1 Class: Monster

Base class for enemies

2 Methods

:
__init__ : Constructor, sets default values
pickHP: Returns healthPool array of enemy HP s
set_attack_mode: sets attack mode (melee or ranged)
set_range: sets range of attack for enemy
get_attack_mode: returns attack mode
get_shield : returns shield value (can be greater than one if enemy has multiple shield layers)
increment_shield: increments shield value
get_range: returns range of attack of enemy
create_orc: creates orc enemy
create_tree: creates tree enemy
set_twin: sets twin boss value (true or false)
create_boss: creates the twins
set_damage: sets damage value
get_damage: gets damage value
get_shield: gets shield value
set_owner: sets owner (useful for batch operations ie multiple deletes/deaths)
set_health: Creates the equation-type health bars
set_position: sets the spatial coordinates (zero is top left of the map)
move: moves sprite (could use an animation function)
set_image: sets sprite image
distance_to: returns Euclidian norm of the vector starting in center of self position and ending in other's center
move_towards: Naive pathfinding. Constructs vector from self to other, normalises it, converts it to an integer so that movement is restricted to a grid abstraction, checks for collision and for neighbouring enemies, and then changes dx/dy accordingly to follow the vector
knockback: Moves sprite in direction opposite to that of the current movement vector, to simulate knockback in battle.
Ai : nave ai function, if player is within range move towards him
death: handles damage to and death of monster (removes self.owner from monster_group)
death_twins: similar to the above function, but this one checks if the first twin is dead before dealing damage to the second one

3 Class: Block

Base class for player

__init__ : constructor, sets default values
equip_item: equips appropriate item, effect_type if the type of the item and effect_value is ammo/charges
increment_powercharge: increments power spell's charge
get_powercharge: returns remaining charges for power spell
get_pie: returns pie status (true if player has pie, false otherwise)
set_owner: sets owner object, useful for death/removal etc
set_life: sets HP value
set_position: sets position based on coordinates x, y (0 is top left of the map, MAP_SIZE is defined in main game loop)
move : moves sprite after checking for collisions. Based on input from event handler **get_spldmg**: returns current spell's damage
get_invisitibility: returns invisibility status. While active invisibility grants immunity to damage
get_ammo: returns remaining ammo/charges of current weapon
get_ammoDmg: returns damage of current weapon
set_ammo: sets ammo value of ammoType to value
set_ammoType: sets current weapon type
get_ammoType: returns current weapon type
set_Damages: sets damage for each weapon type according to it's ammo
set_ammoDmg: sets current ammo's damage value
increment_invisibility: increments self.invisibility value (for damage immunity)
increment_ammo: increments current remaining ammo values by value
death: handles damage, plays hit sound,sets a brief invisibility window after getting hit and removes player from game if he is dead
set_image: sets sprites
set_direction: flips image to account for left/right facing player (removes the need for right sprites, just flip the left ones)

4 Class: Codex

Default codex class
codexViewer: displays the appropriate page of the codex (an image)

5 Class: Combat

Handles combat
combat: handles enemies attacking player and particles/missiles stricking enemies
combat_player_attack: handles player melee attack, accounts for twin's special conditions

6 Class: Item

Default item class
__init__: constructor, sets default values
set_owner: useful for removing/handling objects in groups
set_position: sets spatial coordinates
set_image: sets sprite
distance_to: returns Euclidian norm of vector starting in self center and ending in other center
effects: sets current effect_type and effect_value (usually charges)
pick_up: automatically give item to player if he collides with it

7 Class: Particle

Handles particle effects
__init__: sets default values
display: displays particle
move: moves particle based on current angle and speed
function particle_create: creates particles and appends them to particle_group for management

8 File : questioner.py

Handles Riddle
get_key: gets user niput
display_box: helper function: displays the message in a box in the middle of the screen
ask: displays the riddle using display_box, handles input

9 File: shop.py

Handles actual shop/Riddle Master using questioner functions
ShopGenerator: generates and displays the Riddle Master event and checks the answer