

---

# **aiidalab-widgets-base**

**unknown**

**May 07, 2021**



# CONTENTS

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Contribute to AWB</b>	<b>5</b>
<b>3</b>	<b>List of widgets</b>	<b>7</b>
3.1	Dealing with codes . . . . .	7
3.2	Dealing with computers . . . . .	7
3.3	Dealing with external databases . . . . .	7
3.4	Export AiiDA database . . . . .	7
3.5	Dealing with AiiDA nodes . . . . .	7
3.6	Dealing with AiiDA processes . . . . .	8
3.7	Dealing with structures . . . . .	8
<b>4</b>	<b>aiidalab_widgets_base package</b>	<b>9</b>
4.1	Subpackages . . . . .	9
4.2	Submodules . . . . .	10
4.3	aiidalab_widgets_base.codes module . . . . .	10
4.4	aiidalab_widgets_base.computers module . . . . .	11
4.5	aiidalab_widgets_base.databases module . . . . .	15
4.6	aiidalab_widgets_base.dicts module . . . . .	18
4.7	aiidalab_widgets_base.export module . . . . .	18
4.8	aiidalab_widgets_base.misc module . . . . .	18
4.9	aiidalab_widgets_base.nodes module . . . . .	19
4.10	aiidalab_widgets_base.process module . . . . .	21
4.11	aiidalab_widgets_base.structures module . . . . .	26
4.12	aiidalab_widgets_base.structures_multi module . . . . .	31
4.13	aiidalab_widgets_base.viewers module . . . . .	31
4.14	aiidalab_widgets_base.wizard module . . . . .	34
4.15	Module contents . . . . .	37
<b>5</b>	<b>Indices and tables</b>	<b>59</b>
5.1	How to cite . . . . .	59
5.2	Acknowledgements . . . . .	59
	<b>Python Module Index</b>	<b>61</b>
	<b>Index</b>	<b>63</b>



AiiDALab Widgets Base (AWB) is a set of handy widgets that are used in AiiDALab to create applications.



## INTRODUCTION

AiiDALab Widgets Base (AWB) is a library of widgets that is used in AiiDALab when building applications. The main goal of most of the widgets is to provide a user friendly interface to interact with [AiiDA](#) objects. Some other widgets are more of a general purpose with a goal to help you building powerful applications.





## CONTRIBUTE TO AWB

If you would like to contribute to the project you can do it in multiple ways:

- [Create issues](#) in case you found a bug or you need a feature that is not yet implemented.
- Contribute new widgets.
- Help us with fixing the [existing issues](#).
- Extend the documentation.



## LIST OF WIDGETS

Here we collect the information about all the widgets available in AWB and provide a simple example of how to use them.

### 3.1 Dealing with codes

[issue: #185]

### 3.2 Dealing with computers

[issue: #186 ]

### 3.3 Dealing with external databases

[issue: #187]

### 3.4 Export AiiDA database

[issue: #188]

### 3.5 Dealing with AiiDA nodes

[issue: #189]

## 3.6 Dealing with AiiDA processes

[issue: #190]

## 3.7 Dealing with structures

[issue: #191]

## AIIDALAB\_WIDGETS\_BASE PACKAGE

### 4.1 Subpackages

#### 4.1.1 aiidalab\_widgets\_base.data package

##### Module contents

Useful functions that provide access to some data.

```
class aiidalab_widgets_base.data.LigandSelectorWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_selection.Dropdown

    Class to select ligands that are returned as Atoms object

    __init__ (value=0, description='Select ligand', **kwargs)
        Public constructor

    __module__ = 'aiidalab_widgets_base.data'

    __trait_default_generators = {}

    property anchoring_atom
        Return anchoring atom chemical symbol.

    rotate (align_to=(0, 0, 1), remove_anchor=False)
        Rotate group in such a way that vector which was (-1,-1,-1) is alligned with align_to.
```

#### 4.1.2 aiidalab\_widgets\_base.utils package

##### Module contents

Some utility functions used across the repository.

```
aiidalab_widgets_base.utils.find_ranges (iterable)
    Yield range of consecutive numbers.

aiidalab_widgets_base.utils.get_ase_from_file (fname, format=None)
    Get ASE structure object.

aiidalab_widgets_base.utils.list_to_string_range (lst, shift=1)
    Converts a list like [0, 2, 3, 4] into a string like '1 3..5'.

    Shift used when e.g. for a user interface numbering starts from 1 not from 0

aiidalab_widgets_base.utils.predefine_settings (obj, **kwargs)
    Specify some pre-defined settings.
```

`aiidalab_widgets_base.utils.string_range_to_list` (*strng, shift=- 1*)  
Converts a string like '1 3..5' into a list like [0, 2, 3, 4].

Shift used when e.g. for a user interface numbering starts from 1 not from 0

`aiidalab_widgets_base.utils.valid_arguments` (*arguments, valid\_args*)  
Check whether provided arguments are valid.

## 4.2 Submodules

### 4.3 aiidalab\_widgets\_base.codes module

Module to manage AiiDA codes.

**class** `aiidalab_widgets_base.codes.AiiDACodeSetup` (*\*\*kwargs*)

Bases: `ipywidgets.widgets.widget_box.VBox`

Class that allows to setup AiiDA code

`__init__` (*\*\*kwargs*)

Public constructor

`__module__` = 'aiidalab\_widgets\_base.codes'

`_setup_code` (*\_=None*)

Setup an AiiDA code.

`_trait_default_generators` = {}

**append\_text**

A trait for unicode strings.

**computer**

A trait type representing a Union type.

**description**

A trait for unicode strings.

**exists** ()

Returns True if the code exists, returns False otherwise.

**input\_plugin**

A trait for unicode strings.

**label**

A trait for unicode strings.

**prepend\_text**

A trait for unicode strings.

**remote\_abs\_path**

A trait for unicode strings.

**class** `aiidalab_widgets_base.codes.CodeDropdown` (*\*\*kwargs*)

Bases: `ipywidgets.widgets.widget_box.VBox`

Code selection widget. Attributes:

`selected_code`(Unicode or Code): Trait that points to the selected Code instance. It can be set either to an AiiDA Code instance or to a code label (will automatically be replaced by the corresponding Code instance). It is linked to the 'value' trait of the *self.dropdown* widget.

`codes(Dict)`: Trait that contains a dictionary (label => Code instance) for all codes found in the AiiDA database for the selected plugin. It is linked to the 'options' trait of the `self.dropdown` widget.

`allow_hidden_codes(Bool)`: Trait that defines whether to show hidden codes or not.

`allow_disabled_computers(Bool)`: Trait that defines whether to show codes on disabled computers.

`__init__ (input_plugin, description='Select code:', path_to_root='..', **kwargs)`

Dropdown for Codes for one input plugin.

`input_plugin (str)`: Input plugin of codes to show.

`description (str)`: Description to display before the dropdown.

`__module__ = 'aiidalab_widgets_base.codes'`

`static _full_code_label (code)`

`_get_codes ()`

Query the list of available codes.

`_trait_default_generators = {}`

`_validate_selected_code`

`allow_disabled_computers`

A boolean (True, False) trait.

`allow_hidden_codes`

A boolean (True, False) trait.

`codes`

An instance of a Python dict.

One or more traits can be passed to the constructor to validate the keys and/or values of the dict. If you need more detailed validation, you may use a custom validator method.

Changed in version 5.0: Added `key_trait` for validating dict keys.

Changed in version 5.0: Deprecated ambiguous trait, traits args in favor of `value_trait`, `per_key_traits`.

`refresh (_=None)`

Refresh available codes.

The job of this function is to look in AiiDA database, find available codes and put them in the dropdown attribute.

`selected_code`

A trait type representing a Union type.

## 4.4 aiidalab\_widgets\_base.computers module

All functionality needed to setup a computer.

`class aiidalab_widgets_base.computers.AiidaComputerSetup (**kwargs)`

Bases: `ipywidgets.widgets.widget_box.VBox`

Inform AiiDA about a computer.

`__init__ (**kwargs)`

Public constructor

`__module__ = 'aiidalab_widgets_base.computers'`

**`_configure_computer()`**  
Create AuthInfo.

**`_on_setup_computer(_=None)`**  
When setup computer button is pressed.

**`_trait_default_generators = {}`**

**`_validate_mpiprocs_per_machine`**

**`_validate_safe_interval`**

**`append_text`**  
A trait for unicode strings.

**`description`**  
A trait for unicode strings.

**`hostname`**  
A trait for unicode strings.

**`label`**  
A trait for unicode strings.

**`mpiprocs_per_machine`**  
A trait type representing a Union type.

**`mpirun_command`**  
A trait for unicode strings.

**`prepend_text`**  
A trait for unicode strings.

**`safe_interval`**  
A trait type representing a Union type.

**`scheduler`**  
A trait for unicode strings.

**`test(_=None)`**

**`transport`**  
A trait for unicode strings.

**`work_dir`**  
A trait for unicode strings.

**`class aiidalab_widgets_base.computers.ComputerDropdown(**kwargs)`**

Bases: `ipywidgets.widgets.widget_box.VBox`

Widget to select a configured computer.

**Attributes:**

**`selected_computer(Unicode or Computer):`** Trait that points to the selected Computer instance.

It can be set either to an AiiDA Computer instance or to a computer label (will automatically be replaced by the corresponding Computer instance). It is linked to the 'value' trait of `self._dropdown` widget.

`computers(Dict):` Trait that contains a dictionary (label => Computer instance) for all computers found in the AiiDA database. It is linked to the 'options' trait of `self._dropdown` widget.

`allow_select_disabled(Bool):` Trait that defines whether to show disabled computers.



---

```

__init__ (description='Select computer:', path_to_root='../', **kwargs)
    Dropdown for configured AiiDA Computers.

    description (str): Text to display before dropdown.

    path_to_root (str): Path to the app's root folder.

__module__ = 'aiidalab_widgets_base.computers'

_get_computers ()
    Get the list of available computers.

_trait_default_generators = {}

_validate_selected_computer

allow_select_disabled
    A boolean (True, False) trait.

computers
    An instance of a Python dict.

    One or more traits can be passed to the constructor to validate the keys and/or values of the dict. If you
    need more detailed validation, you may use a custom validator method.

    Changed in version 5.0: Added key_trait for validating dict keys.

    Changed in version 5.0: Deprecated ambiguous trait, traits args in favor of value_trait,
    per_key_traits.

refresh (_=None)
    Refresh the list of configured computers.

selected_computer
    A trait type representing a Union type.

class aiidalab_widgets_base.computers.SshComputerSetup (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    Setup password-free access to a computer.

    __init__ (**kwargs)
        Public constructor

    __module__ = 'aiidalab_widgets_base.computers'

    property __password
        Returning the password and immediately destroying it

    property __private_key
        unwrap private key file and setting filename and file content

    property __proxy_password
        Returning the password and immediately destroying it

    static _add_private_key (private_key_fname, private_key_content)
        param private_key_fname: string param private_key_content: bytes

    _configure_proxy (password, proxy_password)
        Configure proxy server.

    _make_host_known (hostname, proxycmd=None)
        Add host information into known_hosts file.

    _observe_proxy_hostname

```

**`_observe_proxy_username`**

**`_on_setup_ssh`** (*mode, change*)  
ATTENTION: modifying the order of operations in this function can lead to unexpected problems

**`static _send_pubkey`** (*hostname, username, password, proxycmd=""*)  
Send a public key to a remote host.

**`static _ssh_keygen`** ()  
Generate ssh key pair.

**`_trait_default_generators`** = {}

**`_validate_port`**

**`_write_ssh_config`** (*proxycmd="", private\_key\_abs\_fname=None*)  
Put host information into the config file.

**`can_login`** (*silent=False*)  
Check if it is possible to login into the remote host.

**`hostname`**  
A trait for unicode strings.

**`is_host_known`** (*hostname=None*)  
Check if the host is known already.

**`is_in_config`** ()  
Check if the config file contains host information.

**`on_setup_ssh`** (*change*)  
Setup ssh, password and private key are supported

**`on_use_diff_proxy_username_change`** (*change*)  
If using different username for proxy check-box is clicked.

**`on_use_proxy_change`** (*change*)  
If proxy check-box is clicked.

**`on_use_verification_mode_change`** (*change*)  
which verification mode is chosen.

**`port`**  
A trait type representing a Union type.

**`proxy_hostname`**  
A trait for unicode strings.

**`proxy_username`**  
A trait for unicode strings.

**`setup_counter`**  
An int trait.

**`use_proxy`**  
A boolean (True, False) trait.

**`username`**  
A trait for unicode strings.

## 4.5 aiidalab\_widgets\_base.databases module

Widgets that allow to query online databases.

**class** aiidalab\_widgets\_base.databases.CodQueryWidget (\*\*kwargs)

Bases: ipywidgets.widgets.widget\_box.VBox

Query structures in Crystallography Open Database (COD) Useful class members: :ivar structure(Atoms): trait that contains the selected structure, None if structure is not selected.

**\_\_init\_\_** (title="", \*\*kwargs)

Public constructor

**\_\_module\_\_** = 'aiidalab\_widgets\_base.databases'

**\_default\_structure**

**\_on\_click\_query** (change)

Call query when the corresponding button is pressed.

**\_on\_select\_structure** (change)

When a structure was selected.

**static \_query** (idn=None, formula=None)

Make the actual query.

**\_trait\_default\_generators** = {'structure': <traitlets.traitlets.DefaultHandler object>

**structure**

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

**class** aiidalab\_widgets\_base.databases.CodeDatabaseWidget (\*\*kwargs)

Bases: ipywidgets.widgets.widget\_box.HBox

Extract the setup of a known computer from the AiiDA code registry.

**\_\_init\_\_** (\*\*kwargs)

Public constructor

**\_\_module\_\_** = 'aiidalab\_widgets\_base.databases'

**\_trait\_default\_generators** = {}

**append\_text**

A trait for unicode strings.

**computer**

A trait for unicode strings.

**description**

A trait for unicode strings.

**input\_plugin**

A trait for unicode strings.

**label**

A trait for unicode strings.

**on\_computer**

A boolean (True, False) trait.

**prepend\_text**

A trait for unicode strings.

**remote\_abs\_path**

A trait for unicode strings.

**update** (*\_=None*)**update\_codes** (*\_=None*)

Read settings from the YAML files and populate self.database with them.

**update\_computers** (*\_=None*)**update\_settings** (*\_=None*)

Update code settings.

**class** aiidalab\_widgets\_base.databases.**ComputerDatabaseWidget** (*\*\*kwargs*)

Bases: ipywidgets.widgets.widget\_box.HBox

Extract the setup of a known computer from the AiiDA code registry.

**\_\_init\_\_** (*\*\*kwargs*)

Public constructor

**\_\_module\_\_** = 'aiidalab\_widgets\_base.databases'

**\_observe\_proxy\_command**

**\_trait\_default\_generators** = {}

**allow\_agent**

A boolean (True, False) trait.

**append\_text**

A trait for unicode strings.

**description**

A trait for unicode strings.

**hostname**

A trait for unicode strings.

**label**

A trait for unicode strings.

**mpiprocs\_per\_machine**

An int trait.

**mpirun\_command**

A trait for unicode strings.

**num\_cores\_per\_mpiproc**

An int trait.

**port**

An int trait.

**prepend\_text**

A trait for unicode strings.

**proxy\_command**

A trait for unicode strings.

**proxy\_hostname**

A trait for unicode strings.

**proxy\_username**  
A trait for unicode strings.

**queue\_name**  
A trait for unicode strings.

**safe\_interval**  
A float trait.

**scheduler**  
A trait for unicode strings.

**shebang**  
A trait for unicode strings.

**transport**  
A trait for unicode strings.

**update** (*\_=None*)

**update\_computers** (*\_=None*)

**update\_settings** (*\_=None*)  
Read settings from the YAML files and populate self.database with them.

**use\_login\_shell**  
A boolean (True, False) trait.

**work\_dir**  
A trait for unicode strings.

**class** aiidalab\_widgets\_base.databases.OptimadeQueryWidget (\*\*kwargs)

Bases: ipywidgets.widgets.widget\_box.VBox

AiiDALab-specific OPTIMADE Query widget

Useful as a widget to integrate with the `aiidalab_widgets_base.structures.StructureManagerWidget`, embedded into applications.

NOTE: *embedded* for `OptimadeQueryFilterWidget` was introduced in `optimade-client` version 2020.11.5.

#### Parameters

- **embedded** (*bool*) – Whether or not to show extra database and provider information. When set to *True*, the extra information will be hidden, this is useful in situations where the widget is used in a Tab or similar, e.g., for the `aiidalab_widgets_base.structures.StructureManagerWidget`.
- **title** (*str*) – Title used for Tab header if employed in `aiidalab_widgets_base.structures.StructureManagerWidget`.

**\_\_init\_\_** (*embedded: bool = True, title: Optional[str] = None, \*\*kwargs*) → None  
Public constructor

**\_\_module\_\_** = 'aiidalab\_widgets\_base.databases'

**\_\_trait\_default\_generators** = {}

**\_\_update\_structure** (*change: dict*) → None  
New structure chosen

**structure**  
A trait whose value must be an instance of a specified class.  
The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

## 4.6 aiidalab\_widgets\_base.dicts module

## 4.7 aiidalab\_widgets\_base.export module

Widgets to manage AiiDA export.

```
class aiidalab_widgets_base.export.ExportButtonWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_button.Button

    Export Node button.

    __init__ (process, **kwargs)
        Public constructor

    __module__ = 'aiidalab_widgets_base.export'

    _trait_default_generators = {}

    export_aiida_subgraph (change=None)
        Perform export when the button is pressed
```

## 4.8 aiidalab\_widgets\_base.misc module

Some useful classes used across the repository.

```
class aiidalab_widgets_base.misc.CopyToClipboardButton (**kwargs)
    Bases: ipywidgets.widgets.widget_button.Button

    Button to copy text to clipboard.

    __init__ (*args, **kwargs)
        Public constructor

    __module__ = 'aiidalab_widgets_base.misc'

    _trait_default_generators = {}

    copy_to_clipboard (change=None)
        Copy text to clipboard.

    value
        A trait for unicode strings.
```

```
class aiidalab_widgets_base.misc.ReversePolishNotation (operators,
                                                    additional_operands=None)

    Bases: object

    Class defining operations for RPN conversion

    __dict__ = mappingproxy({'__module__': 'aiidalab_widgets_base.misc', '__doc__': 'Cla
    __init__ (operators, additional_operands=None)
        Initialize self. See help(type(self)) for accurate signature.

    __module__ = 'aiidalab_widgets_base.misc'
```

**`__weakref__`**  
list of weak references to the object (if defined)

**`convert`** (*expr*)  
Convert expression to postfix.

**`execute`** (*expression*)  
Execute the provided expression.

**`haslessorequalpriority`** (*opa, opb*)  
Priority of the different operators

**`is_operator`** (*opx*)  
Identifies operators

**`static iscloseparenthesis`** (*operator*)  
Identifies closed parentheses.

**`static isopenparenthesis`** (*operator*)  
Identifies open parentheses.

**`static parse_infix_notation`** (*condition*)  
Convert a string containing the expression into a list of operators and operands.

## 4.9 aiidalab\_widgets\_base.nodes module

Widgets to work with AiiDA nodes.

```
class aiidalab_widgets_base.nodes.AiidaNodeTreeNode (**kwargs)
    Bases: ipytreetree.Node
    __init__ (pk, name, **kwargs)
        Public constructor
    __module__ = 'aiidalab_widgets_base.nodes'
    _default_openend
    _trait_default_generators = {'opened': <traitlets.traitlets.DefaultHandler object>}

class aiidalab_widgets_base.nodes.AiidaOutputsTreeNode (**kwargs)
    Bases: ipytreetree.Node
    __init__ (name, parent_pk, **kwargs)
        Public constructor
    __module__ = 'aiidalab_widgets_base.nodes'
    _trait_default_generators = {}
    disabled
        A boolean (True, False) trait.
    icon
        A trait for unicode strings.

class aiidalab_widgets_base.nodes.AiidaProcessNodeTreeNode (**kwargs)
    Bases: aiidalab_widgets_base.nodes.AiidaNodeTreeNode
    __init__ (pk, **kwargs)
        Public constructor
    __module__ = 'aiidalab_widgets_base.nodes'
```

```

        _trait_default_generators = {}
class aiidalab_widgets_base.nodes.CalcFunctionTreeNode (**kwargs)
    Bases: aiidalab_widgets_base.nodes.AiidaProcessNodeTreeNode
    __module__ = 'aiidalab_widgets_base.nodes'
    _trait_default_generators = {}
    icon
        A trait for unicode strings.
class aiidalab_widgets_base.nodes.CalcJobTreeNode (**kwargs)
    Bases: aiidalab_widgets_base.nodes.AiidaProcessNodeTreeNode
    __module__ = 'aiidalab_widgets_base.nodes'
    _trait_default_generators = {}
    icon
        A trait for unicode strings.
class aiidalab_widgets_base.nodes.NodesTreeWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_output.Output
    A tree widget for the structured representation of a nodes graph.
    NODE_TYPE = {<class 'aiida.orm.nodes.process.workflow.workchain.WorkChainNode'>: <cla
    PROCESS_STATE_STYLE = {<ProcessState.EXCEPTED: 'excepted'>: 'danger', <ProcessState.F
    PROCESS_STATE_STYLE_DEFAULT = 'default'
    __init__ (**kwargs)
        Public constructor
    __module__ = 'aiidalab_widgets_base.nodes'
    classmethod _build_tree (root)
        Recursively build a tree nodes graph for a given tree node.
    _convert_to_tree_nodes (old_nodes, new_nodes)
        Convert nodes into tree nodes while re-using already converted nodes.
    classmethod _find_called (root)
    classmethod _find_children (root)
        Find all children of the provided AiiDA node.
    classmethod _find_outputs (root)
    _observe_nodes
    _observe_tree_selected_nodes (change)
    _refresh_output ()
    classmethod _to_tree_node (node, name=None, **kwargs)
        Convert an AiiDA node to a tree node.
    _trait_default_generators = {}
    _update_tree_node (tree_node)
    classmethod _walk_tree (root)
        Breadth-first search of the node tree.
    find_node (pk)

```



**nodes**

An instance of a Python tuple.

**selected\_nodes**

An instance of a Python tuple.

**update** (*\_=None*)

Refresh nodes based on the latest state of the root process and its children.

```
class aiidalab_widgets_base.nodes.UnknownTypeTreeNode (**kwargs)
```

Bases: *aiidalab\_widgets\_base.nodes.AiidaNodeTreeNode*

```
__module__ = 'aiidalab_widgets_base.nodes'
```

```
_trait_default_generators = {}
```

**icon**

A trait for unicode strings.

```
class aiidalab_widgets_base.nodes.WorkChainProcessTreeNode (**kwargs)
```

Bases: *aiidalab\_widgets\_base.nodes.AiidaProcessNodeTreeNode*

```
__module__ = 'aiidalab_widgets_base.nodes'
```

```
_trait_default_generators = {}
```

**icon**

A trait for unicode strings.

## 4.10 aiidalab\_widgets\_base.process module

Widgets to work with processes.

```
class aiidalab_widgets_base.process.CalcJobOutputWidget (**kwargs)
```

Bases: *ipywidgets.widgets.widget\_string.Textarea*

Output of a calculation.

```
__init__ (**kwargs)
```

Public constructor

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_change_calculation
```

```
_trait_default_generators = {}
```

**calculation**

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

**update** ()

Update the displayed output and scroll to its end.

NOTE: when this widgets is called by ProcessFollowerWidget in non-blocking manner the auto-scrolling won't work. There used to be a function for the Textarea widget, but it didn't work properly and got removed. For more information please visit: <https://github.com/jupyter-widgets/ipywidgets/issues/1815>

```
class aiidalab_widgets_base.process.ProcessCallStackWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_string.HTML
```

Widget that shows process call stack.

```
__init__ (title='Process Call Stack', path_to_root='./', **kwargs)
    Public constructor
```

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_trait_default_generators = {}
```

```
calc_info (node)
```

Return a string with the summary of the state of a CalculationNode.

```
process
```

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

```
update ()
```

Update the call stack that is shown.

```
class aiidalab_widgets_base.process.ProcessFollowerWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox
```

A Widget that follows a process until finished.

```
__init__ (process=None, followers=None, update_interval=0.1, path_to_root='./', **kwargs)
    Initiate all the followers.
```

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_trait_default_generators = {}
```

```
follow (detach=False)
```

Initiate following the process with or without blocking.

```
on_completed (function)
```

Run functions after a process has been completed.

```
process
```

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

```
update ()
```

```
class aiidalab_widgets_base.process.ProcessInputsWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox
```

Widget to select and show process inputs.

```
__init__ (process=None, **kwargs)
    Public constructor
```

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_trait_default_generators = {}
```

```
process
```

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

**show\_selected\_input** (*change=None*)

Function that displays process inputs selected in the *inputs* Dropdown widget.

**class** aiidalab\_widgets\_base.process.ProcessListWidget (\*\*kwargs)

Bases: ipywidgets.widgets.widget\_box.VBox

List of AiiDA processes.

*past\_days* (int): Simulations that were submitted in the last *past\_days*.

*incoming\_node* (int, str, Node): Trait that takes node id or uuid and returns the node that must be among the input nodes of the process of interest.

*outgoing\_node* (int, str, Node): Trait that takes node id or uuid and returns the node that must be among the output nodes of the process of interest.

*process\_states* (list): List of allowed process states.

*process\_label* (str): Show process states of type *process\_label*.

*description\_contains* (str): string that should be present in the description of a process node.

**\_\_init\_\_** (*path\_to\_root='./', \*\*kwargs*)

Public constructor

**\_\_module\_\_** = 'aiidalab\_widgets\_base.process'

**\_\_default\_process\_label**

**\_\_follow** (*update\_interval*)

**\_\_trait\_default\_generators** = {'process\_label': <traitlets.traitlets.DefaultHandler obj>

**\_\_validate\_incoming\_node**

**\_\_validate\_outgoing\_node**

**\_\_validate\_process\_label**

**description\_contains**

A trait for unicode strings.

**incoming\_node**

A trait type representing a Union type.

**outgoing\_node**

A trait type representing a Union type.

**past\_days**

An int trait.

**process\_label**

A trait for unicode strings.

**process\_states**

An instance of a Python list.

**start\_autoupdate** (*update\_interval=10*)

**update** (*\_=None*)

Perform the query.

```

class aiidalab_widgets_base.process.ProcessMonitor (**kwargs)
    Bases: traitlets.traitlets.HasTraits

    Monitor a process and execute callback functions at specified intervals.

    __init__ (callbacks=None, on_sealed=None, timeout=None, **kwargs)
        Initialize self. See help(type(self)) for accurate signature.

    __module__ = 'aiidalab_widgets_base.process'

    _monitor_process (process_id)

    _observe_process

    _trait_default_generators = {}

    join ()

    process
        A trait whose value must be an instance of a specified class.

        The value can also be an instance of a subclass of the specified class.

        Subclasses can declare default classes by overriding the class attribute

class aiidalab_widgets_base.process.ProcessNodesTreeWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    A tree widget for the structured representation of a process graph.

    __init__ (title='Process Tree', **kwargs)
        Public constructor

    __module__ = 'aiidalab_widgets_base.process'

    _observe_process

    _observe_tree_selected_nodes (change)

    _trait_default_generators = {}

    process
        A trait whose value must be an instance of a specified class.

        The value can also be an instance of a subclass of the specified class.

        Subclasses can declare default classes by overriding the class attribute

    selected_nodes
        An instance of a Python tuple.

    update (_=None)

class aiidalab_widgets_base.process.ProcessOutputsWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    Widget to select and show process outputs.

    __init__ (process=None, **kwargs)
        Public constructor

    __module__ = 'aiidalab_widgets_base.process'

    _trait_default_generators = {}

    process
        A trait whose value must be an instance of a specified class.

```

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

**show\_selected\_output** (*change=None*)

Function that displays process output selected in the *outputs* Dropdown widget.

```
class aiidalab_widgets_base.process.ProcessReportWidget (**kwargs)
```

Bases: ipywidgets.widgets.widget\_string.HTML

Widget that shows process report.

```
__init__ (title='Process Report', **kwargs)
```

Public constructor

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_trait_default_generators = {}
```

**process**

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

```
update ()
```

Update report that is shown.

```
class aiidalab_widgets_base.process.ProgressBarWidget (**kwargs)
```

Bases: ipywidgets.widgets.widget\_box.VBox

A bar showing the progress of a process.

```
__init__ (title='Progress Bar', **kwargs)
```

Initialize ProgressBarWidget.

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_trait_default_generators = {}
```

**property current\_state**

**process**

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

```
update ()
```

Update the bar.

```
class aiidalab_widgets_base.process.RunningCalcJobOutputWidget (**kwargs)
```

Bases: ipywidgets.widgets.widget\_box.VBox

Show an output of selected running child calculation.

```
__init__ (title='Running Job Output', **kwargs)
```

Public constructor

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_trait_default_generators = {}
```

**process**

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

**update ()**

Update the displayed output.

**class** aiidalab\_widgets\_base.process.**SubmitButtonWidget** (\*\*kwargs)

Bases: ipywidgets.widgets.widget\_box.VBox

Submit button class that creates submit button jupyter widget.

**\_\_init\_\_** (process\_class, inputs\_generator=None, input\_dictionary\_function=None, description='Submit', disable\_after\_submit=True, append\_output=False, \*\*kwargs)  
Submit Button widget.

process\_class (Process): Process class to submit.

inputs\_generator (func): Function that returns inputs dictionary or inputs builder.

input\_dictionary\_function (DEPRECATED): Function that generates input parameters dictionary.

description (str): Description written on the submission button.

disable\_after\_submit (bool): Whether to disable the button after the process was submitted.

append\_output (bool): Whether to clear widget output for each subsequent submission.

**\_\_module\_\_** = 'aiidalab\_widgets\_base.process'

**\_trait\_default\_generators** = {}

**on\_btn\_submit\_press** (\_=None)

When submit button is pressed.

**on\_click** (function)

**on\_submitted** (function)

Run functions after a process has been submitted successfully.

**process**

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

aiidalab\_widgets\_base.process.**get\_running\_calcs** (process)

Takes a process and yeilds running children calculations.

## 4.11 aiidalab\_widgets\_base.structures module

Module to provide functionality to import structures.

**class** aiidalab\_widgets\_base.structures.**BasicStructureEditor** (\*\*kwargs)

Bases: ipywidgets.widgets.widget\_box.VBox

Widget that allows for the basic structure editing.

**\_\_init\_\_** (title="")

Public constructor

```

__module__ = 'aiidalab_widgets_base.structures'
__trait_default_generators = {}

property action_vector
    Define the action vector.

add (_=None)
    Add atoms.

align (_=None)
    Rotate atoms to align action vector with XYZ vector.

camera_orientation
    An instance of a Python list.

copy_sel (_=None)
    Copy selected atoms and shift by 1.0 A along X-axis.

def_axis_p1 (_=None)
    Define the first point of axis.

def_axis_p2 (_=None)
    Define the second point of axis.

def_perpendicular_to_screen (_=None)
    Define a normalized vector perpendicular to the screen.

def_point (_=None)
    Define the action point.

mirror (_=None, norm=None, point=None)
    Mirror atoms on the plane perpendicular to the action vector.

mirror_3p (_=None)
    Mirror atoms on the plane containing action vector and action point.

mod_element (_=None)
    Modify selected atoms into the given element.

remove (_)
    Remove selected atoms.

rotate (_=None)
    Rotate atoms around selected point in space and vector.

sel2com (_)
    Get center of mass of the selection.

selection
    An instance of a Python list.

str2vec (string)

structure
    A trait whose value must be an instance of a specified class.

    The value can also be an instance of a subclass of the specified class.

    Subclasses can declare default classes by overriding the class attribute

translate_dr (_=None)
    Translate by dr along the selected vector.

```

**translate\_dx dy dz** (*\_ = None*)  
Translate by the selected XYZ delta.

**translate\_to\_xyz** (*\_ = None*)  
Translate to the selected XYZ position.

**vec2str** (*vector*)

**class** aiidalab\_widgets\_base.structures.SmilesWidget (\*\*kwargs)

Bases: ipywidgets.widgets.widget\_box.VBox

Conver SMILES into 3D structure.

**SPINNER** = '<i class="fa fa-spinner fa-pulse" style="color:red;" ></i>'

**\_\_init\_\_** (*title=""*)  
Public constructor

**\_\_module\_\_** = 'aiidalab\_widgets\_base.structures'

**\_\_default\_structure**

**\_\_on\_button\_pressed** (*change*)  
Convert SMILES to ase structure when button is pressed.

**\_\_pybel\_opt** (*smile, steps*)  
Optimize a molecule using force field and pybel (needed for complex SMILES).

**\_\_rdkit\_opt** (*smile, steps*)  
Optimize a molecule using force field and rdkit (needed for complex SMILES).

**\_\_trait\_default\_generators** = {'structure': <traitlets.traitlets.DefaultHandler object>

**make\_ase** (*species, positions*)  
Create ase Atoms object.

**mol\_from\_smiles** (*smile, steps=10000*)  
Convert SMILES to ase structure try rdkit then pybel

**structure**  
A trait whose value must be an instance of a specified class.  
The value can also be an instance of a subclass of the specified class.  
Subclasses can declare default classes by overriding the class attribute

**class** aiidalab\_widgets\_base.structures.StructureBrowserWidget (\*\*kwargs)

Bases: ipywidgets.widgets.widget\_box.VBox

Class to query for structures stored in the AiiDA database.

**\_\_init\_\_** (*title=""*)  
Public constructor

**\_\_module\_\_** = 'aiidalab\_widgets\_base.structures'

**\_\_on\_select\_structure** (*\_ = None*)

**\_\_trait\_default\_generators** = {}

**preprocess** ()  
Search structures in AiiDA database and add formula extra to them.

**search** (*\_ = None*)  
Launch the search of structures in AiiDA database.



```

structure
    A trait type representing a Union type.
class aiidalab_widgets_base.structures.StructureExamplesWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    Class to provide example structures for selection.

    __init__ (examples, title="", **kwargs)
        Public constructor

    __module__ = 'aiidalab_widgets_base.structures'

    _default_structure

    _on_select_structure (change)
        When structure is selected.

    _trait_default_generators = {'structure': <traitlets.traitlets.DefaultHandler object>

    static get_example_structures (examples)
        Get the list of example structures.

structure
    A trait whose value must be an instance of a specified class.

    The value can also be an instance of a subclass of the specified class.

    Subclasses can declare default classes by overriding the class attribute

class aiidalab_widgets_base.structures.StructureManagerWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    Upload a structure and store it in AiiDA database.

Attributes: structure(Atoms): trait that contains the selected structure. 'None' if no structure is selected. structure_node(StructureData, CifData): trait that contains AiiDA structure object node_class(str): trait that contains structure_node type (as string).

SUPPORTED_DATA_FORMATS = {'CifData': 'cif', 'StructureData': 'structure'}

__init__ (importers, viewer=None, editors=None, storable=True, node_class=None, **kwargs)

Arguments:

    importers(list): list of tuples each containing the displayed name of importer and the importer
        object. Each object should contain 'structure' trait pointing to the imported structure. The trait
        will be linked to 'structure' trait of this class.

    storable(bool): Whether to provide Store button (together with Store format)

    node_class(str): AiiDA node class for storing the structure. Possible values: 'StructureData',
        'CifData' or None (let the user decide). Note: If your workflows require a specific node class,
        better fix it here.

    __module__ = 'aiidalab_widgets_base.structures'

    _change_structure_node

    _convert_to_structure_node (structure)
        Convert structure of any type to the StructureNode object.

    _default_node_class

    _observe_input_structure

    _observe_structure_node

```

**`_structure_changed`**

**`_structure_editors`** (*editors*)  
 Preparing structure editors.

**`_structure_importers`** (*importers*)  
 Preparing structure importers.

**`_sync_structure_node`** ()  
 Synchronize the `structure_node` trait using the currently provided info.

**`_trait_default_generators`** = {}

**`input_structure`**  
 A trait type representing a Union type.

**`node_class`**  
 A trait for unicode strings.

**`store_structure`** (*\_=None*)  
 Stores the structure in AiiDA database.

**`structure`**  
 A trait type representing a Union type.

**`structure_node`**  
 A trait whose value must be an instance of a specified class.  
 The value can also be an instance of a subclass of the specified class.  
 Subclasses can declare default classes by overriding the class attribute

**`undo`** (*\_*)  
 Undo modifications.

**class** `aiidalab_widgets_base.structures.StructureUploadWidget` (\*\**kwards*)  
 Bases: `ipywidgets.widgets.widget_box.VBox`  
 Class that allows to upload structures from user's computer.

**`__init__`** (*title=""*, *description='Upload Structure'*)  
 Public constructor

**`__module__`** = `'aiidalab_widgets_base.structures'`

**`_on_file_upload`** (*change=None*)  
 When file upload button is pressed.

**`_trait_default_generators`** = {}

**`_validate_and_fix_ase_cell`** (*ase\_structure*, *vacuum\_ang=10.0*)  
 Checks if the ase Atoms object has a cell set, otherwise sets it to bounding box plus specified "vacuum" space

**`structure`**  
 A trait type representing a Union type.

## 4.12 aiidalab\_widgets\_base.structures\_multi module

Module to deal with files containing multiple structures.

**class** aiidalab\_widgets\_base.structures\_multi.**MultiStructureUploadWidget** (\*\*kwargs)  
Bases: ipywidgets.widgets.widget\_box.VBox

Class to deal with archives (tar or zip) containing multiple structures.

**\_\_init\_\_** (text='Upload Zip or Tar archive', node\_class=None, \*\*kwargs)  
Upload multiple structures and store them in AiiDA database.

### Parameters

- **text** (*str*) – Text to display before upload button
- **node\_class** – AiiDA node class for storing the structure. Possible values: 'Structure-Data', 'CifData' or None (let the user decide). Note: If your workflows require a specific node class, better fix it here.

**\_\_module\_\_** = 'aiidalab\_widgets\_base.structures\_multi'

**\_on\_click\_store\_all** (*change*)  
Store all the uploaded structures.

**\_on\_click\_store\_selected** (*change*)

**\_on\_file\_upload** (*change*)  
Process the archive once it is uplodaded.

**\_trait\_default\_generators** = {}

**change\_structure** ()

**get\_ase** (*filepath*)  
Get an ase object containing the structure.

**static get\_description** (*structure\_ase, filepath*)  
Get the structure description automatically.

**property node\_class**

**refresh\_view** ()  
Refresh the structure view.

**select\_structure** (*filepath*)  
Perform structure selection.

**store\_structure** (*filepath, description=None*)  
Store the structure in the AiiDA database.

## 4.13 aiidalab\_widgets\_base.viewers module

Jupyter viewers for AiiDA data objects.

**class** aiidalab\_widgets\_base.viewers.**BandsDataViewer** (\*\*kwargs)  
Bases: ipywidgets.widgets.widget\_box.VBox

Viewer class for BandsData object.

**Parameters bands** (*BandsData*) – BandsData object to be viewed

```
__init__(bands, **kwargs)
    Public constructor
```

```
__module__ = 'aiidalab_widgets_base.viewers'
```

```
_trait_default_generators = {}
```

```
class aiidalab_widgets_base.viewers.DictViewer(**kwargs)
```

```
Bases: ipywidgets.widgets.widget_box.VBox
```

```
__init__(parameter, downloadable=True, **kwargs)
    Public constructor
```

```
__module__ = 'aiidalab_widgets_base.viewers'
```

```
_trait_default_generators = {}
```

**value**

Viewer class for Dict object.

**Parameters**

- **parameter** (*Dict*) – Dict object to be viewed
- **downloadable** (*bool*) – If True, add link/button to download the content of the object

```
class aiidalab_widgets_base.viewers.FolderDataViewer(**kwargs)
```

```
Bases: ipywidgets.widgets.widget_box.VBox
```

Viewer class for FolderData object.

**Parameters**

- **folder** (*FolderData*) – FolderData object to be viewed
- **downloadable** (*bool*) – If True, add link/button to download the content of the selected file in the folder

```
__init__(folder, downloadable=True, **kwargs)
    Public constructor
```

```
__module__ = 'aiidalab_widgets_base.viewers'
```

```
_trait_default_generators = {}
```

```
change_file_view(change=None)
```

```
download(change=None)
    Prepare for downloading.
```

```
class aiidalab_widgets_base.viewers.StructureDataViewer(**kwargs)
```

```
Bases: aiidalab_widgets_base.viewers._StructureDataBaseViewer
```

Viewer class for AiiDA structure objects.

**Attributes:** *structure* (*Atoms*, *StructureData*, *CifData*): Trait that contains a structure object, which was initially provided to the viewer. It can be either directly set to an ASE Atoms object or to AiiDA structure object containing *get\_ase()* method.

*displayed\_structure* (*Atoms*): Trait that contains a structure object that is currently displayed (super cell, for example). The trait is generated automatically and can't be set outside of the class.

```
__init__(structure=None, **kwargs)
    Public constructor
```

```
__module__ = 'aiidalab_widgets_base.viewers'
```

```

    _observe_selection_2
    _observe_selection_adv
    _trait_default_generators = {}
    _update_displayed_structure
    _update_structure_viewer
    _valid_structure
    create_selection_info ()
        Create information to be displayed with selected atoms
    d_from (operand)
    displayed_structure
        A trait whose value must be an instance of a specified class.
        The value can also be an instance of a subclass of the specified class.
        Subclasses can declare default classes by overriding the class attribute
    name_operator (operand)
        Defining the name operator which will handle atom kind names.
    not_operator (operand)
        Reverting the selected atoms.
    parse_advanced_sel (condition=None)
        Apply advanced selection specified in the text field.
    repeat
    structure
        A trait type representing a Union type.
class aiidalab_widgets_base.viewers._StructureDataBaseViewer (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox
    Base viewer class for AiiDA structure or trajectory objects.
        Parameters configure_view (bool) – If True, add configuration tabs
    DEFAULT_SELECTION_COLOR = 'green'
    DEFAULT_SELECTION_OPACITY = 0.2
    DEFAULT_SELECTION_RADIUS = 6
    __init__ (configure_view=True, **kwargs)
        Public constructor
    __module__ = 'aiidalab_widgets_base.viewers'
    _appearance_tab ()
        Defining the appearance tab.
    _default_selection
    _default_supercell
    static _download (payload, filename)
        Download payload as a file named as filename.
    _download_tab ()
        Defining the download tab.

```

```

    _observe_selection
_on_atom_click (_=None)
    Update selection when clicked on atom.
_prepare_payload (file_format=None)
    Prepare binary information.
_render_structure (change=None)
    Render the structure with POVRAY.
_selection_tab ()
    Defining the selection tab.
_trait_default_generators = {'selection': <traitlets.traitlets.DefaultHandler object>
_validate_selection
apply_selection (_=None)
    Apply selection specified in the text field.
download (change=None)
    Prepare a structure for downloading.
highlight_atoms (vis_list, color='green', size=6, opacity=0.2)
    Highlighting atoms according to the provided list.
selection
    An instance of a Python list.
selection_adv
    A trait for unicode strings.
supercell
    An instance of a Python list.
property thumbnail
aiidalab_widgets_base.viewers.register_viewer_widget (key)
    Register widget as a viewer for the given key.
aiidalab_widgets_base.viewers.viewer (obj, downloadable=True, **kwargs)
    Display AiiDA data types in Jupyter notebooks.
        Parameters downloadable (bool) – If True, add link/button to download the content of displayed AiiDA object.
    Returns the object itself if the viewer wasn't found.

```

## 4.14 aiidalab\_widgets\_base.wizard module

The wizard application allows the implication of a Wizard-like GUI.

Authors:

- Carl Simon Adorf <[simon.adorf@epfl.ch](mailto:simon.adorf@epfl.ch)>

```

class aiidalab_widgets_base.wizard.WizardAppWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox
    ICONS = {<State.INIT: 0>: ' ', <State.READY: 2>: ' ', <State.CONFIGURED: 1>: ' ', <
    ICON_SEPARATOR = '\u2000'

```

```
__init__ (steps, **kwargs)
```

Public constructor

```
__module__ = 'aiidalab_widgets_base.wizard'
```

```
__consider_auto_advance (_=None)
```

Determine whether the app should automatically advance to the next step.

This is performed whenever the current step is within the SUCCESS state and has the `auto_advance` attribute set to True.

```
__observe_selected_index
```

```
__on_click_back_button (_)
```

```
__on_click_next_button (_)
```

```
__on_click_reset_button (_)
```

```
__trait_default_generators = {}
```

```
__update_buttons (_)
```

```
__update_step_state (_)
```

```
__update_titles (_)
```

```
can_reset (_)
```

```
classmethod icons (_)
```

Return the icon set and return animated icons based on the current time stamp.

```
reset (step=0)
```

Reset the app up to the given step.

For example, with `step=0` (the default), the whole app is reset. With `step=1`, all but the first step are reset.

```
selected_index
```

An int trait.

```
class aiidalab_widgets_base.wizard.WizardAppWidgetStep (**kwargs)
```

Bases: `traitlets.traitlets.HasTraits`

One step of a `WizardAppWidget`.

```
class State (value)
```

Bases: `enum.Enum`

Each step is always in one specific state.

The state is used to determine:

- 1) how the step is visually presented to the user, and
- 2) whether the next step is accessible (i.e. reached the SUCCESS state).

App developers are encouraged to use the step states to couple application logic and interface. In general, all widget changes should trigger a re-evaluation of the step state, and states also determine whether certain widgets are enabled or disabled.

A step can be in one of the following states:

INIT: The initial state, usually all widgets disabled. READY: The step (widget) is ready for user input (some or all widgets enabled). CONFIGURED: The step is in a consistent configuration awaiting confirmation. ACTIVE: The step is carrying out a runtime operation. SUCCESS: A configuration has been confirmed / a runtime operation successfully finished. FAIL: A runtime operation has failed in an unrecoverable way.

Not all steps must implement all states, for example:

- the first step does not need an INIT state
- a step without runtime process should not have an ACTIVE or FAIL state
- a “review & confirm” step does not require a READY state.
- a step without configuration options (e.g. pure “review & confirm” step)

Important: The next step is only accessible if the current step is within the SUCCESS state!

```
ACTIVE = 3
CONFIGURED = 1
FAIL = -1
INIT = 0
READY = 2
SUCCESS = 4

__module__ = 'aiidalab_widgets_base.wizard'
__module__ = 'aiidalab_widgets_base.wizard'
_trait_default_generators = {}
```

**auto\_advance**

A boolean (True, False) trait.

**can\_reset ()**

**state**

Use a Enum class as model for the data type description. Note that if no default-value is provided, the first enum-value is used as default-value.

```
# -- SINCE: Python 3.4 (or install backport: pip install enum34)
import enum
from traitlets import HasTraits, UseEnum

class Color(enum.Enum):
    red = 1          # -- IMPLICIT: default_value
    blue = 2
    green = 3

class MyEntity(HasTraits):
    color = UseEnum(Color, default_value=Color.blue)

entity = MyEntity(color=Color.red)
entity.color = Color.green      # USE: Enum-value (preferred)
entity.color = "green"         # USE: name (as string)
entity.color = "Color.green"   # USE: scoped-name (as string)
entity.color = 3               # USE: number (as int)
assert entity.color is Color.green
```



## 4.15 Module contents

Reusable widgets for AiiDA lab applications.

```

class aiidalab_widgets_base.AiiDACodeSetup (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    Class that allows to setup AiiDA code

    __annotations__ = {}

    __init__ (**kwargs)
        Public constructor

    __module__ = 'aiidalab_widgets_base.codes'

    _setup_code (_=None)
        Setup an AiiDA code.

    _trait_default_generators = {}

    append_text
        A trait for unicode strings.

    computer
        A trait type representing a Union type.

    description
        A trait for unicode strings.

    exists ()
        Returns True if the code exists, returns False otherwise.

    input_plugin
        A trait for unicode strings.

    label
        A trait for unicode strings.

    prepend_text
        A trait for unicode strings.

    remote_abs_path
        A trait for unicode strings.

class aiidalab_widgets_base.AiidaComputerSetup (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    Inform AiiDA about a computer.

    __annotations__ = {}

    __init__ (**kwargs)
        Public constructor

    __module__ = 'aiidalab_widgets_base.computers'

    _configure_computer ()
        Create AuthInfo.

    _on_setup_computer (_=None)
        When setup computer button is pressed.

    _trait_default_generators = {}

```

```
_validate_mpiprocs_per_machine  
_validate_safe_interval  
append_text  
    A trait for unicode strings.  
description  
    A trait for unicode strings.  
hostname  
    A trait for unicode strings.  
label  
    A trait for unicode strings.  
mpiprocs_per_machine  
    A trait type representing a Union type.  
mpirun_command  
    A trait for unicode strings.  
prepend_text  
    A trait for unicode strings.  
safe_interval  
    A trait type representing a Union type.  
scheduler  
    A trait for unicode strings.  
test (_=None)  
transport  
    A trait for unicode strings.  
work_dir  
    A trait for unicode strings.  
class aiidalab_widgets_base.BasicStructureEditor (**kwargs)  
    Bases: ipywidgets.widgets.widget_box.VBox  
    Widget that allows for the basic structure editing.  
    __annotations__ = {}  
    __init__ (title="")  
        Public constructor  
    __module__ = 'aiidalab_widgets_base.structures'  
    _trait_default_generators = {}  
property action_vector  
    Define the action vector.  
add (_=None)  
    Add atoms.  
align (_=None)  
    Rotate atoms to align action vector with XYZ vector.  
camera_orientation  
    An instance of a Python list.
```

---

```

copy_sel (_=None)
    Copy selected atoms and shift by 1.0 Å along X-axis.

def_axis_p1 (_=None)
    Define the first point of axis.

def_axis_p2 (_=None)
    Define the second point of axis.

def_perpendicular_to_screen (_=None)
    Define a normalized vector perpendicular to the screen.

def_point (_=None)
    Define the action point.

mirror (_=None, norm=None, point=None)
    Mirror atoms on the plane perpendicular to the action vector.

mirror_3p (_=None)
    Mirror atoms on the plane containing action vector and action point.

mod_element (_=None)
    Modify selected atoms into the given element.

remove (_)
    Remove selected atoms.

rotate (_=None)
    Rotate atoms around selected point in space and vector.

sel2com (_)
    Get center of mass of the selection.

selection
    An instance of a Python list.

str2vec (string)

structure
    A trait whose value must be an instance of a specified class.

    The value can also be an instance of a subclass of the specified class.

    Subclasses can declare default classes by overriding the class attribute

translate_dr (_=None)
    Translate by dr along the selected vector.

translate_dxdydz (_=None)
    Translate by the selected XYZ delta.

translate_to_xyz (_=None)
    Translate to the selected XYZ position.

vec2str (vector)

class aiidalab_widgets_base.CodQueryWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    Query structures in Crystallography Open Database (COD) Useful class members: :ivar structure(Atoms): trait
    that contains the selected structure, None if structure is not selected.

    __annotations__ = {}

```

```

__init__ (title="", **kwargs)
    Public constructor

__module__ = 'aiidalab_widgets_base.databases'

__default_structure

__on_click_query (change)
    Call query when the corresponding button is pressed.

__on_select_structure (change)
    When a structure was selected.

static __query (idn=None, formula=None)
    Make the actual query.

__trait_default_generators = {'structure': <traitlets.traitlets.DefaultHandler object>

structure
    A trait whose value must be an instance of a specified class.

    The value can also be an instance of a subclass of the specified class.

    Subclasses can declare default classes by overriding the class attribute

class aiidalab_widgets_base.CodeDatabaseWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.HBox

    Extract the setup of a known computer from the AiiDA code registry.

    __annotations__ = {}

    __init__ (**kwargs)
        Public constructor

    __module__ = 'aiidalab_widgets_base.databases'

    __trait_default_generators = {}

append_text
    A trait for unicode strings.

computer
    A trait for unicode strings.

description
    A trait for unicode strings.

input_plugin
    A trait for unicode strings.

label
    A trait for unicode strings.

on_computer
    A boolean (True, False) trait.

prepend_text
    A trait for unicode strings.

remote_abs_path
    A trait for unicode strings.

update (_=None)

```

**update\_codes** (*\_=None*)  
Read settings from the YAML files and populate self.database with them.

**update\_computers** (*\_=None*)

**update\_settings** (*\_=None*)  
Update code settings.

**class** aiidalab\_widgets\_base.**CodeDropdown** (*\*\*kwargs*)

Bases: ipywidgets.widgets.widget\_box.VBox

Code selection widget. Attributes:

**selected\_code**(Unicode or Code): Trait that points to the selected Code instance. It can be set either to an AiiDA Code instance or to a code label (will automatically be replaced by the corresponding Code instance). It is linked to the 'value' trait of the *self.dropdown* widget.

**codes**(Dict): Trait that contains a dictionary (label => Code instance) for all codes found in the AiiDA database for the selected plugin. It is linked to the 'options' trait of the *self.dropdown* widget.

**allow\_hidden\_codes**(Bool): Trait that defines whether to show hidden codes or not.

**allow\_disabled\_computers**(Bool): Trait that defines whether to show codes on disabled computers.

**\_\_annotations\_\_** = {}

**\_\_init\_\_** (*input\_plugin, description='Select code:', path\_to\_root='./', \*\*kwargs*)  
Dropdown for Codes for one input plugin.

**input\_plugin** (str): Input plugin of codes to show.

**description** (str): Description to display before the dropdown.

**\_\_module\_\_** = 'aiidalab\_widgets\_base.codes'

**static** **\_full\_code\_label** (*code*)

**\_get\_codes** ()  
Query the list of available codes.

**\_trait\_default\_generators** = {}

**\_validate\_selected\_code**

**allow\_disabled\_computers**  
A boolean (True, False) trait.

**allow\_hidden\_codes**  
A boolean (True, False) trait.

**codes**  
An instance of a Python dict.

One or more traits can be passed to the constructor to validate the keys and/or values of the dict. If you need more detailed validation, you may use a custom validator method.

Changed in version 5.0: Added *key\_trait* for validating dict keys.

Changed in version 5.0: Deprecated ambiguous *trait, traits* args in favor of *value\_trait, per\_key\_traits*.

**refresh** (*\_=None*)  
Refresh available codes.

The job of this function is to look in AiiDA database, find available codes and put them in the dropdown attribute.

**selected\_code**

A trait type representing a Union type.

**class** aiidalab\_widgets\_base.ComputerDatabaseWidget (\*\*kwargs)

Bases: ipywidgets.widgets.widget\_box.HBox

Extract the setup of a known computer from the AiiDA code registry.

**\_\_annotations\_\_** = {}

**\_\_init\_\_** (\*\*kwargs)

Public constructor

**\_\_module\_\_** = 'aiidalab\_widgets\_base.databases'

**\_observe\_proxy\_command**

**\_trait\_default\_generators** = {}

**allow\_agent**

A boolean (True, False) trait.

**append\_text**

A trait for unicode strings.

**description**

A trait for unicode strings.

**hostname**

A trait for unicode strings.

**label**

A trait for unicode strings.

**mpiprocs\_per\_machine**

An int trait.

**mpirun\_command**

A trait for unicode strings.

**num\_cores\_per\_mpiproc**

An int trait.

**port**

An int trait.

**prepend\_text**

A trait for unicode strings.

**proxy\_command**

A trait for unicode strings.

**proxy\_hostname**

A trait for unicode strings.

**proxy\_username**

A trait for unicode strings.

**queue\_name**

A trait for unicode strings.

**safe\_interval**

A float trait.

**scheduler**

A trait for unicode strings.

**shebang**

A trait for unicode strings.

**transport**

A trait for unicode strings.

**update** (*\_=None*)**update\_computers** (*\_=None*)**update\_settings** (*\_=None*)

Read settings from the YAML files and populate self.database with them.

**use\_login\_shell**

A boolean (True, False) trait.

**work\_dir**

A trait for unicode strings.

**class** aiidalab\_widgets\_base.**ComputerDropdown** (*\*\*kwargs*)

Bases: ipywidgets.widgets.widget\_box.VBox

Widget to select a configured computer.

**Attributes:**

**selected\_computer(Unicode or Computer): Trait that points to the selected Computer instance.**

It can be set either to an AiiDA Computer instance or to a computer label (will automatically be replaced by the corresponding Computer instance). It is linked to the 'value' trait of *self.\_dropdown* widget.

**computers(Dict): Trait that contains a dictionary (label => Computer instance) for all computers found in the AiiDA database. It is linked to the 'options' trait of *self.\_dropdown* widget.**

**allow\_select\_disabled(Bool): Trait that defines whether to show disabled computers.**

**\_\_annotations\_\_ = {}**

**\_\_init\_\_** (*description='Select computer:', path\_to\_root='./', \*\*kwargs*)

Dropdown for configured AiiDA Computers.

**description** (str): Text to display before dropdown.

**path\_to\_root** (str): Path to the app's root folder.

**\_\_module\_\_ = 'aiidalab\_widgets\_base.computers'**

**\_get\_computers** ()

Get the list of available computers.

**\_trait\_default\_generators = {}**

**\_validate\_selected\_computer**

**allow\_select\_disabled**

A boolean (True, False) trait.

**computers**

An instance of a Python dict.

One or more traits can be passed to the constructor to validate the keys and/or values of the dict. If you need more detailed validation, you may use a custom validator method.

Changed in version 5.0: Added `key_trait` for validating dict keys.

Changed in version 5.0: Deprecated ambiguous `trait`, `traits` args in favor of `value_trait`, `per_key_traits`.

**refresh** (*\_=None*)

Refresh the list of configured computers.

**selected\_computer**

A trait type representing a Union type.

**class** `aiidalab_widgets_base.ExportButtonWidget` (*\*\*kwargs*)

Bases: `ipywidgets.widgets.widget_button.Button`

Export Node button.

`__annotations__` = {}

`__init__` (*process, \*\*kwargs*)

Public constructor

`__module__` = `'aiidalab_widgets_base.export'`

`_trait_default_generators` = {}

**export\_aiida\_subgraph** (*change=None*)

Perform export when the button is pressed

**class** `aiidalab_widgets_base.MultiStructureUploadWidget` (*\*\*kwargs*)

Bases: `ipywidgets.widgets.widget_box.VBox`

Class to deal with archives (tar or zip) containing multiple structures.

`__annotations__` = {}

`__init__` (*text='Upload Zip or Tar archive', node\_class=None, \*\*kwargs*)

Upload multiple structures and store them in AiiDA database.

#### Parameters

- **text** (*str*) – Text to display before upload button
- **node\_class** – AiiDA node class for storing the structure. Possible values: ‘Structure-Data’, ‘CifData’ or None (let the user decide). Note: If your workflows require a specific node class, better fix it here.

`__module__` = `'aiidalab_widgets_base.structures_multi'`

`_on_click_store_all` (*change*)

Store all the uploaded structures.

`_on_click_store_selected` (*change*)

`_on_file_upload` (*change*)

Process the archive once it is uplodaded.

`_trait_default_generators` = {}

**change\_structure** ()

**get\_ase** (*filepath*)

Get an ase object containing the structure.

**static get\_description** (*structure\_ase, filepath*)

Get the structure description automatically.

**property node\_class**



```

refresh_view ()
    Refresh the structure view.

select_structure (filepath)
    Perform structure selection.

store_structure (filepath, description=None)
    Store the structure in the AiiDA database.

class aiidalab_widgets_base.NodesTreeWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_output.Output
    A tree widget for the structured representation of a nodes graph.

NODE_TYPE = {<class 'aiida.orm.nodes.process.workflow.workchain.WorkChainNode'>: <cla
PROCESS_STATE_STYLE = {<ProcessState.EXCEPTED: 'excepted'>: 'danger', <ProcessState.F
PROCESS_STATE_STYLE_DEFAULT = 'default'

__annotations__ = {}

__init__ (**kwargs)
    Public constructor

__module__ = 'aiidalab_widgets_base.nodes'

classmethod _build_tree (root)
    Recursively build a tree nodes graph for a given tree node.

_convert_to_tree_nodes (old_nodes, new_nodes)
    Convert nodes into tree nodes while re-using already converted nodes.

classmethod _find_called (root)

classmethod _find_children (root)
    Find all children of the provided AiiDA node.

classmethod _find_outputs (root)

_observe_nodes

_observe_tree_selected_nodes (change)

_refresh_output ()

classmethod _to_tree_node (node, name=None, **kwargs)
    Convert an AiiDA node to a tree node.

_trait_default_generators = {}

_update_tree_node (tree_node)

classmethod _walk_tree (root)
    Breadth-first search of the node tree.

find_node (pk)

nodes
    An instance of a Python tuple.

selected_nodes
    An instance of a Python tuple.

update (_=None)
    Refresh nodes based on the latest state of the root process and its children.

```

```
class aiidalab_widgets_base.OptimadeQueryWidget (**kwargs)
```

Bases: ipywidgets.widgets.widget\_box.VBox

AiiDALab-specific OPTIMADE Query widget

Useful as a widget to integrate with the `aiidalab_widgets_base.structures.StructureManagerWidget`, embedded into applications.

NOTE: *embedded* for `OptimadeQueryFilterWidget` was introduced in `optimade-client` version 2020.11.5.

#### Parameters

- **embedded** (*bool*) – Whether or not to show extra database and provider information. When set to *True*, the extra information will be hidden, this is useful in situations where the widget is used in a Tab or similar, e.g., for the `aiidalab_widgets_base.structures.StructureManagerWidget`.
- **title** (*str*) – Title used for Tab header if employed in `aiidalab_widgets_base.structures.StructureManagerWidget`.

```
__annotations__ = {}
```

```
__init__ (embedded: bool = True, title: Optional[str] = None, **kwargs) → None
```

Public constructor

```
__module__ = 'aiidalab_widgets_base.databases'
```

```
_trait_default_generators = {}
```

```
_update_structure (change: dict) → None
```

New structure chosen

#### structure

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

```
class aiidalab_widgets_base.ProcessCallStackWidget (**kwargs)
```

Bases: ipywidgets.widgets.widget\_string.HTML

Widget that shows process call stack.

```
__annotations__ = {}
```

```
__init__ (title='Process Call Stack', path_to_root='./', **kwargs)
```

Public constructor

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_trait_default_generators = {}
```

```
calc_info (node)
```

Return a string with the summary of the state of a CalculationNode.

#### process

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

```
update ()
```

Update the call stack that is shown.

---

```

class aiidalab_widgets_base.ProcessFollowerWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    A Widget that follows a process until finished.

    __annotations__ = {}

    __init__ (process=None, followers=None, update_interval=0.1, path_to_root='./', **kwargs)
        Initiate all the followers.

    __module__ = 'aiidalab_widgets_base.process'

    _trait_default_generators = {}

    follow (detach=False)
        Initiate following the process with or without blocking.

    on_completed (function)
        Run functions after a process has been completed.

    process
        A trait whose value must be an instance of a specified class.

        The value can also be an instance of a subclass of the specified class.

        Subclasses can declare default classes by overriding the class attribute

    update ()

class aiidalab_widgets_base.ProcessInputsWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    Widget to select and show process inputs.

    __annotations__ = {}

    __init__ (process=None, **kwargs)
        Public constructor

    __module__ = 'aiidalab_widgets_base.process'

    _trait_default_generators = {}

    process
        A trait whose value must be an instance of a specified class.

        The value can also be an instance of a subclass of the specified class.

        Subclasses can declare default classes by overriding the class attribute

    show_selected_input (change=None)
        Function that displays process inputs selected in the inputs Dropdown widget.

class aiidalab_widgets_base.ProcessListWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    List of AiiDA processes.

    past_days (int): Simulations that were submitted in the last past_days.

    incoming_node (int, str, Node): Trait that takes node id or uuid and returns the node that must be among the
    input nodes of the process of interest.

    outgoing_node (int, str, Node): Trait that takes node id or uuid and returns the node that must be among the
    output nodes of the process of interest.

    process_states (list): List of allowed process states.

```

`process_label` (str): Show process states of type *process\_label*.

`description_contains` (str): string that should be present in the description of a process node.

```
__annotations__ = {}
```

```
__init__ (path_to_root= './', **kwargs)  
    Public constructor
```

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_default_process_label
```

```
_follow (update_interval)
```

```
_trait_default_generators = {'process_label': <traitlets.traitlets.DefaultHandler obj
```

```
_validate_incoming_node
```

```
_validate_outgoing_node
```

```
_validate_process_label
```

```
description_contains  
    A trait for unicode strings.
```

```
incoming_node  
    A trait type representing a Union type.
```

```
outgoing_node  
    A trait type representing a Union type.
```

```
past_days  
    An int trait.
```

```
process_label  
    A trait for unicode strings.
```

```
process_states  
    An instance of a Python list.
```

```
start_autoupdate (update_interval=10)
```

```
update (_=None)  
    Perform the query.
```

```
class aiidalab_widgets_base.ProcessMonitor (**kwargs)
```

```
    Bases: traitlets.traitlets.HasTraits
```

```
    Monitor a process and execute callback functions at specified intervals.
```

```
__annotations__ = {}
```

```
__init__ (callbacks=None, on_sealed=None, timeout=None, **kwargs)  
    Initialize self. See help(type(self)) for accurate signature.
```

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_monitor_process (process_id)
```

```
_observe_process
```

```
_trait_default_generators = {}
```

```
join ()
```

**process**

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

```
class aiidalab_widgets_base.ProcessNodesTreeWidget (**kwargs)
```

Bases: ipywidgets.widgets.widget\_box.VBox

A tree widget for the structured representation of a process graph.

```
__annotations__ = {}
```

```
__init__ (title='Process Tree', **kwargs)
```

Public constructor

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_observe_process
```

```
_observe_tree_selected_nodes (change)
```

```
_trait_default_generators = {}
```

**process**

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

**selected\_nodes**

An instance of a Python tuple.

```
update (_=None)
```

```
class aiidalab_widgets_base.ProcessOutputsWidget (**kwargs)
```

Bases: ipywidgets.widgets.widget\_box.VBox

Widget to select and show process outputs.

```
__annotations__ = {}
```

```
__init__ (process=None, **kwargs)
```

Public constructor

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_trait_default_generators = {}
```

**process**

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

```
show_selected_output (change=None)
```

Function that displays process output selected in the *outputs* Dropdown widget.

```
class aiidalab_widgets_base.ProcessReportWidget (**kwargs)
```

Bases: ipywidgets.widgets.widget\_string.HTML

Widget that shows process report.

```
__annotations__ = {}
```

```
__init__ (title='Process Report', **kwargs)
    Public constructor
```

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_trait_default_generators = {}
```

**process**

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

**update ()**

Update report that is shown.

```
class aiidalab_widgets_base.ProgressBarWidget (**kwargs)
```

```
    Bases: ipywidgets.widgets.widget_box.VBox
```

A bar showing the proggress of a process.

```
__annotations__ = {}
```

```
__init__ (title='Progress Bar', **kwargs)
    Initialize ProgressBarWidget.
```

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_trait_default_generators = {}
```

**property current\_state**

**process**

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

**update ()**

Update the bar.

```
class aiidalab_widgets_base.RunningCalcJobOutputWidget (**kwargs)
```

```
    Bases: ipywidgets.widgets.widget_box.VBox
```

Show an output of selected running child calculation.

```
__annotations__ = {}
```

```
__init__ (title='Running Job Output', **kwargs)
    Public constructor
```

```
__module__ = 'aiidalab_widgets_base.process'
```

```
_trait_default_generators = {}
```

**process**

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

**update ()**

Update the displayed output.

---

```

class aiidalab_widgets_base.SmilesWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    Conver SMILES into 3D structure.

    SPINNER = '<i class="fa fa-spinner fa-pulse" style="color:red;" ></i>'

    __annotations__ = {}

    __init__ (title='')
        Public constructor

    __module__ = 'aiidalab_widgets_base.structures'

    __default_structure

    __on_button_pressed (change)
        Convert SMILES to ase structure when button is pressed.

    __pybel_opt (smile, steps)
        Optimize a molecule using force field and pybel (needed for complex SMILES).

    __rdkit_opt (smile, steps)
        Optimize a molecule using force field and rdkit (needed for complex SMILES).

    __trait_default_generators = {'structure': <traitlets.traitlets.DefaultHandler object>

    make_ase (species, positions)
        Create ase Atoms object.

    mol_from_smiles (smile, steps=10000)
        Convert SMILES to ase structure try rdkit then pybel

    structure
        A trait whose value must be an instance of a specified class.

        The value can also be an instance of a subclass of the specified class.

        Subclasses can declare default classes by overriding the class attribute

class aiidalab_widgets_base.SshComputerSetup (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    Setup password-free access to a computer.

    __annotations__ = {}

    __init__ (**kwargs)
        Public constructor

    __module__ = 'aiidalab_widgets_base.computers'

    property __password
        Returning the password and immediately destroying it

    property __private_key
        unwrap private key file and setting filename and file content

    property __proxy_password
        Returning the password and immediately destroying it

    static __add_private_key (private_key_fname, private_key_content)
        param private_key_fname: string param private_key_content: bytes

    __configure_proxy (password, proxy_password)
        Configure proxy server.

```

**`_make_host_known`** (*hostname*, *proxycmd=None*)  
Add host information into known\_hosts file.

**`_observe_proxy_hostname`**

**`_observe_proxy_username`**

**`_on_setup_ssh`** (*mode*, *change*)  
ATTENTION: modifying the order of operations in this function can lead to unexpected problems

**`static _send_pubkey`** (*hostname*, *username*, *password*, *proxycmd=""*)  
Send a public key to a remote host.

**`static _ssh_keygen`** ()  
Generate ssh key pair.

**`_trait_default_generators`** = {}

**`_validate_port`**

**`_write_ssh_config`** (*proxycmd=""*, *private\_key\_abs\_fname=None*)  
Put host information into the config file.

**`can_login`** (*silent=False*)  
Check if it is possible to login into the remote host.

**`hostname`**  
A trait for unicode strings.

**`is_host_known`** (*hostname=None*)  
Check if the host is known already.

**`is_in_config`** ()  
Check if the config file contains host information.

**`on_setup_ssh`** (*change*)  
Setup ssh, password and private key are supported

**`on_use_diff_proxy_username_change`** (*change*)  
If using different username for proxy check-box is clicked.

**`on_use_proxy_change`** (*change*)  
If proxy check-box is clicked.

**`on_use_verification_mode_change`** (*change*)  
which verification mode is chosen.

**`port`**  
A trait type representing a Union type.

**`proxy_hostname`**  
A trait for unicode strings.

**`proxy_username`**  
A trait for unicode strings.

**`setup_counter`**  
An int trait.

**`use_proxy`**  
A boolean (True, False) trait.

**`username`**  
A trait for unicode strings.



---

```

class aiidalab_widgets_base.StructureBrowserWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    Class to query for structures stored in the AiiDA database.

    __annotations__ = {}

    __init__ (title='')
        Public constructor

    __module__ = 'aiidalab_widgets_base.structures'

    _on_select_structure (_=None)

    _trait_default_generators = {}

    preprocess ()
        Search structures in AiiDA database and add formula extra to them.

    search (_=None)
        Launch the search of structures in AiiDA database.

    structure
        A trait type representing a Union type.

class aiidalab_widgets_base.StructureExamplesWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    Class to provide example structures for selection.

    __annotations__ = {}

    __init__ (examples, title='', **kwargs)
        Public constructor

    __module__ = 'aiidalab_widgets_base.structures'

    _default_structure

    _on_select_structure (change)
        When structure is selected.

    _trait_default_generators = {'structure': <traitlets.traitlets.DefaultHandler object>

    static get_example_structures (examples)
        Get the list of example structures.

    structure
        A trait whose value must be an instance of a specified class.

        The value can also be an instance of a subclass of the specified class.

        Subclasses can declare default classes by overriding the class attribute

class aiidalab_widgets_base.StructureManagerWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    Upload a structure and store it in AiiDA database.

    Attributes: structure(Atoms): trait that contains the selected structure. 'None' if no structure is selected. structure_node(StructureData, CifData): trait that contains AiiDA structure object node_class(str): trait that contains structure_node type (as string).

    SUPPORTED_DATA_FORMATS = {'CifData': 'cif', 'StructureData': 'structure'}

    __annotations__ = {}

```

`__init__` (*importers*, *viewer=None*, *editors=None*, *storable=True*, *node\_class=None*, *\*\*kwargs*)

**Arguments:**

**importers(list): list of tuples each containing the displayed name of importer and the importer object.** Each object should contain 'structure' trait pointing to the imported structure. The trait will be linked to 'structure' trait of this class.

**storable(bool):** Whether to provide Store button (together with Store format)

**node\_class(str): AiiDA node class for storing the structure.** Possible values: 'StructureData', 'CifData' or None (let the user decide). Note: If your workflows require a specific node class, better fix it here.

`__module__ = 'aiidalab_widgets_base.structures'`

`_change_structure_node`

`_convert_to_structure_node` (*structure*)

Convert structure of any type to the StructureNode object.

`_default_node_class`

`_observe_input_structure`

`_observe_structure_node`

`_structure_changed`

`_structure_editors` (*editors*)

Preparing structure editors.

`_structure_importers` (*importers*)

Preparing structure importers.

`_sync_structure_node` ()

Synchronize the structure\_node trait using the currently provided info.

`_trait_default_generators = {}`

`input_structure`

A trait type representing a Union type.

`node_class`

A trait for unicode strings.

`store_structure` (*\_=None*)

Stores the structure in AiiDA database.

`structure`

A trait type representing a Union type.

`structure_node`

A trait whose value must be an instance of a specified class.

The value can also be an instance of a subclass of the specified class.

Subclasses can declare default classes by overriding the class attribute

`undo` (\_)

Undo modifications.

**class** `aiidalab_widgets_base.StructureUploadWidget` (*\*\*kwargs*)

Bases: `ipywidgets.widgets.widget_box.VBox`

Class that allows to upload structures from user's computer.

---

```

__annotations__ = {}

__init__ (title="", description='Upload Structure')
    Public constructor

__module__ = 'aiidalab_widgets_base.structures'

__on_file_upload (change=None)
    When file upload button is pressed.

__trait_default_generators = {}

__validate_and_fix_ase_cell (ase_structure, vacuum_ang=10.0)
    Checks if the ase Atoms object has a cell set, otherwise sets it to bounding box plus specified "vacuum"
    space

structure
    A trait type representing a Union type.

class aiidalab_widgets_base.SubmitButtonWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox
    Submit button class that creates submit button jupyter widget.

    __annotations__ = {}

    __init__ (process_class, inputs_generator=None, input_dictionary_function=None, descrip-
        tion='Submit', disable_after_submit=True, append_output=False, **kwargs)
        Submit Button widget.

        process_class (Process): Process class to submit.

        inputs_generator (func): Function that returns inputs dictionary or inputs builder.

        input_dictionary_function (DEPRECATED): Function that generates input parameters dictionary.

        description (str): Description written on the submission button.

        disable_after_submit (bool): Whether to disable the button after the process was submitted.

        append_output (bool): Whether to clear widget output for each subsequent submission.

    __module__ = 'aiidalab_widgets_base.process'

    __trait_default_generators = {}

    on_btn_submit_press (_=None)
        When submit button is pressed.

    on_click (function)

    on_submitted (function)
        Run functions after a process has been submitted successfully.

process
    A trait whose value must be an instance of a specified class.

    The value can also be an instance of a subclass of the specified class.

    Subclasses can declare default classes by overriding the class attribute

class aiidalab_widgets_base.WizardAppWidget (**kwargs)
    Bases: ipywidgets.widgets.widget_box.VBox

    ICONS = {<State.INIT: 0>: '', <State.READY: 2>: '', <State.CONFIGURED: 1>: '', <
    ICON_SEPARATOR = '\u2000'

```

```

__annotations__ = {}

__init__(steps, **kwargs)
    Public constructor

__module__ = 'aiidalab_widgets_base.wizard'

__consider_auto_advance(=None)
    Determine whether the app should automatically advance to the next step.

    This is performed whenever the current step is within the SUCCESS state and has the auto_advance attribute set to True.

__observe_selected_index

__on_click_back_button(_)

__on_click_next_button(_)

__on_click_reset_button(_)

__trait_default_generators = {}

__update_buttons()

__update_step_state(_)

__update_titles()

can_reset()

classmethod icons()
    Return the icon set and return animated icons based on the current time stamp.

reset(step=0)
    Reset the app up to the given step.

    For example, with step=0 (the default), the whole app is reset. With step=1, all but the first step are reset.

selected_index
    An int trait.

```

```

class aiidalab_widgets_base.WizardAppWidgetStep(**kwargs)
    Bases: traitlets.traitlets.HasTraits

```

One step of a WizardAppWidget.

```

class State(value)
    Bases: enum.Enum

```

Each step is always in one specific state.

The state is used to determine:

- 1) how the step is visually presented to the user, and
- 2) whether the next step is accessible (i.e. reached the SUCCESS state).

App developers are encouraged to use the step states to couple application logic and interface. In general, all widget changes should trigger a re-evaluation of the step state, and states also determine whether certain widgets are enabled or disabled.

A step can be in one of the following states:

INIT: The initial state, usually all widgets disabled. READY: The step (widget) is ready for user input (some or all widgets enabled). CONFIGURED: The step is in a consistent configuration awaiting confirmation. ACTIVE: The step is carrying out a runtime operation. SUCCESS: A

configuration has been confirmed / a runtime operation successfully finished. FAIL: A runtime operation has failed in an unrecoverable way.

Not all steps must implement all states, for example:

- the first step does not need an INIT state
- a step without runtime process should not have an ACTIVE or FAIL state
- a “review & confirm” step does not require a READY state.
- a step without configuration options (e.g. pure “review & confirm” step)

Important: The next step is only accessible if the current step is within the SUCCESS state!

**ACTIVE = 3**

**CONFIGURED = 1**

**FAIL = -1**

**INIT = 0**

**READY = 2**

**SUCCESS = 4**

**\_\_annotations\_\_ = {}**

**\_\_module\_\_ = 'aiidalab\_widgets\_base.wizard'**

**\_\_annotations\_\_ = {}**

**\_\_module\_\_ = 'aiidalab\_widgets\_base.wizard'**

**\_trait\_default\_generators = {}**

**auto\_advance**

A boolean (True, False) trait.

**can\_reset ()**

**state**

Use a Enum class as model for the data type description. Note that if no default-value is provided, the first enum-value is used as default-value.

```
# -- SINCE: Python 3.4 (or install backport: pip install enum34)
import enum
from traitlets import HasTraits, UseEnum

class Color(enum.Enum):
    red = 1          # -- IMPLICIT: default_value
    blue = 2
    green = 3

class MyEntity(HasTraits):
    color = UseEnum(Color, default_value=Color.blue)

entity = MyEntity(color=Color.red)
entity.color = Color.green      # USE: Enum-value (preferred)
entity.color = "green"         # USE: name (as string)
entity.color = "Color.green"   # USE: scoped-name (as string)
entity.color = 3                # USE: number (as int)
assert entity.color is Color.green
```

`aiidalab_widgets_base.register_viewer_widget` (*key*)

Register widget as a viewer for the given key.

`aiidalab_widgets_base.viewer` (*obj*, *downloadable=True*, *\*\*kwargs*)

Display AiiDA data types in Jupyter notebooks.

**Parameters** `downloadable` (*bool*) – If True, add link/button to download the content of displayed AiiDA object.

Returns the object itself if the viewer wasn't found.

`aiidalab-widgets-base` package is released under the MIT license.

## INDICES AND TABLES

- [genindex](#)
- [modindex](#)
- [search](#)

### 5.1 How to cite

Users of AiiDA lab are kindly asked to cite the following publication in their own work:

- A. V. Yakutovich et al., *Comp. Mat. Sci.* 188, 110165 (2021). DOI:[10.1016/j.commatsci.2020.110165](https://doi.org/10.1016/j.commatsci.2020.110165)

### 5.2 Acknowledgements

This work is supported by the [MARVEL National Centre for Competency in Research](#) funded by the [Swiss National Science Foundation](#), as well as by the [MaX European Centre of Excellence](#) funded by the [Horizon 2020 EINFRA-5](#) program, Grant No. 676598.





## PYTHON MODULE INDEX

### a

aiidalab\_widgets\_base, 37  
aiidalab\_widgets\_base.codes, 10  
aiidalab\_widgets\_base.computers, 11  
aiidalab\_widgets\_base.data, 9  
aiidalab\_widgets\_base.databases, 15  
aiidalab\_widgets\_base.dicts, 18  
aiidalab\_widgets\_base.export, 18  
aiidalab\_widgets\_base.misc, 18  
aiidalab\_widgets\_base.nodes, 19  
aiidalab\_widgets\_base.process, 21  
aiidalab\_widgets\_base.structures, 26  
aiidalab\_widgets\_base.structures\_multi,  
31  
aiidalab\_widgets\_base.utils, 9  
aiidalab\_widgets\_base.viewers, 31  
aiidalab\_widgets\_base.wizard, 34



## Symbols

<code>__annotations__</code>	(ai- idalab_widgets_base.AiiDACodeSetup tribute), 37	<code>__annotations__</code>	(ai- idalab_widgets_base.ProcessFollowerWidget tribute), 47
<code>__annotations__</code>	(ai- idalab_widgets_base.AiidaComputerSetup tribute), 37	<code>__annotations__</code>	(ai- idalab_widgets_base.ProcessInputsWidget tribute), 47
<code>__annotations__</code>	(ai- idalab_widgets_base.BasicStructureEditor tribute), 38	<code>__annotations__</code>	(ai- idalab_widgets_base.ProcessListWidget tribute), 48
<code>__annotations__</code>	(ai- idalab_widgets_base.CodQueryWidget tribute), 39	<code>__annotations__</code>	(ai- idalab_widgets_base.ProcessMonitor tribute), 48
<code>__annotations__</code>	(ai- idalab_widgets_base.CodeDatabaseWidget tribute), 40	<code>__annotations__</code>	(ai- idalab_widgets_base.ProcessNodesTreeWidget tribute), 49
<code>__annotations__</code>	(ai- idalab_widgets_base.CodeDropdown tribute), 41	<code>__annotations__</code>	(ai- idalab_widgets_base.ProcessOutputsWidget tribute), 49
<code>__annotations__</code>	(ai- idalab_widgets_base.ComputerDatabaseWidget tribute), 42	<code>__annotations__</code>	(ai- idalab_widgets_base.ProcessReportWidget tribute), 49
<code>__annotations__</code>	(ai- idalab_widgets_base.ComputerDropdown tribute), 43	<code>__annotations__</code>	(ai- idalab_widgets_base.ProgressBarWidget tribute), 50
<code>__annotations__</code>	(ai- idalab_widgets_base.ExportButtonWidget tribute), 44	<code>__annotations__</code>	(ai- idalab_widgets_base.RunningCalcJobOutputWidget tribute), 50
<code>__annotations__</code>	(ai- idalab_widgets_base.MultiStructureUploadWidget tribute), 44	<code>__annotations__</code>	(ai- idalab_widgets_base.SmilesWidget tribute), 51
<code>__annotations__</code>	(ai- idalab_widgets_base.NodesTreeWidget tribute), 45	<code>__annotations__</code>	(ai- idalab_widgets_base.SshComputerSetup tribute), 51
<code>__annotations__</code>	(ai- idalab_widgets_base.OptimadeQueryWidget tribute), 46	<code>__annotations__</code>	(ai- idalab_widgets_base.StructureBrowserWidget tribute), 53
<code>__annotations__</code>	(ai- idalab_widgets_base.ProcessCallStackWidget tribute), 46	<code>__annotations__</code>	(ai- idalab_widgets_base.StructureExamplesWidget tribute), 53
		<code>__annotations__</code>	(ai- idalab_widgets_base.StructureManagerWidget tribute), 53

`__annotations__` (*aiidalab\_widgets\_base.StructureUploadWidget* attribute), 54  
`__annotations__` (*aiidalab\_widgets\_base.SubmitButtonWidget* attribute), 55  
`__annotations__` (*aiidalab\_widgets\_base.WizardAppWidget* attribute), 55  
`__annotations__` (*aiidalab\_widgets\_base.WizardAppWidgetStep* attribute), 57  
`__annotations__` (*aiidalab\_widgets\_base.WizardAppWidgetStep.State* attribute), 57  
`__dict__` (*aiidalab\_widgets\_base.misc.ReversePolishNotation* attribute), 18  
`__init__` () (*aiidalab\_widgets\_base.AiiDACodeSetup* method), 37  
`__init__` () (*aiidalab\_widgets\_base.AiidaComputerSetup* method), 37  
`__init__` () (*aiidalab\_widgets\_base.BasicStructureEditor* method), 38  
`__init__` () (*aiidalab\_widgets\_base.CodQueryWidget* method), 39  
`__init__` () (*aiidalab\_widgets\_base.CodeDatabaseWidget* method), 40  
`__init__` () (*aiidalab\_widgets\_base.CodeDropdown* method), 41  
`__init__` () (*aiidalab\_widgets\_base.ComputerDatabaseWidget* method), 42  
`__init__` () (*aiidalab\_widgets\_base.ComputerDropdown* method), 43  
`__init__` () (*aiidalab\_widgets\_base.ExportButtonWidget* method), 44  
`__init__` () (*aiidalab\_widgets\_base.MultiStructureUploadWidget* method), 44  
`__init__` () (*aiidalab\_widgets\_base.NodesTreeWidget* method), 45  
`__init__` () (*aiidalab\_widgets\_base.OptimadeQueryWidget* method), 46  
`__init__` () (*aiidalab\_widgets\_base.ProcessCallStackWidget* method), 46  
`__init__` () (*aiidalab\_widgets\_base.ProcessFollowerWidget* method), 47  
`__init__` () (*aiidalab\_widgets\_base.ProcessInputsWidget* method), 47  
`__init__` () (*aiidalab\_widgets\_base.ProcessListWidget* method), 48  
`__init__` () (*aiidalab\_widgets\_base.ProcessMonitor* method), 48  
`__init__` () (*aiidalab\_widgets\_base.ProcessNodesTreeWidget* method), 49  
`__init__` () (*aiidalab\_widgets\_base.ProcessOutputsWidget* method), 49  
`__init__` () (*aiidalab\_widgets\_base.ProcessReportWidget* method), 49  
`__init__` () (*aiidalab\_widgets\_base.ProgressBarWidget* method), 50  
`__init__` () (*aiidalab\_widgets\_base.RunningCalc.JobOutputWidget* method), 50  
`__init__` () (*aiidalab\_widgets\_base.SmilesWidget* method), 51  
`__init__` () (*aiidalab\_widgets\_base.SshComputerSetup* method), 51  
`__init__` () (*aiidalab\_widgets\_base.StructureBrowserWidget* method), 53  
`__init__` () (*aiidalab\_widgets\_base.StructureExamplesWidget* method), 53  
`__init__` () (*aiidalab\_widgets\_base.StructureManagerWidget* method), 53  
`__init__` () (*aiidalab\_widgets\_base.StructureUploadWidget* method), 55  
`__init__` () (*aiidalab\_widgets\_base.SubmitButtonWidget* method), 55  
`__init__` () (*aiidalab\_widgets\_base.WizardAppWidget* method), 56  
`__init__` () (*aiidalab\_widgets\_base.codes.AiiDACodeSetup* method), 10  
`__init__` () (*aiidalab\_widgets\_base.codes.CodeDropdown* method), 11  
`__init__` () (*aiidalab\_widgets\_base.computers.AiidaComputerSetup* method), 11  
`__init__` () (*aiidalab\_widgets\_base.computers.ComputerDropdown* method), 12  
`__init__` () (*aiidalab\_widgets\_base.computers.SshComputerSetup* method), 13  
`__init__` () (*aiidalab\_widgets\_base.data.LigandSelectorWidget* method), 9  
`__init__` () (*aiidalab\_widgets\_base.databases.CodQueryWidget* method), 15  
`__init__` () (*aiidalab\_widgets\_base.databases.CodeDatabaseWidget* method), 15  
`__init__` () (*aiidalab\_widgets\_base.databases.ComputerDatabaseWidget* method), 16  
`__init__` () (*aiidalab\_widgets\_base.databases.OptimadeQueryWidget* method), 17  
`__init__` () (*aiidalab\_widgets\_base.export.ExportButtonWidget* method), 18  
`__init__` () (*aiidalab\_widgets\_base.misc.CopyToClipboardButton* method), 18  
`__init__` () (*aiidalab\_widgets\_base.misc.ReversePolishNotation* method), 18  
`__init__` () (*aiidalab\_widgets\_base.nodes.AiidaNodeTreeNode* method), 19  
`__init__` () (*aiidalab\_widgets\_base.nodes.AiidaOutputsTreeNode* method), 19  
`__init__` () (*aiidalab\_widgets\_base.nodes.AiidaProcessTreeNode* method), 19

method), 19	attribute), 37
__init__ () (aiidalab_widgets_base.nodes.NodesTreeWidget module__ (aiidalab_widgets_base.AiidaComputerSetup method), 20	attribute), 37
__init__ () (aiidalab_widgets_base.process.CalcJobOutputWidget module__ (aiidalab_widgets_base.BasicStructureEditor method), 21	attribute), 38
__init__ () (aiidalab_widgets_base.process.ProcessCallStackWidget module__ (aiidalab_widgets_base.CodQueryWidget method), 22	attribute), 40
__init__ () (aiidalab_widgets_base.process.ProcessFollowerWidget module__ (aiidalab_widgets_base.CodeDatabaseWidget method), 22	attribute), 40
__init__ () (aiidalab_widgets_base.process.ProcessInputsWidget module__ (aiidalab_widgets_base.CodeDropdown method), 22	attribute), 41
__init__ () (aiidalab_widgets_base.process.ProcessListWidget module__ (aiidalab_widgets_base.ComputerDatabaseWidget method), 23	attribute), 42
__init__ () (aiidalab_widgets_base.process.ProcessMonitor module__ (aiidalab_widgets_base.ComputerDropdown method), 24	attribute), 43
__init__ () (aiidalab_widgets_base.process.ProcessNodesTreeWidget module__ (aiidalab_widgets_base.ExportButtonWidget method), 24	attribute), 44
__init__ () (aiidalab_widgets_base.process.ProcessOutputsWidget module__ (aiidalab_widgets_base.MultiStructureUploadWidget method), 24	attribute), 44
__init__ () (aiidalab_widgets_base.process.ProcessReportWidget module__ (aiidalab_widgets_base.NodesTreeWidget method), 25	attribute), 45
__init__ () (aiidalab_widgets_base.process.ProgressBarWidget module__ (aiidalab_widgets_base.OptimadeQueryWidget method), 25	attribute), 46
__init__ () (aiidalab_widgets_base.process.RunningCalcJobOutputWidget module__ (aiidalab_widgets_base.ProcessCallStackWidget method), 25	attribute), 46
__init__ () (aiidalab_widgets_base.process.SubmitButtonWidget module__ (aiidalab_widgets_base.ProcessFollowerWidget method), 26	attribute), 47
__init__ () (aiidalab_widgets_base.structures.BasicStructureEditor module__ (aiidalab_widgets_base.ProcessInputsWidget method), 26	attribute), 47
__init__ () (aiidalab_widgets_base.structures.SmilesWidget module__ (aiidalab_widgets_base.ProcessListWidget method), 28	attribute), 48
__init__ () (aiidalab_widgets_base.structures.StructureBrowserWidget module__ (aiidalab_widgets_base.ProcessMonitor method), 28	attribute), 48
__init__ () (aiidalab_widgets_base.structures.StructureExamplesWidget module__ (aiidalab_widgets_base.ProcessNodesTreeWidget method), 29	attribute), 49
__init__ () (aiidalab_widgets_base.structures.StructureManagerWidget module__ (aiidalab_widgets_base.ProcessOutputsWidget method), 29	attribute), 49
__init__ () (aiidalab_widgets_base.structures.StructureUploadWidget module__ (aiidalab_widgets_base.ProcessReportWidget method), 30	attribute), 50
__init__ () (aiidalab_widgets_base.structures_multi.MultiStructureUploadWidget module__ (aiidalab_widgets_base.ProgressBarWidget method), 31	attribute), 50
__init__ () (aiidalab_widgets_base.viewers.BandsDataViewer module__ (aiidalab_widgets_base.RunningCalcJobOutputWidget method), 31	attribute), 50
__init__ () (aiidalab_widgets_base.viewers.DictViewer module__ (aiidalab_widgets_base.SmilesWidget method), 32	attribute), 51
__init__ () (aiidalab_widgets_base.viewers.FolderDataViewer module__ (aiidalab_widgets_base.SshComputerSetup method), 32	attribute), 51
__init__ () (aiidalab_widgets_base.viewers.StructureDataViewer module__ (aiidalab_widgets_base.StructureBrowserWidget method), 32	attribute), 53
__init__ () (aiidalab_widgets_base.viewers._StructureDatabaseViewer module__ (aiidalab_widgets_base.StructureExamplesWidget method), 33	attribute), 53
__init__ () (aiidalab_widgets_base.wizard.WizardAppWidget module__ (aiidalab_widgets_base.StructureManagerWidget method), 34	attribute), 54
__module__ (aiidalab_widgets_base.AiiDACodeSetup module__ (aiidalab_widgets_base.StructureUploadWidget	

attribute), 55	attribute), 22
__module__ (aiidalab_widgets_base.SubmitButtonWidget	__module__ (aiidalab_widgets_base.process.ProcessFollowerWidget
attribute), 55	attribute), 22
__module__ (aiidalab_widgets_base.WizardAppWidget	__module__ (aiidalab_widgets_base.process.ProcessInputsWidget
attribute), 56	attribute), 22
__module__ (aiidalab_widgets_base.WizardAppWidgetStep	__module__ (aiidalab_widgets_base.process.ProcessListWidget
attribute), 57	attribute), 23
__module__ (aiidalab_widgets_base.WizardAppWidgetStep.State	__module__ (aiidalab_widgets_base.process.ProcessMonitor
attribute), 57	attribute), 24
__module__ (aiidalab_widgets_base.codes.AiiDACodeSetup	__module__ (aiidalab_widgets_base.process.ProcessNodesTreeWidget
attribute), 10	attribute), 24
__module__ (aiidalab_widgets_base.codes.CodeDropdown	__module__ (aiidalab_widgets_base.process.ProcessOutputsWidget
attribute), 11	attribute), 24
__module__ (aiidalab_widgets_base.computers.AiidaComputerSetup	__module__ (aiidalab_widgets_base.process.ProcessReportWidget
attribute), 11	attribute), 25
__module__ (aiidalab_widgets_base.computers.ComputerDropdown	__module__ (aiidalab_widgets_base.process.ProgressBarWidget
attribute), 13	attribute), 25
__module__ (aiidalab_widgets_base.computers.SshComputerSetup	__module__ (aiidalab_widgets_base.process.RunningCalcJobOutputWidget
attribute), 13	attribute), 25
__module__ (aiidalab_widgets_base.data.LigandSelectorWidget	__module__ (aiidalab_widgets_base.process.SubmitButtonWidget
attribute), 9	attribute), 26
__module__ (aiidalab_widgets_base.databases.CodQueryWidget	__module__ (aiidalab_widgets_base.structures.BasicStructureEditor
attribute), 15	attribute), 26
__module__ (aiidalab_widgets_base.databases.CodeDatabaseWidget	__module__ (aiidalab_widgets_base.structures.SmilesWidget
attribute), 15	attribute), 28
__module__ (aiidalab_widgets_base.databases.ComputerDatabaseWidget	__module__ (aiidalab_widgets_base.structures.StructureBrowserWidget
attribute), 16	attribute), 28
__module__ (aiidalab_widgets_base.databases.OptimadeQueryWidget	__module__ (aiidalab_widgets_base.structures.StructureExamplesWidget
attribute), 17	attribute), 29
__module__ (aiidalab_widgets_base.export.ExportButtonWidget	__module__ (aiidalab_widgets_base.structures.StructureManagerWidget
attribute), 18	attribute), 29
__module__ (aiidalab_widgets_base.misc.CopyToClipboardButton	__module__ (aiidalab_widgets_base.structures.StructureUploadWidget
attribute), 18	attribute), 30
__module__ (aiidalab_widgets_base.misc.ReversePolishNotation	__module__ (aiidalab_widgets_base.structures_multi.MultiStructureUpload
attribute), 18	attribute), 31
__module__ (aiidalab_widgets_base.nodes.AiidaNodeTreeNode	__module__ (aiidalab_widgets_base.viewers.BandsDataViewer
attribute), 19	attribute), 32
__module__ (aiidalab_widgets_base.nodes.AiidaOutputsTreeNode	__module__ (aiidalab_widgets_base.viewers.DictViewer
attribute), 19	attribute), 32
__module__ (aiidalab_widgets_base.nodes.AiidaProcessTreeNode	__module__ (aiidalab_widgets_base.viewers.FolderDataViewer
attribute), 19	attribute), 32
__module__ (aiidalab_widgets_base.nodes.CalcFunctionTreeNode	__module__ (aiidalab_widgets_base.viewers.StructureDataViewer
attribute), 20	attribute), 32
__module__ (aiidalab_widgets_base.nodes.CalcJobTreeNode	__module__ (aiidalab_widgets_base.viewers._StructureDataBaseViewer
attribute), 20	attribute), 33
__module__ (aiidalab_widgets_base.nodes.NodesTreeWidget	__module__ (aiidalab_widgets_base.wizard.WizardAppWidget
attribute), 20	attribute), 35
__module__ (aiidalab_widgets_base.nodes.UnknownTypeTreeNode	__module__ (aiidalab_widgets_base.wizard.WizardAppWidgetStep
attribute), 21	attribute), 36
__module__ (aiidalab_widgets_base.nodes.WorkChainProcessTreeNode	__module__ (aiidalab_widgets_base.wizard.WizardAppWidgetStep.State
attribute), 21	attribute), 36
__module__ (aiidalab_widgets_base.process.CalcJobOutputWidget	word() (aiidalab_widgets_base.SshComputerSetup
attribute), 21	property), 51
__module__ (aiidalab_widgets_base.process.ProcessCallStackWidget	word() (aiidalab_widgets_base.computers.SshComputerSetup

<code>property)</code> , 13		<code>_consider_auto_advance()</code>	(ai- <i>idalab_widgets_base.wizard.WizardAppWidget</i> method), 35
<code>__private_key()</code>	(ai- <i>idalab_widgets_base.SshComputerSetup</i> property), 51	<code>_convert_to_structure_node()</code>	(ai- <i>idalab_widgets_base.StructureManagerWidget</i> method), 54
<code>__private_key()</code>	(ai- <i>idalab_widgets_base.computers.SshComputerSetup</i> property), 13	<code>_convert_to_structure_node()</code>	(ai- <i>idalab_widgets_base.structures.StructureManagerWidget</i> method), 29
<code>__proxy_password()</code>	(ai- <i>idalab_widgets_base.SshComputerSetup</i> property), 51	<code>_convert_to_tree_nodes()</code>	(ai- <i>idalab_widgets_base.NodesTreeWidget</i> method), 45
<code>__proxy_password()</code>	(ai- <i>idalab_widgets_base.computers.SshComputerSetup</i> property), 13	<code>_convert_to_tree_nodes()</code>	(ai- <i>idalab_widgets_base.nodes.NodesTreeWidget</i> method), 20
<code>__weakref__</code> ( <i>aiidalab_widgets_base.misc.ReversePolishNotation</i> attribute), 18		<code>_default_node_class</code>	(ai- <i>idalab_widgets_base.StructureManagerWidget</i> attribute), 54
<code>_add_private_key()</code>	(ai- <i>idalab_widgets_base.SshComputerSetup</i> static method), 51	<code>_default_node_class</code>	(ai- <i>idalab_widgets_base.structures.StructureManagerWidget</i> attribute), 29
<code>_add_private_key()</code>	(ai- <i>idalab_widgets_base.computers.SshComputerSetup</i> static method), 13	<code>_default_openend</code>	(ai- <i>idalab_widgets_base.nodes.AiidaNodeTreeNode</i> attribute), 19
<code>_appearance_tab()</code>	(ai- <i>idalab_widgets_base.viewers._StructureDataBaseViewer</i> method), 33	<code>_default_process_label</code>	(ai- <i>idalab_widgets_base.ProcessListWidget</i> attribute), 48
<code>_build_tree()</code>	(ai- <i>idalab_widgets_base.NodesTreeWidget</i> method), 45	<code>_default_process_label</code>	(ai- <i>idalab_widgets_base.process.ProcessListWidget</i> attribute), 23
<code>_build_tree()</code>	(ai- <i>idalab_widgets_base.nodes.NodesTreeWidget</i> class method), 20	<code>_default_selection</code>	(ai- <i>idalab_widgets_base.viewers._StructureDataBaseViewer</i> attribute), 33
<code>_change_calculation</code>	(ai- <i>idalab_widgets_base.process.CalcJobOutputWidget</i> attribute), 21	<code>_default_structure</code>	(ai- <i>idalab_widgets_base.CodQueryWidget</i> attribute), 40
<code>_change_structure_node</code>	(ai- <i>idalab_widgets_base.StructureManagerWidget</i> attribute), 54	<code>_default_structure</code>	(ai- <i>idalab_widgets_base.SmilesWidget</i> attribute), 51
<code>_change_structure_node</code>	(ai- <i>idalab_widgets_base.structures.StructureManagerWidget</i> attribute), 29	<code>_default_structure</code>	(ai- <i>idalab_widgets_base.StructureExamplesWidget</i> attribute), 53
<code>_configure_computer()</code>	(ai- <i>idalab_widgets_base.AiidaComputerSetup</i> method), 37	<code>_default_structure</code>	(ai- <i>idalab_widgets_base.databases.CodQueryWidget</i> attribute), 15
<code>_configure_computer()</code>	(ai- <i>idalab_widgets_base.computers.AiidaComputerSetup</i> method), 11	<code>_default_structure</code>	(ai- <i>idalab_widgets_base.structures.SmilesWidget</i> attribute), 28
<code>_configure_proxy()</code>	(ai- <i>idalab_widgets_base.SshComputerSetup</i> method), 51	<code>_default_structure</code>	(ai- <i>idalab_widgets_base.structures.StructureExamplesWidget</i> attribute), 29
<code>_configure_proxy()</code>	(ai- <i>idalab_widgets_base.computers.SshComputerSetup</i> method), 13	<code>_default_supercell</code>	(ai- <i>idalab_widgets_base.viewers._StructureDataBaseViewer</i> attribute), 33
<code>_consider_auto_advance()</code>	(ai- <i>idalab_widgets_base.WizardAppWidget</i> method), 56		



<code>_download()</code>	( <code>aiidalab_widgets_base.viewers._StructureDataBaseViewer</code> static method), 33	<code>_observe_input_structure</code>	( <code>aiidalab_widgets_base.StructureManagerWidget</code> attribute), 54
<code>_download_tab()</code>	( <code>aiidalab_widgets_base.viewers._StructureDataBaseViewer</code> method), 33	<code>_observe_input_structure</code>	( <code>aiidalab_widgets_base.structures.StructureManagerWidget</code> attribute), 29
<code>_find_called()</code>	( <code>aiidalab_widgets_base.NodesTreeWidget</code> class method), 45	<code>_observe_nodes</code>	( <code>aiidalab_widgets_base.NodesTreeWidget</code> attribute), 45
<code>_find_called()</code>	( <code>aiidalab_widgets_base.nodes.NodesTreeWidget</code> class method), 20	<code>_observe_nodes</code>	( <code>aiidalab_widgets_base.nodes.NodesTreeWidget</code> attribute), 20
<code>_find_children()</code>	( <code>aiidalab_widgets_base.NodesTreeWidget</code> class method), 45	<code>_observe_process</code>	( <code>aiidalab_widgets_base.ProcessMonitor</code> attribute), 48
<code>_find_children()</code>	( <code>aiidalab_widgets_base.nodes.NodesTreeWidget</code> class method), 20	<code>_observe_process</code>	( <code>aiidalab_widgets_base.ProcessNodesTreeWidget</code> attribute), 49
<code>_find_outputs()</code>	( <code>aiidalab_widgets_base.NodesTreeWidget</code> class method), 45	<code>_observe_process</code>	( <code>aiidalab_widgets_base.process.ProcessMonitor</code> attribute), 24
<code>_find_outputs()</code>	( <code>aiidalab_widgets_base.nodes.NodesTreeWidget</code> class method), 20	<code>_observe_process</code>	( <code>aiidalab_widgets_base.process.ProcessNodesTreeWidget</code> attribute), 24
<code>_follow()</code>	( <code>aiidalab_widgets_base.ProcessListWidget</code> method), 48	<code>_observe_proxy_command</code>	( <code>aiidalab_widgets_base.ComputerDatabaseWidget</code> attribute), 42
<code>_follow()</code>	( <code>aiidalab_widgets_base.process.ProcessListWidget</code> method), 23	<code>_observe_proxy_command</code>	( <code>aiidalab_widgets_base.databases.ComputerDatabaseWidget</code> attribute), 16
<code>_full_code_label()</code>	( <code>aiidalab_widgets_base.CodeDropdown</code> static method), 41	<code>_observe_proxy_hostname</code>	( <code>aiidalab_widgets_base.SshComputerSetup</code> attribute), 52
<code>_full_code_label()</code>	( <code>aiidalab_widgets_base.codes.CodeDropdown</code> static method), 11	<code>_observe_proxy_hostname</code>	( <code>aiidalab_widgets_base.computers.SshComputerSetup</code> attribute), 13
<code>_get_codes()</code>	( <code>aiidalab_widgets_base.CodeDropdown</code> method), 41	<code>_observe_proxy_username</code>	( <code>aiidalab_widgets_base.SshComputerSetup</code> attribute), 52
<code>_get_codes()</code>	( <code>aiidalab_widgets_base.codes.CodeDropdown</code> method), 11	<code>_observe_proxy_username</code>	( <code>aiidalab_widgets_base.computers.SshComputerSetup</code> attribute), 13
<code>_get_computers()</code>	( <code>aiidalab_widgets_base.ComputerDropdown</code> method), 43	<code>_observe_selected_index</code>	( <code>aiidalab_widgets_base.WizardAppWidget</code> attribute), 56
<code>_get_computers()</code>	( <code>aiidalab_widgets_base.computers.ComputerDropdown</code> method), 13	<code>_observe_selected_index</code>	( <code>aiidalab_widgets_base.wizard.WizardAppWidget</code> attribute), 35
<code>_make_host_known()</code>	( <code>aiidalab_widgets_base.SshComputerSetup</code> method), 51	<code>_observe_selection</code>	( <code>aiidalab_widgets_base.viewers._StructureDataBaseViewer</code> attribute), 33
<code>_make_host_known()</code>	( <code>aiidalab_widgets_base.computers.SshComputerSetup</code> method), 13	<code>_observe_selection_2</code>	( <code>aiidalab_widgets_base.viewers.StructureDataViewer</code> attribute), 33
<code>_monitor_process()</code>	( <code>aiidalab_widgets_base.ProcessMonitor</code> method), 48		
<code>_monitor_process()</code>	( <code>aiidalab_widgets_base.process.ProcessMonitor</code> method), 48		



<i>attribute</i> ), 32		<i>method</i> ), 35	
<code>_observe_selection_adv</code>	(ai- <i>idalab_widgets_base.viewers.StructureDataViewer</i> <i>attribute</i> ), 33	<code>_on_click_store_all()</code>	(ai- <i>idalab_widgets_base.MultiStructureUploadWidget</i> <i>method</i> ), 44
<code>_observe_structure_node</code>	(ai- <i>idalab_widgets_base.StructureManagerWidget</i> <i>attribute</i> ), 54	<code>_on_click_store_all()</code>	(ai- <i>idalab_widgets_base.structures_multi.MultiStructureUploadWidget</i> <i>method</i> ), 31
<code>_observe_structure_node</code>	(ai- <i>idalab_widgets_base.structures.StructureManagerWidget</i> <i>attribute</i> ), 29	<code>_on_click_store_selected()</code>	(ai- <i>idalab_widgets_base.MultiStructureUploadWidget</i> <i>method</i> ), 44
<code>_observe_tree_selected_nodes()</code>	(ai- <i>idalab_widgets_base.NodesTreeWidget</i> <i>method</i> ), 45	<code>_on_click_store_selected()</code>	(ai- <i>idalab_widgets_base.structures_multi.MultiStructureUploadWidget</i> <i>method</i> ), 31
<code>_observe_tree_selected_nodes()</code>	(ai- <i>idalab_widgets_base.ProcessNodesTreeWidget</i> <i>method</i> ), 49	<code>_on_file_upload()</code>	(ai- <i>idalab_widgets_base.MultiStructureUploadWidget</i> <i>method</i> ), 44
<code>_observe_tree_selected_nodes()</code>	(ai- <i>idalab_widgets_base.nodes.NodesTreeWidget</i> <i>method</i> ), 20	<code>_on_file_upload()</code>	(ai- <i>idalab_widgets_base.StructureUploadWidget</i> <i>method</i> ), 55
<code>_observe_tree_selected_nodes()</code>	(ai- <i>idalab_widgets_base.process.ProcessNodesTreeWidget</i> <i>method</i> ), 24	<code>_on_file_upload()</code>	(ai- <i>idalab_widgets_base.structures.StructureUploadWidget</i> <i>method</i> ), 30
<code>_on_atom_click()</code>	(ai- <i>idalab_widgets_base.viewers._StructureDataBaseViewer</i> <i>method</i> ), 34	<code>_on_file_upload()</code>	(ai- <i>idalab_widgets_base.structures_multi.MultiStructureUploadWidget</i> <i>method</i> ), 31
<code>_on_button_pressed()</code>	(ai- <i>idalab_widgets_base.SmilesWidget</i> <i>method</i> ), 51	<code>_on_select_structure()</code>	(ai- <i>idalab_widgets_base.CodQueryWidget</i> <i>method</i> ), 40
<code>_on_button_pressed()</code>	(ai- <i>idalab_widgets_base.structures.SmilesWidget</i> <i>method</i> ), 28	<code>_on_select_structure()</code>	(ai- <i>idalab_widgets_base.StructureBrowserWidget</i> <i>method</i> ), 53
<code>_on_click_back_button()</code>	(ai- <i>idalab_widgets_base.WizardAppWidget</i> <i>method</i> ), 56	<code>_on_select_structure()</code>	(ai- <i>idalab_widgets_base.StructureExamplesWidget</i> <i>method</i> ), 53
<code>_on_click_back_button()</code>	(ai- <i>idalab_widgets_base.wizard.WizardAppWidget</i> <i>method</i> ), 35	<code>_on_select_structure()</code>	(ai- <i>idalab_widgets_base.databases.CodQueryWidget</i> <i>method</i> ), 15
<code>_on_click_next_button()</code>	(ai- <i>idalab_widgets_base.WizardAppWidget</i> <i>method</i> ), 56	<code>_on_select_structure()</code>	(ai- <i>idalab_widgets_base.structures.StructureBrowserWidget</i> <i>method</i> ), 28
<code>_on_click_next_button()</code>	(ai- <i>idalab_widgets_base.wizard.WizardAppWidget</i> <i>method</i> ), 35	<code>_on_select_structure()</code>	(ai- <i>idalab_widgets_base.structures.StructureExamplesWidget</i> <i>method</i> ), 29
<code>_on_click_query()</code>	(ai- <i>idalab_widgets_base.CodQueryWidget</i> <i>method</i> ), 40	<code>_on_setup_computer()</code>	(ai- <i>idalab_widgets_base.AiidaComputerSetup</i> <i>method</i> ), 37
<code>_on_click_query()</code>	(ai- <i>idalab_widgets_base.databases.CodQueryWidget</i> <i>method</i> ), 15	<code>_on_setup_computer()</code>	(ai- <i>idalab_widgets_base.computers.AiidaComputerSetup</i> <i>method</i> ), 12
<code>_on_click_reset_button()</code>	(ai- <i>idalab_widgets_base.WizardAppWidget</i> <i>method</i> ), 56	<code>_on_setup_ssh()</code>	(ai- <i>idalab_widgets_base.SshComputerSetup</i> <i>method</i> ), 52
<code>_on_click_reset_button()</code>	(ai- <i>idalab_widgets_base.wizard.WizardAppWidget</i> <i>method</i> ), 35	<code>_on_setup_ssh()</code>	(ai- <i>idalab_widgets_base.computers.SshComputerSetup</i> <i>method</i> ), 52

<i>method</i> ), 14		<i>method</i> ), 54	
<code>_prepare_payload()</code>	(ai- <i>idalab_widgets_base.viewers._StructureDataBaseViewer</i> <i>method</i> ), 34	<code>_structure_editors()</code>	(ai- <i>idalab_widgets_base.structures.StructureManagerWidget</i> <i>method</i> ), 30
<code>_pybel_opt()</code>	(aiidalab_widgets_base.SmilesWidget <i>method</i> ), 51	<code>_structure_importers()</code>	(ai- <i>idalab_widgets_base.StructureManagerWidget</i> <i>method</i> ), 54
<code>_pybel_opt()</code>	(aiidalab_widgets_base.structures.SmilesWidget <i>method</i> ), 28	<code>_structure_importers()</code>	(ai- <i>idalab_widgets_base.structures.StructureManagerWidget</i> <i>method</i> ), 30
<code>_query()</code>	(aiidalab_widgets_base.CodQueryWidget <i>static method</i> ), 40	<code>_sync_structure_node()</code>	(ai- <i>idalab_widgets_base.StructureManagerWidget</i> <i>method</i> ), 54
<code>_query()</code>	(aiidalab_widgets_base.databases.CodQueryWidget <i>static method</i> ), 15	<code>_sync_structure_node()</code>	(ai- <i>idalab_widgets_base.structures.StructureManagerWidget</i> <i>method</i> ), 30
<code>_rdkit_opt()</code>	(aiidalab_widgets_base.SmilesWidget <i>method</i> ), 51	<code>_to_tree_node()</code>	(ai- <i>idalab_widgets_base.NodesTreeWidget</i> <i>class method</i> ), 45
<code>_rdkit_opt()</code>	(aiidalab_widgets_base.structures.SmilesWidget <i>method</i> ), 28	<code>_to_tree_node()</code>	(ai- <i>idalab_widgets_base.nodes.NodesTreeWidget</i> <i>class method</i> ), 20
<code>_refresh_output()</code>	(ai- <i>idalab_widgets_base.NodesTreeWidget</i> <i>method</i> ), 45	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.AiiDACodeSetup</i> <i>attribute</i> ), 37
<code>_refresh_output()</code>	(ai- <i>idalab_widgets_base.nodes.NodesTreeWidget</i> <i>method</i> ), 20	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.AiidaComputerSetup</i> <i>attribute</i> ), 37
<code>_render_structure()</code>	(ai- <i>idalab_widgets_base.viewers._StructureDataBaseViewer</i> <i>method</i> ), 34	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.BasicStructureEditor</i> <i>attribute</i> ), 38
<code>_selection_tab()</code>	(ai- <i>idalab_widgets_base.viewers._StructureDataBaseViewer</i> <i>method</i> ), 34	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.CodQueryWidget</i> <i>attribute</i> ), 40
<code>_send_pubkey()</code>	(ai- <i>idalab_widgets_base.SshComputerSetup</i> <i>static method</i> ), 52	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.CodeDatabaseWidget</i> <i>attribute</i> ), 40
<code>_send_pubkey()</code>	(ai- <i>idalab_widgets_base.computers.SshComputerSetup</i> <i>static method</i> ), 14	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.CodeDropdown</i> <i>attribute</i> ), 41
<code>_setup_code()</code>	(ai- <i>idalab_widgets_base.AiiDACodeSetup</i> <i>method</i> ), 37	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.ComputerDatabaseWidget</i> <i>attribute</i> ), 42
<code>_setup_code()</code>	(ai- <i>idalab_widgets_base.codes.AiiDACodeSetup</i> <i>method</i> ), 10	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.ComputerDropdown</i> <i>attribute</i> ), 43
<code>_ssh_keygen()</code>	(ai- <i>idalab_widgets_base.SshComputerSetup</i> <i>static method</i> ), 52	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.ExportButtonWidget</i> <i>attribute</i> ), 44
<code>_ssh_keygen()</code>	(ai- <i>idalab_widgets_base.computers.SshComputerSetup</i> <i>static method</i> ), 14	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.MultiStructureUploadWidget</i> <i>attribute</i> ), 44
<code>_structure_changed</code>	(ai- <i>idalab_widgets_base.StructureManagerWidget</i> <i>attribute</i> ), 54	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.NodesTreeWidget</i> <i>attribute</i> ), 44
<code>_structure_changed</code>	(ai- <i>idalab_widgets_base.structures.StructureManagerWidget</i> <i>attribute</i> ), 29	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.NodesTreeWidget</i> <i>attribute</i> ), 44
<code>_structure_editors()</code>	(ai- <i>idalab_widgets_base.StructureManagerWidget</i> <i>method</i> ), 54	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.NodesTreeWidget</i> <i>attribute</i> ), 44

<i>tribute</i> ), 45		<i>tribute</i> ), 55	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.OptimadeQueryWidget attribute</i> ), 46		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.WizardAppWidget attribute</i> ), 56	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.ProcessCallStackWidget attribute</i> ), 46		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.WizardAppWidgetStep attribute</i> ), 57	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.ProcessFollowerWidget attribute</i> ), 47		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.codes.AiiDACodeSetup attribute</i> ), 10	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.ProcessInputsWidget attribute</i> ), 47		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.codes.CodeDropdown attribute</i> ), 11	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.ProcessListWidget attribute</i> ), 48		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.computers.AiidaComputerSetup attribute</i> ), 12	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.ProcessMonitor attribute</i> ), 48		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.computers.ComputerDropdown attribute</i> ), 13	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.ProcessNodesTreeWidget attribute</i> ), 49		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.computers.SshComputerSetup attribute</i> ), 14	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.ProcessOutputsWidget attribute</i> ), 49		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.data.LigandSelectorWidget attribute</i> ), 9	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.ProcessReportWidget attribute</i> ), 50		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.databases.CodQueryWidget attribute</i> ), 15	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.ProgressBarWidget attribute</i> ), 50		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.databases.CodeDatabaseWidget attribute</i> ), 15	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.RunningCalc.JobOutputWidget attribute</i> ), 50		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.databases.ComputerDatabaseWidget attribute</i> ), 16	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.SmilesWidget attribute</i> ), 51		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.databases.OptimadeQueryWidget attribute</i> ), 17	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.SshComputerSetup attribute</i> ), 52		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.export.ExportButtonWidget attribute</i> ), 18	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.StructureBrowserWidget attribute</i> ), 53		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.misc.CopyToClipboardButton attribute</i> ), 18	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.StructureExamplesWidget attribute</i> ), 53		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.nodes.AiidaTreeNodeNode attribute</i> ), 19	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.StructureManagerWidget attribute</i> ), 54		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.nodes.AiidaOutputsTreeNodeNode attribute</i> ), 19	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.StructureUploadWidget attribute</i> ), 55		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.nodes.AiidaProcessTreeNodeNode attribute</i> ), 19	
<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.SubmitButtonWidget attribute</i> ), 55		<code>_trait_default_generators</code> ( <i>ai- idalab_widgets_base.nodes.CalcFunctionTreeNodeNode attribute</i> ), 19	

<i>attribute</i> ), 20		<i>attribute</i> ), 28	
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.nodes.CalcJobTreeNode</i> <i>attribute</i> ), 20	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.structures.StructureBrowserWidget</i> <i>attribute</i> ), 28
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.nodes.NodesTreeWidget</i> <i>attribute</i> ), 20	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.structures.StructureExamplesWidget</i> <i>attribute</i> ), 29
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.nodes.UnknownTypeTreeNode</i> <i>attribute</i> ), 21	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.structures.StructureManagerWidget</i> <i>attribute</i> ), 30
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.nodes.WorkChainProcessTreeNode</i> <i>attribute</i> ), 21	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.structures.StructureUploadWidget</i> <i>attribute</i> ), 30
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.process.CalcJobOutputWidget</i> <i>attribute</i> ), 21	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.structures_multi.MultiStructureUploadWidget</i> <i>attribute</i> ), 31
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.process.ProcessCallStackWidget</i> <i>attribute</i> ), 22	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.viewers.BandsDataViewer</i> <i>attribute</i> ), 32
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.process.ProcessFollowerWidget</i> <i>attribute</i> ), 22	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.viewers.DictViewer</i> <i>attribute</i> ), 32
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.process.ProcessInputsWidget</i> <i>attribute</i> ), 22	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.viewers.FolderDataViewer</i> <i>attribute</i> ), 32
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.process.ProcessListWidget</i> <i>attribute</i> ), 23	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.viewers.StructureDataViewer</i> <i>attribute</i> ), 33
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.process.ProcessMonitor</i> <i>attribute</i> ), 24	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.viewers._StructureDataBaseViewer</i> <i>attribute</i> ), 34
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.process.ProcessNodesTreeWidget</i> <i>attribute</i> ), 24	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.wizard.WizardAppWidget</i> <i>attribute</i> ), 35
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.process.ProcessOutputsWidget</i> <i>attribute</i> ), 24	<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.wizard.WizardAppWidgetStep</i> <i>attribute</i> ), 36
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.process.ProcessReportWidget</i> <i>attribute</i> ), 25	<code>_update_buttons()</code>	(ai- <i>idalab_widgets_base.WizardAppWidget</i> <i>method</i> ), 56
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.process.ProgressBarWidget</i> <i>attribute</i> ), 25	<code>_update_buttons()</code>	(ai- <i>idalab_widgets_base.wizard.WizardAppWidget</i> <i>method</i> ), 35
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.process.RunningCalcJobOutputWidget</i> <i>attribute</i> ), 25	<code>_update_displayed_structure</code>	(ai- <i>idalab_widgets_base.viewers.StructureDataViewer</i> <i>attribute</i> ), 33
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.process.SubmitButtonWidget</i> <i>attribute</i> ), 26	<code>_update_step_state()</code>	(ai- <i>idalab_widgets_base.WizardAppWidget</i> <i>method</i> ), 56
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.structures.BasicStructureEditor</i> <i>attribute</i> ), 27	<code>_update_step_state()</code>	(ai- <i>idalab_widgets_base.wizard.WizardAppWidget</i> <i>method</i> ), 35
<code>_trait_default_generators</code>	(ai- <i>idalab_widgets_base.structures.SmilesWidget</i> <i>attribute</i> ), 27	<code>_update_structure()</code>	(ai- <i>idalab_widgets_base.OptimadeQueryWidget</i> <i>attribute</i> ), 33

<i>method</i> ), 46		<i>attribute</i> ), 48	
<code>_update_structure()</code>	(ai- <i>idalab_widgets_base.databases.OptimadeQueryWidget</i> <i>method</i> ), 17	<code>_validate_process_label</code>	(ai- <i>idalab_widgets_base.process.ProcessListWidget</i> <i>attribute</i> ), 23
<code>_update_structure_viewer</code>	(ai- <i>idalab_widgets_base.viewers.StructureDataViewer</i> <i>attribute</i> ), 33	<code>_validate_safe_interval</code>	(ai- <i>idalab_widgets_base.AiidaComputerSetup</i> <i>attribute</i> ), 38
<code>_update_titles()</code>	(ai- <i>idalab_widgets_base.WizardAppWidget</i> <i>method</i> ), 56	<code>_validate_safe_interval</code>	(ai- <i>idalab_widgets_base.computers.AiidaComputerSetup</i> <i>attribute</i> ), 12
<code>_update_titles()</code>	(ai- <i>idalab_widgets_base.wizard.WizardAppWidget</i> <i>method</i> ), 35	<code>_validate_selected_code</code>	(ai- <i>idalab_widgets_base.CodeDropdown</i> <i>attribute</i> ), 41
<code>_update_tree_node()</code>	(ai- <i>idalab_widgets_base.NodesTreeWidget</i> <i>method</i> ), 45	<code>_validate_selected_code</code>	(ai- <i>idalab_widgets_base.codes.CodeDropdown</i> <i>attribute</i> ), 11
<code>_update_tree_node()</code>	(ai- <i>idalab_widgets_base.nodes.NodesTreeWidget</i> <i>method</i> ), 20	<code>_validate_selected_computer</code>	(ai- <i>idalab_widgets_base.ComputerDropdown</i> <i>attribute</i> ), 43
<code>_valid_structure</code>	(ai- <i>idalab_widgets_base.viewers.StructureDataViewer</i> <i>attribute</i> ), 33	<code>_validate_selected_computer</code>	(ai- <i>idalab_widgets_base.computers.ComputerDropdown</i> <i>attribute</i> ), 13
<code>_validate_and_fix_ase_cell()</code>	(ai- <i>idalab_widgets_base.StructureUploadWidget</i> <i>method</i> ), 55	<code>_validate_selection</code>	(ai- <i>idalab_widgets_base.viewers._StructureDataBaseViewer</i> <i>attribute</i> ), 34
<code>_validate_and_fix_ase_cell()</code>	(ai- <i>idalab_widgets_base.structures.StructureUploadWidget</i> <i>method</i> ), 30	<code>_walk_tree()</code>	( <i>aiidalab_widgets_base.NodesTreeWidget</i> <i>class method</i> ), 45
<code>_validate_incoming_node</code>	(ai- <i>idalab_widgets_base.ProcessListWidget</i> <i>attribute</i> ), 48	<code>_walk_tree()</code>	( <i>aiidalab_widgets_base.nodes.NodesTreeWidget</i> <i>class method</i> ), 20
<code>_validate_incoming_node</code>	(ai- <i>idalab_widgets_base.process.ProcessListWidget</i> <i>attribute</i> ), 23	<code>_write_ssh_config()</code>	(ai- <i>idalab_widgets_base.SshComputerSetup</i> <i>method</i> ), 52
<code>_validate_mpiprocs_per_machine</code>	(ai- <i>idalab_widgets_base.AiidaComputerSetup</i> <i>attribute</i> ), 37	<code>_write_ssh_config()</code>	(ai- <i>idalab_widgets_base.computers.SshComputerSetup</i> <i>method</i> ), 14
<code>_validate_mpiprocs_per_machine</code>	(ai- <i>idalab_widgets_base.computers.AiidaComputerSetup</i> <i>attribute</i> ), 12	<b>A</b>	
<code>_validate_outgoing_node</code>	(ai- <i>idalab_widgets_base.ProcessListWidget</i> <i>attribute</i> ), 48	<code>action_vector()</code>	(ai- <i>idalab_widgets_base.BasicStructureEditor</i> <i>property</i> ), 38
<code>_validate_outgoing_node</code>	(ai- <i>idalab_widgets_base.process.ProcessListWidget</i> <i>attribute</i> ), 23	<code>action_vector()</code>	(ai- <i>idalab_widgets_base.structures.BasicStructureEditor</i> <i>property</i> ), 27
<code>_validate_port</code>	(ai- <i>idalab_widgets_base.SshComputerSetup</i> <i>attribute</i> ), 52	ACTIVE	( <i>aiidalab_widgets_base.wizard.WizardAppWidgetStep.State</i> <i>attribute</i> ), 36
<code>_validate_port</code>	(ai- <i>idalab_widgets_base.computers.SshComputerSetup</i> <i>attribute</i> ), 14	ACTIVE	( <i>aiidalab_widgets_base.WizardAppWidgetStep.State</i> <i>attribute</i> ), 57
<code>_validate_process_label</code>	(ai- <i>idalab_widgets_base.ProcessListWidget</i> <i>method</i> ), 37	add()	( <i>aiidalab_widgets_base.BasicStructureEditor</i> <i>method</i> ), 38
		add()	( <i>aiidalab_widgets_base.structures.BasicStructureEditor</i> <i>method</i> ), 27
		AiiDACodeSetup	( <i>class in aiidalab_widgets_base</i> ), 37



AiiDACodeSetup	(class in <i>aiidalab_widgets_base.codes</i> ), 10	ai-	<i>aiidalab_widgets_base.codes.CodeDropdown</i> attribute), 11
AiidaComputerSetup	(class in <i>aiidalab_widgets_base</i> ), 37	ai-	allow_hidden_codes ( <i>aiidalab_widgets_base.CodeDropdown</i> attribute), 41
AiidaComputerSetup	(class in <i>aiidalab_widgets_base.computers</i> ), 11	ai-	allow_hidden_codes ( <i>aiidalab_widgets_base.codes.CodeDropdown</i> attribute), 11
aiidalab_widgets_base	module, 37		allow_select_disabled ( <i>aiidalab_widgets_base.ComputerDropdown</i> attribute), 43
aiidalab_widgets_base.codes	module, 10		allow_select_disabled ( <i>aiidalab_widgets_base.computers.ComputerDropdown</i> attribute), 13
aiidalab_widgets_base.computers	module, 11		anchoring_atom() ( <i>aiidalab_widgets_base.data.LigandSelectorWidget</i> property), 9
aiidalab_widgets_base.data	module, 9		append_text ( <i>aiidalab_widgets_base.AiiDACodeSetup</i> attribute), 37
aiidalab_widgets_base.databases	module, 15		append_text ( <i>aiidalab_widgets_base.AiidaComputerSetup</i> attribute), 38
aiidalab_widgets_base.dicts	module, 18		append_text ( <i>aiidalab_widgets_base.CodeDatabaseWidget</i> attribute), 40
aiidalab_widgets_base.export	module, 18		append_text ( <i>aiidalab_widgets_base.codes.AiiDACodeSetup</i> attribute), 10
aiidalab_widgets_base.misc	module, 18		append_text ( <i>aiidalab_widgets_base.ComputerDatabaseWidget</i> attribute), 42
aiidalab_widgets_base.nodes	module, 19		append_text ( <i>aiidalab_widgets_base.computers.AiidaComputerSetup</i> attribute), 12
aiidalab_widgets_base.process	module, 21		append_text ( <i>aiidalab_widgets_base.databases.CodeDatabaseWidget</i> attribute), 15
aiidalab_widgets_base.structures	module, 26		append_text ( <i>aiidalab_widgets_base.databases.ComputerDatabaseWidget</i> attribute), 16
aiidalab_widgets_base.structures_multi	module, 31		apply_selection() ( <i>aiidalab_widgets_base.viewers._StructureDataBaseViewer</i> method), 34
aiidalab_widgets_base.utils	module, 9		auto_advance ( <i>aiidalab_widgets_base.wizard.WizardAppWidgetStep</i> attribute), 36
aiidalab_widgets_base.viewers	module, 31		auto_advance ( <i>aiidalab_widgets_base.WizardAppWidgetStep</i> attribute), 57
aiidalab_widgets_base.wizard	module, 34		
AiidaNodeTreeNode	(class in <i>aiidalab_widgets_base.nodes</i> ), 19	ai-	
AiidaOutputsTreeNode	(class in <i>aiidalab_widgets_base.nodes</i> ), 19	ai-	
AiidaProcessNodeTreeNode	(class in <i>aiidalab_widgets_base.nodes</i> ), 19	ai-	
align()	( <i>aiidalab_widgets_base.BasicStructureEditor</i> method), 38		<b>B</b>
align()	( <i>aiidalab_widgets_base.structures.BasicStructureEditor</i> method), 27		BandsDataViewer (class in <i>aiidalab_widgets_base.viewers</i> ), 31
allow_agent	( <i>aiidalab_widgets_base.ComputerDatabaseWidget</i> attribute), 42		BasicStructureEditor (class in <i>aiidalab_widgets_base</i> ), 38
allow_agent	( <i>aiidalab_widgets_base.databases.ComputerDatabaseWidget</i> attribute), 16		BasicStructureEditor (class in <i>aiidalab_widgets_base.structures</i> ), 26
allow_disabled_computers	( <i>aiidalab_widgets_base.CodeDropdown</i> attribute), 41	(ai-	
allow_disabled_computers	( <i>ai-</i>		<b>C</b>
			calc_info() ( <i>aiidalab_widgets_base.process.ProcessCallStackWidget</i> method), 22
			calc_info() ( <i>aiidalab_widgets_base.ProcessCallStackWidget</i> method), 46

CalcFunctionTreeNode (class in ai- computer (aiidalab\_widgets\_base.codes.AiiDACodeSetup  
 idalab\_widgets\_base.nodes), 20 attribute), 10

CalcJobOutputWidget (class in ai- computer (aiidalab\_widgets\_base.databases.CodeDatabaseWidget  
 idalab\_widgets\_base.process), 21 attribute), 15

CalcJobTreeNode (class in ai- ComputerDatabaseWidget (class in ai-  
 idalab\_widgets\_base.nodes), 20 idalab\_widgets\_base), 42

calculation (aiidalab\_widgets\_base.process.CalcJobOutputWidgetDatabaseWidget (class in ai-  
 attribute), 21 idalab\_widgets\_base.databases), 16

camera\_orientation (ai- ComputerDropdown (class in ai-  
 idalab\_widgets\_base.BasicStructureEditor idalab\_widgets\_base), 43  
 attribute), 38 ComputerDropdown (class in ai-  
 ComputerDropdown (class in ai- idalab\_widgets\_base.computers), 12  
 idalab\_widgets\_base.structures.BasicStructureEditor computers (aiidalab\_widgets\_base.ComputerDropdown  
 attribute), 27 attribute), 43

can\_login() (aiidalab\_widgets\_base.computers.SshComputerSetup computers (aiidalab\_widgets\_base.computers.ComputerDropdown  
 method), 14 attribute), 13

can\_login() (aiidalab\_widgets\_base.SshComputerSetup CONFIGURED (aiidalab\_widgets\_base.wizard.WizardAppWidgetStep.State  
 method), 52 attribute), 36

can\_reset() (aiidalab\_widgets\_base.wizard.WizardAppWidget CONFIGURED (aiidalab\_widgets\_base.WizardAppWidgetStep.State  
 method), 35 attribute), 57

can\_reset() (aiidalab\_widgets\_base.wizard.WizardAppWidgetStep() (aiidalab\_widgets\_base.misc.ReversePolishNotation  
 method), 36 method), 19

can\_reset() (aiidalab\_widgets\_base.WizardAppWidget copy\_sel() (aiidalab\_widgets\_base.BasicStructureEditor  
 method), 56 method), 38

can\_reset() (aiidalab\_widgets\_base.WizardAppWidgetStep copy\_sel() (aiidalab\_widgets\_base.structures.BasicStructureEditor  
 method), 57 method), 27

change\_file\_view() (ai- copy\_to\_clipboard() (ai-  
 idalab\_widgets\_base.viewers.FolderDataViewer idalab\_widgets\_base.misc.CopyToClipboardButton  
 method), 32 method), 18

change\_structure() (ai- CopyToClipboardButton (class in ai-  
 idalab\_widgets\_base.MultiStructureUploadWidget idalab\_widgets\_base.misc), 18

change\_structure() (ai- create\_selection\_info() (ai-  
 idalab\_widgets\_base.structures\_multi.MultiStructureUploadWidget idalab\_widgets\_base.viewers.StructureDataViewer  
 method), 31 method), 33

CodeDatabaseWidget (class in ai- current\_state() (ai-  
 idalab\_widgets\_base), 40 idalab\_widgets\_base.process.ProgressBarWidget  
 property), 25

CodeDatabaseWidget (class in ai- current\_state() (ai-  
 idalab\_widgets\_base.databases), 15 idalab\_widgets\_base.ProgressBarWidget  
 property), 50

CodeDropdown (class in aiidalab\_widgets\_base), 41

CodeDropdown (class in ai-  
 idalab\_widgets\_base.codes), 10

codes (aiidalab\_widgets\_base.CodeDropdown at-  
 tribute), 41

codes (aiidalab\_widgets\_base.codes.CodeDropdown at-  
 tribute), 11

CodQueryWidget (class in aiidalab\_widgets\_base),  
 39

CodQueryWidget (class in ai-  
 idalab\_widgets\_base.databases), 15

computer (aiidalab\_widgets\_base.AiiDACodeSetup at-  
 tribute), 37

computer (aiidalab\_widgets\_base.CodeDatabaseWidget  
 attribute), 40

**D**

d\_from() (aiidalab\_widgets\_base.viewers.StructureDataViewer  
 method), 33

def\_axis\_p1() (ai-  
 idalab\_widgets\_base.BasicStructureEditor  
 method), 39

def\_axis\_p1() (ai-  
 idalab\_widgets\_base.structures.BasicStructureEditor  
 method), 27

def\_axis\_p2() (ai-  
 idalab\_widgets\_base.BasicStructureEditor  
 method), 39

def\_axis\_p2() (aiidalab\_widgets\_base.structures.BasicStructureEditor method), 27

def\_perpendicular\_to\_screen() (aiidalab\_widgets\_base.BasicStructureEditor method), 39

def\_perpendicular\_to\_screen() (aiidalab\_widgets\_base.structures.BasicStructureEditor method), 27

def\_point() (aiidalab\_widgets\_base.BasicStructureEditor method), 39

def\_point() (aiidalab\_widgets\_base.structures.BasicStructureEditor method), 27

DEFAULT\_SELECTION\_COLOR (aiidalab\_widgets\_base.viewers.\_StructureDataBaseViewer attribute), 33

DEFAULT\_SELECTION\_OPACITY (aiidalab\_widgets\_base.viewers.\_StructureDataBaseViewer attribute), 33

DEFAULT\_SELECTION\_RADIUS (aiidalab\_widgets\_base.viewers.\_StructureDataBaseViewer attribute), 33

description (aiidalab\_widgets\_base.AiiDACodeSetup attribute), 37

description (aiidalab\_widgets\_base.AiidaComputerSetup attribute), 38

description (aiidalab\_widgets\_base.CodeDatabaseWidget attribute), 40

description (aiidalab\_widgets\_base.codes.AiiDACodeSetup attribute), 10

description (aiidalab\_widgets\_base.ComputerDatabaseWidget attribute), 42

description (aiidalab\_widgets\_base.computers.AiidaComputerSetup attribute), 12

description (aiidalab\_widgets\_base.databases.CodeDatabaseWidget attribute), 15

description (aiidalab\_widgets\_base.databases.ComputerDatabaseWidget attribute), 16

description\_contains (aiidalab\_widgets\_base.process.ProcessListWidget attribute), 23

description\_contains (aiidalab\_widgets\_base.ProcessListWidget attribute), 48

DictViewer (class in aiidalab\_widgets\_base.viewers), 32

disabled (aiidalab\_widgets\_base.nodes.AiidaOutputsTreeNode attribute), 19

displayed\_structure (aiidalab\_widgets\_base.viewers.StructureDataViewer attribute), 33

download() (aiidalab\_widgets\_base.viewers.\_StructureDataBaseViewer method), 34

download() (aiidalab\_widgets\_base.viewers.FolderDataViewer method), 32

execute() (aiidalab\_widgets\_base.misc.ReversePolishNotation method), 19

exists() (aiidalab\_widgets\_base.AiiDACodeSetup method), 37

exists() (aiidalab\_widgets\_base.codes.AiiDACodeSetup method), 10

export\_aiida\_subgraph() (aiidalab\_widgets\_base.export.ExportButtonWidget method), 18

export\_aiida\_subgraph() (aiidalab\_widgets\_base.ExportButtonWidget method), 44

ExportButtonWidget (class in aiidalab\_widgets\_base), 44

ExportButtonWidget (class in aiidalab\_widgets\_base.export), 18

FAIL (aiidalab\_widgets\_base.wizard.WizardAppWidgetStep.State attribute), 36

FAIL (aiidalab\_widgets\_base.WizardAppWidgetStep.State attribute), 57

find\_node() (aiidalab\_widgets\_base.nodes.NodesTreeWidget method), 20

find\_node() (aiidalab\_widgets\_base.NodesTreeWidget method), 45

find\_ranges() (in module aiidalab\_widgets\_base.utils), 9

FolderDataViewer (class in aiidalab\_widgets\_base.viewers), 32

follow() (aiidalab\_widgets\_base.process.ProcessFollowerWidget method), 22

follow() (aiidalab\_widgets\_base.ProcessFollowerWidget method), 47

get\_ase() (aiidalab\_widgets\_base.MultiStructureUploadWidget method), 44

get\_ase() (aiidalab\_widgets\_base.structures\_multi.MultiStructureUploadWidget method), 31

get\_ase\_from\_file() (in module aiidalab\_widgets\_base.utils), 9

get\_description() (aiidalab\_widgets\_base.MultiStructureUploadWidget static method), 44

get\_description() (aiidalab\_widgets\_base.structures\_multi.MultiStructureUploadWidget static method), 31

get\_example\_structures() (aiidalab\_widgets\_base.StructureExamplesWidget static method), 53



`get_example_structures()` (ai- `incoming_node` (ai-  
`idalab_widgets_base.structures.StructureExamplesWidget` `idalab_widgets_base.ProcessListWidget`  
*static method*), 29 *attribute*), 48

`get_running_calcs()` (in module ai- `INIT (aiidalab_widgets_base.wizard.WizardAppWidgetStep.State`  
`idalab_widgets_base.process)`), 26 *attribute*), 36

## H

`haslessorequalpriority()` (ai- `input_plugin (aiidalab_widgets_base.AiiDACodeSetup`  
`idalab_widgets_base.misc.ReversePolishNotation` *attribute*), 37  
*method*), 19 `input_plugin (aiidalab_widgets_base.CodeDatabaseWidget`  
*attribute*), 40

`highlight_atoms()` (ai- `input_plugin (aiidalab_widgets_base.codes.AiiDACodeSetup`  
`idalab_widgets_base.viewers._StructureDatabaseViewer` *attribute*), 10  
*method*), 34

`hostname (aiidalab_widgets_base.AiidaComputerSetup` `input_plugin (aiidalab_widgets_base.databases.CodeDatabaseWidget`  
*attribute*), 38 *attribute*), 15

`hostname (aiidalab_widgets_base.ComputerDatabaseWidget` `input_structure` (ai-  
*attribute*), 42 `idalab_widgets_base.StructureManagerWidget`  
*attribute*), 54

`hostname (aiidalab_widgets_base.computers.AiidaComputerSetup` `input_structure` (ai-  
*attribute*), 12 `idalab_widgets_base.structures.StructureManagerWidget`  
*attribute*), 14 *attribute*), 30

`hostname (aiidalab_widgets_base.databases.ComputerDatabaseWidget` `is_host_known ()` (ai-  
*attribute*), 16 `idalab_widgets_base.computers.SshComputerSetup`  
*method*), 14

`hostname (aiidalab_widgets_base.SshComputerSetup` `is_host_known ()` (ai-  
*attribute*), 52 `idalab_widgets_base.SshComputerSetup`  
*method*), 52

## I

`icon (aiidalab_widgets_base.nodes.AiidaOutputsTreeNode` `is_in_config ()` (ai-  
*attribute*), 19 `idalab_widgets_base.computers.SshComputerSetup`  
*method*), 14

`icon (aiidalab_widgets_base.nodes.CalcFunctionTreeNode` `is_in_config ()` (ai-  
*attribute*), 20 `idalab_widgets_base.SshComputerSetup`  
*method*), 52

`icon (aiidalab_widgets_base.nodes.CalcJobTreeNode` `is_operator ()` (ai-  
*attribute*), 20 `idalab_widgets_base.misc.ReversePolishNotation`  
*method*), 19

`icon (aiidalab_widgets_base.nodes.UnknownTypeTreeNode` `iscloseparenthesis ()` (ai-  
*attribute*), 21 `idalab_widgets_base.misc.ReversePolishNotation`  
*method*), 19

`icon (aiidalab_widgets_base.nodes.WorkChainProcessTreeNode` `isopenparenthesis ()` (ai-  
*attribute*), 21 `idalab_widgets_base.misc.ReversePolishNotation`  
*method*), 19

`ICON_SEPARATOR` (ai- `idalab_widgets_base.wizard.WizardAppWidget`  
*attribute*), 34

`ICON_SEPARATOR` (ai- `idalab_widgets_base.WizardAppWidget` *at-*  
*tribute*), 55

`ICONS (aiidalab_widgets_base.wizard.WizardAppWidget`  
*attribute*), 34

`ICONS (aiidalab_widgets_base.WizardAppWidget`  
*attribute*), 55

`icons () (aiidalab_widgets_base.wizard.WizardAppWidget`  
*class method*), 35

`icons () (aiidalab_widgets_base.WizardAppWidget`  
*class method*), 56

`incoming_node` (ai- `label (aiidalab_widgets_base.AiiDACodeSetup` *at-*  
`idalab_widgets_base.process.ProcessListWidget` *tribute*), 37  
*attribute*), 23 `label (aiidalab_widgets_base.AiidaComputerSetup` *at-*  
*tribute*), 38

## J

`join () (aiidalab_widgets_base.process.ProcessMonitor`  
*method*), 24

`join () (aiidalab_widgets_base.ProcessMonitor`  
*method*), 48

## L

label (*aiidalab\_widgets\_base.CodeDatabaseWidget attribute*), 40 mol\_from\_smiles () (*aiidalab\_widgets\_base.SmilesWidget method*), 51

label (*aiidalab\_widgets\_base.codes.AiiDACodeSetup attribute*), 10 mol\_from\_smiles () (*aiidalab\_widgets\_base.structures.SmilesWidget method*), 28

label (*aiidalab\_widgets\_base.ComputerDatabaseWidget attribute*), 42 mpprocs\_per\_machine (*aiidalab\_widgets\_base.AiidaComputerSetup attribute*), 12

label (*aiidalab\_widgets\_base.databases.CodeDatabaseWidget attribute*), 15 mpprocs\_per\_machine (*aiidalab\_widgets\_base.ComputerDatabaseWidget attribute*), 42

label (*aiidalab\_widgets\_base.databases.ComputerDatabaseWidget attribute*), 16 mpprocs\_per\_machine (*aiidalab\_widgets\_base.ComputerDatabaseWidget attribute*), 12

LigandSelectorWidget (class in *aiidalab\_widgets\_base.data*), 9 mpprocs\_per\_machine (*aiidalab\_widgets\_base.computers.AiidaComputerSetup attribute*), 12

list\_to\_string\_range () (in module *aiidalab\_widgets\_base.utils*), 9 mpprocs\_per\_machine (*aiidalab\_widgets\_base.databases.ComputerDatabaseWidget attribute*), 16

## M

make\_ase () (*aiidalab\_widgets\_base.SmilesWidget method*), 51 mpirun\_command (*aiidalab\_widgets\_base.AiidaComputerSetup attribute*), 38

make\_ase () (*aiidalab\_widgets\_base.structures.SmilesWidget method*), 28 mpirun\_command (*aiidalab\_widgets\_base.ComputerDatabaseWidget attribute*), 42

mirror () (*aiidalab\_widgets\_base.BasicStructureEditor method*), 39 mpirun\_command (*aiidalab\_widgets\_base.computers.AiidaComputerSetup attribute*), 12

mirror () (*aiidalab\_widgets\_base.structures.BasicStructureEditor method*), 27 mpirun\_command (*aiidalab\_widgets\_base.databases.ComputerDatabaseWidget attribute*), 16

mirror\_3p () (*aiidalab\_widgets\_base.BasicStructureEditor method*), 39 mpirun\_command (*aiidalab\_widgets\_base.databases.ComputerDatabaseWidget attribute*), 16

mirror\_3p () (*aiidalab\_widgets\_base.structures.BasicStructureEditor method*), 27

mod\_element () (*aiidalab\_widgets\_base.BasicStructureEditor method*), 39 MultiStructureUploadWidget (class in *aiidalab\_widgets\_base*), 44

mod\_element () (*aiidalab\_widgets\_base.structures.BasicStructureEditor method*), 27 MultiStructureUploadWidget (class in *aiidalab\_widgets\_base.structures\_multi*), 31

## N

module name\_operator () (*aiidalab\_widgets\_base.viewers.StructureDataViewer method*), 33

aiidalab\_widgets\_base, 37 node\_class (*aiidalab\_widgets\_base.StructureManagerWidget attribute*), 54

aiidalab\_widgets\_base.codes, 10 node\_class (*aiidalab\_widgets\_base.structures.StructureManagerWidget attribute*), 30

aiidalab\_widgets\_base.computers, 11 node\_class () (*aiidalab\_widgets\_base.MultiStructureUploadWidget property*), 44

aiidalab\_widgets\_base.data, 9 node\_class () (*aiidalab\_widgets\_base.structures\_multi.MultiStructureUploadWidget property*), 31

aiidalab\_widgets\_base.databases, 15 NODE\_TYPE (*aiidalab\_widgets\_base.nodes.NodesTreeWidget attribute*), 20

aiidalab\_widgets\_base.dicts, 18 NODE\_TYPE (*aiidalab\_widgets\_base.NodesTreeWidget attribute*), 45

aiidalab\_widgets\_base.export, 18 nodes (*aiidalab\_widgets\_base.nodes.NodesTreeWidget attribute*), 20

aiidalab\_widgets\_base.misc, 18

aiidalab\_widgets\_base.nodes, 19

aiidalab\_widgets\_base.process, 21

aiidalab\_widgets\_base.structures, 26

aiidalab\_widgets\_base.structures\_multi, 31

aiidalab\_widgets\_base.utils, 9

aiidalab\_widgets\_base.viewers, 31

aiidalab\_widgets\_base.wizard, 34

nodes (*aiidalab\_widgets\_base.NodesTreeWidget* attribute), 45

NodesTreeWidget (class in *aiidalab\_widgets\_base*), 45

NodesTreeWidget (class in *aiidalab\_widgets\_base.nodes*), 20

not\_operator() (*aiidalab\_widgets\_base.viewers.StructureDataViewer* method), 33

num\_cores\_per\_mpiPROC (*aiidalab\_widgets\_base.ComputerDatabaseWidget* attribute), 42

num\_cores\_per\_mpiPROC (*aiidalab\_widgets\_base.databases.ComputerDatabaseWidget* attribute), 16

## O

on\_btn\_submit\_press() (*aiidalab\_widgets\_base.process.SubmitButtonWidget* method), 26

on\_btn\_submit\_press() (*aiidalab\_widgets\_base.SubmitButtonWidget* method), 55

on\_click() (*aiidalab\_widgets\_base.process.SubmitButtonWidget* method), 26

on\_click() (*aiidalab\_widgets\_base.SubmitButtonWidget* method), 55

on\_completed() (*aiidalab\_widgets\_base.process.ProcessFollowerWidget* method), 22

on\_completed() (*aiidalab\_widgets\_base.ProcessFollowerWidget* method), 47

on\_computer (*aiidalab\_widgets\_base.CodeDatabaseWidget* attribute), 40

on\_computer (*aiidalab\_widgets\_base.databases.CodeDatabaseWidget* attribute), 15

on\_setup\_ssh() (*aiidalab\_widgets\_base.computers.SshComputerSetup* method), 14

on\_setup\_ssh() (*aiidalab\_widgets\_base.SshComputerSetup* method), 52

on\_submitted() (*aiidalab\_widgets\_base.process.SubmitButtonWidget* method), 26

on\_submitted() (*aiidalab\_widgets\_base.SubmitButtonWidget* method), 55

on\_use\_diff\_proxy\_username\_change() (*aiidalab\_widgets\_base.computers.SshComputerSetup* method), 14

on\_use\_diff\_proxy\_username\_change() (*aiidalab\_widgets\_base.SshComputerSetup* method), 52

on\_use\_verification\_mode\_change() (*aiidalab\_widgets\_base.computers.SshComputerSetup* method), 14

on\_use\_verification\_mode\_change() (*aiidalab\_widgets\_base.SshComputerSetup* method), 52

OptiMakeQueryWidget (class in *aiidalab\_widgets\_base*), 45

OptiMakeQueryWidget (class in *aiidalab\_widgets\_base.databases*), 17

outgoing\_node (*aiidalab\_widgets\_base.process.ProcessListWidget* attribute), 23

outgoing\_node (*aiidalab\_widgets\_base.ProcessListWidget* attribute), 48

## P

parse\_advanced\_sel() (*aiidalab\_widgets\_base.viewers.StructureDataViewer* method), 33

parse\_infix\_notation() (*aiidalab\_widgets\_base.misc.ReversePolishNotation* static method), 19

past\_days (*aiidalab\_widgets\_base.process.ProcessListWidget* attribute), 23

past\_days (*aiidalab\_widgets\_base.ProcessListWidget* attribute), 48

ports (*aiidalab\_widgets\_base.ComputerDatabaseWidget* attribute), 42

port (*aiidalab\_widgets\_base.computers.SshComputerSetup* attribute), 14

port (*aiidalab\_widgets\_base.databases.ComputerDatabaseWidget* attribute), 16

port (*aiidalab\_widgets\_base.SshComputerSetup* attribute), 52

predefine\_settings() (in module *aiidalab\_widgets\_base.utils*), 9

prepend\_text (*aiidalab\_widgets\_base.AiiDACodeSetup* attribute), 37

prepend\_text (*aiidalab\_widgets\_base.AiidaComputerSetup* attribute), 38

prepend\_text (*aiidalab\_widgets\_base.CodeDatabaseWidget* attribute), 40

prepend\_text (*aiidalab\_widgets\_base.codes.AiiDACodeSetup* attribute), 10

prepend_text (aiidalab_widgets_base.ComputerDatabaseWidget attribute), 23			
attribute), 42	process_label		(ai-
prepend_text (aiidalab_widgets_base.computers.AiidaComputerDatabaseWidget attribute), 12	aiidalab_widgets_base.ProcessListWidget attribute), 48		
prepend_text (aiidalab_widgets_base.databases.CodeDatabaseWidget attribute), 15	PROCESS_STATE_STYLE		(ai-
attribute), 16	aiidalab_widgets_base.nodes.NodesTreeWidget		
prepend_text (aiidalab_widgets_base.databases.ComputerDatabaseWidget attribute), 20	PROCESS_STATE_STYLE		(ai-
attribute), 16	aiidalab_widgets_base.ProcessListWidget		
preprocess () (aiidalab_widgets_base.StructureBrowserWidget method), 53	aiidalab_widgets_base.NodesTreeWidget attribute), 45		at-
preprocess () (aiidalab_widgets_base.structures.StructureBrowserWidget method), 28	PROCESS_STATE_STYLE_DEFAULT		(ai-
process (aiidalab_widgets_base.process.ProcessCallStackWidget attribute), 22	aiidalab_widgets_base.nodes.NodesTreeWidget		
attribute), 22	PROCESS_STATE_STYLE_DEFAULT		(ai-
process (aiidalab_widgets_base.process.ProcessFollowerWidget attribute), 22	aiidalab_widgets_base.NodesTreeWidget attribute), 45		at-
process (aiidalab_widgets_base.process.ProcessInputsWidget attribute), 22	process_states		(ai-
attribute), 22	aiidalab_widgets_base.process.ProcessListWidget		
process (aiidalab_widgets_base.process.ProcessMonitor attribute), 24	attribute), 23		
attribute), 24	process_states		(ai-
process (aiidalab_widgets_base.process.ProcessNodesTreeWidget attribute), 24	aiidalab_widgets_base.ProcessListWidget attribute), 48		
process (aiidalab_widgets_base.process.ProcessOutputsWidget attribute), 24	ProcessCallStackWidget (class in ai-		
attribute), 24	idalab_widgets_base), 46		
process (aiidalab_widgets_base.process.ProcessReportWidget attribute), 25	ProcessCallStackWidget (class in ai-		
attribute), 25	idalab_widgets_base.process), 21		
process (aiidalab_widgets_base.process.ProgressBarWidget attribute), 25	ProcessFollowerWidget (class in ai-		
attribute), 25	idalab_widgets_base), 46		
process (aiidalab_widgets_base.process.RunningCalcJobOutputWidget attribute), 25	ProcessFollowerWidget (class in ai-		
attribute), 25	idalab_widgets_base.process), 22		
process (aiidalab_widgets_base.process.SubmitButtonWidget attribute), 26	ProcessInputsWidget (class in ai-		
attribute), 26	idalab_widgets_base), 47		
process (aiidalab_widgets_base.ProcessCallStackWidget attribute), 46	ProcessInputsWidget (class in ai-		
attribute), 46	idalab_widgets_base.process), 22		
process (aiidalab_widgets_base.ProcessFollowerWidget attribute), 47	ProcessListWidget (class in ai-		
attribute), 47	idalab_widgets_base), 47		
process (aiidalab_widgets_base.ProcessInputsWidget attribute), 47	ProcessListWidget (class in ai-		
attribute), 47	idalab_widgets_base.process), 23		
process (aiidalab_widgets_base.ProcessMonitor attribute), 48	ProcessMonitor (class in aiidalab_widgets_base),		
attribute), 48	48		
process (aiidalab_widgets_base.ProcessNodesTreeWidget attribute), 49	ProcessMonitor (class in ai-		
attribute), 49	idalab_widgets_base.process), 23		
process (aiidalab_widgets_base.ProcessOutputsWidget attribute), 49	ProcessNodesTreeWidget (class in ai-		
attribute), 49	idalab_widgets_base), 49		
process (aiidalab_widgets_base.ProcessReportWidget attribute), 50	ProcessNodesTreeWidget (class in ai-		
attribute), 50	idalab_widgets_base.process), 24		
process (aiidalab_widgets_base.ProgressBarWidget attribute), 50	ProcessOutputsWidget (class in ai-		
attribute), 50	idalab_widgets_base), 49		
process (aiidalab_widgets_base.RunningCalcJobOutputWidget attribute), 50	ProcessOutputsWidget (class in ai-		
attribute), 50	idalab_widgets_base.process), 24		
process (aiidalab_widgets_base.SubmitButtonWidget attribute), 55	ProcessReportWidget (class in ai-		
attribute), 55	idalab_widgets_base), 49		
process_label	(ai-	ProcessReportWidget (class in ai-	
aiidalab_widgets_base.process.ProcessListWidget		idalab_widgets_base.process), 25	

ProgressBarWidget (class in ai- refresh\_view() (ai-  
*idalab\_widgets\_base*), 50 *idalab\_widgets\_base.MultiStructureUploadWidget*  
 ProgressBarWidget (class in ai- method), 44  
*idalab\_widgets\_base.process*), 25 refresh\_view() (ai-  
 proxy\_command (ai- *idalab\_widgets\_base.structures\_multi.MultiStructureUploadWidget*  
*idalab\_widgets\_base.ComputerDatabaseWidget* method), 31  
 attribute), 42 register\_viewer\_widget() (in module ai-  
 proxy\_command (ai- *idalab\_widgets\_base*), 57  
*idalab\_widgets\_base.databases.ComputerDatabaseWidget* register\_viewer\_widget() (in module ai-  
 attribute), 16 *idalab\_widgets\_base.viewers*), 34  
 proxy\_hostname (ai- remote\_abs\_path (ai-  
*idalab\_widgets\_base.ComputerDatabaseWidget* *idalab\_widgets\_base.AiiDACodeSetup* at-  
 attribute), 42 tribute), 37  
 proxy\_hostname (ai- remote\_abs\_path (ai-  
*idalab\_widgets\_base.computers.SshComputerSetup* *idalab\_widgets\_base.CodeDatabaseWidget*  
 attribute), 14 attribute), 40  
 proxy\_hostname (ai- remote\_abs\_path (ai-  
*idalab\_widgets\_base.databases.ComputerDatabaseWidget* *idalab\_widgets\_base.codes.AiiDACodeSetup*  
 attribute), 16 attribute), 10  
 proxy\_hostname (ai- remote\_abs\_path (ai-  
*idalab\_widgets\_base.SshComputerSetup* *idalab\_widgets\_base.databases.CodeDatabaseWidget*  
 attribute), 52 attribute), 16  
 proxy\_username (ai- remove() (*aiidalab\_widgets\_base.BasicStructureEditor*  
*idalab\_widgets\_base.ComputerDatabaseWidget* method), 39  
 attribute), 42 remove() (*aiidalab\_widgets\_base.structures.BasicStructureEditor*  
 attribute), 27 method), 27  
 proxy\_username (ai- repeat (*aiidalab\_widgets\_base.viewers.StructureDataViewer*  
*idalab\_widgets\_base.computers.SshComputerSetup* attribute), 33  
 attribute), 14 attribute), 33  
 proxy\_username (ai- reset() (*aiidalab\_widgets\_base.wizard.WizardAppWidget*  
*idalab\_widgets\_base.databases.ComputerDatabaseWidget* method), 35  
 attribute), 16 reset() (*aiidalab\_widgets\_base.WizardAppWidget*  
 attribute), 56 method), 56  
 proxy\_username (ai- ReversePolishNotation (class in ai-  
*idalab\_widgets\_base.SshComputerSetup* *idalab\_widgets\_base.misc*), 18  
 attribute), 52 rotate() (*aiidalab\_widgets\_base.BasicStructureEditor*  
 attribute), 39 method), 39  
**Q** rotate() (*aiidalab\_widgets\_base.data.LigandSelectorWidget*  
 queue\_name (*aiidalab\_widgets\_base.ComputerDatabaseWidget* method), 9  
 attribute), 42 method), 9  
 queue\_name (*aiidalab\_widgets\_base.databases.ComputerDatabaseWidget* rotate() (*aiidalab\_widgets\_base.structures.BasicStructureEditor*  
 attribute), 17 method), 27  
 attribute), 17 method), 27  
**R** RunningCalcJobOutputWidget (class in ai-  
 READY (*aiidalab\_widgets\_base.wizard.WizardAppWidgetStep.State* *idalab\_widgets\_base*), 50  
 attribute), 36 *idalab\_widgets\_base.process*), 25  
 READY (*aiidalab\_widgets\_base.WizardAppWidgetStep.State* attribute), 57  
**S**  
 refresh() (*aiidalab\_widgets\_base.CodeDropdown* safe\_interval (ai-  
 method), 41 *idalab\_widgets\_base.AiidaComputerSetup*  
 attribute), 38  
 refresh() (*aiidalab\_widgets\_base.codes.CodeDropdown* safe\_interval (ai-  
 method), 11 *idalab\_widgets\_base.ComputerDatabaseWidget*  
 attribute), 42  
 refresh() (*aiidalab\_widgets\_base.ComputerDropdown* safe\_interval (ai-  
 method), 44 *idalab\_widgets\_base.ComputerDatabaseWidget*  
 attribute), 42  
 refresh() (*aiidalab\_widgets\_base.computers.ComputerDropdown* safe\_interval (ai-  
 method), 13 *idalab\_widgets\_base.computers.AiidaComputerSetup*  
 attribute), 13



attribute), 12		idalab_widgets_base.ProcessNodesTreeWidget
safe_interval	(ai-	attribute), 49
idalab_widgets_base.databases.ComputerDatabaseWidget	selection (aiidalab_widgets_base.BasicStructureEditor	attribute), 39
attribute), 17		attribute), 39
scheduler (aiidalab_widgets_base.AiidaComputerSetup	selection (aiidalab_widgets_base.structures.BasicStructureEditor	attribute), 27
attribute), 38		attribute), 27
scheduler (aiidalab_widgets_base.ComputerDatabaseWidget	selection (aiidalab_widgets_base.viewers._StructureDataBaseViewer	attribute), 34
attribute), 42		attribute), 34
scheduler (aiidalab_widgets_base.computers.AiidaComputerSetup	on_adv	(ai-
attribute), 12		idalab_widgets_base.viewers._StructureDataBaseViewer
scheduler (aiidalab_widgets_base.databases.ComputerDatabaseWidget	attribute), 34	
attribute), 17	setup_counter	(ai-
search () (aiidalab_widgets_base.StructureBrowserWidget	idalab_widgets_base.computers.SshComputerSetup	
method), 53	attribute), 14	
search () (aiidalab_widgets_base.structures.StructureBrowserWidget	counter	(ai-
method), 28	idalab_widgets_base.SshComputerSetup	
sel2com () (aiidalab_widgets_base.BasicStructureEditor	attribute), 52	
method), 39	shebang (aiidalab_widgets_base.ComputerDatabaseWidget	
sel2com () (aiidalab_widgets_base.structures.BasicStructureEditor	attribute), 43	
method), 27	shebang (aiidalab_widgets_base.databases.ComputerDatabaseWidget	
select_structure ()	(ai-	attribute), 17
idalab_widgets_base.MultiStructureUploadWidget	show_selected_input ()	(ai-
method), 45	idalab_widgets_base.process.ProcessInputsWidget	
select_structure ()	(ai-	method), 23
idalab_widgets_base.structures_multi.MultiStructureUploadWidget	show_selected_input ()	(ai-
method), 31	idalab_widgets_base.ProcessInputsWidget	
selected_code	(ai-	method), 47
idalab_widgets_base.CodeDropdown	at-	show_selected_output ()
tribute), 41		(ai-
selected_code	(ai-	method), 25
idalab_widgets_base.codes.CodeDropdown	show_selected_output ()	(ai-
attribute), 11	idalab_widgets_base.ProcessOutputsWidget	
selected_computer	(ai-	method), 49
idalab_widgets_base.ComputerDropdown	SmilesWidget (class in aiidalab_widgets_base), 50	
attribute), 44	SmilesWidget (class in ai-	
selected_computer	(ai-	idalab_widgets_base.structures), 28
idalab_widgets_base.computers.ComputerDropdown	SPINNER (aiidalab_widgets_base.SmilesWidget	at-
attribute), 13	tribute), 51	
selected_index	(ai-	SPINNER (aiidalab_widgets_base.structures.SmilesWidget
idalab_widgets_base.wizard.WizardAppWidget	attribute), 28	
attribute), 35	SshComputerSetup (class in ai-	
selected_index	(ai-	idalab_widgets_base), 51
idalab_widgets_base.WizardAppWidget	at-	SshComputerSetup (class in ai-
tribute), 56		idalab_widgets_base.computers), 13
selected_nodes	(ai-	start_autoupdate ()
idalab_widgets_base.nodes.NodesTreeWidget	idalab_widgets_base.process.ProcessListWidget	(ai-
attribute), 21	method), 23	
selected_nodes	(ai-	start_autoupdate ()
idalab_widgets_base.NodesTreeWidget	at-	idalab_widgets_base.ProcessListWidget
tribute), 45		method), 48
selected_nodes	(ai-	state (aiidalab_widgets_base.wizard.WizardAppWidgetStep
idalab_widgets_base.process.ProcessNodesTreeWidget	attribute), 36	
tribute), 24	state (aiidalab_widgets_base.WizardAppWidgetStep	
selected_nodes	(ai-	attribute), 57

store\_structure() (ai- attribute), 54  
 idalab\_widgets\_base.MultiStructureUploadWidgetstructure\_node (ai-  
 method), 45 idalab\_widgets\_base.structures.StructureManagerWidget

store\_structure() (ai- attribute), 30  
 idalab\_widgets\_base.StructureManagerWidget StructureBrowserWidget (class in ai-  
 method), 54 idalab\_widgets\_base), 52

store\_structure() (ai- StructureBrowserWidget (class in ai-  
 idalab\_widgets\_base.structures.StructureManagerWidget idalab\_widgets\_base.structures), 28  
 method), 30 StructureDataViewer (class in ai-  
 idalab\_widgets\_base.viewers), 32

store\_structure() (ai- idalab\_widgets\_base.viewers), 32  
 idalab\_widgets\_base.structures\_multi.MultiStructureUploadWidgetExamplesWidget (class in ai-  
 method), 31 idalab\_widgets\_base), 53

str2vec() (aiidalab\_widgets\_base.BasicStructureEditor StructureExamplesWidget (class in ai-  
 method), 39 idalab\_widgets\_base.structures), 29

str2vec() (aiidalab\_widgets\_base.structures.BasicStructureEditor StructureManagerWidget (class in ai-  
 method), 27 idalab\_widgets\_base), 53

string\_range\_to\_list() (in module ai- StructureManagerWidget (class in ai-  
 idalab\_widgets\_base.utils), 10 idalab\_widgets\_base.structures), 29

structure (aiidalab\_widgets\_base.BasicStructureEditor StructureUploadWidget (class in ai-  
 attribute), 39 idalab\_widgets\_base), 54

structure (aiidalab\_widgets\_base.CodQueryWidget StructureUploadWidget (class in ai-  
 attribute), 40 idalab\_widgets\_base.structures), 30

structure (aiidalab\_widgets\_base.databases.CodQueryWidget SubmitButtonWidget (class in ai-  
 attribute), 15 idalab\_widgets\_base), 55

structure (aiidalab\_widgets\_base.databases.OptimadeQueryWidget SubmitButtonWidget (class in ai-  
 attribute), 17 idalab\_widgets\_base.process), 26

structure (aiidalab\_widgets\_base.OptimadeQueryWidget SUCCESS (aiidalab\_widgets\_base.wizard.WizardAppWidgetStep.State  
 attribute), 46 attribute), 36

structure (aiidalab\_widgets\_base.SmilesWidget at- SUCCESS (aiidalab\_widgets\_base.WizardAppWidgetStep.State  
 tribute), 51 attribute), 57

structure (aiidalab\_widgets\_base.StructureBrowserWidget percentcell (aiidalab\_widgets\_base.viewers.\_StructureDataBaseViewer  
 attribute), 53 attribute), 34

structure (aiidalab\_widgets\_base.StructureExamplesWidget SUPPORTED\_DATA\_FORMATS (ai-  
 attribute), 53 idalab\_widgets\_base.StructureManagerWidget  
 attribute), 53

structure (aiidalab\_widgets\_base.StructureManagerWidget SUPPORTED\_DATA\_FORMATS (ai-  
 attribute), 54 idalab\_widgets\_base.structures.StructureManagerWidget  
 attribute), 27 attribute), 29

structure (aiidalab\_widgets\_base.structures.SmilesWidget  
 attribute), 28

structure (aiidalab\_widgets\_base.structures.StructureBrowserWidget (aiidalab\_widgets\_base.AiidaComputerSetup  
 attribute), 28 method), 38

structure (aiidalab\_widgets\_base.structures.StructureExamplesWidget (aiidalab\_widgets\_base.computers.AiidaComputerSetup  
 attribute), 29 method), 12

structure (aiidalab\_widgets\_base.structures.StructureManagerWidget percentcell (aiidalab\_widgets\_base.viewers.\_StructureDataBaseViewer  
 attribute), 30 attribute), 34

structure (aiidalab\_widgets\_base.structures.StructureUploadWidget upload\_base\_dr () (ai-  
 attribute), 30 idalab\_widgets\_base.BasicStructureEditor  
 method), 39

structure (aiidalab\_widgets\_base.StructureUploadWidget  
 attribute), 55

structure (aiidalab\_widgets\_base.viewers.StructureDataViewer translate\_dr () (ai-  
 attribute), 33 idalab\_widgets\_base.structures.BasicStructureEditor  
 method), 27

structure\_node (ai- translate\_dx dy dz () (ai-  
 idalab\_widgets\_base.StructureManagerWidget idalab\_widgets\_base.BasicStructureEditor

method), 39  
translate\_dx dy dz () (ai- idalab\_widgets\_base.structures.BasicStructureEditor method), 27  
translate\_to\_xyz () (ai- idalab\_widgets\_base.BasicStructureEditor method), 39  
translate\_to\_xyz () (ai- idalab\_widgets\_base.structures.BasicStructureEditor method), 28  
transport (aiidalab\_widgets\_base.AiidaComputerSetup attribute), 38  
transport (aiidalab\_widgets\_base.ComputerDatabaseWidget attribute), 43  
transport (aiidalab\_widgets\_base.computers.AiidaComputerSetup attribute), 12  
transport (aiidalab\_widgets\_base.databases.ComputerDatabaseWidget attribute), 17  
update () (aiidalab\_widgets\_base.ProcessCallStackWidget method), 46  
update () (aiidalab\_widgets\_base.ProcessFollowerWidget method), 47  
update () (aiidalab\_widgets\_base.ProcessListWidget method), 48  
update () (aiidalab\_widgets\_base.ProcessNodesTreeWidget method), 49  
update () (aiidalab\_widgets\_base.ProcessReportWidget method), 50  
update () (aiidalab\_widgets\_base.ProgressBarWidget method), 50  
update () (aiidalab\_widgets\_base.RunningCalc.JobOutputWidget method), 50  
update\_codes () (ai- idalab\_widgets\_base.CodeDatabaseWidget method), 40  
update\_codes () (ai- idalab\_widgets\_base.databases.CodeDatabaseWidget method), 16  
update\_computers () (ai- idalab\_widgets\_base.CodeDatabaseWidget method), 41  
update\_computers () (ai- idalab\_widgets\_base.ComputerDatabaseWidget method), 43  
update\_computers () (ai- idalab\_widgets\_base.databases.CodeDatabaseWidget method), 16  
update\_computers () (ai- idalab\_widgets\_base.databases.ComputerDatabaseWidget method), 17  
update\_computers () (ai- idalab\_widgets\_base.CodeDatabaseWidget method), 41  
update\_computers () (ai- idalab\_widgets\_base.ComputerDatabaseWidget method), 43  
update\_computers () (ai- idalab\_widgets\_base.databases.CodeDatabaseWidget method), 16  
update\_computers () (ai- idalab\_widgets\_base.databases.ComputerDatabaseWidget method), 17  
update\_settings () (ai- idalab\_widgets\_base.CodeDatabaseWidget method), 41  
update\_settings () (ai- idalab\_widgets\_base.ComputerDatabaseWidget method), 43  
update\_settings () (ai- idalab\_widgets\_base.databases.CodeDatabaseWidget method), 16  
update\_settings () (ai- idalab\_widgets\_base.databases.ComputerDatabaseWidget method), 17  
use\_login\_shell (ai- idalab\_widgets\_base.ComputerDatabaseWidget attribute), 43  
use\_login\_shell (ai- idalab\_widgets\_base.databases.ComputerDatabaseWidget attribute), 17  
use\_proxy (aiidalab\_widgets\_base.computers.SshComputerSetup attribute), 14  
use\_proxy (aiidalab\_widgets\_base.SshComputerSetup attribute), 52

U

undo () (aiidalab\_widgets\_base.StructureManagerWidget method), 54  
undo () (aiidalab\_widgets\_base.structures.StructureManagerWidget method), 30  
UnknownTypeTreeNode (class in ai- idalab\_widgets\_base.nodes), 21  
update () (aiidalab\_widgets\_base.CodeDatabaseWidget method), 40  
update () (aiidalab\_widgets\_base.ComputerDatabaseWidget method), 43  
update () (aiidalab\_widgets\_base.databases.CodeDatabaseWidget method), 16  
update () (aiidalab\_widgets\_base.databases.ComputerDatabaseWidget method), 17  
update () (aiidalab\_widgets\_base.nodes.NodesTreeWidget method), 21  
update () (aiidalab\_widgets\_base.NodesTreeWidget method), 45  
update () (aiidalab\_widgets\_base.process.Calc.JobOutputWidget method), 21  
update () (aiidalab\_widgets\_base.process.ProcessCallStackWidget method), 22  
update () (aiidalab\_widgets\_base.process.ProcessFollowerWidget method), 22  
update () (aiidalab\_widgets\_base.process.ProcessListWidget method), 23  
update () (aiidalab\_widgets\_base.process.ProcessNodesTreeWidget method), 24  
update () (aiidalab\_widgets\_base.process.ProcessReportWidget method), 25  
update () (aiidalab\_widgets\_base.process.ProgressBarWidget method), 25  
update () (aiidalab\_widgets\_base.process.RunningCalc.JobOutputWidget method), 26



username (*aiidalab\_widgets\_base.computers.SshComputerSetup*  
attribute), 14

username (*aiidalab\_widgets\_base.SshComputerSetup*  
attribute), 52

## V

valid\_arguments() (in module *ai-*  
*idalab\_widgets\_base.utils*), 10

value (*aiidalab\_widgets\_base.misc.CopyToClipboardButton*  
attribute), 18

value (*aiidalab\_widgets\_base.viewers.DictViewer* at-  
tribute), 32

vec2str() (*aiidalab\_widgets\_base.BasicStructureEditor*  
method), 39

vec2str() (*aiidalab\_widgets\_base.structures.BasicStructureEditor*  
method), 28

viewer() (in module *aiidalab\_widgets\_base*), 58

viewer() (in module *aiidalab\_widgets\_base.viewers*),  
34

## W

WizardAppWidget (class in *aiidalab\_widgets\_base*),  
55

WizardAppWidget (class in *ai-*  
*idalab\_widgets\_base.wizard*), 34

WizardAppWidgetStep (class in *ai-*  
*idalab\_widgets\_base*), 56

WizardAppWidgetStep (class in *ai-*  
*idalab\_widgets\_base.wizard*), 35

WizardAppWidgetStep.State (class in *ai-*  
*idalab\_widgets\_base*), 56

WizardAppWidgetStep.State (class in *ai-*  
*idalab\_widgets\_base.wizard*), 35

work\_dir (*aiidalab\_widgets\_base.AiidaComputerSetup*  
attribute), 38

work\_dir (*aiidalab\_widgets\_base.ComputerDatabaseWidget*  
attribute), 43

work\_dir (*aiidalab\_widgets\_base.computers.AiidaComputerSetup*  
attribute), 12

work\_dir (*aiidalab\_widgets\_base.databases.ComputerDatabaseWidget*  
attribute), 17

WorkChainProcessTreeNode (class in *ai-*  
*idalab\_widgets\_base.nodes*), 21