



GROUP 9

# Our team of 4

- Jennifer
- Michael
- Emily
- Amanda



# Nazo: Original Design

- Narratively driven isometric 3D puzzle game with a farming simulator attached to it for progression.
  - Two world states: Puzzle and Farm
  - The player's goal would be to complete the puzzle and utilize the farm to grow the crops required to progress.
- Our original project ran into several problems however...

# Problems With Original Design

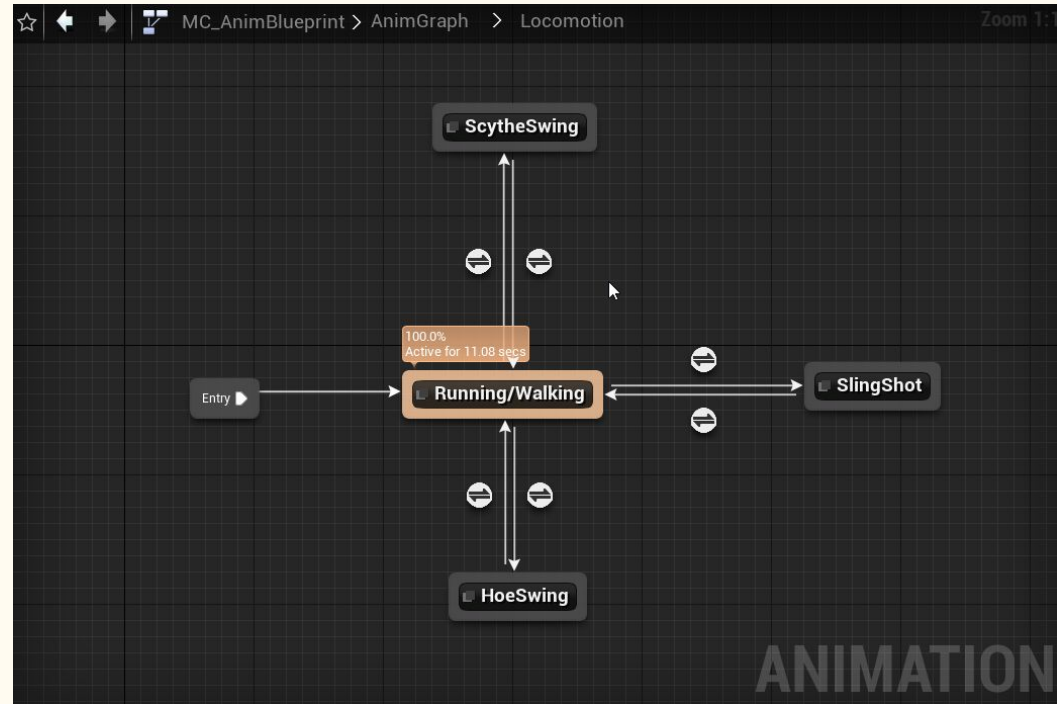
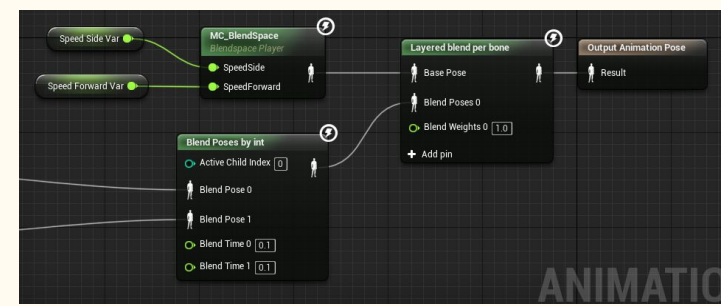
- **Narratively Driven Puzzle Game**
  - Originally we wanted a fully fleshed out story supported by a fully fleshed out puzzle.
  - Each of these goals require not only a large amount of development time, but a large amount of playtime to get players invested.
- **Bad style of game for the purpose of this class**
  - Great Idea for a game, but not suited for this class.
    - Takes too long to grab the player's interest
    - Designed mechanics were too restrictive for a 5-15 minute experience.
- **Over-scoped**
  - The game we originally set out to design involved far too much time investment than what we could devote in the timeframe of this course.

# Refactored: The New Nazo

- Farm based wave game
  - Grow crops
  - Beetles come at night to destroy your crops so you must defend against them
- Goal of the game:
  - Farm crops to make enough money to save the farm

# AI - Animation

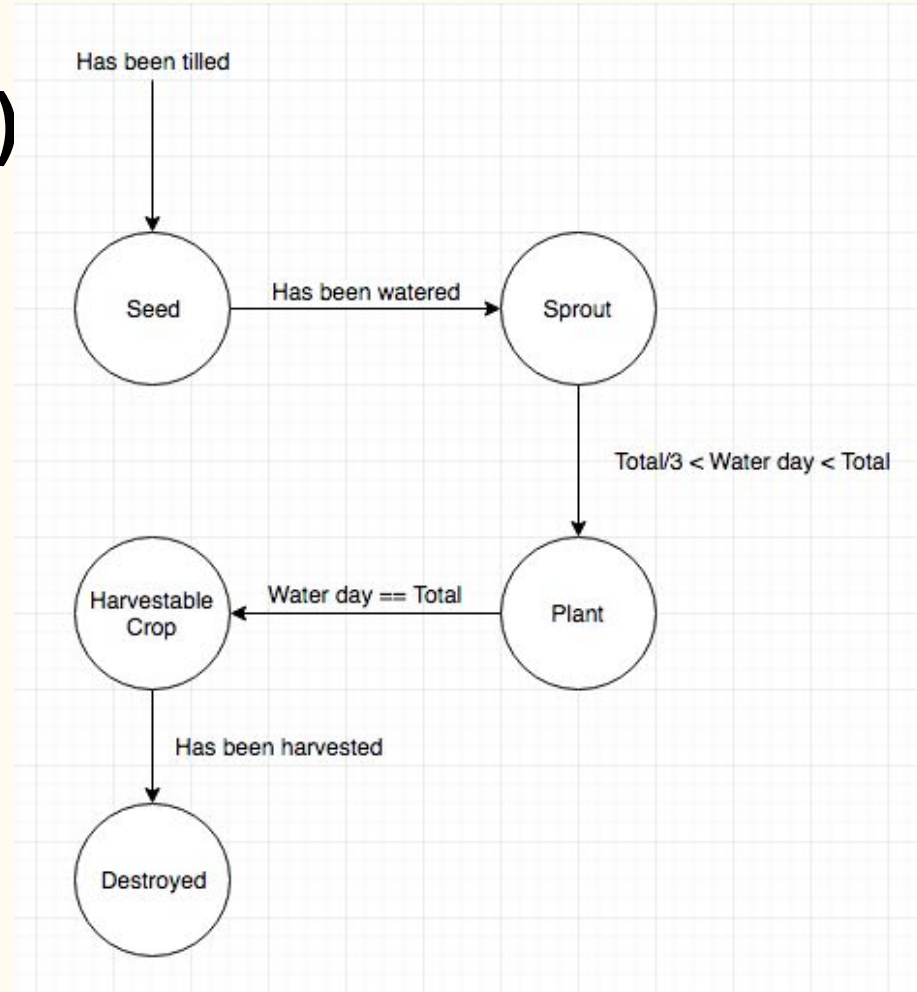
- Alternates between animation states
- Merges animations
  - Merges walking/running blend states with attack animations
  - Breaks up top from bottom based on a bone in skeleton
- AnimGraph determines state to be in based on boolean values, and if the animation had been completed



# AI Design - Crops (FSM)

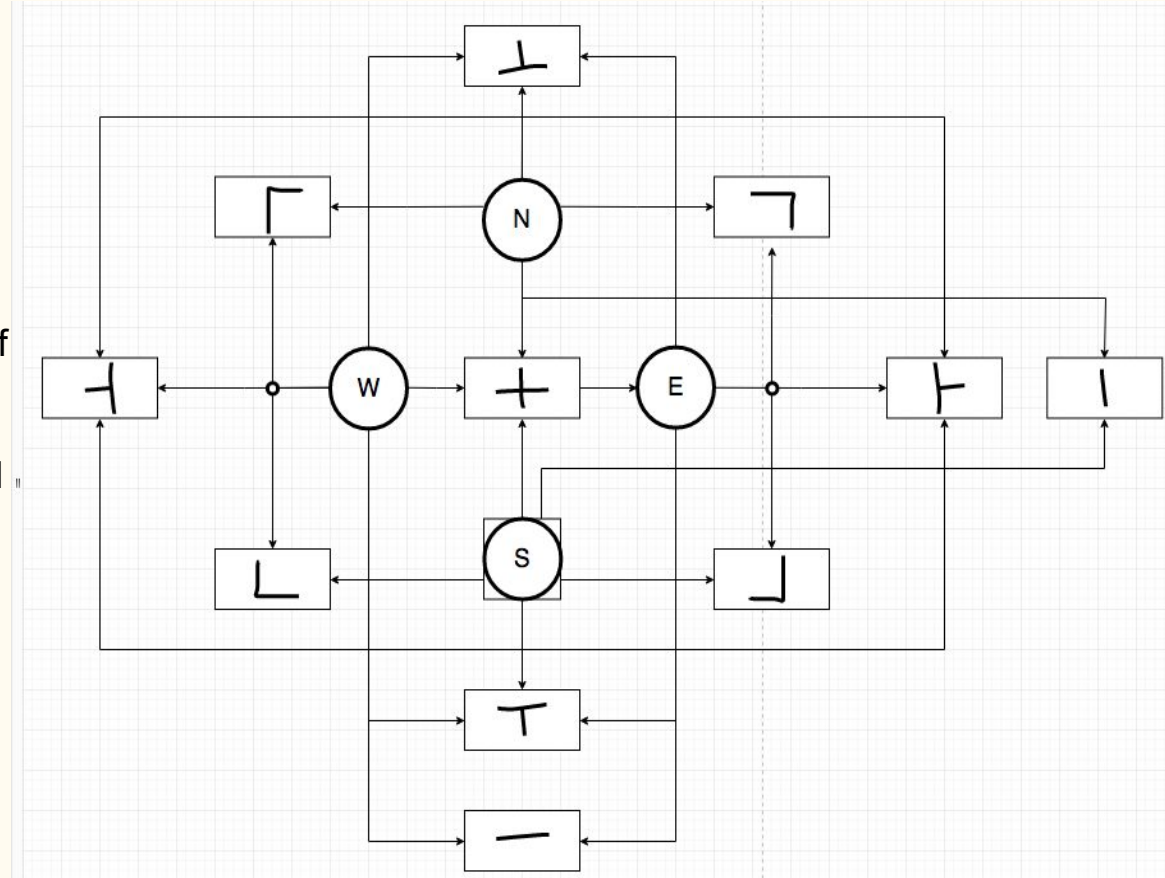
- Crop (FSM)

- Seed state - no water/plot tilled
- Sprout state - watered at least once
- Plant state - Total water days not met, greater than 1/3rd of total
- Crop - Can harvest, water days complete, is destroyed on harvest



# AI Design - Fence (FSM)

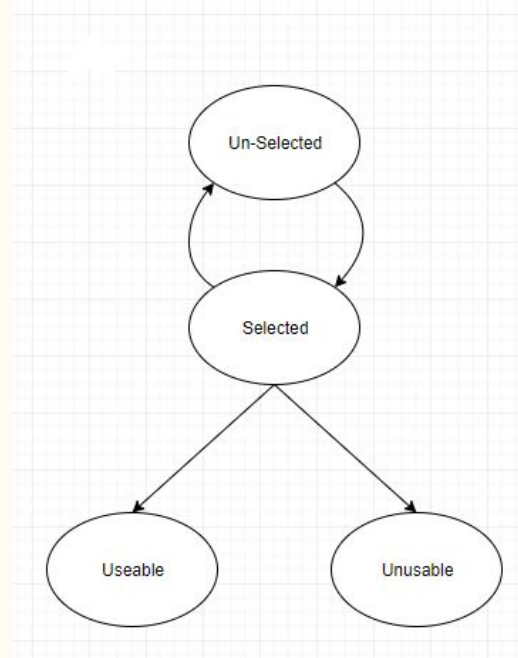
- Fence (FSM)
  - Uses collision detection
    - Detects other fences
  - Based on N/S/E/W
    - Rotates 4 meshes
    - Picks mesh on number of points it hits
  - 11 States total
  - Each coordinate toggles a bool value for if a fence is detected





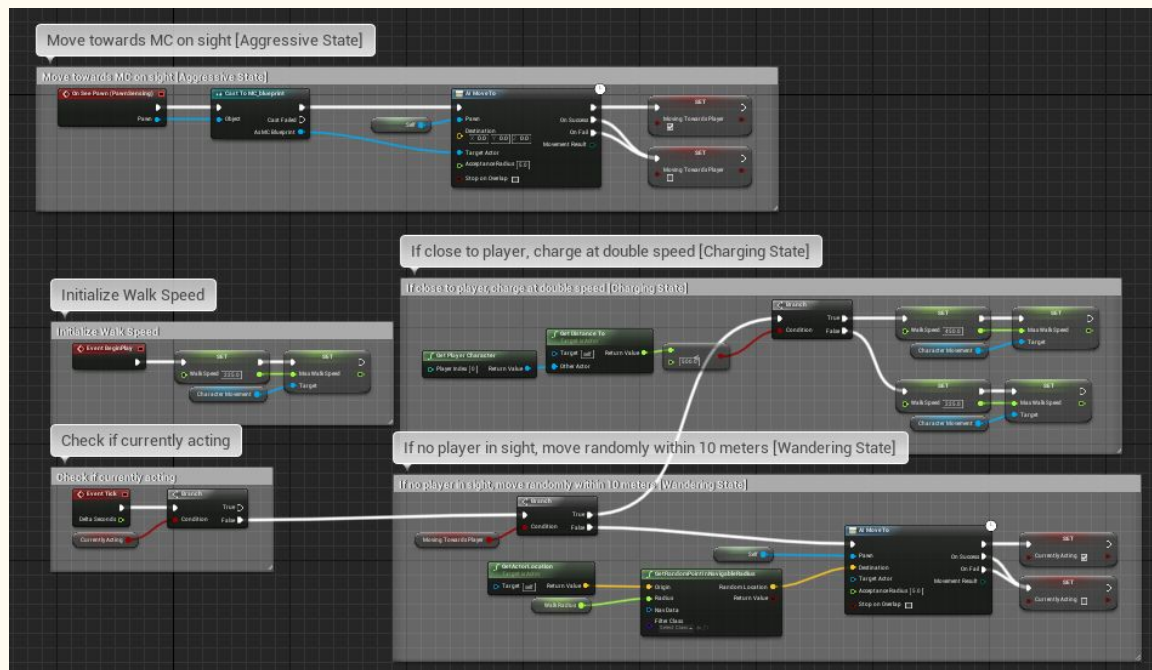
# AI Design - Grid AI (State Machine)

- Updates on gridtick, instead of FPS
- Checks if the forward vector from the player is hitting it or not, and highlights the panel its hitting
- Basis for most planting and building
- Generates on game start, can be modified via data driven variables to certain sizes and shapes
- Breaks blocks into ones that are usable and ones that are not
  - If useable, can place plants based on the block type
    - If soil, can place a plant
    - If not soil, can place traps/fences/defenses



## Iteration 1: Beetle AI Design [Basic FSM]

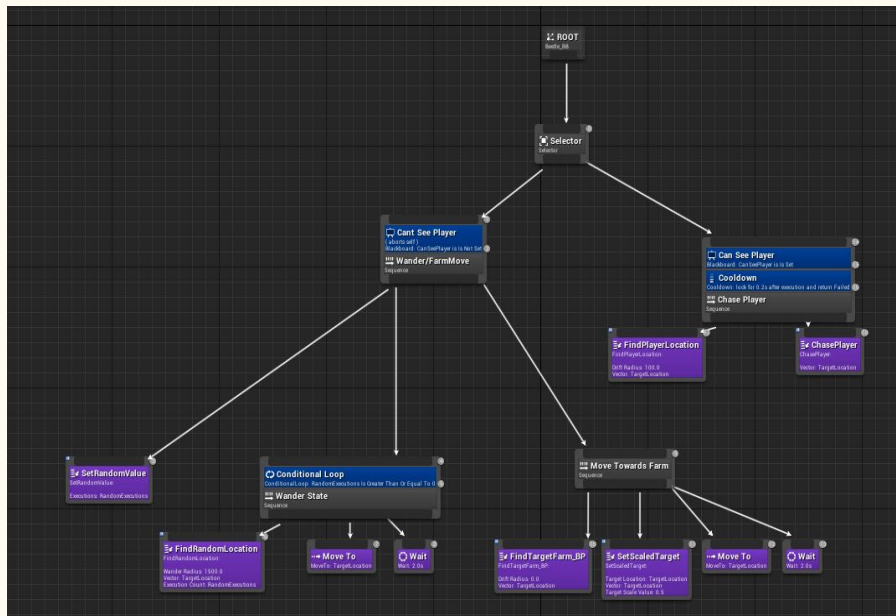
## Original Beetle Design prior to refactor:



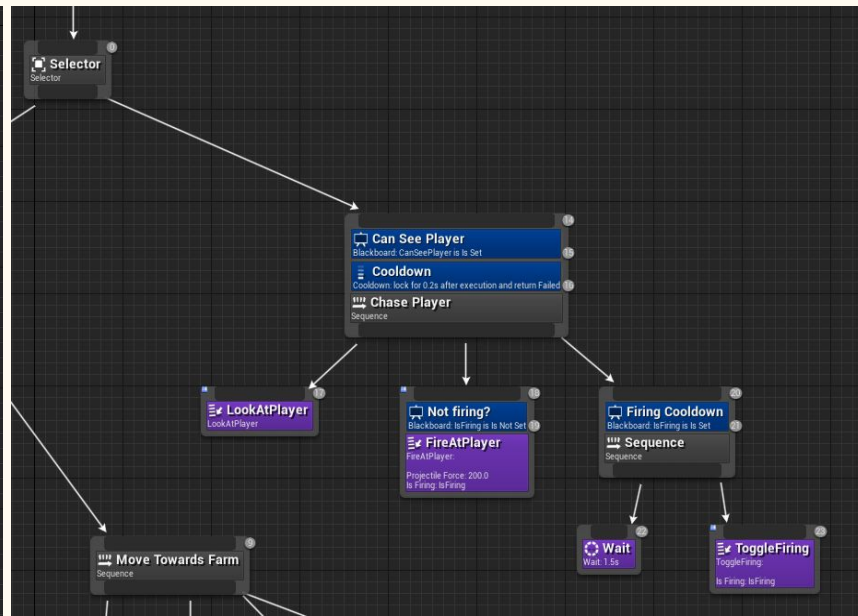
- Operated as a very simple Finite State Machine
- Worked fine but was inefficient [Updates per tick]
- Refactoring needed more complex behaviors

# Iteration 2: Beetle AI Design w/ Wasp [BT]

BeetleAI BT - Parent

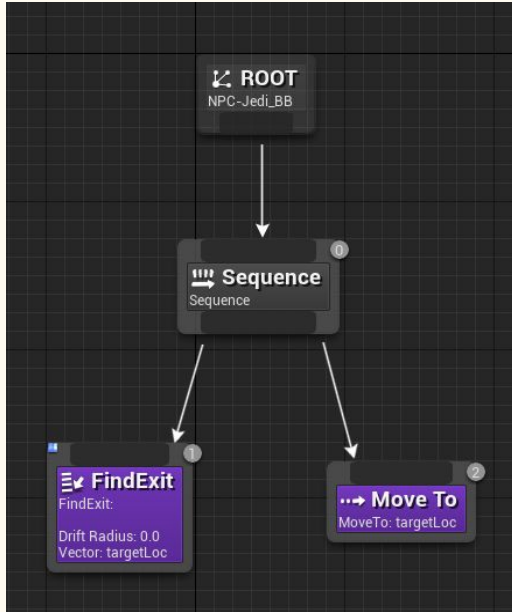


WaspAI BT Modifications - Child



# AI Design - BT simplification

NPC - Shop/MarketMan/Jedi



- Swapping to Behavior Trees allowed for adding complex behaviors
- It also allowed for establishing a Parent/child relationship with AI patterns
- Ultimately they are nicer to work with and understand.

# Playtest Day

- Clarify game
  - Make tutorial more clear
  - Make shop more clear
  - Add in crosshairs
- Found bugs
  - Fix aiming
  - Fix beetle waves not spawning
  - Fix pending kill error
  - Fix flying NPC
  - Fix watering can planting seeds
- Improve
  - Make a hotbar
  - Shop closes when walk away
  - Shop button detection bigger
  - Add melee weapon
  - Speed up animation
- Things that were fixed:
  - Fix flying NPC
  - Pending kill error
  - Make a hotbar
  - Shop closes when walk away
  - Shop button detection bigger
  - Fix watering can planting seeds

# Playtest and Iterations

- Iteration 1

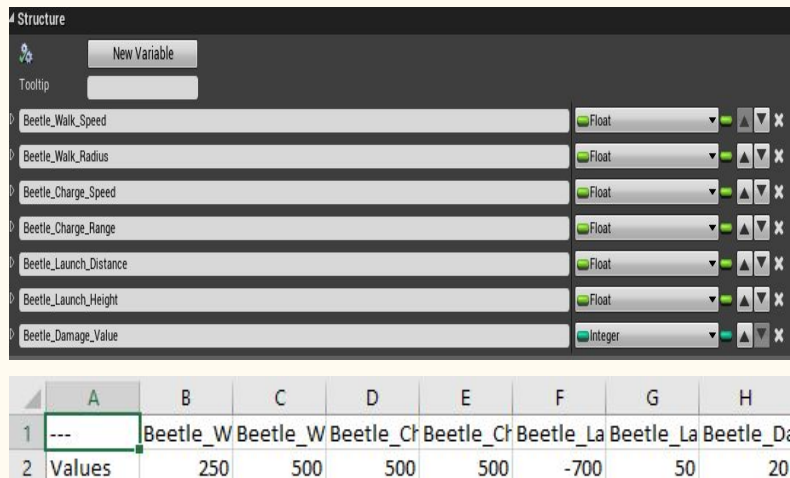
- Bug fixes
- Updated UI
- Hotbar
  - Great success
- Check icons
- New Inventory
- Check market functionality
- Smoother beetle movement
  - Attacks plants
- Plant health

- Iteration 2

- Icon testing
- Hotbar
  - Great success
- Market functioning much more smoothly
- Beetle waves got too hard
  - Need to be balanced
- NPC got stuck on fences when it tried to move away for the night
- Less confusion on controls, but needs more consolidation
- Need to test secondary enemy AI

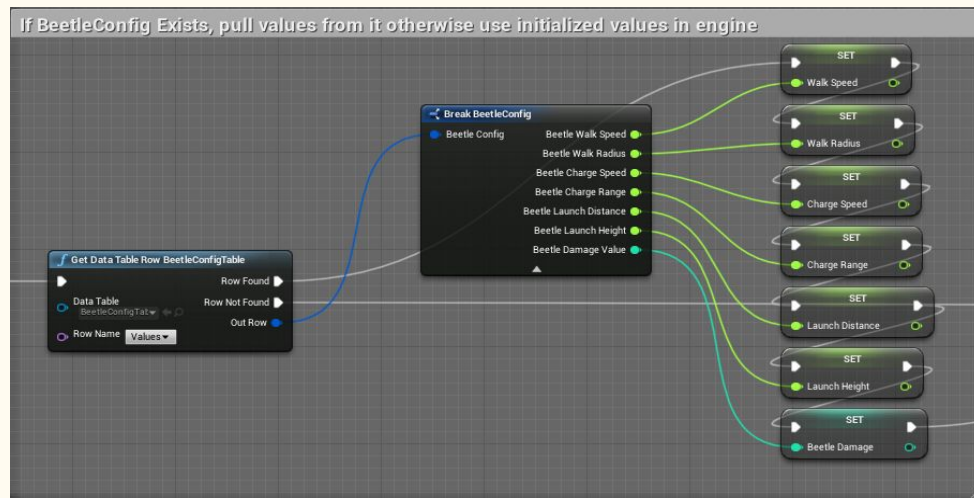
# Data Driven Design

- We utilized Unreal's CSV importer to initiate the values of all enemy attributes, allowing us to set them outside the editor.



The image shows the Unreal Engine Structure panel on the left, listing variables for a beetle: Beetle\_Walk\_Speed (Float), Beetle\_Walk\_Radius (Float), Beetle\_Charge\_Speed (Float), Beetle\_Charge\_Range (Float), Beetle\_Launch\_Distance (Float), Beetle\_Launch\_Height (Float), and Beetle\_Damage\_Value (Integer). On the right is a CSV data table with 8 columns (A-H) and 2 rows. Row 1 contains headers for each attribute, and Row 2 contains their initial values.

	A	B	C	D	E	F	G	H
1	---	Beetle_W	Beetle_W	Beetle_Ch	Beetle_Ch	Beetle_La	Beetle_La	Beetle_Da
2	Values	250	500	500	500	-700	50	20

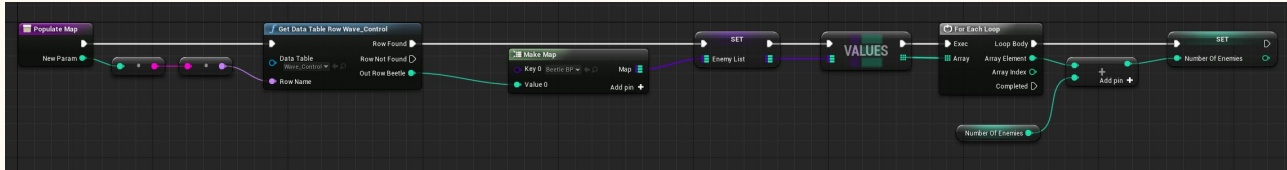


	Beetle_Walk_Speed	Beetle_Walk_Radius	Beetle_Charge_Speed	Beetle_Charge_Range	Beetle_Launch_Distance	Beetle_Launch_Height	Beetle_Damage_Value
Values	250.000000	500.000000	500.000000	500.000000	-1000.000000	50.000000	20

# Data Driven Design Continued

- **Wave Control**

- Used to control the number of beetles that spawn per wave
- Great for iteration if a specific wave was too easy/ too hard



Beetle	Number Of Enemies
1	3
2	4
3	5
4	5
5	6
6	7
7	7
8	9
9	8
10	10
11	10
12	10
13	11
14	11
15	12
16	12
17	13
18	13
19	14
20	15



# Issues with AI

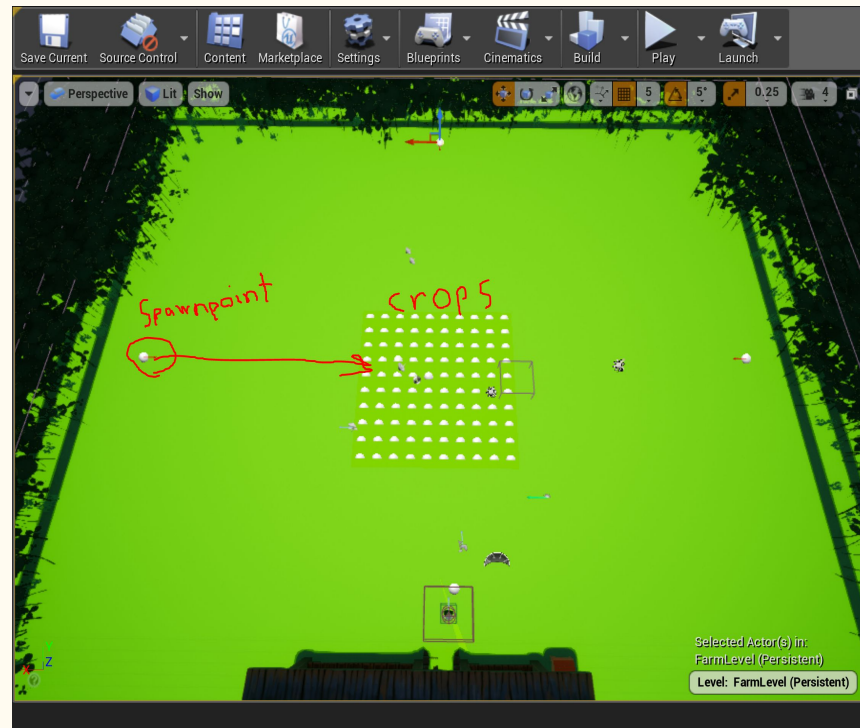
(Fixed) NPC launching when player ran into it

(WIP) Waves not spawning

(Fixed) Finding a way for the Beetle to navigate over to the crops from spawn point:

Target Location  $T$ , Beetle Location  $B_i$

$$(T - B_i) * 0.25 + B_i$$



# UI/Game/Balance Issues

- Finding a new UI system that can handle a large amount of items
  - Solved by changing the UI to a hot bar that interacts with the inventory
- Balancing the amount of income
- Number of enemies per wave

# Software Design - Good Design

Having a grid system to deal with the placement of objects

Using data tables for large amounts of data

OOP Practices - Parenting/Hierarchy/Interfaces

Custom events

Switching beetle/NPC from FSM to Behaviour tree - helped a whole ton with scaling

# Software Design - Bad Design

Scoping - we realized the game was going to be too big, and had to change from a puzzle type game to a wave defense in order to get it done on time

Scaling - Some things were planned out to scale as we added more to it, and some things did not

Player controls started getting overly complicated, we had to simplify them

Weapon toggle didn't scale well with increased items, needed to be swapped out with a hot bar

Dialogue had to be swapped to a data table, previous was print text variables, scaled badly

Some code repeated, wasted time copying from place to place, until we decided to use hierarchy instead

# Questions?