Intro

Building Blocks

Launcher

More Easy Stuff

Some Stuff That Looks Easy Because I'm Leaving Most Of It Out

But Wait!

Questions

?

Intro

whatisthis

Android App Assimilation

A quick tour of Android integration points

prepared for Strange Loop 2011

whoami

Logan Johnson

Lead Developer, fanforce

@loganj

http://gplus.to/loganj

whymanwhy

whatisthis

Android App Assimilation

A quick tour of Android integration points

prepared for Strange Loop 2011

whoami

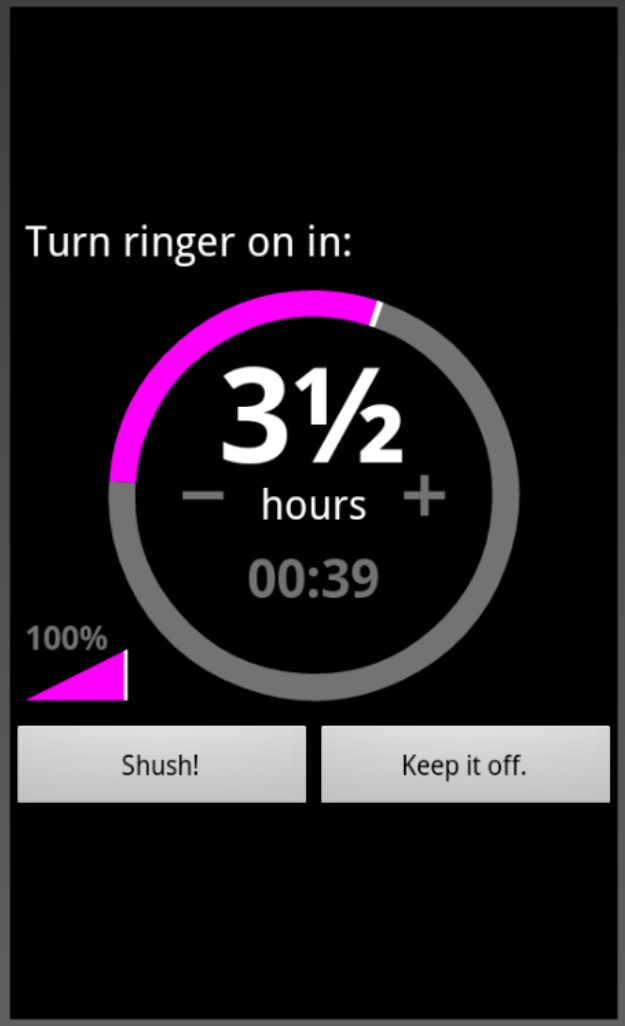
Logan Johnson

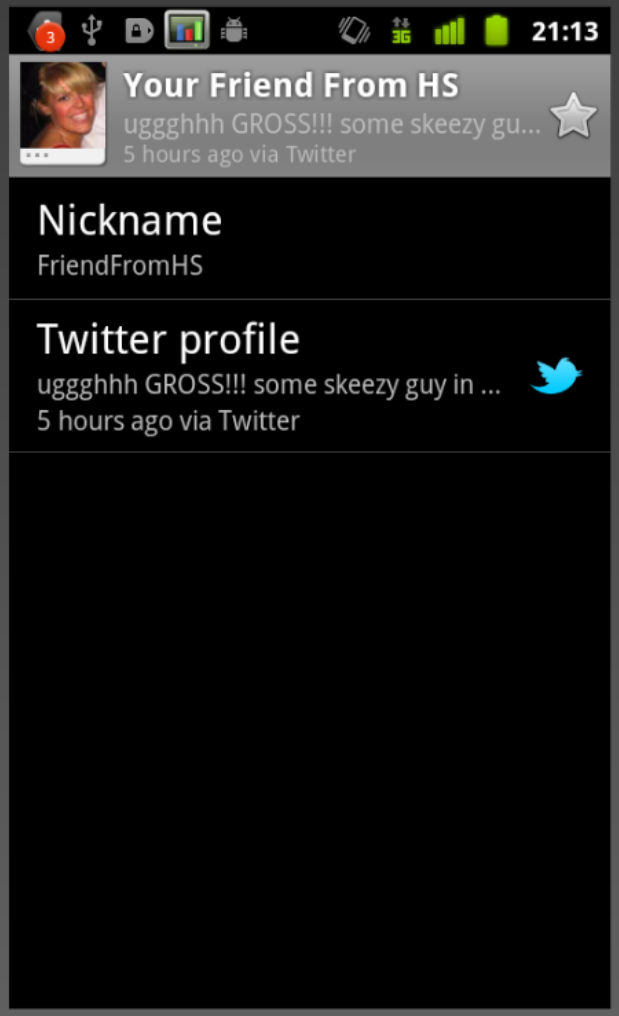
Lead Developer, Fanforce

@loganj

http://gplus.to/loganj

whymanwhy


Turn ringer on in:
3½
hours
00:39
100%
Shush!
Keep it off.


21:13
Your Friend From HS
uggghhh GROSS!!! some skeezy gu...
5 hours ago via Twitter
Nickname
FriendFromHS
Twitter profile
uggghhh GROSS!!! some skeezy guy in ...
5 hours ago via Twitter

Intro

whatisthis

Android App Assimilation

A quick tour of Android integration points

prepared for Strange Loop 2011

whoami

Logan Johnson

Lead Developer, fanforce

@loganj

http://gplus.to/loganj

whymanwhy

Building Blocks

Intents

ContentProviders

- provide access to your app's data on request
- classes in android.content define, assist with a high-level protocol

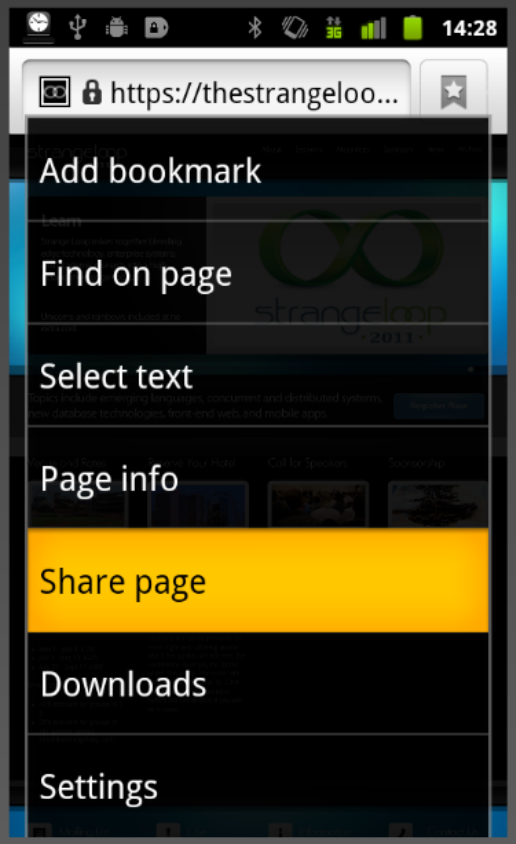
Providers

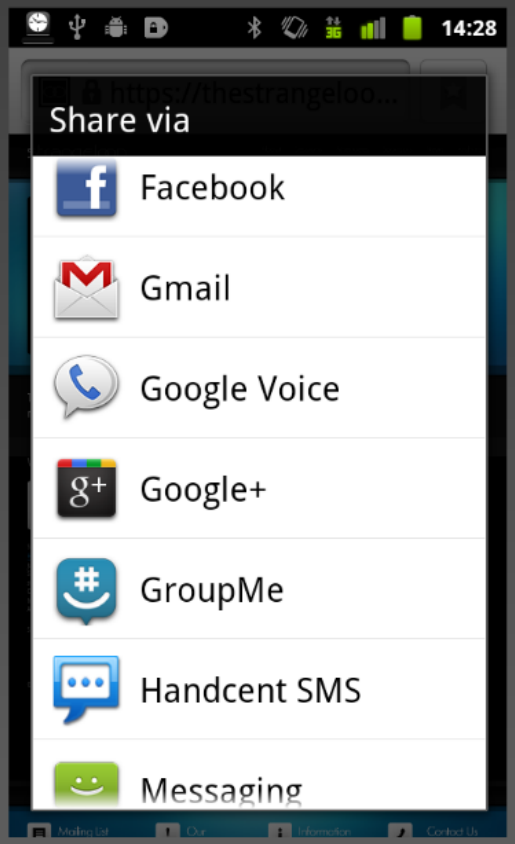
- really client libraries
- ContentProviders + Intents + interface

RemoteViews

- Declarative view definition
- Packages a layout with basic behavior
- "Basic behavior" == PendingIntents for onClick
- Helpful utility methods for common views

Intents





Refresher

- explicit: new Intent(className, url)
- implicit: new Intent(ACTION_VIEW, url)
- implicit intents are "caught" by intent filters
- Intents are Parcelable
- PendingIntents for deferred firing
- may be broadcast by the system

Refresher

- explicit: new Intent(className, url)
- implicit: new Intent(ACTION_VIEW, url)
- implicit intents are "caught" by intent filters
- Intents are Parcelable
- PendingIntents for deferred firing
- may be broadcast by the system

ContentProviders

- provide access to your app's data on request
- classes in android.content define, assist with a high-level protocol

Data Model

_ID	foo	bar	baz
Long	Float	String	byte[]

- Tabular, no joins
- Can also provide InputStreams and OutputStreams

Operations

- operations are encoded as Uris
- Client is ContentResolver or CursorLoader
- query result is a Cursor over rows
- inserts/updates are packed into ContentValues
- there's a batching mechanism, ContentProviderOperations

Data Model

_ID	foo	bar	baz
Long	Float	String	byte[]

- Tabular, no joins
- Can also provide InputStreams and OutputStreams

Operations

- operations are encoded as Uris
- Client is ContentResolver or CursorLoader
- query result is a Cursor over rows
- inserts/updates are packed into ContentValues
- there's a batching mechanism, ContentProviderOperations

Providers

- really client libraries
- ContentProviders + Intents + interface

RemoteViews

- Declarative view definition
- Packages a layout with basic behavior
- "Basic behavior" == PendingIntents for onClick
- Helpful utility methods for common views

Building Blocks

Intents

ContentProviders

- provide access to your app's data on request
- classes in android.content define, assist with a high-level protocol

Providers

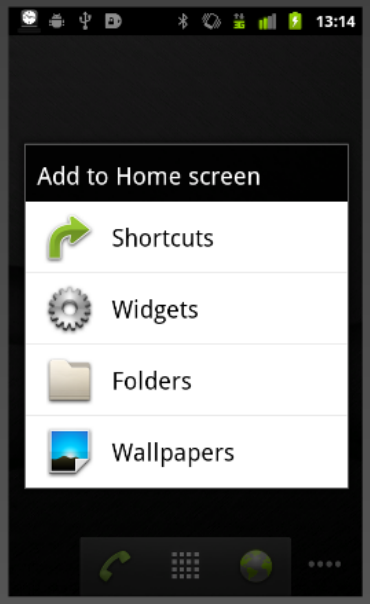
- really client libraries
- ContentProviders + Intents + interface

RemoteViews

- Declarative view definition
- Packages a layout with basic behavior
- "Basic behavior" == PendingIntents for onClick
- Helpful utility methods for common views

Launcher

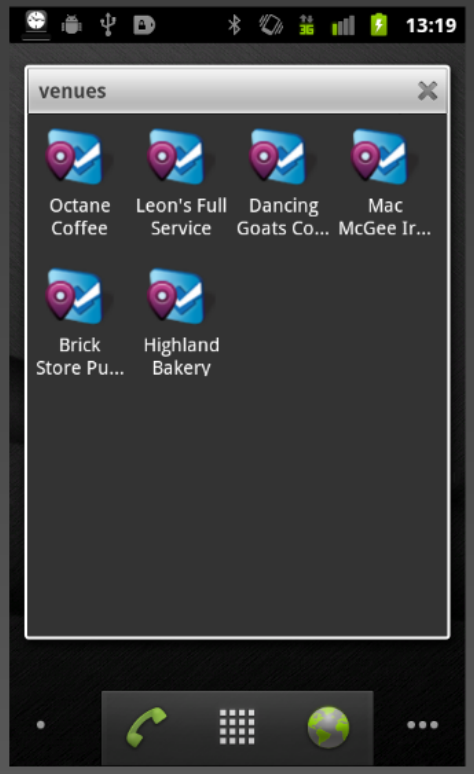
Shortcuts

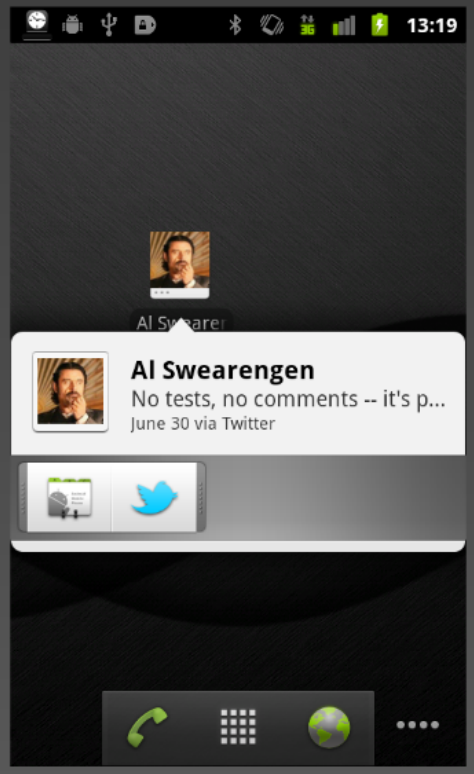


AppWidgets

Live Folders

Shortcuts





Manifest

```
<activity
  a:name=".ListTracks"
  a:label="@string/list_tracks_label"
  a:launchMode="singleTop">

  <intent-filter>
    <action
      a:name="android.intent.action.CREATE_SHORTCUT" />
    <category
      a:name="android.intent.category.DEFAULT" />
  </intent-filter>

</activity>
```

Activity Impl (1/2)

```
public class ListTracks extends Activity {

  private boolean mCreating;

  public void onCreate(Bundle icicle) {

    mCreating = ACTION_CREATE_SHORTCUT.equals(getIntent().getAction());

    mListView.setOnItemClickListener(new OnItemClickListener() {
      public void onItemClick(AdapterView> parent, View view, int pos, long id) {
        Track track = (Track) parent.getAdapter().getItem(position);
        if (mCreating) {
          Intent shortcut = createShortcut(track);
          setResult(RESULT_OK, shortcut);
          finish();
        }
      }
    });

  }

  // ...
}
```

Activity Impl (2/2)

```
import static android.content.Intent;

public class ListTracks extends Activity {
  // ...

  private Intent createShortcut(Track track) {
    Intent listSessions = new Intent(ACTION_MAIN);
    listSessions.setClassName(this, ListSessions.class.getName());
    listSessions.putExtra(ListSessions.EXTRA_TRACK_ID, track.getId());

    Intent result = new Intent();
    result.putExtra(EXTRA_SHORTCUT_INTENT, listSessions);
    result.putExtra(EXTRA_SHORTCUT_NAME, track.getName());
    result.putExtra(EXTRA_SHORTCUT_ICON_RESOURCE,
    Intent.ShortcutIconResource.fromContext(this, R.drawable.track_shortcut_icon));

    return result;
  }
}
```

Manifest

```
<activity
  a:name=".ListTracks"
  a:label="@string/list_tracks_label"
  a:launchMode="singleTop">

  <intent-filter>
    <action
      a:name="android.intent.action.CREATE_SHORTCUT" />
    <category
      a:name="android.intent.category.DEFAULT" />
  </intent-filter>


</activity>
```

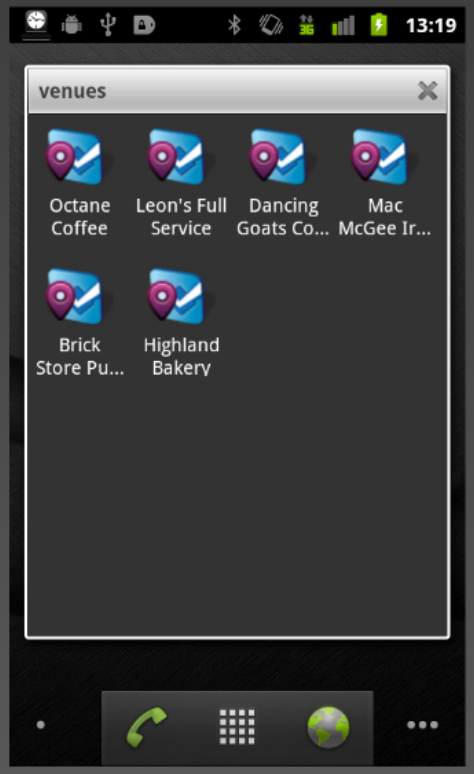
Activity Impl (1/2)

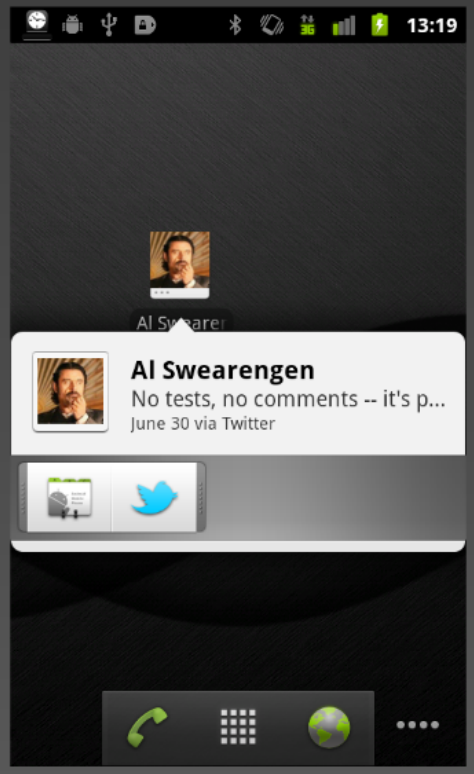
```
public class ListTracks extends Activity {

  private boolean mCreatingShortcut;

  public void onCreate(Bundle icicle) {

    mCreatingShortcut = ACTION_CREATE_SHORTCUT.equals(getIntent().getAction());

    mListView.setOnItemClickListener(new OnItemClickListener() {
      public void onItemClick(AdapterView<?> parent, View view, int pos, long id) {
        Track track = (Track) parent.getAdapter().getItem(position);
        if (mCreatingShortcut) {
          Intent shortcut = createShortcut(track);
          setResult(RESULT_OK, shortcut);
          finish();
        }
      }
    });

  }

  // ...
}
```

Activity Impl (2/2)

```
import static android.content.Intent.*;

public class ListTracks extends Activity {
  //...

  private Intent createShortcut(Track track) {
    Intent listSessions = new Intent(ACTION_MAIN);
    listSessions.setClassName(this, ListSessions.class.getName());
    listSessions.putExtra(ListSessions.EXTRA_TRACK_ID, track.getId());

    Intent result = new Intent();
    result.putExtra(EXTRA_SHORTCUT_INTENT, listSessions);
    result.putExtra(EXTRA_SHORTCUT_NAME, track.getName());
    result.putExtra(EXTRA_SHORTCUT_ICON_RESOURCE,
    Intent.ShortcutIconResource.fromContext(this, R.drawable.track_shortcut_icon));

    return result;
  }

}
```

Shortcuts





Manifest

```
<activity
  a:name=".ListTracks"
  a:label="@string/list_tracks_label"
  a:launchMode="singleTop">

  <intent-filter>
    <action
      a:name="android.intent.action.CREATE_SHORTCUT" />
    <category
      a:name="android.intent.category.DEFAULT" />
  </intent-filter>

</activity>
```

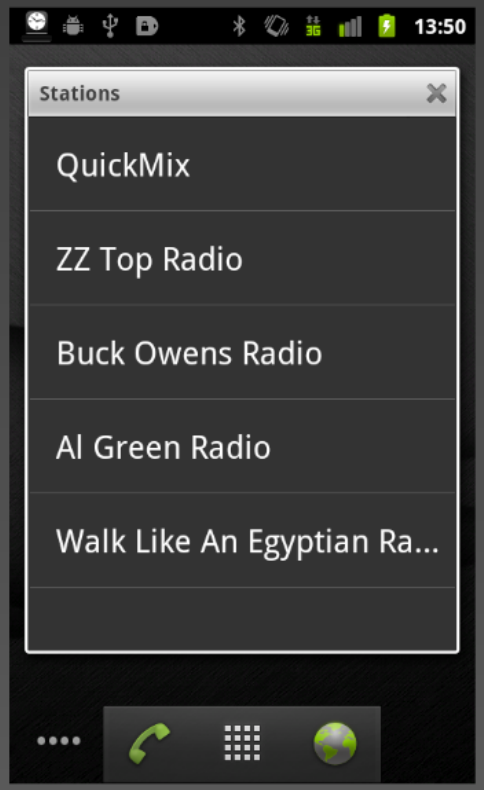
Activity Impl (1/2)

```
public class ListTracks extends Activity {

  private boolean mCreating;

  public void onCreate(Bundle icicle) {

    mCreating = ACTION_CREATE_SHORTCUT.equals(getIntent().getAction());

    mListView.setOnItemClickListener(new OnItemClickListener() {
      public void onItemClick(AdapterView> parent, View view, int pos, long id) {
        Track track = (Track) parent.getAdapter().getItem(position);
        if (mCreating) {
          Intent shortcut = createShortcut(track);
          setResult(RESULT_OK, shortcut);
          finish();
        }
      }
    });

  }

  // ...
}
```

Activity Impl (2/2)

```
import static android.content.Intent;

public class ListTracks extends Activity {
  // ...

  private Intent createShortcut(Track track) {
    Intent listSessions = new Intent(ACTION_MAIN);
    listSessions.setClassName(this, ListSessions.class.getName());
    listSessions.putExtra(ListSessions.EXTRA_TRACK_ID, track.getId());

    Intent result = new Intent();
    result.putExtra(EXTRA_SHORTCUT_INTENT, listSessions);
    result.putExtra(EXTRA_SHORTCUT_NAME, track.getName());
    result.putExtra(EXTRA_SHORTCUT_ICON_RESOURCE,
    Intent.ShortcutIconResource.fromContext(this, R.drawable.track_shortcut_icon));

    return result;
  }
}
```

Live Folders



Manifest

```
<activity
  a:name=".ListTracks"
  a:label="@string/list_tracks_label"
  a:launchMode="singleTop">

  <intent-filter>
    <action
      a:name="android.intent.action.CREATE_LIVE_FOLDER" />
    <category
      a:name="android.intent.category.DEFAULT" />
  </intent-filter>

</activity>
```

Activity Impl (1/2)

```
import static android.content.Intent.*;

public class ListTracks extends Activity {

  public void onCreate(Bundle bundle) {

    creatingLiveFolder = ACTION_CREATE_LIVE_FOLDER.equals(getIntent().getAction());

    mListView.setOnItemClickListener(new OnItemClickListener() {
      public void onItemClick(AdapterView<?> parent, View view, int pos, long id) {

        Track track = (Track) parent.getAdapter().getItem(pos);

        if (mCreatingLiveFolder) {
          Intent liveFolder = createLiveFolder(track);
          setResult(RESULT_OK, liveFolder);
          finish();
        }
      }
    });
  }

  // ...
}
```

ContentProvider

```
public class TrackFolderProvider extends ContentProvider {

  final static HashMap<String,String> PROJECTION = new HashMap<String,String>();
  static {
    PROJECTION.put(LiveFolders._ID, ...);
    PROJECTION.put(LiveFolders.NAME, ...);
    PROJECTION.put(LiveFolders.DESCRIPTION, ...);
    PROJECTION.put(LiveFolders.INTENT, ...);
    // other
    PROJECTION.put(LiveFolders.ICON_BITMAP, ...);
    // or
    PROJECTION.put(LiveFolders.ICON_PACKAGE, ...);
    PROJECTION.put(LiveFolders.ICON_RESOURCE, ...);
  }

  public Cursor query(Uri uri, final String[] proj, String selection,
                      String[] args, String sort) {
    long trackId = ContentUris.parseId(uri);
    SQLiteQueryBuilder qb = new SQLiteQueryBuilder();
    qb.appendWhere(Tracks._ID + "=" + trackId);
    qb.setProjectionMap(PROJECTION);
    // ...
    return qb.query(...);
  }
}
```

Activity Impl (2/2)

```
import static android.provider.LiveFolders.*;

public class ListTracks extends Activity {
  // ...

  private Intent createLiveFolder(Track track) {
    Uri uri = TrackFolderProvider.CONTENT_URI(track.getId());
    Intent result = new Intent();
    result.setData(uri);
    result.putExtra(EXTRA_LIVE_FOLDER_NAME, track.getName());
    result.putExtra(EXTRA_LIVE_FOLDER_ICON,
        ShortcutIconResource.fromContext(this, R.drawable.track_shortcut_icon));
    result.putExtra(EXTRA_LIVE_FOLDER_DISPLAY_MODE, DISPLAY_MODE_LIST);
    result.putExtra(EXTRA_LIVE_FOLDER_BASE_INTENT, ...);
    return result;
  }

}
```

Manifest

```
<activity
  a:name=".ListTracks"
  a:label="@string/list_tracks_label"
  a:launchMode="singleTop">

  <intent-filter>
    <action
      a:name="android.intent.action.CREATE_LIVE_FOLDER" />
    <category
      a:name="android.intent.category.DEFAULT" />
  </intent-filter>

</activity>
```

Activity Impl (1/2)

```
import static android.content.Intent.*;

public class ListTracks extends Activity {

  public void onCreate(Bundle icicle) {

    mCreatingLiveFolder = ACTION_CREATE_LIVE_FOLDER.equals(getIntent().getAction());

    mListView.setOnItemClickListener(new OnItemClickListener() {
      public void onItemClick(AdapterView<?> parent, View view, int pos, long id) {

        Track track = (Track) parent.getAdapter().getItem(pos);

        if (mCreatingLiveFolder) {
          Intent liveFolder = createLiveFolder(track);
          setResult(RESULT_OK, liveFolder);
          finish();
        }
      }
    });
  }

  // ...
}
```

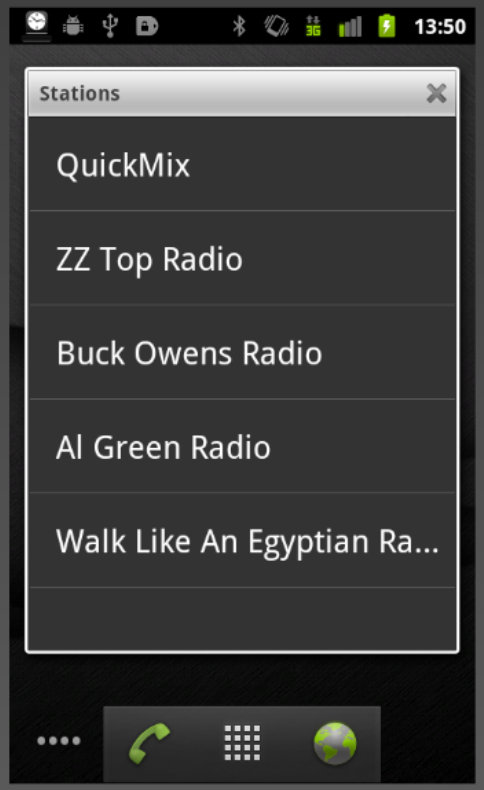
Activity Impl (2/2)

```
import static android.provider.LiveFolders.*;

public class ListTracks extends Activity {
  // ...

  private Intent createLiveFolder(Track track) {
    Uri uri = TrackFolderProvider.contentUri(track.getId());
    Intent result = new Intent();
    result.setData(uri);
    result.putExtra(EXTRA_LIVE_FOLDER_NAME, track.getName())
    result.putExtra(EXTRA_LIVE_FOLDER_ICON,
        ShortcutIconResource.fromContext(this, R.drawable.track_shortcut_icon));
    result.putExtra(EXTRA_LIVE_FOLDER_DISPLAY_MODE, DISPLAY_MODE_LIST);
    result.putExtra(EXTRA_LIVE_FOLDER_BASE_INTENT, ...);
    return result;
  }

}
```

ContentProvider

```
public class TrackFolderProvider extends ContentProvider {

  final static HashMap<String,String> PROJECTION = new HashMap<String,String>();
  static {
    PROJECTION.put(LiveFolders._ID, ...);
    PROJECTION.put(LiveFolders.NAME, ...);
    PROJECTION.put(LiveFolders.DESCRIPTION, ...);
    PROJECTION.put(LiveFolders.INTENT, ...);
    // either
      PROJECTION.put(LiveFolders.ICON_BITMAP, ...);
    // or
      PROJECTION.put(LiveFolders.ICON_PACKAGE, ...);
      PROJECTION.put(LiveFolders.ICON_RESOURCE, ...);
  }

  public Cursor query(Uri uri, final String[] proj, String selection,
                      String[] args, String sort) {
    long trackId = ContentUris.parseId(uri);
    SQLiteQueryBuilder qb = new SQLiteQueryBuilder();
    qb.appendWhere(Session.TRACK_ID + "=" + trackId);
    qb.setProjectionMap(PROJECTION);
    // ...
    return qb.query(...);
  }
}
```

Live Folders



Manifest

```
<activity
  a:name=".ListTracks"
  a:label="@string/list_tracks_label"
  a:launchMode="singleTop">

  <intent-filter>
    <action
      a:name="android.intent.action.CREATE_LIVE_FOLDER" />
    <category
      a:name="android.intent.category.DEFAULT" />
  </intent-filter>

</activity>
```

Activity Impl (1/2)

```
import static android.content.Intent.*;

public class ListTracks extends Activity {

  public void onCreate(Bundle bundle) {

    creatingLiveFolder = ACTION_CREATE_LIVE_FOLDER.equals(getIntent().getAction());

    mListView.setOnItemClickListener(new OnItemClickListener() {
      public void onItemClick(AdapterView<?> parent, View view, int pos, long id) {

        Track track = (Track) parent.getAdapter().getItem(pos);

        if (mCreatingLiveFolder) {
          Intent liveFolder = createLiveFolder(track);
          setResult(RESULT_OK, liveFolder);
          finish();
        }
      }
    });
  }

  // ...
}
```

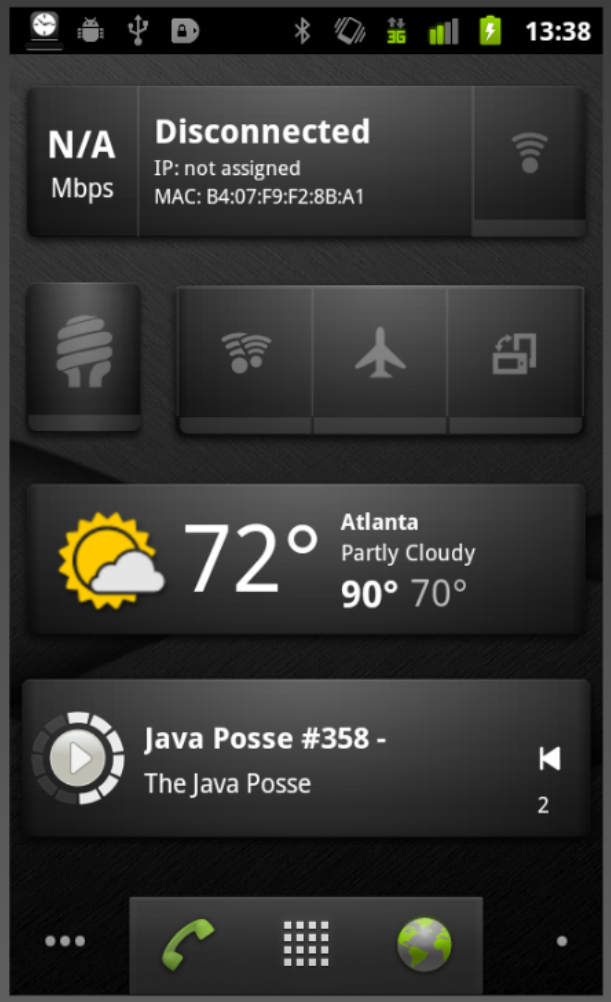
ContentProvider

```
public class TrackFolderProvider extends ContentProvider {

  final static HashMap<String,String> PROJECTION = new HashMap<String,String>();
  static {
    PROJECTION.put(LiveFolders._ID, ...);
    PROJECTION.put(LiveFolders.NAME, ...);
    PROJECTION.put(LiveFolders.DESCRIPTION, ...);
    PROJECTION.put(LiveFolders.INTENT, ...);
    // other
    PROJECTION.put(LiveFolders.ICON_BITMAP, ...);
    // or
    PROJECTION.put(LiveFolders.ICON_PACKAGE, ...);
    PROJECTION.put(LiveFolders.ICON_RESOURCE, ...);
  }

  public Cursor query(Uri uri, final String[] proj, String selection,
                      String[] args, String sort) {
    long trackId = ContentUris.parseId(uri);
    SQLiteQueryBuilder qb = new SQLiteQueryBuilder();
    qb.appendWhere(Tracks._ID + "=" + trackId);
    qb.setProjectionMap(PROJECTION);
    // ...
    return qb.query(...);
  }
}
```

Activity Impl (2/2)

```
import static android.provider.LiveFolders.*;

public class ListTracks extends Activity {
  // ...

  private Intent createLiveFolder(Track track) {
    Uri uri = TrackFolderProvider.CONTENT_URI(track.getId());
    Intent result = new Intent();
    result.setData(uri);
    result.putExtra(EXTRA_LIVE_FOLDER_NAME, track.getName());
    result.putExtra(EXTRA_LIVE_FOLDER_ICON,
        ShortcutIconResource.fromContext(this, R.drawable.track_shortcut_icon));
    result.putExtra(EXTRA_LIVE_FOLDER_DISPLAY_MODE, DISPLAY_MODE_LIST);
    result.putExtra(EXTRA_LIVE_FOLDER_BASE_INTENT, ...);
    return result;
  }

}
```

AppWidgets



Manifest

```
<receiver a:name=".NextSessionWidget"
          a:label="@string/next_session_widget_label">

  <intent-filter>
    <action
      a:name="android.appwidget.action.APPWIDGET_UPDATE" />
  </intent-filter>

  <meta-data a:name="android.appwidget.provider"
      a:resource="@xml/next_session_widget" />

</receiver>
```

Configuration

```
<appwidget-provider xmlns:a="..."
    a:minWidth="300dip"
    a:minHeight="60dip"
    a:updatePeriodMillis="86000"
    a:initialLayout="@layout/next_session_widget" />
```

AppWidgetProvider (1/3)

```
import static android.appwidget.AppWidgetManager.*;

public class NextSessionWidget
             extends AppWidgetProvider {

  @Override
  public void onUpdate(Context context,
                       AppWidgetManager appWidgetManager,
                       int[] appWidgetIds) {

    Intent intent = new Intent(context,
                               StrangeLoopService.class);

    intent.setAction(ACTION_APPWIDGET_UPDATE);
    context.startService(intent);
  }

}
```

Service

```
public class StrangeLoopService extends IntentService {

  @Override
  public void onHandleIntent(Intent i) {

    if (ACTION_APPWIDGET_UPDATE.equals(i.getAction())) {
      AppWidgetManager am
          = AppWidgetManager.getInstance(this);

      Class widgetClass = NextSessionWidget.class;
      String cName = new ComponentName(this, widgetClass);

      int[] widgetIds = am.getAppWidgetIds(cName);
      for (int i = 0; i < widgetIds.length; i++) {
        NextSessionWidget.update(this, am, widgetIds[i]);
      }
    }
  }
}
```

AppWidgetProvider (2/3)

```
public class NextSessionWidget
             extends AppWidgetProvider {
  // ...

  public static void update( Context ctx,
                             AppWidgetManager am,
                             int widgetId ) {

    RemoteViews updatedView = buildRemoteView();

    am.updateAppWidget(widgetId, updatedView);

  }

}
```

AppWidgetProvider (3/3)

```
public class NextSessionWidget
             extends AppWidgetProvider {
  // ...

  private RemoteView buildRemoteView() {
    Session next = getNextSession();
    String layout = R.layout.next_session_widget;

    RemoteViews rv
      = new RemoteViews(ctx.getPackageName(), layout);
    rv.setTextViewText(R.id.title, next.getTitle());
    rv.setOnClickPendingIntent(
      R.id.session,
      new Intent(ACTION_VIEW, next.getContentUri());
    );

    return rv;
  }
}
```

Manifest

```
<receiver a:name=".NextSessionWidget"
          a:label="@string/next_session_widget_label">

  <intent-filter>
    <action
      a:name="android.appwidget.action.APPWIDGET_UPDATE" />
  </intent-filter>

  <meta-data a:name="android.appwidget.provider"
      a:resource="@xml/next_session_widget" />

</receiver>
```

Configuration

```
<appwidget-provider xmlns:a="..."
    a:minWidth="300dip"
    a:minHeight="60dip"
    a:updatePeriodMillis="90000"
    a:initialLayout="@layout/next_session_widget" />
```

AppWidgetProvider (1/3)

```
import static android.appwidget.AppWidgetManager.*;

public class NextSessionWidget
             extends AppWidgetProvider {

  @Override
  public void onUpdate(Context context,
                       AppWidgetManager appWidgetManager,
                       int[] appWidgetIds) {

    Intent intent = new Intent(context,
                               StrangeLoopService.class);

    intent.setAction(ACTION_APPWIDGET_UPDATE);
    context.startService(intent);
  }


}
```

Service

```
public class StrangeLoopService extends IntentService {

  @Override
  public void onHandleIntent(Intent i) {

    if (ACTION_APPWIDGET_UPDATE.equals(i.getAction())) {
      AppWidgetManager am
        = AppWidgetManager.getInstance(this);

      Class widgetClass = NextSessionWidget.class;
      String cName = new ComponentName(this, widgetClass);

      int[] widgetIds = am.getAppWidgetIds(cName);
      for (int i = 0; i < widgetIds.length; i++) {
          NextSessionWidget.update(this, am, widgetIds[i]);
      }
    }
  }
}
```

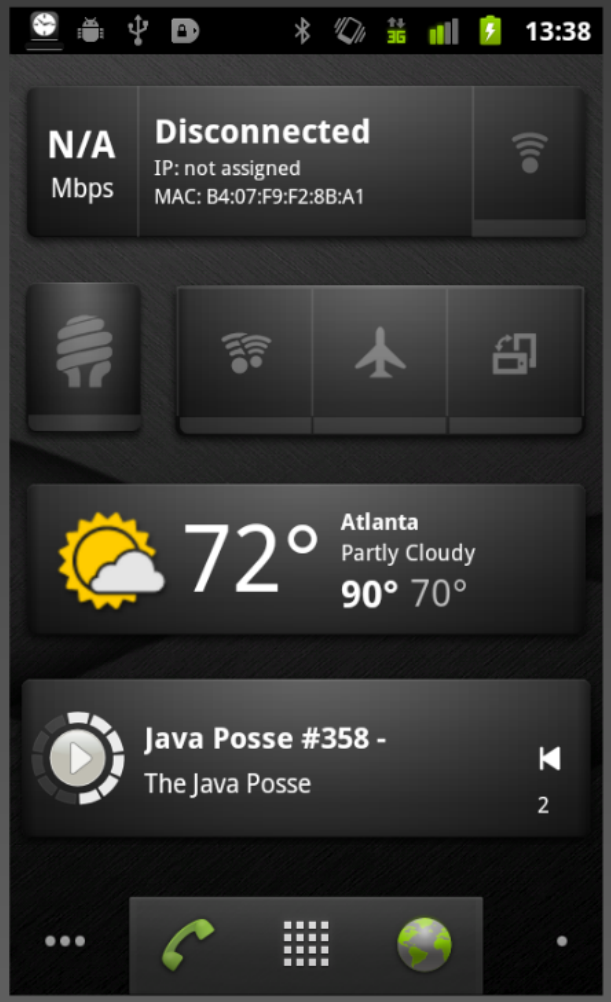
AppWidgetProvider (2/3)

```
public class NextSessionWidget
             extends AppWidgetProvider {
  // ...

  public static void update( Context ctx,
                             AppWidgetManager am,
                             int widgetId ) {

    RemoteViews updatedView = buildRemoteView();

    am.updateAppWidget(widgetId, updatedView);

  }

}
```

AppWidgetProvider (3/3)

```
public class NextSessionWidget
             extends AppWidgetProvider {
  // ...

  private RemoteView buildRemoteView() {
    Session next = getNextSession();
    String layout = R.layout.next_session_widget;

    RemoteViews rv
      = new RemoteViews(ctx.getPackageName(), layout);
    rv.setTextViewText(R.id.title, next.getTitle());
    rv.setOnClickPendingIntent(
      R.id.session,
      new Intent(ACTION_VIEW, next.getContentUri());
    );

    return rv;
  }

}
```

AppWidgets



Manifest

```
<receiver a:name=".NextSessionWidget"
          a:label="@string/next_session_widget_label">

  <intent-filter>
    <action
      a:name="android.appwidget.action.APPWIDGET_UPDATE" />
  </intent-filter>

  <meta-data a:name="android.appwidget.provider"
      a:resource="@xml/next_session_widget" />

</receiver>
```

Configuration

```
<appwidget-provider xmlns:a="..."
    a:minWidth="300dip"
    a:minHeight="60dip"
    a:updatePeriodMillis="86000"
    a:initialLayout="@layout/next_session_widget" />
```

AppWidgetProvider (1/3)

```
import static android.appwidget.AppWidgetManager.*;

public class NextSessionWidget
             extends AppWidgetProvider {

  @Override
  public void onUpdate(Context context,
                       AppWidgetManager appWidgetManager,
                       int[] appWidgetIds) {

    Intent intent = new Intent(context,
                               StrangeLoopService.class);

    intent.setAction(ACTION_APPWIDGET_UPDATE);
    context.startService(intent);
  }

}
```

Service

```
public class StrangeLoopService extends IntentService {

  @Override
  public void onHandleIntent(Intent i) {

    if (ACTION_APPWIDGET_UPDATE.equals(i.getAction())) {
      AppWidgetManager am
          = AppWidgetManager.getInstance(this);

      Class widgetClass = NextSessionWidget.class;
      String cName = new ComponentName(this, widgetClass);

      int[] widgetIds = am.getAppWidgetIds(cName);
      for (int i = 0; i < widgetIds.length; i++) {
        NextSessionWidget.update(this, am, widgetIds[i]);
      }
    }
  }
}
```

AppWidgetProvider (2/3)

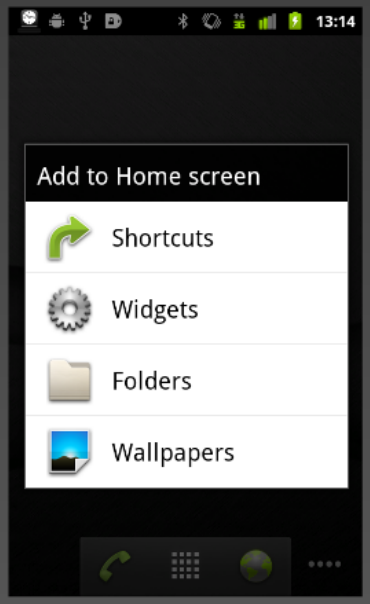
```
public class NextSessionWidget
             extends AppWidgetProvider {
  // ...

  public static void update( Context ctx,
                             AppWidgetManager am,
                             int widgetId ) {

    RemoteViews updatedView = buildRemoteView();

    am.updateAppWidget(widgetId, updatedView);

  }

}
```

AppWidgetProvider (3/3)

```
public class NextSessionWidget
             extends AppWidgetProvider {
  // ...

  private RemoteView buildRemoteView() {
    Session next = getNextSession();
    String layout = R.layout.next_session_widget;

    RemoteViews rv
      = new RemoteViews(ctx.getPackageName(), layout);
    rv.setTextViewText(R.id.title, next.getTitle());
    rv.setOnClickPendingIntent(
      R.id.session,
      new Intent(ACTION_VIEW, next.getContentUri());
    );

    return rv;
  }
}
```

Launcher

Shortcuts



AppWidgets

Live Folders

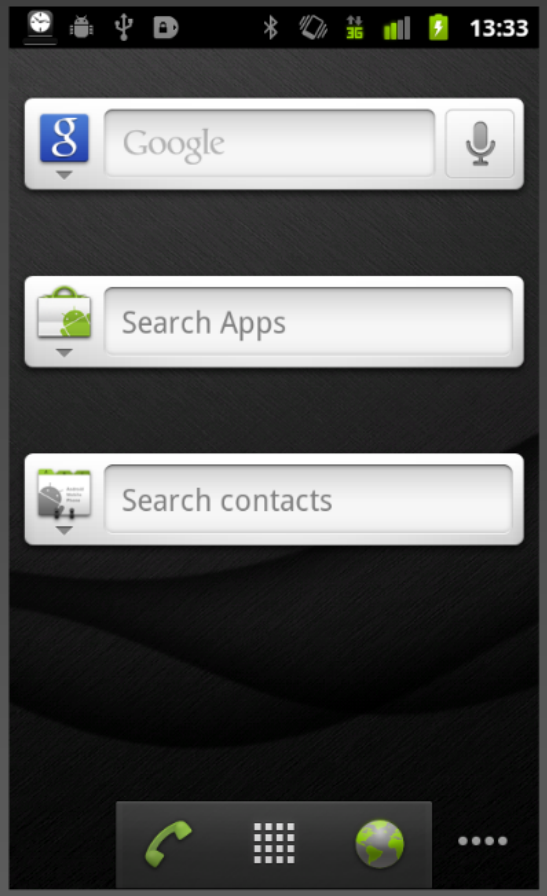
More Easy Stuff

Quick Search Box

QuickContactBadge

Clipboard

Quick Search Box



Searchable Activity

ContentProvider

Searchable Activity

Activity - Manifest

```
<activity a:name=".SearchActivity" >

  <intent-filter>
    <action a:name="android.intent.action.SEARCH" />
  </intent-filter>

  <meta-data a:name="android.app.searchable"
             a:resource="@xml/session_search"/>

</activity>
```

Activity - Config

```
<searchable xmlns:a="..."

  a:label="@string/app_label"

  a:searchSuggestAuthority="com.strangeloop.searchSuggestions"

  a:searchSuggestIntentAction="android.intent.action.VIEW"

  a:searchSettingsDescription="@string/search_settings_desc"

  a:includeInGlobalSearch="true">

</searchable>
```

Activity - Impl

```
public class SearchActivity extends Activity {

  private void handleIntent(Intent intent) {

    if (ACTION_SEARCH.equals(intent.getAction())) {
      String q = intent.getStringExtra(SearchManager.QUERY);
      handleQuery(q);
    } else if (ACTION_VIEW.equals(intent.getAction())) {
      Intent view = new Intent(ACTION_VIEW, intent.getData());
      startActivity(view)
      finish();
    }

  }

}
```

Activity - Manifest

```
<activity a:name=".SearchActivity" >

  <intent-filter>
    <action a:name="android.intent.action.SEARCH" />
  </intent-filter>

  <meta-data a:name="android.app.searchable"
             a:resource="@xml/session_search"/>

</activity>
```

Activity - Config

```
<searchable xmlns:a="..."
  a:label="@string/app_label"
  a:searchSuggestAuthority="com.strangeloop.searchSuggestions"
  a:searchSuggestIntentAction="android.intent.action.VIEW"
  a:searchSettingsDescription="@string/search_settings_desc"
  a:includeInGlobalSearch="true">
</searchable>
```

Activity - Impl

```
public class SearchActivity extends Activity {

  private void handleIntent(Intent intent) {

    if (ACTION_SEARCH.equals(intent.getAction())) {
      String q = intent.getStringExtra(SearchManager.QUERY);
      handleQuery(q);
    } else if (ACTION_VIEW.equals(intent.getAction())) {
      Intent view = new Intent(ACTION_VIEW, intent.getData());
      startActivity(view)
      finish();
    }

  }

}
```

Searchable Activity

Activity - Manifest

```
<activity a:name=".SearchActivity" >

  <intent-filter>
    <action a:name="android.intent.action.SEARCH" />
  </intent-filter>

  <meta-data a:name="android.app.searchable"
             a:resource="@xml/session_search"/>

</activity>
```

Activity - Config

```
<searchable xmlns:a="..."

  a:label="@string/app_label"

  a:searchSuggestAuthority="com.strangeloop.searchSuggestions"

  a:searchSuggestIntentAction="android.intent.action.VIEW"

  a:searchSettingsDescription="@string/search_settings_desc"

  a:includeInGlobalSearch="true">

</searchable>
```

Activity - Impl

```
public class SearchActivity extends Activity {

  private void handleIntent(Intent intent) {

    if (ACTION_SEARCH.equals(intent.getAction())) {
      String q = intent.getStringExtra(SearchManager.QUERY);
      handleQuery(q);
    } else if (ACTION_VIEW.equals(intent.getAction())) {
      Intent view = new Intent(ACTION_VIEW, intent.getData());
      startActivity(view)
      finish();
    }

  }

}
```

ContentProvider

ContentProvider (1/2)

```
import static android.app.SearchManager.*

public class SearchProvider extends ContentProvider {

  final static HashMap<String,String> PROJ_SUGGESTIONS
    = new HashMap<String,String>();

  static {
    PROJ_SUGGESTIONS.put(BaseColumns._ID, ...);
    PROJ_SUGGESTIONS.put(SUGGEST_COLUMN_TEXT_1, ...);
    PROJ_SUGGESTIONS.put(SUGGEST_COLUMN_ICON_1, ...);
    PROJ_SUGGESTIONS.put(SUGGEST_COLUMN_INTENT_DATA, ...);
  }

  // ...

}
```

ContentProvider (2/2)

```
public class SearchProvider extends ContentProvider {
  // ...

  private static final int SEARCH = 0;
  private static final int SUGGEST = 1;

  final static UriMatcher uriMatcher = buildUriMatcher();
  private static UriMatcher buildUriMatcher() {
    UriMatcher m =  new UriMatcher(NO_MATCH);
    m.addUri(AUTHORITY, "sessions", SEARCH);
    m.addURI(AUTHORITY, SUGGEST_URI_PATH_QUERY, SUGGEST);
    return m;
  }

  // ...
}
```

ContentProvider (1/2)

```
import static android.app.SearchManager.*

public class SearchProvider extends ContentProvider {

  final static HashMap<String,String> PROJ_SUGGESTIONS
    = new HashMap<String,String>();

  static {
    PROJ_SUGGESTIONS.put(BaseColumns._ID, ...);
    PROJ_SUGGESTIONS.put(SUGGEST_COLUMN_TEXT_1, ...);
    PROJ_SUGGESTIONS.put(SUGGEST_COLUMN_ICON_1, ...);
    PROJ_SUGGESTIONS.put(SUGGEST_COLUMN_INTENT_DATA, ...);
  }

  // ...

}
```

ContentProvider (2/2)

```
public class SearchProvider extends ContentProvider {
  // ...

  private static final int SEARCH = 0;
  private static final int SUGGEST = 1;

  final static UriMatcher uriMatcher = buildUriMatcher();
  private static UriMatcher buildUriMatcher() {
    UriMatcher m =  new UriMatcher(NO_MATCH);
    m.addUri(AUTHORITY, "sessions", SEARCH);
    m.addURI(AUTHORITY, SUGGEST_URI_PATH_QUERY, SUGGEST);
    return m;
  }

  // ...
}
```

ContentProvider

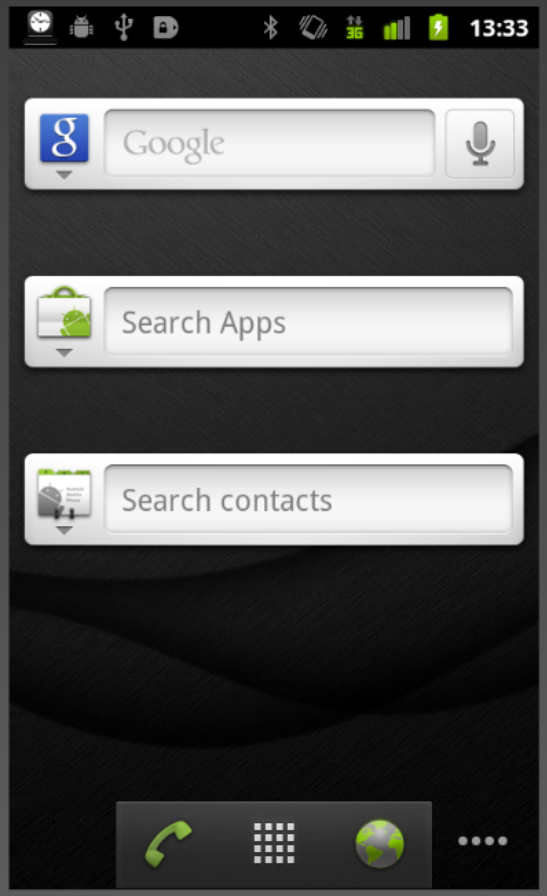
ContentProvider (1/2)

```
import static android.app.SearchManager.*

public class SearchProvider extends ContentProvider {

  final static HashMap<String,String> PROJ_SUGGESTIONS
    = new HashMap<String,String>();

  static {
    PROJ_SUGGESTIONS.put(BaseColumns._ID, ...);
    PROJ_SUGGESTIONS.put(SUGGEST_COLUMN_TEXT_1, ...);
    PROJ_SUGGESTIONS.put(SUGGEST_COLUMN_ICON_1, ...);
    PROJ_SUGGESTIONS.put(SUGGEST_COLUMN_INTENT_DATA, ...);
  }

  // ...

}
```

ContentProvider (2/2)

```
public class SearchProvider extends ContentProvider {
  // ...

  private static final int SEARCH = 0;
  private static final int SUGGEST = 1;

  final static UriMatcher uriMatcher = buildUriMatcher();
  private static UriMatcher buildUriMatcher() {
    UriMatcher m =  new UriMatcher(NO_MATCH);
    m.addUri(AUTHORITY, "sessions", SEARCH);
    m.addURI(AUTHORITY, SUGGEST_URI_PATH_QUERY, SUGGEST);
    return m;
  }

  // ...
}
```

Quick Search Box



Searchable Activity

ContentProvider

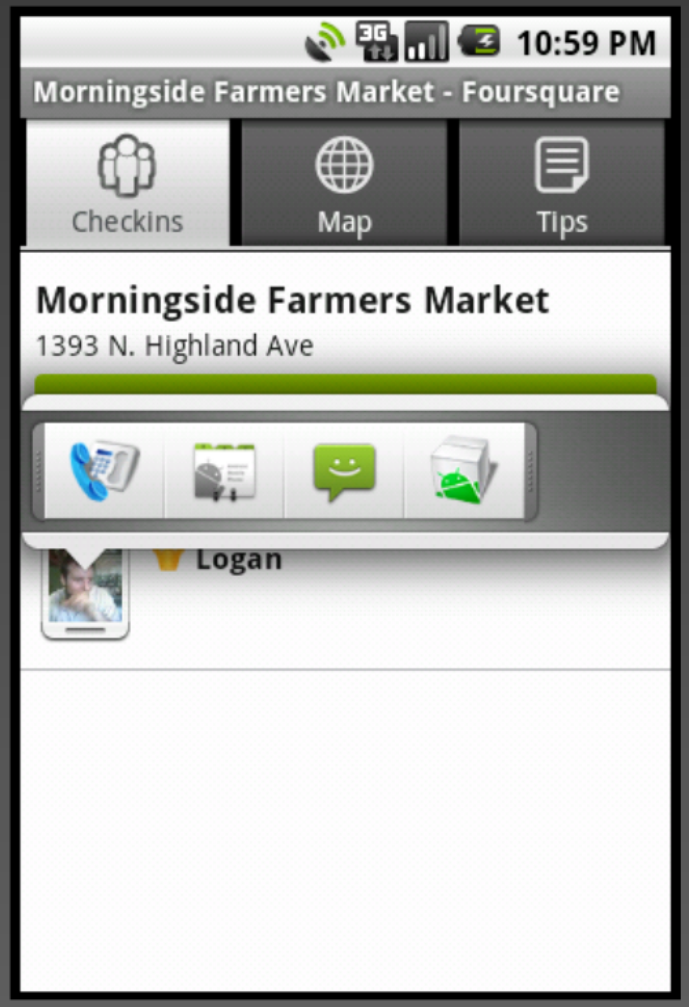
QuickContactBadge

layout

```
<QuickContactBadge
  a:id="@+id/qcFriendPhoto"
  a:layout_height="50dip"
  a:layout_width="44dip"
  a:layout_alignParentTop="true"
  a:layout_alignParentLeft="true"
  a:padding="3dip"
  a:src="@drawable/contact_silhouette"
  a:scaleType="centerInside"
  style="?android:attr/quickContactBadgeStyleWindowSmall"/>
```

code

```
QuickContactBadge qcb = findViewById(R.id.qcFriendPhoto);
qcb.setImageBitmap( friend.getProfilePic() );
qcb.assignContactUri( findContact(friend) );
```



layout

```
<QuickContactBadge
  a:id="@+id/qcFriendPhoto"
  a:layout_height="50dip"
  a:layout_width="44dip"
  a:layout_alignParentTop="true"
  a:layout_alignParentLeft="true"
  a:padding="3dip"
  a:src="@drawable/contact_silhouette"
  a:scaleType="centerInside"
  style="?android:attr/quickContactBadgeStyleWindowSmall"/>
```

code

```
QuickContactBadge qcb = findViewById(R.id.qcFriendPhoto);

qcb.setImageBitmap( friend.getProfilePic() );

qcb.assignContactUri( findContact(friend) );
```

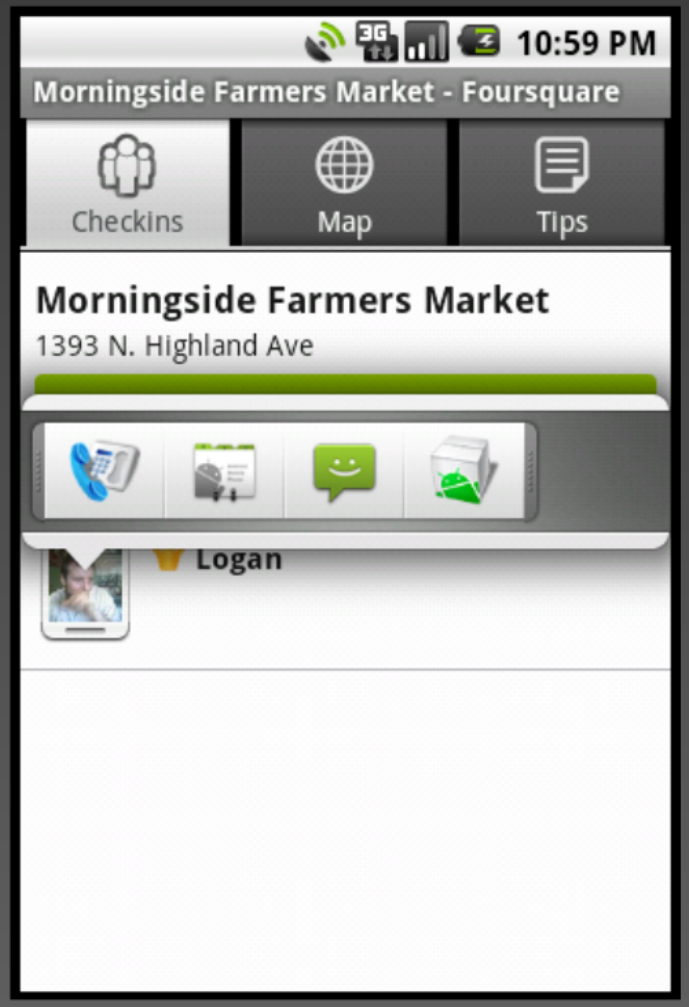
QuickContactBadge

layout

```
<QuickContactBadge
  a:id="@+id/qcFriendPhoto"
  a:layout_height="50dip"
  a:layout_width="44dip"
  a:layout_alignParentTop="true"
  a:layout_alignParentLeft="true"
  a:padding="3dip"
  a:src="@drawable/contact_silhouette"
  a:scaleType="centerInside"
  style="?android:attr/quickContactBadgeStyleWindowSmall"/>
```

code

```
QuickContactBadge qcb = findViewById(R.id.qcFriendPhoto);
qcb.setImageBitmap( friend.getProfilePic() );
qcb.assignContactUri( findContact(friend) );
```



Clipboard

Copy

```
ClipData clip = ClipData.newPlainText("track", "mobile");

// or...

clip = ClipData.newIntent("View Mobile Track", viewTrackIntent(mobileTrackId));

// or...

clip = ClipData.newUri(getContentResolver(), "track info", trackUri(mobileTrackId));
clip.addItem(new ClipData.Item(trackUri(toolsTrackId)));
clip.addItem(new ClipData.Item(trackUri(webTrackId)));

ClipboardManager clipboard
  = (ClipboardManager) getSystemService(Context.CLIPBOARD_SERVICE);

clipboard.setPrimaryClip(clip);
```

Paste

```
if ( clipboard.hasPrimaryClip() ) {

  ClipData clipData = clipboard.getPrimaryClip();
  ClipData.Item item = clipData.getItemAt(0);

  String paste = item.getText();

  // or...
  String pasteText = item.coerceToText(context);

  // or...
  Uri pasteUri = item.getUri();

  // or...
  Intent pasteIntent = item.getItem();

  // or...

  // ...

}
```

Copy

```
ClipData clip = ClipData.newPlainText("track", "mobile");

// or...

clip = ClipData.newIntent("View Mobile Track", viewTrackIntent(mobileTrackId));

// or...

clip = ClipData.newUri(getContentResolver(), "track info", trackUri(mobileTrackId));
clip.addItem(new ClipData.Item(trackUri(toolsTrackId)));
clip.addItem(new ClipData.Item(trackUri(webTrackId)));

ClipboardManager clipboard
  = (ClipboardManager) getSystemService(Context.CLIPBOARD_SERVICE);

clipboard.setPrimaryClip(clip);
```

Paste

```
if ( clipboard.hasPrimaryClip() ) {

  ClipData clipData = clipboard.getPrimaryClip();
  ClipData.Item item = clipData.getItemAt(0);

  String paste = item.getText();

  // or...
  String pasteText = item.coerceToText(context);

  // or...
  Uri pasteUri = item.getUri();

  // or...
  Intent pasteIntent = item.getItem();

  // or...

  // ...

}
```

Clipboard

Copy

```
ClipData clip = ClipData.newPlainText("track", "mobile");

// or...

clip = ClipData.newIntent("View Mobile Track", viewTrackIntent(mobileTrackId));

// or...

clip = ClipData.newUri(getContentResolver(), "track info", trackUri(mobileTrackId));
clip.addItem(new ClipData.Item(trackUri(toolsTrackId)));
clip.addItem(new ClipData.Item(trackUri(webTrackId)));

ClipboardManager clipboard
  = (ClipboardManager) getSystemService(Context.CLIPBOARD_SERVICE);

clipboard.setPrimaryClip(clip);
```

Paste

```
if ( clipboard.hasPrimaryClip() ) {

  ClipData clipData = clipboard.getPrimaryClip();
  ClipData.Item item = clipData.getItemAt(0);

  String paste = item.getText();

  // or...
  String pasteText = item.coerceToText(context);

  // or...
  Uri pasteUri = item.getUri();

  // or...
  Intent pasteIntent = item.getItem();

  // or...

  // ...

}
```

More Easy Stuff

Quick Search Box

QuickContactBadge

Clipboard

Some Stuff That Looks Easy Because I'm Leaving Most Of It Out

Accounts

Contacts

Accounts

Manifest

```
<service a:name=".AuthenticatorService"
         a:exported="true"
         a:process=":auth">

  <intent-filter>
    <action a:name="android.accounts.AccountAuthenticator" />
  </intent-filter>

  <meta-data a:name="android.accounts.AccountAuthenticator"
             a:resource="@xml/authenticator" />

</service>
```

AuthenticatorService

```
public final class AuthenticatorService extends Service {

  public static final String ACCOUNT_TYPE = "com.strangeloop";

  public void onCreate() {
    StrangeLoop strangeLoop = (StrangeLoop)getApplication();
    mAuthenticator = new Authenticator(this);
  }

  public IBinder onBind(Intent intent) {
    String action = intent.getAction();
    if (ACTION_AUTHENTICATOR_INTENT.equals(action)) {
      return mAuthenticator.getIBinder();
    }
    return null;
  }

}
```

Authenticator

```
final class Authenticator extends AbstractAccountAuthenticator {

  Authenticator(Context ctx) {
    mContext = ctx;
  }

  @Override
  public Bundle addAccount(AccountAuthenticatorResponse response, ...) throws ... {

    final Intent intent = new Intent(mContext, LoginActivity.class);
    intent.putExtra(LoginActivity.FROM_ADDACCOUNT, true);
    intent.putExtra(AccountManager.KEY_ACCOUNT_AUTHENTICATOR_RESPONSE, response);

    Bundle reply = new Bundle();
    reply.putParcelable(AccountManager.KEY_INTENT, intent);

    return reply;
  }

}
```

LoginActivity

```
class LoginActivity extends Activity {

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    boolean addAccount = Intent.getBooleanExtra(FROM_ADDACCOUNT);
    AccountAuthenticatorResponse response = null;
    if (addAccount) {
      response = Intent.getParcelableExtra(KEY_ACCOUNT_AUTHENTICATOR_RESPONSE);
      response.onRequestContinued();
    }

    // ...

    if (addAccount) {
      Account account = new Account(username, AuthenticatorService.ACCOUNT_TYPE);
      AccountManager am = AccountManager.get(context);
      am.addAccountExplicitly(account, password, null);

      Bundle result = new Bundle();
      result.putString(AccountManager.KEY_ACCOUNT_NAME, username);
      result.putString(AccountManager.KEY_ACCOUNT_TYPE, STRANGELOOP_ACCOUNT_TYPE);
      result.putString(AccountManager.KEY_AUTHTOKEN, password);

      response.onResult(result);
    }
```

Manifest

```
<service a:name=".AuthenticatorService"
         a:exported="true"
         a:process=":auth">

  <intent-filter>
    <action a:name="android.accounts.AccountAuthenticator" />
  </intent-filter>

  <meta-data a:name="android.accounts.AccountAuthenticator"
             a:resource="@xml/authenticator" />

</service>
```

AuthenticatorService

```
public final class AuthenticatorService extends Service {

  public static final String ACCOUNT_TYPE = "com.strangeloop";

  public void onCreate() {
    StrangeLoop strangeLoop = (StrangeLoop)getApplication();
    mAuthenticator = new Authenticator(this);
  }

  public IBinder onBind(Intent intent) {
    String action = intent.getAction();
    if (ACTION_AUTHENTICATOR_INTENT.equals(action)) {
      return mAuthenticator.getIBinder();
    }
    return null;
  }

}
```

Authenticator

```
final class Authenticator extends AbstractAccountAuthenticator {

  Authenticator(Context ctx) {
    mContext = ctx;
  }

  @Override
  public Bundle addAccount(AccountAuthenticatorResponse response, ...) throws ... {

    final Intent intent = new Intent(mContext, LoginActivity.class);
    intent.putExtra(LoginActivity.PARAM_ADDACCOUNT, true);
    intent.putExtra(AccountManager.KEY_ACCOUNT_AUTHENTICATOR_RESPONSE, response);

    Bundle reply = new Bundle();
    reply.putParcelable(AccountManager.KEY_INTENT, intent);

    return reply;
  }

}
```

LoginActivity

```
class LoginActivity extends Activity {

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    boolean addAccount = intent.getBooleanExtra(PARAM_ADDACCOUNT);
    AccountAuthenticatorResponse response = null;
    if (addAccount) {
      response = intent.getParcelableExtra(KEY_ACCOUNT_AUTHENTICATOR_RESPONSE);
      response.onRequestContinued();
    }

    // ...

    if (addAccount) {
      Account account = new Account(username, AuthenticatorService.ACCOUNT_TYPE);
      AccountManager am = AccountManager.get(context);
      am.addAccountExplicitly(account, password, null);

      Bundle result = new Bundle();
      result.putString(AccountManager.KEY_ACCOUNT_NAME, username);
      result.putString(AccountManager.KEY_ACCOUNT_TYPE, STRANGELOOP_ACCOUNT_TYPE);
      result.putString(AccountManager.KEY_AUTHTOKEN, password);

      response.onResult(result);
    } }
```

Accounts

Manifest

```
<service a:name=".AuthenticatorService"
         a:exported="true"
         a:process=":auth">

  <intent-filter>
    <action a:name="android.accounts.AccountAuthenticator" />
  </intent-filter>

  <meta-data a:name="android.accounts.AccountAuthenticator"
             a:resource="@xml/authenticator" />

</service>
```

AuthenticatorService

```
public final class AuthenticatorService extends Service {

  public static final String ACCOUNT_TYPE = "com.strangeloop";

  public void onCreate() {
    StrangeLoop strangeLoop = (StrangeLoop)getApplication();
    mAuthenticator = new Authenticator(this);
  }

  public IBinder onBind(Intent intent) {
    String action = intent.getAction();
    if (ACTION_AUTHENTICATOR_INTENT.equals(action)) {
      return mAuthenticator.getIBinder();
    }
    return null;
  }

}
```

Authenticator

```
final class Authenticator extends AbstractAccountAuthenticator {

  Authenticator(Context ctx) {
    mContext = ctx;
  }

  @Override
  public Bundle addAccount(AccountAuthenticatorResponse response, ...) throws ... {

    final Intent intent = new Intent(mContext, LoginActivity.class);
    intent.putExtra(LoginActivity.FROM_ADDACCOUNT, true);
    intent.putExtra(AccountManager.KEY_ACCOUNT_AUTHENTICATOR_RESPONSE, response);

    Bundle reply = new Bundle();
    reply.putParcelable(AccountManager.KEY_INTENT, intent);

    return reply;
  }

}
```

LoginActivity

```
class LoginActivity extends Activity {

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    boolean addAccount = Intent.getBooleanExtra(FROM_ADDACCOUNT);
    AccountAuthenticatorResponse response = null;
    if (addAccount) {
      response = Intent.getParcelableExtra(KEY_ACCOUNT_AUTHENTICATOR_RESPONSE);
      response.onRequestContinued();
    }

    // ...

    if (addAccount) {
      Account account = new Account(username, AuthenticatorService.ACCOUNT_TYPE);
      AccountManager am = AccountManager.get(context);
      am.addAccountExplicitly(account, password, null);

      Bundle result = new Bundle();
      result.putString(AccountManager.KEY_ACCOUNT_NAME, username);
      result.putString(AccountManager.KEY_ACCOUNT_TYPE, STRANGELOOP_ACCOUNT_TYPE);
      result.putString(AccountManager.KEY_AUTHTOKEN, password);

      response.onResult(result);
    }
```

Contacts

Adding Contacts (1/5)

```
ArrayList<ContentProviderOperation> addContact(Account acc,
                                               Attendee att
                                               int backRef) {

    ArrayList<ContentProviderOperation> addContact
      = new ArrayList<ContentProviderOperation>();

    addContact.add( baseRecord(acc, att) );

    if ( friend.getEmail() != null ) {
        addContact.add( email(att, backRef) );
    }

    addContact.add( strangeLoop(att, backRef) );

    return addContact;
}
```

Adding Contacts (2/5)

```
import static android.content.ContentProviderOperation.*;

ContentProviderOperation baseRecord(Account acc,
                                    Attendee att) {

  ContentProviderOperation.Builder op
    = newInsert(RawContacts.CONTENT_URI);

  op.withValue(RawContacts.ACCOUNT_NAME, acc.name);
  op.withValue(RawContacts.ACCOUNT_TYPE, acc.type);
  op.withValue(RawContacts.SYNC1, att.getId());
  op.withValue(RawContacts.SOURCE_ID, att.getId());

  return op.build();

}
```

Adding Contacts (3/5)

```
ContentProviderOperation email(Attendee att, int backRef) {

  ContentProviderOperation.Builder op
    = ContentProviderOperation.newInsert(Data.CONTENT_URI);

  op.withValueBackReference(StructuredName.RAW_CONTACT_ID,
                            backRef);
  op.withValue(Data.MIMETYPE, Email.CONTENT_ITEM_TYPE);
  op.withValue(Email.DATA, att.getEmail());

  return op.build();

}
```

Adding Contacts (4/5)

```
final static String PROFILE_MIMETYPE
  = "vnd.android.cursor.item/com.strangeloop.profile"

ContentProviderOperation strangeLoop(Attendee att,
                                     int backRef) {

  ContentProviderOperation.Builder op
    = ContentProviderOperation.newInsert(Data.CONTENT_URI);

  op.withValueBackReference(Data.RAW_CONTACT_ID, backRef);
  op.withValue(Data.MIMETYPE, PROFILE_MIMETYPE);
  op.withValue(Data.DATA1, attendee.getId());
  op.withValue(Data.DATA2, "StrangeLoop");
  op.withValue(Data.DATA3, "View profile");

  return op.build();

}
```

Adding Contacts (5/5)

```
import static com.strangeloop.AuthenticatorService.*;

Account account = new Account(username, ACCOUNT_TYPE);
ArrayList<ContentProviderOperation> ops
  = new ArrayList<ContentProviderOperation>();

for (Attendee att : attendees) {
  ops.addAll( addContact(account, att, ops.size()) );
}

ContentResolver resolver = getContentResolver();
resolver.applyBatch(ContactsContract.AUTHORITY, ops);
```

Adding Contacts (1/5)

```
ArrayList<ContentProviderOperation> addContact(Account acc,
                                               Attendee att
                                               int backRef) {

    ArrayList<ContentProviderOperation> addContact
      = new ArrayList<ContentProviderOperation>();

    addContact.add( baseRecord(acc, att) );

    if ( friend.getEmail() != null ) {
        addContact.add( email(att, backRef) );
    }

    addContact.add( strangeLoop(att, backRef) );

    return addContact;
}
```

Adding Contacts (2/5)

```
import static android.content.ContentProviderOperation.*;

ContentProviderOperation baseRecord(Account acc,
                                    Attendee att) {

  ContentProviderOperation.Builder op
    = newInsert(RawContacts.CONTENT_URI);

  op.withValue(RawContacts.ACCOUNT_NAME, acc.name);
  op.withValue(RawContacts.ACCOUNT_TYPE, acc.type);
  op.withValue(RawContacts.SYNC1, att.getId());
  op.withValue(RawContacts.SOURCE_ID, att.getId());

  return op.build();

}
```

Adding Contacts (3/5)

```
ContentProviderOperation email(Attendee att, int backRef) {

  ContentProviderOperation.Builder op
    = ContentProviderOperation.newInsert(Data.CONTENT_URI);

  op.withValueBackReference(StructuredName.RAW_CONTACT_ID,
                            backRef);
  op.withValue(Data.MIMETYPE, Email.CONTENT_ITEM_TYPE);
  op.withValue(Email.DATA, att.getEmail());

  return op.build();

}
```

Adding Contacts (4/5)

```
final static string PROFILE_MIMETYPE
 = "vnd.android.cursor.item/com.strangeloop.profile"

ContentProviderOperation strangeLoop(Attendee att,
                                     int backRef) {

  ContentProviderOperation.Builder op
    = ContentProviderOperation.newInsert(Data.CONTENT_URI);

  op.withValueBackReference(Data.RAW_CONTACT_ID, backRef);
  op.withValue(Data.MIMETYPE, PROFILE_MIMETYPE);
  op.withValue(Data.DATA1, attendee.getId());
  op.withValue(Data.DATA2, "StrangeLoop");
  op.withValue(Data.DATA3, "View profile");

  return op.build();

}
```

Adding Contacts (5/5)

```
import static com.strangeloop.AuthenticatorService.*;

Account account = new Account(username, ACCOUNT_TYPE);
ArrayList<ContentProviderOperation> ops
  = new ArrayList<ContentProviderOperation>();

for (Attendee att : attendees) {
  ops.addAll( addContact(account, att, opList.size()) );
}

ContentResolver resolver = getContentResolver();
resolver.applyBatch(ContactsContract.AUTHORITY, ops);
```

Contacts

Adding Contacts (1/5)

```
ArrayList<ContentProviderOperation> addContact(Account acc,
                                               Attendee att
                                               int backRef) {

    ArrayList<ContentProviderOperation> addContact
      = new ArrayList<ContentProviderOperation>();

    addContact.add( baseRecord(acc, att) );

    if ( friend.getEmail() != null ) {
        addContact.add( email(att, backRef) );
    }

    addContact.add( strangeLoop(att, backRef) );

    return addContact;
}
```

Adding Contacts (2/5)

```
import static android.content.ContentProviderOperation.*;

ContentProviderOperation baseRecord(Account acc,
                                    Attendee att) {

  ContentProviderOperation.Builder op
    = newInsert(RawContacts.CONTENT_URI);

  op.withValue(RawContacts.ACCOUNT_NAME, acc.name);
  op.withValue(RawContacts.ACCOUNT_TYPE, acc.type);
  op.withValue(RawContacts.SYNC1, att.getId());
  op.withValue(RawContacts.SOURCE_ID, att.getId());

  return op.build();

}
```

Adding Contacts (3/5)

```
ContentProviderOperation email(Attendee att, int backRef) {

  ContentProviderOperation.Builder op
    = ContentProviderOperation.newInsert(Data.CONTENT_URI);

  op.withValueBackReference(StructuredName.RAW_CONTACT_ID,
                            backRef);
  op.withValue(Data.MIMETYPE, Email.CONTENT_ITEM_TYPE);
  op.withValue(Email.DATA, att.getEmail());

  return op.build();

}
```

Adding Contacts (4/5)

```
final static String PROFILE_MIMETYPE
  = "vnd.android.cursor.item/com.strangeloop.profile"

ContentProviderOperation strangeLoop(Attendee att,
                                     int backRef) {

  ContentProviderOperation.Builder op
    = ContentProviderOperation.newInsert(Data.CONTENT_URI);

  op.withValueBackReference(Data.RAW_CONTACT_ID, backRef);
  op.withValue(Data.MIMETYPE, PROFILE_MIMETYPE);
  op.withValue(Data.DATA1, attendee.getId());
  op.withValue(Data.DATA2, "StrangeLoop");
  op.withValue(Data.DATA3, "View profile");

  return op.build();

}
```

Adding Contacts (5/5)

```
import static com.strangeloop.AuthenticatorService.*;

Account account = new Account(username, ACCOUNT_TYPE);
ArrayList<ContentProviderOperation> ops
  = new ArrayList<ContentProviderOperation>();

for (Attendee att : attendees) {
  ops.addAll( addContact(account, att, ops.size()) );
}

ContentResolver resolver = getContentResolver();
resolver.applyBatch(ContactsContract.AUTHORITY, ops);
```

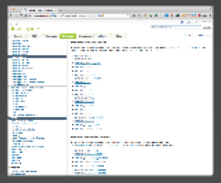
Some Stuff That Looks Easy Because I'm Leaving Most Of It Out

Accounts

Contacts

But Wait!

look for Intents

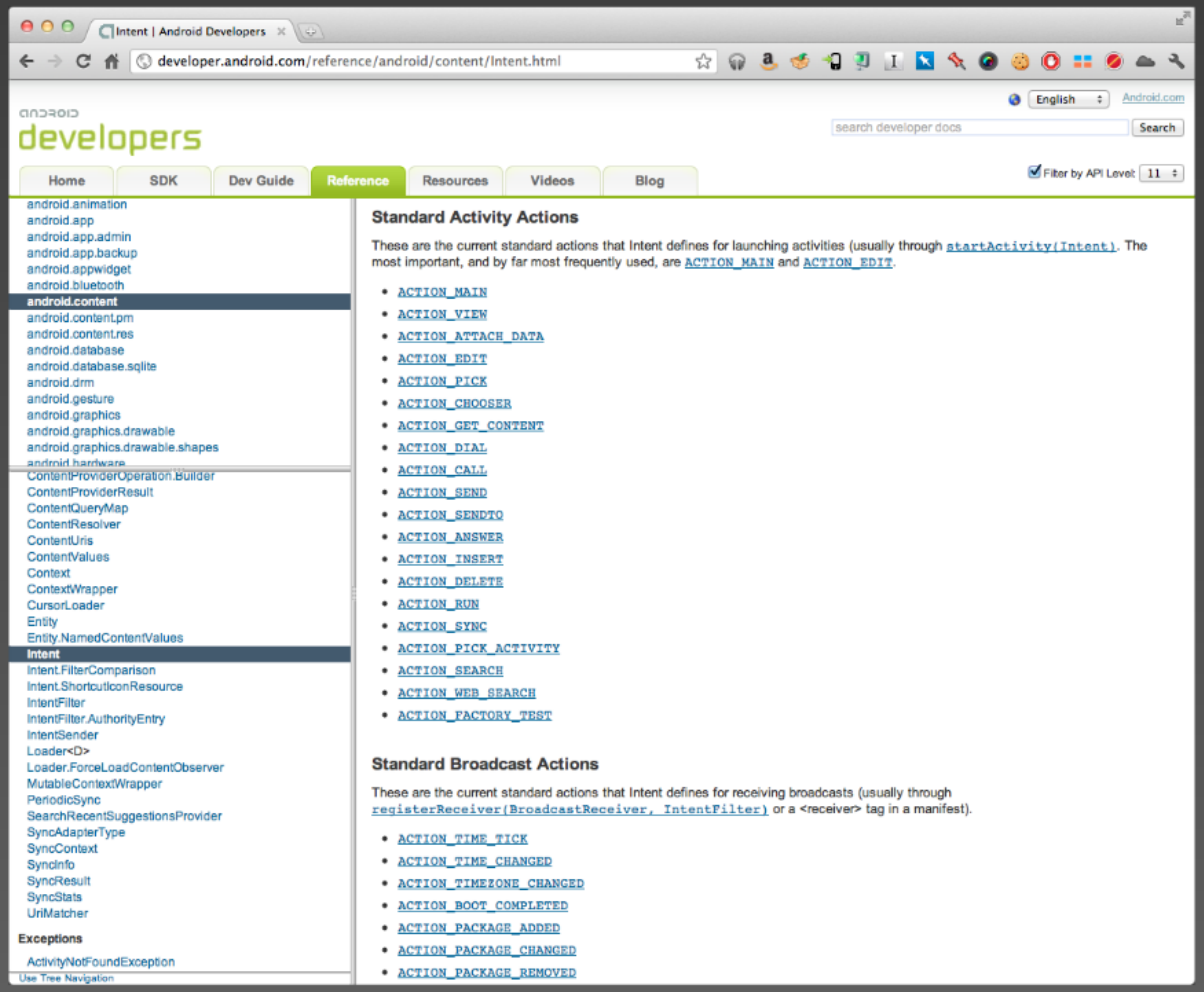


look in android.provider

go spelunking

```
<receiver a:name="NewPhotoReceiver">
  <intent-filter>
    <action
      a:name="com.android.camera.NEW_PICTURE"/>
    <data a:mimeType="image/*"/>
  </intent-filter>
</receiver>
```

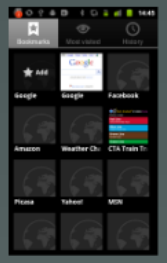
look for Intents



look in android.provider

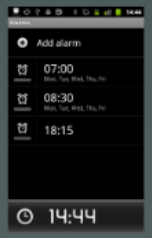
.Browser

- bookmarks
- history
- search history



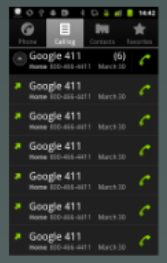
.AlarmClock

- set an alarm



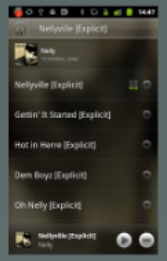
.CallLog

- who called whom
- call duration



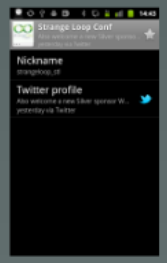
.MediaStore

- audio, images, video
- capture from hardware
- direct updates or MediaScanner



.ContactsContract

- all contact info
- great examples for contract definition or ContentProvider mapping problems

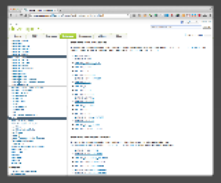


go spelunking

```
<receiver a:name="NewPhotoReceiver">
  <intent-filter>
    <action
      a:name="com.android.camera.NEW_PICTURE"/>
    <data a:mimeType="image/*"/>
  </intent-filter>
</receiver>
```

But Wait!

look for Intents



look in android.provider

go spelunking

```
<receiver a:name="NewPhotoReceiver">
  <intent-filter>
    <action
      a:name="com.android.camera.NEW_PICTURE"/>
    <data a:mimeType="image/*"/>
  </intent-filter>
</receiver>
```

Questions