

CODING TASK II: GAME COMPONENT

Version 1.0
BSc Computing for Games

Dr Michael Scott

Introduction

In this assignment, you are required to propose and write a game component for an existing game. You will then present a demonstration of your work.

Games are often comprised of a rich architecture incorporating many components. As such, coding tasks in the games industry may require you to develop bespoke code on a particular aspect of a game. Examples include: level generators; event triggers; conflict mechanics; character behaviours; physics; and so on. Through this project, you will become acquainted with techniques and methods that help you to work effectively to write such components; thereby, gaining knowledge of coding and its various aspects.

"Engage with the community and support each other. This is important. Upload your code to GitHub and receive feedback from experienced peers. Review your peers' work yourself and really consider what 'quality' actually means. Debate, argue, and question others about it— an open and sustained discourse is an excellent way for all to learn!"

This assignment is formed of several parts:

- (a) **Write** a brief proposal that will:
 - i. **identify** a component of game architecture;
 - ii. **describe** what will be created;
 - iii. **illustrate** the key user stories;
 - iv. and **identify** the existing game the component will integrate into;
- (b) **Write** a draft computer program that will:
 - i. **implement** the game component;
 - ii. and **address** the requirements highlighted in the proposal;
- (c) **Write** a final computer program that will:
 - i. **revise** any issues raised by your tutor or your peers;
- (d) **Present** an executable version of your final program.

Note: All submissions must be clearly distinctive. Members of the same development group must **not** target the same component.

Assignment Setup

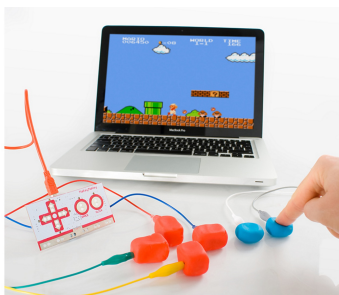
This assignment is a **programming task** and so regular pushes to a GitHub repository is expected. Fork the GitHub repository at the following URL:

<https://github.com/Falmouth-Games-Academy/comp110-coding-task-2>

Use the existing directory structure and, as required, extend this structure with sub-directories. Ensure that you maintain the `readme.md` file. Optionally, a `changelog.md` may be created.

Modify the `.gitignore` to the defaults for either **Python** or **C++** depending on which language you wish to use for this assignment. Please, also ensure that you add IDE-specific files and folders to `.gitignore`.

Setup a Trello board to manage this assignment.



The MaKey MaKey allows a multitude of materials to be used to create videogame controllers.

Part A

Part A consists of a **single formative submission**. This work is **individual** and will be assessed on a **threshold** basis. The following criteria are used to determine a pass or fail:

- (a) Submission is timely;
- (b) Proposed game component is relevant to the existing game;
- (c) Coding task is non-trivial and achievable.

To complete Part A, write your proposal in the `readme.md` document and add user stories to the backlog of the Trello board. Show these to your tutor. If acceptable, this will be signed-off.

You will receive immediate **informal feedback**.

Part B

Part B is a **single formative submission**. This work is **individual** and will be assessed on a **threshold** basis. The following criteria are used to determine a pass or fail:

- (a) Submission is timely;
- (b) Enough work is available to conduct a meaningful review;
- (c) A broadly appropriate review of a peer's work is submitted.

To complete Part B, prepare a draft version of the source code for your game component. This may or may not be integrated into the existing game. Ensure that the source code and related assets are pushed to GitHub and a pull request is made prior to the scheduled code review session. Then, attend the scheduled code review session.

You will receive **peer feedback** within 3 working days after the code review session.

Part C

Part C is a **single summative submission**. This work is **individual** and will be assessed on a **criterion-referenced** basis. The following criteria are used to allocate marks:

- (a) Appropriateness of Requirements;
- (b) Appropriateness of Design;
- (c) Functional Coherence of Executable Solution;
- (d) Maintainability of Source Code;
- (e) Sophistication of Source Code;
- (f) Professional Practice;

To complete Part C, revise the source code for your game component based on the feedback you have received. Integrate your source code into the existing game. Then, upload the source code and related assets to the LearningSpace. Please note, the LearningSpace will only accept a single `.zip` file (and the `.rar` format must not be used).

You will receive **formal feedback** three weeks after the final deadline.

Part D

Part D is **not grade-bearing**; however, **participation is mandatory**. Failure to attend will result in a grade capped at 40% (D-).

To complete Part D, attend the scheduled demo session. Ensure that a working executable demo of your work has been pushed into GitHub prior to this session.

You will receive immediate **informal feedback**.

Additional Guidance

Do not begin programming the game component until your tutor has reviewed your proposal. It is important that the main requirements are firmly specified and are not too broad. This will help ensure that you do not overburden yourself.

Please remember to commit frequently and to push your source code and related assets to the GitHub repository. This will make it easier for you to maintain a backup of your work. It will also help you to measure your productivity.

Poor planning and poor time management can have a significant impact on this assignment. It is very easy to underestimate how much work is involved in first learning programming concepts and then actually applying them in order to write a computer program. As some of you may have already discovered, programming is quite unlike other subjects in that it cannot be “crammed” into a last minute deluge. Sustain a steady pace across the duration of the course. Do a little programming every day, if you can!

In order to obtain higher marks, you are being expected to focus on the quality of your code, rather than just its functionality. This can be supported through disciplined working practices. Such practice may include regular peer review. It may also include the use of lint tools and reference to style guidelines.

If you encounter difficulty, please email your tutor for support. For problems relating specifically to source code, please make a pull request on GitHub.

FAQ

- **What is the deadline for this assignment?**

Falmouth University policy states that deadlines must only be specified on LearningSpace. Please examine the assignment area where you located this document.

- **How can I create a .zip of my source code?**

See: <http://i.stack.imgur.com/NhsQe.png>.

- **What should I do to seek help?**

You can email your tutor for informal clarifications. For informal feedback, make a pull request on GitHub.

- **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.

Additional Resources

- To be advised by your tutor after submission of the proposal.

Marking Rubric

Criterion	Weight	F (0 – 39)	D (40 – 49)	C (50 – 59)	B (60 – 69)	A (70 – 79)	A* (80 – 100)
Satisfactory Preparation of Proposal	5%	The proposal is inappropriate and/or is late.					The proposal has been signed-off by your tutor by the deadline.
Satisfactory Completion of Peer-Review Tasks	5%	No work was submitted for peer-review and/or no peer-review has been submitted and/or either is late.					Work submitted for peer-review on time and reviews of peers' work submitted on time.
Appropriateness of Requirements	5%	No user stories are provided.	Few user stories are appropriately formatted, distinguishable, and easily measured.	Some user stories are appropriately formatted, distinguishable, and easily measured.	Most user stories are appropriately formatted, distinguishable, and easily measured. The scope and relevance of all requirements is appropriate.	Nearly all user stories are appropriately formatted, distinguishable, and easily measured. The scope and relevance of all requirements is appropriate.	All user stories are appropriately formatted, distinguishable, and easily measured. The scope and relevance of all requirements is appropriate.
Appropriateness of Design	10%	No design is presented.	The design is very flawed and/or very poorly described.	The design is flawed and/or poorly described.	The design is acceptable and adequately described.	The design is sound and well described.	The design is exceptional and very well described.
Functional Coherence	15%	The component is non-functional.	Few requirements have been met. There are many obvious bugs.	Some requirements have been met. There are some obvious bugs.	Many requirements have been met. There are few obvious bugs.	The game component is fit-for-purpose. There are almost no obvious bugs.	The game component is fit-for-purpose. There are no obvious bugs.
Sophistication	25%	No insight into the appropriate use of programming constructs is evident from the source code.	Little insight into the appropriate use of programming constructs is evident from the source code.	Some insight into the appropriate use of programming constructs is evident from the source code.	Much insight into the appropriate use of programming constructs is evident from the source code. The program is structured appropriately.	Significant insight into the appropriate use of programming constructs is evident from the source code. The program is structured effectively, such that there is high cohesion and low coupling.	Exemplary insight into the appropriate use of programming constructs is evident from the source code. The program is structured very effectively, such that there is very high cohesion and very low coupling.
Maintainability	25%	The source code cannot be maintained.	There are many problems which affect the maintainability of the source code.	There are some problems which affect the maintainability of the source code. Some clear and appropriate comments are present.	There are few problems which affect the maintainability of the source code. Many clear and appropriate comments are present.	There are almost no problems which affect the maintainability of the source code. Source code is well commented. Doc strings (or equivalent) are provided.	There are no problems which affect the maintainability of the source code. Source code is exceptionally well commented. Appropriate doc strings (or equivalent) are provided.
Professional Practice	10%	GitHub has not been used.	Source code and assets have been checked into the repository only just before a deadline.	Source code and assets have seldom been checked into the repository.	Source code and assets have regularly been checked into the repository. An attempt has been made to document the project using <code>readme.md</code> and <code>changelog.md</code> .	Source code and assets have regularly been checked into the repository. The first check-in to the repository is in the first half of the semester. The project is appropriately documented using <code>readme.md</code> and <code>changelog.md</code> . There is evidence of some engagement with the Falmouth Games Academy community (e.g. reviewing peers' pull requests).	Source code and assets have regularly been checked into the repository. The first check-in to the repository is in the first quarter of the semester. The project is exemplary documented using <code>readme.md</code> and <code>changelog.md</code> . There is evidence of much engagement with the Falmouth Games Academy community (e.g. reviewing peers' pull requests).