

## Abstract Art Generator

Generated by Doxygen 1.8.17



<b>1 Hierarchical Index</b>	<b>1</b>
1.1 Class Hierarchy	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 File Index</b>	<b>5</b>
3.1 File List	5
<b>4 Class Documentation</b>	<b>7</b>
4.1 canvas.canvas Class Reference	7
4.1.1 Detailed Description	7
4.1.2 Constructor & Destructor Documentation	7
4.1.2.1 __init__()	8
4.1.3 Member Function Documentation	8
4.1.3.1 draw_to_canvas()	8
4.1.3.2 generate_bg()	8
4.1.3.3 get_canvas()	9
4.2 circle_generator.circle_generator Class Reference	9
4.2.1 Detailed Description	9
4.2.2 Member Function Documentation	10
4.2.2.1 draw()	10
4.3 color_palette.color_palette Class Reference	10
4.3.1 Detailed Description	11
4.3.2 Constructor & Destructor Documentation	11
4.3.2.1 __init__()	11
4.3.3 Member Function Documentation	12
4.3.3.1 draw_ui_dynamic()	12
4.3.3.2 draw_ui_static()	12
4.3.3.3 events()	12
4.3.3.4 get_background_color()	13
4.3.3.5 get_colors_from_palette()	13
4.3.3.6 get_foreground_colors()	13
4.3.3.7 get_name_of_palette()	13
4.3.3.8 refresh_ui_static()	14
4.4 curves_generator.curves_generator Class Reference	14
4.4.1 Detailed Description	14
4.4.2 Member Function Documentation	14
4.4.2.1 draw()	14
4.5 dots_generator.dots_generator Class Reference	15
4.5.1 Detailed Description	15
4.5.2 Member Function Documentation	15
4.5.2.1 draw()	16

4.6 fpolygons_generator.fpolygons_generator Class Reference	16
4.6.1 Detailed Description	17
4.6.2 Member Function Documentation	17
4.6.2.1 draw()	17
4.7 generator.generator Class Reference	17
4.7.1 Detailed Description	18
4.7.2 Member Function Documentation	18
4.7.2.1 draw()	18
4.8 generator_storage.generator_storage Class Reference	18
4.8.1 Detailed Description	19
4.9 help.help Class Reference	19
4.9.1 Detailed Description	20
4.9.2 Constructor & Destructor Documentation	20
4.9.2.1 __init__()	20
4.9.3 Member Function Documentation	20
4.9.3.1 draw_ui_dynamic()	20
4.9.3.2 draw_ui_static()	20
4.9.3.3 events()	21
4.10 hpolygons_generator.hpolygons_generator Class Reference	21
4.10.1 Detailed Description	21
4.10.2 Member Function Documentation	22
4.10.2.1 draw()	22
4.11 layer.layer Class Reference	22
4.11.1 Detailed Description	23
4.11.2 Constructor & Destructor Documentation	23
4.11.2.1 __init__()	23
4.11.3 Member Function Documentation	24
4.11.3.1 draw_ui_dynamic()	24
4.11.3.2 draw_ui_static()	24
4.11.3.3 events()	24
4.11.3.4 get_layer_complexity()	25
4.11.3.5 get_layer_shape()	25
4.11.3.6 get_layer_size()	25
4.11.3.7 get_layer_style()	25
4.12 line_generator.line_generator Class Reference	26
4.12.1 Detailed Description	26
4.12.2 Member Function Documentation	26
4.12.2.1 draw()	26
4.13 overlay.overlay Class Reference	27
4.13.1 Detailed Description	27
4.13.2 Constructor & Destructor Documentation	28
4.13.2.1 __init__()	28

4.13.3 Member Function Documentation	28
4.13.3.1 draw_ui_dynamic()	28
4.13.3.2 draw_ui_static()	28
4.13.3.3 events()	28
4.14 ring_generator.ring_generator Class Reference	29
4.14.1 Detailed Description	29
4.14.2 Member Function Documentation	29
4.14.2.1 draw()	29
4.15 square_generator.square_generator Class Reference	30
4.15.1 Detailed Description	30
4.15.2 Member Function Documentation	30
4.15.2.1 draw()	31
4.16 ui_controller.ui_controller Class Reference	31
4.16.1 Detailed Description	32
4.16.2 Constructor & Destructor Documentation	32
4.16.2.1 __init__()	32
4.16.3 Member Function Documentation	33
4.16.3.1 draw_ui_dynamic()	33
4.16.3.2 draw_ui_static()	33
4.16.3.3 process_events()	33
4.16.3.4 run()	33
4.16.4 Member Data Documentation	33
4.16.4.1 resolutions_list	34
4.17 widget.widget Class Reference	34
4.17.1 Detailed Description	35
4.17.2 Constructor & Destructor Documentation	35
4.17.2.1 __init__()	35
4.17.3 Member Function Documentation	35
4.17.3.1 draw_ui_dynamic()	35
4.17.3.2 draw_ui_static()	35
4.17.3.3 events()	36
4.18 widget_storage.widget_storage Class Reference	36
4.18.1 Detailed Description	36
<b>5 File Documentation</b>	<b>37</b>
5.1 assets.py File Reference	37
5.1.1 Detailed Description	38
5.1.2 Author(s)	38
5.1.3 Function Documentation	38
5.1.3.1 text_to_screen()	38
5.2 canvas.py File Reference	38
5.2.1 Detailed Description	39

5.2.2 Author(s) . . . . .	39
5.3 circle_generator.py File Reference . . . . .	39
5.3.1 Detailed Description . . . . .	39
5.3.2 Author(s) . . . . .	39
5.3.3 Variable Documentation . . . . .	39
5.3.3.1 art_styles_list . . . . .	40
5.4 color_palette.py File Reference . . . . .	40
5.4.1 Detailed Description . . . . .	40
5.4.2 Author(s) . . . . .	40
5.5 curves_generator.py File Reference . . . . .	40
5.5.1 Detailed Description . . . . .	41
5.5.2 Author(s) . . . . .	41
5.5.3 Variable Documentation . . . . .	41
5.5.3.1 art_styles_list . . . . .	41
5.6 dots_generator.py File Reference . . . . .	41
5.6.1 Detailed Description . . . . .	41
5.6.2 Author(s) . . . . .	41
5.6.3 Variable Documentation . . . . .	42
5.6.3.1 art_styles_list . . . . .	42
5.7 fpolygons_generator.py File Reference . . . . .	42
5.7.1 Detailed Description . . . . .	42
5.7.2 Author(s) . . . . .	42
5.7.3 Variable Documentation . . . . .	42
5.7.3.1 art_styles_list . . . . .	43
5.8 generator.py File Reference . . . . .	43
5.8.1 Detailed Description . . . . .	43
5.8.2 Author(s) . . . . .	43
5.9 generator_storage.py File Reference . . . . .	43
5.9.1 Detailed Description . . . . .	44
5.9.2 Author(s) . . . . .	44
5.9.3 Variable Documentation . . . . .	44
5.9.3.1 generator_storage . . . . .	44
5.10 help.py File Reference . . . . .	44
5.10.1 Detailed Description . . . . .	44
5.10.2 Author(s) . . . . .	44
5.11 hpolygons_generator.py File Reference . . . . .	44
5.11.1 Detailed Description . . . . .	45
5.11.2 Author(s) . . . . .	45
5.11.3 Variable Documentation . . . . .	45
5.11.3.1 art_styles_list . . . . .	45
5.12 layer.py File Reference . . . . .	45
5.12.1 Detailed Description . . . . .	46

5.12.2 Author(s)	46
5.13 line_generator.py File Reference	46
5.13.1 Detailed Description	46
5.13.2 Author(s)	46
5.13.3 Variable Documentation	46
5.13.3.1 art_styles_list	46
5.14 overlay.py File Reference	47
5.14.1 Detailed Description	47
5.14.2 Author(s)	47
5.15 square_generator.py File Reference	47
5.15.1 Detailed Description	47
5.15.2 Author(s)	47
5.15.3 Variable Documentation	47
5.15.3.1 art_styles_list	48
5.16 ui_controller.py File Reference	48
5.16.1 Detailed Description	48
5.16.2 Author(s)	48
5.17 widget.py File Reference	48
5.17.1 Detailed Description	49
5.17.2 Author(s)	49
5.18 widget_storage.py File Reference	49
5.18.1 Detailed Description	49
5.18.2 Author(s)	49
5.18.3 Variable Documentation	49
5.18.3.1 widgets	49
<b>Index</b>	<b>51</b>





# Chapter 1

## Hierarchical Index

### 1.1 Class Hierarchy

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

canvas.canvas . . . . .	7
generator_storage.generator_storage . . . . .	18
ui_controller.ui_controller . . . . .	31
widget_storage.widget_storage . . . . .	36
ABC	
generator.generator . . . . .	17
widget.widget . . . . .	34
generator	
circle_generator.circle_generator . . . . .	9
curves_generator.curves_generator . . . . .	14
dots_generator.dots_generator . . . . .	15
fpolygons_generator.fpolygons_generator . . . . .	16
hpolygons_generator.hpolygons_generator . . . . .	21
line_generator.line_generator . . . . .	26
ring_generator.ring_generator . . . . .	29
square_generator.square_generator . . . . .	30
widget	
color_palette.color_palette . . . . .	10
help.help . . . . .	19
layer.layer . . . . .	22
overlay.overlay . . . . .	27



## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">canvas.canvas</a>	
The canvas class . . . . .	7
<a href="#">circle_generator.circle_generator</a>	
Class to draw circles . . . . .	9
<a href="#">color_palette.color_palette</a>	
The color palette widget class . . . . .	10
<a href="#">curves_generator.curves_generator</a>	
Class to draw curves . . . . .	14
<a href="#">dots_generator.dots_generator</a>	
Class to draw dots . . . . .	15
<a href="#">fpolygons_generator.fpolygons_generator</a>	
Class to draw filled polygons . . . . .	16
<a href="#">generator.generator</a>	
Abstract class for generators to extend . . . . .	17
<a href="#">generator_storage.generator_storage</a>	
Storage for program generators . . . . .	18
<a href="#">help.help</a>	
The help widget class . . . . .	19
<a href="#">hpolygons_generator.hpolygons_generator</a>	
Class to draw hollow polygons . . . . .	21
<a href="#">layer.layer</a>	
The layer widget class . . . . .	22
<a href="#">line_generator.line_generator</a>	
Class to draw lines . . . . .	26
<a href="#">overlay.overlay</a>	
The overlay widget class . . . . .	27
<a href="#">ring_generator.ring_generator</a>	
Class to draw rings . . . . .	29
<a href="#">square_generator.square_generator</a>	
Class to draw squares . . . . .	30
<a href="#">ui_controller.ui_controller</a>	
The <a href="#">ui_controller</a> class . . . . .	31
<a href="#">widget.widget</a>	
An abstract class for widgets to extend . . . . .	34
<a href="#">widget_storage.widget_storage</a>	
Storage for program widgets . . . . .	36



## Chapter 3

# File Index

### 3.1 File List

Here is a list of all documented files with brief descriptions:

<a href="#">assets.py</a>	Stores various assets useful to other modules . . . . .	37
<a href="#">canvas.py</a>	Defines the canvas class . . . . .	38
<a href="#">circle_generator.py</a>	Defines the circle_generator class . . . . .	39
<a href="#">color_palette.py</a>	Defines the color_palette class . . . . .	40
<a href="#">curves_generator.py</a>	Defines the curves_generator class . . . . .	40
<a href="#">dots_generator.py</a>	Defines the dots_generator class . . . . .	41
<a href="#">fpolygons_generator.py</a>	Defines the fpolygons_generator class . . . . .	42
<a href="#">generator.py</a>	Defines the abstract class generator . . . . .	43
<a href="#">generator_storage.py</a>	Defines and initializes the generator_storage class . . . . .	43
<a href="#">help.py</a>	Defines the help class . . . . .	44
<a href="#">hpolygons_generator.py</a>	Defines the hpolygons_generator class . . . . .	44
<a href="#">layer.py</a>	Defines the layer class . . . . .	45
<a href="#">line_generator.py</a>	Defines the line_generator class . . . . .	46
<a href="#">overlay.py</a>	Defines the overlay class . . . . .	47
<a href="#">square_generator.py</a>	Defines the square_generator class . . . . .	47
<a href="#">ui_controller.py</a>	Defines and initializes the ui_controller class . . . . .	48
<a href="#">widget.py</a>	Defines the widget abstract class . . . . .	48
<a href="#">widget_storage.py</a>	Defines and initializes the widget_storage class . . . . .	49



## Chapter 4

# Class Documentation

### 4.1 canvas.canvas Class Reference

The canvas class.

#### Public Member Functions

- def `__init__` (self, x, y, width, height, display\_width, display\_height, window)  
*Initializes the canvas.*
- def `draw` (self)  
*Draws the canvas to the ui.*
- def `draw_to_canvas` (self)  
*Draws layers to the canvas.*
- def `generate_bg` (self, color)  
*Fill the canvas background with a color.*
- def `get_canvas` (self)  
*Gets the pygame surface the canvas draws on.*
- def `get_height` (self)
- def `get_width` (self)

#### 4.1.1 Detailed Description

The canvas class.

Provides the canvas the program draws on along with functions for drawing layers to the canvas and drawing the canvas to the ui.

#### 4.1.2 Constructor & Destructor Documentation

#### 4.1.2.1 `__init__()`

```
def canvas.canvas.__init__ (
    self,
    x,
    y,
    width,
    height,
    display_width,
    display_height,
    window )
```

Initializes the canvas.

##### Parameters

<i>x</i>	Horizontal position to draw the canvas at on the ui.
<i>y</i>	Vertical position to draw the canvas at on the ui.
<i>width</i>	Width of the canvas surface.
<i>height</i>	Height of the canvas surface.
<i>display_width</i>	Width of the ui's canvas display port.
<i>display_height</i>	Height of the ui's canvas display port.
<i>window</i>	Ui window to draw the canvas to.

### 4.1.3 Member Function Documentation

#### 4.1.3.1 `draw_to_canvas()`

```
def canvas.canvas.draw_to_canvas (
    self )
```

Draws layers to the canvas.

Calls any drawing widgets to draw to the canvas.

#### 4.1.3.2 `generate_bg()`

```
def canvas.canvas.generate_bg (
    self,
    color )
```

Fill the canvas background with a color.

##### Parameters

<i>color</i>	Color to fill the background with.
--------------	------------------------------------



#### 4.1.3.3 get\_canvas()

```
def canvas.canvas.get_canvas (
    self )
```

Gets the pygame surface the canvas draws on.

##### Returns

The pygame surface the canvas draws on.

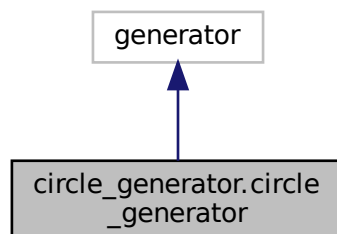
The documentation for this class was generated from the following file:

- [canvas.py](#)

## 4.2 circle\_generator.circle\_generator Class Reference

Class to draw circles.

Inheritance diagram for circle\_generator.circle\_generator:



### Public Member Functions

- def [draw](#) (layer, complexity, cp, style, magnitude)  
*Draws circlces to a layer.*

#### 4.2.1 Detailed Description

Class to draw circles.

## 4.2.2 Member Function Documentation

### 4.2.2.1 draw()

```
def circle_generator.circle_generator.draw (
    layer,
    complexity,
    cp,
    style,
    magnitude )
```

Draws cirlces to a layer.

#### Parameters

<i>layer</i>	The layer to draw to.
<i>complexity</i>	The complexity of the layer.
<i>cp</i>	The color palette to draw with.
<i>style</i>	The style of the layer.
<i>magnitude</i>	The magnitude of the layer.

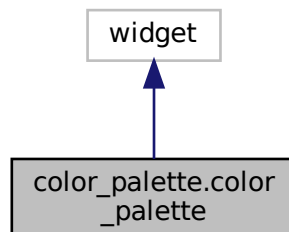
The documentation for this class was generated from the following file:

- [circle\\_generator.py](#)

## 4.3 color\_palette.color\_palette Class Reference

The color palette widget class.

Inheritance diagram for color\_palette.color\_palette:



## Public Member Functions

- def `__init__` (self, x, y, window, ui\_manager)  
*Initializes the color palette widget.*
- def `draw_ui_dynamic` (self)  
*Draws the dynamic ui elements for the color palette widget.*
- def `draw_ui_static` (self)  
*Draws the static ui elements for the color palette widget.*
- def `events` (self, event)  
*Processes pygame events for the color palette widget.*
- def `get_background_color` (self)  
*Get the background color in hex form.*
- def `get_colors_from_palette` (self)  
*Get the list of colors for the current palette in hex form.*
- def `get_foreground_colors` (self)  
*Get the list of foreground colors in hex form.*
- def `get_name_of_palette` (self)  
*Get the name of the current palette.*
- def `randomize` (self)  
*Randomize the current color palette and background color.*
- def `refresh_ui_static` (self)  
*Refreshes the static ui elements for the color palette widget.*

## Public Attributes

- `background_index_buttons`

### 4.3.1 Detailed Description

The color palette widget class.

Provides a ui and functionality to specify the current color palette and the background color.

### 4.3.2 Constructor & Destructor Documentation

#### 4.3.2.1 `__init__()`

```
def color_palette.color_palette.__init__ (
    self,
    x,
    y,
    window,
    ui_manager )
```

Initializes the color palette widget.

**Parameters**

<i>x</i>	Horizontal position to draw the widget at on the ui.
<i>y</i>	Vertical position to draw the widget at on the ui.
<i>window</i>	Ui window to draw the widget to.
<i>ui_manager</i>	Pygame_gui element manager to tie pygame_gui elements to.

### 4.3.3 Member Function Documentation

#### 4.3.3.1 draw\_ui\_dynamic()

```
def color_palette.color_palette.draw_ui_dynamic (
    self )
```

Draws the dynamic ui elements for the color palette widget.

Draws the text, lock icons, and color swatches.

#### 4.3.3.2 draw\_ui\_static()

```
def color_palette.color_palette.draw_ui_static (
    self )
```

Draws the static ui elements for the color palette widget.

Draws the palette dropdown, lock buttons, and color swatch buttons.

#### 4.3.3.3 events()

```
def color_palette.color_palette.events (
    self,
    event )
```

Processes pygame events for the color palette widget.

Handles the palette dropdown, lock buttons, and background color buttons.

**Parameters**

<i>event</i>	The pygame event being processed.
--------------	-----------------------------------

#### 4.3.3.4 get\_background\_color()

```
def color_palette.color_palette.get_background_color (
    self )
```

Get the background color in hex form.

##### Returns

The background color.

#### 4.3.3.5 get\_colors\_from\_palette()

```
def color_palette.color_palette.get_colors_from_palette (
    self )
```

Get the list of colors for the current palette in hex form.

##### Returns

A list of the palette colors.

#### 4.3.3.6 get\_foreground\_colors()

```
def color_palette.color_palette.get_foreground_colors (
    self )
```

Get the list of foreground colors in hex form.

##### Returns

A list of the palette colors excluding the background color.

#### 4.3.3.7 get\_name\_of\_palette()

```
def color_palette.color_palette.get_name_of_palette (
    self )
```

Get the name of the current palette.

##### Returns

The palette name.

#### 4.3.3.8 refresh\_ui\_static()

```
def color_palette.color_palette.refresh_ui_static (
    self )
```

Refreshes the static ui elements for the color palette widget.

Changes how many color swatch buttons display based on the length of the color palette.

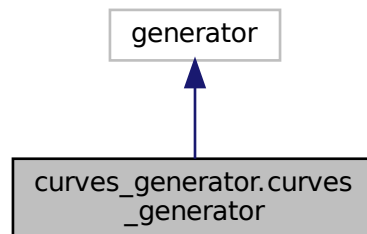
The documentation for this class was generated from the following file:

- [color\\_palette.py](#)

## 4.4 curves\_generator.curves\_generator Class Reference

Class to draw curves.

Inheritance diagram for curves\_generator.curves\_generator:



### Public Member Functions

- def [draw](#) (layer, complexity, cp, style, magnitude)  
*Draws curves to a layer.*

#### 4.4.1 Detailed Description

Class to draw curves.

#### 4.4.2 Member Function Documentation

##### 4.4.2.1 draw()

```
def curves_generator.curves_generator.draw (
    layer,
    complexity,
    cp,
    style,
    magnitude )
```

Draws curves to a layer.

## Parameters

<i>layer</i>	The layer to draw to.
<i>complexity</i>	The complexity of the layer.
<i>cp</i>	The color palette to draw with.
<i>style</i>	The style of the layer.
<i>magnitude</i>	The magnitude of the layer.

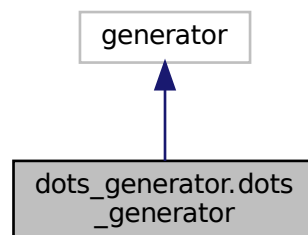
The documentation for this class was generated from the following file:

- [curves\\_generator.py](#)

## 4.5 dots\_generator.dots\_generator Class Reference

Class to draw dots.

Inheritance diagram for dots\_generator.dots\_generator:



### Public Member Functions

- def [draw](#) (layer, complexity, cp, style, magnitude)  
*Draws dots to a layer.*

#### 4.5.1 Detailed Description

Class to draw dots.

#### 4.5.2 Member Function Documentation

#### 4.5.2.1 draw()

```
def dots_generator.dots_generator.draw (
    layer,
    complexity,
    cp,
    style,
    magnitude )
```

Draws dots to a layer.

##### Parameters

<i>layer</i>	The layer to draw to.
<i>complexity</i>	The complexity of the layer.
<i>cp</i>	The color palette to draw with.
<i>style</i>	The style of the layer.
<i>magnitude</i>	The magnitude of the layer.

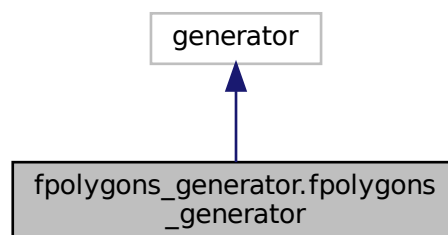
The documentation for this class was generated from the following file:

- [dots\\_generator.py](#)

## 4.6 fpolygons\_generator.fpolygons\_generator Class Reference

Class to draw filled polygons.

Inheritance diagram for fpolygons\_generator.fpolygons\_generator:



### Public Member Functions

- def [draw](#) (layer, complexity, cp, style, magnitude)  
*Draws filled polygons to a layer.*



### 4.6.1 Detailed Description

Class to draw filled polygons.

### 4.6.2 Member Function Documentation

#### 4.6.2.1 draw()

```
def fpolygons_generator.fpolygons_generator.draw (
    layer,
    complexity,
    cp,
    style,
    magnitude )
```

Draws filled polygons to a layer.

#### Parameters

<i>layer</i>	The layer to draw to.
<i>complexity</i>	The complexity of the layer.
<i>cp</i>	The color palette to draw with.
<i>style</i>	The style of the layer.
<i>magnitude</i>	The magnitude of the layer.

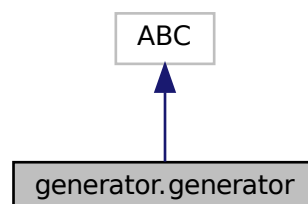
The documentation for this class was generated from the following file:

- [fpolygons\\_generator.py](#)

## 4.7 generator.generator Class Reference

Abstract class for generators to extend.

Inheritance diagram for generator.generator:



## Public Member Functions

- def [draw](#) (layer, complexity, cp, style, magnitude)  
*Draws to a layer.*

### 4.7.1 Detailed Description

Abstract class for generators to extend.

### 4.7.2 Member Function Documentation

#### 4.7.2.1 draw()

```
def generator.generator.draw (
    layer,
    complexity,
    cp,
    style,
    magnitude )
```

Draws to a layer.

#### Parameters

<i>layer</i>	The layer to draw to.
<i>complexity</i>	The complexity of the layer.
<i>cp</i>	The color palette to draw with.
<i>style</i>	The style of the layer.
<i>magnitude</i>	The magnitude of the layer.

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

- [generator.py](#)

## 4.8 generator\_storage.generator\_storage Class Reference

Storage for program generators .

## Public Member Functions

- def `__init__` (self)

## Public Attributes

- [circle\\_generator](#)  
*The color palette widget.*

### 4.8.1 Detailed Description

Storage for program generators .

Allows all other modules to access program generators.

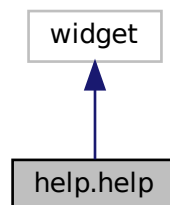
The documentation for this class was generated from the following file:

- [generator\\_storage.py](#)

## 4.9 help.help Class Reference

The help widget class.

Inheritance diagram for help.help:



## Public Member Functions

- def [\\_\\_init\\_\\_](#) (self, x, y, window, ui\_manager)  
*Initializes the help widget.*
- def [draw\\_ui\\_dynamic](#) (self)  
*Draws the dynamic ui elements for the help widget.*
- def [draw\\_ui\\_static](#) (self)  
*Draws the static ui elements for the help widget.*
- def [events](#) (self, event)  
*Processes pygame events for the help widget.*

### 4.9.1 Detailed Description

The help widget class.

Displays a ui help button that displays the program instructions when clicked.

### 4.9.2 Constructor & Destructor Documentation

#### 4.9.2.1 `__init__()`

```
def help.help.__init__ (
    self,
    x,
    y,
    window,
    ui_manager )
```

Initializes the help widget.

##### Parameters

<i>x</i>	Horizontal position to draw the widget at on the ui.
<i>y</i>	Vertical position to draw the widget at on the ui.
<i>window</i>	Ui window to draw the widget to.
<i>ui_manager</i>	Pygame_gui element manager to tie pygame_gui elements to.

### 4.9.3 Member Function Documentation

#### 4.9.3.1 `draw_ui_dynamic()`

```
def help.help.draw_ui_dynamic (
    self )
```

Draws the dynamic ui elements for the help widget.

Draws a dialog with the instructions for using the program.

#### 4.9.3.2 `draw_ui_static()`

```
def help.help.draw_ui_static (
    self )
```

Draws the static ui elements for the help widget.

Draws a button with "help" written on it.

#### 4.9.3.3 events()

```
def help.help.events (
    self,
    event )
```

Processes pygame events for the help widget.

If event in the help button being pressed display the instructions dialog.

##### Parameters

<i>event</i>	The pygame event being processed.
--------------	-----------------------------------

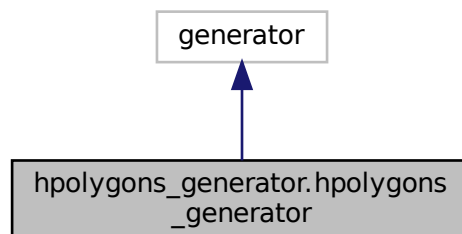
The documentation for this class was generated from the following file:

- [help.py](#)

## 4.10 hpolygons\_generator.hpolygons\_generator Class Reference

Class to draw hollow polygons.

Inheritance diagram for hpolygons\_generator.hpolygons\_generator:



### Public Member Functions

- def [draw](#) (layer, complexity, cp, style, magnitude)  
*Draws hollow polygons to a layer.*

#### 4.10.1 Detailed Description

Class to draw hollow polygons.

## 4.10.2 Member Function Documentation

### 4.10.2.1 draw()

```
def hpolygons_generator.hpolygons_generator.draw (
    layer,
    complexity,
    cp,
    style,
    magnitude )
```

Draws hollow polygons to a layer.

#### Parameters

<i>layer</i>	The layer to draw to.
<i>complexity</i>	The complexity of the layer.
<i>cp</i>	The color palette to draw with.
<i>style</i>	The style of the layer.
<i>magnitude</i>	The magnitude of the layer.

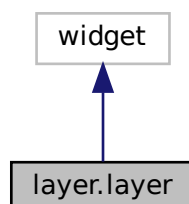
The documentation for this class was generated from the following file:

- [hpolygons\\_generator.py](#)

## 4.11 layer.layer Class Reference

The layer widget class.

Inheritance diagram for layer.layer:



## Public Member Functions

- def `__init__` (self, x, y, window, ui\_manager, layer\_num)  
*Initializes the layer widget.*
- def `clean_layer` (self)  
*Clean the layer by setting it to be blank and see-through.*
- def `draw_canvas` (self)  
*Draw the layer to self.layer based on the current widget settings.*
- def `draw_ui_dynamic` (self)  
*Draws the dynamic ui elements for the layer widget.*
- def `draw_ui_static` (self)  
*Draws the static ui elements for the layer widget.*
- def `events` (self, event)  
*Processes pygame events for the layer widget.*
- def `get_layer_complexity` (self)
- def `get_layer_shape` (self)
- def `get_layer_size` (self)
- def `get_layer_style` (self)
- def `randomize` (self)  
*Randomize the shape, style, complexity, and size of the layer drawing algorithm.*

## Public Attributes

- `layer`  
*The pygame surface the layer draws to.*

### 4.11.1 Detailed Description

The layer widget class.

Provides a ui and functionality to specify a drawing algorithm and draw to a pygame surface.

### 4.11.2 Constructor & Destructor Documentation

#### 4.11.2.1 `__init__()`

```
def layer.layer.__init__ (  
    self,  
    x,  
    y,  
    window,  
    ui_manager,  
    layer_num )
```

Initializes the layer widget.

**Parameters**

<i>x</i>	Horizontal position to draw the widget at on the ui.
<i>y</i>	Vertical position to draw the widget at on the ui.
<i>window</i>	Ui window to draw the widget to.
<i>ui_manager</i>	Pygame_gui element manager to tie pygame_gui elements to.

**4.11.3 Member Function Documentation****4.11.3.1 draw\_ui\_dynamic()**

```
def layer.layer.draw_ui_dynamic (
    self )
```

Draws the dynamic ui elements for the layer widget.

Draws the text and lock icons.

**4.11.3.2 draw\_ui\_static()**

```
def layer.layer.draw_ui_static (
    self )
```

Draws the static ui elements for the layer widget.

Draws the shape and style dropdowns, the complexity and size sliders, and lock button.

**4.11.3.3 events()**

```
def layer.layer.events (
    self,
    event )
```

Processes pygame events for the layer widget.

Handles the shape and style dropdowns, the complexity and size sliders, and lock button.

**Parameters**

<i>event</i>	The pygame event being processed.
--------------	-----------------------------------



#### 4.11.3.4 get\_layer\_complexity()

```
def layer.layer.get_layer_complexity (
    self )
```

##### Returns

The layer complexity.

#### 4.11.3.5 get\_layer\_shape()

```
def layer.layer.get_layer_shape (
    self )
```

##### Returns

The layer shape.

#### 4.11.3.6 get\_layer\_size()

```
def layer.layer.get_layer_size (
    self )
```

##### Returns

The layer size.

#### 4.11.3.7 get\_layer\_style()

```
def layer.layer.get_layer_style (
    self )
```

##### Returns

The layer style.

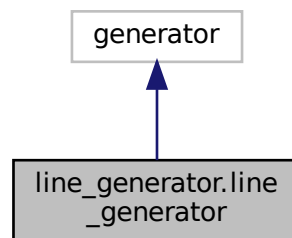
The documentation for this class was generated from the following file:

- [layer.py](#)

## 4.12 line\_generator.line\_generator Class Reference

Class to draw lines.

Inheritance diagram for line\_generator.line\_generator:



### Public Member Functions

- def [draw](#) (layer, complexity, cp, style, magnitude)  
*Draws lines to a layer.*

#### 4.12.1 Detailed Description

Class to draw lines.

#### 4.12.2 Member Function Documentation

##### 4.12.2.1 draw()

```

def line_generator.line_generator.draw (
    layer,
    complexity,
    cp,
    style,
    magnitude )
  
```

Draws lines to a layer.

##### Parameters

<i>layer</i>	The layer to draw to.
<i>complexity</i>	The complexity of the layer.
<i>cp</i>	The color palette to draw with.
<i>style</i>	The style of the layer.
<i>magnitude</i>	The magnitude of the layer.

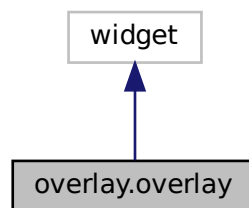
The documentation for this class was generated from the following file:

- [line\\_generator.py](#)

## 4.13 overlay.overlay Class Reference

The overlay widget class.

Inheritance diagram for overlay.overlay:



### Public Member Functions

- `def __init__ (self, x, y, window, ui_manager, layer_num)`  
*Initializes the overlay widget.*
- `def clean_layer (self)`  
*Clean the overlay by setting it to be blank and see-through.*
- `def draw_canvas (self)`  
*Draw the currently selected overlay image to self.overlay\_layer.*
- `def draw_ui_dynamic (self)`  
*Draws the dynamic ui elements for the overlay widget.*
- `def draw_ui_static (self)`  
*Draws the static ui elements for the overlay widget.*
- `def events (self, event)`  
*Processes pygame events for the overlay widget.*

### Public Attributes

- `overlay_layer`  
*The pygame surface the overlay draws to.*

#### 4.13.1 Detailed Description

The overlay widget class.

Provides a ui and functionality to specify an overlay and draw to a pygame surface.

## 4.13.2 Constructor & Destructor Documentation

### 4.13.2.1 `__init__()`

```
def overlay.overlay.__init__ (
    self,
    x,
    y,
    window,
    ui_manager,
    layer_num )
```

Initializes the overlay widget.

#### Parameters

<i>x</i>	Horizontal position to draw the widget at on the ui.
<i>y</i>	Vertical position to draw the widget at on the ui.
<i>window</i>	Ui window to draw the widget to.
<i>ui_manager</i>	Pygame_gui element manager to tie pygame_gui elements to.

## 4.13.3 Member Function Documentation

### 4.13.3.1 `draw_ui_dynamic()`

```
def overlay.overlay.draw_ui_dynamic (
    self )
```

Draws the dynamic ui elements for the overlay widget.

Draws the text and overlay thumbnails.

### 4.13.3.2 `draw_ui_static()`

```
def overlay.overlay.draw_ui_static (
    self )
```

Draws the static ui elements for the overlay widget.

Draws the overlay selection buttons.

### 4.13.3.3 `events()`

```
def overlay.overlay.events (
    self,
    event )
```

Processes pygame events for the overlay widget.

Handles the overlay selection buttons.

## Parameters

<i>event</i>	The pygame event being processed.
--------------	-----------------------------------

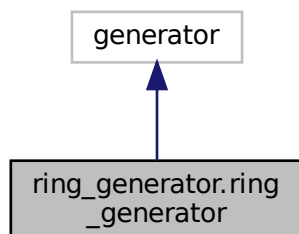
The documentation for this class was generated from the following file:

- [overlay.py](#)

## 4.14 ring\_generator.ring\_generator Class Reference

Class to draw rings.

Inheritance diagram for ring\_generator.ring\_generator:



### Public Member Functions

- def [draw](#) (layer, complexity, cp, style, magnitude)  
*Draws rings to a layer.*

#### 4.14.1 Detailed Description

Class to draw rings.

#### 4.14.2 Member Function Documentation

##### 4.14.2.1 draw()

```
def ring_generator.ring_generator.draw (  
    layer,  
    complexity,  
    cp,  
    style,  
    magnitude )
```

Draws rings to a layer.

**Parameters**

<i>layer</i>	The layer to draw to.
<i>complexity</i>	The complexity of the layer.
<i>cp</i>	The color palette to draw with.
<i>style</i>	The style of the layer.
<i>magnitude</i>	The magnitude of the layer.

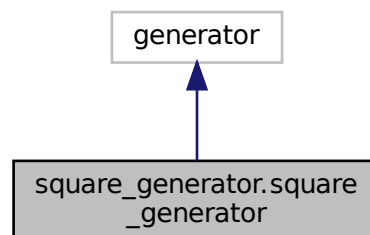
The documentation for this class was generated from the following file:

- ring\_generator.py

## 4.15 square\_generator.square\_generator Class Reference

Class to draw squares.

Inheritance diagram for square\_generator.square\_generator:



### Public Member Functions

- def [draw](#) (layer, complexity, cp, style, magnitude)  
*Draws squares to a layer.*

#### 4.15.1 Detailed Description

Class to draw squares.

#### 4.15.2 Member Function Documentation

## 4.15.2.1 draw()

```
def square_generator.square_generator.draw (
    layer,
    complexity,
    cp,
    style,
    magnitude )
```

Draws squares to a layer.

## Parameters

<i>layer</i>	The layer to draw to.
<i>complexity</i>	The complexity of the layer.
<i>cp</i>	The color palette to draw with.
<i>style</i>	The style of the layer.
<i>magnitude</i>	The magnitude of the layer.

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

- [square\\_generator.py](#)

## 4.16 ui\_controller.ui\_controller Class Reference

The [ui\\_controller](#) class.

## Public Member Functions

- def [\\_\\_init\\_\\_](#) (self)  
*Initializes [ui\\_controller](#).*
- def [draw\\_ui\\_dynamic](#) (self)  
*Draws the dynamic ui.*
- def [draw\\_ui\\_static](#) (self)  
*Draws the static ui.*
- def [export\\_art](#) (self)  
*Exports the canvas to a png image.*
- def [process\\_events](#) (self)  
*Processes pygame events.*
- def [run](#) (self)  
*Main loop.*

## Public Attributes

- [canvas](#)  
*The canvas to draw the generated art on.*
- [isrunning](#)  
*A boolean that specifies if the program is running, program terminates if False.*
- [ui\\_manager](#)  
*Manages pygame\_gui elements and events.*
- [window](#)  
*Program window.*

## Static Public Attributes

- tuple `canvas_display_size` = (int(`SW`//1.8), int(`SH`//1.8))  
*Canvas ui display port size.*
- tuple `canvas_pos` = ((`SW` - `canvas_display_size`[0])//2, (`SH` - `canvas_display_size`[1])//2)  
*Position of the canvas on the ui.*
- tuple `canvas_size` = (3840, 2160)  
*Canvas internal size.*
- list `export_resolution` = `resolutions_list`[0]  
*Current canvas export resolution.*
- tuple `help_pos` = (284, 60)  
*Position of the help widget.*
- tuple `layer_one_pos` = (`_ui_menus_left`, 60)  
*Position of the layer one widget.*
- tuple `layer_three_pos` = (`_ui_menus_left`, `layer_two_pos`[1]+200)  
*Position of the layer three widget.*
- tuple `layer_two_pos` = (`_ui_menus_left`, `layer_one_pos`[1]+200)  
*Position of the layer two widget.*
- tuple `overlay_pos` = (0, `palette_pos`[1]+155)  
*Position of the overlay widget.*
- tuple `palette_pos` = (`_ui_menus_right`, 60)  
*Position of the color palette widget.*
- list `resolutions_list`  
*Possible canvas export resolutions.*
- int `SH` = 720  
*Application window height.*
- int `SW` = 1280  
*Application window width.*

### 4.16.1 Detailed Description

The `ui_controller` class.

A high level class that calls all other modules. Orchestrates pygame event handling, ui drawing, art generation, randomization, and art exporting.

### 4.16.2 Constructor & Destructor Documentation

#### 4.16.2.1 `__init__()`

```
def ui_controller.ui_controller.__init__ (
    self )
```

Initializes `ui_controller`.

Initializes pygame, the application window, the canvas, and all widgets.



### 4.16.3 Member Function Documentation

#### 4.16.3.1 draw\_ui\_dynamic()

```
def ui_controller.ui_controller.draw_ui_dynamic (
    self )
```

Draws the dynamic ui.

Draws the background color and some text itself and calls canvas and widgets for all other drawing.

#### 4.16.3.2 draw\_ui\_static()

```
def ui_controller.ui_controller.draw_ui_static (
    self )
```

Draws the static ui.

Draws generation and export controls itself and calls widgets for all other drawing.

#### 4.16.3.3 process\_events()

```
def ui_controller.ui_controller.process_events (
    self )
```

Processes pygame events.

Handles generation and export controls itself and calls events() in widgets for all other event processing.

#### 4.16.3.4 run()

```
def ui_controller.ui_controller.run (
    self )
```

Main loop.

Draws static ui then enters loop where it processes events and draws the dynamic ui.

### 4.16.4 Member Data Documentation

#### 4.16.4.1 resolutions\_list

```
list ui_controller.ui_controller.resolutions_list [static]
```

**Initial value:**

```
= [
    "4K: 3840x2160",
    "Full HD: 1920x1080",
    "HD: 1280x720"
]
```

Possible canvas export resolutions.

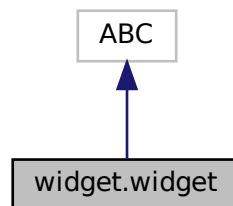
The documentation for this class was generated from the following file:

- [ui\\_controller.py](#)

## 4.17 widget.widget Class Reference

An abstract class for widgets to extend.

Inheritance diagram for widget.widget:



### Public Member Functions

- def `__init__` (self, x, y, window, ui\_manager)  
*Initializes the widget.*
- def `draw_canvas` (self)  
*Draw to the canvas.*
- def `draw_ui_dynamic` (self)  
*Draw ui elements that need to be refreshed each frame.*
- def `draw_ui_static` (self)  
*Draw ui elements that only need to be drawn once.*
- def `events` (self, event)  
*Handle pygame events for the widget.*
- def `randomize` (self)  
*Randomize the widget settings.*
- def `refresh_ui_static` (self)  
*Refresh the static ui elements.*

### 4.17.1 Detailed Description

An abstract class for widgets to extend.

Provides an interface widgets typically use.

### 4.17.2 Constructor & Destructor Documentation

#### 4.17.2.1 `__init__()`

```
def widget.widget.__init__ (
    self,
    x,
    y,
    window,
    ui_manager )
```

Initializes the widget.

##### Parameters

<i>x</i>	Horizontal position to draw the widget at on the ui.
<i>y</i>	Vertical position to draw the widget at on the ui.
<i>window</i>	Ui window to draw the widget to.
<i>ui_manager</i>	Pygame_gui element manager to tie pygame_gui elements to.

### 4.17.3 Member Function Documentation

#### 4.17.3.1 `draw_ui_dynamic()`

```
def widget.widget.draw_ui_dynamic (
    self )
```

Draw ui elements that need to be refreshed each frame.

For our purposes draws everything that isn't a pygame\_gui element.

#### 4.17.3.2 `draw_ui_static()`

```
def widget.widget.draw_ui_static (
    self )
```

Draw ui elements that only need to be drawn once.

For our purposes draws pygame\_gui elements.

#### 4.17.3.3 events()

```
def widget.widget.events (
    self,
    event )
```

Handle pygame events for the widget.

##### Parameters

<i>event</i>	The event to be processed.
--------------	----------------------------

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

- [widget.py](#)

## 4.18 widget\_storage.widget\_storage Class Reference

Storage for program widgets.

### Public Member Functions

- `def __init__ (self)`

### Public Attributes

- [color\\_palette](#)  
*The color palette widget.*
- [help](#)  
*The help button widget.*
- [layer\\_one](#)  
*The layer one widget.*
- [layer\\_three](#)  
*The layer three widget.*
- [layer\\_two](#)  
*The layer two widget.*

#### 4.18.1 Detailed Description

Storage for program widgets.

Allows all other modules to access program widgets.

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

- [widget\\_storage.py](#)

## Chapter 5

# File Documentation

### 5.1 assets.py File Reference

Stores various assets useful to other modules.

#### Functions

- def `assets.text_to_screen` (window, text, color, pos, font\_size)  
*Draws text to ui.*

#### Variables

- tuple `assets.active_color` = (90, 90, 90)  
*Color used to indicate active settings.*
- `assets.background_color` = pg.Color("#322f3d")  
*Background\_color of ui.*
- list `assets.font_sizes` = [12, 14, 18, 24, 30, 40]  
*Font size numbers that correspond with defined font sizes.*
- list `assets.fonts` = [xs\_font, small\_font, medium\_font, large\_font, xl\_font, xxl\_font]  
*List of font sizes.*
- tuple `assets.inactive_color` = (20, 20, 20)  
*Color used to indicate inactive settings.*
- `assets.large_font` = pg.freetype.Font("Basic-Regular.ttf", 24)  
*Large font.*
- `assets.lock_disabled` = pg.transform.scale(pg.image.load("assets/lock\_disabled.png"), (20, 20))  
*Lock disabled graphic.*
- `assets.lock_enabled` = pg.transform.scale(pg.image.load("assets/lock\_enabled.png"), (20, 20))  
*Lock enabled graphic.*
- `assets.logo` = pg.image.load("assets/logo.png")  
*Program logo.*
- `assets.medium_font` = pg.freetype.Font("Basic-Regular.ttf", 18)  
*Medium font.*
- `assets.small_font` = pg.freetype.Font("Basic-Regular.ttf", 14)  
*Small font.*

- `assets.ui_color = pg.Color("#DFD6FF")`  
*Color used for smaller text elements.*
- tuple `assets.ui_h1_color = (250, 250, 250)`  
*Color used for larger text elements.*
- `assets.xl_font = pg.freetype.Font("Basic-Regular.ttf", 30)`  
*Extra large font.*
- `assets.xs_font = pg.freetype.Font("Basic-Regular.ttf", 12)`  
*Extra small font.*
- `assets.xxl_font = pg.freetype.Font("Basic-Regular.ttf", 40)`  
*Extra extra large font.*

### 5.1.1 Detailed Description

Stores various assets useful to other modules.

### 5.1.2 Author(s)

- Created by Jessica Dawson on 03/16/2022.

### 5.1.3 Function Documentation

#### 5.1.3.1 `text_to_screen()`

```
def assets.text_to_screen (
    window,
    text,
    color,
    pos,
    font_size )
```

Draws text to ui.

#### Parameters

<i>window</i>	Ui window to draw to.
<i>text</i>	Text to draw.
<i>color</i>	Color of text.
<i>pos</i>	Position of text.
<i>font_size</i>	Size of text.

## 5.2 `canvas.py` File Reference

Defines the canvas class.

## Classes

- class [canvas.canvas](#)

*The canvas class.*

### 5.2.1 Detailed Description

Defines the canvas class.

### 5.2.2 Author(s)

- Created by Jessica Dawson on 03/16/2022.

## 5.3 circle\_generator.py File Reference

Defines the circle\_generator class.

## Classes

- class [circle\\_generator.circle\\_generator](#)

*Class to draw circles.*

## Variables

- list [circle\\_generator.art\\_styles\\_list](#)

### 5.3.1 Detailed Description

Defines the circle\_generator class.

### 5.3.2 Author(s)

- Created by Jessica Dawson on 03/17/2022.

### 5.3.3 Variable Documentation

### 5.3.3.1 art\_styles\_list

```
list circle_generator.art_styles_list
```

#### Initial value:

```
1 = [  
2     "Chaotic",  
3     "Striped Horizontal",  
4     "Striped Vertical",  
5     "Mosaic",  
6     "Cornered",  
7     "Centered",  
8     "Empty"  
9 ]
```

## 5.4 color\_palette.py File Reference

Defines the color\_palette class.

### Classes

- class [color\\_palette.color\\_palette](#)  
*The color palette widget class.*

### 5.4.1 Detailed Description

Defines the color\_palette class.

### 5.4.2 Author(s)

- Created by Jessica Dawson on 03/16/2022.

## 5.5 curves\_generator.py File Reference

Defines the curves\_generator class.

### Classes

- class [curves\\_generator.curves\\_generator](#)  
*Class to draw curves.*

### Variables

- list [curves\\_generator.art\\_styles\\_list](#)



### 5.5.1 Detailed Description

Defines the curves\_generator class.

### 5.5.2 Author(s)

- Created by Jessica Dawson on 03/17/2022.

### 5.5.3 Variable Documentation

#### 5.5.3.1 art\_styles\_list

list curves\_generator.art\_styles\_list

Initial value:

```
1 = [  
2     "Chaotic",  
3     "Striped Horizontal",  
4     "Striped Vertical",  
5     "Mosaic",  
6     "Cornered",  
7     "Centered",  
8     "Empty"  
9 ]
```

## 5.6 dots\_generator.py File Reference

Defines the dots\_generator class.

### Classes

- class [dots\\_generator.dots\\_generator](#)  
*Class to draw dots.*

### Variables

- list **dots\_generator.art\_styles\_list**

### 5.6.1 Detailed Description

Defines the dots\_generator class.

### 5.6.2 Author(s)

- Created by Jessica Dawson on 03/17/2022.

### 5.6.3 Variable Documentation

#### 5.6.3.1 art\_styles\_list

```
list dots_generator.art_styles_list
```

**Initial value:**

```
1 = [  
2     "Chaotic",  
3     "Striped Horizontal",  
4     "Striped Vertical",  
5     "Mosaic",  
6     "Cornered",  
7     "Centered",  
8     "Empty"  
9 ]
```

## 5.7 fpolygons\_generator.py File Reference

Defines the fpolygons\_generator class.

### Classes

- class [fpolygons\\_generator.fpolygons\\_generator](#)  
*Class to draw filled polygons.*

### Variables

- list `fpolygons_generator.art_styles_list`

#### 5.7.1 Detailed Description

Defines the fpolygons\_generator class.

#### 5.7.2 Author(s)

- Created by Jessica Dawson on 03/17/2022.

#### 5.7.3 Variable Documentation

### 5.7.3.1 art\_styles\_list

```
list fpolygons_generator.art_styles_list
```

**Initial value:**

```
1 = [  
2     "Chaotic",  
3     "Striped Horizontal",  
4     "Striped Vertical",  
5     "Mosaic",  
6     "Cornered",  
7     "Centered",  
8     "Empty"  
9 ]
```

## 5.8 generator.py File Reference

Defines the abstract class generator.

### Classes

- class [generator.generator](#)  
*Abstract class for generators to extend.*

### 5.8.1 Detailed Description

Defines the abstract class generator.

### 5.8.2 Author(s)

- Created by Jessica Dawson on 03/17/2022.

## 5.9 generator\_storage.py File Reference

Defines and initializes the generator\_storage class.

### Classes

- class [generator\\_storage.generator\\_storage](#)  
*Storage for program generators .*

### Variables

- [generator\\_storage.generator\\_storage](#) = None  
*Instance of generator\_storage to access generators through.*

### 5.9.1 Detailed Description

Defines and initializes the `generator_storage` class.

### 5.9.2 Author(s)

- Created by Jessica Dawson on 03/16/2022.

### 5.9.3 Variable Documentation

#### 5.9.3.1 `generator_storage`

```
generator_storage.generator_storage = None
```

Instance of `generator_storage` to access generators through.

Import this instance and access widgets with `widgets.widget_name()`

## 5.10 `help.py` File Reference

Defines the `help` class.

### Classes

- class `help.help`  
*The help widget class.*

#### 5.10.1 Detailed Description

Defines the `help` class.

#### 5.10.2 Author(s)

- Created by Jessica Dawson on 03/16/2022.

## 5.11 `hpolygons_generator.py` File Reference

Defines the `hpolygons_generator` class.

## Classes

- class [hpolygons\\_generator.hpolygons\\_generator](#)  
*Class to draw hollow polygons.*

## Variables

- list [hpolygons\\_generator.art\\_styles\\_list](#)

### 5.11.1 Detailed Description

Defines the hpolygons\_generator class.

### 5.11.2 Author(s)

- Created by Jessica Dawson on 03/17/2022.

### 5.11.3 Variable Documentation

#### 5.11.3.1 art\_styles\_list

list hpolygons\_generator.art\_styles\_list

##### Initial value:

```
1 = [  
2     "Chaotic",  
3     "Striped Horizontal",  
4     "Striped Vertical",  
5     "Mosaic",  
6     "Cornered",  
7     "Centered",  
8     "Empty"  
9 ]
```

## 5.12 layer.py File Reference

Defines the layer class.

## Classes

- class [layer.layer](#)  
*The layer widget class.*

### 5.12.1 Detailed Description

Defines the layer class.

### 5.12.2 Author(s)

- Created by Aamina Hussain on 03/17/2022.

## 5.13 line\_generator.py File Reference

Defines the line\_generator class.

### Classes

- class [line\\_generator.line\\_generator](#)  
*Class to draw lines.*

### Variables

- list [line\\_generator.art\\_styles\\_list](#)

### 5.13.1 Detailed Description

Defines the line\_generator class.

### 5.13.2 Author(s)

- Created by Jessica Dawson on 03/17/2022.

### 5.13.3 Variable Documentation

#### 5.13.3.1 art\_styles\_list

```
list line_generator.art_styles_list
```

#### Initial value:

```
1 = [  
2     "Chaotic",  
3     "Striped Horizontal",  
4     "Striped Vertical",  
5     "Mosaic",  
6     "Cornered",  
7     "Centered",  
8     "Empty"  
9 ]
```

## 5.14 overlay.py File Reference

Defines the overlay class.

### Classes

- class `overlay.overlay`  
*The overlay widget class.*

#### 5.14.1 Detailed Description

Defines the overlay class.

#### 5.14.2 Author(s)

- Created by Jessica Dawson on 03/17/2022.

## 5.15 square\_generator.py File Reference

Defines the square\_generator class.

### Classes

- class `square_generator.square_generator`  
*Class to draw squares.*

### Variables

- list `square_generator.art_styles_list`

#### 5.15.1 Detailed Description

Defines the square\_generator class.

#### 5.15.2 Author(s)

- Created by Jessica Dawson on 03/17/2022.

#### 5.15.3 Variable Documentation

### 5.15.3.1 art\_styles\_list

```
list square_generator.art_styles_list
```

#### Initial value:

```
1 = [  
2     "Chaotic",  
3     "Striped Horizontal",  
4     "Striped Vertical",  
5     "Mosaic",  
6     "Cornered",  
7     "Centered",  
8     "Empty"  
9 ]
```

## 5.16 ui\_controller.py File Reference

Defines and initializes the ui\_controller class.

### Classes

- class [ui\\_controller.ui\\_controller](#)  
*The ui\_controller class.*

### Variables

- `ui_controller.controller` = `ui_controller()`

### 5.16.1 Detailed Description

Defines and initializes the ui\_controller class.

### 5.16.2 Author(s)

- Created by Jessica Dawson on 03/16/2022.
- Modified by Aamina Hussain on 03/17/2022.

## 5.17 widget.py File Reference

Defines the widget abstract class.

### Classes

- class [widget.widget](#)  
*An abstract class for widgets to extend.*



### 5.17.1 Detailed Description

Defines the widget abstract class.

### 5.17.2 Author(s)

- Created by Jessica Dawson on 03/16/2022.

## 5.18 widget\_storage.py File Reference

Defines and initializes the widget\_storage class.

### Classes

- class `widget_storage.widget_storage`  
*Storage for program widgets.*

### Variables

- `widget_storage.widgets` = None  
*Instance of widget\_storage to access widgets through.*

### 5.18.1 Detailed Description

Defines and initializes the widget\_storage class.

### 5.18.2 Author(s)

- Created by Jessica Dawson on 03/16/2022.
- Modified by Aamina Hussain on 03/17/2022.

### 5.18.3 Variable Documentation

#### 5.18.3.1 widgets

```
widget_storage.widgets = None
```

Instance of widget\_storage to access widgets through.

Import this instance and access widgets with `widgets.widget_name`



# Index

- `__init__`
  - `canvas.canvas`, 7
  - `color_palette.color_palette`, 11
  - `help.help`, 20
  - `layer.layer`, 23
  - `overlay.overlay`, 28
  - `ui_controller.ui_controller`, 32
  - `widget.widget`, 35
- `art_styles_list`
  - `circle_generator.py`, 39
  - `curves_generator.py`, 41
  - `dots_generator.py`, 42
  - `fpolygons_generator.py`, 42
  - `hpolygons_generator.py`, 45
  - `line_generator.py`, 46
  - `square_generator.py`, 47
- `assets.py`, 37
  - `text_to_screen`, 38
- `canvas.canvas`, 7
  - `__init__`, 7
  - `draw_to_canvas`, 8
  - `generate_bg`, 8
  - `get_canvas`, 9
- `canvas.py`, 38
- `circle_generator.circle_generator`, 9
  - `draw`, 10
- `circle_generator.py`, 39
  - `art_styles_list`, 39
- `color_palette.color_palette`, 10
  - `__init__`, 11
  - `draw_ui_dynamic`, 12
  - `draw_ui_static`, 12
  - `events`, 12
  - `get_background_color`, 12
  - `get_colors_from_palette`, 13
  - `get_foreground_colors`, 13
  - `get_name_of_palette`, 13
  - `refresh_ui_static`, 13
- `color_palette.py`, 40
- `curves_generator.curves_generator`, 14
  - `draw`, 14
- `curves_generator.py`, 40
  - `art_styles_list`, 41
- `dots_generator.dots_generator`, 15
  - `draw`, 15
- `dots_generator.py`, 41
  - `art_styles_list`, 42
- `draw`
  - `circle_generator.circle_generator`, 10
  - `curves_generator.curves_generator`, 14
  - `dots_generator.dots_generator`, 15
  - `fpolygons_generator.fpolygons_generator`, 17
  - `generator.generator`, 18
  - `hpolygons_generator.hpolygons_generator`, 22
  - `line_generator.line_generator`, 26
  - `ring_generator.ring_generator`, 29
  - `square_generator.square_generator`, 30
- `draw_to_canvas`
  - `canvas.canvas`, 8
- `draw_ui_dynamic`
  - `color_palette.color_palette`, 12
  - `help.help`, 20
  - `layer.layer`, 24
  - `overlay.overlay`, 28
  - `ui_controller.ui_controller`, 33
  - `widget.widget`, 35
- `draw_ui_static`
  - `color_palette.color_palette`, 12
  - `help.help`, 20
  - `layer.layer`, 24
  - `overlay.overlay`, 28
  - `ui_controller.ui_controller`, 33
  - `widget.widget`, 35
- `events`
  - `color_palette.color_palette`, 12
  - `help.help`, 20
  - `layer.layer`, 24
  - `overlay.overlay`, 28
  - `widget.widget`, 35
- `fpolygons_generator.fpolygons_generator`, 16
  - `draw`, 17
- `fpolygons_generator.py`, 42
  - `art_styles_list`, 42
- `generate_bg`
  - `canvas.canvas`, 8
- `generator.generator`, 17
  - `draw`, 18
- `generator.py`, 43
- `generator_storage`
  - `generator_storage.py`, 44
- `generator_storage.generator_storage`, 18
- `generator_storage.py`, 43
  - `generator_storage`, 44
- `get_background_color`

- color\_palette.color\_palette, 12
- get\_canvas
  - canvas.canvas, 9
- get\_colors\_from\_palette
  - color\_palette.color\_palette, 13
- get\_foreground\_colors
  - color\_palette.color\_palette, 13
- get\_layer\_complexity
  - layer.layer, 24
- get\_layer\_shape
  - layer.layer, 25
- get\_layer\_size
  - layer.layer, 25
- get\_layer\_style
  - layer.layer, 25
- get\_name\_of\_palette
  - color\_palette.color\_palette, 13
- help.help, 19
  - \_\_init\_\_, 20
  - draw\_ui\_dynamic, 20
  - draw\_ui\_static, 20
  - events, 20
- help.py, 44
- hpolygons\_generator.hpolygons\_generator, 21
  - draw, 22
- hpolygons\_generator.py, 44
  - art\_styles\_list, 45
- layer.layer, 22
  - \_\_init\_\_, 23
  - draw\_ui\_dynamic, 24
  - draw\_ui\_static, 24
  - events, 24
  - get\_layer\_complexity, 24
  - get\_layer\_shape, 25
  - get\_layer\_size, 25
  - get\_layer\_style, 25
- layer.py, 45
- line\_generator.line\_generator, 26
  - draw, 26
- line\_generator.py, 46
  - art\_styles\_list, 46
- overlay.overlay, 27
  - \_\_init\_\_, 28
  - draw\_ui\_dynamic, 28
  - draw\_ui\_static, 28
  - events, 28
- overlay.py, 47
- process\_events
  - ui\_controller.ui\_controller, 33
- refresh\_ui\_static
  - color\_palette.color\_palette, 13
- resolutions\_list
  - ui\_controller.ui\_controller, 33
- ring\_generator.ring\_generator, 29
  - draw, 29
- run
  - ui\_controller.ui\_controller, 33
- square\_generator.py, 47
  - art\_styles\_list, 47
- square\_generator.square\_generator, 30
  - draw, 30
- text\_to\_screen
  - assets.py, 38
- ui\_controller.py, 48
- ui\_controller.ui\_controller, 31
  - \_\_init\_\_, 32
  - draw\_ui\_dynamic, 33
  - draw\_ui\_static, 33
  - process\_events, 33
  - resolutions\_list, 33
  - run, 33
- widget.py, 48
- widget.widget, 34
  - \_\_init\_\_, 35
  - draw\_ui\_dynamic, 35
  - draw\_ui\_static, 35
  - events, 35
- widget\_storage.py, 49
  - widgets, 49
- widget\_storage.widget\_storage, 36
- widgets
  - widget\_storage.py, 49