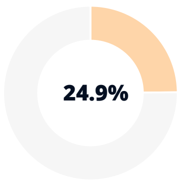
Plagiarism report

particle.py

Scan details

Scan time:	Total Pages:	Total Words:
April 13th, 2024 at 9:29 UTC	3	700

Plagiarism Detection



Types of plagiarism		Words
● Identical	**0%**	0
● Minor Changes	**0%**	0
● Paraphrased	**24.9%**	174
○ Omitted Words	**0%**	0

AI Content Detection

Text coverage

- ● AI text
- ○ Human text

Plagiarism Results: (1)

python - How do I rotate an image around its center using Pygame? - Stac... **24.9%**

https://stackoverflow.com/questions/4183208/how-do-i-rotate-an-image-around-its-center-using-pygame

Stack Overflow About ...

```python
class Particle:
    """
    Represents a particle in the game, which can be used for visual effects like explosions, sparks, etc.

    Each particle has a type, position, velocity, and is associated with a specific animation frame. The particl
    behavior (e.g., movement and animation) is updated each frame, and it can be rendered to the screen.

    Attributes:
        game: The main game object which this particle is part of. This allows access to game-wide resources and
        type (str): A string representing the type of the particle (e.g., 'leaf', 'sparks'). This can be used tc
                    particle behaviors or rendering styles.
        pos (list): The position of the particle in the game world as a list [x, y].
        velocity (list): The velocity of the particle as a list [vx, vy], representing movement along the x and
        animation (Animation): An Animation object representing the particle's current animation state. It contr
                               which image/frame is displayed for the particle.
        frame (int): The initial frame of the animation to start with. Allows particles of the same type to star
                     at different animation states for variety.
    """

    def __init__(self, game, p_type, pos, velocity=[0, 0], frame=0):
        """
        Initializes a new Particle instance with the given type, position, velocity, and animation frame.

        Parameters:
            game: The game object this particle is associated with.
            p_type (str): The type of the particle.
            pos (list): The starting position of the particle in the game world.
            velocity (list, optional): The initial velocity of the particle. Defaults to [0, 0].
            frame (int, optional): The initial frame of the particle's animation. Defaults to 0.
        """
        self.game = game
        self.type = p_type
        self.pos = list(pos)
        self.velocity = list(velocity)
        self.animation = self.game.assets['particle/' + p_type].copy()
        self.animation.frame = frame

    def update(self):
        """
        Updates the particle's state. This includes moving the particle according to its velocity and updating i
        animation. If the animation is completed, it might mark the particle for removal.

        Returns:
            bool: True if the particle is to be removed (e.g., because its animation has finished), False otherw
        """
        kill = False
        if self.animation.done:
            kill = True

        self.pos[0] += self.velocity[0]
        self.pos[1] += self.velocity[1]

        self.animation.update()

        return kill

    def render(self, surf, offset=(0, 0)):
        """
        Renders the particle on the given surface.

        The particle's current animation frame is drawn at the particle's position. This position is adjusted by
        the given offset, which allows for camera movement or scrolling backgrounds.

        Parameters:
            surf (pygame.Surface): The Pygame surface to draw the particle on.
            offset (tuple, optional): A tuple (x, y) representing the offset to apply to the particle's positior
                                      This is useful for rendering the particle relative to a moving camera.
                                      Defaults to (0, 0).
        """
        img = self.animation.img()
        surf.blit(img, (self.pos[0] - offset[0] - img.get_width() // 2, self.pos[1] - offset[1] - img.get_height
```