Progress Log (May)

Created by	Ⓞ Omar Owis
Created time	@May 3, 2023 1:58 AM
Tags	

Focus: Combat system pt3

May 1st 2023

Nothing was done on this day

May 2nd 2023

Nothing was done on this day

May 3rd 2023

- Last state of enum ability behavior is saved upon clicking off the ability
- Changed directory path for saved behaviors to "Assets/_Core/Scriptable Objects/AbilityBehaviors"
- Old behavior is deleted when changing behaviors in enum

May 4th 2023

- Bug fix: can select "(empty)" in ability behavior enum
- The selected ability behavior info is now displayed on the ability itself
    - Editing the ability behavior info on the ability changes the info on the actual ability behavior scriptable object

May 5th 2023

Nothing was done on this day

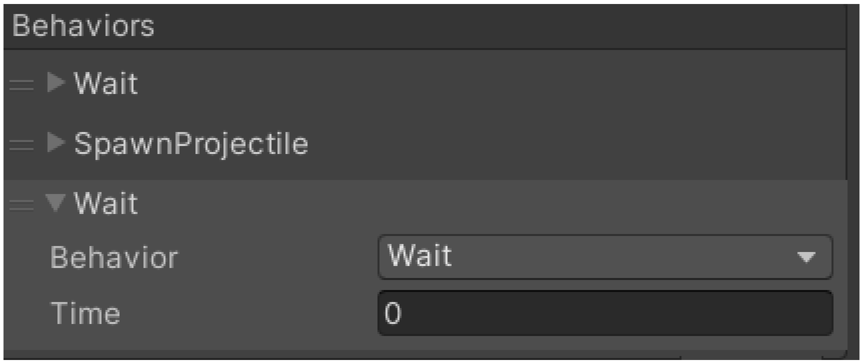
May 6th 2023

- Changed Ability Behavior save location to save on the ability asset itself, rather than saving to a directory file.
- Extended Ability Behavior enum to a reorderable list:
    - Functional adding of ability behaviors on ability
    - Semi-functional removing from arbitrary index, does not rename element
    - Functional changing of ability behavior (buggy)
    - Functional drawing and serialization of each ability behavior, works with lists elements
    - Semi-functional internal ability behaviors list: adding, removing, and changing of ability behaviors is updated in the internal ability behavior list. Does not swap in internal list
    - Functional last enum state saving
- Changed “(empty)” to “(remove)”
- TODO:
    - ~~Selecting “(remove)” on the enum removes element from internal and reorderable lists~~
    - ~~Swapping functionality~~
    - Rename ability behaviors to match current index when swapping and removing
    - ~~Hide info/variables of Ability Behaviors functionality~~

May 7th 2023

- Implemented hide info/variables of Ability Behaviors

- Bug fix: Selecting “(remove)” on the enum removes element from internal and reorderable lists
- Implemented ability behavior name label



Behavior: Wait

Time: 0

May 8th - 12th 2023

Nothing were done on these days

May 13th 2023

- Implemented swapping functionality in reorderable list
- Code refactoring for ability editor
- Fixed enum occasional incorrect naming bug
- Implemented index renaming when removing behaviors

May 14th - 18th 2023

Researching combat and game development theory

May 19th 2023

- Implemented character data scriptable object
- Implemented Input manager scriptable object
- Refactoring input code

- Changed and defined character stats (What is a Character?)
    - Health
    - Speed
    - Luck
    - Power
    - Cooldown Reduction
    - Defence
- Refactoring stat panel and tooltip code

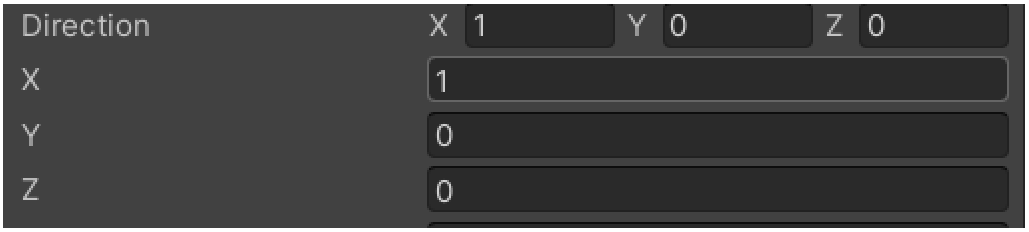
May 20th 2023

- Implemented camera locking and character locking when inventory is open
- Implemented character data resetting on application quitting
- Added Stat Duration ability behavior
- Added Changed callback to modifiable stat class. Is invoked when stat value is changed
- Added Change callback to character data object. Is invoked when any modifiable stat is changed.
- Added new Heal ability behavior
- Added new Damage ability behavior
- Bug Fix: When creating new ability behaviors, the selected id for old behaviors changes upon opening ability inspector.

May 21th 2023

- Added control movement state in player state machine
- Added Linear Move ability behavior
- BUGS TO FIX:

  - ~~Linear move ability behavior moves player relative to camera rather than relative to player~~
  - ~~Ability custom editor shows vector as an element and also shows vector children as elements~~



Direction X 1 Y 0 Z 0

X 1

Y 0

Z 0

  - ~~Rotating camera while performing linear move, changes direction of the path. (Same with dash)~~
  - Ability continues activation even after game exit

May 22th 2023

- Bug fixes:
  - Ability custom editor no longer shows children of an element
  - Linear move ability behavior moves player relative to camera rather than relative to player
  - Rotating camera while performing linear move, changes direction of the path. (Same with dash)
  - When switch both super and sub state in the same frame, the super state of the sub state changes accordingly
  - Abilities inside Combo Ability gets initialized correctly
- Code refactoring:
  - Combo handler is called from combo ability object.
  - Cooldown handler is called from ability object and combo handler singleton

May 23th 2023

- Implemented Spawnable Object mono-behavior class
    - Has functionality for spawning object in the world without mono-behavior
    - Has functionality for destroying it self without mono-behavior
    - Has functionality for following rigid-body parent (using rigid-body fixed link)
- Implemented Pull collider prefab and Push collider prefab
    - Given the force, size of the collider, and position in the world, the prefab will pull/push any rigid-body into the center
    - Objects inherits from Spawnable Object mono-behavior class to support spawnability and remove on non-mono-behavior classes (Ability Behaviors)
- Implemented push and pull colliders ability behaviors
    - Has functionality to draw gizmos in editor scene for debugging purposes
    - Behaviors spawns a pull/push collider into the world for a specified duration, and then deletes itself once the duration has passed.

May 24th & 25th 2023

- Refactoring Inventory code
- Added namespace to procedural terrain generation code and editor code
- Added namespace to player movement state machine code
- Re-organization of project folder structure

May 26th - 30th 2023

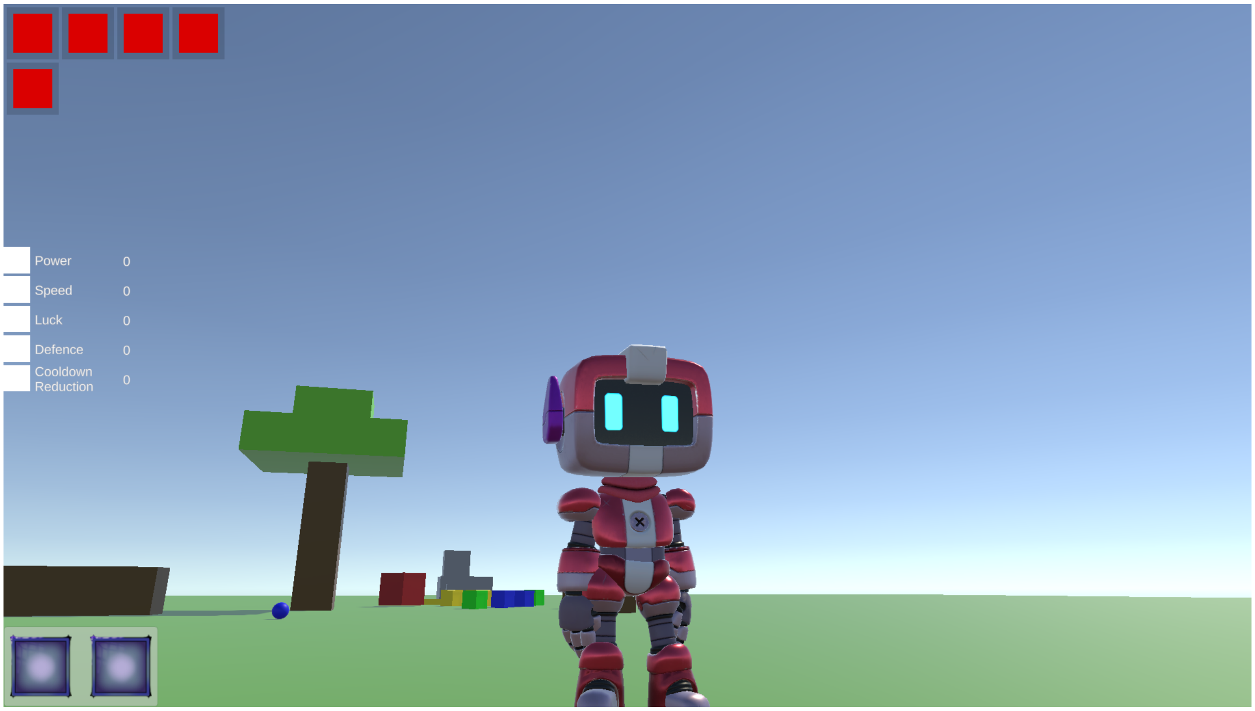
Nothing was done on these days

May 31th 2023

- Switched ability controller input detection from loop based to event based
- Created charge ability scriptable object skeleton

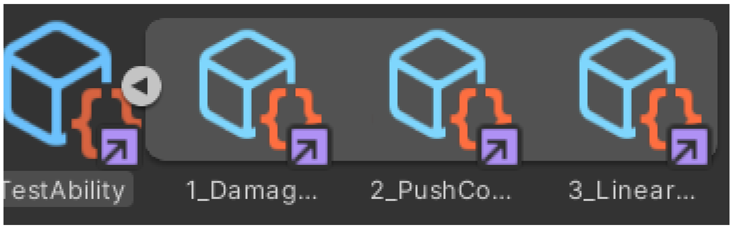
Monthly Summary:

days without progress: 13 days out of 31



Power 0
Speed 0
Luck 0
Defence 0
Cooldown Reduction 0

Ability System

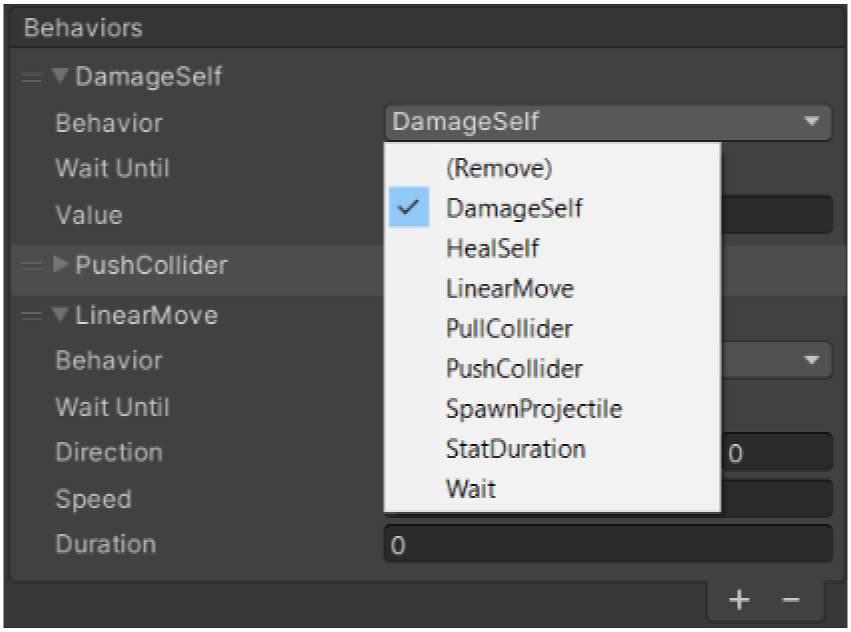


New ability system utilizes the software command pattern principle and C# aysnc functionality. Given a list of commands, in this case ability behaviors, the program will execute each command in sequence. The developer has the choice to make the program to wait until the ability behavior is done executing or execute the next following behavior sequentially, more information about ability behaviors in the ability behavior section. New abilities can be easily added and implemented through unity’s inspector.

Types of Abilities:

- Single Ability SO: Holds a list of behaviors. Upon activation through the ability controller script, the listed behaviors are executed in order. (More information in March 2023 MS)
- Combo Ability SO: Holds a list of abilities. On each activation by the ability controller, the next ability in the ability sequence is activated. (More information in March 2023 MS)
- Charge Ability SO: Holds a list of behaviors. Same as Single ability. However before activation, when player charges up ability by holding activation key and when released the ability activates. The charge ability object has the following properties:
    - Charge Rate (float): Determines the rate at which the charging occurs, in rate/frame. Clamped between 0 and 1
    - Activation Threshold (float): Determines the minimum charge level needed to activate the ability. Clamped between 0 and 1
    - Effectiveness Factor (Animation Curve): Determines the effectiveness of the ability
    - Auto Activate (bool): If true, automatically activates ability when charging reaches 100%. Otherwise, ability is only activated when player lets go of the button

Ability Editor



Behaviors

DamageSelf

Behavior: DamageSelf

Wait Until

Value

PushCollider

LinearMove

Behavior

Wait Until

Direction: 0

Speed

Duration: 0

- Uses reorderable list to serialize ability behaviors variables.
- Uses C# reflection to find and automatically update behaviors enum
- Automatically creates and deletes ability behavior
- Re-ordering functionality
- Middle of the index deletion and creation
- Automatically renames ability behavior index when reordering, deleting, or creating ability behaviors

Ability Behaviors

Damage Self

Given a damage amount (float), in heart units, the behavior will damage the caster for that amount.

Heal Self

Given a healing amount (float), in heart units, the behavior will heal the caster for that amount.

Linear Move

Given a direction relative to the player (vector3), speed (float), and duration (int), the behavior will move the player in the given direction and speed for a duration.

Pull Collider

Given a position relative to the player (vector3), size (vector3), pull force (float), follow player (bool), and duration (int). The behavior will spawn a collider, for the given duration, that will pull any rigid-body object to the center. If the follow player option is checked, the collider will follow the player around. Developers can visualize the collider in editor mode if draw gizmo option is checked.

Push Collider

Given a position relative to the player (vector3), size (vector3), push force (float), follow player (bool), and duration (int). The behavior will spawn a collider, for the given duration, that will push any rigid-body object to the center. If the follow player option is checked, the collider will follow the player around. Developers can visualize the collider in editor mode if draw gizmo option is checked.

Spawn Projectile

Given a Hand (custom enum), GameObject, damage (float), speed (float). The behavior will spawn a the given game-object from the specified hand facing the shooting direction

Stat Duration

Given a duration (int), Appiled Value (float), stat type (custom enum), Stat modification type (custom enum). The behavior will add a modifier to the given stat type and remove the same modifier after duration time has passed.

Wait

Given a duration (int). The behavior will stall ability for execution for given duration.

New Systems

IDamageable Interface

- Defines if the entity can take damage/heal. Can be used on any object such as: walls, barrels, enemies, etc.…

- Parameters of interface:
  - Max Health: the maximum health the entity is allowed, in hearts.
  - Current Health: the current health the entity has, in heats.
  - Damage(float): entity takes damage defined in parameter
  - Heal(float): entity heals amount defined in parameter

Input Manger

- Centralized event-relayer used by scripts.
- Detects key presses using unity’s new input system
- Triggers relevant events when keys are pressed

Character Data

- Contains the following modifiable stats (What is a Character?):
  - Health: Defines the maximum health a character can hold at a time
  - Speed: Defines how fast the character moves when sprinting
  - Luck: Defines how often character/NPC hit a critical hit. When character lands critical hit, base damage is multiplied by 2.25
  - Power: Increases damage dealt by character’s abilities.
  - Cooldown Reduction: Reduces cooldowns for all abilities.
  - Defence: Defines how much damage is negated when hit