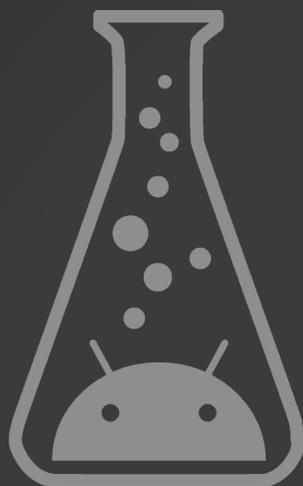


ANDROID
developer lab



ANDROID
developer lab

Introduction to Honeycomb APIs

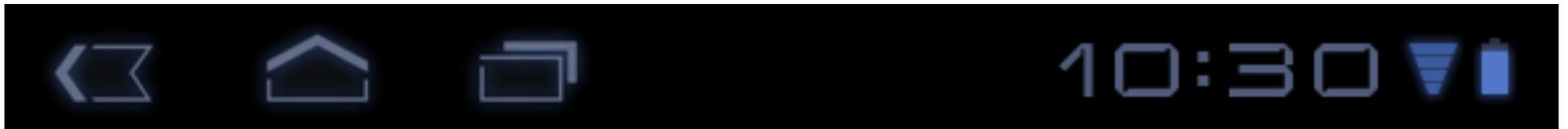
Q3 2011

Honeycomb

- Focused on tablets
- Huge release, many updates and new features
- New holographic system theme
- Version 3.0 (base), 3.1 and 3.2 (point releases), API levels 11/12/13

System Bar

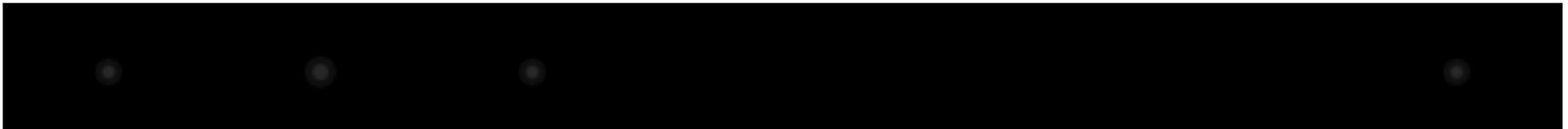
- System-wide navigation and status
- Orientation agnostic
- Always there with varying height
 - ~48dp-56dp
 - design flexible layouts
 - can use `display.getHeight()/getWidth()`



System Bar

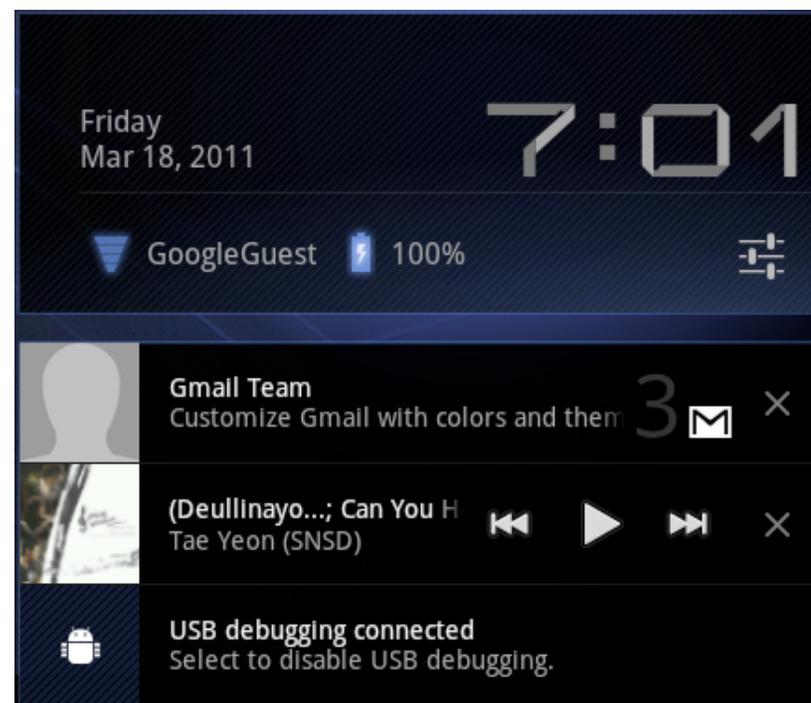
- Lights out mode

```
mView.setSystemUiVisibility(View.STATUS_BAR_HIDDEN);  
mView.setSystemUiVisibility(View.STATUS_BAR_VISIBLE);
```

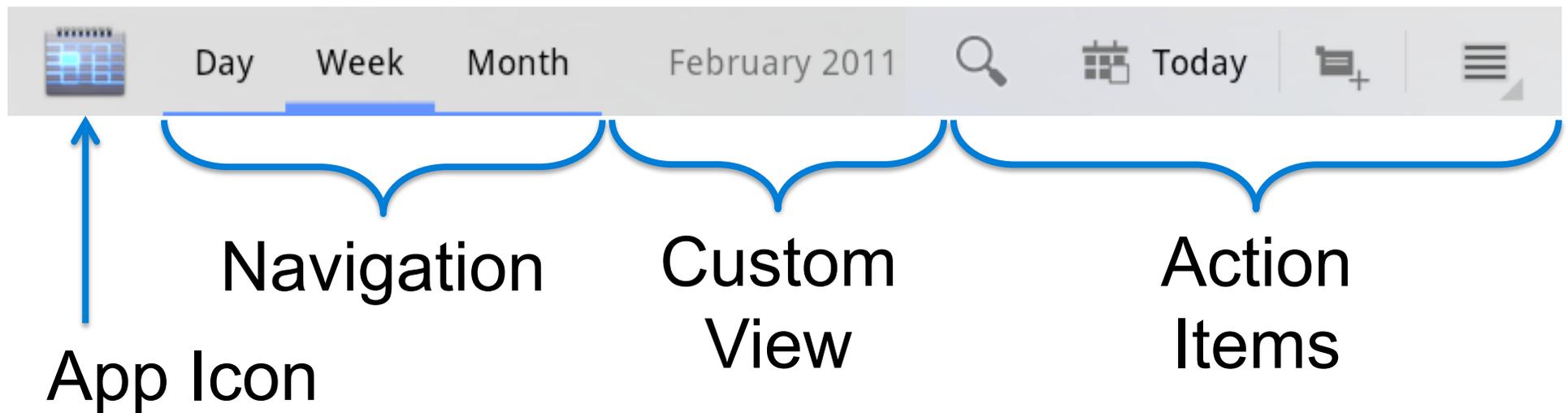


Notifications

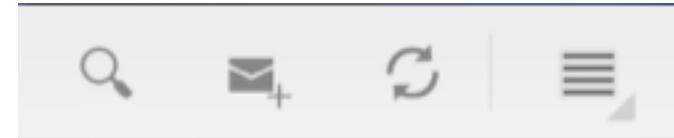
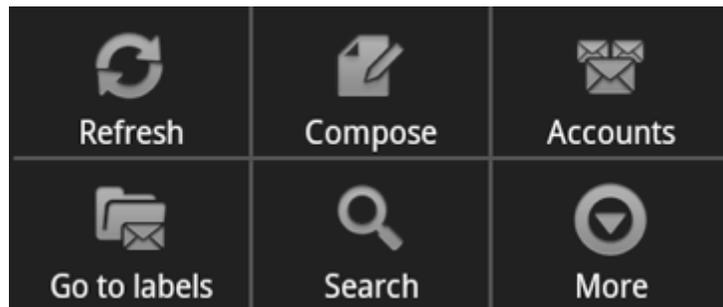
- Android's great notifications, improved
- Dismiss individually
- Customizable
 - Larger icon
 - Actionable buttons



Action Bar



Action Bar (Action Items)



- Menu items from Options Menu
- Easily configured via menu resource file

```
<item android:id="@+id/menu_add"  
    android:icon="@drawable/ic_menu_save"  
    android:title="@string/menu_save"  
    android:showAsAction="ifRoom|withText" />
```

Action Bar (Action Items)

```
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getActivity().getMenuInflater();
    inflater.inflate(R.menu.my_menu, menu);
    return true;
}
```

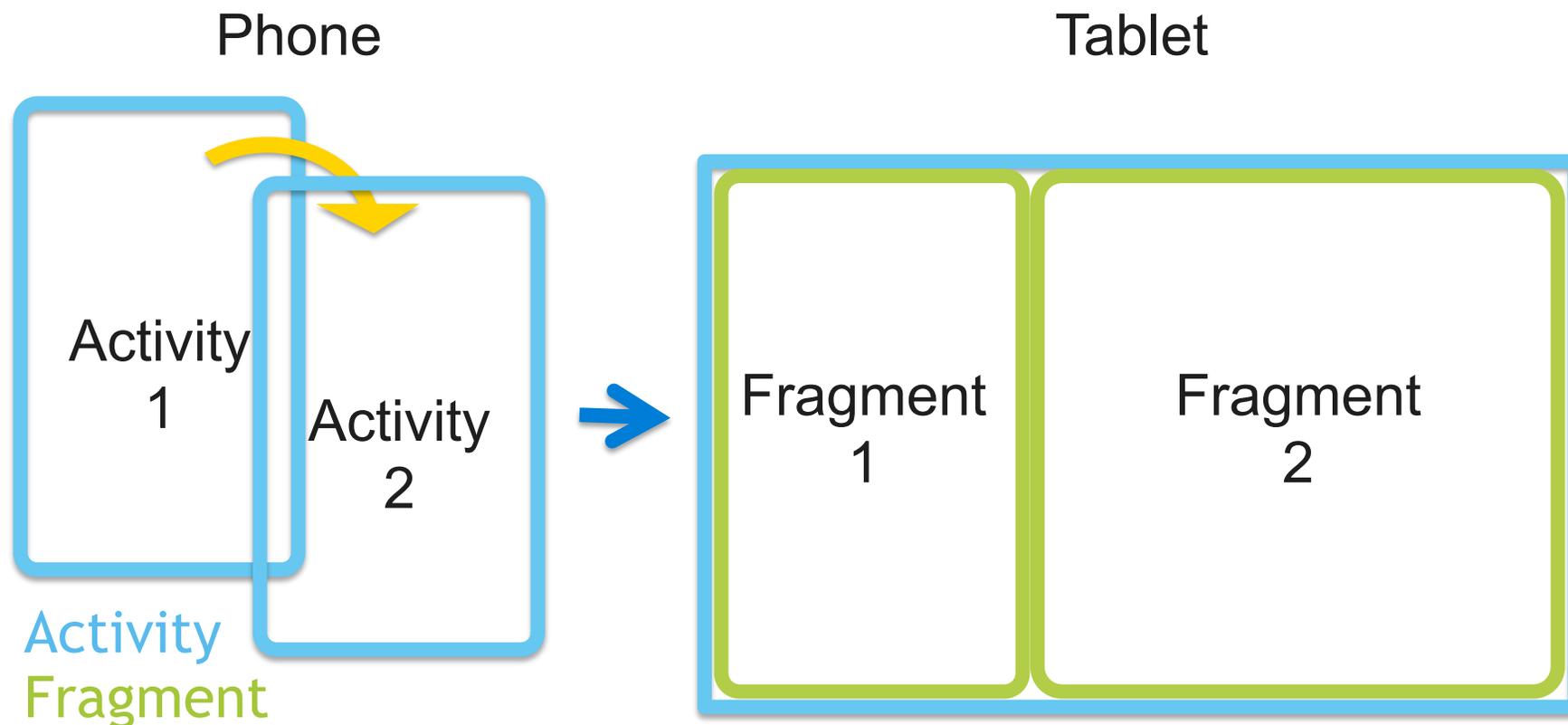
```
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case android.R.id.home:
            // app icon in Action Bar clicked; go home
            return true;
        case R.id.my_menu_item:
            // app menu item selected
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

Redesigned Home Screen Widgets

- Users can interact with home screen widgets in new ways like flipping and scrolling
- New widgets: ListView, GridView, StackView...
- Resizable (from 3.1)

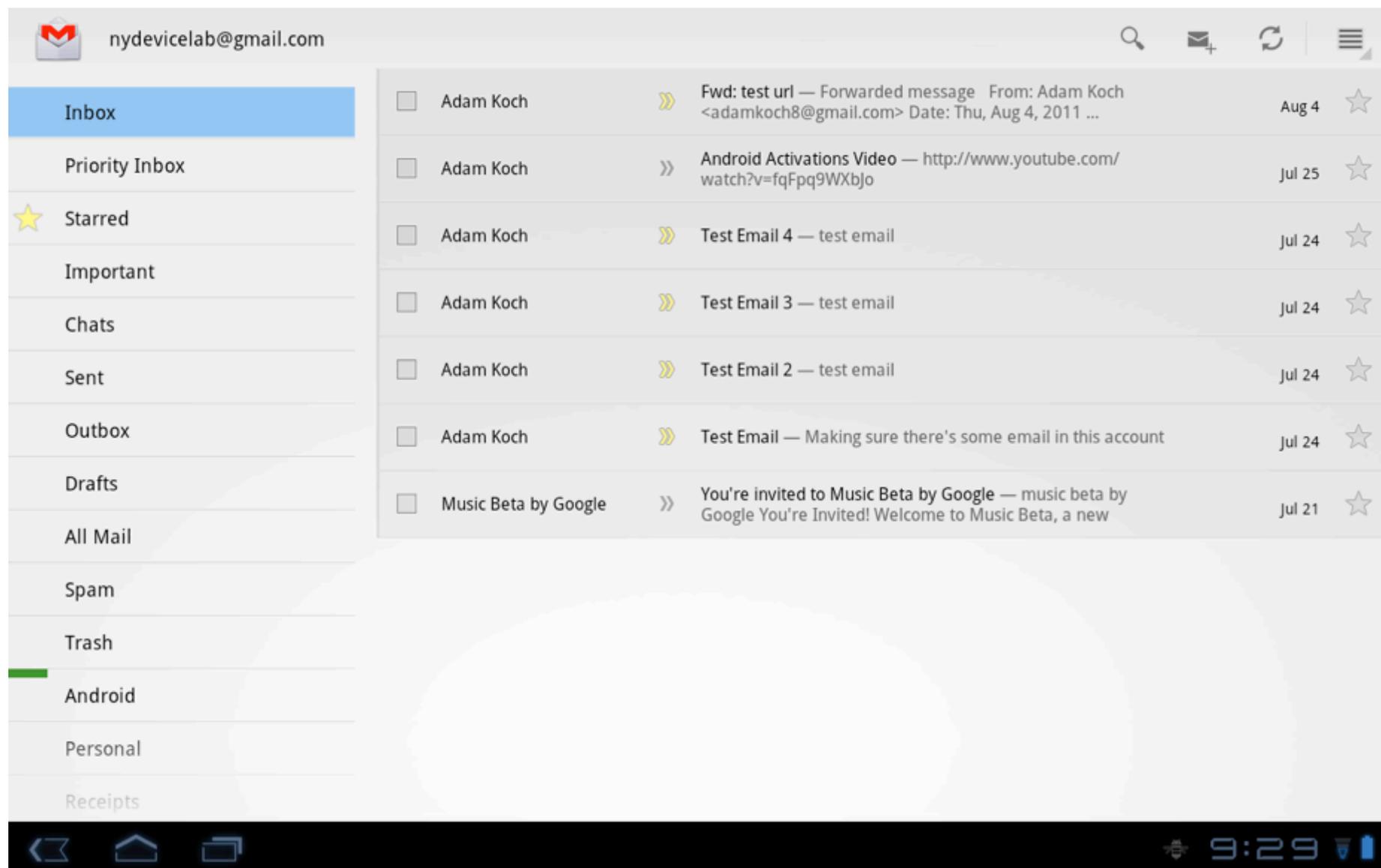


Fragments

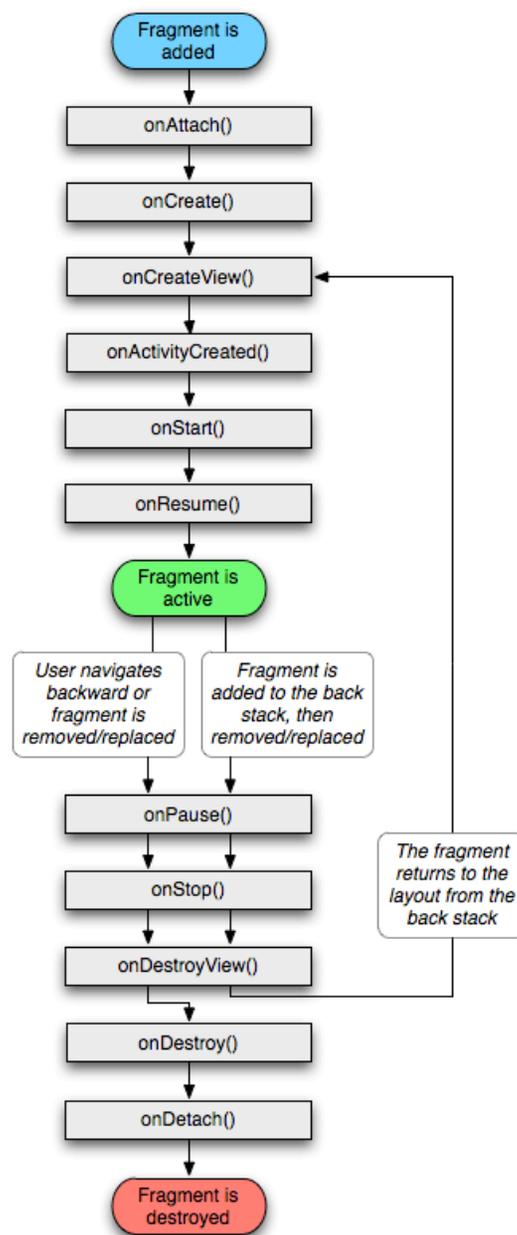


Re-think your UI, don't just let it stretch!

Fragments



Fragments – Lifecycle





Fragments – Other Uses

- Award for best named method:
 - `onRetainNonConfigurationInstance()`
- Instead use:
 - `setRetainInstance(true)`
- Fragments without UI
 - Retain state through configuration changes
 - Use in conjunction with `AsyncTask`

Fragments – Summary

- Reusable UI components within an Activity
- Has its own lifecycle and back stack. Its lifecycle is affected by the host Activity's lifecycle
- Attach to a `ViewGroup` in the Activity view hierarchy through `<fragment>` in XML or programmatically
- Act as a background worker (`findFragmentByTag`)
- Can be added, removed and replaced via `FragmentManager`
- Can communicate with each other via `FragmentManager`



Fragments Example

```
// Get FragmentManager
FragmentManager fragmentManager = getFragmentManager();

// Create new fragment and transaction
Fragment newFragment = new ExampleFragment();

FragmentManager.beginTransaction()
    .beginTransaction();

// Replace view and add to back stack
transaction.replace(R.id.fragment_container, newFragment);
transaction.addToBackStack(null);

// Commit
transaction.commit();
```

Loaders

- Easy way to asynchronously load data in an Activity or Fragment
- Monitors data source and deliver results when content changes
- Automatically reconnect after configuration change



CursorLoader Example

- Implement `LoaderManager.LoaderCallbacks`

```
public Loader<Cursor> onCreateLoader(int id, Bundle args) {  
    ...  
    return new CursorLoader(  
        getActivity(), mUri, mProjection,  
        mSelection, mSelectionArgs, mSortOrder)  
    }  
  
public void onLoadFinished(Loader<Cursor> loader,  
    Cursor data) {  
    mAdapter.swapCursor(data);  
}  
  
public void onLoaderReset(Loader<Cursor> loader) {  
    mAdapter.swapCursor(null);  
}
```

CursorLoader Example

- Init loader in `onCreate()`

```
SimpleCursorAdapter mAdapterer;  
  
public void onCreate(Bundle savedInstanceState) {  
    ...  
    mAdapterer = new SimpleCursorAdapter(...);  
    setListAdapter(mAdapterer);  
    getLoaderManager().initLoader(0, null, this);  
}
```

Clipboard Framework – Copy & Paste

- Supports 3 types of content
 - Text
 - URI
 - Intent
- At any time, only one clip on the clipboard
- For each clip (`clipData`), it can store multiple items of the same type
- You decide what MIME types can be handled by your app

Drag and Drop

- A drag begins by calling

```
view.startDrag(dragData, shadow, null, 0);
```

- To accept a drop implement

```
View.OnDragListener
```

- Use `clipData` to store “drag” data

Hardware Acceleration

- Speed up standard widgets, drawables – all drawing operations on View's Canvas
- Can be set at the Activity, Window and View levels
- Default is disabled

```
<application android:hardwareAccelerated="true">  
  ...  
</application>
```

RenderScript

- High performance 3D rendering and compute API
- Written in C99 (a dialect of C)
- Pros: portability, performance, usability
- Cons: new APIs, debugging, fewer features (compared to OpenGL)

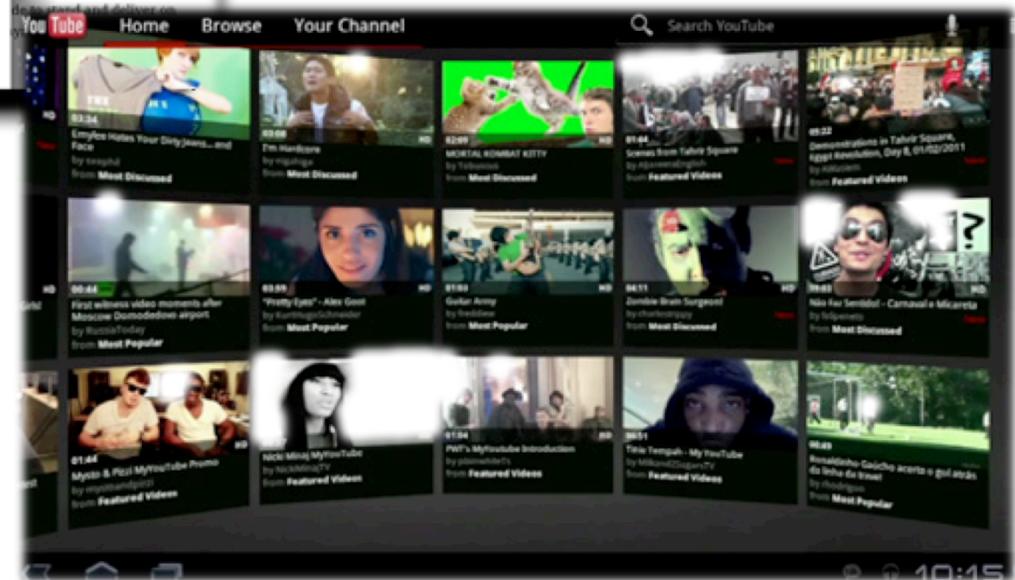


RenderScript – Sample Apps



Google Books

YouTube



Property Animation Framework

- New animation system that can animate any object's properties
- Changes objects and their behavior as well
- Can animate changes to a `ViewGroup`
- `ViewPropertyAnimator` (3.1+) makes animations even simpler and more efficient



Property Animation Framework

- Simple property animation:

```
ObjectAnimator.ofFloat(myView, "alpha", 0f)
    .setDuration(500)
    .start();
```

- Even better using `ViewPropertyAnimator`:

```
myView.animate().setDuration(500).alpha(0);
```



Enterprise

- Support for encrypted storage
- New device administration policy support
 - Encrypted storage
 - Password expiration
 - Password history
 - Password complex character required



Media – Updates from Android 3.0, 3.1, 3.2

- HTTP Live Streaming
- Pluggable DRM framework
- Inline playback of HTML5 `<video>`
- MTP/PTP
- RTP
- Updated Media Formats
 - Raw ADTS AAC, FLAC...





Your App & Honeycomb

Design With Tablets in Mind

- Use density independent pixels (dp)
- Design flexible layouts
- Centralize dimensions using `dimens.xml`
- Keep application logic and UI separate
- Support landscape and don't assume portrait



Updating Your App for Honeycomb

- Test holographic theme
- Update for ActionBar
- Add higher resolution graphics
- Tweak layouts, spacing, font sizes
- Fragments

```
<manifest ... >  
    <uses-sdk android:minSdkVersion="4"  
            android:targetSdkVersion="11" />  
</manifest>
```



Compatibility Library

- Not really a compatibility library anymore, more of a support library
- Works back to API Level 4 (Donut / 1.6)
- Provides:
 - Fragments
 - Loaders
 - ViewPager / PagerAdapter - neat!
 - LruCache
 - and more...



Screen Size Support – Updated in 3.2

- Screen compatibility mode
- Optimizations for a wider range of tablets
- New numeric selectors
 - smallestWidth (res/layout-sw720dp)
 - width (res/layout-w600dp)
 - height (res/layout-h720dp)



Looking Forward

- Ice Cream Sandwich – very tasty dessert





For more, visit
developer.android.com

Copyrights and trademarks

- Android, Google are registered trademarks of Google Inc.
- All other trademarks and copyrights are the property of their respective owners.

