Game Engine Programming Inventory-System: Post-Mortem Report

A video game with a robot

Description automatically generated

**Figure 1: Unreal Engine 5, Example image of completed Inventory**

# Abstract

# This is a Document that explains how this Inventory system has been created, the original inspiration for the project, and how it shaped into the finished product; Explaining the issues faced and overcome. An examination of what went well and what could have been done better to be developed in future projects.

# Introduction

During this Post-Mortem report the recently completed inventory is going to be examined. This is a chance to disassemble the whole project and have a look from the planning to implementation what was the expected results in comparison to the finished product. How everything that was achieved finished and what could have been improved upon in the stages of creating the inventory system. In the hopes that for future projects this Post-Mortem report can be a reference point and something to learn from. Firstly, look into the background research for Inventory and how the ideas were made to create this style of inventory. After this it will be the Implementation, explaining what mechanics were added and what were not added because of issues faced along the way. Then finally, the evaluation looks into what was achieved and what could have been improved upon and a comparison of the two engines.

**Student Number: 22036987**

**Student Name: Riley Macfarlane**

# Background & Research

**Figure 2: DayZ standalone PC v0.29.113822, The Inventory showcasing the different sizes of varying objects. (Ashley Pomeroy, Dec. 2013)**

A screenshot of a video game

Description automatically generated

**Figure 3: Minecraft PC rd-132211, This is the simplified Minecraft inventory**

A screenshot of a video game

Description automatically generated

# Minecraft's Inventory: items are placed inside a Slot which is inside a grid. All the items take up the same amount of space. One slot

inside the grid and some items can stack some cannot.

# During the idea phase of the project, one idea was to base the Inventory System in a similar way to how the game DayZ has achieved theirs. A brief explanation of DayZ is as explained on their main website 'DayZ is a gritty, authentic, open-world survival horror hybrid-MMO game, players follow a single goal: Survive by any means necessary' (Bohemia Interactive, T&D N/A). This game has an inventory not too dissimilar from the game Tetris (Tetris Holding, Alexey Pajitnov in 1984). It implements a complex grid-based system. Each different item is associated with a different number and shape placement of squares; this will then need to fit into the player's backpack which is also several squares depending on the size of their backpack. After some research (Unreal Engine - Spatial Inventory Tutorial, Reids Channel, Released to YouTube on Nov 17, 2020) Quickly the decision was made to simplify this considering the level of skill required to develop this system to a high standard and my skill level.

# Due to this, it was decided that it would be better to investigate a more simplified grid-based system. The decision was based on Minecraft's Inventory.

# The main functionalities that would be taken from the Minecraft inventory system are the grid-based system, click-drag to move items into different slots and dropping items back into the game world by clicking anywhere outside of the inventory.

# After researching multiple different tutorials on the internet (Night Run Studio, released to YouTube on 30th Jan 2024, Gorka Games, Released to YouTube Nov 12 2022). Two tutorials were found, one for each engine, they looked to be descriptive, professionally created and with a small number of changes catered to exactly what I had in mind for my system (Unity Inventory System Tutorial, By Dan Pos, Released to YouTube on Dec 26 2021; Unreal Engine 5 Tutorial - Inventory System Ryan Laley, Released to YouTube on Oct 11, 2022). Learning from each of the tutorials assisted in developing an understanding of how to go about creating this project for each of the engines. What could be seen so far is that it would not be difficult to make each system the same if not similar to each other.

# A screenshot of a computer Description automatically generatedImplementations

**Figure 4: Unreal Engine: Blueprints, Showcasing the multiple different functions and ‘RemoveFromInventory’ being called in the main event graph in the Inventory system.**

# The first step in the Unity Project was to create the inventory system script this is the event system for the whole project. On start, it is responsible for adding a pre-determined amount of slots to the inventory. During the running of the program, it runs all the functions that allow the inventory to work which will then be sent to and updated by the UI. One of these functions is the picking up of an item. The System then needs to check certain parameters: can the item be added to a stack of existing items if there is enough space? If not or there are not any existing items add it to an empty inventory slot. The size of the Inventory System can also be changed by a public int inside before running.

# In the Unreal project, the inventory system blueprint works similarly to the Unity project but the difference is that you call and implement the event of all the blueprint-built components that run the inventory system such as the UI elements or the specific slot for the inventory. If a component is created almost anywhere on this project, it will be passed into the inventory system to trigger the event to happen when instructed. The inventory system is the event system that controls all the code included with the behind-the-scenes use and the UI of the inventory. The functions that have been created inside the inventory system are also functionalities that allow each component to complete its task.

# A screenshot of a computer Description automatically generated

**Figure 5: Unity Engine: Inspector, Image Example of an 'InventoryItemData' Scriptable Object**

# In both Engines, each item can be added to a data table where it can take in certain parameters: Name, ID, stack size, description and image icon. The ID was created so that the Inventory System can differentiate each item from one another; this is a way of making sure the items are placed exactly where expected. The stack size determines the item's maximum amount it can hold in each instance of a slot. The original idea for the item description was that it was going to be visible when hovering over the item for pickup but unfortunately, this was not achieved in the final version. Finally, the image icon is the icon which will be displayed when it is added to the inventory.

# In both engines it has been implemented in such a way that would be very simple for an external programmer to come and insert new items into the Item Data table. In Unreal this can be done by simply creating a new actor blueprint class inserting the desired mesh for the item and adding collisions then simply adding a new row to the data table inserting all the parameters I have previously explained finally, just select the newly created Actor class in the item class drop-down menu and a new item has been created ready for use in the scene. An Item Data component has been created which has certain preset parameters when a new actor is created the Data table and Quantity of each are pre-determined. A similar system has been implemented into the Unity Project by using scriptable objects named 'InventoryItemData'. This script allows for scriptable objects to be created using previously defined parameters similar to the Unreal project Name, ID etc. This was achieved by adding public variables in the code base of the scriptable object.

# Other mechanics that are in the Unity project include the movement of Items to different slots inside the inventories, Functionality for the creation of multiple inventories (chests), in Unity the interactions between the player inventory and the chests, drag and drop mechanic to move and finally splitting items with a shift click and swapping alternate items on click.

# Evaluation

The focus of this section is to appreciate what was achieved during the project; managed to implement core functionality to get an inventory system that works as intended. On both Engines, the task that was set out has been completed to a positive level in the time frame given. This project has been very beneficial in terms of new information gained in the engines and general coding knowledge acquired. As mentioned in the previous paragraph a large amount of extra functionality was also implemented being the chests and the mechanics to move items around the inventory.

Unfortunately, the movement of items from between the chests and the player was not achieved in Unreal. The creation of the alternate inventories was implemented but issues occurred when finalising the blueprint and there was not an understandable reason for this. It would be a good idea to go back through and try to fix the issue once again. It would be beneficial to figure out what the issue was to be able to learn from it. Ideally in the final project, it would have been better if the ability to use an item to get certain buffs also equip items such as tools, weapons or maybe even crafting two items together to create another. For example, if the player was able to eat a certain food to get a speed increase, use a sword for a melee attack or craft some string and sticks into a bow which could then also be equipped.

The inventory system created would be perfect for a video game that can upgrade or change inventory slot sizes depending on game progression. The continuation of play, while the inventory system is open, could add tension to the player in a wider video game setting. For example, in a game where the player is being chased the use of the inventory would need to be quick and the player would need to make sure that they have everything that is needed to be used or equipped in the hot bar. Both Engines are very similar in their feel and use.

# The Main difference I noticed between the two engines that in Unreal you can visually implement the code using the blueprints. Personally, this was beneficial to me in Unreal. Although more time-consuming, having the ability to visualize where the code is pointing and connect the pins helped me to understand it. Unreal uses more of a computer's GPU in comparison to Unity. This is because the graphics and the physics are more complex than that in Unity.

# References

<https://theswissbay.ch/pdf/Gentoomen%20Library/Game%20Development/Programming/Postmortems%20From%20Game%20Developers.pdf>

POSTMORTEMS FROM GAME BY DEVELOPER AUSTIN GROSSMAN, Published by CMP Books, Distributed across the U.S and Canada in 2003

https://women-and-dreams.blogspot.com/2013/12/dayz-and-confuz.html

POSTED BY ASHLEY POMEROY AT TUESDAY, DECEMBER 24, 2013

https://dayz.com/

Bohemia Interactive (2024), DayZ Released in December 2018, Video game

<https://minecraft.net/en-us>

Mojang Studios and Xbox Studios, Minecraft, Released May 2009, Video game

https://tetris.com/play-tetris

**by** Alexey Pajitnov in 1984, Tetris Holding

<https://www.youtube.com/watch?v=0wKB_rxtqh4&list=PLSR2vNOypvs6eIxvTu-rYjw2Eyw57nZrU>

Name: Let's Make An Inventory System In Unity!

**By** Night Run Studio, Released to YouTube on 30th Jan 2024

<https://www.youtube.com/watch?v=khjUY5To7qI&t=11s>

Name: How to Make a Simple Inventory System in Unreal Engine 5

**By** Gorka Games, Released to YouTube Nov 12 2022

<https://www.youtube.com/watch?v=vHT4MhmwacE&list=PL4G2bSPE_8umjCYXbq0v5IoV-Wi_WAxC3>

Name: Unreal Engine 5 Tutorial - Inventory System

**By** Ryan Laley, Released to YouTube on Oct 11, 2022

<https://www.youtube.com/playlist?list=PL-hj540P5Q1hLK7NS5fTSNYoNJpPWSL24>

Name: Unity Inventory System Tutorial

**By** Dan Pos, Released to YouTube on Dec 26 2021

<https://www.youtube.com/watch?v=4CjpBoKl6s8&t=1177s>

Name: Unreal Engine - Spatial Inventory Tutorial

**By** Reids ChannelReleased to YouTube on Nov 17, 2020