Lab 1 Report

Name: 胡紹宇
Student ID: 112598045
Date: 2024/2/29

1 Test Plan

1.1 Test requirements

The Lab 1 requires to (1) select **15 methods** from **6 classes** of the SUT (GeoProject), (2) design Unit test cases based on the experience or intuition for the selected methods, (3) develop test scripts to implement the test cases, (4) execute the test script on the selected methods, and (5) report the test results.

In particular, based on the statement coverage criterion, the **test requirements** for Lab 1 are to design test cases for each selected method so that "*each statement of the method will be covered by at least one test case* and *the minimum statement coverage is 40%*".

1.2 Strategy

To satisfy the test requirements listed in Section 1, a proposed strategy is to

(1) select those public methods that are easy to understand and have primitive types of input and output parameters (if possible).

(2) set the objective of the minimum statement coverage to be 50% initially and (if necessary) adjust the objective based on the time available.

(3) learn the necessary skills and tools as soon as possible.

(4) design the test cases for those selected methods by considering

i. the possible **valid values** and **combinations** of the input parameters.

ii. the **boundary values** of the input parameters.

1.3 Test activities

To implement the proposed strategy, the following activities are planned to perform.

No.	Activity Name	Plan hours	Schedule Date
1	Study GeoProject	0.25	2024/2/24
2	Learn JUnit	0.25	2024/2/24
3	Design test cases for the selected methods	1	2024/2/29
4	Implement test cases	2	2024/2/29
5	Perform test	2	2024/2/29
6	Complete Lab1 report	1	2024/2/29

1.4 Success criteria

All test cases designed for the selected methods must pass (or "90% of all test cases must pass) and *the statement coverage should have achieved at least 80%*.

2 Test Design

To fulfill the test requirements listed in section 1.1, the following methods are selected and corresponding test cases are designed.

No.	Class	Method	Test Objective	Inputs	Expected Outputs
1	Info	id()	Info	N/A	taipeiId
2	Info	lat()	Info	N/A	25.104397
3	Info	lon()	Info	N/A	121.597366
4	Info	time()	Info	N/A	1708757706
5	Info	value()	Info	N/A	taipei
6	Info	toString()	Info	N/A	Info [lat=25.104597, lon=121.597366, time=1708757706, value=taipei, id=Optional.of(taipeiId)]
7	Coverage	getHashes()	Coverage	N/A	wsqqqjhw9yw767cn, xn77jk5fzrzf8rc2
8	Coverage	getRatio()	Coverage	N/A	0.5
9	Coverage	getHashLength()	Coverage	N/A	16
10	Coverage	toString()	Coverage	N/A	Coverage [hashes=[wsqqqjhw9yw767cn, xn77jk5fzrzf8rc2], ratio=0.5]
11	CoverageLongs	getHashes()	CoverageLongs	N/A	8703398685819821410, 7729896939150581108
12	CoverageLongs	getRatio()	CoverageLongs	N/A	0.5
13	CoverageLongs	getCount()	CoverageLongs	N/A	2
14	Base32	encodeBase32()	Base32	75324, 4	29jw
15	Base32	decodeBase32()	Base32	29jw	75324

3 Test Implementation

The design of test cases specified in Section 2 was implemented using JUnit 4. The test scripts of 3 selected test cases are given below. The rest of test script implementations can be found in the link (or JUnit files).

No.	Test method	Source code

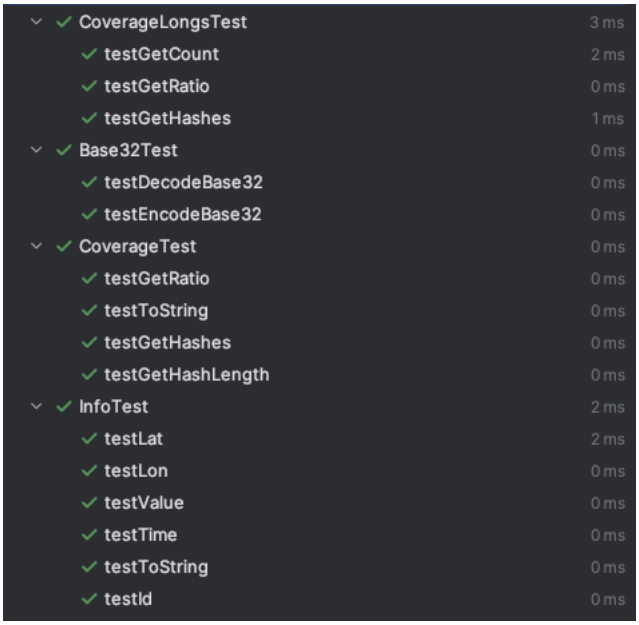
1	Base32.encodeBase32 ()	@Test
		public void testEncodeBase32() {
		String encode = Base32.encodeBase32(75324, 4);
		assertEquals("29jw", encode);
		}

2	Coverage.getHashes()	@Test
		public void testGetHashes() {
		final Set<String> actualHashes = coverage.getHashes();
		
		assertTrue(actualHashes.contains(taipei101GeoHash));
		assertTrue(actualHashes.contains(tokyoSkytreeGeoHash));
		assertFalse(actualHashes.contains(seoulTowerGeoHash)); // not added
		}

3	CoverageLongs.getHashes()	@Test
		public void testGetHashes() {
		final long[] actualHashes = coverageLongs.getHashes();
		assertEquals(hashes[0], actualHashes[0]);
		assertEquals(hashes[1], actualHashes[1]);
		}

4 Test Results

4.1 JUnit test result snapshot



Test Summary

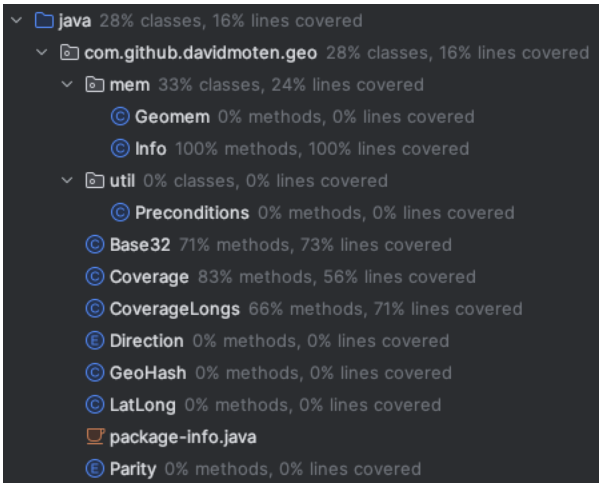
15 tests	0 failures	0 ignored	0.005s duration	100% successful

Packages | Classes

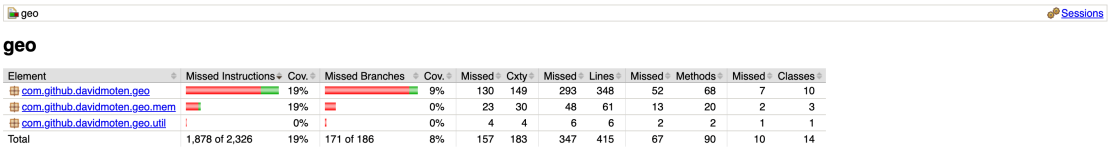
Package	Tests	Failures	Ignored	Duration	Success rate
com.github.davidmoten.geo	9	0	0	0.003s	100%
com.github.davidmoten.geo.mem	6	0	0	0.002s	100%

4.2 Code coverage snapshot

- Coverage of each selected method

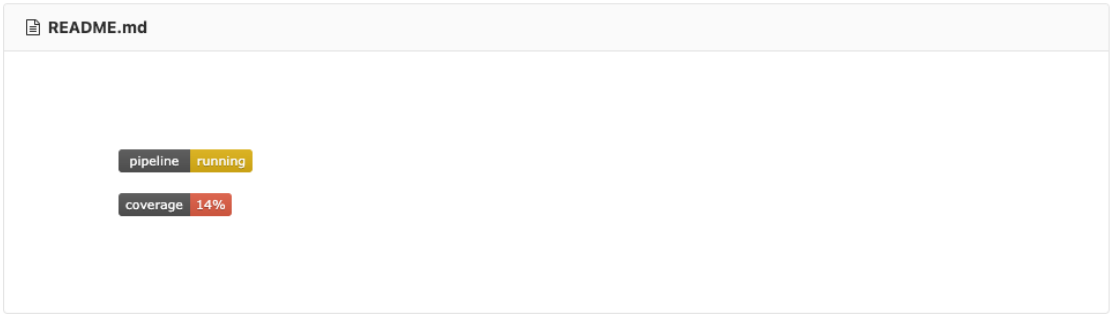


- Total coverage

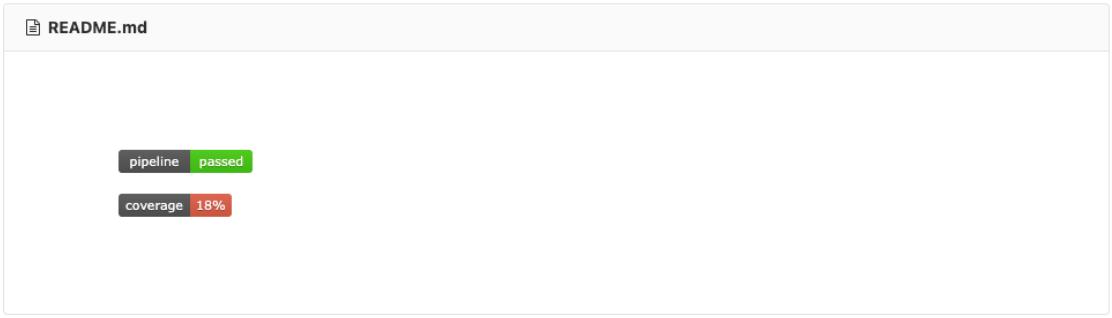


4.3 CI result snapshot (3 iterations for CI)

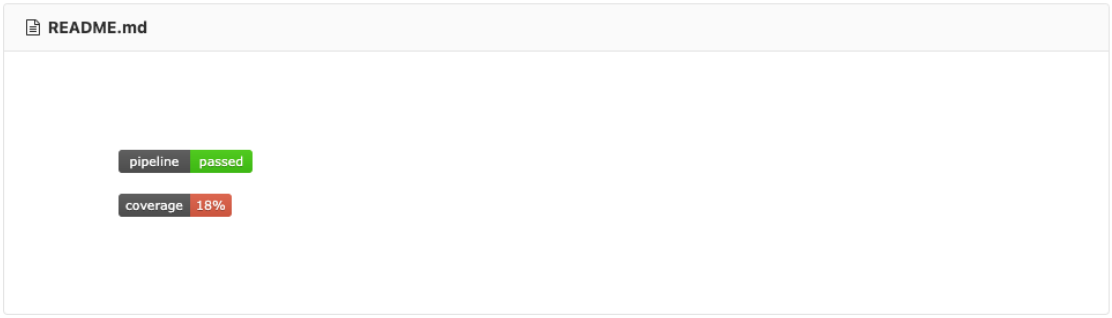
- CI#1



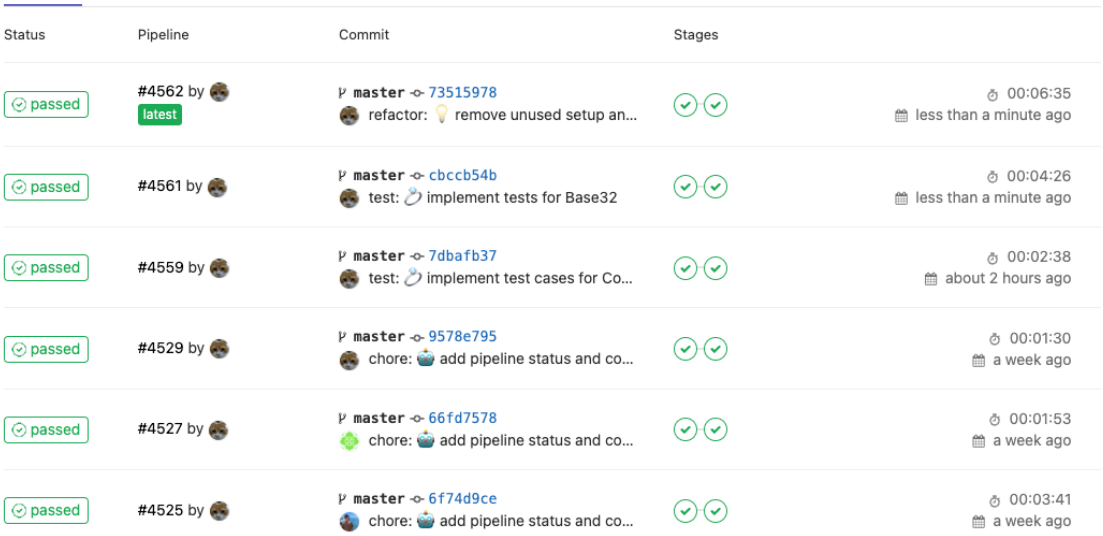
- CI#2



- CI#3



- CI Pipeline



Status	Pipeline	Commit	Stages	
passed	#4562 by latest	master 73515978 refactor: remove unused setup an...		00:06:35 less than a minute ago
passed	#4561 by	master cbccb54b test: implement tests for Base32		00:04:26 less than a minute ago
passed	#4559 by	master 7dbafb37 test: implement test cases for Co...		00:02:38 about 2 hours ago
passed	#4529 by	master 9578e795 chore: add pipeline status and co...		00:01:30 a week ago
passed	#4527 by	master 66fd7578 chore: add pipeline status and co...		00:01:53 a week ago
passed	#4525 by	master 6f74d9ce chore: add pipeline status and co...		00:03:41 a week ago

5 Summary

In Lab 1, **15 test cases have been designed and implemented using JUnit**. The test is conducted in 3 CI and **the execution results of the 15 test methods are all passed**. **The total statement coverage of the test is 28%.** Thus, the test requirements described in Section 1 are satisfied. Some lessons learned in this Lab are …

1. Learned how to use JUnit
2. Learned how to generate test reports and test coverage reports