
SIFT Documentation

Release 1.1.0a1

SIFT Developers

Dec 04, 2019

Contents

1	Design Overview	3
1.1	Main Window	3
1.2	Workspace	3
1.3	Document	3
1.4	Scene Graph	4
2	uwsift package	5
2.1	Subpackages	5
2.2	Submodules	11
2.3	uwsift.common module	11
2.4	uwsift.queue module	16
2.5	uwsift.version module	16
2.6	Module contents	16
3	How to Contribute	17
4	Indices and tables	19
	Python Module Index	21
	Index	23

SIFT, Satellite Information Familiarization Tool, is a GUI application for viewing and analyzing earth-observing satellite data. This documentation is meant for developers of SIFT or those interested in the low-level details (programming interfaces, public APIs, overall designs, etc). For general information on the use of SIFT, official releases, and installation instructions of those releases see the SIFT [home page](#). Note that the library and python package for the SIFT project are named `uwsift`.

For source code and other information now provided by the home page see the [GitHub repository](#). Some of the original design documentation of SIFT can be found on the repository's [wiki](#). This site's documentation will slowly replace the low-level documents on the wiki.

SIFT’s software design revolves around a few key components:

- Main Window (GUI)
- Workspace
- Document
- Scene Graph

Each of these components is described in the sections below. Other components involved in accomplishing SIFT’s feature.

1.1 Main Window

Currently the main window for the SIFT GUI connects all other components and helper objects. This may change in future versions of SIFT. By defining things this way the main window has access to UI events and can connect them to the other SIFT components that need to use them like those listed below.

1.2 Workspace

The Workspace acts as the manager of on-disk or remote data. It will handle importing requested datasets, caching binary data, and storing dataset metadata in a database for easier querying. Since the Workspace manages all of the cached data it is also the best place that SIFT components will go for variations on the data (different resolutions, data within a polygon, etc).

1.3 Document

The Document acts as the “model” of the Model-View-Controller design of SIFT. Through the Document a developer can get access to individual layer objects containing metadata, layer order, and animation order. In the future as other

features are added to SIFT the Document may provide user profile or configuration information.

1.4 Scene Graph

The Scene Graph wraps all map canvas elements visual elements. It handles connecting all mouse events from the map canvas like pan and zoom events. The majority of this components responsibility is to map SIFT functions to the python `vispy` library.

2.1 Subpackages

2.1.1 uwsift.control package

Submodules

uwsift.control.doc_ws_as_timeline_scene module

uwsift.control.file_behaviors module

uwsift.control.layer_tree module

uwsift.control.rgb_behaviors module

Module contents

.py

PURPOSE

REFERENCES

REQUIRES

author R.K.Garcia <rayg@ssec.wisc.edu>

copyright 2014 by University of Wisconsin Regents, see AUTHORS for more details

license GPLv3, see LICENSE for more details

`uwsift.control.main()`

2.1.2 uwsift.model package

Submodules

uwsift.model.composite_recipes module

uwsift.model.document module

uwsift.model.layer module

layer.py

PURPOSE Document Layer & LayerDoc

REFERENCES

REQUIRES

author R.K.Garcia <rayg@ssec.wisc.edu>

copyright 2014 by University of Wisconsin Regents, see AUTHORS for more details

license GPLv3, see LICENSE for more details

class `uwsift.model.layer.DocAlgebraicLayer` (*doc, info, *args, **kwargs*)

Bases: `uwsift.model.layer.DocCompositeLayer`

A value field derived from other value fields algebraically

class `uwsift.model.layer.DocBasicLayer` (*doc, info, *args, **kwargs*)

Bases: `uwsift.model.layer.DocLayer`

A layer consistent of a simple scalar floating point value field, which can have color maps applied

class `uwsift.model.layer.DocCompositeLayer` (*doc, info, *args, **kwargs*)

Bases: `uwsift.model.layer.DocLayer`

A layer which combines other layers, be they basic or composite themselves

class `uwsift.model.layer.DocLayer` (*doc, info, *args, **kwargs*)

Bases: `collections.ChainMap`

Container for layer metadata

Use dictionary-like access for metadata information.

children

return dictionary of weakrefs to layers we require in order to function :return:

dataset_name

default_display_name

display_name

instrument

is_flat_field

return whether the layer can be represented as a flat numerical field or not (RGB layers cannot) :return:
bool

is_valid

invalid layers cannot be displayed (are never visible) valid layers may or may not be visible visibility is managed by the scenegraph validity is managed by the document example of an invalid layer: an RGB or algebraic layer that's insufficiently specified to actually display, be it through lack of data or lack of projection information however, an invalid layer may still be configurable in order to allow it to become valid and then visible :return: bool

kind

which kind of layer it is - RGB, Algebraic, etc. This can also be tested by the class of the layer typically. We may deprecate this eventually? :return:

parent

parent layer, if any :return:

platform**product_family_key**

Unique key for this layer and its group of siblings

sched_time**uuid**

UUID of the layer, which for basic layers is likely to be the UUID of the dataset in the workspace. :return:

class `uwsift.model.layer.DocRGBLayer` (*doc, recipe, info, *args, **kwargs*)

Bases: `uwsift.model.layer.DocCompositeLayer`

a**b****central_wavelength**

dep_info (*key, include_alpha=False*)

g**has_deps****instrument****is_flat_field**

return whether the layer can be represented as a flat numerical field or not (RGB layers cannot) :return: bool

is_valid

invalid layers cannot be displayed (are never visible) valid layers may or may not be visible visibility is managed by the scenegraph validity is managed by the document example of an invalid layer: an RGB or algebraic layer that's insufficiently specified to actually display, be it through lack of data or lack of projection information however, an invalid layer may still be configurable in order to allow it to become valid and then visible :return: bool

platform

product_family_keys (*include_alpha=False*)

r**recipe_layers_match****scene****sched_time****shared_origin**

`shared_projections`

`update_metadata_from_dependencies()`

recalculate origin and dimension information based on new upstream :return:

class `uwsift.model.layer.Mixing`

Bases: `enum.Enum`

An enumeration.

ADD = 2

NORMAL = 1

SUBTRACT = 3

UNKNOWN = 0

`uwsift.model.layer.main()`

`uwsift.model.shapes` module

Module contents

2.1.3 `uwsift.ui` package

Submodules

`uwsift.ui.change_colormap_dialog_ui` module

`uwsift.ui.config_rgb_layer_ui` module

`uwsift.ui.create_algebraic_dialog_ui` module

`uwsift.ui.custom_widgets` module

`uwsift.ui.export_image_dialog_ui` module

`uwsift.ui.open_cache_dialog_ui` module

`uwsift.ui.open_file_wizard_ui` module

`uwsift.ui.pov_main_ui` module

Module contents

`.py`

PURPOSE

REFERENCES

REQUIRES

author R.K.Garcia <rayg@ssec.wisc.edu>

copyright 2014 by University of Wisconsin Regents, see AUTHORS for more details

license GPLv3, see LICENSE for more details

`uwsift.ui.main()`

2.1.4 uwsift.util package

Submodules

uwsift.util.default_paths module

Utility functions and classes

Directory Constants:

- `WORKSPACE_DB_DIR`: Default location to place the workspace metadata database(s)
- `WORKSPACE_CACHE_DIR`: Default workspace cache directory Where any raster data may be cached.
- `DOCUMENT_SETTINGS_DIR`: Default document user settings/profiles directory
-

Module contents

`uwsift.util.check_grib_definition_dir()`

`uwsift.util.check_imageio_deps()`

`uwsift.util.get_package_data_dir()`

Return location of the package 'data' directory.

When frozen the data directory is placed in 'sift_data' of the root package directory.

`uwsift.util.prefix_share_dir()`

2.1.5 uwsift.view package

Subpackages

uwsift.view.timeline package

Submodules

uwsift.view.timeline.common module

uwsift.view.timeline.items module

uwsift.view.timeline.scene module

Module contents

Submodules

uwsift.view.cameras module

`uwsift.view.colormap` module

`uwsift.view.colormap_dialogs` module

`uwsift.view.colormap_editor` module

`uwsift.view.create_algebraic` module

`uwsift.view.export_image` module

`uwsift.view.layer_details` module

`uwsift.view.open_file_wizard` module

`uwsift.view.probes` module

`uwsift.view.rgb_config` module

`uwsift.view.scene_graph` module

`uwsift.view.texture_atlas` module

`uwsift.view.tile_calculator` module

`uwsift.view.transform` module

`uwsift.view.visuals` module

Module contents

View owns the GPU and the user interface. Delegates from the Model and elsewhere provide it information it needs to react rapidly.

2.1.6 `uwsift.workspace` package

Submodules

`uwsift.workspace.collector` module

`uwsift.workspace.goesr_pug` module

`uwsift.workspace.guidebook` module

`uwsift.workspace.importer` module

`uwsift.workspace.matrix` module

`uwsift.workspace.metadatabase` module

`uwsift.workspace.workspace` module

Module contents

2.2 Submodules

2.3 uwsift.common module

2.3.1 .py

PURPOSE Support calculations, namedtuples and constants used throughout the library and application.

REFERENCES

REQUIRES numpy numba

author R.K.Garcia <rayg@ssec.wisc.edu>

copyright 2015 by University of Wisconsin Regents, see AUTHORS for more details

license GPLv3, see LICENSE for more details

class `uwsift.common.Box` (*bottom, left, top, right*)

Bases: `tuple`

bottom

Alias for field number 0

left

Alias for field number 1

right

Alias for field number 3

top

Alias for field number 2

class `uwsift.common.CompositeType`

Bases: `enum.Enum`

Type of non-luminance image layers.

ARITHMETIC = 2

RGB = 1

class `uwsift.common.Coordinate` (*deg_north, deg_east*)

Bases: `tuple`

deg_east

Alias for field number 1

deg_north

Alias for field number 0

class `uwsift.common.Flags`

Bases: `set`

A set of enumerated Flags which may ultimately be represented as a bitfield, but observing set interface

class `uwsift.common.Info`

Bases: `enum.Enum`

Standard keys for info dictionaries Note: some fields correspond to database fields in workspace.metadatabase !

```
CATEGORY = 'category'
CELL_HEIGHT = 'cell_height'
CELL_WIDTH = 'cell_width'
CENTRAL_WAVELENGTH = 'nominal_wavelength'
CLIM = 'clim'
COLORMAP = 'colormap'
DATASET_NAME = 'dataset_name'
DISPLAY_FAMILY = 'display_family'
DISPLAY_NAME = 'display_name'
DISPLAY_TIME = 'display_time'
FAMILY = 'family'
INSTRUMENT = 'instrument'
KIND = 'kind'
LONG_NAME = 'long_name'
OBS_DURATION = 'obsduration'
OBS_TIME = 'obstime'
ORIGIN_X = 'origin_x'
ORIGIN_Y = 'origin_y'
PATHNAME = 'path'
PLATFORM = 'platform'
PROJ = 'proj4'
SCENE = 'scene'
SCHED_TIME = 'timeline'
SERIAL = 'serial'
SHAPE = 'shape'
SHORT_NAME = 'short_name'
STANDARD_NAME = 'standard_name'
UNITS = 'units'
UNIT_CONVERSION = 'unit_conversion'
UNKNOWN = '???'
UUID = 'uuid'
VALID_RANGE = 'valid_range'
```

```
class uwsift.common.Instrument
```

```
    Bases: enum.Enum
```

```
    An enumeration.
```



```

ABI = 'ABI'
AHI = 'AHI'
AMI = 'AMI'
GFS = 'GFS'
NAM = 'NAM'
SEVIRI = 'SEVIRI'
UNKNOWN = '???'

```

```
class uwsift.common.Kind
```

```
Bases: enum.Enum
```

Kind of entities we're working with.

```

COMPOSITE = 1
CONTOUR = 6
IMAGE = 1
OUTLINE = 2
RGB = 4
SHAPE = 3
UNKNOWN = 0

```

```
class uwsift.common.Platform
```

```
Bases: enum.Enum
```

An enumeration.

```

GOES_16 = 'G16'
GOES_17 = 'G17'
GOES_18 = 'G18'
HIMAWARI_8 = 'Himawari-8'
HIMAWARI_9 = 'Himawari-9'
MSG10 = 'Meteosat-10'
MSG11 = 'Meteosat-11'
MSG8 = 'Meteosat-8'
MSG9 = 'Meteosat-9'
NWP = 'NWP'
UNKNOWN = '???'

```

```
class uwsift.common.Point(y, x)
```

```
Bases: tuple
```

```
x
```

Alias for field number 1

```
y
```

Alias for field number 0

class `uwsift.common.Presentation`

Bases: `tuple`

Presentation information for a layer.

`z_order` comes from the layerset

a_order

Alias for field number 3

climits

Alias for field number 5

colormap

Alias for field number 4

gamma

Alias for field number 6

kind

Alias for field number 1

mixing

Alias for field number 7

uuid

Alias for field number 0

visible

Alias for field number 2

class `uwsift.common.Resolution`

Bases: `tuple`

Pixel resolution (km per pixel).

dx

Alias for field number 1

dy

Alias for field number 0

class `uwsift.common.Span(s, d)`

Bases: `tuple`

d

Alias for field number 1

e

static from_s_e (*s: datetime.datetime, e: datetime.datetime*)

is_instantaneous

s

Alias for field number 0

class `uwsift.common.State`

Bases: `enum.Enum`

State for products in document.

ARRIVING = 2

ATTACHED = 5

```

CACHED = 3
DANGLING = -1
ONSCREEN = 6
POTENTIAL = 1
UNKNOWN = 0

class uwsift.common.Tool
    Bases: enum.Enum
    Names for cursor tools.
    PAN_ZOOM = 'pan_zoom'
    POINT_PROBE = 'point_probe'
    REGION_PROBE = 'region_probe'

class uwsift.common.ViewBox
    Bases: tuple
    Combination of Box + Resolution.
    bottom
        Alias for field number 0
    dx
        Alias for field number 5
    dy
        Alias for field number 4
    left
        Alias for field number 1
    right
        Alias for field number 3
    top
        Alias for field number 2

class uwsift.common.ZList (zmax: int = None, content: Iterable[Any] = None)
    Bases: collections.abc.MutableSequence, typing.Generic
    List indexed from high Z to low Z For z-ordered tracks, we want to have - contiguous Z order values from high to low (negative) - elements assigned negative stay negative, likewise with positive (negative Z implies inactive non-document track) - no Z value is repeated - insertions are correctly handled - append also works, by default arriving as most-negative z-order - assignment off either end gets snapped to the contiguous next value
    append (val, start_negative: bool = False, not_if_present: bool = False)
        S.append(value) – append value to the end of the sequence
    bottom_z
    index (value [, start [, stop ]]) → integer – return first index of value.
        Raises ValueError if the value is not present.
        Supporting start and stop arguments is optional, but recommended.
    insert (z: int, val)
        insert a value such that it lands at index z as needed: displace any content with z>=0 upward displace any content with z<0 downward

```

items () → Iterable[Tuple[int, Any]]

keys ()

merge_subst (*new_values: Iterable[Tuple[int, Any]]*)

batch merge of substitutions raises IndexError if any of them is outside current range

min_max

move (*to_z: int, val*)

prepend (*val*)

to_dict (*inverse=False*) → dict

top_z

values ()

`uwsift.common.get_font_size` (*pref_size*)

Get a font size that looks good on this platform.

This is a HACK and can be replaced by PyQt5 font handling after migration.

2.4 uwsift.queue module

2.5 uwsift.version module

2.6 Module contents

CHAPTER 3

How to Contribute

Information on contributing to SIFT can be found on our [GitHub Wiki](<https://github.com/ssec/sift/wiki/Contributing>).

CHAPTER 4

Indices and tables

- `genindex`
- `modindex`
- `search`

U

uwsift, 16
uwsift.common, 11
uwsift.control, 5
uwsift.model.layer, 6
uwsift.ui, 8
uwsift.util, 9
uwsift.util.default_paths, 9
uwsift.version, 16
uwsift.view, 10
uwsift.view.timeline, 9

A

a (*uwsift.model.layer.DocRGBLayer attribute*), 7
 a_order (*uwsift.common.Presentation attribute*), 14
 ABI (*uwsift.common.Instrument attribute*), 12
 ADD (*uwsift.model.layer.Mixing attribute*), 8
 AHI (*uwsift.common.Instrument attribute*), 13
 AMI (*uwsift.common.Instrument attribute*), 13
 append() (*uwsift.common.ZList method*), 15
 ARITHMETIC (*uwsift.common.CompositeType attribute*), 11
 ARRIVING (*uwsift.common.State attribute*), 14
 ATTACHED (*uwsift.common.State attribute*), 14

B

b (*uwsift.model.layer.DocRGBLayer attribute*), 7
 bottom (*uwsift.common.Box attribute*), 11
 bottom (*uwsift.common.ViewBox attribute*), 15
 bottom_z (*uwsift.common.ZList attribute*), 15
 Box (*class in uwsift.common*), 11

C

CACHED (*uwsift.common.State attribute*), 14
 CATEGORY (*uwsift.common.Info attribute*), 12
 CELL_HEIGHT (*uwsift.common.Info attribute*), 12
 CELL_WIDTH (*uwsift.common.Info attribute*), 12
 CENTRAL_WAVELENGTH (*uwsift.common.Info attribute*), 12
 central_wavelength (*uwsift.model.layer.DocRGBLayer attribute*), 7
 check_grib_definition_dir() (*in module uwsift.util*), 9
 check_imageio_deps() (*in module uwsift.util*), 9
 children (*uwsift.model.layer.DocLayer attribute*), 6
 CLIM (*uwsift.common.Info attribute*), 12
 climits (*uwsift.common.Presentation attribute*), 14
 COLORMAP (*uwsift.common.Info attribute*), 12
 colormap (*uwsift.common.Presentation attribute*), 14
 COMPOSITE (*uwsift.common.Kind attribute*), 13
 CompositeType (*class in uwsift.common*), 11

CONTOUR (*uwsift.common.Kind attribute*), 13
 Coordinate (*class in uwsift.common*), 11

D

d (*uwsift.common.Span attribute*), 14
 DANGLING (*uwsift.common.State attribute*), 15
 DATASET_NAME (*uwsift.common.Info attribute*), 12
 dataset_name (*uwsift.model.layer.DocLayer attribute*), 6
 default_display_name (*uwsift.model.layer.DocLayer attribute*), 6
 deg_east (*uwsift.common.Coordinate attribute*), 11
 deg_north (*uwsift.common.Coordinate attribute*), 11
 dep_info() (*uwsift.model.layer.DocRGBLayer method*), 7
 DISPLAY_FAMILY (*uwsift.common.Info attribute*), 12
 DISPLAY_NAME (*uwsift.common.Info attribute*), 12
 display_name (*uwsift.model.layer.DocLayer attribute*), 6
 DISPLAY_TIME (*uwsift.common.Info attribute*), 12
 DocAlgebraicLayer (*class in uwsift.model.layer*), 6
 DocBasicLayer (*class in uwsift.model.layer*), 6
 DocCompositeLayer (*class in uwsift.model.layer*), 6
 DocLayer (*class in uwsift.model.layer*), 6
 DocRGBLayer (*class in uwsift.model.layer*), 7
 dx (*uwsift.common.Resolution attribute*), 14
 dx (*uwsift.common.ViewBox attribute*), 15
 dy (*uwsift.common.Resolution attribute*), 14
 dy (*uwsift.common.ViewBox attribute*), 15

E

e (*uwsift.common.Span attribute*), 14

F

FAMILY (*uwsift.common.Info attribute*), 12
 Flags (*class in uwsift.common*), 11
 from_s_e() (*uwsift.common.Span static method*), 14

G

`g` (*uwsift.model.layer.DocRGBLayer attribute*), 7
`gamma` (*uwsift.common.Presentation attribute*), 14
`get_font_size()` (*in module uwsift.common*), 16
`get_package_data_dir()` (*in module uwsift.util*), 9
`GFS` (*uwsift.common.Instrument attribute*), 13
`GOES_16` (*uwsift.common.Platform attribute*), 13
`GOES_17` (*uwsift.common.Platform attribute*), 13
`GOES_18` (*uwsift.common.Platform attribute*), 13

H

`has_deps` (*uwsift.model.layer.DocRGBLayer attribute*), 7
`HIMAWARI_8` (*uwsift.common.Platform attribute*), 13
`HIMAWARI_9` (*uwsift.common.Platform attribute*), 13

I

`IMAGE` (*uwsift.common.Kind attribute*), 13
`index()` (*uwsift.common.ZList method*), 15
`Info` (*class in uwsift.common*), 11
`insert()` (*uwsift.common.ZList method*), 15
`Instrument` (*class in uwsift.common*), 12
`INSTRUMENT` (*uwsift.common.Info attribute*), 12
`instrument` (*uwsift.model.layer.DocLayer attribute*), 6
`instrument` (*uwsift.model.layer.DocRGBLayer attribute*), 7
`is_flat_field` (*uwsift.model.layer.DocLayer attribute*), 6
`is_flat_field` (*uwsift.model.layer.DocRGBLayer attribute*), 7
`is_instantaneous` (*uwsift.common.Span attribute*), 14
`is_valid` (*uwsift.model.layer.DocLayer attribute*), 6
`is_valid` (*uwsift.model.layer.DocRGBLayer attribute*), 7
`items()` (*uwsift.common.ZList method*), 15

K

`keys()` (*uwsift.common.ZList method*), 16
`Kind` (*class in uwsift.common*), 13
`KIND` (*uwsift.common.Info attribute*), 12
`kind` (*uwsift.common.Presentation attribute*), 14
`kind` (*uwsift.model.layer.DocLayer attribute*), 7

L

`left` (*uwsift.common.Box attribute*), 11
`left` (*uwsift.common.ViewBox attribute*), 15
`LONG_NAME` (*uwsift.common.Info attribute*), 12

M

`main()` (*in module uwsift.control*), 5
`main()` (*in module uwsift.model.layer*), 8

`main()` (*in module uwsift.ui*), 9
`merge_subst()` (*uwsift.common.ZList method*), 16
`min_max` (*uwsift.common.ZList attribute*), 16
`Mixing` (*class in uwsift.model.layer*), 8
`mixing` (*uwsift.common.Presentation attribute*), 14
`move()` (*uwsift.common.ZList method*), 16
`MSG10` (*uwsift.common.Platform attribute*), 13
`MSG11` (*uwsift.common.Platform attribute*), 13
`MSG8` (*uwsift.common.Platform attribute*), 13
`MSG9` (*uwsift.common.Platform attribute*), 13

N

`NAM` (*uwsift.common.Instrument attribute*), 13
`NORMAL` (*uwsift.model.layer.Mixing attribute*), 8
`NWP` (*uwsift.common.Platform attribute*), 13

O

`OBS_DURATION` (*uwsift.common.Info attribute*), 12
`OBS_TIME` (*uwsift.common.Info attribute*), 12
`ONSCREEN` (*uwsift.common.State attribute*), 15
`ORIGIN_X` (*uwsift.common.Info attribute*), 12
`ORIGIN_Y` (*uwsift.common.Info attribute*), 12
`OUTLINE` (*uwsift.common.Kind attribute*), 13

P

`PAN_ZOOM` (*uwsift.common.Tool attribute*), 15
`parent` (*uwsift.model.layer.DocLayer attribute*), 7
`PATHNAME` (*uwsift.common.Info attribute*), 12
`Platform` (*class in uwsift.common*), 13
`PLATFORM` (*uwsift.common.Info attribute*), 12
`platform` (*uwsift.model.layer.DocLayer attribute*), 7
`platform` (*uwsift.model.layer.DocRGBLayer attribute*), 7
`Point` (*class in uwsift.common*), 13
`POINT_PROBE` (*uwsift.common.Tool attribute*), 15
`POTENTIAL` (*uwsift.common.State attribute*), 15
`prefix_share_dir()` (*in module uwsift.util*), 9
`prepend()` (*uwsift.common.ZList method*), 16
`Presentation` (*class in uwsift.common*), 13
`product_family_key` (*uwsift.model.layer.DocLayer attribute*), 7
`product_family_keys()` (*uwsift.model.layer.DocRGBLayer method*), 7
`PROJ` (*uwsift.common.Info attribute*), 12

R

`r` (*uwsift.model.layer.DocRGBLayer attribute*), 7
`recipe_layers_match` (*uwsift.model.layer.DocRGBLayer attribute*), 7
`REGION_PROBE` (*uwsift.common.Tool attribute*), 15
`Resolution` (*class in uwsift.common*), 14
`RGB` (*uwsift.common.CompositeType attribute*), 11
`RGB` (*uwsift.common.Kind attribute*), 13

right (*uwsift.common.Box* attribute), 11
 right (*uwsift.common.ViewBox* attribute), 15

S

s (*uwsift.common.Span* attribute), 14
 SCENE (*uwsift.common.Info* attribute), 12
 scene (*uwsift.model.layer.DocRGBLayer* attribute), 7
 SCHED_TIME (*uwsift.common.Info* attribute), 12
 sched_time (*uwsift.model.layer.DocLayer* attribute), 7
 sched_time (*uwsift.model.layer.DocRGBLayer* attribute), 7
 SERIAL (*uwsift.common.Info* attribute), 12
 SEVIRI (*uwsift.common.Instrument* attribute), 13
 SHAPE (*uwsift.common.Info* attribute), 12
 SHAPE (*uwsift.common.Kind* attribute), 13
 shared_origin (*uwsift.model.layer.DocRGBLayer* attribute), 7
 shared_projections (*uwsift.model.layer.DocRGBLayer* attribute), 7
 SHORT_NAME (*uwsift.common.Info* attribute), 12
 Span (*class in uwsift.common*), 14
 STANDARD_NAME (*uwsift.common.Info* attribute), 12
 State (*class in uwsift.common*), 14
 SUBTRACT (*uwsift.model.layer.Mixing* attribute), 8

T

to_dict () (*uwsift.common.ZList* method), 16
 Tool (*class in uwsift.common*), 15
 top (*uwsift.common.Box* attribute), 11
 top (*uwsift.common.ViewBox* attribute), 15
 top_z (*uwsift.common.ZList* attribute), 16

U

UNIT_CONVERSION (*uwsift.common.Info* attribute), 12
 UNITS (*uwsift.common.Info* attribute), 12
 UNKNOWN (*uwsift.common.Info* attribute), 12
 UNKNOWN (*uwsift.common.Instrument* attribute), 13
 UNKNOWN (*uwsift.common.Kind* attribute), 13
 UNKNOWN (*uwsift.common.Platform* attribute), 13
 UNKNOWN (*uwsift.common.State* attribute), 15
 UNKNOWN (*uwsift.model.layer.Mixing* attribute), 8
 update_metadata_from_dependencies () (*uwsift.model.layer.DocRGBLayer* method), 8
 UUID (*uwsift.common.Info* attribute), 12
 uuid (*uwsift.common.Presentation* attribute), 14
 uuid (*uwsift.model.layer.DocLayer* attribute), 7
 uwsift (*module*), 16
 uwsift.common (*module*), 11
 uwsift.control (*module*), 5
 uwsift.model.layer (*module*), 6
 uwsift.ui (*module*), 8
 uwsift.util (*module*), 9
 uwsift.util.default_paths (*module*), 9
 uwsift.version (*module*), 16

uwsift.view (*module*), 10
 uwsift.view.timeline (*module*), 9

V

VALID_RANGE (*uwsift.common.Info* attribute), 12
 values () (*uwsift.common.ZList* method), 16
 VBox (*class in uwsift.common*), 15
 visible (*uwsift.common.Presentation* attribute), 14

X

x (*uwsift.common.Point* attribute), 13

Y

y (*uwsift.common.Point* attribute), 13

Z

ZList (*class in uwsift.common*), 15