



SUNGARD GLOBAL SERVICES

Development Tools for Large Teams

September 8, 2012

Todd Ricker





A few questions

- » Other than implementing application functionality, where do your developers spend their time?
- » What is the quality of your team's code? How brittle is it?
- » When things break, how long does it take the team to know something is broken?
- » When things break, do people self-mobilize to fix the problem?
- » Can your QA staff do their own builds?
- » Do you have a one click build?



Goals: Agile / Continuous integration / Devops

- » Eliminate manual processes
- » Stringent and continual testing
- » Frequent, low-ceremony releases
- » Cultural shift to shared accountability
- » Transparency



Shared accountability: An example

RE: Build failed in Jenkins: [REDACTED]-integration-test #2519



Sent: Wed 8/29/2012 11:40 AM

To:

That's me. I'll fix it

-----Original Message-----

From: Jenkins CI1 [mailto:jenkins.admin.noreply@sungard.com]

Sent: Wednesday, August 29, 2012 11:39 AM



Solutions

- » Continuous integration
 - Continuously compile, test, and deploy
- » Software quality tools
- » Unit test coverage tools
- » Javadoc coverage tools
- » Peer code reviews
- » Automated functional tests
- » Database change provisioning



Devops

- » The tools and processes tread the line between many roles
 - Development
 - Systems administrations
 - QA / Testing staff
 - Build engineers



Tools: Continuous integration

» What is it?

- Short answer: Build (and test) automation
- The practice of frequently integrating one's new or changed code with the existing code repository – should occur frequently enough that no intervening window remains between commit and build, and such that no errors can arise without developers noticing them and correcting them immediately

Source: Wikipedia.org

http://en.wikipedia.org/wiki/Continuous_integration



Tools: Continuous integration

- » Jenkins (Open Source)
- » Hudson (Open Source)
- » CruiseControl (Open Source)
- » JetBrains TeamCity
- » Atlassian Bamboo
- » And many others...



*Live Jenkins demo
coming up...*



Tools: Software quality – Sonar

- » “Sonar is an open platform to manage code quality”
source: <http://www.sonarsource.org/>
- » The 7 axes of code quality
 - Architecture & design
 - Duplications
 - Unit tests
 - Complexity
 - Potential bugs - <http://findbugs.sourceforge.net/>
 - Code rules - <http://checkstyle.sourceforge.net/>
 - Comments

Tools: Sonar – Project overview page

Home Configuration Log in Search

Version Sprint8 - 17 Aug 2012 03:01 [Time changes...](#)

Dashboard

- Hotspots
- Reviews
- Components
- Violations Drilldown
- Time Machine
- Clouds
- Design
- Libraries

sonar

Total Quality
69.1%
92.3% Architecture
93.7% Design
0.0% Test
90.4% Code

Violations
6,934
Rules compliance
86.2%

Lines of code
122,864
294,003 lines
42,763 statements
1,700 files

Classes
1,845
68 packages
8,693 methods
7,235 accessors

Comments
40.7%
84,398 lines
93.1% docu. API
593 undocu. API

Duplications
5.9%
17,481 lines
552 blocks
195 files

Complexity
2.3 /method
10.8 /class
11.7 /file
Total: 19,889

Methods Classes

Events [All](#)

Date	Type	Message	Details
17 Aug 2012	Version	Sprint8	
18 Apr 2012	Alert	Green (was Orange)	
17 Apr 2012	Profile	Sonar Way with Findbugs version 1	(i)
17 Apr 2012	Alert	Orange	(i)
17 Apr 2012	Profile	Sonar way with Findbugs version 1	(i)
21 Jul 2011	Version	0.1-SNAPSHOT	

Powered by [SonarSource](#) - Open Source [LGPL](#) - v.2.14 - [Plugins](#) - [Documentation](#) - [Ask a question](#)



Tools: Sonar – Project overview page

★ Version Sprint8 - 27 Aug 2012 03:00 [Time changes...](#)

Total Quality
69.2%

92.3% Architecture
93.7% Design
0.0% Test
90.6% Code

Lines of code
123,924 ▲
295,706 lines ▲
43,194 statements ▲
1,700 files ▲

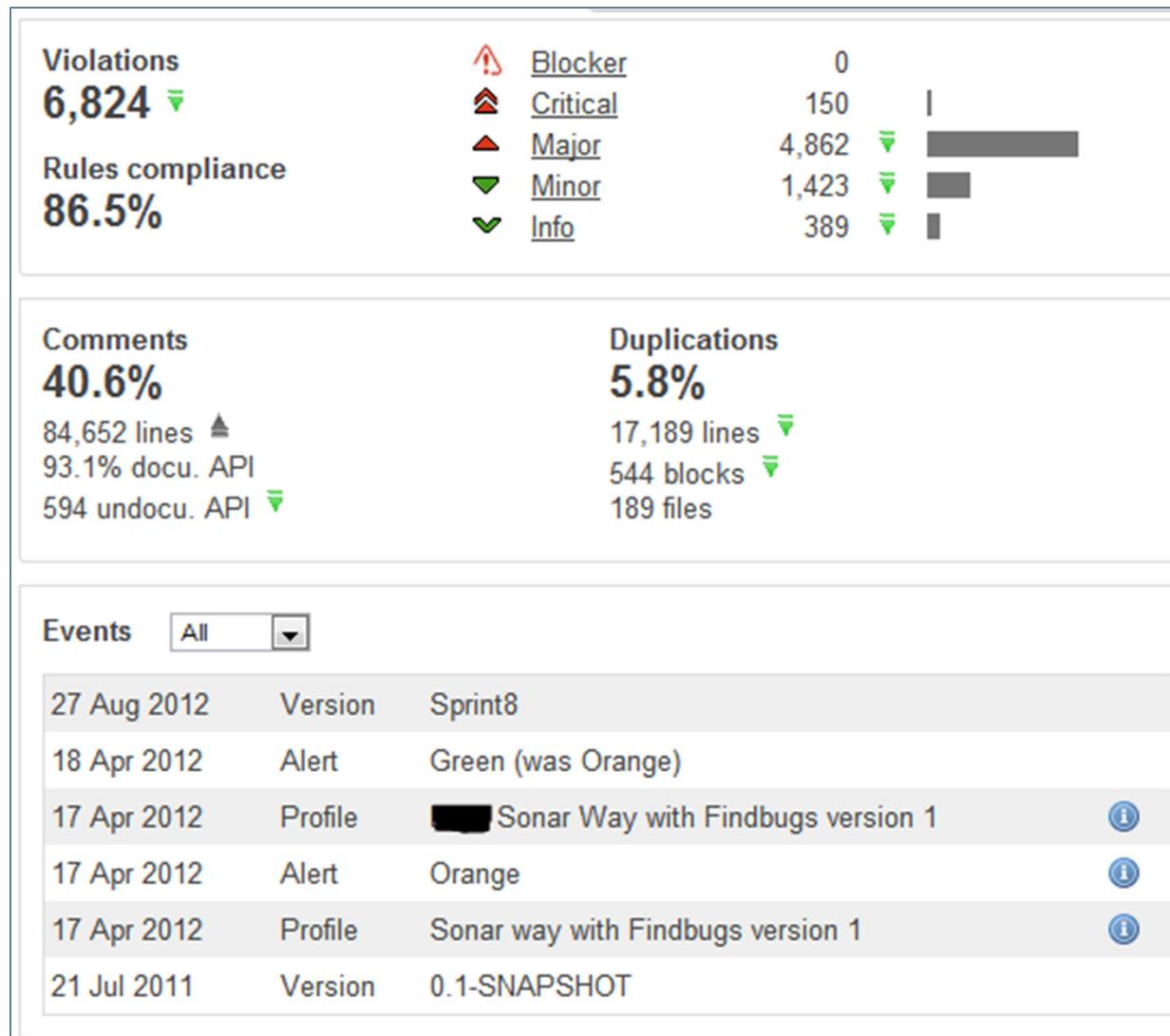
Classes
1,847 ▲
68 packages
8,752 methods ▲
7,259 accessors ▲

Complexity
2.3 /method
10.9 /class
11.8 /file
Total: 20,098 ▲

Methods Classes



Tools: Sonar – Project overview page





Tools: Sonar – 30 Day Delta

Home Configuration Log in Search

Dashboard

Hotspots
Reviews
Components
Violations Drilldown
Time Machine
Clouds
Design
Libraries

sonar

Version Sprint8 - 17 Aug 2012 03:01

Total Quality **69.1%** 92.3% Architecture
93.7% Design
0.0% Test
90.4% Code

Violations **6,934 (-533)**
On new code: **576**
On old code: **-1109**

Rules compliance **86.2% (+1.6)**

Comments **40.7% (-0.1)**
84,398 lines (+3,025)
93.1% docu. API (+0.4)

Duplications **5.9% (-0.4)**
17,481 lines (-225)
552 blocks (-9)
195 files (-3)

Events All

Date	Type	Message	Details
17 Aug 2012	Version	Sprint8	
18 Apr 2012	Alert	Green (was Orange)	
17 Apr 2012	Profile	Sonar Way with Findbugs version 1	
17 Apr 2012	Alert	Orange	
17 Apr 2012	Profile	Sonar way with Findbugs version 1	
21 Jul 2011	Version	0.1-SNAPSHOT	

Lines of code **122,864 (+4,558)**
294,003 lines (+11,269)
42,763 statements (+1,617)
1,700 files (+57)

Classes **1,845 (+65)**
68 packages (+2)
8,693 methods (+436)
7,235 accessors (+229)

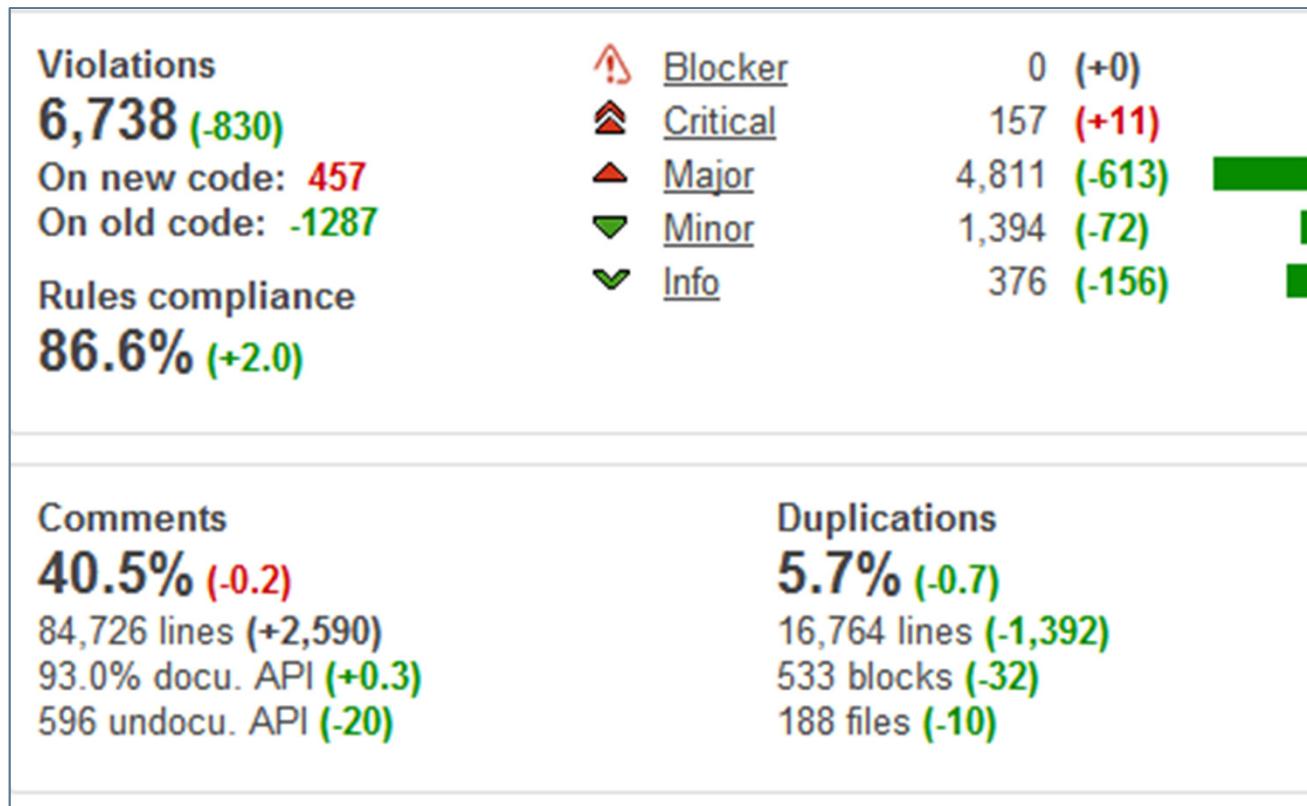
Complexity **2.3 /method(+0.0)**
10.8 /class(+0.0)
11.7 /file(+0.0)
Total: 19,889 (+716)

Key: [REDACTED]
Language: java
Profile: [REDACTED] [Sonar Way with Findbugs \(version 1\)](#)
Alerts: [RSS Feed](#)

Powered by [SonarSource](#) - Open Source [LGPL](#) - v.2.14 - [Plugins](#) - [Documentation](#) - Ask a question



Tools: Sonar – 30 Day Delta



Tools: Sonar – Violations

The screenshot shows the SonarQube interface for a project named "Sonar Way with Findbugs". The left sidebar includes links for Home, Dashboard, Hotspots, Reviews, Components, Violations Drilldown (which is selected), Time Machine, Clouds, Design, and Libraries. A "sonar" logo is also present.

The main area displays a "Violations Drilldown" report. It starts with a "Severity" summary table:

Severity	Count
Blocker	0
Critical	151
Major	4,928
Minor	1,440
Info	415

Below this is a "Rule" summary table:

Rule	Count
Signature Declare Throws Exception	1,324
IfStmts Must Use Braces	1,008
Constructor Calls Overridable Method	408
Malicious code vulnerability - May expose internal representation by returning reference to mutable object	336
Malicious code vulnerability - May expose internal representation by incorporating reference to mutable object	334
Visibility Modifier	333

Further down, there's a detailed view of violations for a file named "BeanImpl". The "Violations" tab is selected, showing 35 violations, all of which are "Signature Declare Throws Exception". The code snippet shown is:

```
142 *
143 *
144 *
145 */
146 public String Action() throws Exception {
```

A tooltip for line 146 states: "A method/constructor shouldn't explicitly throw java.lang.Exception". The code continues below:

```
147     ActionRunner runner = new ActionRunner() {
148         @Override
149         public String action() {
150             // clear dto and results
```



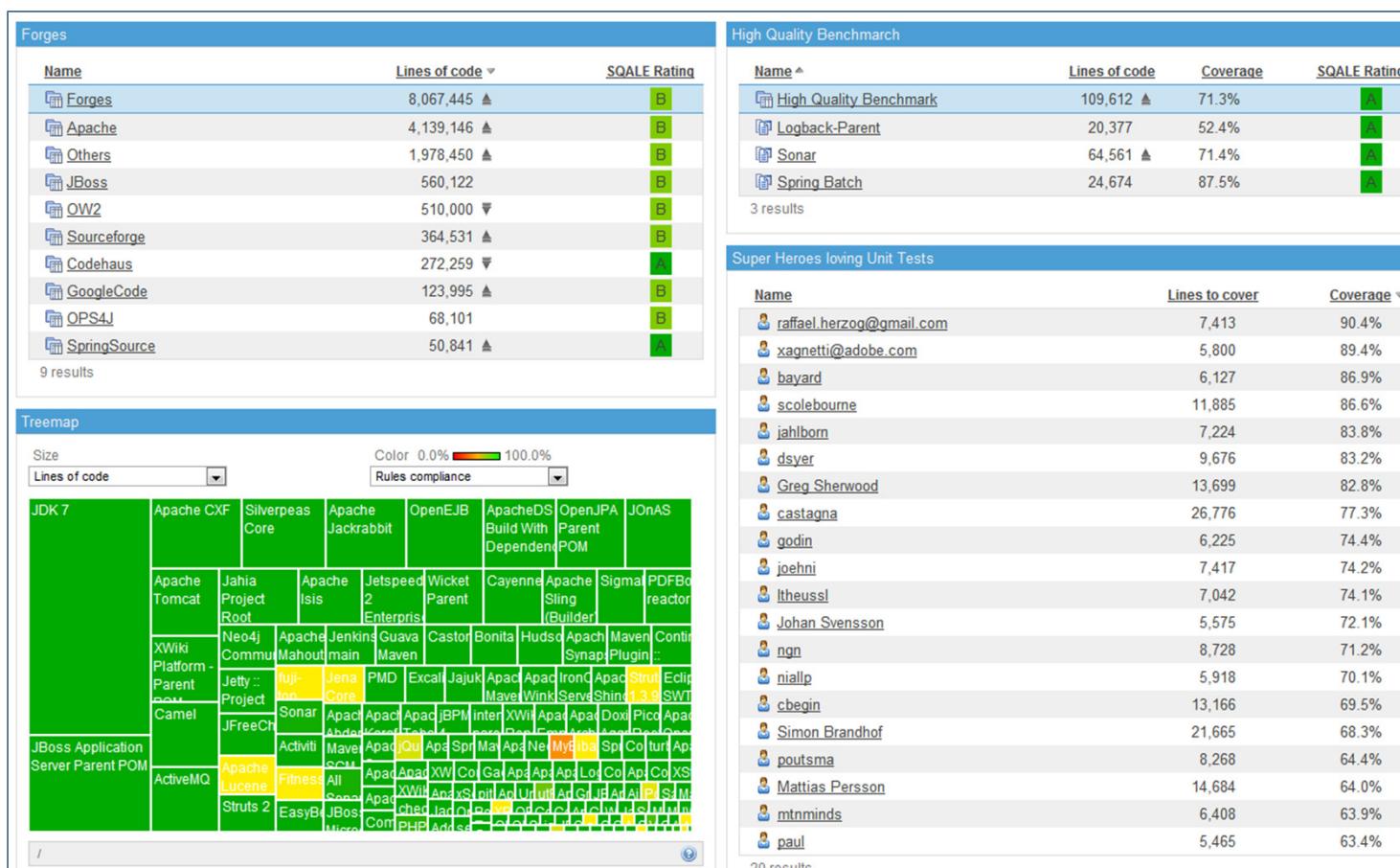
Tools: Sonar – Violations – 30 Day Delta

The screenshot shows the SonarQube web interface. The left sidebar has a blue background and lists navigation items: Home, Dashboard, Hotspots, Reviews, Components, Violations Drilldown (which is selected and highlighted in blue), Time Machine, Clouds, Design, Libraries, and a Sonar logo. The main content area has a white background. At the top, there's a dropdown menu labeled "Profile" set to "Sonar Way with Findbugs - Added over 30 days". Below it, a "Severity" section shows counts for Blocker (0), Critical (+23), Major (+361), Minor (+142), and Info (+50). A "Rule" section lists specific violation types with their counts: Performance - Inefficient use of keySet iterator instead of entrySet iterator (+3), Correctness - Method call passes null for nonnull parameter (+2), Correctness - Nullcheck of value previously dereferenced (+2), Bad practice - Class defines compareTo(...) and uses Object.equals() (+2), Correctness - Illegal format string (+2), and Performance - Method invokes inefficient Number constructor; use static valueOf instead (+2). Below these sections, there are two lists of files. The first list shows file icons and their delta values: +105, +76, +66, +59, +50, and +32. The second list shows document icons and their delta values: +32, +20, +19, +16, +14, and +10. At the bottom, a footer bar contains links: Powered by [SonarSource](#), Open Source [LGPL](#), v.2.14, [Plugins](#), [Documentation](#), and [Ask a question](#).



Tools: Sonar

- » Live Sonar instance tracking many OSS projects:
 - = <http://nemo.sonarsource.org/>



www.sungard.com/globalservices/learnmore



Tools: Unit test coverage

- » ***"I don't think anybody tests enough of anything"*** – A Conversation with Java's Creator, James Gosling, by Bill Venners, 2002
<http://www.artima.com/intv/goslingD4.html>
- » In a large codebase it is virtually impossible to “manually” gauge Unit test coverage
- » Options
 - Cobertura – Java, (Open Source)
 - Atlassian Clover – Java, Groovy
 - Ncover – .NET



Tools: Unit test coverage – Cobertura

- » How does it work?
 - 3 Steps
 1. Instrumentation - added to your Java byte-code
 2. Execute unit test
 3. Generate html report



Tools: Unit Test Coverage – Cobertura

- » Easy configuration with Maven

```
<reporting>
  <plugins>
    <plugin>
      <groupId>org.codehaus.mojo</groupId>
      <artifactId>cobertura-maven-plugin</artifactId>
      <version>2.5.1</version>
      <configuration>
        <instrumentation/>
      </configuration>
    </plugin>
  </plugins>
</reporting>
```

- » To run with maven:

mvn cobertura:cobertura

Tools: Unit Test Coverage - Cobertura

The screenshot shows the Cobertura Coverage Report interface. On the left, there's a sidebar with a 'Packages' section listing several packages under 'All'. The main area is titled 'Coverage Report - All Packages' and contains a table with columns: Package, # Classes, Line Coverage, Branch Coverage, and Complexity. The table lists numerous packages with their respective class counts and coverage percentages. A color-coded bar for each package indicates the percentage of code covered. At the bottom of the report, it says 'Report generated by Cobertura 1.9.4.1 on 8/21/12 2:11 PM.'

Package	# Classes	Line Coverage	Branch Coverage	Complexity
All Packages	662	1% / 193/17908	1% / 111/7848	2.285
android.support.v4.app	4	0% / 0/48	0% / 0/62	5.4
com.actionbarsherlock	25	0% / 0/140	0% / 0/60	2.098
com.actionbarsherlock.app	14	0% / 0/509	0% / 0/92	1.276
com.actionbarsherlock.internal	7	0% / 0/613	0% / 0/413	3.917
com.actionbarsherlock.internal.app	7	0% / 0/598	0% / 0/247	1.772
com.actionbarsherlock.internal.nineoldandroids	33	0% / 0/1369	0% / 0/652	2.603
com.actionbarsherlock.internal.nineoldandroids.animation	1	0% / 0/37	0% / 0/24	2.5
com.actionbarsherlock.internal.nineoldandroids.view	1	0% / 0/116	0% / 0/38	2.045
com.actionbarsherlock.internal.nineoldandroids.view.animation	3	0% / 0/77	0% / 0/52	2.55
com.actionbarsherlock.internal.nineoldandroids.widget	4	0% / 0/52	0% / 0/8	1.207
com.actionbarsherlock.internal.view	35	0% / 0/2134	0% / 0/1164	2.18
com.actionbarsherlock.internal.view.menu	56	0% / 0/2993	0% / 0/1719	2.923
com.actionbarsherlock.internal.widget	15	0% / 0/194	0% / 0/80	1.447
com.actionbarsherlock.view	25	0% / 0/688	0% / 0/259	2.88
com.actionbarsherlock.widget	35	0% / 0/343	0% / 0/82	1.47
com.github.kevinsawicki.wishlist	26	0% / 0/220	0% / 0/48	1.818
com.github.mobile	25	1% / 4/380	0% / 0/136	2.623
com.github.mobile.accounts	5	6% / 4/60	16% / 4/24	1.933
com.github.mobile.core	10	6% / 16/244	1% / 2/130	3.077
com.github.mobile.core.commit	8	4% / 4/89	11% / 2/18	1.583
com.github.mobile.core.issue	9	11% / 32/269	26% / 40/152	3.476
com.github.mobile.core.repo	4	0% / 0/61	0% / 0/90	8
com.github.mobile.core.user	7	20% / 10/50	29% / 7/24	3.222
com.github.mobile.persistence	11	0% / 0/213	0% / 0/50	1.896
com.github.mobile.sync	5	0% / 0/61	0% / 0/6	1.667
com.github.mobile.ui	23	0% / 0/528	0% / 0/185	1.836
com.github.mobile.ui.comment	4	0% / 0/42	0% / 0/6	1.25
com.github.mobile.ui.commit	22	0% / 0/643	0% / 0/213	2.398
com.github.mobile.ui.gist	32	0% / 0/630	0% / 0/179	2.008
com.github.mobile.ui.issue	80	0% / 0/1592	0% / 0/451	2.015
com.github.mobile.ui.repo	23	0% / 0/409	0% / 0/124	1.957
com.github.mobile.ui.user	38	0% / 0/697	0% / 0/314	2.566
com.github.mobile.util	25	18% / 123/651	19% / 56/294	3.355
com.viewpagerindicator	40	0% / 0/1156	0% / 0/452	2.487

Tools: Unit Test Coverage - Cobertura

```
92      /**
93      * Add HTML colorization to a block of Java code.
94      *
95      * @param text The block of Java code.
96      * @return The same block of Java code with added span tags.
97      *         Newlines are preserved.
98      */
99      public String process(final String text)
100     {
101        if (text == null)
102          throw new IllegalArgumentException("\"text\" can not be null.");
103
104        StringBuffer ret = new StringBuffer();
105
106        // This look is really complicated because it preserves all
107        // combinations of \r, \n, \r\n, and \n\r
108        int begin, end, nextCR;
109        begin = 0;
110        end = text.indexOf('\n', begin);
111        nextCR = text.indexOf('\r', begin);
112        if ((nextCR != -1) && ((end == -1) || (nextCR < end)))
113          end = nextCR;
114        while (end != -1)
115        {
116          ret.append(processLine(text.substring(begin, end)) + "<br>");
117
118          if ((end + 1 < text.length())
119              && ((text.charAt(end + 1) == '\n') || (text
120                  .charAt(end + 1) == '\r')))
121          {
122            ret.append(text.substring(end, end + 1));
123            begin = end + 2;
124          }
125          else
126          {
127            ret.append(text.charAt(end));
128            begin = end + 1;
129          }
130
131          end = text.indexOf('\n', begin);
132          nextCR = text.indexOf('\r', begin);
133          if ((nextCR != -1) && ((end == -1) || (nextCR < end)))
134            end = nextCR;
135        }
136        ret.append(processLine(text.substring(begin)));
137
138        return ret.toString();
139    }
140}
```



JavaDoc coverage

- » Doc check
 - Extension of Javadoc tool from Sun / Oracle
 - Generates an html report documenting Javadoc coverage levels and specific areas with omissions
- » *A good idea for consultants delivering new APIs to clients*



JavaDoc coverage

API Specification Errors Executive Summary for DocCheck Sample Code

Generated by DocCheck doclet, version 1.2b2,
Fri, 11 Jul 2003.

For a summary by package, see [Package Summary](#)
For a statistical breakdown, see [Package Statistics](#)

Missing Comments

Package	Number	Percent
All Packages	122	49%
testPkg1.*	40	51%
testPkg2.*	82	53%

Minor Errors Only

Package	Number	Percent
All Packages	67	27%
okpkg.*	1	33%



JavaDoc coverage

API Specification Errors Errors in Package: testPkg1

Package

Category 1: Package Error

- No documentation for package. Need a [package.html](#) file.

Class/Interface

Name	Missing Comments		Minor Errors		
	Class/Iface	Member	Tag	Text/Link	Warning
Test1	1	19	12	8	--
Test2	--	19	12	8	--

Explanation of Terms

Member

An inner class, inner interface, field, constructor, or method.

Note:

This table shows total numbers for the minor errors, rather than the number of items that have such an error. (If there was one method missing 4 tags, this table would show "4" in the Tags column.)



Automated functional testing

- » Selenium 2.0 – WebDriver API
- » Improvement over 1.0; no longer uses JavaScript trickery to drive the browsers; now uses native browser calls
- » Drivers for Firefox, Chrome, IE, Safari, *and even Android*
- » Tests can be written in Java, Ruby, C#, Python, Perl, PHP



Automated functional testing

- » *I choose the Ruby WebDriver API because:*
 - It can run “headless” in a CI environment
 - Headless - Ruby gem for Xvfb, a lightweight virtual framebuffer
- » And for the fact that I can write my tests in Ruby



Automated functional testing - Simplest

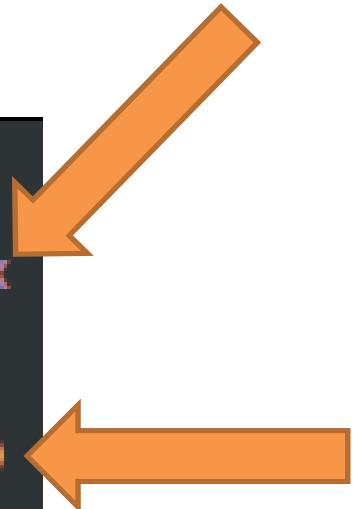
```
require "selenium-webdriver"

driver = Selenium::WebDriver.for :firefox
driver.navigate.to "http://google.com"

element = driver.find_element(:name, 'q')
element.send_keys "Hello WebDriver!"
element.submit

puts driver.title

driver.quit
```



Source: <http://code.google.com/p/selenium/wiki/RubyBindings>

Automated functional testing – Advanced

```
#!/usr/bin/env ruby

require 'selenium-webdriver'
require 'headless'
require 'test/unit'

module TestHelper

    Runheadless = true

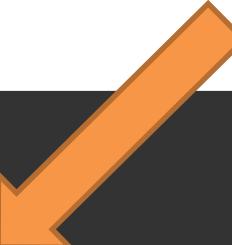
    def whoami
        "#{self.class.name}"
    end

    def setup
        if Runheadless
            @headless = Headless.new
            @headless.start
        end

        @driver = Selenium::WebDriver.for :firefox
        @driver.manage.timeouts.implicit_wait = 30
        @verification_errors = []
        @errors = true
    end
end
```



Automated functional testing - Advanced



```
def teardown
  if @errors
    t = Time.now

    @driver.save_screenshot(t.strftime("hrnt-#{whoami}-fail-%m%d%Y-%H%M.png"))
    @driver.quit
    assert false;
  end
  @driver.quit
  if Runheadless
    @headless.destroy
  end
  assert_equal [], @verification_errors
end

def verify(&blk)
  yield
  rescue Test::Unit::AssertionFailedError => ex
    @verification_errors << ex
end
```



Code Reviews

- » The benefits
 - Improved Quality
 - Professional Development
- » The challenges
 - Logistics, scheduling, out sick, conference rooms, etc.
 - Keeping the tone and scope of reviews constructive
 - Making it “automatic”



Code reviews

- » Make it part of your “culture”
 - Do them on a set schedule and make sure they happen
 - Assign the reviewee as the “host”, responsible for the meeting logistics
 - Set the expectation that reviewers come prepared
- » Use a tool!
 - Atlassian Crucible – proprietary - any version control system
 - Gerrit – Open Source – git based
 - Many others...



Simple code review framework

- » 1 reviewee, 2 reviewers
- » Reviewee sets up review, pulling in any code in question
- » Reviewers have a few days to review code at their leisure. Comments are added via the tool.
- » Reviewee and reviewers get together face to face and go through the code and address all comments
- » Document and prioritize any to-do items that come out of the review



Development tools: Database provisioning

» Liquibase

- “...an open source (Apache 2.0 Licensed), database-independent library for tracking, managing and applying database changes.”
source: <http://www.liquibase.org/>
- Store database structure and updates in your version control system
- Replay DDL changes and rebuild at will. Apply only most recent changesets.
- Push to any flavor of database using Java JDBC drivers
- We develop locally on MySQL and push to integration and production DB2 database servers



Development tools – Database provisioning

» Liquibase

```
<?xml version="1.0" encoding="UTF-8"?>

<databaseChangeLog
    xmlns="http://www.liquibase.org/xml/ns/dbchangelog"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.liquibase.org/xml/ns/dbchangelog
        http://www.liquibase.org/xml/ns/dbchangelog/dbchangelog-2.0.xsd">

    <changeSet id="1" author="bob">
        <createTable tableName="department">
            <column name="id" type="int">
                <constraints primaryKey="true" nullable="false"/>
            </column>
            <column name="name" type="varchar(50)">
                <constraints nullable="false"/>
            </column>
            <column name="active" type="boolean" defaultValueBoolean="true"/>
        </createTable>
    </changeSet>

</databaseChangeLog>
```

source: <http://www.liquibase.org/quickstart>



Development tools – Database provisioning

- » Liquibase

```
liquibase --driver=com.mysql.jdbc.Driver \
    --classpath=/path/to/classes \
    --changeLogFile=com/example/db.changelog.xml \
    --url="jdbc:mysql://localhost/example" \
    --username=user \
    --password=asdf \
    migrate
```

source: <http://www.liquibase.org/quickstart>

- » Or better yet, configure maven: mvn liquibase:update
- » We use one changelog per sprint



Development tools – Database provisioning

- » Liquibase – loading CSV **spell out** reference data

```
<loadData tableName="users" file="com/sample/users.csv">
    <column name="id" type="NUMERIC"/>
    <column name="firstname" type="STRING"/>
    <column name="lastname" type="STRING"/>
    <column name="username" type="STRING"/>
</loadData>
```

source: http://www.liquibase.org/manual/load_data



What is Jenkins?

- » A continuous integration system!
 - Everything discussed thus far should be automated with Jenkins jobs



Jenkins continuous integration

- » Install
 - Download jenkins.war
 - java –jar jenkins.war
 - (Or use Apache Tomcat)
- » Demo: <http://192.168.56.2:8080/>



SUNGARD GLOBAL SERVICES

Contact

Todd Ricker
Senior Manager
SunGard Global Services
340 Madison Ave., 7th Floor
New York, NY 10173

+1 917 727 8633 phone
Todd.ricker@sungard.com

Copyright © 2012 by SunGard Data Systems (or its subsidiaries, "SunGard"). All rights reserved. No parts of this document may be reproduced, transmitted or stored electronically without SunGard's prior written permission.

This document contains SunGard's confidential or proprietary information. By accepting this document, you agree that: (A)(1) if a pre-existing contract containing disclosure and use restrictions exists between your company and SunGard, you and your company will use this information subject to the terms of the pre-existing contract; or (2) if no such pre-existing contract exists, you and your Company agree to protect this information and not reproduce or disclose the information in any way; and (B) SunGard makes no warranties, express or implied, in this document, and SunGard shall not be liable for damages of any kind arising out of use of this document. This document is not a contract of employment, does not guarantee continued employment for any period, and does not alter the at-will status of employment.