PyPunk2 Documentation

Release 0.1

Saxon Landers (ackwell)

June 19, 2013

CONTENTS

Contents:

CHAPTER

ONE

API REFERENCE

Contents:

1.1 Core

Child classes:

1.1.1 Engine

class pypunk.core.**Engine**(*width*, *height*[, *frame_rate=60*[, *title="PyPunk"*]])
Main game class. Manages game loop.

Parameters

- **width** – The width of your game.
- **height** – The height of your game.
- **frame_rate** – The game framerate, in frames per second.
- **title** – The window caption for your game.

paused = True
If the game should stop updating/rendering.

start()
Starts the PyPunk loop. Should be called after setting PP.world so as to start running the game.

close([*event = None*])
Signals the Engine to stop it's loop, and exit the game.

1.1.2 World

class pypunk.core.**World**
Updated by Engine, main game container that holds all currently active Entities.

Useful for organization, eg. "Menu", "Level1", etc.

Todo

Usage example, etc.

Note: Make sure to call `super().__init__()` if overriding to ensure World instance is set up correctly.

visible = True
If False, the World will not be rendered.

self.camera = Point()
Point used to modify location entities are drawn at.

begin()
Override this; called when the World is changed, and is set to the currently active world.

end()
Override this; called when the World is changed, and the active world is no longer this.

update()
render()
Executed by the game loop, updates/renders all contained Entities.

Note: If you override these to give your World update/render code, remeber to call `super().update()` or your Entities will not be updated/rendered.

mouse_x
mouse_y
Read only. X/Y position of the mouse in the World.

add(*e*)
remove(*e*)
Adds/removes Entity `e` to/from the World at the end of the frame. Returns the added/removed Entity.

add_list(*entity[, entity [...]]*)
remove_list(*entity[, entity [...]]*)
Adds/removes passed Entities to/from the World at the end of the frame. If the first argument is a `list`, it will be used instead.

remove_all()
Removes all Entities from the World at the end of the frame.

Todo

Mask handling.

bring_to_front(*e*)
send_to_back(*e*)
Brings/sends the Entity `e` to the front/back of it's containing layer. Returns `True` if successful.

bring_forward(*e*)
send_backward(*e*)
Brings/sends the Entity `e` once position towards the front/back of it's containing layer. Returns `True` if successful.

is_at_front(*e*)
is_at_back(*e*)
Returns whether the Entity `e` is at the front/back of it's layer.

Todo

Collide Functions.

---

count
Read only. Number of Entities that are in the World.

type_count (*t*)
Returns the number of Entities of the type `t` that are in the World.

class_count (*c*)
Returns the number of Entities of the class `c` that are in the World.

layer_count (*l*)
Returns the number of Entities on the layer `l`.

first
Read only. The first Entity in the World update order.

layers
Read only. Number of Entity layers the World has.

type_first (*t*)
The first Entity of type `t`.

class_first (*c*)
The first Entity of class `c`.

layer_first (*l*)
layer_last (*l*)
The first/last Entity on layer `l`.

farthest
nearest
Read only. The Entity that will be rendered first/last by the World.

layer_farthest
layer_nearest
Read only. The Entity that will be rendered first/last by the World.

unique_types
Read only. The number of different types that have been added to the World.

get_type (*t*, *into*)
Adds all Entities of type `t` to provided list `into`.

get_class (*c*, *into*)
Adds all Entities of class `c` to provided list `into`.

get_layer (*l*, *into*)
Adds all Entities on layer `l` to provided list `into`.

get_all (*into*)
Adds all Entities in World to provided list `into`.

1.1.3 Entity

class pypunk.core.**Entity**
Main game Entity class, updated by World.

Parameters

- **x** – X position to place the Entity.

- **y** – Y position to place the Entity.
- **graphic** – Graphic to assign to the Entity.
- **mask** – Mask to assign to the Entity.

Note: Make sure to call `super().__init__()` if overriding to ensure World instance is set up correctly.

---

visible = True
If the Entity should render.

collideable = True
If the Entity should respond to collision checks.

x
y
X/Y position of the Entity in the World

width
height
Width/Height of the Entity's hitbox

origin_x
origin_y
X/Y origin of the Entity's hitbox

render_target = None
The Screen object to draw the entity onto. Leave as None to render to the primary window.

added()
removed()
Override these, called when the Entity is added/removed to/from a world.

update()
Override this, called every frame by the current World as part of the main game loop.

render()
Renders the Entity's Graphic. If you override this to implement additional behaviour, remember to call `super().render()` to ensure the Entity is drawn.

collide(*t*, *x*, *y*)
Checks for a collision between the Entity, positioned at `(x, y)`, and an Entity of type `t`. Returns the first Entity collided with, or `None` if there was no collision.

collide_types(*types*, *x*, *y*)
Same as `collide()`, but checks against a list of Entity types `types`.

collide_with(*e*, *x*, *y*)
Same as `collide()`, but checks against a single Entity instance, `e`.

collide_rect(*x*, *y*, *r_x*, *r_y*, *r_width*, *r_height*)
Returns whether the Entity, positioned at `(x, y)`, overlaps the specified rectangle at `(r_x, r_y)` with dimensions `r_width x r_height`.

collide_point(*x*, *y*, *p_x*, *p_y*)
Returns whether this Entity, positioned at `(x, y)`, overlaps the specified position `(p_x, p_y)`

Todo

•collide_into

- collide_types_into
- on_camera

---

world
Read only. The World object this Entity has been added to.

center_x
center_y
The center x/y position of the Entity's hitbox area.

left
right
top
bottom
The left/right/top/bottom-most position of the Entity's hitbox.

layer
The rendering layer of this entity. Higher layers are rendered first.

type
The collision type, used for collision checks.

Todo

Mask support.

---

graphic
Graphic object to render to the screen during the render loop.

add_graphic(*g*)
Adds the Graphic `g` to the Entity via a Graphicslist

set_hitbox(*width*, *height*, *origin_x*, *origin_y*)
Sets the Entity's hitbox properties.

set_hitbox_to(*o*)
Sets the Entity's hitbox to math that of the provided object `o`.

set_origin(*x=0*, *y=0*)
Sets the origin of the Entity to `(x, y)`.

center_origin()
Sets the Entity's origin to `(width/2, height/2)`.

Todo

- distance_from
- distance_to_point
- distance_to_rect
- move_by
- move_to
- move_towards
- move_collide_x
- move_collide_y

clamp_horizontal(*left*, *right*, *padding=0*)
clamp_vertical(*top*, *bottom*, *padding=0*)
Clamps the Entity's hitbox on the x/y axis, between `(left, right)`/`(top, bottom)`, with optional additional `padding`.

CHAPTER

TWO

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

PYTHON MODULE INDEX

p