Comp 55 Application Development

JumpStart

By

Drew Overgaard, Isabelle Tran, Josiah Yoshimura,

Kyle Hartman

<https://basecamp.com/2707759/projects/11638120>

March 24th, 2016

Individual Contributions Breakdown

Drew Overgaard:

Progress Report and Plan of Work (70%)

User Interface Design and Implementation (100%)

Algorithms and Data Structures (40%)

UML Class Diagram (10%)

Interactions Diagram (20%)

Isabelle Tran:

Progress Report and Plan of Work (30%)

UML Class Diagram (10%)

Interactions Diagram (60%)

Algorithms and Data Structures (40%)

Josiah Yoshimura:

UML Class Diagram (70%)

Algorithms and Data Structures (10%)

Kyle Hartman:

UML Class Diagram (10%)

System Architecture and System Design (100%)

Algorithms and Data Structures (10%)

Table of Contents, Cover Page and Individual Contribution Breakdown (100%)

Interactions Diagram (20%)

Table of Contents

UML Class Design………………………………………………………………………………...3

Description of Classes and Main Relationships………………………………..………………….4

Interactions Diagram………………………………………………………………………………6

System Architecture and System Design………………………………………………………….7

Algorithms and Data Structures…………………………………………………………………...8

User Interface and Implementation………………………………………………………………..9

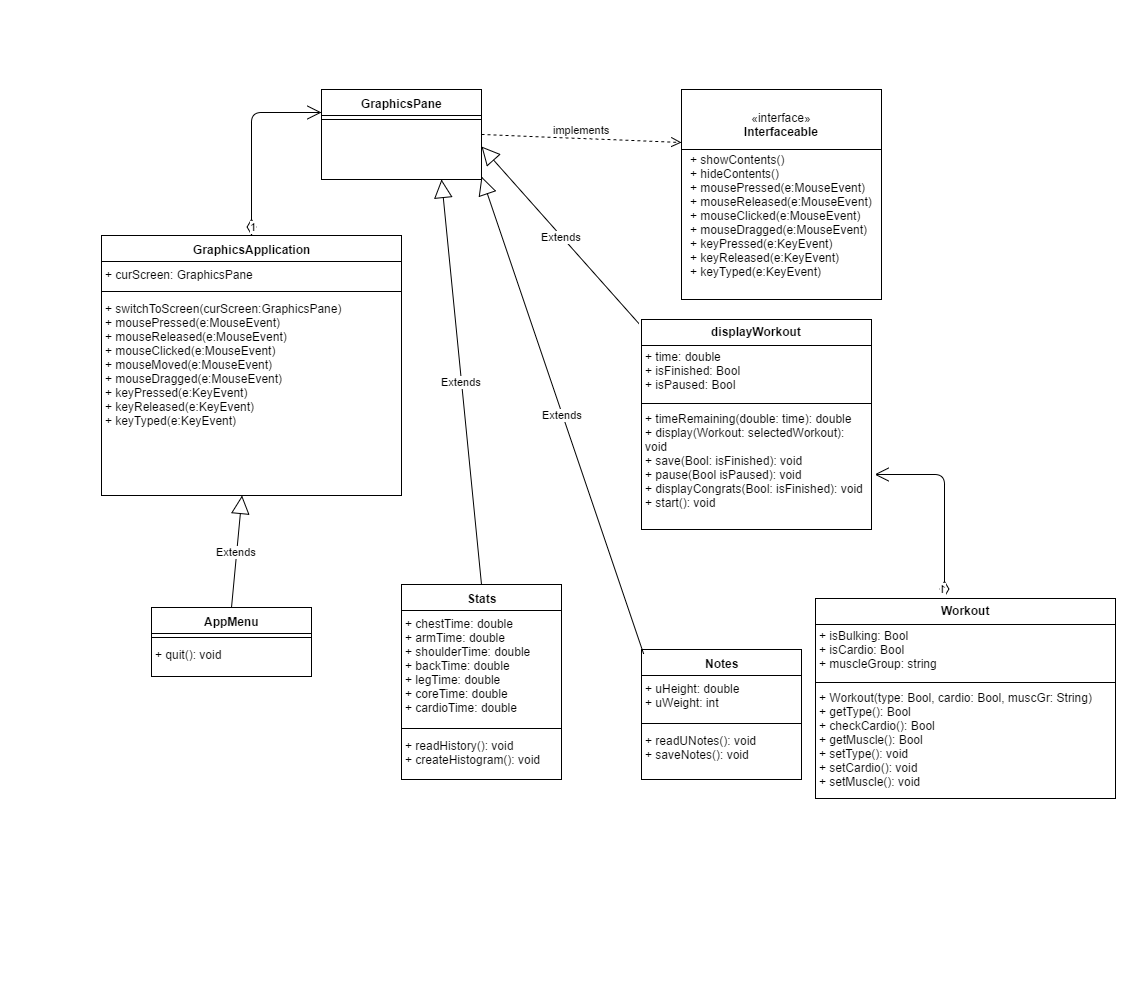
Progress Report…………………………………………………………………………………..10

Plan of Work……………………………………………….……………………………….........11

Breakdown of Responsibilities…………………………………………………………………..12

References……………………………………………………………………………………….13

UML Diagram

Class Diagram

Description of Classes and Main Relationships

**GraphicsApplication**

This will allow us to switch between the GraphicsPanes. Each GrapicsPane will be one of our program features. So Workout, Stats, and Notes will all extend from GraphicsPane. I think the only thing we really have to remember is to save progress before switching or quitting.

**GraphicsPane**

This will allow us to display the information contained in our classes. Workout will need to display pictures of the various exercises. Stats might be more difficult as the histogram will have to be built from a file containing the information on the user. Notes should be easy as it will be mainly text.

**displayWorkout**

This is the core of the program. Images will need to be opened and displayed using the GraphicsPane methods. Additionally, some information will need to be saved in a text file for stats. A Workout object will be passed to the displayWorkout method, and whichever workout is passed will determine what workout is displayed.

**WorkoutSpecs**

This is the workout layout. It has some Booleans to determine what type of workout it is. It is used by displayWorkout. Each specification corresponds to a specific workout’s images.

**Stats**

Stats will open the file that is saved from workout in order to show a histogram to the user. The file will need to be closed each time the user switches or quits. We’re not sure yet, but it might be useful to make a histogram class.

**Notes**

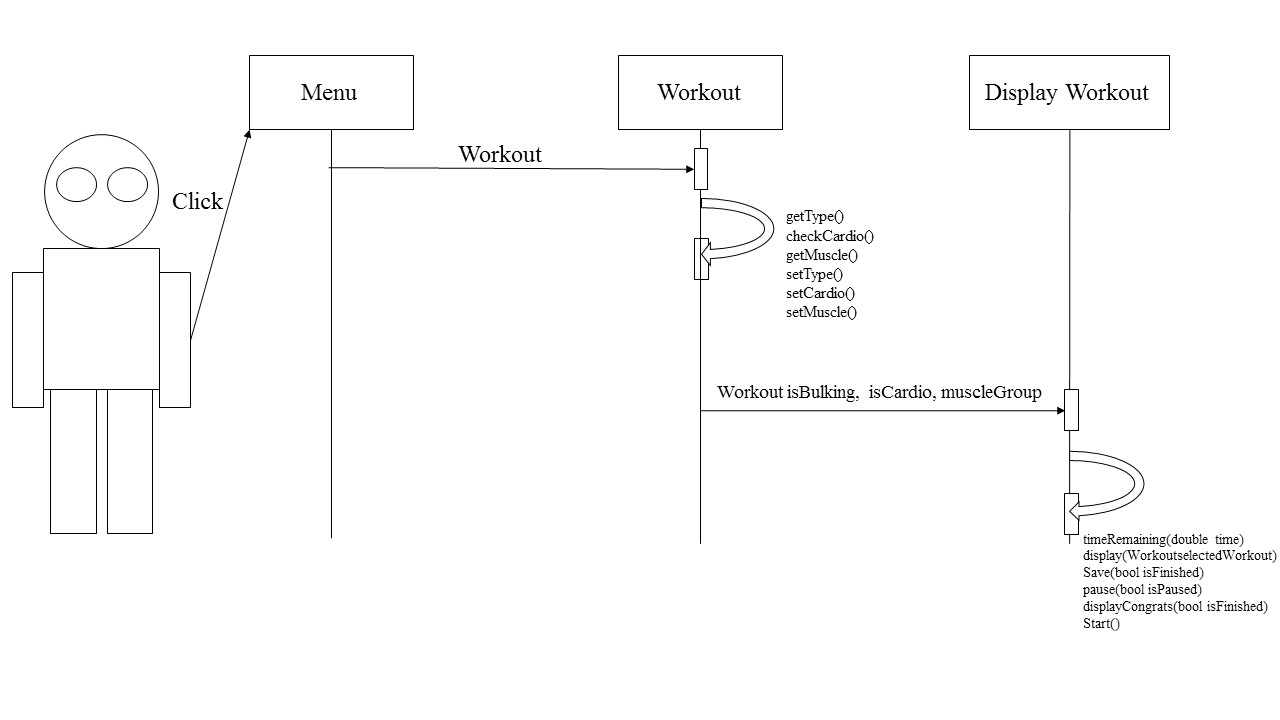
Notes will have its own file for notetaking that is opened when the user switched to this feature. The file will need to be saved and closed each time the user switches or quits.

**AppMenu**

This should have the different features displayed. Its unique option is quit, which lets the user exit the application.

Interactions Diagram

One of the design principles we will use is DRY which is don’t repeat yourself, we will have some code which will be similar in nature but we do not want to just copy and paste code to make it easier on us. We will also use KISS which is ‘keep it simple stupid’, we do not want to have really complex code and so we will try and keep our code as simple as possible and easy to understand. Another design principle we will be using is SRP which is the ‘Single Responsibility Principle’. We will be making sure that our class’s will be in charge of one thing only, for example are stats and notes class will be in charge of all of the statistics and all of the note taking, respectively.



System Architecture and System Design

1. Mapping Subsystems to Hardware

Our system only needs an Operating system to run, the user will not have to access any kind of web browser or web server to use or receive any of the information that will be in our program.

1. Network Protocol

Our system will be run on a single machine and will not need any communication protocol.

Hardware Requirements

1. Our System will need a screen display and some disk storage to store the program and all of the user information. We will have a color display and for resolution we would like to detect the computer systems native resolution and have our program fit to the screen. We would like to also have a minimum of 1 Gigabyte of hard disk space to store the files the program along with its pictures and user information.

Algorithms and Data Structures

1. Yes, our system will need an algorithm to calculate user progress and display a statistical analysis of their progress. We will store the user progress including, number of workouts completed, which muscle groups the workouts completed were in, total time exercised, and the number of workouts not completed. Based on this data we will create a bar graph to show users which areas they can improve upon and which areas they are doing well in

File will contain type of workout, muscle group, and amount of time completed (0-7 min) or other for cardio

User does a workout

With these values the histogram will update with each workout

Statistics class will have variables for each muscle group and continue adding amount of time (ex: chestProgress += chestProgress)

Workout class writes progress to a file

Statistics class will read the file after every time it has been written to by workout class

1. Yes, we will need to use an array to store the workouts for each muscle group. We will also need an array to store the images for workouts. We should use an array because it will be less complex to implement and we won’t need to edit the data, remove, or move items in the array, which means we won’t need to use a more complex data structure such as a linked list.

User Interface Design and Implementation

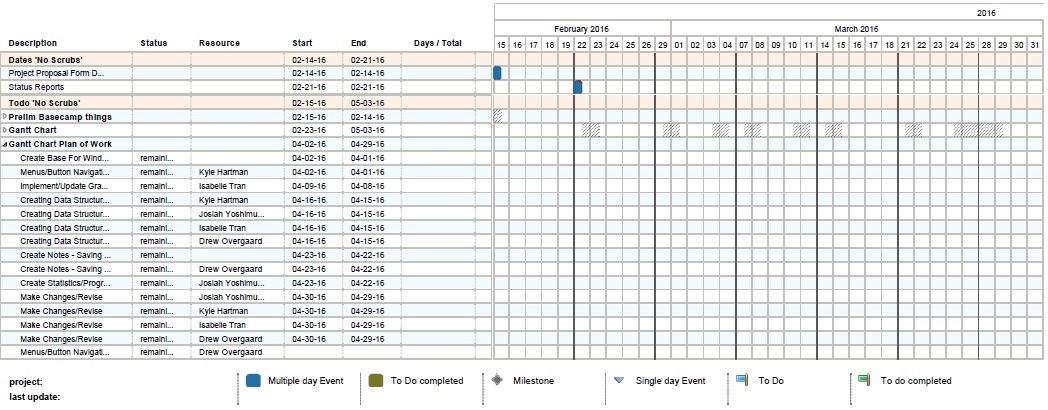
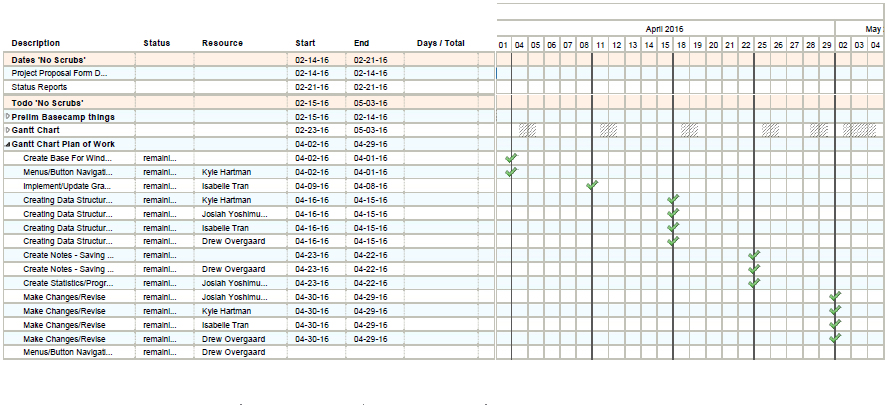
1. We have made two changes to the initial screen mock-ups that we created in report #1. Firstly, we removed the edit and save buttons on the notes screen. We removed these buttons because we decided to allow the user to input text into the textbox at any time without having to press a button and the changes they make will automatically be saved when they quit notes or move to a different window within the application. The second change that we’ve made to the initial screen mock-ups was the implementation of a drop down menu instead of a quit button on every window besides the main menu. This drop down menu will allow the user to switch from one window of the application to another instead of previously only being able to quit to the main menu. This is more efficient and will save the user time.
2. One thing we learned from demonstrating our paper prototype is that the quit to main menu button was cumbersome. A suggestion we received during our demonstration was to implement a drop down menu instead that would allow the user to switch to any of the windows in the application. This allows the user to switch to a different menu without having to go back to the main menu. We also received suggestions to color code our graphs in the statistics window to differentiate from workouts of the same type. The addition of a congratulations message when the user completes a workout.

Progress Report and Plan of Work

1. Progress Report

As of the week of March 21, we have not implemented any of our use cases. Our interface will provide a menu allowing the user to select a feature of the program. The first feature we want to implement is the workout portion to start as our foundation. To begin, we will construct a base class called WorkOut. Within the class we can use functions to set the difficulty, set muscle group to work out, create a Boolean for cutting and bulking as well as cardio which will additionally require a cardio class. We will also need a set and get time function to allow users to only complete as much of the 7-minute workout as they wish. The time completed of most recent workout will be stored in a function which will display itself to the user the next time they open the application.

To set muscle group within our WorkOut class we will enumerate each muscle group (chest, arms, shoulders, back, legs, and core). We will use the ACM library to access the workout routine images that correspond to each workout enumeration. The timer will start at 7-minutes and count down and be displayed to the user. After we have completed the WorkOut class we will work on creating a Statistics class that will heavily work with functions in the WorkOut class which will need to get the time in order to create the progress histogram.

B.) Plan of Work

C) Breakdown of Responsibilities

Josiah – Responsible for cresting the statistics class and testing it.

Drew – Responsible for creating the notes class and testing it. Will help with coordinating the integration of the program.

Kyle- Responsible for creating the notes class and testing it, coordinating the integration of the program and responsible for the user interface.

Isabelle – Responsible for creating the workout class and testing it.

Every group member will test each other’s classes in order to find bugs that may have been missed during programming. We will also all test the final integrated system.

References

http://www.theguardian.com/lifeandstyle/2011/jan/07/fitness-regime-stay-motivated

Dixon, Rachel. “How to Stick to a Fitness Regime.” *The Guardian* 7 Jan. 2011. Web. 23 Mar. 2016.

http://www.bodybuilding.com/

“Bodybuilding.com.” Bodybuilding.com, n.d. Web. 23 Mar. 2016.

https://human.biodigital.com/signin.html

“BioDigital human: Explore the body in 3D!” *BioDigital*. 2016. Web. 23 Mar. 2016.

http://www.tiim.info/the\_nxt\_project/web/design.php

Kahrimanovic, Almir. *Design*. 2016. Web. 23 Mar. 2016.

<http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/bell/>

Bell, Donald, et al. *The class diagram*. 15 Sept. 2004. Web. 23 Mar. 2016.