Improved coverage analysis for LibreOffice's CI

Linus  linus@thebehrens.net
Thorsten thb@libreoffice.org
Urs Svante svante.schubert@libreoffice.org
A Prototype Fund project

The Prototype Fund is run by Open Knowledge Foundation Germany, and funded via Bundesministerium für Bildung und Forschung (BMBF).
for LibreOffice’s continuous integration:

0. develop the glue to integrate many different data providers
1. get new, shiny tools!
2. create incentives for developers and QA “to do the right thing”
3. provide automated means to locate “features” in LibreOffice
Our Team

Mr. **C++ & DevOps**
Thorsten Behrens

Mr. **Python & UX**
Linus Behrens

Mr. **Java & Graph**
Urs Svante Schubert

Cover-Rest project page:
https://cover-rest.gitlab.io/
Glue code for CI tools

Simplify n:m:o Problem

Project Stretch Goal: Improve (Coverage) Tool Modularity

n: Programming Languages

each Language has

m: Coverage Tools

which might need integration in o: CI systems
New, shiny tools!
Coverage – Jacoco Demo

Dashboard > CodeCov-Jacoco-Java-Jenkins-Demo > #56 > Jacoco > (default)

Coverage Summary

<table>
<thead>
<tr>
<th>name</th>
<th>Instruction</th>
<th>branch</th>
<th>complexity</th>
<th>line</th>
<th>method</th>
</tr>
</thead>
<tbody>
<tr>
<td>(default)</td>
<td>M: 0 C: 45</td>
<td>M: 0 C: 0</td>
<td>M: 0 C: 0</td>
<td>M: 0 C: 14</td>
<td>M: 0 C: 0</td>
</tr>
<tr>
<td></td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Coverage Breakdown by Source File

<table>
<thead>
<tr>
<th>name</th>
<th>Instruction</th>
<th>branch</th>
<th>complexity</th>
<th>line</th>
<th>method</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddsSomeNumbers</td>
<td>M: 0 C: 20</td>
<td>M: 0 C: 0</td>
<td>M: 0 C: 0</td>
<td>M: 0 C: 6</td>
<td>M: 0 C: 3</td>
</tr>
<tr>
<td></td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>AddsSomeNumbersTest</td>
<td>M: 0 C: 25</td>
<td>M: 0 C: 0</td>
<td>M: 0 C: 0</td>
<td>M: 0 C: 8</td>
<td>M: 0 C: 3</td>
</tr>
<tr>
<td></td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>
Coverage – Cobertura Demo

Code Coverage

Cobertura Coverage Report

Trend

- Classes: 100%
- Conditionals: 60%
- Files: 100%
- Lines: 95%
- Methods: 100%
- Packages: 100%

Project Coverage summary

<table>
<thead>
<tr>
<th>Name</th>
<th>Classes</th>
<th>Conditionals</th>
<th>Files</th>
<th>Lines</th>
<th>Methods</th>
<th>Packages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cobertura Coverage Report</td>
<td>1/1</td>
<td>60%</td>
<td>100%</td>
<td>1/1</td>
<td>95%</td>
<td>100%</td>
</tr>
<tr>
<td>com.example.simple</td>
<td>100%</td>
<td>60%</td>
<td>100%</td>
<td>95%</td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Coverage Breakdown by Package

- com.example.simple:
  - Classes: 100%
  - Conditionals: 60%
  - Files: 100%
  - Lines: 95%
  - Methods: 100%
Coverage – API plugin using jacoco.xml

Coverage of 'Code Coverage API Plugin: jacoco.xml'

<table>
<thead>
<tr>
<th>Component</th>
<th>Covered</th>
<th>Missed</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Branch</td>
<td>457</td>
<td>399</td>
<td>856</td>
</tr>
<tr>
<td>Instruction</td>
<td>6401</td>
<td>3632</td>
<td>10033</td>
</tr>
<tr>
<td>Line</td>
<td>1472</td>
<td>857</td>
<td>2329</td>
</tr>
<tr>
<td>Method</td>
<td>384</td>
<td>243</td>
<td>627</td>
</tr>
<tr>
<td>Class</td>
<td>61</td>
<td>21</td>
<td>82</td>
</tr>
<tr>
<td>Fino</td>
<td>45</td>
<td>14</td>
<td>59</td>
</tr>
<tr>
<td>Package</td>
<td>13</td>
<td>1</td>
<td>14</td>
</tr>
</tbody>
</table>

Coverage overview

Coverage trend
## Coverage details

### Package Overview

<table>
<thead>
<tr>
<th>Package</th>
<th>File</th>
<th>Line Coverage</th>
<th>Line Coverage</th>
<th>Branch Coverage</th>
<th>Branch Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>io.jenkins.plugins.coverage.detector</td>
<td>AntPathReportDetector.java</td>
<td>80.00%</td>
<td></td>
<td>n/a</td>
<td></td>
</tr>
<tr>
<td>io.jenkins.plugins.coverage</td>
<td>BuildUtils.java</td>
<td>60.00%</td>
<td></td>
<td></td>
<td>25.00%</td>
</tr>
<tr>
<td>io.jenkins.plugins.coverage.targets</td>
<td>Chartable.java</td>
<td>n/a</td>
<td></td>
<td></td>
<td>n/a</td>
</tr>
<tr>
<td>io.jenkins.plugins.coverage.adapter</td>
<td>CoberturaReportAdapter.java</td>
<td>70.59%</td>
<td></td>
<td>50.00%</td>
<td></td>
</tr>
<tr>
<td>io.jenkins.plugins.coverage</td>
<td>CompatibleObjectInputStream.java</td>
<td>66.67%</td>
<td></td>
<td>50.00%</td>
<td></td>
</tr>
<tr>
<td>io.jenkins.plugins.coverage.model</td>
<td>Coverage.java</td>
<td>100.00%</td>
<td></td>
<td>100.00%</td>
<td></td>
</tr>
<tr>
<td>io.jenkins.plugins.coverage</td>
<td>CoverageAction.java</td>
<td>59.09%</td>
<td></td>
<td>50.00%</td>
<td></td>
</tr>
<tr>
<td>io.jenkins.plugins.coverage.adapter</td>
<td>CoverageAdapter.java</td>
<td>100.00%</td>
<td></td>
<td></td>
<td>n/a</td>
</tr>
<tr>
<td>io.jenkins.plugins.coverage.adapter</td>
<td>CoverageAdapterDescriptor.java</td>
<td>100.00%</td>
<td></td>
<td></td>
<td>n/a</td>
</tr>
<tr>
<td>io.jenkins.plugins.coverage.targets</td>
<td>CoverageAggregationRule.java</td>
<td>88.89%</td>
<td></td>
<td>100.00%</td>
<td></td>
</tr>
</tbody>
</table>

Showing 1 to 10 of 59 entries
Jenkins has a host of plugins, and an active community (quite some software engineering research groups there too):

- Jacoco
- Cobertura
- Forensics-api
- Llvm-cov
- Opencover
- ...

More tools from the Jenkins ecosystem
Create incentives

1. automate what is possible for reviews
2. nudge developers by suggesting changes
3. create incentives for developers – e.g. have nice metrics right inside patch review
Metrics visible at patch
Feature map – “code cognita”
Determine control flow by “feature”
“Feature” load test in LibreOffice

Test loading Documents
- Test Document „text“
- Test Document „text + image“

Coverage Tool
- LibreOffice
- LibreOffice Source Code

Coverage Results
- Coverage „text“
- Coverage „text + image“
Compute location of unique features

Subtracting of Coverages (DIFF)

Coverage „text + image“

- Coverage „text“

= Lines solely used by „image“ feature!
Compute location of unique features

Subtracting of Coverages (DIFF)

Coverage „text + image“
- Coverage „text“
= Lines solely used by „image“ feature!
Develop a “code cognita” map
Questions? Answers! :)