# 159.251 – Assignment 2 – 2018

Due date: **November 20th at 8.00pm (Beijing Time)**
Worth: **19% of final grade**

**Read carefully as there are many things that you should be aware of.**

## Overview

You are to work in **self-selected pairs** to create the program defined below, using ***git*** to manage source code contribution and integration between the two developers. All project issues and changes should be tracked using ***issue tracker software*** (feel free to select any issue tracker software that you would like to work with).

**Note: Both members of the group will receive the same mark,** unless it is clear that the work is predominantly that of a single person. These will be sorted out on a case-by-case basis. The partition of the work is entirely up to you and your project partner.

Part of this assignment is to become familiar with using ***git*** for version control which we'll cover soon in the "version control in software development" lecture. There's a quick-intro guide at the end of the assignment to get you started. You will need to use Bitbucket (http://www.bitbucket.com) –a free git hosting service. Bitbucket provides free ***private*** repositories. For Github, you can requiest an *academic licence* which will allow to create private repositories.

**IMPORTANT:** Choose meaningful names for your repository (e.g. **251-Assignment2-FirstName1-FirstName2**). For example, if the first student is Ma and second student is Lee, then their repository should be (**251-Assignment2-Ma-Lee**), otherwise I end up getting emails from Bitbucket telling me that member *Whistle* has invited me to join project *Celery*. This isn't helpful.

# Tasks

## There are several parts to this assignment:

1. Developing a *ToDo Script*.
2. Creating and maintaining a *git* repository on your local machine for your source code, and also on a remote repository to provide a central server accessible to both members, and also accessible for marking.
3. Keeping an audit trail of commits in the *git* repository. These will form part of the marking.
4. Make sure that you use gip features of ***branching*** and ***merging***.
5. Keep track of changes and issues – using *issue tracker* software.
6. Make sure you write clean code, and write comments to explain your code.
7. Use a code quality checking tool to report some metrics from the project.

# 1. The Todo Shell Scripting

### 1. Shell and scripting

A team of developers manage todo lists in their local directories as markdown files. Each mark- down file represents a project and they contain a list of todos. Markdown format:

# Project title
## Tasks
[ ] (A) Todo 1 due:2010-01-23
[ ] (C) Todo 2 due:2010-01-21
[X] (B) Todo 3 due:2010-01-10

Write a script that:
1. Checks that the project title matches the file name, moves files that don't match to a directory named ./error_files
2. Copies todos without a due date to directory named ./no_duedates
3. Combines all project todos (todos without errors) in a single file

# 2. Software development

You are provided with the source code for a basic web application that can be used to manage todo lists. The application uses Apache Wicket (a Java web framework.) You are required to implement additional functionality:
1. Clear completed tasks from the list (2/45)

2. A filter to show either completed, active or all tasks (5/45)
3. Show the total number of incomplete tasks (5/45)
4. Add a due date, project title field in the add task form (3/45)

5. Add persistence (e.g. XML-based serialisation, SQLite database) support for the application (cleared tasks must be archived) (8/45)

6. Import the tasks from the todo list text file into your application (5/45)
7. Add an indicator in the webpage so that it is easy to distinguish the environment it is running in (development or production) (2/45)

Once development is done, you need to report metrics data using the Eclipse Metrics Plugin (http://metrics2.sourceforge.net). Code quality report from PMD should also be submitted with your assignment. The following metrics should be collected and reported clearerly in a separate file that must be submitted with the project (this should be in your git repopository).

The metrics that you need to report are:

Code Size (per class): Lines of Code (LOC) and Number of Methods (NOM)

Code Complexity: Cyclomatic Complexity and Coupling Between Objects (CBO) and God Classes.

Code Quality Report from PMD. Use only Java Basic rules such as Naming Convention for classes and variables (extract the full report and included with your submission).

**Methodoloy**

**Use an Agile methodology. Due to the short time I recoment Extreme Programing with short Sprints of 4 days per Sprint.**

# Submitting your assignment

All submission is to be done using Stream:

share your program on your ***private*** git repository (GitHub or Bitbucket) with me by sending a share invitation to p.shaw@massey.ac.nz and jacobstringer@windowslive.com

**This is to track commits on your git repository or bitbucket.**

## Include a Readme.md file in the top level of the project

The *Readme.md* is a text file with a Markdown syntax that contains:

1. the names & IDs of BOTH MEMBERS of the group.
2. clear instructions on how to run your program, and if there are any other folders, what they contain.
3. for each student, a couple of git ***commit IDs*** that show the work of each individual member of the group.

4. Metrics analysis report from Eclipse Metrics Plugin and code quality analysis report from PMD.
5. any other features you feel worth mentioning.

## Who submits what?

Only one member of a group should submit a complete project, the other just submits the *Readme.md* file:

- **member A**: submit (through Stream) a single compressed (e.g., zip or tar) file that contain the assignment
    - **name the compressed file with both members' FirstName_LastName and ID numbers** (e.g. James_Smith-87817172- Susan_Jones 01002023.zip)
- **member B** : submit just the README.md file containing the your name and that of the partner who is submitting the zip/tar file. This is so Stream knows that you've submitted something, otherwise it won't let a mark be entered.

Read more about **.md** files here (http://stackoverflow.com/questions/5922882/what-file-uses-md-extension-and-how-should-i-edit-them)

# Assessment

Your assessment will be based on the following criteria:

| Criteria | Worth |
|---|---|
| Correct implementation of the **todo script** | 5 |
| Correct implementation of the todo java program | 5 |
| Appropriate use of git/bibbuck **FROM THE START OF DEVELOPMENT** | 4 |
| Appropriate use of issue tracking features to track changes/issues **FROM THE START OF DEVELOPMENT.** | 3 |
| Check code quality and **report the specific size and complexity metrics.** | 1.5 |
| Overall code quality, including exception handling and comments to explain the code. | 1.5 |
| **Total** | **19** |

# Late Submissions Penalties

There is a penalty for late submissions. The penalty is 10% deducted from the total possible mark for every day delay in submission (one day late – 90%, two days – 80% etc).