Android Tools for Eclipse

EclipseDay at Googleplex 2010
August 26th, 2010

Xavier Ducrohet - Google Inc.

Google™

Android plug-ins for Eclipse

- Provide familiar workflow
  - build
  - launch configuration
  - debug

- Hide the android-specific soup
  - Android build is non standard
  - Debugger setup rely on other tools
  - Deal with the external tools

Google™

Current State

ADT 0.9.7 is public (05/2010)

- SDK Tools, revision 6

Next: ADT 0.9.8

- SDK Tools, revision 7
- Entirely developed in the open
- In stabilization (tools_r7 branch)
- Released "soon"

Working on next version in master branch

- SDK tools, revision 8
- ADT ?.?.?

Google™

Android Projects

- 3 types of projects
  - regular app project
    - code + resources
    - generate APK
  - Test Project
    - Generate test APK
  - Library project
    - code + resources
    - used by regular app projects

Google™

Android Library Projects

- Goal is reusing code and resources
  - library code accesses library resources.
- Typical use cases
  - Free and paid versions
  - Custom widget reused by 2+ apps
- Problem
  - Resources are accessed using integer IDs.
  - IDs are generated from resources (R.java)
  - Only one pool of resources per application
  - Library Projects should work on earlier versions of Android

Google™

Building applications with Libraries

- aapt generates R.java from all resources
  - app project + libraries
  - no namespace => specific order for overriding values
- New problem
  - Java constants are inlined.
  - => Can't compile library code before generating final R.java
- Solution
  - compile library resources in the application project
  - Limitation: Library projects are source only.

Google™

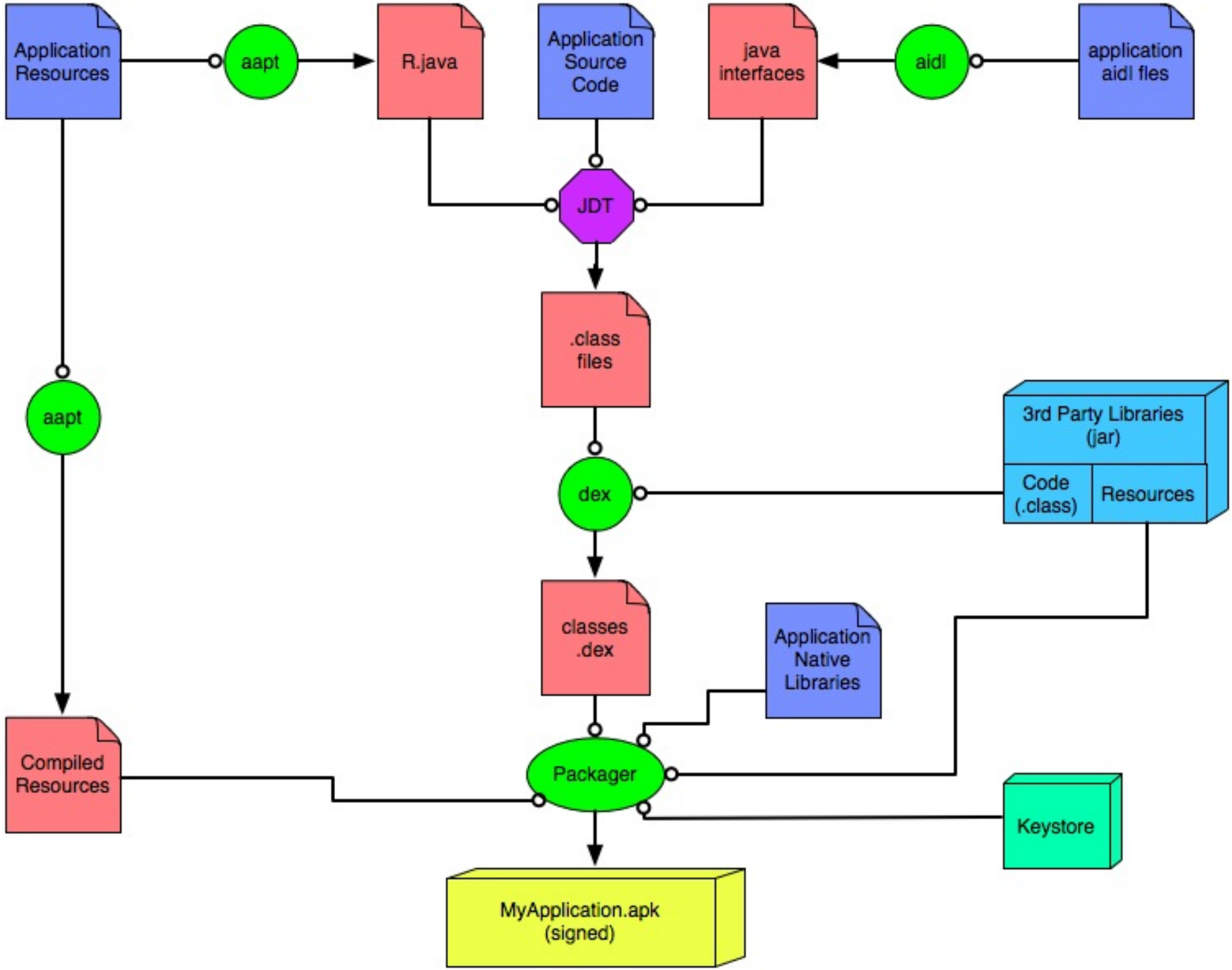
Android Library Projects

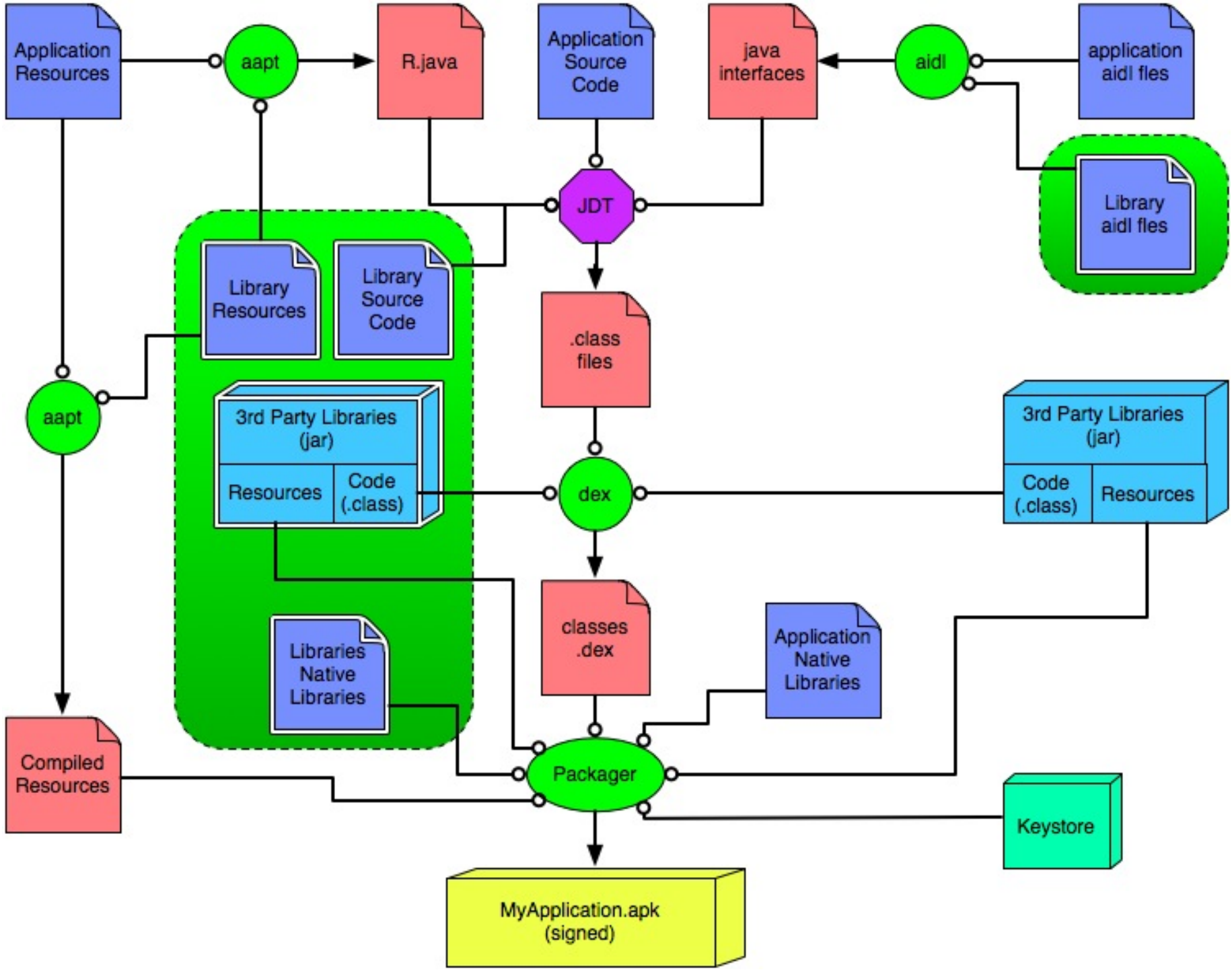
- Contains
  - Manifest (unused for now)
  - Source code
  - Resources

- Build steps of the Library Project
  - Compile resources
  - Compile aidl
  - Compile Java Code

- Actually, no usable output
- Compilation done for verification only.
- Application Manifest must manually declare what the library (ies) declares.

Google™


Application Resources
aapt
R.java
Application Source Code
java interfaces
aidl
application aidl fles
JDT
.class files
aapt
3rd Party Libraries (jar)
Code (.class)
Resources
dex
classes .dex
Application Native Libraries
Compiled Resources
Packager
Keystore
MyApplication.apk (signed)


Application Resources
aapt
R.java
Application Source Code
java interfaces
aidl
application aidl fles
Library aidl fles
JDT
Library Resources
Library Source Code
.class files
aapt
3rd Party Libraries (jar)
Resources
Code (.class)
dex
3rd Party Libraries (jar)
Code (.class)
Resources
classes .dex
Application Native Libraries
Libraries Native Libraries
Compiled Resources
Packager
Keystore
MyApplication.apk (signed)

Android Application Projects

application: com.example.myapp
library: com.example.mylibrary

Build process:

- compile resources and generate 2 identical R.java
  - com.example.myapp.R
  - com.example.mylibrary.R

- compile library code and app code.
  - linked source folder.
  - use *gen* folder of the main application

Google™

Library Projects Setup

All configuration done in default.properties

- platform version target
- isLibrary flag
- library dependencies

No Eclipse project references in the build path.

Why? Single configuration file compatible with Ant.

- Mix Eclipse / Ant users in a Team.
- Develop with Eclipse, automated build release with Ant.

Google™

Library Projects

Demo

Google™

Library Projects

Introduced in r6 / ADT 0.9.7

New in r7 / ADT 0.9.8

- Library project with library dependencies
- Fix a lot of issues related to linked folder in Eclipse
  - much more robust
  - support for any name in the source folders
  - support for 2+ source folders

What's next?

- Manifest merge
- Fix issue related to custom view attributes
- Issue with duplicate error/warning, refresh, ...

Google™

Editing Android files

- Java is handled by JDT
- XML files
  - Android Manifest
  - Values (strings, colors, ...)
  - Layouts
  - Menu definition
  - Settings definition

- 9-patch bitmaps
  - not yet integrated into ADT

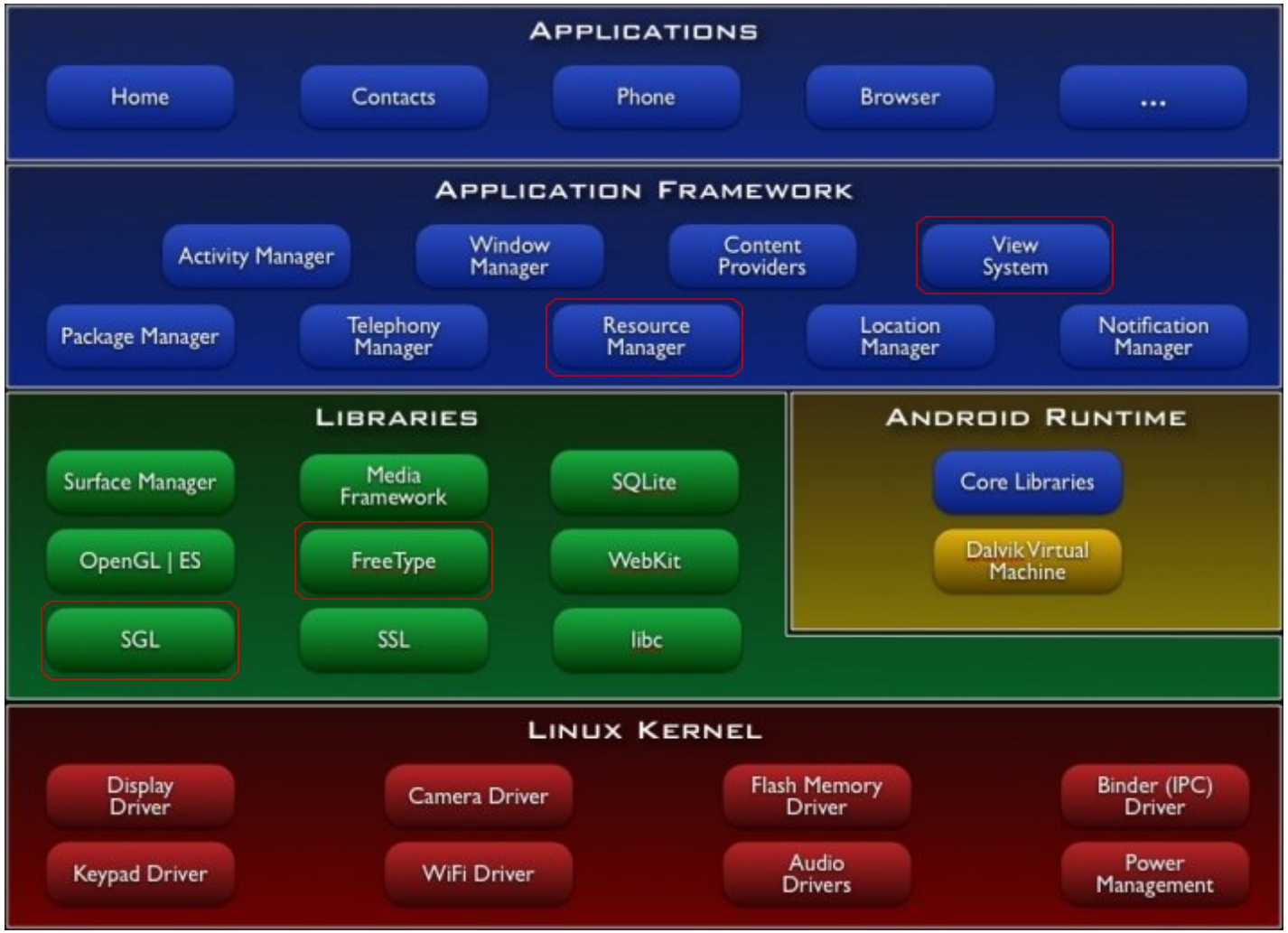
Google™

Layout Editor

- Lots of challenges
  - Rendering fidelity
  - Complex user interactivity
  - UI for a lot of attributes

Google™

Layout Rendering: Architecture



Layout Rendering: *layoutlib*

- Library bundled with the SDK
  - One per platform version (1.5, 1.6, 2.1, 2.2, ...)
  - 100% Java
  - Loaded dynamically by ADT
  - Stateless

- Android View System
- 2D Drawing API reimplemented on top of Java2D

- Resource Manager API used by View System
  - Query Resources
  - Resolve Theme/reference
- Resources parsed by ADT

Google™

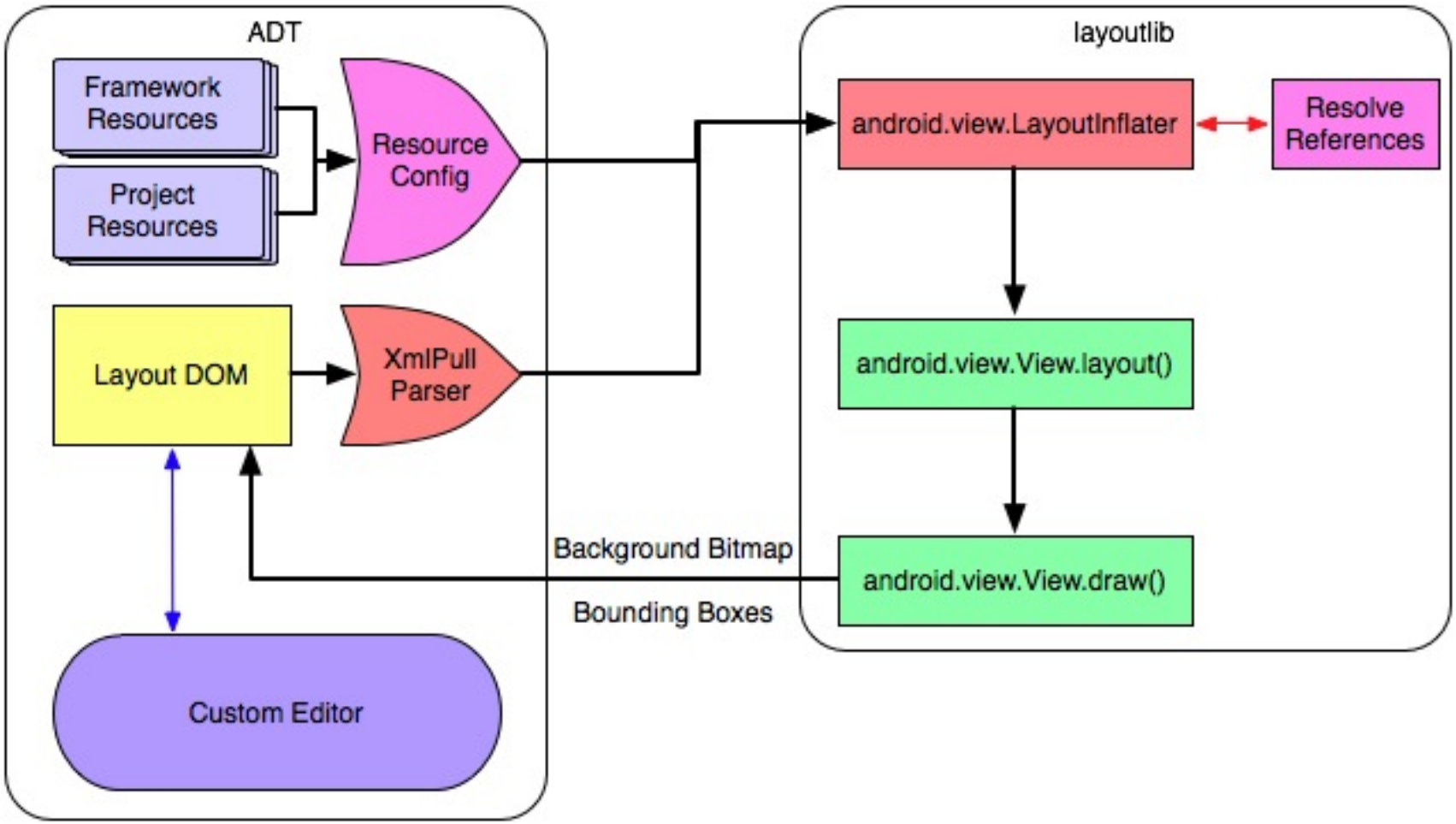
Layoutlib challenges

- Catching up
  - support for all drawing modes
    - added gradients in 2.1
    - a lot more to do
    - back port to previous versions?

- Keeping up with frameworks changes
  - new dependencies not part of layoutlib
  - new options that changes the interactivity

Google™

Layout Rendering: *layoutlib*



Google™

Layout Editor - Custom Views

- Support with some limitations
  - Constructor, onDraw
  - Don't access platform-only API
    - database
    - sensor
    - start thread, ...
- View.isInEditMode

Google™

Layout Editor

- Current Public version
  - Rendering
  - Property View for attributes
  - *Very* basic drag and drop (palette => layout)

- Next version
  - New canvas / interactivity
  - Full drag and drop support
    - palette to layout
    - layout to layout
  - base framework done
    - need to work on visual clue
    - tune interactivity
  - Developed in parallel with current version

Google™

Layout Editor

Demo

Google™

Layout Editor

Longer term:

- new palette
- new properties pane
- new resource selector
- layout-opt integration

Incremental changes directly in each ADT release.

Google™

Testing

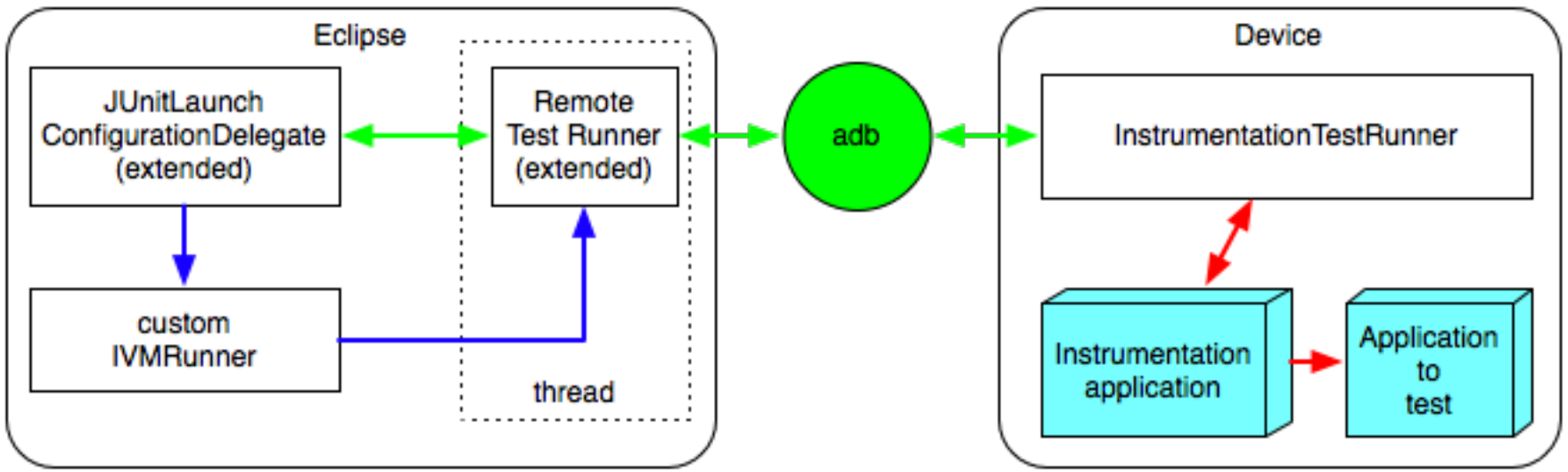
- *android.jar* has no code
  - Cannot run tests on the desktop JVM
- Android Instrumentation Framework
  - Runs JUnit tests on the device
  - Basic command line:

```
adb shell am instrument ...
```

  - Text output
- Integration into Eclipse

Google™

Running JUnit Tests



Google™

Running JUnit Tests

Demo

Google™

Profiling

- TraceView
  - Standalone Tool (SWT) to see traces
- hprof files
  - Non standard, but converter available

Both now controlled through DDMS and integrated in ADT.

Google™

Profiling Layout

HierarchyViewer

- Connect to device to get runtime layout.
- Display layout tree.
- Pixel perfect display, loupe.

Originally wrote in Swing.

New version being rewritten in SWT with planned integration in Eclipse as a plug-in.

Google™

The Future is open

SDK Tools, rev 7 / ADT 0.9.8

- developed in the open.
- No internal changes (mirror only).
- Builds coming from the *tools_r7* branch (announced on android-contrib).

SDK Tools, rev 8 / ADT ?.?.? development ongoing in the open in *master* branch.

Google™

External contributions

Already accepting contributions, but

- They come out of nowhere.
- They don't follow style.
- We don't know about the feature before the full patch is uploaded.

What needs to happen?

- publish roadmap / todo list.
- publish clear guidelines on contributing to the tools.
- use the external bug tracker exclusively.
- use android-contrib for all discussions.

Google™

Useful Links

- http://developer.android.com
  - SDK / ADT download
  - Dev Guide, API reference
  - android-developers mailing list
  - http://stackoverflow.com/
- http://source.android.com
  - Android source code
  - Dev Tools source code
  - android-contrib mailing list.

Google™