CS378 - Mobile Computing

Content Providers

Content Providers

- One of the four primary application components:
  - activities
  - content providers
  - services
  - broadcast receivers

Android Applications

- Recall…
- Each application runs as a different user on the OS
- private files, user id, separate process with its own instance of the Dalvik VM
- Content Providers act as a bridge between applications to share data

Content Providers

- Many of the built in applications have content providers to allow other apps to access data
- Examples of built in content providers
  - MediaStore
  - CallLog
  - Browser
  - ContactsContract (contacts)
  - Settings
  - UserDictionary
- http://developer.android.com/reference/android/provider/package-summary.html

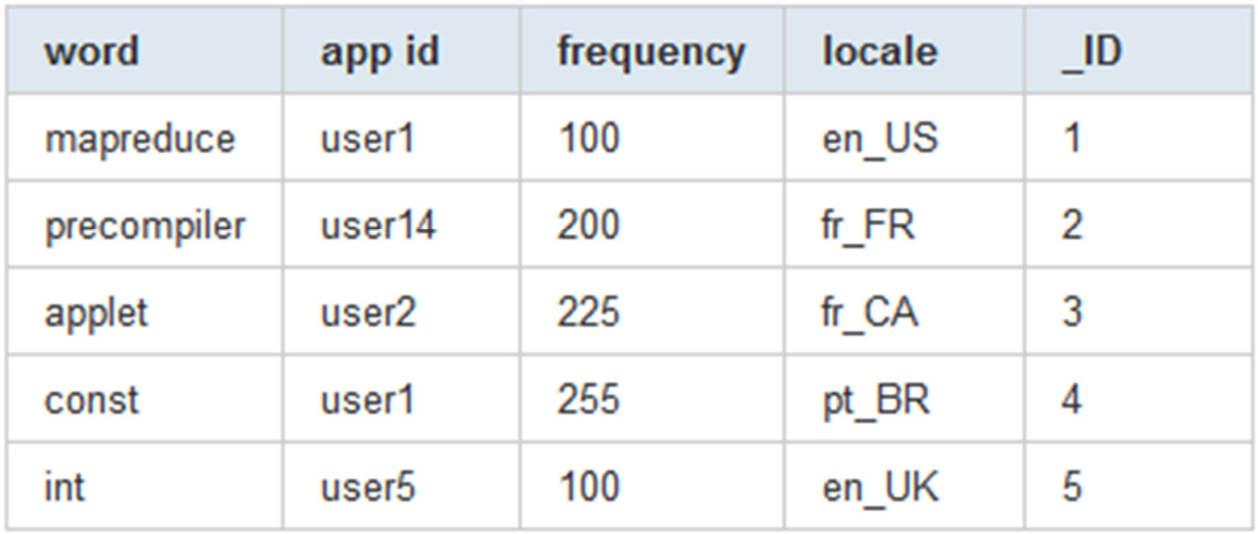
Content Providers

- Provide access to a central data repository
  - ability to read and write to centralized data
- data presented by Content Provider in the form of a table
  - like table from relational database
- Each row in data table one "piece" of data in repository

Example Table

- Data from user dictionary

Table 1: Sample user dictionary table.



word	app id	frequency	locale	_ID
mapreduce	user1	100	en_US	1
precompiler	user14	200	fr_FR	2
applet	user2	225	fr_CA	3
const	user1	255	pt_BR	4
int	user5	100	en_UK	5

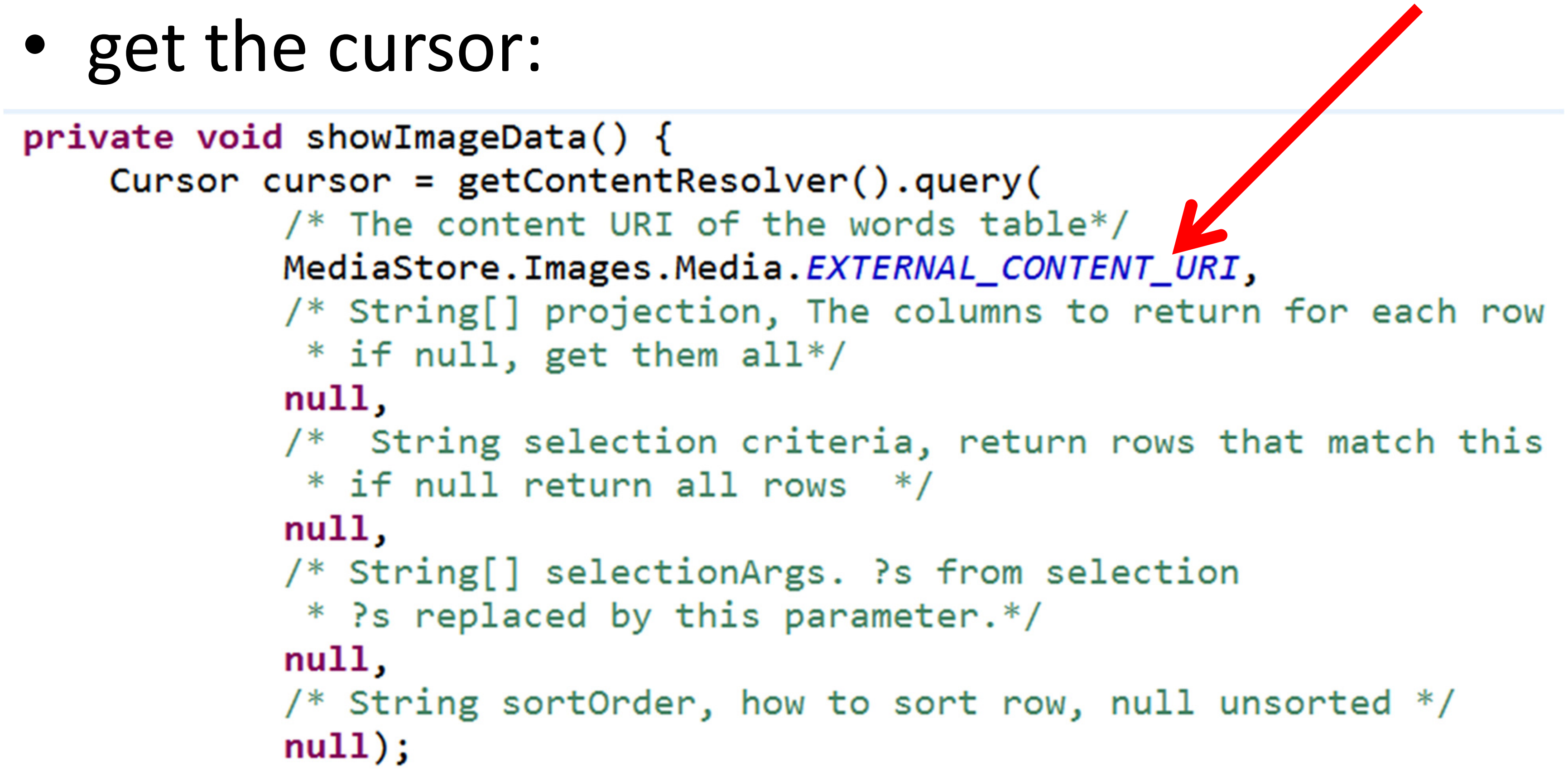
- primary key optional
- _ID column required to bind data from provider to a ListView

Accessing Content Provider

- Use a ContentResolver client object in your app
- ContentResolver communicates with ContentProvider
- matching methods in each class
- example: query() method
- Create a cursor via content resolver to move through rows of table

Accessing Content via Provider

- Example: Exploring Images on a device
- MediaStore.Images.Media class is one of the ContentProviders
- get the cursor:



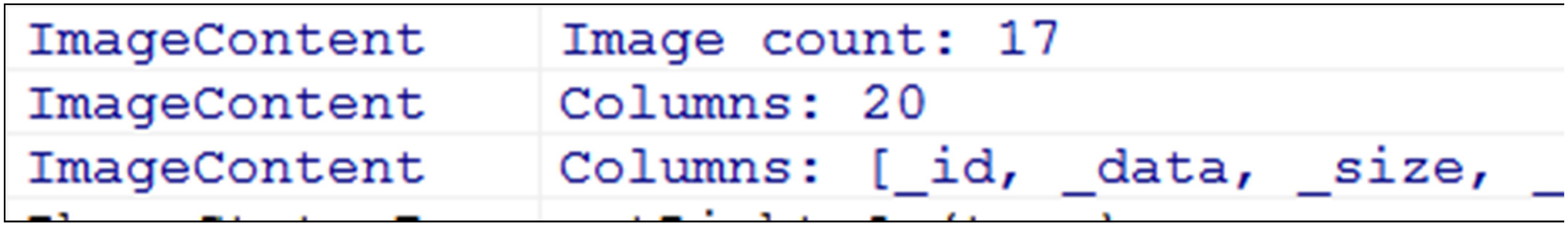
```java
private void showImageData() {
    Cursor cursor = getContentResolver().query(
            /* The content URI of the words table*/
            MediaStore.Images.Media.EXTERNAL_CONTENT_URI,
            /* String[] projection, The columns to return for each row
             * if null, get them all*/
            null,
            /*  String selection criteria, return rows that match this
             * if null return all rows   */
            null,
            /* String[] selectionArgs. ?s from selection
             * ?s replaced by this parameter.*/
            null,
            /* String sortOrder, how to sort row, null unsorted */
            null);
```

Accessing Content via Provider

- After obtaining cursor:

```
Log.d(TAG, "Image count: " + cursor.getCount());
Log.d(TAG, "Columns: "  + cursor.getColumnCount());
String[] columns = cursor.getColumnNames();

Log.d(TAG, "Columns: " + Arrays.toString(columns));
```

- result:



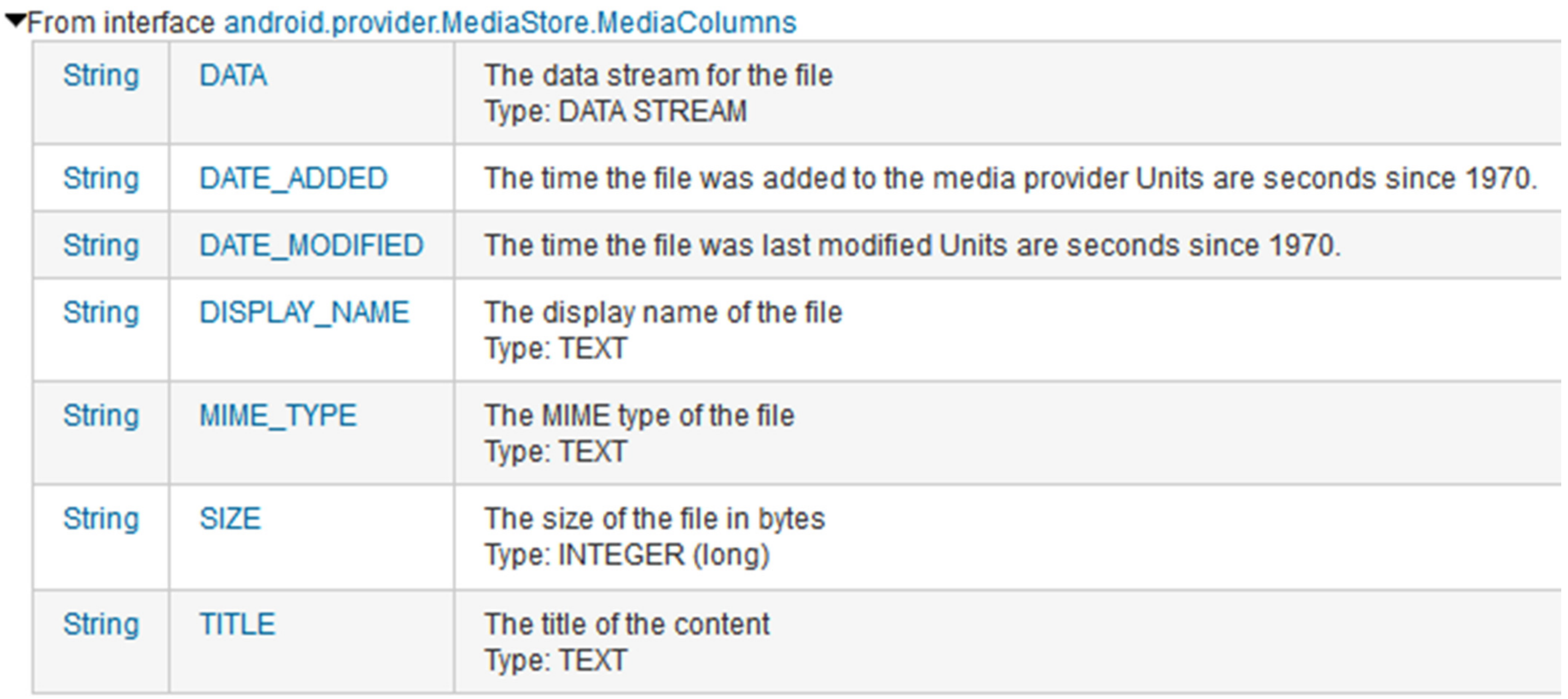
```
ImageContent    Image count: 17
ImageContent    Columns: 20
ImageContent    Columns: [_id, _data, _size, _
```

MediaStore.Images.Media

- Columns from table:
- According to Logcat:
- [_id, _data, _size, _display_name, mime_type, title, date_added, date_modified, description, picasa_id, isprivate, latitude, longitude, datetaken, orientation, mini_thumb_magic, bucket_id, bucket_display_name, width, height]

MediaStore.Images.Media

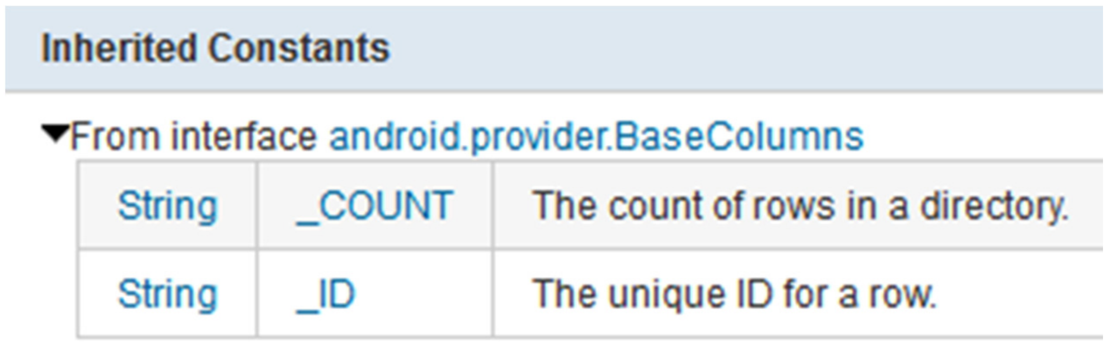
- Columns documented in ContentProvider classes and interfaces



From interface android.provider.MediaStore.MediaColumns

String	DATA	The data stream for the file Type: DATA STREAM
String	DATE_ADDED	The time the file was added to the media provider Units are seconds since 1970.
String	DATE_MODIFIED	The time the file was last modified Units are seconds since 1970.
String	DISPLAY_NAME	The display name of the file Type: TEXT
String	MIME_TYPE	The MIME type of the file Type: TEXT
String	SIZE	The size of the file in bytes Type: INTEGER (long)
String	TITLE	The title of the content Type: TEXT

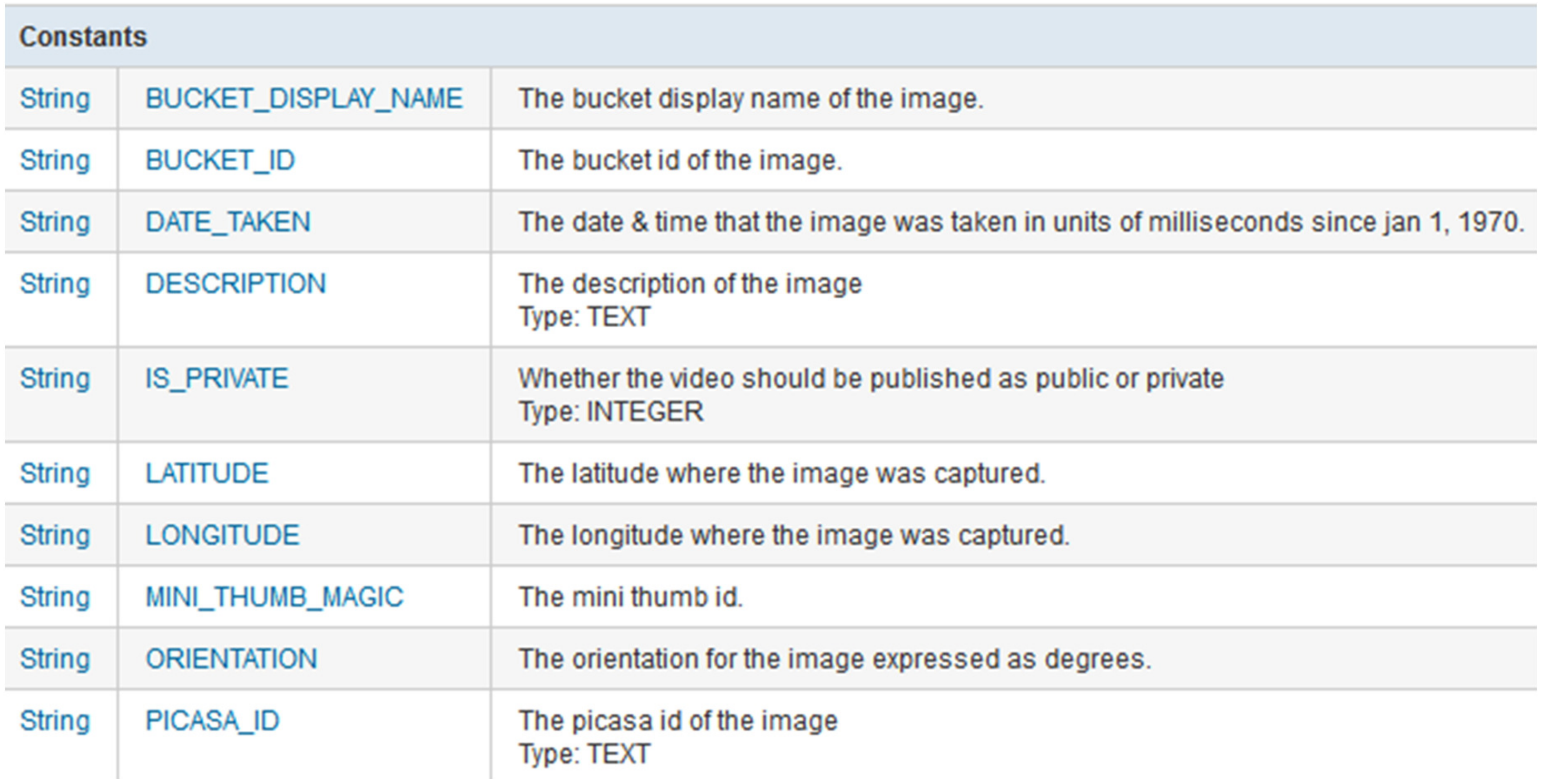
MediaStore.Images.Media Columns



Inherited Constants

From interface android.provider.BaseColumns

String	_COUNT	The count of rows in a directory.
String	_ID	The unique ID for a row.



Constants

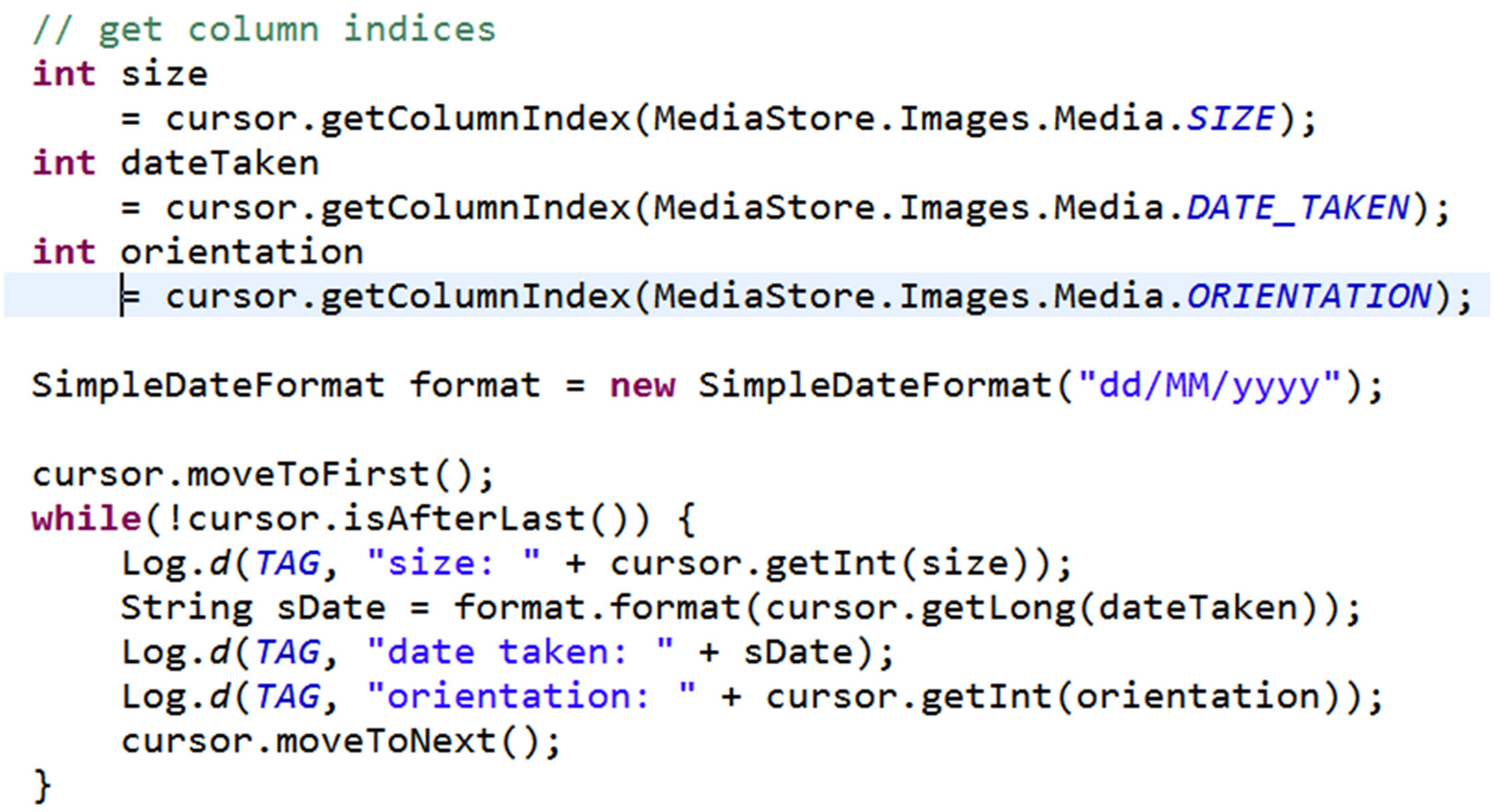
String	BUCKET_DISPLAY_NAME	The bucket display name of the image.
String	BUCKET_ID	The bucket id of the image.
String	DATE_TAKEN	The date & time that the image was taken in units of milliseconds since jan 1, 1970.
String	DESCRIPTION	The description of the image Type: TEXT
String	IS_PRIVATE	Whether the video should be published as public or private Type: INTEGER
String	LATITUDE	The latitude where the image was captured.
String	LONGITUDE	The longitude where the image was captured.
String	MINI_THUMB_MAGIC	The mini thumb id.
String	ORIENTATION	The orientation for the image expressed as degrees.
String	PICASA_ID	The picasa id of the image Type: TEXT

Selection Columns

- Limit Columns returned with projection argument to query method that creates Cursor

```
String[] projection = {MediaStore.Images.Media.DATE_TAKEN,
        MediaStore.Images.Media.SIZE,
        MediaStore.Images.Media.ORIENTATION};

cursor = getContentResolver().query(
        MediaStore.Images.Media.EXTERNAL_CONTENT_URI,
        projection,
        null,
        null,
        MediaStore.Images.Media.SIZE);
```

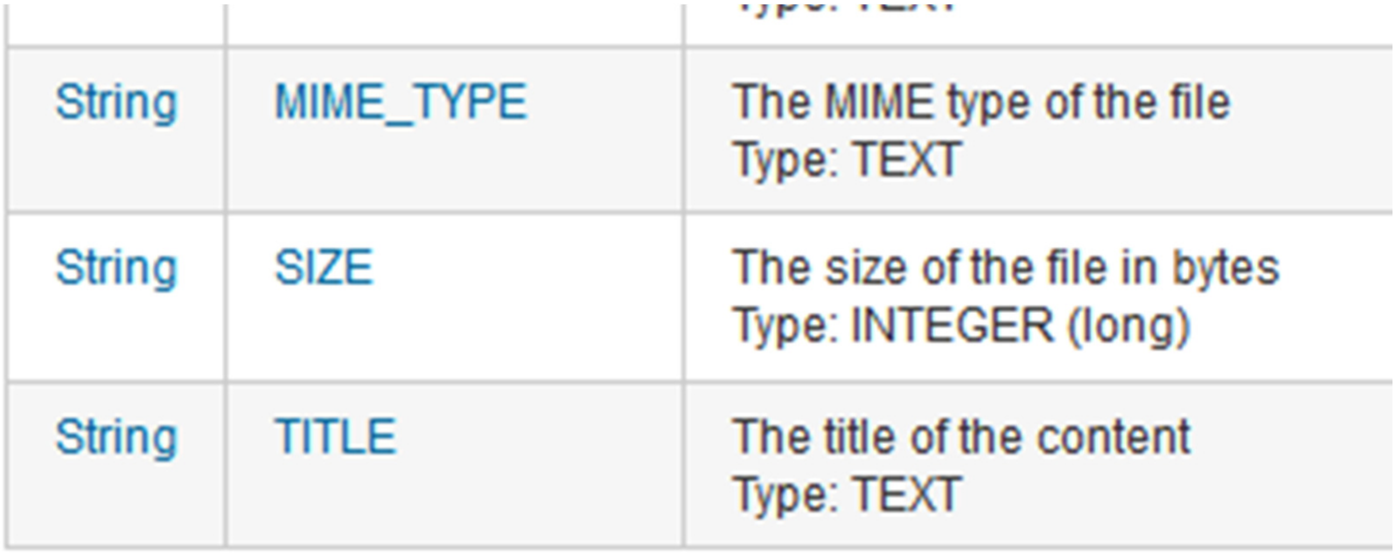
Showing Data in Logcat



```java
// get column indices
int size
    = cursor.getColumnIndex(MediaStore.Images.Media.SIZE);
int dateTaken
    = cursor.getColumnIndex(MediaStore.Images.Media.DATE_TAKEN);
int orientation
    = cursor.getColumnIndex(MediaStore.Images.Media.ORIENTATION);

SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");

cursor.moveToFirst();
while(!cursor.isAfterLast()) {
    Log.d(TAG, "size: " + cursor.getInt(size));
    String sDate = format.format(cursor.getLong(dateTaken));
    Log.d(TAG, "date taken: " + sDate);
    Log.d(TAG, "orientation: " + cursor.getInt(orientation));
    cursor.moveToNext();
}
```

Getting Data from Row

- Must determine how what form column data is in, use getX method
- refer to constants from ContentProvider class
- careful - some INTEGERS longs

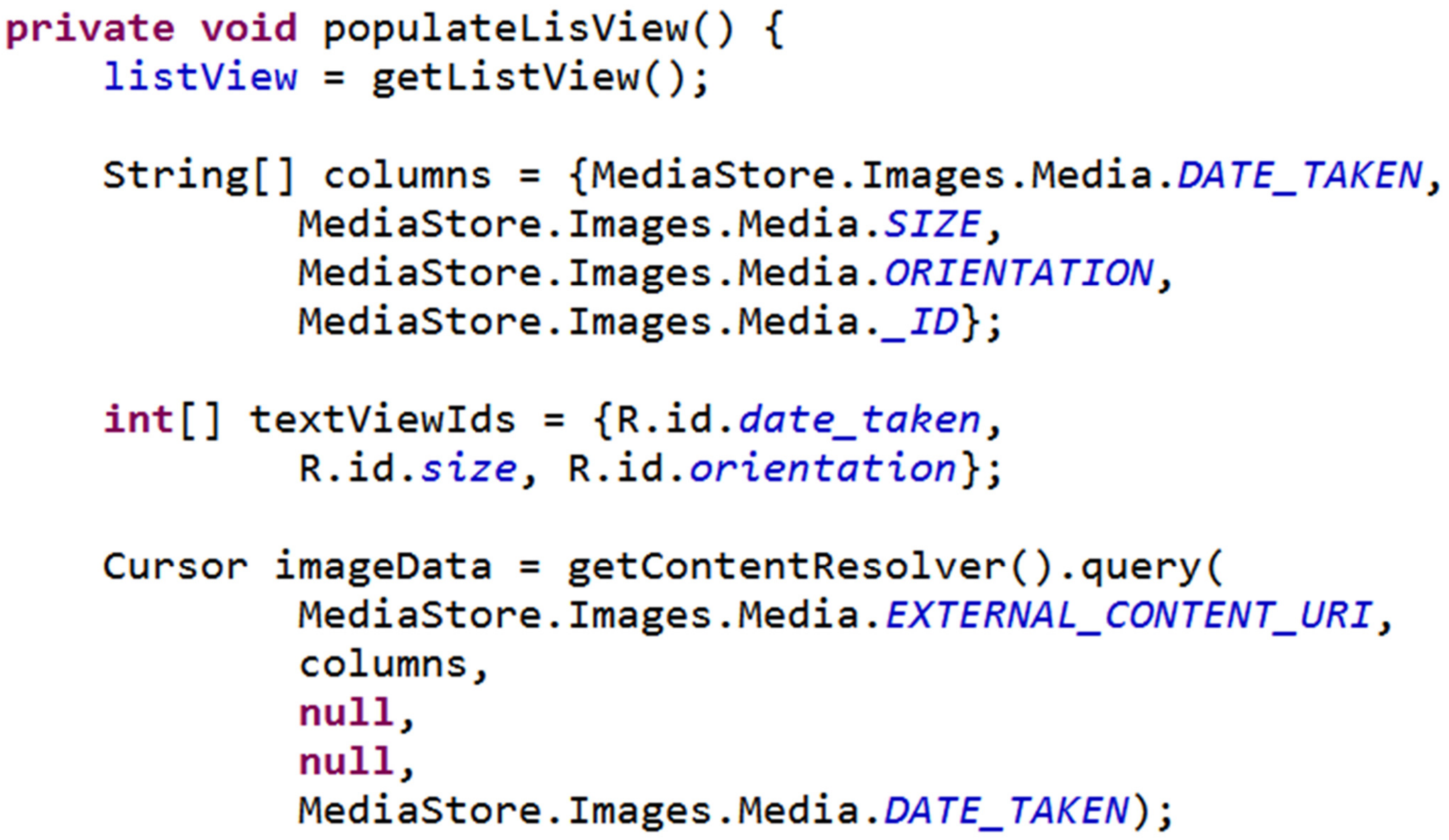


String	MIME_TYPE	The MIME type of the file Type: TEXT
String	SIZE	The size of the file in bytes Type: INTEGER (long)
String	TITLE	The title of the content Type: TEXT

Displaying Data in ListView

- Like MovieRating Example
- Specify columns to get from ContentProvider
- Create view that will hold data
- Obtain cursor from ContentProvider
- Use ListAdapter to convert data from Cursor to Views
- Sub class adapter to format text

Display Data from ContentProvider



```java
private void populateLisView() {
    listView = getListView();

    String[] columns = {MediaStore.Images.Media.DATE_TAKEN,
            MediaStore.Images.Media.SIZE,
            MediaStore.Images.Media.ORIENTATION,
            MediaStore.Images.Media._ID};

    int[] textViewIds = {R.id.date_taken,
            R.id.size, R.id.orientation};

    Cursor imageData = getContentResolver().query(
            MediaStore.Images.Media.EXTERNAL_CONTENT_URI,
            columns,
            null,
            null,
            MediaStore.Images.Media.DATE_TAKEN);
```

Display Data from ContentProvider

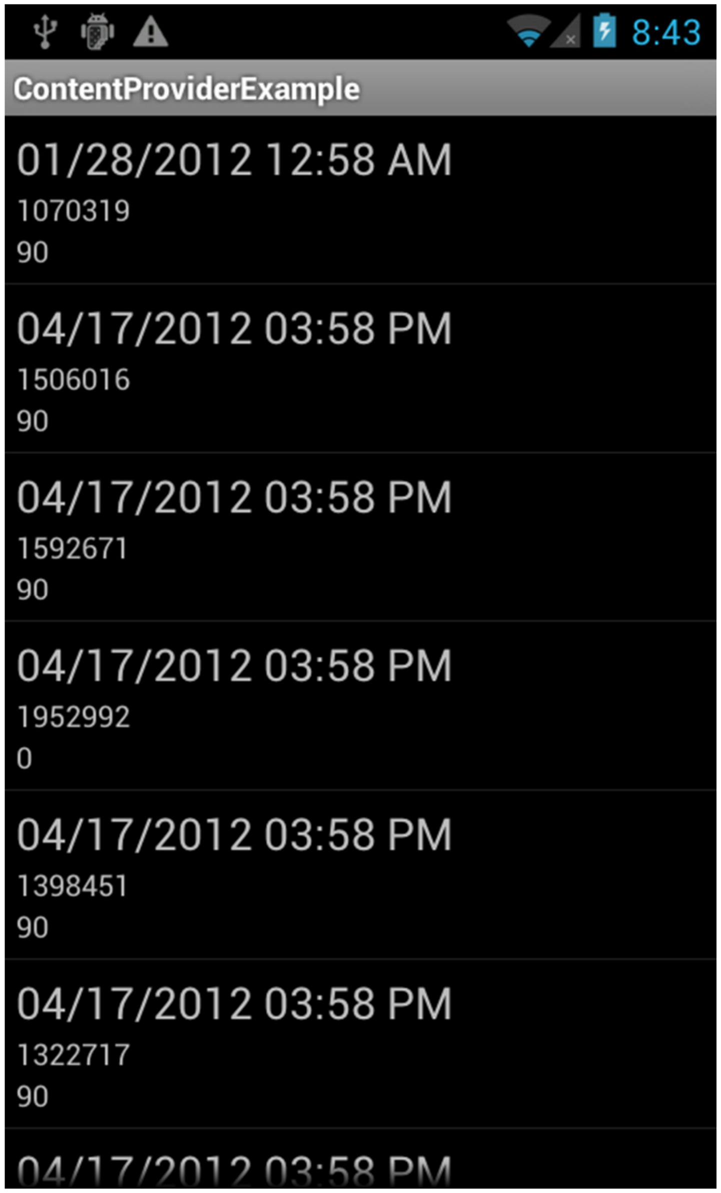
- rest of populateListView from ListActivity

```
    ListAdapter adapter = new MyAdapter(this,
            R.layout.list_item_view,
            imageData, columns, textViewIds);

    Log.d(TAG, "count: " + adapter.getCount());

    setListAdapter(adapter);
}
```

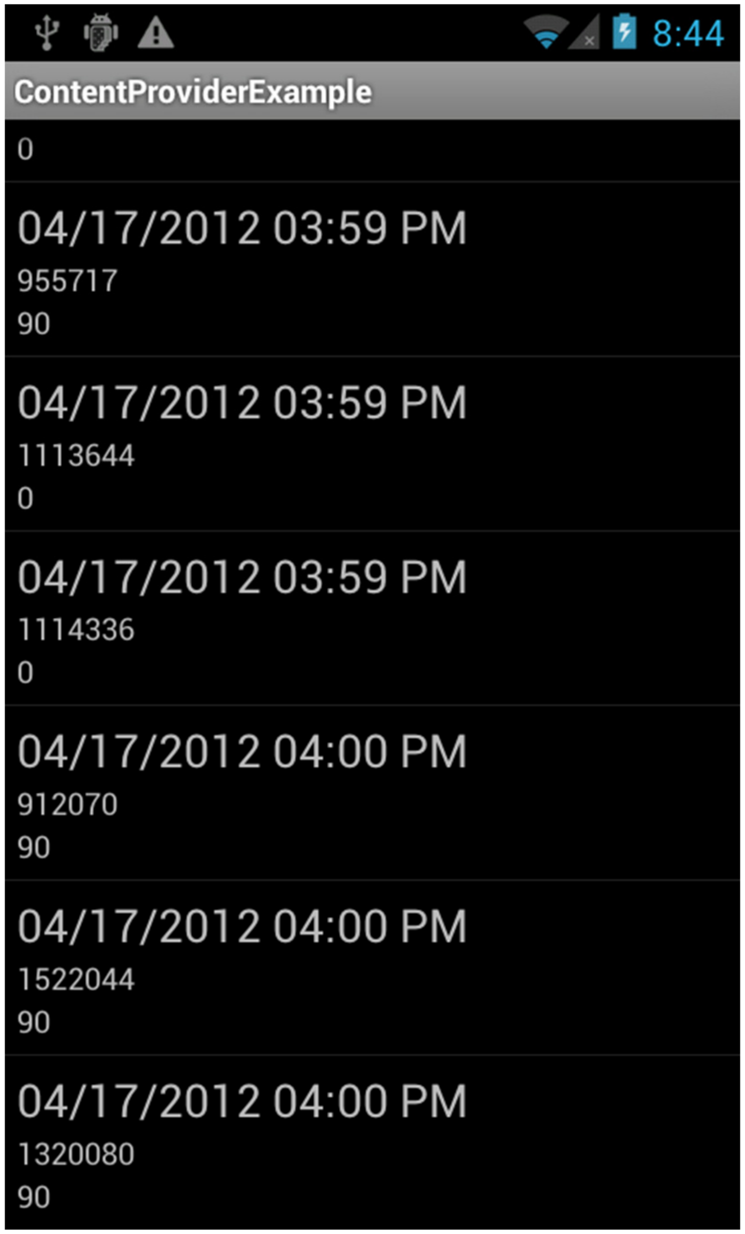
Subclass Adapter

```
private static class MyAdapter
    extends SimpleCursorAdapter {

    static String format = "MM/dd/yyyy hh:mm a";

    private MyAdapter(Context c, int layout,
            Cursor cur, String[] from, int[] to) {
        super(c,layout, cur, from, to);
    }

    public void setViewText(TextView v, String text) {
        if (v.getId() == R.id.date_taken) {
            text = getDate(Long.parseLong(text), format);
        }
        v.setText(text);
    }
```

Results



```
01/28/2012 12:58 AM
1070319
90
04/17/2012 03:58 PM
1506016
90
04/17/2012 03:58 PM
1592671
90
04/17/2012 03:58 PM
1952992
0
04/17/2012 03:58 PM
1398451
90
04/17/2012 03:58 PM
1322717
90
04/17/2012 03:58 PM
```



```
0
04/17/2012 03:59 PM
955717
90
04/17/2012 03:59 PM
1113644
0
04/17/2012 03:59 PM
1114336
0
04/17/2012 04:00 PM
912070
90
04/17/2012 04:00 PM
1522044
90
04/17/2012 04:00 PM
1320080
90
```

Content Provider Capabilities

- Possible to update, insert, and delete data via a ContentProvider
- insert, update, and delete methods part of ContentResolver class
- important to guard against malicious input from user
  - sql statements
  - http://developer.android.com/guide/topics/providers/content-provider-basics.html#Injection

Creating ContentProvider

- It is possible to implement a ContentProvider for your app
- You may need / want to provide a ContentProvider if:
  - You want to offer complex data or files to other applications.
  - You want to allow users to copy complex data from your app into other apps.
  - You want to provide custom search suggestions using the search framework.
- Not normally necessary

Bonus - Oracle v. Google

Summary and Timeline

- Oracle is suing Google for its use of Java in the Android operating system
- brief timeline from pc world
- August 2005 -- Google buys Android Inc. Soon after, it discusses the possibility of licensing Java from Sun.
- October 2005 -- Andy Rubin, the head of Google's Android division, writes in an email that Google can either adopt Microsoft's C# for Android or "do Java anyway and defend our decision, perhaps making enemies along the way." Over the next several months, Google and Sun continue to negotiate for a Java license but fail to reach a deal.
- February 2006 -- Sun supposedly offers Google a three-year Java license for US$20 million plus 10 percent of Google's Android-related revenue, capped at $25 million. Google rejects the offer.

TimeLine Continued

- November 2007 -- Google announces publicly that it is developing Android, which includes a Java-compatible virtual machine called Dalvik.
- October 2008 -- HTC releases the first Android phone, the HTC Dream.
- January 2010 -- Oracle acquires Sun and inherits its Java patents and copyrights.
- July 2010 -- Oracle meets with Google's lawyers to discuss Oracle's patent infringement allegations.
- Aug. 6, 2010 -- Rubin receives an email from a Google engineer stating that the alternatives to Java "all suck" and that "we need to negotiate a license for Java under the terms we need."
- Aug. 12, 2010 -- Oracle files a lawsuit against Google, accusing it of infringing seven Java patents and its Java copyrights. Google denies any wrongdoing and calls the lawsuit a "baseless attack" on Google and open-source developers.

Timeline Continued

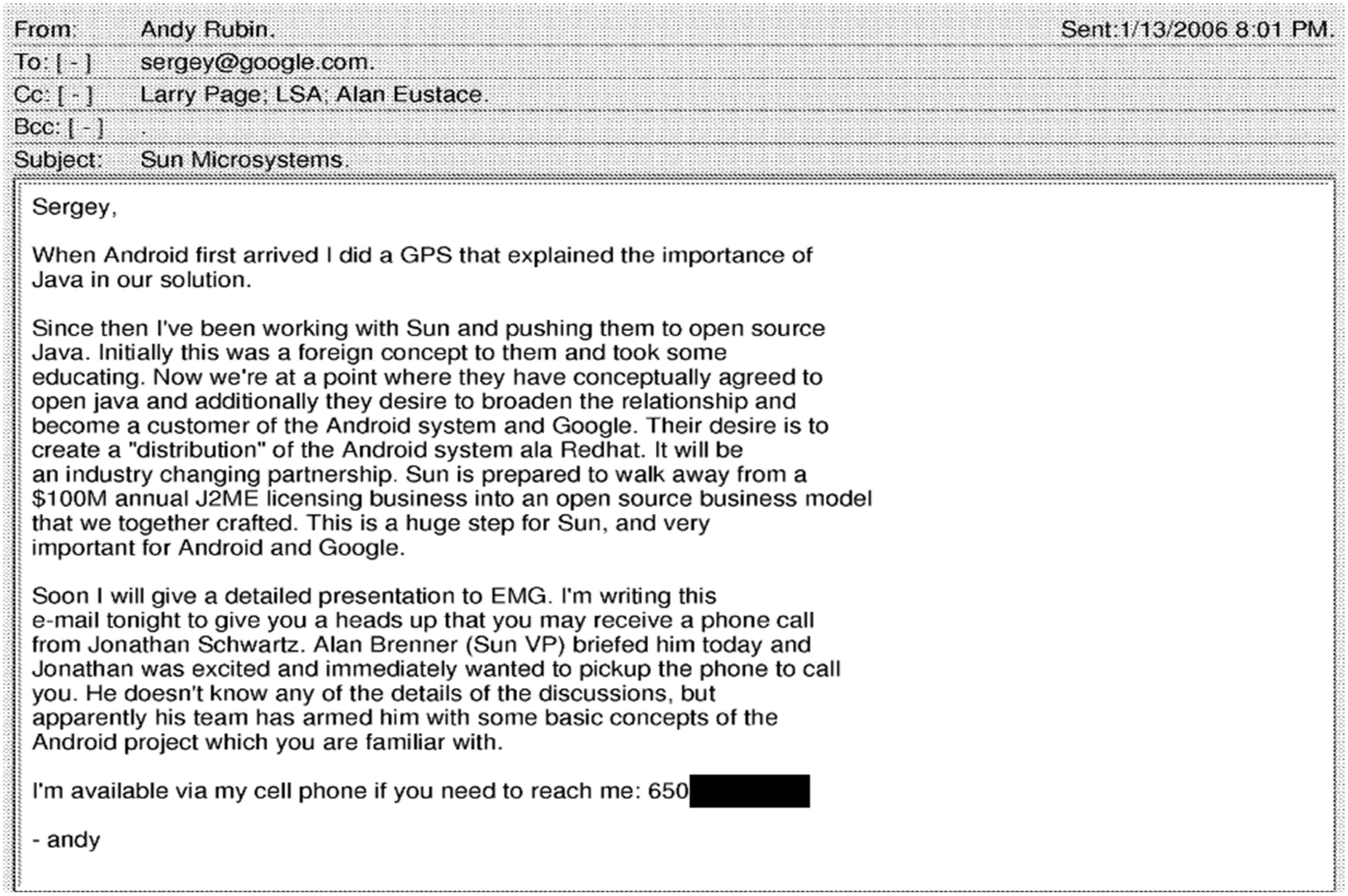
- January 2011 -- Android accounts for one-third of all smartphone sales, Canalys says.
- February 2011 -- Google asks the U.S. Patent and Trademark Office to reexamine Oracle's patents, arguing they shouldn't have been issued. By the time the trial starts, only two of the seven patents remain in the suit.
- June 2011 -- A court filing reveals that Oracle is seeking between $1.4 billion and $6.1 billion in damages.
- July 2011 -- A judge rules that Oracle "overreached" with its damages estimate and tells it to recalculate.

Timeline Continued

- September 2011 -- The CEOs of Oracle and Google, Larry Ellison and Larry Page, are ordered to hold settlement talks but can't reach agreement.
- November 2011 -- Android accounts for more than 50 percent of smartphone sales, Gartner says.
- March 2012 -- The two sides are ordered to hold more settlement talks but still can't reach a deal.
- April 2012 -- An eight-week jury trial is scheduled to begin Monday, April 16, at the U.S. District Court in San Francisco.

Some of Oracle's Evidence

Case3:10-cv-03561-WHA Document398-10 Filed09/06/11 Page2 of 2



From:	Andy Rubin.	Sent:1/13/2006 8:01 PM.
To: [-]	sergey@google.com.	
Cc: [-]	Larry Page; LSA; Alan Eustace.	
Bcc: [-]	.	
Subject:	Sun Microsystems.	

Sergey,

When Android first arrived I did a GPS that explained the importance of Java in our solution.

Since then I've been working with Sun and pushing them to open source Java. Initially this was a foreign concept to them and took some educating. Now we're at a point where they have conceptually agreed to open java and additionally they desire to broaden the relationship and become a customer of the Android system and Google. Their desire is to create a "distribution" of the Android system ala Redhat. It will be an industry changing partnership. Sun is prepared to walk away from a $100M annual J2ME licensing business into an open source business model that we together crafted. This is a huge step for Sun, and very important for Android and Google.

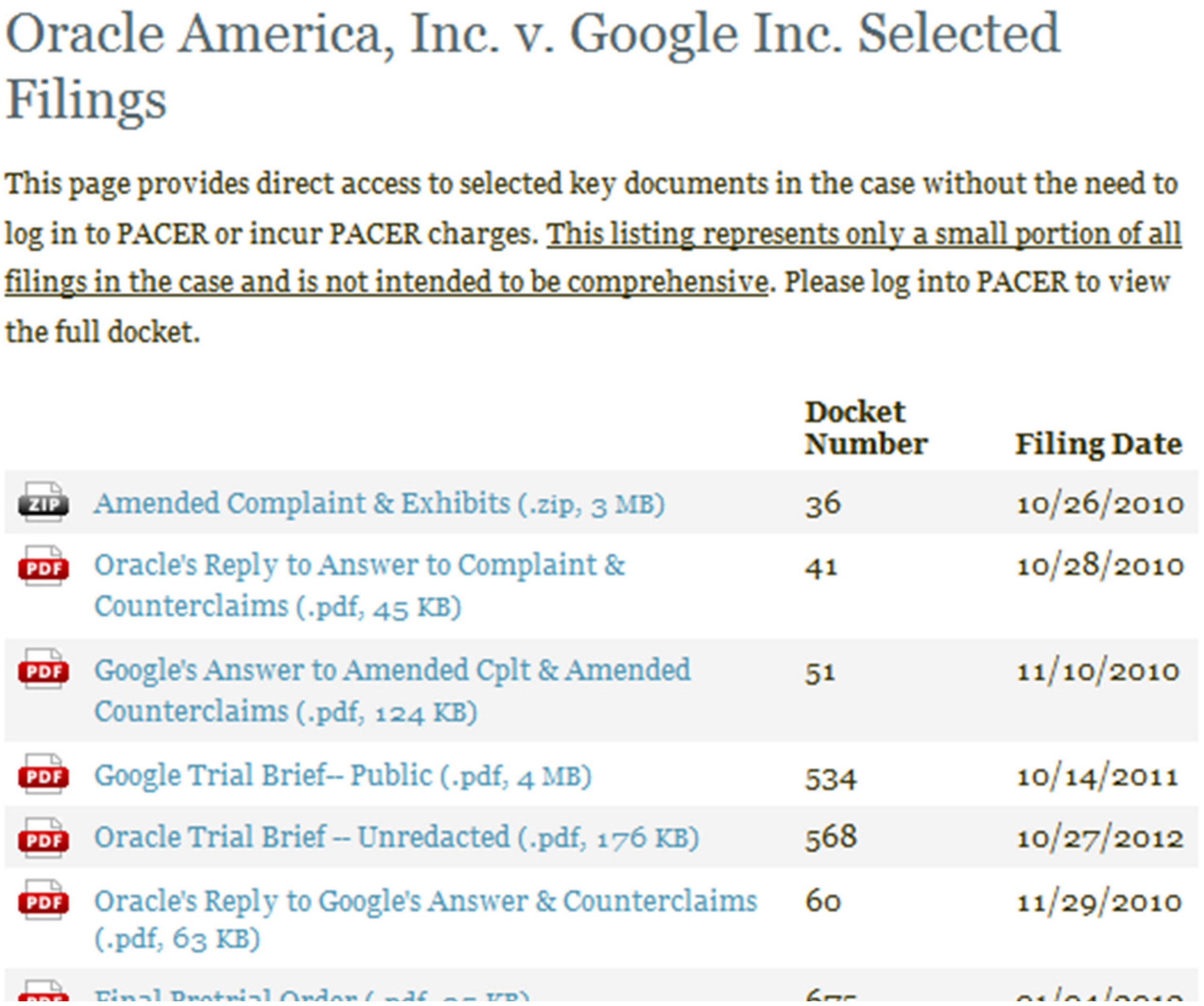
Soon I will give a detailed presentation to EMG. I'm writing this e-mail tonight to give you a heads up that you may receive a phone call from Jonathan Schwartz. Alan Brenner (Sun VP) briefed him today and Jonathan was excited and immediately wanted to pickup the phone to call you. He doesn't know any of the details of the discussions, but apparently his team has armed him with some basic concepts of the Android project which you are familiar with.

I'm available via my cell phone if you need to reach me: 650[redacted]

- andy

Court Briefs

- http://cand.uscourts.gov/wha/oraclevgoogle/docs



Oracle America, Inc. v. Google Inc. Selected Filings

This page provides direct access to selected key documents in the case without the need to log in to PACER or incur PACER charges. This listing represents only a small portion of all filings in the case and is not intended to be comprehensive. Please log into PACER to view the full docket.

	Docket Number	Filing Date
Amended Complaint & Exhibits (.zip, 3 MB)	36	10/26/2010
Oracle's Reply to Answer to Complaint & Counterclaims (.pdf, 45 KB)	41	10/28/2010
Google's Answer to Amended Cplt & Amended Counterclaims (.pdf, 124 KB)	51	11/10/2010
Google Trial Brief-- Public (.pdf, 4 MB)	534	10/14/2011
Oracle Trial Brief -- Unredacted (.pdf, 176 KB)	568	10/27/2012
Oracle's Reply to Google's Answer & Counterclaims (.pdf, 63 KB)	60	11/29/2010
Final Pretrial Order (.pdf, 25 KB)	675	01/04/2012

Court Briefs

- The Introduction and Overview portions are good summaries of the positions
- read over the summaries provided