**Assignment Brief – BTEC**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Programme** | | Level 3 Extended Diploma in Creative Media Production (Games Development) | | | |
| **Unit number(s) and title covered** | | Unit 71: Object-Oriented Design for Computer Games | | | |
| **Assignment number & title** | | Assignment One: The Purpose of OO Design | | | |
| **Student name** | | *Lewis Hawkins* | | | |
| **Assessor** | | James Shaun | **Internal Verifier** | *David Matravers* | |
| **Date issued** | | *11.10.2018* | **Submission deadline** | *08.11.2018 at* ***4.30pm*** | |
| **Assessment Criteria** | **To achieve the criteria, the evidence must show that the student is able to:** | | | | **Assessor confirm met** |
| P1 | Describe the purpose of object-oriented design for games with some appropriate use of subject terminology | | | | Yes |
| M1 | Explain the purpose of object-oriented design for games with reference to detailed illustrative examples and with generally correct use of subject terminology | | | | Yes |
| D1 | Comprehensively explain the purpose of object- oriented design for games with elucidated examples and consistently using subject terminology correctly | | | | No |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Assessor feedback - 1st submission** | | | | | | | | | |
| *Task No* | *Targeted Criteria* | *Met* | *Comment* | | | | | | |
| 1 | P1 | Y | Great start Lewis, you have given a range of explanations as to the purpose of object orientated design for games with some clear examples.  At times some of your explanations could be further developed and complete and you also miss opportunities to explain how OOD methods are the best for the develop of a game over others. I am recommending you for a resubmission to improve this assessment. | | | | | | |
| 1 | M1 | Y |
| 1 | D1 | N |
| **Did the learner meet the original deadline or agreed extension?** | | | | Yes ☐ | | | | | |
| **Assessor signature** | | | **H:\My Pictures\Jim Signature - Copy.jpg** | | | | **Date** | | 05/01/2019 |
| **Resubmission authorised?** | | | | Yes ☐ | | | | | |
| **New agreed deadline date for submission** *\* must be within 10 days of receiving original assignment back* | | | | **26/01/2019** | | | | | |
| **Lead Internal Verifier signature** | | | **H:\My Pictures\Jim Signature - Copy.jpg** | | | **Date** | | 05/01/2019 | |
| **Assessor feedback - Resubmission** | | | | | | | | | |
|  | | | | | | | | | |
| **Assessor signature** (resubmission only) | | |  | | **Date** | | | |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Scenario** | | | | |
| You work as a freelance games journalist that has been approached by Player magazine in Glasgow, who have shown an interest in you to become a monthly columnist in their esteemed publication.  They have asked for you to demonstrate your work by writing an article on aspects and fundamentals of object orientated design in games, using examples so that readers of all ages will find it accessible, though there will be areas where simplification is difficult. | | | | |
| **Tasks and criteria covered** | | | | |
| **Task 1**  You must write an article which comprehensively explains the purpose of object-orientated design for games. This will cover a definition what an object is, alongside how the object orientated design process is used to improve the design process for games development.  You make comprehensive arguments to explain why using an object orientated approach is the best method for games development and should include how features of OOD can improve communication between teams for the designing process. You should also comment on how the selection and use of objects as well as how recycling game objects and classes can be effective in time saving for developers or prototyping.  Your article should include an explanation of the following and how these are well suited to the games design and development process of a 2D or 3D games. You should take care to use detailed examples that support your points.   * Game objects *(swords, characters, walls, scenery, pickups)* * Properties (*colour, size, speed, sounds, health, damage etc)* * Actions and events (*Input, clicks, button press, collision, destroy)* * Inheritance *(parent/child)* * Reusability *(prefabs, DLC, sequels)* * Maintenance *(Improve, fix, optimise)* * Communication *(between teams)* * Real world modelling   [Task Covers P1, M1, D1] | | | | |
| **Evidence you must produce for this task** | | | | |
| Report  Supporting Images | | | | |
| **Sources of information** | | | | |
| **Sources of information**  **Textbooks**  Baylis P, Freedman A, Procter N et al – BTEC Level 3 National Creative Media Production, Student Book  (Pearson, 2010) ISBN 978-1846906725  Baylis P, Freedman A, Procter N et al – BTEC Level 3 National Creative Media Production, Teaching Resource  Pack (Pearson, 2010) ISBN 978-1846907371  Gold J— Object-Oriented Game Development (Addison-Wesley, 2004) ISBN 978-0321176608  Makar J – Macromedia Flash MX Game Design Demystified (Macromedia, 2002) ISBN 978-0735713987  Miles R and Hamilton K – Learning UML 2.0 (O’Reilly Media Inc, 2006) ISBN 978-0596009823  Overmars M – ‘Learning Object-Oriented Design by Creating Games’ in Potentials  (the journal of the Institute of Electrical and Electronic Engineers), December 2004-January 2005, Volume 23,  Issue 5, pages 11-13 (available from www.cs.uu.nl/research/techreps/repo/CS-2004/2004-057.pdf)  Rollings A and Morris D – Game Architecture and Design: NRG Programming (New Riders, 2003)  ISBN 978- 0735713634  Swamy N and Swamy N – Basic Game Design and Creation for Fun and Learning (Charles River Media, 2006)  ISBN 978-1584504467  **Websites**  www.cs.uu.nl/research/techreps/repo/CS-2004/2004-057.pdf – article on learning object-oriented design by  creating games, by M Overmars, author of Game Maker software (available from www.yoyogames.com/  make)  www.developer.com/design/ – software development resources and articles  www.devmaster.net/articles/oo-game-design/ – game development encyclopaedia  www.gamasutra.com – respected website for all things game development, sister publication to the respected  print magazine Game Developer; excellent game developer resources  www.macromedia.com/devnet/mx/director/articles/oop\_dir\_flash.html – article on designing and  implementing objects  www.tdan.com/special003.htm – special feature on event progress diagrams | | | | |
| **Student checklist** | | | | **Complete?** |
| Proofread work | | | |  |
| Reference / Bibliography (if applicable) | | | |  |
| All pages attached and numbered – including introduction/conclusion/front sheet | | | |  |
| **Authenticity of Evidence Student declaration** | | | | |
| I certify that the evidence submitted for this assignment is my own.  I have clearly referenced any sources used in the work.  I understand that false declaration of authenticity (i.e. plagiarised work) is a form of academic misconduct and the relevant College procedures will be instigated if I am found to be in contravention of these. | | | | |
| **Student signature** |  | **Date of submission** | 08/11/18 | |
| **Re-authentication of Evidence Student declaration (for resubmission only)** | | | | |
| **Student signature** |  | **Date of resubmission** | 25/01/19 | |

NB. Students – the assignment starts on the first page **after** these front sheets, i.e. Page 1.

* For your convenience, page numbers have been inserted into the footer. **Please keep them**.
* You may choose to add a contents table (ToC) in this section.
* Please **do** **not use text boxes** for the main body of your written answers.
* Please make sure that images/screenshots are correctly formatted, laid out and labelled. A table of Figures (ToF) may also be added if you wish.
* Make sure you use Page (or Section) Breaks whenever a new page is required. (Rather than adding large numbers of Return/Paragraph characters.) Ensure that new Section breaks continue with correct orientation and correct page numbers.
* Ensure that you have referenced your work throughout, using references in text and that you also have a reference list and full bibliography at the end of the work according to the current **Harvard Referencing** conventions. **Failure to do so will make your work more difficult to authenticate.**

**Task One**

**Game Objects:** Game Objects are entities within the game which are split into two categories; physical and non-physical.

Physical are either visible to the player (Building), or interactable (Trigger collider). Physical game objects are made to build the game world full of visuals and interactable. Without the use of Physical game objects, you would not have a game.

Non-Physical are game objects that don’t interact and are often invisible. They’re used to hold scripts and other components but can also be UI elements like the HUD. In games there often needs to be a master script of some kind, which controls everything in the background, while also serving as a root script other scripts can pull data from. It can also be used to call important methods that multiple scripts might want to use.

**Properties:** A property is a variable that’s used to control an action, such as speed, spawn position, items to drop, and boolens that control different behaviours.

They’re used to set variables at runtime or to be set on “Start(); / OnValidate();” and are essential for storing temporary, and default data. One way they can also be used is to create variations of an object through changing them (Rate of Fire, Attacks, Colour, Speed).

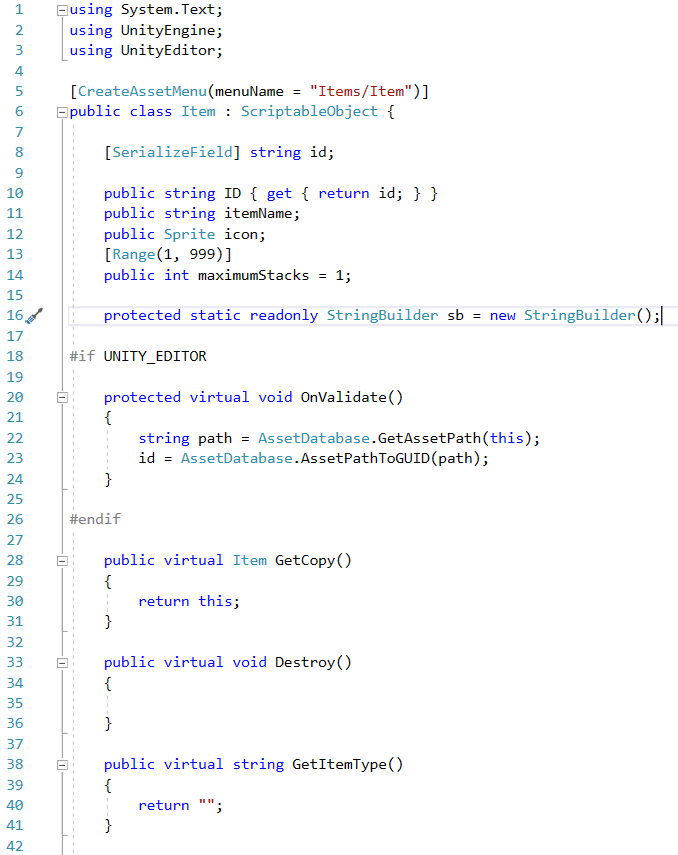
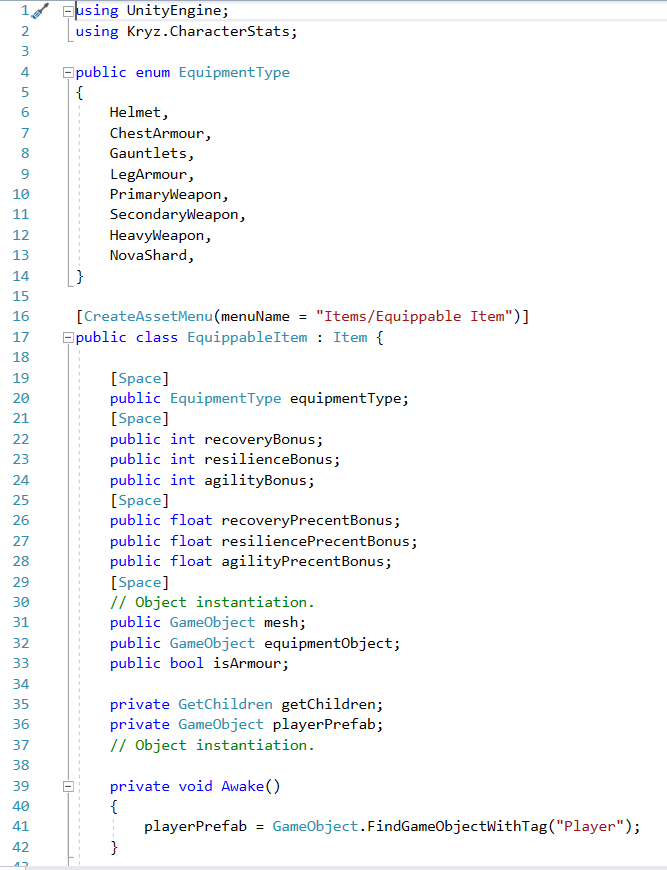
On a gun for example, there needs to be multiple floats, ints and boolens which determine rate of fire, impact force, magazine size, and damage. Different guns will require these stats to be varied to avoid them from being a simple cosmetic change. If the properties are public or private while using “[SerializeField]” they can be changed in the inspector and prefabbed to create a variant with different stats.

**Actions/Events:** An action is the response an event (Move forward, shoot, die, increase score.), Events are the inputs and collisions which lead to an action.

Actions are what makes the game work. There can be as many events as needed, but withut actions they’re usless. Actions are esentual to making a game feel alive through player interaction, however it doesn’t need to be caused by a player. Actions can be called at anytime either on Start(), timer, or player events.

Events are used to move the player, shoot a gun, open a door, etc, and are required to allow the player to make decisions. They’re used in many ways to not only do the basics, but increase the interactability of the game world. The game worlds interactability is key to making it feel alive and real, which is what every developer should aim for.

**Inheritance:** Inheritance enables objects to take on properties of another object. The property is created in one class, and is can be taken using subclasses. Below is an example of a class (Left) and a subclass (Right) which I created.



Inheritance is a useful tool as it allows the programmer to separate classes making them easier to reuse and read. It also creates less stress on the machine as its only possessing code it needs and isn’t going through unneeded functions. Also, it allows the developer to use the script execution order (Unity engine), instead of having to write more code just to make sure it’s read in the right order.

**Reusability:** Making code reusable is arguably the most important priority besides functionality and optimisation. Making code reusable allows for less time to be spent on programming and more on other development aspects. This means one class can be used for all agents instead of having a different one just for a slight change such as flying or animation.

My code above is a good example of making code reusable, as it takes advantage of Scriptable Objects. A Scriptable Object is like a regular MonoBehaviour except it doesn’t have to be attached to an object. Instead it acts as a data file which can be taken and used by a game object and used to store other objects to be used later.

A good example of how these can be used is an inventory like what i’m using it for. They make it easy to create a new item in seconds (All I need to do is create one in the project window and assign the properties.).

**Maintenance:** Maintaining code is key to making it readable and and as optimised as possible. In C# a good way of organising code is to split up public and [SerializeField] variables using [Header(“Example”)]. This not only splits the code in the script, but also in the inspector.

It’s also good practice to create notes “//Example note” to remind yourself or others what the code is for.

Another big part of maintenance is optimisation. The best ways to optimise code is to use as little of it as possible, and to not use functions such as “GameObject.FindGameObjectOfType<Rigidbody> ();” unless there’s no other way of getting the component.

**Communication** is key to working with a team, as things need to be done in a certain order and before deadlines. It’s very important for the leader of the group to know what everyone’s doing, as without this knowledge the group can fall apart.

During enterprise lessons I took the job of leader and made sure the whole group was working on specific areas and working to deadlines.

**Real World Modelling** is the technique of modelling an entity after the real-world counterpart. This can be useful when making realistic vehicles, and AI, however making something too realistic can be problematic for the game. To counter this issue while keeping the realism developers program AI (for example) to make mistakes and in the case of Batman: Arkham Night, to continue looking away when the player is behind them.

Forza Horizon 3 is a good example of how real-world modelling can be used in games. This is a game about racing realistic cars through a vast open world, with amazing graphics. The developers used real-world modelling to create the vehicles as realistic as possible, which is a property the developers pride the game on.

Another use of real-world modelling is realistic physics. Physics are key to making a world believable and if done right the developer can rely on them to handle a good proportion of the game.