URI
- Row ID appended as path to end

Primary way to interact with a specific row

- When querying for specific row
- When updating or deleting specific row
- Behaves the same as passing the table URI with “_id=?” selection criteria

Data Interaction and Row URIs

Content Provider URI Handling

- Need to handle table URIs
- Need to handle row URIs

Data Interaction and Row URIs

UriMatcher and row URIs

- Match Row URI separate from table URI
- Supports wildcard matching for ID value
- Use # in place of ID value in addURI call

```
content://com.jwhh.jim.notekeeper.provider/notes/#
```

Data Interaction and Row URIs

Content Provider handling of row URIs

- Database needs “_id=?” selection criteria
- Extract _ID value from URI
- Use ContentUris.parseId

Interacting with Other Applications

Content providers accessible by other apps

- App can interact with data
- App may not now much about the data

Can make more information available

- Make contract class available
- Identify the mime type of the data returned by each URI

Make Contract Class Available

Build jar or Android library

- Contains only the contract class
- http://bit.ly/buildandroidlib

Other app references jar/library

- Allows app to use the constants contained in the contract class

“OtherApp” Using Content Provider

```
public Loader<Cursor> onCreateLoader(int id, Bundle args) {

  Uri uri =
    Uri.parse("content://com.jwhh.jim.notekeeper.provider/courses");

  String[] columns = {
    "_id",
    "course_title",
    "course_id" };

  return new CursorLoader(this, uri, columns,
    null, null, "course_title");

}
```

“OtherApp” Using Content Provider

```
#import com.jwhh.jim.notekeeper.NoteKeeperProviderContract.Courses;

public Loader<Cursor> onCreateLoader(int id, Bundle args) {
  Uri uri = Courses.CONTENT_URI;

  String[] columns = {
    "_id",
    "course_title",
    "course_id" };

  return new CursorLoader(this, uri, columns,
    null, null, "course_title");
}
```

“OtherApp” Using Content Provider

```
#import com.jwhh.jim.notekeeper.NoteKeeperProviderContract.Courses;

public Loader<Cursor> onCreateLoader(int id, Bundle args) {

  Uri uri = Courses.CONTENT_URI;


  String[] columns = {
    Courses._ID,
    Courses.COLUMN_COURSE_TITLE,
    Courses.COLUMN_COURSE_ID };

  return new CursorLoader(this, uri, columns,
    null, null, Courses.COLUMN_COURSE_TITLE);

}
```

Identify Each URIs Mime Type

Mime types describe data

- Many common mime types exist

Our data doesn't fit common mime types

- Cursor based data
- Data is application defined
- We need to construct or own mime type

Identify Each URIs Mime Type

Constructing our mime type

- Table name
- Content provider authority
- Identify as vendor mime type
- Whether returns one or multiple rows

Identify Each URIs Mime Type

```
vnd.android.cursor.item/
                        vnd.com.jwhh.jim.notekeeper.provider.courses
 vnd.android.cursor.dir/
```

Identify Each URIs Mime Type

```
content://com.jwhh.jim.notekeeper.provider/courses
```

```
vnd.android.cursor.dir/vnd.com.jwhh.jim.notekeeper.provider.courses
```

Identify Each URIs Mime Type

`content://com.jwhh.jim.notekeeper.provider/courses/8`

`vnd.android.cursor.item/vnd.com.jwhh.jim.notekeeper.provider.courses`

Identify Each URIs Mime Type

Implement getType method

- Return appropriate mime type
- Build with simple string concatenation

ContentResolver helpful constants

- CURSOR_DIR_BASE_TYPE
- CURSOR_ITEM_BASE_TYPE

Summary

ContentResolver

- Provides access to ContentProvider
- Exposes methods for each operation
- Methods receive a target URI
- Delegates operation to ContentProvider

Summary

ContentProvider operations

- query, insert, update, delete
- Normally backed by SQLite operation
- Need to handle table URIs
- Need to handle row URIs

Summary

Row URI

- References specific row in a table
- Based on table URI
- Row ID appended as path

Insert method

- Returns row URI of inserted row

Handling row URIs in query, update, delete

- Use UriMatcher URI pattern ending in #
- Perform DB operation using “_id = ?” selection criteria

Summary

Improving interaction with other apps

- Expose contract class in jar/library
- Implement getType method to return mime types of supported URIs