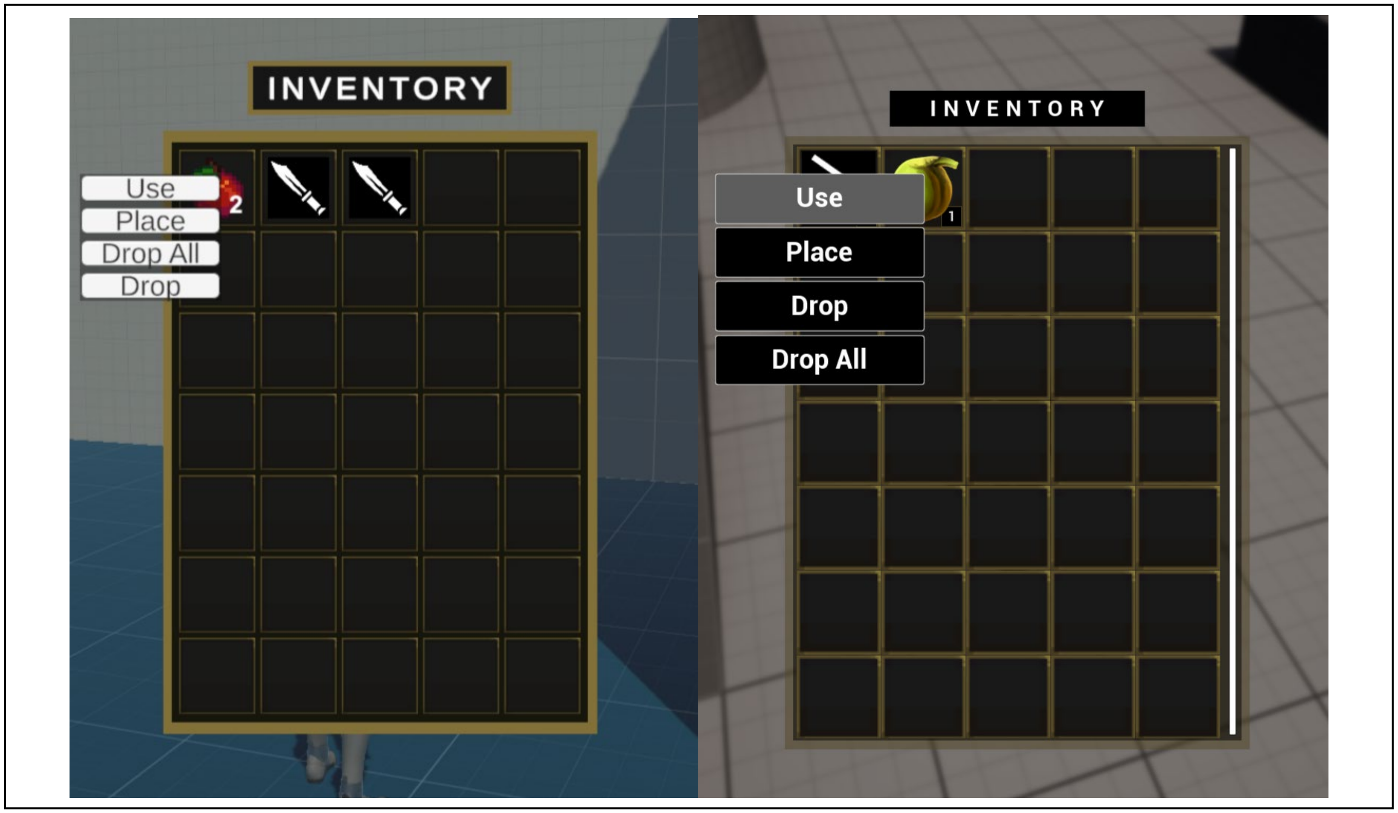
Game Engine Programming

Inventory System Implementations Post-Mortem Report

Student Number: 22000237



Introduction

Two similar inventory systems were made in both Unreal Engine and Unity. The inventory system included a grid-based interface that allowed players to drag and drop items to any slot in the inventory. Moreover, an item description will be displayed when the mouse pointer is hovering over the item. Players can perform various actions with the item by navigating the item menu when right-clicking on it, including using, dropping, and placing the item. Furthermore, the player can pick up items in the game world when the items are within the interaction range of the player's character.

Background & Research

Before the implementation of the inventory in both engines, research had been done on the tools and systems provided in both engines, the logic of the inventory and object interaction system, and UI designs of inventory in various games.

Engine Systems and tools

By studying the lectures (Hompstead, 2023), table 1 shows the tools and systems that will be used to create a more developer-friendly and data-driven system for the developer to change and extend the system being created.

Engine Usage	Unity	Unreal
Handling item data	Scriptable Objects	Data Structure/Data Table
Create Events	Event Trigger, Predefined events in “Unity.EventSystems” library	Predefined Events/ Custom Events
Able to change variables without navigating the script	Attributes	Inputs in Custom Events

Table 1. Tools and systems in each engine for the implementation.

Initial design inspired by games

Some functionalities of the implementation are inspired by different games as shown in Table 2.

Games	**Features**
Minecraft	Grid-based
Baldur’s Gate 3	Item description when the mouse hovers over an item
Hitman 3	Item pickup and place down

Table 2. Game features inspire implementation.

Implementations

The following will talk about the implementation of each subsection of the inventory system.

Inventory User interface

Engine Functionality	Unity	Unreal Engine
Trigger Event with UI	Button	Button
Assign slots	Grid Layout Group Component	Wrap Box
Assign buttons vertically in the item menu	Grid Layout Group Component	Vertical Box
Create additional children in a widget	UI objects can have multiple children	Overlay
Ignored by the mouse's ray cast	Add a Canvas Group component and uncheck "Block Raycasts"	Set the visibility to "Non Hit_Testable"

Table 3. Features used for the user interface in each engine.

Implementation of UI in Widgets is used for creating the user interface of the inventory, table 3 shows the features used.

Importing 2D/3D assets

To add mesh and sprite for the object created, assets will be imported into the engine project file. Table 4 shows the ways assets are imported in both engines.

Engine Assets	Unity	Unreal Engine
3D mesh and 2D sprites	Unity asset store imported by using package manager	Quixel Bridge
2D sprites from outside sources	-dragging the files downloaded from outside sources and dropping them in the project window -The texture type has been set to sprite before usage	-dragging the PNG files downloaded from outside sources and dropping them in the content drawer -converted to textures when imported

Table 4. Ways of importing assets in each engine

Item data

To make the implementation more data-driven and keep different properties of the item in the same place can avoid every item's property being hard-coded when accessing them in the functions and events.

UNITY

Firstly, a scriptable object for items is created by a script to create custom item types that contain different properties, so it can avoid copies of values by saving different related instances in it to save memory when accessing it in scripts. Secondly, new item instances are created by adding a new one in the project window (Figure 1) and setting up the variable of each property in the inspector (Figure 2). Moreover, the item's properties can(Figure 3) be accessed in scripts with the variable of item class. Lastly, the properties of the item class can be extended and changed by navigating the item scripts and adding extra item instances in the project window.

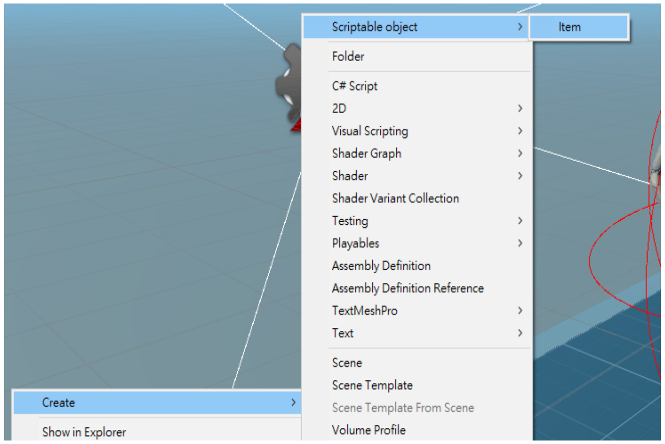
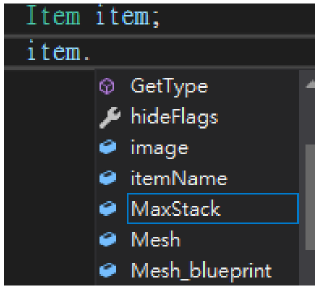
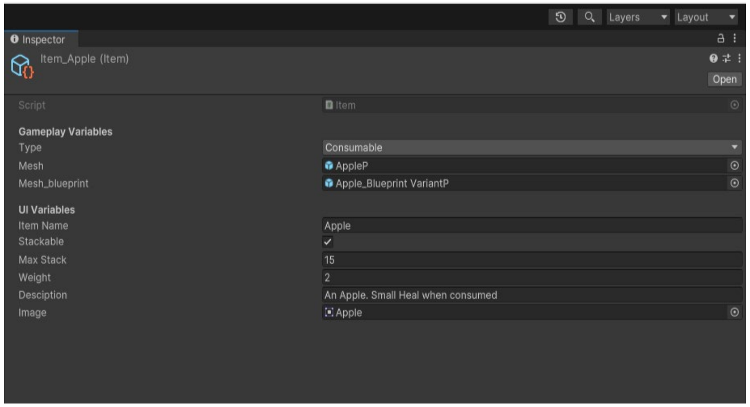


Figure 1: A new scriptable object instance can be created in the project window.



```
Item item;
item.
        GetType
        hideFlags
        image
        itemName
        MaxStack
        Mesh
        Mesh_blueprint
```

Figure 3: Accessing the item properties in scripts.



Item_Apple (Item)

Script	Item
Gameplay Variables	
Type	Consumable
Mesh	AppleP
Mesh_blueprint	Apple_Blueprint VariantP
UI Variables	
Item Name	Apple
Stackable	✓
Max Stack	15
Weight	2
Desciption	An Apple. Small Heal when consumed
Image	Apple

Figure 2: Variables in the instance can be set in the inspector.

UNREAL

Firstly, a "Structure" blueprint is used to create custom item types like scriptable objects in Unity, so accessing different properties of the item can be done with the item ID using the script shown in Figure 4.

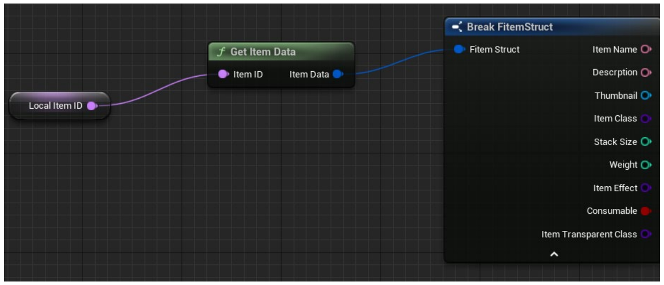


Figure 4: nodes to access item data with the item ID.

Secondly, data table is used to handle all the instances with the item structure, so accessing the properties of an item instance from the data table can be done by the script shown in Figure 5.

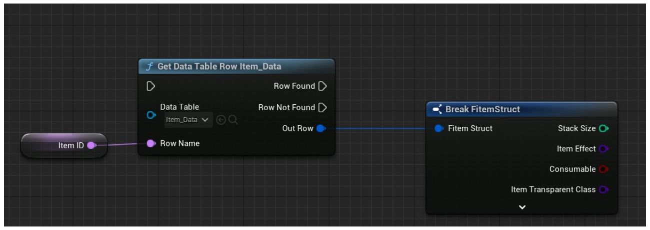


Figure 5: nodes to access item data in data table with the item ID.

Lastly, the users can expand and change the item property by adding extra variables or altering variables in the structure blueprint. Moreover, they can add an extra item instance by adding a new row and setting up the properties of the new item in the data table.

Inventory system

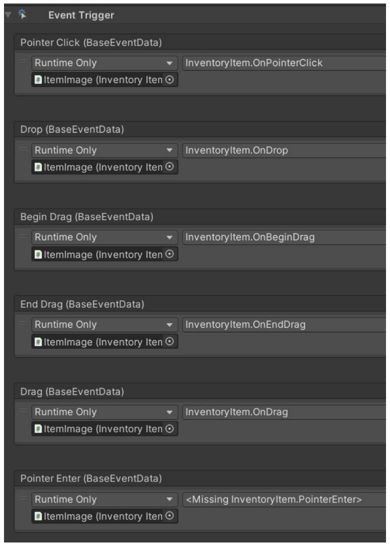
The inventory system includes finding an empty slot and adding an item in it or stacking the same items in a single slot in the inventory when an item is picked up. Moreover, reducing the quantity or removing the item when it is consumed or taken out.

UNITY

The implementation of the inventory system is created by synthesising the inventory system created by Coco Code (2022), and it is created by using scripts as a manager to run the functions.

UNREAL

The implementation of the inventory system is created by synthesising the inventory system created by Ryan Laley (2022), and it is created by using an actor component as a manager to run the functions.



Event Trigger

Pointer Click (BaseEventData)
Runtime Only | InventoryItem.OnPointerClick
ItemImage (Inventory Iten

Drop (BaseEventData)
Runtime Only | InventoryItem.OnDrop
ItemImage (Inventory Iten

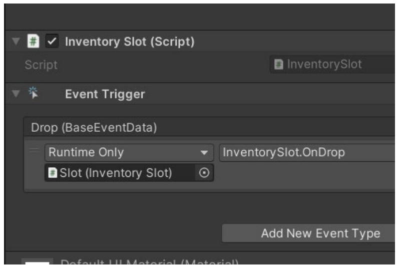
Begin Drag (BaseEventData)
Runtime Only | InventoryItem.OnBeginDrag
ItemImage (Inventory Iten

End Drag (BaseEventData)
Runtime Only | InventoryItem.OnEndDrag
ItemImage (Inventory Iten

Drag (BaseEventData)
Runtime Only | InventoryItem.OnDrag
ItemImage (Inventory Iten

Pointer Enter (BaseEventData)
Runtime Only | <Missing InventoryItem.PointerEnter>
ItemImage (Inventory Iten

Figure 5: Event trigger in item image gameobject.



Inventory Slot (Script)
Script | InventorySlot

Event Trigger

Drop (BaseEventData)
Runtime Only | InventorySlot.OnDrop
Slot (Inventory Slot)

Default UI Material (Material)

Figure 6: Event trigger in item slot gameobject.

Drap-and-Drop

The item drag-and-drop system enables the player to move the item to the desired slot in the inventory.

UNITY

The drag-and-drop is implemented by synthesising the item placing logic created by Coco Code (2022).

The following is the flow diagram of the drag-and-drop:

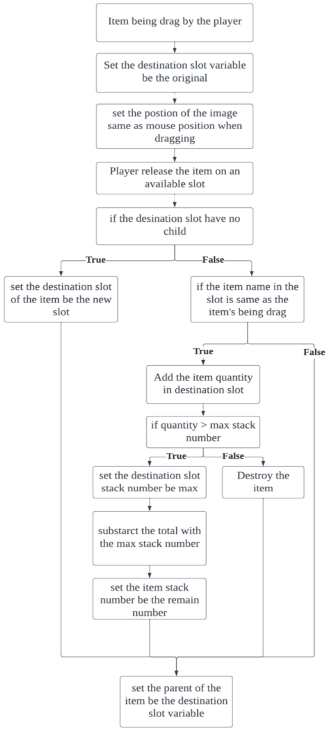

Item being drag by the player
Set the destination slot variable be the original
set the postion of the image same as mouse position when dragging
Player release the item on an available slot
if the desination slot have no child
True
False
set the destination slot of the item be the new slot
if the item name in the slot is same as the item's being drag
True
False
Add the item quantity in destination slot
if quantity > max stack number
True
False
set the destination slot stack number be max
Destroy the item
substarct the total with the max stack number
set the item stack number be the remain number
set the parent of the item be the destination slot variable

Figure 4: Flow diagram of the drag-and-drop logic in Unity.

Event Triggers are used on the item image (Figure 5) and item slot (Figure 6) respectively with the “PointerEventData” from the Unity event system to enable the game to detect mouse events such as dragging or clicking. So, the drag-and-drop functionalities will be called when the corresponding mouse event is triggered.

UNREAL

The drag and drop is implemented by synthesising the item placing tutorial created by Ryan Laley (2022).

The following is the flow diagram of the drag and drop:

A Drag Drop Operation blueprint is created to pass the inventory actor and ID of the item through the slot and display the item icon with the “WBP_DragPreview” widget to follow the mouse pointer when dragging.

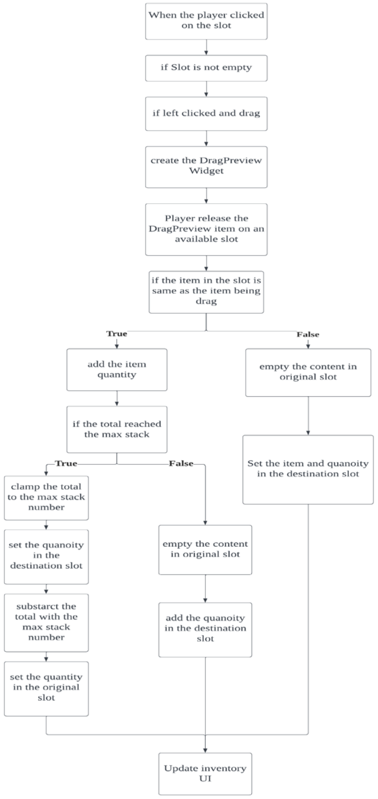

When the player clicked on the slot
if Slot is not empty
if left clicked and drag
create the DragPreview Widget
Player release the DragPreview item on an available slot
if the item in the slot is same as the item being drag
True
False
add the item quantity
empty the content in original slot
if the total reached the max stack
Set the item and quanoity in the destination slot
True
False
clamp the total to the max stack number
set the quanoity in the destination slot
substarct the total with the max stack number
set the quantity in the original slot
empty the content in original slot
add the quanoity in the destination slot
Update inventory UI

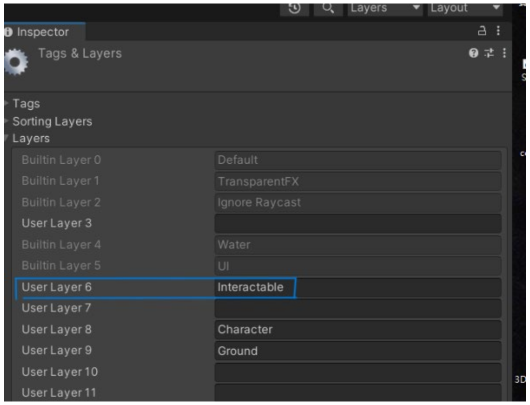
Figure 7: Flow diagram of the drag-and-drop logic in Unreal.

Object Interaction Check

If the player wants to interact with the interactive object, such as item pickup or chest opening, the object needs to be within the character’s interact range.

UNITY

The object interaction is implemented by following the item placing tutorial created by Dan Pos (2022).



Builtin Layer 0	Default
Builtin Layer 1	TransparentFX
Builtin Layer 2	Ignore Raycast
User Layer 3	
Builtin Layer 4	Water
Builtin Layer 5	UI
User Layer 6	Interactable
User Layer 7	
User Layer 8	Character
User Layer 9	Ground
User Layer 10	
User Layer 11	

Figure 8: “Interactable” Layer.

Firstly, An Interactable Layer is made for the item objects in the game world (Figure 8), so the interaction range will only check for the gameobject with the layer set to Interactable. Secondly, by using the “OverlapSphereNonAlloc‟ function in the Unity physics library, a spherical range is created at the child empty gameobject position in the player character to check any gameobject with an Interactable layer is in front of the character.

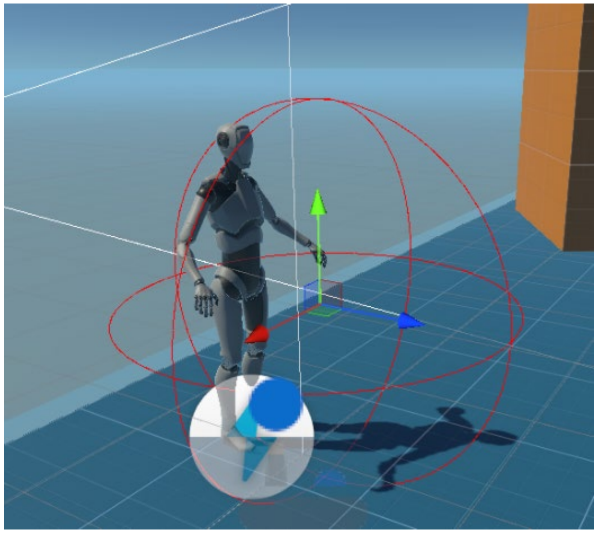


Figure 9: Interaction check range in Unity.

Lastly, Gizmos are used to make the range visible, which will make it easier for further changes in the range (Figure 9). Moreover, the user can change the size of the interact range in the Player Interaction component of the Player Character in the inspector with the use of serialize field in the script, Moreover, the position of the range can be changed by moving the “InteractRange” gameobject in Player Character through the scene or the inspector.

UNREAL

The object interaction is implemented by synthesising the item placing logic created by Ryan Laley (2022).

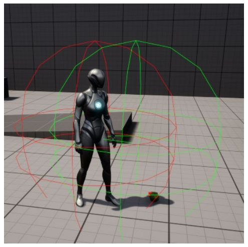


Figure 10: Capsule trace to check whether the player character is in front of the object.

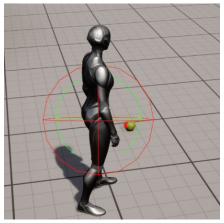


Figure 11: Sphere trace to check whether the player is looking at the object.

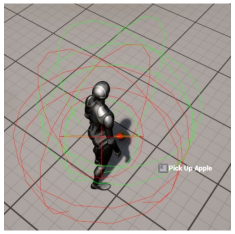


Figure 12: A prompt will be displayed when both conditions of tracing are true,

Firstly, An Interactive trace channel is added by creating a new trace channel in the project settings. When creating each item actor that can be traced by the interact range, the collision presets of the mesh should be set to custom to enable the mesh to block the interactive trace.

Secondly, traces are used for checking whether the object is in range to interact. Firstly, a capsule trace is used to check whether the player character is in front of the object (Figure 10). Secondly, a sphere trace is used to check whether the player is looking at the object (Figure 11). If both conditions are true, a prompt will be displayed, and the player will be able to interact with the object when the corresponding button is pressed (Figure 12).

Lastly, if the users want to change the interact range, they can set the Draw Debug Type variable to make it visible and the radius of both traces' ranges can be customed in the inventory system of "BP_PlayerCharacter".

Item placing

The player can place an item from the inventory where they desire. When the player selects the "Place" button in the item menu, the inventory closes and a preview of the item will follow the player's position and where the camera is looking. When the left mouse button is clicked, the item will spawn at the preview's position.

UNITY

The item placing is implemented by synthesising the item placing logic created by Seabass (2020).

Figure 13 shows the flow of the item placing:

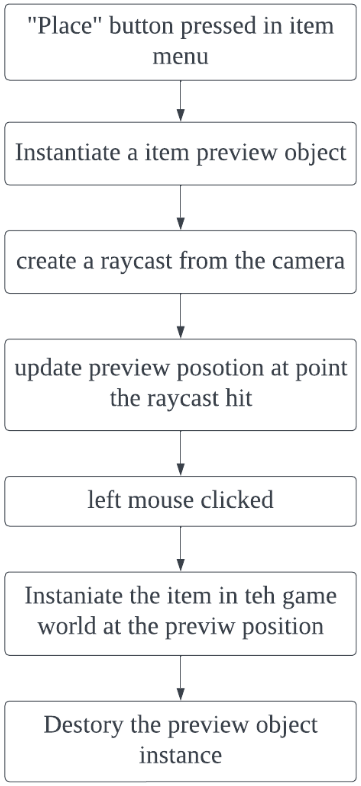

"Place" button pressed in item menu
Instantiate a item preview object
create a raycast from the camera
update preview posotion at point the raycast hit
left mouse clicked
Instaniate the item in teh game world at the previw position
Destory the preview object instance

Figure 13: Flow diagram of the item placing logic in Unity.

Raycast from the Unity Physics library is used for setting the item preview position to be on the ground that the camera looking at. Moreover, the raycast will only hit game objects set to the “Ground” layer so it will not be blocked by other unrelated game objects.

UNREAL

The item placing is implemented by synthesising the item placing logic created by Matt Aspland (2021) and Seredias(2022).

Figure 14 shows the flow of the item placing:

Line tracing is used for a similar functionality as Raycast in Unity. However, an extra line tracing is used to limit the maximum distance of the original trace with the line tracing pointing vertically down to the ground. The item preview will be positioned on the hit point of the second trace when the first trace didn’t hit anything.

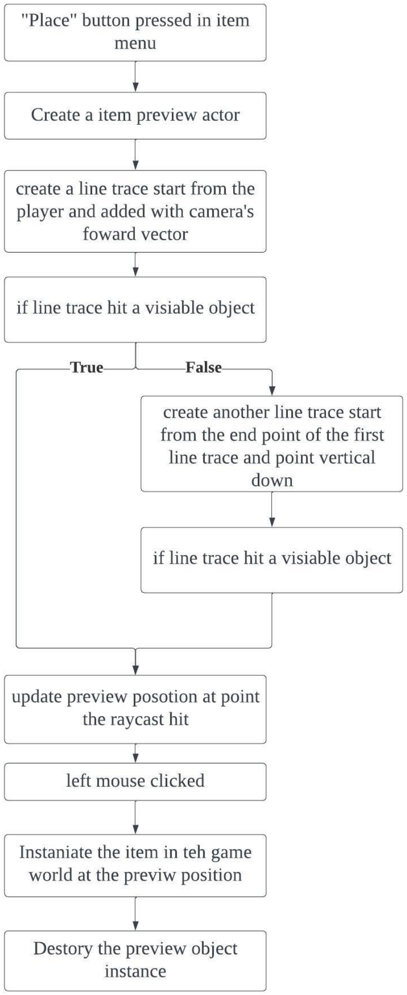

"Place" button pressed in item menu
Create a item preview actor
create a line trace start from the player and added with camera's foward vector
if line trace hit a visiable object
True
False
create another line trace start from the end point of the first line trace and point vertical down
if line trace hit a visiable object
update preview posotion at point the raycast hit
left mouse clicked
Instaniate the item in teh game world at the previw position
Destory the preview object instance

Figure 14: Flow diagram of the item placing logic in Unreal

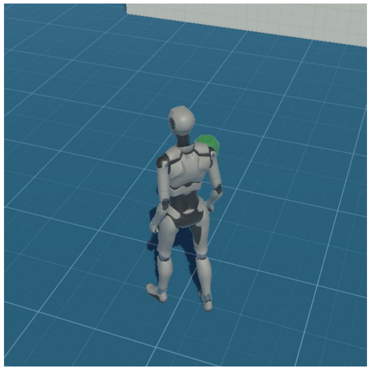


Figure 17: Item preview is blocked by the character in Unity.

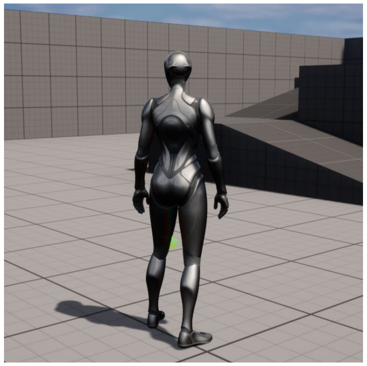


Figure 18: Item preview is blocked by the character in Unreal.

Evaluation

A working basic grid-based inventory is successfully achieved in both engines, where players can both get items from the game world into their inventory and place or drop items back into the game world. In addition, including the use of dragging and dropping items between different slots. However, the item placement in Unreal does not work as intended. The item preview is positioned at the wrong position when the first trace not hitting the ground (shown in Figure 15).

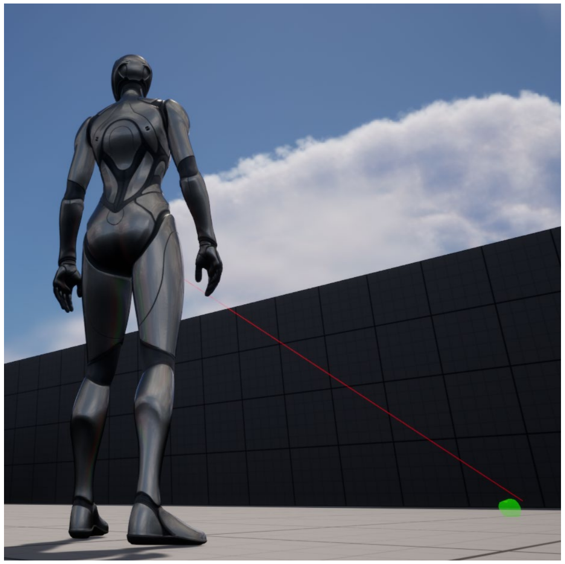


Figure 14: The item preview is positioned in the unintended position in Unreal.

Further Improvements

A quick slot can be implemented to enable players to navigate the item without opening the inventory such as the hot bar in Minecraft (Figure 15). Moreover, a weight limit bar like the weight limit in Baldur's Gate 3 (Figure 16) also can be implemented to stop the player from picking up items when the limit is reached.

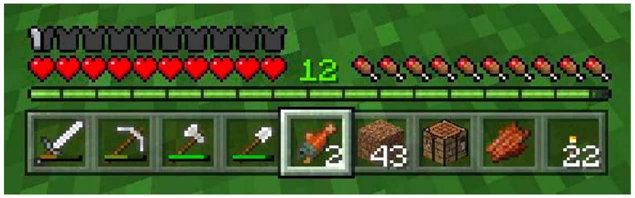


Figure 15: Toolbar in Minecraft. (Mojang Studios, 2009). [image from Reddit]

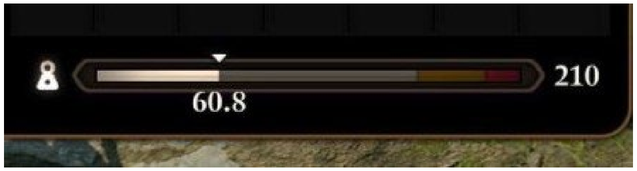


Figure 16: Weight limit bar in Baldur's Gate 3. (Larian Studios, 2023) [image from DOT ESPORTS]

For the implemented functionalities. Firstly, dropping items from inventory can be improved as the implementation only spawns the item objects in front of the character. Checks such as checking whether the character is standing near a wall are missing, so the object may spawn at the other side of the wall. Secondly, placing items also can be improved as the item preview is blocked by the character, so it's hard for the player to place the item (Figures 17 & 18).

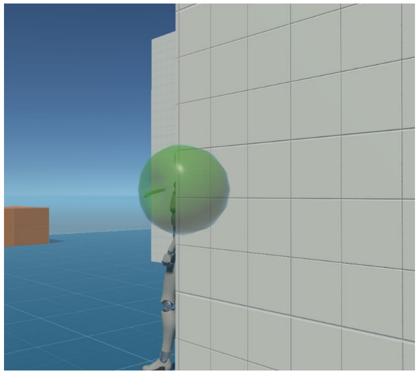


Figure 19 The item preview appear on the wall that blocked ray from the camera.

Furthermore, the interaction check should be able to ensure that the player only interacts with the item closest to the character when multiple items are presented in the range.

Implementation comparison between engines

Firstly, for the interaction system, using a separated empty game object as the range in Unity makes it easier to navigate the range than doing a tracing function in Unreal, as the range is visible in the scene view so the user can change the position of the range by moving in the object without entering play mode every time in Unity while Unreal required to change the variables in the properties of the player blueprint and able to see the range in player mode only.

Secondly, the implementation of the item in a slot is different, in Unity, an item image object will be created in a slot when a new item is added to the inventory, and drag-and-drop is created by changing the parent of the item image. In Unreal, every slot widget already has a non-visible empty item image and text by default, so the item will be displayed by setting the corresponding image and quantity of the item when an item is added to the inventory. Also, passing the item image and quantity data between the slots to implement the drag and drop.

Last but not least, the tracing of the item preview in item placement is different between Unity and Unreal, In Unity, the ray of the ray casting is set by the "ScreenPointToRay" function of the camera, which the ray is forced to start from the camera, which caused the problem shown in Figure 19 that the preview will appear on the wall that blocked the camera. While the start point and end point of the line tracing can be determined in unreal, which can avoid the problem in Unity.

References

Hompstead, J. (2023) LECTURE #6: EVENTS & MORE. *Game engine programming*[online]. Available from:https://blackboard.uwe.ac.uk/ultra/courses/_361722_1/cl/outline [Accessed 8 December]

Hompstead, J. (2023) LECTURE #7: DATA & ASSETS. *Game engine programming*[online]. Available from:https://blackboard.uwe.ac.uk/ultra/courses/_361722_1/cl/outline [Accessed 8 December]

Hompstead, J. (2023) LECTURE #8: DESIGNER TOOLS. *Game engine programming*[online]. Available from:https://blackboard.uwe.ac.uk/ultra/courses/_361722_1/cl/outline [Accessed 8 December]

Coco Code(2022). Unity INVENTORY: A Definitive Tutorial. *YouTube* [video]. 29 September. Available from: https://www.youtube.com/watch?v=oJAE6CbsQQA&t=1446s [Accessed 26 December]

Coco Code(2022). Drag and drop in Unity UI - create your own inventory UI !. *YouTube* [video]. 8 September. Available from: https://www.youtube.com/watch?v=kWRyZ3hb1Vc&t=2s [Accessed 26 December]

Dan Pos(2022). Unity Interaction Tutorial | How To Interact With Any Game Object (Open Chests & Doors etc). *YouTube* [video]. 3 April. Available from: https://www.youtube.com/watch?v=THmW4YolDok [Accessed 3 January]

Seabass (2020). Unity RTS - Building Placement Tutorial. *YouTube* [video]. 8 August. Available from: https://www.youtube.com/watch?v=Omu0A4Mk5pE [Accessed 6 January]

Ryan Laley(2022) Unreal Engine 5 Tutorial - Inventory System Part 1: Inventory Component. *YouTube* [video]. 11 October. Available from: https://www.youtube.com/watch?v=vHT4MhmwacE&t=496s [Accessed 12 December]

Ryan Laley(2022) Unreal Engine 5 Tutorial - Inventory System Part 2: Interaction System. *YouTube* [video]. 12 October. Available from: https://www.youtube.com/watch?v=F0Q_6mOlc-M&t=161s [Accessed 12 December]

Ryan Laley(2022) Unreal Engine 5 Tutorial - Inventory System Part 3: Item Data Component. *YouTube* [video]. 24 October. Available from: https://www.youtube.com/watch?v=TKtvgRUTuuA&t=2s [Accessed 12 December]

Ryan Laley(2022) Unreal Engine 5 Tutorial - Inventory System Part 4: Item Replication. *YouTube* [video]. 18 October. Available from: https://www.youtube.com/watch?v=4bXYl11sFew&t=6s [Accessed 12 December]

Ryan Laley(2022) Unreal Engine 5 Tutorial - Inventory System Part 5: Add to Inventory *YouTube* [video]. 31 October. Available from: https://www.youtube.com/watch?v=-cdzw9YE-MM&t=1022s [Accessed 13 December]

Ryan Laley(2022) Unreal Engine 5 Tutorial - Inventory System Part 7: Inventory Widget. *YouTube* [video].25 November. Available from: https://www.youtube.com/watch?v=yx3umVORujE&t=552s [Accessed 13 December]

Ryan Laley(2022) Unreal Engine 5 Tutorial - Inventory System Part 8: Drag and Drop Widget. *YouTube* [video]. 6 December. Available from: https://www.youtube.com/watch?v=QKmqjE-ZDrM&t=364s [Accessed 14 December]

Ryan Laley(2022) Unreal Engine 5 Tutorial - Inventory System Part 9: Drag and Drop. *YouTube* [video]. 16 December. Available from: https://www.youtube.com/watch?v=v94b-yp3Umw&t=2s [Accessed 14 December]

Ryan Laley(2022) Unreal Engine 5 Tutorial - Inventory System Part 10: Drop and Swap. *YouTube* [video]. 19 December. Available from: https://www.youtube.com/watch?v=vHT4MhmwacE&t=496s [Accessed 16 December]

Ryan Laley(2022) Unreal Engine 5 Tutorial - Inventory System Part 11: Container Inventory. *YouTube* [video]. 26 December. Available from: https://www.youtube.com/watch?v=l6eXqK0zHHI [Accessed 16 December]

Ryan Laley(2023) Unreal Engine 5 Tutorial - Inventory System Part 12: Container Drag Drop. *YouTube* [video].2 January. Available from: https://www.youtube.com/watch?v=Sj4pWx84dH8 [Accessed 17 December]

Ryan Laley(2023) Unreal Engine 5 Tutorial - Inventory System Part 13: Action Menu. *YouTube* [video]. 10 January. Available from: https://www.youtube.com/watch?v=qpk758MvW24 [Accessed 21 December]

Ryan Laley(2023) Unreal Engine 5 Tutorial - Inventory System Part 14: Dropping Items. *YouTube* [video]. 18 January. Available from: https://www.youtube.com/watch?v=3c4ZRsqjGB8 [Accessed 22 December]

Ryan Laley(2023) Unreal Engine 5 Tutorial - Inventory System Part 15: Display Message. *YouTube* [video]. 30 January. Available from: https://www.youtube.com/watch?v=4jqk8RK6sV4 [Accessed 22 December]

Ryan Laley(2023) Unreal Engine 5 Tutorial - Inventory System Part 16: Use Item. *YouTube* [video]. 11 October. Available from: https://www.youtube.com/watch?v=aZOI0Pn6q40 [Accessed 23 December]

Matt Aspland (2021) How To Place Down And Rotate Objects In Real Time - Unreal Engine 4 Tutorial. *YouTube* [video].12 February. Available from: https://www.youtube.com/watch?v=fQjTrHYUygQ [Accessed 26 December]

Seredias (2022) Unreal Engine Placing Furniture Tutorial. *YouTube* [video].8 October. Available from: https://www.youtube.com/watch?v=fQjTrHYUygQ [Accessed 26 December]

Viana,B , Andrews.H and Ernst,A. (2023) Baldur's Gate 3: How ‘add to wares' works and when to use it in BG3. **DOT ESPORTS** [online]. Available from: https://dotesports.com/baldurs-gate/news/baldurs-gate-3-how-add-to-wares-works-and-when-to-use-it-in-bg3 [Accessed 14 January]

Iliveformyotp.(2020) Any game designers here? Need some feedback on my redesign of Minecraft hotbar UI.[Reddit] Available from: https://www.reddit.com/r/UI_Design/comments/nr1pkd/any_game_designers_here_need_some_feedback_on_my/ [Accessed 14 January]