Best Practice for Andorid Development

KYU CHO

Best Practice for Performance

- **Two Basic Rules**
- Don’t do work that you don’t need to do.
- Don’t allocate memory if you can avoid it.

Avoid Creating Unnecessary Objects

- Object Creation is Never Free

- **Solution**
- Use direct implementation.
- Use substring of the original data for returning value.
- Use array of ints over array of Integer objects.
  - Ex) Two parallel Foo[] and Bar[] > Single array of custome (Foo, Bar) object.

Prefer Static Over Virtual

- Use Static Method
  - Static method invokes about 15% - 20% faster.
- Use Static Final For Constatnts
  - <clinit>
  - No <clinit>

```
static int intVal = 42;
static String strVal = "Hello, world!";
```

```
static final int intVal = 42;
static final String strVal = "Hello, world!";
```

Avoid Internal Getters/Setters

- Use them in the public interface only, and avoid to use them within a class.
  - Virtual mehod calls are much more expensive than instance filed look ups.
- Use JIT (Just In Time, Android 2.2 or above)
  - Without JIT direct filed access is 3x faster
  - With JIT 7x faster than invoking getter function
- Use ProGuard
  - It can inline accessors
  - http://developer.android.com/tools/help/proguard.html

Use Enhanced For Loop Syntax

```
static class Foo {
    int mSplat;
}

Foo[] mArray = ...
```

```
public void one() {
    int sum = 0;
    Foo[] localArray = mArray;
    int len = localArray.length;

    for (int i = 0; i < len; ++i) {
        sum += localArray[i].mSplat;
    }
}
```

```
public void zero() {
    int sum = 0;
    for (int i = 0; i < mArray.length; ++i) {
        sum += mArray[i].mSplat;
    }
}
```

```
public void two() {
    int sum = 0;
    for (Foo a : mArray) {
        sum += a.mSplat;
    }
}
```

Usefull Tips

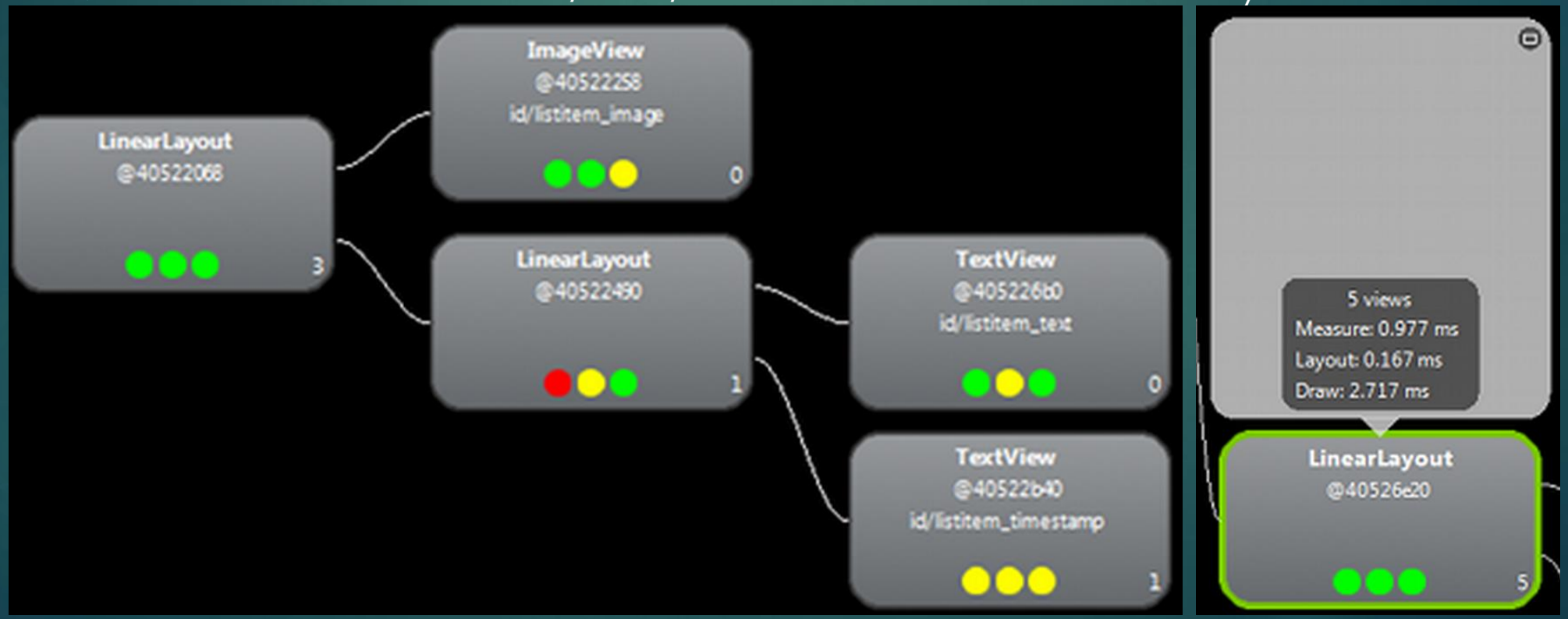
- Avoid Using Floating-Point
  - Float is 2x slower than int on Android-powered devides.
- Use the Libraries
  - Ex) System.arraycopy() is 9x faster than a hand-coded loop
  - Ex) String.indexOf()
- Always Measure with tools
  - Caliper
    - https://code.google.com/p/caliper/
  - Tracview
    - http://developer.android.com/tools/debugging/debugging-tracing.html

Improving Layout Performance

- Optimizing Layout Hierachies
- Re-using Layouts with <include/>
- Call On Demand Layout with <ViewStub/>

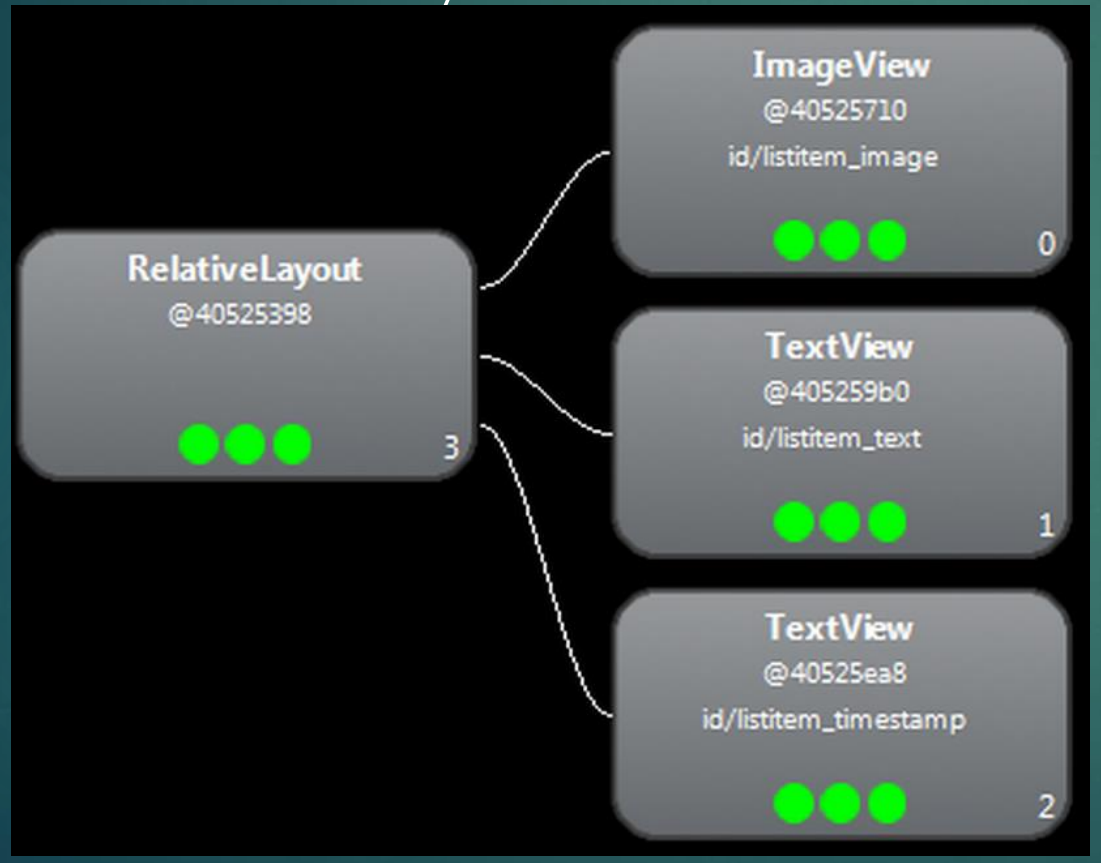
Optimizing Layout Hierachies

- Inspect Your Layout using Hierachy Viewer
  - tool is available in <sdk>/tools/. Then click Load View Hierachy to view.



Revise Your Layout

- User flatter layout, that shallow and wide, rather than narrow and



Now rendering a list item takes:

- Measure: 0.598ms
- Layout: 0.110ms
- Draw: 2.146ms

Re-using Layouts with <include/> and <merge/>

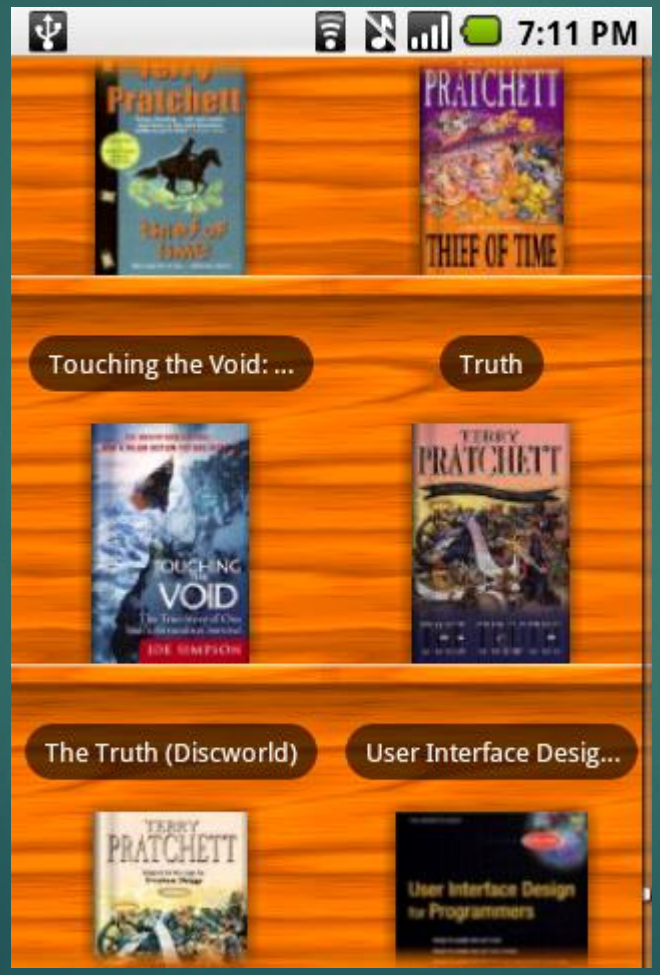
- Attaching titlebar.xml

```
<LinearLayout xmlns:android="http://schemas.android.
    android:orientation="vertical"
    android:layout_width=”match_parent”
    android:layout_height=”match_parent”
    android:background="@color/app_bg"
    android:gravity="center_horizontal">

    <include layout="@layout/titlebar"/>

    <TextView android:layout_width=”match_parent”
              android:layout_height="wrap_content"
              android:text="@string/hello"
              android:padding="10dp" />
</LinearLayout>
```

```
<merge xmlns:android="http://schemas.android.com/apk/res/an

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/add"/>

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/delete"/>

</merge>
```

Call On Demand Layout with <ViewStub/>

It is used to load only when you actually use or need it.



Define and Load a 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();
```

More Best Practice

- Best Practice For
  - Interaction & Engagement
  - User Interface
  - User Imput
  - Background jobs
  - **Performance**
  - Security & Privacy
  - Testing

- http://developer.android.com/training/best-ux.html

Resources

- http://developer.android.com/training/articles/perf-tips.html
- https://code.google.com/p/caliper/
- http://developer.android.com/tools/debugging/debugging-tracing.html
- http://developer.android.com/training/improving-layouts/optimizing-layout.html
- http://developer.android.com/training/improving-layouts/reusing-layouts.html
- http://developer.android.com/training/improving-layouts/loading-ondemand.html