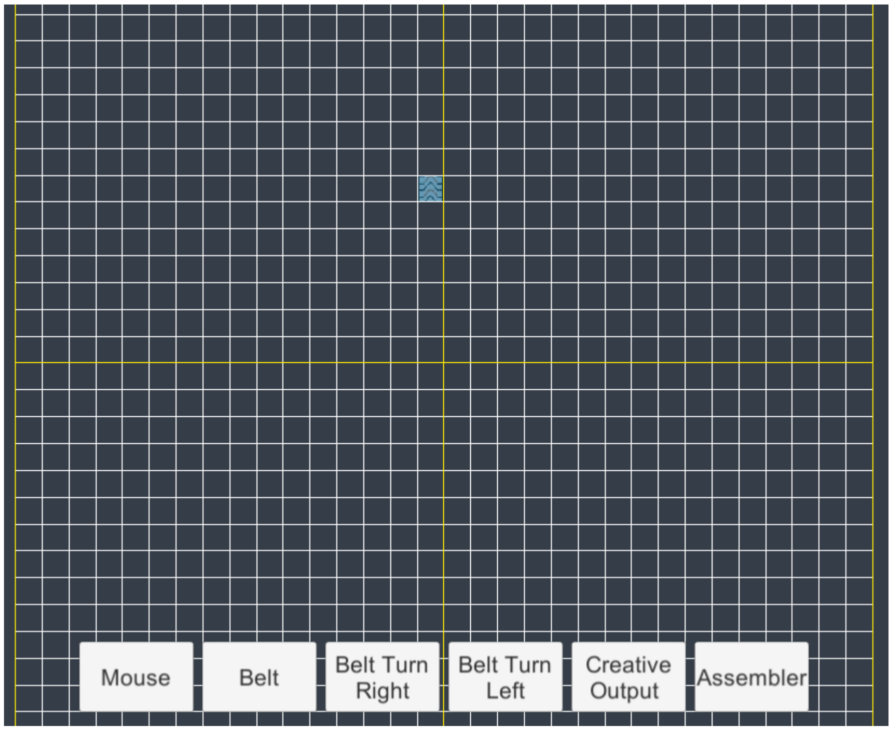
Brandon Coffey

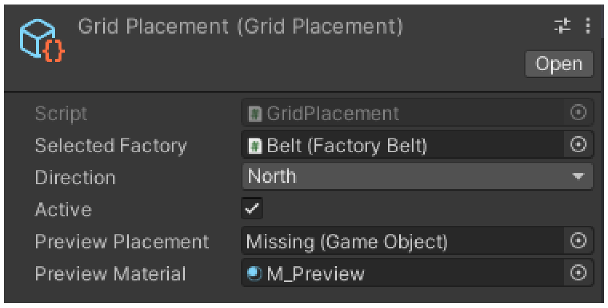
Project 03 Planning – Factorio Systems

For Project 03, I worked to recreate the Belt and Automation system from Factorio.

To start, I worked on a Grid and Chunk system, which is generated at runtime. I can specify how large the chunks are and for now, it only generates 4 chunks (Potential for unlimited chunks as the player explores). These chunks consist of 256 Grid Objects, though they are only instantiated when an object is placed. When the player clicks, I send a ray-cast and see what collider was hit. If the collider was one of the chunks, a grid object is created that location.



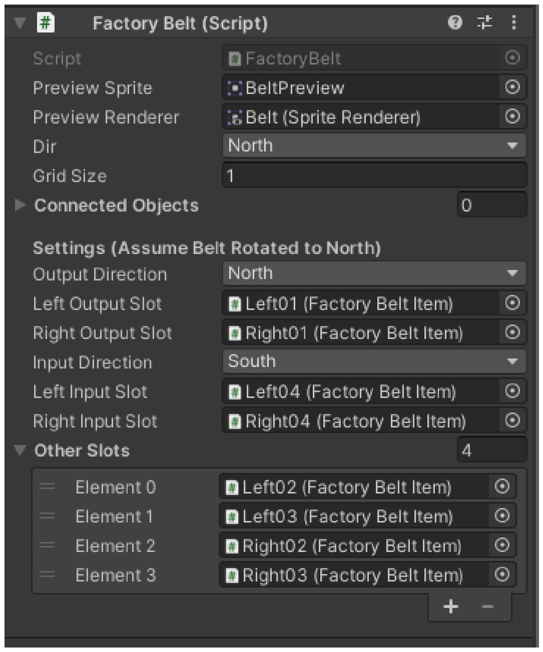
To setup a diverse placement system, I created a Scriptable Object called Grid Placement. This object contains data for the current selected factory object, as chosen by the 6 buttons at the bottom of the screen, and the current rotation, as changed by the key R. This data is sent to the created grid object and used to instantiate the selected factory object under that grid position.


Grid Placement (Grid Placement)

Open

Script	GridPlacement
Selected Factory	Belt (Factory Belt)
Direction	North
Active	✓
Preview Placement	Missing (Game Object)
Preview Material	M_Preview

After this was set up, I needed some factory objects to work with, starting with the belt. Factorio has three different kinds of belts: straights, left turns, and right turns (I will not focus on different speeds or underground belts). These are all three their own factory object, but essentially, they consist of input slots, output slots, and in-between slots.



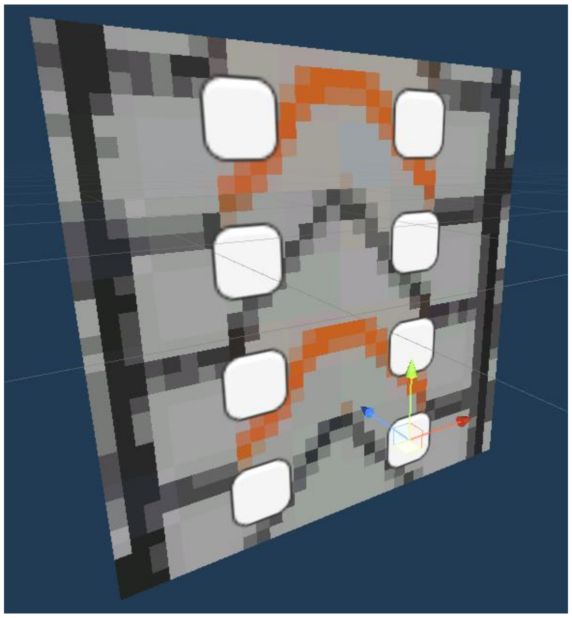
Factory Belt (Script)

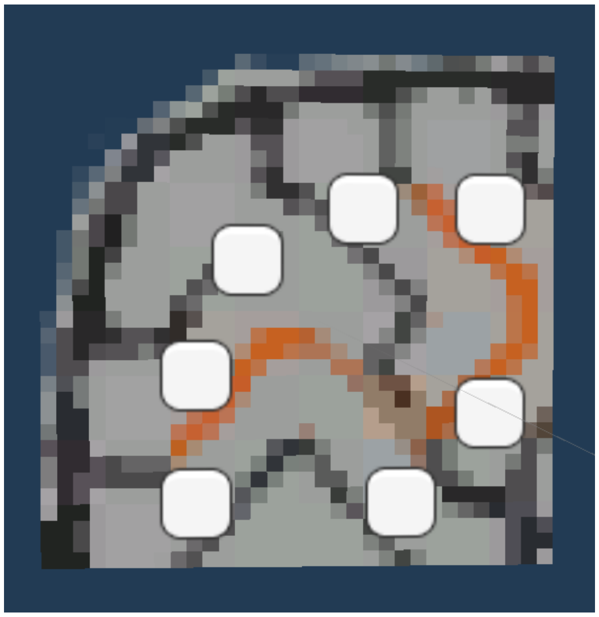
Script	FactoryBelt
Preview Sprite	BeltPreview
Preview Renderer	Belt (Sprite Renderer)
Dir	North
Grid Size	1
Connected Objects	0

Settings (Assume Belt Rotated to North)

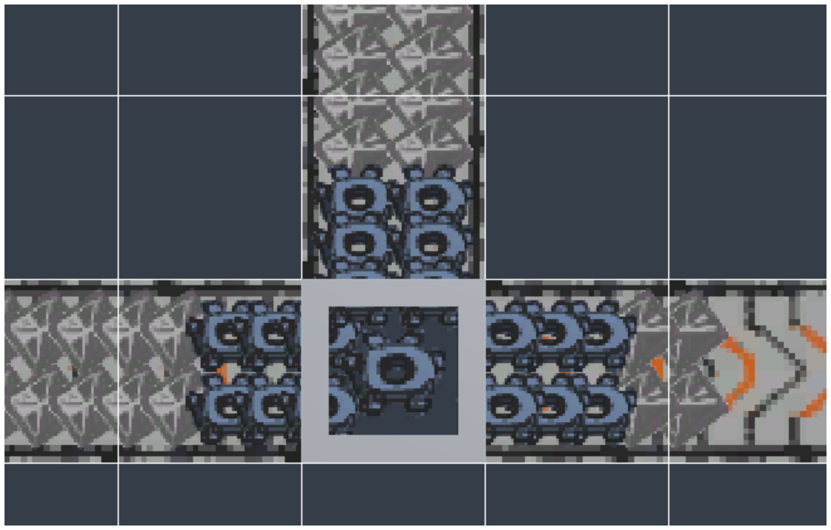
Output Direction	North
Left Output Slot	Left01 (Factory Belt Item)
Right Output Slot	Right01 (Factory Belt Item)
Input Direction	South
Left Input Slot	Left04 (Factory Belt Item)
Right Input Slot	Right04 (Factory Belt Item)
Other Slots	4
Element 0	Left02 (Factory Belt Item)
Element 1	Left03 (Factory Belt Item)
Element 2	Right02 (Factory Belt Item)
Element 3	Right03 (Factory Belt Item)

Each slot on the belt can contain an item and stores the previous and next slot. Each belt slot has their own update function that runs if they have an item and plays a simple animation to move the item along. If the next slot is empty, they will transfer their item to that slot.

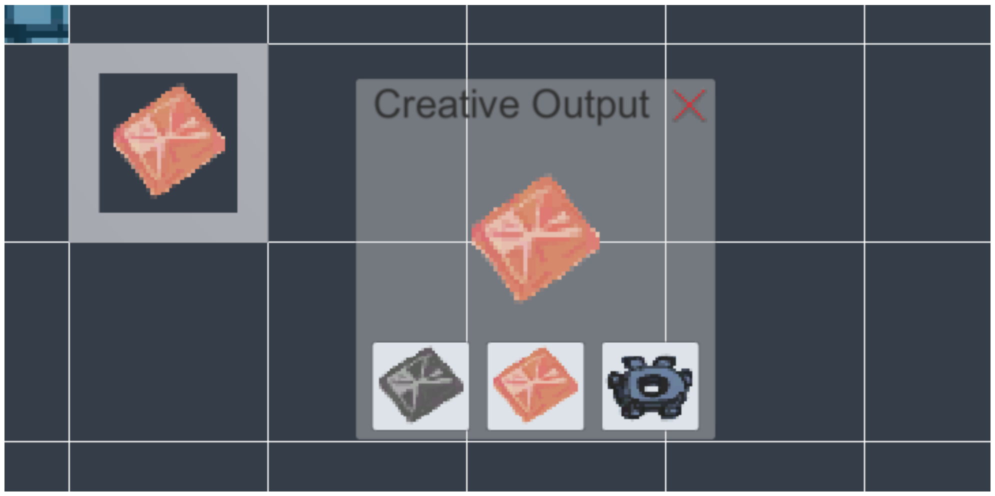




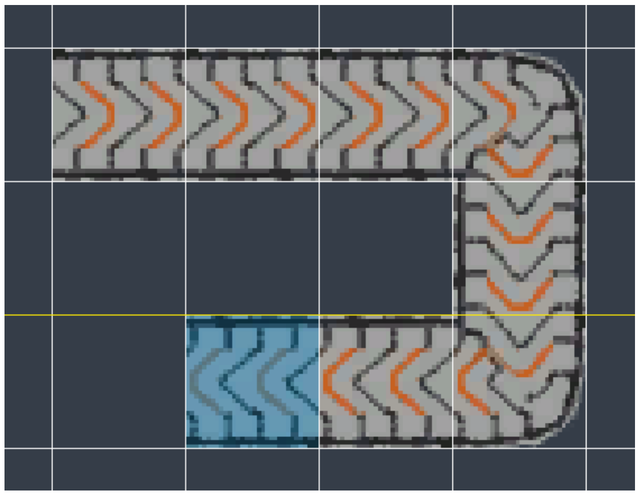
When a factory object is placed, it must take the information of its neighbors to work with adjacent objects. For the belt, its input and output slots connect to neighboring belts as soon as either belt is placed next to each other. I then created a creative output block to let me test items on the belts. This was the simplest object, as it just contains a single item that it constantly outputs into neighboring belts.



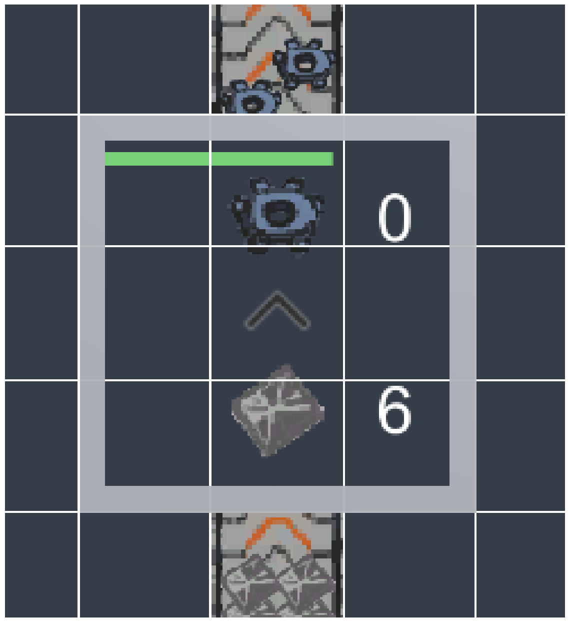
To better match Factorio's gameplay, I created a few objects to work with, being iron plates, copper plates, and gears. When the player clicks on the creative output menu, a UI pops up that allows the player to change what item is stored in the creative output. This UI panel is stored under a singleton panel to allow any object to activate the panel and send its data through, removing any messy references from the UI.



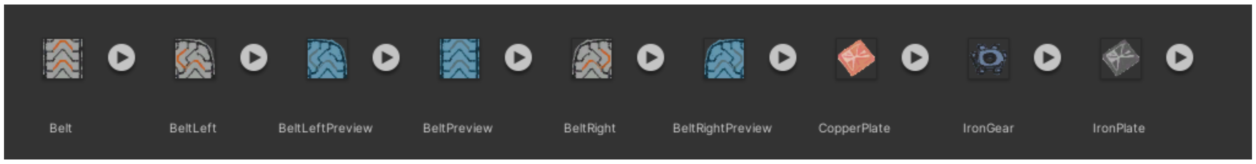
To add to the appeal, I also implemented some sound effects directly from Factorio, as well as some visual feedback in the form of a blue colored preview of where the object will be placed. This was accomplished by matching a game-object's art to the selected object, the rotation to the current rotation value, and the position to where the mouse was hovering over. This preview was desperately needed and significantly made the game feel more like Factorio, as well as the placement and breaking sounds of each object.



For the final object I recreated from Factorio, I worked on an assembling machine. This machine is programmed with a recipe item and an output item and displays both on the factory object itself. For simplicities sake, I only implemented a recipe to turn iron plates into gears, as shown below. This works by identifying input and output belts and then waiting for an input that matches iron plates. It has a stored crafting time and displays a green bar at the top to show progress on its progress.



Last but certainly not least, I drew up some art for the game, as basic unity objects and sprites do not get much appeal across. This art closely matches the actual objects in Factorio, and the belts could be animated eventually.



Thus, with the factory complete, my job was done. A core element of Factorio that I did not implement was the inserter, as I figured that would be more difficult than anything else. I will work on this sometime in the future, but for now I was inspired by games like Satisfactory and Dyson Sphere program to have direct input and output through the belts. Either way, I have created a working belt system with an output block and an assembler. The grid system is expandable and everything is connected through scriptable objects, making this a dynamic system that could be expanded upon.