The PongVerse

UML Class Diagram

ShrinkEnlarge

+ active_time : int
+ probability : int
+ name : str
+ description : str
+ icon : str
+ width : float
+ height : float
+ settings : GlobalSettings
+ owner : str
+ image : pygame.Surface
+ rect : pygame.Rect

+ ShrinkEnlarge(self, ball_owner: str, width: float, height: float, settings: GlobalSettings)
+ affect_playerA(self, player_A: Paddle)
+ affect_playerA(self, player_B: Paddle)
+ run_powerup(self, paddleA: Paddle, paddleB: Paddle)
+ revert_powerup(self, paddleA: Paddle, paddleB: Paddle)

Freeze

+ active_time : int
+ probability : int
+ name : str
+ description : str
+ icon : str
+ width : float
+ height : float
+ settings : GlobalSettings
+ owner : str
+ image : pygame.Surface
+ rect : pygame.Rect

+ Freeze(self, ball_owner: str, width: float, height: float, settings: GlobalSettings)
+ affect_playerA(self, player_A: Paddle)
+ affect_playerA(self, player_B: Paddle)
+ run_powerup(self, paddleA: Paddle, paddleB: Paddle)
+ revert_powerup(self, paddleA: Paddle, paddleB: Paddle)

MultilpeBalls

+ active_time : int
+ probability : int
+ name : str
+ description : str
+ icon : str
+ width : float
+ height : float
+ settings : GlobalSettings
+ owner : str
+ image : pygame.Surface
+ rect : pygame.Rect

+ MultipleBalls(self, ball_owner: str, width: float, height: float, settings: GlobalSettings)
+ affect_playerA(self, player_A: Paddle)
+ affect_playerA(self, player_B: Paddle)
+ run_powerup(self, paddleA: Paddle, paddleB: Paddle)
+ revert_powerup(self, paddleA: Paddle, paddleB: Paddle)

FasterPaddle

+ active_time : int
+ probability : int
+ name : str
+ description : str
+ icon : str
+ width : float
+ height : float
+ settings : GlobalSettings
+ owner : str
+ image : pygame.Surface
+ rect : pygame.Rect

+ FasterPaddle(self, ball_owner: str, width: float, height: float, settings: GlobalSettings)
+ affect_playerA(self, player_A: Paddle)
+ affect_playerA(self, player_B: Paddle)
+ run_powerup(self, paddleA: Paddle, paddleB: Paddle)
+ revert_powerup(self, paddleA: Paddle, paddleB: Paddle)

DoubleScore

+ active_time : int
+ probability : int
+ name : str
+ description : str
+ icon : str
+ width : float
+ height : float
+ settings : GlobalSettings
+ owner : str
+ image : pygame.Surface
+ rect : pygame.Rect

+ DoubleScore(self, ball_owner: str, width: float, height: float, settings: GlobalSettings)
+ affect_playerA(self, player_A: Paddle)
+ affect_playerA(self, player_B: Paddle)
+ run_powerup(self, paddleA: Paddle, paddleB: Paddle)
+ revert_powerup(self, paddleA: Paddle, paddleB: Paddle)

Shield

+ active_time : int
+ probability : int
+ name : str
+ description : str
+ icon : str
+ width : float
+ height : float
+ settings : GlobalSettings
+ owner : str
+ image : pygame.Surface
+ rect : pygame.Rect

+ Shield(self, ball_owner: str, width: float, height: float, settings: GlobalSettings)
+ affect_playerA(self, player_A: Paddle)
+ affect_playerA(self, player_B: Paddle)
+ run_powerup(self, paddleA: Paddle, paddleB: Paddle)
+ revert_powerup(self, paddleA: Paddle, paddleB: Paddle)

Extends

PowerUp

+ settings : GlobalSettings
+ visible_time : int
+ width : float
+ height : float
+ owner : str
+ image : pygame.Surface
+ rect : pygame.Rect

+ PowerUp(self, ball_owner: str, width: float, height: float, settings: GlobalSettings)
+ affect_playerA(self, player_A: Paddle)
+ affect_playerB(self, player_B: Paddle)
+ run_powerup(self, paddleA: Paddle, paddleB: Paddle)
+ revert_powerup(self, paddleA: Paddle, paddleB: Paddle)
+ draw(self, filename: str)

Use

GlobalSettings

+ MENU_TITLE: str
+ SETTINGS_TITLE: str
+ CREDITS_TITLE: str
+ GAME_TITLE: str
+ GAME_TITLE_VANILLA: str
+ INSTRUCTIONS_TITLE: str
+ INSTRUCTIONS_TITLE_VANILLA: str
+ BLACK: tuple
+ WHITE: tuple
+ BLUE: tuple
+ LIGHT_BLUE: tuple
+ GOLDEN: tuple
+ RED: tuple
+ GRAY: tuple
+ FONT_TYPE_DEFAULT: str
+ FONT_TYPE_MENU: str
+ WIN_SCORE: int
+ SCORE_ADDER_A: int
+ SCORE_ADDER_B: int
+ MIN_ADDITIONAL_BALLS: int
+ MAX_ADDITIONAL_BALLS: int
+ PADDLE_ROUND_CORNERS_A: int
+ PADDLE_ROUND_CORNERS_B: int
+ PADDLE_ROUND_CORNERS_SHIELD: int
+ POWERUP_VISIBLE_TIME: int
+ POWERUP_NAME_VISIBLE_TIME: int
+ resolution: tuple
+ width: int
+ height: int
+ font_size_default: int
+ font_size_powerup: int
+ font_size_small_powerup: int
+ font_size_menu: int
+ title_size: int
+ subtitle_size: int
+ body_size: int
+ small_body_size: int
+ music_on: bool
+ button_width: int
+ button_height: int
+ small_button_width: int
+ small_button_height: int
+ button_gap: int
+ novaims_icon_size: tuple
+ novaims_img_load: pygame.Transform
+ sound_on_icon_size: tuple
+ sound_on_img_load: pygame.Transform
+ creator_icon_size: tuple
+ creator_1_img_load: pygame.Transform
+ creator_2_img_load: pygame.Transform
+ creator_3_img_load: pygame.Transform
+ creator_4_img_load: pygame.Transform
+ player_icon_size: tuple
+ player_a_icon_pos: tuple
+ player_b_icon_pos: tuple
+ pos_score_a: tuple
+ pos_score_b: tuple
+ field_divider_initial_pos: tuple
+ field_divider_max_pos: tuple
+ winner_icon_size: tuple
+ background_img: pygame.Transform
+ powerup_width: float
+ powerup_height: float
+ right_x_alignment: float
+ left_x_alignment: float
+ player_keys_width: float
+ player_keys_height: float
+ player_a_up: pygame.Transform
+ player_a_down: pygame.Transform
+ player_b_up: pygame.Transform
+ player_b_down: pygame.Transform
+ powerup_icon_pos_list: dict
+ escape_key_width: float
+ escape_key_height: float
+ escape_key: tuple
+ ball_width: float
+ ball_height: float
+ initial_pos_x: float
+ initial_pos_y: float
+ paddle_width_a: float
+ paddle_height_a: float
+ paddle_width_b: float
+ paddle_height_b: float
+ initial_pos_x_a: float
+ initial_pos_y_a: float
+ initial_pos_x_b: float
+ initial_pos_y_b: float
+ paddle_speed_a: float
+ paddle_speed_b: float
+ default_paddle_speed: float
+ faster_paddle_speed: float
+ powerup_width: float
+ powerup_height: float
+ powerup_field_width: tuple
+ powerup_field_height: tuple

+ GlobalSettings(self, initial_resolution: tuple, initial_music_on: bool)

PongVerse

+ settings: GlobalSettings
+ vanilla: bool
+ powerup_visible: Powerup (or None)
+ powerup_active: Powerup (or None)
+ powerup_visible_time: pygame.Time
+ powerup_active_time: pygame.Time
+ all_sprites_list: pygame.sprite.Group
+ additional_balls: list[Ball]
+ ball_owner: str (or None)
+ powerup_owner: str (or None)
+ screen: pygame.Surface
+ default_font: pygame.Font
+ powerup_font: pygame.Font
+ small_powerup_font: pygame.Font
+ clock: pygame.Clock
+ scoreA: int
+ scoreB: int
+ triggered: bool
+ random_second: int

+ PongVerse(self, vanilla: bool, settings: GlobalSettings)
+ displayWinConditionInstructions(self, body_font: pygame.font, subtitle_font: pygame.Font, left_x_alignment: float)
+ displayPowerupsInstructions(self, body_font: pygame.font, small_body_font: pygame.Font, subtitle_font: pygame.font)
+ displayPlayerKeysInstructions(self, body_font: pygame.font, subtitle_font: pygame.Font, left_x_alignment: float)
+ instructions(self)
+ pressedKeys(self, keys: pygame.key, paddleA: Paddle, paddleB: Paddle)
+ setPowerupVisible(self, timer: pygame.time)
+ setPowerupInvisible(self)
+ setPowerupActive(self, paddleA: Paddle, paddleB: Paddle)
+ setPowerupInactive(self, paddleA: Paddle, paddleB: Paddle)
+ handleVisiblePowerup(self, ball: Ball, paddleA: Paddle, paddleB: Paddle)
+ handleActivePowerup(self, paddleA: Paddle, paddleB: Paddle)
+ handleMultipleBalls(self, paddleA: Paddle, paddleB: Paddle)
+ displayScores(self)
+ setStaticElements(self, background_img: pygame.image, playerA_icon: pygame.transform, playerB_icon: pygame.transform)
+ instanceNewBall(self)
+ powerupActivatedSound(self)
+ winnerSound(self)
+ play(self)
+ winnerScreen(self)

Interface

+ settings: GlobalSettings
+ screen: pygame.Display
+ clock: pygame.Time
+ default_font: pygame.Font
+ menu_font: pygame.Font
+ button_width_center: int
+ button_height_pos: int
+ width_center: int

+ Interface(self, settings: GlobalSettings)
+ setWindowTitle(title: str)
+ clickSound(self)
+ drawRect(self, color: tuple, width: float, height: float, default_size: bool)
+ displayReturnButton(self, mouse_pos: tuple, return_rect: pygame.rect, return_text_rect: pygame.Rect, return_text: pygame.surface.Surface)
+ setTitle(self, title: str)
+ mainProgramLoop(self, main_loop: int, return_rect: pygame.rect)
+ getResolutionDict(self)
+ mainMenu(self)
+ creatorsScreen(self)
+ settingsScreen(self)

Use

Ball

+ settings: GlobalSettings
+ filename: str
+ width: float
+ height: float
+ image: pygame.surface
+ rect: pygame.rect
+ x_velocity: randint
+ y_velocity: randint
+ velocity: list
+ ball_free: bool

+ Ball(self, filename: str, width: float, height: float, settings: GlobalSettings)
+ update(self, move: bool)
+ bounce(self)
+ bounceUpDown(self)
+ handleBallCollision(self, paddleA: Paddle, paddleB: Paddle)
+ resetBall(self,)
+ handleBallMotion(self, scoreA: int, scoreB: int, ball_owner: str, triggered: bool)

Paddle

+ height: float
+ border_radius: int
+ image: pygame.image
+ rect: pygame.rect
+ settings: GlobalSettings
+ color: tuple

+ Paddle(self, color, width, height, border_radius)
+ moveUp(self, pixels: int)
+ moveDown(self, pixels: int, window_height: int)

Use