Lab 2 Report

Name: 陳日揚
Student ID: 106598014
Date: 2018/4/9

1 Test Plan

1.1 Test requirements

Lab2 要求從 GeoProject 中(共 6 個 Classes，50 個 Methods)挑選出 15 個 methods 來進行測試。首先，必須先理解及分析待測程式的功能及目的為何？本次 Lab 要求採用 ISP (Input Space Partition)的技巧來設計 test cases，因此我們必須根據待測程式的介面(Interface)或功能(Function)來找出特徵(Characteristic)，並利用這些特徵來將 test input 切格成不同的 blocks。接著，將設計的 Test cases 轉換為可實際執行測試的程式碼。最後將測試的結果記錄在本測試報告中。

根據題目要求，15 個待測的 Methods 皆必須被我們所設計出來的 Test cases 包含。同時，整體程式的 statement coverage 要達到 40% (較 Lab1 高)。

1.2 Strategy

為了達成 Section1.1 所描述的需求，將採用以下的策略：

(1) 挑選與 Lab1 相同的 15 個 test cases。

(2) 挑選參數值與回傳值只包含 primitive type 的 Methods 以方便測試。

(3) 在設計測試案例前先分析與理解待測程式執行的內容，同時也必須瞭解該領域的 domain knowledge 才能正確地定義出測試案例的輸出是否正確。

(4) 採用 ISP (Input Space Partition)的技巧來設計 test cases。

1.3 Test activities

下列為本次測試過程中所包含的活動。

No.	Activity Name	Plan hours	Schedule Date
1	Study GeoProject	0.5Hr	2018/4/7
2	Learn ISP Slide	3Hr	2018/4/7
3	Design test cases for the selected methods	3Hr	2018/4/7-2018/4/8
4	Implement test cases	1.5Hr	2018/4/8
5	Perform test	0.5Hr	2018/4/8
6	Complete Lab2 report	3Hr	2018/4/9

1.4 Design Approach

本次 Lab 採用 ISP (Input Space Partition)的技巧來進行測試，ISP 的 Input domain modeling 共有五個步驟，以下說明這次 Lab 實作這五個步驟的情況：

(1) Identify testable functions：
找出欲測試的 15 個 test function。

(2) Find all the parameters：
分析 test functions 所需要 input 的參數類型、名稱以及 output 的類型。

(3) Model the input domain：
這個步驟分成兩種方式，分別為 Interface-based 及 Functionality-based，我採用 Interface-based 的方式根據輸入的參數來尋找 characteristics 及切割 blocks。

(4) Apply a test criterion：
設定通過測試的 statement coverage 標準。

(5) Refine combinations of block into test input：
從每個 block 中選定適合的 test input value 來進行測試。

1.5 Success criteria

因為待測程式是一 Open Source 的 Library，理論上正確率應該非常高，因此本測試所設計之 Test cases 的通過率至少需達成 95%以上，並且單一 Method 的 Statement coverage 必須至少達成 85%(較 Lab1 高)。

2 Test Design

因本測試報告的版面限制，Test case 的詳細設計請參考附件 Excel 檔案(*Lab2 (ISP test case design).xlsx*)。

3 Test Implementation

本次 Lab 使用的測試工具為 Junit 4，下列挑選 Section 2 中設計的三個 Test cases 實作內容，未列出的測試內容可在 GitLab 上查看。

No.	Test method	Source code

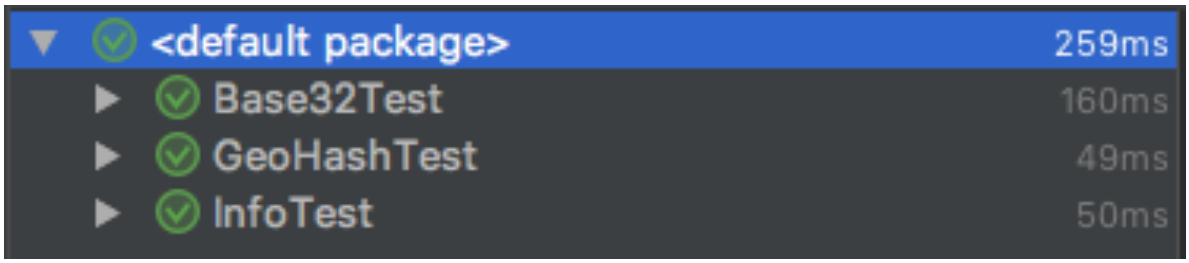
1	encodeBase32Test_T1	`@Test`
		`public void encodeBase32Test_T1() {`
		`  String encode = Base32.encodeBase32(Long.MIN_VALUE, 0);`
		`  assertEquals("-8000000000000", encode);`
		`}`

2	rightTest_T25()	`@Rule`
		`public ExpectedException thrown= ExpectedException.none();`
		
		`@Test`
		`public void rightTest_T25() {`
		`  thrown.expect(IllegalArgumentException.class);`
		`  thrown.expectMessage("adjacent has no meaning for a zero length hash that covers the whole world");`
		`  GeoHash.right("");`
		`}`

3	idTest_T33()	`@Test`
		`public void idTest_T33() {`
		`  Integer i = null;`
		`  Info<String, Integer> info = new Info<String, Integer>(23.25, 120.55, 123, "2", Optional.fromNullable(i));`
		`  assertTrue(!info.id().isPresent());`
		`}`

4 Test Results

4.1 JUnit test result snapshot



Test Summary

47 tests	0 failures	0 ignored	0.099s duration

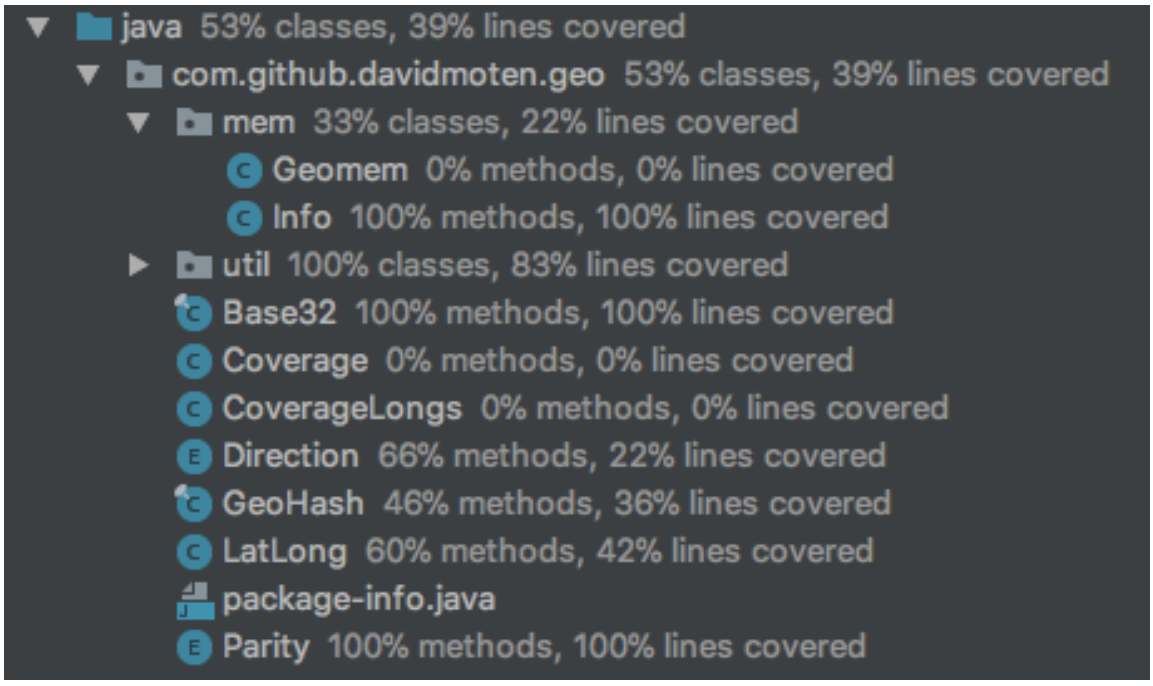
100% successful

Packages | Classes

Package	Tests	Failures	Ignored	Duration	Success rate
com.github.davidmoten.geo	32	0	0	0.086s	100%
com.github.davidmoten.geo.mem	15	0	0	0.013s	100%

4.2 Code coverage snapshot

- Coverage of each selected method



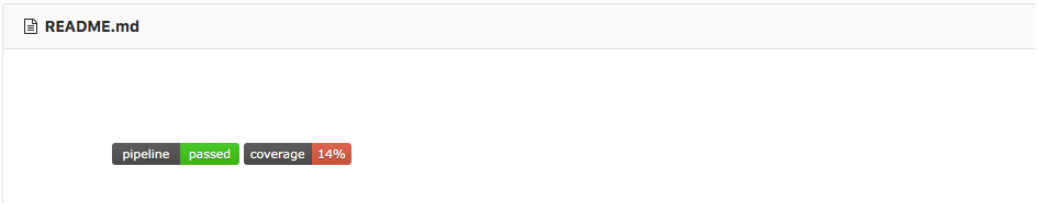
- Total coverage

geo

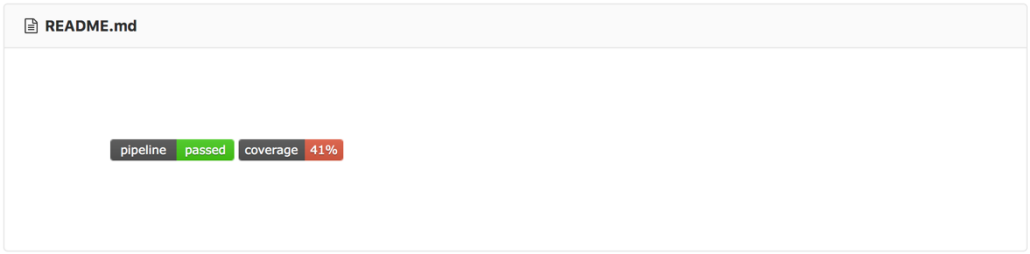
Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.github.davidmoten.geo		48%		33%	105	159	211	354	46	78	4	10
com.github.davidmoten.geo.mem		19%		0%	23	30	48	61	13	20	2	3
com.github.davidmoten.geo.util		59%		75%	2	5	3	8	1	3	0	1
Total	1,338 of 2,379	44%	130 of 186	30%	130	194	262	423	60	101	6	14

4.3 CI result snapshot (3 iterations for CI)

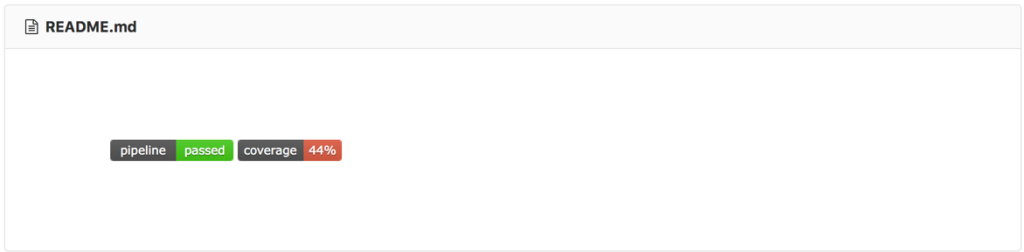
- CI#1



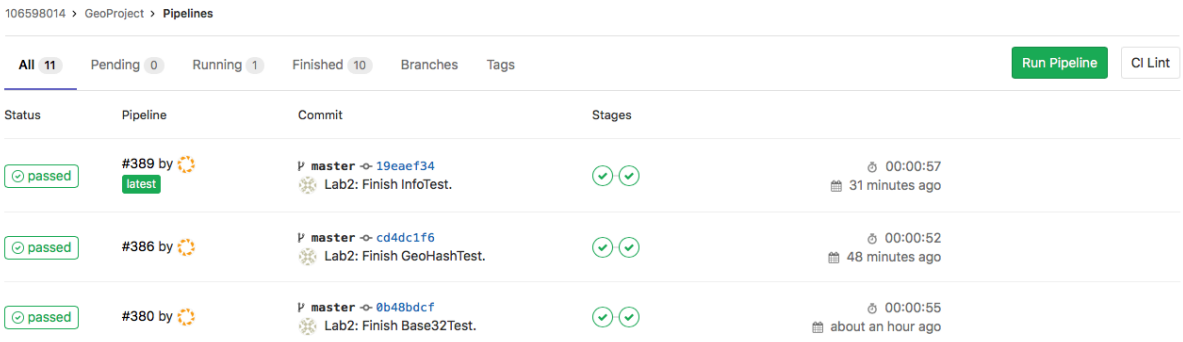
- CI#2



- CI#3



- CI Pipeline



5 Summary

在此次的 Lab2 中，總共設計並利用 JUnit 執行了 15 個單元測試，測試的過程中共執行了三次的 CI。測試結果全數通過，且整體的 Statement coverage 為 44%，有達成 Section 1 所要求的 40%覆蓋率。同時，在這次 Lab 的執行過程中也學習到了 ISP (Input Space Partition)測試的觀念及實作。ISP 測試可用在各種 level 的測試(包含單元測試、整合測試、系統測試等)，且在設計 ISP 的測試案例時並不需具備實作的知識，因此 ISP 測試是最簡單的一種測試方法。