Assignment HW3

Cover Page

Prepared for:
Dr. Mehra Borazjany
Trung Hieu Tran

Prepared by:
Alex Lundin
Ali Haider
SE-4367.0U1-Testing

Assignment Choice:
N/A this week

06 June 7, 2018

Proof of Working Software

GitHub link:

https://github.com/AlexLundinEducational/SE-4367-Testing

Branch Summary:

master – managed by Alex, only fully completed pulls allowed to make TA's life easy. Master only contains assignment material once they reach completed status.

working – flexible branch for team, ideally, this material should build without causing technical debt during the project

Commit for grading:

HW3_TEAM1 Complete, Ready for Merge to master and Ready for Grading

Phase 1 Development

Initial Webex Meeting, Alex handles GUI / Ali handles XML writes

Phase 2 Development

https://autode.sk/2l1kJV2 (0:30)

Alex's side of the project is done

Phase 3 Development

Phase 3 was a cluster of chaos, constant 1 step forward 3 steps backwards. This all stemmed from low quality design decisions. Each one had an escalating effect on the next. We will not deliver any group homework on time if we repeat this. Alex made this clear several times before starting this homework, during the homework and now in the final documentation.

https://autode.sk/2t0NCo2 (1:32)

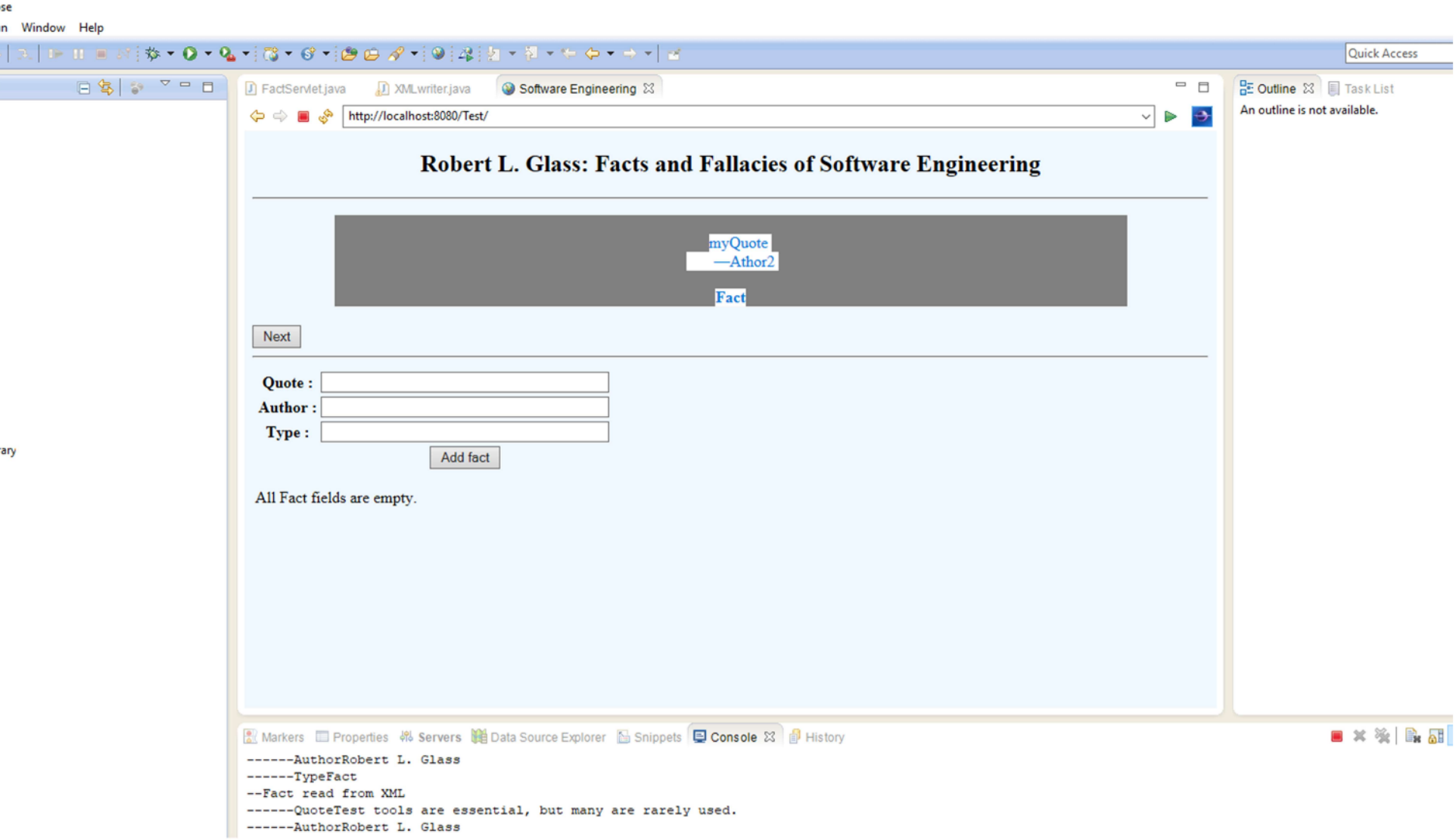
https://autode.sk/2t3ii8g (0:51)

https://autode.sk/2l7H5o0 (1:00)

Proof of Functional build

Alex stayed up until 5am carrying the team, not the original plan.

https://autode.sk/2l4AXg9 (1:17)


Window Help
Quick Access
FactServlet.java
XMLwriter.java
Software Engineering
http://localhost:8080/Test/
Outline
Task List
An outline is not available.
Robert L. Glass: Facts and Fallacies of Software Engineering
myQuote
—Athor2
Fact
Next
Quote :
Author :
Type :
Add fact
All Fact fields are empty.
Markers
Properties
Servers
Data Source Explorer
Snippets
Console
History
------AuthorRobert L. Glass
------TypeFact
--Fact read from XML
------QuoteTest tools are essential, but many are rarely used.
------AuthorRobert L. Glass

Documentation Log

I tagged all modifications as follows

HW3 – any edits to the base functionality

Mesh point – control flow between servlet and XMLwriter

*** - all the issues I had to fix before carrying the final **5** hours of work

1. Reductions
    a. FactServlet.java
        - Removed search feature
2. Refactors
    a. StringConstants.java
        - this is a refactoring of the strings used in the JavaServlet.java file

    b. FactServlet.java
        - this is a refactoring of the strings used in the JavaServlet.java file

3. Modifications
    a. Fact.java
        - Added several helper methods
    b. FactServlet
        - Form table
            - Fields
            - Buttons
        - Control flow
            - Request in control flow to pull string from the fields
            - Use addFact method to implement the addButton in the control flow
            - addFact is the mesh point to XMLwriter.java class
        - Modified
4. Additions
    a. XMLwriter.java
        - Wrapper method combines all necessary control flow for reading existing xml data, adding new data and writing the final combined list

Maintainability Assessment

What did the original programmers do that made it hard to change the software?

The original programmers included too much functionality inside the FactServlet.java class file. This file should have been broken into a few class files. The rule of thumb is 1 method per class file if possible.

What did the original programmers do that made it easy to change the software?

Using XML notation added extremely loose coupling. I didn't have to edit any existing functions. Since these components were grouped together but not constrained it was a breeze to word around them. Not sticky at all.

What would you do differently if you did it again?

I would modularize the XML reader. As is stands currently, there are hard coded strings inside the reader. It would be better to have a method that accepted the xml tag and then did the reading and parsing.

Ali Haider
Alex Lundin

Homework3 **SE 4367: Software Testing, Verification, Validation and QA**

PLEASE PRINT ALL PARTICIPANT NAMES ON TOP OF THE PAGE

Class, Section and Group# SE-4367 Testing, Section 0U1 Group 1
Total Points (Out of 100 points) _____

Instruction:

1. Answer to the problem on a **PDF** file **(PDF file only)** and save it as HW#_YOUR-TEAM#.PDF
2. Submit the **PDF** file to eLearning before the due date

Your job this week is to evolve the server to provide the ability to add new facts through the interface. (Please note that editing the data file directly does not satisfy the assignment.) For full credit, the new facts should be saved in the permanent file.

You can decide how to add this feature, but you must attempt to preserve the integrity of the data file. That is, check the new text to ensure it conforms to minimal syntactic requirements. It is up to you to determine the rules for new facts (what to check for), how to check, and what to do if the facts are not valid.
The original web app had a "community search" feature. If that does not make sense in your system, you may remove that functionality.

As in HW2, when you make the changes, keep a simple documentation log of what you do. Note which components and methods you change and added. Summarize the changes in a few words. You do not need to go to the level of which variables you create or delete or how you change the control flow. I just want the highlights. This should be one page or less. (*Hint: This will much easier if you keep the log while you change the program. Trying to remember later will take more time. That is, waiting until the end will create maintenance debt, always a bad thing.*)

Next, write a short assessment of the maintainability of the software. What did the original programmers do that made it hard to change the software? What did the original programmers do that made it easy to change the software? What would you do differently if you did it again?

Submission

Submit five items in a PDF file:

1. A very simple cover page with all partner names (no more than three per team), and where the TA can access your source files
2. A screen shot showing your working software
3. A one-paragraph description of the data integrity checks
4. The simple documentation log
5. Your maintainability assessment

Grading

We will grade on several factors.

- (20 pts) Whether each item is included
- (15 pts) Whether the modified software works
- (20 pts) Clarity and thoughtfulness of the documentation log
- (20 pts) Clarity and thoughtfulness of the maintainability assessment
- (15 pts) Quality and maintainability of the changes you made to the code (this is why we need access to source files)
- (10 pts) Other factors to be determined while grading such as team collaborations,….