
opencv-proto Documentation

Release 0.3.0

Igor ‘idle sign’ Starikov

Apr 30, 2020

Contents

1	Description	3
2	Requirements	5
3	Table of Contents	7
3.1	Quickstart	7
3.1.1	Color Palette	7
3.1.2	Camera capture	7
3.2	API	8
3.2.1	Application	8
3.2.1.1	Config	9
3.2.1.2	Keys	10
3.2.2	User Interface	10
3.2.2.1	Windows Manager	10
3.2.2.2	Window	10
3.2.2.3	Trackbar	11
3.2.3	Sources	12
3.2.3.1	Image	12
3.2.3.2	Camera	13
3.2.3.3	Video	15
3.2.4	Miscellaneous	18
3.2.4.1	Canvas	18
3.2.4.2	Colors	19
3.2.4.3	Text	19
3.2.5	Primitives	20
3.2.5.1	Legend	20
	Python Module Index	21
	Index	23

<https://github.com/idlesign/opencv-proto>

CHAPTER 1

Description

Allows fast prototyping in Python for OpenCV

Offers primitives and simplified interfaces to streamline prototypes construction in Python.

Facilitates:

- Windows construction and management
- Trackbar construction
- Configuration save/load (including trackbar values)
- Key binding (e.g. for trackbar control, configuration save/load)
- Video capturing and modification
- Work with images
- Work with text
- Frames transformation

CHAPTER 2

Requirements

1. Python 3.6+
2. opencv-python (or variants)

Quick install with third-parties: `$ pip install opencv-proto[all]`

3.1 Quickstart

3.1.1 Color Palette

Let's replace 37 lines of source code from [Trackbar as the Color Palette](#) tutorial with `ocvproto`-based implementation:

```
from ocvproto.toolbox import WindowManager, Canvas

with WindowManager() as wm:
    # Window manager will create a default window for us if none provided.
    # Default window is available through .window property.

    # With the help of .add_trackbar_group() we create three trackbars,
    # to adjust our color values for R, G and B. Batch apply `max` value
    # to all three trackbars.
    rgb = wm.window.add_trackbar_group(['R', 'G', 'B'], max=255)

    # If one doesn't configure their own Application object,
    # Window manager will instantiate one automatically, using default settings.
    for _ in wm.app.loop():
        # Most of the time we'll need a loop to process our frames.
        # Application object (available through .app property) offers us such a loop.

        # Lastly we create a canvas object using RGB value from trackbars,
        # and pass its' frame to .set_frame() shortcut.
        # That shortcut puts the frame into default window.
        wm.set_frame(Canvas(512, 300, color=rgb))
```

3.1.2 Camera capture

Now let's capture video camera stream into `ocvproto.avi` file, being able to adjust blur.

Let's also setup config filepath (`ocvproto.json`) - this allows us to store current trackbar values (`s` key) and load them (`r` key). It is useful to restore settings between sessions.

We bind `z` key to take camera shots.

```
from ocvproto.toolbox import WindowManager, Camera

with WindowManager() as wm:

    # We instruct our application to store settings into file.
    wm.app.set_config('ocvproto.json')

    # We create two trackbars to adjust blur.
    blur = wm.window.add_trackbar_group(['x', 'y'], 'Blur', default=1)

    # We initiate default (first available) camera connection.
    with Camera() as cam:

        # Let's add trackbars for adjustable camera properties (contrast, brightness,
        # etc.).
        wm.window.add_trackbar_group(cam.describe_properties(), default=1, max=255)

        # You can bind keys to actions.
        # Here we bind `z` to trigger .cam.dump_image() to take stills.
        wm.app.bind_key('z', cam.dump_image)

    for _ in wm.app.loop():
        # Read a frame from camera, we'll work with.
        cam.read()

        # Now we blur that frame.
        cam.blur(blur)

        # And we dump the frame into the file.
        # If dumping parameters were not set up before
        # .dump() shortcut will use defaults
        # (e.g. `ocvproto.avi` name, XVID codec).
        cam.dump()

        # Show the frame.
        wm.set_frame(cam)
```

3.2 API

3.2.1 Application

class `ocvproto.app.application.Application`(*, *config*: Union[str, pathlib.Path, None] = None)

Represents ocvproto application.

Parameters `config` – Configuration to be used.

bind_key (*key*: Union[str, int], *func*: Callable)

Binds a key to a function.

Parameters

- **key** –
- **func** –

config_load()

Loads a configuration from config file.

config_save()

Saves current configuration to config file.

hook_register(*key: str, func: Callable*)

Registers a hook.

Parameters

- **key** – Hooks group key.
- **func** – Hook function to add to group.

loop()

Main application loop. Handles keys listening and issues a loop function (see `.set_loop_func()`).

set_config(*config: Union[str, pathlib.Path, None], keys: Tuple[Union[str, int], Union[str, int]] = None*)

Sets configuration from the app.

Parameters

- **config** – Configuration object.
- **keys** – Keys tuple to save and load configuration.

set_loop_func(*func: Callable*)

Sets a function to perform in a main app loop.

3.2.1.1 Config

class `ocvproto.app.config.Config`(*fpath: Union[str, pathlib.Path, None]*)

Represent an interface to configuration file.

Parameters **fpath** – Configuration file path.

get_data(*key: str, default: Any = None*)

Reads data from a config section denoted by key.

Parameters

- **key** –
- **default** – Default values to return.

load()

Loads configuration from file.

save()

Saves configuration to file.

set_data(*key: str, value: Any*)

Places the data into config section denoted by key.

Parameters

- **key** –
- **value** –

3.2.1.2 Keys

class `ocvproto.app.keys.Key`
 Keys registry. Can be used for binding actions.

3.2.2 User Interface

3.2.2.1 Windows Manager

class `ocvproto.ui.wm.WindowManager` (*windows: List[ocvproto.ui.window.Window] = None, app: ocvproto.app.application.Application = None*)

Manages windows.

Parameters

- **windows** – Windows to manage. If not set, one window is automatically constructed.
- **app** – ocvproto application object. Automatically constructed if not set.

config_load (*config: ocvproto.app.config.Config*)
 Updates managed windows using data from the given config.

Parameters config –

config_update (*config: ocvproto.app.config.Config*)
 Updates data gathered from managed windows in the given config.

Parameters config –

iter_trackbars () → Generator[Tuple[ocvproto.ui.window.Window, ocvproto.ui.trackbars.base.Trackbar], None, None]
 Generator yielding managed windows and trackbars.

render ()
 Renders managed windows.

set_frame (*frame: Union[<sphinx.ext.autodoc.importer._MockObject object at 0x7f193ba800f0>, Frame]*)
 Sets frame to be rendered in default window.

Parameters frame –

window
 Default window.

3.2.2.2 Window

class `ocvproto.ui.window.Window` (*name: str = None*)
 Represents a window.

Parameters name – Window name. If not set, automatically generated.

add_trackbar (**trackbars*)
 Add the given trackbars to the window.

Parameters trackbars –

add_trackbar_group (*definitions: Union[int, Dict[str, dict], List[dict], List[str]], prefix: str = "", **common_kwargs*) → Tuple
 A shortcut to batch create trackbars in a declarative way.

Parameters

- **definitions** – Definitions to construct trackbars.
 - **Integer:**
 - * 2 - create two trackbars with generated titles and default params.
 - **List:**
 - * ['one', 'two', 'three'] -
 - create 3 trackbars with the given titles and default params.
 - * [{'keys': 'kl'}, {}] -
 - create 2 trackbars with generated titles and default params.
 - **Dictionary:**
 - * {'y': {'keys': 'kl'}, 'x': {'step': 20}} - create 2 trackbars with the given titles and params.
- **prefix** – Prefix to add to trackbars titles.
- **common_kwargs** – Common keyword arguments to pass to all trackbars.

create (*, *autosize=True*)

Creates a window.

Parameters **autosize** – If try, window is automatically sized to a content.

position (*, *x: int, y: int*)

Positions the window.

render ()

Renders window contents.

resize (*, *width: int, height: int*)

Resizes the window.

Parameters

- **width** –
- **height** –

set_frame (*frame: Union[<sphinx.ext.autodoc.importer._MockObject object at 0x7f193ba800f0>, Frame]*)

Sets current frame for the window.

Parameters **frame** –

3.2.2.3 Trackbar

class `ocvproto.ui.trackbars.base.Trackbar` (*name, *, max: Union[int, float] = None, default: Union[int, float] = None, callback: Callable = None, step: Union[int, float] = None, keys: str = None*)

Represents a trackbar.

Parameters

- **name** – Name to show in UI and address this in opencv api.
- **max** – Max value. Default: 100
- **default** – Default (current) value. Default: 0
- **callback** – Function to be called on trackbar value change through UI.
- **step** – Step to inc/dec trackbar value. Default: 1

- **keys** – Two-letter string to represent keys to inc and dec value.

bind (*window_name: str*)
 Binds the trackabr to the given window.
Parameters **window_name** –

dec ()
 Decrements the current value.

get_value () → Union[int, float]
 Force getting current value.

inc ()
 Increments the current value.

onChange (*val: Union[int, float]*)
 Issued on value change from UI.

value
 Current trackbar value.

3.2.3 Sources

3.2.3.1 Image

class `ocvproto.sources.image.Image` (*src: Union[int, str, <sphinx.ext.autodoc.importer._MockObject object at 0x7f193ba800f0>, Frame] = None*)
 Bases: `ocvproto.sources.base.Source`
 Represents an image.

Parameters **src** – Source id (int), path (str) or frame (np array)

absdiff (*frame: Union[<sphinx.ext.autodoc.importer._MockObject object at 0x7f193ba800f0>, Frame]*) → `ocvproto.frame.Frame`
 Returns absolute difference between the current and a given frame as a new Source.
Parameters **frame** –

blur (*ksize: Tuple[int, int]*) → `ocvproto.frame.Frame`
 Blures the current frame inplace.
Parameters **ksize** – Kernel size tuple (width, height)

canny (*thr_1: int, thr_2: int*) → `ocvproto.frame.Frame`
 Applies Canny Edge Detection algorithm to the current frame inplace.
Parameters

- **thr_1** –
- **thr_2** –

dilate (*element: Union[<sphinx.ext.autodoc.importer._MockObject object at 0x7f193ba800f0>, Frame], iterations: int = None*) → `ocvproto.frame.Frame`
 Dilates the current frame inplace.
Parameters

- **element** –
- **iterations** –

draw_rectangle (*, *pos: Tuple[int, int], width: int, height: int, color: Union[int, str, Tuple[int, int, int]] = None*)
 Draws a rectangle.
Parameters

- **pos** – Top left corner (x, y).
- **width** –
- **height** –
- **color** –

dump (*fpath: Union[str, pathlib.Path] = None*)

Dumps frame into a file.

Parameters **fpath** – Filepath to store image into. If not set, name is generated automatically.

fill (*color: Union[int, str, Tuple[int, int, int]]*)

Fills the canvas with the given color.

Parameters **color** –

frame

Current frame.

height

Height

make_gray () → `ocvproto.frame.Frame`

Makes the current frame grayscale inplace.

make_rgb () → `ocvproto.frame.Frame`

Makes the current frame RGB inplace.

read () → `ocvproto.sources.image.Image`

Read and return current frame.

resize (*width: int, height: int*) → `ocvproto.frame.Frame`

Resizes the current frame inplace.

Parameters

- **width** –
- **height** –

width

Width

3.2.3.2 Camera

class `ocvproto.sources.camera.Camera` (*src: Union[int, str] = 0*)

Bases: `ocvproto.sources.video.Video`

Represents a camera device.

Parameters **src** – Device path (str) or id (int). Default 0.

E.g.:

- `'/dev/video0'`
- `0`
- `1`

absdiff (*frame: Union[<sphinx.ext.autodoc.importer._MockObject object at 0x7f193ba800f0>, Frame]*) → `ocvproto.frame.Frame`

Returns absolute difference between the current and a given frame as a new Source.

Parameters **frame** –

blur (*ksize: Tuple[int, int]*) → ocvproto.frame.Frame

Blures the current frame inplace.

Parameters **ksize** – Kernel size tuple (width, height)

brightness

Brightness

canny (*thr_1: int, thr_2: int*) → ocvproto.frame.Frame

Applies Canny Edge Detection algorithm to the current frame inplace.

Parameters

- **thr_1** –
- **thr_2** –

codec

FOURCC codec alias.

contrast

Contrast

describe_properties () → Dict[str, Any]

Returns descriptions for CV properties found in the class of this object and its bases.

One can initialize trackbars with these descriptions: see Window.add_trackbar_group()

dilate (*element: Union[<sphinx.ext.autodoc.importer._MockObject object at 0x7f193ba800f0>, Frame], iterations: int = None*) → ocvproto.frame.Frame

Dilates the current frame inplace.

Parameters

- **element** –
- **iterations** –

draw_rectangle (*, *pos: Tuple[int, int], width: int, height: int, color: Union[int, str, Tuple[int, int, int]] = None*)

Draws a rectangle.

Parameters

- **pos** – Top left corner (x, y).
- **width** –
- **height** –
- **color** –

dump (*frame: Union[<sphinx.ext.autodoc.importer._MockObject object at 0x7f193ba800f0>, Frame] = None*)

Writes the current or the given frame. Automatically configures writer object is needed.

dump_image (*fpath: Union[str, pathlib.Path] = None*)

Dumps the image into a file.

Parameters **fpath** – Filepath to store image into. If not set, name is generated automatically.

dump_setup (*fpath: Union[str, pathlib.Path] = 'opencvproto.avi', *, width: int = None, height: int = None, fps: Union[int, float] = None, codec: str = 'XVID'*) → <sphinx.ext.autodoc.importer._MockObject object at 0x7f193b9b7780>

Configures write parameters. Returns opencv writer object.

Parameters

- **fpath** – Filepath.

- **width** –
- **height** –
- **fps** – Frames per second.
- **codec** – FOURCC codec alias.

exposure

Exposure

fill (*color: Union[int, str, Tuple[int, int, int]]*)

Fills the canvas with the given color.

Parameters **color** –

frame

Current frame.

gain

Gain

get_image () → `ocvproto.sources.image.Image`

Returns image object from the current frame.

hue

Hue

make_gray () → `ocvproto.frame.Frame`

Makes the current frame grayscale inplace.

make_rgb () → `ocvproto.frame.Frame`

Makes the current frame RGB inplace.

read () → `ocvproto.sources.video.Video`

Read and return current frame.

resize (*width: int, height: int*) → `ocvproto.frame.Frame`

Resizes the current frame inplace.

Parameters

- **width** –
- **height** –

saturation

Saturation

set_property (*name: str, value: int*)

Helper method to set property value.

Parameters

- **name** – Property name.
- **value** –

3.2.3.3 Video

class `ocvproto.sources.video.Property` (*cv_prop: int, *, max: int = None*)

Bases: `object`

Represents a capture video property with restrictions.

class `ocvproto.sources.video.Video` (*src: Union[str, <sphinx.ext.autodoc.importer._MockObject object at 0x7f193ba800f0>]*)

Bases: `ocvproto.sources.base.Source`

Represents a video.

absdiff (*frame: Union[<sphinx.ext.autodoc.importer._MockObject object at 0x7f193ba800f0>, Frame]*) → `ocvproto.frame.Frame`

Returns absolute difference between the current and a given frame as a new Source.

Parameters frame –

blur (*ksize: Tuple[int, int]*) → `ocvproto.frame.Frame`

Blures the current frame inplace.

Parameters ksize – Kernel size tuple (width, height)

canny (*thr_1: int, thr_2: int*) → `ocvproto.frame.Frame`

Applies Canny Edge Detection algorithm to the current frame inplace.

Parameters

- **thr_1** –
- **thr_2** –

codec

FOURCC codec alias.

describe_properties () → `Dict[str, Any]`

Returns descriptions for CV properties found in the class of this object and its bases.

One can initialize trackbars with these descriptions: see `Window.add_trackbar_group()`

dilate (*element: Union[<sphinx.ext.autodoc.importer._MockObject object at 0x7f193ba800f0>, Frame], iterations: int = None*) → `ocvproto.frame.Frame`

Dilates the current frame inplace.

Parameters

- **element** –
- **iterations** –

draw_rectangle (*, *pos: Tuple[int, int], width: int, height: int, color: Union[int, str, Tuple[int, int, int]] = None*)

Draws a rectangle.

Parameters

- **pos** – Top left corner (x, y).
- **width** –
- **height** –
- **color** –

dump (*frame: Union[<sphinx.ext.autodoc.importer._MockObject object at 0x7f193ba800f0>, Frame] = None*)

Writes the current or the given frame. Automatically configures writer object is needed.

dump_image (*fpath: Union[str, pathlib.Path] = None*)

Dumps the image into a file.

Parameters fpath – Filepath to store image into. If not set, name is generated automatically.

dump_setup (*fpath: Union[str, pathlib.Path] = 'ocvproto.avi', *, width: int = None, height: int = None, fps: Union[int, float] = None, codec: str = 'XVID') → <sphinx.ext.autodoc.importer._MockObject object at 0x7f193b9b7780>*

Configures write parameters. Returns opencv writer object.

Parameters

- **fpath** – Filepath.
- **width** –
- **height** –
- **fps** – Frames per second.
- **codec** – FOURCC codec alias.

fill (*color: Union[int, str, Tuple[int, int, int]]*)

Fills the canvas with the given color.

Parameters color –

focus

Focus

fps

FPS

frame

Current frame.

gamma

Gamma

get_image () → `ocvproto.sources.image.Image`

Returns image object from the current frame.

height

Height

make_gray () → `ocvproto.frame.Frame`

Makes the current frame grayscale inplace.

make_rgb () → `ocvproto.frame.Frame`

Makes the current frame RGB inplace.

read () → `ocvproto.sources.video.Video`

Read and return current frame.

resize (*width: int, height: int*) → `ocvproto.frame.Frame`

Resizes the current frame inplace.

Parameters

- **width** –
- **height** –

set_property (*name: str, value: int*)

Helper method to set property value.

Parameters

- **name** – Property name.
- **value** –

sharpness

Sharpness

width
Width

zoom
Zoom

3.2.4 Miscellaneous

3.2.4.1 Canvas

class `ocvproto.misc.canvas.Canvas` (*width: int = 640, height: int = 480, *, channels: int = 3, color: Union[int, str, Tuple[int, int, int]] = None*)

Represents a canvas.

Parameters

- **width** –
- **height** –
- **channels** –
- **color** –

absdiff (*frame: Union[<sphinx.ext.autodoc.importer.MockObject object at 0x7f193ba800f0>, Frame]*) → `ocvproto.frame.Frame`

Returns absolute difference between the current and a given frame as a new Source.

Parameters **frame** –

blur (*ksize: Tuple[int, int]*) → `ocvproto.frame.Frame`

Blures the current frame inplace.

Parameters **ksize** – Kernel size tuple (width, height)

canny (*thr_1: int, thr_2: int*) → `ocvproto.frame.Frame`

Applies Canny Edge Detection algorithm to the current frame inplace.

Parameters

- **thr_1** –
- **thr_2** –

dilate (*element: Union[<sphinx.ext.autodoc.importer.MockObject object at 0x7f193ba800f0>, Frame], iterations: int = None*) → `ocvproto.frame.Frame`

Dilates the current frame inplace.

Parameters

- **element** –
- **iterations** –

draw_rectangle (**, pos: Tuple[int, int], width: int, height: int, color: Union[int, str, Tuple[int, int, int]] = None*)

Draws a rectangle.

Parameters

- **pos** – Top left corner (x, y).
- **width** –
- **height** –
- **color** –

dump (*fpath: Union[str, pathlib.Path] = None*)

Dumps frame into a file.

Parameters **fpath** – Filepath to store image into. If not set, name is generated automatically.

fill (*color: Union[int, str, Tuple[int, int, int]]*)
 Fills the canvas with the given color.
Parameters **color** –

height
 Height

make_gray () → ocvproto.frame.Frame
 Makes the current frame grayscale inplace.

make_rgb () → ocvproto.frame.Frame
 Makes the current frame RGB inplace.

resize (*width: int, height: int*) → ocvproto.frame.Frame
 Resizes the current frame inplace.
Parameters

- **width** –
- **height** –

width
 Width

3.2.4.2 Colors

`ocvproto.misc.colors.COLORS = {'beige': (245, 245, 220), 'black': (0, 0, 0), 'blue': (0, 0, 255), 'cyan': (0, 255, 255), 'darkblue': (0, 0, 128), 'darkcyan': (0, 128, 128), 'darkred': (128, 0, 0), 'darkteal': (0, 128, 128), 'darkviolet': (128, 0, 128), 'darkyellow': (128, 128, 0), 'gray': (128, 128, 128), 'green': (0, 255, 0), 'lightblue': (173, 216, 230), 'lightcyan': (224, 255, 255), 'lightgray': (220, 220, 220), 'lightgreen': (153, 255, 153), 'lightteal': (153, 255, 255), 'lightviolet': (255, 153, 255), 'lightyellow': (255, 255, 153), 'magenta': (255, 0, 255), 'red': (255, 0, 0), 'teal': (0, 128, 128), 'violet': (128, 0, 128), 'yellow': (255, 255, 0)}`
 Color aliases to RGB tuples map.

to_rgb (*value: Union[int, str, Tuple[int, int, int]]*) → Tuple[int, int, int]
 Translates the given color value to RGB tuple.
Parameters **value** –

3.2.4.3 Text

class ocvproto.misc.text.**Text** (*val: str = None, *, face: str = None, scale: float = None, color: Union[int, str, Tuple[int, int, int]] = None, pos: Tuple[int, int] = None, weight: int = None*)

Represents a text that can be placed into a frame.

Parameters

- **val** – Text value itself.
- **face** – Font face alias (see `.face_map` keys). Default: normal
- **scale** – Scale factor. Default: 1
- **color** – Color RGB tuple or alias (see `COLORS`). Default: white
- **pos** – Position tuple (x, y) in frame from top-left. Default: (20, 20)
- **weight** – Line thickness. Default: 1

put_on (*frame, text: str = None, *, pos: Tuple[int, int] = None*)

Applies text to the frame.

Parameters

- **frame** – Frame to apply text to.
- **text** – Text value to set on frame. If not set, value from initializer is used.
- **pos** – Position tuple (x, y) in frame from top-left. Default: (20, 20)

classmethod **put_on_demo** (*frame: <sphinx.ext.autodoc.importer._MockObject object at 0x7f193ba800f0>, text: str = 'Test Text 1 2 3 4 5'*)

Demonstrates available font faces applying all of them to the frame.

Parameters

- **frame** – Frame to apply text to.
- **text** – Text value to on frame.

3.2.5 Primitives

3.2.5.1 Legend

class `ocvproto.primitives.legend.Legend` (*labels: Sequence[str], pos: Tuple[int, int] = None, width: int = None, gap: int = None*)

Bases: `object`

Represents a color-legend for labels.

Parameters

- **labels** – Strings to get colors for.
- **pos** – Position (x, y) to place top left legend corner. Default: (20, 20)
- **width** – Default: 250
- **gap** – Base gap (also a height for each color stripe). Default: 25

put_on (*frame: ocvproto.frame.Frame, *, pos: Tuple[int, int] = None*)

Applies the legend to the frame.

Parameters

- **frame** – Frame to apply the legend to.
- **pos** – Position (x, y) to place top left legend corner. Default: (20, 20)

O

- `ocvproto.app.application`, 8
- `ocvproto.app.config`, 9
- `ocvproto.app.keys`, 10
- `ocvproto.misc.canvas`, 18
- `ocvproto.misc.colors`, 19
- `ocvproto.misc.text`, 19
- `ocvproto.primitives.legend`, 20
- `ocvproto.sources.camera`, 13
- `ocvproto.sources.image`, 12
- `ocvproto.sources.video`, 15
- `ocvproto.ui.trackbars.base`, 11
- `ocvproto.ui.window`, 10
- `ocvproto.ui.wm`, 10

A

absdiff() (*ocvproto.misc.canvas.Canvas method*), 18
absdiff() (*ocvproto.sources.camera.Camera method*), 13
absdiff() (*ocvproto.sources.image.Image method*), 12
absdiff() (*ocvproto.sources.video.Video method*), 16
add_trackbar() (*ocvproto.ui.window.Window method*), 10
add_trackbar_group() (*ocvproto.ui.window.Window method*), 10
Application (*class in ocvproto.app.application*), 8

B

bind() (*ocvproto.ui.trackbars.base.Trackbar method*), 12
bind_key() (*ocvproto.app.application.Application method*), 8
blur() (*ocvproto.misc.canvas.Canvas method*), 18
blur() (*ocvproto.sources.camera.Camera method*), 13
blur() (*ocvproto.sources.image.Image method*), 12
blur() (*ocvproto.sources.video.Video method*), 16
brightness (*ocvproto.sources.camera.Camera attribute*), 14

C

Camera (*class in ocvproto.sources.camera*), 13
canny() (*ocvproto.misc.canvas.Canvas method*), 18
canny() (*ocvproto.sources.camera.Camera method*), 14
canny() (*ocvproto.sources.image.Image method*), 12
canny() (*ocvproto.sources.video.Video method*), 16
Canvas (*class in ocvproto.misc.canvas*), 18
codec (*ocvproto.sources.camera.Camera attribute*), 14
codec (*ocvproto.sources.video.Video attribute*), 16
COLORS (*in module ocvproto.misc.colors*), 19
Config (*class in ocvproto.app.config*), 9
config_load() (*ocvproto.app.application.Application method*), 9

config_load() (*ocvproto.ui.wm.WindowManager method*), 10
config_save() (*ocvproto.app.application.Application method*), 9
config_update() (*ocvproto.ui.wm.WindowManager method*), 10
contrast (*ocvproto.sources.camera.Camera attribute*), 14
create() (*ocvproto.ui.window.Window method*), 11

D

dec() (*ocvproto.ui.trackbars.base.Trackbar method*), 12
describe_properties() (*ocvproto.sources.camera.Camera method*), 14
describe_properties() (*ocvproto.sources.video.Video method*), 16
dilate() (*ocvproto.misc.canvas.Canvas method*), 18
dilate() (*ocvproto.sources.camera.Camera method*), 14
dilate() (*ocvproto.sources.image.Image method*), 12
dilate() (*ocvproto.sources.video.Video method*), 16
draw_rectangle() (*ocvproto.misc.canvas.Canvas method*), 18
draw_rectangle() (*ocvproto.sources.camera.Camera method*), 14
draw_rectangle() (*ocvproto.sources.image.Image method*), 12
draw_rectangle() (*ocvproto.sources.video.Video method*), 16
dump() (*ocvproto.misc.canvas.Canvas method*), 18
dump() (*ocvproto.sources.camera.Camera method*), 14
dump() (*ocvproto.sources.image.Image method*), 13
dump() (*ocvproto.sources.video.Video method*), 16
dump_image() (*ocvproto.sources.camera.Camera method*), 14
dump_image() (*ocvproto.sources.video.Video method*), 16
dump_setup() (*ocvproto.sources.camera.Camera method*), 14

`dump_setup()` (*ocvproto.sources.video.Video method*), 16

E

`exposure` (*ocvproto.sources.camera.Camera attribute*), 15

F

`fill()` (*ocvproto.misc.canvas.Canvas method*), 19
`fill()` (*ocvproto.sources.camera.Camera method*), 15
`fill()` (*ocvproto.sources.image.Image method*), 13
`fill()` (*ocvproto.sources.video.Video method*), 17
`focus` (*ocvproto.sources.video.Video attribute*), 17
`fps` (*ocvproto.sources.video.Video attribute*), 17
`frame` (*ocvproto.sources.camera.Camera attribute*), 15
`frame` (*ocvproto.sources.image.Image attribute*), 13
`frame` (*ocvproto.sources.video.Video attribute*), 17

G

`gain` (*ocvproto.sources.camera.Camera attribute*), 15
`gamma` (*ocvproto.sources.video.Video attribute*), 17
`get_data()` (*ocvproto.app.config.Config method*), 9
`get_image()` (*ocvproto.sources.camera.Camera method*), 15
`get_image()` (*ocvproto.sources.video.Video method*), 17
`get_value()` (*ocvproto.ui.trackbars.base.Trackbar method*), 12

H

`height` (*ocvproto.misc.canvas.Canvas attribute*), 19
`height` (*ocvproto.sources.image.Image attribute*), 13
`height` (*ocvproto.sources.video.Video attribute*), 17
`hook_register()` (*ocvproto.app.application.Application method*), 9
`hue` (*ocvproto.sources.camera.Camera attribute*), 15

I

`Image` (*class in ocvproto.sources.image*), 12
`inc()` (*ocvproto.ui.trackbars.base.Trackbar method*), 12
`iter_trackbars()` (*ocvproto.ui.wm.WindowManager method*), 10

K

`Key` (*class in ocvproto.app.keys*), 10

L

`Legend` (*class in ocvproto.primitives.legend*), 20
`load()` (*ocvproto.app.config.Config method*), 9
`loop()` (*ocvproto.app.application.Application method*), 9

M

`make_gray()` (*ocvproto.misc.canvas.Canvas method*), 19
`make_gray()` (*ocvproto.sources.camera.Camera method*), 15
`make_gray()` (*ocvproto.sources.image.Image method*), 13
`make_gray()` (*ocvproto.sources.video.Video method*), 17
`make_rgb()` (*ocvproto.misc.canvas.Canvas method*), 19
`make_rgb()` (*ocvproto.sources.camera.Camera method*), 15
`make_rgb()` (*ocvproto.sources.image.Image method*), 13
`make_rgb()` (*ocvproto.sources.video.Video method*), 17

O

`ocvproto.app.application` (*module*), 8
`ocvproto.app.config` (*module*), 9
`ocvproto.app.keys` (*module*), 10
`ocvproto.misc.canvas` (*module*), 18
`ocvproto.misc.colors` (*module*), 19
`ocvproto.misc.text` (*module*), 19
`ocvproto.primitives.legend` (*module*), 20
`ocvproto.sources.camera` (*module*), 13
`ocvproto.sources.image` (*module*), 12
`ocvproto.sources.video` (*module*), 15
`ocvproto.ui.trackbars.base` (*module*), 11
`ocvproto.ui.window` (*module*), 10
`ocvproto.ui.wm` (*module*), 10
`onChange()` (*ocvproto.ui.trackbars.base.Trackbar method*), 12

P

`position()` (*ocvproto.ui.window.Window method*), 11
`Property` (*class in ocvproto.sources.video*), 15
`put_on()` (*ocvproto.misc.text.Text method*), 19
`put_on()` (*ocvproto.primitives.legend.Legend method*), 20
`put_on_demo()` (*ocvproto.misc.text.Text class method*), 19

R

`read()` (*ocvproto.sources.camera.Camera method*), 15
`read()` (*ocvproto.sources.image.Image method*), 13
`read()` (*ocvproto.sources.video.Video method*), 17
`render()` (*ocvproto.ui.window.Window method*), 11
`render()` (*ocvproto.ui.wm.WindowManager method*), 10
`resize()` (*ocvproto.misc.canvas.Canvas method*), 19
`resize()` (*ocvproto.sources.camera.Camera method*), 15

`resize()` (*ocvproto.sources.image.Image method*), 13
`resize()` (*ocvproto.sources.video.Video method*), 17
`resize()` (*ocvproto.ui.window.Window method*), 11

S

`saturation` (*ocvproto.sources.camera.Camera attribute*), 15
`save()` (*ocvproto.app.config.Config method*), 9
`set_config()` (*ocvproto.app.application.Application method*), 9
`set_data()` (*ocvproto.app.config.Config method*), 9
`set_frame()` (*ocvproto.ui.window.Window method*), 11
`set_frame()` (*ocvproto.ui.wm.WindowManager method*), 10
`set_loop_func()` (*ocvproto.app.application.Application method*), 9
`set_property()` (*ocvproto.sources.camera.Camera method*), 15
`set_property()` (*ocvproto.sources.video.Video method*), 17
`sharpness` (*ocvproto.sources.video.Video attribute*), 17

T

`Text` (*class in ocvproto.misc.text*), 19
`to_rgb()` (*in module ocvproto.misc.colors*), 19
`Trackbar` (*class in ocvproto.ui.trackbars.base*), 11

V

`value` (*ocvproto.ui.trackbars.base.Trackbar attribute*), 12
`Video` (*class in ocvproto.sources.video*), 15

W

`width` (*ocvproto.misc.canvas.Canvas attribute*), 19
`width` (*ocvproto.sources.image.Image attribute*), 13
`width` (*ocvproto.sources.video.Video attribute*), 17
`Window` (*class in ocvproto.ui.window*), 10
`window` (*ocvproto.ui.wm.WindowManager attribute*), 10
`WindowManager` (*class in ocvproto.ui.wm*), 10

Z

`zoom` (*ocvproto.sources.video.Video attribute*), 18