

Rationalizing Android Development

Philipp Kumar



Who am I?



Philipp Kumar

- **akquinet tech@spree GmbH**
- **Mobile Solutions**
- Focus: **Android** and its **Enterprise Integration**

Who are we?

UI Design

OSGi

Trainings

Android

JBoss

Development

Java / JEE

Consulting

akquinet



Modular Systems

Web Portals

User Experience

Open Source

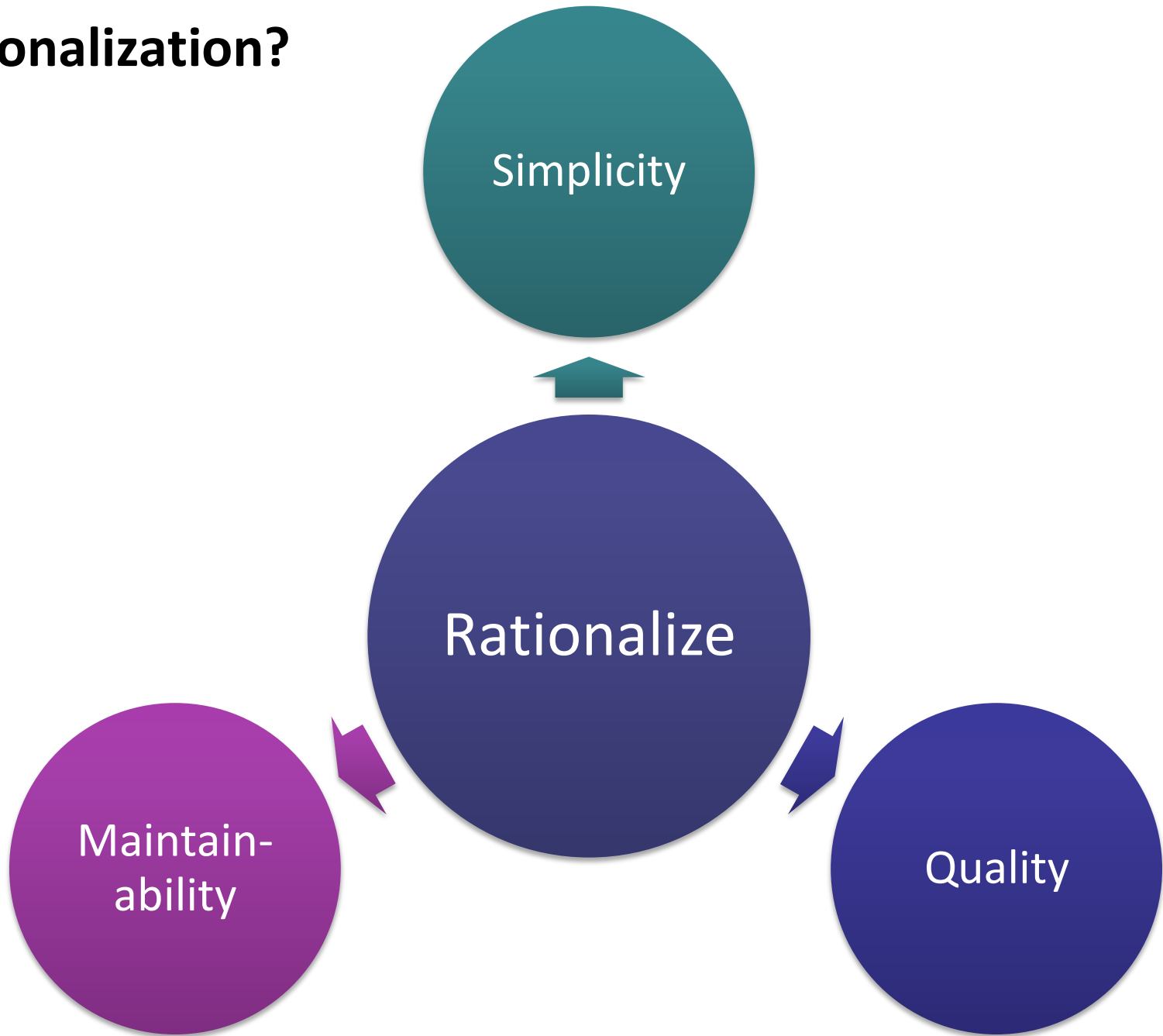
Integration

Mobile Systems

Who are you?



Rationalization?



DEMO

Android Development

What we have...

- **Android SDK**
 - Compiler, Packager, Emulator, ...
- **Eclipse Plugin**
 - Android IDE

Now imagine...



... more complexity.

- **One** screen -> **Multitude** of screens
- **Offline** app -> **Server** interaction
- **Java only** -> Java plus **native** C/C++ components
- **Foreground** only -> **Background** tasks/services
- **10** Lines of Code -> **>10000** Lines of Code
- ...

DEMO

A real-world app

Again, what we have...

- **Android SDK**
 - Compiler, Packager, Emulator, ...
- **Eclipse Plugin**
 - Android IDE

Again, what we have...

- **Android SDK**
 - Compiler, Packager, Emulator, ...
- **Eclipse Plugin**
 - Android IDE

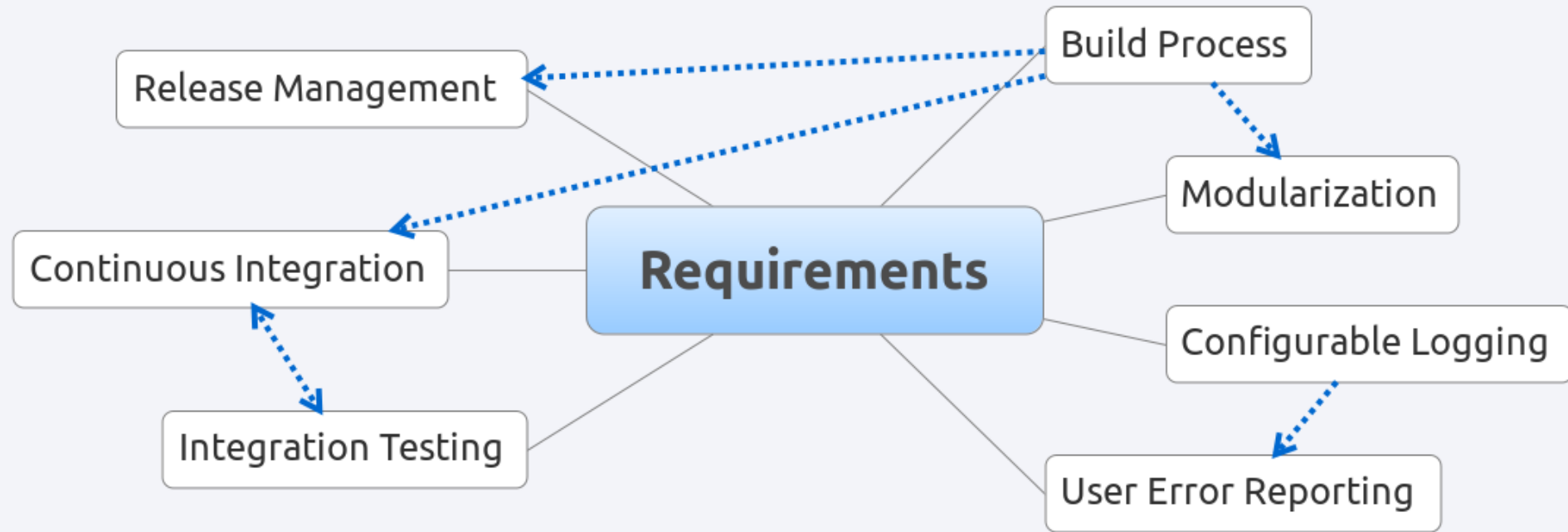
insufficient...

Again, what we have...

- **Android SDK**
 - Compiler, Packager, Emulator, ...
- **Eclipse Plugin**
 - Android IDE

insufficient...
why?

- **Correctness**
 - testable, consistent, market-ready
- **Portability**
 - address Android fragmentation
- **Maintainability**
 - **simple ↔ comprehensible → changeable/extensible**

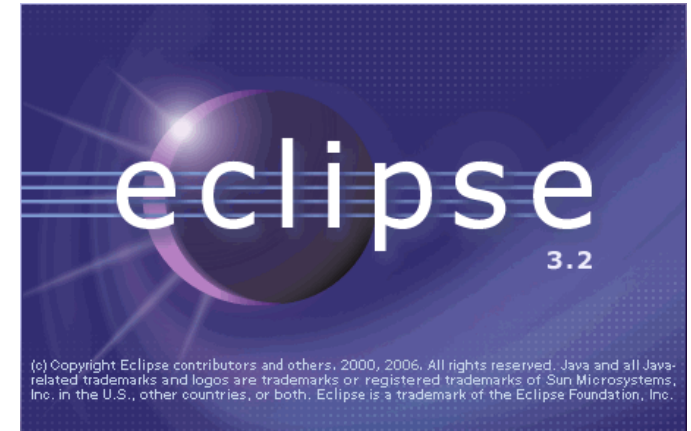


Build Process

- Easy

but ...

- **Platform-dependent**
 - Non-reproducible
 - Error-prone
- **Cannot be automated**



- Can be automated
- Reproducible



but ...

- **No standardized build process**
- **No dependency management**
- **Maintenance nightmare**

Solution: Use Maven!

Standardized **Descriptive** **Modular**



Apache Maven Project
<http://maven.apache.org/>

Release Process

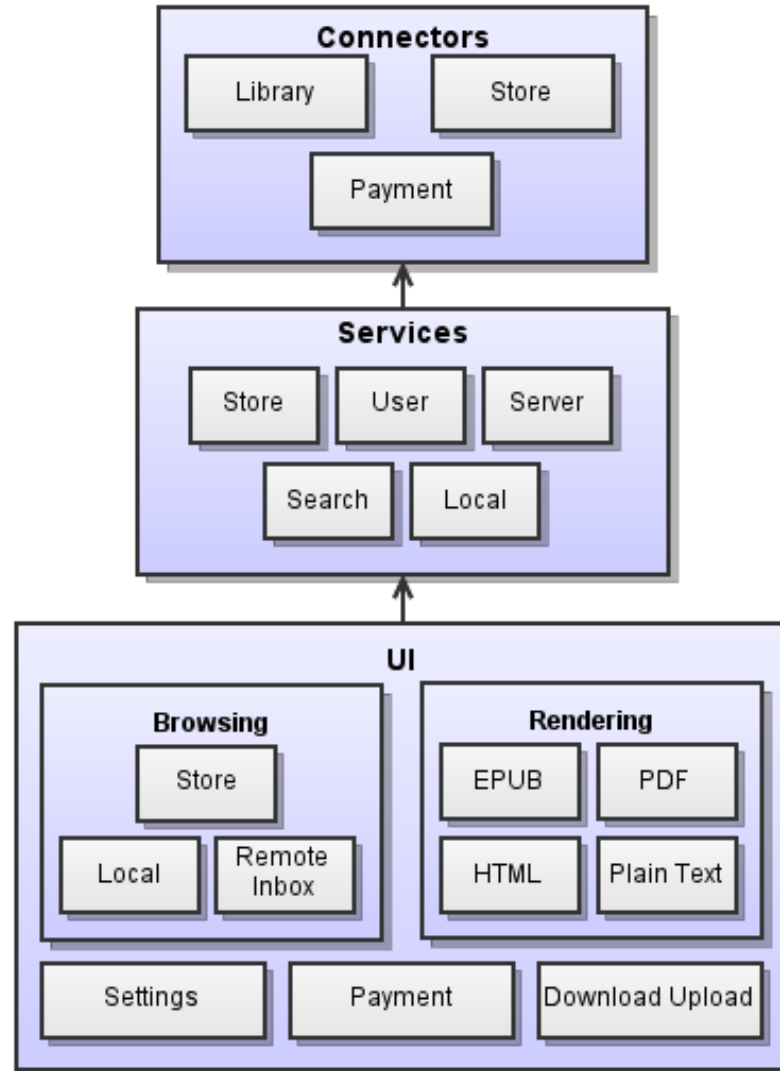
**Dependency
Management**



- **Divide** your Android projects into **smaller** projects
- Subprojects can depend on
 - **Ordinary Java** projects
 - **Android** projects

<http://code.google.com/p/maven-android-plugin>

<https://github.com/akquinet/android-rindirect>



```
phil@janus: ~/projects/txtr/trunk/android
E/WindowManager( 2488): Overwriting rotation value from 0
D/dalvikvm( 2390): GC EXPLICIT freed 52 objects / 2328 bytes in 69ms
D/LocalServiceImpl(13307): onCreate()
I/global (13307): Default buffer size used in BufferedReader constructor. It would be better to be explicit if an 8k-char buffer is required.
D/dalvikvm( 2390): GC EXPLICIT freed 2 objects / 48 bytes in 61ms
I/CONNECTOR(13307): Logger created : com.txtr.android.connector.AndroidLogger@481960a9
I/CONNECTOR(13307): Encryptor created : com.txtr.connector.impl.Sha1Encryptor@48196920
I/global ( 2488): Default buffer size used in BufferedInputStream constructor. It would be better to be explicit if an 8k buffer is required.
I/com.txtr.android.connector.ServerConnector(13307): Connector created and configured com.txtr.connector.impl.TxtrConnector@481959f8
I/LocalServiceImpl(13307): Try to use the SDCard
I/LocalServiceImpl(13307): SDCard OK
I/global (13307): Default buffer size used in BufferedReader constructor. It would be better to be explicit if an 8k-char buffer is required.
I/global (13307): Default buffer size used in BufferedWriter constructor. It would be better to be explicit if an 8k-char buffer is required.
I/global (13307): Default buffer size used in BufferedReader constructor. It would be better to be explicit if an 8k-char buffer is required.
I/global (13307): Default buffer size used in BufferedWriter constructor. It would be better to be explicit if an 8k-char buffer is required.
I/LocalServiceImpl(13307): Device IME 353942040003176
I/LocalServiceImpl(13307): Salt -5760250510723452889
I/LocalServiceImpl(13307): Dev : 353942040003176-5760250510723452889
I/LocalServiceImpl(13307): Finger Print : CVC0kVVDjhZ08HDztpgT
I/LocalServiceImpl(13307): Base64 finger print : 01ZDb1ZrVkrQqFpRQmhEenRwZ10=
I/AdobeUtils(13307): Device.xml already created
I/LocalServiceImpl(13307): Device IME 353942040003176
I/LocalServiceImpl(13307): Salt -5760250510723452889
I/LocalServiceImpl(13307): Dev : 353942040003176-5760250510723452889
I/LocalServiceImpl(13307): Finger Print : CVC0kVVDjhZ08HDztpgT
I/LocalServiceImpl(13307): Base64 finger print : 01ZDb1ZrVkrQqFpRQmhEenRwZ10=
I/AdobeUtils(13307): Device.xml already created
I/global (13307): Default buffer size used in BufferedReader constructor. It would be better to be explicit if an 8k-char buffer is required.
I/LocalServiceImpl(13307): Try to use the SDCard
I/LocalServiceImpl(13307): SDCard OK
I/LocalServiceImpl(13307): Try to find /mnt/sdcard/txtr/at6d89.epub
I/LocalServiceImpl(13307): Has reloaded : Siddhartha
I/global (13307): Default buffer size used in BufferedReader constructor. It would be better to be explicit if an 8k-char buffer is required.
I/LocalServiceImpl(13307): Try to use the SDCard
I/LocalServiceImpl(13307): SDCard OK
I/LocalServiceImpl(13307): Try to find /mnt/sdcard/txtr/ap8dz9.epub
I/LocalServiceImpl(13307): Has reloaded : White Fang
I/LocalServiceImpl(13307): Reloading document from Digital Editions
I/AdobeDocumentsReader(13307): This device has no Adobe manifest. Searched at /mnt/sdcard/Digital Editions/manifest.xml
I/LocalServiceImpl(13307): 0 document reloaded from ADE
I/LocalServiceImpl(13307): Device IME 353942040003176
I/LocalServiceImpl(13307): Salt -5760250510723452889
I/LocalServiceImpl(13307): Dev : 353942040003176-5760250510723452889
I/LocalServiceImpl(13307): Finger Print : CVC0kVVDjhZ08HDztpgT
I/LocalServiceImpl(13307): Base64 finger print : 01ZDb1ZrVkrQqFpRQmhEenRwZ10=
I/AdobeUtils(13307): Device.xml already created
I/LocalServiceImpl(13307): Local Service >> Bind Method
I/ActivityManager( 2488): Displayed activity com.txtr.android/.VanillaIntroActivity: 424 ms (total 377306 ms)
I/Downloader(13307): User Service bound
I/Launcher( 2602): onWindowFocusChanged(false)
I/CheckinService( 2955): Preparing to send checkin request
I/EventLogService( 2955): Accumulating logs since 1295907330519
I/CheckinTask( 2955): Sending checkin request (1452 bytes)
I/com.txtr.android.IntroActivity$1(13307): Forwarding to next activity...
I/ActivityManager( 2488): Starting activity: Intent { cmp=com.txtr.android/.FirstTimeUseActivity (has extras) }
I/CheckinTask( 2955): Checkin success: https://android.clients.google.com/checkin (1 requests sent)
I/CheckinService( 2955): From server: Intent { act=android.server.checkin.FOTA_CANCEL }
V/AlarmManager( 2488): set: Alarm(481f7b68 type 0 com.google.android.gsf)
I/ActivityManager( 2488): Displayed activity com.txtr.android/.FirstTimeUseActivity: 397 ms (total 397 ms)
```

Logging

What Android provides

- Logging mechanism
- Different log levels
- Log level can be set at runtime (per-device)
 - Requires appropriate rights!

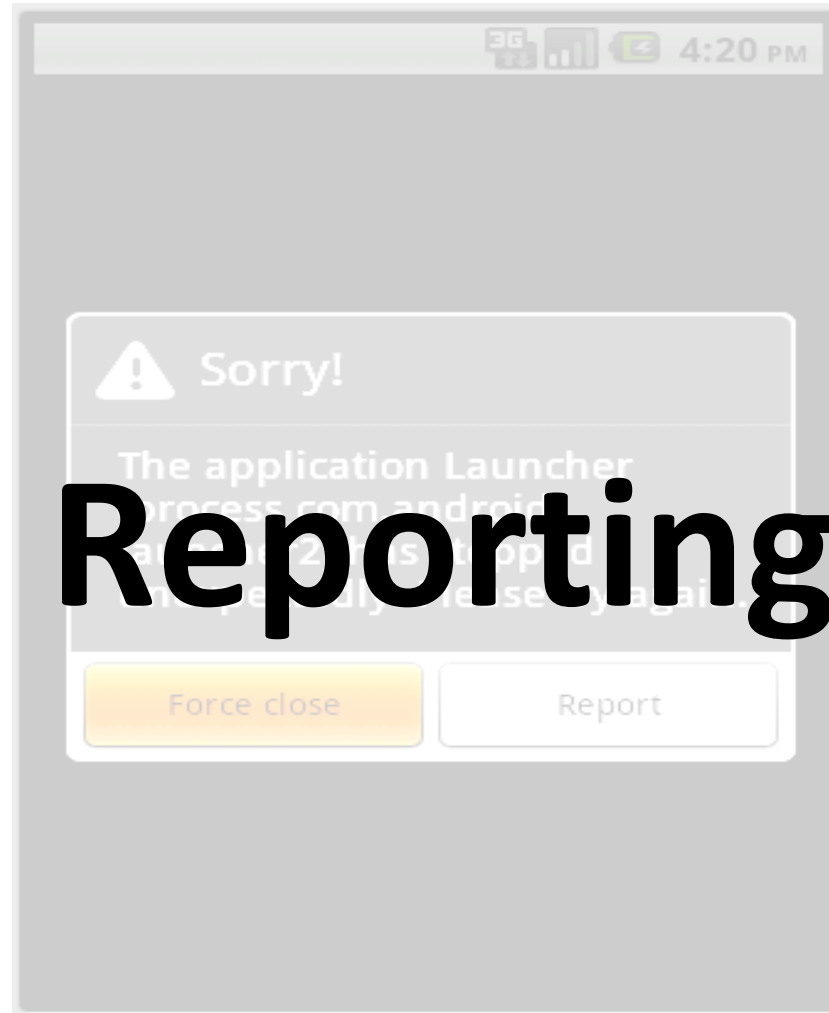
What we needed

- Per default, **do not log anything!**
- Enable/configure logging at runtime **if needed**
- Configure logging **per app**

androlog framework

- Wrapper on top of Android's Log system
- Enable/disable logging
- Configure loglevel
- Open-source

<https://github.com/akquinet/androlog>

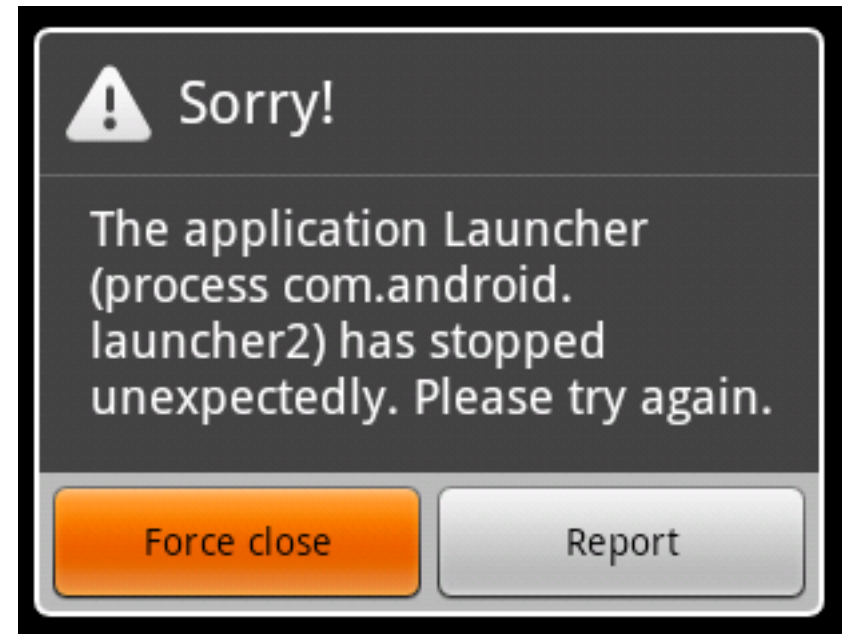


Reporting

What Android 2.2 provides

- „Send report“ prompt on error
- Report contains
 - **Exception** stack trace
 - Custom **user message**

=> **Insufficient**



What we needed

- **Pre-Android-2.2** reporting!
- Report **context** information
- Report **device** information

Androlog... again 😊

- Catch exceptions and send a **more detailed report**
 - containing log messages
 - containing device info
- Send reports via
 - Email
 - HTTP Post
 - ...
 - extendable

<https://github.com/akquinet/androlog>

Integration Testing

Automatically test the whole app on a device or emulator

- Supported in Android via **Instrumentation**
 - Start application under test...
 - ... deliver events...
 - ... and define assertions on resulting behavior.

DEMO

Integration Testing

Marvin: Android Test Framework

- **Control** Activities and Services
- **Define** Assertions
- **Inject** events
 - **Ease writing and maintenance** of tests

<https://github.com/akquinet/android-marvin>

Continuous Integration

Continuous Integration

- **Build** the app on a **dedicated** server
- **Build automatically** after every commit
- **Test** by automatically **starting emulators**
 - ... and run tests **covering all Android versions**
- **Send mails** immediately on build/test failure
- **Build nightly** beta versions

We use

- Hudson/Jenkins + Android plugin

Release Process

TODOs on Release

- Run **tests**
- **Enforce** rules
 - e.g. app is not „debuggable“
- **Zip-align** the APK
- **Sign** APK with market key

Can be automated...

- ... using Maven 😊
- Releasing takes 2 minutes
- repeatable!

Conclusion

Developing apps for Android...

- ... is non-trivial.
- **Maintainability**
 - **Use Maven** to build your app
 - **Modularize** your app
- **Correctness, Portability**
 - Use androlog to get **error reports** from users
 - Write **automated tests**, e.g. with Marvin
 - Set up Hudson to **build/test** your app **continuously**

Current work in progress

- Easy **injection** of services and resources
- Automatically compute **quality metrics**
- Improve **IDE support**
- ...

<http://blog.akquinet.de>

DEMO

Current Work in Progress

