PRE-PRODUCTION TASKS & PITCHES

Version 2.1
BSc Computing for Games
COMP230

Dr Michael Scott

Introduction

You will prepare a pre-production prototype of a game-related product. You will work with peers drawn from other courses in the Games Academy in a team. You will be involved in ideation, but will primarily adopt a role as a 'technical specialist' to develop game components and facilitate play-testing.

Game development is a diverse ecology. Tackling the challenges of multidisciplinarity is important—notably, appreciating the roles of computing experts.

This assignment is formed of several parts:

(A) **Present**, as an **individual**, a 10-minute 'world pitch' that will:
   i. **describe** an original world;
   ii. and **reveal affordances** to be leveraged as the context for a game.

(B) **Present**, as a **group**, a 20-minute 'game concept pitch' that will:
   i. **extend** the world that you devised and/or those devised by peers;
   ii. **illustrate** the key concept of the game;
   iii. and **justify** a business case for the game.

(C) **Implement**, as a **group**, a pre-production prototype that will:
   i. **illustrate** the core gameplay and unique selling points;
   ii. and **facilitate** play testing.

(D) **Prepare**, as a **group**, a video of the pre-production prototype that will:
   i. **highlights** the key features of the game;
   ii. and **responds** to issues identified during play-testing.

(E) **Present**, as an **individual**, a 5-minute 'business case' that will:
   i. **clarify** the product and its design;
   ii. and **demonstrate** the final pre-production prototype.

(F) **Demonstrate**, as a **group**, the pre-production prototype:
   i. **demonstrate** your academic integrity;
   ii. as well as **showcase** your individual contributions.

"The first 90 percent of the code accounts for the first 90 percent of the development time.

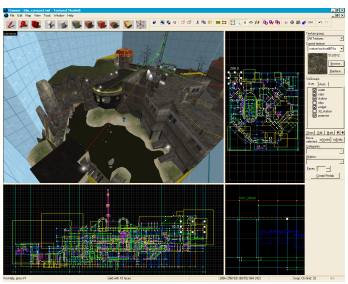
"The remaining 10 percent of the code accounts for the other 90 percent of the development time."

— Tom Cargill

"Hofstadter's Law:

"It always takes longer than you expect, even when you take into account Hofstadter's Law."

— Douglas Hofstadter



Don't make *Hammer*, but don't under-estimate the value of tool programming.

Assignment Setup

This assignment is a **product development task**.

You will need to devise an approach to communication, task management, and version control with your peers. Strive to be as adaptable as possible and heed the advise of your product owner.

You may use any languages, tools, and development environment available within the Games Academy.

Part A

Part A consists of a **single summative submission**. This work is **individual** and will be assessed on a **criterion-referenced** basis. Please refer to the marking rubric at the end of the brief for details on the criteria.

To complete Part A, devise an original world. Then, prepare a slide-deck and handout that describe that world. You will either: (i) present these as a live 10-minute presentation to tutors; or (ii) record a 10-minute video. Attend the relevant sessions and then upload your materials to the LearningSpace.

You will receive **formal feedback** from your **tutor** 3-weeks after the deadline.

Part B

Part B consists of a **single summative submission**. This work is **collaborative** and will be assessed on a **criterion-referenced** basis. Please refer to the marking rubric at the end of the brief for details on the criteria.

To complete Part B, devise an game concept with your peers. You are expected to build upon the worlds of one or more members of your team, amalgamating ideas where neccessary. Prepare a slide-deck and handout that describe your game concept. You will either: (i) present these as a live 10-minute presentation to tutors; or (ii) record a 10-minute video. Attend the relevant sessions and then upload your materials to the LearningSpace.

You will receive **formal feedback** from your **tutor** 3-weeks after the deadline.

Part C

Part C is a **multiple formative submissions**. This work is **collaborative** and will be assessed on a **threshold** basis. The following criteria are used to determine a pass or fail:

(a) Playable builds are made available in a timely manner;

(b) Enough work is available to conduct a meaningful review.

To complete Part C, prepare draft versions of the pre-production prototype ahead of the **sprint review meetings** with your product owner. Ensure that the source code and related assets are made available prior to **EACH** scheduled meeting. Then, attend the scheduled meetings.

You will receive immediate **informal feedback** from your **product owner**.

Part D

Part D is a **single summative submission**. This work is **collaborative** and will be assessed on a **criterion-referenced** basis. Please refer to the marking rubric at the end of the brief for details on the criteria.

To complete Part D, revise the pre-production prototype based on the feedback you have received through play-testing, crits, and review by the product owner. Strive to finish any incomplete features. Then record a a 5-minute video of the gameplay implemented in the pre-production prototype. The video must showcase the core elements of the game and draw attention to design changes derived through play-testing. Fading through a few different versions of the build is advised. Upload the video to the LearningSpace.

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

Part E

Part E 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) The title and high concept of the product

(b) The target audience and market

(c) The concept's unique selling points and how they distinguish it from competitors

(d) The design's technical and production feasibility

(e) The project's commercial feasibility

To complete Part E, prepare a presentation slide-deck. Practice the delivery of the presentation to ensure it fits within the time constraint. Ensure that you are comfortable with the presentation medium and discuss any concerns with your tutor. Then, attend the scheduled pitch session.

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

Part F

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

(a) A working and playable build is presented;

(b) Enough work is available to conduct a meaningful review;

(c) Individual contributions are both clearly articulated and explained.

To complete Part F, prepare a live demonstration of the pre-production prototype. You are permitted to demo any version of the pre-production prototype, but it must be a playable and working version. Then, attend the scheduled demo-day.

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

Additional Guidance

Avoid poor planning and time management. By now this will be a familiar phrase, but it is no less true. In particular, avoid underestimating the effort required to implement even simple software; always consider scope. From the pitch stage, you should consider very carefully what is feasible.

For the most part, your work will be marked as a group effort. However we want to avoid the situation where students try to "coast" through the assignment on their fellow group members' work, and equally the situation where one member of the group takes the lion's share of the work and prevents the others from contributing effectively. Marks will be weighted by a multiplier for **individual contribution**, which aims to penalise both of these behaviours. We assess this by several means, including but not limited to: sprint reviews; individual vivas; feedback from your peers; attribution in the source code; and commit logs. Any student who has contributed their *fair share* of effort to the project will receive a fair % for their effort, so any student who is putting in the appropriate level of effort has no need to worry. Note that effort is not the same as productivity.

The first step in planning your implementation should be to break your concept down into **user stories**. Your user stories should be **distinguishable** (i.e. there should be little overlap between them) and **easily measured** (i.e. it should be easy to tell when each user story has been implemented). They should also be **comprehensive**, i.e. the user stories should completely capture the desired functionality of the game, with no gaps. Imagine giving your user stories to a developer who has never seen a product of this type before. Would they be able to implement the software correctly, or would they miss key features?

Your game will be assessed on **conceptual coherence** and **player engagement**. This loosely corresponds to the **functional coherence** criterion you have previously been familiar with: how well the finished product corresponds to the user stories, and whether it has any obvious bugs. Correspondence to user stories runs both ways: implementing features that were not present in the design ("feature creep") is just as bad as neglecting to implement features. However, the strength of the concept and its ability to engage players are integrated into the definition. It mustn't just work—it should also be fun!

Your source code will also be assessed as part of your **specialist contribution**. This is a single criterion consisting of **both sophistication and maintainability**. Your code will also be assessed on **sophistication**. To succeed on a project of this size and complexity, you will need to make use of appropriate algorithms, data structures, libraries, and object oriented programming concepts. Appropriateness to the task at hand is key: you will **not** receive credit for complexity where something simpler would have sufficed. **Maintainability** is important in all programming projects, but doubly so when working in a team. Use **comments** liberally to improve code comprehension, and carefully choose the **names** for your files, classes, functions and variables. Use a well-established commenting convention for **high-level documentation**. The open-source tool Doxygen supports several such conventions. Also ensure that all code corresponds to a sensible and consistent **formatting style**: indentation, whitespace, placement of curly braces, etc. Hard-coded **literals** (numbers and strings) within the source should be avoided, with values instead defined as constants together in a single place. Consider allowing some literal values, where appropriate, to be "tinkered" without changing the source code, e.g. by defining them in an external file read at startup.

FAQ

- **What is the deadline for this assignment?**
  Falmouth University policy states that 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.
- **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

- Keith, C. (2010) Agile Game Development with Scrum. Pearson Education.
- `http://www.gamasutra.com/blogs/RogerPaffrath/20131115/204871/What_NOT_to_do_when_starting_as_an_indie_game_developer.php`
- `https://www.youtube.com/watch?v=on7end04lPY`

Marking Rubric (Pre-Production Prototype)

Criteria marked with a ‡ are shared by the group. Criteria marked with a † are weighted by individual contribution to a shared deliverable. All other criteria are individual.

Criterion	Weight	Refer for Resubmission	Basic Proficiency	Novice Competency	Novice Proficiency	Professional Competency	Professional Proficiency
Sprint Reviews	40%	Inadequate participation in sprint reviews or breach of academic integrity.	The student participates adequately in all sprint reviews (excepting mitigating circumstances). A meaningfully playable build is available in most sprint reviews.				
Conceptual Coherence	10% ‡	Few user stories are distinguishable and easily measured.	Some user stories are distinguishable and easily measured.	Most user stories are distinguishable and easily measured. User stories (as implemented) correspond to some conceptually cohesion.	Nearly all user stories are distinguishable and easily measured. User stories (as implemented) correspond to much conceptually cohesion.	All user stories are distinguishable and easily measured. User stories (as implemented) correspond to considerable conceptually cohesion.	All user stories are distinguishable and easily measured. User stories (as implemented) correspond to significant conceptually cohesion.
Player Engagement	10% ‡	Few user stories have been implemented and/or the code fails to compile or run. Unplayable and/or serious bugs.	Some user stories have been implemented. Some ability to engage, but with many serious bugs.	Many user stories have been implemented. Much ability to engage, but with a few serious bugs. There is some evidence of feature creep.	Almost all user stories have been implemented. Considerable engagement, but with a few minor bugs. There is little evidence of feature creep.	All user stories have been implemented. Significant engagement. There is almost no evidence of feature creep. Some bugs, purely cosmetic and/or superficial in nature, are detected.	All user stories have been implemented. Extensive engagement. There is no evidence of feature creep. Few to no bugs are detected.
Value Added	10%	Unable to evidence value-added or very poor prioritisation.	Some value added.	Much value added.	Considerable value added.	Significant value added. Some evidence of play-testing and data analysis to improve prototype.	Extensive value added. Much evidence of play-testing and data analysis to improve prototype.
Specialist Contribution (Technical)	30% †	Little insight into the appropriate use of programming constructs is evident from the source code. The program structure is poor or non-existant. The code is only sporadically commented, if at all, or comments are unclear. Few identifier names are clear or inappropriate. Code formatting hinders readability.	Some insight into the appropriate use of programming constructs is evident from the source code. The program structure is adequate. The code is well commented. Some identifier names are descriptive and appropriate. An attempt has been made to adhere to a consistent formatting style. There is little obvious duplication of code or of literal values.	Much insight into the appropriate use of programming constructs is evident from the source code. The program structure is appropriate. The code is reasonably well commented. Most identifier names are descriptive and appropriate. Most code adheres to the an appropriate formatting style. There is almost no obvious duplication of code or of literal values.	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. The code is reasonably well commented, accompanied by documentation. Almost all identifier names are descriptive and appropriate. Almost all code adheres to the an appropriate formatting style. There is no obvious duplication of code or of literal values. Some literal values can be easily "tinkered".	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. The code is very well commented, with some ability to generate documentation. All identifier names are descriptive and appropriate. All code adheres to the an appropriate formatting style. There is no obvious duplication of code or of literal values. Most literal values are, where appropriate, easily "tinkered" outside of the source.	Extensive insight into the appropriate use of programming constructs is evident from the source code. The program structure is extremely effective. There is very high cohesion and very low coupling. The code is commented extremely well, with much ability to generate documentation. All identifier names are descriptive and appropriate. All code adheres to the an appropriate formatting style. There is no duplication of code or of literal values. Nearly all literal values are, where appropriate, easily "tinkered" outside of the source.
Individual Contribution	Multiplier for criteria marked †	The student has failed to contribute their "fair share" to the project, or has actively prevented others from doing so.					The student has contributed their "fair share" to the project, and has facilitated others in doing so.

Marking Rubric (World and Game Pitches)

Criteria marked with a ‡ are shared by the group. All other criteria are individual.

Criterion	Weight	Refer for Resubmission	Basic Proficiency	Novice Competency	Novice Proficiency	Professional Competency	Professional Proficiency
Basic Competency Threshold	40%	No individual and/or group pitch is delivered, or either pitch is inappropriate.	A broadly appropriate 'world pitch' and 'game concept pitch' are both delivered.				
Communication	20%	The individual pitch was delivered with little enthusiasm, or communication was inappropriate in both pitches.	Has the student communicated their World concept clearly? Are the game's unique selling points clearly identified?				
Supporting Materials	10%	There were no supporting materials in either pitch.	Did the student use suitable supporting materials? Does the pitch make good use of supporting materials to help communicate the core features of the game?				
Planning	10%	Either pitch was poorly timed.	Was the pitch content well planned, and also structured to the available time-frame? Is the pitch structured to suit the available time-frame?				
Professionalism	10%	Little to no innovation and/or creativity. Either pitch was unprofessional.	Was the pitch delivered in a professional way? Does the pitch demonstrate a commercial awareness of the market the game will be competing in?				
Commercial Feasibility (Game Pitch Only)	10%	No budget or production plan are proposed, or they are inadequate.	How confident is the marker that the game concept is viable?				