

Hamilton College CS110 Graphics Library

1.0

Generated by Doxygen 1.6.1

Fri Jul 7 11:02:21 2017

Contents

1	Namespace Index	1
1.1	Package List	1
2	Class Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	Namespace Documentation	7
4.1	Package cs110graphics	7
4.1.1	Detailed Description	8
4.1.2	Function Documentation	8
4.1.2.1	RunWithYieldDelay	8
4.1.2.2	StartGraphicsSystem	8
5	Class Documentation	11
5.1	cs110graphics._RunWithYieldDelay Class Reference	11
5.1.1	Detailed Description	11
5.2	cs110graphics.Circle Class Reference	12
5.2.1	Detailed Description	12
5.2.2	Member Function Documentation	12
5.2.2.1	set_radius	12
5.3	cs110graphics.Event Class Reference	13
5.3.1	Detailed Description	13
5.3.2	Member Function Documentation	13
5.3.2.1	get_button	13
5.3.2.2	get_description	13
5.3.2.3	get_key	13
5.3.2.4	get_mouse_location	14

5.3.2.5	get_root_mouse_location	14
5.4	cs110graphics.EventHandler Class Reference	15
5.4.1	Detailed Description	15
5.4.2	Member Function Documentation	15
5.4.2.1	handle_key_release	15
5.4.2.2	handle_key_press	15
5.4.2.3	handle_mouse_enter	16
5.4.2.4	handle_mouse_leave	16
5.4.2.5	handle_mouse_move	16
5.4.2.6	handle_mouse_press	16
5.4.2.7	handle_mouse_release	16
5.5	cs110graphics.Fillable Class Reference	17
5.5.1	Detailed Description	17
5.5.2	Member Function Documentation	18
5.5.2.1	get_border_color	18
5.5.2.2	get_border_width	18
5.5.2.3	get_fill_color	18
5.5.2.4	get_pivot	18
5.5.2.5	rotate	18
5.5.2.6	scale	18
5.5.2.7	set_border_color	18
5.5.2.8	set_border_width	18
5.5.2.9	set_fill_color	18
5.5.2.10	set_pivot	19
5.6	cs110graphics.GraphicalObject Class Reference	20
5.6.1	Detailed Description	20
5.6.2	Member Function Documentation	20
5.6.2.1	add_handler	20
5.6.2.2	get_center	21
5.6.2.3	get_depth	21
5.6.2.4	move	21
5.6.2.5	move_to	21
5.6.2.6	set_depth	21
5.7	cs110graphics.Image Class Reference	22
5.7.1	Detailed Description	22
5.7.2	Member Function Documentation	23

5.7.2.1	move	23
5.7.2.2	move_to	23
5.7.2.3	resize	23
5.7.2.4	rotate	23
5.7.2.5	scale	23
5.7.2.6	size	23
5.8	cs110graphics.Oval Class Reference	24
5.8.1	Detailed Description	24
5.8.2	Member Function Documentation	24
5.8.2.1	set_radii	24
5.9	cs110graphics.Polygon Class Reference	25
5.9.1	Detailed Description	25
5.10	cs110graphics.Rectangle Class Reference	26
5.10.1	Detailed Description	26
5.10.2	Member Function Documentation	26
5.10.2.1	set_side_lengths	26
5.11	cs110graphics.Square Class Reference	27
5.11.1	Detailed Description	27
5.11.2	Member Function Documentation	27
5.11.2.1	set_side_length	27
5.12	cs110graphics.Text Class Reference	28
5.12.1	Detailed Description	28
5.12.2	Member Function Documentation	28
5.12.2.1	move	28
5.12.2.2	move_to	29
5.12.2.3	set_size	29
5.12.2.4	set_text	29
5.13	cs110graphics.Timer Class Reference	30
5.13.1	Detailed Description	30
5.13.2	Member Function Documentation	30
5.13.2.1	set_function	30
5.13.2.2	set_interval	30
5.13.2.3	start	30
5.13.2.4	stop	31
5.14	cs110graphics.Window Class Reference	32
5.14.1	Detailed Description	32

5.14.2	Member Function Documentation	32
5.14.2.1	add	32
5.14.2.2	remove	33
5.14.2.3	set_background	33
5.14.2.4	set_height	33
5.14.2.5	set_title	33
5.14.2.6	set_width	33

Chapter 1

Namespace Index

1.1 Package List

Here are the packages with brief descriptions (if available):

[cs110graphics](#) (Contains a CPython-friendly version of a Tkinter based graphics library) [7](#)

Chapter 2

Class Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

cs110graphics._RunWithYieldDelay	11
cs110graphics.Event	13
cs110graphics.EventHandler	15
cs110graphics.GraphicalObject	20
cs110graphics.Fillable	17
cs110graphics.Circle	12
cs110graphics.Oval	24
cs110graphics.Polygon	25
cs110graphics.Rectangle	26
cs110graphics.Square	27
cs110graphics.Image	22
cs110graphics.Text	28
cs110graphics.Timer	30
cs110graphics.Window	32

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

cs110graphics._RunWithYieldDelay (A class which uses a function which returns a generator to rerun until the generator stops generating numbers)	11
cs110graphics.Circle (A circle, which can be added to a Window object)	12
cs110graphics.Event (An event which gets bound to an object)	13
cs110graphics.EventHandler (Handles an event)	15
cs110graphics.Fillable (This window is a parent class of any object which can have its colors modified)	17
cs110graphics.GraphicalObject (This window is a parent class of any object which can be put into Window)	20
cs110graphics.Image (An image, which can be added to a Window object)	22
cs110graphics.Oval (An oval, which can be added to a Window object)	24
cs110graphics.Polygon (A Polygon , which can be added to a Window object)	25
cs110graphics.Rectangle (A rectangle, which can be added to a Window object)	26
cs110graphics.Square (A square, which can be added to a Window object)	27
cs110graphics.Text (Text which can be added to a Window object)	28
cs110graphics.Timer (A class which continually runs a function after a delay)	30
cs110graphics.Window (This window acts as a canvas which other objects can be put onto) . . .	32

Chapter 4

Namespace Documentation

4.1 Package cs110graphics

Contains a CSPy-friendly version of a Tkinter based graphics library.

Classes

- class [Window](#)
This window acts as a canvas which other objects can be put onto.
- class [Event](#)
An event which gets bound to an object.
- class [EventHandler](#)
Handles an event.
- class [GraphicalObject](#)
This window is a parent class of any object which can be put into [Window](#).
- class [Fillable](#)
This window is a parent class of any object which can have its colors modified.
- class [Image](#)
An image, which can be added to a [Window](#) object.
- class [Text](#)
[Text](#) which can be added to a [Window](#) object.
- class [Polygon](#)
A [Polygon](#), which can be added to a [Window](#) object.
- class [Circle](#)
A circle, which can be added to a [Window](#) object.
- class [Oval](#)

An oval, which can be added to a [Window](#) object.

- class [Square](#)

A square, which can be added to a [Window](#) object.

- class [Rectangle](#)

A rectangle, which can be added to a [Window](#) object.

- class [Timer](#)

A class which continually runs a function after a delay.

- class [_RunWithYieldDelay](#)

A class which uses a function which returns a generator to rerun until the generator stops generating numbers.

Functions

- def [StartGraphicsSystem](#)

This initializes the graphics system.

- def [RunWithYieldDelay](#)

A wrapper for the [_RunWithYieldDelay](#) class.

4.1.1 Detailed Description

Contains a CSPy-friendly version of a Tkinter based graphics library. Paul Magnus '18, Ines Ayara '20, Matthew R. Jenkins '20

Summer 2017

4.1.2 Function Documentation

4.1.2.1 def cs110graphics.RunWithYieldDelay (window, func)

A wrapper for the [_RunWithYieldDelay](#) class. THIS SHOULD BE USED INSTEAD OF CREATING AN [_RunWithYieldDelay](#) INSTANCE.

Required Parameters:

- window - [Window](#)
- func - function which returns a generator of int

4.1.2.2 def cs110graphics.StartGraphicsSystem (first_function, width = 400, height = 400, background = "white", name = "Graphics Window")

This initializes the graphics system. Required Parameters:

- first_function - func

Optional Parameters:

- width - int
- height - int
- background - string
- name - string

Chapter 5

Class Documentation

5.1 cs110graphics._RunWithYieldDelay Class Reference

A class which uses a function which returns a generator to rerun until the generator stops generating numbers.

Public Member Functions

- `def __init__`

5.1.1 Detailed Description

A class which uses a function which returns a generator to rerun until the generator stops generating numbers. NOTE: DO NOT INITIALIZE THIS CLASS ANYWHERE IN YOUR PROGRAM. THE WRAPPER FUNCTION `RunWithYieldDelay` SHOULD BE USED INSTEAD.

Required Parameters:

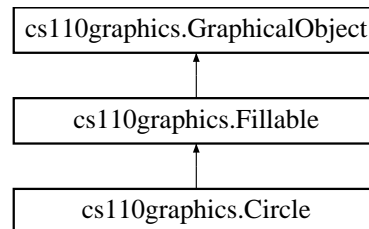
- `window` - [Window](#) - the window which the object with yield delay is on.
- `func` - function which returns a generator of int - a function with a few necessary parameters which allow it to run with yield delay. A function needs to return a generator of int, needs a yield statement with an int which represents the delay (in milliseconds), and it needs a `raise StopIteration` statement at the end of the function.

The documentation for this class was generated from the following file:

- `cs110graphics.py`

5.2 cs110graphics.Circle Class Reference

A circle, which can be added to a [Window](#) object. Inheritance diagram for cs110graphics.Circle::



Public Member Functions

- def `__init__`
- def `set_radius`
Sets the radius of the [Circle](#).

5.2.1 Detailed Description

A circle, which can be added to a [Window](#) object. Required Parameters:

- window - [Window](#) - the window which the object will be added to.

Optional Parameters:

- radius - int - sets the radius of the [Circle](#). (default: 40)
- center - tuple - sets the center of the [Circle](#). (default: (200, 200))

5.2.2 Member Function Documentation

5.2.2.1 def cs110graphics.Circle.set_radius (self, radius)

Sets the radius of the [Circle](#). Required Parameters:

- radius - int

The documentation for this class was generated from the following file:

- cs110graphics.py

5.3 cs110graphics.Event Class Reference

An event which gets bound to an object.

Public Member Functions

- def `__init__`
- def `get_button`
Returns the mouse button that is attached to the event.
- def `get_description`
Returns the description of the event.
- def `get_key`
Returns the keyboard key that is attached to the event.
- def `get_mouse_location`
Returns a tuple of the x and y coordinates of the mouse location in the canvas.
- def `get_root_mouse_location`
Returns a tuple of the x and y coordinates of the mouse location in the window.

5.3.1 Detailed Description

An event which gets bound to an object. Used by [EventHandler](#) objects.

Required Parameters:

- event - TkEvent - The event which the user want applied an an object.

5.3.2 Member Function Documentation

5.3.2.1 def cs110graphics.Event.get_button (self)

Returns the mouse button that is attached to the event. Returns None if the button fails to exist (like if the [Event](#) handles a key press).

5.3.2.2 def cs110graphics.Event.get_description (self)

Returns the description of the event.

5.3.2.3 def cs110graphics.Event.get_key (self)

Returns the keyboard key that is attached to the event. Returns None if the key fails to exist (like if the [Event](#) handles a mouse press).

5.3.2.4 `def cs110graphics.Event.get_mouse_location (self)`

Returns a tuple of the x and y coordinates of the mouse location in the canvas.

5.3.2.5 `def cs110graphics.Event.get_root_mouse_location (self)`

Returns a tuple of the x and y coordinates of the mouse location in the window.

The documentation for this class was generated from the following file:

- `cs110graphics.py`

5.4 cs110graphics.EventHandler Class Reference

Handles an event.

Public Member Functions

- def `__init__`
- def `handle_key_press`
Handles a key press.
- def `handke_key_release`
Handles a key release.
- def `handle_mouse_enter`
Handles when a mouse enters an object.
- def `handle_mouse_leave`
Handles when a mouse leaves an object.
- def `handle_mouse_move`
Handles a mouse move.
- def `handle_mouse_press`
Handles a mouse press.
- def `handle_mouse_release`
Handles a mouse release.

5.4.1 Detailed Description

Handles an event. These are overloaded by the user, so by default they're empty except for the pass command.

5.4.2 Member Function Documentation

5.4.2.1 `def cs110graphics.EventHandler.handke_key_release (self, event)`

Handles a key release. Optional Parameters:

- event - [Event](#) - when included, you can use any [Event](#) method whenever this function is run.

5.4.2.2 `def cs110graphics.EventHandler.handle_key_press (self, event)`

Handles a key press. Optional Parameters:

- event - [Event](#) - when included, you can use any [Event](#) method whenever this function is run.

5.4.2.3 def cs110graphics.EventHandler.handle_mouse_enter (*self*, *event*)

Handles when a mouse enters an object. Optional Parameters:

- event - [Event](#) - when included, you can use any [Event](#) method whenever this function is run.

5.4.2.4 def cs110graphics.EventHandler.handle_mouse_leave (*self*, *event*)

Handles when a mouse leaves an object. Optional Parameters:

- event - [Event](#) - when included, you can use any [Event](#) method whenever this function is run.

5.4.2.5 def cs110graphics.EventHandler.handle_mouse_move (*self*, *event*)

Handles a mouse move. Optional Parameters:

- event - [Event](#) - when included, you can use any [Event](#) method whenever this function is run.

5.4.2.6 def cs110graphics.EventHandler.handle_mouse_press (*self*, *event*)

Handles a mouse press. Optional Parameters:

- event - [Event](#) - when included, you can use any [Event](#) method whenever this function is run.

5.4.2.7 def cs110graphics.EventHandler.handle_mouse_release (*self*, *event*)

Handles a mouse release. Optional Parameters:

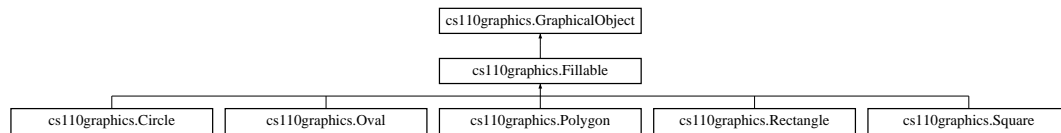
- event - [Event](#) - when included, you can use any [Event](#) method whenever this function is run.

The documentation for this class was generated from the following file:

- cs110graphics.py

5.5 cs110graphics.Fillable Class Reference

This window is a parent class of any object which can have its colors modified. Inheritance diagram for cs110graphics.Fillable::



Public Member Functions

- def `__init__`
- def `get_border_color`
Returns the border color of a *Fillable*.
- def `get_border_width`
Returns the border width of a *Fillable*.
- def `get_fill_color`
Returns the depth of a *Fillable*.
- def `get_pivot`
Returns the pivot point of a *Fillable*.
- def `rotate`
Rotates the object.
- def `scale`
Scales the *Fillable* up or down depending on the factor.
- def `set_border_color`
Sets the border color of the *Fillable*.
- def `set_border_width`
Sets the border width of the *Fillable*.
- def `set_fill_color`
Sets the fill color of the *Fillable*.
- def `set_pivot`
Sets the pivot point of the *Fillable*.

5.5.1 Detailed Description

This window is a parent class of any object which can have its colors modified. No constructor exists in this class, but its methods are used by other objects that extend/inherit this class.

5.5.2 Member Function Documentation

5.5.2.1 `def cs110graphics.Fillable.get_border_color (self)`

Returns the border color of a [Fillable](#).

5.5.2.2 `def cs110graphics.Fillable.get_border_width (self)`

Returns the border width of a [Fillable](#).

5.5.2.3 `def cs110graphics.Fillable.get_fill_color (self)`

Returns the depth of a [Fillable](#).

5.5.2.4 `def cs110graphics.Fillable.get_pivot (self)`

Returns the pivot point of a [Fillable](#).

5.5.2.5 `def cs110graphics.Fillable.rotate (self, degrees)`

Rotates the object. Required Parameters:

- degrees - int

5.5.2.6 `def cs110graphics.Fillable.scale (self, factor)`

Scales the [Fillable](#) up or down depending on the factor. Required Parameters:

- factor - float

5.5.2.7 `def cs110graphics.Fillable.set_border_color (self, color)`

Sets the border color of the [Fillable](#). Required Parameters:

- color - string

5.5.2.8 `def cs110graphics.Fillable.set_border_width (self, width)`

Sets the border width of the [Fillable](#). Required Parameters:

- width - int

5.5.2.9 `def cs110graphics.Fillable.set_fill_color (self, color)`

Sets the fill color of the [Fillable](#). Required Parameters:

- color - string

5.5.2.10 `def cs110graphics.Fillable.set_pivot (self, pivot)`

Sets the pivot point of the [Fillable](#). Required Parameters:

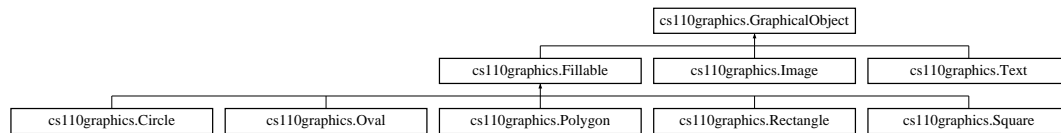
- *pivot* - tuple of (int * int)

The documentation for this class was generated from the following file:

- cs110graphics.py

5.6 cs110graphics.GraphicalObject Class Reference

This window is a parent class of any object which can be put into [Window](#). Inheritance diagram for `cs110graphics.GraphicalObject`:



Public Member Functions

- `def __init__`
- `def add_handler`
Adds a handler to the graphical object.
- `def get_center`
Returns the center of the graphical object.
- `def get_depth`
Returns the depth of the graphical object.
- `def move`
Moves a graphical object dx pixels horizontally and dy pixels vertically.
- `def move_to`
Moves a graphical object to a point.
- `def set_depth`
Sets the depth of the [GraphicalObject](#).

5.6.1 Detailed Description

This window is a parent class of any object which can be put into [Window](#). No constructor exists in this class, but its methods are used by other objects that extend/inherit this class.

5.6.2 Member Function Documentation

5.6.2.1 `def cs110graphics.GraphicalObject.add_handler (self, handler_object)`

Adds a handler to the graphical object. Required Parameters:

- `handler_object` - an object with a [GraphicalObject](#) representation within it (such as an object which has a [Circle](#) object in it)

5.6.2.2 def cs110graphics.GraphicalObject.get_center (self)

Returns the center of the graphical object.

5.6.2.3 def cs110graphics.GraphicalObject.get_depth (self)

Returns the depth of the graphical object.

5.6.2.4 def cs110graphics.GraphicalObject.move (self, dx, dy)

Moves a graphical object dx pixels horizontally and dy pixels vertically. Required Parameters:

- dx - int
- dy - int

Reimplemented in [cs110graphics.Image](#), and [cs110graphics.Text](#).

5.6.2.5 def cs110graphics.GraphicalObject.move_to (self, point)

Moves a graphical object to a point. Required Parameters:

- point - tuple of (int * int)

Reimplemented in [cs110graphics.Image](#), and [cs110graphics.Text](#).

5.6.2.6 def cs110graphics.GraphicalObject.set_depth (self, depth)

Sets the depth of the [GraphicalObject](#). Required Parameters:

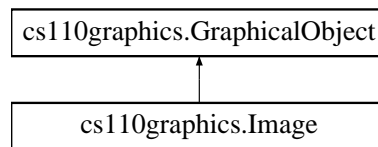
- depth - int

The documentation for this class was generated from the following file:

- cs110graphics.py

5.7 cs110graphics.Image Class Reference

An image, which can be added to a [Window](#) object. Inheritance diagram for cs110graphics.Image::



Public Member Functions

- def `__init__`
- def `move`
Moves a graphical object dx pixels horizontally and dy pixels vertically.
- def `move_to`
Moves a graphical object to a point.
- def `resize`
Resizes the [Image](#).
- def `rotate`
Rotates an object by degrees.
- def `scale`
Scales the image according to the factor.
- def `size`
Returns a tuple of the width and height of the image.

5.7.1 Detailed Description

An image, which can be added to a [Window](#) object. Required Parameters:

- window - [Window](#) - the window which the object will be added to.
- image_loc - str - The name of an image within the current working directory . (If the current working directory is /foo/bar, then the image the user wants to use has to be in that directory. There is no support for using internet links at this time.)

Optional Parameters:

- center - tuple of int * int - sets the center of the [Image](#). (default: (200, 200))
- width - int - sets the width of the image. (default: 25)
- height - int - sets the height of the image. (default: 25)

5.7.2 Member Function Documentation

5.7.2.1 `def cs110graphics.Image.move (self, dx, dy)`

Moves a graphical object dx pixels horizontally and dy pixels vertically. Required Parameters:

- dx - int
- dy - int

Reimplemented from [cs110graphics.GraphicalObject](#).

5.7.2.2 `def cs110graphics.Image.move_to (self, point)`

Moves a graphical object to a point. Required Parameters:

- point - tuple of (int * int)

Reimplemented from [cs110graphics.GraphicalObject](#).

5.7.2.3 `def cs110graphics.Image.resize (self, width, height)`

Resizes the [Image](#). Required Parameters:

- width - int
- height - int

5.7.2.4 `def cs110graphics.Image.rotate (self, degrees)`

Rotates an object by degrees. Required Parameters:

- degrees - int

5.7.2.5 `def cs110graphics.Image.scale (self, factor)`

Scales the image according to the factor. Required Parameters:

- factor - float

5.7.2.6 `def cs110graphics.Image.size (self)`

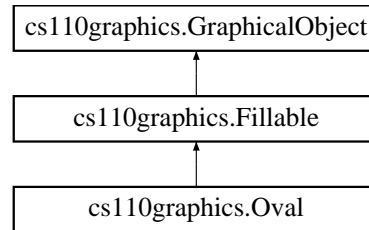
Returns a tuple of the width and height of the image.

The documentation for this class was generated from the following file:

- cs110graphics.py

5.8 cs110graphics.Oval Class Reference

An oval, which can be added to a [Window](#) object. Inheritance diagram for cs110graphics.Oval::



Public Member Functions

- def `__init__`
- def `set_radii`

Sets the horizontal and vertical radii of the [Oval](#).

5.8.1 Detailed Description

An oval, which can be added to a [Window](#) object. Required Parameters:

- window - [Window](#) - the window which the object will be added to.

Optional Parameters:

- radiusX - int - sets the radius of the [Oval](#). (default: 40)
- radiusY - int - sets the radius of the [Oval](#). (default: 60)
- center - tuple - sets the center of the [Oval](#). (default: (200, 200))

5.8.2 Member Function Documentation

5.8.2.1 def cs110graphics.Oval.set_radii (self, radiusX, radiusY)

Sets the horizontal and vertical radii of the [Oval](#). Required Parameters:

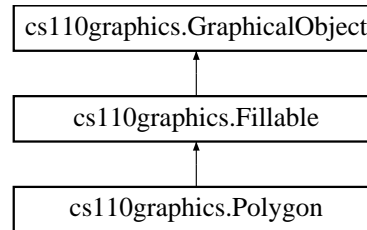
- radiusX - int
- radiusY - int

The documentation for this class was generated from the following file:

- cs110graphics.py

5.9 cs110graphics.Polygon Class Reference

A [Polygon](#), which can be added to a [Window](#) object. Inheritance diagram for cs110graphics.Polygon::



Public Member Functions

- `def __init__`

5.9.1 Detailed Description

A [Polygon](#), which can be added to a [Window](#) object. Required Parameters:

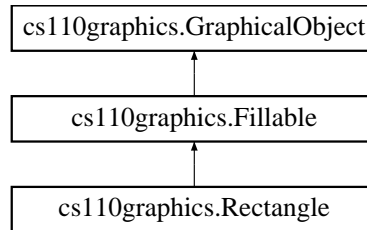
- window - [Window](#) - the window which the object will be added to.
- points - list of tuples of `int * int` - each tuple corresponds to an xy point.

The documentation for this class was generated from the following file:

- cs110graphics.py

5.10 cs110graphics.Rectangle Class Reference

A rectangle, which can be added to a [Window](#) object. Inheritance diagram for cs110graphics.Rectangle::



Public Member Functions

- def `__init__`
- def `set_side_lengths`
Sets the width and height of the [Rectangle](#).

5.10.1 Detailed Description

A rectangle, which can be added to a [Window](#) object. Required Parameters:

- window - [Window](#) - the window which the object will be added to.

Optional Parameters:

- width - int - sets the width of the [Square](#). (default: 40)
- height - int - sets the height of the [Square](#). (default: 40)
- center - tuple - sets the center of the [Square](#). (default: (200, 200))

5.10.2 Member Function Documentation

5.10.2.1 def cs110graphics.Rectangle.set_side_lengths (self, width, height)

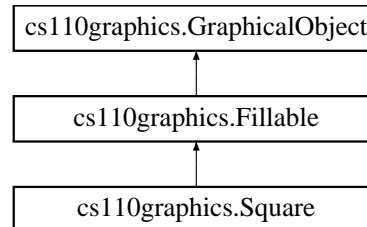
Sets the width and height of the [Rectangle](#).

The documentation for this class was generated from the following file:

- cs110graphics.py

5.11 cs110graphics.Square Class Reference

A square, which can be added to a [Window](#) object. Inheritance diagram for cs110graphics.Square::



Public Member Functions

- def `__init__`
- def `set_side_length`
Sets the side length of the [Square](#).

5.11.1 Detailed Description

A square, which can be added to a [Window](#) object. Required Parameters:

- window - [Window](#) - the window which the object will be added to.

Optional Parameters:

- side_length - int - sets the side length of the [Square](#). (default: 40)
- center - tuple - sets the center of the [Square](#). (default: (200, 200))

5.11.2 Member Function Documentation

5.11.2.1 def cs110graphics.Square.set_side_length (self, side_length)

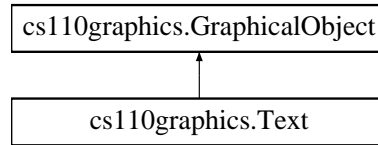
Sets the side length of the [Square](#).

The documentation for this class was generated from the following file:

- cs110graphics.py

5.12 cs110graphics.Text Class Reference

[Text](#) which can be added to a [Window](#) object. Inheritance diagram for `cs110graphics.Text`:



Public Member Functions

- `def __init__`
- `def move`
Moves a graphical object dx pixels horizontally and dy pixels vertically.
- `def move_to`
Moves a graphical object to a point.
- `def set_size`
Sets the point size of the text.
- `def set_text`
Sets the text.

5.12.1 Detailed Description

[Text](#) which can be added to a [Window](#) object. Required Parameters:

- `window` - [Window](#) - the window which the object will be added to.
- `text` - str - The text which is displayed.

Optional Parameters:

- `center` - tuple of int * int - sets the center of the [Image](#). (default: (200, 200))
- `width` - int - sets the size of the text. (default: 12)

5.12.2 Member Function Documentation

5.12.2.1 `def cs110graphics.Text.move (self, dx, dy)`

Moves a graphical object dx pixels horizontally and dy pixels vertically. Required Parameters:

- `dx` - int
- `dy` - int

Reimplemented from [cs110graphics.GraphicalObject](#).

5.12.2.2 `def cs110graphics.Text.move_to (self, point)`

Moves a graphical object to a point. Required Parameters:

- *point* - tuple of (int * int)

Reimplemented from [cs110graphics.GraphicalObject](#).

5.12.2.3 `def cs110graphics.Text.set_size (self, size)`

Sets the point size of the text. Required Parameters:

- *size* - int

5.12.2.4 `def cs110graphics.Text.set_text (self, text)`

Sets the text. Required Parameters:

- *text* - string

The documentation for this class was generated from the following file:

- cs110graphics.py

5.13 cs110graphics.Timer Class Reference

A class which continually runs a function after a delay.

Public Member Functions

- def `__init__`
- def `set_function`
Sets the function which is going to be run.
- def `set_interval`
Sets the interval between executions of the function.
- def `start`
Starts the timer.
- def `stop`
Stops the timer.

5.13.1 Detailed Description

A class which continually runs a function after a delay. Required Parameters:

- window - `Window` - the window which the timer will use to start and stop the animation.
- interval - int - the time (in milliseconds) that the function will refresh.
- func - function - the function which will be run.

5.13.2 Member Function Documentation

5.13.2.1 def cs110graphics.Timer.set_function (self, func)

Sets the function which is going to be run. Required Parameters:

- func - function

5.13.2.2 def cs110graphics.Timer.set_interval (self, interval)

Sets the interval between executions of the function. Required Parameters:

- interval - int

5.13.2.3 def cs110graphics.Timer.start (self)

Starts the timer.

5.13.2.4 def cs110graphics.Timer.stop (*self*)

Stops the timer.

The documentation for this class was generated from the following file:

- cs110graphics.py

5.14 cs110graphics.Window Class Reference

This window acts as a canvas which other objects can be put onto.

Public Member Functions

- def `__init__`
- def `add`
Adds an object of type [GraphicalObject](#) to the [Window](#) object.
- def `remove`
Removes an object of type [GraphicalObject](#) to the [Window](#) object, assuming the object being deleted exists.
- def `set_background`
Sets the background color of the canvas.
- def `set_height`
Sets the height of the canvas.
- def `set_title`
Sets the title of the window holding the canvas.
- def `set_width`
Sets the width of the canvas.

5.14.1 Detailed Description

This window acts as a canvas which other objects can be put onto. Required Parameters:

- width - int - Width of canvas.
- height - int - Height of canvas.
- background - str - Background color of canvas. Can be either the name of a color ("yellow"), or a hex code ("#FFFF00").
- name - str - The title of the window.
- first_function - proc(Window) - When the window is created, it runs this function after everything is run. (default: None)
- master - unknown type - necessary for the creation of the Tkinter widgets. (default: None)

5.14.2 Member Function Documentation

5.14.2.1 def cs110graphics.Window.add (self, graphic)

Adds an object of type [GraphicalObject](#) to the [Window](#) object. Required Parameters:

- graphic - [GraphicalObject](#)

5.14.2.2 def cs110graphics.Window.remove (*self*, *graphic*)

Removes an object of type [GraphicalObject](#) to the [Window](#) object, assuming the object being deleted exists.
Required Parameters:

- *graphic* - [GraphicalObject](#)

5.14.2.3 def cs110graphics.Window.set_background (*self*, *background*)

Sets the background color of the canvas. Required Parameters:

- *background* - string

5.14.2.4 def cs110graphics.Window.set_height (*self*, *height*)

Sets the height of the canvas. Required Parameters:

- *height* - int

5.14.2.5 def cs110graphics.Window.set_title (*self*, *name*)

Sets the title of the window holding the canvas. Required Parameters:

- *name* - string

5.14.2.6 def cs110graphics.Window.set_width (*self*, *width*)

Sets the width of the canvas. Required Parameters:

- *width* - height

The documentation for this class was generated from the following file:

- cs110graphics.py

Index

- add
 - cs110graphics::Window, 32
- add_handler
 - cs110graphics::GraphicalObject, 20
- cs110graphics, 7
 - RunWithYieldDelay, 8
 - StartGraphicsSystem, 8
- cs110graphics::_RunWithYieldDelay, 11
- cs110graphics::Circle, 12
 - set_radius, 12
- cs110graphics::Event, 13
 - get_button, 13
 - get_description, 13
 - get_key, 13
 - get_mouse_location, 13
 - get_root_mouse_location, 14
- cs110graphics::EventHandler, 15
 - handle_key_release, 15
 - handle_key_press, 15
 - handle_mouse_enter, 15
 - handle_mouse_leave, 16
 - handle_mouse_move, 16
 - handle_mouse_press, 16
 - handle_mouse_release, 16
- cs110graphics::Fillable, 17
 - get_border_color, 18
 - get_border_width, 18
 - get_fill_color, 18
 - get_pivot, 18
 - rotate, 18
 - scale, 18
 - set_border_color, 18
 - set_border_width, 18
 - set_fill_color, 18
 - set_pivot, 18
- cs110graphics::GraphicalObject, 20
 - add_handler, 20
 - get_center, 20
 - get_depth, 21
 - move, 21
 - move_to, 21
 - set_depth, 21
- cs110graphics::Image, 22
 - move, 23
 - move_to, 23
 - resize, 23
 - rotate, 23
 - scale, 23
 - size, 23
- cs110graphics::Oval, 24
 - set_radii, 24
- cs110graphics::Polygon, 25
- cs110graphics::Rectangle, 26
 - set_side_lengths, 26
- cs110graphics::Square, 27
 - set_side_length, 27
- cs110graphics::Text, 28
 - move, 28
 - move_to, 28
 - set_size, 29
 - set_text, 29
- cs110graphics::Timer, 30
 - set_function, 30
 - set_interval, 30
 - start, 30
 - stop, 30
- cs110graphics::Window, 32
 - add, 32
 - remove, 32
 - set_background, 33
 - set_height, 33
 - set_title, 33
 - set_width, 33
- get_border_color
 - cs110graphics::Fillable, 18
- get_border_width
 - cs110graphics::Fillable, 18
- get_button
 - cs110graphics::Event, 13
- get_center
 - cs110graphics::GraphicalObject, 20
- get_depth
 - cs110graphics::GraphicalObject, 21
- get_description
 - cs110graphics::Event, 13
- get_fill_color
 - cs110graphics::Fillable, 18
- get_key

- cs110graphics::Event, 13
- get_mouse_location
 - cs110graphics::Event, 13
- get_pivot
 - cs110graphics::Fillable, 18
- get_root_mouse_location
 - cs110graphics::Event, 14
- handle_key_release
 - cs110graphics::EventHandler, 15
- handle_key_press
 - cs110graphics::EventHandler, 15
- handle_mouse_enter
 - cs110graphics::EventHandler, 15
- handle_mouse_leave
 - cs110graphics::EventHandler, 16
- handle_mouse_move
 - cs110graphics::EventHandler, 16
- handle_mouse_press
 - cs110graphics::EventHandler, 16
- handle_mouse_release
 - cs110graphics::EventHandler, 16
- move
 - cs110graphics::GraphicalObject, 21
 - cs110graphics::Image, 23
 - cs110graphics::Text, 28
- move_to
 - cs110graphics::GraphicalObject, 21
 - cs110graphics::Image, 23
 - cs110graphics::Text, 28
- remove
 - cs110graphics::Window, 32
- resize
 - cs110graphics::Image, 23
- rotate
 - cs110graphics::Fillable, 18
 - cs110graphics::Image, 23
- RunWithYieldDelay
 - cs110graphics, 8
- scale
 - cs110graphics::Fillable, 18
 - cs110graphics::Image, 23
- set_background
 - cs110graphics::Window, 33
- set_border_color
 - cs110graphics::Fillable, 18
- set_border_width
 - cs110graphics::Fillable, 18
- set_depth
 - cs110graphics::GraphicalObject, 21
- set_fill_color
 - cs110graphics::Fillable, 18
- set_function
 - cs110graphics::Timer, 30
- set_height
 - cs110graphics::Window, 33
- set_interval
 - cs110graphics::Timer, 30
- set_pivot
 - cs110graphics::Fillable, 18
- set_radii
 - cs110graphics::Oval, 24
- set_radius
 - cs110graphics::Circle, 12
- set_side_length
 - cs110graphics::Square, 27
- set_side_lengths
 - cs110graphics::Rectangle, 26
- set_size
 - cs110graphics::Text, 29
- set_text
 - cs110graphics::Text, 29
- set_title
 - cs110graphics::Window, 33
- set_width
 - cs110graphics::Window, 33
- size
 - cs110graphics::Image, 23
- start
 - cs110graphics::Timer, 30
- StartGraphicsSystem
 - cs110graphics, 8
- stop
 - cs110graphics::Timer, 30