tamcolors

Release 2.0.0

Mar 01, 2021

Contents:

1 about **3**

2 pip install tamcolors **5**

3 links **7**

4 table tennis example **9**

5 basic example **11**

6 icon example **13**
6.1 tamcolors 13

7 Indices and tables **71**

Python Module Index **73**

Index **75**


TTTTTT AAAA MM MM CCCC OOOOOO LL OOOOOO RRRRRR SSSS
TTTTTT AAAAAA MMM MMM CCCC OOOOOO LL OOOOOO RRRRRR SSSS
TT AA AA MMMMMMM CC OO OO LL OO OO RR RR SS
TT AA AA MM M MM CC OO OO LL OO OO RR RR SS
TT AAAAAA MM MM CC OO OO LL OO OO RRRRRR SSSS
TT AAAAAA MM MM CC OO OO LL OO OO RRRRR SSSS
TT AA AA MM MM CC OO OO LL OO OO RRRR SS
TT AA AA MM MM CC OO OO LL OO OO RR RR SS
TT AA AA MM MM CCCC OOOOOO LLLL OOOOOO RR RR SSSS
TT AA AA MM MM CCCC OOOOOO LLLL OOOOOO RR R SSSS

CHAPTER 1

about

tamcolors is a terminal game library which supports multiplayer and audio. tamcolors gives a buffer which lets the user set the character, foreground color and background color which can draw at a stable FPS of 25 on all supported console.

CHAPTER 2

pip install tamcolors

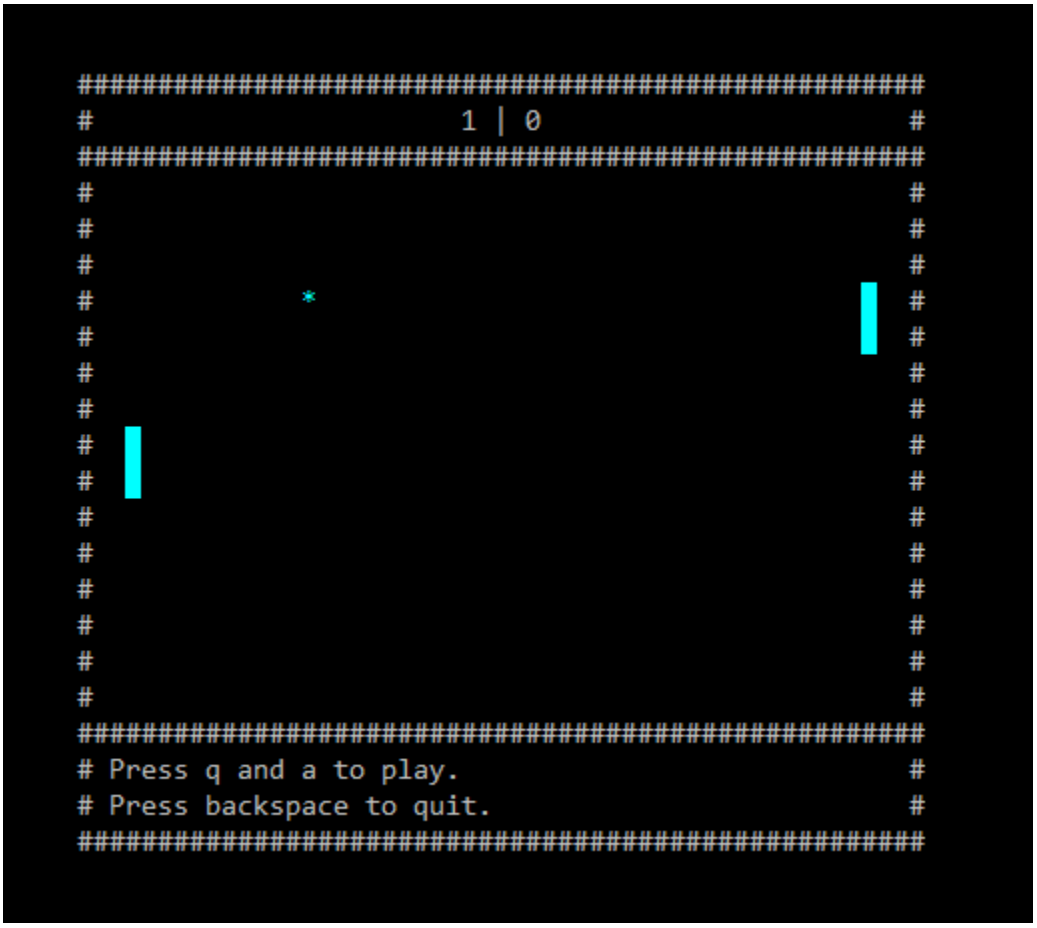
CHAPTER 3

links

- github
- pypi
- read the docs
- youtube
- patreon
- facebook

CHAPTER 4

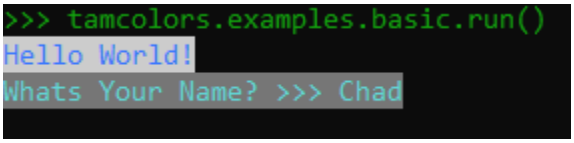
table tennis example




```
import tamcolors
tamcolors.examples.tabletennis.run()
```

CHAPTER 5

basic example




```
import tamcolors
tamcolors.examples.basic_console.run()
```

```
from tamcolors.tam_basic import console
from tamcolors.tam_io.tam_colors import *
console.printc("Hello", "World!", ("light blue", "white"), same_color=True)
name = console.inputc("Whats Your Name? >>> ", ("light aqua", "gray"))
console.clear()
console.printc("Hello, ", ("default", "default"), name, (GREEN, WHITE), "!", ("gray",
↪"light aqua"), sep="")
```

CHAPTER 6

icon example

```
import tamcolors
tamcolors.examples.icon.run()
```

6.1 tamcolors

6.1.1 tamcolors package

Subpackages

tamcolors.examples package

Submodules

tamcolors.examples.alpha module

class tamcolors.examples.alpha.**TAMAlpha**

Bases: *tamcolors.tam.tam_loop.TAMFrame*

draw(*tam_surface*, *loop_data*, **args*)

info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

update(*tam_loop*, *keys*, *loop_data*, **args*)

info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

tamcolors.examples.alpha.**run**()

tamcolors.examples.basic module

tamcolors.examples.clouds module

class tamcolors.examples.clouds.**Clouds**
Bases: *tamcolors.tam.tam_loop.TAMFrame*

draw(*tam_surface*, *loop_data*, **args*)
info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

update(*tam_loop*, *keys*, *loop_data*, **args*)
info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

tamcolors.examples.clouds.**run**()

tamcolors.examples.colors module

class tamcolors.examples.colors.**TAMCOLORS**
Bases: *tamcolors.tam.tam_loop.TAMFrame*

draw(*tam_surface*, *loop_data*, **args*)
info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

update(*tam_loop*, *keys*, *loop_data*, **args*)
info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

tamcolors.examples.colors.**run**()

tamcolors.examples.connection_loopback module

tamcolors.examples.connection_loopback.**run**()

tamcolors.examples.connection_multi_player module

tamcolors.examples.connection_multi_player.**run**()

tamcolors.examples.host_loopback module

tamcolors.examples.host_loopback.**run**()

tamcolors.examples.host_multi_player module

class tamcolors.examples.host_multi_player.**HostMultiPlayer**
Bases: *tamcolors.tam.tam_loop.TAMFrame*

draw(*tam_surface*, *loop_data*, *other_surfaces*, *other_data*)
info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

update(*tam_loop*, *keys*, *loop_data*, *other_handlers*, *other_keys*, *other_data*)
info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

tamcolors.examples.host_multi_player.**run**()

tamcolors.examples.icon module

class tamcolors.examples.icon.**BootLogo**
Bases: *tamcolors.tam.tam_loop.TAMFrame*

draw(*tam_surface*, *loop_data*, **args*)
info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

update(*tam_loop*, *keys*, *loop_data*, **args*)
info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

tamcolors.examples.icon.**run**()

tamcolors.examples.rgb_color module

class tamcolors.examples.rgb_color.**RGBCOLOR**
Bases: *tamcolors.tam.tam_loop.TAMFrame*

draw(*tam_surface*, *loop_data*, **args*)
info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

update(*tam_loop*, *keys*, *loop_data*, **args*)
info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

tamcolors.examples.rgb_color.**run**()

tamcolors.examples.sandy_check module

tamcolors.examples.sandy_check.**run**()

tamcolors.examples.tabletennis module

class tamcolors.examples.tabletennis.**Ball**(*tam_surface*, *way=False*)
Bases: object

get_y()

update()

winner()

class tamcolors.examples.tabletennis.**Racket**(*x*, *ball*, *tam_surface*, *ai=False*)
Bases: object

update(*key_manager*)

class tamcolors.examples.tabletennis.**TableTennis**
Bases: *tamcolors.tam.tam_loop.TAMFrame*

draw(*tam_surface*, *loop_data*, **args*)
info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

update(*tam_loop*, *keys*, *loop_data*, **args*)
info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

tamcolors.examples.tabletennis.**run**()

tamcolors.examples.tam_key_manager module

class tamcolors.examples.tam_key_manager.**TAMKeyManager**
Bases: *tamcolors.tam.tam_loop.TAMFrame*

draw(*tam_surface*, *loop_data*, **args*)
info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

update(*tam_loop*, *keys*, *loop_data*, **args*)
info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

tamcolors.examples.tam_key_manager.**run**()

tamcolors.examples.tam_keys module

class tamcolors.examples.tam_keys.**TAMKeys**
Bases: *tamcolors.tam.tam_loop.TAMFrame*

draw(*tam_surface*, *loop_data*, **args*)
info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

update(*tam_loop*, *keys*, *loop_data*, **args*)
info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

tamcolors.examples.tam_keys.**run**()

tamcolors.examples.tam_list_surface module

class tamcolors.examples.tam_list_surface.**TAMListSurface**
Bases: *tamcolors.tam.tam_loop.TAMFrame*

draw(*tam_surface*, *loop_data*, **args*)
info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

update(*tam_loop*, *keys*, *loop_data*, **args*)
info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

tamcolors.examples.tam_list_surface.**run**()

tamcolors.examples.tam_loop module

class tamcolors.examples.tam_loop.**TAMLoopHelloWorld**
Bases: *tamcolors.tam.tam_loop.TAMFrame*

draw(*tam_surface*, *loop_data*, **args*)
info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

update(*tam_loop*, *keys*, *loop_data*, **args*)
info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

tamcolors.examples.tam_loop.**run**()

tamcolors.examples.tam_print module

class tamcolors.examples.tam_print.**TAMPrint**
Bases: *tamcolors.tam.tam_loop.TAMFrame*

draw(*tam_surface*, *loop_data*, **args*)
info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

update(*tam_loop*, *keys*, *loop_data*, **args*)
info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

tamcolors.examples.tam_print.**run**()

tamcolors.examples.tam_text_box module

class tamcolors.examples.tam_text_box.**TAMPrint**
Bases: *tamcolors.tam.tam_loop.TAMFrame*

draw(*tam_surface*, *loop_data*, **args*)
info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

update(*tam_loop*, *keys*, *loop_data*, **args*)
info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

tamcolors.examples.tam_text_box.**run**()

Module contents

tamcolors.tam package

Submodules

tamcolors.tam.tam_loop module

class tamcolors.tam.tam_loop.**TAMFrame**(*fps*, *char*, *foreground_color*, *background_color*, *min_width=0*, *max_width=1000*, *min_height=0*, *max_height=1000*)

Bases: object

done(*tam_loop*, *loop_data*, *other_handlers*, *other_data*)
info: called when TAMLoop is terminating and can only be called once :param tam_loop: TAMLoop :param loop_data: dict :param other_handlers: dict :param other_data: dict :return:

draw(*tam_surface*, *loop_data*, *other_surfaces*, *other_data*)
info: will draw frame onto terminal :param tam_surface: TAMSurface :param loop_data: object :param other_surfaces: dict :param other_data: dict :return:

frame_done(*tam_loop*, *loop_data*, *other_handlers*, *other_data*)
info: called when clean up the frame and can only be called once :param tam_loop: TAMLoop :param loop_data: dict :param other_handlers: dict :param other_data: dict :return:

get_defaults()
info: gets defaults :return: (str, int, int)

get_fps()
info: returns the frame fps :return: int

get_height_min_and_max()
info: returns min and max height :return: (int, int)

get_width_min_and_max()
info: returns min and max width :return: (int, int)

make_surface_ready(*tam_surface*, *screen_width*, *screen_height*)
info: will make surface ready for frame :param tam_surface: TAMSurface :param screen_width: int: 0 - inf :param screen_height: int: 0 - inf :return:

update(*tam_loop*, *keys*, *loop_data*, *other_handlers*, *other_keys*, *other_data*)
info: will update terminal :param tam_loop: TAMLoop :param keys: list, tuple :param loop_data: objects :param other_handlers: dict :param other_keys: dict :param other_data: dict :return:

class tamcolors.tam.tam_loop.**TAMLoop**(*tam_frame*, *io=None*, *only_any_os=False*, *color_change_key=('ESCAPE', 'SPECIAL')*, *loop_data=None*, *stability_check=False*, *tam_color_defaults=True*, *highest_mode_lock=False*, *preferred_mode=None*, *name=None*, *identifier_id=None*, *start_data=None*, *receivers=None*, *other_handlers=None*, *thread_count=20*, *enable_loop_log=False*, *loop_log_key=('F1', 'SPECIAL')*, *loop_log_level=10*, *enable_loop_fps=True*, *loop_fps_key=('F2', 'SPECIAL')*)

Bases: *tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler*

add_frame_stack(*frame*)
info: will add a TAMFrame to stack :param frame: TAMFrame :return:

add_receiver(*receiver*)

done()
info: will stop tam loop :return: None

get_all_receiver_names()

get_receiver_settings()
info: gets the receiver settings :return: dict

pop_frame_stack()
info: will remove TAMFrame from stack :return: TAMFrame or None

remove_receiver(*receiver_name*)

classmethod run_application(**args*, ***kwargs*)
info: will run tam loop as an application note: when tam loop is done running the program will quit if tam loop has an error and the frame does not catch it the error will be printed onto the screen and the program will quit after user input :param args: :param kwargs: :return:

thread_task(*func*, **args*, ***kwargs*)

exception tamcolors.tam.tam_loop.**TAMLoopError**
Bases: Exception

tamcolors.tam.tam_loop_io_handler module

class tamcolors.tam.tam_loop_io_handler.**TAMLoopIOHandler**(*io*, *name=None*, *identifier_id=None*, *color_change_key=('ESCAPE', 'SPECIAL')*, *start_data=None*, *loop_data=None*, *tam_color_defaults=True*, *highest_mode_lock=False*, *preferred_mode=None*, *reset_io=True*)
Bases: object

done()
info: will stop tam loop :return: None

enable_key_state_mode(*enable=True*)
info: Will enable or disable key state mode :param enable: bool :return: None

freeze_handler()
info: will freeze event loop :return:

get_color_16(*spot*)
info: Will get color from color palette 16 :param spot: int :return: RGBA

get_color_16_pal_256(*spot*)
info: Will get color from color palette 2 :param spot: int :return: RGBA

get_color_2(*spot*)
info: Will get color from color palette 2 :param spot: int :return: RGBA

get_color_256(*spot*)
info: Will get color from color palette 256 :param spot: int :return: RGBA

get_dimensions()
info: Will get the dimensions of the terminal :return: tuple

get_full_name()
info: get full name :return: tuple

get_identifier_id()
info: get identifier id :return: bytes or None

get_io()
infl: will get the IO :return: IO

get_keyboard_name(*default_to_us_english=True*)
info: Will get the keyboard language name :param default_to_us_english: bool :return: str

get_loop_data()
info: get loop data :return: object

get_name()
info: get name :return: str or None

get_start_data()
info: get start data :return: object

is_key_state_mode_enabled()
info: Will get the status of key_state :return: bool

is_running()
info: None has not ran, True is running, False has ran :return: bool or None

pump_keys()
info: will get keys from the key queue :return: list

reset_colors_to_console_defaults()
info: will reset colors to console defaults :return: None

run()
info: will call tam loop :return: None

run_with_profiler()
info: will run with a profiler and print out data when done :return: None

set_color_16(*spot*, *color*)
info: sets a color value :param spot: int :param color: RGBA :return: None

set_color_16_pal_256(*spot*, *color*)
info: sets a color value :param spot: int :param color: int :return: None

set_color_2(*spot*, *color*)
info: sets a color value :param spot: int :param color: RGBA :return: None

set_color_256(*spot*, *color*)
info: sets a color value :param spot: int :param color: RGBA :return: None

set_loop_data(*loop_data*)
info: set loop data :param loop_data: object :return: None

set_start_data(*start_data*)
info: set start data :param start_data: object :return: None

set_tam_color_defaults()
info: will set console colors to tam defaults :return: None

unfreeze_handler()
info: will unfreeze event loop :return:

tamcolors.tam.tam_loop_io_tcp_handler module

class tamcolors.tam.tam_loop_io_tcp_handler.**TAMLoopIOTCPHandler**(*io*, *name=None*, *identifier_id=None*, *color_change_key=('ESCAPE', 'SPECIAL')*, *start_data=None*, *loop_data=None*, *tam_color_defaults=True*, *highest_mode_lock=False*, *preferred_mode=None*, *reset_io=True*)

Bases: *tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler*

done()
info: will stop tam loop :return: None

is_running()
info: None has not ran, True is running, False has ran :return: bool or None

tamcolors.tam.tam_loop_receiver module

class tamcolors.tam.tam_loop_receiver.**TAMLoopReceiver**(*name*)

Bases: abc.ABC

done()
info: Will stop the receiver :return: None

get_handler()
info: Will get an io if available :return: TAMLoopIOHandler or None

get_name()
info: Will get the receiver name :return: str

get_receiver_settings()
info: will get the receiver io settings :return: dict or None

get_running()
info: Checks if receiver is running :return: bool

set_receiver_settings(*settings*)
info: will set the receiver io settings :param settings: dict :return: None

tamcolors.tam.tam_loop_tcp_receiver module

class tamcolors.tam.tam_loop_tcp_receiver.**TAMLoopTCPReceiver**(*host='127.0.0.1'*, *port=4444*, *ipv6=False*, *listen_count=10*, *connection_password=''*, *address_white_list=None*, *address_black_list=None*, *encryption=None*, *object_packer=None*, *our_information=None*)

Bases: *tamcolors.tam.tam_loop_receiver.TAMLoopReceiver*

done()
info: Will stop the receiver :return: None

get_handler()
info: Will get an io if available :return: TAMLoopIOHandler or None

Module contents

tamcolors.tam_basic package

Submodules

tamcolors.tam_basic.basic module

Module contents

tamcolors.tam_c package

Module contents

tamcolors.tam_io package

Submodules

tamcolors.tam_io.ansi_256_drivers module

class tamcolors.tam_io.ansi_256_drivers.**ANSI256ChangerDriver**(**args*, ***kwargs*)
Bases: *tamcolors.tam_io.tam_drivers.ColorChangerDriver*, abc.ABC

class tamcolors.tam_io.ansi_256_drivers.**ANSI256ColorDriver**(**args*, ***kwargs*)
Bases: *tamcolors.tam_io.tam_drivers.FullColorDriver*, abc.ABC

draw(*tam_surface*)
info: Will draw TAMSurface to console :param tam_surface: TAMSurface :return: None

get_printc_mode()
Gets the modes used by printc and inputc :return: str

inputc(*output*, *color*)
info: Will get input from the console in color :param output: str :param color: COLOR :return: str

printc(*output*, *color*, *flush*, *stderr*)
info: Will print to the console in color :param output: str :param color: COLOR :param flush: bool :param stderr: std :return:

set_mode(*mode*)
info: will set the color mode :param mode: int: key to color mode :return:

start()
info: operations for IO to start :return: None

tamcolors.tam_io.ansi_true_color_drivers module

class tamcolors.tam_io.ansi_true_color_drivers.**ANSITrueFullColorDriver**(**args*, ***kwargs*)
Bases: *tamcolors.tam_io.tam_drivers.FullColorDriver*, abc.ABC

draw(*tam_surface*)
info: Will draw TAMSurface to console :param tam_surface: TAMSurface :return: None

get_printc_mode()
Gets the modes used by printc and inputc :return: str

inputc(*output*, *color*)
info: Will get input from the console in color :param output: str :param color: COLOR :return: str

printc(*output*, *color*, *flush*, *stderr*)
info: Will print to the console in color :param output: str :param color: COLOR :param flush: bool :param stderr: std :return:

set_mode(*mode*)
info: will set the color mode :param mode: int: key to color mode :return:

start()
info: operations for IO to start :return: None

tamcolors.tam_io.any_drivers module

class tamcolors.tam_io.any_drivers.**ANYFullColorDriver**(**args*, ***kwargs*)
Bases: *tamcolors.tam_io.tam_drivers.FullColorDriver*, abc.ABC

draw(*tam_surface*)
info: Will draw TAMSurface to console :param tam_surface: TAMSurface :return: None

get_printc_mode()
Gets the modes used by printc and inputc :return: str

inputc(*value*, *color*)
info: will get user input with color :param value: str :param color: tuple: (int, int) :return: str

printc(*output*, *color*, *flush*, *stderr*)
info: will print out user output with color :param output: str :param color: tuple: (int, int) :param flush: boolean :param stderr: boolean :return: None

class tamcolors.tam_io.any_drivers.**ANYKeyDriver**(**args*, ***kwargs*)
Bases: *tamcolors.tam_io.tam_drivers.KeyDriver*, abc.ABC

get_key()
info: Gets an input from the terminal :return: tuple or false

get_key_dict(*language=None*)
info: Gets a dict of all the keys :param language: str or None :return: dict

get_keyboard_name(*default_to_us_english=True*)
info: Will get the keyboard language name :param default_to_us_english: bool :return: str

wait_key(*rest_time=0.0001*, *attempts=300000*)
info: Get an input from the terminal :param: rest_time: float: rest time from checking if a key is down :param: attempts: int: number of attempts to get a key :return: tuple or false

class tamcolors.tam_io.any_drivers.**ANYSoundDriver**(**args*, ***kwargs*)
Bases: *tamcolors.tam_io.tam_drivers.SoundDriver*, abc.ABC

close_sound(*sound_id*)
info: will close sound :param sound_id: int :return: None

get_sound_length(*sound_id*)
info: will get sound lenght :param sound_id: int :return: int

get_sound_position(*sound_id*)
info: will get the time spot of the song :param sound_id: int :return: int

is_sound_playing(*sound_id*)
info: will check if sound is playing :param sound_id: int :return: bool

open_sound(*file*, *sound_id*)
info: will open .wav sound :param file: str :param sound_id: int :return: None

pause_sound(*sound_id*)
info: will pause sound :param sound_id: int :return: None

play_sound(*sound_id*, *reset_sound=True*)
info: will play sound :param sound_id: int :param reset_sound: bool :return: None

set_sound_position(*sound_id*, *spot*)
info: will set the spot of the sound :param sound_id: int :param spot: int :return: None

class tamcolors.tam_io.any_drivers.**ANYUtilitiesDriver**(**args*, ***kwargs*)
Bases: *tamcolors.tam_io.tam_drivers.UtilitiesDriver*, abc.ABC

clear()
info: Will clear the console :return: None

get_dimensions()
info: Gets the dimensions of console :return: (int, int): (row, column)

show_console_cursor(*show*)
info: Will show or hide console cursor :param show: int :return: None

tamcolors.tam_io.io_tam module

class tamcolors.tam_io.io_tam.**IO**(*identifier*, *mode_2=True*, *mode_16_pal_256=True*, *mode_16=True*, *mode_256=True*, *mode_rgb=True*, *key_driver_operational=True*, *color_driver_operational=True*, *color_changer_driver_operational=True*, *utilities_driver_operational=True*, *sound_driver_operational=True*)

Bases: *tamcolors.tam_io.io_tam.RawIO*, abc.ABC

classmethod able_to_execute()
info: checks that io is stable in current environment :return: bool

apply_snapshot(*snapshot*)
info: apply snapshot state to IO :param snapshot: dict :return: None

clear()
info: Will clear the console :return: None

close_sound(*sound_id*)
info: will close sound :param sound_id: int :return: None

color_changer_driver_operational()
info: checks if the color changer driver is operational :return: bool

color_driver_operational()
info: checks if the color driver is operational :return: bool

done()
info: operations for IO to stop :return: None

draw(*tam_surface*)
info: Will draw TAMSurface to console :param tam_surface: TAMSurface :return: None

enable_console_keys(*enable*)
info: will enable console keys :param enable: boool :return: None

enable_event_bus(*bus=True*)
info: will enable the event bus :param bus: bool :return: None

enable_key_state_mode(*enable=True*)
info: Will enable or disable key state mode :param enable: bool :return: None

get_color_16(*spot*)
info: Will get color from color palette 16 :param spot: int :return: RGBA

get_color_16_pal_256(*spot*)
info: Will get color from color palette 2 :param spot: int :return: int

get_color_2(*spot*)
info: Will get color from color palette 2 :param spot: int :return: RGBA

get_color_256(*spot*)
info: Will get color from color palette 256 :param spot: int :return: RGBA

get_dimensions()
info: Gets the dimensions of console :return: (int, int): (row, column)

get_event()
info: will get event :yield: tuple

get_info_dict()
info: will get the identifier dict :return: dict

get_key()
info: Gets an input from the terminal :return: tuple or false

get_key_dict(*language=None*)
info: Gets a dict of all the keys :param language: str or None :return: dict

get_keyboard_name(*default_to_us_english=True*)
info: Will get the keyboard language name :param default_to_us_english: bool :return: str

get_mode()
info: will return the current color mode :return: int

get_modes()
info: will return a tuple of all color modes :return: (int, int, ...)

get_printc_mode()
Gets the modes used by printc and inputc :return: str

get_snapshot()
info: get snapshot of IO :return: dict

get_sound_length(*sound_id*)
info: will get sound length :param sound_id: int :return: int

get_sound_position(*sound_id*)
info: will get the time spot of the song :param sound_id: int :return: int

inputc(*output*, *color*)
info: Will get input from the console in color :param output: str :param color: COLOR :return: str

is_console_cursor_enabled()
info: will check if console cursor is enabled :return: bool

is_console_keys_enabled()
info: will check if console keys enabled :return: bool

is_event_bus_enabled()
info: will check if event bus is enabled :return: bool

is_key_state_mode_enabled()
info: Will get the status of key_state :return: bool

is_running()
info: checks if IO has been started :return: bool

is_sound_playing(*sound_id*)
info: will check if sound is playing :param sound_id: int :return: bool

key_driver_operational()
info: checks if the key driver is operational :return: bool

open_sound(*file*, *sound_id*)
info: will open .wav sound :param file: str :param sound_id: int :return: None

pause_sound(*sound_id*)
info: will pause sound :param sound_id: int :return: None

play_sound(*sound_id*, *reset_sound=True*)
info: will play sound :param sound_id: int :param reset_sound: bool :return: None

prime_event_bus()
info: will repeat the last event for every type other than keys :return: None

printc(*output*, *color*, *flush*, *stderr*)
info: Will print to the console in color :param output: str :param color: COLOR :param flush: bool :param stderr: std :return:

reset_colors_to_console_defaults()
info: will reset colors to console defaults :return: None

rest_sound(*sound_id*)
info: will reset sound :param sound_id: int :return: None

set_color_16(*spot*, *color*)
info: sets a color value :param spot: int :param color: RGBA :return: None

set_color_16_pal_256(*spot*, *color*)
info: sets a color value :param spot: int :param color: int :return: None

set_color_2(*spot*, *color*)
info: sets a color value :param spot: int :param color: RGBA :return: None

set_color_256(*spot*, *color*)
info: sets a color value :param spot: int :param color: RGBA :return: None

set_mode(*mode*)
info: will set the color mode :param mode: int: key to color mode :return: None

set_sound_position(*sound_id*, *spot*)
info: will set the spot of the sound :param sound_id: int :param spot: int :return: None

set_tam_color_defaults()
info: will set console colors to tam defaults :return: None

show_console_cursor(*show*)
info: Will show or hide console cursor :param show: bool :return: None

sound_driver_operational()
info: checks if the sound driver is operational :return: bool

start()
info: operations for IO to start :return: None

utilities_driver_operational()
info: checks if the utilities driver is operational :return: bool

wait_key(*rest_time=0.0001*, *attempts=None*)
info: Get an input from the terminal :param: rest_time: float: rest time from checking if a key is down :param: attempts: int or None: number of attempts to get a key :return: tuple or false

class tamcolors.tam_io.io_tam.**RawIO**
Bases: abc.ABC

classmethod able_to_execute()
info: checks that io is stable in current environment :return: bool

apply_snapshot(*snapshot*)
info: apply snapshot state to IO :param snapshot: dict :return: None

clear()
info: Will clear the console :return: None

close_sound(*sound_id*)
info: will close sound :param sound_id: int :return: None

color_change_driver_operational()
info: checks if the color changer driver is operational :return: bool

color_changer_driver_operational()
info: checks if the color changer driver is operational :return: bool

color_driver_operational()
info: checks if the color driver is operational :return: bool

done()
info: operations for IO to stop :return: None

draw(*tam_surface*)
info: Will draw TAMSurface to console :param tam_surface: TAMSurface :return: None

enable_console_keys(*enable*)
info: will enable console keys :param enable: boool :return: None

enable_event_bus(*bus=True*)
info: will enable the event bus :param bus: bool :return: None

enable_key_state_mode(*enable=True*)
info: Will enable or disable key state mode :param enable: bool :return: None

get_color_16(*spot*)
info: Will get color from color palette 16 :param spot: int :return: RGBA

get_color_16_pal_256(*spot*)
info: Will get color from color palette 2 :param spot: int :return: int

get_color_2(*spot*)
info: Will get color from color palette 2 :param spot: int :return: RGBA

get_color_256(*spot*)
info: Will get color from color palette 256 :param spot: int :return: RGBA

get_dimensions()
info: Gets the dimensions of console :return: (int, int): (row, column)

get_event()
info: will get event :yield: tuple

get_info_dict()
info: will get the identifier dict :return: dict

get_key()
info: Gets an input from the terminal :return: tuple or false

get_key_dict(*language=None*)
info: Gets a dict of all the keys :param language: str or None :return: dict

get_keyboard_name(*default_to_us_english=True*)
info: Will get the keyboard language name :param default_to_us_english: bool :return: str

get_mode()
info: will return the current color mode :return: int

get_modes()
info: will return a tuple of all color modes :return: (str, str, . . .)

get_printc_mode()
Gets the modes used by printc and inputc :return: str

get_snapshot()
info: get snapshot of IO :return: dict

get_sound_length(*sound_id*)
info: will get sound length :param sound_id: int :return: int

get_sound_position(*sound_id*)
info: will get the time spot of the song :param sound_id: int :return: int

inputc(*output*, *color*)
info: Will get input from the console in color :param output: str :param color: COLOR :return: str

is_console_cursor_enabled()
info: will check if console cursor is enabled :return: bool

is_console_keys_enabled()
info: will check if console keys enabled :return: bool

is_event_bus_enabled()
info: will check if event bus is enabled :return: bool

is_key_state_mode_enabled()
info: Will get the status of key_state :return: bool

is_running()
info: checks if IO has been started :return: bool

is_sound_playing(*sound_id*)
info: will check if sound is playing :param sound_id: int :return: bool

key_driver_operational()
info: checks if the key driver is operational :return: bool

open_sound(*file*, *sound_id*)
info: will open .wav sound :param file: str :param sound_id: int :return: None

pause_sound(*sound_id*)
info: will pause sound :param sound_id: int :return: None

play_sound(*sound_id*, *reset_sound=True*)
info: will play sound :param sound_id: int :param reset_sound: bool :return: None

prime_event_bus()
info: will repeat the last event for every type other than keys :return: None

printc(*output*, *color*, *flush*, *stderr*)
info: Will print to the console in color :param output: str :param color: COLOR :param flush: bool :param stderr: std :return:

reset_colors_to_console_defaults()
info: will reset colors to console defaults :return: None

rest_sound(*sound_id*)
info: will reset sound :param sound_id: int :return: None

set_color_16(*spot*, *color*)
info: sets a color value :param spot: int :param color: RGBA :return: None

set_color_16_pal_256(*spot*, *color*)
info: sets a color value :param spot: int :param color: int :return: None

set_color_2(*spot*, *color*)
info: sets a color value :param spot: int :param color: RGBA :return: None

set_color_256(*spot*, *color*)
info: sets a color value :param spot: int :param color: RGBA :return: None

set_mode(*mode*)
info: will set the color mode :param mode: int: key to color mode :return:

set_sound_position(*sound_id*, *spot*)
info: will set the spot of the sound :param sound_id: int :param spot: int :return: None

set_tam_color_defaults()
info: will set console colors to tam defaults :return: None

show_console_cursor(*show*)
info: Will show or hide console cursor :param show: int :return: None

sound_driver_operational()
info: checks if the sound driver is operational :return: bool

start()
info: operations for IO to start :return: None

utilities_driver_operational()
info: checks if the utilities driver is operational :return: bool

wait_key(*rest_time=0.0001*, *attempts=300000*)
info: Get an input from the terminal :param: rest_time: float: rest time from checking if a key is down :param: attempts: int: number of attempts to get a key :return: tuple or false

exception tamcolors.tam_io.io_tam.**TAMSoundError**
Bases: Exception

tamcolors.tam_io.null_drivers module

class tamcolors.tam_io.null_drivers.**NULLFullColorDriver**(**args*, ***kwargs*)
Bases: *tamcolors.tam_io.tam_drivers.FullColorDriver*, abc.ABC

get_printc_mode()
Gets the modes used by printc and inputc :return: str

inputc(*output*, *color*)
info: Will get input from the console in color :param output: str :param color: COLOR :return: str

printc(*output*, *color*, *flush*, *stderr*)
info: Will print to the console in color :param output: str :param color: COLOR :param flush: bool :param stderr: std :return:

class tamcolors.tam_io.null_drivers.**NULLKeyDriver**(**args*, ***kwargs*)
Bases: *tamcolors.tam_io.tam_drivers.KeyDriver*, abc.ABC

get_key()
info: Gets an input from the terminal :return: tuple or false

classmethod get_key_dict(*language=None*)
info: Gets a dict of all the keys :param language: str or None :return: dict

get_keyboard_name(*default_to_us_english=True*)
info: Will get the keyboard language name :param default_to_us_english: bool :return: str

wait_key(*rest_time=0.0001*, *attempts=300000*)
info: Get an input from the terminal :param: rest_time: float: rest time from checking if a key is down :param: attempts: int: number of attempts to get a key :return: tuple or false

class tamcolors.tam_io.null_drivers.**NULLSoundDriver**(**args*, ***kwargs*)
Bases: *tamcolors.tam_io.tam_drivers.SoundDriver*, abc.ABC

close_sound(*sound_id*)
info: will close sound :param sound_id: int :return: None

get_sound_length(*sound_id*)
info: will get sound lenght :param sound_id: int :return: int

get_sound_position(*sound_id*)
info: will get the time spot of the song :param sound_id: int :return: int

is_sound_playing(*sound_id*)
info: will check if sound is playing :param sound_id: int :return: bool

open_sound(*file*, *sound_id*)
info: will open .wav sound :param file: str :param sound_id: int :return: None

pause_sound(*sound_id*)
info: will pause sound :param sound_id: int :return: None

play_sound(*sound_id*, *reset_sound=True*)
info: will play sound :param sound_id: int :param reset_sound: bool :return: None

set_sound_position(*sound_id*, *spot*)
info: will set the spot of the sound :param sound_id: int :param spot: int :return: None

class tamcolors.tam_io.null_drivers.**NULLUtilitiesDriver**(**args*, ***kwargs*)
Bases: *tamcolors.tam_io.tam_drivers.UtilitiesDriver*, abc.ABC

get_dimensions()
info: Gets the dimensions of console :return: (int, int): (row, column)

tamcolors.tam_io.tam_colors module

class tamcolors.tam_io.tam_colors.**Color**(*mode_16*, *mode_256*, *mode_rgb*, *mode_16_pal_256=None*, *mode_2=None*, *_color_id=None*)
Bases: *tamcolors.utils.immutable_cache.ImmutableCache*, *tamcolors.utils.object_packer.FastHandObjectPacker*

classmethod from_bytes(*object_byte_array*)
info: from bytes to object :param object_byte_array: bytearray :return: object

has_alpha
info: Checks if color has any alpha :return: bool

mode_16
info: Gets mode 16 :return: int

mode_16_pal_256
info: Gets mode 16 pal 256 :return: int

mode_2
info: Gets mode 2 :return: int

mode_256
info: Gets mode 256 :return: int

mode_rgb
info: Gets mode rgb :return: RGBA

place_color_over(*old_color*, *override_alpha*)
info: Will calculate what the new color will be :param old_color: Color :param override_alpha: bool :return: color

to_bytes()
info: object to bytes :return: bytes

transparent_value

class tamcolors.tam_io.tam_colors.**RGBA**(*r*, *g*, *b*, *a=255*, *is_default=False*)
Bases: *tamcolors.utils.immutable_cache.ImmutableCache*, *tamcolors.utils.object_packer.FastHandObjectPacker*

a
info: Will get the a value :return: int

b
info: Will get the b value :return: int

classmethod from_bytes(*object_byte_array*)
info: from bytes to object :param object_byte_array: bytearray :return: object

g
info: Will get the g value :return: int

is_default
info: See if color is default :return: bool

r
info: Will get the r value :return: int

to_bytes()
info: object to bytes :return: bytes

tamcolors.tam_io.tam_drivers module

class tamcolors.tam_io.tam_drivers.**ColorChangerDriver**(*identifier*, *mode_2=True*, *mode_16_pal_256=True*, *mode_16=True*, *mode_256=True*, *mode_rgb=True*, *key_driver_operational=True*, *color_driver_operational=True*, *color_changer_driver_operational=True*, *utilities_driver_operational=True*, *sound_driver_operational=True*)
Bases: *tamcolors.tam_io.tam_drivers.TAMDriver*, abc.ABC

get_color_16(*spot*)
info: Will get color from color palette 16 :param spot: int :return: RGBA

get_color_16_pal_256(*spot*)
info: Will get color from color palette 16 :param spot: int :return: int

get_color_2(*spot*)
info: Will get color from color palette 2 :param spot: int :return: RGBA

get_color_256(*spot*)
info: Will get color from color palette 256 :param spot: int :return: RGBA

set_color_16(*spot*, *color*)
info: sets a color value :param spot: int :param color: RGBA :return: None

set_color_16_pal_256(*spot*, *color*)
info: sets a color value :param spot: int :param color: int :return: None

set_color_2(*spot*, *color*)
info: sets a color value :param spot: int :param color: RGBA :return: None

set_color_256(*spot*, *color*)
info: sets a color value :param spot: int :param color: RGBA :return: None

set_mode(*mode*)
info: will set the color mode :param mode: int: key to color mode :return:

class tamcolors.tam_io.tam_drivers.**ColorDriver**(*identifier*, *mode_2=True*, *mode_16_pal_256=True*, *mode_16=True*, *mode_256=True*, *mode_rgb=True*, *key_driver_operational=True*, *color_driver_operational=True*, *color_changer_driver_operational=True*, *utilities_driver_operational=True*, *sound_driver_operational=True*)
Bases: *tamcolors.tam_io.tam_drivers.TAMDriver*, abc.ABC

draw(*tam_surface*)
info: Will draw TAMSurface to console :param tam_surface: TAMSurface :return: None

get_printc_mode()
Gets the modes used by printc and inputc :return: str

inputc(*output*, *color*)
info: Will get input from the console in color :param output: str :param color: COLOR :return: str

printc(*output*, *color*, *flush*, *stderr*)
info: Will print to the console in color :param output: str :param color: COLOR :param flush: bool :param stderr: std :return:

set_mode(*mode*)
info: will set the color mode :param mode: int: key to color mode :return:

class tamcolors.tam_io.tam_drivers.**FullColorDriver**(*identifier*, *mode_2=True*, *mode_16_pal_256=True*, *mode_16=True*, *mode_256=True*, *mode_rgb=True*, *key_driver_operational=True*, *color_driver_operational=True*, *color_changer_driver_operational=True*, *utilities_driver_operational=True*, *sound_driver_operational=True*)
Bases: *tamcolors.tam_io.tam_drivers.ColorDriver*, *tamcolors.tam_io.tam_drivers.ColorChangerDriver*, abc.ABC

class tamcolors.tam_io.tam_drivers.**KeyDriver**(*identifier*, *mode_2=True*, *mode_16_pal_256=True*, *mode_16=True*, *mode_256=True*, *mode_rgb=True*, *key_driver_operational=True*, *color_driver_operational=True*, *color_changer_driver_operational=True*, *utilities_driver_operational=True*, *sound_driver_operational=True*)

Bases: *tamcolors.tam_io.tam_drivers.TAMDriver*, abc.ABC

enable_console_keys(*enable*)
info: will enable console keys :param enable: boool :return: None

get_key()
info: Gets an input from the terminal :return: tuple or false

get_key_dict(*language=None*)
info: Gets a dict of all the keys :param language: str or None :return: dict

get_keyboard_name(*default_to_us_english=True*)
info: Will get the keyboard language name :param default_to_us_english: bool :return: str

wait_key(*rest_time=0.0001*, *attempts=300000*)
info: Get an input from the terminal :param: rest_time: float: rest time from checking if a key is down :param: attempts: int: number of attempts to get a key :return: tuple or false

class tamcolors.tam_io.tam_drivers.**SoundDriver**(*identifier*, *mode_2=True*, *mode_16_pal_256=True*, *mode_16=True*, *mode_256=True*, *mode_rgb=True*, *key_driver_operational=True*, *color_driver_operational=True*, *color_changer_driver_operational=True*, *utilities_driver_operational=True*, *sound_driver_operational=True*)

Bases: *tamcolors.tam_io.tam_drivers.TAMDriver*, abc.ABC

close_sound(*sound_id*)
info: will close sound :param sound_id: int :return: None

get_sound_length(*sound_id*)
info: will get sound lenght :param sound_id: int :return: int

get_sound_position(*sound_id*)
info: will get the time spot of the song :param sound_id: int :return: int

is_sound_playing(*sound_id*)
info: will check if sound is playing :param sound_id: int :return: bool

open_sound(*file*, *sound_id*)
info: will open .wav sound :param file: str :param sound_id: int :return: None

pause_sound(*sound_id*)
info: will pause sound :param sound_id: int :return: None

play_sound(*sound_id*, *reset_sound=True*)
info: will play sound :param sound_id: int :param reset_sound: bool :return: None

set_sound_position(*sound_id*, *spot*)
info: will set the spot of the sound :param sound_id: int :param spot: int :return: None

class tamcolors.tam_io.tam_drivers.**TAMDriver**(*identifier*, *mode_2=True*, *mode_16_pal_256=True*, *mode_16=True*, *mode_256=True*, *mode_rgb=True*, *key_driver_operational=True*, *color_driver_operational=True*, *color_changer_driver_operational=True*, *utilities_driver_operational=True*, *sound_driver_operational=True*)

Bases: *tamcolors.tam_io.io_tam.IO*, abc.ABC

classmethod able_to_execute()
info: checks that io is stable in current environment :return: bool

done()
info: operations for IO to stop :return: None

start()
info: operations for IO to start :return: None

class tamcolors.tam_io.tam_drivers.**UtilitiesDriver**(*identifier*, *mode_2=True*, *mode_16_pal_256=True*, *mode_16=True*, *mode_256=True*, *mode_rgb=True*, *key_driver_operational=True*, *color_driver_operational=True*, *color_changer_driver_operational=True*, *utilities_driver_operational=True*, *sound_driver_operational=True*)

Bases: *tamcolors.tam_io.tam_drivers.TAMDriver*, abc.ABC

clear()
info: Will clear the console :return: None

get_dimensions()
info: Gets the dimensions of console :return: (int, int): (row, column)

show_console_cursor(*show*)
info: Will show or hide console cursor :param show: int :return: None

tamcolors.tam_io.tam_identifier module

class tamcolors.tam_io.tam_identifier.**TAMIdentifier**(*name*, **drivers*)

Bases: object

get_all_drivers()
info: Get a list of all the tam drivers being used :return: list

get_info_dict()
info: Get raw info about this identifier :return: dict

get_io()
info: Gets the IO that the selected drivers made :return: IO

get_name()
info: Get the general name of the drivers :return: str

get_system()
info: Get the system :return: str

classmethod identify()
info: Will find the best drivers for current environment :return: TAMIdentifier

stable()
info: Sees if this IO is able to execute :return: bool

tamcolors.tam_io.tam_keys module

terminal keys supported on all platforms

class tamcolors.tam_io.tam_keys.**Keyboard**(*name*, *key_spot_dict*, *key_dict=None*, *key_state_dict=None*)

Bases: object

code_to_key(*code*)
info: will get key from code :param code: object :return: tuple or False

code_to_key_state(*code*)
info: will get key from code :param code: object :return: tuple or False

get_key_dict()
info: will get key dict :return: dict or None

get_key_list()

get_key_spot_dict()
info: will get key spot dict :return: dict

get_key_state_dict()
info: will get key state dict :return: dict or None

get_name()
info: will get keyboard name :return: str

key_to_spot(*key*)
info: will get spot from key :param key: tuple :return: int or None

static split_code_dict(*code_dict*)
info: will split a code dict :param code_dict: dict: {tuple: (object, object)} :return: dict, dict

spot_to_key(*spot*)
info: will get key from spot :param spot: int :return: tuple or None

tamcolors.tam_io.tam_surface module

class tamcolors.tam_io.tam_surface.**TAMSurface**(*width*, *height*, *char*, *foreground_color*, *background_color*)

Bases: *tamcolors.utils.object_packer.FastHandObjectPacker*

clear()
info: clears TAMSurface :return:

copy()
info: copy's TAMSurface :return: TAMSurface

draw_onto(*tam_surface*, *start_x=0*, *start_y=0*, *surface_start_x=0*, *surface_start_y=0*, *surface_size_x=-1*, *surface_size_y=-1*, *override_alpha=False*)
info: will draw tam_surface or part of a TAMSurface onto another TAMSurface :param tam_surface: TAMSurface :param start_x: int :param start_y: int :param surface_start_x: int :param surface_start_y: int :param surface_size_x: int: 0 - inf :param surface_size_y: int: 0 - inf :param override_alpha: bool :return:

classmethod from_bytes(*object_byte_array*)
info: from bytes to object :param object_byte_array: bytearray :return: object

get_cross_rect(*tam_surface*, *start_x=0*, *start_y=0*, *surface_start_x=0*, *surface_start_y=0*, *surface_size_x=-1*, *surface_size_y=-1*)
info: will draw tam_surface or part of a TAMSurface onto another TAMSurface :param tam_surface: TAMSurface :param start_x: int :param start_y: int :param surface_start_x: int :param surface_start_y: int :param surface_size_x: int: 0 - inf :param surface_size_y: int: 0 - inf :return:

get_defaults()
info: gets defaults :return: (str, int, int)

get_dimensions()
info: gets surface dimensions :return: (int, int)

get_from_raw_spot(*spot*)
info: gets spot info :param spot: x: int: 0 - (len(tam_surface) - 1) :return: (int, int, int) or None

get_raw_spot(*x*, *y*)
info: return -1 if not a spot :param x: int :param y: int :return: int

get_raw_surface()
info: gets raw surface :return: (list, list, list)

get_spot(*x*, *y*)
info: gets spot info :param x: int :param y: int :return: (int, int, int) or None

replace_alpha_chars(*alpha_replacement=None*)

Parameters **alpha_replacement** – None or str

Returns

set_defaults_and_clear(*char*, *foreground_color*, *background_color*)
info: clears surface and resets defaults :param char: str: len of 1 :param foreground_color: int: 0 - inf :param background_color: int: 0 - inf :return:

set_dimensions_and_clear(*width*, *height*)
info: clears and resizes TAMSurface :param width: int: 0 - inf :param height: int: 0 - inf :return:

set_spot(*x*, *y*, *char*, *foreground_color*, *background_color*, *override_alpha=False*)
info: sets a single spot on the surface :param x: int :param y: int :param char: str: len of 1 :param foreground_color: int :param background_color: int :param override_alpha: bool :return:

to_bytes()
info: object to bytes :return: bytes

tamcolors.tam_io.tcp_io module

tamcolors.tam_io.tcp_io.**get_tcp_io**(*receiver*, *wait=True*)
info: will return TCPObjectConnector connected to an io object :param receiver: TCPReceiver :param wait: bool :return: TCPObjectConnector

tamcolors.tam_io.tcp_io.**run_tcp_connection**(*connection*, *io=None*)
info: will run a tcp connection :param connection: TCPConnection :param io: IO or None: None will uses default IO :return: None

tamcolors.tam_io.uni_drivers module

class tamcolors.tam_io.uni_drivers.**UNISharedData**(*identifier*, *mode_2=True*, *mode_16_pal_256=True*, *mode_16=True*, *mode_256=True*, *mode_rgb=True*, *key_driver_operational=True*, *color_driver_operational=True*, *color_changer_driver_operational=True*, *utilities_driver_operational=True*, *sound_driver_operational=True*)

Bases: *tamcolors.tam_io.tam_drivers.TAMDriver*, abc.ABC

classmethod able_to_execute()
info: checks that io is stable in current environment :return: bool

class tamcolors.tam_io.uni_drivers.**UNIUtilitiesDriver**(*identifier*, *mode_2=True*, *mode_16_pal_256=True*, *mode_16=True*, *mode_256=True*, *mode_rgb=True*, *key_driver_operational=True*, *color_driver_operational=True*, *color_changer_driver_operational=True*, *utilities_driver_operational=True*, *sound_driver_operational=True*)

Bases: *tamcolors.tam_io.tam_drivers.UtilitiesDriver*, *tamcolors.tam_io.uni_drivers.UNISharedData*, abc.ABC

clear()
info: Will clear the console :return: None

done()
info: operations for IO to stop :return: None

get_dimensions()
info: Gets the dimensions of console :return: (int, int): (row, column)

show_console_cursor(*show*)
info: Will show or hide console cursor :param show: int :return: None

tamcolors.tam_io.win_drivers module

class tamcolors.tam_io.win_drivers.**WINFullColorDriver**(**args*, ***kwargs*)

Bases: *tamcolors.tam_io.tam_drivers.FullColorDriver*, *tamcolors.tam_io.win_drivers.WinSharedData*, abc.ABC

done()
info: operations for IO to stop :return: None

draw(*tam_surface*)
info: will draw tam surface to terminal :param tam_surface: TAMSurface :return:

get_printc_mode()
Gets the modes used by printc and inputc :return: str

inputc(*output*, *color*)
info: will get user input with color :param output: str :param color: tuple: (int, int) :return: str

printc(*output*, *color*, *flush*, *stderr*)
info: will print out user output with color :param output: str :param color: tuple: (int, int) :param flush: boolean :param stderr: boolean :return: None

set_mode(*mode*)
info: will set the color mode :param mode: int: key to color mode :return:

class tamcolors.tam_io.win_drivers.**WINKeyDriver**(**args*, ***kwargs*)
Bases: *tamcolors.tam_io.tam_drivers.KeyDriver*, *tamcolors.tam_io.win_drivers.WinSharedData*, abc.ABC

get_key()
info: will get single key input or return False :return: str or False

get_key_dict(*language=None*)
info: Gets a dict of all the keys :param language: str or None :return: dict

get_keyboard_name(*default_to_us_english=True*)
info: Will get the keyboard language name :param default_to_us_english: bool :return: str

class tamcolors.tam_io.win_drivers.**WINSoundDriver**(*identifier*, *mode_2=True*, *mode_16_pal_256=True*, *mode_16=True*, *mode_256=True*, *mode_rgb=True*, *key_driver_operational=True*, *color_driver_operational=True*, *color_changer_driver_operational=True*, *utilities_driver_operational=True*, *sound_driver_operational=True*)
Bases: *tamcolors.tam_io.tam_drivers.SoundDriver*, *tamcolors.tam_io.win_drivers.WinSharedData*, abc.ABC

close_sound(*sound_id*)
info: will close sound :param sound_id: int :return: None

get_sound_length(*sound_id*)
info: will get sound length :param sound_id: int :return: int

get_sound_position(*sound_id*)
info: will get the time spot of the song :param sound_id: int :return: int

is_sound_playing(*sound_id*)
info: will check if sound is playing :param sound_id: int :return: bool

open_sound(*file*, *sound_id*)
info: will open .wav sound :param file: str :param sound_id: int :return: None

pause_sound(*sound_id*)
info: will pause sound :param sound_id: int :return: None

play_sound(*sound_id*, *reset_sound=True*)
info: will play sound :param sound_id: int :param reset_sound: bool :return: None

set_sound_position(*sound_id*, *spot*)
info: will set the spot of the sound :param sound_id: int :param spot: int :return: None

class tamcolors.tam_io.win_drivers.**WINUtilitiesDriver**(**args*, ***kwargs*)
Bases: *tamcolors.tam_io.tam_drivers.UtilitiesDriver*, *tamcolors.tam_io.win_drivers.WinSharedData*, abc.ABC

clear()
info: will clear the screen :return:

done()
info: operations for IO to stop :return: None

get_dimensions()
info: will get teh terminal dimensions :return: (int, int)

show_console_cursor(*show*)
info: Will show or hide console cursor :param show: bool :return: None

start()
info: operations for IO to start :return: None

class tamcolors.tam_io.win_drivers.**WinSharedData**(*identifier*, *mode_2=True*, *mode_16_pal_256=True*, *mode_16=True*, *mode_256=True*, *mode_rgb=True*, *key_driver_operational=True*, *color_driver_operational=True*, *color_changer_driver_operational=True*, *utilities_driver_operational=True*, *sound_driver_operational=True*)
Bases: *tamcolors.tam_io.tam_drivers.TAMDriver*, abc.ABC

classmethod able_to_execute()
info: checks that io is stable in current environment :return: bool

Module contents

tamcolors.tam_tools package

Submodules

tamcolors.tam_tools.tam_color_palette module

class tamcolors.tam_tools.tam_color_palette.**TAMColorPalette**(*color_range=range(0, 16)*, *color_rules=None*)
Bases: object

get_color(*key*)
info: gets a color from the color palette :param key: object :return:

get_rule(*key*)
info: gets a instance of TAMColorPaletteRule or None :param key: object :return: instance of TAMColorPaletteRule or None

key_present(*key*)
info: checks if key is in color palette :param key: object :return: bool

set_color(*key*, *color*)
info sets a color to the color palette :param key: object :param color: Color :return:

set_rule(*key*, *rule*)
info: sets a color rule :param key: object :param rule: instance of TAMColorPaletteRule :return:

update()
info: updates color_palette rules :return:

exception tamcolors.tam_tools.tam_color_palette.**TAMColorPaletteError**
Bases: Exception

class tamcolors.tam_tools.tam_color_palette.**TAMColorPaletteRule**
Bases: object

update(*color_palette*, *key*)

class tamcolors.tam_tools.tam_color_palette.**TAMCycleColor**(*colors*, *clock=1*)
Bases: *tamcolors.tam_tools.tam_color_palette.TAMColorPaletteRule*

get_clock()
info: gets the clock rate :return: int

set_clock(*clock*)
info: sets the clock :param clock: int :return: int: 1 - inf

set_colors(*colors*)
info: sets all the colors :param colors: tuple or int: [Color, Color, Color, . . .] :return:

update(*color_palette*, *key*)
info: will update the color_palette color :param color_palette: TAMColorPalette :param key: object :return:

class tamcolors.tam_tools.tam_color_palette.**TAMDefaultColor**(*color*)
Bases: *tamcolors.tam_tools.tam_color_palette.TAMColorPaletteRule*

get_color()
info: gets the default color :return: Color

set_color(*color*)
info: sets the default color :param color: Color :return:

update(*color_palette*, *key*)
info: will update the color_palette color :param color_palette: TAMColorPalette :param key: object :return:

tamcolors.tam_tools.tam_fade module

tamcolors.tam_tools.tam_fade.**tam_fade_in**(*surface*, *char*, *foreground_color*, *background_color*, *rand=(True, False, False, False, False)*, *reverse=False*)
info: makes a fade in or fade out via TAMFilm :param surface: TAMSurface :param char: single block char :param foreground_color: Color :param background_color: Color :param rand: list: [True, bool, bool, bool, . . .] :param reverse: bool :return: TAMFilm

tamcolors.tam_tools.tam_film module

class tamcolors.tam_tools.tam_film.**TAMFilm**(*tam_surfaces=None*, *circular=False*)
Bases: object

append(*tam_surface*)
info :param tam_surface: :return:

done()
info: returns true if film is done :return: bool

get(*spot*)
info will get a TAMSurface :param spot: int: 0 - len(self._surface_list) :return: TAMSurface

get_circular()
info: gets the circular value :return: bool

peak()
info: gets the current frame :return: TAMSurface or None

pop()
info: will pop the last TAMSurface :return: TAMSurface or None

set(*spot*, *tam_surface*)
info: sets a frame :param spot: int: 0 - len(self._surface_list) :param tam_surface: TAMSurface :return:

set_circular(*circular*)
info: can set circular value :param circular: True :return:

slide()
info will get the next TAMSurface :return: TAMSurface or None

exception tamcolors.tam_tools.tam_film.**TAMFilmError**
Bases: Exception

tamcolors.tam_tools.tam_icon module

tamcolors.tam_tools.tam_icon.**get_icon**()

tamcolors.tam_tools.tam_key_manager module

class tamcolors.tam_tools.tam_key_manager.**TAMKeyManager**(*all_keys={('t', 'WHITESPACE'), ('n', 'WHITESPACE'), (' ', 'WHITESPACE'), ('!', 'NORMAL'), ('"', 'NORMAL'), ('#', 'NORMAL'), ('$', 'NORMAL'), ('%', 'NORMAL'), ('&', 'NORMAL'), ("'", 'NORMAL'), ('(', 'NORMAL'), (')', 'NORMAL'), ('*', 'NORMAL'), ('+', 'NORMAL'), (',', 'NORMAL'), ('-', 'NORMAL'), ('.', 'NORMAL'), ('/', 'NORMAL'), ('0', 'NORMAL'), ('1', 'NORMAL'), ('2', 'NORMAL'), ('3', 'NORMAL'), ('4', 'NORMAL'), ('5', 'NORMAL'), ('6', 'NORMAL'), ('7', 'NORMAL'), ('8', 'NORMAL'), ('9', 'NORMAL'), (':', 'NORMAL'), (';', 'NORMAL'), ('<', 'NORMAL'), ('=', 'NORMAL'), ('>', 'NORMAL'), ('?', 'NORMAL'), ('@', 'NORMAL'), ('A', 'NORMAL'), ('B', 'NORMAL'), ('BACKSPACE', 'SPECIAL'), ('C', 'NORMAL'), ('D', 'NORMAL'), ('DELETE', 'SPECIAL'), ('DOWN', 'SPECIAL'), ('E', 'NORMAL'), ('ESCAPE', 'SPECIAL'), ('F', 'NORMAL'), ('F1', 'SPECIAL'), ('F12', 'SPECIAL'), ('F12_SHIFT', 'SPECIAL'), ('F1_SHIFT', 'SPECIAL'), ('F2', 'SPECIAL'), ('F2_SHIFT', 'SPECIAL'), ('F3', 'SPECIAL'), ('F3_SHIFT', 'SPECIAL'), ('F4', 'SPECIAL'), ('F4_SHIFT', 'SPECIAL'), ('F5', 'SPECIAL'), ('F5_SHIFT', 'SPECIAL'), ('F6', 'SPECIAL'), ('F6_SHIFT', 'SPECIAL'), ('F7', 'SPECIAL'), ('F7_SHIFT', 'SPECIAL'), ('F8', 'SPECIAL'), ('F8_SHIFT', 'SPECIAL'), ('F9', 'SPECIAL'), ('F9_SHIFT', 'SPECIAL'), ('G', 'NORMAL'), ('H', 'NORMAL'),*

get_key_state(*key*)
info: will get a state of a key :param key: str :return: int

get_raw_user_input()
info: will get the raw user input :return: list or tuple: [(str, str), . . .]

get_user_input()
info: will the next key the user enters :return: tuple, list or None: (str, str)

get_user_input_generator()
info: yields a key at a time :return: list or tuple: (str, str)

silent_key_state(*key*)
info: will get a key state and make it 0 :param key: str :return: int

update(*keys*)
info: will update key manager with next set of keys :param keys: tuple or list: [(str, str), . . .] :return:

tamcolors.tam_tools.tam_list_surface module

tamcolors.tam_tools.tam_list_surface.**tam_list_surface**(*chars*, *foreground_colors*, *background_colors*)
info: makes a list into a TAMSurface :param chars: list, tuple :param foreground_colors: list, tuple, Color :param background_colors: list, tuple, Color :return: TAMSurface

tamcolors.tam_tools.tam_menu module

class tamcolors.tam_tools.tam_menu.**TAMButtonRule**
Bases: object

draw(*tam_surface*)

get_action()

get_position()

off()

on()

run_action()

set_action(*func*)

set_position(*x*, *y*)

update()

class tamcolors.tam_tools.tam_menu.**TAMMenu**(*buttons*, *call_key*, *goto_map*, *on=0*)
Bases: object

draw(*tam_surface*)
info: draw all the buttons of the menu :param tam_surface: TAMSurface :return:

get_buttons()
info: gets the buttons :return: list

get_call_key()
info: gets the call key :return: str

get_goto_map()
info: gets the goto map :return: dict

get_on()
info: gets the on value :return: int

static simple_menu_builder(*buttons*, *call_on*, *up_keys=('UP',)*, *down_keys=('DOWN',)*, *on=0*)
info: a simple way of making a TAMMenu :param buttons: TAMButtonRule :param call_on: str :param up_keys: list or tuple: (str, ...) :param down_keys: list or tuple: (str, ...) :param on: int :return: TAMMenu

update(*keys*)
info: update the menu :param keys: list or tuple: [(str, str), (str, str, ...), ...] :return:

class tamcolors.tam_tools.tam_menu.**TAMTextBoxButton**(*text*, *x*, *y*, *width*, *height*, *char*, *foreground_color*, *background_color*, *action_func*, *on_foreground_color*, *on_background_color*, *on_char='/'*, *clock=-1*, *center_vertical=True*, *center_horizontal=False*, *vertical_space=1*, *vertical_start=1*, *char_background=' '*)
Bases: *tamcolors.tam_tools.tam_menu.TAMButtonRule*

draw(*tam_surface*)
info: draws button onto surface :param tam_surface: :return:

get_action()
info: gets action function :return: fuction

get_position()
info: gets the position :return:

off()
info: puts the button onto the off state :return:

on()
info: puts the button onto the on state :return:

run_action()
" info: calls the action function :return:

set_action(*func*)
info: sets the actions :param func: function :return:

set_position(*x*, *y*)
info: sets the position :param x: int :param y: int :return:

update()
info: updates the button :return:

class tamcolors.tam_tools.tam_menu.**TAMTextButton**(*text*, *x*, *y*, *foreground_color*, *background_color*, *action_func*, *on_foreground_color*, *on_background_color*, *on_chars='*'*)
Bases: *tamcolors.tam_tools.tam_menu.TAMButtonRule*

draw(*tam_surface*)
info: draws button onto surface :param tam_surface: :return:

get_action()
info: gets action function :return: fuction

get_position()
info: gets the position :return:

off()
info: puts the button onto the off state :return:

on()
info: puts the button onto the on state :return:

run_action()
" info: calls the action function :return:

set_action(*func*)
info: sets the actions :param func: function :return:

set_position(*x*, *y*)
info: sets the position :param x: int :param y: int :return:

update()
info: updates the button :return:

tamcolors.tam_tools.tam_placing module

tamcolors.tam_tools.tam_placing.**bottom_left**(*x*, *y*, *width*, *height*)
info: gets the bottom left x, y :param x: int :param y: int :param width: int: 0 - inf :param height: int: 0 - inf :return: (int, int)

tamcolors.tam_tools.tam_placing.**bottom_right**(*x*, *y*, *width*, *height*)
info: gets the bottom right x, y :param x: int :param y: int :param width: int: 0 - inf :param height: int: 0 - inf :return: (int, int)

tamcolors.tam_tools.tam_placing.**center**(*x*, *y*, *width*, *height*)
info: gets the center x, y :param x: int :param y: int :param width: int: 0 - inf :param height: int: 0 - inf :return: (int, int)

tamcolors.tam_tools.tam_placing.**top_left**(*x*, *y*, *width*, *height*)
info: gets the top left x, y :param x: int :param y: int :param width: int: 0 - inf :param height: int: 0 - inf :return: (int, int)

tamcolors.tam_tools.tam_placing.**top_right**(*x*, *y*, *width*, *height*)
info: gets the top right x, y :param x: int :param y: int :param width: int: 0 - inf :param height: int: 0 - inf :return: (int, int)

tamcolors.tam_tools.tam_print module

tamcolors.tam_tools.tam_print.**tam_print**(*tam_surface*, *x*, *y*, *text*, *foreground_color*, *background_color*, *error_bad_char=False*, *bad_char=”*)
info: tam_print will draw string on a tam_surface :param tam_surface: TAMSurface :param x: int :param y: int :param text: object with __str__ :param foreground_color: int: -1 - inf: use current foreground_color :param background_color: int: -1 - inf: use current background_color :param error_bad_char: bool :param bad_char: str :return:

tamcolors.tam_tools.tam_str module

exception tamcolors.tam_tools.tam_str.**TAMStrError**
Bases: Exception

tamcolors.tam_tools.tam_str.**make_tam_str**(*text*, *end_line='\n'*, *bad_char=None*)
info: formats str into a tam str :param text: str :param end_line: str :param bad_char: str :return: str

tamcolors.tam_tools.tam_text_box module

class tamcolors.tam_tools.tam_text_box.**TAMTextBox**(*text*, *width*, *height*, *char*, *foreground_color*, *background_color*, *clock=-1*, *center_vertical=True*, *center_horizontal=False*, *vertical_space=1*, *vertical_start=1*, *char_background=' '*)
Bases: object

done()
info: True if text box has placed all chars onto to its self :return: bool

draw(*tam_surface*, *start_x=0*, *start_y=0*)
info: draws the text box on to another surface :param tam_surface: TAMSurface :param start_x: int :param start_y: int :return:

get_char()
info: gets char :return: str

get_colors()
info: gets textbox color :return: (int, int)

get_dimensions()
info: gets surface dimensions :return: (int, int)

get_text()
info: gets text :return: str

set_char(*char*)
info: sets char :return:

set_colors(*foreground_color*, *background_color*)
info: sets textbox colors :param foreground_color: 0 - inf :param background_color: 0 - inf :return:

update()
info updates the text box :return:

Module contents

tamcolors.tests package

Subpackages

tamcolors.tests.tam_basic_tests package

Submodules

tamcolors.tests.tam_basic_tests.basic_tests module

Module contents

tamcolors.tests.tam_io_tests package

Submodules

tamcolors.tests.tam_io_tests.any_drivers_tests module

class tamcolors.tests.tam_io_tests.any_drivers_tests.**AnyIOTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_events()

test_get_dimensions()

test_get_io()

test_get_key()

test_get_keyboard_name()

test_get_modes()

static test_reset_colors_to_console_defaults()

test_set_slash_get_mode()

static test_set_tam_color_defaults()

static test_snapshot()

test_sound()

tamcolors.tests.tam_io_tests.any_drivers_tests.**get_any_io**()

tamcolors.tests.tam_io_tests.io_tam_tests module

class tamcolors.tests.tam_io_tests.io_tam_tests.**IOTAMTest**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test__draw_onto()

test__draw_onto_2()

test__draw_onto_3()

static test__write_to_output_stream()

static test__write_to_output_stream_2()

class tamcolors.tests.tam_io_tests.io_tam_tests.**RAWIOTest**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_same_doc_string()

test_same_methods()

tamcolors.tests.tam_io_tests.tam_colors_tests module

class tamcolors.tests.tam_io_tests.tam_colors_tests.**ColorTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_color_eq()

test_color_eq_2()

test_color_eq_3()

test_color_init()

test_color_init_2()

test_color_ne()

test_color_ne_2()

test_get_property()

test_get_property_2()

class tamcolors.tests.tam_io_tests.tam_colors_tests.**RGBATests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_get_property()

test_get_property_2()

test_rgba_eq()

test_rgba_eq_2()

test_rgba_eq_3()

test_rgba_init()

test_rgba_init_2()

test_rgba_ne()

test_rgba_ne_2()

class tamcolors.tests.tam_io_tests.tam_colors_tests.**TAMColorTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_colors()

tamcolors.tests.tam_io_tests.tam_keys_tests module

class tamcolors.tests.tam_io_tests.tam_keys_tests.**TAMKeyTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_keys()

tamcolors.tests.tam_io_tests.tam_surface_tests module

class tamcolors.tests.tam_io_tests.tam_surface_tests.**TAMSurfaceTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_clear()

test_copy()

test_copy_2()

test_draw_onto()

test_draw_onto_2()

test_draw_onto_3()

test_draw_onto_4()

test_draw_onto_5()

test_draw_onto_6()

test_draw_onto_7()

test_draw_onto_8()

test_get_cross_rect()

test_get_cross_rect_2()

test_get_defaults()

test_get_dimensions()

test_get_dimensions_2()

test_get_dimensions_3()

test_get_from_raw_spot()

test_get_raw_spot()

test_get_raw_surface()

test_get_raw_surface_2()

test_get_spot()

test_replace_alpha_chars_1()

test_replace_alpha_chars_2()

test_replace_alpha_chars_3()

test_set_defaults_and_clear()

test_set_defaults_and_clear_2()

test_set_dimensions_and_clear()

test_set_dimensions_and_clear_2()

test_set_spot()

test_set_spot_2()

test_set_spot_3()

test_set_spot_4()

test_set_spot_5()

test_set_spot_6()

test_set_spot_7()

test_set_spot_8()

test_set_spot_9()

test_surface_eq()

test_surface_eq_2()

test_surface_eq_3()

test_surface_eq_4()

test_surface_eq_5()

test_surface_init()

test_surface_len()

test_surface_len_2()

test_surface_len_3()

test_surface_len_4()

test_surface_ne()

test_surface_ne_2()

test_surface_ne_3()

test_surface_ne_4()

test_surface_str()

test_surface_str_2()

test_surface_str_3()

test_to_bytes_from_bytes()

tamcolors.tests.tam_io_tests.uni_drivers_tests module

tamcolors.tests.tam_io_tests.win_drivers_tests module

class tamcolors.tests.tam_io_tests.win_drivers_tests.**WinDriversTests**(*methodName='runTest'*)

Bases: unittest.case.TestCase

test__console_color_count()

test__draw_16()

static test__draw_2()

test__get_buffer_dimension()

static test__print()

static test__print_2()

static test__print_3()

test__processes_special_color_1()

test__processes_special_color_2()

test__processes_special_color_3()

test__processes_special_color_4()

test__processes_special_color_5()

test__processes_special_color_6()

test__spot_swap()

test_able_to_execute()

static test_close()

test_done()

test_full_play()

test_get_dimensions()

test_get_key()

test_get_key_2()

test_get_key_3()

test_get_key_4()

test_get_key_5()

test_get_key_dict()

test_get_keyboard_name()

test_get_keyboard_name_2()

test_get_keyboard_name_3()

test_get_keyboard_name_4()

test_get_modes()

static test_play()

test_position()

test_reset_colors_to_console_defaults()

test_rest()

test_set_slash_get_mode()

test_set_tam_color_defaults()

test_start()

class tamcolors.tests.tam_io_tests.win_drivers_tests.**WinGlobalsTests**(*methodName='runTest'*)

Bases: unittest.case.TestCase

test_win_stable()

tamcolors.tests.tam_io_tests.win_drivers_tests.**get_win_io**()

Module contents

tamcolors.tests.tam_tests package

Submodules

tamcolors.tests.tam_tests.tam_loop_tcp_receiver_tests module

class tamcolors.tests.tam_tests.tam_loop_tcp_receiver_tests.**TAMLoopTCPReceiver**(*methodName='r*
Bases: *tamcolors.tests.test_multi_task_helper.MultiTaskHelper*, unittest.case.TestCase

test_loop_tcp_receiver()

tamcolors.tests.tam_tests.tam_loop_tests module

class tamcolors.tests.tam_tests.tam_loop_tests.**TAMFrameTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test__done()

test__done_2()

test__frame_done()

test__frame_done_2()

test_draw()

test_frame_init()

test_get_defaults()

test_get_defaults_2()

test_get_fps()

test_get_fps_2()

test_get_height_min_and_max()

test_get_height_min_and_max_2()

test_get_width_min_and_max()

test_get_width_min_and_max_2()

test_make_surface_ready()

test_make_surface_ready_2()

test_make_surface_ready_3()

test_update()

class tamcolors.tests.tam_tests.tam_loop_tests.**TAMLoopTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_loop_init()

test_preferred_mode()

test_preferred_mode_2()

test_preferred_mode_3()

test_reset_colors_to_console_defaults()

test_run()

test_set_tam_color_defaults()

test_stack()

Module contents

tamcolors.tests.tam_tools_tests package

Submodules

tamcolors.tests.tam_tools_tests.tam_color_palette_tests module

class tamcolors.tests.tam_tools_tests.tam_color_palette_tests.**TAMColorPaletteTests**(*methodNam*
Bases: unittest.case.TestCase

test_get_color()

test_get_color_2()

test_get_color_3()

test_get_rule()

test_get_rule_2()

test_getitem()

test_getitem_2()

test_getitem_3()

test_init_color_palette()

test_init_color_palette_2()

test_key_present()

test_key_present_2()

test_set_color()

test_set_color_2()

test_set_color_3()

test_set_rule()

test_set_rule_2()

test_setitem()

test_setitem_2()

test_setitem_3()

test_str()

test_str_2()

test_update()

class tamcolors.tests.tam_tools_tests.tam_color_palette_tests.**TAMCycleColorTests**(*methodName=*
Bases: unittest.case.TestCase

test_get_clock()

test_init_cycle_color()

test_set_clock()

test_set_colors()

test_update()

test_update_2()

class tamcolors.tests.tam_tools_tests.tam_color_palette_tests.**TAMDefaultColorTests**(*methodNa*
Bases: unittest.case.TestCase

test_get_color()

test_init_default_color()

test_set_color()

test_update()

tamcolors.tests.tam_tools_tests.tam_fade_tests module

class tamcolors.tests.tam_tools_tests.tam_fade_tests.**TAMFilmFadeInTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_tam_fade_in()

test_tam_fade_in_2()

test_tam_fade_in_3()

test_tam_fade_in_4()

tamcolors.tests.tam_tools_tests.tam_film_tests module

class tamcolors.tests.tam_tools_tests.tam_film_tests.**TAMFilmTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_append()

test_done()

test_done_2()

test_done_3()

test_get()

test_get_2()

test_get_3()

test_get_circular()

test_get_circular_2()

test_getitem()

test_getitem_2()

test_getitem_3()

test_init_film()

test_len()

test_len_2()

test_next()

test_next_2()

test_peak()

test_peak_2()

test_peak_3()

test_pop()

test_set()

test_set_2()

test_set_3()

test_set_circular()

test_setitem()

test_setitem_2()

test_setitem_3()

test_slide()

test_slide_2()

tamcolors.tests.tam_tools_tests.tam_key_manager_tests module

class tamcolors.tests.tam_tools_tests.tam_key_manager_tests.**TAMKeyManagerTests**(*methodName='r*

Bases: unittest.case.TestCase

get_raw_user_input_2()

get_raw_user_input_3()

test_get_key_state()

test_get_key_state_2()

test_get_raw_user_input()

test_get_user_input()

test_get_user_input_2()

test_get_user_input_3()

test_get_user_input_generator()

test_get_user_input_generator_2()

test_get_user_input_generator_3()

test_init_key_manger()

test_iter()

test_silent_key_state()

test_silent_key_state_2()

test_silent_key_state_3()

test_update()

test_update_2()

tamcolors.tests.tam_tools_tests.tam_list_surface_tests module

class tamcolors.tests.tam_tools_tests.tam_list_surface_tests.**TAMListSurfaceTests**(*methodName=*
Bases: unittest.case.TestCase

test_tam_list_surface()

test_tam_list_surface_2()

test_tam_list_surface_3()

test_tam_list_surface_4()

test_tam_list_surface_5()

tamcolors.tests.tam_tools_tests.tam_menu_tests module

class tamcolors.tests.tam_tools_tests.tam_menu_tests.**TAMMenuTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_draw()

test_get_buttons()

test_get_buttons_2()

test_get_buttons_3()

test_get_call_key()

test_get_call_key_2()

test_get_goto_map()

test_get_on()

test_get_on_2()

test_simple_menu_builder()

test_simple_menu_builder_2()

test_simple_menu_builder_3()

test_simple_menu_builder_4()

test_simple_menu_builder_5()

test_tam_menu()

test_tam_menu_2()

test_update()

test_update_2()

test_update_3()

class tamcolors.tests.tam_tools_tests.tam_menu_tests.**TAMTextBoxButtonTests**(*methodName='runTest*
Bases: unittest.case.TestCase

static test_call_TextBoxButton()

test_draw()

test_draw_2()

test_get_action()

test_get_position()

test_get_position_2()

test_init_TextBoxButton()

test_init_TextBoxButton_2()

test_off()

test_off_2()

test_on()

test_on_2()

static test_run_action()

test_set_action()

test_set_position()

test_set_position_2()

test_str_TextBoxButton()

test_str_TextBoxButton_2()

static test_update()

class tamcolors.tests.tam_tools_tests.tam_menu_tests.**TAMTextButtonTests**(*methodName='runTest'*)

Bases: unittest.case.TestCase

static test_call_TextButton()

test_draw()

test_draw_2()

test_get_action()

test_get_position()

test_get_position_2()

test_init_TextButton()

test_init_TextButton_2()

test_off()

test_off_2()

test_on()

test_on_2()

static test_run_action()

test_set_action()

test_set_position()

test_set_position_2()

test_str_TextButton()

test_str_TextButton_2()

static test_update()

tamcolors.tests.tam_tools_tests.tam_placing_tests module

class tamcolors.tests.tam_tools_tests.tam_placing_tests.**TAMPlacingTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test__get_center()

test__get_center_2()

test__get_dimensions_wrapper()

test__get_dimensions_wrapper_2()

test__get_dimensions_wrapper_3()

test__get_other_side()

test__get_other_side_2()

test_bottom_left()

test_bottom_left_2()

test_bottom_right()

test_bottom_right_2()

test_center()

test_center_2()

test_top_left()

test_top_left_2()

test_top_right()

test_top_right_2()

tamcolors.tests.tam_tools_tests.tam_print_tests module

class tamcolors.tests.tam_tools_tests.tam_print_tests.**TAMPrintTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_tam_print()

test_tam_print_2()

test_tam_print_3()

test_tam_print_4()

test_tam_print_5()

tamcolors.tests.tam_tools_tests.tam_str_tests module

class tamcolors.tests.tam_tools_tests.tam_str_tests.**MakeTAMStrTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_make_tam_str()

test_make_tam_str_2()

test_make_tam_str_3()

test_make_tam_str_4()

test_make_tam_str_5()

tamcolors.tests.tam_tools_tests.tam_text_box_tests module

class tamcolors.tests.tam_tools_tests.tam_text_box_tests.**TAMTextBoxTests**(*methodName='runTest'*)

Bases: unittest.case.TestCase

test_done()

test_done_2()

test_draw()

test_draw_2()

test_draw_3()

test_get_char()

test_get_char_2()

test_get_colors()

test_get_colors_2()

test_get_dimensions()

test_get_dimensions_2()

test_get_text()

test_get_text_2()

test_set_char()

test_set_char_2()

test_set_colors()

test_set_colors_2()

test_set_colors_3()

test_tam_text_box_init()

test_tam_text_box_str()

test_tam_text_box_str_2()

test_update()

Module contents

tamcolors.tests.tests_tests package

Submodules

tamcolors.tests.tests_tests.test_multi_task_helper_tests module

class tamcolors.tests.tests_tests.test_multi_task_helper_tests.**MultiTaskHelperTests**(*methodNa*
Bases: *tamcolors.tests.test_multi_task_helper.MultiTaskHelper*, unittest.case.TestCase

test_assert_no_main_process()

test_assert_no_main_thread()

test_assert_process()

test_assert_thread()

test_hand_down()

Module contents

tamcolors.tests.utils_tests package

Submodules

tamcolors.tests.utils_tests.compress_tests module

class tamcolors.tests.utils_tests.compress_tests.**CompressTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_1()

test_2()

test_3()

test_4()

tamcolors.tests.utils_tests.encryption_tests module

class tamcolors.tests.utils_tests.encryption_tests.**EncryptionTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_build_setting()

test_encrypt_decrypt()

test_init()

test_simple_encrypt_decrypt()

class tamcolors.tests.utils_tests.encryption_tests.**NoEncryptionTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_init()

tamcolors.tests.utils_tests.identifier_tests module

class tamcolors.tests.utils_tests.identifier_tests.**IdentifierTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_generate_identifier()

test_get_identifier_bytes()

test_globals()

tamcolors.tests.utils_tests.immutable_cache_tests module

class tamcolors.tests.utils_tests.immutable_cache_tests.**DummyCache**(**args*, ***kwargs*)

Bases: *tamcolors.utils.immutable_cache.ImmutableCache*

class tamcolors.tests.utils_tests.immutable_cache_tests.**ImmutableCacheTests**(*methodName='runTe*

Bases: unittest.case.TestCase

test_cache()

test_init()

test_new()

tamcolors.tests.utils_tests.log_tests module

class tamcolors.tests.utils_tests.log_tests.**LogTests**(*methodName='runTest'*)

Bases: unittest.case.TestCase

test_critical()

test_debug()

static test_enable_and_disable_logging()

test_log_stream()

tamcolors.tests.utils_tests.object_packer_tests module

class tamcolors.tests.utils_tests.object_packer_tests.**FastHandObjectPackerTests**(*methodName='*

Bases: unittest.case.TestCase

test_dunder_bytes()

test_tam_color_1()

test_tam_color_2()

test_tam_surface_1()

test_tam_surface_2()

class tamcolors.tests.utils_tests.object_packer_tests.**ObjectPackerFunctionsTests**(*methodName=*

Bases: unittest.case.TestCase

test_save_and_load_data_1()

test_save_and_load_data_2()

test_save_and_load_data_3()

test_save_and_load_data_4()

test_save_and_load_int_1()

test_save_and_load_int_2()

test_save_and_load_int_3()

test_save_and_load_int_4()

class tamcolors.tests.utils_tests.object_packer_tests.**ObjectPackerJsonTests**(*methodName='runTe*
Bases: unittest.case.TestCase

test_bool()

test_bytearray()

test_bytes()

test_dict()

test_fast_hand_object()

test_float()

test_int()

test_large_data()

test_list()

test_none()

test_set()

test_str()

test_tuple()

tamcolors.tests.utils_tests.tcp_tests module

class tamcolors.tests.utils_tests.tcp_tests.**TCPObjectWrapper**(*methodName='runTest'*)
Bases: *tamcolors.tests.test_multi_task_helper.MultiTaskHelper*, unittest.case.TestCase

test_object_wrapper()

class tamcolors.tests.utils_tests.tcp_tests.**TCPTests**(*methodName='runTest'*)
Bases: *tamcolors.tests.test_multi_task_helper.MultiTaskHelper*, unittest.case.TestCase

test_ping()

test_ping_ipv6()

tamcolors.tests.utils_tests.timer_tests module

class tamcolors.tests.utils_tests.timer_tests.**TimerTests**(*methodName='runTest'*)
Bases: unittest.case.TestCase

test_lap()

test_offset_sleep()

test_timer()

tamcolors.tests.utils_tests.transport_optimizer_tests module

class tamcolors.tests.utils_tests.transport_optimizer_tests.**TransportOptimizerTests**(*methodNa*
Bases: unittest.case.TestCase

test_large_data()

test_large_tam_surface_data()

test_same_data()

test_simple_data()

Module contents

Submodules

tamcolors.tests.all_tests module

tamcolors.tests.all_tests.**load_tests**(*loader*, *other_tests=None*, *pattern=None*)

tamcolors.tests.all_tests.**stability_check**(*ret_bool=True*, *run_slow=False*)
info: run all TAM tests :param ret_bool: bool :param run_slow: bool :return: (int, int) or bool: (test_pasted, test_ran) or True if all test pasted

tamcolors.tests.all_tests.**tests_main**(*run_slow=False*)
info: the main way tamcolors run tests :param run_slow: bool: will run slow tests :return: bool

tamcolors.tests.test_multi_task_helper module

class tamcolors.tests.test_multi_task_helper.**MultiTaskHelper**
Bases: object

multiple_processes_helper(*tasks*, *timeout=120*)
info: will run tasks in a thread :param tasks: list or tuple: [(func, tuple, dict), ...] :param timeout: int :return:

multiple_threads_helper(*tasks*, *timeout=120*)
info: will run tasks in a thread :param tasks: list or tuple: [(func, tuple, dict), ...] :param timeout: int :return:

classmethod task(*func*, **args*, ***kwargs*)
info: will make a Task :param func: Function :param args: tuple :param kwargs: dict :return: Task

tamcolors.tests.test_multi_task_helper.**process_runner**(*error_ret*, *func*, **args*, ***kwargs*)
info: will run the task :param error_ret: list: away to return an error :param func: function :param args: tuple :param kwargs: dict :return:

tamcolors.tests.test_utils module

tamcolors.tests.test_utils.**enable_slow_tests**(*enable=True*)
info: will enable slow tests :param enable: bool :return:

tamcolors.tests.test_utils.**is_slow_tests_enabled**()
info: will check iif slow tests are enabled :return: bool

tamcolors.tests.test_utils.**slow_test**(*function*)
info: slow_test decorators :param function: function :return: function

Module contents

tamcolors.utils package

Submodules

tamcolors.utils.compress module

tamcolors.utils.compress.**compress**(*data*)
info: will compress data :param data: bytes :return: bytes

tamcolors.utils.compress.**decompress**(*data*)
info: will defcompress data :param data: bytes :return: bytes

tamcolors.utils.encryption module

class tamcolors.utils.encryption.**Encryption**(*rsa_key_size=4096*, *aes_key_size=256*, *nonce_key_size=64*, *authenticator=b”*, *max_random=20*)
Bases: object

static check_sandy_check_key(*sandy_key*, *data*)
info: Checks if the sandy key is equal to the data key. :param sandy_key: bytes: sha hash key :param data: bytes: :return: bool

decrypt(*data*)
info: Decrypt message. The data will also be checked for corruption. :param data: bytes :return: bytes: Unencrypted data.

encrypt(*data*)
info: Encrypts data. :param data: bytes :return: bytes: Encrypted data.

encrypt_with_public_key(*key*, *data*)
info: Encrypts data with public key. :param key: bytes or ras key object :param data: bytes :return: bytes: encrypted data

get_raw_private_key()
info: Gets the raw private key. :return: bytes: PEM encoding

get_raw_public_key()
info: gets the raw public key. :return: bytes: PEM encoding

static make_sandy_check_key(*data*)
info: Hashes data. :param data: bytes :return: bytes: Hash key.

exception tamcolors.utils.encryption.**EncryptionError**
Bases: Exception

tamcolors.utils.identifier module

tamcolors.utils.identifier.**generate_identifier_bytes**(*identifier_file='/home/docs/checkouts/readthedocs.org/u*
identifier_size=500)

tamcolors.utils.identifier.**get_identifier_bytes**(*identifier_file='/home/docs/checkouts/readthedocs.org/user_bu*

tamcolors.utils.immutable_cache module

class tamcolors.utils.immutable_cache.**ImmutableCache**
Bases: object

tamcolors.utils.log module

class tamcolors.utils.log.**Log**(*size=5000*)
Bases: object

first_msg_id()

last_msg_id()

log(*msg*)

read(*log_id*)

tamcolors.utils.log.**critical**(*msg*)
info: log at critical level :param msg: str :return:

tamcolors.utils.log.**debug**(*msg*)
info: log at debug level :param msg: str :return:

tamcolors.utils.log.**disable_logging**()
info: will disable logging :return:

tamcolors.utils.log.**enable_logging**(*level=10*)
info: will enable logging at a level :param level: log level :return:

tamcolors.utils.log.**error**(*msg*)
info: log at debug level :param msg: str :return:

tamcolors.utils.log.**format_message**(*func*)

tamcolors.utils.log.**info**(*msg*)
info: log at info level :param msg: str :return:

tamcolors.utils.log.**warning**(*msg*)
info: log at warning level :param msg: str :return:

tamcolors.utils.object_packer module

class tamcolors.utils.object_packer.**FastHandObjectPacker**
Bases: object

classmethod from_bytes(*object_byte_array*)
info: from bytes to object :param object_byte_array: bytearray :return: object

classmethod start_from_bytes(*object_bytes*)
info: from bytes to object :param object_bytes: bytearray, bytes, list, tuple :return: object

to_bytes()
info: object to bytes :return: bytes

class tamcolors.utils.object_packer.**ObjectPackerJson**(*fast_hand_object_packer_objects=None*)
Bases: object

dumps(*data*)
info: object to bytes :param data: object :return: bytes

loads(*data*)
info: bytes to object :param data: bytearray, bytes, list, tuple :return: object

exception tamcolors.utils.object_packer.**ObjectPackerJsonError**
Bases: Exception

tamcolors.utils.object_packer.**load_data**(*object_byte_array*)
info: Loads bytes :param object_byte_array: bytearray :return: bytes

tamcolors.utils.object_packer.**load_int**(*object_byte_array*)
info: Loads an unsigned int :param object_byte_array: bytearray :return: int

tamcolors.utils.object_packer.**save_data**
info: Saves bytes :param data: bytes :return: bytes

tamcolors.utils.object_packer.**save_int**
info: Saves an unsigned int :param number: int :return: bytes

tamcolors.utils.tcp module

class tamcolors.utils.tcp.**TCPBase**(*connection*, *address*, *port*, *connection_password*, *encryption=None*, *object_packer=None*, *our_information=None*)
Bases: object

close()
info: will close the connection :return: None

get_address()
info: will get the address :return: str

get_data()
info: will compress and encrypt data if encryption is enabled :return: bytes

is_open()
info: will check if connection is open :return: bool

send_data(*data*)
info: will compress and decrypt if encryption is enabled :param data: bytes :return: None

class tamcolors.utils.tcp.**TCPConnection**(*host='127.0.0.1'*, *port=4444*, *ipv6=False*, *connection_password=''*, *user_name=None*, *user_id=None*, *encryption=None*, *object_packer=None*, *our_information=None*)
Bases: *tamcolors.utils.tcp.TCPBase*

get_other_data()
info: will get other data :return: object

exception tamcolors.utils.tcp.**TCPError**
Bases: Exception

class tamcolors.utils.tcp.**TCPHost**(**args*, ***kwargs*)
Bases: *tamcolors.utils.tcp.TCPBase*

get_other_data()
info: will get other data :return: object

get_user_id()
info: get connection id :return: bytes

get_user_name()
info: get connection name :return: str

class tamcolors.utils.tcp.**TCPObjectConnector**(*tcp_connection*, *object_packer=None*, *no_return=None*, *optimizer=None*, *none_generator=None*)
Bases: object

close()
info: will close the object :return:

get_connection()
info: will get connection :return: TCPBase

is_open()
info: will check if object is still open :return: bool

class tamcolors.utils.tcp.**TCPObjectWrapper**(*tcp_connection*, *obj*, *object_packer=None*)
Bases: object

close()
info: will close the object :return:

get_connection()
info: will get connection :return: TCPBase

is_open()
info: will check if object is still open :return: bool

class tamcolors.utils.tcp.**TCPReceiver**(*host='127.0.0.1'*, *port=4444*, *ipv6=False*, *listen_count=10*, *connection_password=''*, *address_white_list=None*, *address_black_list=None*, *encryption=None*, *object_packer=None*, *our_information=None*)
Bases: object

close()
info: will close tcp receiver connection :return: None

get_host_connection(*wait=True*)
info: will get a new host connection :param wait: bool: if true will wait for a connection :return: None or TCPHost

tamcolors.utils.timer module

class tamcolors.utils.timer.**TickRateTracker**(*frame=1*)
Bases: object

tick()
info: will add a tick :return:

tick_rate()
info: will get tick rate :return: int

class tamcolors.utils.timer.**Timer**(*time_corruption=0*)
Bases: object

lap()
Gets time difference from last lap. :return: float

offset_sleep(*sleep_time*)
Will sleep the thread for a length of time based of the lap time. :param sleep_time: float :return: float

tamcolors.utils.transport_optimizer module

class tamcolors.utils.transport_optimizer.**LastReceivedCache**
Bases: object

class tamcolors.utils.transport_optimizer.**LastSentCache**
Bases: object

Module contents

Module contents

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

t

tamcolors, 69
tamcolors.examples, 17
tamcolors.examples.alpha, 13
tamcolors.examples.clouds, 14
tamcolors.examples.colors, 14
tamcolors.examples.connection_loopback, 14
tamcolors.examples.connection_multi_player, 14
tamcolors.examples.host_loopback, 14
tamcolors.examples.host_multi_player, 14
tamcolors.examples.icon, 15
tamcolors.examples.rgb_color, 15
tamcolors.examples.sandy_check, 15
tamcolors.examples.tabletennis, 15
tamcolors.examples.tam_key_manager, 16
tamcolors.examples.tam_keys, 16
tamcolors.examples.tam_list_surface, 16
tamcolors.examples.tam_loop, 17
tamcolors.examples.tam_print, 17
tamcolors.examples.tam_text_box, 17
tamcolors.tam, 22
tamcolors.tam.tam_loop, 18
tamcolors.tam.tam_loop_io_handler, 19
tamcolors.tam.tam_loop_io_tcp_handler, 21
tamcolors.tam.tam_loop_receiver, 21
tamcolors.tam.tam_loop_tcp_receiver, 22
tamcolors.tam_basic, 22
tamcolors.tam_c, 22
tamcolors.tam_io, 40
tamcolors.tam_io.ansi_256_drivers, 22
tamcolors.tam_io.ansi_true_color_drivers, 23
tamcolors.tam_io.any_drivers, 23
tamcolors.tam_io.io_tam, 25
tamcolors.tam_io.null_drivers, 30
tamcolors.tam_io.tam_colors, 31
tamcolors.tam_io.tam_drivers, 32
tamcolors.tam_io.tam_identifier, 35
tamcolors.tam_io.tam_keys, 36
tamcolors.tam_io.tam_surface, 36
tamcolors.tam_io.tcp_io, 37
tamcolors.tam_io.uni_drivers, 38
tamcolors.tam_io.win_drivers, 38
tamcolors.tam_tools, 47
tamcolors.tam_tools.tam_color_palette, 40
tamcolors.tam_tools.tam_fade, 41
tamcolors.tam_tools.tam_film, 41
tamcolors.tam_tools.tam_icon, 42
tamcolors.tam_tools.tam_key_manager, 43
tamcolors.tam_tools.tam_list_surface, 44
tamcolors.tam_tools.tam_menu, 44
tamcolors.tam_tools.tam_placing, 46
tamcolors.tam_tools.tam_print, 46
tamcolors.tam_tools.tam_str, 47
tamcolors.tam_tools.tam_text_box, 47
tamcolors.tests, 65
tamcolors.tests.all_tests, 64
tamcolors.tests.tam_basic_tests, 48
tamcolors.tests.tam_io_tests, 52
tamcolors.tests.tam_io_tests.any_drivers_tests, 48
tamcolors.tests.tam_io_tests.io_tam_tests, 48
tamcolors.tests.tam_io_tests.tam_colors_tests, 49
tamcolors.tests.tam_io_tests.tam_keys_tests, 49
tamcolors.tests.tam_io_tests.tam_surface_tests, 49
tamcolors.tests.tam_io_tests.win_drivers_tests, 51
tamcolors.tests.tam_tests, 54

tamcolors.tests.tam_tests.tam_loop_tcp_remote_tests, 53
tamcolors.tests.tam_tests.tam_loop_tests, 53
tamcolors.tests.tam_tools_tests, 60
tamcolors.tests.tam_tools_tests.tam_color_picker_tests, 54
tamcolors.tests.tam_tools_tests.tam_fade_tests, 55
tamcolors.tests.tam_tools_tests.tam_film_tests, 55
tamcolors.tests.tam_tools_tests.tam_key_manager_tests, 56
tamcolors.tests.tam_tools_tests.tam_list_surface_tests, 57
tamcolors.tests.tam_tools_tests.tam_menu_tests, 57
tamcolors.tests.tam_tools_tests.tam_placing_tests, 59
tamcolors.tests.tam_tools_tests.tam_print_tests, 59
tamcolors.tests.tam_tools_tests.tam_str_tests, 59
tamcolors.tests.tam_tools_tests.tam_text_box_tests, 60
tamcolors.tests.test_multi_task_helper, 64
tamcolors.tests.test_utils, 64
tamcolors.tests.tests_tests, 61
tamcolors.tests.tests_tests.test_multi_task_helper_tests, 61
tamcolors.tests.utils_tests, 64
tamcolors.tests.utils_tests.compress_tests, 61
tamcolors.tests.utils_tests.encryption_tests, 61
tamcolors.tests.utils_tests.identifier_tests, 61
tamcolors.tests.utils_tests.immutable_cache_tests, 62
tamcolors.tests.utils_tests.log_tests, 62
tamcolors.tests.utils_tests.object_packer_tests, 62
tamcolors.tests.utils_tests.tcp_tests, 63
tamcolors.tests.utils_tests.timer_tests, 63
tamcolors.tests.utils_tests.transport_optimizer_tests, 64
tamcolors.utils, 69
tamcolors.utils.compress, 65
tamcolors.utils.encryption, 65
tamcolors.utils.identifier, 65
tamcolors.utils.immutable_cache, 66
tamcolors.utils.log, 66
tamcolors.utils.object_packer, 66
tamcolors.utils.tcp, 67
tamcolors.utils.timer, 68
tamcolors.utils.transport_optimizer, 69

Index

A

a (*tamcolors.tam_io.tam_colors.RGBA attribute*), 32
able_to_execute() (*tamcolors.tam_io.io_tam.IO class method*), 25
able_to_execute() (*tamcolors.tam_io.io_tam.RawIO class method*), 27
able_to_execute() (*tamcolors.tam_io.tam_drivers.TAMDriver class method*), 35
able_to_execute() (*tamcolors.tam_io.uni_drivers.UNISharedData class method*), 38
able_to_execute() (*tamcolors.tam_io.win_drivers.WinSharedData class method*), 40
add_frame_stack() (*tamcolors.tam.tam_loop.TAMLoop method*), 18
add_receiver() (*tamcolors.tam.tam_loop.TAMLoop method*), 18
ANSI256ChangerDriver (*class in tamcolors.tam_io.ansi_256_drivers*), 22
ANSI256ColorDriver (*class in tamcolors.tam_io.ansi_256_drivers*), 22
ANSITrueFullColorDriver (*class in tamcolors.tam_io.ansi_true_color_drivers*), 23
ANYFullColorDriver (*class in tamcolors.tam_io.any_drivers*), 23
AnyIOTests (*class in tamcolors.tests.tam_io_tests.any_drivers_tests*), 48
ANYKeyDriver (*class in tamcolors.tam_io.any_drivers*), 23
ANYSoundDriver (*class in tamcolors.tam_io.any_drivers*), 24
ANYUtilitiesDriver (*class in tamcolors.tam_io.any_drivers*), 24
append() (*tamcolors.tam_tools.tam_film.TAMFilm method*), 41
apply_snapshot() (*tamcolors.tam_io.io_tam.IO method*), 25
apply_snapshot() (*tamcolors.tam_io.io_tam.RawIO method*), 27

B

b (*tamcolors.tam_io.tam_colors.RGBA attribute*), 32
Ball (*class in tamcolors.examples.tabletennis*), 15
BootLogo (*class in tamcolors.examples.icon*), 15
bottom_left() (*in module tamcolors.tam_tools.tam_placing*), 46
bottom_right() (*in module tamcolors.tam_tools.tam_placing*), 46

C

center() (*in module tamcolors.tam_tools.tam_placing*), 46
check_sandy_check_key() (*tamcolors.utils.encryption.Encryption static method*), 65
clear() (*tamcolors.tam_io.any_drivers.ANYUtilitiesDriver method*), 24
clear() (*tamcolors.tam_io.io_tam.IO method*), 25
clear() (*tamcolors.tam_io.io_tam.RawIO method*), 27
clear() (*tamcolors.tam_io.tam_drivers.UtilitiesDriver method*), 35
clear() (*tamcolors.tam_io.tam_surface.TAMSurface method*), 36
clear() (*tamcolors.tam_io.uni_drivers.UNIUtilitiesDriver method*), 38
clear() (*tamcolors.tam_io.win_drivers.WINUtilitiesDriver method*), 39
close() (*tamcolors.utils.tcp.TCPBase method*), 67
close() (*tamcolors.utils.tcp.TCPObjectConnector method*), 68
close() (*tamcolors.utils.tcp.TCPObjectWrapper method*), 68
close() (*tamcolors.utils.tcp.TCPReceiver method*), 68
close_sound() (*tamcolors.tam_io.any_drivers.ANYSoundDriver*

method), 24
close_sound() (*tamcolors.tam_io.io_tam.IO method*), 25
close_sound() (*tamcolors.tam_io.io_tam.RawIO method*), 27
close_sound() (*tamcolors.tam_io.null_drivers.NULLSoundDriver method*), 31
close_sound() (*tamcolors.tam_io.tam_drivers.SoundDriver method*), 34
close_sound() (*tamcolors.tam_io.win_drivers.WINSoundDriver method*), 39
Clouds (*class in tamcolors.examples.clouds*), 14
code_to_key() (*tamcolors.tam_io.tam_keys.Keyboard method*), 36
code_to_key_state() (*tamcolors.tam_io.tam_keys.Keyboard method*), 36
Color (*class in tamcolors.tam_io.tam_colors*), 31
color_change_driver_operational() (*tamcolors.tam_io.io_tam.RawIO method*), 27
color_changer_driver_operational() (*tamcolors.tam_io.io_tam.IO method*), 25
color_changer_driver_operational() (*tamcolors.tam_io.io_tam.RawIO method*), 28
color_driver_operational() (*tamcolors.tam_io.io_tam.IO method*), 25
color_driver_operational() (*tamcolors.tam_io.io_tam.RawIO method*), 28
ColorChangerDriver (*class in tamcolors.tam_io.tam_drivers*), 32
ColorDriver (*class in tamcolors.tam_io.tam_drivers*), 33
ColorTests (*class in tamcolors.tests.tam_io_tests.tam_colors_tests*), 49
compress() (*in module tamcolors.utils.compress*), 65
CompressTests (*class in tamcolors.tests.utils_tests.compress_tests*), 61
copy() (*tamcolors.tam_io.tam_surface.TAMSurface method*), 36
critical() (*in module tamcolors.utils.log*), 66

D

debug() (*in module tamcolors.utils.log*), 66
decompress() (*in module tamcolors.utils.compress*), 65
decrypt() (*tamcolors.utils.encryption.Encryption method*), 65
disable_logging() (*in module tamcolors.utils.log*), 66
done() (*tamcolors.tam.tam_loop.TAMFrame method*), 18
done() (*tamcolors.tam.tam_loop.TAMLoop method*), 18
done() (*tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method*), 19
done() (*tamcolors.tam.tam_loop_io_tcp_handler.TAMLoopIOTCPHandle method*), 21
done() (*tamcolors.tam.tam_loop_receiver.TAMLoopReceiver method*), 21
done() (*tamcolors.tam.tam_loop_tcp_receiver.TAMLoopTCPReceiver method*), 22
done() (*tamcolors.tam_io.io_tam.IO method*), 25
done() (*tamcolors.tam_io.io_tam.RawIO method*), 28
done() (*tamcolors.tam_io.tam_drivers.TAMDriver method*), 35
done() (*tamcolors.tam_io.uni_drivers.UNIUtilitiesDriver method*), 38
done() (*tamcolors.tam_io.win_drivers.WINFullColorDriver method*), 38
done() (*tamcolors.tam_io.win_drivers.WINUtilitiesDriver method*), 40
done() (*tamcolors.tam_tools.tam_film.TAMFilm method*), 41
done() (*tamcolors.tam_tools.tam_text_box.TAMTextBox method*), 47
draw() (*tamcolors.examples.alpha.TAMAlpha method*), 13
draw() (*tamcolors.examples.clouds.Clouds method*), 14
draw() (*tamcolors.examples.colors.TAMCOLORS method*), 14
draw() (*tamcolors.examples.host_multi_player.HostMultiPlayer method*), 14
draw() (*tamcolors.examples.icon.BootLogo method*), 15
draw() (*tamcolors.examples.rgb_color.RGBCOLOR method*), 15
draw() (*tamcolors.examples.tabletennis.TableTennis method*), 16
draw() (*tamcolors.examples.tam_key_manager.TAMKeyManager method*), 16
draw() (*tamcolors.examples.tam_keys.TAMKeys method*), 16
draw() (*tamcolors.examples.tam_list_surface.TAMListSurface method*), 16
draw() (*tamcolors.examples.tam_loop.TAMLoopHelloWorld method*), 17
draw() (*tamcolors.examples.tam_print.TAMPrint method*), 17
draw() (*tamcolors.examples.tam_text_box.TAMPrint method*), 17
draw() (*tamcolors.tam.tam_loop.TAMFrame method*), 18
draw() (*tamcolors.tam_io.ansi_256_drivers.ANSI256ColorDriver*

method), 22
draw() (tamcolors.tam_io.ansi_true_color_drivers.ANSITrueFullColorDriver method), 23
draw() (tamcolors.tam_io.any_drivers.ANYFullColorDriver method), 23
draw() (tamcolors.tam_io.io_tam.IO method), 25
draw() (tamcolors.tam_io.io_tam.RawIO method), 28
draw() (tamcolors.tam_io.tam_drivers.ColorDriver method), 33
draw() (tamcolors.tam_io.win_drivers.WINFullColorDriver method), 38
draw() (tamcolors.tam_tools.tam_menu.TAMButtonRule method), 44
draw() (tamcolors.tam_tools.tam_menu.TAMMenu method), 44
draw() (tamcolors.tam_tools.tam_menu.TAMTextBoxButton method), 45
draw() (tamcolors.tam_tools.tam_menu.TAMTextButton method), 45
draw() (tamcolors.tam_tools.tam_text_box.TAMTextBox method), 47
draw_onto() (tamcolors.tam_io.tam_surface.TAMSurface method), 36
DummyCache (class in tamcolors.tests.utils_tests.immutable_cache_tests), 62
dumps() (tamcolors.utils.object_packer.ObjectPackerJson method), 66

E

enable_console_keys() (tamcolors.tam_io.io_tam.IO method), 25
enable_console_keys() (tamcolors.tam_io.io_tam.RawIO method), 28
enable_console_keys() (tamcolors.tam_io.tam_drivers.KeyDriver method), 34
enable_event_bus() (tamcolors.tam_io.io_tam.IO method), 25
enable_event_bus() (tamcolors.tam_io.io_tam.RawIO method), 28
enable_key_state_mode() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 19
enable_key_state_mode() (tamcolors.tam_io.io_tam.IO method), 25
enable_key_state_mode() (tamcolors.tam_io.io_tam.RawIO method), 28
enable_logging() (in module tamcolors.utils.log), 66
enable_slow_tests() (in module tamcolors.tests.test_utils), 64
encrypt() (tamcolors.utils.encryption.Encryption method), 65
encrypt_with_public_key() (tamcolors.utils.encryption.Encryption method), 65
Encryption (class in tamcolors.utils.encryption), 65
EncryptionError, 65
EncryptionTests (class in tamcolors.tests.utils_tests.encryption_tests), 61
error() (in module tamcolors.utils.log), 66

F

FastHandObjectPacker (class in tamcolors.utils.object_packer), 66
FastHandObjectPackerTests (class in tamcolors.tests.utils_tests.object_packer_tests), 62
first_msg_id() (tamcolors.utils.log.Log method), 66
format_message() (in module tamcolors.utils.log), 66
frame_done() (tamcolors.tam.tam_loop.TAMFrame method), 18
freeze_handler() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 19
from_bytes() (tamcolors.tam_io.tam_colors.Color class method), 31
from_bytes() (tamcolors.tam_io.tam_colors.RGBA class method), 32
from_bytes() (tamcolors.tam_io.tam_surface.TAMSurface class method), 36
from_bytes() (tamcolors.utils.object_packer.FastHandObjectPacker class method), 66
FullColorDriver (class in tamcolors.tam_io.tam_drivers), 33

G

g (tamcolors.tam_io.tam_colors.RGBA attribute), 32
generate_identifier_bytes() (in module tamcolors.utils.identifier), 65
get() (tamcolors.tam_tools.tam_film.TAMFilm method), 41
get_action() (tamcolors.tam_tools.tam_menu.TAMButtonRule method), 44
get_action() (tamcolors.tam_tools.tam_menu.TAMTextBoxButton method), 45
get_action() (tamcolors.tam_tools.tam_menu.TAMTextButton method), 45

get_address() (tamcolors.utils.tcp.TCPBase method), 67
get_all_drivers() (tamcolors.tam_io.tam_identifier.TAMIdentifier method), 35
get_all_receiver_names() (tamcolors.tam.tam_loop.TAMLoop method), 18
get_any_io() (in module tamcolors.tests.tam_io_tests.any_drivers_tests), 48
get_buttons() (tamcolors.tam_tools.tam_menu.TAMMenu method), 44
get_call_key() (tamcolors.tam_tools.tam_menu.TAMMenu method), 44
get_char() (tamcolors.tam_tools.tam_text_box.TAMTextBox method), 47
get_circular() (tamcolors.tam_tools.tam_film.TAMFilm method), 42
get_clock() (tamcolors.tam_tools.tam_color_palette.TAMCycleColor method), 41
get_color() (tamcolors.tam_tools.tam_color_palette.TAMColorPalette method), 40
get_color() (tamcolors.tam_tools.tam_color_palette.TAMDefaultColor method), 41
get_color_16() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 19
get_color_16() (tamcolors.tam_io.io_tam.IO method), 25
get_color_16() (tamcolors.tam_io.io_tam.RawIO method), 28
get_color_16() (tamcolors.tam_io.tam_drivers.ColorChangerDriver method), 32
get_color_16_pal_256() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 19
get_color_16_pal_256() (tamcolors.tam_io.io_tam.IO method), 25
get_color_16_pal_256() (tamcolors.tam_io.io_tam.RawIO method), 28
get_color_16_pal_256() (tamcolors.tam_io.tam_drivers.ColorChangerDriver method), 32
get_color_2() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 19
get_color_2() (tamcolors.tam_io.io_tam.IO method), 25
get_color_2() (tamcolors.tam_io.io_tam.RawIO method), 28
get_color_2() (tamcolors.tam_io.tam_drivers.ColorChangerDriver method), 32
get_color_256() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 19
get_color_256() (tamcolors.tam_io.io_tam.IO method), 25
get_color_256() (tamcolors.tam_io.io_tam.RawIO method), 28
get_color_256() (tamcolors.tam_io.tam_drivers.ColorChangerDriver method), 32
get_colors() (tamcolors.tam_tools.tam_text_box.TAMTextBox method), 47
get_connection() (tamcolors.utils.tcp.TCPObjectConnector method), 68
get_connection() (tamcolors.utils.tcp.TCPObjectWrapper method), 68
get_cross_rect() (tamcolors.tam_io.tam_surface.TAMSurface method), 37
get_data() (tamcolors.utils.tcp.TCPBase method), 67
get_defaults() (tamcolors.tam.tam_loop.TAMFrame method), 18
get_defaults() (tamcolors.tam_io.tam_surface.TAMSurface method), 37
get_dimensions() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 19
get_dimensions() (tamcolors.tam_io.any_drivers.ANYUtilitiesDriver method), 24
get_dimensions() (tamcolors.tam_io.io_tam.IO method), 25
get_dimensions() (tamcolors.tam_io.io_tam.RawIO method), 28
get_dimensions() (tamcolors.tam_io.null_drivers.NULLUtilitiesDriver method), 31
get_dimensions() (tamcolors.tam_io.tam_drivers.UtilitiesDriver method), 35
get_dimensions() (tamcolors.tam_io.tam_surface.TAMSurface method), 37

get_dimensions() (tamcolors.tam_io.uni_drivers.UNIUtilitiesDriver method), 38
get_dimensions() (tamcolors.tam_io.win_drivers.WINUtilitiesDriver method), 40
get_dimensions() (tamcolors.tam_tools.tam_text_box.TAMTextBox method), 47
get_event() (tamcolors.tam_io.io_tam.IO method), 25
get_event() (tamcolors.tam_io.io_tam.RawIO method), 28
get_fps() (tamcolors.tam.tam_loop.TAMFrame method), 18
get_from_raw_spot() (tamcolors.tam_io.tam_surface.TAMSurface method), 37
get_full_name() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 19
get_goto_map() (tamcolors.tam_tools.tam_menu.TAMMenu method), 44
get_handler() (tamcolors.tam.tam_loop_receiver.TAMLoopReceiver method), 21
get_handler() (tamcolors.tam.tam_loop_tcp_receiver.TAMLoopTCPReceiver method), 22
get_height_min_and_max() (tamcolors.tam.tam_loop.TAMFrame method), 18
get_host_connection() (tamcolors.utils.tcp.TCPReceiver method), 68
get_icon() (in module tamcolors.tam_tools.tam_icon), 42
get_identifier_bytes() (in module tamcolors.utils.identifier), 65
get_identifier_id() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 19
get_info_dict() (tamcolors.tam_io.io_tam.IO method), 25
get_info_dict() (tamcolors.tam_io.io_tam.RawIO method), 28
get_info_dict() (tamcolors.tam_io.tam_identifier.TAMIdentifier method), 35
get_io() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 20
get_io() (tamcolors.tam_io.tam_identifier.TAMIdentifier method), 35
get_key() (tamcolors.tam_io.any_drivers.ANYKeyDriver method), 24
get_key() (tamcolors.tam_io.io_tam.IO method), 26
get_key() (tamcolors.tam_io.io_tam.RawIO method), 28
get_key() (tamcolors.tam_io.null_drivers.NULLKeyDriver method), 30
get_key() (tamcolors.tam_io.tam_drivers.KeyDriver method), 34
get_key() (tamcolors.tam_io.win_drivers.WINKeyDriver method), 39
get_key_dict() (tamcolors.tam_io.any_drivers.ANYKeyDriver method), 24
get_key_dict() (tamcolors.tam_io.io_tam.IO method), 26
get_key_dict() (tamcolors.tam_io.io_tam.RawIO method), 28
get_key_dict() (tamcolors.tam_io.null_drivers.NULLKeyDriver class method), 30
get_key_dict() (tamcolors.tam_io.tam_drivers.KeyDriver method), 34
get_key_dict() (tamcolors.tam_io.tam_keys.Keyboard method), 36
get_key_dict() (tamcolors.tam_io.win_drivers.WINKeyDriver method), 39
get_key_list() (tamcolors.tam_io.tam_keys.Keyboard method), 36
get_key_spot_dict() (tamcolors.tam_io.tam_keys.Keyboard method), 36
get_key_state() (tamcolors.tam_tools.tam_key_manager.TAMKeyManager method), 43
get_key_state_dict() (tamcolors.tam_io.tam_keys.Keyboard method), 36
get_keyboard_name() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 20
get_keyboard_name() (tamcolors.tam_io.any_drivers.ANYKeyDriver method), 24
get_keyboard_name() (tamcolors.tam_io.io_tam.IO method), 26
get_keyboard_name() (tamcolors.tam_io.io_tam.RawIO method), 28
get_keyboard_name() (tamcolors.tam_io.null_drivers.NULLKeyDriver method), 30
get_keyboard_name() (tamcol-

ors.tam_io.tam_drivers.KeyDriver method), 34
get_keyboard_name() (*tamcolors.tam_io.win_drivers.WINKeyDriver method*), 39
get_loop_data() (*tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method*), 20
get_mode() (*tamcolors.tam_io.io_tam.IO method*), 26
get_mode() (*tamcolors.tam_io.io_tam.RawIO method*), 28
get_modes() (*tamcolors.tam_io.io_tam.IO method*), 26
get_modes() (*tamcolors.tam_io.io_tam.RawIO method*), 28
get_name() (*tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method*), 20
get_name() (*tamcolors.tam.tam_loop_receiver.TAMLoopReceiver method*), 21
get_name() (*tamcolors.tam_io.tam_identifier.TAMIdentifier method*), 35
get_name() (*tamcolors.tam_io.tam_keys.Keyboard method*), 36
get_on() (*tamcolors.tam_tools.tam_menu.TAMMenu method*), 45
get_other_data() (*tamcolors.utils.tcp.TCPConnection method*), 67
get_other_data() (*tamcolors.utils.tcp.TCPHost method*), 67
get_position() (*tamcolors.tam_tools.tam_menu.TAMButtonRule method*), 44
get_position() (*tamcolors.tam_tools.tam_menu.TAMTextBoxButton method*), 45
get_position() (*tamcolors.tam_tools.tam_menu.TAMTextButton method*), 46
get_printc_mode() (*tamcolors.tam_io.ansi_256_drivers.ANSI256ColorDriver method*), 22
get_printc_mode() (*tamcolors.tam_io.ansi_true_color_drivers.ANSITrueFullColorDriver method*), 23
get_printc_mode() (*tamcolors.tam_io.any_drivers.ANYFullColorDriver method*), 23
get_printc_mode() (*tamcolors.tam_io.io_tam.IO method*), 26
get_printc_mode() (*tamcolors.tam_io.io_tam.RawIO method*), 28
get_printc_mode() (*tamcolors.tam_io.null_drivers.NULLFullColorDriver method*), 30
get_printc_mode() (*tamcolors.tam_io.tam_drivers.ColorDriver method*), 33
get_printc_mode() (*tamcolors.tam_io.win_drivers.WINFullColorDriver method*), 38
get_raw_private_key() (*tamcolors.utils.encryption.Encryption method*), 65
get_raw_public_key() (*tamcolors.utils.encryption.Encryption method*), 65
get_raw_spot() (*tamcolors.tam_io.tam_surface.TAMSurface method*), 37
get_raw_surface() (*tamcolors.tam_io.tam_surface.TAMSurface method*), 37
get_raw_user_input() (*tamcolors.tam_tools.tam_key_manager.TAMKeyManager method*), 44
get_raw_user_input_2() (*tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManag method*), 56
get_raw_user_input_3() (*tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManag method*), 56
get_receiver_settings() (*tamcolors.tam.tam_loop.TAMLoop method*), 18
get_receiver_settings() (*tamcolors.tam.tam_loop_receiver.TAMLoopReceiver method*), 21
get_rule() (*tamcolors.tam_tools.tam_color_palette.TAMColorPalette method*), 40
get_running() (*tamcolors.tam.tam_loop_receiver.TAMLoopReceiver method*), 21
get_snapshot() (*tamcolors.tam_io.io_tam.IO method*), 26
get_snapshot() (*tamcolors.tam_io.io_tam.RawIO method*), 28
get_sound_length() (*tamcolors.tam_io.any_drivers.ANYSoundDriver method*), 24
get_sound_length() (*tamcolors.tam_io.io_tam.IO method*), 26
get_sound_length() (*tamcolors.tam_io.io_tam.RawIO method*), 29
get_sound_length() (*tamcolors.tam_io.null_drivers.NULLSoundDriver

method), 31
get_sound_length() (tamcolors.tam_io.tam_drivers.SoundDriver method), 34
get_sound_length() (tamcolors.tam_io.win_drivers.WINSoundDriver method), 39
get_sound_position() (tamcolors.tam_io.any_drivers.ANYSoundDriver method), 24
get_sound_position() (tamcolors.tam_io.io_tam.IO method), 26
get_sound_position() (tamcolors.tam_io.io_tam.RawIO method), 29
get_sound_position() (tamcolors.tam_io.null_drivers.NULLSoundDriver method), 31
get_sound_position() (tamcolors.tam_io.tam_drivers.SoundDriver method), 34
get_sound_position() (tamcolors.tam_io.win_drivers.WINSoundDriver method), 39
get_spot() (tamcolors.tam_io.tam_surface.TAMSurface method), 37
get_start_data() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 20
get_system() (tamcolors.tam_io.tam_identifier.TAMIdentifier method), 35
get_tcp_io() (in module tamcolors.tam_io.tcp_io), 37
get_text() (tamcolors.tam_tools.tam_text_box.TAMTextBox method), 47
get_user_id() (tamcolors.utils.tcp.TCPHost method), 67
get_user_input() (tamcolors.tam_tools.tam_key_manager.TAMKeyManager method), 44
get_user_input_generator() (tamcolors.tam_tools.tam_key_manager.TAMKeyManager method), 44
get_user_name() (tamcolors.utils.tcp.TCPHost method), 67
get_width_min_and_max() (tamcolors.tam.tam_loop.TAMFrame method), 18
get_win_io() (in module tamcolors.tests.tam_io_tests.win_drivers_tests), 52
get_y() (tamcolors.examples.tabletennis.Ball method), 15

H

has_alpha (tamcolors.tam_io.tam_colors.Color attribute), 31
HostMultiPlayer (class in tamcolors.examples.host_multi_player), 14

I

IdentifierTests (class in tamcolors.tests.utils_tests.identifier_tests), 61
identify() (tamcolors.tam_io.tam_identifier.TAMIdentifier class method), 35
ImmutableCache (class in tamcolors.utils.immutable_cache), 66
ImmutableCacheTests (class in tamcolors.tests.utils_tests.immutable_cache_tests), 62
info() (in module tamcolors.utils.log), 66
inputc() (tamcolors.tam_io.ansi_256_drivers.ANSI256ColorDriver method), 23
inputc() (tamcolors.tam_io.ansi_true_color_drivers.ANSITrueFullColo method), 23
inputc() (tamcolors.tam_io.any_drivers.ANYFullColorDriver method), 23
inputc() (tamcolors.tam_io.io_tam.IO method), 26
inputc() (tamcolors.tam_io.io_tam.RawIO method), 29
inputc() (tamcolors.tam_io.null_drivers.NULLFullColorDriver method), 30
inputc() (tamcolors.tam_io.tam_drivers.ColorDriver method), 33
inputc() (tamcolors.tam_io.win_drivers.WINFullColorDriver method), 38
IO (class in tamcolors.tam_io.io_tam), 25
IOTAMTest (class in tamcolors.tests.tam_io_tests.io_tam_tests), 48
is_console_cursor_enabled() (tamcolors.tam_io.io_tam.IO method), 26
is_console_cursor_enabled() (tamcolors.tam_io.io_tam.RawIO method), 29
is_console_keys_enabled() (tamcolors.tam_io.io_tam.IO method), 26
is_console_keys_enabled() (tamcolors.tam_io.io_tam.RawIO method), 29
is_default (tamcolors.tam_io.tam_colors.RGBA attribute), 32
is_event_bus_enabled() (tamcolors.tam_io.io_tam.IO method), 26
is_event_bus_enabled() (tamcolors.tam_io.io_tam.RawIO method), 29
is_key_state_mode_enabled() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 20

is_key_state_mode_enabled() (tamcolors.tam_io.io_tam.IO method), 26
is_key_state_mode_enabled() (tamcolors.tam_io.io_tam.RawIO method), 29
is_open() (tamcolors.utils.tcp.TCPBase method), 67
is_open() (tamcolors.utils.tcp.TCPObjectConnector method), 68
is_open() (tamcolors.utils.tcp.TCPObjectWrapper method), 68
is_running() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 20
is_running() (tamcolors.tam.tam_loop_io_tcp_handler.TAMLoopIOTCPHandler method), 21
is_running() (tamcolors.tam_io.io_tam.IO method), 26
is_running() (tamcolors.tam_io.io_tam.RawIO method), 29
is_slow_tests_enabled() (in module tamcolors.tests.test_utils), 64
is_sound_playing() (tamcolors.tam_io.any_drivers.ANYSoundDriver method), 24
is_sound_playing() (tamcolors.tam_io.io_tam.IO method), 26
is_sound_playing() (tamcolors.tam_io.io_tam.RawIO method), 29
is_sound_playing() (tamcolors.tam_io.null_drivers.NULLSoundDriver method), 31
is_sound_playing() (tamcolors.tam_io.tam_drivers.SoundDriver method), 34
is_sound_playing() (tamcolors.tam_io.win_drivers.WINSoundDriver method), 39

K

key_driver_operational() (tamcolors.tam_io.io_tam.IO method), 26
key_driver_operational() (tamcolors.tam_io.io_tam.RawIO method), 29
key_present() (tamcolors.tam_tools.tam_color_palette.TAMColorPalette method), 40
key_to_spot() (tamcolors.tam_io.tam_keys.Keyboard method), 36
Keyboard (class in tamcolors.tam_io.tam_keys), 36
KeyDriver (class in tamcolors.tam_io.tam_drivers), 33

L

lap() (tamcolors.utils.timer.Timer method), 68
last_msg_id() (tamcolors.utils.log.Log method), 66
LastReceivedCache (class in tamcolors.utils.transport_optimizer), 69
LastSentCache (class in tamcolors.utils.transport_optimizer), 69
load_data() (in module tamcolors.utils.object_packer), 67
load_int() (in module tamcolors.utils.object_packer), 67
load_tests() (in module tamcolors.tests.all_tests), 64
loads() (tamcolors.utils.object_packer.ObjectPackerJson method), 67
Log (class in tamcolors.utils.log), 66
log() (tamcolors.utils.log.Log method), 66
LogTests (class in tamcolors.tests.utils_tests.log_tests), 62

M

make_sandy_check_key() (tamcolors.utils.encryption.Encryption static method), 65
make_surface_ready() (tamcolors.tam.tam_loop.TAMFrame method), 18
make_tam_str() (in module tamcolors.tam_tools.tam_str), 47
MakeTAMStrTests (class in tamcolors.tests.tam_tools_tests.tam_str_tests), 59
mode_16 (tamcolors.tam_io.tam_colors.Color attribute), 31
mode_16_pal_256 (tamcolors.tam_io.tam_colors.Color attribute), 31
mode_2 (tamcolors.tam_io.tam_colors.Color attribute), 31
mode_256 (tamcolors.tam_io.tam_colors.Color attribute), 31
mode_rgb (tamcolors.tam_io.tam_colors.Color attribute), 31
multiple_processes_helper() (tamcolors.tests.test_multi_task_helper.MultiTaskHelper method), 64
multiple_threads_helper() (tamcolors.tests.test_multi_task_helper.MultiTaskHelper method), 64
MultiTaskHelper (class in tamcolors.tests.test_multi_task_helper), 64
MultiTaskHelperTests (class in tamcolors.tests.tests_tests.test_multi_task_helper_tests), 61

N

NoEncryptionTests (class in tamcolors.tests.utils_tests.encryption_tests), 61

NULLFullColorDriver (*class in tamcolors.tam_io.null_drivers*), 30
NULLKeyDriver (*class in tamcolors.tam_io.null_drivers*), 30
NULLSoundDriver (*class in tamcolors.tam_io.null_drivers*), 30
NULLUtilitiesDriver (*class in tamcolors.tam_io.null_drivers*), 31

O

ObjectPackerFunctionsTests (*class in tamcolors.tests.utils_tests.object_packer_tests*), 62
ObjectPackerJson (*class in tamcolors.utils.object_packer*), 66
ObjectPackerJsonError, 67
ObjectPackerJsonTests (*class in tamcolors.tests.utils_tests.object_packer_tests*), 63
off() (*tamcolors.tam_tools.tam_menu.TAMButtonRule method*), 44
off() (*tamcolors.tam_tools.tam_menu.TAMTextBoxButton method*), 45
off() (*tamcolors.tam_tools.tam_menu.TAMTextButton method*), 46
offset_sleep() (*tamcolors.utils.timer.Timer method*), 68
on() (*tamcolors.tam_tools.tam_menu.TAMButtonRule method*), 44
on() (*tamcolors.tam_tools.tam_menu.TAMTextBoxButton method*), 45
on() (*tamcolors.tam_tools.tam_menu.TAMTextButton method*), 46
open_sound() (*tamcolors.tam_io.any_drivers.ANYSoundDriver method*), 24
open_sound() (*tamcolors.tam_io.io_tam.IO method*), 26
open_sound() (*tamcolors.tam_io.io_tam.RawIO method*), 29
open_sound() (*tamcolors.tam_io.null_drivers.NULLSoundDriver method*), 31
open_sound() (*tamcolors.tam_io.tam_drivers.SoundDriver method*), 34
open_sound() (*tamcolors.tam_io.win_drivers.WINSoundDriver method*), 39

P

pause_sound() (*tamcolors.tam_io.any_drivers.ANYSoundDriver method*), 24
pause_sound() (*tamcolors.tam_io.io_tam.IO method*), 26
pause_sound() (*tamcolors.tam_io.io_tam.RawIO method*), 29
pause_sound() (*tamcolors.tam_io.null_drivers.NULLSoundDriver method*), 31
pause_sound() (*tamcolors.tam_io.tam_drivers.SoundDriver method*), 34
pause_sound() (*tamcolors.tam_io.win_drivers.WINSoundDriver method*), 39
peak() (*tamcolors.tam_tools.tam_film.TAMFilm method*), 42
place_color_over() (*tamcolors.tam_io.tam_colors.Color method*), 31
play_sound() (*tamcolors.tam_io.any_drivers.ANYSoundDriver method*), 24
play_sound() (*tamcolors.tam_io.io_tam.IO method*), 26
play_sound() (*tamcolors.tam_io.io_tam.RawIO method*), 29
play_sound() (*tamcolors.tam_io.null_drivers.NULLSoundDriver method*), 31
play_sound() (*tamcolors.tam_io.tam_drivers.SoundDriver method*), 34
play_sound() (*tamcolors.tam_io.win_drivers.WINSoundDriver method*), 39
pop() (*tamcolors.tam_tools.tam_film.TAMFilm method*), 42
pop_frame_stack() (*tamcolors.tam.tam_loop.TAMLoop method*), 19
prime_event_bus() (*tamcolors.tam_io.io_tam.IO method*), 26
prime_event_bus() (*tamcolors.tam_io.io_tam.RawIO method*), 29
printc() (*tamcolors.tam_io.ansi_256_drivers.ANSI256ColorDriver method*), 23
printc() (*tamcolors.tam_io.ansi_true_color_drivers.ANSITrueFullColo method*), 23
printc() (*tamcolors.tam_io.any_drivers.ANYFullColorDriver method*), 23
printc() (*tamcolors.tam_io.io_tam.IO method*), 27
printc() (*tamcolors.tam_io.io_tam.RawIO method*), 29
printc() (*tamcolors.tam_io.null_drivers.NULLFullColorDriver method*), 30
printc() (*tamcolors.tam_io.tam_drivers.ColorDriver method*), 33

printc() (*tamcolors.tam_io.win_drivers.WINFullColorDriver method*), 39
process_runner() (*in module tamcolors.tests.test_multi_task_helper*), 64
pump_keys() (*tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method*), 20

R

r (*tamcolors.tam_io.tam_colors.RGBA attribute*), 32
Racket (*class in tamcolors.examples.tabletennis*), 15
RawIO (*class in tamcolors.tam_io.io_tam*), 27
RAWIOTest (*class in tamcolors.tests.tam_io_tests.io_tam_tests*), 48
read() (*tamcolors.utils.log.Log method*), 66
remove_receiver() (*tamcolors.tam.tam_loop.TAMLoop method*), 19
replace_alpha_chars() (*tamcolors.tam_io.tam_surface.TAMSurface method*), 37
reset_colors_to_console_defaults() (*tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method*), 20
reset_colors_to_console_defaults() (*tamcolors.tam_io.io_tam.IO method*), 27
reset_colors_to_console_defaults() (*tamcolors.tam_io.io_tam.RawIO method*), 29
rest_sound() (*tamcolors.tam_io.io_tam.IO method*), 27
rest_sound() (*tamcolors.tam_io.io_tam.RawIO method*), 29
RGBA (*class in tamcolors.tam_io.tam_colors*), 32
RGBATests (*class in tamcolors.tests.tam_io_tests.tam_colors_tests*), 49
RGBCOLOR (*class in tamcolors.examples.rgb_color*), 15
run() (*in module tamcolors.examples.alpha*), 13
run() (*in module tamcolors.examples.clouds*), 14
run() (*in module tamcolors.examples.colors*), 14
run() (*in module tamcolors.examples.connection_loopback*), 14
run() (*in module tamcolors.examples.connection_multi_player*), 14
run() (*in module tamcolors.examples.host_loopback*), 14
run() (*in module tamcolors.examples.host_multi_player*), 15
run() (*in module tamcolors.examples.icon*), 15
run() (*in module tamcolors.examples.rgb_color*), 15
run() (*in module tamcolors.examples.sandy_check*), 15
run() (*in module tamcolors.examples.tabletennis*), 16
run() (*in module tamcolors.examples.tam_key_manager*), 16
run() (*in module tamcolors.examples.tam_keys*), 16
run() (*in module tamcolors.examples.tam_list_surface*), 16
run() (*in module tamcolors.examples.tam_loop*), 17
run() (*in module tamcolors.examples.tam_print*), 17
run() (*in module tamcolors.examples.tam_text_box*), 17
run() (*tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method*), 20
run_action() (*tamcolors.tam_tools.tam_menu.TAMButtonRule method*), 44
run_action() (*tamcolors.tam_tools.tam_menu.TAMTextBoxButton method*), 45
run_action() (*tamcolors.tam_tools.tam_menu.TAMTextButton method*), 46
run_application() (*tamcolors.tam.tam_loop.TAMLoop class method*), 19
run_tcp_connection() (*in module tamcolors.tam_io.tcp_io*), 37
run_with_profiler() (*tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method*), 20

S

save_data (*in module tamcolors.utils.object_packer*), 67
save_int (*in module tamcolors.utils.object_packer*), 67
send_data() (*tamcolors.utils.tcp.TCPBase method*), 67
set() (*tamcolors.tam_tools.tam_film.TAMFilm method*), 42
set_action() (*tamcolors.tam_tools.tam_menu.TAMButtonRule method*), 44
set_action() (*tamcolors.tam_tools.tam_menu.TAMTextBoxButton method*), 45
set_action() (*tamcolors.tam_tools.tam_menu.TAMTextButton method*), 46
set_char() (*tamcolors.tam_tools.tam_text_box.TAMTextBox method*), 47
set_circular() (*tamcolors.tam_tools.tam_film.TAMFilm method*), 42
set_clock() (*tamcolors.tam_tools.tam_color_palette.TAMCycleColor method*), 41

set_color() (tamcolors.tam_tools.tam_color_palette.TAMColorPalette method), 40
set_color() (tamcolors.tam_tools.tam_color_palette.TAMDefaultColors method), 41
set_color_16() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 20
set_color_16() (tamcolors.tam_io.io_tam.IO method), 27
set_color_16() (tamcolors.tam_io.io_tam.RawIO method), 29
set_color_16() (tamcolors.tam_io.tam_drivers.ColorChangerDriver method), 32
set_color_16_pal_256() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 20
set_color_16_pal_256() (tamcolors.tam_io.io_tam.IO method), 27
set_color_16_pal_256() (tamcolors.tam_io.io_tam.RawIO method), 29
set_color_16_pal_256() (tamcolors.tam_io.tam_drivers.ColorChangerDriver method), 33
set_color_2() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 20
set_color_2() (tamcolors.tam_io.io_tam.IO method), 27
set_color_2() (tamcolors.tam_io.io_tam.RawIO method), 29
set_color_2() (tamcolors.tam_io.tam_drivers.ColorChangerDriver method), 33
set_color_256() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 20
set_color_256() (tamcolors.tam_io.io_tam.IO method), 27
set_color_256() (tamcolors.tam_io.io_tam.RawIO method), 29
set_color_256() (tamcolors.tam_io.tam_drivers.ColorChangerDriver method), 33
set_colors() (tamcolors.tam_tools.tam_color_palette.TAMCycleColors method), 41
set_colors() (tamcolors.tam_tools.tam_text_box.TAMTextBox method), 47
set_defaults_and_clear() (tamcolors.tam_io.tam_surface.TAMSurface method), 37
set_dimensions_and_clear() (tamcolors.tam_io.tam_surface.TAMSurface method), 37
set_loop_data() (tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method), 20
set_mode() (tamcolors.tam_io.ansi_256_drivers.ANSI256ColorDriver method), 23
set_mode() (tamcolors.tam_io.ansi_true_color_drivers.ANSITrueFullColorDriver method), 23
set_mode() (tamcolors.tam_io.io_tam.IO method), 27
set_mode() (tamcolors.tam_io.io_tam.RawIO method), 30
set_mode() (tamcolors.tam_io.tam_drivers.ColorChangerDriver method), 33
set_mode() (tamcolors.tam_io.tam_drivers.ColorDriver method), 33
set_mode() (tamcolors.tam_io.win_drivers.WINFullColorDriver method), 39
set_position() (tamcolors.tam_tools.tam_menu.TAMButtonRule method), 44
set_position() (tamcolors.tam_tools.tam_menu.TAMTextBoxButton method), 45
set_position() (tamcolors.tam_tools.tam_menu.TAMTextButton method), 46
set_receiver_settings() (tamcolors.tam.tam_loop_receiver.TAMLoopReceiver method), 21
set_rule() (tamcolors.tam_tools.tam_color_palette.TAMColorPalette method), 40
set_sound_position() (tamcolors.tam_io.any_drivers.ANYSoundDriver method), 24
set_sound_position() (tamcolors.tam_io.io_tam.IO method), 27
set_sound_position() (tamcolors.tam_io.io_tam.RawIO method), 30
set_sound_position() (tamcolors.tam_io.null_drivers.NULLSoundDriver method), 31
set_sound_position() (tamcolors.tam_io.tam_drivers.SoundDriver method), 34
set_sound_position() (tamcol-

ors.tam_io.win_drivers.WINSoundDriver method), 39
set_spot() (*tamcolors.tam_io.tam_surface.TAMSurface method*), 37
set_start_data() (*tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method*), 20
set_tam_color_defaults() (*tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method*), 20
set_tam_color_defaults() (*tamcolors.tam_io.io_tam.IO method*), 27
set_tam_color_defaults() (*tamcolors.tam_io.io_tam.RawIO method*), 30
show_console_cursor() (*tamcolors.tam_io.any_drivers.ANYUtilitiesDriver method*), 24
show_console_cursor() (*tamcolors.tam_io.io_tam.IO method*), 27
show_console_cursor() (*tamcolors.tam_io.io_tam.RawIO method*), 30
show_console_cursor() (*tamcolors.tam_io.tam_drivers.UtilitiesDriver method*), 35
show_console_cursor() (*tamcolors.tam_io.uni_drivers.UNIUtilitiesDriver method*), 38
show_console_cursor() (*tamcolors.tam_io.win_drivers.WINUtilitiesDriver method*), 40
silent_key_state() (*tamcolors.tam_tools.tam_key_manager.TAMKeyManager method*), 44
simple_menu_builder() (*tamcolors.tam_tools.tam_menu.TAMMenu static method*), 45
slide() (*tamcolors.tam_tools.tam_film.TAMFilm method*), 42
slow_test() (*in module tamcolors.tests.test_utils*), 64
sound_driver_operational() (*tamcolors.tam_io.io_tam.IO method*), 27
sound_driver_operational() (*tamcolors.tam_io.io_tam.RawIO method*), 30
SoundDriver (*class in tamcolors.tam_io.tam_drivers*), 34
split_code_dict() (*tamcolors.tam_io.tam_keys.Keyboard static method*), 36
spot_to_key() (*tamcolors.tam_io.tam_keys.Keyboard method*), 36
stability_check() (*in module tamcolors.tests.all_tests*), 64
stable() (*tamcolors.tam_io.tam_identifier.TAMIdentifier method*), 36
start() (*tamcolors.tam_io.ansi_256_drivers.ANSI256ColorDriver method*), 23
start() (*tamcolors.tam_io.ansi_true_color_drivers.ANSITrueFullColorI method*), 23
start() (*tamcolors.tam_io.io_tam.IO method*), 27
start() (*tamcolors.tam_io.io_tam.RawIO method*), 30
start() (*tamcolors.tam_io.tam_drivers.TAMDriver method*), 35
start() (*tamcolors.tam_io.win_drivers.WINUtilitiesDriver method*), 40
start_from_bytes() (*tamcolors.utils.object_packer.FastHandObjectPacker class method*), 66

T

TableTennis (*class in tamcolors.examples.tabletennis*), 15
tam_fade_in() (*in module tamcolors.tam_tools.tam_fade*), 41
tam_list_surface() (*in module tamcolors.tam_tools.tam_list_surface*), 44
tam_print() (*in module tamcolors.tam_tools.tam_print*), 46
TAMAlpha (*class in tamcolors.examples.alpha*), 13
TAMButtonRule (*class in tamcolors.tam_tools.tam_menu*), 44
TAMColorPalette (*class in tamcolors.tam_tools.tam_color_palette*), 40
TAMColorPaletteError, 41
TAMColorPaletteRule (*class in tamcolors.tam_tools.tam_color_palette*), 41
TAMColorPaletteTests (*class in tamcolors.tests.tam_tools_tests.tam_color_palette_tests*), 54
TAMCOLORS (*class in tamcolors.examples.colors*), 14
tamcolors (*module*), 69
tamcolors.examples (*module*), 17
tamcolors.examples.alpha (*module*), 13
tamcolors.examples.clouds (*module*), 14
tamcolors.examples.colors (*module*), 14
tamcolors.examples.connection_loopback (*module*), 14
tamcolors.examples.connection_multi_player (*module*), 14
tamcolors.examples.host_loopback (*module*), 14
tamcolors.examples.host_multi_player (*module*), 14
tamcolors.examples.icon (*module*), 15
tamcolors.examples.rgb_color (*module*), 15
tamcolors.examples.sandy_check (*module*), 15

tamcolors.examples.tabletennis (*module*), 15
tamcolors.examples.tam_key_manager (*module*), 16
tamcolors.examples.tam_keys (*module*), 16
tamcolors.examples.tam_list_surface (*module*), 16
tamcolors.examples.tam_loop (*module*), 17
tamcolors.examples.tam_print (*module*), 17
tamcolors.examples.tam_text_box (*module*), 17
tamcolors.tam (*module*), 22
tamcolors.tam.tam_loop (*module*), 18
tamcolors.tam.tam_loop_io_handler (*module*), 19
tamcolors.tam.tam_loop_io_tcp_handler (*module*), 21
tamcolors.tam.tam_loop_receiver (*module*), 21
tamcolors.tam.tam_loop_tcp_receiver (*module*), 22
tamcolors.tam_basic (*module*), 22
tamcolors.tam_c (*module*), 22
tamcolors.tam_io (*module*), 40
tamcolors.tam_io.ansi_256_drivers (*module*), 22
tamcolors.tam_io.ansi_true_color_drivers (*module*), 23
tamcolors.tam_io.any_drivers (*module*), 23
tamcolors.tam_io.io_tam (*module*), 25
tamcolors.tam_io.null_drivers (*module*), 30
tamcolors.tam_io.tam_colors (*module*), 31
tamcolors.tam_io.tam_drivers (*module*), 32
tamcolors.tam_io.tam_identifier (*module*), 35
tamcolors.tam_io.tam_keys (*module*), 36
tamcolors.tam_io.tam_surface (*module*), 36
tamcolors.tam_io.tcp_io (*module*), 37
tamcolors.tam_io.uni_drivers (*module*), 38
tamcolors.tam_io.win_drivers (*module*), 38
tamcolors.tam_tools (*module*), 47
tamcolors.tam_tools.tam_color_palette (*module*), 40
tamcolors.tam_tools.tam_fade (*module*), 41
tamcolors.tam_tools.tam_film (*module*), 41
tamcolors.tam_tools.tam_icon (*module*), 42
tamcolors.tam_tools.tam_key_manager (*module*), 43
tamcolors.tam_tools.tam_list_surface (*module*), 44
tamcolors.tam_tools.tam_menu (*module*), 44
tamcolors.tam_tools.tam_placing (*module*), 46
tamcolors.tam_tools.tam_print (*module*), 46
tamcolors.tam_tools.tam_str (*module*), 47
tamcolors.tam_tools.tam_text_box (*module*), 47
tamcolors.tests (*module*), 65
tamcolors.tests.all_tests (*module*), 64
tamcolors.tests.tam_basic_tests (*module*), 48
tamcolors.tests.tam_io_tests (*module*), 52
tamcolors.tests.tam_io_tests.any_drivers_tests (*module*), 48
tamcolors.tests.tam_io_tests.io_tam_tests (*module*), 48
tamcolors.tests.tam_io_tests.tam_colors_tests (*module*), 49
tamcolors.tests.tam_io_tests.tam_keys_tests (*module*), 49
tamcolors.tests.tam_io_tests.tam_surface_tests (*module*), 49
tamcolors.tests.tam_io_tests.win_drivers_tests (*module*), 51
tamcolors.tests.tam_tests (*module*), 54
tamcolors.tests.tam_tests.tam_loop_tcp_receiver_tes (*module*), 53
tamcolors.tests.tam_tests.tam_loop_tests (*module*), 53
tamcolors.tests.tam_tools_tests (*module*), 60
tamcolors.tests.tam_tools_tests.tam_color_palette_t (*module*), 54
tamcolors.tests.tam_tools_tests.tam_fade_tests (*module*), 55
tamcolors.tests.tam_tools_tests.tam_film_tests (*module*), 55
tamcolors.tests.tam_tools_tests.tam_key_manager_tes (*module*), 56
tamcolors.tests.tam_tools_tests.tam_list_surface_te (*module*), 57
tamcolors.tests.tam_tools_tests.tam_menu_tests (*module*), 57
tamcolors.tests.tam_tools_tests.tam_placing_tests (*module*), 59
tamcolors.tests.tam_tools_tests.tam_print_tests (*module*), 59
tamcolors.tests.tam_tools_tests.tam_str_tests (*module*), 59
tamcolors.tests.tam_tools_tests.tam_text_box_tests (*module*), 60
tamcolors.tests.test_multi_task_helper (*module*), 64
tamcolors.tests.test_utils (*module*), 64
tamcolors.tests.tests_tests (*module*), 61
tamcolors.tests.tests_tests.test_multi_task_helper_ (*module*), 61
tamcolors.tests.utils_tests (*module*), 64

tamcolors.tests.utils_tests.compress_tests (*module*), 61
tamcolors.tests.utils_tests.encryption_tests (*module*), 61
tamcolors.tests.utils_tests.identifier_tests (*module*), 61
tamcolors.tests.utils_tests.immutable_cache_tests (*module*), 62
tamcolors.tests.utils_tests.log_tests (*module*), 62
tamcolors.tests.utils_tests.object_packer_tests (*module*), 62
tamcolors.tests.utils_tests.tcp_tests (*module*), 63
tamcolors.tests.utils_tests.timer_tests (*module*), 63
tamcolors.tests.utils_tests.transport_optimizer_tests (*module*), 64
tamcolors.utils (*module*), 69
tamcolors.utils.compress (*module*), 65
tamcolors.utils.encryption (*module*), 65
tamcolors.utils.identifier (*module*), 65
tamcolors.utils.immutable_cache (*module*), 66
tamcolors.utils.log (*module*), 66
tamcolors.utils.object_packer (*module*), 66
tamcolors.utils.tcp (*module*), 67
tamcolors.utils.timer (*module*), 68
tamcolors.utils.transport_optimizer (*module*), 69
TAMColorTests (*class in tamcolors.tests.tam_io_tests.tam_colors_tests*), 49
TAMCycleColor (*class in tamcolors.tam_tools.tam_color_palette*), 41
TAMCycleColorTests (*class in tamcolors.tests.tam_tools_tests.tam_color_palette_tests*), 54
TAMDefaultColor (*class in tamcolors.tam_tools.tam_color_palette*), 41
TAMDefaultColorTests (*class in tamcolors.tests.tam_tools_tests.tam_color_palette_tests*), 55
TAMDriver (*class in tamcolors.tam_io.tam_drivers*), 34
TAMFilm (*class in tamcolors.tam_tools.tam_film*), 41
TAMFilmError, 42
TAMFilmFadeInTests (*class in tamcolors.tests.tam_tools_tests.tam_fade_tests*), 55
TAMFilmTests (*class in tamcolors.tests.tam_tools_tests.tam_film_tests*), 55
TAMFrame (*class in tamcolors.tam.tam_loop*), 18
TAMFrameTests (*class in tamcolors.tests.tam_tools_tests.tam_loop_tests*), 53
TAMIdentifier (*class in tamcolors.tam_io.tam_identifier*), 35
TAMKeyManager (*class in tamcolors.examples.tam_key_manager*), 16
TAMKeyManager (*class in tamcolors.tam_tools.tam_key_manager*), 43
TAMKeyManagerTests (*class in tamcolors.tests.tam_tools_tests.tam_key_manager_tests*), 56
TAMKeys (*class in tamcolors.examples.tam_keys*), 16
TAMKeyTests (*class in tamcolors.tests.tam_io_tests.tam_keys_tests*), 49
TAMListSurface (*class in tamcolors.examples.tam_list_surface*), 16
TAMListSurfaceTests (*class in tamcolors.tests.tam_tools_tests.tam_list_surface_tests*), 57
TAMLoop (*class in tamcolors.tam.tam_loop*), 18
TAMLoopError, 19
TAMLoopHelloWorld (*class in tamcolors.examples.tam_loop*), 17
TAMLoopIOHandler (*class in tamcolors.tam.tam_loop_io_handler*), 19
TAMLoopIOTCPHandler (*class in tamcolors.tam.tam_loop_io_tcp_handler*), 21
TAMLoopReceiver (*class in tamcolors.tam.tam_loop_receiver*), 21
TAMLoopTCPReceiver (*class in tamcolors.tam.tam_loop_tcp_receiver*), 22
TAMLoopTCPReceiver (*class in tamcolors.tests.tam_tests.tam_loop_tcp_receiver_tests*), 53
TAMLoopTests (*class in tamcolors.tests.tam_tests.tam_loop_tests*), 53
TAMMenu (*class in tamcolors.tam_tools.tam_menu*), 44
TAMMenuTests (*class in tamcolors.tests.tam_tools_tests.tam_menu_tests*), 57
TAMPlacingTests (*class in tamcolors.tests.tam_tools_tests.tam_placing_tests*), 59
TAMPrint (*class in tamcolors.examples.tam_print*), 17
TAMPrint (*class in tamcolors.examples.tam_text_box*), 17
TAMPrintTests (*class in tamcolors.tests.tam_tools_tests.tam_print_tests*), 59
TAMSoundError, 30
TAMStrError, 47
TAMSurface (*class in tamcolors.tam_io.tam_surface*), 36
TAMSurfaceTests (*class in tamcolors.tests.tam_io_tests.tam_surface_tests*),

49
TAMTextBox (*class in tamcolors.tam_tools.tam_text_box*), 47
TAMTextBoxButton (*class in tamcolors.tam_tools.tam_menu*), 45
TAMTextBoxButtonTests (*class in tamcolors.tests.tam_tools_tests.tam_menu_tests*), 57
TAMTextBoxTests (*class in tamcolors.tests.tam_tools_tests.tam_text_box_tests*), 60
TAMTextButton (*class in tamcolors.tam_tools.tam_menu*), 45
TAMTextButtonTests (*class in tamcolors.tests.tam_tools_tests.tam_menu_tests*), 58
task() (*tamcolors.tests.test_multi_task_helper.MultiTaskHelper class method*), 64
TCPBase (*class in tamcolors.utils.tcp*), 67
TCPConnection (*class in tamcolors.utils.tcp*), 67
TCPError, 67
TCPHost (*class in tamcolors.utils.tcp*), 67
TCPObjectConnector (*class in tamcolors.utils.tcp*), 68
TCPObjectWrapper (*class in tamcolors.tests.utils_tests.tcp_tests*), 63
TCPObjectWrapper (*class in tamcolors.utils.tcp*), 68
TCPReceiver (*class in tamcolors.utils.tcp*), 68
TCPTests (*class in tamcolors.tests.utils_tests.tcp_tests*), 63
test_1() (*tamcolors.tests.utils_tests.compress_tests.CompressTests method*), 61
test_2() (*tamcolors.tests.utils_tests.compress_tests.CompressTests method*), 61
test_3() (*tamcolors.tests.utils_tests.compress_tests.CompressTests method*), 61
test_4() (*tamcolors.tests.utils_tests.compress_tests.CompressTests method*), 61
test__console_color_count() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 51
test__done() (*tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method*), 53
test__done_2() (*tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method*), 53
test__draw_16() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 51
test__draw_2() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests static method*), 51
test__draw_onto() (*tamcolors.tests.tam_io_tests.io_tam_tests.IOTAMTest method*), 48
test__draw_onto_2() (*tamcolors.tests.tam_io_tests.io_tam_tests.IOTAMTest method*), 48
test__draw_onto_3() (*tamcolors.tests.tam_io_tests.io_tam_tests.IOTAMTest method*), 48
test__frame_done() (*tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method*), 53
test__frame_done_2() (*tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method*), 53
test__get_buffer_dimension() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 51
test__get_center() (*tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method*), 59
test__get_center_2() (*tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method*), 59
test__get_dimensions_wrapper() (*tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method*), 59
test__get_dimensions_wrapper_2() (*tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method*), 59
test__get_dimensions_wrapper_3() (*tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method*), 59
test__get_other_side() (*tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method*), 59
test__get_other_side_2() (*tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method*), 59
test__print() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests static method*), 51
test__print_2() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests static method*), 51
test__print_3() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests static method*), 51
test__processes_special_color_1() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 51
test__processes_special_color_2() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 51
test__processes_special_color_3() (*tam-*

colors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method), 51
test__processes_special_color_4() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 51
test__processes_special_color_5() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 51
test__processes_special_color_6() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 51
test__spot_swap() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 52
test__write_to_output_stream() (*tamcolors.tests.tam_io_tests.io_tam_tests.IOTAMTest static method*), 48
test__write_to_output_stream_2() (*tamcolors.tests.tam_io_tests.io_tam_tests.IOTAMTest static method*), 48
test_able_to_execute() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 52
test_append() (*tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method*), 55
test_assert_no_main_process() (*tamcolors.tests.tests_tests.test_multi_task_helper_tests.MultiTaskHelperTests method*), 61
test_assert_no_main_thread() (*tamcolors.tests.tests_tests.test_multi_task_helper_tests.MultiTaskHelperTests method*), 61
test_assert_process() (*tamcolors.tests.tests_tests.test_multi_task_helper_tests.MultiTaskHelperTests method*), 61
test_assert_thread() (*tamcolors.tests.tests_tests.test_multi_task_helper_tests.MultiTaskHelperTests method*), 61
test_bool() (*tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerJsonTests method*), 63
test_bottom_left() (*tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method*), 59
test_bottom_left_2() (*tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method*), 59
test_bottom_right() (*tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method*), 59
test_bottom_right_2() (*tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method*), 59
test_build_setting() (*tamcolors.tests.utils_tests.encryption_tests.EncryptionTests method*), 61
test_bytearray() (*tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerJsonTests method*), 63
test_bytes() (*tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerJsonTests method*), 63
test_cache() (*tamcolors.tests.utils_tests.immutable_cache_tests.ImmutableCacheTests method*), 62
test_call_TextBoxButton() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests static method*), 57
test_call_TextButton() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests static method*), 58
test_center() (*tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method*), 59
test_center_2() (*tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method*), 59
test_clear() (*tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method*), 49
test_close() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests static method*), 52
test_color_eq() (*tamcolors.tests.tam_io_tests.tam_colors_tests.ColorTests method*), 49
test_color_eq_2() (*tamcolors.tests.tam_io_tests.tam_colors_tests.ColorTests method*), 49
test_color_eq_3() (*tamcolors.tests.tam_io_tests.tam_colors_tests.ColorTests method*), 49
test_color_init() (*tamcolors.tests.tam_io_tests.tam_colors_tests.ColorTests method*), 49
test_color_init_2() (*tamcolors.tests.tam_io_tests.tam_colors_tests.ColorTests method*), 49
test_color_ne() (*tamcolors.tests.tam_io_tests.tam_colors_tests.ColorTests method*), 49
test_color_ne_2() (*tamcolors.tests.tam_io_tests.tam_colors_tests.ColorTests method*), 49
test_colors() (*tamcolors.tests.tam_io_tests.tam_colors_tests.TAMColorTests method*), 49
test_copy() (*tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerJsonTests method*), 63

ors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 49
test_copy_2() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_critical() (tamcolors.tests.utils_tests.log_tests.LogTests method), 62
test_debug() (tamcolors.tests.utils_tests.log_tests.LogTests method), 62
test_dict() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerJsonTests method), 63
test_done() (tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method), 52
test_done() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 55
test_done() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_done_2() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 55
test_done_2() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_done_3() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 55
test_draw() (tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method), 53
test_draw() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_draw() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method), 57
test_draw() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method), 58
test_draw() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_draw_2() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method), 57
test_draw_2() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method), 58
test_draw_2() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_draw_3() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_draw_onto() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_draw_onto_2() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_draw_onto_3() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_draw_onto_4() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_draw_onto_5() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_draw_onto_6() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_draw_onto_7() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_draw_onto_8() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_dunder_bytes() (tamcolors.tests.utils_tests.object_packer_tests.FastHandObjectPackerTe method), 62
test_enable_and_disable_logging() (tamcolors.tests.utils_tests.log_tests.LogTests static method), 62
test_encrypt_decrypt() (tamcolors.tests.utils_tests.encryption_tests.EncryptionTests method), 61
test_events() (tamcolors.tests.tam_io_tests.any_drivers_tests.AnyIOTests method), 48
test_fast_hand_object() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerJsonTests method), 63
test_float() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerJsonTests method), 63
test_frame_init() (tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method), 53
test_full_play() (tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method), 52
test_generate_identifier() (tamcol-

ors.tests.utils_tests.identifier_tests.IdentifierTests method), 61
test_get() (tamcol-ors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 55
test_get_2() (tamcol-ors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 55
test_get_3() (tamcol-ors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 55
test_get_action() (tamcol-ors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method), 57
test_get_action() (tamcol-ors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method), 58
test_get_buttons() (tamcol-ors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_get_buttons_2() (tamcol-ors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_get_buttons_3() (tamcol-ors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_get_call_key() (tamcol-ors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_get_call_key_2() (tamcol-ors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_get_char() (tamcol-ors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_get_char_2() (tamcol-ors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_get_circular() (tamcol-ors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 55
test_get_circular_2() (tamcol-ors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 55
test_get_clock() (tamcol-ors.tests.tam_tools_tests.tam_color_palette_tests.TAMCycleColorTests method), 54
test_get_color() (tamcol-ors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_get_color() (tamcol-ors.tests.tam_tools_tests.tam_color_palette_tests.TAMDefaultColorTests method), 55
test_get_color_2() (tamcol-ors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPalet method), 54
test_get_color_3() (tamcol-ors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPalet method), 54
test_get_colors() (tamcol-ors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_get_colors_2() (tamcol-ors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_get_cross_rect() (tamcol-ors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_get_cross_rect_2() (tamcol-ors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_get_defaults() (tamcol-ors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_get_defaults() (tamcol-ors.tests.tam_tests.tam_loop_tests.TAMFrameTests method), 53
test_get_defaults_2() (tamcol-ors.tests.tam_tests.tam_loop_tests.TAMFrameTests method), 53
test_get_dimensions() (tamcol-ors.tests.tam_io_tests.any_drivers_tests.AnyIOTests method), 48
test_get_dimensions() (tamcol-ors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_get_dimensions() (tamcol-ors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method), 52
test_get_dimensions() (tamcol-ors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_get_dimensions_2() (tamcol-ors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_get_dimensions_2() (tamcol-ors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_get_dimensions_3() (tamcol-ors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_get_fps() (tamcol-ors.tests.tam_tests.tam_loop_tests.TAMFrameTests method), 53
test_get_fps_2() (tamcol-ors.tests.tam_tests.tam_loop_tests.TAMFrameTests method), 53
test_get_from_raw_spot() (tamcol-

ors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_get_goto_map() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method*), 57
test_get_height_min_and_max() (*tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method*), 53
test_get_height_min_and_max_2() (*tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method*), 53
test_get_identifier_bytes() (*tamcolors.tests.utils_tests.identifier_tests.IdentifierTests method*), 62
test_get_io() (*tamcolors.tests.tam_io_tests.any_drivers_tests.AnyIOTests method*), 48
test_get_key() (*tamcolors.tests.tam_io_tests.any_drivers_tests.AnyIOTests method*), 48
test_get_key() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 52
test_get_key_2() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 52
test_get_key_3() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 52
test_get_key_4() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 52
test_get_key_5() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 52
test_get_key_dict() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 52
test_get_key_state() (*tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method*), 56
test_get_key_state_2() (*tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method*), 56
test_get_keyboard_name() (*tamcolors.tests.tam_io_tests.any_drivers_tests.AnyIOTests method*), 48
test_get_keyboard_name() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 52
test_get_keyboard_name_2() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 52
test_get_keyboard_name_3() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 52
test_get_keyboard_name_4() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 52
test_get_modes() (*tamcolors.tests.tam_io_tests.any_drivers_tests.AnyIOTests method*), 48
test_get_modes() (*tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method*), 52
test_get_on() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method*), 57
test_get_on_2() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method*), 57
test_get_position() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method*), 58
test_get_position() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method*), 58
test_get_position_2() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method*), 58
test_get_position_2() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method*), 58
test_get_property() (*tamcolors.tests.tam_io_tests.tam_colors_tests.ColorTests method*), 49
test_get_property() (*tamcolors.tests.tam_io_tests.tam_colors_tests.RGBATests method*), 49
test_get_property_2() (*tamcolors.tests.tam_io_tests.tam_colors_tests.ColorTests method*), 49
test_get_property_2() (*tamcolors.tests.tam_io_tests.tam_colors_tests.RGBATests method*), 49
test_get_raw_spot() (*tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method*), 50
test_get_raw_surface() (*tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method*), 50
test_get_raw_surface_2() (*tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method*), 50
test_get_raw_user_input() (*tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManager method*), 56
test_get_rule() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method*), 57

ors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_get_rule_2() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_get_spot() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_get_text() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_get_text_2() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_get_user_input() (tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method), 56
test_get_user_input_2() (tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method), 56
test_get_user_input_3() (tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method), 56
test_get_user_input_generator() (tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method), 56
test_get_user_input_generator_2() (tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method), 56
test_get_user_input_generator_3() (tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method), 56
test_get_width_min_and_max() (tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method), 53
test_get_width_min_and_max_2() (tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method), 53
test_getitem() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_getitem() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 55
test_getitem_2() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_getitem_2() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 55
test_getitem_3() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_getitem_3() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 55
test_globals() (tamcolors.tests.utils_tests.identifier_tests.IdentifierTests method), 62
test_hand_down() (tamcolors.tests.tests_tests.test_multi_task_helper_tests.MultiTaskHelperTests method), 61
test_init() (tamcolors.tests.utils_tests.encryption_tests.EncryptionTests method), 61
test_init() (tamcolors.tests.utils_tests.encryption_tests.NoEncryptionTests method), 61
test_init() (tamcolors.tests.utils_tests.immutable_cache_tests.ImmutableCacheTests method), 62
test_init_color_palette() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_init_color_palette_2() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_init_cycle_color() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMCycleColorPaletteTests method), 54
test_init_default_color() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMDefaultColorPaletteTests method), 55
test_init_film() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 55
test_init_key_manger() (tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method), 56
test_init_TextBoxButton() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method), 58
test_init_TextBoxButton_2() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method), 58
test_init_TextButton() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method), 58
test_init_TextButton_2() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method), 58
test_int() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerJsonTests method), 63
test_iter() (tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method), 56
test_key_present() (tamcolors.

ors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_key_present_2() (*tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method*), 54
test_keys() (*tamcolors.tests.tam_io_tests.tam_keys_tests.TAMKeyTests method*), 49
test_lap() (*tamcolors.tests.utils_tests.timer_tests.TimerTests method*), 63
test_large_data() (*tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerJsonTests method*), 63
test_large_data() (*tamcolors.tests.utils_tests.transport_optimizer_tests.TransportOptimizerTests method*), 64
test_large_tam_surface_data() (*tamcolors.tests.utils_tests.transport_optimizer_tests.TransportOptimizerTests method*), 64
test_len() (*tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method*), 55
test_len_2() (*tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method*), 55
test_list() (*tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerJsonTests method*), 63
test_log_stream() (*tamcolors.tests.utils_tests.log_tests.LogTests method*), 62
test_loop_init() (*tamcolors.tests.tam_tests.tam_loop_tests.TAMLoopTests method*), 53
test_loop_tcp_receiver() (*tamcolors.tests.tam_tests.tam_loop_tcp_receiver_tests.TAMLoopTCPReceiver method*), 53
test_make_surface_ready() (*tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method*), 53
test_make_surface_ready_2() (*tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method*), 53
test_make_surface_ready_3() (*tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method*), 53
test_make_tam_str() (*tamcolors.tests.tam_tools_tests.tam_str_tests.MakeTAMStrTests method*), 59
test_make_tam_str_2() (*tamcolors.tests.tam_tools_tests.tam_str_tests.MakeTAMStrTests method*), 59
test_make_tam_str_3() (*tamcolors.tests.tam_tools_tests.tam_str_tests.MakeTAMStrTests method*), 60
test_make_tam_str_4() (*tamcolors.tests.tam_tools_tests.tam_str_tests.MakeTAMStrTests method*), 60
test_make_tam_str_5() (*tamcolors.tests.tam_tools_tests.tam_str_tests.MakeTAMStrTests method*), 60
test_new() (*tamcolors.tests.utils_tests.immutable_cache_tests.ImmutableCacheTests method*), 62
test_next() (*tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method*), 55
test_next_2() (*tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method*), 55
test_none() (*tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerJsonTests method*), 63
test_object_wrapper() (*tamcolors.tests.utils_tests.tcp_tests.TCPObjectWrapper method*), 63
test_off() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method*), 58
test_off() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method*), 58
test_off_2() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method*), 58
test_off_2() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method*), 58
test_offset_sleep() (*tamcolors.tests.utils_tests.timer_tests.TimerTests method*), 63
test_on() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBox method*), 58
test_on() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBu method*), 58
test_on_2() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method*), 58
test_on_2() (*tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method*), 58
test_peak() (*tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method*), 56
test_peak_2() (*tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method*), 56

test_peak_3() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 56
test_ping() (tamcolors.tests.utils_tests.tcp_tests.TCPTests method), 63
test_ping_ipv6() (tamcolors.tests.utils_tests.tcp_tests.TCPTests method), 63
test_play() (tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests static method), 52
test_pop() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 56
test_position() (tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method), 52
test_preferred_mode() (tamcolors.tests.tam_tests.tam_loop_tests.TAMLoopTests method), 53
test_preferred_mode_2() (tamcolors.tests.tam_tests.tam_loop_tests.TAMLoopTests method), 53
test_preferred_mode_3() (tamcolors.tests.tam_tests.tam_loop_tests.TAMLoopTests method), 53
test_replace_alpha_chars_1() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_replace_alpha_chars_2() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_replace_alpha_chars_3() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_reset_colors_to_console_defaults() (tamcolors.tests.tam_io_tests.any_drivers_tests.AnyIOTests static method), 48
test_reset_colors_to_console_defaults() (tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method), 52
test_reset_colors_to_console_defaults() (tamcolors.tests.tam_tests.tam_loop_tests.TAMLoopTests method), 53
test_rest() (tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method), 52
test_rgba_eq() (tamcolors.tests.tam_io_tests.tam_colors_tests.RGBATests method), 49
test_rgba_eq_2() (tamcolors.tests.tam_io_tests.tam_colors_tests.RGBATests method), 49
test_rgba_eq_3() (tamcolors.tests.tam_io_tests.tam_colors_tests.RGBATests method), 49
test_rgba_init() (tamcolors.tests.tam_io_tests.tam_colors_tests.RGBATests method), 49
test_rgba_init_2() (tamcolors.tests.tam_io_tests.tam_colors_tests.RGBATests method), 49
test_rgba_ne() (tamcolors.tests.tam_io_tests.tam_colors_tests.RGBATests method), 49
test_rgba_ne_2() (tamcolors.tests.tam_io_tests.tam_colors_tests.RGBATests method), 49
test_run() (tamcolors.tests.tam_tests.tam_loop_tests.TAMLoopTests method), 53
test_run_action() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests static method), 58
test_run_action() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests static method), 58
test_same_data() (tamcolors.tests.utils_tests.transport_optimizer_tests.TransportOptimizer method), 64
test_same_doc_string() (tamcolors.tests.tam_io_tests.io_tam_tests.RAWIOTest method), 48
test_same_methods() (tamcolors.tests.tam_io_tests.io_tam_tests.RAWIOTest method), 48
test_save_and_load_data_1() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerFunctionsTe method), 62
test_save_and_load_data_2() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerFunctionsTe method), 62
test_save_and_load_data_3() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerFunctionsTe method), 62
test_save_and_load_data_4() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerFunctionsTe method), 62
test_save_and_load_int_1() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerFunctionsTe method), 62
test_save_and_load_int_2() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerFunctionsTe method), 62
test_save_and_load_int_3() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerFunctionsTe method), 63

test_save_and_load_int_4() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerFunctionsTests method), 63
test_set() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 56
test_set() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerJsonTests method), 63
test_set_2() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 56
test_set_3() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 56
test_set_action() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method), 58
test_set_action() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method), 58
test_set_char() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_set_char_2() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_set_circular() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 56
test_set_clock() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMCyclesColorsTests method), 54
test_set_color() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_set_color() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMDefaultColorsTests method), 55
test_set_color_2() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_set_color_3() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_set_colors() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMCyclesColorsTests method), 54
test_set_colors() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_set_colors_2() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_set_colors_3() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_set_defaults_and_clear() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_set_defaults_and_clear_2() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_set_dimensions_and_clear() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_set_dimensions_and_clear_2() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_set_position() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method), 58
test_set_position() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method), 58
test_set_position_2() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method), 58
test_set_position_2() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method), 58
test_set_rule() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPalette method), 54
test_set_rule_2() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPalette method), 54
test_set_slash_get_mode() (tamcolors.tests.tam_io_tests.any_drivers_tests.AnyIOTests method), 48
test_set_slash_get_mode() (tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method), 52
test_set_spot() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_set_spot_2() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_set_spot_3() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_set_spot_4() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_set_spot_5() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50

test_set_spot_6() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_set_spot_7() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_set_spot_8() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_set_spot_9() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_set_tam_color_defaults() (tamcolors.tests.tam_io_tests.any_drivers_tests.AnyIOTests static method), 48
test_set_tam_color_defaults() (tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method), 52
test_set_tam_color_defaults() (tamcolors.tests.tam_tests.tam_loop_tests.TAMLoopTests method), 53
test_setitem() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_setitem() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 56
test_setitem_2() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_setitem_2() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 56
test_setitem_3() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_setitem_3() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 56
test_silent_key_state() (tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method), 56
test_silent_key_state_2() (tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method), 56
test_silent_key_state_3() (tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method), 56
test_simple_data() (tamcolors.tests.utils_tests.transport_optimizer_tests.TransportOptimizerTests method), 64
test_simple_encrypt_decrypt() (tamcolors.tests.utils_tests.encryption_tests.EncryptionTests method), 61
test_simple_menu_builder() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_simple_menu_builder_2() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_simple_menu_builder_3() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_simple_menu_builder_4() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_simple_menu_builder_5() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_slide() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 56
test_slide_2() (tamcolors.tests.tam_tools_tests.tam_film_tests.TAMFilmTests method), 56
test_snapshot() (tamcolors.tests.tam_io_tests.any_drivers_tests.AnyIOTests static method), 48
test_sound() (tamcolors.tests.tam_io_tests.any_drivers_tests.AnyIOTests method), 48
test_stack() (tamcolors.tests.tam_tests.tam_loop_tests.TAMLoopTests method), 53
test_start() (tamcolors.tests.tam_io_tests.win_drivers_tests.WinDriversTests method), 52
test_str() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPalette method), 54
test_str() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerJsonTests method), 63
test_str_2() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPalette method), 54
test_str_TextBoxButton() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method), 58
test_str_TextBoxButton_2() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests method), 58
test_str_TextButton() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method), 58
test_str_TextButton_2() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests method), 58

test_surface_eq() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 50
test_surface_eq_2() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_eq_3() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_eq_4() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_eq_5() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_init() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_len() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_len_2() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_len_3() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_len_4() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_ne() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_ne_2() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_ne_3() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_ne_4() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_str() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_str_2() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_surface_str_3() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_tam_color_1() (tamcolors.tests.utils_tests.object_packer_tests.FastHandObjectPackerTests method), 62
test_tam_color_2() (tamcolors.tests.utils_tests.object_packer_tests.FastHandObjectPackerTests method), 62
test_tam_fade_in() (tamcolors.tests.tam_tools_tests.tam_fade_tests.TAMFilmFadeInTests method), 55
test_tam_fade_in_2() (tamcolors.tests.tam_tools_tests.tam_fade_tests.TAMFilmFadeInTests method), 55
test_tam_fade_in_3() (tamcolors.tests.tam_tools_tests.tam_fade_tests.TAMFilmFadeInTests method), 55
test_tam_fade_in_4() (tamcolors.tests.tam_tools_tests.tam_fade_tests.TAMFilmFadeInTests method), 55
test_tam_list_surface() (tamcolors.tests.tam_tools_tests.tam_list_surface_tests.TAMListSurfaceTests method), 57
test_tam_list_surface_2() (tamcolors.tests.tam_tools_tests.tam_list_surface_tests.TAMListSurfaceTests method), 57
test_tam_list_surface_3() (tamcolors.tests.tam_tools_tests.tam_list_surface_tests.TAMListSurfaceTests method), 57
test_tam_list_surface_4() (tamcolors.tests.tam_tools_tests.tam_list_surface_tests.TAMListSurfaceTests method), 57
test_tam_list_surface_5() (tamcolors.tests.tam_tools_tests.tam_list_surface_tests.TAMListSurfaceTests method), 57
test_tam_menu() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_tam_menu_2() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_tam_print() (tamcolors.tests.tam_tools_tests.tam_print_tests.TAMPrintTests method), 59
test_tam_print_2() (tamcolors.tests.tam_tools_tests.tam_print_tests.TAMPrintTests method), 59
test_tam_print_3() (tamcolors.tests.tam_tools_tests.tam_print_tests.TAMPrintTests method), 59
test_tam_print_4() (tamcolors.tests.tam_tools_tests.tam_print_tests.TAMPrintTests method), 59
test_tam_print_5() (tamcolors.tests.tam_tools_tests.tam_print_tests.TAMPrintTests method), 59
test_tam_surface_1() (tamcolors.tests.utils_tests.object_packer_tests.FastHandObjectPackerTests method), 62

test_tam_surface_2() (tamcolors.tests.utils_tests.object_packer_tests.FastHandObjectPackerTests method), 62
test_tam_text_box_init() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_tam_text_box_str() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_tam_text_box_str_2() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_timer() (tamcolors.tests.utils_tests.timer_tests.TimerTests method), 63
test_to_bytes_from_bytes() (tamcolors.tests.tam_io_tests.tam_surface_tests.TAMSurfaceTests method), 51
test_top_left() (tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method), 59
test_top_left_2() (tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method), 59
test_top_right() (tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method), 59
test_top_right_2() (tamcolors.tests.tam_tools_tests.tam_placing_tests.TAMPlacingTests method), 59
test_tuple() (tamcolors.tests.utils_tests.object_packer_tests.ObjectPackerTests method), 63
test_update() (tamcolors.tests.tam_tests.tam_loop_tests.TAMFrameTests method), 53
test_update() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMColorPaletteTests method), 54
test_update() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMCycleColorTests method), 55
test_update() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMDefaultColorTests method), 55
test_update() (tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method), 56
test_update() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_update() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextBoxButtonTests static method), 58
test_update() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMTextButtonTests static method), 58
test_update() (tamcolors.tests.tam_tools_tests.tam_text_box_tests.TAMTextBoxTests method), 60
test_update_2() (tamcolors.tests.tam_tools_tests.tam_color_palette_tests.TAMCycleColorTests method), 55
test_update_2() (tamcolors.tests.tam_tools_tests.tam_key_manager_tests.TAMKeyManagerTests method), 56
test_update_2() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_update_3() (tamcolors.tests.tam_tools_tests.tam_menu_tests.TAMMenuTests method), 57
test_win_stable() (tamcolors.tests.tam_io_tests.win_drivers_tests.WinGlobalsTests method), 52
tests_main() (in module tamcolors.tests.all_tests), 64
thread_task() (tamcolors.tam.tam_loop.TAMLoop method), 19
tick() (tamcolors.utils.timer.TickRateTracker method), 68
tick_rate() (tamcolors.utils.timer.TickRateTracker method), 68
TickRateTracker (class in tamcolors.utils.timer), 68
Timer (class in tamcolors.utils.timer), 68
TimerTests (class in tamcolors.tests.utils_tests.timer_tests), 63
to_bytes() (tamcolors.tam_io.tam_colors.Color method), 32
to_bytes() (tamcolors.tam_io.tam_colors.RGBA method), 32
TAMColorPaletteTests (tamcolors.tam_io.tam_surface.TAMSurface method), 37
TAMCycleColorTests (tamcolors.utils.object_packer.FastHandObjectPacker method), 66
TAMDefaultColorTests (in module tamcolors.tam_tools.tam_placing), 46
top_right() (in module tamcolors.tam_tools.tam_placing), 46
transparent_value (tamcolors.tam_io.tam_colors.Color attribute), 32
TransportOptimizerTests (class in tamcolors.tests.utils_tests.transport_optimizer_tests), 64

U

unfreeze_handler() (*tamcolors.tam.tam_loop_io_handler.TAMLoopIOHandler method*), 20
UNISharedData (*class in tamcolors.tam_io.uni_drivers*), 38
UNIUtilitiesDriver (*class in tamcolors.tam_io.uni_drivers*), 38
update() (*tamcolors.examples.alpha.TAMAlpha method*), 13
update() (*tamcolors.examples.clouds.Clouds method*), 14
update() (*tamcolors.examples.colors.TAMCOLORS method*), 14
update() (*tamcolors.examples.host_multi_player.HostMultiPlayer method*), 14
update() (*tamcolors.examples.icon.BootLogo method*), 15
update() (*tamcolors.examples.rgb_color.RGBCOLOR method*), 15
update() (*tamcolors.examples.tabletennis.Ball method*), 15
update() (*tamcolors.examples.tabletennis.Racket method*), 15
update() (*tamcolors.examples.tabletennis.TableTennis method*), 16
update() (*tamcolors.examples.tam_key_manager.TAMKeyManager method*), 16
update() (*tamcolors.examples.tam_keys.TAMKeys method*), 16
update() (*tamcolors.examples.tam_list_surface.TAMListSurface method*), 16
update() (*tamcolors.examples.tam_loop.TAMLoopHelloWorld method*), 17
update() (*tamcolors.examples.tam_print.TAMPrint method*), 17
update() (*tamcolors.examples.tam_text_box.TAMPrint method*), 17
update() (*tamcolors.tam.tam_loop.TAMFrame method*), 18
update() (*tamcolors.tam_tools.tam_color_palette.TAMColorPalette method*), 40
update() (*tamcolors.tam_tools.tam_color_palette.TAMColorPaletteRule method*), 41
update() (*tamcolors.tam_tools.tam_color_palette.TAMCycleColor method*), 41
update() (*tamcolors.tam_tools.tam_color_palette.TAMDefaultColor method*), 41
update() (*tamcolors.tam_tools.tam_key_manager.TAMKeyManager method*), 44
update() (*tamcolors.tam_tools.tam_menu.TAMButtonRule method*), 44
update() (*tamcolors.tam_tools.tam_menu.TAMMenu method*), 45
update() (*tamcolors.tam_tools.tam_menu.TAMTextBoxButton method*), 45
update() (*tamcolors.tam_tools.tam_menu.TAMTextButton method*), 46
update() (*tamcolors.tam_tools.tam_text_box.TAMTextBox method*), 47
utilities_driver_operational() (*tamcolors.tam_io.io_tam.IO method*), 27
utilities_driver_operational() (*tamcolors.tam_io.io_tam.RawIO method*), 30
UtilitiesDriver (*class in tamcolors.tam_io.tam_drivers*), 35

W

wait_key() (*tamcolors.tam_io.any_drivers.ANYKeyDriver method*), 24
wait_key() (*tamcolors.tam_io.io_tam.IO method*), 27
wait_key() (*tamcolors.tam_io.io_tam.RawIO method*), 30
wait_key() (*tamcolors.tam_io.null_drivers.NULLKeyDriver method*), 30
wait_key() (*tamcolors.tam_io.tam_drivers.KeyDriver method*), 34
warning() (*in module tamcolors.utils.log*), 66
WinDriversTests (*class in tamcolors.tests.tam_io_tests.win_drivers_tests*), 51
WINFullColorDriver (*class in tamcolors.tam_io.win_drivers*), 38
WinGlobalsTests (*class in tamcolors.tests.tam_io_tests.win_drivers_tests*), 52
WINKeyDriver (*class in tamcolors.tam_io.win_drivers*), 39
winner() (*tamcolors.examples.tabletennis.Ball method*), 15
WinSharedData (*class in tamcolors.tam_io.win_drivers*), 40
WINSoundDriver (*class in tamcolors.tam_io.win_drivers*), 39
WINUtilitiesDriver (*class in tamcolors.tam_io.win_drivers*), 39