Swift Project Guide
CPSC 223W

(Rev. 2-20200903)

Overview

Projects provide an opportunity to work with others to develop a large application over the semester. This will allow you to practice designing and writing software that is robust and maintainable.

Learning Objectives

At the end of the semester you are expected to:

1. Identify a project that can be implemented in Swift.
2. Write a design document that will guide the development of the project.
3. Design and write code that makes appropriate use of fundamental programming constructs and data structures (e.g., expressions, conditions, loops, user-defined functions, primitive data types).
4. Design and write code that makes appropriate use of object-oriented concepts (e.g., classes, objects, methods, composition, and inheritance).
5. Design and write code that makes appropriate use of advanced programming concepts (e.g., error handling).
6. Write code that follows coding standards.
7. Write appropriate comments that help other developers understand and reuse code.
8. Design unit tests to evaluate the project's correctness and completeness.
9. Analyze and interpret compilation errors, unit test results, and code behavior to debug code.
10. Write code that utilizes programming-language constructs and software libraries from documentation.

Requirements

You will work in groups of three students to identify and develop any software application using Swift. The application needs to apply concepts we discussed in class, use an object-oriented design, and requires around ten weeks for development. We will have several consultations to ensure that you complete your project in time. In addition, your project will need to implement the following components.

1. **User Interface**
   Your Swift application must provide a way for users to interact with its functionalities. It is highly recommended that you use Xcode's Interface Builder as we are discussing it in class and it offers an easy-to-use interface. If your group prefers to use a different platform (e.g., SwiftUI) you are more than welcome to do so. Similarly, you are free to develop another type of application, such as web applications that use HTML or Javascript. However, take note that other interfaces will not be discussed in class and it is up to you to do research on them.
2. **Storage**
   Your Swift application needs to store data that allows it to perform advanced functionalities. Data can be stored in files, databases, or through a web-based API. It is up to the team to do research in implementing these functionalities.
3. **Unit testing**
   Your Swift application needs to have unit tests that will verify the correctness of all the modules in your code.

Optionally, you are encouraged to make use of a Web API. They are heavily used in many Swift applications and this is a good opportunity to learn using them. You can use APIs to implement various features such as authentication (e.g., login using Facebook), pushing information (e.g., posting messages or uploading data), or retrieving information (e.g., read messages or retrieve images).

Grading

There will be four code "check-ins" across the semester where your group will present your progress (i.e., Weeks 8, 10, 12, and 15). Check-ins will always be done in our Thursday class. This will help ensure that you complete your project in time. All groups will present their final project in the last week of class (Week 16).

Project check in

Although you will work in groups, each person will be graded individually according to their contribution to the project. I will use the following rubric to grade each member of the group.

Progress toward final project (50%)
I will measure progress by the number of features that your team has implemented while considering the complexity of the feature.

Design Document Refinement (5%)
During the meeting I will assess whether the team reflected on their design document and adjusted it according to their progress. The design document does not need to change if the group thinks they are still on track to complete the project as seen in the latest version of the document.

Contribution (30%)
I will measure each member's contribution by looking at the code, unit tests, or documentation committed into the GItHub repository. As this is my only way to check for contribution, you must make sure that you commit code using your own GitHub account. Avoid pushing code for someone else.

Peer evaluation (10%)
You will rate each other's contribution to the project and I will use this as part of the check-in grade.

Reflection (5%)
I will check whether you reflected on your team's progress and your plans for improvement.

Final project

At the end of the semester, you will be presenting your project to the class. I will evaluate your final project according to the following rubric.

Functionality and design (40%)
I assess your code's functionality by checking the correctness of the features you implemented. I will also check the appropriateness of your selected object-oriented designs and Swift components to implement a particular functionality.

Unit tests (10%)
I will check the correctness of your unit tests and it's coverage in testing all components of your code.

Readability (10%)
I will assess your code's readability by checking whether your code conforms to Swift's style guide (https://swift.org/documentation/api-design-guidelines/). I will check for comments that describe your classes, data members, and member functions as well as complex implementations.

Contribution (30%)

I will measure each member's contribution by looking at the code, unit tests, or documentation committed into the GItHub repository. As this is my only way to check for contribution, you must make sure that you commit code using your own GitHub account. Avoid pushing code for someone else.

Peer evaluation (10%)
You will rate each other's contribution to the project.

Final project grade

Each project check in will contribute 10% to your final project grade (i.e., 4 x 10% = 40%), while your final project will contribute 60% to your final project grade. Adding up all check-ins and final project results in a total grade of 100%.

Submission

Each team will have a GitHub repository where you will submit your code. You are expected to use GitHub's functionalities for adding, committing, and pushing code to the repository. Additionally you are encouraged to create branches and make pull requests to make it easy to contribute to the project.

I will use the latest commit to the master/main branch of the repository for the project check-ins and final project. You are also expected to update your design document as much as possible so it is easy to track your project's functionalities.