ContentProvider

Today's Topics

ContentProvider & ContentResolver

ContentResolver methods

CursorLoader

Implementing ContentProviders

ContentProvider

Represents a repository of structured data

Encapsulates data sets

Enforces data access permissions

ContentProvider

Intended for inter-application data sharing

Clients access ContentProviders through a ContentResolver

ContentResolver

Presents a database-style interface for reading & writing data

- query, insert, update, delete, etc.

Provides additional services such as change notification

ContentResolver

Get reference to ContentResolver by calling Context.getContentResolver()

ContentProvider & ContentResolver

Together these classes let code running in one process access data managed by another process

Android ContentProviders

Browser – bookmarks, history

Call log– telephone usage

Contacts – contact data

Media – media database

UserDictionary – database for predictive spelling

Many more

ContentProvider Data Model

Data represented logically as database tables

_ID	artist
13	Lady Gaga
44	Frank Sinatra
45	Elvis Presley
53	Barbara Streisand

URI

Content providers referenced by URIs

The format of the URI identifies specific data sets managed by specific ContentProviders

FORMAT

CONTENT://AUTHORITY/PATH/ID

CONTENT – SCHEME INDICATING DATA THAT IS MANAGED BY A CONTENT PROVIDER

AUTHORITY – ID FOR THE CONTENT PROVIDER

PATH – 0 OR MORE SEGMENTS INDICATING THE TYPE OF DATA TO BE ACCESSED

ID – A SPECIFIC RECORD BEING REQUESTED

Example: Contacts URI

ContactsContract.Contacts.CONTENT_URI =
  “content://com.android.contacts/contacts/”

ContentResolver.query()

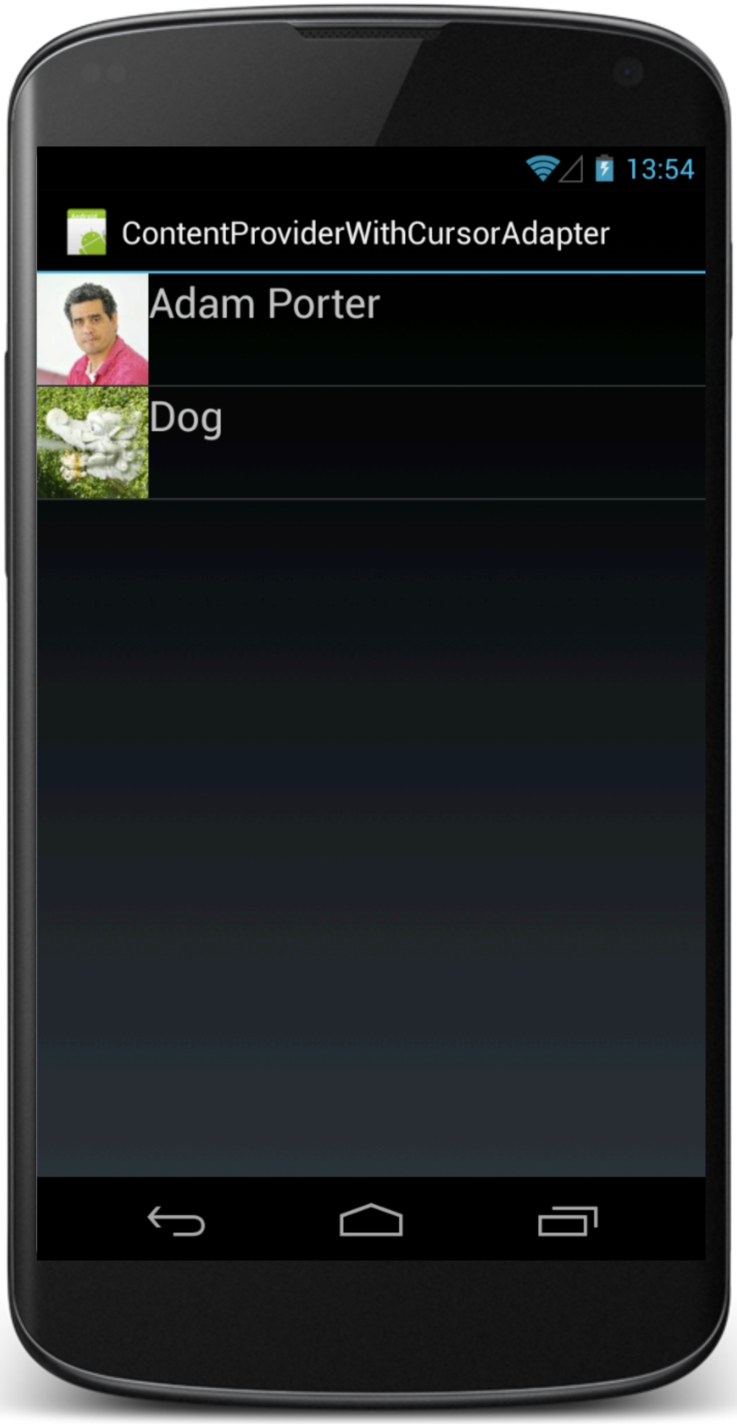
```
Cursor query (
    Uri uri,                    // ContentProvider Uri
    String[] projection         // Columns to retrieve
    String selection            // SQL selection pattern
    String[] selectionArgs      // SQL pattern args
    String sortOrder            // Sort order
    )
```

Returns a Cursor for iterating over the set of results

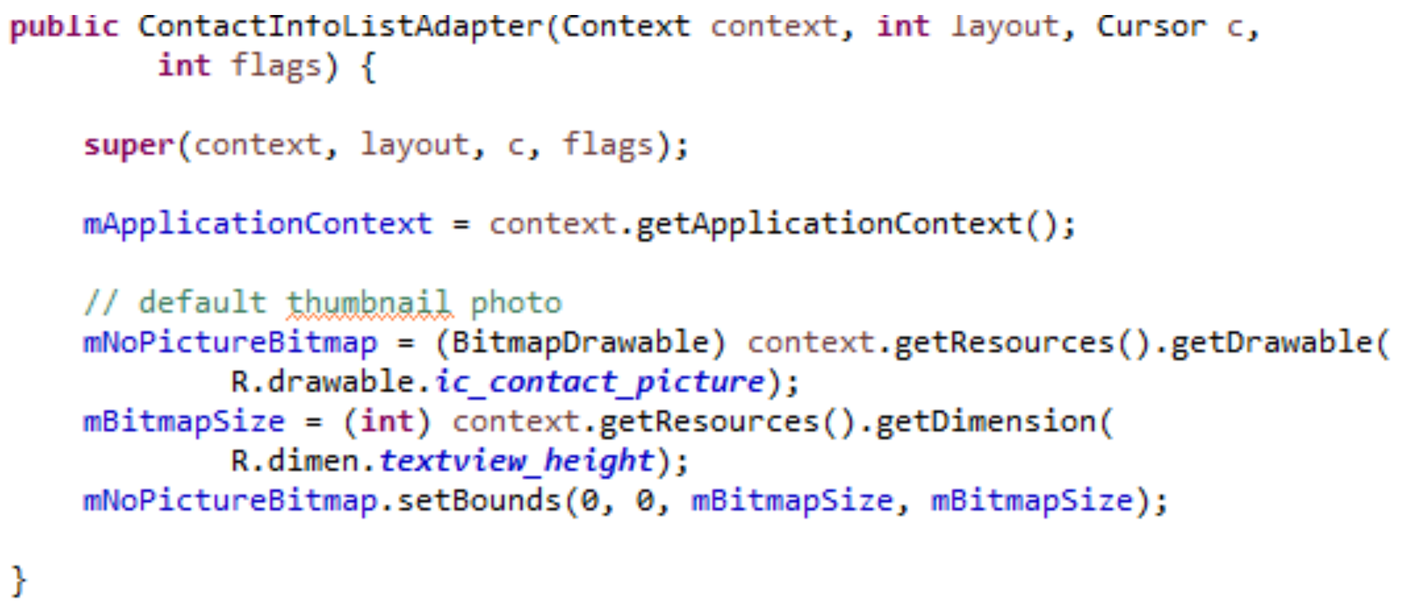
ContentProviderWithCursorAdapter

Extracts Contact information from the android Contacts ContentProvider

Displays each contact's name and photo, if available


13:54
ContentProviderWithCursorAdapter
Adam Porter
Dog

ContentProviderWithCursorAdapter



```java
public ContactInfoListAdapter(Context context, int layout, Cursor c,
        int flags) {

    super(context, layout, c, flags);

    mApplicationContext = context.getApplicationContext();

    // default thumbnail photo
    mNoPictureBitmap = (BitmapDrawable) context.getResources().getDrawable(
            R.drawable.ic_contact_picture);
    mBitmapSize = (int) context.getResources().getDimension(
            R.dimen.textview_height);
    mNoPictureBitmap.setBounds(0, 0, mBitmapSize, mBitmapSize);

}
```

ContentProviderWithCursorAdapter

```
// Create and return a new contact data view
@Override
public View newView(Context context, Cursor cursor, ViewGroup parent) {

    LayoutInflater inflater = (LayoutInflater) context
            .getSystemService(Context.LAYOUT_INFLATER_SERVICE);

    return inflater.inflate(R.layout.list_item, parent, false);

}
```

ContentProviderWithCursorAdapter

```java
// Update and return a contact data view
@Override
public void bindView(View view, Context context, Cursor cursor) {

    TextView textView = (TextView) view.findViewById(R.id.name);
    textView.setText(cursor.getString(cursor
            .getColumnIndex(Contacts.DISPLAY_NAME)));

    // Default photo
    BitmapDrawable photoBitmap = mNoPictureBitmap;

    // Get actual thumbnail photo if it exists
    String photoContentUri = cursor.getString(cursor
            .getColumnIndex(Contacts.PHOTO_THUMBNAIL_URI));
```

ContentProviderWithCursorAdapter

```java
    if (null != photoContentUri) {

        InputStream input = null;

        try {

            // Read thumbnail data from input stream
            input = context.getContentResolver().openInputStream(
                    Uri.parse(photoContentUri));

            if (input != null) {

                photoBitmap = new BitmapDrawable(
                        mApplicationContext.getResources(), input);
                photoBitmap.setBounds(0, 0, mBitmapSize, mBitmapSize);

            }
        } catch (FileNotFoundException e) {

            Log.i(TAG, "FileNotFoundException");

        }
    }

    // Set thumbnail image
    textView.setCompoundDrawables(photoBitmap, null, null, null);

}
```

CursorLoader

Conducting intensive operations on the main thread can affect application responsiveness

CursorLoader uses an AsyncTask to perform queries on a background thread

Using a CursorLoader

Implement LoaderManager's LoaderCallbacks interface

Create and initialize a cursor loader

initLoader()

INITIALIZE AND ACTIVATE A LOADER

```
Loader<D> initLoader(
        int id,
        Bundle args,
        LoaderCallbacks<D> callback)
```

LoaderCallbacks

Called to instantiate and return a new Loader for the specified ID

```
Loader<D> onCreateLoader (
                    int id,
                    Bundle args)
```

LoaderCallbacks

Called when a previously created loader has finished loading

```
void onLoadFinished(
            Loader<D> loader,
            D data)
```

LoaderCallbacks

Called when a previously created loader is reset

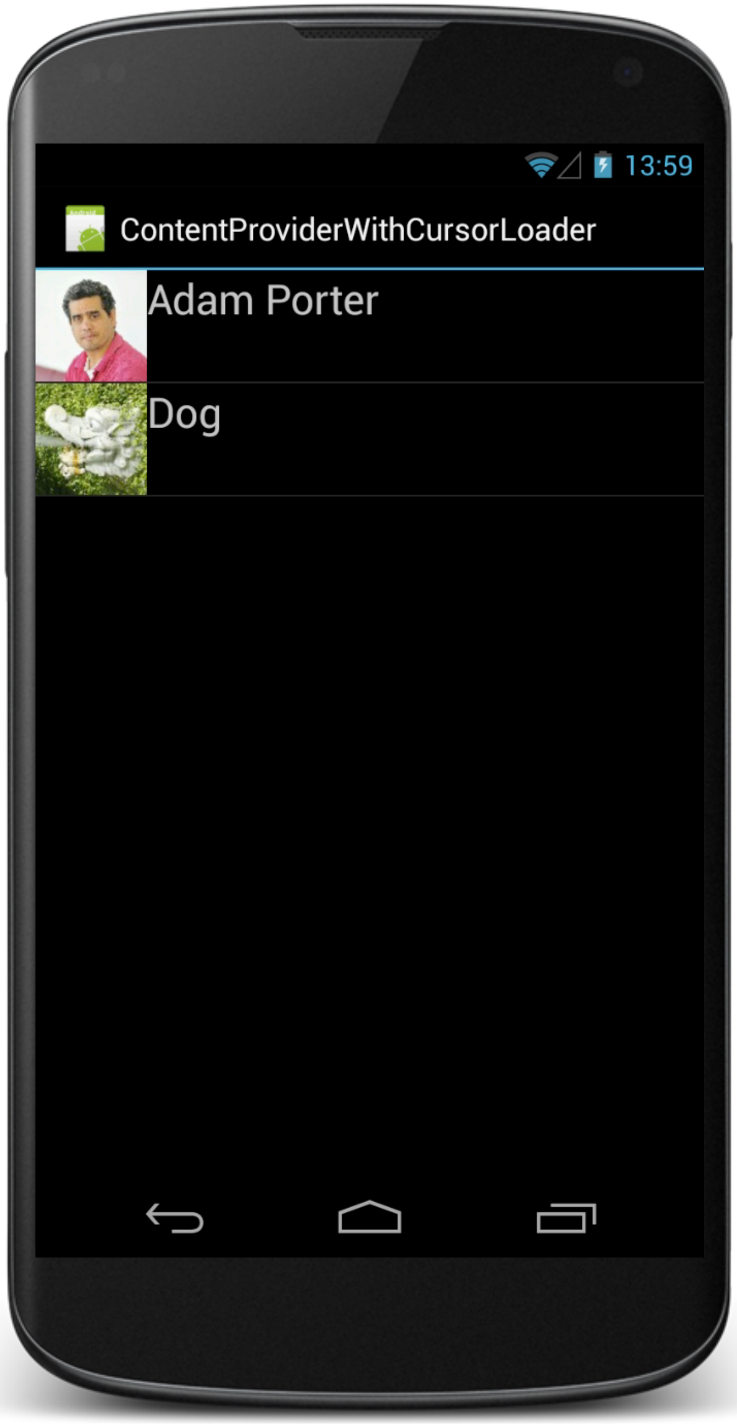
```
void onLoaderReset (
                    Loader<D> loader)
```

ContentProviderWithCursorLoader

Extracts Contact information from the android Contacts ContentProvider

Displays each contact's name and photo, if available

But it uses a CursorLoader when querying the ContentProvider



Adam Porter

Dog

ContentProviderWithCursorLoader

```java
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    // Create and set empty adapter
    mAdapter = new ContactInfoListAdapter(this, R.layout.list_item, null, 0);
    setListAdapter(mAdapter);

    // Initialize the loader
    getLoaderManager().initLoader(0, null, this);

}
```

ContentProviderWithCursorLoader

```
// Called when a new Loader should be created
// Returns a new CursorLoader

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

    // String used to filter contacts with empty or missing names or are unstarred
    String select = "((" + Contacts.DISPLAY_NAME + " NOTNULL) AND ("
            + Contacts.DISPLAY_NAME + " != " ) AND (" + Contacts.STARRED
            + "== 1))";

    // String used for defining the sort order
    String sortOrder = Contacts._ID + " ASC";

    return new CursorLoader(this, Contacts.CONTENT_URI, CONTACTS_ROWS,
            select, null, sortOrder);
}
```

ContentProviderWithCursorLoader

```
// Called when the Loader has finished loading its data
@Override
public void onLoadFinished(Loader<Cursor> loader, Cursor data) {

    // Swap the new cursor into the List adapter
    mAdapter.swapCursor(data);

}

// Called when the last Cursor provided to onLoadFinished()
// is about to be closed

@Override
public void onLoaderReset(Loader<Cursor> loader) {

    // set List adapter's cursor to null
    mAdapter.swapCursor(null);
}
```

ContentResolver.delete()

```
int delete (
 Uri url,                    // content Uri
 String where,               // SQL sel. pattern
 String[] selectArgs         // SQL pattern args
 )
```

RETURNS THE NUMBER OF ROWS DELETED

ContentResolver.insert()

```
Uri insert (
   Uri url,                          // content Uri
   ContentValues values              // values
 )
```

Returns the Uri of the inserted row

ContentResolver.update()

```
int update(
  Uri url,                      // content Uri
  ContentValues values          // new field values
  String where,                 // SQL sel. pattern
  String[] selectionArgs        // SQL pattern args
)
```

Returns the number of rows updated

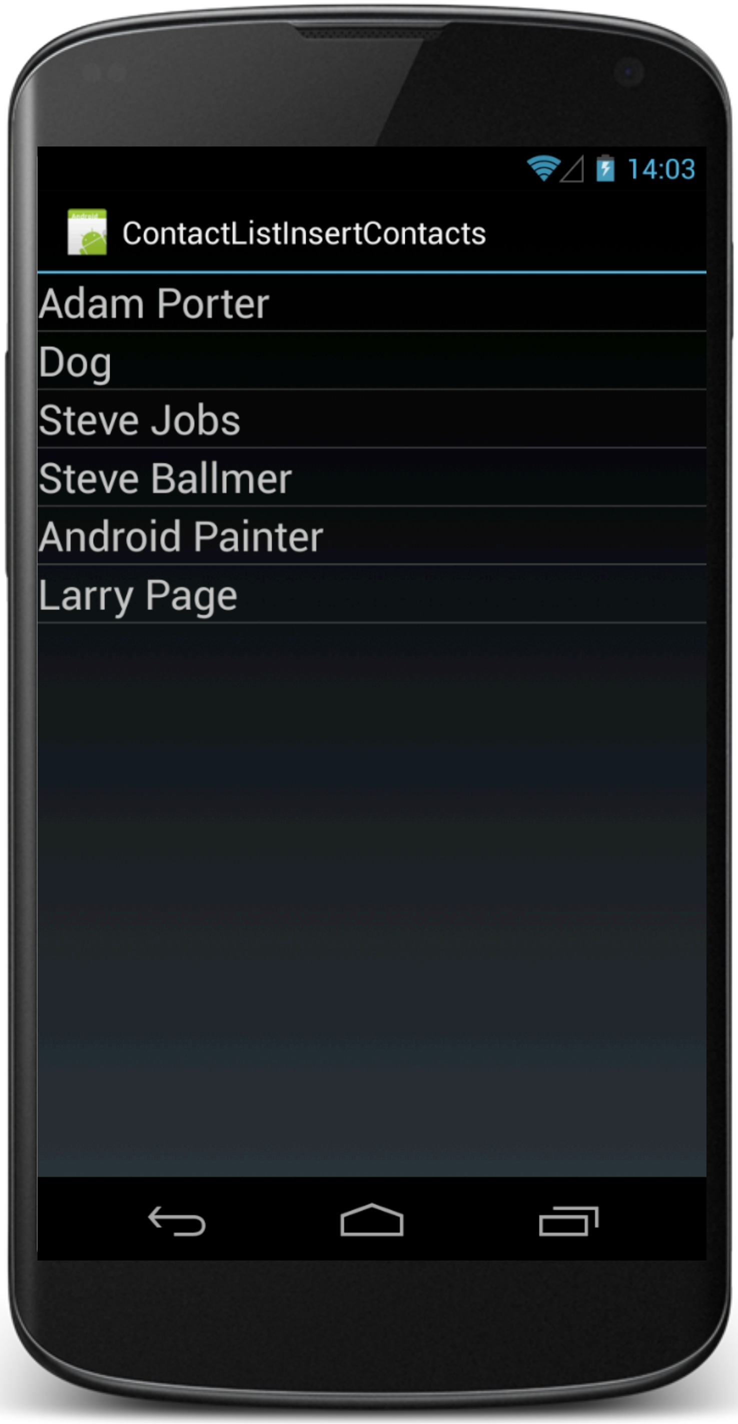
ContentProviderInsertContacts

Application reads contact information from the Android Contacts ContentProvider

Inserts several new contacts into Contacts ContentProvider

Displays old and new contacts

Deletes these new contacts on exit


14:03
ContactListInsertContacts
Adam Porter
Dog
Steve Jobs
Steve Ballmer
Android Painter
Larry Page

ContentProviderInsertContacts

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    // Get Account information
    // Must have a Google account set up on your device
    mAccountList = AccountManager.get(this).getAccountsByType("com.google");
    mType = mAccountList[0].type;
    mName = mAccountList[0].name;

    // Insert new contacts
    insertAllNewContacts();

    // Create and set empty list adapter
    mAdapter = new SimpleCursorAdapter(this, R.layout.list_layout, null,
            columnsToDisplay, resourceIds, 0);
    setListAdapter(mAdapter);

    // Initialize a CursorLoader
    getLoaderManager().initLoader(0, null, this);

}
```

ContentProviderInsertContacts

```
// Insert all new contacts into Contacts ContentProvider
private void insertAllNewContacts() {

    // Set up a batch operation on Contacts ContentProvider
    ArrayList<ContentProviderOperation> batchOperation = new ArrayList<ContentProviderOperation>();

    for (String name : mNames) {
        addRecordToBatchInsertOperation(name, batchOperation);
    }

    try {

        // Apply all batched operations
        getContentResolver().applyBatch(ContactsContract.AUTHORITY,
                batchOperation);

    } catch (RemoteException e) {
        Log.i(TAG, "RemoteException");
    } catch (OperationApplicationException e) {
        Log.i(TAG, "RemoteException");
    }

}
```

ContentProviderInsertContacts

```
// Insert named contact into Contacts ContentProvider
private void addRecordToBatchInsertOperation(String name,
        List<ContentProviderOperation> ops) {

    int position = ops.size();

    // First part of operation
    ops.add(ContentProviderOperation.newInsert(RawContacts.CONTENT_URI)
            .withValue(RawContacts.ACCOUNT_TYPE, mType)
            .withValue(RawContacts.ACCOUNT_NAME, mName)
            .withValue(Contacts.STARRED, 1).build());

    // Second part of operation
    ops.add(ContentProviderOperation.newInsert(Data.CONTENT_URI)
            .withValueBackReference(Data.RAW_CONTACT_ID, position)
            .withValue(Data.MIMETYPE, StructuredName.CONTENT_ITEM_TYPE)
            .withValue(StructuredName.DISPLAY_NAME, name).build());

}
```

Creating a ContentProvider

Implement a storage system for the data

Define a Contract Class to support users of your ContentProvider

Implement a ContentProvider subclass

Declare and configure content provider in AndroidManifest.xml

ContentProviderCustom

Application defines a ContentProvider for ID/string pairs

ContentProviderCustom

```
// Delete some or all data items
@Override
public synchronized int delete(Uri uri, String selection,
        String[] selectionArgs) {

    int numRecordsRemoved = 0;

    // If last segment is the table name, delete all data items
    if (isTableUri(uri)) {

        numRecordsRemoved = db.size();
        db.clear();

    // If last segment is the digit, delete data item with that ID
    } else if (isItemUri(uri)) {

        Integer requestId = Integer.parseInt(uri.getLastPathSegment());

        if (null != db.get(requestId)) {

            db.remove(requestId);

            numRecordsRemoved++;
        }
    }

    //return number of items deleted
    return numRecordsRemoved;
}
```

ContentProviderCustom

```java
// Return MIME type for given uri
@Override
public synchronized String getType(Uri uri) {

    String contentType = DataContract.CONTENT_ITEM_TYPE;

    if (isTableUri(uri)) {

        contentType = DataContract.CONTENT_DIR_TYPE;

    }

    return contentType;
}

// Insert specified value into ContentProvider
@Override
public synchronized Uri insert(Uri uri, ContentValues value) {


    if (value.containsKey(DataContract.DATA)) {

        DataRecord dataRecord = new DataRecord(value.getAsString(DataContract.DATA));
        db.put(dataRecord.getID(), dataRecord);

        // return Uri associated with newly-added data item
        return Uri.withAppendedPath(DataContract.CONTENT_URI,
                String.valueOf(dataRecord.getID()));

    }
    return null;
}
```

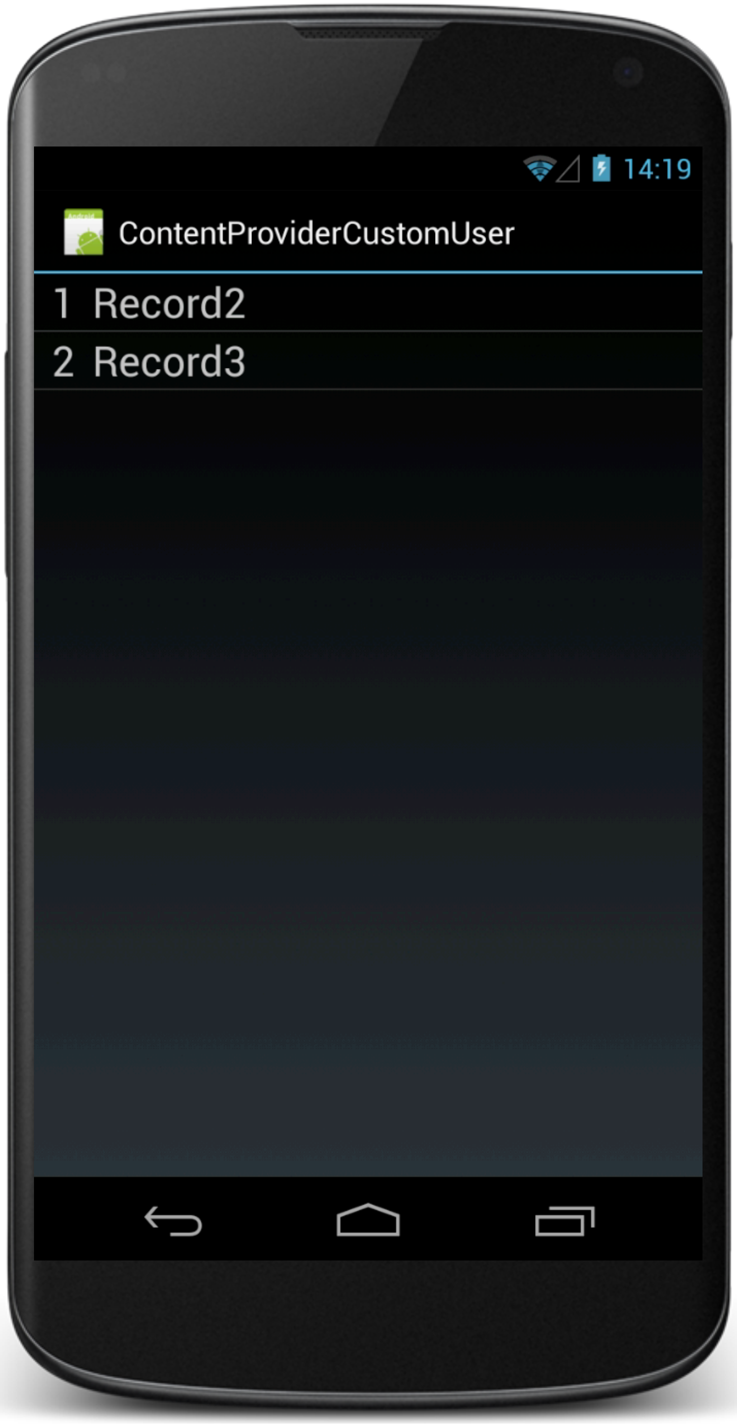
ContentProviderCustom

```java
// return all or some rows from ContentProvider based on specified Uri
// all other parameters are ignored

@Override
public synchronized Cursor query(Uri uri, String[] projection,
        String selection, String[] selectionArgs, String sortOrder) {

    // Create simple cursor
    MatrixCursor cursor = new MatrixCursor(DataContract.ALL_COLUMNS);

    if (isTableUri(uri)) {

        // Add all rows to cursor
        for (int idx = 0; idx < db.size(); idx++) {

            DataRecord dataRecord = db.get(db.keyAt(idx));
            cursor.addRow(new Object[] { dataRecord.getID(),
                    dataRecord.getData() });

        }
    } else if (isItemUri(uri)){

        // Add single row to cursor
        Integer requestId = Integer.parseInt(uri.getLastPathSegment());

        if (null != db.get(requestId)) {

            DataRecord dr = db.get(requestId);
            cursor.addRow(new Object[] { dr.getID(), dr.getData() });

        }
    }
    return cursor;
}
```

ContentProviderCustomUser

Reads ID/String pairs from the ContentProvider we just examined

Displays the data in a ListView



```
1 Record2
2 Record3
```

ContentProviderCustomUser

```java
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    ContentResolver contentResolver = getContentResolver();

    ContentValues values = new ContentValues();

    // Insert first record
    values.put(DataContract.DATA, "Record1");
    Uri firstRecordUri = contentResolver.insert(DataContract.CONTENT_URI, values);

    values.clear();

    // Insert second record
    values.put(DataContract.DATA, "Record2");
    contentResolver.insert(DataContract.CONTENT_URI, values);

    values.clear();

    // Insert third record
    values.put(DataContract.DATA, "Record3");
    contentResolver.insert(DataContract.CONTENT_URI, values);

    // Delete first record
    contentResolver.delete(firstRecordUri, null, null);

    // Create and set cursor and list adapter
    Cursor c = contentResolver.query(DataContract.CONTENT_URI, null, null, null,
            null);

    setListAdapter(new SimpleCursorAdapter(this, R.layout.list_layout, c,
            DataContract.ALL_COLUMNS, new int[] { R.id.idString,
                    R.id.data }, 0));

}
```

Next Time

Service