

Configuration of JaCoCo using IntelliJ Idea.

Step 1: Make sure the project is building and test cases are running successfully.

Step 2: Create new Run/Debug configuration. Go to Run > Create/Edit configuration. As shown in Figure 1-1, choose “All in package” as test kind. Also make sure to choose the configuration for JaCoCo run as shown in Figure 1-2.

Step 3: Apply the newly created run profile and run JaCoCo with test coverage. Go to Run > “Run JaCoCo with Coverage”.

Step 4: Export the generated coverage results as shown in Figure 1-3.

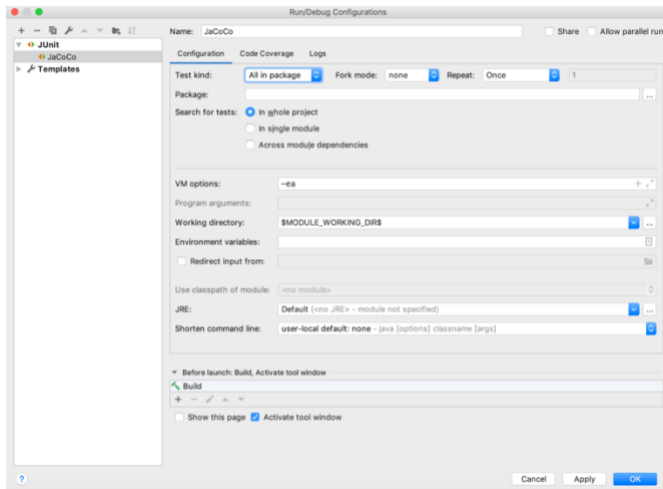


Figure 1-Error! No text of specified style in document.-1: JaCoCo Configuration in IntelliJ Idea IDE

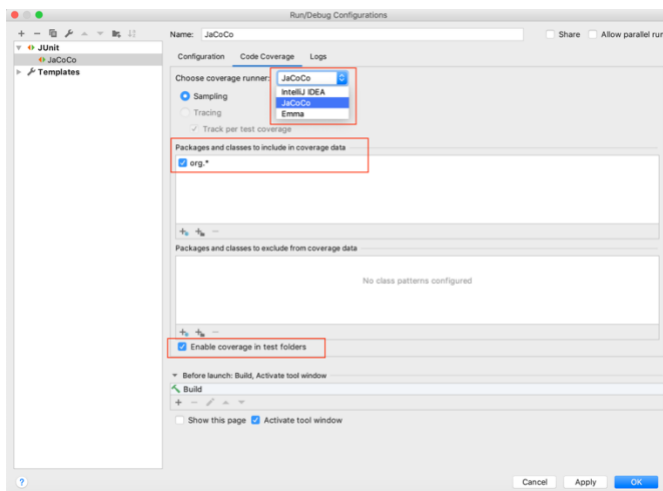


Figure 1-Error! No text of specified style in document.-2: JaCoCo Configuration in IntelliJ Idea IDE



Element	Manual Inspections	CW	Missed Branches	CW	Missed Cmts	CW	Missed Cmts	Missed Cmts	Missed Cmts	Missed Cmts	Missed Cmts	Missed Cmts
org.apache.commons.collections	70%	488	2,017	1,212	1,177	185	1,278	4	147			
org.apache.commons.collections.buffer	91%	60%	77%	548	2,734	648	1,787	117	1,561	3	172	
org.apache.commons.collections.buffer.BufferedIterator	91%	60%	79%	107	1,047	1,047	4,019	370	964	164	1,047	
org.apache.commons.collections.buffer.BufferIterator	90%	60%	80%	109	1,234	648	1,787	85	793	739	1,047	
org.apache.commons.collections.buffer.CircularBuffer	90%	60%	80%	160	541	209	1,203	299	268	268	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator	90%	60%	80%	102	178	173	1,719	37	503	1	53	
org.apache.commons.collections.buffer.CircularBufferIterator2	90%	60%	80%	38	474	209	1,203	299	268	268	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator3	90%	60%	80%	77	92	478	209	1,203	54	354	354	
org.apache.commons.collections.buffer.CircularBufferIterator4	90%	60%	80%	101	154	209	1,203	299	268	268	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator5	90%	60%	80%	84	504	1,203	299	268	268	268	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator6	90%	60%	80%	77	327	134	680	34	208	0	25	
org.apache.commons.collections.buffer.CircularBufferIterator7	90%	60%	80%	80	272	82	145	145	145	145	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator8	90%	60%	80%	84	67	482	83	173	235	205	0	20
org.apache.commons.collections.buffer.CircularBufferIterator9	90%	60%	80%	27	144	144	144	144	144	144	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator10	90%	60%	80%	67	138	30	483	2	158	0	16	
org.apache.commons.collections.buffer.CircularBufferIterator11	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator12	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator13	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator14	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator15	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator16	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator17	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator18	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator19	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator20	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator21	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator22	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator23	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator24	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator25	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator26	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator27	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator28	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator29	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator30	90%	60%	80%	11	101	101	101	101	101	101	1,047	
org.apache.commons.collections.buffer.CircularBufferIterator31	90%	60%	80%	11	101	101	101	101	101			

Figure 1-Error! No text of specified style in document.-4 Sample JaCoCo report

Metric 3 is about test suit effectiveness, and we chose PIT tool (<http://pitest.org>) to calculate mutation score. IntelliJ Idea has a plugin that simply provides results when we run mutation test on the selected test suits. The tool adds a 'Run configuration' that allows to execute PIT within IDE.

Usage: Run->Edit Configurations->Defaults->Pit Runner or just simply right click on project's parent directory and choose to run PITest from the context menu as shown in Figure 2-1. Note here, it will apply the mutation on all the classes in the project since we are choosing the same option.



After completion of running mutation testing, it will generate the test reports like shown in Figure 2-2.

Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage	Mutation Coverage
52	92% 3398/3678	87% 2798/3205

Breakdown by Package

Name	Number of Classes	Line Coverage	Mutation Coverage
org.apache.commons.codec	3	82% 23/28	100% 7/7
org.apache.commons.codec.binary	9	96% 648/677	91% 676/742
org.apache.commons.codec.cli	1	0% 0/53	0% 0/30
org.apache.commons.codec.digest	12	86% 651/759	81% 724/890
org.apache.commons.codec.language	13	98% 1226/1255	90% 935/1043
org.apache.commons.codec.language.bm	7	92% 378/413	94% 156/166
org.apache.commons.codec.net	7	96% 472/493	92% 300/327

Report generated by PIT 1.4.3

Figure 2-2 PIT test coverage report

Configuration of MetricsReloaded plugin in IntelliJ Idea IDE for static code analysis.

MetricsReloaded plugin is used for static code analysis. We used this plugin to get Halstead Volume in order to calculate Metric 5.

The tool itself is very easy to use, we just need to install the plugin in IntelliJ Idea (<https://plugins.jetbrains.com/plugin/93-metricsreloaded>), specify the metrics and metrics calculation scope, and run it (Figure 3).

Usage: Choose Analyze > “Calculate Metric” from the menu.

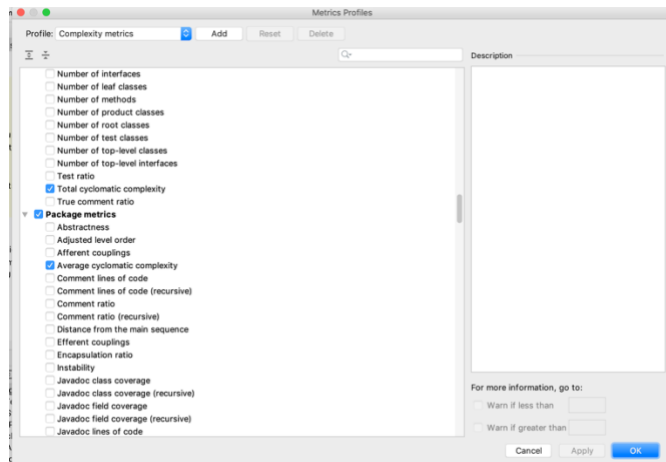


Figure 3 MetricsReloaded plugin configuration and usage