ParticlePy

grimmigerFuchs

Nov 20, 2021

CONTENTS:

1 ParticlePy Modules **1**
- 1.1 particlepy.particle . . . 1
- 1.2 particlepy.shape . . . 3
- 1.3 particlepy.math . . . 8

2 Indices and tables **9**

Python Module Index **11**

Index **13**

CHAPTER

ONE

PARTICLEPY MODULES

1.1 particlepy.particle

class Particle(*shape: particlepy.shape.Shape*, *position: Tuple[float, float]*, *velocity: Tuple[float, float]*, *delta_radius: float*, *data: Optional[dict] = None*, *alive: bool = True*)

Bases: `object`

This is the particle class. It simulates the physics of a particle and can be used in a particle system (*ParticleSystem*)

Parameters

- **shape** (*particlepy.shape.Shape*) – Visual particle shape
- **position** (*Tuple[float, float]*) – Center position
- **velocity** (*Tuple[float, float]*) – Velocity
- **delta_radius** (*float*) – Radius decrease value
- **data** (*dict, optional*) – A dictionary for extra data, defaults to *None*
- **alive** (*bool, optional*) – *True* if particle should be alive, and *False* if otherwise, defaults to *True*

Attributes

- **shape** (*particlepy.shape.Shape*) – Visual particle shape
- **position** (*List[float, float]*) – Center position
- **velocity** (*List[float, float]*) – Velocity (can be modified with gravity)
- **delta_radius** (*float*) – Radius decrease value
- **progress** (*float*) – A variable ranging from 0 to 1 to represent the lifespan
- **inverted_progress** (*float*) – A variable ranging from 1 to 0 to represent the lifespan
- **time** (*float*) – A simple timer
- **data** (*dict*) – A dictionary for extra data
- **alive** (*bool*) – *True* if particle is alive, and *False* if otherwise

kill()

Sets attribute `alive` *False*

render(*surface: pygame.Surface*)

Renders the particle on given surface

Parameters **surface** (pygame.Surface) – The surface on which the particle is being rendered on

revive()
Sets attribute alive *True*

update(*delta_time: float*, *gravity: Optional[Tuple[float, float]] = None*)
Updates position, velocity, progress, etc. of particle and kills it, if radius <= 0

Parameters

- **delta_time** (*float*) – A value to let the particle move according to frame time
- **gravity** (*Tuple[float, float], optional*) – Affects the velocity and 'pulls' it in a direction, defaults to None

class ParticleSystem(*data: Optional[dict] = None*, *alive: bool = True*)
Bases: object

The particle system class. It is used to manage particles in a group

Parameters

- **data** (*dict, optional*) – A dictionary for extra data, defaults to None
- **alive** (*bool, optional*) – *True* if particle system should be alive, and *False* if otherwise, defaults to *True*

Attributes

- **particles** (List[*Particle*])
- **data** (*dict*) – A dictionary for extra data
- **alive** (*bool*) – *True* if particle system is alive, and *False* if otherwise

clear()
Clears the particle list

emit(*particle:* particlepy.particle.Particle)
Creates a new particle

Parameters **particle** (*Particle*) – Particle which is being created

Raises **Exception** – Particle system is not alive, not able to add particles

kill()
Sets alive *False*

make_shape()
Makes the surface of all particles in system

render(*surface: pygame.Surface*)
Renders surface of all particles on given surface

Parameters **surface** (pygame.Surface) – Surface on which the particles are being rendered

revive()
Sets alive *True*

update(*delta_time: float*, *gravity: Optional[Tuple[float, float]] = None*)
Calls *Particle.update()* for every particle in system

Parameters

- **delta_time** (*float*) – A value to let the particles move according to frame time

- **gravity** (*Tuple[float, float], optional*) – Affects the velocity and 'pulls' particles in a direction, defaults to None

1.2 particlepy.shape

class BaseForm(*radius: float*, *color: Tuple[int, int, int]*, *alpha: int = 255*, *angle: float = 0*)
Bases: *particlepy.shape.Shape*, abc.ABC

The basic form class. Is used as shape argument in *particlepy.particle.Particle*. Is subclassed to create other shapes, e.g. *Circle* or *Rect*

Parameters

- **radius** (*float*) – Radius of shape
- **color** (*Tuple[int, int, int]*) – Color of shape
- **alpha** (*int, optional*) – Transparency of shape *(0 - 255 → RGBA)*, defaults to *255*
- **angle** (*float, optional*) – Degrees of rotation, defaults to *0*

Attributes

- **radius** (*float*) – Radius of shape
- **_orig_radius** (*float*) – Radius of shape when being instanced. Property is *BaseForm.orig_radius()*
- **color** (*List[int, int, int]*) – Color of shape
- **_orig_color** (*Tuple[int, int, int]*) – Color of shape when being instanced. Property is *BaseForm.orig_color()*
- **alpha** (*int*) – Transparency of shape, ranges from *0* to *255*
- **_orig_alpha** (*int*) – Transparency of shape when being instanced. Property is BaseForm.orig_alpha()
- **angle** (*int*) – Degrees of rotation of shape
- **_orig_angle** (*float*) – Angle of shape when being instanced. Property is BaseForm.orig_angle()
- **surface** (pygame.Surface) – Pygame surface of shape
- **rect** (pygame.Rect) – Pygame Rect of surface. Position does not affect anything

check_size_above_zero()
Checks if surface size is above *null*

Returns *True* if surface size above *null*, *False* if otherwise

Return type bool

decrease(*delta: float*)
Decreases radius of shape by delta_radius

Parameters **delta** (*float*) – Radius decrease value

get_progress() → Tuple[float, float]
Returns tuple of two floats: *progress* and *inverted_progress*

Returns *progress* and *inverted_progress*

Return type Tuple[float, float]

`make_shape()`
Creates shape for shape surface. Can be modified to make different shapes and effects.

`make_surface()` → pygame.Surface
Makes the surface by also calling *Shape.make_shape()*

Returns Surface of shape

Return type `pygame.Surface`

`property orig_color`
Returns `_orig_color`

Returns `_orig_color`

Return type Tuple[int]

`property orig_radius`
Returns `_orig_radius`

Returns `_orig_radius`

Return type float

`class Circle`(*radius: float*, *color: Tuple[int, int, int]*, *alpha: int = 255*, *angle: float = 0*)
Bases: *particlepy.shape.BaseForm*, `abc.ABC`

Circle shape class. Is subclass of *BaseForm* and inherits all attributes and methods

Parameters

- **radius** (*float*) – Radius of shape
- **color** (*Tuple[int, int, int]*) – Color of shape
- **alpha** (*int, optional*) – Transparency of shape *(0 - 255 → RGBA)*, defaults to *255*
- **angle** (*float, optional*) – Degrees of rotation, defaults to *0*

Attributes

- **radius** (*float*) – Radius of shape
- **_orig_radius** (*float*) – Radius of shape when being instanced. Property is `Circle.orig_radius()`
- **color** (*List[int, int, int]*) – Color of shape
- **_orig_color** (*Tuple[int, int, int]*) – Color of shape when being instanced. Property is `Circle.orig_color()`
- **alpha** (*int*) – Transparency of shape, ranges from *0* to *255*
- **_orig_alpha** (*int*) – Transparency of shape when being instanced. Property is `Circle.orig_alpha()`
- **angle** (*int*) – Degrees of rotation of shape
- **_orig_angle** (*float*) – Angle of shape when being instanced. Property is `Circle.orig_angle()`
- **surface** (`pygame.Surface`) – Pygame surface of shape
- **rect** (`pygame.Rect`) – Pygame Rect of `surface`. Position does not affect anything

`make_shape()`
Makes a circle

`class Image(surface: pygame.Surface, size: Tuple[int, int], alpha: int = 255, angle: float = 0)`
Bases: `particlepy.shape.Shape`, `abc.ABC`

Image shape class. Is subclass of *Shape* and inherits all attributes and methods and adds to it

Parameters

- **surface** (`pygame.Surface`) – Surface of shape
- **size** (*Tuple[int, int]*) – Scaled size of surface
- **alpha** (*int, optional*) – Transparency of shape *(0 - 255 → RGBA)*, defaults to *255*
- **angle** (*float, optional*) – Degrees of rotation, defaults to *0*

Attributes

- **alpha** (*int*) – Transparency of shape, ranges from *0* to *255*
- **_orig_alpha** (*int*) – Transparency of shape when being instanced. Property is `Image.orig_alpha()`
- **angle** (*int*) – Degrees of rotation of shape
- **_orig_angle** (*float*) – Angle of shape when being instanced. Property is `Image.orig_angle()`
- **size** (*List[int, int]*) – Scaled size of surface
- **_orig_size** (*Tuple[int, int]*) – Scaled size of surface shape when being instanced. Property is *Image.orig_size()*
- **surface** (`pygame.Surface`) – Pygame surface of shape
- **_orig_surface** (`pygame.Surface`) – Surface of shape when being instanced. Property is *Image.orig_surface()*
- **rect** (`pygame.Rect`) – Pygame Rect of `surface`. Position does not affect anything

`check_size_above_zero()` → bool
Checks if surface size is above *null*

Returns *True* if surface size above *null*, *False* if otherwise

Return type bool

`decrease(delta: float)`
Decreases size by `attr`

`get_progress()` → Tuple[float, float]
Returns `progress` and `inverted_progress` of shape

Returns `progress`, `inverted_progress`

Return type Tuple[float, float]

`make_shape()`
Is being called by *Image.make_surface()* and used to make the visual representation of the shape

`make_surface()` → pygame.Surface
Makes the surface by also calling *Image.make_shape()*

Returns Surface of shape

Return type `pygame.Surface`

property orig_size
Returns _orig_size

Returns _orig_size

Return type Tuple[int, int]

property orig_surface
Returns _orig_surface

Returns _orig_surface

Return type Tuple[int, int]

class Rect(*radius: float*, *color: Tuple[int, int, int]*, *alpha: int = 255*, *angle: float = 0*)
Bases: particlepy.shape.BaseForm, abc.ABC

Rectangle shape class. Is subclass of BaseForm and inherits all attributes and methods

Parameters

- **radius** (*float*) – Radius of shape
- **color** (*Tuple[int, int, int]*) – Color of shape
- **alpha** (*int, optional*) – Transparency of shape *(0 - 255 → RGBA)*, defaults to *255*
- **angle** (*float, optional*) – Degrees of rotation, defaults to *0*

Attributes

- **radius** (*float*) – Radius of shape
- **_orig_radius** (*float*) – Radius of shape when being instanced. Property is Rect.orig_radius()
- **color** (*List[int, int, int]*) – Color of shape
- **_orig_color** (*Tuple[int, int, int]*) – Color of shape when being instanced. Property is Rect.orig_color()
- **alpha** (*int*) – Transparency of shape, ranges from *0* to *255*
- **_orig_alpha** (*int*) – Transparency of shape when being instanced. Property is Rect.orig_alpha()
- **angle** (*int*) – Degrees of rotation of shape
- **_orig_angle** (*float*) – Angle of shape when being instanced. Property is Rect.orig_angle()
- **surface** (pygame.Surface) – Pygame surface of shape
- **rect** (pygame.Rect) – Pygame Rect of surface. Position does not affect anything

make_shape()
Makes a rectangle

class Shape(*alpha: int = 255*, *angle: float = 0*)
Bases: object

This is the shape class. It is only used to subclass and use as a base for shapes.

Parameters

- **alpha** (*int, optional*) – Transparency of shape *(0 - 255 → RGBA)*, defaults to *255*
- **angle** (*float, optional*) – Degrees of rotation, defaults to *0*

Attributes

- **alpha** (*int*) – Transparency of shape, ranges from *0* to *255*
- **_orig_alpha** (*int*) – Transparency of shape when being instanced. Property is *Shape.orig_alpha()*
- **angle** (*float*) – Degrees of rotation
- **_orig_angle** (*float*) – Angle of shape when being instanced. Property is *Shape.orig_angle()*

check_size_above_zero() → bool
Checks if surface size is above *null*

Returns *True* if surface size above *null*, *False* if otherwise

Return type bool

decrease(*delta: float*)
Decreases size by `attr`

get_progress() → Tuple[float, float]
Returns `progress` and `inverted_progress` of shape

Returns `progress`, `inverted_progress`

Return type Tuple[float, float]

make_shape()
Is being called by *Shape.make_surface()* and used to make the visual representation of the shape

make_surface() → pygame.Surface
Makes the surface by also calling *Shape.make_shape()*

Returns Surface of shape

Return type `pygame.Surface`

property orig_alpha
Returns original alpha

Returns `_orig_alpha`

Return type int

property orig_angle
Returns original angle

Returns `_orig_angle`

Return type float

rotate(*surface: pygame.Surface*, *angle: float*)
Rotates shape by angle

Notes

Only exists because of pygame issue 2464.

1.3 particlepy.math

fade_alpha(*particle:* particlepy.particle.Particle, *alpha: int*, *progress: float*) → float
Fades `alpha` (transparency) of argument `particle` over life span (`progress`) to new color (`color`)

Parameters

- **particle** (*particlepy.particle.Particle*) – Particle to alpha color with
- **alpha** (*int*) – Transparency to fade to
- **progress** (*float*) – Life span identifier: `particlepy.particle.Particle.progress` or `particlepy.particle.Particle.inverted_progress`

Returns New alpha of particle

Return type float

fade_color(*particle:* particlepy.particle.Particle, *color: Tuple[int, int, int]*, *progress: float*) → list
Fades color of `particle` over life span (`progress`) to new color (`color`)

Parameters

- **particle** (*particlepy.particle.Particle*) – Particle to fade color with
- **color** (*Tuple[int, int, int]*) – Color to fade to
- **progress** (*float*) – Life span identifier: `particlepy.particle.Particle.progress` or `particlepy.particle.Particle.inverted_progress`

Returns New color of particle

Return type List[float]

Raises **AssertionError** – If `particle.shape` not *particlepy.shape.BaseForm*

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

p

particlepy.math, 8
particlepy.particle, 1
particlepy.shape, 3

INDEX

B

BaseForm (*class in particlepy.shape*), 3

C

check_size_above_zero() (*BaseForm method*), 3
check_size_above_zero() (*Image method*), 5
check_size_above_zero() (*Shape method*), 7
Circle (*class in particlepy.shape*), 4
clear() (*ParticleSystem method*), 2

D

decrease() (*BaseForm method*), 3
decrease() (*Image method*), 5
decrease() (*Shape method*), 7

E

emit() (*ParticleSystem method*), 2

F

fade_alpha() (*in module particlepy.math*), 8
fade_color() (*in module particlepy.math*), 8

G

get_progress() (*BaseForm method*), 3
get_progress() (*Image method*), 5
get_progress() (*Shape method*), 7

I

Image (*class in particlepy.shape*), 5

K

kill() (*Particle method*), 1
kill() (*ParticleSystem method*), 2

M

make_shape() (*BaseForm method*), 4
make_shape() (*Circle method*), 4
make_shape() (*Image method*), 5
make_shape() (*ParticleSystem method*), 2
make_shape() (*Rect method*), 6
make_shape() (*Shape method*), 7
make_surface() (*BaseForm method*), 4
make_surface() (*Image method*), 5
make_surface() (*Shape method*), 7
module
    particlepy.math, 8
    particlepy.particle, 1
    particlepy.shape, 3

O

orig_alpha (*Shape property*), 7
orig_angle (*Shape property*), 7
orig_color (*BaseForm property*), 4
orig_radius (*BaseForm property*), 4
orig_size (*Image property*), 5
orig_surface (*Image property*), 6

P

Particle (*class in particlepy.particle*), 1
particlepy.math
    module, 8
particlepy.particle
    module, 1
particlepy.shape
    module, 3
ParticleSystem (*class in particlepy.particle*), 2

R

Rect (*class in particlepy.shape*), 6
render() (*Particle method*), 1
render() (*ParticleSystem method*), 2
revive() (*Particle method*), 2
revive() (*ParticleSystem method*), 2
rotate() (*in module particlepy.shape*), 7

S

Shape (*class in particlepy.shape*), 6

U

update() (*Particle method*), 2
update() (*ParticleSystem method*), 2