UML Comparison

My original UML looked at the game on a larger scale of a full game. This in turn utilized 3 classes each tied to the core objects of the game, however when making the game I only have the one additional class of projectile this manages the firing and the vectors of the object whilst also inheriting the GameObject features thus enabling me to reduce the clutter in the GameObject files.

The blocks class would have been added if the blocks were used in a capacity of which multiple levels were created to store the locations along with rotation and states.

Target class wasn’t used due to only using the targets in a small compacity of collision checking and initializing.

The projectile class is larger than I expected but isn’t cluttered, its structured in such a way that makes it easy to follow and the functions have clear singular uses.

Pseudocode

Start projectile movement

On mouse click store x and y.

On mouse release store x and y.

Mouse on click x – mouse on release x = movement vector x

Mouse on click y – mouse on release y = movement vector y

Start fire

Variable for x

Variable for y

x iteratively equals movement vector x \* integer \* the game time divided by 1000

y iteratively equals movement vector y \* integer \* the game time divided by 1000

start if

gravity applied to movement y vector if launched

end if

end fire

update sprite x and y positions.

End projectile movement