Project X: Proposal Worksheet

Preston Lee

Contents

1 Abstract

Project X is a self-defined team project that uses the combined brain power of your team to define, design, build, test and demo a meaningful Java programming project. This worksheet will take you through the initial steps team and project planning steps. **This a proposal that must be "approved" by the instructor prior to development.**

2 Build the team!

Self assemble yourselfs into teams of your choosing. Team requirements:

1. Three or four people. *No exceptions.*
2. At least one person with *no* previous Java courses.
3. At least one person with previous Java courses.
4. Captain must have no previous Java courses.

2.1 Pick some cool names.

Every project team needs some cool names, as well as a captain. Fill in the blanks:

Team Captain:
Team Code Name:
Project Code Name:

2.2 Gather member information.

Fill in the following table with your team information

Table 1: Team member information.

Number	Name	ASURite	Previous Course	Email
(e.g.)	Alice Anderson	aanderson	CST 100 (Java)	aanderson
#1				
#2				
#3				
#4				

3 BYO project.

3.1 Define the problem.

Define a one-paragraph problem statement that you will attempt to solve. (Tip: Pick a *practical* problem that you think you and your team can realistically solve in about a months time.)

3.2 Demonstrate your knowledge.

For this proposal to be approved, your project **must** prove your knowledge of course material by demonstrating *all* of the following:

Timeliness Real-world projects can't wait until the last minute, and neither should you. The development of your project should be paced throughout the time you are given, **not** crammed into the last week.

Abstraction Your must "reuse" code (in a meaningful way) by using the concepts of abstraction, inheritance and well as polymorphic references where sensical in your application.

Input The application must accept input from the user, disk, network or other external source.

Output Programs must provide meaningful real-time output to the use. It's ok to write to disk, but you still must print to the screen or put up some sort of GUI. (**Bonus**: Implement a GUI using Swing or SWT.)

Exception Handling Any/All user, disk, network etc. I/O must be "solid": written with thorough exception handling practices to provide a reasonable level of application robustness. Use try/catch blocks and checked/unchecked exceptions where appropriate. (**Bonus**: Make use of a network connection.)

Code Documentation Code with documentation on usage and programmer thinking is a magnitude more valuable than code without. All

code must be thoroughly comment in JavaDoc format. (**Bonus**: Provide "howto" documentation on all reusable classes.)

Automated Test Cases Provide a suite of "unit tests" that allow you to quickly run regression tests against your code base. You must also include "negative" test cases: code which intentionally calls functions with invalid input to verify that the code fails the way it is supposed to. For example, passing null, invalid numbers etc. to functions expecting "correct" input should do something reasonable. (**Bonus**: Use a unit test framework.)

3.3 Project summary.

In 2-3 paragraphs, summarize the purpose, input, behavior, and expected output of your application.

3.4 Target user(s).

In 1-2 paragraphs, describe the intended user of your application. (If you're writing a game, for example, is it for kids or adults? ...fun or educational purposes? ...paying customers or free access?)

3.5 Milestones.

The team will need to hit all of the following milestones for full credit.

Thursday, September 16th Proposals due.

Thursday, September 23th Last chance acceptance date.

Thursday, October 7th Sanity check. Your team (led by the captain) will walk me through your progress thus far. By this point I expect you to have a good start on the project objectives as well as a good working relationship (and process) with the team.

Thursday, October 28th Demo and presentation. Your team–lead by the captain–will deliver a well rehearsed project presentation and application demonstration in front of the class.

4 Define success.

All team members will receive the same grade for the project. Additionally, the team captain will deliver a live 10-15 minute demo in front of the class prior to grading.

4.1 Grading breakdown.

Total points possible: 36. (Tip: This is a lot, so take the project seriously.)

20% In-class demo. *Be prepared.* Be professional. Quality of content, delivery and project demo will be considered. In other words, make sure your project doesn't "blow up" in front of the class! You will have projector access to show slides, code, and other additional resources you deem fit.

40% Given by instructor, based on the previously defined criteria.

40% Defined by your team, below.

4.2 Custom grading criteria.

Define at least **four** specific ways your project should be graded. (For example: *"Correctly retrieves automated Google search results"*, or *"Renders MineSweeper board correctly."*) These must be reasonably challenging criteria for a team of 3-4 people given over a month of development time. Trivial criteria such as *"Application accepts invalid user input without crashing."* will be rejected.