CREATIVE COMPUTING PROJECT

Version 3.1
Computing
COMP140

Brian McDonald

Introduction

In this assignment, you are required to **create** a game or other playful experience which **interfaces** with a custom system.

Experimentation, ingenuity, and creativity are at the heart of everything that creative developers do. To this end, building your own custom interface is the perfect place to exercise these characteristics. However, you will also gain invaluable exposure to working with computer hardware and embedded systems. In recent years, there has been considerable growth in the development of new fabrication technologies, such as 3D printers. In addition, electronics, from primitive transistors to complex computer chips, have all become much cheaper. Accessibility to these tools has, therefore, unveiled an unprecedented opportunity to invent and innovate in this space. Increasingly, app developers are augmenting mobile software with new wearable devices. With the advent of VR, AR and XR and the demand for augmenting reality through new sensing technology, developers are increasingly expected to consider new interfaces in the design of entertainment products

"As soon as we started programming, we found out to our surprise that it wasn't as easy to get programs right as we had thought. Debugging had to be discovered. I can remember the exact instant when I realized that a large part of my life from then on was going to be spent in finding mistakes in my own programs."

— Maurice Wilkes

"C++ is history repeated as tragedy. Java is history repeated as farce."

— Scott McKay

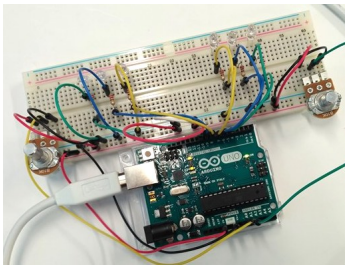
This assignment is formed of several parts:

(A) **Write**, as an **individual**, prototype game or other playful experience (see the contracts below) which interfaces with a custom controller or sensing system
(B) **Write**, as an **individual**, a final version of your project
(C) **Present**, as an **individual**, a practical demo of the computer program to your tutor that will:
   i. **demonstrate** your academic integrity;
   ii. as well as **demonstrate** your **individual** programming & hardware knowledge.

Contracts

To give you a wide brief and accommodate a broad range of applications you will **choose one** of the following two contracts:

(A) You have been commissioned to develop a **game controller and game** for a **keynote games expo event** that pushes the boundaries of what is possible with common domestic objects and gameplay. Create a game that utilises everyday objects as interfaces to a game world.
(B) You have been commissioned to develop an **interactive exhibit** for the **Natural History Musueum** that explores the interaction between the natural word and the digital. Create a digital environment that is influenced and effected by real world sensors and vice versa.



Pong Controller example from session 18-19

As this module is about pushing your creativity in creating an experience which blends a physical controller and a digital application, we are going to enforce the following **constraints** and make some **recommendations**

Constraints

(A) You should not recreate a twin stick joypad
(B) You should not build a custom joystick/fighting stick
(C) You should avoid buttons, other sensors could be used instead!
(D) You should not submit a bare bread board, please build a case or housing around the controller or sensing system
(E) You should have a traditional control scheme (keyboard, mouse, keyboard), so your experience functionality can be quickly tested
(F) The main **readme.md** should be updated with all sources used to create the project

Recommendations

(A) You should focus on the interaction between the control system and the digital environment
(B) You should consider a 2D interface to keep the experience simple
(C) You should avoid using 3D printed elements in your controller
(D) You should use found objects/recycled objects for your controller

Assignment Setup

Fork the GitHub repository at:

https://gamesgit.falmouth.ac.uk/scm/ mattwatkins/comp140-assignment-1.git

Use the existing directory structure, the Unity Project should be placed inside the **Unity Project** folder and the Arduino project files should be placed inside **Arduino Project** folder. Ensure that you maintain the readme.md file.

Part A

Part A consists of a **single formative submission**. You should demonstrate your progress to a tutor in the timetabled session in **Week 6**

You will receive **immediate informal** feedback from your **tutor**.

Part B

Part B consists of a **single summative submissions**. You should download your project from GitHub, and submit a **zip file** which contains the following

(A) The Unity Project including all source code and assets
(B) The Arduino Project
(C) Two images of the controller: one of the wiring and another with case/housing
(D) Video footage of the controller and game/experience being played
(E) readme.md with references to all sources and assets used in the project

Part C

To complete Part C, implement the final changes to your project. Prepare a practical demonstration of the project. Ensure that the source code and related assets are pushed to GitHub and a pull request is made prior to the scheduled viva session. Then, attend the scheduled viva session via Mircosoft Teams.

You will receive **immediate informal** feedback from your **tutor**.

Additional Guidance

Nobody learns in a vacuum: you are allowed, and indeed encouraged, to discuss your work with your peers. However you must be very careful to avoid falling into **academic misconduct**, in particular **plagiarism**. If any part of your solution is **not your own individual work**, you must make this as clear as possible in your submission, for example in source code comments.

FAQ

- **What is the deadline for this assignment?**
  Each worksheet has its own formative deadline, specified on that worksheet and also communicated in class. Falmouth University policy states that summative deadlines must only be specified on the MyFalmouth system.
- **What should I do to seek help?**
  You can email your tutor for informal clarifications. For informal feedback, make a pull request on GitHub.
- **How will I receive feedback on my work?**
  You will be given verbal feedback on your work during the session in which it is marked. If you require more in-depth feedback or discussion, please book an appointment with your tutor.
- **Is this a mistake?**
  If you have discovered an issue with the brief itself, the source files are available at:

  https://github.com/Falmouth-Games-Academy/bsc-assignment-briefs
  .
  Please make a pull request and comment accordingly.
- **What coding standards are we using on this assignment?**
  We are using the Microsoft's coding styles for C#

  https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions

Additional Resources

- Wilkinson, K. and Petrich, M. (2014) The Art of Tinkering: Meet 150 Markers Working at the Intersection of Art, Science & Technology. Weldon Owen: London.
- Alicia Gibb. Building Open Source Hardware: DIY Manufacturing for Hackers and Makers. Addison Wesley, 2014.
- Jeremy Blum. Exploring Arduino: Tools and Techniques for Engineering Wizardry. John Wiley, 2013.
- Kelly, K. (2014) Cool Tools: A Catalogue of Possibilities. Cool Tools.
- https://www.sitepoint.com/heuristic-evaluation-guide/
- https://www.usability.gov/how-to-and-tools/methods/heuristic-evaluation.html
- https://github.com/arduino/Arduino/blob/master/.gitignore
- https://gitignore.io/

Marking Rubric

All submissions and assessment criteria for this assignment are individual.

Criterion	Weight	Near Pass	Adequate	Competent	Very Good	Excellent	Outstanding
Basic Competency Threshold	30%	At least one part is missing or is inadequate.	Adequate ability to generate ideas, problem solving, concepts, technical competency and proposals in response to set briefs and/or self-initiated activity. The work demonstrates an adequate, ethically informed, real-world experience of industry/business environments and markets. Enough work is available to hold a meaningful discussion. Clear evidence of programming knowledge. Constraints followed No breaches of academic integrity.				
PROCESS: Sophistication of Code	15%	No insight into the appropriate use of programming constructs is evident from the source code. No attempt to structure the program (e.g. one monolithic function).	Little insight into the appropriate use of programming constructs is evident from the source code. The program structure is poor.	Some insight into the appropriate use of programming constructs is evident from the source code. The program structure is adequate.	Much insight into the appropriate use of programming constructs is evident from the source code. The program structure is appropriate.	Considerable insight into the appropriate use of programming constructs is evident from the source code. The program structure is effective. There is high cohesion and low coupling.	Significant insight into the appropriate use of programming constructs is evident from the source code. The program structure is very effective. There is high cohesion and low coupling.
PROCESS: Maintainability of Code	10%	There are no comments in the source code, or comments are misleading. Most variable names are unclear or inappropriate. Code formatting hinders readability.	The source code is only sporadically commented, or comments are unclear. Some identifier names are unclear or inappropriate. Code formatting is inconsistent or does not aid readability.	The source code is somewhat well commented. Some identifier names are descriptive and appropriate. An attempt has been made to adhere to Microsoft's formatting style. There is little obvious duplication of code or of literal values.	The source code is reasonably well commented. Most identifier names are descriptive and appropriate. Most code adheres to the Microsoft's formatting style. There is almost no obvious duplication of code or of literal values.	The source code is reasonably well commented in the Microsoft's style Almost all identifier names are descriptive and appropriate. Almost all code adheres to the Microsoft's formatting style. There is no obvious duplication of code or of literal values. Some literal values can be easily changed in the Unity Editor.	The source code is very well commented, with Python doc-strings. All identifier names are descriptive and appropriate. All source code adheres to the Microsoft's formatting style. There is no obvious duplication of code or of literal values. Most literal values are, where appropriate, easily changed in the Unity Editor.
PROCESS: Functionality of Physical Prototype	10%	No physical prototype, or the prototype is not functional.	The physical prototype has a little functionality. There are critical technical and/or constructional flaws.	The physical prototype has some functionality. There are major technical and/or constructional flaws.	The physical prototype has much functionality. If any, only minor technical and/or constructional flaws.	The physical prototype has considerable functionality. If any, only minor technical and/or constructional flaws.	The physical prototype has significant functionality. If any, only superficial technical and/or constructional flaws.
PROCESS: Electronics Sophistication	5%	Solution lacks even basic use of electronic components.	Solution has some sophistication in terms of electronics. Little insight insight into electronic circuits.	Solution has some sophistication in terms of electronics. Some insight into electronic circuits.	Solution leverages electronic components with much effectiveness. Much insight into electronic circuits.	Solution leverages electronic components with much effectiveness. Considerable insight into electronic circuits.	Solution leverages electronic components with much effectiveness. Significant insight into electronic circuits.

Criterion	Weight	Near Pass	Adequate	Competent	Very Good	Excellent	Outstanding
PROCESS:Physical Form Factor Sophistication	10%	No physical prototype, or it is limited to a breadboard without housing.	Physical form factor has a little sophistication. Little insight into human-computer interaction.	Physical form factor has some sophistication. Some insight into human-computer interaction.	Physical form factor has much sophistication and sturdiness. Much insight into human-computer interaction.	Physical form factor has much sophistication and sturdiness. Considerable insight into human-computer interaction.	Physical form factor has considerable sophistication and sturdiness. Significant insight into human-computer interaction. Controller has both practical and aesthetic value.
INDUSTRY: Creative Response to Brief	10%	No creativity. The work is a clone of an existing work with mere cosmetic alterations.	Little creativity. The work is derivative of existing works, with only minor alterations.	Some creativity. The work is derivative of existing works, demonstrating little divergent and/or subversive thinking.	Much creativity. The work is somewhat novel, demonstrating some divergent and/or subversive thinking.	Considerable creativity. The work is novel, demonstrating significant divergent and/or subversive thinking.	Significant creativity. The work is highly original, with strong evidence of divergent and/or subversive thinking.
INDUSTRY: Use of Version Control	10%	Version control (e.g. GitHub) has not been used.	Source code has been checked into version control (e.g. GitHub).	Source code has been checked into version control (e.g. GitHub) at least once per week. Sensible commit messages are present.	Source code has been checked into version control (e.g. GitHub) several times per week. Commit messages are clear, concise and relevant. There is evidence of somewhat meaningful engagement with peers (e.g. code review). Comments to peers are somewhat constructive and provide some insight.	Source code has been checked into version control (e.g. GitHub) several times per week. Commit messages are clear, concise and relevant. There is evidence of meaningful engagement with peers (e.g. code review). Comments to peers are reasonably constructive and provide much insight.	Source code has been checked into version control (e.g. GitHub) many times per week. Commit messages are clear, concise and relevant. There is evidence of effective engagement with peers (e.g. code review). Comments to peers are reasonably constructive and provide considerable insight.