FactoryBuilding system deconstruction

Evan Daveikis 991 721 245

The FactoryBuilding system will be the basis of what the player interacts with in the game.

What problem does the system solve?

The game needs to way to manage multiple different types of buildings and remain extensible. Buildings need to have inputs and outputs, and each will have different logic depending on how they handle products.

What does it do? How does it do these things?

This system will allow for buildings of varying sizes, with any number of inputs/outputs and allowing each to have custom logic to serve its functions.

It will do these things by breaking each building down into smaller pieces: Each building will be made of one or more "Tiles", and each tile can have 0 or more inputs and outputs. For example, a simple conveyor will be a single tile with 1 input and 1 output, while an assembler may have 4 tiles, with 2 inputs and 1 output.

These buildings will then have functions for when items enter/leave, each update cycle, etc that subclasses can use to implement their required functionality.

Which parts of this system will be achieved by composition (components) and which parts will be achieved by inheritance? What components will be required?

Each building will be composed of a subclass of FactoryBuilding, one or more tiles, a SpriteRenderer, and a collider to support mouse events. This makes up the physical composition of the building objects.

Building-specific logic will be implemented through inheritance. As an example, an Assembler will inherit from FactoryBuilding, overriding the events for when items enter. When it detects enough

items have entered, it will send another item out through the building's output. A conveyor would detect items entering, wait some time, and then send the item out through the output.

What is the functionality of the base classes?

The base FactoryBuilding class will keep track of all the tiles in the building, the building's rotation, and any other information required for all buildings (like overridable functions, utility functions, a list of stored products, etc).

What is the specific functionality of each inheriting class?

See Planning.txt for specifics, but here is the high-level overview:

Outbox: The final destination for products. Products enter and add to the global store/count of that product, then are destroyed.

Miner: Can only be placed on resource WorldTiles. Sends that product out through the Output at specified intervals.

Conveyor: Used to send items from one place to other. Accepts items on three sides, outputs them at the last side.

Assembler/combiner: Takes in items and produces new ones. 2x2 building, accepts 4 inputs on one side, outputs them on 1 tile of the opposite side

Underground conveyor: Used to get items across conveyors. Actually 2 buildings: an input side and an output side. Input has 1 input, Output has 1 output.

Splitter: 2x1 building: has 2 inputs at the back and 2 outputs at the front. Will swap sides when outputting (splits inputs equally between both sides)