Developing for Android
with
Eclipse

EclipseDay at Googleplex 2009
August 27th, 2009

Xavier Ducrohet - Google Inc.
Why custom plug-ins?

- Android build is complex
- Setting up debugger is non-trivial
- Lots of external tools

Goals of the plug-ins?

- Provide familiar work flow
- Hide all the android-specific stuff
Build Process

Application Resources → aapt → R.java → Application Source Code → aidl → aidl files

JDT → .class files

aapt → .dex files → dex → classes .dex

Compiled Resources → apkbuilder → MyApplication.apk (signed)

3rd Party Libraries (.jar) → Code (.class) → Resources → Keystore
Android Projects

- Project natures
  - Java (JDT)
  - Android

- Two IncrementalProjectBuilder
  - PreCompiler
    - Resources -> R.java (aapt)
    - Compile aidl files (aidl)
  - Package builder
    - Convert to Dalvik bytecode (dx.jar)
    - Compile resources into binary XMLs (aapt)
    - Package everything into APK
    - Sign with debug key
Plugin - Device communication

DDMS

<table>
<thead>
<tr>
<th>Name</th>
<th>Online</th>
</tr>
</thead>
<tbody>
<tr>
<td>SERIALNUMBER</td>
<td>Online</td>
</tr>
<tr>
<td>emulator-tcp-5555</td>
<td>Online</td>
</tr>
<tr>
<td>system_process</td>
<td>514</td>
</tr>
<tr>
<td>com.google.process.content</td>
<td>559</td>
</tr>
<tr>
<td>com.google.android.home</td>
<td>575</td>
</tr>
<tr>
<td>com.google.android.phone</td>
<td>577</td>
</tr>
</tbody>
</table>
Debugging on Devices

- Secure device => Cannot debug any app
- debuggable = true in manifest to enable debugger
- Don't ship with this!
DDMS

- Handle connection to devices through adb
- Basic tools
  - device/app list
  - logcat
  - heap / thread views
  - emulator control
  - Screen capture
  - File Explorer
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
XML Editors

- "Advanced"
  - Form based
  - WYSIWYG
- Text Editor
  - Default XML text editor
  - Custom content assist
- Resource Manager
  - Load each project resources
  - customize editors with project content (content assist)
  - Resource Explorer
- Refactoring
  - Extract Strings
Layout Editor

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

- Current version
  - Rendering
  - Property View for attributes
  - Very basic drag and drop
Layout Rendering: Architecture

Applications:
- Home
- Contacts
- Phone
- Browser
- ...

Application Framework:
- Activity Manager
- Window Manager
- Content Providers
- View System
- Package Manager
- Telephony Manager
- Resource Manager
- Location Manager
- Notification Manager

Libraries:
- Surface Manager
- Media Framework
- SQLite
- OpenGL | ES
- FreeType
- WebKit
- SGL
- SSL
- libc

Android Runtime:
- Core Libraries
- Dalvik Virtual Machine

Linux Kernel:
- Display Driver
- Camera Driver
- Flash Memory Driver
- Binder (IPC) Driver
- Keypad Driver
- WiFi Driver
- Audio Drivers
- Power Management
Layout Rendering: layoutlib

- Library bundled with the SDK
  - 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
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
Running JUnit Tests
Profiling

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

- [http://developer.android.com](http://developer.android.com)
  - SDK / ADT download
  - Dev Guide, API reference
  - Developer mailing lists

- [http://source.android.com](http://source.android.com)
  - Android source code
  - Dev Tools source code
  - Platform mailing lists