

## 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 <code>__init__()</code>	8
4.1.3 Member Function Documentation	8
4.1.3.1 <code>draw_to_canvas()</code>	8
4.1.3.2 <code>generate_bg()</code>	8
4.1.3.3 <code>get_canvas()</code>	9
4.2 color_palette.color_palette Class Reference	9
4.2.1 Detailed Description	10
4.2.2 Constructor & Destructor Documentation	10
4.2.2.1 <code>__init__()</code>	10
4.2.3 Member Function Documentation	11
4.2.3.1 <code>draw_ui_dynamic()</code>	11
4.2.3.2 <code>draw_ui_static()</code>	11
4.2.3.3 <code>events()</code>	11
4.2.3.4 <code>get_background_color()</code>	11
4.2.3.5 <code>get_colors_from_palette()</code>	12
4.2.3.6 <code>get_foreground_colors()</code>	12
4.2.3.7 <code>get_name_of_palette()</code>	12
4.2.3.8 <code>refresh_ui_static()</code>	12
4.3 help.help Class Reference	13
4.3.1 Detailed Description	13
4.3.2 Constructor & Destructor Documentation	13
4.3.2.1 <code>__init__()</code>	13
4.3.3 Member Function Documentation	14
4.3.3.1 <code>draw_ui_dynamic()</code>	14
4.3.3.2 <code>draw_ui_static()</code>	14
4.3.3.3 <code>events()</code>	14
4.4 ui_controller.ui_controller Class Reference	15
4.4.1 Detailed Description	16
4.4.2 Constructor & Destructor Documentation	16
4.4.2.1 <code>__init__()</code>	16

4.4.3 Member Function Documentation	16
4.4.3.1 draw_ui_dynamic()	16
4.4.3.2 draw_ui_static()	16
4.4.3.3 process_events()	16
4.4.3.4 run()	17
4.4.4 Member Data Documentation	17
4.4.4.1 resolutions_list	17
4.5 widget.widget Class Reference	17
4.5.1 Detailed Description	18
4.5.2 Constructor & Destructor Documentation	18
4.5.2.1 __init__()	18
4.5.3 Member Function Documentation	19
4.5.3.1 draw_ui_dynamic()	19
4.5.3.2 draw_ui_static()	19
4.5.3.3 events()	19
4.6 widget_storage.widget_storage Class Reference	19
4.6.1 Detailed Description	20
<b>5 File Documentation</b>	<b>21</b>
5.1 assets.py File Reference	21
5.1.1 Detailed Description	22
5.1.2 Author(s)	22
5.1.3 Function Documentation	22
5.1.3.1 text_to_screen()	22
5.2 canvas.py File Reference	22
5.2.1 Detailed Description	23
5.2.2 Author(s)	23
5.3 color_palette.py File Reference	23
5.3.1 Detailed Description	23
5.3.2 Author(s)	23
5.4 help.py File Reference	23
5.4.1 Detailed Description	24
5.4.2 Author(s)	24
5.5 ui_controller.py File Reference	24
5.5.1 Detailed Description	24
5.5.2 Author(s)	24
5.6 widget.py File Reference	24
5.6.1 Detailed Description	24
5.6.2 Author(s)	25
5.7 widget_storage.py File Reference	25
5.7.1 Detailed Description	25
5.7.2 Author(s)	25

---

5.7.3 Variable Documentation . . . . .	25
5.7.3.1 widgets . . . . .	25
<b>Index</b>	<b>27</b>



# Chapter 1

## Hierarchical Index

### 1.1 Class Hierarchy

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

canvas.canvas . . . . .	7
ui_controller.ui_controller . . . . .	15
widget_storage.widget_storage . . . . .	19
ABC	
widget.widget . . . . .	17
widget	
color_palette.color_palette . . . . .	9
help.help . . . . .	13





## 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="#">color_palette.color_palette</a>		
	The color palette widget class . . . . .	9
<a href="#">help.help</a>		
	The help widget class . . . . .	13
<a href="#">ui_controller.ui_controller</a>		
	The <a href="#">ui_controller</a> class . . . . .	15
<a href="#">widget.widget</a>		
	An abstract class for widgets to extend . . . . .	17
<a href="#">widget_storage.widget_storage</a>		
	Storage for program widgets . . . . .	19



## 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 . . . . .	21
<a href="#">canvas.py</a>	Defines the canvas class . . . . .	22
<a href="#">color_palette.py</a>	Defines the color_palette class . . . . .	23
<a href="#">help.py</a>	Defines the help class . . . . .	23
<a href="#">ui_controller.py</a>	Defines and initializes the ui_controller class . . . . .	24
<a href="#">widget.py</a>	Defines the widget abstract class . . . . .	24
<a href="#">widget_storage.py</a>	Defines and initializes the widget_storage class . . . . .	25



## 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.*

#### 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.

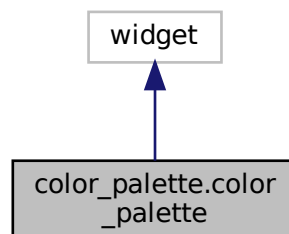
The documentation for this class was generated from the following file:

- [canvas.py](#)

## 4.2 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`

### 4.2.1 Detailed Description

The color palette widget class.

Provides ui settings to change the current color palette and the background color.

### 4.2.2 Constructor & Destructor Documentation

#### 4.2.2.1 `__init__()`

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

Initializes the color palette widget.

#### Parameters

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



## 4.2.3 Member Function Documentation

### 4.2.3.1 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.

### 4.2.3.2 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.

### 4.2.3.3 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.
--------------	-----------------------------------

### 4.2.3.4 get\_background\_color()

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

Get the background color in hex form.

#### Returns

The background color.

#### 4.2.3.5 `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.

#### 4.2.3.6 `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.

#### 4.2.3.7 `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.

#### 4.2.3.8 `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.

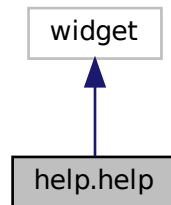
The documentation for this class was generated from the following file:

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

## 4.3 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.*

### 4.3.1 Detailed Description

The help widget class.

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

### 4.3.2 Constructor & Destructor Documentation

#### 4.3.2.1 `__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.

### 4.3.3 Member Function Documentation

#### 4.3.3.1 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.

#### 4.3.3.2 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.

#### 4.3.3.3 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](#)

## 4.4 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`
- `isrunning`
- `ui_manager`
- `window`

### Static Public Attributes

- tuple `canvas_display_size` = (int(`SW`//1.8), int(`SH`//1.8))
- tuple `canvas_pos` = ((`SW` - `canvas_display_size`[0])//2, (`SH` - `canvas_display_size`[1])//2)
- tuple `canvas_size` = (3840, 2160)
- list `export_resolution` = `resolutions_list`[0]  
*Current canvas export resolution.*
- tuple `help_pos` = (284, 60)
- tuple `layer_one_pos` = (`ui_menus_left`, 60)
- tuple `layer_three_pos` = (`ui_menus_left`, `layer_two_pos`[1]+200)
- tuple `layer_two_pos` = (`ui_menus_left`, `layer_one_pos`[1]+200)
- tuple `overlay_pos` = (0, `palette_pos`[1]+155)
- tuple `palette_pos` = (`ui_menus_right`, 60)
- list `resolutions_list`  
*Possible canvas export resolutions.*
- int `SH` = 720  
*Application window height.*
- int `SW` = 1280  
*Application window width.*
- int `ui_menus_left` = 18
- int `ui_menus_right` = `SW`-270

### 4.4.1 Detailed Description

The `ui_controller` class.

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

### 4.4.2 Constructor & Destructor Documentation

#### 4.4.2.1 `__init__()`

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

Initializes `ui_controller`.

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

### 4.4.3 Member Function Documentation

#### 4.4.3.1 `draw_ui_dynamic()`

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

Draws the dynamic ui.

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

#### 4.4.3.2 `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.

#### 4.4.3.3 `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.

#### 4.4.3.4 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.

### 4.4.4 Member Data Documentation

#### 4.4.4.1 resolutions\_list

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

**Initial value:**

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

Possible canvas export resolutions.

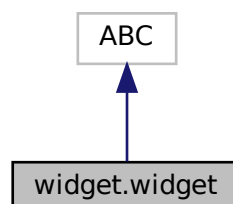
The documentation for this class was generated from the following file:

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

## 4.5 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.*

### 4.5.1 Detailed Description

An abstract class for widgets to extend.

Provides an interface widgets typically use.

### 4.5.2 Constructor & Destructor Documentation

#### 4.5.2.1 `__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.



### 4.5.3 Member Function Documentation

#### 4.5.3.1 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.

#### 4.5.3.2 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.

#### 4.5.3.3 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](#)

## 4.6 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.*

### 4.6.1 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 color\_palette.py File Reference

Defines the color\_palette class.

## Classes

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

### 5.3.1 Detailed Description

Defines the color\_palette class.

### 5.3.2 Author(s)

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

## 5.4 help.py File Reference

Defines the help class.

## Classes

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

### 5.4.1 Detailed Description

Defines the help class.

### 5.4.2 Author(s)

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

## 5.5 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()`

### 5.5.1 Detailed Description

Defines and initializes the ui\_controller class.

### 5.5.2 Author(s)

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

## 5.6 widget.py File Reference

Defines the widget abstract class.

### Classes

- class `widget.widget`  
*An abstract class for widgets to extend.*

### 5.6.1 Detailed Description

Defines the widget abstract class.

### 5.6.2 Author(s)

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

## 5.7 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.*

### 5.7.1 Detailed Description

Defines and initializes the widget\_storage class.

### 5.7.2 Author(s)

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

### 5.7.3 Variable Documentation

#### 5.7.3.1 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`, 10
  - `help.help`, 13
  - `ui_controller.ui_controller`, 16
  - `widget.widget`, 18
- `assets.py`, 21
  - `text_to_screen`, 22
- `canvas.canvas`, 7
  - `__init__`, 7
  - `draw_to_canvas`, 8
  - `generate_bg`, 8
  - `get_canvas`, 9
- `canvas.py`, 22
- `color_palette.color_palette`, 9
  - `__init__`, 10
  - `draw_ui_dynamic`, 11
  - `draw_ui_static`, 11
  - `events`, 11
  - `get_background_color`, 11
  - `get_colors_from_palette`, 11
  - `get_foreground_colors`, 12
  - `get_name_of_palette`, 12
  - `refresh_ui_static`, 12
- `color_palette.py`, 23
- `draw_to_canvas`
  - `canvas.canvas`, 8
- `draw_ui_dynamic`
  - `color_palette.color_palette`, 11
  - `help.help`, 14
  - `ui_controller.ui_controller`, 16
  - `widget.widget`, 19
- `draw_ui_static`
  - `color_palette.color_palette`, 11
  - `help.help`, 14
  - `ui_controller.ui_controller`, 16
  - `widget.widget`, 19
- `events`
  - `color_palette.color_palette`, 11
  - `help.help`, 14
  - `widget.widget`, 19
- `generate_bg`
  - `canvas.canvas`, 8
- `get_background_color`
  - `color_palette.color_palette`, 11
- `get_canvas`
  - `canvas.canvas`, 9
- `get_colors_from_palette`
  - `color_palette.color_palette`, 11
- `get_foreground_colors`
  - `color_palette.color_palette`, 12
- `get_name_of_palette`
  - `color_palette.color_palette`, 12
- `help.help`, 13
  - `__init__`, 13
  - `draw_ui_dynamic`, 14
  - `draw_ui_static`, 14
  - `events`, 14
- `help.py`, 23
- `process_events`
  - `ui_controller.ui_controller`, 16
- `refresh_ui_static`
  - `color_palette.color_palette`, 12
- `resolutions_list`
  - `ui_controller.ui_controller`, 17
- `run`
  - `ui_controller.ui_controller`, 16
- `text_to_screen`
  - `assets.py`, 22
- `ui_controller.py`, 24
- `ui_controller.ui_controller`, 15
  - `__init__`, 16
  - `draw_ui_dynamic`, 16
  - `draw_ui_static`, 16
  - `process_events`, 16
  - `resolutions_list`, 17
  - `run`, 16
- `widget.py`, 24
- `widget.widget`, 17
  - `__init__`, 18
  - `draw_ui_dynamic`, 19
  - `draw_ui_static`, 19
  - `events`, 19
- `widget_storage.py`, 25
  - `widgets`, 25
- `widget_storage.widget_storage`, 19
- `widgets`
  - `widget_storage.py`, 25