moderngl_window Documentation

Release 1.3.0

Einar Forselv

Nov 11, 2019

Contents:

1 Programming Guide **1**
- 1.1 Installation . . . 1

2 API Reference **3**
- 2.1 moderngl_window . . . 3
- 2.2 moderngl_window.conf.Settings . . . 4
  - 2.2.1 Methods . . . 4
  - 2.2.2 Attributes . . . 6
- 2.3 moderngl_window.context . . . 8
  - 2.3.1 base.window.WindowConfig . . . 8
  - 2.3.2 base.BaseWindow . . . 13
  - 2.3.3 glfw.GLFW . . . 17
  - 2.3.4 headless.Headless . . . 21
  - 2.3.5 pyglet.Window . . . 25
  - 2.3.6 pyqt5Window . . . 29
  - 2.3.7 pyside2Window . . . 34
  - 2.3.8 sdl2.Window . . . 38
- 2.4 moderngl_window.geometry . . . 41
- 2.5 moderngl_window.loaders . . . 43
  - 2.5.1 base.BaseLoader . . . 43
  - 2.5.2 texture.t2d.Loader . . . 44
  - 2.5.3 program.single.Loader . . . 45
  - 2.5.4 program.separate.Loader . . . 47
  - 2.5.5 texture.array.Loader . . . 48
  - 2.5.6 scene.wavefront.Loader . . . 49
  - 2.5.7 scene.gltf2.Loader . . . 50
  - 2.5.8 scene.stl.Loader . . . 52
  - 2.5.9 data.json.Loader . . . 53
  - 2.5.10 data.text.Loader . . . 54
  - 2.5.11 data.binary.Loader . . . 55
- 2.6 moderngl_window.meta . . . 56
  - 2.6.1 base.ResourceDescription . . . 56
  - 2.6.2 texture.TextureDescription . . . 57
  - 2.6.3 program.ProgramDescription . . . 59
  - 2.6.4 scene.SceneDescription . . . 61
  - 2.6.5 data.DataDescription . . . 63

2.7 moderngl_window.finders . . . 64
2.7.1 base.BaseFilesystemFinder . . . 64
2.7.2 texture.FilesystemFinder . . . 65
2.7.3 program.FilesystemFinder . . . 65
2.7.4 scene.FilesystemFinder . . . 65
2.7.5 data.FilesystemFinder . . . 66
2.8 moderngl_window.opengl . . . 66
2.8.1 opengl.projection.Projection3D . . . 66
2.8.2 opengl.vao.VAO . . . 67
2.9 moderngl_window.resources . . . 70
2.9.1 base.BaseRegistry . . . 70
2.9.2 textures.Textures . . . 71
2.9.3 programs.Programs . . . 72
2.9.4 scenes.Scenes . . . 73
2.9.5 base.DataFiles . . . 74
2.10 moderngl_window.timers . . . 74
2.10.1 base.BaseTimer . . . 74
2.10.2 clock.Timer . . . 75
2.11 moderngl_window.scene . . . 76
2.11.1 Camera . . . 76
2.11.2 KeyboardCamera . . . 77
2.11.3 Scene . . . 79
2.11.4 Node . . . 80
2.11.5 Mesh . . . 81
2.11.6 Material . . . 83
2.11.7 MaterialTexture . . . 83
2.11.8 MeshProgram . . . 84

3 Indices and tables **85**

Python Module Index **87**

Index **89**

CHAPTER 1

Programming Guide

1.1 Installation

Install instructions

CHAPTER 2

API Reference

2.1 moderngl_window

General helper functions aiding in the boostrapping of this library.

moderngl_window.**setup_basic_logging**(*level: int*)
Set up basic logging

Parameters **level** (*int*) – The log level

moderngl_window.**activate_context**(*window: moderngl_window.context.base.window.BaseWindow = None*, *ctx: moderngl.context.Context = None*)
Register the active window and context. If only a window is supplied the context is taken from the window.

Keyword Arguments

- **window** (window) – The currenty active window
- **ctx** (*moderngl.Context*) – The active moderngl context

moderngl_window.**window**()
Obtain the active window

moderngl_window.**ctx**()
Obtain the active context

moderngl_window.**get_window_cls**(*window: str = None*) → Type[moderngl_window.context.base.window.BaseWindow]
Attept to obtain a window class using the full dotted python path. This can be used to import custom or modified window classes.

Parameters **window** (*str*) – Name of the window

Returns A reference to the requested window class. Raises exception if not found.

moderngl_window.**get_local_window_cls**(*window: str = None*) → Type[moderngl_window.context.base.window.BaseWindow]
Attept to obtain a window class in the moderngl_window package using short window names such as `pyglet` or `glfw`.

Parameters **window** (*str*) – Name of the window

Returns A reference to the requested window class. Raises exception if not found.

moderngl_window.**find_window_classes**() → List[str]
Find available window packages

Returns A list of avaialble window packages

moderngl_window.**create_window_from_settings**() → moderngl_window.context.base.window.BaseWindow
Creates a window using configured values in *moderngl_window.conf.Settings.WINDOW*. This will also activate the window/context.

Returns The Window instance

moderngl_window.**run_window_config**(*config_cls: moderngl_window.context.base.window.WindowConfig*, *timer=None*, *args=None*) → None
Run an WindowConfig entering a blocking main loop

Parameters

- **config_cls** – The WindowConfig class to render
- **args** – Override sys.args

moderngl_window.**parse_args**(*args=None*)
Parse arguments from sys.argv

2.2 moderngl_window.conf.Settings

moderngl_window.conf.**Settings**
Bag of settings values. New attributes can be freely added runtime. Various apply* methods are supplied so the user have full control over how settings values are initialized. This is especially useful for more custom usage. And instance of the *Settings* class is created when the *conf* module is imported.

Attribute names must currently be in upper case to be recognized.

Some examples of usage:

```
from moderngl_window.conf import settings

# Mandatory settings values
try:
    value = settings.VALUE
except KeyError:
    raise ValueError("This settings value is required")

# Fallback in code
value = getattr(settings, 'VALUE', 'default_value')

# Pretty printed string represenation for easy inspection
print(settings)
```

2.2.1 Methods

Settings.**__init__**()
Initialize settings with default values

Settings.**apply_default_settings**() → None
Apply keys and values from the default settings module located in this package. This is to ensure we always have the minimnal settings for the system to run.

If replacing or customizing the settings class you must always apply default settings to ensure compatibility when new settings are added.

Settings.**apply_settings_from_env**() → None
Apply settings from MODERNGL_WINDOW_SETTINGS_MODULE environment variable. If the enviroment variable is undefined no action will be taken. Normally this would be used to easily be able to switch between different configuration by setting env vars before executing the program.

Example:

```
import os
from moderngl_window.conf import settings

os.environ['MODERNGL_WINDOW_SETTINGS_MODULE'] = 'python.path.to.module'
settings.apply_settings_from_env()
```

Raises **ImproperlyConfigured if the module was not found** –

Settings.**apply_from_module_name**(*settings_module_name: str*) → None
Apply settings from a python module by supplying the full pythonpath to the module.

Parameters **settings_module_name** (*str*) – Full python path to the module

Raises **ImproperlyConfigured if the module was not found** –

Settings.**apply_from_dict**(*data: dict*) → None
Apply settings values from a dictionary

Example:

```
>> from moderngl_window.conf import settings
>> settings.apply_dict({'SOME_VALUE': 1})
>> settings.SOME_VALUE
1
```

Settings.**apply_from_module**(*module: module*) → None
Apply settings values from a python module

Example:

```
my_settings.py module containing the following line:
SOME_VALUE = 1

>> from moderngl_window.conf import settings
>> import my_settings
>> settings.apply_module(my_settings)
>> settings.SOME_VALUE
1
```

Settings.**apply_from_cls**(*cls*) → None
Apply settings values from a class namespace

Example:

```
>> from moderngl_window.conf import settings
>> class MySettings:
>>     SOME_VALUE = 1
>>
>> settings.apply(MySettings)
>> settings.SOME_VALUE
1
```

Settings.**apply_from_iterable**(*iterable: Union[collections.abc.Iterable, generator]*) → None
Apply (key, value) pairs from an interable or generator

Settings.**to_dict**()
Create a dict represenation of the settings Only uppercase attributes are included

Returns dict represenation

Return type dict

2.2.2 Attributes

Settings.**WINDOW**
Window/screen properties. Most importantly the `class` attribute decides what class should be used to handle the window.

```
# Default values
WINDOW = {
    "gl_version": (3, 3),
    "class": "moderngl_window.context.pyglet.Window",
    "size": (1280, 720),
    "aspect_ratio": 16 / 9,
    "fullscreen": False,
    "resizable": True,
    "title": "ModernGL Window",
    "vsync": True,
    "cursor": True,
    "samples": 0,
}
```

Other Properties:

- `gl_version`: The minimum required major/minor OpenGL version
- `size`: The window size to open.
- `aspect_ratio` is the enforced aspect ratio of the viewport.
- `fullscreen`: True if you want to create a context in fullscreen mode
- `resizable`: If the window should be resizable. This only applies in windowed mode.
- `vsync`: Only render one frame per screen refresh
- `title`: The visible title on the window in windowed mode
- `cursor`: Should the mouse cursor be visible on the screen? Disabling this is also useful in windowed mode when controlling the camera on some platforms as moving the mouse outside the window can cause issues.
- `Samples`: Number if samples used in multusampling. Values above 1 enables multisampling.

The created window frame buffer will by default use:

- RGBA8 (32 bit per pixel)
- 24 bit depth buffer
- Double buffering
- color and depth buffer is cleared for every frame

`Settings.`**`SCREENSHOT_PATH`**
Absolute path to the directory screenshots will be saved by the screenshot module. Screenshots will end up in the project root of not defined. If a path is configured, the directory will be auto-created.

`Settings.`**`PROGRAM_FINDERS`**
Finder classes for locating programs/shaders.

```
# Default values
PROGRAM_FINDERS = [
    "moderngl_window.finders.program.FileSystemFinder",
]
```

`Settings.`**`TEXTURE_FINDERS`**
Finder classes for locating textures.

```
# Default values
TEXTURE_FINDERS = [
    "moderngl_window.finders.texture.FileSystemFinder",
]
```

`Settings.`**`SCENE_FINDERS`**
Finder classes for locating scenes.

```
# Default values
SCENE_FINDERS = [
    "moderngl_window.finders.scene.FileSystemFinder",
]
```

`Settings.`**`DATA_FINDERS`**
Finder classes for locating data files.

```
# Default values
DATA_FINDERS = [
    "moderngl_window.finders.data.FileSystemFinder",
]
```

`Settings.`**`PROGRAM_DIRS`**
Lists of *str* or *pathlib.Path* used by `FileSystemFinder` to looks for programs/shaders.

`Settings.`**`TEXTURE_DIRS`**
Lists of *str* or *pathlib.Path* used by `FileSystemFinder` to looks for textures.

`Settings.`**`SCENE_DIRS`**
Lists of *str* or *pathlib.Path* used by `FileSystemFinder` to looks for scenes (obj, gltf, stl etc).

`Settings.`**`DATA_DIRS`**
Lists of *str* or *pathlib.Path* used by `FileSystemFinder` to looks for data files.

`Settings.`**`PROGRAM_LOADERS`**
Classes responsible for loading programs/shaders.

```
# Default values
PROGRAM_LOADERS = [
    'moderngl_window.loaders.program.single.Loader',
    'moderngl_window.loaders.program.separate.Loader',
]
```

Settings.**TEXTURE_LOADERS**
Classes responsible for loading textures.

```
# Default values
TEXTURE_LOADERS = [
    'moderngl_window.loaders.texture.t2d.Loader',
    'moderngl_window.loaders.texture.array.Loader',
]
```

Settings.**SCENE_LOADERS**
Classes responsible for loading scenes.

```
# Default values
SCENE_LOADERS = [
    "moderngl_window.loaders.scene.gltf.GLTF2",
    "moderngl_window.loaders.scene.wavefront.ObjLoader",
    "moderngl_window.loaders.scene.stl_loader.STLLoader",
]
```

Settings.**DATA_LOADERS**
Classes responsible for loading data files.

```
# Default values
DATA_LOADERS = [
    'moderngl_window.loaders.data.binary.Loader',
    'moderngl_window.loaders.data.text.Loader',
    'moderngl_window.loaders.data.json.Loader',
]
```

2.3 moderngl_window.context

2.3.1 base.window.WindowConfig

moderngl_window.context.base.window.**WindowConfig**
Creating a WindowConfig instance is the simplest interface this library provides to open and window, handle inputs and provide simple shortcut method for loading basic resources. It's appropriate for projects with basic needs.

Example:

```
class MyConfig(mglw.WindowConfig):
    gl_version = (3, 3)
    window_size = (1920, 1080)
    aspect_ratio = 16 / 9
    title = "My Config"
    resizable = False
    samples = 8
```

(continues on next page)

(continued from previous page)

```
    def __init__(self, **kwargs):
        super().__init__(**kwargs)
        # Do other initialization here

    def render(self, time: float, frametime: float):
        # Render stuff here with ModernGL

    def resize(self, width: int, height: int):
        print("Window was resized. buffer size is {} x {}".format(width, height))

    def mouse_position_event(self, x, y):
        print("Mouse position:", x, y)

    def mouse_press_event(self, x, y, button):
        print("Mouse button {} pressed at {}, {}".format(button, x, y))

    def mouse_release_event(self, x: int, y: int, button: int):
        print("Mouse button {} released at {}, {}".format(button, x, y))

    def key_event(self, key, action, modifiers):
        print(key, action, modifiers)
```

Methods

WindowConfig.**__init__**(*ctx: moderngl.context.Context = None*, *wnd: moderngl_window.context.base.window.BaseWindow = None*, *timer: moderngl_window.timers.base.BaseTimer = None*, ***kwargs*)

Initialize the window config

Keyword Arguments

- **ctx** (*moderngl.Context*) – The moderngl context
- **wnd** – The window instance
- **timer** – The timer instance

WindowConfig.**render**(*time: float*, *frame_time: float*)

Renders the assigned effect

Parameters

- **time** (*float*) – Current time in seconds
- **frame_time** (*float*) – Delta time from last frame in seconds

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

Called every time the window is resized in case the we need to do internal adjustments.

Parameters

- **width** (*int*) – width in buffer size (not window size)
- **height** (*int*) – height in buffer size (not window size)

WindowConfig.**key_event**(*key: Any*, *action: Any*, *modifiers: moderngl_window.context.base.keys.KeyModifiers*)

Called for every key press and release. Depending on the library used, key events may trigger repeating events during the pressed duration based on the configured key repeat on the users operating system.

Parameters

- **key** – The key that was press. Compare with self.wnd.keys.
- **action** – self.wnd.keys.ACTION_PRESS or ACTION_RELEASE
- **modifiers** – Modifier state for shift and ctrl

WindowConfig.**mouse_position_event**(*x: int*, *y: int*)
Reports the current mouse cursor position in the window

Parameters

- **x** (*int*) – X postion of the mouse cursor
- **Iint)** (*y*) – Y position of the mouse cursor

WindowConfig.**mouse_press_event**(*x: int*, *y: int*, *button: int*)
Called when a mouse button in pressed

Parameters

- **x** (*int*) – X position the press occured
- **y** (*int*) – Y position the press occured
- **button** (*int*) – 1 = Left button, 2 = right button

WindowConfig.**mouse_release_event**(*x: int*, *y: int*, *button: int*)
Called when a mouse button in released

Parameters

- **x** (*int*) – X position the release occured
- **y** (*int*) – Y position the release occured
- **button** (*int*) – 1 = Left button, 2 = right button

WindowConfig.**load_texture_2d**(*path: str*, *flip=True*, *mipmap=False*, *mipmap_levels: Tuple[int, int] = None*, *anisotropy=1.0*, ***kwargs*) → moderngl.texture.Texture
Loads a 2D texture

Parameters **path** (*str*) – Path to the texture relative to search directories

Keyword Arguments

- **flip** (*boolean*) – Flip the image horisontally
- **mipmap** (*bool*) – Generate mipmaps. Will generate max possible levels unless *mipmap_levels* is defined.
- **mipmap_levels** (*tuple*) – (base, max_level) controlling mipmap generation. When defined the *mipmap* parameter is automatically *True*
- **anisotropy** (*float*) – Number of samples for anisotropic filtering
- ****kwargs** – Additonal parameters to TextureDescription

Returns Texture instance

Return type moderngl.Texture

WindowConfig.**load_texture_array**(*path: str*, *layers: int = 0*, *flip=True*, *mipmap=False*, *mipmap_levels: Tuple[int, int] = None*, *anisotropy=1.0*, ***kwargs*) → moderngl.texture_array.TextureArray
Loads a texture array.

Parameters **path** (*str*) – Path to the texture relative to search directories

Keyword Arguments

- **layers** (*int*) – How many layers to split the texture into vertically
- **flip** (*boolean*) – Flip the image horisontally
- **mipmap** (*bool*) – Generate mipmaps. Will generate max possible levels unless *mipmap_levels* is defined.
- **mipmap_levels** (*tuple*) – (base, max_level) controlling mipmap generation. When defined the *mipmap* parameter is automatically *True*
- **anisotropy** (*float*) – Number of samples for anisotropic filtering
- ****kwargs** – Additonal parameters to TextureDescription

Returns The texture instance

Return type moderngl.TextureArray

WindowConfig.**load_program**(*path=None*, *vertex_shader=None*, *geometry_shader=None*, *fragment_shader=None*, *tess_control_shader=None*, *tess_evaluation_shader=None*) → moderngl.program.Program

Loads a shader program.

Note that *path* should only be used if all shaders are defined in the same glsl file separated by defines.

Keyword Arguments

- **path** (*str*) – Path to a single glsl file
- **vertex_shader** (*str*) – Path to vertex shader
- **geometry_shader** (*str*) – Path to geometry shader
- **fragment_shader** (*str*) – Path to fragment shader
- **tess_control_shader** (*str*) – Path to tessellation control shader
- **tess_evaluation_shader** (*str*) – Path to tessellation eval shader

Returns The program instance

Return type moderngl.Program

WindowConfig.**load_text**(*path: str*, ***kwargs*) → str

Load a text file.

Parameters

- **path** (*str*) – Path to the file relative to search directories
- ****kwargs** – Additional parameters to DataDescription

Returns Contents of the text file

Return type str

WindowConfig.**load_json**(*path: str*, ***kwargs*) → dict

Load a json file

Parameters

- **path** (*str*) – Path to the file relative to search directories
- ****kwargs** – Additional parameters to DataDescription

Returns Contents of the json file

Return type dict

WindowConfig.**load_binary**(*path: str*, ***kwargs*) → bytes
Load a file in binary mode.

Parameters

- **path** (*str*) – Path to the file relative to search directories
- ****kwargs** – Additional parameters to DataDescription

Returns The byte data of the file

Return type bytes

WindowConfig.**load_scene**(*path: str*, *cache=False*, *attr_names=<class 'moderngl_window.geometry.attributes.AttributeNames'>*, *kind=None*, ***kwargs*) → moderngl_window.scene.scene.Scene
Loads a scene.

Keyword Arguments

- **path** (*str*) – Path to the file relative to search directories
- **cache** (*str*) – Use the loader caching system if present
- **attr_names** (*AttributeNames*) – Attrib name config
- **kind** (*str*) – Override loader kind
- ****kwargs** – Additional parameters to SceneDescription

Returns The scene instance

Return type Scene

Attributes

WindowConfig.**window_size**
Size of the window.

```
# Default value
window_size = (1280, 720)
```

WindowConfig.**resizable**
Determines of the window should be resizable

```
# Default value
resizable = True
```

WindowConfig.**gl_version**
The minimum required OpenGL version required

```
# Default value
gl_version = (3, 3)
```

WindowConfig.**title**
Title of the window

```
# Default value
title = "Example"
```

WindowConfig.**aspect_ratio**
The endorced aspect ratio of the viewport. When specified back borders will be calulcated both vertically and horizontally if needed.

This property can be set to `None` to disable the fixed viewport system.

```
# Default value
aspect_ratio = 16 / 9
```

WindowConfig.**cursor**
Determines if the mouse cursor should be visible inside the window. If enabled on some platforms

```
# Default value
cursor = True
```

WindowConfig.**samples**
Number of samples to use in multisampling.

```
# Default value
samples = 4
```

WindowConfig.**resource_dir**
Absolute path to your resource directory containing textures, scenes, shaders/programs or data files. The `load_` methods in this class will look for resources in this path. This attribute can be a `str` or a `pathlib.Path`.

```
# Default value
resource_dir = None
```

WindowConfig.**log_level**
Sets the log level for this library using the standard *logging* module.

```
# Default value
log_level = logging.INFO
```

2.3.2 base.BaseWindow

Methods

BaseWindow.**__init__**(*title='ModernGL', gl_version=(3, 3), size=(1280, 720), resizable=True, fullscreen=False, vsync=True, aspect_ratio=1.7777777777777777, samples=4, cursor=True, **kwargs*)
Initialize a window instance.

Parameters

- **title** (*str*) – The window title
- **gl_version** (*tuple*) – Major and minor version of the opengl context to create
- **size** (*tuple*) – Window size x, y
- **resizable** (*bool*) – Should the window be resizable?
- **fullscreen** (*bool*) – Open window in fullsceeen mode
- **vsync** (*bool*) – Enable/disable vsync
- **aspect_ratio** (*float*) – The desired aspect ratio. Can be set to None.

- **samples** (*int*) – Number of MSAA samples for the default framebuffer
- **cursor** (*bool*) – Enable/disable displaying the cursor inside the window

BaseWindow.**init_mgl_context**() → None
Create or assign a ModernGL context. If no context is supplied a context will be created using the window's gl_version.

Keyword Arguments **ctx** – An optional custom ModernGL context

BaseWindow.**is_key_pressed**(*key*) → bool
Returns: The press state of a key

BaseWindow.**close**() → None
Signal for the window to close

BaseWindow.**use**()
Bind the window's framebuffer

BaseWindow.**clear**(*red=0.0*, *green=0.0*, *blue=0.0*, *alpha=0.0*, *depth=1.0*, *viewport=None*)
Binds and clears the default framebuffer

Parameters

- **red** (*float*) – color component
- **green** (*float*) – color component
- **blue** (*float*) – color component
- **alpha** (*float*) – alpha component
- **depth** (*float*) – depth value
- **viewport** (*tuple*) – The viewport

BaseWindow.**render**(*time=0.0*, *frame_time=0.0*) → None
Renders a frame by calling the configured render callback

Keyword Arguments

- **time** (*float*) – Current time in seconds
- **frame_time** (*float*) – Delta time from last frame in seconds

BaseWindow.**swap_buffers**() → None
Library specific buffer swap method. Must be overridden.

BaseWindow.**resize**(*width*, *height*) → None
Should be called every time window is resized so the example can adapt to the new size if needed

BaseWindow.**destroy**() → None
A library specific destroy method is required

BaseWindow.**set_default_viewport**() → None
Calculates the viewport based on the configured aspect ratio. Will add black borders and center the viewport if the window do not match the configured viewport.

If aspect ratio is None the viewport will be scaled to the entire window size regardless of size.

BaseWindow.**print_context_info**()
Prints moderngl context info.

Attributes

BaseWindow.**keys**
Window specific key constants

BaseWindow.**ctx**
The ModernGL context for the window

Type moderngl.Context

BaseWindow.**fbo**
The default framebuffer

Type moderngl.Framebuffer

BaseWindow.**title**
Window title

Type str

BaseWindow.**gl_version**
(major, minor) required OpenGL version

Type Tuple[int, int]

BaseWindow.**width**
The current window width

Type int

BaseWindow.**height**
The current window height

Type int

BaseWindow.**size**
current window size

Type Tuple[int, int]

BaseWindow.**buffer_size**
tuple with the current window buffer size

Type Tuple[int, int]

BaseWindow.**pixel_ratio**
The frambuffer/window size ratio

Type float

BaseWindow.**viewport**
current window viewport

Type Tuple[int, int, int, int]

BaseWindow.**frames**
Number of frames rendered

Type int

BaseWindow.**resizable**
Window is resizable

Type bool

BaseWindow.**fullscreen**
Window is in fullscreen mode

Type bool

BaseWindow.**config**
Get the current WindowConfig instance

BaseWindow.**vsync**
vertical sync enabled/disabled

Type bool

BaseWindow.**aspect_ratio**
Aspect ratio configured for the viewport

Type float

BaseWindow.**samples**
Number of Multisample anti-aliasing (MSAA) samples

Type float

BaseWindow.**cursor**
Should the mouse cursor be visible inside the window?

Type bool

BaseWindow.**render_func**
The render callable

This property can also be used to assign a callable.

Type callable

BaseWindow.**resize_func**
The resize callable

This property can also be used to assign a callable.

Type callable

BaseWindow.**key_event_func**
The key_event callable

This property can also be used to assign a callable.

Type callable

BaseWindow.**mouse_position_event_func**
The mouse_position callable

This property can also be used to assign a callable.

Type callable

BaseWindow.**mouse_press_event_func**
The mouse_press callable

This property can also be used to assign a callable.

Type callable

BaseWindow.**mouse_release_event_func**
The mouse_release callable

This property can also be used to assign a callable.

Type callable

BaseWindow.**is_closing**
Is the window about to close?

Type bool

BaseWindow.**modifiers**
(KeyModifiers) The current keyboard modifiers

BaseWindow.**gl_version_code**
Generates the version code integer for the selected OpenGL version.

gl_version (4, 1) returns 410

Type int

2.3.3 glfw.GLFW

Methods

Window.**__init__**(***kwargs*)
Initialize a window instance.

Parameters

- **title** (*str*) – The window title
- **gl_version** (*tuple*) – Major and minor version of the opengl context to create
- **size** (*tuple*) – Window size x, y
- **resizable** (*bool*) – Should the window be resizable?
- **fullscreen** (*bool*) – Open window in fullsceeen mode
- **vsync** (*bool*) – Enable/disable vsync
- **aspect_ratio** (*float*) – The desired aspect ratio. Can be set to None.
- **samples** (*int*) – Number of MSAA samples for the default framebuffer
- **cursor** (*bool*) – Enable/disable displaying the cursor inside the window

Window.**init_mgl_context**() → None
Create or assign a ModernGL context. If no context is supplied a context will be created using the window's gl_version.

Keyword Arguments **ctx** – An optional custom ModernGL context

Window.**is_key_pressed**(*key*) → bool
Returns: The press state of a key

Window.**close**() → None
Suggest to glfw the window should be closed soon

Window.**use**()
Bind the window's framebuffer

Window.**clear**(*red=0.0*, *green=0.0*, *blue=0.0*, *alpha=0.0*, *depth=1.0*, *viewport=None*)
Binds and clears the default framebuffer

Parameters

- **red** (*float*) – color component

- **green** (*float*) – color component
- **blue** (*float*) – color component
- **alpha** (*float*) – alpha component
- **depth** (*float*) – depth value
- **viewport** (*tuple*) – The viewport

Window.**render**(*time=0.0*, *frame_time=0.0*) → None
Renders a frame by calling the configured render callback

Keyword Arguments

- **time** (*float*) – Current time in seconds
- **frame_time** (*float*) – Delta time from last frame in seconds

Window.**swap_buffers**()
Swap buffers, increment frame counter and pull events

Window.**resize**(*width*, *height*) → None
Should be called every time window is resized so the example can adapt to the new size if needed

Window.**destroy**()
Gracefully terminate GLFW

Window.**set_default_viewport**() → None
Calculates the viewport based on the configured aspect ratio. Will add black borders and center the viewport if the window do not match the configured viewport.

If aspect ratio is None the viewport will be scaled to the entire window size regardless of size.

Window.**print_context_info**()
Prints moderngl context info.

Window Specific Methods

Window.**glfw_window_resize_callback**(*window*, *width*, *height*)
Window resize callback for glfw

Parameters

- **window** – The window
- **width** – New width
- **height** – New height

Window.**glfw_mouse_event_callback**(*window*, *xpos*, *ypos*)
Mouse event callback from glfw. Translates the events forwarding them to `cursor_event()`.

Parameters

- **window** – The window
- **xpos** – viewport x pos
- **ypos** – viewport y pos

Window.**glfw_mouse_button_callback**(*window*, *button*, *action*, *mods*)
Handle mouse button events and forward them to the example

Parameters

- **window** – The window
- **button** – The button creating the event
- **action** – Button action (press or release)
- **mods** – They modifiers such as ctrl or shift

Window.**glfw_key_event_callback**(*window*, *key*, *scancode*, *action*, *mods*)
Key event callback for glfw. Translates and forwards keyboard event to `keyboard_event()`

Parameters

- **window** – Window event origin
- **key** – The key that was pressed or released.
- **scancode** – The system-specific scancode of the key.
- **action** – `GLFW_PRESS`, `GLFW_RELEASE` or `GLFW_REPEAT`
- **mods** – Bit field describing which modifier keys were held down.

Attributes

Window.**keys**
GLFW specific key constants

Window.**ctx**
The ModernGL context for the window

Type moderngl.Context

Window.**fbo**
The default framebuffer

Type moderngl.Framebuffer

Window.**title**
Window title

Type str

Window.**gl_version**
(major, minor) required OpenGL version

Type Tuple[int, int]

Window.**width**
The current window width

Type int

Window.**height**
The current window height

Type int

Window.**size**
current window size

Type Tuple[int, int]

Window.**buffer_size**
tuple with the current window buffer size

Type Tuple[int, int]

Window.**pixel_ratio**
The frambuffer/window size ratio

Type float

Window.**viewport**
current window viewport

Type Tuple[int, int, int, int]

Window.**frames**
Number of frames rendered

Type int

Window.**resizable**
Window is resizable

Type bool

Window.**fullscreen**
Window is in fullscreen mode

Type bool

Window.**config**
Get the current WindowConfig instance

Window.**vsync**
vertical sync enabled/disabled

Type bool

Window.**aspect_ratio**
Aspect ratio configured for the viewport

Type float

Window.**samples**
Number of Multisample anti-aliasing (MSAA) samples

Type float

Window.**cursor**
Should the mouse cursor be visible inside the window?

Type bool

Window.**render_func**
The render callable

This property can also be used to assign a callable.

Type callable

Window.**resize_func**
The resize callable

This property can also be used to assign a callable.

Type callable

Window.**key_event_func**
The key_event callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_position_event_func**
The mouse_position callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_press_event_func**
The mouse_press callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_release_event_func**
The mouse_release callable

This property can also be used to assign a callable.

Type callable

Window.**is_closing**
Checks if the window is scheduled for closing

Type bool

Window.**modifiers**
(KeyModifiers) The current keyboard modifiers

Window.**gl_version_code**
Generates the version code integer for the selected OpenGL version.

gl_version (4, 1) returns 410

Type int

2.3.4 headless.Headless

Methods

Window.**__init__**(***kwargs*)
Initialize a window instance.

Parameters

- **title** (*str*) – The window title
- **gl_version** (*tuple*) – Major and minor version of the opengl context to create
- **size** (*tuple*) – Window size x, y
- **resizable** (*bool*) – Should the window be resizable?
- **fullscreen** (*bool*) – Open window in fullsceeen mode
- **vsync** (*bool*) – Enable/disable vsync
- **aspect_ratio** (*float*) – The desired aspect ratio. Can be set to None.
- **samples** (*int*) – Number of MSAA samples for the default framebuffer
- **cursor** (*bool*) – Enable/disable displaying the cursor inside the window

Window.**init_mgl_context**() → None
Create an standalone context and framebuffer

Window.**is_key_pressed**(*key*) → bool
Returns: The press state of a key

Window.**close**() → None
Signal for the window to close

Window.**use**()
Bind the window's framebuffer

Window.**clear**(*red=0.0*, *green=0.0*, *blue=0.0*, *alpha=0.0*, *depth=1.0*, *viewport=None*)
Binds and clears the default framebuffer

Parameters

- **red** (*float*) – color component
- **green** (*float*) – color component
- **blue** (*float*) – color component
- **alpha** (*float*) – alpha component
- **depth** (*float*) – depth value
- **viewport** (*tuple*) – The viewport

Window.**render**(*time=0.0*, *frame_time=0.0*) → None
Renders a frame by calling the configured render callback

Keyword Arguments

- **time** (*float*) – Current time in seconds
- **frame_time** (*float*) – Delta time from last frame in seconds

Window.**swap_buffers**() → None
Placeholder. We currently don't do double buffering in headless mode. This may change in the future.

Window.**resize**(*width*, *height*) → None
Should be called every time window is resized so the example can adapt to the new size if needed

Window.**destroy**() → None
A library specific destroy method is required

Window.**set_default_viewport**() → None
Calculates the viewport based on the configured aspect ratio. Will add black borders and center the viewport if the window do not match the configured viewport.

If aspect ratio is None the viewport will be scaled to the entire window size regardless of size.

Window.**print_context_info**()
Prints moderngl context info.

Attributes

Window.**keys**

Window.**ctx**
The ModernGL context for the window

Type moderngl.Context

Window.**fbo**
The default framebuffer

Type moderngl.Framebuffer

Window.**title**
Window title

Type str

Window.**gl_version**
(major, minor) required OpenGL version

Type Tuple[int, int]

Window.**width**
The current window width

Type int

Window.**height**
The current window height

Type int

Window.**size**
current window size

Type Tuple[int, int]

Window.**buffer_size**
tuple with the current window buffer size

Type Tuple[int, int]

Window.**pixel_ratio**
The frambuffer/window size ratio

Type float

Window.**viewport**
current window viewport

Type Tuple[int, int, int, int]

Window.**frames**
Number of frames rendered

Type int

Window.**resizable**
Window is resizable

Type bool

Window.**fullscreen**
Window is in fullscreen mode

Type bool

Window.**config**
Get the current WindowConfig instance

Window.**vsync**
vertical sync enabled/disabled

Type bool

Window.**aspect_ratio**
Aspect ratio configured for the viewport

Type float

Window.**samples**
Number of Multisample anti-aliasing (MSAA) samples

Type float

Window.**cursor**
Should the mouse cursor be visible inside the window?

Type bool

Window.**render_func**
The render callable

This property can also be used to assign a callable.

Type callable

Window.**resize_func**
The resize callable

This property can also be used to assign a callable.

Type callable

Window.**key_event_func**
The key_event callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_position_event_func**
The mouse_position callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_press_event_func**
The mouse_press callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_release_event_func**
The mouse_release callable

This property can also be used to assign a callable.

Type callable

Window.**is_closing**
Is the window about to close?

Type bool

Window.**modifiers**
(KeyModifiers) The current keyboard modifiers

Window.**gl_version_code**
Generates the version code integer for the selected OpenGL version.

gl_version (4, 1) returns 410

Type int

2.3.5 pyglet.Window

Methods

Window.**__init__**(***kwargs*)
Initialize a window instance.

Parameters

- **title** (*str*) – The window title
- **gl_version** (*tuple*) – Major and minor version of the opengl context to create
- **size** (*tuple*) – Window size x, y
- **resizable** (*bool*) – Should the window be resizable?
- **fullscreen** (*bool*) – Open window in fullsceeen mode
- **vsync** (*bool*) – Enable/disable vsync
- **aspect_ratio** (*float*) – The desired aspect ratio. Can be set to None.
- **samples** (*int*) – Number of MSAA samples for the default framebuffer
- **cursor** (*bool*) – Enable/disable displaying the cursor inside the window

Window.**init_mgl_context**() → None
Create or assign a ModernGL context. If no context is supplied a context will be created using the window's gl_version.

Keyword Arguments **ctx** – An optional custom ModernGL context

Window.**is_key_pressed**(*key*) → bool
Returns: The press state of a key

Window.**close**() → None
Close the pyglet window directly

Window.**use**()
Bind the window's framebuffer

Window.**clear**(*red=0.0*, *green=0.0*, *blue=0.0*, *alpha=0.0*, *depth=1.0*, *viewport=None*)
Binds and clears the default framebuffer

Parameters

- **red** (*float*) – color component
- **green** (*float*) – color component
- **blue** (*float*) – color component
- **alpha** (*float*) – alpha component
- **depth** (*float*) – depth value
- **viewport** (*tuple*) – The viewport

Window.**render**(*time=0.0*, *frame_time=0.0*) → None
Renders a frame by calling the configured render callback

Keyword Arguments

- **time** (*float*) – Current time in seconds
- **frame_time** (*float*) – Delta time from last frame in seconds

Window.**swap_buffers**() → None
Swap buffers, increment frame counter and pull events

Window.**resize**(*width*, *height*) → None
Should be called every time window is resized so the example can adapt to the new size if needed

Window.**destroy**()
Destroy the pyglet window

Window.**set_default_viewport**() → None
Calculates the viewport based on the configured aspect ratio. Will add black borders and center the viewport if the window do not match the configured viewport.

If aspect ratio is None the viewport will be scaled to the entire window size regardless of size.

Window.**print_context_info**()
Prints moderngl context info.

Window Specific Methods

Window.**on_mouse_press**(*x: int*, *y: int*, *button*, *mods*)
Handle mouse press events and forward to standard methods

Parameters

- **x** – x position of the mouse when pressed
- **y** – y position of the mouse when pressed
- **button** – The pressed button
- **mods** – Modifiers

Window.**on_key_release**(*symbol*, *modifiers*)
Pyglet specific key release callback.

Forwards and translates the events to standard methods.

Parameters

- **symbol** – The symbol of the pressed key
- **modifiers** – Modifier state (shift, ctrl etc.)

Window.**on_mouse_drag**(*x*, *y*, *dx*, *dy*, *buttons*, *modifiers*)
Pyglet specific mouse drag event.

When a mouse button is pressed this is the only way to capture mouse posision events

Window.**on_key_press**(*symbol*, *modifiers*)
Pyglet specific key press callback.

Forwards and translates the events to the standard methods.

Parameters

- **symbol** – The symbol of the pressed key

- **modifiers** – Modifier state (shift, ctrl etc.)

Window.**on_mouse_release**(*x: int*, *y: int*, *button*, *mods*)
Handle mouse release events and forward to standard methods

Parameters

- **x** – x position when moutse button was released
- **y** – y position when moutse button was released
- **button** – The button pressed
- **mods** – Modifiers

Window.**on_mouse_motion**(*x*, *y*, *dx*, *dy*)
Pyglet specific mouse motion callback.

Forwards and traslates the event to the standard methods.

Parameters

- **x** – x position of the mouse
- **y** – y position of the mouse
- **dx** – delta x position
- **dy** – delta y position of the mouse

Window.**on_resize**(*width: int*, *height: int*)
Pyglet specific callback for window resize events forwarding to standard methods

Parameters

- **width** – New window width
- **height** – New window height

Attributes

Window.**keys**
Pyglet specific key constants

Window.**ctx**
The ModernGL context for the window

Type moderngl.Context

Window.**fbo**
The default framebuffer

Type moderngl.Framebuffer

Window.**title**
Window title

Type str

Window.**gl_version**
(major, minor) required OpenGL version

Type Tuple[int, int]

Window.**width**
The current window width

Type int

Window.**height**
The current window height

Type int

Window.**size**
current window size

Type Tuple[int, int]

Window.**buffer_size**
tuple with the current window buffer size

Type Tuple[int, int]

Window.**pixel_ratio**
The frambuffer/window size ratio

Type float

Window.**viewport**
current window viewport

Type Tuple[int, int, int, int]

Window.**frames**
Number of frames rendered

Type int

Window.**resizable**
Window is resizable

Type bool

Window.**fullscreen**
Window is in fullscreen mode

Type bool

Window.**config**
Get the current WindowConfig instance

Window.**vsync**
vertical sync enabled/disabled

Type bool

Window.**aspect_ratio**
Aspect ratio configured for the viewport

Type float

Window.**samples**
Number of Multisample anti-aliasing (MSAA) samples

Type float

Window.**cursor**
Should the mouse cursor be visible inside the window?

Type bool

Window.**render_func**
The render callable

This property can also be used to assign a callable.

Type callable

Window.**resize_func**
The resize callable

This property can also be used to assign a callable.

Type callable

Window.**key_event_func**
The key_event callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_position_event_func**
The mouse_position callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_press_event_func**
The mouse_press callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_release_event_func**
The mouse_release callable

This property can also be used to assign a callable.

Type callable

Window.**is_closing**
Check pyglet's internal exit state

Window.**modifiers**
(KeyModifiers) The current keyboard modifiers

Window.**gl_version_code**
Generates the version code integer for the selected OpenGL version.

gl_version (4, 1) returns 410

Type int

2.3.6 pyqt5Window

Methods

Window.**__init__**(***kwargs*)
Initialize a window instance.

Parameters

- **title** (*str*) – The window title

- **gl_version** (*tuple*) – Major and minor version of the opengl context to create
- **size** (*tuple*) – Window size x, y
- **resizable** (*bool*) – Should the window be resizable?
- **fullscreen** (*bool*) – Open window in fullsceeen mode
- **vsync** (*bool*) – Enable/disable vsync
- **aspect_ratio** (*float*) – The desired aspect ratio. Can be set to None.
- **samples** (*int*) – Number of MSAA samples for the default framebuffer
- **cursor** (*bool*) – Enable/disable displaying the cursor inside the window

Window.**init_mgl_context**() → None
Create or assign a ModernGL context. If no context is supplied a context will be created using the window's gl_version.

Keyword Arguments **ctx** – An optional custom ModernGL context

Window.**is_key_pressed**(*key*) → bool
Returns: The press state of a key

Window.**close**() → None
Signal for the window to close

Window.**use**()
Bind the window's framebuffer

Window.**clear**(*red=0.0*, *green=0.0*, *blue=0.0*, *alpha=0.0*, *depth=1.0*, *viewport=None*)
Binds and clears the default framebuffer

Parameters

- **red** (*float*) – color component
- **green** (*float*) – color component
- **blue** (*float*) – color component
- **alpha** (*float*) – alpha component
- **depth** (*float*) – depth value
- **viewport** (*tuple*) – The viewport

Window.**render**(*time=0.0*, *frame_time=0.0*) → None
Renders a frame by calling the configured render callback

Keyword Arguments

- **time** (*float*) – Current time in seconds
- **frame_time** (*float*) – Delta time from last frame in seconds

Window.**swap_buffers**() → None
Swap buffers, set viewport, trigger events and increment frame counter

Window.**resize**(*width: int*, *height: int*) → None
Replacement for Qt's resizeGL method.

Parameters

- **width** – New window width
- **height** – New window height

Window.**destroy**() → None
Quit the Qt application to exit the window gracefully

Window.**set_default_viewport**() → None
Calculates the viewport based on the configured aspect ratio. Will add black borders and center the viewport if the window do not match the configured viewport.

If aspect ratio is None the viewport will be scaled to the entire window size regardless of size.

Window.**print_context_info**()
Prints moderngl context info.

Window Specific Methods

Window.**close_event**(*event*) → None
The standard PyQt close events

Parameters **event** – The qtevent instance

Window.**mouse_release_event**(*event*) → None
Forward mouse release events to standard methods

Parameters **event** – The qtevent instance

Window.**key_release_event**(*event*) → None
Process Qt key release events forwarding them to standard methods

Parameters **event** – The qtevent instance

Window.**mouse_move_event**(*event*) → None
Forward mouse cursor position events to standard methods

Parameters **event** – The qtevent instance

Window.**key_pressed_event**(*event*) → None
Process Qt key press events forwarding them to standard methods

Parameters **event** – The qtevent instance

Window.**mouse_press_event**(*event*) → None
Forward mouse press events to standard methods

Parameters **event** – The qtevent instance

Attributes

Window.**keys**
PyQt5 specific key constants

Window.**ctx**
The ModernGL context for the window

Type moderngl.Context

Window.**fbo**
The default framebuffer

Type moderngl.Framebuffer

Window.**title**
Window title

Type str

Window.**gl_version**
(major, minor) required OpenGL version

Type Tuple[int, int]

Window.**width**
The current window width

Type int

Window.**height**
The current window height

Type int

Window.**size**
current window size

Type Tuple[int, int]

Window.**buffer_size**
tuple with the current window buffer size

Type Tuple[int, int]

Window.**pixel_ratio**
The frambuffer/window size ratio

Type float

Window.**viewport**
current window viewport

Type Tuple[int, int, int, int]

Window.**frames**
Number of frames rendered

Type int

Window.**resizable**
Window is resizable

Type bool

Window.**fullscreen**
Window is in fullscreen mode

Type bool

Window.**config**
Get the current WindowConfig instance

Window.**vsync**
vertical sync enabled/disabled

Type bool

Window.**aspect_ratio**
Aspect ratio configured for the viewport

Type float

Window.**samples**
Number of Multisample anti-aliasing (MSAA) samples

Type float

Window.**cursor**
Should the mouse cursor be visible inside the window?

Type bool

Window.**render_func**
The render callable

This property can also be used to assign a callable.

Type callable

Window.**resize_func**
The resize callable

This property can also be used to assign a callable.

Type callable

Window.**key_event_func**
The key_event callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_position_event_func**
The mouse_position callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_press_event_func**
The mouse_press callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_release_event_func**
The mouse_release callable

This property can also be used to assign a callable.

Type callable

Window.**is_closing**
Is the window about to close?

Type bool

Window.**modifiers**
(KeyModifiers) The current keyboard modifiers

Window.**gl_version_code**
Generates the version code integer for the selected OpenGL version.

gl_version (4, 1) returns 410

Type int

2.3.7 pyside2Window

Methods

Window.**__init__**(***kwargs*)
Initialize a window instance.

Parameters

- **title** (*str*) – The window title
- **gl_version** (*tuple*) – Major and minor version of the opengl context to create
- **size** (*tuple*) – Window size x, y
- **resizable** (*bool*) – Should the window be resizable?
- **fullscreen** (*bool*) – Open window in fullsceeen mode
- **vsync** (*bool*) – Enable/disable vsync
- **aspect_ratio** (*float*) – The desired aspect ratio. Can be set to None.
- **samples** (*int*) – Number of MSAA samples for the default framebuffer
- **cursor** (*bool*) – Enable/disable displaying the cursor inside the window

Window.**init_mgl_context**() → None
Create or assign a ModernGL context. If no context is supplied a context will be created using the window's gl_version.

Keyword Arguments **ctx** – An optional custom ModernGL context

Window.**is_key_pressed**(*key*) → bool
Returns: The press state of a key

Window.**close**() → None
Signal for the window to close

Window.**use**()
Bind the window's framebuffer

Window.**clear**(*red=0.0*, *green=0.0*, *blue=0.0*, *alpha=0.0*, *depth=1.0*, *viewport=None*)
Binds and clears the default framebuffer

Parameters

- **red** (*float*) – color component
- **green** (*float*) – color component
- **blue** (*float*) – color component
- **alpha** (*float*) – alpha component
- **depth** (*float*) – depth value
- **viewport** (*tuple*) – The viewport

Window.**render**(*time=0.0*, *frame_time=0.0*) → None
Renders a frame by calling the configured render callback

Keyword Arguments

- **time** (*float*) – Current time in seconds
- **frame_time** (*float*) – Delta time from last frame in seconds

Window.**swap_buffers**() → None
Swap buffers, set viewport, trigger events and increment frame counter

Window.**resize**(*width: int*, *height: int*) → None
Replacement for Qt's `resizeGL` method.

Parameters

- **width** – New window width
- **height** – New window height

Window.**destroy**() → None
Quit the Qt application to exit the window gracefully

Window.**set_default_viewport**() → None
Calculates the viewport based on the configured aspect ratio. Will add black borders and center the viewport if the window do not match the configured viewport.

If aspect ratio is None the viewport will be scaled to the entire window size regardless of size.

Window.**print_context_info**()
Prints moderngl context info.

Window Specific Methods

Window.**close_event**(*event*) → None
The standard PyQt close events

Parameters **event** – The qtevent instance

Window.**mouse_release_event**(*event*) → None
Forward mouse release events to standard methods

Parameters **event** – The qtevent instance

Window.**key_release_event**(*event*)
Process Qt key release events forwarding them to standard methods

Parameters **event** – The qtevent instance

Window.**mouse_move_event**(*event*) → None
Forward mouse cursor position events to standard methods

Parameters **event** – The qtevent instance

Window.**key_pressed_event**(*event*)
Process Qt key press events forwarding them to standard methods

Parameters **event** – The qtevent instance

Window.**mouse_press_event**(*event*) → None
Forward mouse press events to standard methods

Parameters **event** – The qtevent instance

Attributes

Window.**keys**
PySide2 specific key constants

Window.**ctx**
The ModernGL context for the window

Type moderngl.Context

Window.**fbo**
The default framebuffer

Type moderngl.Framebuffer

Window.**title**
Window title

Type str

Window.**gl_version**
(major, minor) required OpenGL version

Type Tuple[int, int]

Window.**width**
The current window width

Type int

Window.**height**
The current window height

Type int

Window.**size**
current window size

Type Tuple[int, int]

Window.**buffer_size**
tuple with the current window buffer size

Type Tuple[int, int]

Window.**pixel_ratio**
The frambuffer/window size ratio

Type float

Window.**viewport**
current window viewport

Type Tuple[int, int, int, int]

Window.**frames**
Number of frames rendered

Type int

Window.**resizable**
Window is resizable

Type bool

Window.**fullscreen**
Window is in fullscreen mode

Type bool

Window.**config**
Get the current WindowConfig instance

Window.**vsync**
vertical sync enabled/disabled

Type bool

Window.**aspect_ratio**
Aspect ratio configured for the viewport

Type float

Window.**samples**
Number of Multisample anti-aliasing (MSAA) samples

Type float

Window.**cursor**
Should the mouse cursor be visible inside the window?

Type bool

Window.**render_func**
The render callable

This property can also be used to assign a callable.

Type callable

Window.**resize_func**
The resize callable

This property can also be used to assign a callable.

Type callable

Window.**key_event_func**
The key_event callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_position_event_func**
The mouse_position callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_press_event_func**
The mouse_press callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_release_event_func**
The mouse_release callable

This property can also be used to assign a callable.

Type callable

Window.**is_closing**
Is the window about to close?

Type bool

Window.**modifiers**
(KeyModifiers) The current keyboard modifiers

Window.**gl_version_code**
Generates the version code integer for the selected OpenGL version.

gl_version (4, 1) returns 410

Type int

2.3.8 sdl2.Window

Methods

Window.**__init__**(***kwargs*)
Initialize a window instance.

Parameters

- **title** (*str*) – The window title
- **gl_version** (*tuple*) – Major and minor version of the opengl context to create
- **size** (*tuple*) – Window size x, y
- **resizable** (*bool*) – Should the window be resizable?
- **fullscreen** (*bool*) – Open window in fullsceeen mode
- **vsync** (*bool*) – Enable/disable vsync
- **aspect_ratio** (*float*) – The desired aspect ratio. Can be set to None.
- **samples** (*int*) – Number of MSAA samples for the default framebuffer
- **cursor** (*bool*) – Enable/disable displaying the cursor inside the window

Window.**init_mgl_context**() → None
Create or assign a ModernGL context. If no context is supplied a context will be created using the window's gl_version.

Keyword Arguments **ctx** – An optional custom ModernGL context

Window.**is_key_pressed**(*key*) → bool
Returns: The press state of a key

Window.**close**() → None
Signal for the window to close

Window.**use**()
Bind the window's framebuffer

Window.**clear**(*red=0.0*, *green=0.0*, *blue=0.0*, *alpha=0.0*, *depth=1.0*, *viewport=None*)
Binds and clears the default framebuffer

Parameters

- **red** (*float*) – color component
- **green** (*float*) – color component
- **blue** (*float*) – color component
- **alpha** (*float*) – alpha component
- **depth** (*float*) – depth value
- **viewport** (*tuple*) – The viewport

Window.**render**(*time=0.0*, *frame_time=0.0*) → None
Renders a frame by calling the configured render callback

Keyword Arguments

- **time** (*float*) – Current time in seconds
- **frame_time** (*float*) – Delta time from last frame in seconds

Window.**swap_buffers**() → None
Swap buffers, set viewport, trigger events and increment frame counter

Window.**resize**(*width*, *height*) → None
Resize callback

Parameters

- **width** – New window width
- **height** – New window height

Window.**destroy**() → None
Gracefully close the window

Window.**set_default_viewport**() → None
Calculates the viewport based on the configured aspect ratio. Will add black borders and center the viewport if the window do not match the configured viewport.

If aspect ratio is None the viewport will be scaled to the entire window size regardless of size.

Window.**print_context_info**()
Prints moderngl context info.

Window Specific Methods

Window.**process_events**() → None
Handle all queued events in sdl2 dispatching events to standard methods

Attributes

Window.**keys**
SDL2 specific key constants

Window.**ctx**
The ModernGL context for the window

Type moderngl.Context

Window.**fbo**
The default framebuffer

Type moderngl.Framebuffer

Window.**title**
Window title

Type str

Window.**gl_version**
(major, minor) required OpenGL version

Type Tuple[int, int]

Window.**width**
The current window width

Type int

Window.**height**
The current window height

Type int

Window.**size**
current window size

Type Tuple[int, int]

Window.**buffer_size**
tuple with the current window buffer size

Type Tuple[int, int]

Window.**pixel_ratio**
The frambuffer/window size ratio

Type float

Window.**viewport**
current window viewport

Type Tuple[int, int, int, int]

Window.**frames**
Number of frames rendered

Type int

Window.**resizable**
Window is resizable

Type bool

Window.**fullscreen**
Window is in fullscreen mode

Type bool

Window.**config**
Get the current WindowConfig instance

Window.**vsync**
vertical sync enabled/disabled

Type bool

Window.**aspect_ratio**
Aspect ratio configured for the viewport

Type float

Window.**samples**
Number of Multisample anti-aliasing (MSAA) samples

Type float

Window.**cursor**
Should the mouse cursor be visible inside the window?

Type bool

Window.**render_func**
The render callable

This property can also be used to assign a callable.

Type callable

Window.**resize_func**
The resize callable

This property can also be used to assign a callable.

Type callable

Window.**key_event_func**
The key_event callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_position_event_func**
The mouse_position callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_press_event_func**
The mouse_press callable

This property can also be used to assign a callable.

Type callable

Window.**mouse_release_event_func**
The mouse_release callable

This property can also be used to assign a callable.

Type callable

Window.**is_closing**
Is the window about to close?

Type bool

Window.**modifiers**
(KeyModifiers) The current keyboard modifiers

Window.**gl_version_code**
Generates the version code integer for the selected OpenGL version.

gl_version (4, 1) returns 410

Type int

2.4 moderngl_window.geometry

moderngl_window.geometry.**bbox**(*size=(1.0, 1.0, 1.0)*, *name=None*, *attr_names=<class 'moderngl_window.geometry.attributes.AttributeNames'>*)
Generates a bounding box with (0.0, 0.0, 0.0) as the center. This is simply a box with LINE_STRIP as draw mode.

Keyword Arguments

- **size** (*tuple*) – x, y, z size of the box
- **name** (*str*) – Optional name for the VAO
- **attr_names** (*AttributeNames*) – Attribute names

Returns A *moderngl_window.opengl.vao.VAO* instance

moderngl_window.geometry.**quad_fs**(*attr_names=<class 'moderngl_window.geometry.attributes.AttributeNames'>*, *normals=True*, *uvs=True*, *name=None*) → moderngl_window.opengl.vao.VAO

Creates a screen aligned quad using two triangles with normals and texture coordiantes.

Keyword Arguments

- **attr_names** (*AttributeNames*) – Attrib name config
- **normals** (*bool*) – Include normals in VAO
- **uvs** (*bool*) – Include normals in VAO
- **name** (*str*) – Optional name for the VAO

Returns A `demosys.opengl.vao.VAO` instance.

moderngl_window.geometry.**quad_2d**(*size=(1.0, 1.0)*, *pos=(0.0, 0.0)*, *normals=True*, *uvs=True*, *attr_names=<class 'moderngl_window.geometry.attributes.AttributeNames'>*, *name=None*) → moderngl_window.opengl.vao.VAO

Creates a 2D quad VAO using 2 triangles with normals and texture coordinates.

Keyword Arguments

- **size** (*tuple*) – width and height
- **pos** (*float*) – Center position x and y
- **normals** (*bool*) – Include normals in VAO
- **uvs** (*bool*) – Include normals in VAO
- **attr_names** (*AttributeNames*) – Attrib name config
- **name** (*str*) – Optional name for the VAO

Returns A `VAO` instance.

moderngl_window.geometry.**cube**(*size=(1.0, 1.0, 1.0)*, *center=(0.0, 0.0, 0.0)*, *normals=True*, *uvs=True*, *name=None*, *attr_names=<class 'moderngl_window.geometry.attributes.AttributeNames'>*) → moderngl_window.opengl.vao.VAO

Creates a cube VAO with normals and texture coordinates

Keyword Arguments

- **width** (*float*) – Width of the cube
- **height** (*float*) – Height of the cube
- **depth** (*float*) – Depth of the cube
- **center** – center of the cube as a 3-component tuple
- **normals** – (bool) Include normals
- **uvs** – (bool) include uv coordinates

- **name** (*str*) – Optional name for the VAO
- **attr_names** (*AttributeNames*) – Attribute names

Returns A *moderngl_window.opengl.vao.VAO* instance

moderngl_window.geometry.**sphere**(*radius=0.5*, *sectors=32*, *rings=16*, *normals=True*, *uvs=True*, *name: str = None*, *attr_names=<class 'moderngl_window.geometry.attributes.AttributeNames'>*) → moderngl_window.opengl.vao.VAO

Creates a sphere.

Keyword Arguments

- **radius** (*float*) – Radius or the sphere
- **rings** (*int*) – number or horizontal rings
- **sectors** (*int*) – number of vertical segments
- **normals** (*bool*) – Include normals in the VAO
- **uvs** (*bool*) – Include texture coordinates in the VAO
- **name** (*str*) – An optional name for the VAO
- **attr_names** (*AttributeNames*) – Attribute names

Returns A VAO instance

2.5 moderngl_window.loaders

2.5.1 base.BaseLoader

Method

BaseLoader.**__init__**(*meta*)

Initialize loader.

Loaders take a ResourceDescription instance containing all the parameters needed to load and initialize this data.

Parameters **meta** (*ResourceDescription*) – The resource to load

classmethod BaseLoader.**supports_file**(*meta*)

Check if the loader has a supported file extension.

What extensions are supported can be defiened in the *file_extensions* class attribute.

BaseLoader.**load**() → Any

Loads a resource.

When creating a loader this is the only method that needs to be implemented.

Returns The loaded resource

BaseLoader.**find_data**(*path*)

Find resource using data finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

BaseLoader.**find_program**(*path*)
Find resource using program finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

BaseLoader.**find_texture**(*path*)
Find resource using texture finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

BaseLoader.**find_scene**(*path*)
Find resource using scene finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Attributes

BaseLoader.**kind = None**
The kind of resource this loaded supports. This can be used when file extensions is not enough to decide what loader should be selected.

BaseLoader.**file_extensions = []**
A list defining the file extensions accepted by this loader.

Example:

```
# Loader will match .xyz and .xyz.gz files.
file_extensions = [
    ['.xyz'],
    ['.xyz', '.gz'],
]
```

BaseLoader.**ctx**
ModernGL context

Type moderngl.Context

2.5.2 texture.t2d.Loader

Method

Loader.**__init__**(*meta*)
Initialize loader.

Loaders take a ResourceDescription instance containing all the parameters needed to load and initialize this data.

Parameters **meta** (*ResourceDescription*) – The resource to load

classmethod Loader.**supports_file**(*meta*)
Check if the loader has a supported file extension.

What extensions are supported can be defiened in the *file_extensions* class attribute.

Loader.**load**()
Load a 2d texture as configured in the supplied `TextureDescription`

Returns The Texture instance

Return type moderngl.Texture

Loader.**find_data**(*path*)
Find resource using data finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_program**(*path*)
Find resource using program finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_texture**(*path*)
Find resource using texture finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_scene**(*path*)
Find resource using scene finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Attributes

Loader.**kind = '2d'**

Loader.**file_extensions = []**

Loader.**ctx**
ModernGL context

Type moderngl.Context

2.5.3 program.single.Loader

Method

Loader.**__init__**(*meta*)
Initialize loader.

Loaders take a ResourceDescription instance containing all the parameters needed to load and initialize this data.

Parameters **meta** (*ResourceDescription*) – The resource to load

classmethod Loader.**supports_file**(*meta*)
Check if the loader has a supported file extension.

What extensions are supported can be defiened in the *file_extensions* class attribute.

Loader.**load**() → moderngl.program.Program
Loads a shader program from a single glsl file.

Each shader type is separated by preprocessors

- VERTEX_SHADER
- FRAGMENT_SHADER
- GEOMETRY_SHADER
- TESS_CONTROL_SHADER
- TESS_EVALUATION_SHADER

Example:

```
#version 330

#if defined VERTEX_SHADER

in vec3 in_position;
in vec2 in_texcoord_0;
out vec2 uv0;

void main() {
    gl_Position = vec4(in_position, 1);
    uv0 = in_texcoord_0;
}

#elif defined FRAGMENT_SHADER

out vec4 fragColor;
uniform sampler2D texture0;
in vec2 uv0;

void main() {
    fragColor = texture(texture0, uv0);
}
#endif
```

Returns The Program instance

Return type moderngl.Program

Loader.**find_data**(*path*)
Find resource using data finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_program**(*path*)
Find resource using program finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_texture**(*path*)
Find resource using texture finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_scene**(*path*)
Find resource using scene finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Attributes

Loader.**kind = 'single'**

Loader.**file_extensions = []**

Loader.**ctx**
ModernGL context

Type moderngl.Context

2.5.4 program.separate.Loader

Method

Loader.**__init__**(*meta*)
Initialize loader.

Loaders take a ResourceDescription instance containing all the parameters needed to load and initialize this data.

Parameters **meta** (*ResourceDescription*) – The resource to load

classmethod Loader.**supports_file**(*meta*)
Check if the loader has a supported file extension.

What extensions are supported can be defiened in the *file_extensions* class attribute.

Loader.**load**() → moderngl.program.Program
Loads a shader program were each shader is a separate file.

This detected and dictated by the `kind` in the `ProgramDescription`.

Returns The Program instance

Return type moderngl.Program

Loader.**find_data**(*path*)
Find resource using data finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_program**(*path*)
Find resource using program finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_texture**(*path*)
Find resource using texture finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_scene**(*path*)
Find resource using scene finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader Specific Methods

Loader.**load_shader**(*shader_type: str*, *path: str*)
Load a single shader

Attributes

Loader.**kind = 'separate'**

Loader.**file_extensions = []**

Loader.**ctx**
ModernGL context

Type moderngl.Context

2.5.5 texture.array.Loader

Method

Loader.**__init__**(*meta*)
Initialize loader.

Loaders take a ResourceDescription instance containing all the parameters needed to load and initialize this data.

Parameters **meta** (*ResourceDescription*) – The resource to load

classmethod Loader.**supports_file**(*meta*)
Check if the loader has a supported file extension.

What extensions are supported can be defiened in the *file_extensions* class attribute.

Loader.**load**()
Load a texture array as described by the supplied TextureDescription`

Returns The TextureArray instance

Return type moderngl.TextureArray

Loader.**find_data**(*path*)
Find resource using data finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_program**(*path*)
Find resource using program finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_texture**(*path*)
Find resource using texture finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_scene**(*path*)
Find resource using scene finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Attributes

Loader.**kind = 'array'**

Loader.**file_extensions = []**

Loader.**ctx**
ModernGL context

Type moderngl.Context

2.5.6 scene.wavefront.Loader

Method

Loader.**__init__**(*meta: moderngl_window.meta.scene.SceneDescription*)
Initialize loader.

Loaders take a ResourceDescription instance containing all the parameters needed to load and initialize this data.

Parameters **meta** (*ResourceDescription*) – The resource to load

classmethod Loader.**supports_file**(*meta*)
Check if the loader has a supported file extension.

What extensions are supported can be defiened in the *file_extensions* class attribute.

Loader.**load**()
Loads a wavefront/obj file including materials and textures

Returns The Scene instance

Return type Scene

Loader.**find_data**(*path*)
Find resource using data finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_program**(*path*)
Find resource using program finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_texture**(*path*)
Find resource using texture finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_scene**(*path*)
Find resource using scene finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Attributes

Loader.**kind = 'wavefront'**

Loader.**file_extensions = [['.obj'], ['.obj', '.gz'], ['.bin']]**

Loader.**ctx**
ModernGL context

Type moderngl.Context

2.5.7 scene.gltf2.Loader

Method

Loader.**__init__**(*meta: moderngl_window.meta.scene.SceneDescription*)
Initialize loading GLTF 2 scene.

Supported formats:

- gltf json format with external resources
- gltf embedded buffers
- glb Binary format

classmethod Loader.**supports_file**(*meta*)
Check if the loader has a supported file extension.

What extensions are supported can be defiened in the *file_extensions* class attribute.

Loader.**load**() → moderngl_window.scene.scene.Scene
Load a GLTF 2 scene including referenced textures.

Returns The scene instance

Return type Scene

Loader.**find_data**(*path*)
Find resource using data finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_program**(*path*)
Find resource using program finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_texture**(*path*)
Find resource using texture finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_scene**(*path*)
Find resource using scene finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader Specific Methods

Loader.**load_gltf**()
Loads a gltf json file parsing its contents

Loader.**load_glb**()
Loads a binary gltf file parsing its contents

Loader.**load_materials**()
Load materials referenced in gltf metadata

Loader.**load_nodes**()
Load nodes referenced in gltf metadata

Loader.**load_node**(*meta*, *parent=None*)
Load a single node

Loader.**load_images**()
Load images referenced in gltf metadata

Loader.**load_textures**()
Load textures referenced in gltf metadata

Loader.**load_samplers**()
Load samplers referenced in gltf metadata

Loader.**load_meshes**()
Load meshes referenced in gltf metadata

Attributes

Loader.**kind = 'gltf'**

Loader.**file_extensions = [['.gltf'], ['.glb']]**

Loader.**ctx**
ModernGL context

Type moderngl.Context

Loader Specific Attributes

Loader.**supported_extensions = []**
Supported GLTF extensions https://github.com/KhronosGroup/glTF/tree/master/extensions

2.5.8 scene.stl.Loader

Method

Loader.**__init__**(*meta*)
Initialize loader.

Loaders take a ResourceDescription instance containing all the parameters needed to load and initialize this data.

Parameters **meta** (*ResourceDescription*) – The resource to load

classmethod Loader.**supports_file**(*meta*)
Check if the loader has a supported file extension.

What extensions are supported can be defiened in the *file_extensions* class attribute.

Loader.**load**() → moderngl_window.scene.scene.Scene
Loads and stl scene/file

Returns The Scene instance

Return type Scene

Loader.**find_data**(*path*)
Find resource using data finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_program**(*path*)
Find resource using program finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_texture**(*path*)
Find resource using texture finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_scene**(*path*)
Find resource using scene finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Attributes

Loader.**kind = 'stl'**

Loader.**file_extensions = [['.stl'], ['.stl', '.gz']]**

Loader.**ctx**
ModernGL context

Type moderngl.Context

2.5.9 data.json.Loader

Method

Loader.**__init__**(*meta*)
Initialize loader.

Loaders take a ResourceDescription instance containing all the parameters needed to load and initialize this data.

Parameters **meta** (*ResourceDescription*) – The resource to load

classmethod Loader.**supports_file**(*meta*)
Check if the loader has a supported file extension.

What extensions are supported can be defiened in the *file_extensions* class attribute.

Loader.**load**() → dict
Load a file as json

Returns The json contents

Return type dict

Loader.**find_data**(*path*)
Find resource using data finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_program**(*path*)
Find resource using program finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_texture**(*path*)
Find resource using texture finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_scene**(*path*)
Find resource using scene finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Attributes

Loader.**kind = 'json'**

Loader.**file_extensions = [['.json']]**

Loader.**ctx**
ModernGL context

Type moderngl.Context

2.5.10 data.text.Loader

Method

Loader.**__init__**(*meta*)
Initialize loader.

Loaders take a ResourceDescription instance containing all the parameters needed to load and initialize this data.

Parameters **meta** (*ResourceDescription*) – The resource to load

classmethod Loader.**supports_file**(*meta*)
Check if the loader has a supported file extension.

What extensions are supported can be defiened in the *file_extensions* class attribute.

Loader.**load**() → str
Load a file in text mode.

Returns The string contents of the file

Return type str

Loader.**find_data**(*path*)
Find resource using data finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_program**(*path*)
Find resource using program finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_texture**(*path*)
Find resource using texture finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_scene**(*path*)
Find resource using scene finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Attributes

Loader.**kind = 'text'**

Loader.**file_extensions = [['.txt']]**

Loader.**ctx**
ModernGL context

Type moderngl.Context

2.5.11 data.binary.Loader

Method

Loader.**__init__**(*meta*)
Initialize loader.

Loaders take a ResourceDescription instance containing all the parameters needed to load and initialize this data.

Parameters **meta** (*ResourceDescription*) – The resource to load

classmethod Loader.**supports_file**(*meta*)
Check if the loader has a supported file extension.

What extensions are supported can be defiened in the *file_extensions* class attribute.

Loader.**load**() → bytes
Load a file in binary mode

Returns The bytes contents of the file

Return type bytes

Loader.**find_data**(*path*)
Find resource using data finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_program**(*path*)
Find resource using program finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_texture**(*path*)
Find resource using texture finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Loader.**find_scene**(*path*)
Find resource using scene finders.

This is mainly a shortcut method to simplify the task.

Parameters **path** – Path to resource

Attributes

Loader.**kind = 'binary'**

Loader.**file_extensions = []**

Loader.**ctx**
ModernGL context

Type moderngl.Context

2.6 moderngl_window.meta

2.6.1 base.ResourceDescription

moderngl_window.meta.base.**ResourceDescription**
Description of any resource. Resource descriptions are required to load a resource. This class can be extended to add more specific properties.

Methods

ResourceDescription.**__init__**(***kwargs*)
Initialize a resource description

Parameters ****kwargs** – Attributes describing the resource to load

Attributes

ResourceDescription.**path**
The path to a resource when a single file is specified

Type str

ResourceDescription.**resolved_path**
The resolved path by a finder.

The absolute path to the resource can optionally be assigned by a loader class.

Type pathlib.Path

ResourceDescription.**attrs**
All keywords arguments passed to the resource

Type dict

ResourceDescription.**label**
optional name for the resource

Assigning a label is not mandatory but can help when aliasing resources. Some prefer to preload all needed resources and fetch them later by the label. This can he a lot less chaotic in larger applications.

Type str

ResourceDescription.**kind**
default resource kind.

The resource `kind` is directly matched with the `kind` in loder classes.

This property also supports assignment and is useful if the `kind` is detected based in the the attribute values.

```
description.kind = 'something'
```

Type str

ResourceDescription.**loader_cls**
The loader class for this resource.

This property is assigned to during the loading stage were a loader class is assigned based on the *kind*.

Type Type

ResourceDescription.**default_kind = None**
The default kind for this resource type

Type str

ResourceDescription.**resource_type = None**
A unique identifier for the resource type

Type str

2.6.2 texture.TextureDescription

moderngl_window.meta.texture.**TextureDescription**
Describes a texture to load.

Example:

```
# Loading a 2d texture
TextureDescription(path='textures/wood.png')

# Loading a 2d texture with mimpmaps with anisotropy
TextureDescription(path='textures/wood.png', mipmap=True, anisotropy=16.0)

# Loading texture array containing 10 layers
TextureDescription(path='textures/tiles.png', layers=10, kind='array')
```

Methods

TextureDescription.**__init__**(*path: str = None*, *kind: str = None*, *flip=True*, *mipmap=False*, *mipmap_levels: Tuple[int, int] = None*, *anisotropy=1.0*, *image=None*, *layers=None*, ***kwargs*)
Describes a texture resource

Parameters

- **path** (*str*) – path to resource relative to search directories
- **flip** (*boolean*) – Flip the image horisontally
- **mipmap** (*bool*) – Generate mipmaps. Will generate max possible levels unless *mipmap_levels* is defined.
- **mipmap_levels** (*tuple*) – (base, max_level) controlling mipmap generation. When defined the *mipmap* parameter is automatically *True*.
- **anisotropy** (*float*) – Number of samples for anisotropic filtering
- **kind** (*str*) – The kind of loader to use
- **image** – PIL image for when loading embedded resources
- **layers** – (int): Number of layers for texture arrays
- ****kwargs** – Any optional/custom attributes

Attributes

`TextureDescription.`**`mipmap`**
If mipmaps should be generated

Type bool

`TextureDescription.`**`image`**
PIL image when loading embedded resources

Type Image

`TextureDescription.`**`layers`**
Number of layers in texture array

Type int

`TextureDescription.`**`anisotropy`**
Number of samples for anisotropic filtering

Type float

`TextureDescription.`**`mipmap_levels`**
base, max_level for mipmap generation

Type Tuple[int, int]

`TextureDescription.`**`flip`**
If the image should be flipped horisontally

Type bool

Inherited Attributes

`TextureDescription.`**`path`**
The path to a resource when a single file is specified

Type str

`TextureDescription.`**`resolved_path`**
The resolved path by a finder.

The absolute path to the resource can optionally be assigned by a loader class.

Type pathlib.Path

`TextureDescription.`**`attrs`**
All keywords arguments passed to the resource

Type dict

`TextureDescription.`**`label`**
optional name for the resource

Assigning a label is not mandatory but can help when aliasing resources. Some prefer to preload all needed resources and fetch them later by the label. This can he a lot less chaotic in larger applications.

Type str

`TextureDescription.`**`kind`**
default resource kind.

The resource `kind` is directly matched with the `kind` in loder classes.

This property also supports assignment and is useful if the `kind` is detected based in the the attribute values.

```
description.kind = 'something'
```

Type str

TextureDescription.**loader_cls**
The loader class for this resource.

This property is assigned to during the loading stage were a loader class is assigned based on the *kind*.

Type Type

TextureDescription.**default_kind = '2d'**

TextureDescription.**resource_type = 'textures'**

2.6.3 program.ProgramDescription

moderngl_window.meta.program.**ProgramDescription**
Describes a program to load

By default a program can be loaded in the following ways:

- By supplying a *path* to s single glsl file containing all shaders
- By supplying several paths to separate files containing each shader trype. For example `vertex_shader`, `fragment_shader` .. etc.

```
# Single glsl file containing all shaders
ProgramDescription(path='programs/myprogram.glsl')

# Multiple shader files
ProgramDescription(
    vertex_shader='programs/myprogram_vs.glsl',
    fragment_shader='programs/myprogram_fs.glsl',
    geometry_shader='programs/myprogram_gs.glsl',
)
```

Methods

ProgramDescription.**__init__**(*path: str = None*, *kind: str = None*, *reloadable=False*, *vertex_shader: str = None*, *geometry_shader: str = None*, *fragment_shader: str = None*, *tess_control_shader: str = None*, *tess_evaluation_shader: str = None*, ***kwargs*)
Create a program description

Keyword Arguments

- **path** (*str*) – path to the resource realive to search directories
- **kind** (*str*) – The kind of loader to use
- **reloadable** (*bool*) – Should this program be reloadable
- **vertex_shader** (*str*) – Path to vertex shader file
- **geometry_shader** (*str*) – Path to geometry shader
- **fragment_shader** (*str*) – Path to fragmet shader
- **tess_control_shader** (*str*) –

- **tess_evaluation_shader** (*str*) – Path to tess eval shader
- ****kwargs** – Optional custom attributes

Attributes

ProgramDescription.**tess_evaluation_shader**
Relative path to tessellation evaluation shader

Type str

ProgramDescription.**vertex_shader**
Relative path to vertex shader

Type str

ProgramDescription.**geometry_shader**
Relative path to geometry shader

Type str

ProgramDescription.**reloadable**
if this program is reloadable

Type bool

ProgramDescription.**fragment_shader**
Relative path to fragment shader

Type str

ProgramDescription.**tess_control_shader**
Relative path to tess control shader

Type str

Inherited Attributes

ProgramDescription.**path**
The path to a resource when a single file is specified

Type str

ProgramDescription.**resolved_path**
The resolved path by a finder.

The absolute path to the resource can optionally be assigned by a loader class.

Type pathlib.Path

ProgramDescription.**attrs**
All keywords arguments passed to the resource

Type dict

ProgramDescription.**label**
optional name for the resource

Assigning a label is not mandatory but can help when aliasing resources. Some prefer to preload all needed resources and fetch them later by the label. This can he a lot less chaotic in larger applications.

Type str

ProgramDescription.**kind**
default resource kind.

The resource `kind` is directly matched with the `kind` in loder classes.

This property also supports assignment and is useful if the `kind` is detected based in the the attribute values.

```
description.kind = 'something'
```

Type str

ProgramDescription.**loader_cls**
The loader class for this resource.

This property is assigned to during the loading stage were a loader class is assigned based on the *kind*.

Type Type

ProgramDescription.**default_kind = None**

ProgramDescription.**resource_type = 'programs'**

2.6.4 scene.SceneDescription

moderngl_window.meta.scene.**SceneDescription**
Describes a scene to load.

The correct loader is resolved by looking at the file extension. This can be overriden by specifying a `kind` that maps directly to a specific loader class.

```
# Wavefront/obj file
SceneDescription(path='scenes/cube.obj')

# stl file
SceneDescription(path='scenes/crater.stl')

# GLTF 2 file
SceneDescription(path='scenes/sponza.gltf')
```

The user can also override what buffer/attribute names should be used by specifying `attr_names`.

A `cache` option is also available as some scene loaders supports converting the file into a different format on the fly to speed up loading.

Methods

SceneDescription.**__init__**(*path=None*, *kind=None*, *cache=False*, *attr_names=<class 'moderngl_window.geometry.attributes.AttributeNames'>*, ***kwargs*)
Create a scene description.

Keyword Arguments

- **path** (*str*) – Path to resource
- **kind** (*str*) – Loader kind
- **cache** (*str*) – Use the loader caching system if present
- **attr_names** (*AttributeNames*) – Attrib name config

- ****kwargs** – Optional custom attributes

Attributes

SceneDescription.**attr_names**
Attribute name config

Type AttributeNames

SceneDescription.**cache**
Use cache feature in scene loader

Type bool

Inherited Attributes

SceneDescription.**path**
The path to a resource when a single file is specified

Type str

SceneDescription.**resolved_path**
The resolved path by a finder.

The absolute path to the resource can optionally be assigned by a loader class.

Type pathlib.Path

SceneDescription.**attrs**
All keywords arguments passed to the resource

Type dict

SceneDescription.**label**
optional name for the resource

Assigning a label is not mandatory but can help when aliasing resources. Some prefer to preload all needed resources and fetch them later by the label. This can he a lot less chaotic in larger applications.

Type str

SceneDescription.**kind**
default resource kind.

The resource `kind` is directly matched with the `kind` in loder classes.

This property also supports assignment and is useful if the `kind` is detected based in the the attribute values.

```
description.kind = 'something'
```

Type str

SceneDescription.**loader_cls**
The loader class for this resource.

This property is assigned to during the loading stage were a loader class is assigned based on the *kind*.

Type Type

SceneDescription.**default_kind = None**

SceneDescription.**resource_type = 'scenes'**

2.6.5 data.DataDescription

moderngl_window.meta.data.**DataDescription**
Describes data file to load.

This is a generic resource description type for loading resources that are not textures, programs and scenes. That loaded class is used depends on the `kind` or the file extension.

Currently used to load:

- text files
- json files
- binary files

```
# Describe a text file. Text loader is used based on file extension
DataDescription(path='data/text.txt')

# Describe a json file. Json loader is used based on file extension
DataDescription(path='data/data.json')

# Describe a binary file. Specify a binary loader should be used.
DataDescription(path='data/data.bin', kind='binary')
```

Methods

DataDescription.**__init__**(*path=None*, *kind=None*, ***kwargs*)
Initialize the resource description.

Keyword Arguments

- **path** (*str*) – Relative path to the resource
- **kind** (*str*) – The resource kind deciding loader class
- ****kwargs** – Additional custom attributes

Attributes

DataDescription.**path**
The path to a resource when a single file is specified

Type str

DataDescription.**resolved_path**
The resolved path by a finder.

The absolute path to the resource can optionally be assigned by a loader class.

Type pathlib.Path

DataDescription.**attrs**
All keywords arguments passed to the resource

Type dict

DataDescription.**label**
optional name for the resource

Assigning a label is not mandatory but can help when aliasing resources. Some prefer to preload all needed resources and fetch them later by the label. This can he a lot less chaotic in larger applications.

Type str

DataDescription.**kind**
default resource kind.

The resource `kind` is directly matched with the `kind` in loder classes.

This property also supports assignment and is useful if the `kind` is detected based in the the attribute values.

```
description.kind = 'something'
```

Type str

DataDescription.**loader_cls**
The loader class for this resource.

This property is assigned to during the loading stage were a loader class is assigned based on the *kind*.

Type Type

DataDescription.**default_kind = None**

DataDescription.**resource_type = 'data'**

2.7 moderngl_window.finders

2.7.1 base.BaseFilesystemFinder

moderngl_window.finders.base.**BaseFilesystemFinder**
Base class for searching filesystem directories

Methods

BaseFilesystemFinder.**__init__**()
Initialize finder class by looking up the paths referenced in `settings_attr`.

BaseFilesystemFinder.**find**(*path: pathlib.Path*) → pathlib.Path
Finds a file in the configured paths returning its absolute path.

Parameters **path** (*pathlib.Path*) – The path to find

Returns The absolute path to the file or None if not found

Attributes

BaseFilesystemFinder.**settings_attr = None**
Name of the attribute in *Settings* containing a list of paths the finder should seach in.

Type str

2.7.2 texture.FilesystemFinder

moderngl_window.finders.texture.**FilesystemFinder**
Find textures in settings.TEXTURE_DIRS

Methods

FilesystemFinder.**__init__**()
Initialize finder class by looking up the paths referenced in settings_attr.

FilesystemFinder.**find**(*path: pathlib.Path*) → pathlib.Path
Finds a file in the configured paths returning its absolute path.

Parameters **path** (*pathlib.Path*) – The path to find

Returns The absolute path to the file or None if not found

Attributes

FilesystemFinder.**settings_attr = 'TEXTURE_DIRS'**

2.7.3 program.FilesystemFinder

moderngl_window.finders.program.**FilesystemFinder**
Find shaders in settings.PROGRAM_DIRS

Methods

FilesystemFinder.**__init__**()
Initialize finder class by looking up the paths referenced in settings_attr.

FilesystemFinder.**find**(*path: pathlib.Path*) → pathlib.Path
Finds a file in the configured paths returning its absolute path.

Parameters **path** (*pathlib.Path*) – The path to find

Returns The absolute path to the file or None if not found

Attributes

FilesystemFinder.**settings_attr = 'PROGRAM_DIRS'**

2.7.4 scene.FilesystemFinder

moderngl_window.finders.scene.**FilesystemFinder**
Find scenes in settings.SCENE_DIRS

Methods

`FilesystemFinder.__init__()`
Initialize finder class by looking up the paths referenced in `settings_attr`.

`FilesystemFinder.find(path: pathlib.Path)` → pathlib.Path
Finds a file in the configured paths returning its absolute path.

Parameters **path** (*pathlib.Path*) – The path to find

Returns The absolute path to the file or None if not found

Attributes

`FilesystemFinder.settings_attr = 'SCENE_DIRS'`

2.7.5 data.FilesystemFinder

`moderngl_window.finders.data.FilesystemFinder`
Find data in `settings.DATA_DIRS`

Methods

`FilesystemFinder.__init__()`
Initialize finder class by looking up the paths referenced in `settings_attr`.

`FilesystemFinder.find(path: pathlib.Path)` → pathlib.Path
Finds a file in the configured paths returning its absolute path.

Parameters **path** (*pathlib.Path*) – The path to find

Returns The absolute path to the file or None if not found

Attributes

`FilesystemFinder.settings_attr = 'DATA_DIRS'`

2.8 moderngl_window.opengl

2.8.1 opengl.projection.Projection3D

`moderngl_window.opengl.projection.Projection3D`
3D Projection

Methods

`Projection3D.__init__(aspect_ratio=1.7777777777777777, fov=75.0, near=1.0, far=100.0)`
Create a 3D projection

Keyword Arguments

- **aspect_ratio** (*float*) – Sspect ratio

- **fov** (*float*) – Field of view
- **near** (*float*) – Near plane value
- **far** (*float*) – Far plane value

Projection3D.**update**(*aspect_ratio: float = None*, *fov: float = None*, *near: float = None*, *far: float = None*) → None
Update the projection matrix

Keyword Arguments

- **aspect_ratio** (*float*) – Sspect ratio
- **fov** (*float*) – Field of view
- **near** (*float*) – Near plane value
- **far** (*float*) – Far plane value

Projection3D.**tobytes**() → bytes
Get the byte representation of the projection matrix

Returns byte representation of the projection matrix

Return type bytes

Attributes

Projection3D.**aspect_ratio**
The projection's aspect ratio

Type float

Projection3D.**fov**
Current field of view

Type float

Projection3D.**near**
Current near plane value

Type float

Projection3D.**far**
Current far plane value

Type float

Projection3D.**matrix**
Current numpy projection matrix

Type np.ndarray

Projection3D.**projection_constants**
(x, y) projection constants for the current projection. This is for example useful when reconstructing a view position of a fragment from a linearized depth value.

2.8.2 opengl.vao.VAO

moderngl_window.opengl.vao.**VAO**
Represents a vertex array object.

This is a wrapper class over `moderngl.VertexArray` to make interactions with programs/shaders simpler. Named buffers are added correspoding with attribute names in a vertex shader. When rendering the VAO an internal `moderngl.VertextArray` is created by automatically creating a buffer mapping compatible with the supplied program. This program is cached internally.

The shader program doesn't need to use all the buffers registered in this wrapper. When a subset is used only the used buffers are mapped and the approproate padding is calculated when iterleaved data is used.

There is no requirements to use this class, but most methods in the system creating vertexbuffers will return this type. You can obtain a single `moderngl.VertexBuffer` instance by calling *VAO.instance()* method if you prefer to work directly on moderngl instances.

Example:

```
# Separate buffers
vao = VAO(name="test", mode=moderngl.POINTS)
vao.buffer(positions, '3f', ['in_position'])
vao.buffer(velocities, '3f', ['in_velocities'])

# Interleaved
vao = VAO(name="test", mode=moderngl.POINTS)
vao.buffer(interleaved_data, '3f 3f', ['in_position', 'in_velocities'])
```

```
# GLSL vertex shader in attributes
in vec3 in_position;
in vec3 in_velocities;
```

Methods

VAO.**__init__**(*name=''*, *mode=4*)

Create and empty VAO with a name and default render mode.

Example:

```
VAO(name="cube", mode=moderngl.TRIANGLES)
```

Keyword Arguments

- **name** (*str*) – Optional name for debug purposes
- **mode** (*int*) – Default draw mode

VAO.**render**(*program: moderngl.program.Program*, *mode=None*, *vertices=-1*, *first=0*, *instances=1*)

Render the VAO.

An internal `moderngl.VertexBuffer` with compatible buffer bindings is automatically created on the fly and cached internally.

Parameters **program** – The `moderngl.Program`

Keyword Arguments

- **mode** – Override the draw mode (`TRIANGLES` etc)
- **vertices** (*int*) – The number of vertices to transform
- **first** (*int*) – The index of the first vertex to start with
- **instances** (*int*) – The number of instances

VAO.**render_indirect**(*program: moderngl.program.Program*, *buffer*, *mode=None*, *count=-1*, *, *first=0*)
The render primitive (mode) must be the same as the input primitive of the GeometryShader. The draw commands are 5 integers: (count, instanceCount, firstIndex, baseVertex, baseInstance).

Parameters

- **program** – The `moderngl.Program`
- **buffer** – The `moderngl.Buffer` containing indirect draw commands

Keyword Arguments

- **mode** (*int*) – By default `TRIANGLES` will be used.
- **count** (*int*) – The number of draws.
- **first** (*int*) – The index of the first indirect draw command.

VAO.**transform**(*program: moderngl.program.Program*, *buffer: moderngl.buffer.Buffer*, *mode=None*, *vertices=-1*, *first=0*, *instances=1*)
Transform vertices. Stores the output in a single buffer.

Parameters

- **program** – The `moderngl.Program`
- **buffer** – The `moderngl.buffer` to store the output

Keyword Arguments

- **mode** – Draw mode (for example `moderngl.POINTS`)
- **vertices** (*int*) – The number of vertices to transform
- **first** (*int*) – The index of the first vertex to start with
- **instances** (*int*) – The number of instances

VAO.**buffer**(*buffer, buffer_format: str, attribute_names: List[str]*)
Register a buffer/vbo for the VAO. This can be called multiple times. adding multiple buffers (interleaved or not).

Parameters

- **buffer** – The buffer data. Can be `numpy.array`, `moderngl.Buffer` or `bytes`.
- **buffer_format** (*str*) – The format of the buffer. (eg. `3f 3f` for interleaved positions and normals).
- **attribute_names** – A list of attribute names this buffer should map to.

Returns The `moderngl.Buffer` instance object. This is handy when providing `bytes` and `numpy.array`.

VAO.**index_buffer**(*buffer*, *index_element_size=4*)
Set the index buffer for this VAO.

Parameters **buffer** – `moderngl.Buffer`, `numpy.array` or `bytes`

Keyword Arguments **index_element_size** (*int*) – Byte size of each element. 1, 2 or 4

VAO.**instance**(*program: moderngl.program.Program*) → moderngl.vertex_array.VertexArray
Obtain the `moderngl.VertexArray` instance for the program.

The instance is only created once and cached internally.

Parameters **program** (*moderngl.Program*) – The program

Returns instance

Return type moderngl.VertexArray

VAO.**release**(*buffer=True*)
Destroy all iternally cached vaos and release all buffers.

Keyword Arguments **buffers** (*bool*) – also release buffers

VAO.**get_buffer_by_name**(*name: str*) → moderngl_window.opengl.vao.BufferInfo
Get the BufferInfo associated with a specific attribute name

If no buffer is associated with the name *None* will be returned.

Parameters **name** (*str*) – Name of the mapped attribute

Returns BufferInfo instance

Return type BufferInfo

Attributes

VAO.**ctx**
The actite moderngl context

Type moderngl.Context

2.9 moderngl_window.resources

2.9.1 base.BaseRegistry

moderngl_window.resources.base.**BaseRegistry**
Base class for all resource pools

Methods

BaseRegistry.**__init__**()
Initialize internal attributes

BaseRegistry.**load**(*meta: moderngl_window.meta.base.ResourceDescription*) → Any
Loads a resource using the configured finders and loaders.

Parameters **meta** (*ResourceDescription*) – The resource description

BaseRegistry.**add**(*meta: moderngl_window.meta.base.ResourceDescription*) → None
Adds a resource description without loading it. The resource is loaded and returned when load_pool() is called.

Parameters **meta** (*ResourceDescription*) – The resource description

BaseRegistry.**load_pool**() → Generator[Tuple[moderngl_window.meta.base.ResourceDescription, Any], None, None]
Loads all the data files using the configured finders.

This is only relevant when resource have been added to this pool using add().

Returns Generator of (meta, resoure) tuples

BaseRegistry.**resolve_loader**(*meta: moderngl_window.meta.base.ResourceDescription*) → None
Attempts to assign a loader class to a ResourceDecription.

Parameters **meta** (*ResourceDescription*) – The resource description instance

Attributes

`BaseRegistry.`**`settings_attr = None`**
The name of the attribute in *Settings* containting a list of loader classes.

Type str

`BaseRegistry.`**`count`**
The number of resource descriptions added. This is only relevant when using *add* and *load_pool*.

Type int

`BaseRegistry.`**`loaders`**
Loader classes for this resource type

Type Generator

2.9.2 textures.Textures

`moderngl_window.resources.textures.`**`Textures`**
Handles texture resources

Methods

`Textures.`**`__init__`**`()`
Initialize internal attributes

`Textures.`**`load`**`(`*meta: moderngl_window.meta.texture.TextureDescription*`)` → Union[moderngl.texture.Texture, moderngl.texture_array.TextureArray]
Loads a texture with the configured loaders.

Parameters **meta** (*TextureDescription*) – The resource description

Returns 2d texture

Return type moderngl.Texture

Returns texture array if `layers` is supplied

Return type moderngl.TextureArray

`Textures.`**`add`**`(`*meta: moderngl_window.meta.base.ResourceDescription*`)` → None
Adds a resource description without loading it. The resource is loaded and returned when `load_pool()` is called.

Parameters **meta** (*ResourceDescription*) – The resource description

`Textures.`**`load_pool`**`()` → Generator[Tuple[moderngl_window.meta.base.ResourceDescription, Any], None, None]
Loads all the data files using the configured finders.

This is only relevant when resource have been added to this pool using `add()`.

Returns Generator of (meta, resoure) tuples

`Textures.`**`resolve_loader`**`(`*meta: moderngl_window.meta.base.ResourceDescription*`)` → None
Attempts to assign a loader class to a ResourceDecription.

Parameters **meta** (*ResourceDescription*) – The resource description instance

Attributes

Textures.**settings_attr = 'TEXTURE_LOADERS'**

Textures.**count**
The number of resource descriptions added. This is only relevant when using *add* and *load_pool*.

Type int

Textures.**loaders**
Loader classes for this resource type

Type Generator

2.9.3 programs.Programs

moderngl_window.resources.programs.**Programs**
Handle program loading

Methods

Programs.**__init__**()
Initialize internal attributes

Programs.**load**(*meta: moderngl_window.meta.program.ProgramDescription*) → moderngl.program.Program
Loads a shader program with the configured loaders

Parameters **meta** (*ProgramDescription*) – The resource description

Returns The shader program

Return type moderngl.Program

Programs.**add**(*meta: moderngl_window.meta.base.ResourceDescription*) → None
Adds a resource description without loading it. The resource is loaded and returned when `load_pool()` is called.

Parameters **meta** (*ResourceDescription*) – The resource description

Programs.**load_pool**() → Generator[Tuple[moderngl_window.meta.base.ResourceDescription, Any], None, None]
Loads all the data files using the configured finders.

This is only relevant when resource have been added to this pool using `add()`.

Returns Generator of (meta, resoure) tuples

Programs.**resolve_loader**(*meta: moderngl_window.meta.program.ProgramDescription*) → None
Resolve program loader.

Determines if the references resource is a single or multiple glsl files unless `kind` is specified.

Parameters **meta** (*ProgramDescription*) – The resource description

Attributes

Programs.**settings_attr = 'PROGRAM_LOADERS'**

Programs.**count**
The number of resource descriptions added. This is only relevant when using *add* and *load_pool*.

Type int

`Programs.`**`loaders`**
Loader classes for this resource type

Type Generator

2.9.4 scenes.Scenes

`moderngl_window.resources.scenes.`**`Scenes`**
Handles scene loading

Methods

`Scenes.`**`__init__`**`()`
Initialize internal attributes

`Scenes.`**`load`**`(`*meta: moderngl_window.meta.scene.SceneDescription*`)` → moderngl_window.scene.scene.Scene
Load a scene with the configured loaders.

Parameters **meta** (*SceneDescription*) – The resource description

Returns The loaded scene

Return type `Scene`

`Scenes.`**`add`**`(`*meta: moderngl_window.meta.base.ResourceDescription*`)` → None
Adds a resource description without loading it. The resource is loaded and returned when `load_pool()` is called.

Parameters **meta** (*ResourceDescription*) – The resource description

`Scenes.`**`load_pool`**`()` → Generator[Tuple[moderngl_window.meta.base.ResourceDescription, Any], None, None]
Loads all the data files using the configured finders.

This is only relevant when resource have been added to this pool using `add()`.

Returns Generator of (meta, resoure) tuples

`Scenes.`**`resolve_loader`**`(`*meta: moderngl_window.meta.base.ResourceDescription*`)` → None
Attempts to assign a loader class to a ResourceDecription.

Parameters **meta** (*ResourceDescription*) – The resource description instance

Attributes

`Scenes.`**`settings_attr = 'SCENE_LOADERS'`**

`Scenes.`**`count`**
The number of resource descriptions added. This is only relevant when using *add* and *load_pool*.

Type int

`Scenes.`**`loaders`**
Loader classes for this resource type

Type Generator

2.9.5 base.DataFiles

moderngl_window.resources.data.**DataFiles**
Registry for requested data files

Methods

DataFiles.**__init__**()
Initialize internal attributes

DataFiles.**load**(*meta: moderngl_window.meta.data.DataDescription*) → Any
Load data file with the configured loaders.

Parameters **meta** (*DataDescription*) – the resource description

Returns The loaded resource

Return type Any

DataFiles.**add**(*meta: moderngl_window.meta.base.ResourceDescription*) → None
Adds a resource description without loading it. The resource is loaded and returned when `load_pool()` is called.

Parameters **meta** (*ResourceDescription*) – The resource description

DataFiles.**load_pool**() → Generator[Tuple[moderngl_window.meta.base.ResourceDescription, Any], None, None]
Loads all the data files using the configured finders.

This is only relevant when resource have been added to this pool using `add()`.

Returns Generator of (meta, resoure) tuples

DataFiles.**resolve_loader**(*meta: moderngl_window.meta.base.ResourceDescription*) → None
Attempts to assign a loader class to a ResourceDecription.

Parameters **meta** (*ResourceDescription*) – The resource description instance

Attributes

DataFiles.**settings_attr = 'DATA_LOADERS'**

DataFiles.**count**
The number of resource descriptions added. This is only relevant when using *add* and *load_pool*.

Type int

DataFiles.**loaders**
Loader classes for this resource type

Type Generator

2.10 moderngl_window.timers

2.10.1 base.BaseTimer

moderngl_window.timers.base.**BaseTimer**
A timer controls the time passed into the the render function. This can be used in creative ways to control the

current time such as basing it on current location in an audio file.

All methods must be implemented.

Methods

BaseTimer.**__init__**()
Initialize self. See help(type(self)) for accurate signature.

BaseTimer.**next_frame**() → Tuple[float, float]
Get timer information for the next frame.

Returns The frametime and current time

Return type Tuple[float, float]

BaseTimer.**start**()
Start the timer initially or resume after pause

BaseTimer.**pause**()
Pause the timer

BaseTimer.**toggle_pause**()
Toggle pause state

BaseTimer.**stop**() → Tuple[float, float]
Stop the timer. Should only be called once when stopping the timer.

Returns Tuple[float, float]> Current position in the timer, actual running duration

Attributes

BaseTimer.**is_paused**
The pause state of the timer

Type bool

BaseTimer.**is_running**
Is the timer currently running?

Type bool

BaseTimer.**time**
Get the current time in seconds

The current time can also be assigned to this attribute.

Returns The current time in seconds

Return type float

2.10.2 clock.Timer

moderngl_window.timers.clock.**Timer**
Timer based on python `time`.

Methods

Timer.**__init__**(***kwargs*)
Initialize self. See help(type(self)) for accurate signature.

Timer.**next_frame**() → Tuple[float, float]
Get the time and frametime for the next frame. This should only be called once per frame.

Returns current time and frametime

Return type Tuple[float, float]

Timer.**start**()
Start the timer by recoding the current `time.time()` preparing to report the number of seconds since this timestamp.

Timer.**pause**()
Pause the timer by setting the internal pause time using `time.time()`

Timer.**toggle_pause**()
Toggle the paused state

Timer.**stop**() → Tuple[float, float]
Stop the timer. Should only be called once when stopping the timer.

Returns Current position in the timer, actual running duration

Return type Tuple[float, float]

Attributes

Timer.**is_paused**
The pause state of the timer

Type bool

Timer.**is_running**
Is the timer currently running?

Type bool

Timer.**time**
Get the current time in seconds

Returns The current time in seconds

2.11 moderngl_window.scene

2.11.1 Camera

moderngl_window.scene.**Camera**
Simple camera class containing projection

Methods

Camera.**__init__**(*fov=60.0*, *aspect_ratio=1.0*, *near=1.0*, *far=100.0*)
Initialize camera using a specific projection

Keyword Arguments

- **fov** (*float*) – Field of view
- **aspect_ratio** (*float*) – Aspect ratio
- **near** (*float*) – Near plane
- **far** (*float*) – Far plane

`Camera.`**`set_position`**(*x*, *y*, *z*) → None
Set the 3D position of the camera.

Parameters

- **x** (*float*) – x position
- **y** (*float*) – y position
- **z** (*float*) – z position

`Camera.`**`look_at`**(*vec=None*, *pos=None*) → numpy.ndarray
Look at a specific point

Either `vec` or `pos` needs to be supplied.

Keyword Arguments

- **vec** (*pyrr.Vector3*) – position
- **pos** (*tuple/list*) – list of tuple `[x, y, x]` / `(x, y, x)`

Returns Camera matrix

Return type numpy.ndarray

Attributes

`Camera.`**`matrix`**
The current view matrix for the camera

Type numpy.ndarray

2.11.2 KeyboardCamera

`moderngl_window.scene.`**`KeyboardCamera`**
Camera controlled by mouse and keyboard

Methods

`KeyboardCamera.`**`__init__`**(*keys: moderngl_window.context.base.keys.BaseKeys*, *fov=60.0*, *aspect_ratio=1.0*, *near=1.0*, *far=100.0*)
Initialize the camera

Parameters **keys** (*BaseKeys*) – The key constants for the current window type

Keyword Arguments

- **fov** (*float*) – Field of view
- **aspect_ratio** (*float*) – Aspect ratio
- **near** (*float*) – near plane

- **far** (*float*) – far plane

KeyboardCamera.**key_input**(*key*, *action*, *modifiers*) → None
Process key inputs and move camera

Parameters

- **key** – The key
- **action** – key action release/press
- **modifiers** – key modifier states such as ctrl or shit

KeyboardCamera.**move_left**(*activate*) → None
The camera should be continiously moving to the left.

Parameters **activate** (*bool*) – Activate or deactivate this state

KeyboardCamera.**move_right**(*activate*) → None
The camera should be continiously moving to the right.

Parameters **activate** (*bool*) – Activate or deactivate this state

KeyboardCamera.**move_forward**(*activate*) → None
The camera should be continiously moving forward.

Parameters **activate** (*bool*) – Activate or deactivate this state

KeyboardCamera.**move_backward**(*activate*) → None
The camera should be continiously moving backwards.

Parameters **activate** (*bool*) – Activate or deactivate this state

KeyboardCamera.**move_up**(*activate*) → None
The camera should be continiously moving up.

Parameters **activate** (*bool*) – Activate or deactivate this state

KeyboardCamera.**move_down**(*activate*)
The camera should be continiously moving down.

Parameters **activate** (*bool*) – Activate or deactivate this state

KeyboardCamera.**move_state**(*direction*, *activate*) → None
Set the camera position move state.

Parameters

- **direction** – What direction to update
- **activate** – Start or stop moving in the direction

KeyboardCamera.**rot_state**(*x*, *y*) → None
Set the rotation state of the camera. This value is normally the current mouse position.

Parameters

- **x** – viewport x pos
- **y** – viewport y pos

Inherited Methods

KeyboardCamera.**set_position**(*x*, *y*, *z*) → None
Set the 3D position of the camera.

Parameters

- **x** (*float*) – x position
- **y** (*float*) – y position
- **z** (*float*) – z position

KeyboardCamera.**look_at**(*vec=None*, *pos=None*) → numpy.ndarray
Look at a specific point

Either `vec` or `pos` needs to be supplied.

Keyword Arguments

- **vec** (*pyrr.Vector3*) – position
- **pos** (*tuple/list*) – list of tuple `[x, y, x]` / `(x, y, x)`

Returns Camera matrix

Return type numpy.ndarray

Attributes

KeyboardCamera.**matrix**
The current view matrix for the camera

Type numpy.ndarray

2.11.3 Scene

Methods

Scene.**__init__**(*name*, ***kwargs*)
Create a scene with a name.

Parameters **name** (*str*) – Unique name or path for the scene

Scene.**draw**(*projection_matrix: numpy.ndarray = None*, *camera_matrix: numpy.ndarray = None*, *time=0.0*) → None
Draw all the nodes in the scene.

Parameters

- **projection_matrix** (*ndarray*) – projection matrix (bytes)
- **camera_matrix** (*ndarray*) – camera_matrix (bytes)
- **time** (*float*) – The current time

Scene.**draw_bbox**(*projection_matrix=None*, *camera_matrix=None*, *children=True*) → None
Draw scene and mesh bounding boxes.

Parameters

- **projection_matrix** (*ndarray*) – mat4 projection
- **camera_matrix** (*ndarray*) – mat4 camera matrix
- **children** (*bool*) – Will draw bounding boxes for meshes as well

Scene.**apply_mesh_programs**(*mesh_programs=None*) → None
Applies mesh programs to meshes. If not mesh programs are passed in we assign default ones.

Parameters **mesh_programs** (*list*) – List of mesh programs to assign

Scene.**calc_scene_bbox**() → None
Calculate scene bbox

Scene.**prepare**() → None
prepare the scene for rendering.

Calls apply_mesh_programs() assining default meshprograms if needed and sets the model matrix.

Scene.**destroy**() → None
Destroys the scene data and vertex buffers

Attributes

Scene.**ctx**
The current context

Type moderngl.Context

Scene.**model_matrix**
The current model matrix

This property is settable.

Type numpy.ndarray

2.11.4 Node

moderngl_window.scene.**Node**
A generic scene node containing a mesh or camera and/or a container for other nodes. Nodes and their children represents the scene tree.

Methods

Node.**__init__**(*camera=None*, *mesh=None*, *matrix=None*)
Create a node.

Keyword Arguments

- **camera** – Camera to store in the node
- **mesh** – Mesh to store in the node
- **matrix** – The node's matrix

Node.**add_child**(*node*)
Add a child to this node

Parameters **node** (*Node*) – Node to add as a child

Node.**draw**(*projection_matrix=None*, *camera_matrix=None*, *time=0*)
Draw node and children.

Keyword Arguments

- **projection_matrix** (*bytes*) – projection matrix
- **camera_matrix** (*bytes*) – camera_matrix
- **time** (*float*) – The current time

Node.**draw_bbox**(*projection_matrix*, *camera_matrix*, *program*, *vao*)
Draw bounding box around the node and children.

Keyword Arguments

- **projection_matrix** (*bytes*) – projection matrix
- **camera_matrix** (*bytes*) – camera_matrix
- **program** (*moderngl.Program*) – The program to render the bbox
- **vao** – The vertex array representing the bounding box

Node.**calc_global_bbox**(*view_matrix*, *bbox_min*, *bbox_max*)
Recursive calculation of scene bbox.

Keyword Arguments

- **view_matrix** (*numpy.ndarray*) – view matrix
- **bbox_min** – min bbox values
- **bbox_max** – max bbox values

Node.**calc_model_mat**(*model_matrix*)
Calculate the model matrix related to all parents.

Parameters **model_matrix** (*numpy.ndarray*) – model matrix

Attributes

Node.**children**
List of children

Type list

2.11.5 Mesh

moderngl_window.scene.**Mesh = <class 'moderngl_window.scene.mesh.Mesh'>**
Mesh info and geometry

Methods

Mesh.**__init__**(*name*, *vao=None*, *material=None*, *attributes=None*, *bbox_min=None*, *bbox_max=None*)
Initialize mesh.

Parameters **name** (*str*) – name of the mesh

Keyword Arguments

- **vao** (VAO) – geometry
- **material** (*Msterial*) – material for the mesh
- **attributes** (*dict*) – Details info about each mesh attribute (dict)
- **bbox_min** – xyz min values
- **bbox_max** – xyz max values

Attributes example:

```
{
    "NORMAL": {"name": "in_normal", "components": 3, "type": GL_FLOAT},
    "POSITION": {"name": "in_position", "components": 3, "type": GL_FLOAT}
}
```

Mesh.**draw**(*projection_matrix=None*, *model_matrix=None*, *camera_matrix=None*, *time=0.0*)
Draw the mesh using the assigned mesh program

Keyword Arguments

- **projection_matrix** (*bytes*) – projection_matrix
- **view_matrix** (*bytes*) – view_matrix
- **camera_matrix** (*bytes*) – camera_matrix

Mesh.**draw_bbox**(*proj_matrix*, *model_matrix*, *cam_matrix*, *program*, *vao*)
Renders the bounding box for this mesh.

Parameters

- **proj_matrix** – Projection matrix
- **model_matrix** – View/model matrix
- **cam_matrix** – Camera matrix
- **program** – The moderngl.Program rendering the bounding box
- **vao** – The vao mesh for the bounding box

Mesh.**add_attribute**(*attr_type*, *name*, *components*)
Add metadata about the mesh :param attr_type: POSITION, NORMAL etc :param name: The attribute name used in the program :param components: Number of floats

Mesh.**calc_global_bbox**(*view_matrix*, *bbox_min*, *bbox_max*)
Calculates the global bounding.

Parameters

- **view_matrix** – View matrix
- **bbox_min** – xyz min
- **bbox_max** – xyz max

Returns Combined bbox

Return type bbox_min, bbox_max

Mesh.**has_normals**() → bool

Returns Does the mesh have a normals?

Return type bool

Mesh.**has_uvs**(*layer=0*) → bool

Returns Does the mesh have texture coordinates?

Return type bool

2.11.6 Material

moderngl_window.scene.**Material**
Generic material

Methods

Material.**__init__**(*name*)
Initialize material.

Parameters **name** (*str*) – Name of the material

Attributes

Material.**name**
Name of the material

Type str

Material.**color**
RGBA color

Type Tuple[float, float, float, float]

Material.**mat_texture**
instance

Type MaterialTexture

Material.**double_sided**
Material surface is double sided?

Type bool

2.11.7 MaterialTexture

moderngl_window.scene.**MaterialTexture**
Wrapper for textures used in materials. Contains a texture and a sampler object.

Methods

MaterialTexture.**__init__**(*texture: moderngl.texture.Texture = None*, *sampler: moderngl.sampler.Sampler = None*)
Initialize instance.

Parameters

- **texture** (*moderngl.Texture*) – Texture instance
- **sampler** (*moderngl.Sampler*) – Sampler instance

Attributes

`MaterialTexture.`**`texture`**
Texture instance

Type moderngl.Texture

`MaterialTexture.`**`sampler`**
Sampler instance

Type moderngl.Sampler

2.11.8 MeshProgram

`moderngl_window.scene.`**`MeshProgram`**
Describes how a mesh is rendered using a specific shader program

Methods

`MeshProgram.`**`draw`**(*mesh*, *projection_matrix: numpy.ndarray = None*, *model_matrix: numpy.ndarray = None*, *camera_matrix: numpy.ndarray = None*, *time=0.0*)
Draw code for the mesh

Parameters **`mesh`** (*`Mesh`*) – The mesh to render

Keyword Arguments

- **`projection_matrix`** (*`numpy.ndarray`*) – projection_matrix (bytes)
- **`model_matrix`** (*`numpy.ndarray`*) – view_matrix (bytes)
- **`camera_matrix`** (*`numpy.ndarray`*) – camera_matrix (bytes)
- **`time`** (*`float`*) – The current time

`MeshProgram.`**`apply`**(*mesh*)
Determine if this `MeshProgram` should be applied to the mesh. Can return self or some `MeshProgram` instance to support dynamic `MeshProgram` creation

Parameters **`mesh`** – The mesh to inspect

Attributes

`MeshProgram.`**`ctx`**
The current context

Type moderngl.Context

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

m

moderngl_window, 3
moderngl_window.conf, 4
moderngl_window.context.base.window, 8
moderngl_window.context.glfw.window, 17
moderngl_window.context.headless.window, 21
moderngl_window.context.pyglet.window, 25
moderngl_window.context.pyqt5.window, 29
moderngl_window.context.pyside2.window, 33
moderngl_window.context.sdl2.window, 38
moderngl_window.finders.base, 64
moderngl_window.finders.data, 66
moderngl_window.finders.program, 65
moderngl_window.finders.scene, 65
moderngl_window.finders.texture, 64
moderngl_window.geometry, 41
moderngl_window.loaders.base, 43
moderngl_window.loaders.data.binary, 55
moderngl_window.loaders.data.json, 53
moderngl_window.loaders.data.text, 54
moderngl_window.loaders.program.separate, 47
moderngl_window.loaders.program.single, 45
moderngl_window.loaders.scene.gltf2, 50
moderngl_window.loaders.scene.stl, 52
moderngl_window.loaders.scene.wavefront, 49
moderngl_window.loaders.texture.array, 48
moderngl_window.loaders.texture.t2d, 44
moderngl_window.meta.base, 56
moderngl_window.meta.data, 62
moderngl_window.meta.program, 59
moderngl_window.meta.scene, 61
moderngl_window.meta.texture, 57
moderngl_window.opengl.projection, 66
moderngl_window.opengl.vao, 67
moderngl_window.resources.base, 70
moderngl_window.resources.data, 73
moderngl_window.resources.programs, 72
moderngl_window.resources.scenes, 73
moderngl_window.resources.textures, 71
moderngl_window.scene, 79
moderngl_window.timers.base, 74
moderngl_window.timers.clock, 75

Index

Symbols

__init__() (*moderngl_window.conf.Settings method*), 4
__init__() (*moderngl_window.context.base.window.BaseWindow method*), 13
__init__() (*moderngl_window.context.base.window.WindowConfig method*), 9
__init__() (*moderngl_window.context.glfw.window.Window method*), 17
__init__() (*moderngl_window.context.headless.window.Window method*), 21
__init__() (*moderngl_window.context.pyglet.window.Window method*), 25
__init__() (*moderngl_window.context.pyqt5.window.Window method*), 29
__init__() (*moderngl_window.context.pyside2.window.Window method*), 34
__init__() (*moderngl_window.context.sdl2.window.Window method*), 38
__init__() (*moderngl_window.finders.base.BaseFilesystemFinder method*), 64
__init__() (*moderngl_window.finders.data.FilesystemFinder method*), 66
__init__() (*moderngl_window.finders.program.FilesystemFinder method*), 65
__init__() (*moderngl_window.finders.scene.FilesystemFinder method*), 66
__init__() (*moderngl_window.finders.texture.FilesystemFinder method*), 65
__init__() (*moderngl_window.loaders.base.BaseLoader method*), 43
__init__() (*moderngl_window.loaders.data.binary.Loader method*), 55
__init__() (*moderngl_window.loaders.data.json.Loader method*), 53
__init__() (*moderngl_window.loaders.data.text.Loader method*), 54
__init__() (*moderngl_window.loaders.program.separate.Loader method*), 47
__init__() (*moderngl_window.loaders.program.single.Loader method*), 45
__init__() (*moderngl_window.loaders.scene.gltf2.Loader method*), 50
__init__() (*moderngl_window.loaders.scene.stl.Loader method*), 52
__init__() (*moderngl_window.loaders.scene.wavefront.Loader method*), 49
__init__() (*moderngl_window.loaders.texture.array.Loader method*), 48
__init__() (*moderngl_window.loaders.texture.t2d.Loader method*), 44
__init__() (*moderngl_window.meta.base.ResourceDescription method*), 56
__init__() (*moderngl_window.meta.data.DataDescription method*), 63
__init__() (*moderngl_window.meta.program.ProgramDescription method*), 59
__init__() (*moderngl_window.meta.scene.SceneDescription method*), 61
__init__() (*moderngl_window.meta.texture.TextureDescription method*), 57
__init__() (*moderngl_window.opengl.projection.Projection3D method*), 66
__init__() (*moderngl_window.opengl.vao.VAO method*), 68
__init__() (*moderngl_window.resources.base.BaseRegistry method*), 70
__init__() (*moderngl_window.resources.data.DataFiles method*), 74
__init__() (*moderngl_window.resources.programs.Programs method*), 72
__init__() (*moderngl_window.resources.scenes.Scenes method*), 73
__init__() (*moderngl_window.resources.textures.Textures method*), 71
__init__() (*moderngl_window.scene.Camera method*), 76
__init__() (*moderngl_window.scene.KeyboardCamera method*), 77

__init__() (*moderngl_window.scene.Material method*), 83
__init__() (*moderngl_window.scene.MaterialTexture method*), 83
__init__() (*moderngl_window.scene.Mesh method*), 81
__init__() (*moderngl_window.scene.Node method*), 80
__init__() (*moderngl_window.scene.Scene method*), 79
__init__() (*moderngl_window.timers.base.BaseTimer method*), 75
__init__() (*moderngl_window.timers.clock.Timer method*), 76

A

activate_context() (*in module moderngl_window*), 3
add() (*moderngl_window.resources.base.BaseRegistry method*), 70
add() (*moderngl_window.resources.data.DataFiles method*), 74
add() (*moderngl_window.resources.programs.Programs method*), 72
add() (*moderngl_window.resources.scenes.Scenes method*), 73
add() (*moderngl_window.resources.textures.Textures method*), 71
add_attribute() (*moderngl_window.scene.Mesh method*), 82
add_child() (*moderngl_window.scene.Node method*), 80
anisotropy (*moderngl_window.meta.texture.TextureDescription attribute*), 58
apply() (*moderngl_window.scene.MeshProgram method*), 84
apply_default_settings() (*moderngl_window.conf.Settings method*), 4
apply_from_cls() (*moderngl_window.conf.Settings method*), 5
apply_from_dict() (*moderngl_window.conf.Settings method*), 5
apply_from_iterable() (*moderngl_window.conf.Settings method*), 6
apply_from_module() (*moderngl_window.conf.Settings method*), 5
apply_from_module_name() (*moderngl_window.conf.Settings method*), 5
apply_mesh_programs() (*moderngl_window.scene.Scene method*), 79
apply_settings_from_env() (*moderngl_window.conf.Settings method*), 5
aspect_ratio (*moderngl_window.context.base.window.BaseWindow attribute*), 16
aspect_ratio (*moderngl_window.context.base.window.WindowConfig attribute*), 12
aspect_ratio (*moderngl_window.context.glfw.window.Window attribute*), 20
aspect_ratio (*moderngl_window.context.headless.window.Window attribute*), 23
aspect_ratio (*moderngl_window.context.pyglet.window.Window attribute*), 28
aspect_ratio (*moderngl_window.context.pyqt5.window.Window attribute*), 32
aspect_ratio (*moderngl_window.context.pyside2.window.Window attribute*), 37
aspect_ratio (*moderngl_window.context.sdl2.window.Window attribute*), 40
aspect_ratio (*moderngl_window.opengl.projection.Projection3D attribute*), 67
attr_names (*moderngl_window.meta.scene.SceneDescription attribute*), 62
attrs (*moderngl_window.meta.base.ResourceDescription attribute*), 56
attrs (*moderngl_window.meta.data.DataDescription attribute*), 63
attrs (*moderngl_window.meta.program.ProgramDescription attribute*), 60
attrs (*moderngl_window.meta.scene.SceneDescription attribute*), 62
attrs (*moderngl_window.meta.texture.TextureDescription attribute*), 58

B

BaseFilesystemFinder (*in module moderngl_window.finders.base*), 64
BaseRegistry (*in module moderngl_window.resources.base*), 70
BaseTimer (*in module moderngl_window.timers.base*), 74
bbox() (*in module moderngl_window.geometry*), 41
buffer() (*moderngl_window.opengl.vao.VAO method*), 69
buffer_size (*moderngl_window.context.base.window.BaseWindow attribute*), 15
buffer_size (*moderngl_window.context.glfw.window.Window attribute*), 19
buffer_size (*moderngl_window.context.headless.window.Window attribute*), 23

buffer_size (*moderngl_window.context.pyglet.window.Window attribute*), 28
buffer_size (*moderngl_window.context.pyqt5.window.Window attribute*), 32
buffer_size (*moderngl_window.context.pyside2.window.Window attribute*), 36
buffer_size (*moderngl_window.context.sdl2.window.Window attribute*), 40

C

cache (*moderngl_window.meta.scene.SceneDescription attribute*), 62
calc_global_bbox() (*moderngl_window.scene.Mesh method*), 82
calc_global_bbox() (*moderngl_window.scene.Node method*), 81
calc_model_mat() (*moderngl_window.scene.Node method*), 81
calc_scene_bbox() (*moderngl_window.scene.Scene method*), 80
Camera (*in module moderngl_window.scene*), 76
children (*moderngl_window.scene.Node attribute*), 81
clear() (*moderngl_window.context.base.window.BaseWindow method*), 14
clear() (*moderngl_window.context.glfw.window.Window method*), 17
clear() (*moderngl_window.context.headless.window.Window method*), 22
clear() (*moderngl_window.context.pyglet.window.Window method*), 25
clear() (*moderngl_window.context.pyqt5.window.Window method*), 30
clear() (*moderngl_window.context.pyside2.window.Window method*), 34
clear() (*moderngl_window.context.sdl2.window.Window method*), 38
close() (*moderngl_window.context.base.window.BaseWindow method*), 14
close() (*moderngl_window.context.glfw.window.Window method*), 17
close() (*moderngl_window.context.headless.window.Window method*), 22
close() (*moderngl_window.context.pyglet.window.Window method*), 25
close() (*moderngl_window.context.pyqt5.window.Window method*), 30
close() (*moderngl_window.context.pyside2.window.Window method*), 34
close() (*moderngl_window.context.sdl2.window.Window method*), 38
close_event() (*moderngl_window.context.pyqt5.window.Window method*), 31
close_event() (*moderngl_window.context.pyside2.window.Window method*), 35
color (*moderngl_window.scene.Material attribute*), 83
config (*moderngl_window.context.base.window.BaseWindow attribute*), 16
config (*moderngl_window.context.glfw.window.Window attribute*), 20
config (*moderngl_window.context.headless.window.Window attribute*), 23
config (*moderngl_window.context.pyglet.window.Window attribute*), 28
config (*moderngl_window.context.pyqt5.window.Window attribute*), 32
config (*moderngl_window.context.pyside2.window.Window attribute*), 36
config (*moderngl_window.context.sdl2.window.Window attribute*), 40
count (*moderngl_window.resources.base.BaseRegistry attribute*), 71
count (*moderngl_window.resources.data.DataFiles attribute*), 74
count (*moderngl_window.resources.programs.Programs attribute*), 72
count (*moderngl_window.resources.scenes.Scenes attribute*), 73
count (*moderngl_window.resources.textures.Textures attribute*), 72
create_window_from_settings() (*in module moderngl_window*), 4
ctx (*moderngl_window.context.base.window.BaseWindow attribute*), 15
ctx (*moderngl_window.context.glfw.window.Window attribute*), 19
ctx (*moderngl_window.context.headless.window.Window attribute*), 22
ctx (*moderngl_window.context.pyglet.window.Window attribute*), 27
ctx (*moderngl_window.context.pyqt5.window.Window attribute*), 31
ctx (*moderngl_window.context.pyside2.window.Window attribute*), 35
ctx (*moderngl_window.context.sdl2.window.Window attribute*), 39
ctx (*moderngl_window.loaders.base.BaseLoader attribute*), 44
ctx (*moderngl_window.loaders.data.binary.Loader attribute*), 55
ctx (*moderngl_window.loaders.data.json.Loader attribute*), 53
ctx (*moderngl_window.loaders.data.text.Loader attribute*), 54
ctx (*moderngl_window.loaders.program.separate.Loader attribute*), 48

ctx (*moderngl_window.loaders.program.single.Loader attribute*), 47
ctx (*moderngl_window.loaders.scene.gltf2.Loader attribute*), 51
ctx (*moderngl_window.loaders.scene.stl.Loader attribute*), 52
ctx (*moderngl_window.loaders.scene.wavefront.Loader attribute*), 50
ctx (*moderngl_window.loaders.texture.array.Loader attribute*), 49
ctx (*moderngl_window.loaders.texture.t2d.Loader attribute*), 45
ctx (*moderngl_window.opengl.vao.VAO attribute*), 70
ctx (*moderngl_window.scene.MeshProgram attribute*), 84
ctx (*moderngl_window.scene.Scene attribute*), 80
ctx() (*in module moderngl_window*), 3
cube() (*in module moderngl_window.geometry*), 42
cursor (*moderngl_window.context.base.window.BaseWindow attribute*), 16
cursor (*moderngl_window.context.base.window.WindowConfig attribute*), 13
cursor (*moderngl_window.context.glfw.window.Window attribute*), 20
cursor (*moderngl_window.context.headless.window.Window attribute*), 24
cursor (*moderngl_window.context.pyglet.window.Window attribute*), 28
cursor (*moderngl_window.context.pyqt5.window.Window attribute*), 32
cursor (*moderngl_window.context.pyside2.window.Window attribute*), 37
cursor (*moderngl_window.context.sdl2.window.Window attribute*), 40

D

DATA_DIRS (*moderngl_window.conf.Settings attribute*), 7
DATA_FINDERS (*moderngl_window.conf.Settings attribute*), 7
DATA_LOADERS (*moderngl_window.conf.Settings attribute*), 8
DataDescription (*in module moderngl_window.meta.data*), 63
DataFiles (*in module moderngl_window.resources.data*), 74
default_kind (*moderngl_window.meta.base.ResourceDescription attribute*), 57
default_kind (*moderngl_window.meta.data.DataDescription attribute*), 64
default_kind (*moderngl_window.meta.program.ProgramDescription attribute*), 61
default_kind (*moderngl_window.meta.scene.SceneDescription attribute*), 62
default_kind (*moderngl_window.meta.texture.TextureDescription attribute*), 59
destroy() (*moderngl_window.context.base.window.BaseWindow method*), 14
destroy() (*moderngl_window.context.glfw.window.Window method*), 18
destroy() (*moderngl_window.context.headless.window.Window method*), 22
destroy() (*moderngl_window.context.pyglet.window.Window method*), 26
destroy() (*moderngl_window.context.pyqt5.window.Window method*), 30
destroy() (*moderngl_window.context.pyside2.window.Window method*), 35
destroy() (*moderngl_window.context.sdl2.window.Window method*), 39
destroy() (*moderngl_window.scene.Scene method*), 80
double_sided (*moderngl_window.scene.Material attribute*), 83
draw() (*moderngl_window.scene.Mesh method*), 82
draw() (*moderngl_window.scene.MeshProgram method*), 84
draw() (*moderngl_window.scene.Node method*), 80
draw() (*moderngl_window.scene.Scene method*), 79
draw_bbox() (*moderngl_window.scene.Mesh method*), 82
draw_bbox() (*moderngl_window.scene.Node method*), 80
draw_bbox() (*moderngl_window.scene.Scene method*), 79

F

far (*moderngl_window.opengl.projection.Projection3D attribute*), 67
fbo (*moderngl_window.context.base.window.BaseWindow attribute*), 15
fbo (*moderngl_window.context.glfw.window.Window attribute*), 19
fbo (*moderngl_window.context.headless.window.Window attribute*), 22
fbo (*moderngl_window.context.pyglet.window.Window attribute*), 27
fbo (*moderngl_window.context.pyqt5.window.Window attribute*), 31
fbo (*moderngl_window.context.pyside2.window.Window attribute*), 36
fbo (*moderngl_window.context.sdl2.window.Window attribute*), 39

file_extensions (*moderngl_window.loaders.base.BaseLoader attribute*), 44
file_extensions (*moderngl_window.loaders.data.binary.Loader attribute*), 55
file_extensions (*moderngl_window.loaders.data.json.Loader attribute*), 53
file_extensions (*moderngl_window.loaders.data.text.Loader attribute*), 54
file_extensions (*moderngl_window.loaders.program.separate.Loader attribute*), 48
file_extensions (*moderngl_window.loaders.program.single.Loader attribute*), 47
file_extensions (*moderngl_window.loaders.scene.gltf2.Loader attribute*), 51
file_extensions (*moderngl_window.loaders.scene.stl.Loader attribute*), 52
file_extensions (*moderngl_window.loaders.scene.wavefront.Loader attribute*), 50
file_extensions (*moderngl_window.loaders.texture.array.Loader attribute*), 49
file_extensions (*moderngl_window.loaders.texture.t2d.Loader attribute*), 45
FilesystemFinder (*in module moderngl_window.finders.data*), 66
FilesystemFinder (*in module moderngl_window.finders.program*), 65
FilesystemFinder (*in module moderngl_window.finders.scene*), 65
FilesystemFinder (*in module moderngl_window.finders.texture*), 65
find() (*moderngl_window.finders.base.BaseFilesystemFinder method*), 64
find() (*moderngl_window.finders.data.FilesystemFinder method*), 66
find() (*moderngl_window.finders.program.FilesystemFinder method*), 65
find() (*moderngl_window.finders.scene.FilesystemFinder method*), 66
find() (*moderngl_window.finders.texture.FilesystemFinder method*), 65
find_data() (*moderngl_window.loaders.base.BaseLoader method*), 43
find_data() (*moderngl_window.loaders.data.binary.Loader method*), 55
find_data() (*moderngl_window.loaders.data.json.Loader method*), 53
find_data() (*moderngl_window.loaders.data.text.Loader method*), 54
find_data() (*moderngl_window.loaders.program.separate.Loader method*), 47
find_data() (*moderngl_window.loaders.program.single.Loader method*), 46
find_data() (*moderngl_window.loaders.scene.gltf2.Loader method*), 50
find_data() (*moderngl_window.loaders.scene.stl.Loader method*), 52
find_data() (*moderngl_window.loaders.scene.wavefront.Loader method*), 49
find_data() (*moderngl_window.loaders.texture.array.Loader method*), 48
find_data() (*moderngl_window.loaders.texture.t2d.Loader method*), 45
find_program() (*moderngl_window.loaders.base.BaseLoader method*), 43
find_program() (*moderngl_window.loaders.data.binary.Loader method*), 55
find_program() (*moderngl_window.loaders.data.json.Loader method*), 53
find_program() (*moderngl_window.loaders.data.text.Loader method*), 54
find_program() (*moderngl_window.loaders.program.separate.Loader method*), 47
find_program() (*moderngl_window.loaders.program.single.Loader method*), 46
find_program() (*moderngl_window.loaders.scene.gltf2.Loader method*), 51
find_program() (*moderngl_window.loaders.scene.stl.Loader method*), 52
find_program() (*moderngl_window.loaders.scene.wavefront.Loader method*), 49
find_program() (*moderngl_window.loaders.texture.array.Loader method*), 48
find_program() (*moderngl_window.loaders.texture.t2d.Loader method*), 45
find_scene() (*moderngl_window.loaders.base.BaseLoader*

method), 44
find_scene() (*moderngl_window.loaders.data.binary.Loader method*), 55
find_scene() (*moderngl_window.loaders.data.json.Loader method*), 53
find_scene() (*moderngl_window.loaders.data.text.Loader method*), 54
find_scene() (*moderngl_window.loaders.program.separate.Loader method*), 48
find_scene() (*moderngl_window.loaders.program.single.Loader method*), 47
find_scene() (*moderngl_window.loaders.scene.gltf2.Loader method*), 51
find_scene() (*moderngl_window.loaders.scene.stl.Loader method*), 52
find_scene() (*moderngl_window.loaders.scene.wavefront.Loader method*), 50
find_scene() (*moderngl_window.loaders.texture.array.Loader method*), 49
find_scene() (*moderngl_window.loaders.texture.t2d.Loader method*), 45
find_texture() (*moderngl_window.loaders.base.BaseLoader method*), 44
find_texture() (*moderngl_window.loaders.data.binary.Loader method*), 55
find_texture() (*moderngl_window.loaders.data.json.Loader method*), 53
find_texture() (*moderngl_window.loaders.data.text.Loader method*), 54
find_texture() (*moderngl_window.loaders.program.separate.Loader method*), 47
find_texture() (*moderngl_window.loaders.program.single.Loader method*), 46
find_texture() (*moderngl_window.loaders.scene.gltf2.Loader method*), 51
find_texture() (*moderngl_window.loaders.scene.stl.Loader method*), 52
find_texture() (*moderngl_window.loaders.scene.wavefront.Loader method*), 50
find_texture() (*moderngl_window.loaders.texture.array.Loader method*), 49
find_texture() (*moderngl_window.loaders.texture.t2d.Loader method*), 45
find_window_classes() (*in module moderngl_window*), 4
flip (*moderngl_window.meta.texture.TextureDescription attribute*), 58
fov (*moderngl_window.opengl.projection.Projection3D attribute*), 67
fragment_shader (*moderngl_window.meta.program.ProgramDescription attribute*), 60
frames (*moderngl_window.context.base.window.BaseWindow attribute*), 15
frames (*moderngl_window.context.glfw.window.Window attribute*), 20
frames (*moderngl_window.context.headless.window.Window attribute*), 23
frames (*moderngl_window.context.pyglet.window.Window attribute*), 28
frames (*moderngl_window.context.pyqt5.window.Window attribute*), 32
frames (*moderngl_window.context.pyside2.window.Window attribute*), 36
frames (*moderngl_window.context.sdl2.window.Window attribute*), 40
fullscreen (*moderngl_window.context.base.window.BaseWindow attribute*), 15
fullscreen (*moderngl_window.context.glfw.window.Window attribute*), 20
fullscreen (*moderngl_window.context.headless.window.Window attribute*), 23
fullscreen (*moderngl_window.context.pyglet.window.Window attribute*), 28
fullscreen (*moderngl_window.context.pyqt5.window.Window attribute*), 32
fullscreen (*moderngl_window.context.pyside2.window.Window attribute*), 36
fullscreen (*moderngl_window.context.sdl2.window.Window attribute*), 40

G

geometry_shader (*moderngl_window.meta.program.ProgramDescription attribute*), 60
get_buffer_by_name() (*moderngl_window.opengl.vao.VAO method*),

70
get_local_window_cls() (*in module moderngl_window*), 3
get_window_cls() (*in module moderngl_window*), 3
gl_version (*moderngl_window.context.base.window.BaseWindow attribute*), 15
gl_version (*moderngl_window.context.base.window.WindowConfig attribute*), 12
gl_version (*moderngl_window.context.glfw.window.Window attribute*), 19
gl_version (*moderngl_window.context.headless.window.Window attribute*), 23
gl_version (*moderngl_window.context.pyglet.window.Window attribute*), 27
gl_version (*moderngl_window.context.pyqt5.window.Window attribute*), 32
gl_version (*moderngl_window.context.pyside2.window.Window attribute*), 36
gl_version (*moderngl_window.context.sdl2.window.Window attribute*), 39
gl_version_code (*moderngl_window.context.base.window.BaseWindow attribute*), 17
gl_version_code (*moderngl_window.context.glfw.window.Window attribute*), 21
gl_version_code (*moderngl_window.context.headless.window.Window attribute*), 24
gl_version_code (*moderngl_window.context.pyglet.window.Window attribute*), 29
gl_version_code (*moderngl_window.context.pyqt5.window.Window attribute*), 33
gl_version_code (*moderngl_window.context.pyside2.window.Window attribute*), 37
gl_version_code (*moderngl_window.context.sdl2.window.Window attribute*), 41
glfw_key_event_callback() (*moderngl_window.context.glfw.window.Window method*), 19
glfw_mouse_button_callback() (*moderngl_window.context.glfw.window.Window method*), 18
glfw_mouse_event_callback() (*moderngl_window.context.glfw.window.Window method*), 18
glfw_window_resize_callback() (*moderngl_window.context.glfw.window.Window method*), 18

H

has_normals() (*moderngl_window.scene.Mesh method*), 82
has_uvs() (*moderngl_window.scene.Mesh method*), 82
height (*moderngl_window.context.base.window.BaseWindow attribute*), 15
height (*moderngl_window.context.glfw.window.Window attribute*), 19
height (*moderngl_window.context.headless.window.Window attribute*), 23
height (*moderngl_window.context.pyglet.window.Window attribute*), 28
height (*moderngl_window.context.pyqt5.window.Window attribute*), 32
height (*moderngl_window.context.pyside2.window.Window attribute*), 36
height (*moderngl_window.context.sdl2.window.Window attribute*), 40

I

image (*moderngl_window.meta.texture.TextureDescription attribute*), 58
index_buffer() (*moderngl_window.opengl.vao.VAO method*), 69
init_mgl_context() (*moderngl_window.context.base.window.BaseWindow method*), 14
init_mgl_context() (*moderngl_window.context.glfw.window.Window method*), 17
init_mgl_context() (*moderngl_window.context.headless.window.Window method*), 21
init_mgl_context() (*moderngl_window.context.pyglet.window.Window method*), 25
init_mgl_context() (*moderngl_window.context.pyqt5.window.Window method*), 30
init_mgl_context() (*moderngl_window.context.pyside2.window.Window method*), 34
init_mgl_context() (*moderngl_window.context.sdl2.window.Window method*), 38
instance() (*moderngl_window.opengl.vao.VAO method*), 69
is_closing (*moderngl_window.context.base.window.BaseWindow attribute*), 17
is_closing (*moderngl_window.context.glfw.window.Window attribute*), 21

is_closing (*moderngl_window.context.headless.window.Window attribute*), 24
is_closing (*moderngl_window.context.pyglet.window.Window attribute*), 29
is_closing (*moderngl_window.context.pyqt5.window.Window attribute*), 33
is_closing (*moderngl_window.context.pyside2.window.Window attribute*), 37
is_closing (*moderngl_window.context.sdl2.window.Window attribute*), 41
is_key_pressed() (*moderngl_window.context.base.window.BaseWindow method*), 14
is_key_pressed() (*moderngl_window.context.glfw.window.Window method*), 17
is_key_pressed() (*moderngl_window.context.headless.window.Window method*), 22
is_key_pressed() (*moderngl_window.context.pyglet.window.Window method*), 25
is_key_pressed() (*moderngl_window.context.pyqt5.window.Window method*), 30
is_key_pressed() (*moderngl_window.context.pyside2.window.Window method*), 34
is_key_pressed() (*moderngl_window.context.sdl2.window.Window method*), 38
is_paused (*moderngl_window.timers.base.BaseTimer attribute*), 75
is_paused (*moderngl_window.timers.clock.Timer attribute*), 76
is_running (*moderngl_window.timers.base.BaseTimer attribute*), 75
is_running (*moderngl_window.timers.clock.Timer attribute*), 76

K

key_event() (*moderngl_window.context.base.window.WindowConfig method*), 9
key_event_func (*moderngl_window.context.base.window.BaseWindow attribute*), 16
key_event_func (*moderngl_window.context.glfw.window.Window attribute*), 20
key_event_func (*moderngl_window.context.headless.window.Window attribute*), 24
key_event_func (*moderngl_window.context.pyglet.window.Window attribute*), 29
key_event_func (*moderngl_window.context.pyqt5.window.Window attribute*), 33
key_event_func (*moderngl_window.context.pyside2.window.Window attribute*), 37
key_event_func (*moderngl_window.context.sdl2.window.Window attribute*), 41
key_input() (*moderngl_window.scene.KeyboardCamera method*), 78
key_pressed_event() (*moderngl_window.context.pyqt5.window.Window method*), 31
key_pressed_event() (*moderngl_window.context.pyside2.window.Window method*), 35
key_release_event() (*moderngl_window.context.pyqt5.window.Window method*), 31
key_release_event() (*moderngl_window.context.pyside2.window.Window method*), 35
KeyboardCamera (*in module moderngl_window.scene*), 77
keys (*moderngl_window.context.base.window.BaseWindow attribute*), 15
keys (*moderngl_window.context.glfw.window.Window attribute*), 19
keys (*moderngl_window.context.headless.window.Window attribute*), 22
keys (*moderngl_window.context.pyglet.window.Window attribute*), 27
keys (*moderngl_window.context.pyqt5.window.Window attribute*), 31
keys (*moderngl_window.context.pyside2.window.Window attribute*), 35
keys (*moderngl_window.context.sdl2.window.Window attribute*), 39
kind (*moderngl_window.loaders.base.BaseLoader attribute*), 44
kind (*moderngl_window.loaders.data.binary.Loader attribute*), 55
kind (*moderngl_window.loaders.data.json.Loader attribute*), 53
kind (*moderngl_window.loaders.data.text.Loader attribute*), 54
kind (*moderngl_window.loaders.program.separate.Loader attribute*), 48
kind (*moderngl_window.loaders.program.single.Loader attribute*), 47
kind (*moderngl_window.loaders.scene.gltf2.Loader attribute*), 51

kind (*moderngl_window.loaders.scene.stl.Loader attribute*), 52
kind (*moderngl_window.loaders.scene.wavefront.Loader attribute*), 50
kind (*moderngl_window.loaders.texture.array.Loader attribute*), 49
kind (*moderngl_window.loaders.texture.t2d.Loader attribute*), 45
kind (*moderngl_window.meta.base.ResourceDescription attribute*), 56
kind (*moderngl_window.meta.data.DataDescription attribute*), 64
kind (*moderngl_window.meta.program.ProgramDescription attribute*), 60
kind (*moderngl_window.meta.scene.SceneDescription attribute*), 62
kind (*moderngl_window.meta.texture.TextureDescription attribute*), 58

L

label (*moderngl_window.meta.base.ResourceDescription attribute*), 56
label (*moderngl_window.meta.data.DataDescription attribute*), 63
label (*moderngl_window.meta.program.ProgramDescription attribute*), 60
label (*moderngl_window.meta.scene.SceneDescription attribute*), 62
label (*moderngl_window.meta.texture.TextureDescription attribute*), 58
layers (*moderngl_window.meta.texture.TextureDescription attribute*), 58
load() (*moderngl_window.loaders.base.BaseLoader method*), 43
load() (*moderngl_window.loaders.data.binary.Loader method*), 55
load() (*moderngl_window.loaders.data.json.Loader method*), 53
load() (*moderngl_window.loaders.data.text.Loader method*), 54
load() (*moderngl_window.loaders.program.separate.Loader method*), 47
load() (*moderngl_window.loaders.program.single.Loader method*), 45
load() (*moderngl_window.loaders.scene.gltf2.Loader method*), 50
load() (*moderngl_window.loaders.scene.stl.Loader method*), 52
load() (*moderngl_window.loaders.scene.wavefront.Loader method*), 49
load() (*moderngl_window.loaders.texture.array.Loader method*), 48
load() (*moderngl_window.loaders.texture.t2d.Loader method*), 44
load() (*moderngl_window.resources.base.BaseRegistry method*), 70
load() (*moderngl_window.resources.data.DataFiles method*), 74
load() (*moderngl_window.resources.programs.Programs method*), 72
load() (*moderngl_window.resources.scenes.Scenes method*), 73
load() (*moderngl_window.resources.textures.Textures method*), 71
load_binary() (*moderngl_window.context.base.window.WindowConfig method*), 11
load_glb() (*moderngl_window.loaders.scene.gltf2.Loader method*), 51
load_gltf() (*moderngl_window.loaders.scene.gltf2.Loader method*), 51
load_images() (*moderngl_window.loaders.scene.gltf2.Loader method*), 51
load_json() (*moderngl_window.context.base.window.WindowConfig method*), 11
load_materials() (*moderngl_window.loaders.scene.gltf2.Loader method*), 51
load_meshes() (*moderngl_window.loaders.scene.gltf2.Loader method*), 51
load_node() (*moderngl_window.loaders.scene.gltf2.Loader method*), 51
load_nodes() (*moderngl_window.loaders.scene.gltf2.Loader method*), 51
load_pool() (*moderngl_window.resources.base.BaseRegistry method*), 70
load_pool() (*moderngl_window.resources.data.DataFiles method*), 74
load_pool() (*moderngl_window.resources.programs.Programs method*), 72
load_pool() (*moderngl_window.resources.scenes.Scenes method*), 73
load_pool() (*moderngl_window.resources.textures.Textures method*), 71
load_program() (*moderngl_window.context.base.window.WindowConfig method*), 11
load_samplers() (*moderngl_window.loaders.scene.gltf2.Loader method*), 51
load_scene() (*moderngl_window.context.base.window.WindowConfig method*), 12
load_shader() (*moderngl_window.loaders.program.separate.Loader*

method), 48
load_text() (moderngl_window.context.base.window.WindowConfig method), 11
load_texture_2d() (moderngl_window.context.base.window.WindowConfig method), 10
load_texture_array() (moderngl_window.context.base.window.WindowConfig method), 10
load_textures() (moderngl_window.loaders.scene.gltf2.Loader method), 51
loader_cls (moderngl_window.meta.base.ResourceDescription attribute), 56
loader_cls (moderngl_window.meta.data.DataDescription attribute), 64
loader_cls (moderngl_window.meta.program.ProgramDescription attribute), 61
loader_cls (moderngl_window.meta.scene.SceneDescription attribute), 62
loader_cls (moderngl_window.meta.texture.TextureDescription attribute), 59
loaders (moderngl_window.resources.base.BaseRegistry attribute), 71
loaders (moderngl_window.resources.data.DataFiles attribute), 74
loaders (moderngl_window.resources.programs.Programs attribute), 73
loaders (moderngl_window.resources.scenes.Scenes attribute), 73
loaders (moderngl_window.resources.textures.Textures attribute), 72
log_level (moderngl_window.context.base.window.WindowConfig attribute), 13
look_at() (moderngl_window.scene.Camera method), 77
look_at() (moderngl_window.scene.KeyboardCamera method), 79

M

mat_texture (moderngl_window.scene.Material attribute), 83
Material (in module moderngl_window.scene), 83
MaterialTexture (in module moderngl_window.scene), 83
matrix (moderngl_window.opengl.projection.Projection3D attribute), 67
matrix (moderngl_window.scene.Camera attribute), 77
matrix (moderngl_window.scene.KeyboardCamera attribute), 79
Mesh (in module moderngl_window.scene), 81
MeshProgram (in module moderngl_window.scene), 84
mipmap (moderngl_window.meta.texture.TextureDescription attribute), 58
mipmap_levels (moderngl_window.meta.texture.TextureDescription attribute), 58
model_matrix (moderngl_window.scene.Scene attribute), 80
moderngl_window (module), 3
moderngl_window.conf (module), 4
moderngl_window.context.base.window (module), 8, 13
moderngl_window.context.glfw.window (module), 17
moderngl_window.context.headless.window (module), 21
moderngl_window.context.pyglet.window (module), 25
moderngl_window.context.pyqt5.window (module), 29
moderngl_window.context.pyside2.window (module), 33
moderngl_window.context.sdl2.window (module), 38
moderngl_window.finders.base (module), 64
moderngl_window.finders.data (module), 66
moderngl_window.finders.program (module), 65
moderngl_window.finders.scene (module), 65
moderngl_window.finders.texture (module), 64
moderngl_window.geometry (module), 41
moderngl_window.loaders.base (module), 43
moderngl_window.loaders.data.binary (module), 55
moderngl_window.loaders.data.json (module), 53
moderngl_window.loaders.data.text (module), 54
moderngl_window.loaders.program.separate (module), 47
moderngl_window.loaders.program.single (module), 45
moderngl_window.loaders.scene.gltf2 (module), 50
moderngl_window.loaders.scene.stl (module), 52
moderngl_window.loaders.scene.wavefront (module), 49
moderngl_window.loaders.texture.array (module), 48
moderngl_window.loaders.texture.t2d (module), 44
moderngl_window.meta.base (module), 56
moderngl_window.meta.data (module), 62
moderngl_window.meta.program (module), 59
moderngl_window.meta.scene (module), 61

moderngl_window.meta.texture (*module*), 57
moderngl_window.opengl.projection (*module*), 66
moderngl_window.opengl.vao (*module*), 67
moderngl_window.resources.base (*module*), 70
moderngl_window.resources.data (*module*), 73
moderngl_window.resources.programs (*module*), 72
moderngl_window.resources.scenes (*module*), 73
moderngl_window.resources.textures (*module*), 71
moderngl_window.scene (*module*), 76, 77, 79–84
moderngl_window.timers.base (*module*), 74
moderngl_window.timers.clock (*module*), 75
modifiers (*moderngl_window.context.base.window.BaseWindow attribute*), 17
modifiers (*moderngl_window.context.glfw.window.Window attribute*), 21
modifiers (*moderngl_window.context.headless.window.Window attribute*), 24
modifiers (*moderngl_window.context.pyglet.window.Window attribute*), 29
modifiers (*moderngl_window.context.pyqt5.window.Window attribute*), 33
modifiers (*moderngl_window.context.pyside2.window.Window attribute*), 37
modifiers (*moderngl_window.context.sdl2.window.Window attribute*), 41
mouse_move_event() (*moderngl_window.context.pyqt5.window.Window method*), 31
mouse_move_event() (*moderngl_window.context.pyside2.window.Window method*), 35
mouse_position_event() (*moderngl_window.context.base.window.WindowConfig method*), 10
mouse_position_event_func (*moderngl_window.context.base.window.BaseWindow attribute*), 16
mouse_position_event_func (*moderngl_window.context.glfw.window.Window attribute*), 21
mouse_position_event_func (*moderngl_window.context.headless.window.Window attribute*), 24
mouse_position_event_func (*moderngl_window.context.pyglet.window.Window attribute*), 29
mouse_position_event_func (*moderngl_window.context.pyqt5.window.Window attribute*), 33
mouse_position_event_func (*moderngl_window.context.pyside2.window.Window attribute*), 37
mouse_position_event_func (*moderngl_window.context.sdl2.window.Window attribute*), 41
mouse_press_event() (*moderngl_window.context.base.window.WindowConfig method*), 10
mouse_press_event() (*moderngl_window.context.pyqt5.window.Window method*), 31
mouse_press_event() (*moderngl_window.context.pyside2.window.Window method*), 35
mouse_press_event_func (*moderngl_window.context.base.window.BaseWindow attribute*), 16
mouse_press_event_func (*moderngl_window.context.glfw.window.Window attribute*), 21
mouse_press_event_func (*moderngl_window.context.headless.window.Window attribute*), 24
mouse_press_event_func (*moderngl_window.context.pyglet.window.Window attribute*), 29
mouse_press_event_func (*moderngl_window.context.pyqt5.window.Window attribute*), 33
mouse_press_event_func (*moderngl_window.context.pyside2.window.Window attribute*), 37
mouse_press_event_func (*moderngl_window.context.sdl2.window.Window attribute*), 41
mouse_release_event() (*moderngl_window.context.base.window.WindowConfig method*), 10
mouse_release_event() (*moderngl_window.context.pyqt5.window.Window method*), 31
mouse_release_event() (*moderngl_window.context.pyside2.window.Window method*), 35
mouse_release_event_func (*moderngl_window.context.base.window.BaseWindow attribute*), 16
mouse_release_event_func (*moderngl_window.context.glfw.window.Window attribute*), 21
mouse_release_event_func (*moderngl_window.context.headless.window.Window*

attribute), 24
mouse_release_event_func (*moderngl_window.context.pyglet.window.Window attribute*), 29
mouse_release_event_func (*moderngl_window.context.pyqt5.window.Window attribute*), 33
mouse_release_event_func (*moderngl_window.context.pyside2.window.Window attribute*), 37
mouse_release_event_func (*moderngl_window.context.sdl2.window.Window attribute*), 41
move_backward() (*moderngl_window.scene.KeyboardCamera method*), 78
move_down() (*moderngl_window.scene.KeyboardCamera method*), 78
move_forward() (*moderngl_window.scene.KeyboardCamera method*), 78
move_left() (*moderngl_window.scene.KeyboardCamera method*), 78
move_right() (*moderngl_window.scene.KeyboardCamera method*), 78
move_state() (*moderngl_window.scene.KeyboardCamera method*), 78
move_up() (*moderngl_window.scene.KeyboardCamera method*), 78

N

name (*moderngl_window.scene.Material attribute*), 83
near (*moderngl_window.opengl.projection.Projection3D attribute*), 67
next_frame() (*moderngl_window.timers.base.BaseTimer method*), 75
next_frame() (*moderngl_window.timers.clock.Timer method*), 76
Node (*in module moderngl_window.scene*), 80

O

on_key_press() (*moderngl_window.context.pyglet.window.Window method*), 26
on_key_release() (*moderngl_window.context.pyglet.window.Window method*), 26
on_mouse_drag() (*moderngl_window.context.pyglet.window.Window method*), 26
on_mouse_motion() (*moderngl_window.context.pyglet.window.Window method*), 27
on_mouse_press() (*moderngl_window.context.pyglet.window.Window method*), 26
on_mouse_release() (*moderngl_window.context.pyglet.window.Window method*), 27
on_resize() (*moderngl_window.context.pyglet.window.Window method*), 27

P

parse_args() (*in module moderngl_window*), 4
path (*moderngl_window.meta.base.ResourceDescription attribute*), 56
path (*moderngl_window.meta.data.DataDescription attribute*), 63
path (*moderngl_window.meta.program.ProgramDescription attribute*), 60
path (*moderngl_window.meta.scene.SceneDescription attribute*), 62
path (*moderngl_window.meta.texture.TextureDescription attribute*), 58
pause() (*moderngl_window.timers.base.BaseTimer method*), 75
pause() (*moderngl_window.timers.clock.Timer method*), 76
pixel_ratio (*moderngl_window.context.base.window.BaseWindow attribute*), 15
pixel_ratio (*moderngl_window.context.glfw.window.Window attribute*), 20
pixel_ratio (*moderngl_window.context.headless.window.Window attribute*), 23
pixel_ratio (*moderngl_window.context.pyglet.window.Window attribute*), 28
pixel_ratio (*moderngl_window.context.pyqt5.window.Window attribute*), 32
pixel_ratio (*moderngl_window.context.pyside2.window.Window attribute*), 36
pixel_ratio (*moderngl_window.context.sdl2.window.Window attribute*), 40
prepare() (*moderngl_window.scene.Scene method*), 80
print_context_info() (*moderngl_window.context.base.window.BaseWindow method*), 14
print_context_info() (*moderngl_window.context.glfw.window.Window method*), 18
print_context_info() (*moderngl_window.context.headless.window.Window method*), 22

print_context_info() (*moderngl_window.context.pyglet.window.Window method*), 26
print_context_info() (*moderngl_window.context.pyqt5.window.Window method*), 31
print_context_info() (*moderngl_window.context.pyside2.window.Window method*), 35
print_context_info() (*moderngl_window.context.sdl2.window.Window method*), 39
process_events() (*moderngl_window.context.sdl2.window.Window method*), 39
PROGRAM_DIRS (*moderngl_window.conf.Settings attribute*), 7
PROGRAM_FINDERS (*moderngl_window.conf.Settings attribute*), 7
PROGRAM_LOADERS (*moderngl_window.conf.Settings attribute*), 7
ProgramDescription (*in module moderngl_window.meta.program*), 59
Programs (*in module moderngl_window.resources.programs*), 72
Projection3D (*in module moderngl_window.opengl.projection*), 66
projection_constants (*moderngl_window.opengl.projection.Projection3D attribute*), 67

Q

quad_2d() (*in module moderngl_window.geometry*), 42
quad_fs() (*in module moderngl_window.geometry*), 42

R

release() (*moderngl_window.opengl.vao.VAO method*), 70
reloadable (*moderngl_window.meta.program.ProgramDescription attribute*), 60
render() (*moderngl_window.context.base.window.BaseWindow method*), 14
render() (*moderngl_window.context.base.window.WindowConfig method*), 9
render() (*moderngl_window.context.glfw.window.Window method*), 18
render() (*moderngl_window.context.headless.window.Window method*), 22
render() (*moderngl_window.context.pyglet.window.Window method*), 25
render() (*moderngl_window.context.pyqt5.window.Window method*), 30
render() (*moderngl_window.context.pyside2.window.Window method*), 34
render() (*moderngl_window.context.sdl2.window.Window method*), 38
render() (*moderngl_window.opengl.vao.VAO method*), 68
render_func (*moderngl_window.context.base.window.BaseWindow attribute*), 16
render_func (*moderngl_window.context.glfw.window.Window attribute*), 20
render_func (*moderngl_window.context.headless.window.Window attribute*), 24
render_func (*moderngl_window.context.pyglet.window.Window attribute*), 28
render_func (*moderngl_window.context.pyqt5.window.Window attribute*), 33
render_func (*moderngl_window.context.pyside2.window.Window attribute*), 37
render_func (*moderngl_window.context.sdl2.window.Window attribute*), 40
render_indirect() (*moderngl_window.opengl.vao.VAO method*), 68
resizable (*moderngl_window.context.base.window.BaseWindow attribute*), 15
resizable (*moderngl_window.context.base.window.WindowConfig attribute*), 12
resizable (*moderngl_window.context.glfw.window.Window attribute*), 20
resizable (*moderngl_window.context.headless.window.Window attribute*), 23
resizable (*moderngl_window.context.pyglet.window.Window attribute*), 28
resizable (*moderngl_window.context.pyqt5.window.Window attribute*), 32
resizable (*moderngl_window.context.pyside2.window.Window attribute*), 36
resizable (*moderngl_window.context.sdl2.window.Window attribute*), 40
resize() (*moderngl_window.context.base.window.BaseWindow method*), 14
resize() (*moderngl_window.context.base.window.WindowConfig method*), 9
resize() (*moderngl_window.context.glfw.window.Window method*), 18
resize() (*moderngl_window.context.headless.window.Window method*), 22
resize() (*moderngl_window.context.pyglet.window.Window method*), 26
resize() (*moderngl_window.context.pyqt5.window.Window method*), 30
resize() (*moderngl_window.context.pyside2.window.Window method*), 35
resize() (*moderngl_window.context.sdl2.window.Window*

method), 39
resize_func (*moderngl_window.context.base.window.BaseWindow attribute*), 16
resize_func (*moderngl_window.context.glfw.window.Window attribute*), 20
resize_func (*moderngl_window.context.headless.window.Window attribute*), 24
resize_func (*moderngl_window.context.pyglet.window.Window attribute*), 29
resize_func (*moderngl_window.context.pyqt5.window.Window attribute*), 33
resize_func (*moderngl_window.context.pyside2.window.Window attribute*), 37
resize_func (*moderngl_window.context.sdl2.window.Window attribute*), 41
resolve_loader() (*moderngl_window.resources.base.BaseRegistry method*), 70
resolve_loader() (*moderngl_window.resources.data.DataFiles method*), 74
resolve_loader() (*moderngl_window.resources.programs.Programs method*), 72
resolve_loader() (*moderngl_window.resources.scenes.Scenes method*), 73
resolve_loader() (*moderngl_window.resources.textures.Textures method*), 71
resolved_path (*moderngl_window.meta.base.ResourceDescription attribute*), 56
resolved_path (*moderngl_window.meta.data.DataDescription attribute*), 63
resolved_path (*moderngl_window.meta.program.ProgramDescription attribute*), 60
resolved_path (*moderngl_window.meta.scene.SceneDescription attribute*), 62
resolved_path (*moderngl_window.meta.texture.TextureDescription attribute*), 58
resource_dir (*moderngl_window.context.base.window.WindowConfig attribute*), 13
resource_type (*moderngl_window.meta.base.ResourceDescription attribute*), 57
resource_type (*moderngl_window.meta.data.DataDescription attribute*), 64
resource_type (*moderngl_window.meta.program.ProgramDescription attribute*), 61
resource_type (*moderngl_window.meta.scene.SceneDescription attribute*), 62
resource_type (*moderngl_window.meta.texture.TextureDescription attribute*), 59
ResourceDescription (*in module moderngl_window.meta.base*), 56
rotate() (*moderngl_window.scene.KeyboardCamera method*), 78
run_window_config() (*in module moderngl_window*), 4

S

sampler (*moderngl_window.scene.MaterialTexture attribute*), 84
samples (*moderngl_window.context.base.window.BaseWindow attribute*), 16
samples (*moderngl_window.context.base.window.WindowConfig attribute*), 13
samples (*moderngl_window.context.glfw.window.Window attribute*), 20
samples (*moderngl_window.context.headless.window.Window attribute*), 24
samples (*moderngl_window.context.pyglet.window.Window attribute*), 28
samples (*moderngl_window.context.pyqt5.window.Window attribute*), 32
samples (*moderngl_window.context.pyside2.window.Window attribute*), 37
samples (*moderngl_window.context.sdl2.window.Window attribute*), 40
SCENE_DIRS (*moderngl_window.conf.Settings attribute*), 7
SCENE_FINDERS (*moderngl_window.conf.Settings attribute*), 7
SCENE_LOADERS (*moderngl_window.conf.Settings attribute*), 8
SceneDescription (*in module moderngl_window.meta.scene*), 61
Scenes (*in module moderngl_window.resources.scenes*), 73
SCREENSHOT_PATH (*moderngl_window.conf.Settings attribute*), 7
set_default_viewport() (*moderngl_window.context.base.window.BaseWindow method*), 14
set_default_viewport() (*moderngl_window.context.glfw.window.Window method*), 18

set_default_viewport() (*moderngl_window.context.headless.window.Window method*), 22
set_default_viewport() (*moderngl_window.context.pyglet.window.Window method*), 26
set_default_viewport() (*moderngl_window.context.pyqt5.window.Window method*), 31
set_default_viewport() (*moderngl_window.context.pyside2.window.Window method*), 35
set_default_viewport() (*moderngl_window.context.sdl2.window.Window method*), 39
set_position() (*moderngl_window.scene.Camera method*), 77
set_position() (*moderngl_window.scene.KeyboardCamera method*), 78
Settings (*in module moderngl_window.conf*), 4
settings_attr (*moderngl_window.finders.base.BaseFilesystemFinder attribute*), 64
settings_attr (*moderngl_window.finders.data.FilesystemFinder attribute*), 66
settings_attr (*moderngl_window.finders.program.FilesystemFinder attribute*), 65
settings_attr (*moderngl_window.finders.scene.FilesystemFinder attribute*), 66
settings_attr (*moderngl_window.finders.texture.FilesystemFinder attribute*), 65
settings_attr (*moderngl_window.resources.base.BaseRegistry attribute*), 71
settings_attr (*moderngl_window.resources.data.DataFiles attribute*), 74
settings_attr (*moderngl_window.resources.programs.Programs attribute*), 72
settings_attr (*moderngl_window.resources.scenes.Scenes attribute*), 73
settings_attr (*moderngl_window.resources.textures.Textures attribute*), 72
setup_basic_logging() (*in module moderngl_window*), 3
size (*moderngl_window.context.base.window.BaseWindow attribute*), 15
size (*moderngl_window.context.glfw.window.Window attribute*), 19
size (*moderngl_window.context.headless.window.Window attribute*), 23
size (*moderngl_window.context.pyglet.window.Window attribute*), 28
size (*moderngl_window.context.pyqt5.window.Window attribute*), 32
size (*moderngl_window.context.pyside2.window.Window attribute*), 36
size (*moderngl_window.context.sdl2.window.Window attribute*), 40
sphere() (*in module moderngl_window.geometry*), 43
start() (*moderngl_window.timers.base.BaseTimer method*), 75
start() (*moderngl_window.timers.clock.Timer method*), 76
stop() (*moderngl_window.timers.base.BaseTimer method*), 75
stop() (*moderngl_window.timers.clock.Timer method*), 76
supported_extensions (*moderngl_window.loaders.scene.gltf2.Loader attribute*), 52
supports_file() (*moderngl_window.loaders.base.BaseLoader class method*), 43
supports_file() (*moderngl_window.loaders.data.binary.Loader class method*), 55
supports_file() (*moderngl_window.loaders.data.json.Loader class method*), 53
supports_file() (*moderngl_window.loaders.data.text.Loader class method*), 54
supports_file() (*moderngl_window.loaders.program.separate.Loader class method*), 47
supports_file() (*moderngl_window.loaders.program.single.Loader class method*), 45
supports_file() (*moderngl_window.loaders.scene.gltf2.Loader class method*), 50
supports_file() (*moderngl_window.loaders.scene.stl.Loader class method*), 52
supports_file() (*moderngl_window.loaders.scene.wavefront.Loader class method*), 49
supports_file() (*moderngl_window.loaders.texture.array.Loader*

class method), 48
supports_file() (*moderngl_window.loaders.texture.t2d.Loader class method*), 44
swap_buffers() (*moderngl_window.context.base.window.BaseWindow method*), 14
swap_buffers() (*moderngl_window.context.glfw.window.Window method*), 18
swap_buffers() (*moderngl_window.context.headless.window.Window method*), 22
swap_buffers() (*moderngl_window.context.pyglet.window.Window method*), 26
swap_buffers() (*moderngl_window.context.pyqt5.window.Window method*), 30
swap_buffers() (*moderngl_window.context.pyside2.window.Window method*), 34
swap_buffers() (*moderngl_window.context.sdl2.window.Window method*), 39

T

tess_control_shader (*moderngl_window.meta.program.ProgramDescription attribute*), 60
tess_evaluation_shader (*moderngl_window.meta.program.ProgramDescription attribute*), 60
texture (*moderngl_window.scene.MaterialTexture attribute*), 84
TEXTURE_DIRS (*moderngl_window.conf.Settings attribute*), 7
TEXTURE_FINDERS (*moderngl_window.conf.Settings attribute*), 7
TEXTURE_LOADERS (*moderngl_window.conf.Settings attribute*), 8
TextureDescription (*in module moderngl_window.meta.texture*), 57
Textures (*in module moderngl_window.resources.textures*), 71
time (*moderngl_window.timers.base.BaseTimer attribute*), 75
time (*moderngl_window.timers.clock.Timer attribute*), 76
Timer (*in module moderngl_window.timers.clock*), 75
title (*moderngl_window.context.base.window.BaseWindow attribute*), 15
title (*moderngl_window.context.base.window.WindowConfig attribute*), 12
title (*moderngl_window.context.glfw.window.Window attribute*), 19
title (*moderngl_window.context.headless.window.Window attribute*), 23
title (*moderngl_window.context.pyglet.window.Window attribute*), 27
title (*moderngl_window.context.pyqt5.window.Window attribute*), 31
title (*moderngl_window.context.pyside2.window.Window attribute*), 36
title (*moderngl_window.context.sdl2.window.Window attribute*), 39
to_dict() (*moderngl_window.conf.Settings method*), 6
tobytes() (*moderngl_window.opengl.projection.Projection3D method*), 67
toggle_pause() (*moderngl_window.timers.base.BaseTimer method*), 75
toggle_pause() (*moderngl_window.timers.clock.Timer method*), 76
transform() (*moderngl_window.opengl.vao.VAO method*), 69

U

update() (*moderngl_window.opengl.projection.Projection3D method*), 67
use() (*moderngl_window.context.base.window.BaseWindow method*), 14
use() (*moderngl_window.context.glfw.window.Window method*), 17
use() (*moderngl_window.context.headless.window.Window method*), 22
use() (*moderngl_window.context.pyglet.window.Window method*), 25
use() (*moderngl_window.context.pyqt5.window.Window method*), 30
use() (*moderngl_window.context.pyside2.window.Window method*), 34
use() (*moderngl_window.context.sdl2.window.Window method*), 38

V

VAO (*in module moderngl_window.opengl.vao*), 67
vertex_shader (*moderngl_window.meta.program.ProgramDescription attribute*), 60
viewport (*moderngl_window.context.base.window.BaseWindow attribute*), 15
viewport (*moderngl_window.context.glfw.window.Window attribute*), 20
viewport (*moderngl_window.context.headless.window.Window attribute*), 23

viewport (*moderngl_window.context.pyglet.window.Window attribute*), 28
viewport (*moderngl_window.context.pyqt5.window.Window attribute*), 32
viewport (*moderngl_window.context.pyside2.window.Window attribute*), 36
viewport (*moderngl_window.context.sdl2.window.Window attribute*), 40
vsync (*moderngl_window.context.base.window.BaseWindow attribute*), 16
vsync (*moderngl_window.context.glfw.window.Window attribute*), 20
vsync (*moderngl_window.context.headless.window.Window attribute*), 23
vsync (*moderngl_window.context.pyglet.window.Window attribute*), 28
vsync (*moderngl_window.context.pyqt5.window.Window attribute*), 32
vsync (*moderngl_window.context.pyside2.window.Window attribute*), 36
vsync (*moderngl_window.context.sdl2.window.Window attribute*), 40

W

width (*moderngl_window.context.base.window.BaseWindow attribute*), 15
width (*moderngl_window.context.glfw.window.Window attribute*), 19
width (*moderngl_window.context.headless.window.Window attribute*), 23
width (*moderngl_window.context.pyglet.window.Window attribute*), 27
width (*moderngl_window.context.pyqt5.window.Window attribute*), 32
width (*moderngl_window.context.pyside2.window.Window attribute*), 36
width (*moderngl_window.context.sdl2.window.Window attribute*), 39
WINDOW (*moderngl_window.conf.Settings attribute*), 6
window() (*in module moderngl_window*), 3
window_size (*moderngl_window.context.base.window.WindowConfig attribute*), 12
WindowConfig (*in module moderngl_window.context.base.window*), 8