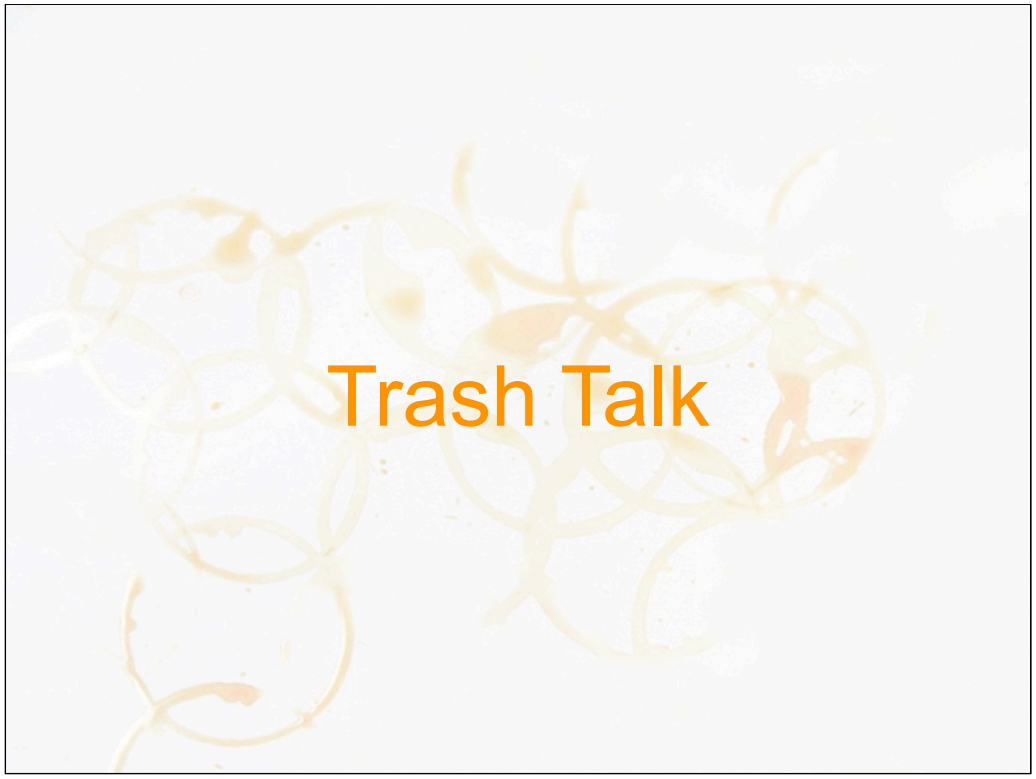
Android UI Development: Tips, Tricks, and Techniques

Romain Guy
Chet Haase
Android UI Toolkit Team
Google


Totally Terrific
Android UI Development:
Tips, Tricks, and Techniques
Romain Guy
Chet Haase
Android UI Toolkit Team
Google


Trash Talk

Trash Talk
or
Garbage Zero

Avoid creating garbage,
when *necessary* and *possible*

Statics as Temporaries

- Instead of a temporary object:

```
public boolean pointInArea(int x, int y, Area area) {
  Point testPoint = new Point(x, y);
  return area.intersect(testPoint);
}
```

- Consider a static instead:

```
static final Point tmpPoint = new Point();

public boolean pointInArea(int x, int y, Area area) {
  tmpPoint.x = x;
  tmpPoint.y = y;
  return area.intersect(tmpPoint.yPoint);
}
```

AutoBoxing

- Autoboxing creates Objects

```
float x = 5;
Float y = x;
doSomething(x);

void doSomething(Float z) {}
```

is equivalent to

```
float x = 5;
Float y = new Float(x);
doSomething(new Float(x));

void doSomething(Float z) {}
```

De-Autoboxing

- Use primitive types whenever possible
  - –Avoids Object creation
- Use types you need for the situation
  - –Avoids autoboxing back and forth

Obliterator

- The enhanced for() loop is great
- ... but creates garbage

```
for (Node node : nodeList) {}
```

is equivalent to

```
Iterator iter = nodeList.iterator();
while (iter.hasNext()) {}
```

- Consider a size check first:

```
if (nodeList.size() > 0) {
  for (Node node : nodeList) {}
}
```

Image is Everything

- Recycle those Bitmaps
  - –Device resources are limited
- Finalizers will clear them ... eventually
- You might think this would help

```
// done using this one, clear reference
myBitmap = null;
```

- But you really want to do this

```
// done using this one, recycle it
myBitmap.recycle();
```

- Don’t wait for the finalizer to do the work if you need that memory *now*

Varargh

- Parameters to varargs method packaged into a temporary array

```
void someMethod(float... args) {}

someMethod(5f);
```

is equivalent to

```
someMethod(new float[]{5});
```

Gener-ick

- T doesn't stand for "primitive Type"

```
public class MyClass<T> {
  T myVar;
  MyClass<T>(T arg) {
    myVar = arg;
  }
}
```

- Generics only deal with Objects; primitive types get autoboxed

```
float f;
MyClass<Float> myObject =
  new MyClass<Float>(f);
```

which is equivalent to

```
MyClass<Float> myObject =
  new MyClass<Float>(new Float(f));
```

Tools: Allocation Tracking

- Limit allocations to find problems

```
int prevLimt = -1;
try {
  prevLimit =
Debug.setAllocationLimit(0);
  // Do stuff
} finally {
  Debug.setAllocationLimit(-1);
}
```

- Count the allocations being made

```
Debug.startAllocationCounting();
// do stuff
int allocCount = Debug.getThreadAllocCount();
Debug.stopAllocationCounting);
```

Tools: DDMS

- Visual tool helps track allocations down to the object/file/line number
- (demo)

Watch the Garbage... But Don't Be Silly

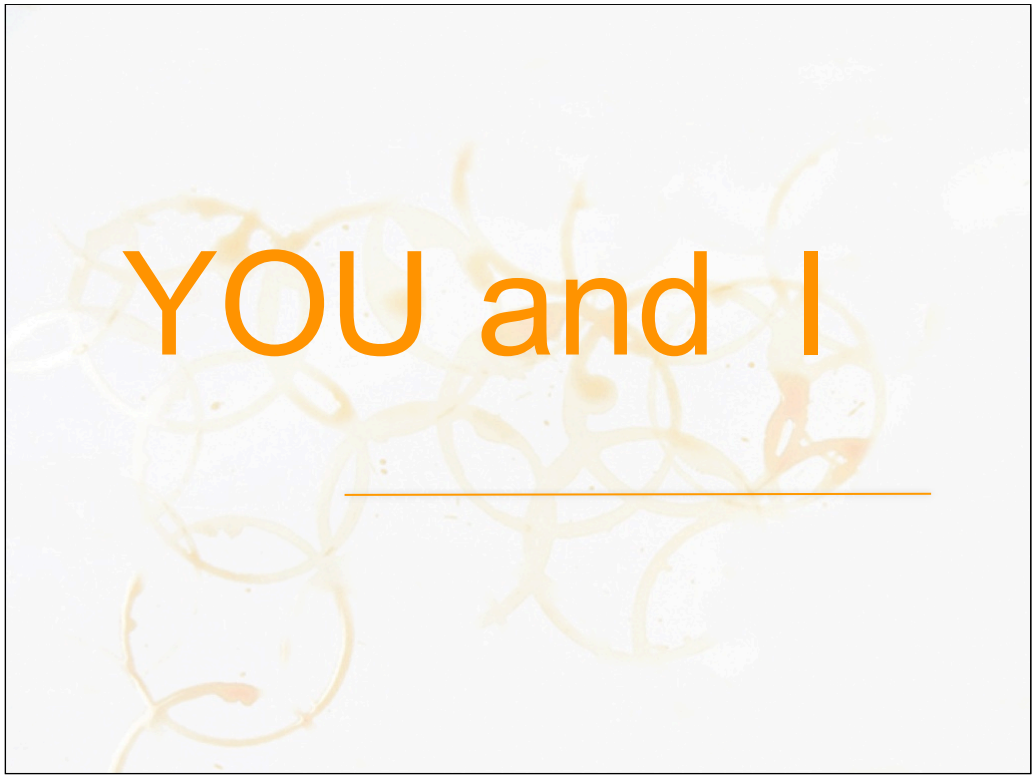
- *As Michael Abrash might have said:*
  - *"Premature optimization is the ~~Root~~ ViewRoot of all evil"*
- Minor garbage is irrelevant in most cases
- But if you have GCs at critical points in your application, consider Garbage Zero
  - Example: animations

Tools: hat

- DDMS
- Heap Analysis Tool is used to track down memory leaks
- adb shell dumpsys meminfo <process>

Memory leaks

- Be careful with Context
- Be careful with static fields
- Avoid non-static inner classes
- Use weak references


YOU and I

Responsiveness

- Single-threaded UI
- Don’t block the UI thread
  - Also called main thread
- AsyncTask
  - Worker thread and UI thread messaging
- Handler
  - Messaging

Overinvalidating

- Only redraw what you must
- (demo)

Fewer is better

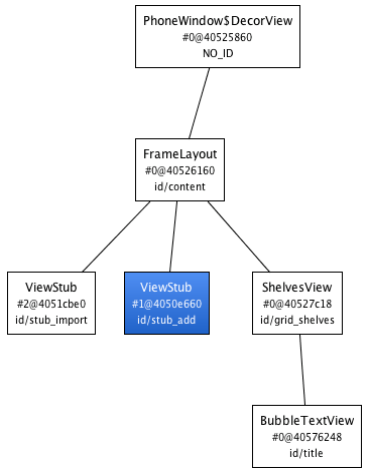
- Many views
  - Slower layout
  - Slower drawing
  - Slower startup time
- Deep hierarchies
  - Memory
  - Slow...
  - StackOverflowException

HiearchyViewer

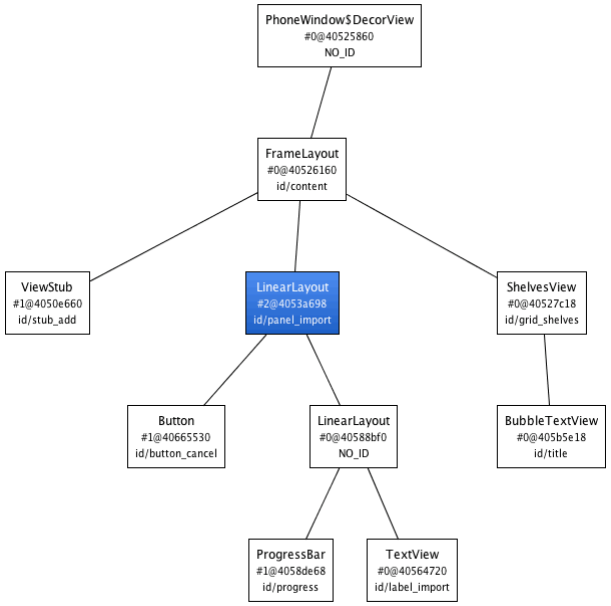
Layout optimizations

- Custom views
- Custom layouts
- <merge />
- ViewStub
- Compound drawables
- layoutopt

ViewStub


PhoneWindow$DecorView
#0@40525860
NO_ID
FrameLayout
#0@40526160
id/content
ViewStub
#2@4051cbe0
id/stub_import
ViewStub
#1@4050e660
id/stub_add
ShelvesView
#0@40527c18
id/grid_shelves
BubbleTextView
#0@40576248
id/title

ViewStub


PhoneWindow$DecorView
#0@40525860
NO_ID
FrameLayout
#0@40526160
id/content
ViewStub
#1@4050e660
id/stub_add
LinearLayout
#2@4053a698
id/panel_import
ShelvesView
#0@40527c18
id/grid_shelves
Button
#1@40665530
id/button_cancel
LinearLayout
#0@40588bf0
NO_ID
BubbleTextView
#0@405b5e18
id/title
ProgressBar
#1@4058de68
id/progress
TextView
#0@40564720
id/label_import

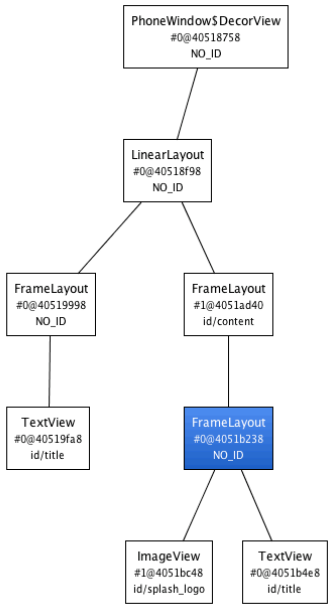
ViewStub

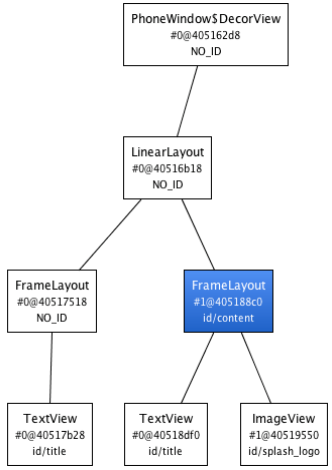
```
<ViewStub
    android:id="@+id/stub_import"
    android:inflatedId="@+id/panel_import"

    android:layout="@layout/progress_overlay"

    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom" />
```

ViewStub

```
findViewById(R.id.stub_import).setVisibility(View.VISIBLE);
// or
View importPanel = ((ViewStub)
        findViewById(R.id.stub_import)).inflate();
```

<merge/>


PhoneWindow$DecorView
#0@40518758
NO_ID
LinearLayout
#0@40518f98
NO_ID
FrameLayout
#0@40519998
NO_ID
FrameLayout
#1@4051ad40
id/content
TextView
#0@40519fa8
id/title
FrameLayout
#0@4051b238
NO_ID
ImageView
#1@4051bc48
id/splash_logo
TextView
#0@4051b4e8
id/title


PhoneWindow$DecorView
#0@405162d8
NO_ID
LinearLayout
#0@40516b18
NO_ID
FrameLayout
#0@40517518
NO_ID
FrameLayout
#1@405188c0
id/content
TextView
#0@40517b28
id/title
TextView
#0@40518df0
id/title
ImageView
#1@40519550
id/splash_logo

<merge/>

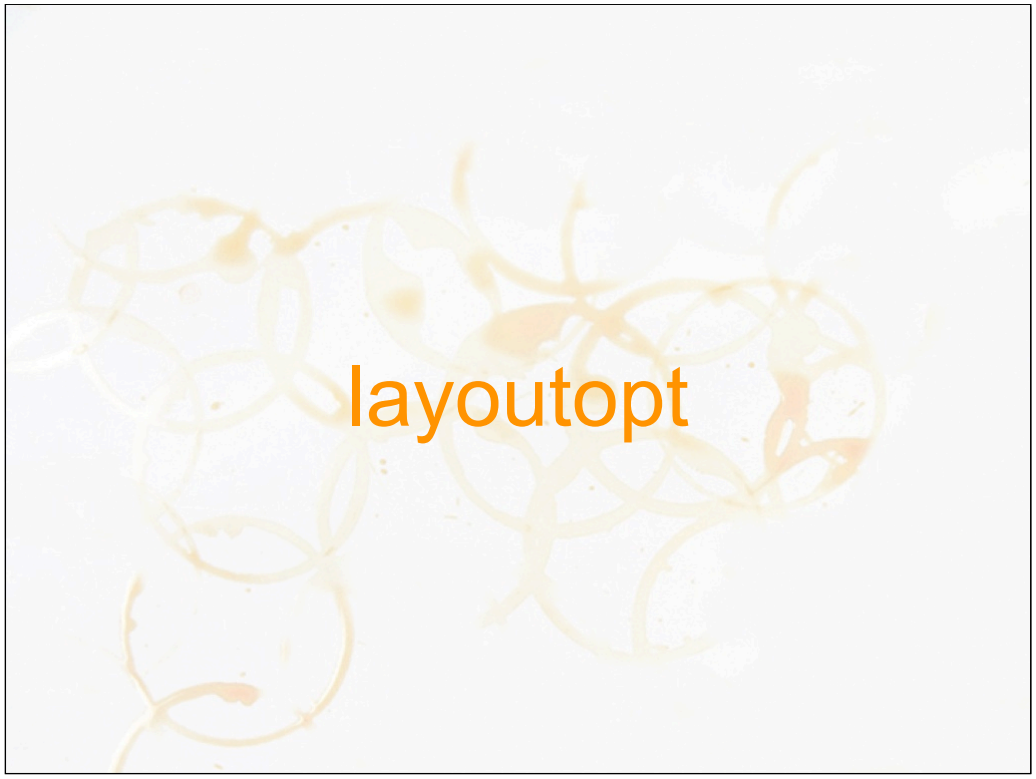
```
<!-- The merge tag must be the root tag -->
<merge xmlns:android="http://schemas.android.com/apk/res/android">
    <!-- Content -->
</merge>
```

Compound drawables

```
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content">

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/icon" />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/hello" />

</LinearLayout>
```

Compound drawables

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    android:drawableLeft="@drawable/icon" />
```


layoutopt

ListView

```
1 public View getView(int position, View convertView, ViewGroup parent) {
2     View item = mInflater.inflate(R.layout.list_item_icon_text, null);

3     ((TextView) item.findViewById(R.id.text)).setText(DATA[position]);
4     ((ImageView) item.findViewById(R.id.icon)).setImageBitmap(
5             (position & 1) == 1 ? mIcon1 : mIcon2);

6     return item;
7 }
```

ListView

```
1 public View getView(int position, View convertView, ViewGroup parent) {
2     if (convertView == null) {
3         convertView = mInflater.inflate(R.layout.item, parent, false);
4     }

5     ((TextView) convertView.findViewById(R.id.text)).setText(DATA[position]);
6     ((ImageView) convertView.findViewById(R.id.icon)).setImageBitmap(
7             (position & 1) == 1 ? mIcon1 : mIcon2);

8     return convertView;
9 }
```

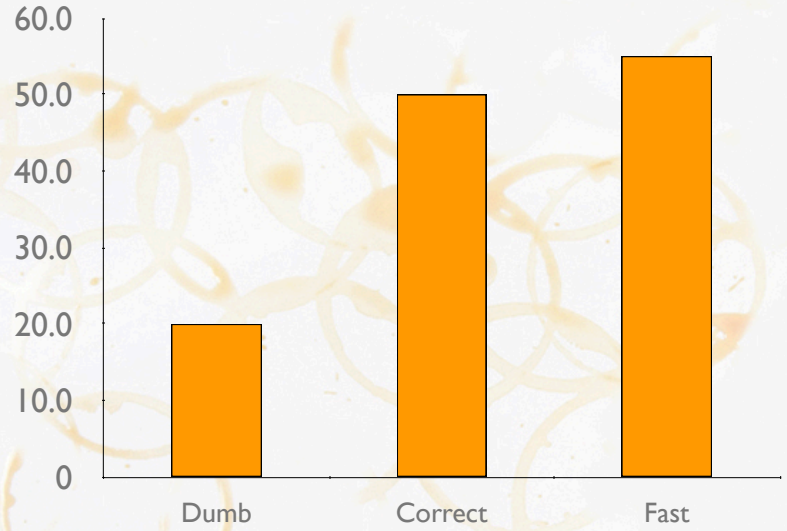
ListView

```
static class ViewHolder {
    TextView text;
    ImageView icon;
}
```

ListView

```
1  public View getView(int position, View convertView, ViewGroup parent) {
2      ViewHolder holder;
3
4      if (convertView == null) {
5          convertView = mInflater.inflate(R.layout.list_item_icon_text,
6                  parent, false);
7          holder = new ViewHolder();
8          holder.text = (TextView) convertView.findViewById(R.id.text);
9          holder.icon = (ImageView) convertView.findViewById(R.id.icon);
10
11         convertView.setTag(holder);
12     } else {
13         holder = (ViewHolder) convertView.getTag();
14     }
15
16     holder.text.setText(DATA[position]);
17     holder.icon.setImageBitmap((position & 1) ==? mIcon1 : mIcon2);
18
19     return convertView;
20 }
```

ListView



List of 10,000 items on NexusOne, Android 2.2

Graphics optimizations

- Pre-scale bitmaps
- Use compatible bitmaps
  - ARGB_8888 to draw on 32 bits window
- Avoid blending
- Use View drawing caches
  - View.setDrawingCacheEnabled(true)
- View.isOpaque()

For More Information

- Android developer site
  - –developer.android.com
- Romain
  - –@romainguy
  - –curious-creature.org
- Chet
  - –@chethaase
  - –graphics-geek.blogspot.com