
netdef Documentation

Release 1.0.7

Frode Holmer

Jul 20, 2021

User guide:

1	Netdef	3
1.1	Summary	3
1.2	Features	3
1.3	Use Cases	4
1.4	Getting started	4
2	Application architecture	5
3	Installation	10
4	Quickstart	11
4.1	Make-project	11
4.2	Setup your application	11
4.3	Launch application	12
4.4	Examples	12
5	Webadmin	13
5.1	Override root endpoint	15
5.2	Override <i>Webadmin</i> → <i>Home</i>	15
5.3	Override <i>Webadmin</i> → <i>Tools</i>	16
6	Configuration	17
6.1	Extended interpolation	18
6.2	Default configs	21
6.3	Built-in Controllers and Rules	23
7	Advanced	24
7.1	Project layout	24
7.2	Add a controller	27
7.3	Add a rule	28
7.4	Create a custom controller	28
7.5	Create a custom source	30
7.6	Create a custom rule	32
7.7	Putting it all together	34
8	Credits	40
8.1	Contributors	40

9	Changelog	40
9.1	1.0.7	40
9.2	1.0.6	41
9.3	1.0.5	41
9.4	1.0.4	42
9.5	1.0.3	42
9.6	1.0.2	43
9.7	1.0.1	43
9.8	1.0.0	43
10	Build from source	43
10.1	Python	43
10.2	psutil	44
10.3	Netdef package	44
10.4	Docs	45
11	Built-in configs	46
11.1	Controller configs	46
11.2	Rule configs	51
12	netdef package	54
12.1	netdef.__main__	54
12.2	netdef.service	55
12.3	netdef.windows_service	56
12.4	netdef.systemd_service	58
12.5	netdef.utils	59
12.6	netdef.testutils	60
13	netdef.Controllers package	61
13.1	Controllers	62
13.2	Abstract base controllers	62
13.3	Built-in controller modules	66
14	netdef.Engines package	84
14.1	Abstract baseclass	85
14.2	Expressions	86
14.3	Built-in engine modules	88
14.4	Webadmin	89
15	netdef.Interfaces package	104
15.1	Abstract base	104
15.2	Internal classes	105
15.3	Built-in Interfaces	105
16	netdef.Rules package	108
16.1	Rules	108
16.2	Abstract base	109
16.3	Built-in rule modules	112
17	netdef.Shared package	114
17.1	Internal	114
17.2	Shared	114
17.3	SharedConfig	115
17.4	SharedExpressions	115
17.5	SharedQueues	116

17.6 SharedSources	118
18 netdef.Sources package	118
18.1 Sources	119
18.2 Abstract base	119
18.3 Built-in Interfaces	122
19 Indices and tables	127
Python Module Index	128
Index	130

1 Netdef

- Documentation: <https://netdef.readthedocs.io/en/latest/>
- GitHub: <https://github.com/fholmer/netdef>
- GitLab: <https://gitlab.com/fholmer/netdef>
- PyPI: <https://pypi.org/project/netdef/>
- License: GNU Lesser General Public License v3 or later (LGPLv3+)

1.1 Summary

An application framework with built-in drivers (Controllers), data holders (Sources) and config parsers (Rules). Also includes a web interface for configuration and troubleshooting.

1.2 Features

- Abstract base classes for creating custom controllers, sources and rules
- The configuration is done using configparser with extended interpolation
- Start a new netdef project with cookiecutter or make-project. Templates available at <https://gitlab.com/fholmer/netdef-project/>
- Built-in Controllers:
 - OpcUa server / client (freeopcua)
 - TcpModbus server / client (pymodbus)
 - icmp ping / url ping
 - XmlRpc client
 - trigger events by using crontab format (crontab)
 - disk, memory and CPU monitoring (psutil)
 - MQTT client (using a simple messaging format called `DataMessage`) (paho-mqtt)

- Simple RESTJson client
- Simple Influxdb logger ([influxdb](#))
- Built-in Rules:
 - Generic CSV config parser
 - Generic INI config parser
 - Generic Yaml config parser ([PyYAML](#))
- Built-in application engines:
 - threaded engine with stdout/stderr only
 - threaded engine with web-interface (webadmin)
 - serve webadmin behind nginx reverse proxy

1.3 Use Cases

Netdef is useful if you want to create a middleware that can translate a protocol into a completely different protocol or data format into a completely different data format.

1.4 Getting started

First install make-project:

```
$ python3 -m pip install --user make
```

Create your first application:

```
$ python3 -m make project gl:fholmer/netdef-project/minimal-app
```

When asked for *project_name* type *Test-App*:

```
project_name? [First-App]: Test-App
```

Setup development environment for your application:

```
$ cd Test-App
$ python3 -m venv venv
$ source venv/bin/activate
$ pip install wheel
$ pip install -r requirements-dev.txt
$ pip install -r requirements.txt
$ python -m test_app -i .
```

Run:

```
$ python -m test_app -r .
```

CTRL-C to exit

Package your application:

```
$ python setup.py bdist_wheel
```

Exit development environment:

```
$ deactivate
```

Prepare deployment:

```
$ sudo mkdir -p /opt/test-app
$ sudo chown $USER:$USER /opt/test-app/
$ python3 -m venv /opt/test-app/
```

Deploy your application:

```
$ source /opt/test-app/bin/activate
$ pip install ./dist/Test_App-0.1.0-py3-none-any.whl
$ python -m test_app -i /opt/test-app/
```

Install as service:

```
$ sudo /opt/test-app/bin/Test-App-Service -u $USER --install /opt/test-app/
```

Enable and run:

```
$ sudo systemctl --system daemon-reload
$ sudo systemctl enable test-app-service.service
$ sudo systemctl start test-app-service.service
```

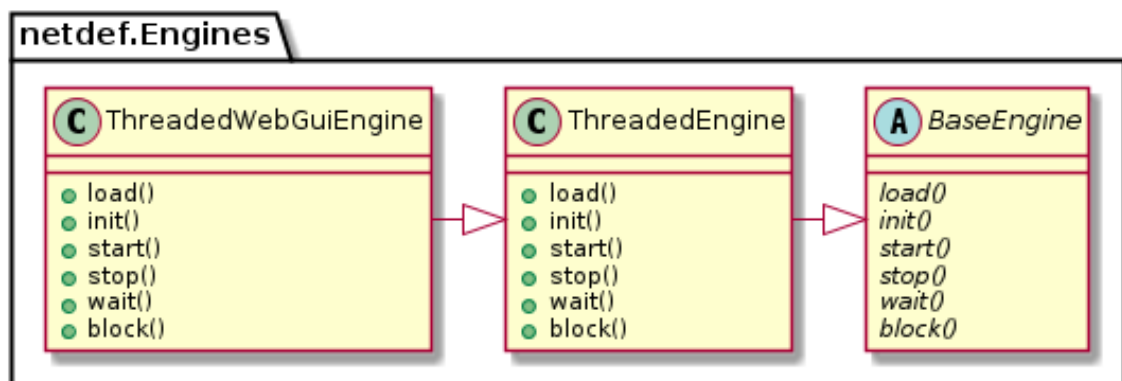
2 Application architecture

When you create an application using the Netdef framework your application consists of:

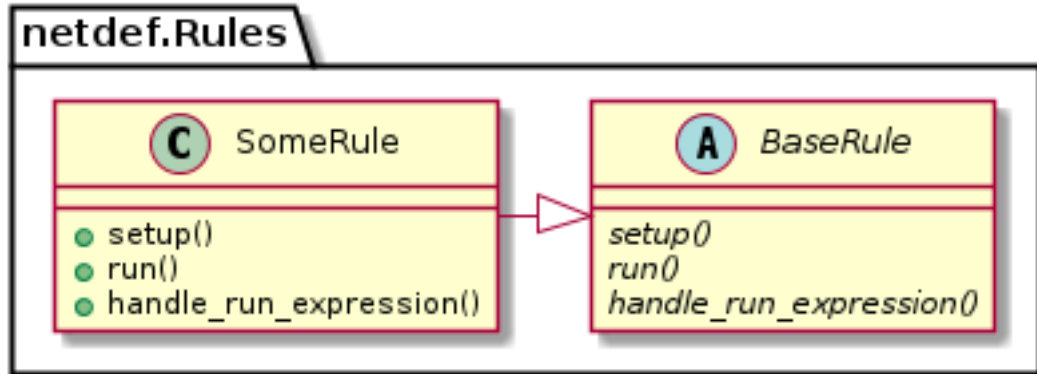
- Exactly one *engine*.
- At least one *rule*.
- At least one *source*.
- At least one *controller*.
- At least one *expression*.

Glossary

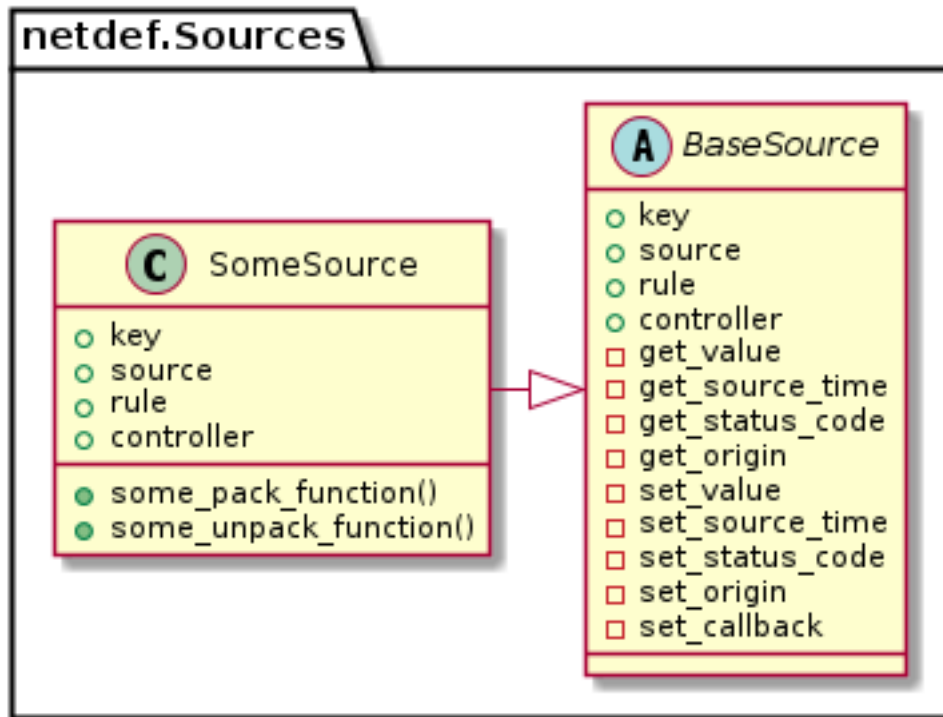
engine The engine is an instance of `netdef.Engines.ThreadedEngine`.



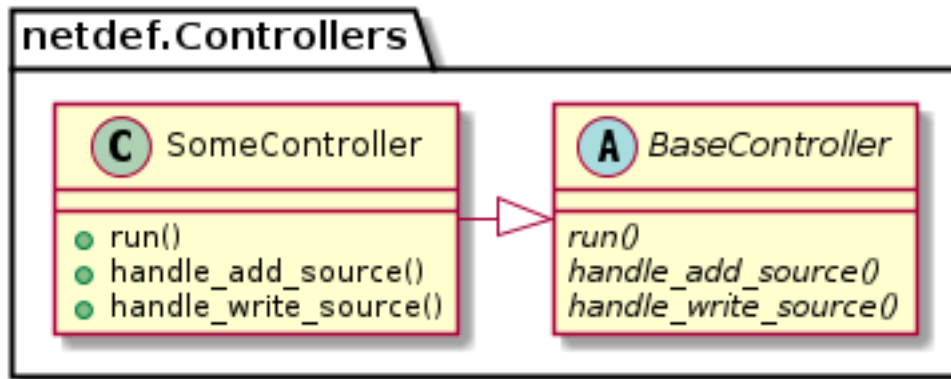
rule A rule is an instance derived from `netdef.Rules.BaseRule`.



source A source is an instance derived from `netdef.Sources.BaseSource`.



controller A controller is an instance derived from `netdef.Controllers.BaseController`.

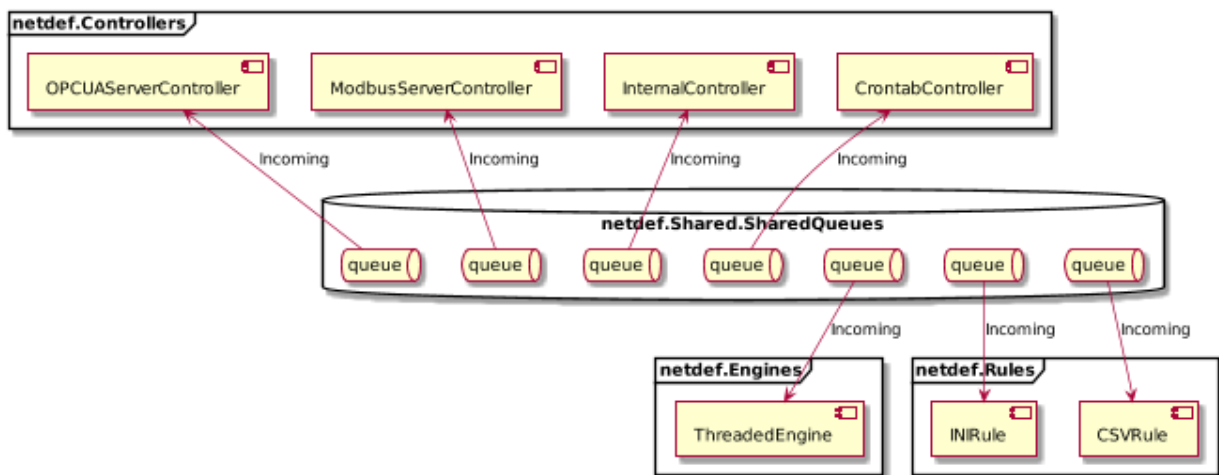


expression A python callable that is executed by engine when a associated source changes its value. The associated sources are arguments to the callable. See *netdef.Engines.expression.Expression*.

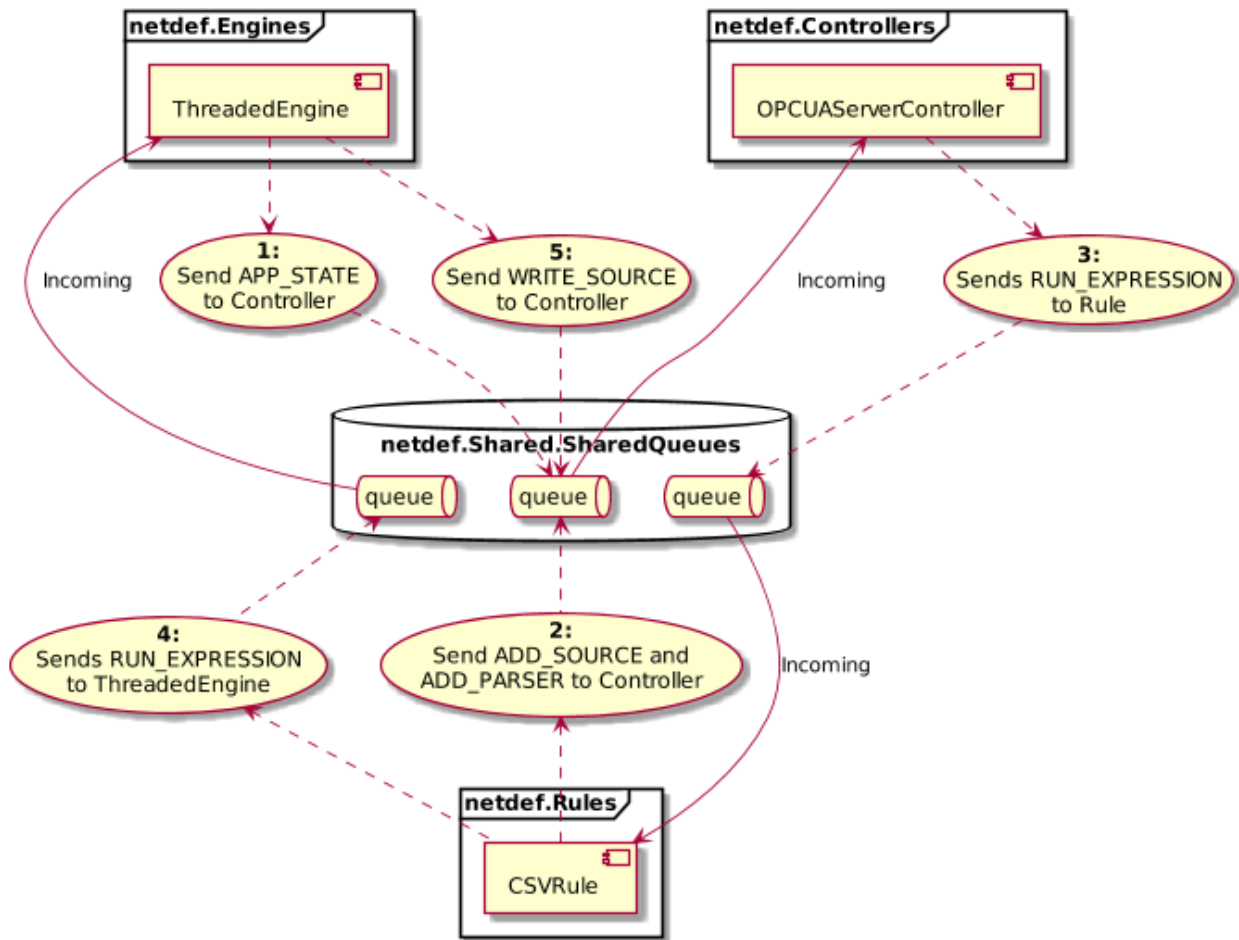
```
# Example:
def expression(arg1, arg2):
    print("expression was called")
    print("This is a netdef.Engines.expression.Expression.Argument:", arg1)
    print("This is the associated source instance:", arg1.instance)
    print("The name of the associated controller:", arg1.instance.controller)
```

Shared queues

All instances have their own *incoming* queue. This queue is available to the other instances in the shared object. See *netdef.Shared.SharedQueues.SharedQueues*



The instances communicate with each other by registering messages in the recipient's queue. The example below shows a project with one controller and one rule:



The most important message types in your application are APP_STATE, ADD_SOURCE, ADD_PARSER, WRITE_SOURCE and RUN_EXPRESSION. See `netdef.Shared.SharedQueues.MessageType`

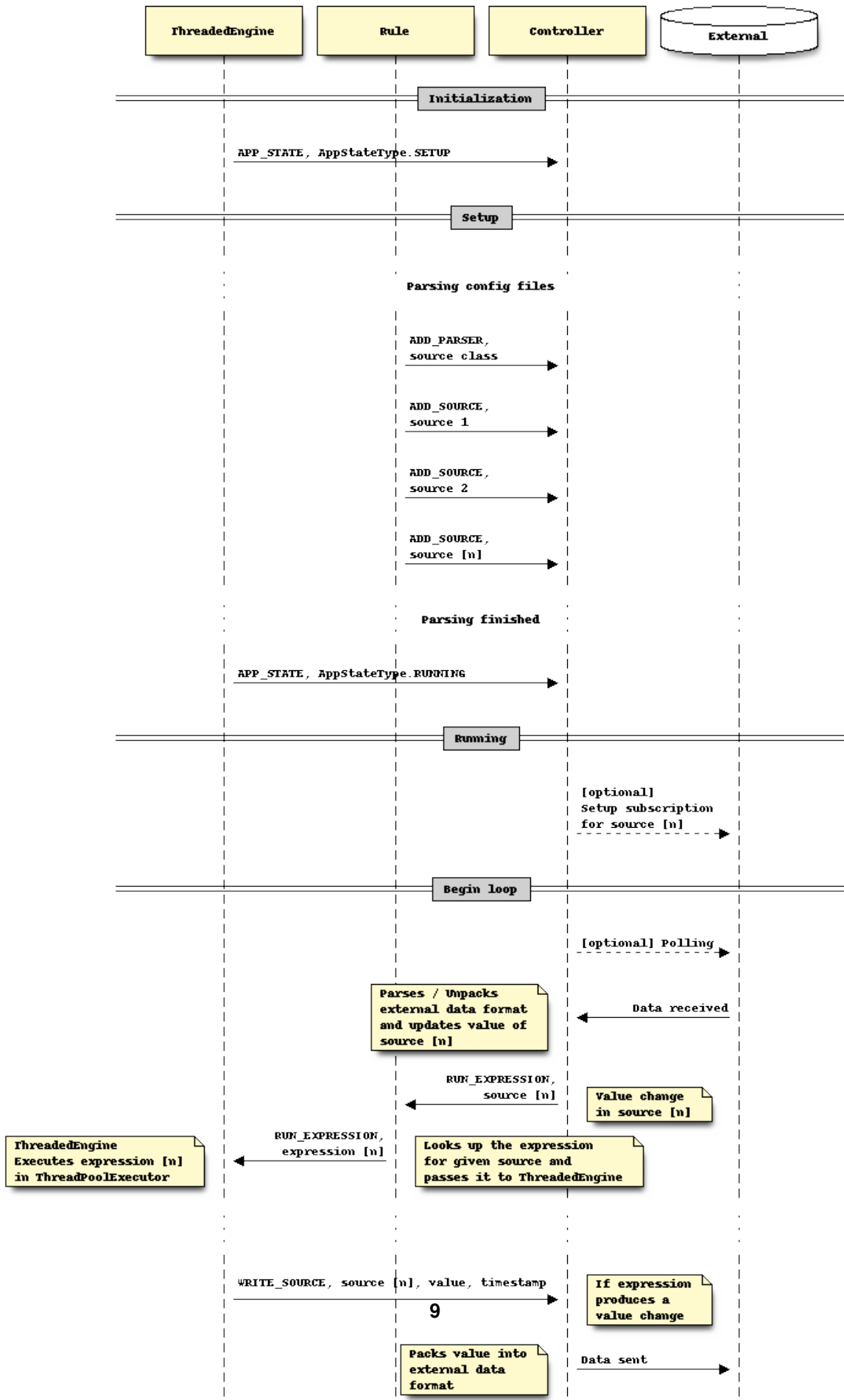
The message flow will in most cases be as follows:

At application initialization:

- The *engine* will send APP_STATE to all active controllers.
- Every *rule* will send ADD_PARSER or/and ADD_SOURCE to a specific *controller* depending on what is in the configuration files.
- The *engine* will send a new APP_STATE to all active controllers.

Repeats until application is terminated:

- Every *controller* will send RUN_EXPRESSION back to a specific *rule* on data changes.
- The specific *rule* will then collect the associated *expression* to be evaluated depending on given data change and send RUN_EXPRESSION to the *engine*.
- If the *expression* generate a new data change then a WRITE_SOURCE message is sent back directly to *controller*.



3 Installation

Netdef is implemented in [Python](#) and supports Python 3.5.3+.

Prerequisites

- Debian:

Python3 requirements can be installed by typing:

```
$ sudo apt-get install python3 python3-pip python3-venv
```

Requirements for building psutil:

```
$ sudo apt-get install build-essential python3-dev
```

Ensure you have installed python 3.5.3 or newer. You can check this by typing:

```
$ python3 -V  
Python 3.5.3
```

- Windows:

Ensure you have installed Python 3.5.3 or newer. You can check this by opening command prompt and type:

```
> py -3 -V  
Python 3.5.3
```

If `py.exe` is not found then you have to download and install [Python 3.5.3](#) or newer.

Create an Virtual environment

- Linux:

```
$ python3 -m venv venv
```

- Windows:

```
> py -3 -m venv venv
```

Activate the environment

- Linux:

```
$ source venv/bin/activate
```

- Windows:

```
> venv\Scripts\activate
```

Install Netdef

- Linux:

```
$ pip install netdef
```

- Windows:

```
> pip install netdef
```

4 Quickstart

Netdef will require a specific project structure:

```
/First-App
  /setup.py
  /config
    /default.conf
  /first_app
    /Controllers
    /Engines
      /__init__.py
      /templates
      /webadmin
    /Interfaces
    /Rules
    /Sources
    /__init__.py
    /__main__.py
    /defaultconfig.py
    /main.py
```

Pre made project templates are available using make-project or cookiecutter

4.1 Make-project

First install make-project:

```
$ python3 -m pip install make
```

Create your first application:

```
$ python3 -m make project gl:fholmer/netdef-project/minimal-app
```

The rest of this documentation assumes that your application is called First-App

4.2 Setup your application

Create a virtual environment for your application:

```
$ cd First-App
$ python3 -m venv venv
$ source venv/bin/activate
```

Install dependencies:

```
$ pip install -r requirements-dev.txt
$ pip install -r requirements.txt
```

Link your application into the virtual environment site-packages:

```
$ pip install -e .
```

Create config and log folders for your app:

```
$ First-App --init .
```

4.3 Launch application

There are several ways to run your application.

You can use the the entrypoint:

```
$ First-App --run .
```

Or you can use the package module:

```
$ python -m first_app --run .
```

There is also a simple launcher script:

```
$ python launchApp.py
```

You don't have to activate the virtual environment to run your application. You can run it directly by using absolute paths:

```
$ cd /  
$ [insert-abs-path-to-proj]/venv/bin/First-App --run [insert-abs-path-to-proj]
```

4.4 Examples

Create a wheel package:

```
$ source venv/bin/activate  
$ python setup.py bdist_wheel  
$ deactivate
```

Deploy to /opt/first_app

```
$ mkdir -p /opt/first_app  
$ python3 -m venv /opt/first_app  
$ /opt/first_app/bin/pip install [path-to-first-app-wheel]  
$ /opt/first_app/bin/First-App -i /opt/first_app
```

Confirm that the application is working:

```
$ /opt/first_app/bin/First-App -r /opt/first_app
```

Create a systemd service unit file:

```
$ sudo /opt/first_app/bin/First-App-Service -u $USER -i /opt/first_app
```

Confirm that the unit-file looks correct:

```
$ cat /etc/systemd/system/first_app.service
```

```

[Unit]
Description=First-App
After=syslog.target network-online.target

[Service]
Type=simple
User=TODO-INSERT-MY-USERNAME
Group=TODO-INSERT-MY-USERNAME
Environment=PYTHONUNBUFFERED=true

WorkingDirectory=/opt/first_app
ExecStart=/opt/first_app/bin/First-App -r /opt/first_app

StandardOutput=syslog
StandardError=syslog

[Install]
WantedBy=multi-user.target

```

5 Webadmin

Webadmin is a simple web interface to configure and debug your application. You can customize basic behaviour in `default.conf`. It is recommended to add these options in its own file `webadmin.conf` and reference this file in `default.conf`

Here is a basic example:

Listing 1: default.conf

```

[config]
webadmin_conf = config/webadmin.conf

```

Listing 2: webadmin.conf

```

[webadmin]
host = 0.0.0.0
port = 8000
users.admin.user = admin
users.admin.password =
users.admin.password_hash = pbkdf2:sha256:150000$$N2b3ky8d$
↪$51fbf24e48d498bd5543d60a86bd94927fd4d6eb123bf2d81a7401666eaaa5c0
users.admin.roles = admin
secret_key = 1b50383ec6945aff8993f018feb568fa
on = 1
home_on = 1
config_on = 1
installationrepo_on = 1
tools_on = 1
settings_on = 1
sources_on = 1
expressions_on = 1
statistics_on = 1
security_webadmin_on = 1
security_certificates_on = 1
ssl_certificate =

```

(continues on next page)

```
ssl_certificate_key =
ssl_on = 0
```

Table 1: Webadmin

Section	Key	Default	Description
webadmin	Config	Default	Description
webadmin	host	0.0.0.0	Webserver host address
webadmin	port	8000	Webserver tcp port
webadmin	users.admin.user	admin	Username
webadmin	users.admin.password		Plain text password. If password_hash is set then this option is ignored.
webadmin	users.admin.password_hash		Password hash generated with <code>python -m netdef -ga</code> command
webadmin	users.admin.roles	admin	name of user role.
webadmin	secret_key		Secret flask session key. Can be generated with <code>python -m netdef -ga</code>
webadmin	on	1	Enable Webadmin. <ul style="list-style-type: none"> • 0 – disabled. • 1 – enabled.
webadmin	home_on	1	Enable <i>Webadmin</i> → <i>Home</i> .
webadmin	config_on	1	Enable <i>Webadmin</i> → <i>Config</i> .
webadmin	tools_on	1	Enable <i>Webadmin</i> → <i>Tools</i> .
webadmin	installationrepo_on		Enable <i>Webadmin</i> → <i>Tools</i> → <i>Upgrade</i> .
webadmin	security_webadmin_on		Enable <i>Webadmin</i> → <i>Tools</i> → <i>Webadmin</i> . <p>[config] <code>webadmin_conf=config/webadmin.conf</code></p> The default value is 1 if <code>webadmin_conf</code> exists in <code>[config]</code>
webadmin	security_certificates_on		Enable <i>Webadmin</i> → <i>Tools</i> → <i>Certificates</i> .
webadmin	settings_on	1	Enable <i>Webadmin</i> → <i>Settings</i> .
webadmin	sources_on	1	Enable <i>Webadmin</i> → <i>Sources</i> .
webadmin	expressions_on	1	Enable <i>Webadmin</i> → <i>Expressions</i> .
webadmin	statistics_on	1	Enable <i>Webadmin</i> → <i>Statistics</i> .
webadmin	ssl_certificate		File path to ssl certificate. Required if <code>ssl_on=1</code> .
webadmin	ssl_certificate_key		File path to ssl certificate key. Required if <code>ssl_on=1</code> .
webadmin	ssl_on	0	Enable https.
webadmin_views	[viewident]	0	[viewident] is the unique name of a <i>MyBaseView</i> <ul style="list-style-type: none"> • 0 – disabled. • 1 – enabled. <p style="text-align: center;">Listing 3: Example</p> <p>[webadmin_views] <code>Home = 1</code></p>
webadmin_views	Home	1	Enable Home view.
webadmin_views	FileModel	1	Enable FileModel view.
webadmin_views	SettingsModel	1	Enable SettingsModel view.
webadmin_views	SourcesModel	1	Enable SourcesModel view.
webadmin_views	ExpressionsView	1	Enable ExpressionsView view.
webadmin_views	StatisticsModel	1	Enable StatisticsModel view.

Continued on next page

Table 1 – continued from previous page

Section	Key	Default	Description
webadmin_views	Tools	1	Enable Tools view.

5.1 Override root endpoint

A common use case is to integrate an existing flask app into the root endpoint (/) of the webserver. The example shows how this is done by retrieving the webadmin WSGI app and register a new endpoint at '/'

first_app/main.py:

```
# function that register my custom flask app
def init_app(app):
    @app.route('/')
    def hello_world():
        return 'Hello, World!'
    return app

def main():
    ...

    engine = ThreadedWebGuiEngine.ThreadedWebGuiEngine(shared)

    # init my custom flask app as soon as the webgui engine is initialized.
    init_app(engine.get_flask_app())

    engine.add_controller_classes(controllers)
    engine.add_source_classes(sources)
    engine.add_rule_classes(rules)
    engine.load(__package__, 'netdef')
    engine.init()
    engine.start()
    engine.block() # until ctrl-c or SIG_TERM
    engine.stop()
    ...
```

5.2 Override Webadmin→Home

Copy the default html template.

netdef/Engines/templates/home.html:

```
{% extends 'home/home.html' %}

{% block home %}

    <p>Application version: {{version}}</p>

{% endblock home %}
```

Paste it into your application with extended information:

first_app/Engines/templates/home.html:

```

1  {% extends 'home/home.html' %}
2
3  {% block home %}
4
5      <p>{{app_name}} version: {{app_version}}</p>
6      <p>netdef version: {{netdef_version}}</p>
7      <p>Python version: {{py_version}}</p>
8      <p>Platform version: {{sys_version}}</p>
9
10 {% endblock home %}

```

Now you only have to override the Home View by creating following file:

first_app/Engines/webadmin/Home.py:

```

1  import sys
2  import datetime
3  import platform
4  from flask import current_app
5  from flask_admin import expose
6
7  from netdef.Engines.webadmin import Views, Home
8
9  from netdef import __version__ as netdef_version
10 from ... import __version__ as app_version
11 from ... import __package__ as app_name
12
13 @Views.register("Home")
14 def setup(admin):
15     Home.setup(admin, MyNewHome(name='Home', endpoint='home'))
16
17 class MyNewHome(Home.Home):
18     @expose("/")
19     def index(self):
20         return self.render(
21             'home.html',
22             app_name=app_name,
23             app_version=app_version,
24             netdef_version=netdef_version,
25             py_version=sys.version,
26             sys_version=str(platform.version())
27         )

```

- At line 13 we replace the default *Webadmin*→*Home* with your own
- At line 17 we override the default Home class with our extended functionality

5.3 Override *Webadmin*→*Tools*

Copy the default html template.

netdef/Engines/templates/tools.html:

```
{% extends 'tools/tools.html' %}
```

Paste it into your application with extended information:

first_app/Engines/templates/tools.html:


```

1  {% extends 'tools/tools.html' %}
2  {% block system_panel %}
3      <div class="panel panel-default">
4          <div class="panel-heading">System</div>
5          <div class="panel-body">
6              <p>Uptime: {{sys_uptime}}</p>
7              <div class="container">
8                  <div class="row">
9                      <a href="/cmd_dir/" class="btn btn-default col-md-2" role=
↪ "button">
10                         <span class="glyphicon glyphicon-list" aria-hidden="true">
↪ </span>
11                             dir
12                         </a>
13                     </div>
14                 </div>
15             </div>
16         </div>
17     {% endblock system_panel %}

```

Now you only have to override the Tools View by creating following file:

first_app/Engines/webadmin/Tools.py:

```

1  from flask import stream_with_context, Response
2  from flask_admin import expose
3  from netdef.Engines.webadmin import Views, Tools
4
5  @Views.register("Tools")
6  def setup(admin):
7      Tools.setup(admin, MoreTools(name='Tools', endpoint='tools'))
8
9  class MoreTools(Tools.Tools):
10     @expose("/cmd_dir/")
11     def hg_log(self):
12         return Response(
13             stream_with_context(
14                 Tools.stdout_from_terminal_as_generator(
15                     "dir",
16                     pre="Command:\n\n  hg log -r .:\n\nResult:\n\n",
17                     post=""
18                 )
19             )
20         )

```

- At line 5 we replace the default *Webadmin*→*Tools* with your own
- At line 9 we override the default Tools class with our extended functionality

6 Configuration

Config files is parsed at startup using the `configparser` module. Multiple strings and files is read in following order:

- (str) mypackage.defaultconfig:default_config_string
- (file) config/default.conf

- (file) config/default. [osname] .conf where **osname** is **nt** on windows and **posix** on linux.
- (files) all files found in [config] section in default.conf
- (file) config/default.conf.lock

6.1 Extended interpolation

Extended interpolation is using `${section:option}` to denote a value from a foreign section. Example:

Listing 4: default.conf

```
[OPCUAClientController]
endpoint = opc.tcp://${client:host}:${client:port}/freeopcua/server/
user = ${client:user}
password = ${client:password}

[OPCUAServerController]
endpoint = opc.tcp://${server:host}:${server:port}/freeopcua/server/
user = ${server:user}
password = ${server:password}

[client]
host = 10.10.1.13
port = 4841
user = CommonUser
password = 7T-SECRET_PASS-PhsTh7yVpV9jKTShAXcOdL8KmO4m3MUY3EPu7

[server]
host = 0.0.0.0
port = 4841
user = ${client:user}
password = ${client:password}
```

By using extended interpolation in combination with [config] section you can move application secrets into its own config file:

Listing 5: default.conf

```
[config]
secrets_conf = config/secrets.conf

[OPCUAClientController]
endpoint = opc.tcp://${client:host}:${client:port}/freeopcua/server/
user = ${client:user}
password = ${client:password}

[OPCUAServerController]
endpoint = opc.tcp://${server:host}:${server:port}/freeopcua/server/
user = ${server:user}
password = ${server:password}
```

Listing 6: secrets.conf

```
[client]
host = 10.10.1.13
port = 4841
```

(continues on next page)

(continued from previous page)

```
user = CommonUser
password = 7T-SECRET_PASS-PhsTh7yVpV9jKTShAXcOdL8KmO4m3MUY3EPu7

[server]
host = 0.0.0.0
port = 4841
user = ${client:user}
password = ${client:password}
```


6.2 Default configs

Table 2: General configs

Section	Key	Default	Description
general	identifier	[appidnt]	Name of application. <ul style="list-style-type: none"> • [appidnt] – is the unique name of your application. The name have to match in order for your application to accept the config file.
general	version	1	Version of your configfile. If you have to break compatibility in the future you can bump the config version to reject outdated config files
config	[unique key]	[filename]	Name of a configfile to be parsed. <ul style="list-style-type: none"> • [unique key] – is just a unique key • [filepath] – is the actual filename. File path relative to project folder. <p style="text-align: center;">Listing 7: Example</p> <pre>[config] my_conf = config/my_configuration.conf more_things = config/more_configs.conf</pre>
logging	loglevel	20	Default logging level for the application <ul style="list-style-type: none"> • 1 – All • 10 – Debug • 20 – Info • 30 – Warning • 40 – Error • 50 – Critical
logging	logformat	%(asctime)-15s %(levelname)-9s: %(name)-11s: %(message)s	Logging format for the application
logging	logdatefmt	%Y-%m-%d %H:%M:%S	Date time format
logging	to_console	1	<ul style="list-style-type: none"> • 0 – Suppress output to stdout • 1 – Write output to stdout
logging	to_file	0	<ul style="list-style-type: none"> • 0 – Disable logfile • 1 – Write output to logfile
logging	logfile	log/application.log	Path to logfile is relative to project folder.
logginglevels	[module name]	20	<ul style="list-style-type: none"> • [module name] is the name of a python module that is using the logging module <p>Values:</p> <ul style="list-style-type: none"> • 1 – All • 10 – Debug • 20 – Info • 30 – Warning • 40 – Error • 50 – Critical <p style="text-align: center;">Listing 8: Example</p> <pre>[logginglevels] werkzeug = 40 InternalController = 10</pre>

Table 3: Aliases

Section	Key	Default	Description
controller_aliases	[unique key]	[controllername]	Create multiple controller instances of same class Listing 12: Example [controllers] <code>CommTestController = 1</code> [controller_aliases] <code>FastPingController=CommTestController</code> <code>SlowPingController=CommTestController</code>
source_aliases	[unique key]	[sourcename]	Create multiple sources based on an existing source Listing 13: Example [sources] <code>IntegerSource = 1</code> [source_aliases] <code>IntStatusSource = IntegerSource</code> <code>IntCommandSource = IntegerSource</code>

Table 4: Thread pool configs

Section	Key	Default	Description
ExpressionExecutor	max_workers	[cpu_count * 10]	Number of thread pool workers to be available in <code>netdef.Engines.ThreadedEngine</code>

Table 5: Webadmin

Section	Key	Default	Description
webadmin	Config	Default	Description
webadmin	host	0.0.0.0	Webserver host address
webadmin	port	8000	Webserver tcp port
webadmin	users.admin.user	admin	Username
webadmin	users.admin.password		Plain text password. If password_hash is set then this option is ignored.
webadmin	users.admin.password_hash		Password hash generated with <code>python -m netdef -ga</code> command
webadmin	users.admin.roles	admin	name of user role.
webadmin	secret_key		Secret flask session key. Can be generated with <code>python -m netdef -ga</code>
webadmin	on	1	Enable Webadmin. • 0 – disabled. • 1 – enabled.
webadmin	home_on	1	Enable <i>Webadmin</i> → <i>Home</i> .
webadmin	config_on	1	Enable <i>Webadmin</i> → <i>Config</i> .
webadmin	tools_on	1	Enable <i>Webadmin</i> → <i>Tools</i> .
webadmin	installationrepo_on	oh	Enable <i>Webadmin</i> → <i>Tools</i> → <i>Upgrade</i> .

Continued on next page

Table 5 – continued from previous page

Section	Key	Default	Description
webadmin	security_webadmin_on	0	Enable <i>Webadmin</i> → <i>Tools</i> → <i>Webadmin</i> . [config] webadmin_conf=config/webadmin.conf The default value is 1 if <i>webadmin_conf</i> exists in [config]
webadmin	security_certificates_on	0	Enable <i>Webadmin</i> → <i>Tools</i> → <i>Certificates</i> .
webadmin	settings_on	1	Enable <i>Webadmin</i> → <i>Settings</i> .
webadmin	sources_on	1	Enable <i>Webadmin</i> → <i>Sources</i> .
webadmin	expressions_on	1	Enable <i>Webadmin</i> → <i>Expressions</i> .
webadmin	statistics_on	1	Enable <i>Webadmin</i> → <i>Statistics</i> .
webadmin	ssl_certificate		File path to ssl certificate. Required if <i>ssl_on</i> =1.
webadmin	ssl_certificate_key		File path to ssl certificate key. Required if <i>ssl_on</i> =1.
webadmin	ssl_on	0	Enable https.
webadmin_views	[viewident]	0	[viewident] is the unique name of a <i>MyBaseView</i> <ul style="list-style-type: none"> • 0 – disabled. • 1 – enabled. <p style="text-align: center;">Listing 14: Example</p> <pre>[webadmin_views] Home = 1</pre>
webadmin_views	Home	1	Enable Home view.
webadmin_views	FileModel	1	Enable FileModel view.
webadmin_views	SettingsModel	1	Enable SettingsModel view.
webadmin_views	SourcesModel	1	Enable SourcesModel view.
webadmin_views	ExpressionsView	1	Enable ExpressionsView view.
webadmin_views	StatisticsModel	1	Enable StatisticsModel view.
webadmin_views	Tools	1	Enable Tools view.

Table 6: Upgrade application

Section	Key	Default	Description
auto_update	on	0	
auto_update	no_index	0	
auto_update	pre_release	0	
auto_update	force_reinstall	0	
auto_update	find_links		
auto_update	trusted_host		
auto_update	minimal_timeout	0	
auto_update	package	[appid]	

6.3 Built-in Controllers and Rules

You can look opp the correct *Built-in configs* in API Reference

7 Advanced

7.1 Project layout

create a Project folder:

```
$ mkdir First-App
$ cd First-App

$ mkdir config
$ mkdir log
$ mkdir first_app
```

- First-App, The Project name.
- config, applications default configfiles
- log, application.log is created in this folder
- first_app, the python package with your applications files

```
/First-App
  /setup.py
  /config
    /default.conf
  /first_app
    /Controllers
    /Engines
      /__init__.py
      /templates
      /webadmin
    /Interfaces
    /Rules
    /Sources
    /__init__.py
    /__main__.py
    /defaultconfig.py
    /main.py
```

setup.py:

- package_data, make sure to include html templates
- entry_points, the entry point will make it easy to launch application

```
from setuptools import setup, find_packages
from first_app import __version__ as app_version

NAME = "First-App"
MAIN_PACKAGE = "first_app"

setup(
    name=NAME,
    version=app_version,
    packages=find_packages(exclude=['contrib', 'docs', 'tests', 'config']),
    install_requires=[
        'netdef'
    ],
```

(continues on next page)

(continued from previous page)

```
package_data={
    MAIN_PACKAGE: [
        'Engines/templates/*.html',
        'Engines/templates/**/*.html'
    ]
},
entry_points={
    'console_scripts': [
        '{NAME}={MAIN_PACKAGE}.__main__:cli'.format(NAME=NAME, MAIN_PACKAGE=MAIN_
↵PACKAGE),
    ],
},
)
```

first_app/__init__.py:

```
__version__ = '0.1.0'
```

first_app/__main__.py:

```
from netdef.__main__ import entrypoint

def run_app():
    from . import main

def get_template_config():
    from . import defaultconfig
    return defaultconfig.template_config_string

def cli():
    # entrypoint: console_scripts
    entrypoint(run_app, get_template_config)

if __name__ == '__main__':
    # entrypoint: python -m console_scripts
    entrypoint(run_app, get_template_config)
```

first_app/defaultconfig.py:

```
template_config_string = \
"""[general]
identifier = First-App
version = 1
"""

default_config_string = \
"""[general]
[config]
[ExpressionExecutor]
[webadmin]
host = 0.0.0.0
port = 8000
user = admin
password = admin

[webadmin_views]
```

(continues on next page)

(continued from previous page)

```
[logging]
logglevel = 20
loggformat = %(asctime)-15s %(levelname)-9s: %(name)-11s: %(message)s
loggdatefmt = %Y-%m-%d %H:%M:%S
to_console = 1
to_file = 1

[logginglevels]
werkzeug = 40

[rules]

[controllers]

[controller_aliases]

[sources]

[source_aliases]
"""
```

first_app/main.py:

```
import os
from netdef.Controllers import Controllers
from netdef.Sources import Sources
from netdef.Rules import Rules
from netdef.Engines import ThreadedWebGuiEngine
from netdef.Shared import Shared
from netdef.utils import setup_logging, handle_restart
from . import defaultconfig

def main():
    # init shared-module
    try:
        install_path = os.path.dirname(__file__)
        proj_path = os.getcwd()
        config_string = defaultconfig.default_config_string
        shared = Shared.Shared("First-App", install_path, proj_path, config_string)
    except ValueError as error:
        print(error)
        raise SystemExit(1)

    # configure logging
    setup_logging(shared.config)

    controllers = Controllers.Controllers(shared)
    controllers.load(__package__, 'netdef')

    sources = Sources.Sources(shared)
    sources.load(__package__, 'netdef')

    rules = Rules.Rules(shared)
    rules.load(__package__, 'netdef')

    # the engine connects webadmin, controllers, sources and rules.
    engine = ThreadedWebGuiEngine.ThreadedWebGuiEngine(shared)
```

(continues on next page)

(continued from previous page)

```
engine.add_controller_classes(controllers)
engine.add_source_classes(sources)
engine.add_rule_classes(rules)
engine.load(['__package__', 'netdef'])

engine.init()
engine.start()
engine.block() # until ctrl-c or SIG_TERM
engine.stop()

# if restart-button in webadmin is pressed:
handle_restart(shared, engine)

main()
```

config/default.conf:

```
[general]
identifier = First-App
version = 1
```

7.2 Add a controller

Built-in controllers can be activated by adding special values to the config file.

You can look opp the correct *Built-in configs* in API Reference

In this tutorial we will activate the CrontabController and the OPCUAServerController

We will have to merge the two configs into one and add them to config/default.conf

```
[controllers]
CrontabController = 1
OPCUAServerController = 1

[sources]
CrontabSource = 1
VariantSource = 1

[CrontabSource]
controller = CrontabController

[VariantSource]
controller = OPCUAServerController

[CrontabController]
[OPCUAServerController]
```

We also have to merge required packages into requirements.txt:

```
crontab
freeopcua
```

Next step is to start using the controllers and sources by setting up a Rule.

7.3 Add a rule

Built-in rules can be activated by adding special values to the config file, just like the controllers. There is currently only one built-in rule we can use.

Add the config for CSVRule to `config/default.conf` and replace the example rules with a `hello_world` rule like this:

```
[rules]
CSVRule = 1

[CSVRule]
hello_world_rule = 1

[hello_world_rule]
csv = config/hello_world_rule.csv
py = config/hello_world_rule.py
```

We now have to create the csv and py file:

`config/hello_world_rule.csv`

```
CrontabSource,VariantSource
*/2 * * * * *,ns=2;s=hello_world
```

`config/hello_world_rule.py`

```
def expression(cron, oua):
    if cron.new:
        oua.set = "Hello, world"

    if cron.update:
        oua.set = "Hello, world {}".format(int(cron.value))
```

Now you can try to launch the application:

```
$ pip install -r requirements.txt
$ python -m first_app -r .
```

7.4 Create a custom controller

Copy the included template to create a custom controller.

`netdef/Controllers/NewControllerTemplate.py`:

```
import datetime
import logging

from netdef.Controllers import BaseController, Controllers
from netdef.Sources.BaseSource import StatusCode

# import my supported sources
from netdef.Sources.NewSourceTemplate import NewSourceTemplate

@Controllers.register("NewControllerTemplate")
class NewControllerTemplate(BaseController.BaseController):
```

(continues on next page)

(continued from previous page)

```
def __init__(self, name, shared):
    super().__init__(name, shared)
    self.logger = logging.getLogger(self.name)
    self.logger.info("init")
    self.one_config_entry = self.shared.config.config(
        self.name, "one_config_entry", "default_value"
    )

def run(self):
    "Main loop. Will exit when receiving interrupt signal"
    self.logger.info("Running")
    while not self.has_interrupt():
        self.loop_incoming() # dispatch handle_* functions
        self.loop_outgoing() # dispatch poll_* functions
    self.logger.info("Stopped")

def handle_readall(self, incoming):
    raise NotImplementedError

def handle_add_source(self, incoming):
    self.logger.debug("'Add source' event for %s", incoming.key)
    self.add_source(incoming.key, incoming)

def handle_read_source(self, incoming):
    raise NotImplementedError

def handle_write_source(self, incoming, value, source_time):
    self.logger.debug(
        "'Write source' event to %s. value: %s at: %s",
        incoming.key,
        value,
        source_time,
    )

def poll_outgoing_item(self, item):
    if isinstance(item, NewSourceTemplate): # My
        # TODO: get new value somehow
        address = item.unpack_address()
        new_val = get_the_new_value_somehow(address)
        stime = datetime.datetime.utcnow()
        status_ok = True # Why not
        cmp_oldew = False # compare old and new value?

        if self.update_source_instance_value(
            item, new_val, stime, status_ok, cmp_oldew
        ):
            self.send_outgoing(item)
```

Paste it into your application with a new name:

first_app/Controllers/CmdController.py:

```
1 import logging
2 import datetime
3 from netdef.Controllers import BaseController, Controllers
4 from netdef.Sources.BaseSource import StatusCode
5
```

(continues on next page)

(continued from previous page)

```
6 # import my supported sources
7 from netdef.Sources.NewSourceTemplate import NewSourceTemplate
8
9 @Controllers.register("CmdController")
10 class CmdController(BaseController.BaseController):
11     def __init__(self, name, shared):
12         super().__init__(name, shared)
13
14 ...
```

Line 9 and 10 is changed to the same name as the file. Line 7 have to be replaced at a later time to a custom or built-in source

To activate the controller we have to merge following config to default.conf:

```
[controllers]
CmdController = 1

[CmdController]
```

Result after merge:

```
[controllers]
CrontabController = 1
OPCUAServerController = 1
CmdController = 1

[CrontabController]

[OPCUAServerController]

[CmdController]
```

7.5 Create a custom source

Copy the included template to create a custom source for your controller.

netdef/Sources/NewSourceTemplate.py:

```
from netdef.Interfaces.DefaultInterface import DefaultInterface
from netdef.Sources import BaseSource, Sources

@Sources.register("NewSourceTemplate")
class NewSourceTemplate(BaseSource.BaseSource):
    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.interface = DefaultInterface

    # TODO: add a address for your new controller
    def unpack_address(self):
        return self.key
```

Paste it into your application with a new name:

first_app/Sources/CmdSource.py:

```

1 from netdef.Sources import BaseSource, Sources
2 from netdef.Interfaces.DefaultInterface import DefaultInterface
3
4 @Sources.register("CmdSource")
5 class CmdSource(BaseSource.BaseSource):
6     def __init__(self, *args, **kwargs):
7         super().__init__(*args, **kwargs)
8         self.interface = DefaultInterface
9
10     # TODO: add a address for your new controller
11     def unpack_address(self):
12         return self.key

```

Line 4 and 5 is changed to the same name as the file.

Change line 7 in your custom controller:

first_app/Controllers/CmdController.py:

```

1 import logging
2 import datetime
3 from netdef.Controllers import BaseController, Controllers
4 from netdef.Sources.BaseSource import StatusCode
5
6 # import my new source
7 from ..Sources.CmdSource import CmdSource
8 ...

```

To activate the source we have to merge following config to default.conf:

```

[sources]
CmdSource = 1

[CmdSource]
controller = CmdController

```

Result:

```

[controllers]
CrontabController = 1
OPCUAServerController = 1
CmdController = 1

[sources]
CrontabSource = 1
VariantSource = 1
CmdSource = 1

[CrontabSource]
controller = CrontabController

[VariantSource]
controller = OPCUAServerController

[CmdSource]
controller = CmdController

[CrontabController]

```

(continues on next page)

```
[OPCUAServerController]

[CmdController]
```

7.6 Create a custom rule

Copy the included template to create a custom rule.

netdef/Rules/NewRuleTemplate.py:

```
import logging
import pathlib

from netdef.Rules import BaseRule, Rules
from netdef.Rules.utils import import_file

SourceInfo = BaseRule.SourceInfo
ExpressionInfo = BaseRule.ExpressionInfo

@Rules.register("NewTemplateRule")
class NewTemplateRule(BaseRule.BaseRule):
    def __init__(self, name, shared):
        super().__init__(name, shared)
        self.logger = logging.getLogger(name)
        self.logger.info("init")

        config = self.shared.config.config
        self.proj_path = pathlib.Path(config("proj", "path", ".")).absolute()

    def setup(self):
        self.logger.info("Running setup")

        # example:
        self.setup_example()

        # sub rule example:
        for name, active in self.shared.config.get_dict(self.name).items():
            if int(active):
                self.setup_sub_rule(name)
        self.logger.info("Done parsing")

    def setup_sub_rule(self, name):
        raise NotImplementedError

    def setup_example(self):
        # example_expression_module = self.import_py_file("config/example_expression.
↪py")

        # config/example_expression.py:
        # def expression(internal):
        #     if internal.new or internal.update:
        #         print(internal)
```

(continues on next page)

(continued from previous page)

```
self.add_new_parser("InternalSource")

source_count = self.add_new_expression(
    ExpressionInfo(
        example_expression_module,
        [SourceInfo("InternalSource", "intern_test_1")],
    )
)
self.update_statistics(self.name + ".example", 0, 1, source_count)

def import_py_file(self, rel_file):
    full_file = pathlib.Path(self.proj_path).joinpath(rel_file)
    nice_name = full_file.name
    return import_file(str(full_file), self.name, nice_name)

def run(self):
    self.logger.info("Running")
    while not self.has_interrupt():
        self.loop_incoming() # dispatch handle_* functions
    self.logger.info("Stopped")

def handle_run_expression(self, incoming, value, source_time, status_code):
    expressions = self.get_expressions(incoming)
    self.logger.debug(
        "Received %s. Found expressions %s", incoming.key, len(expressions)
    )
    if expressions:
        self.send_expressions_to_engine(
            incoming, expressions, value, source_time, status_code
        )
    )
```

Paste it into your application with a new name:

first_app/Rules/FirstAppRule.py:

```
1 import logging
2 import pathlib
3 from .utils import import_file
4 from . import BaseRule, Rules
5
6 SourceInfo = BaseRule.SourceInfo
7 ExpressionInfo = BaseRule.ExpressionInfo
8
9 @Rules.register("FirstAppRule")
10 class FirstAppRule(BaseRule.BaseRule):
11     def __init__(self, name, shared):
12         super().__init__(name, shared)
13         self.logger = logging.getLogger(name)
14         self.logger.info("init")
```

Line 9 and 10 is changed to the same name as the file.

To activate the rule we have to merge following config to default.conf:

```
[rules]
FirstAppRule = 1

[FirstAppRule]
```

Result:

```
[rules]
FirstAppRule = 1

[FirstAppRule]

[controllers]
CrontabController = 1
OPCUAServerController = 1
CmdController = 1

[sources]
CrontabSource = 1
VariantSource = 1
CmdSource = 1

[CrontabSource]
controller = CrontabController

[VariantSource]
controller = OPCUAServerController

[CmdSource]
controller = CmdController

[CrontabController]

[OPCUAServerController]

[CmdController]
```

7.7 Putting it all together

In this example we want to pass following commands to the subprocess module:

- echo hello
- ls -lah .
- ./simple_script.sh
- echo Don't break the

We could hard code these commands in the controller but it is more flexible to create a source for each command. And we also want to read these commands from a config file so it will be easy to reuse, change or extend the commands.

To achieve this we just implement a method in the source that returns the command. the command can be extracted from the sources key:

first_app/Sources/CmdSource.py:

```
1 from netdef.Sources import BaseSource, Sources
2 from netdef.Interfaces.DefaultInterface import DefaultInterface
3
4 @Sources.register("CmdSource")
5 class CmdSource(BaseSource.BaseSource):
6     def __init__(self, *args, **kwargs):
7         super().__init__(*args, **kwargs)
```

(continues on next page)

(continued from previous page)

```
8     self.interface = DefaultInterface
9
10    def get_command_and_args(self, args=None):
11        if args:
12            return self.key + " " + args
13        else:
14            return self.key
```

The controller can retrieve the command to run by calling `get_command_and_args`
first_app/Controllers/CmdController.py:

```
1  import logging
2  import datetime
3  import subprocess
4  import shlex
5
6  from netdef.Controllers import BaseController, Controllers
7  from netdef.Sources.BaseSource import StatusCode
8
9  from ..Sources.CmdSource import CmdSource
```

We will use `subprocess` and `shlex` from standard library to execute commands. To keep it simple we can create a wrapper function that run a command and return the result from stdout. In case of error the function return the error as text instead. Also, charset decoding errors is replaced with "?".

```
10 def stdout_from_terminal(command_as_str, err_msg=None):
11     command_args = shlex.split(command_as_str)
12     try:
13         res = subprocess.run(command_args, stdout=subprocess.PIPE).stdout
14         return str(res, errors="replace")
15     except Exception as error:
16         if err_msg is None:
17             return str(error)
18         else:
19             return err_msg
```

We create an option `value_as_args` to use the value from the source to be added as an argument to the command. the option is read from config file.

```
20 @Controllers.register("CmdController")
21 class CmdController(BaseController.BaseController):
22     def __init__(self, name, shared):
23         super().__init__(name, shared)
24         self.logger = logging.getLogger(self.name)
25         self.logger.info("init")
26         self.value_as_args = self.shared.config.config(self.name, "value_as_args", 1)
```

The run method will be very simple in this tutorial. Normally this is where we create a polling loop or setup subscriptions and await events. In this example we only wait for `WRITE_SOURCE` messages. So we only have to iterate the message queue:

```
27 def run(self):
28     "Main loop. Will exit when receiving interrupt signal"
29     self.logger.info("Running")
30     while not self.has_interrupt():
```

(continues on next page)

(continued from previous page)

```
31     self.loop_incoming() # dispatch handle_* functions
32     self.logger.info("Stopped")
```

The rule will always send the source instance at startup as a `ADD_SOURCE` message. we have to receive the message and keep it in our controller. We can use `netdef.Controllers.BaseController.BaseController.add_source`

```
33 def handle_add_source(self, incoming):
34     self.logger.debug("'Add source' event for %s", incoming.key)
35     self.add_source(incoming.key, incoming)
```

When an expression changes the value on one of our sources we will receive a `WRITE_SOURCE` message. We have to verify that the received source is in our source list and that we know how to handle it.

To check if it is one of ours we use `netdef.Controllers.BaseController.BaseController.has_source`

To check if we know how to handle it we check if it is an instance of the source we created `CmdSource`.

```
36 def handle_write_source(self, incoming, value, source_time):
37     self.logger.debug("'Write source' event to %s. value: %s at: %s", incoming.key,
↪value, source_time)
38     if not self.has_source(incoming.key):
39         self.logger.error(
40             "%s not found",
41             incoming.key
42         )
43         return
44
45     if not isinstance(incoming, CmdSource):
46         self.logger.error(
47             "Got write event for %s, but only CmdSource is supported",
48             type(incoming)
49         )
50         return
```

We have verified that the source is an instance of `CmdSource`. Knowing this we can safely call `CmdSource.get_command_and_args` to get the command.

```
51     if self.value_as_args:
52         cmd_as_str = incoming.get_command_and_args(value)
53     else:
54         cmd_as_str = incoming.get_command_and_args()
55
56     new_val = stdout_from_terminal(cmd_as_str)
57     stime = datetime.datetime.utcnow()
58     status_ok = True # Why not
59     cmp_oldew = False # compare old and new value?
```

At last we create and send a `RUN_EXPRESSION` message using `netdef.Controllers.BaseController.BaseController.update_source_instance_value` and `netdef.Controllers.BaseController.BaseController.send_outgoing`

```
60     if self.update_source_instance_value(incoming, new_val, stime, status_ok, cmp_
↪oldew):
61         self.send_outgoing(incoming)
```

We now have to create the configfile and expression that is parsed by rule. The command list can be a simple text file:

config/command_rule.txt:

```
1 echo hello
2 ls -lah .
3 ./simple_script.sh
4 echo Don\'t break the
```

The expression is a python file. The rule expect to find a function called `expression()`

config/command_rule.py:

```
1 import logging
2 logger = logging.getLogger(__name__ + ":expression")
3
4 def expression(intern, cmd):
5     # triggers at startup
6     if intern.new:
7
8         if "hello" in cmd.key:
9             arg = "world"
10        elif "Don\\\'t break the" in cmd.key:
11            arg = "circle"
12        else:
13            arg = ""
14
15        logger.info("{}: Send command arg: {}".format(cmd.key, arg))
16        cmd.set = arg
17
18    if cmd.new or cmd.update:
19        logger.info("{}: Result: {}".format(cmd.key, cmd.value))
```

Now we are ready to create the rule

first_app/Rules/FirstAppRule.py:

```
1 import logging
2 import pathlib
3 from netdef.Rules.utils import import_file
4 from netdef.Rules import BaseRule, Rules
5
6 SourceInfo = BaseRule.SourceInfo
7 ExpressionInfo = BaseRule.ExpressionInfo
```

We will look for the config file and expression file relative to the project folder.

```
8 @Rules.register("FirstAppRule")
9 class FirstAppRule(BaseRule.BaseRule):
10     def __init__(self, name, shared):
11         super().__init__(name, shared)
12         self.logger = logging.getLogger(name)
13         self.logger.info("init")
14         self.proj_path = shared.config.config("proj", "path")
15
16     def read_list(self, rel_file):
17         full_file = pathlib.Path(self.proj_path).joinpath(rel_file)
18         lines = open(str(full_file), "r").readlines()
19         return [l.strip() for l in lines]
```

(continues on next page)

(continued from previous page)

```
20
21 def import_py_file(self, rel_file):
22     full_file = pathlib.Path(self.proj_path).joinpath(rel_file)
23     nice_name = full_file.name
24     return import_file(str(full_file), self.name, nice_name)
```

TODO

```
25 def setup(self):
26     self.logger.info("Running setup")
27     self.setup_commands()
28     self.logger.info("Done parsing")
29
30 def setup_commands(self):
31     command_expression_module = self.import_py_file("config/command_rule.py")
32     command_list = self.read_list("config/command_rule.txt")
33
34     source_count = 0
35     for command in command_list:
36         source_count += self.add_new_expression(
37             ExpressionInfo(
38                 command_expression_module,
39                 [
40                     SourceInfo("InternalSource", "generic"),
41                     SourceInfo("CmdSource", command)
42                 ]
43             )
44         )
45     self.update_statistics(self.name + ".commands", 0, 1, source_count)
```

TODO

```
46 def run(self):
47     self.logger.info("Running")
48     while not self.has_interrupt():
49         self.loop_incoming() # dispatch handle_* functions
50     self.logger.info("Stopped")
```

TODO

```
51 def handle_run_expression(self, incoming, value, source_time, status_code):
52     expressions = self.get_expressions(incoming)
53     self.logger.debug("Received %s. Found expressions %s", incoming.key,
↪ len(expressions))
54     if expressions:
55         self.send_expressions_to_engine(incoming, expressions, value, source_time,
↪ status_code)
```

TODO

config/default.ini

```
1 [rules]
2 FirstAppRule = 1
3
4 [FirstAppRule]
5
```

(continues on next page)

(continued from previous page)

```
6 [sources]
7 CmdSource = 1
8 InternalSource = 1
9
10 [CmdSource]
11 controller = CmdController
12
13 [InternalSource]
14 controller = InternalController
15
16 [controllers]
17 CmdController = 1
18 InternalController = 1
19
20 [InternalController]
21 send_init_event = 1
22
23 [CmdController]
24 value_as_args = 1
```

TODO

tests/test_command_rule.py

```
1 from netdef.testutils import MockExpression
2 from netdef.Sources.InternalSource import InternalSource
3 from first_app.Sources.CmdSource import CmdSource
4
5 def test_hello():
6     mock = MockExpression(
7         module="config/command_rule.py",
8         intern=InternalSource("generic"),
9         cmd=CmdSource("echo hello")
10    )
11    mock.intern.update_value(None, stat_init=True)
12    mock.cmd.assert_called_once_with("world")
13    mock.intern.assert_not_called()
14
15
16 def test_circle():
17     mock = MockExpression(
18         module="config/command_rule.py",
19         intern=InternalSource("generic"),
20         cmd=CmdSource("echo Don\\'t break the")
21    )
22    mock.intern.update_value(None, stat_init=True)
23    mock.cmd.assert_called_once_with("circle")
24    mock.intern.assert_not_called()
25
26
27 def test_ls():
28     mock = MockExpression(
29         module="config/command_rule.py",
30         intern=InternalSource("generic"),
31         cmd=CmdSource("ls -lah .")
32    )
33    mock.intern.update_value(None, stat_init=True)
```

(continues on next page)

```
34 mock.cmd.assert_called_once_with("")
35 mock.intern.assert_not_called()
```

TODO

8 Credits

8.1 Contributors

- Frode Holmer <fholmer+netdef@gmail.com>
- Vegard Haugland <vegard@haugland.at>

9 Changelog

9.1 1.0.7

2021-07-20

Enhancements

- SecurityCertificatesView: Added OpcUa certs
- Controllers: call setup-function in module if found at startup
- Rules: call setup-function in module if found at startup
- Sources: call setup-function in module if found at startup
- BaseSource: call setup-function in source instance if found at startup
- Added testutils.MockShared
- OPCUAServerController: insert SourceTimestamp and ServerTimestamp if missing
- OPCUAServerController: Added option for debug_statistics
- Added: SubprocessController
- Webadmin: Added a simple web interface for tracemalloc and gc

Bug fixes

- Values from controller is frozen before RUN_EXPRESSION is sent.
- CommTestSource: remove url_path from host
- Rules.utils: relative import of filenames did not work properly

Incompatible API changes

- handle_run_expression function in BaseRule now have 4 required arguments. All custom rules have to be updated.

9.2 1.0.6

2020-02-28

Enhancements

- Webadmin: Added role based user table
- Added testutils.MockExpression to simplify testing of expressions
- Added: InfluxDBLogger
- Improved ModbusClientController
- BaseController: Improved message queue helper function
- InternalController: Improved persistent storage
- OPCUAServerController: added config for auto_build_folders
- ModbusServerController: new option: daemon_threads

Bug fixes

- ModbusServerController: Attempt to bind to socket for three minutes before throwing exception.

Incompatible API changes

- MQTTDataMessageController: renamed from MQTTDataAccessController
- Webadmin: [webadmin]user/password keyword has changed.
- InternalController: changed persistent storage filenames

9.3 1.0.5

2019-11-07

Enhancements

- OPCUAClientController: Improved configuration
- OPCUAServerController: Added legacy support for basic128rsa15 and basic256.
- Webadmin: Added SecurityWebadminView and SecurityCertificatesView
- ModbusServerController: Attempt to bind to socket for one minute before throwing exception. (Handle CLOSE_WAIT state)

Bug fixes

- Webadmin: Changed height of file edit textarea to 20 rows
- Webadmin: Fixed routing.BuildError when you don't have the permission to access the requested resource.
- CSVRule: expression can now be a modulename or a python-file
- Fixed Windows service.
- OPCUAServerController: Fixed TypeError
- OPCUAServerController: Only add subscription if exists

Incompatible API changes

- InternalController: changed persistent storage filenames

9.4 1.0.4

2019-08-19

Enhancements

- ModbusServerController: get modbus framer by calling self.get_framer
- Added FloatInterface and StringInterface
- Display a 10 second restart timer in webadmin on restart
- InternalController: config entry send_init_event trigger event at startup
- Added an experimental yaml parser
- Added an experimental ini parser
- Source value can be changed from webadmin -> Sources -> Edit
- Added create_interface function to expression arguments
- Added persistent storage to InternalController
- Added new message type APP_STATE
- Added Alpha version of ConcurrentWebRequestController
- Added simple installer for Systemd services

Bug fixes

- OPCUAServerController: Fixed a varianttype bug
- Fixed pyinstaller hook file
- BaseRule is rewritten to store expression info in shared module. This fixes a problem with multiple rules sharing same sources.
- Fixed a problem where the name of a controller or rule and module name had to be equal.
- OPCUAClientController: specify security mode in configfile
- OPCUAServerController: reject X509IdentityToken
- OPCUAServerController: force timestamp on values (from clients) where timestamp is none

Incompatible API changes

- OPCUAServerController: startup statuscode changed from BadNoData to BadWaitingForInitialData
- BaseRule: rule_name_from_key no longer accept * as a rule name
- BaseController: fetch_one_incoming returns tuple

9.5 1.0.3

2019-06-16

Enhancements

- SystemMonitorController: monitor disk partition usage
- Display update options in webadmin -> Tools -> Upgrade
- BaseRule: call setup-function in expressions if found at startup
- Added docs

- OPCUAServerController: OPCUA controller will set statuscode BadNoData on startup.
- Added BaseAsyncController
- Webadmin: / redirects to admin page. /admin redirects to /admin/home.
- Allow for existing flask apps to be integrated in Webadmin

Bug fixes

- Added requirements and missing interface
- Added extendable blocks in html templates
- Tools.setup got a view argument

Incompatible API changes

- Expression: interface attribute have been removed from expressions arguments

9.6 1.0.2

2019-05-25

Enhancements

- Added support for Windows services. require pywin32 package on windows

9.7 1.0.1

2019-05-17

Enhancements

- Added CrontabController
- Added MQTTDataAccessController
- Added RESTJsonController
- Added SystemMonitorController
- Added simple user/pass to OPCUAServerController

Bug fixes

- Fixed netdef endpoint

9.8 1.0.0

2019-04-30

- First public release

10 Build from source

10.1 Python

Normally you don't have to compile python. On Windows you can download pre-compiled binaries, and most linux distros have a pre-installed version of python.

Compiling to a relative directory:

```
$ mkdir ~/Python-3.8/
$ cd ~/Python-3.8/
$ wget https://www.python.org/ftp/python/3.8.1/Python-3.8.1.tgz
$ tar zxvf Python-3.8.1.tgz
$ Python-3.8.1/configure
$ make
$ make install DESTDIR=.
```

Or absolute directory:

```
$ mkdir /opt/Python-3.8/
$ cd /opt/Python-3.8/
$ wget https://www.python.org/ftp/python/3.8.1/Python-3.8.1.tgz
$ tar zxvf Python-3.8.1.tgz
$ Python-3.8.1/configure --prefix=/opt/Python-3.8
$ make
$ make install
```

10.2 psutil

Normally you don't have to compile yourself. `pip install` should compile automatically. If automatic compilation fails you can try to specify include dirs and library dirs:

```
$ pip install --global-option=build_ext \
  --global-option="-I~/Python-3.8/usr/local/include/python3.8" \
  --global-option="-L~/Python-3.8/usr/local/lib" \
  psutil
```

10.3 Netdef package

Debian

Install requirements:

```
# python 3.5 +
$ sudo apt-get install python3 python3-pip python3-venv

# source control management
$ sudo apt-get install mercurial

# requirements for building psutil
$ sudo apt-get install build-essential python3-dev
```

Get sources:

```
$ hg clone git+ssh://git@gitlab.com:fholmer/netdef.git
$ cd netdef
```

Setup virtual environment:

```
$ python3 -m venv venv
$ source venv/bin/activate
```

Build sdist and wheel:

```
$ python setup.py sdist
$ python setup.py bdist_wheel
```

Windows

Install requirements:

Get [Python](#) and [Mercurial](#)

Get sources:

```
> hg clone git+ssh://git@gitlab.com:fholmer/netdef.git
> cd netdef
```

Setup an virtual environment:

```
> py -3 -m venv venv
> venv\Scripts\activate
```

Build sdist and wheel

```
> python setup.py sdist
> python setup.py bdist_wheel
```

10.4 Docs

Debian

Install requirements

```
# requirements for building psutil
$ sudo apt-get install build-essential python3-dev

# requirements for pdf
$ sudo apt-get install texlive-latex-recommended texlive-latex-extra texlive-fonts-
↪recommended latexmk

# requirements for pdf multi language
$ sudo apt-get install texlive-lang-european texlive-lang-english

# requirements for UML diagram
$ sudo apt-get install plantuml
```

Setup virtual environment:

```
$ python3 -m venv venv
$ source venv/bin/activate
```

Build docs:

```
$ cd docs
$ make html
$ make latexpdf
```

UML diagrams:

Note: This is only needed if UML diagrams is out of date:

```
$ plantuml -tsvg docs/_static/uml/
```

11 Built-in configs

- *Controller configs*
 - *CommTestController*
 - *ConcurrentWebRequestController*
 - *CrontabController*
 - *InfluxDBLoggerController*
 - *InternalController*
 - *ModbusServerController*
 - *MQTTDataMessageController*
 - *NewControllerTemplate*
 - *OPCUAServerController*
 - *SubprocessController*
 - *SystemMonitorController*
 - *XmlRpcController*
 - *ZmqDataAccessController*
- *Rule configs*
 - *CSVRule*
 - *InfluxDBLoggerRule*
 - *INIRule*
 - *YAMLRule*

11.1 Controller configs

CommTestController

Listing 15: config/default.conf

```
[controllers]  
CommTestController = 1  
  
[sources]
```

(continues on next page)

(continued from previous page)

```
CommTestSource = 1

[CommTestSource]
controller = CommTestController

[CommTestController]
```

ConcurrentWebRequestController

Listing 16: config/default.conf

```
[controllers]
ConcurrentWebRequestController = 1

[sources]
ConcurrentWebRequestSource = 1

[ConcurrentWebRequestSource]
controller = ConcurrentWebRequestController

[ConcurrentWebRequestController]
```

Listing 17: requirements.txt

```
aihttp
```

CrontabController

Listing 18: config/default.conf

```
[controllers]
CrontabController = 1

[sources]
CrontabSource = 1

[CrontabSource]
controller = CrontabController

[CrontabController]
```

Listing 19: requirements.txt

```
crontab
```

InfluxDBLoggerController

Listing 20: config/default.conf

```
[rules]
InfluxDBLoggerRule = 1

[controllers]
InfluxDBLoggerController = 1

[sources]
InfluxDBLoggerSource = 1

[InfluxDBLoggerSource]
controller = InfluxDBLoggerController

[InfluxDBLoggerRule]
auto_logging_on = 1

[InfluxDBLoggerController]
dsn = influxdb:///Database-Name
```

Listing 21: requirements.txt

```
influxdb
```

InternalController

Listing 22: config/default.conf

```
[controllers]
InternalController = 1

[sources]
InternalSource = 1

[InternalSource]
controller = InternalController

[InternalController]
```

ModbusServerController

Listing 23: config/default.conf

```
[controllers]
ModbusServerController = 1

[sources]
HoldingRegisterSource = 1

[HoldingRegisterSource]
controller = ModbusServerController

[ModbusServerController]
```

(continues on next page)

(continued from previous page)

```
[ModbusServerController_devices]
ModbusServerController_device0 = 1

[ModbusServerController_device0]
```

Listing 24: requirements.txt

```
pymodbus
```

MQTTDataMessageController

Listing 25: config/default.conf

```
[controllers]
MQTTDataMessageController = 1

[sources]
MQTTDataMessageSource = 1

[MQTTDataMessageSource]
controller = MQTTDataMessageController

[MQTTDataMessageController]
```

Listing 26: requirements.txt

```
paho-mqtt
```

NewControllerTemplate

Listing 27: config/default.conf

```
[controllers]
NewControllerTemplate = 1

[sources]
NewSourceTemplate = 1

[NewSourceTemplate]
controller = NewControllerTemplate

[NewControllerTemplate]
```

OPCUAServerController

Listing 28: config/default.conf

```
[controllers]
OPCUAServerController = 1
```

(continues on next page)

(continued from previous page)

```
[sources]
VariantSource = 1
BytestringSource = 1

[VariantSource]
controller = OPCUAServerController

[BytestringSource]
controller = OPCUAServerController

[OPCUAServerController]
```

Listing 29: requirements.txt

```
freeopcua
```

SubprocessController

Listing 30: config/default.conf

```
[controllers]
SubprocessController = 1

[sources]
SubprocessSource = 1

[SubprocessSource]
controller = SubprocessController

[SubprocessController]
```

SystemMonitorController

Listing 31: config/default.conf

```
[controllers]
SystemMonitorController = 1

[sources]
SystemMonitorSource = 1

[SystemMonitorSource]
controller = SystemMonitorController

[SystemMonitorByteSource]
controller = SystemMonitorController

[SystemMonitorPercentSource]
controller = SystemMonitorController
```

Listing 32: requirements.txt

```
psutil
```

XmlRpcController

Listing 33: config/default.conf

```
[controllers]
XmlRpcController = 1

[sources]
XmlRpcMethodCallSource = 1

[XmlRpcMethodCallSource]
controller = XmlRpcController

[XmlRpcController]
```

ZmqDataAccessController

Listing 34: config/default.conf

```
[controllers]
ZmqDataAccessController = 1

[sources]
ZmqDataAccessSource = 1

[ZmqDataAccessSource]
controller = ZmqDataAccessController

[ZmqDataAccessController]
```

Listing 35: requirements.txt

```
pyzmq
```

11.2 Rule configs

CSVRule

Listing 36: config/default.conf

```
[rules]
CSVRule = 1

[CSVRule]
example_rule_101 = 1
example_rule_102 = 1
```

(continues on next page)

```
[example_rule_101]
csv = config/example_rule_101.csv
py = config/example_rule_101.py

[example_rule_102]
csv = config/example_rule_102.csv
py = config/example_rule_102.py
```

Listing 37: config/example_rule_101.csv

```
IntegerSource,TextSource
example-data1-as-int,example-data1-as-text
example-data2-as-int,example-data2-as-text
```

Listing 38: config/example_rule_101.py

```
def setup(shared):
    pass

def expression(intdata, textdata):
    pass
```

InfluxDBLoggerRule

Listing 39: config/default.conf

```
[rules]
InfluxDBLoggerRule = 1

[InfluxDBLoggerRule]
auto_logging_on = 1
```

Listing 40: requirements.txt

```
influxdb
```

INIRule

Listing 41: config/default.conf

```
[rules]
INIRule = 1

[INIRule]
example_rule_101 = config/example_rule_101.ini
example_rule_102 = config/example_rule_102.ini
```

Listing 42: config/example_rule_101.ini

```
[example_rule_101]
on = 1
parsers = IntegerSource, TextSource
module = config/example_rule_101.py
setup = setup
expression = expression
arguments =
    IntegerSource(example-data1-as-int), TextSource(example-data1-as-text)
    IntegerSource(example-data2-as-int), TextSource(example-data2-as-text)
```

Listing 43: config/example_rule_101.py

```
def setup(shared):
    pass

def expression(intdata, textdata):
    pass
```

YAMLRule

Listing 44: config/default.conf

```
[rules]
YAMLRule = 1

[YAMLRule]
example_rule_101 = config/example_rule_101.yaml
example_rule_102 = config/example_rule_102.yaml
```

Listing 45: config/example_rule_101.yaml

```
parsers:
- source: IntegerSource
- source: TextSource

expressions:
- module: config/example_rule_101.py
  setup: setup
  expression: expression
  arguments:
  - source: IntegerSource
    key: example-data1-as-int
  - source: TextSource
    key: example-data1-as-text
- module: config/example_rule_101.py
  setup: setup
  expression: expression
  arguments:
  - source: IntegerSource
    key: example-data2-as-int
```

(continues on next page)

(continued from previous page)

```
- source: TextSource
  key:     example-data2-as-text
```

Listing 46: config/example_rule_101.py

```
def setup(shared):
    pass

def expression(intdata, textdata):
    pass
```

Listing 47: requirements.txt

```
PyYAML
```

12 netdef package

- *netdef.__main__*
- *netdef.service*
- *netdef.windows_service*
- *netdef.systemd_service*
- *netdef.utils*
- *netdef.testutils*

12.1 netdef.__main__

`netdef.__main__.cli()`
entrypoint for use in `setup.py`:

```
entry_points={
    'console_scripts': [
        '{NAME}={MAIN_PACKAGE}.__main__:cli'.format(NAME=NAME, MAIN_PACKAGE=MAIN_
↪PACKAGE),
    ],
},
```

`netdef.__main__.create_project(proj_path, template_config_callback)`

Create project structure in given folder. Add content from `template_config_callback` into `config/default.ini`

Parameters

- **proj_path** (*str*) – project folder
- **template_config_callback** (*str*) – config text

`netdef.__main__.entrypoint(run_callback, template_config_callback)`

Entrypoint to be used in your application. Parses Command line arguments and dispatch functions.

Example from `First-App/first_app/__main__.py`:

```
from netdef.__main__ import entrypoint

def run_app():
    from . import main

def get_template_config():
    from . import defaultconfig
    return defaultconfig.template_config_string

def cli():
    # entrypoint: console_scripts
    entrypoint(run_app, get_template_config)

if __name__ == '__main__':
    # entrypoint: python -m console_scripts
    entrypoint(run_app, get_template_config)
```

`netdef.__main__.framework_entrypoint()`

The main entrypoint for the netdef package. Used by `cli()`.

Parses command line arguments and dispatch functions

`netdef.__main__.generate_certificate(interactive=True)`

Generate ssl certificates using openssl. Files is created in project folder.

- `certificate.pem.key`
- `certificate.pem`
- `certificate.der.key`
- `certificate.der`

Prints result to stdout.

Parameters `interactive` (*bool*) – ask for CN if True.

`netdef.__main__.generate_webadmin_auth(interactive=True)`

Generate a user and password in ini-format. Prints result to stdout. Can be copy-pasted into `config/default.conf`

Parameters `interactive` (*bool*) – ask for user/pass if True. Generate automatically if not.

12.2 netdef.service

`netdef.service.get_service(*args, **kwargs)`

Note: This function is only implemented for Windows and Systemd based linux distributions

Returns the Service-class to use as argument in `run_service()`

Parameters

- **svc_name** – name of the service
- **exe_name** – filename of the service
- **app_callback** – a function that will start your application

- **template_callback** – a function that returns template config

Returns GenericApplicationService

Example:

```
from netdef.service import get_service, run_service

def run_app():
    from . import main

def get_template_config():
    from . import defaultconfig
    return defaultconfig.template_config_string

application_service = get_service("First-App", "First-App-Service", run_app, get_
↳template_config)
run_service(application_service)
```

netdef.service.**run_service**(*args, **kwargs)

Note: This function is only implemented for Windows and Systemd based linux distributions

Parameters app_service_class – service class from *get_service()*

Create an instance of app_service_class and run as service

Example:

```
from netdef.service import get_service, run_service

def run_app():
    from . import main

def get_template_config():
    from . import defaultconfig
    return defaultconfig.template_config_string

application_service = get_service("First-App", "First-App-Service", run_app, get_
↳template_config)
run_service(application_service)
```

12.3 netdef.windows_service

class netdef.windows_service.GenericApplicationService(*args*)

Bases: sphinx.ext.autodoc.importer._MockObject

SvcDoRun()

SvcStop()

application = None

netdef.windows_service.**get_service**(*svc_name*, *exe_name*, *app_callback*, *tem-*
plate_callback=None)

Note: This function is only implemented for Windows and Systemd based linux distributions

Returns the Service-class to use as argument in `run_service()`

Parameters

- **svc_name** – name of the service
- **exe_name** – filename of the service
- **app_callback** – a function that will start your application
- **template_callback** – a function that returns template config

Returns `GenericApplicationService`

Example:

```
from netdef.service import get_service, run_service

def run_app():
    from . import main

def get_template_config():
    from . import defaultconfig
    return defaultconfig.template_config_string

application_service = get_service("First-App", "First-App-Service", run_app, get_
↳template_config)
run_service(application_service)
```

`netdef.windows_service.run_service(app_service_class)`

Note: This function is only implemented for Windows and Systemd based linux distributions

Parameters `app_service_class` – service class from `get_service()`

Create an instance of `app_service_class` and run as service

Example:

```
from netdef.service import get_service, run_service

def run_app():
    from . import main

def get_template_config():
    from . import defaultconfig
    return defaultconfig.template_config_string

application_service = get_service("First-App", "First-App-Service", run_app, get_
↳template_config)
run_service(application_service)
```

12.4 netdef.systemd_service

`netdef.systemd_service` can also be invoked directly using the `-m` switch of the interpreter with `proj_path` as argument.

This example installs the project in current directory as a service:

```
$ python -m netdef.systemd_service -i .
```

```
class netdef.systemd_service.ApplicationService(svc_name, exe_name, app_callback,  
                                                template_callback)
```

Bases: tuple

app_callback

Alias for field number 2

exe_name

Alias for field number 1

svc_name

Alias for field number 0

template_callback

Alias for field number 3

```
netdef.systemd_service.get_service(svc_name, exe_name, app_callback, template_callback)
```

Note: This function is only implemented for Windows and Systemd based linux distributions

Returns the Service-class to use as argument in `run_service()`

Parameters

- **svc_name** – name of the service
- **exe_name** – filename of the service
- **app_callback** – a function that will start your application
- **template_callback** – a function that returns template config

Returns GenericApplicationService

Example:

```
from netdef.service import get_service, run_service

def run_app():
    from . import main

def get_template_config():
    from . import defaultconfig
    return defaultconfig.template_config_string

application_service = get_service("First-App", "First-App-Service", run_app, get_
→template_config)
run_service(application_service)
```

```
netdef.systemd_service.install_service(proj_path, service_file, svc_name, user)
```

Note: This function is only implemented for Systemd based linux distributions

Creates a systemd service file in /etc/systemd/system/

`netdef.systemd_service.run_service(app_service_class)`

Note: This function is only implemented for Windows and Systemd based linux distributions

Parameters `app_service_class` – service class from `get_service()`

Create an instance of `app_service_class` and run as service

Example:

```
from netdef.service import get_service, run_service

def run_app():
    from . import main

def get_template_config():
    from . import defaultconfig
    return defaultconfig.template_config_string

application_service = get_service("First-App", "First-App-Service", run_app, get_
→template_config)
run_service(application_service)
```

12.5 netdef.utils

`netdef.utils.handle_restart(shared, engine)`

By calling this function your application will restart on SystemExit if `shared.restart_on_exit` is True.

Parameters

- **shared** – instance of `netdef.Shared.Shared`
- **engine** – instance or subclass of `netdef.Engines.BaseEngine.BaseEngine`

Example:

```
from netdef.utils import handle_restart
...
engine.init()
engine.start()
engine.block() # until ctrl-c or SIG_TERM
engine.stop()
handle_restart(shared, engine)
```

`netdef.utils.setup_logging(config)`

Parse the config file for:

```
[logging]
logglevel
loggformat
loggdatefmt
loggfile
to_console
to_file
```

Then the logging module is set according to the configs

Parameters `config` – instance of `netdef.Shared.SharedConfig.Config`

Example:

```
...
from netdef.Shared import Shared
from netdef.utils import setup_logging
shared = Shared.Shared("First-App", install_path, proj_path, config_string)
setup_logging(shared.config)
...
```

12.6 netdef.testutils

class `netdef.testutils.MockExpression` (**kwargs)

Bases: `object`

Example:

```
from netdef.testutils import MockExpression

def test_hello():
    mock = MockExpression(
        module="config/command_rule.py",
        intern=InternalSource("generic"),
        cmd=CmdSource("echo hello")
    )
    mock.intern.update_value(None, stat_init=True)
    mock.cmd.assert_called_once_with("world")
    mock.intern.assert_not_called()
```

get_callbacks ()

get_module ()

Returns the expression module

set_init_values (**kwargs)

set_none_values (**kwargs)

class `netdef.testutils.MockShared` (`config_string=""`)

Bases: `netdef.Shared.Shared.Shared`

class `netdef.testutils.MockSource` (`expression`, `source`)

Bases: `object`

assert_any_call (`value`)

assert_called ()

assert_called_once ()

assert_called_once_with (*value*)

assert_called_with (*value*)

assert_not_called ()

assert_value (*value*)

A helper function to assert value and timestamp

call_args

call_args_list

call_count

update_value (*val*, *stime=None*, *stat_none=False*, *stat_init=False*, *stat_good=False*,
stat_invalid=False, *run_expression=True*)

A Helper function to update values in expression

13 netdef.Controllers package

- *Controllers*
- *Abstract base controllers*
 - *BaseController*
 - *BaseAsyncController*
- *Built-in controller modules*
 - *CommTestController*
 - *ConcurrentWebRequestController*
 - *CrontabController*
 - *InfluxDBLoggerController*
 - *InternalController*
 - *ModbusClientController*
 - *ModbusServerController*
 - *MQTTDataMessageController*
 - *OPCUAClientController*
 - *OPCUAServerController*
 - *RESTJsonController*
 - *SubprocessController*
 - *SystemMonitorController*
 - *XmlRpcController*
 - *ZmqDataAccessController*

13.1 Controllers

class netdef.Controllers.Controllers.**Controllers** (*shared=None*)

Bases: object

A collection of all loaded controllers

add_shared_object (*shared*)

init ()

load (*base_packages*)

Imports controller modules. Creates queue instances associated with the given controllers.

Example:

```
from netdef.Controllers import Controllers
controllers = Controllers.Controllers(shared)
controllers.load([__package__, 'netdef'])
```

netdef.Controllers.Controllers.**register** (*name, classref=None*)

A decorator to register controllers. Example:

```
from netdef.Controllers import BaseController, Controllers

@Controllers.register("NewControllerTemplate")
class NewControllerTemplate(BaseController.BaseController):
    def __init__(self, name, shared):
        ...
```

Can also be called as a normal function:

```
from netdef.Controllers import BaseController, Controllers

def setup(shared):
    Controllers.register("NewControllerTemplate", NewControllerTemplate)

class NewControllerTemplate(BaseController.BaseController):
    def __init__(self, name, shared):
        ...
```

Parameters

- **name** (*str*) – Name of the controller class
- **classref** (*object*) – Should be *None* if used as a decorator and a *class* if called as a function

Returns A callable that returns a *class* if used as a decorator and a *class* if called as a normal function

13.2 Abstract base controllers

BaseController

This is an abstract baseclass

class netdef.Controllers.BaseController.**BaseController** (*name, shared*)

Bases: object

Abstract class for controllers.

Parameters

- **name** (*str*) – Name to be used in logfiles
- **shared** – a reference to the shared object

add_interrupt (*interrupt*)

Setup the interrupt signal

add_logger (*name*)

Setup logging module

add_parser (*parser*)

Add parser if not already exists

add_source (*name, init_value*)

Add a source to the storage dict. Override if something else is needed.

clear_incoming (*until_empty=True, until_messagetype=None*)

Delete all messages from incoming queue.

Parameters

- **until_empty** (*bool*) – If True the function will block until queue is empty. If False it will block forever.
- **until_messagetype** (*MessageType*) – Block until given messagetype is received

Example:

```
...
while not self.has_interrupt():
    reconnect = False
    try:
        if reconnect:
            self.clear_incoming()
            self.try_reconnect()
        # main loop
        while not self.has_interrupt():
            self.loop_incoming()
            self.loop_outgoing()
    except ConnectionError:
        reconnect = True
...
```

fetch_one_incoming ()

Returns one message from the queue.

Returns tuple of (messagetype, incoming)

Return type tuple(*MessageType, BaseSource*)

get_parsers ()

Return parser storage

get_source (*name*)

Return named source

get_sources ()

Return source storage

handle_add_parser (*incoming*)
Add parser to controller if not already exists

handle_add_source (*incoming*)

handle_app_state (*app_state*)
Override if controller need to react to application states

handle_app_state_running ()
Override if controller need to react to running state

handle_app_state_setup ()
Override if controller need to react to setup state

handle_read_source (*incoming*)

handle_readall (*incoming*)

handle_tick (*incoming*)
Answer the tick message

handle_write_source (*incoming, value, source_time*)

has_interrupt ()
Returns True if the interrupt signal is received

has_source (*name*)
Return True if source name is found

init_parsers (*parsers*)
Setup the parser storage as a list. Override if something else is needed.

init_queue ()
Setup the message queue and timeout

init_sources (*sources*)
Setup the source storage as a dict. Override if something else is needed.

loop_incoming (*until_empty=True, until_timeout=0.0, until_messagetype=None, until_app_state=None*)
Get every message from the queue and dispatch the associated handler function.

Parameters

- **until_empty** (*bool*) – Blocking until queue is empty
- **until_timeout** (*float*) – Timeout in seconds. 0.0 blocks forever.
- **until_messagetype** (*MessageType*) – Blocking until given messagetype is dispatched
- **until_app_state** (*AppStateType*) – Blocking until given app_state is dispatched

loop_outgoing ()
Check every source and call the poll_outgoing_item function

loop_until_app_state_running ()
Usefull if you want your controller to block while ADD_SOURCE and ADD_PARSER is dispatched

Example:

```
def run(self):
    self.loop_until_app_state_running()
    while not self.has_interrupt():
        try:
```

(continues on next page)

(continued from previous page)

```
self.handle_connection()
while not self.has_interrupt():
    self.loop_incoming()
    self.loop_outgoing()
except ConnectionError:
    self.handle_conn_error()
```

poll_outgoing_item (*item*)

run ()

Override this function in controller. Example:

```
def run(self):
    self.logger.info("Running")

    while not self.has_interrupt():
        self.loop_incoming() # dispatch handle_* functions
        self.loop_outgoing() # dispatch poll_* functions

    self.logger.info("Stopped")
```

send_outgoing (*outgoing*)

Send RUN_EXPRESSION message on valuechange

sleep (*seconds*)

” Sleep by waiting for the interrupt. Should be used instead of time.sleep. Override if sleep should be interrupted by even more signals

statistics_update ()

classmethod update_source_instance_status (*source_instance, status_ok, old-new_check*)

Updates state on given source_instance Returns True if source_instance have triggered a value change

static update_source_instance_value (*source_instance, value, stime, status_ok, old-new_check*)

Updates value, timestamp and state on given source_instance Returns True if source_instance have triggered a value change

BaseAsyncController

```
class netdef.Controllers.BaseAsyncController.BaseAsyncController (name,
                                                                    shared)
```

Bases: *netdef.Controllers.BaseController.BaseController*

Tip: Development Status :: 5 - Production/Stable

get_event_loop ()

handle_app_state_running ()

Override if controller need to react to running state

init_asyncio ()

loop_incoming_until_interrupt ()

run ()

Override this function in controller. Example:

```

def run(self):
    self.logger.info("Running")

    some_client = SomeAsyncioClient()

    # Start polling of the blocking incoming queue in a thread executor
    self.loop.run_in_executor(None, self.loop.incoming_until_interrupt)

    # TODO: define a coroutine that stops your async client when called.
    async def stop_some_client():
        await some_client.stop()

    # register coroutine to be run at interrupt / shutdown
    self.loop.create_task(self.run_async_on_interrupt(stop_some_client))

    # TODO: start your client coroutine
    self.loop.run_until_complete(some_client.start())

    self.logger.info("Stopped")

```

`run_async_on_interrupt` (callback)

Built-in controller modules:

13.3 Built-in controller modules

CommTestController

class `netdef.Controllers.CommTestController.CommTestController` (*name, shared*)
 Bases: `netdef.Controllers.BaseAsyncController.BaseAsyncController`

Tip: Development Status :: 5 - Production/Stable

This class will send TCP or ICMP ping requests based on sources received in ADD_SOURCE messages and store the result into given sources. When result is stored into a source this class will send the changed source in a RUN_EXPRESSION message to the source's rule.

Parameters

- **name** (*str*) – Name of controller
- **shared** (`netdef.Shared`) – Instance of applications shared object.

Configuration:

- **timeout** – Connection timeout in seconds
- **interval** – Poll interval in seconds
- **test_type** – Available types: [tcpip, ping]
- **max_concurrent_sockets** – Max number of simultaneous open sockets.
- **disable** – If disabled this controller will enter running state but all messages will be discarded.

Defaults:

```
[CommTestController]
timeout = 2
interval = 10
test_type = tcpip
max_concurrent_sockets = 1000
disable = 0
```

loop_outgoing_until_interrupt ()
Main coroutine. loops until interrupt is set.

run ()
Main thread loop. Will exit when receiving interrupt signal Sets up

ConcurrentWebRequestController

class netdef.Controllers.ConcurrentWebRequestController.**ConcurrentWebRequestController** (*name*, *share*)

Bases: *netdef.Controllers.BaseAsyncController.BaseAsyncController*

Danger: Development Status :: 3 - Alpha

Basically just a web scraper. Can scrape multiple web pages simultaneously.

IO is handled by this controller. The poll interval and program flow is implemented in ConcurrentWebRequestSource

get_client_session (item)
Returns a aiohttp session. Add new session to source if not found. Session will be initialized with basic auth and a default timeout

handle_add_source (incoming)
Add source to controller

handle_write_source (incoming, value, source_time)
execute a command if given value is the name of a command

init_task_limit ()
Read configuration

loop_outgoing_until_interrupt ()
Main async loop.

process_task (item, method)
Retrives data from web site and packs it into the source

process_web_request_item (item, method, session)
handle IO by interfacing with the sources data generator

run ()
Main sync loop

class netdef.Controllers.ConcurrentWebRequestController.**NextInterval** (*timestamp*)
Bases: object

Call next() to retrieve seconds to next interval, and which interval it is

add (interval)

has_interval ()

next (*now*)
spans
start

CrontabController

class `netdef.Controllers.CrontabController.CrontabController` (*name, shared*)
Bases: `netdef.Controllers.BaseController.BaseController`

Tip: Development Status :: 5 - Production/Stable

poll_outgoing_item (*item*)
Check if it is time to trigger event for given source
Parameters **item** – source instance to check

run ()
Main loop. Will exit when receiving interrupt signal

InfluxDBLoggerController

class `netdef.Controllers.InfluxDBLoggerController.InfluxDBLoggerController` (*name, shared*)
Bases: `netdef.Controllers.BaseController.BaseController`

Danger: Development Status :: 3 - Alpha

A logging controller. Its purpose is to store every write event into influxdb.

handle_write_source (*incoming, value, source_time*)
Write given value and timestamp into influxdb

Parameters

- **incoming** (`InfluxDBLoggerSource`) – source instance
- **value** – frozen value if instance
- **source_time** (`datetime.datetime`) – value timestamp

run ()
Main loop. Will exit when receiving interrupt signal

InternalController

class `netdef.Controllers.InternalController.InternalController` (*name, shared*)
Bases: `netdef.Controllers.BaseController.BaseController`

Tip: Development Status :: 5 - Production/Stable

Internal variables that works just like any other value from a controller. Can trigger events on valuechanges. State can be cached to disk.

Parameters

- **name** (*str*) – The name is used i logfile and default.ini
- **shared** (*Shared*) – Instance of applications shared object.

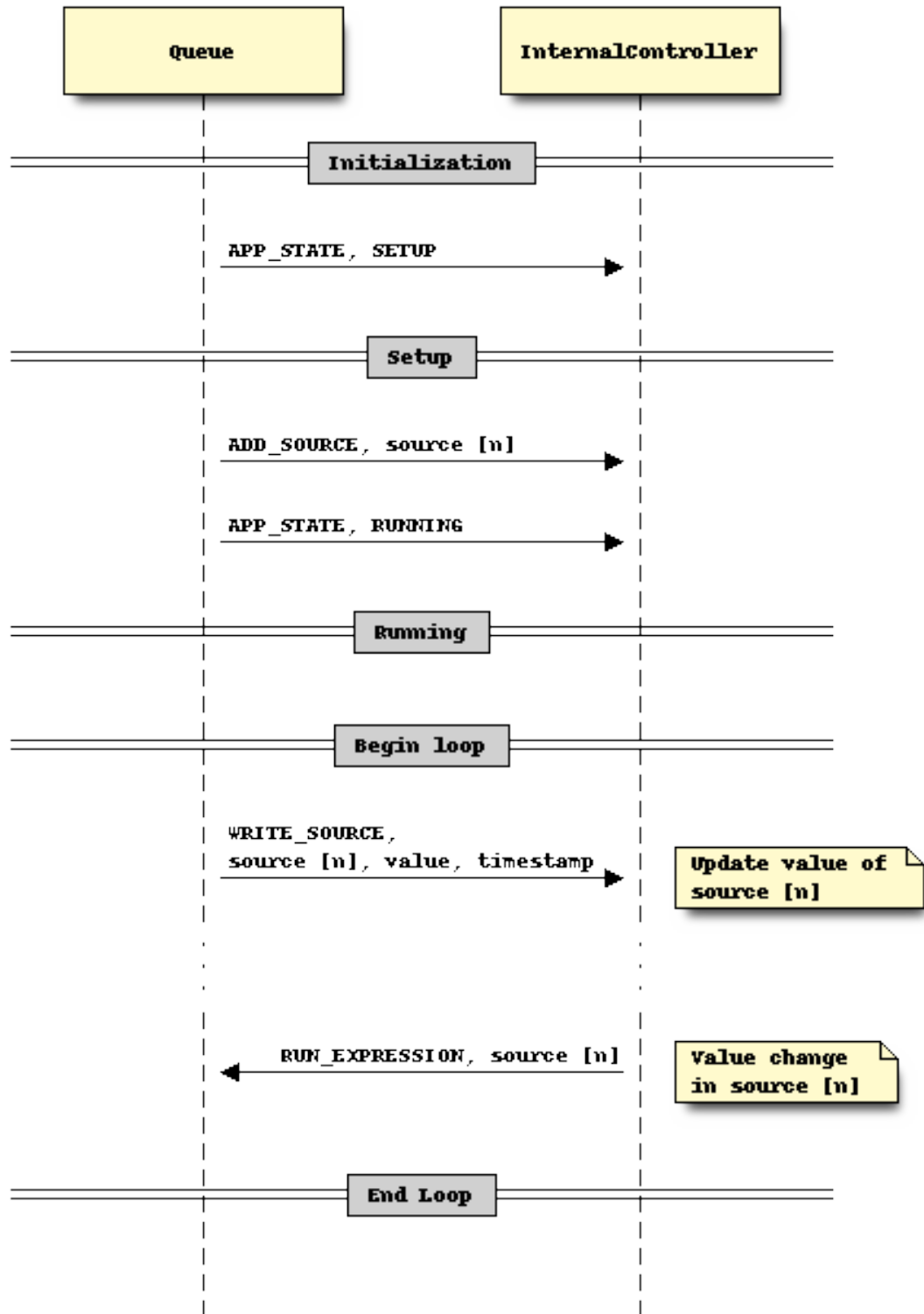
Configuration

```
[InternalController]
send_init_event = 0
send_events = 0
persistent_value = 0
key_in_filename = 0
```

Options

- **send_init_event** – trigger RUN_EXPRESSION with StatusCode.INITIAL for every source at startup
- **send_events** – trigger a RUN_EXPRESSION message for every WRITE_SOURCE message
- **persistent_value** – store values to disk
- **key_in_filename** – use source key as prefix in filename for persistent storage

Sequence diagram



get_cache_filename (*key*)

Generate sha256 hash to be used as filename. If config key_in_filename=1 then key will be prefixed to the hexdigest. Valid characters: a-z A-Z 0-9 _-

Parameters **key** (*str*) – string to encode

Returns filename

handle_add_source (*incoming*)

Add given source instance to internal source list

Parameters **incoming** (*InternalSource*) – source instance

handle_write_source (*incoming, value, source_time*)

Update internal dict with new value.

Parameters

- **incoming** (*InternalSource*) – source instance
- **value** – frozen value of instance
- **source_time** (*datetime.datetime*) – value timestamp

poll_outgoing_item (*item*)

Check if given source should be cached to disk.

Parameters **item** (*InternalSource*) – source instance

run ()

Main loop. Will exit when receiving interrupt signal

store_to_disk (*item=None*)

Store sources into files at [proj-path]/db/internal/

ModbusClientController

class netdef.Controllers.ModbusClientController.**ModbusClientController** (*name, shared*)

Bases: *netdef.Controllers.BaseController.BaseController*

Tip: Development Status :: 5 - Production/Stable

Read and write holding registers of a modbus device.

Parameters

- **name** (*str*) – The name is used i logfile and default.ini
- **shared** (*Shared*) – reference to the global shared instance

Settings:

```
[ModbusClientController]

# connection
host = 127.0.0.1
port = 5020

# RUN_EXPRESSION is only sent if value has changed
oldnew_comparision = 1

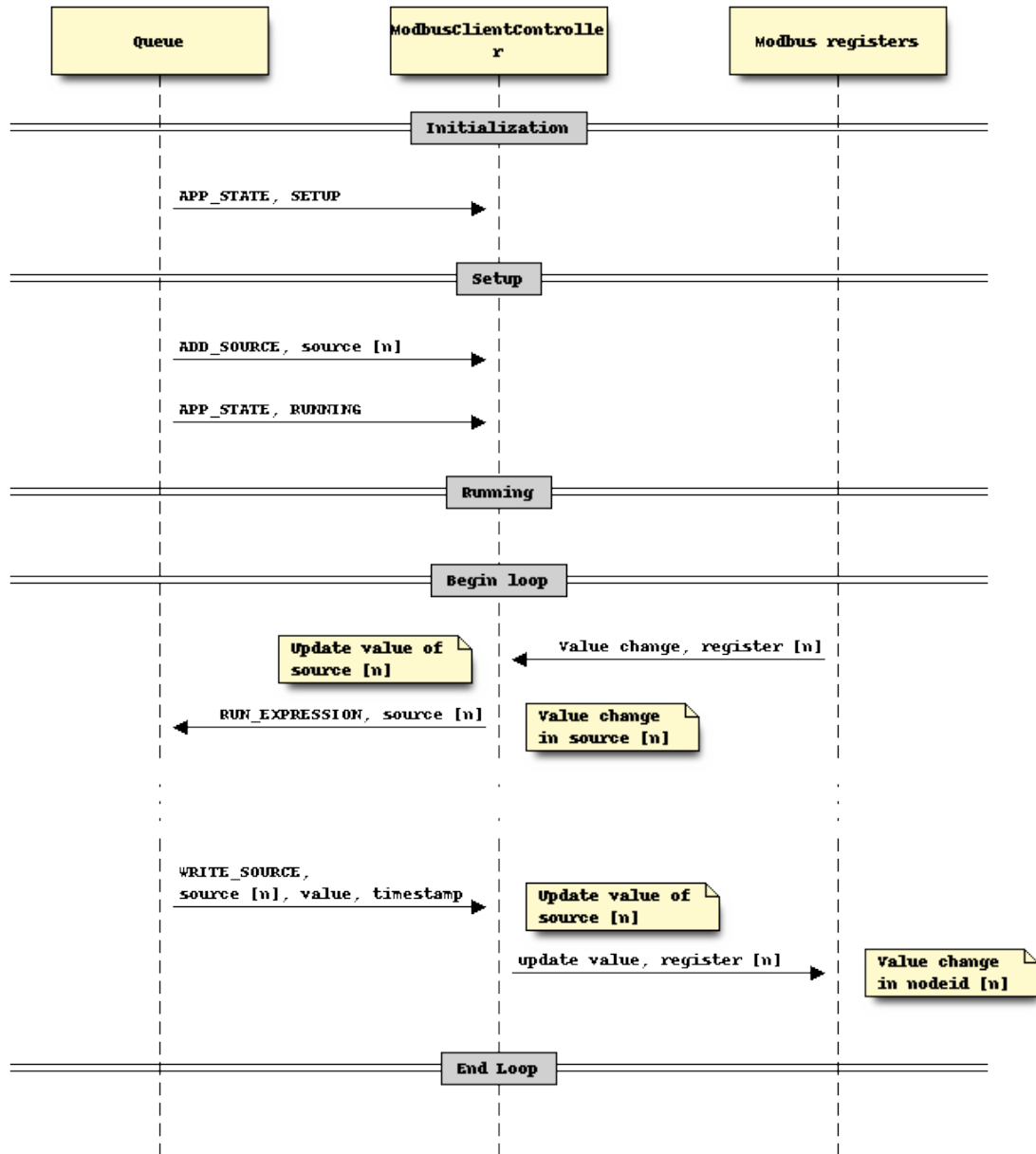
# cooldown on connection error og write error
reconnect_timeout = 20

# Buffer or clear write requests recieved during cooldown
clear_writes_on_disconnect = 1
```

(continues on next page)

```
# Polling interval  
poll_interval = 0.5
```

Sequence diagram:



handle_add_source (*incoming*)

Add given source instance to internal source list

Parameters **incoming** (*HoldingRegisterSource*) – source instance

handle_write_source (*incoming, value, source_time*)

Write given value to the connected modbus device.

Parameters

- **incoming** (*HoldingRegisterSource*) – source instance
- **value** – frozen value of instance
- **source_time** (*datetime.datetime*) – value timestamp

poll_outgoing_item (*item*)

Poll given source for its value in the modbus device

Parameters **item** (*HoldingRegisterSource*) – source instance

run ()

Main loop. Will exit when receiving interrupt signal

safe_disconnect ()

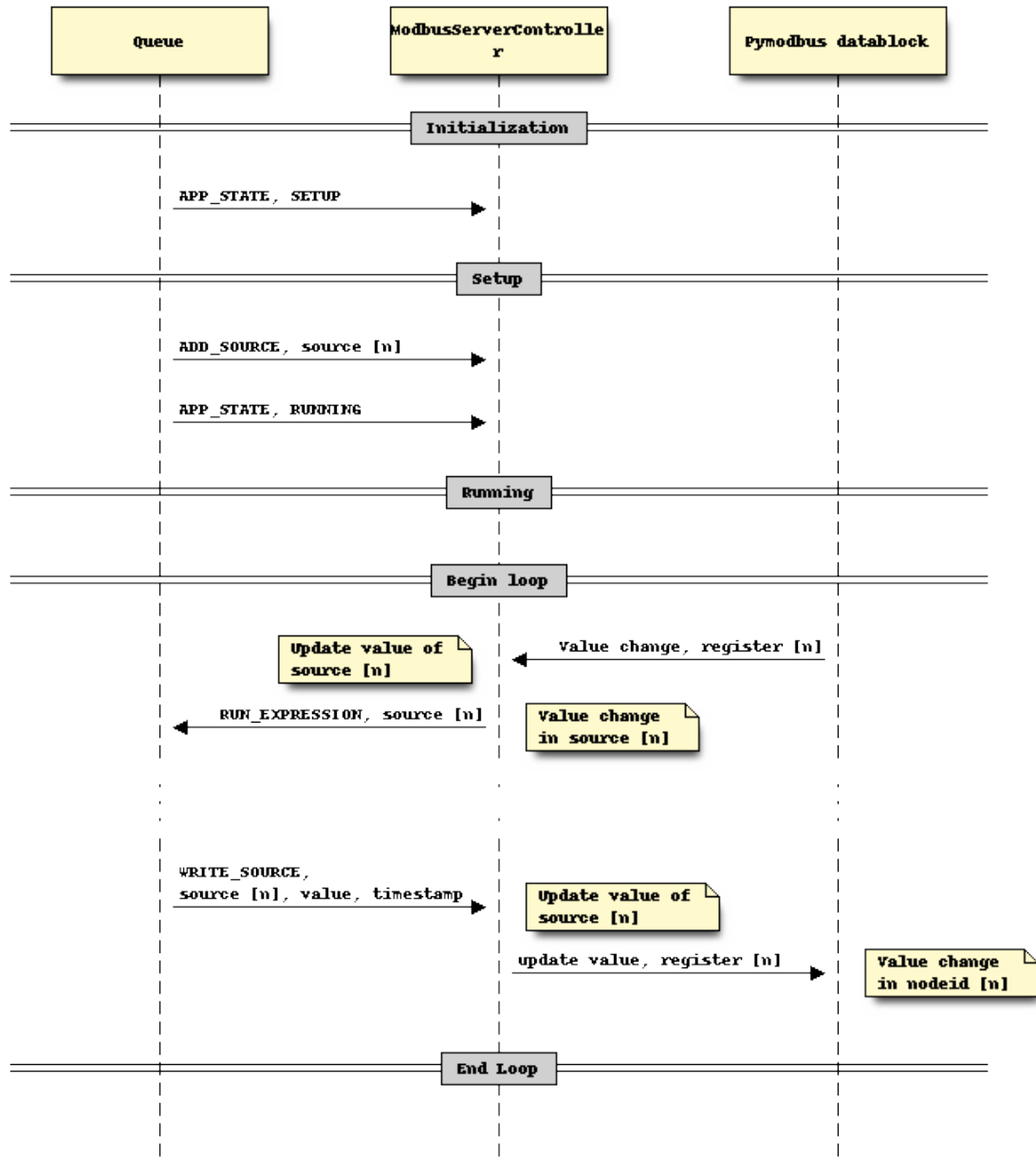
Close the tcp socket if it is connected

ModbusServerController

```
class netdef.Controllers.ModbusServerController.ModbusServerController ( name,  
 shared)  
Bases:  netdef.Controllers.BaseController.BaseController 
```

Tip: Development Status :: 5 - Production/Stable

Sequence diagram:



get_framer ()

Returns the framer to be used. Override this function to return a custom framer

get_modbus_server_context ()

Iter the devicelist section in config-file and builds a ModbusServerContext object

Returns an ModbusServerContext instance

handle_add_source (*incoming*)

handle_datachange (*unit, address, value, is_internal*)

handle_write_source (*incoming, value, source_time*)

init_server (*context, framer, identity, host, port*)

run ()
Main loop. Will exit when receiving interrupt signal

class `netdef.Controllers.ModbusServerController.MyContext (*args, **kwargs)`
Bases: `pymodbus.datastore.context.ModbusSlaveContext`

setValues (*fx, address, values, is_internal=False*)
Sets the datastore with the supplied values

Parameters

- **fx** – The function we are working with
- **address** – The starting address
- **values** – The new values to be set

class `netdef.Controllers.ModbusServerController.MyController (*args, **kwargs)`
Bases: `pymodbus.server.sync.ModbusTcpServer`

daemon_threads = False

service_actions ()
Called by the `serve_forever()` loop.

May be overridden by a subclass / Mixin to implement any code that needs to be run during the loop.

MQTTDataMessageController

class `netdef.Controllers.MQTTDataMessageController.MQTTDataMessageController (name, shared)`
Bases: `netdef.Controllers.BaseController.BaseController`

Danger: Development Status :: 3 - Alpha
--

get_key (*topic*)

get_topic (*topic*)

handle_add_source (*incoming*)

handle_write_source (*incoming, value, source_time*)

loop_mqtt ()

mqtt_connect ()

mqtt_safe_disconnect ()

on_connect (*client, userdata, flags, rc*)

on_disconnect (*client, userdata, rc*)

on_message (*client, userdata, msg*)

publish_data_item (*topic, payload*)

run ()
Main loop. Will exit when receiving interrupt signal

OPCUAClientController

```
class netdef.Controllers.OPCUAClientController.OPCUAClientController (name,  
                                                                    shared)  
    Bases: netdef.Controllers.BaseController.BaseController
```

Caution: Development Status :: 4 - Beta
--

```
config (key, default)
```

```
handle_add_source (incoming)
```

```
handle_write_source (incoming, value, source_time)
```

```
loop_outgoing ()
```

```
    Check every source and call the poll_outgoing_item function
```

```
run ()
```

```
    Main loop. Will exit when receiving interrupt signal
```

```
safe_disconnect ()
```

```
send_datachange (nodeid, value, stime, status_ok)
```

```
class netdef.Controllers.OPCUAClientController.SubHandler (parent)  
    Bases: object
```

```
    Client to subscription. It will receive events from server
```

```
datachange_notification (node, value, data)
```

```
event_notification (event)
```

```
status_change_notification (status)
```

OPCUAServerController

```
class netdef.Controllers.OPCUAServerController.CustomAnonInternalSession (internal_server,  
                                                                    as-  
                                                                    pace,  
                                                                    sub-  
                                                                    mgr,  
                                                                    name,  
                                                                    user=<sphinx.ext.autodoc  
                                                                    ob-  
                                                                    ject>,  
                                                                    ex-  
                                                                    ter-  
                                                                    nal=False)
```

```
    Bases: opcua.server.internal_server.InternalSession
```

```
    Custom InternalSession will set timestamp when missing
```

```
write (params)
```

```
class netdef.Controllers.OPCUAServerController.CustomInternalSession (internal_server,  
aspace,  
sub-  
mgr,  
name,  
user=<sphinx.ext.autodoc.impo  
ob-  
ject>,  
exter-  
nal=False)
```

Bases: *netdef.Controllers.OPCUAServerController.CustomAnonInternalSession*

This custom InternalSession will block anonymous access

```
activate_session (params)
```

```
class netdef.Controllers.OPCUAServerController.CustomServer (shelffile=None,  
iserver=None)
```

Bases: *opcua.server.server.Server*

Custom Server that enables Basic128Rsa15 and Basic256

```
class netdef.Controllers.OPCUAServerController.OPCUAServerController (name,  
shared)
```

Bases: *netdef.Controllers.BaseController.BaseController*

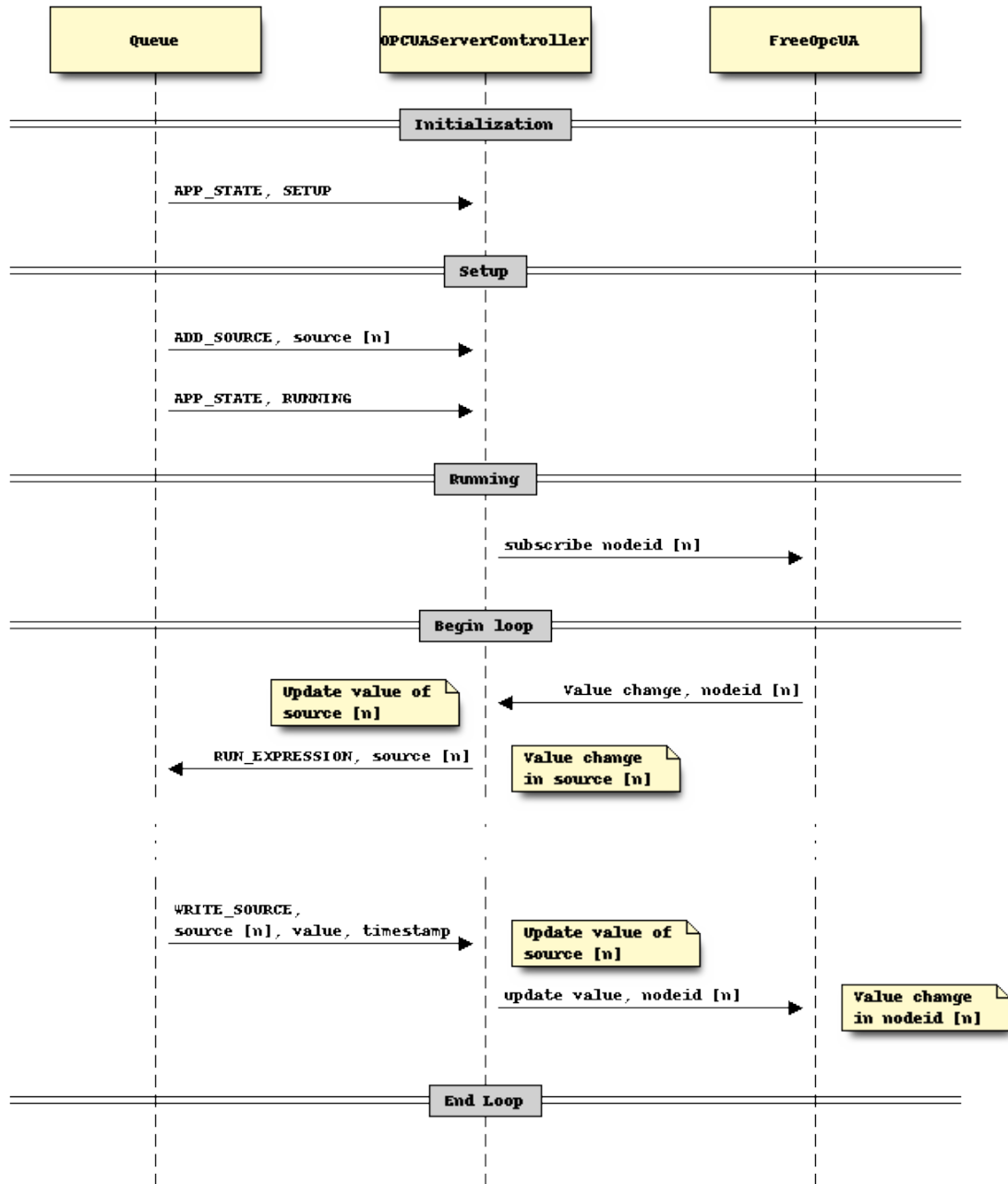
Tip: Development Status :: 5 - Production/Stable

This Controller will start a freeopcua server instance and will add a nodeid for all sources received in ADD_SOURCE messages.

When a client writes a new value this event will be forwarded to the associated source and a RUN_EXPRESSION message will be sent.

When a WRITE_SOURCE message is received the value for the associated source will be updated in the server and all connected clients will receive a value update

Sequence diagram:



add_folder (*parent, foldername*)

Add a folder in server

add_variablenode (*parent, ref, val, varianttype*)

Create and add a variable in server and return the variable node

build_folders (*parent, ref, sep*)

create_datavalue (*val, datatype, statuscode, timestamp*)

Create a value for the server that keep the correct datatype

create_monitored_items (*event, dispatcher*)

write a warning to logfile if the client add a nodeid that does not exists

get_default_value (*incoming*)
Returns the default value of the source value

get_nodeid (*incoming*)
Returns the nodeid from the source

get_varianttype (*incoming*)
Returns the varianttype from the source

handle_add_source (*incoming*)
Add a source to the server

handle_write_source (*incoming, value, source_time*)
Receive a value change from an expression and update the server

is_writable (*incoming*)
Returns True if source is writable for the opcua client

modify_monitored_items (*event, dispatcher*)

run ()
Main loop. Will exit when receiving interrupt signal

send_datachange (*nodeid, value, stime, status_ok, ua_status_code*)
Triggers a RUN_EXPRESSION message for given source

class netdef.Controllers.OPCUAServerController.**SubHandler** (*controller*)
Bases: object

The subscription handler for the server. Will send value changes i server to the controller.

datachange_notification (*node, val, data*)

event_notification (*event*)

RESTJsonController

class netdef.Controllers.RESTJsonController.**RESTJsonController** (*name, shared*)
Bases: *netdef.Controllers.BaseController.BaseController*

Tip: Development Status :: 5 - Production/Stable

connect ()

handle_add_source (*incoming*)

handle_read_source (*incoming*)

handle_readall (*incoming*)

handle_write_source (*incoming, value*)

loop_outgoing ()
Check every source and call the poll_outgoing_item function

parse_item (*item*)

run ()
Main loop. Will exit when receiving interrupt signal

send_datachange (*key, source_time, value*)

urlerrorhandling ()

SubprocessController

class netdef.Controllers.SubprocessController.**NextInterval** (*timestamp*)

Bases: object

Call next() to retrieve seconds to next interval, and which interval it is

add (*interval*)

has_interval ()

next (*now*)

spans

start

class netdef.Controllers.SubprocessController.**SubprocessController** (*name*,
shared)

Bases: *netdef.Controllers.BaseController.BaseController*

Danger: Development Status :: 3 - Alpha
--

handle_add_source (*incoming*)

handle_write_source (*incoming*, *value*, *source_time*)

parse_item (*item*)

parse_response (*response*)

poll_outgoing_item (*item*)

run ()

Main loop. Will exit when receiving interrupt signal

send_datachange (*source_key*, *value*, *source_time*, *status_ok*)

setup_interval_plan ()

netdef.Controllers.SubprocessController.**stdout_from_terminal** (*command_as_str*,
err_msg=None)

SystemMonitorController

class netdef.Controllers.SystemMonitorController.**DataItem** (*source_type*, *key*, *interval*,
func, *args=None*)

Bases: object

args

Arguments for self.func callback

func

Callback to retrieve value

get_value ()

Returns value of self.func callback

interval

Poll interval

key
Unique identifier

next
Next scheduled call to `self.func`

ready()
Returns True if interval for this item has elapsed.

source_type
Reference to a `SystemMonitorSource` class

class `netdef.Controllers.SystemMonitorController`.**SystemMonitorController** (*name*, *shared*)
Bases: `netdef.Controllers.BaseController`.`BaseController`

Tip: Development Status :: 5 - Production/Stable

handle_add_source (*incoming*)

handle_write_source (*incoming*, *value*, *source_time*)

poll_data ()
Iter the dict of `DataItem` and get values.

run ()
Main loop. Will exit when receiving interrupt signal

send_datachange (*source_key*, *value*, *stime*, *status_ok*)

`netdef.Controllers.SystemMonitorController`.**get_clean_mount_point_name** (*node*)
Replace / or with .

Example:

```
for disk in psutil.disk_partitions():
    print (get_clean_mount_point_name(disk.mountpoint))
```

Parameters `node` (*str*) – name of mountpoint

Returns new node name

`netdef.Controllers.SystemMonitorController`.**get_data_items_dict** (*mempoll*, *cpupoll*, *poll*, *checkdisk*, *diskpoll*)

Create a dict with items to monitor.

Parameters

- **mempoll** (*int*) – poll interval for memory callbacks
- **cpupoll** (*int*) – poll interval for cpu callbacks
- **poll** (*int*) – general poll interval
- **checkdisk** (*bool*) – Set True to poll disk drives
- **diskpoll** (*int*) – poll interval for disk drives

Returns dict of `DataItem`

`netdef.Controllers.SystemMonitorController.get_proc()`
Helperfunction.

Returns `psutil.Process`

`netdef.Controllers.SystemMonitorController.get_vm()`
Helperfunction.

Returns `psutil.virtual_memory`

`netdef.Controllers.SystemMonitorController.statistics_update(item)`
Write internal statistics to the Statistics singleton if activated

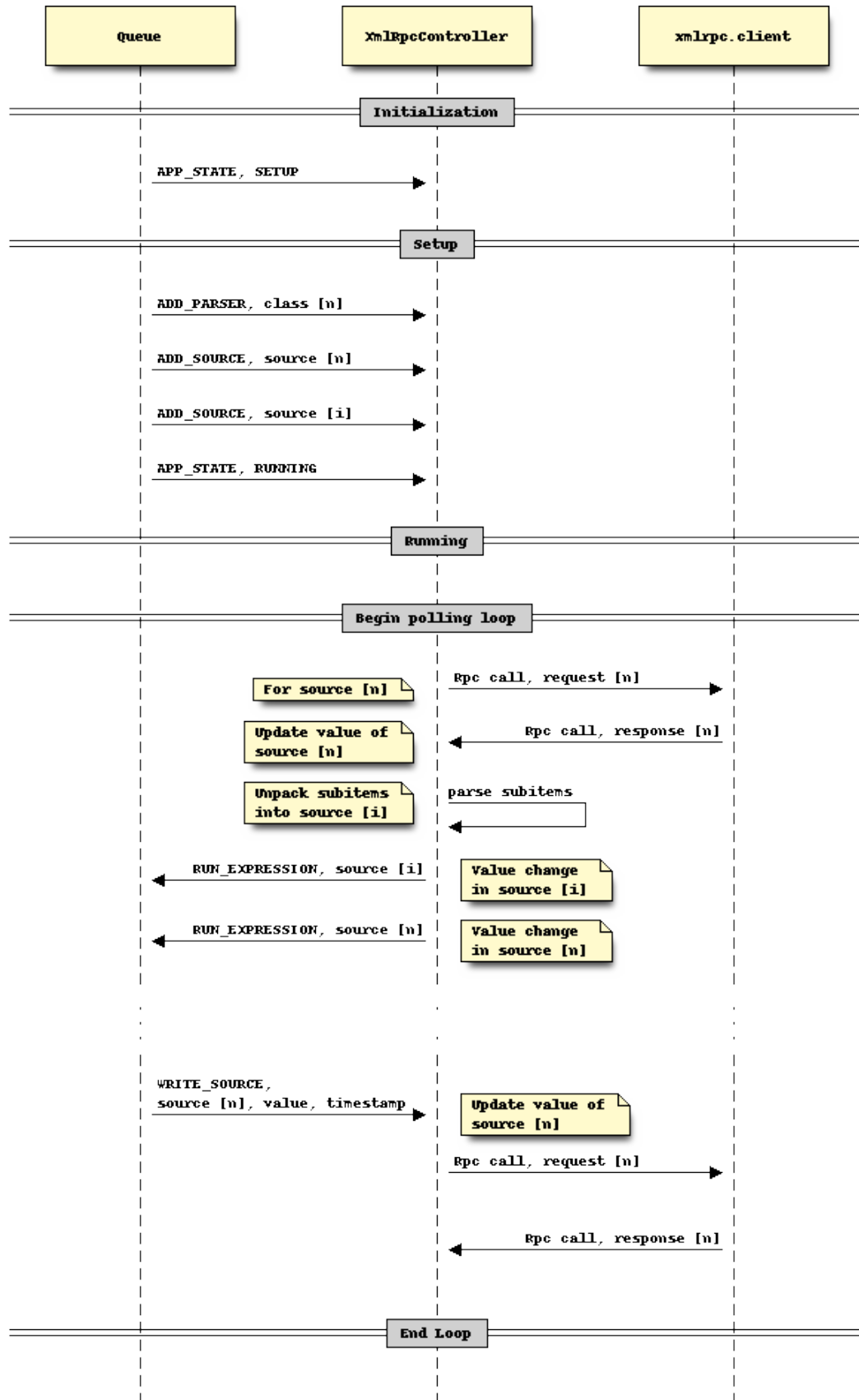
XmlRpcController

class `netdef.Controllers.XmlRpcController.XmlRpcController` (*name, shared*)

Bases: `netdef.Controllers.BaseController.BaseController`

Tip: Development Status :: 5 - Production/Stable

Sequence diagram:



```

handle_add_source (incoming)
handle_read_source (incoming)
handle_readall (incoming)
handle_write_source (incoming, value, source_time)
parse_item (item)
parse_response (response)
poll_outgoing_item (item)
rpc_call (item, value)
run ()
    Main loop. Will exit when receiving interrupt signal
send_datachange (key, source_time, value)

```

ZmqDataAccessController

```

class netdef.Controllers.ZmqDataAccessController.ZmqDataAccessController (name,
                                                                    shared)
    Bases: netdef.Controllers.BaseController.BaseController

```

<p>Danger: Development Status :: 3 - Alpha</p>

```

connect ()
handle_add_source (incoming)
handle_write_source (incoming, value, source_time)
loop_subscribers ()
run ()
    Main loop. Will exit when receiving interrupt signal

```

14 netdef.Engines package

- *Abstract baseclass*
 - *BaseEngine*
- *Expressions*
 - *Expression*
 - *Collector*
- *Built-in engine modules*
 - *ThreadedEngine*
 - *ThreadedWebGuiEngine*

- *NginxWebGuiReverseProxy*
- *Webadmin*
 - *AdminIndex*
 - *ExpressionsView*
 - *FileModel*
 - *MyBaseView*
 - *SettingsModel*
 - *SourcesModel*
 - *StatisticsModel*
 - *Tools*
 - *SecurityWebadminView*
 - *SecurityCertificatesView*
 - *Views*

14.1 Abstract baseclass

BaseEngine

This is an abstract baseclass

```

class netdef.Engines.BaseEngine.BaseEngine (shared=None)
    Bases: object

    add_controller_classes (controllers)

    add_rule_classes (rules)

    add_shared_object (shared)

    add_source_classes (sources)

    static block ()

    init ()

    load (base_package)

    start ()

    stop ()

    wait ()

class netdef.Engines.BaseEngine.BaseExpressionExecutor (name, shared)
    Bases: object

    add_interrupt (interrupt)

    add_name (name)

    add_shared (shared)

    handle_run_expression (source_item, expressions, value, source_time, status_code)

```

```
has_interrupt ()
init_queue ()
loop_incoming ()
run ()
```

14.2 Expressions

Expression

```
class netdef.Engines.expression.Expression.Argument (source_instance, instigator,
                                                    frozen_value=None)
```

Bases: object

A wrapper for source instances.

Parameters

- **source_instance** (*BaseSource*) – An source instance
- **instigator** (*boolean*) – True if given source instance triggered the execution

controller

Returns the controller attribute from source instance

create_interface (value=None)

Wrap given value into the source interface. (See interface attr of *netdef.Sources.BaseSource.BaseSource*)

Parameters **value** (*object*) – value to be wrapped

Returns An interface instance

Return type *netdef.Interfaces.DefaultInterface*

get

Returns the value from source instance. NB! this is not a *frozen* copy of the value. It may change if the controller updates the value.

instance

reference to the source instance

key

Returns the key attribute from source instance

new

Returns True if source triggered the expression and this is the first value. (StatusCode.INITIAL)

set

Write a new value to the source. This will trigger a WRITE_SOURCE message to the controller.

status_ok

Returns True if value is StatusCode.GOOD or StatusCode.INITIAL

update

Returns True if source triggered the expression. (StatusCode.GOOD or INVALID)

value

a frozen copy of the value in self.instance.get

```
class netdef.Engines.expression.Expression.Expression (expression, filename)
    Bases: object
```

A class containing a reference to the expression-function and references to the source-instances that will become arguments to the expression function

Parameters

- **expression** (*callable*) – A reference to the actual function
- **filename** (*str*) – Filename of the module where the function is found

```
add_arg (arg)
    arg: this should be a source instance
```

```
add_kwarg (keyword, arg)
    This could be anything. This function exist for you to extend arguments for the expressions. netdef itself do not use this
```

```
disable ()
    If there is problems with the expression it can be automaticly disabled by calling this function
```

```
execute (args, kwargs)
    Execute the expression-function with given arguments
```

```
get_args (source_instance=None, frozen_value=None)
    Wrap each source-instance into its own Argument instance Return a tuple of Arguments
```

```
get_kwargs ()
```

Collector

```
class netdef.Engines.expression.Collector.Collector (fn, wait, mode)
    Bases: object
```

Takes a function but does not call it right away. After the given wait time has elapsed the function is called based on the given mode.

Parameters

- **fn** (*callable*) – a function or callable
- **wait** (*float*) – seconds to wait
- **mode** (*Mode*) – how to call the callable

```
__call__ (*args)
    Add arguments to a queue. Only the first call will acquire self.lock and sleep until wait time has elapsed. After sleep the arguments in queue is used to call the function self.fn based on the chosen mode.
```

```
class netdef.Engines.expression.Collector.Mode
    Bases: enum.Enum
```

collector modes

```
FIRST = 1
    Use arguments from the first call
```

```
FIRST_WITH_EVENT = 4
    Use arguments from the first call and an additional argument called event
```

```
LAST = 2
    Use arguments from the last call
```

LAST_WITH_EVENT = 5

Use arguments from the last call and an additional argument called event

LIST_ALL = 3

Convert arguments to lists with every call

`netdef.Engines.expression.Collector.collect (wait, mode)`

A decorator for expressions.

Usage:

```
from netdef.Engines.expression.Collector import collect, Mode

@collect (wait=0.1, mode=Mode.LIST_ALL)
def expression (c1, c2, c3):
    pass
```

14.3 Built-in engine modules

ThreadedEngine

class `netdef.Engines.ThreadedEngine.ExpressionExecutor (*args, **kwargs)`

Bases: `netdef.Engines.BaseEngine.BaseExpressionExecutor`

handle_run_expression (*source_item, expressions, value, source_time, status_code*)

loop_futures ()

run ()

class `netdef.Engines.ThreadedEngine.ThreadedEngine (shared)`

Bases: `netdef.Engines.BaseEngine.BaseEngine`

static block ()

init ()

load (*base_package*)

start ()

stop ()

wait ()

ThreadedWebGuiEngine

class `netdef.Engines.ThreadedWebGuiEngine.ThreadedWebGuiEngine (shared)`

Bases: `netdef.Engines.ThreadedEngine.ThreadedEngine`

Integrates a simple werkzeug webserver to serve flask_admin webpages

block ()

Run webserver and wait for KeyboardInterrupt

get_flask_app ()

Returns the main flask app.

Common use case is to integrate an existing flask app.

main.py Example:


```

def init_app(app):

    @app.route('/')
    def hello_world():
        return 'Hello, World!'

    return app

def main():
    ...

    engine = ThreadedWebGuiEngine.ThreadedWebGuiEngine(shared)

    # here we go
    init_app(engine.get_flask_app())

    engine.add_controller_classes(controllers)
    engine.add_source_classes(sources)
    engine.add_rule_classes(rules)
    engine.load([__package__, 'netdef'])
    engine.init()
    engine.start()
    engine.block() # until ctrl-c or SIG_TERM
    engine.stop()
    ...

```

init()

load(*base_package*)

netdef.Engines.ThreadedWebGuiEngine.**init_app**(*app, webadmin_views, shared*)
 Configure flask. Setup flask_admin and flask_login

netdef.Engines.ThreadedWebGuiEngine.**make_admin_users_dict**(*config, section*)

NgixWebGuiReverseProxy

class netdef.Engines.NgixWebGuiReverseProxy.**NgixReverseProxy**(*shared*)
 Bases: *netdef.Engines.ThreadedWebGuiEngine.ThreadedWebGuiEngine*

block()

Run webserver and wait for KeyboardInterrupt

14.4 Webadmin

AdminIndex

class netdef.Engines.webadmin.AdminIndex.**LoginForm**(*formdata=None, obj=None, prefix="", data=None, meta=None, **kwargs*)

Bases: *wtforms.form.Form*

get_user()

login = **<UnboundField(StringField, (), {'validators': [*<wtforms.validators.Required***

```

    password = <UnboundField(PasswordField, ()), {'validators': [<wtforms.validators.Requi
    validate_login (field)
class netdef.Engines.webadmin.AdminIndex.MyAdminIndexView (name=None,          cat-
                                                    egor
y=None,
                                                    endpo
int=None,
                                                    url=None,          tem-
                                                    plate='admin/index.html',
                                                    menu_cla
ss_name=None,
                                                    menu_ico
n_type=None,
                                                    menu_ico
n_value=None)

Bases: flask_admin.base.AdminIndexView

command_result_view()

index()

login_view()

logout_view()

restart_view()

restarting = 0

shutdown_view()

shuttingdown = 0

class netdef.Engines.webadmin.AdminIndex.User (userid, roles)
    Bases: flask_login.mixins.UserMixin

    has_role (roles)

netdef.Engines.webadmin.AdminIndex.shutdown_server()

```

ExpressionsView

```

class netdef.Engines.webadmin.ExpressionsView.ExpressionsModel (expression)
    Bases: object

    function_arguments

    function_name

    module_filename

class netdef.Engines.webadmin.ExpressionsView.ExpressionsModelForm (formdata=None,
                                                    obj=None,
                                                    prefix="",
                                                    data=None,
                                                    meta=None,
                                                    **kwargs)

Bases: wtforms.form.Form

function_arguments = <UnboundField(StringField, ('function_arguments',), {})>

function_name = <UnboundField(StringField, ('function_name',), {})>

module_filename = <UnboundField(StringField, ('module_filename',), {})>

```

```

class netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView(model,
                                                                    name=None,
                                                                    cate-
                                                                    gory=None,
                                                                    end-
                                                                    point=None,
                                                                    url=None,
                                                                    static_folder=None,
                                                                    menu_class_name=None,
                                                                    menu_icon_type=None,
                                                                    menu_icon_value=None)

Bases: netdef.Engines.webadmin.MyBaseView.MyBaseView, flask_admin.model.base.
BaseModelView

action_view()
    Mass-model action view.

ajax_lookup()

ajax_update()
    Edits a single column of a record in list view.

can_create = False
can_delete = False
can_edit = False

column_list = ('module_filename', 'function_name', 'function_arguments')
column_searchable_list = ('module_filename', 'function_name', 'function_arguments')
column_sortable_list = ()

create_view()
    Create model view

delete_view()
    Delete model view. Only POST method is allowed.

details_view()
    Details model view

edit_view()
    Edit model view

export(export_type)

form
    alias of ExpressionsModelForm

get_list(page, sort_field, sort_desc, search, filters, page_size=None)
    Return a paginated and sorted list of models from the data source.

    Must be implemented in the child class.

```

Parameters

- **page** – Page number, 0 based. Can be set to None if it is first page.
- **sort_field** – Sort column name or None.
- **sort_desc** – If set to True, sorting is in descending order.
- **search** – Search query

- **filters** – List of filter tuples. First value in a tuple is a search index, second value is a search value.
- **page_size** – Number of results. Defaults to ModelView's page_size. Can be overridden to change the page_size limit. Removing the page_size limit requires setting page_size to 0 or False.

get_pk_value (*model_*)

Return PK value from a model object.

index_view ()

List view

init_search ()

Initialize search. If data provider does not support search, *init_search* will return False.

is_accessible ()

Override this method to add permission checks.

Flask-Admin does not make any assumptions about the authentication system used in your application, so it is up to you to implement it.

By default, it will allow access for everyone.

static sampling (*selection, offset=0, limit=None*)

`netdef.Engines.webadmin.ExpressionsView.setup(admin)`

FileModel

class `netdef.Engines.webadmin.FileModel.Files` (*base_path, *args, **kwargs*)

Bases: `netdef.Engines.webadmin.MyBaseView.MyBaseView`, `flask_admin.contrib.fileadmin.FileAdmin`

action_view ()

allowed_extensions = ('txt', 'conf', 'csv', 'der', 'pam', 'key', 'zip', 'gz', '7z', 'p...

can_download = True

delete ()

Delete view method

download (*path=None*)

Download view method.

Parameters path – File path.

edit ()

Edit view method

edit_template = 'admin/fileedit.html'

editable_extensions = ('txt', 'conf', 'csv', 'py', 'ini', 'yaml')

get_edit_form ()

Create form class for file editing view.

Override to implement customized behavior.

index (*path=None*)

index_view (*path=None*)

Index view method

Parameters path – Optional directory path. If not provided, will use the base directory

is_accessible()

Override this method to add permission checks.

Flask-Admin does not make any assumptions about the authentication system used in your application, so it is up to you to implement it.

By default, it will allow access for everyone.

is_accessible_path(path)

Verify if the provided path is accessible for the current user.

Override to customize behavior.

Parameters path – Relative path to the root

list_template = 'admin/filelist.html'

mkdir(path=None)

Directory creation view method

Parameters path – Optional directory path. If not provided, will use the base directory

rename()

Rename view method

upload(path=None)

Upload view method

Parameters path – Optional directory path. If not provided, will use the base directory

```
class netdef.Engines.webadmin.FileModel.InstallationRepo(base_path, *args,  
**kwargs)
```

Bases: `netdef.Engines.webadmin.MyBaseView.MyBaseView`, `flask_admin.contrib.fileadmin.FileAdmin`

action_view()

allowed_extensions = ('zip', 'whl', 'gz')

can_download = True

can_rename = False

delete()

Delete view method

download(path=None)

Download view method.

Parameters path – File path.

edit()

Edit view method

index(path=None)

index_view(path=None)

Index view method

Parameters path – Optional directory path. If not provided, will use the base directory

is_accessible()

Override this method to add permission checks.

Flask-Admin does not make any assumptions about the authentication system used in your application, so it is up to you to implement it.

By default, it will allow access for everyone.

```
list_template = 'admin/filelist.html'
```

```
mkdir (path=None)
```

Directory creation view method

Parameters path – Optional directory path. If not provided, will use the base directory

```
rename ()
```

Rename view method

```
upload (path=None)
```

Upload view method

Parameters path – Optional directory path. If not provided, will use the base directory

```
netdef.Engines.webadmin.FileModel.setup (admin)
```

MyBaseView

```
class netdef.Engines.webadmin.MyBaseView.MyBaseView (name=None, category=None,  
endpoint=None, url=None,  
static_folder=None,  
static_url_path=None,  
menu_class_name=None,  
menu_icon_type=None,  
menu_icon_value=None)
```

Bases: flask_admin.base.BaseView

```
has_role (roles)
```

```
inaccessible_callback (name, **kwargs)
```

Handle the response to inaccessible views.

By default, it throw HTTP 403 error. Override this method to customize the behaviour.

```
is_accessible ()
```

Override this method to add permission checks.

Flask-Admin does not make any assumptions about the authentication system used in your application, so it is up to you to implement it.

By default, it will allow access for everyone.

SettingsModel

```
class netdef.Engines.webadmin.SettingsModel.SettingsModel (section, key, value)
```

Bases: object

```
key
```

```
section
```

```
value
```

```

class netdef.Engines.webadmin.SettingsModel.SettingsModelForm (formdata=None,
                                                                obj=None,
                                                                prefix="",
                                                                data=None,
                                                                meta=None,
                                                                **kwargs)

Bases: wtforms.form.Form

key = <UnboundField(StringField, ('key',), {})>
section = <UnboundField(StringField, ('section',), {})>
value = <UnboundField(StringField, ('value',), {})>

class netdef.Engines.webadmin.SettingsModel.SettingsModelView (model,
                                                                name=None,
                                                                category=None,
                                                                endpoint=None,
                                                                url=None,
                                                                static_folder=None,
                                                                menu_class_name=None,
                                                                menu_icon_type=None,
                                                                menu_icon_value=None)

Bases: netdef.Engines.webadmin.MyBaseView.MyBaseView, flask_admin.model.base.
BaseModelView

action_view ()
    Mass-model action view.

ajax_lookup ()

ajax_update ()
    Edits a single column of a record in list view.

can_create = False
can_delete = False
can_edit = False
column_list = ('section', 'key', 'value')
column_searchable_list = 'key'
column_sortable_list = ()

create_view ()
    Create model view

delete_view ()
    Delete model view. Only POST method is allowed.

details_view ()
    Details model view

edit_view ()
    Edit model view

export (export_type)

form
    alias of SettingsModelForm

```

get_list (*page, sort_field, sort_desc, search, filters, page_size=None*)
Return a paginated and sorted list of models from the data source.

Must be implemented in the child class.

Parameters

- **page** – Page number, 0 based. Can be set to None if it is first page.
- **sort_field** – Sort column name or None.
- **sort_desc** – If set to True, sorting is in descending order.
- **search** – Search query
- **filters** – List of filter tuples. First value in a tuple is a search index, second value is a search value.
- **page_size** – Number of results. Defaults to ModelView's page_size. Can be overridden to change the page_size limit. Removing the page_size limit requires setting page_size to 0 or False.

get_pk_value (*model_*)
Return PK value from a model object.

index_view ()
List view

init_search ()
Initialize search. If data provider does not support search, *init_search* will return False.

is_accessible ()
Override this method to add permission checks.

Flask-Admin does not make any assumptions about the authentication system used in your application, so it is up to you to implement it.

By default, it will allow access for everyone.

static sampling (*selection, offset=0, limit=None*)

`netdef.Engines.webadmin.SettingsModel.setup(admin)`

SourcesModel

```
class netdef.Engines.webadmin.SourcesModel.SourcesModelForm (formdata=None,  
                                                         obj=None, pre-  
                                                         fix="", data=None,  
                                                         meta=None,  
                                                         **kwargs)
```

Bases: `wtforms.form.Form`

```
key = <UnboundField(StringField, ('key',), {'render_kw': {'readonly': True}})>
```

process (**args, **kwargs*)
Take form, object data, and keyword arg input and have the fields process them.

Parameters

- **formdata** – Used to pass data coming from the enduser, usually `request.POST` or equivalent.
- **obj** – If `formdata` is empty or not provided, this object is checked for attributes matching form field names, which will be used for field values.

- **data** – If provided, must be a dictionary of data. This is only used if `formdata` is empty or not provided and `obj` does not contain an attribute named the same as the field.
- ****kwargs** – If `formdata` is empty or not provided and `obj` does not contain an attribute named the same as a field, form will assign the value of a matching keyword argument to the field, if one exists.

```

set_origin = <UnboundField(StringField, ('set_origin',), {'render_kw': {'readonly': True}})>
set_source_time = <UnboundField(StringField, ('set_source_time',), {'render_kw': {'readonly': True}})>
set_status_code = <UnboundField(StringField, ('set_status_code',), {'render_kw': {'readonly': True}})>
set_value = <UnboundField(StringField, ('set_value',), {'render_kw': {'readonly': True}})>
source = <UnboundField(StringField, ('source',), {'render_kw': {'readonly': True}})>
source_datatype = <UnboundField(StringField, ('source_datatype',), {'render_kw': {'readonly': True}})>
class netdef.Engines.webadmin.SourcesModel.SourcesModelView(model, name=None,
                                                            category=None,
                                                            endpoint=None,
                                                            url=None,
                                                            static_folder=None,
                                                            menu_class_name=None,
                                                            menu_icon_type=None,
                                                            menu_icon_value=None)
    Bases: netdef.Engines.webadmin.MyBaseView.MyBaseView, flask_admin.model.base.BaseModelView
    action_view()
        Mass-model action view.
    ajax_lookup()
    ajax_update()
        Edits a single column of a record in list view.
    can_create = False
    can_delete = False
    can_edit = True
    can_view_details = True
    column_details_list = ('key', 'rule', 'source', 'controller', 'value_as_string', 'status_code')
    column_list = ('key', 'rule', 'source', 'controller', 'value_as_string', 'status_code')
    column_searchable_list = ('key', 'rule', 'source', 'controller', 'value')
    column_sortable_list = ()
    create_view()
        Create model view
    delete_view()
        Delete model view. Only POST method is allowed.
    details_view()
        Details model view
    edit_view()
        Edit model view

```

export (*export_type*)

form

alias of *SourcesModelForm*

get_list (*page, sort_field, sort_desc, search, filters, page_size=None*)

Return a paginated and sorted list of models from the data source.

Must be implemented in the child class.

Parameters

- **page** – Page number, 0 based. Can be set to None if it is first page.
- **sort_field** – Sort column name or None.
- **sort_desc** – If set to True, sorting is in descending order.
- **search** – Search query
- **filters** – List of filter tuples. First value in a tuple is a search index, second value is a search value.
- **page_size** – Number of results. Defaults to ModelView's page_size. Can be overridden to change the page_size limit. Removing the page_size limit requires setting page_size to 0 or False.

get_one (*ref*)

Return one model by its id.

Must be implemented in the child class.

Parameters **id** – Model id

get_pk_value (*model_*)

Return PK value from a model object.

index_view ()

List view

init_search ()

Initialize search. If data provider does not support search, *init_search* will return False.

is_accessible ()

Override this method to add permission checks.

Flask-Admin does not make any assumptions about the authentication system used in your application, so it is up to you to implement it.

By default, it will allow access for everyone.

static sampling (*selection, offset=0, limit=None*)

update_model (*form, model*)

Update model from the form.

Returns True if operation succeeded.

Must be implemented in the child class.

Parameters

- **form** – Form instance
- **model** – Model instance

`netdef.Engines.webadmin.SourcesModel.setup(admin)`

StatisticsModel

```
class netdef.Engines.webadmin.StatisticsModel.StatisticsModel (key, value)
    Bases: object

    key

    value

class netdef.Engines.webadmin.StatisticsModel.StatisticsModelForm (formdata=None,
                                                                    obj=None,
                                                                    prefix="",
                                                                    data=None,
                                                                    meta=None,
                                                                    **kwargs)

    Bases: wtforms.form.Form

    key = <UnboundField(StringField, ('key',), {})>
    value = <UnboundField(StringField, ('value',), {})>

class netdef.Engines.webadmin.StatisticsModel.StatisticsModelView (model,
                                                                    name=None,
                                                                    cate-
                                                                    gory=None,
                                                                    end-
                                                                    point=None,
                                                                    url=None,
                                                                    static_folder=None,
                                                                    menu_class_name=None,
                                                                    menu_icon_type=None,
                                                                    menu_icon_value=None)

    Bases: netdef.Engines.webadmin.SourcesModel.SourcesModelView

    action_view()
        Mass-model action view.

    ajax_lookup()

    ajax_update()
        Edits a single column of a record in list view.

    can_create = False

    can_delete = False

    can_edit = False

    can_view_details = False

    column_list = ('key', 'value')

    column_searchable_list = 'key'

    column_sortable_list = ()

    create_view()
        Create model view

    delete_view()
        Delete model view. Only POST method is allowed.

    details_view()
        Details model view
```

edit_view()

Edit model view

export (*export_type*)

form

alias of *StatisticsModelForm*

get_list (*page, sort_field, sort_desc, search, filters, page_size=None*)

Return a paginated and sorted list of models from the data source.

Must be implemented in the child class.

Parameters

- **page** – Page number, 0 based. Can be set to None if it is first page.
- **sort_field** – Sort column name or None.
- **sort_desc** – If set to True, sorting is in descending order.
- **search** – Search query
- **filters** – List of filter tuples. First value in a tuple is a search index, second value is a search value.
- **page_size** – Number of results. Defaults to ModelView's page_size. Can be overridden to change the page_size limit. Removing the page_size limit requires setting page_size to 0 or False.

get_pk_value (*model_*)

Return PK value from a model object.

index_view()

List view

```
netdef.Engines.webadmin.StatisticsModel.setup(admin)
```

Tools

```
class netdef.Engines.webadmin.Tools.Tools (name=None, category=None,
                                             endpoint=None, url=None,
                                             static_folder=None, static_url_path=None,
                                             menu_class_name=None,
                                             menu_icon_type=None,
                                             menu_icon_value=None)
```

Bases: *netdef.Engines.webadmin.MyBaseView.MyBaseView*

autoupgrade()

autoupgrade_upgrade()

echo()

index()

is_accessible()

Override this method to add permission checks.

Flask-Admin does not make any assumptions about the authentication system used in your application, so it is up to you to implement it.

By default, it will allow access for everyone.

```
logfile()
```

```
netdef.Engines.webadmin.Tools.get_update_cmd(executable, no_index, pre, force_reinstall,  
                                              find_links, trusted_host, minimal_timeout,  
                                              package)
```

```
netdef.Engines.webadmin.Tools.setup(admin, view=None)
```

```
netdef.Engines.webadmin.Tools.stdout_from_terminal(*command, err_msg=None)
```

```
netdef.Engines.webadmin.Tools.stdout_from_terminal_as_generator(*command,  
                                                                err_msg=None,  
                                                                pre="",  
                                                                post="")
```

SecurityWebadminView

```
class netdef.Engines.webadmin.SecurityWebadminView.BasicSecurityForm(formdata=None,  
                                                                    obj=None,  
                                                                    pre-  
                                                                    fix="",  
                                                                    data=None,  
                                                                    meta=None,  
                                                                    **kwargs)
```

Bases: `netdef.Engines.webadmin.SecurityWebadminView.SecurityForm`

```
new_flask_secret = <UnboundField(SelectField, ('Renew session cookie',), {'choices':
```

```
old_password = None
```

```
validate_password = None
```

```
class netdef.Engines.webadmin.SecurityWebadminView.SecurityForm(formdata=None,  
                                                                obj=None,  
                                                                prefix="",  
                                                                data=None,  
                                                                meta=None,  
                                                                **kwargs)
```

Bases: `wtforms.form.Form`

```
confirm = <UnboundField>PasswordField, ('Repeat Password',), {}>
```

```
login = <UnboundField(StringField, ('Login', [<wtforms.validators.Required object>]),
```

```
new_flask_secret = <UnboundField(SelectField, ('Renew session cookie',), {'choices':
```

```
old_password = <UnboundField>PasswordField, ('Current password',), {}>
```

```
password = <UnboundField>PasswordField, ('New Password',), {}>
```

```
ssl_certificate = <UnboundField(SelectField, ('SSL Certificate',), {'default': functools.
```

```
ssl_certificate_key = <UnboundField(SelectField, ('SSL Key',), {'default': functools.
```

```
ssl_on = <UnboundField(SelectField, ('HTTPS On',), {'default': functools.partial(<bou
```

```
update_on = <UnboundField(SelectField, ('Package upgrade',), {'default': functools.pa
```

```
update_pre_release = <UnboundField(SelectField, ('Accept pre-releases',), {'default':
```

```
static validate_old_password(form, field)
```

```
static validate_password(form, field)
```

```

class netdef.Engines.webadmin.SecurityWebadminView.SecurityWebadminView (name=None,
                                                                    cat-
                                                                    e-
                                                                    gory=None,
                                                                    end-
                                                                    point=None,
                                                                    url=None,
                                                                    static_folder=None,
                                                                    static_url_path=None,
                                                                    menu_class_name=None,
                                                                    menu_icon_type=None,
                                                                    menu_icon_value=None)

```

Bases: *netdef.Engines.webadmin.MyBaseView.MyBaseView*

```
choices_crts = [('', 'None')]
```

```
choices_keys = [('', 'None')]
```

```
index()
```

```
is_accessible()
```

Override this method to add permission checks.

Flask-Admin does not make any assumptions about the authentication system used in your application, so it is up to you to implement it.

By default, it will allow access for everyone.

```
setup_conf_secrets_and_https (webadmin_conf, form)
```

```
setup_conf_userdata (webadmin_conf, form)
```

```
setup_form_defaults (form)
```

```
update_usertable (form)
```

```
usertable_is_empty ()
```

```
netdef.Engines.webadmin.SecurityWebadminView.setup (admin, view=None)
```

SecurityCertificatesView

```

class netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesForm (formdata=None,
                                                                    obj=None,
                                                                    pre-
                                                                    fix="",
                                                                    data=None,
                                                                    meta=None,
                                                                    **kwargs)

```

Bases: *wtforms.form.Form*

```
basicConstraints = <UnboundField(StringField, ('basicConstraints',), {'default': 'CA:'>
```

```
cn = <UnboundField(StringField, ('Common name',), {'default': 'build-14281606-project->
```

```
current_password = <UnboundField>PasswordField, ('Current password',), {}>
```

```
days = <UnboundField(IntegerField, ('Days valid',), {'default': 7300}>
```

```
dns_1 = <UnboundField(StringField, ('DNS.1',), {'default': 'build-14281606-project-47>
```

```
dns_2 = <UnboundField(StringField, ('DNS.2',), {'default': '', 'validators': [<wtfo>
```

```

dns_3 = <UnboundField(StringField, ('DNS.3',), {'default': '', 'validators': [<wtfo
extendedKeyUsage = <UnboundField(StringField, ('extendedKeyUsage',), {'default': 'cri
form_opts = <flask_admin.form.FormOpts object>
gen_opcua = <UnboundField(SelectField, ('OpcUa certificate',), {'default': '1', 'choi
gen_webadmin = <UnboundField(SelectField, ('Webadmin certificate',), {'default': '1',
ip_1 = <UnboundField(StringField, ('IP.1',), {'default': '127.0.0.1', 'validators':
ip_2 = <UnboundField(StringField, ('IP.2',), {'default': '172.17.0.2', 'validators':
ip_3 = <UnboundField(StringField, ('IP.3',), {'default': '', 'validators': [<wtform
ip_4 = <UnboundField(StringField, ('IP.4',), {'default': '', 'validators': [<wtform
ip_5 = <UnboundField(StringField, ('IP.5',), {'default': '', 'validators': [<wtform
keyUsage = <UnboundField(StringField, ('keyUsage',), {'default': 'critical, cRLSign,
subjectAltName = <UnboundField(HiddenField, ('subjectAltName:',), {})>
uri_1 = <UnboundField(StringField, ('URI.1',), {'default': <function get_uri>, 'valid
uri_2 = <UnboundField(StringField, ('URI.2',), {'default': '', 'validators': [<wtfo
uri_3 = <UnboundField(StringField, ('URI.3',), {'default': '', 'validators': [<wtfo
static validate_current_password(form, field)

```

```

class netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView(name=None,
cat-
e-
gory=None,
end-
point=None,
url=None,
static_folder=None,
static_url_path=None,
menu_class_name=None,
menu_icon_type=None,
menu_icon_value=None)

```

Bases: *netdef.Engines.webadmin.MyBaseView.MyBaseView*

index()

is_accessible()

Override this method to add permission checks.

Flask-Admin does not make any assumptions about the authentication system used in your application, so it is up to you to implement it.

By default, it will allow access for everyone.

netdef.Engines.webadmin.SecurityCertificatesView.get_uri()

netdef.Engines.webadmin.SecurityCertificatesView.setup(admin, view=None)

Views

class *netdef.Engines.webadmin.Views.Views* (*shared=None*)

Bases: *object*

A collection of all loaded webadmin views

add_shared_object (*shared*)

load (*base_packages*)

setup (*admin*)

netdef.Engines.webadmin.Views.**register** (*name*)

A decorator to register webadmin views. Example:

```
from netdef.Engines.webadmin import Views

@Views.register("NewView")
def setup(admin, view=None):
    if not view:
        view = NewView(name='NewView', endpoint='newview')
    admin.add_view(view)
    ...
```

15 netdef.Interfaces package

- *Abstract base*
 - *Default interface*
- *Internal classes*
 - *Datamessage*
 - *Tick*
- *Built-in Interfaces*
 - *BytestringInterface*
 - *CommTestInterface*
 - *ConcurrentWebRequestInterface*
 - *FloatInterface*
 - *InfluxDBLoggerInterface*
 - *IntegerInterface*
 - *StringInterface*
 - *UnitOfValueInterface*

15.1 Abstract base

Default interface

class netdef.Interfaces.DefaultInterface.**DefaultInterface** (*value*)

Bases: object

Abstract base class

15.2 Internal classes

Datamessage

```
class netdef.Interfaces.datamessage.DataDefinition (key, default, datatype, access, extension)
```

```
    Bases: netdef.Interfaces.datamessage.datamessage.AbstractBase
```

```
    access
```

```
    datatype
```

```
    default
```

```
    extension
```

```
    classmethod from_uri (uri)
```

```
    static is_uri (uri)
```

```
    key
```

```
class netdef.Interfaces.datamessage.DataMessage (key, value, source_time, status_code, origin, extension)
```

```
    Bases: netdef.Interfaces.datamessage.datamessage.AbstractBase
```

```
    extension
```

```
    classmethod from_uri (uri)
```

```
    static is_uri (uri)
```

```
    key
```

```
    origin
```

```
    source_time
```

```
    status_code
```

```
    value
```

Tick

```
class netdef.Interfaces.internal.tick.Tick (controller)
```

```
    Bases: object
```

```
    tick ()
```

```
    timediff ()
```

15.3 Built-in Interfaces

BytestringInterface

```
class netdef.Interfaces.BytestringInterface.ByteStringInterface (value)
```

```
    Bases: netdef.Interfaces.DefaultInterface.DefaultInterface
```

CommTestInterface

```
class netdef.Interfaces.CommTestInterface.CommTestInterface (value)
    Bases: netdef.Interfaces.DefaultInterface.DefaultInterface

    available
    delay

class netdef.Interfaces.CommTestInterface.Value (value)
    Bases: object

    available
    delay
```

ConcurrentWebRequestInterface

```
class netdef.Interfaces.ConcurrentWebRequestInterface.ConcurrentWebRequestInterface (value)
    Bases: netdef.Interfaces.DefaultInterface.DefaultInterface

    available
    data
    delay

class netdef.Interfaces.ConcurrentWebRequestInterface.Value (value)
    Bases: object

    available
    data
    delay
```

FloatInterface

```
class netdef.Interfaces.FloatInterface.FloatInterface (value)
    Bases: netdef.Interfaces.DefaultInterface.DefaultInterface
```

InfluxDBLoggerInterface

```
class netdef.Interfaces.InfluxDBLoggerInterface.InfluxDBLoggerInterface (value)
    Bases: netdef.Interfaces.DefaultInterface.DefaultInterface

class netdef.Interfaces.InfluxDBLoggerInterface.Value (key, source, rule, con-
                                                         troller, value, source_time,
                                                         status_code)

    Bases: tuple

    controller
        Alias for field number 3

    key
        Alias for field number 0

    rule
        Alias for field number 2
```

source
Alias for field number 1

source_time
Alias for field number 5

status_code
Alias for field number 6

value
Alias for field number 4

IntegerInterface

class netdef.Interfaces.IntegerInterface.**IntegerInterface** (*value*)
Bases: *netdef.Interfaces.DefaultInterface.DefaultInterface*

Interface that facilitates bit manipulation in an integer

bit (*offset*)
returns True or False

bits (**offsets*)
Returns True or False List

clearbit (*offset*)
Changes bit in value to False. No return value.

clearbits (**offsets*)
Changes bits in value to False. No return value.

setbit (*offset, bit=True*)
Changing bit in value to True. Can also change to False if bit = False Does not return any value.

setbits (**offsets, bit=True*)
Changing bits in value to True. Can also change to False if bit = False Does not return any value.

StringInterface

class netdef.Interfaces.StringInterface.**StringInterface** (*value*)
Bases: *netdef.Interfaces.DefaultInterface.DefaultInterface*

UnitOfValueInterface

class netdef.Interfaces.UnitOfValueInterface.**ByteUnitInterface** (*value*)
Bases: *netdef.Interfaces.DefaultInterface.DefaultInterface*

get_value_and_unit ()

class netdef.Interfaces.UnitOfValueInterface.**NoUnitInterface** (*value*)
Bases: *netdef.Interfaces.DefaultInterface.DefaultInterface*

get_value_and_unit ()

class netdef.Interfaces.UnitOfValueInterface.**PercentUnitInterface** (*value*)
Bases: *netdef.Interfaces.DefaultInterface.DefaultInterface*

get_value_and_unit ()

netdef.Interfaces.UnitOfValueInterface.**bytes2human** (*n*)

16 netdef.Rules package

- *Rules*
 - *Rules*
 - *utils*
- *Abstract base*
 - *BaseRule*
- *Built-in rule modules*
 - *CSVRule*
 - *InfluxDBLoggerRule*
 - *INIRule*
 - *YAMLRule*

16.1 Rules

Rules

```
class netdef.Rules.Rules.Rules (shared=None)
    Bases: object
    add_shared_object (shared)
    init ()
    load (base_packages)
```

```
netdef.Rules.Rules.register (name, classref=None)
    A decorator to register rules. Example:
```

```
from netdef.Rules import BaseRule, Rules

@Rules.register("NewRuleTemplate")
class NewRuleTemplate(BaseRule.BaseRule):
    def __init__(self, name, shared):
        ...
```

Can also be called as a normal function:

```
from netdef.Rules import BaseRule, Rules

def setup(shared):
    Rules.register("NewRuleTemplate", NewRuleTemplate)

class NewRuleTemplate(BaseRule.BaseRule):
    def __init__(self, name, shared):
        ...
```

Parameters

- **name** (*str*) – Name of the rule class
- **classref** (*object*) – Should be *None* if used as a decorator and a *class* if called as a function

Returns A callable that returns a *class* if used as a decorator and a *class* if called as a normal function

utils

`netdef.Rules.utils.get_module_from_string(mod_str, package, abs_root, location_name, mod_name)`

`netdef.Rules.utils.import_file(abs_pyfile, location_name, mod_name)`

`netdef.Rules.utils.load_entrypoint(entrypoint, package=None)`

16.2 Abstract base

This is an abstract baseclass

BaseRule

class `netdef.Rules.BaseRule.BaseRule` (*name, shared*)

Bases: `object`

Abstract class for rules.

Parameters

- **name** (*str*) – Name to be used in logfiles
- **shared** (`netdef.Shared.Shared`) – a reference to the shared object

add_class_to_controller (*source_name, controller_name=None*)

Sends ADD_PARSER to controls. Controllers will use static functions defined in these classes to decode / encode values etc.

Parameters

- **source_name** (*str*) – source name as string
- **controller_name** (*str*) – controller name as string

add_instance_to_controller (*item_instance*)

Send ADD_SOURCE to controller of given source.

Parameters **item_instance** (`netdef.Sources.BaseSource`) – source instance

add_interrupt (*interrupt*)

Setup the interrupt signal

add_new_expression (*expr_info*)

This function does too many things:

1. Updates `shared.expressions.instances` (indirectly via `self.maintain_searches`)
2. Associate the sources with expressions as arguments
3. Finds sources and sends them to controllers with ADD_SOURCE message

add_new_parser (*source_name*, *controller_name=None*)

It is not always easy for a controller to understand what kind data that a source regards as value. Some controllers do not even know which source to update with data.

Therefore the source classes has static functions that the controller can use to find out these things.

Use this function to add a source class to a controller as a parser.

Parameters

- **source_name** (*str*) – source as string
- **controller_name** (*str*) – controller as string

convert_to_instance (*item_name*, *source_name*, *controller_name*, *rule_name*, *defaultvalue*)

Uses the source name to find the actual source class. Make a instance off the given source class, returns the instance

Parameters

- **item_name** (*str*) – item as string
- **source_name** (*str*) – source as string
- **controller_name** (*str*) – controller as string
- **rule_name** (*str*) – rule as string
- **defaultvalue** – could be anything.

Returns instance of source

get_existing_instance (*source_instance*)

get_expressions (*instance*)

Returns all expression that is associated with the given instance

Returns list or None

static get_module_from_string (*mod_str*, *package=None*, *abs_root=None*, *location_name=None*, *mod_name=None*)

get_ticks ()

handle_run_expression (*incoming*, *value*, *source_time*, *status_code*)

has_existing_instance (*source_instance*)

Returns True if the source we are working on already exists. This is important, because we do not want more than one source instance for each value...

has_interrupt ()

Returns True if the interrupt signal is received

init_queue ()

Setup the message queue and timeout

loop_incoming ()

Get every message from the queue and dispatch the associated handler function

maintain_searches (*source_instance*, *expression*)

Keeps shared.expressions.instances updated

process_ticks ()

rule_name_from_key (*key*, *default_rule_name*)

Check if rule name is valid.

Parameters

- **key** (*str*) – the source key
- **default_rule_name** (*str*) – rule name to use if not found by given key

Returns rule name

Return type str

Raises **ValueError** – if rule does not exists

run ()

Override this function in rule. Example:

```
def run(self):
    self.logger.info("Running")

    while not self.has_interrupt():
        self.loop_incoming() # dispatch handle_* functions

    self.logger.info("Stopped")
```

send_expressions_to_engine (*item_instance, expressions, value, source_time, status_code*)

Send RUN_EXPRESSION to the engine

Parameters

- **item_instance** – the source instance that triggered the expressions
- **expressions** (*list*) – list of expressions

send_ticks ()

setup ()

Implement the following:

1. Open and read a configuration file
2. Create SourceInfo for the sources found in config
3. Create instance of expression found in config
4. Create source instances based on data in SourceInfo
5. Link source instances to expression.
6. Send ADD_SOURCE and ADD_PARSER to controllers

setup_done ()

Update useful statistics

setup_ticks ()

sleep (*seconds*)

” Sleep by waiting for the interrupt. Should be used instead of time.sleep. Override if sleep should be interrupted by even more signals

source_and_controller_from_key (*key, controller=None*)

Check if controller name is valid. Returns a valid (key, controller) tuple

Parameters

- **key** (*str*) – the source key
- **controller** (*str*) – controller name to use if not found by given key

Returns tuple of key and controller

Return type tuple

Raises **ValueError** – if controller does not exists

update_statistics (*namespace, error_count, expression_count, source_count*)

Write useful info to Statistics-singleton

```
class netdef.Rules.BaseRule.ExpressionInfo (module, arguments, func='expression',  
                                         setup='setup')
```

Bases: object

This is a data class that *describes* an expression. The rule shall create an expression based on this *description*

arguments

func

module

setup

```
class netdef.Rules.BaseRule.SourceInfo (typename, key, controller=None, default-  
                                         value=None, setup='setup')
```

Bases: object

This is a data class that *describes* a source. The rule shall create a source instance based on this *description*

controller

defaultvalue

get_setup_func (*instance*)

key

setup

typename

16.3 Built-in rule modules

CSVRule

```
class netdef.Rules.CSVRule.CSVRule (name, shared)
```

Bases: *netdef.Rules.BaseRule.BaseRule*

Tip: Development Status :: 5 - Production/Stable

handle_run_expression (*incoming, value, source_time, status_code*)

run ()

Main loop. Will exit when receiving interrupt signal

setup ()

Parse config files

setup_csv_rule (*name*)

Parse CSV file.

InfluxDBLoggerRule

```
class netdef.Rules.InfluxDBLoggerRule.InfluxDBLoggerRule (name, shared)  
    Bases: netdef.Rules.BaseRule.BaseRule  
  
    handle_run_expression (incoming, value, source_time, status_code)  
  
    run ()  
        Main loop. Will exit when receiving interrupt signal. Calls setup_auto_logging() once at startup  
  
    setup ()  
  
    setup_auto_logging ()  
        Autogenerate logging expressions and sources for every source that is already created by other rules
```

INIRule

```
class netdef.Rules.INIRule.INIRule (name, shared)  
    Bases: netdef.Rules.BaseRule.BaseRule
```

Caution: Development Status :: 4 - Beta
--

```
    handle_run_expression (incoming, value, source_time, status_code)  
  
    run ()  
        Main loop. Will exit when receiving interrupt signal  
  
    setup ()  
        Parse config files  
  
    setup_ini_rule (name, rel_inifile)  
        parse given ini-file
```

YAMLRule

```
class netdef.Rules.YAMLRule.YAMLRule (name, shared)  
    Bases: netdef.Rules.BaseRule.BaseRule
```

Danger: Development Status :: 3 - Alpha
--

```
    handle_run_expression (incoming, value, source_time, status_code)  
  
    run ()  
        Main loop. Will exit when receiving interrupt signal  
  
    setup ()  
        Parse config files  
  
    setup_yaml_rule (name, rel_yamlfile)  
        parse given yaml-file
```

17 netdef.Shared package

- *Internal*
- *Shared*
- *SharedConfig*
- *SharedExpressions*
- *SharedQueues*
- *SharedSources*

17.1 Internal

class netdef.Shared.Internal.Statistics

Bases: object

A singleton class to store statistics as key-value pair. Can be turned off for performance or security.

Can be imported from Rules, Controllers and Expressions.

Example:

```
import psutil
from netdef.Shared.Internal import Statistics
from netdef.Sources.SystemMonitorSource import bytes2human

if Statistics.on:
    uss = psutil.Process().memory_full_info().uss
    Statistics.set("process.memory.startup", bytes2human(uss))
```

static get (*key*)

static get_dict ()

on = True

static set (*key*, *value*)

statistics = {}

17.2 Shared

class netdef.Shared.Shared.Shared (*identifier*, *install_path*, *proj_path*, *default_config_string*)

Bases: object

Shared memory for the application. This is the class of the *shared* instance that is passed to all controllers, rules, engines and expressions. You will use this class to read configs, get message queues etc.

Parameters

- **identifier** (*str*) – a unique identifier for this app.
- **install_path** (*str*) – Full filepath to application package location
- **proj_path** (*str*) – Full filepath to project location

- **default_config_string** (*str*) – initial config text for SharedConfig.Config

17.3 SharedConfig

class netdef.Shared.SharedConfig.**Config** (*identifier*, *install_path*, *proj_path*, *default_config_string*, *read_from_files=True*)

Bases: object

A *wrapper* class for the configparser module in standard python library.

Parameters

- **identifier** (*str*) – a unique identifier for this app.
- **install_path** (*str*) – Full filepath to application package location
- **proj_path** (*str*) – Full filepath to project location
- **default_config_string** (*str*) – initial config text for configparser

add_section (*section*)

config (*section*, *key*, *defaultvalue=None*, *add_if_not_exists=True*)

get_dict (*section*)

get_full_list ()

is_hidden_value (*section*, *key*)

read_default (*config_path*)

set_config (*section*, *key*, *value*)

set_hidden_value (*section*, *key*)

verify (*proj_path*, *config_path*)

17.4 SharedExpressions

class netdef.Shared.SharedExpressions.**ExpressionInstances**

Bases: object

add_expression (*item*)

add_expression_in_source_ref (*ref*, *expression*)

get_expressions_by_source_ref (*ref*)

has_expression_in_source_ref (*ref*, *expression*)

has_source_ref (*ref*)

class netdef.Shared.SharedExpressions.**SharedExpressions**

Bases: object

instances = <netdef.Shared.SharedExpressions.ExpressionInstances object>

17.5 SharedQueues

```
class netdef.Shared.SharedQueues.MessageType
```

```
    Bases: enum.Enum
```

```
    An enumeration.
```

```
    ADD_PARSER = 6
```

```
        Instruct the controller to use the given source class as a parser
```

```
    ADD_SOURCE = 2
```

```
        Instruct the controller to update the given source's value from external datasource
```

```
    APP_STATE = 9
```

```
        Inform the controller of application state
```

```
    READ_ALL = 1
```

```
        warning: Not implemented yet
```

```
    READ_SOURCE = 3
```

```
        warning: Not implemented yet
```

```
    REMOVE_SOURCE = 7
```

```
        warning: Not implemented yet
```

```
    RUN_EXPRESSION = 5
```

```
        Instruct the rule or engine to execute the given expression's function
```

```
    TICK = 8
```

```
        Instruct the controller to send a reply
```

```
    WRITE_SOURCE = 4
```

```
        Instruct the controller to update external datasource from the given source's value
```

```
class netdef.Shared.SharedQueues.AppStateType
```

```
    Bases: enum.Enum
```

```
    An enumeration.
```

```
    RUNNING = 2
```

```
    SETUP = 1
```

```
class netdef.Shared.SharedQueues.SharedQueues (maxsize=0)
```

```
    Bases: object
```

```
    Message queues for all controllers, rules and the engine
```

```
    add_controller (name)
```

```
        Create a incoming queue for given controller'
```

```
    add_rule (name)
```

```
        Create a incoming queue for given rule'
```

```
    get_messages_to_controller (name)
```

```
        Returns the incoming queue for given controller
```

```
    get_messages_to_engine ()
```

```
        Returns the incoming queue for the engine
```

```
    get_messages_to_rule (name)
```

```
        Returns the incoming queue for given rule
```

```
    run_expressions_in_engine (source_instance, expressions, value, source_time, status_code)
```

```
        Send a RUN_EXPRESSION message to the engine.
```

Parameters

- **source_instance** – the source that triggered given expressions
- **expressions** (*list*) – list of expressions

run_expressions_in_rule (*source_instance*)

Send a RUN_EXPRESSION message to given rule.

Parameters **source_instance** – the source

send_message_to_controller (*messagetype, controllername, message_object*)

Send a message to given controller

Parameters

- **messagetype** (*self.MessageType*) –
- **controllername** (*str*) –
- **message_object** – usually a source instance. can also be a tuple.

send_message_to_engine (*messagetype, message_object*)

Send a message to the engine

Parameters

- **messagetype** (*self.MessageType*) – probably Mes-
sageType.RUN_EXPRESSION
- **message_object** – usually a source instance.

send_message_to_rule (*messagetype, rule_name, message_object*)

Send a message to given rule

Parameters

- **messagetype** (*self.MessageType*) –
- **rule_name** (*str*) –
- **message_object** – usually a source instance.

send_running_state_to_controller (*controllername*)

Send a APP_STATE message to given controller

Parameters **controllername** – the controller

send_setup_state_to_controller (*controllername*)

Send a APP_STATE message to given controller

Parameters **controllername** – the controller

write_value_to_controller (*source_instance, value, source_time*)

Send a WRITE_SOURCE message to given controller

Parameters

- **source_instance** – the source
- **value** – new value. datatype have to match the given source
- **source_time** (*datetime.datetime*) – timestamp in utc

17.6 SharedSources

```
class netdef.Shared.SharedSources.SharedSources
```

```
Bases: object
```

```
classes contain a dict (classes.items) with uninitiated sources classes. (key is name from config, value is class) Used by rules when parsing config files and finding the right source.
```

```
instances contains a list of all sources (instances.items) instances created by the rules.
```

```
classes = <netdef.Shared.SharedSources.SourceClasses object>
```

```
instances = <netdef.Shared.SharedSources.SourceInstances object>
```

```
class netdef.Shared.SharedSources.SourceClasses
```

```
Bases: object
```

```
add_item(source_name, classobj)
```

```
get_item(name)
```

```
init_items(items)
```

```
class netdef.Shared.SharedSources.SourceInstances
```

```
Bases: object
```

```
add_item(item)
```

```
get_item_by_ref(ref)
```

```
has_item_ref(ref)
```

18 netdef.Sources package

- *Sources*
- *Abstract base*
 - *BaseSource*
- *Built-in Interfaces*
 - *BytestringSource*
 - *CommTestSource*
 - *ConcurrentWebRequestSource*
 - *CrontabSource*
 - *DictSource*
 - *FloatSource*
 - *HoldingRegisterSource*
 - *InfluxDBLoggerSource*
 - *IntegerSource*
 - *InternalSource*
 - *MQTTDataMessageSource*

- *SubprocessSource*
- *SystemMonitorSource*
- *TextSource*
- *VariantSource*
- *XmlRpcMethodCallSource*
- *ZmqDataAccessSource*

18.1 Sources

```
class netdef.Sources.Sources.Sources (shared=None)
    Bases: object
    add_shared_object (shared)
    init ()
    load (base_packages)
```

netdef.Sources.Sources.**register** (name, classref=None)
A decorator to register sources. Example:

```
from netdef.Sources import BaseSource, Sources

@Sources.register("NewSourceTemplate")
class NewSourceTemplate(BaseSource.BaseSource):
    def __init__(self, name, shared):
        ...
```

Can also be called as a normal function:

```
from netdef.Sources import BaseSource, Sources

def setup(shared):
    Sources.register("NewSourceTemplate", NewSourceTemplate)

class NewSourceTemplate(BaseSource.BaseSource):
    def __init__(self, name, shared):
        ...
```

Parameters

- **name** (*str*) – Name of the source class
- **classref** (*object*) – Should be *None* if used as a decorator and a *class* if called as a function

Returns A callable that returns a *class* if used as a decorator and a *class* if called as a normal function

18.2 Abstract base

BaseSource

class netdef.Sources.BaseSource.**BaseSource** (*key=None, value=None, controller=None, source=None, rule=None*)

Bases: object

can_set_value_from_string ()

Returns True if the value can be converted from string to its given datatype. Only builtins.int, str and float have built-in support, but additional types can be implemented by this function and *set_value_from_string*

static can_unpack_subitems (*value*)

Function that confirms / decides on input data a known list. If so, then unpack_subitems can be used afterwards.

Example:

```
def parse_response(self, response):
    for parser in self.get_parsers():
        if parser.can_unpack_subitems(response):
            yield from parser.unpack_subitems(response)
```

static can_unpack_value (*value*)

Function that confirms / determines if the input data is compatible with this class. If so, unpack_value should be used afterwards.

Example:

```
def parse_item(self, item):
    for parser in self.get_parsers():
        if parser.can_unpack_value(item):
            key, source_time, value = parser.unpack_value(item)
            self.send_datachange(key, source_time, value)
```

copy_get_value ()

Shallow copy of the value

copy_value ()

Shallow copy of the value

get

Get the value that is updated by the controller

get_reference ()

Used to identify similar sources. if two instances return the same reference this means that one instance is redundant and can be replaced

pack_add_source ()

Used if source must be added to external system. I.e. a subscription. Can be overridden and customized.

static pack_subitems (*value*)

Creates output that can be used to query for a list of inputs

pack_value (*value*)

Function that converts key and values into a format that the source uses. Can be overridden and adapted to the controller it is to be used in.

Example:


```
def handle_write_source(self, incoming, value, source_time):
    data = incoming.pack_value(value, source_time)
    topic, payload = incoming.make_message(incoming.key, data)
    self.publish_data_item(topic, payload)
```

register_set_callback (*set_callback*)

Register the callback that sends WRITE_SOURCE message to the controller queue.

set

Get the value that is updated by expressions

set_value_from_string (*value, stime=None, status_ok=True, origin=""*)

Converts given value to correct datatype and sends a WRITE_SOURCE message to controller.

This function is called when a value change is triggered from *Webadmin* → *Sources* → *Edit*

Parameters

- **value** – value to be set
- **or datetime.datetime) stime** (*(None)*) – timestamp when the value was changed
- **status_ok** (*bool*) – True if value is good
- **origin** (*str*) – who set the value

static unpack_subitems (*value*)

Function that parses response from source and yield items found in value. This can be overridden and adapted to the controller it is to be used in.

Example:

```
def parse_response(self, response):
    for parser in self.get_parsers():
        if parser.can_unpack_subitems(response):
            yield from parser.unpack_subitems(response)
```

static unpack_value (*key, source_time, value*)

Function that parses response from source and returns following tuple: (key, source_time, value) Key can then be used to find the right instance and update values.

Can be overridden and adapted to the controller it is to be used in.

Returns tuple(key, source_time, value)

Return type tuple

Example:

```
def parse_item(self, item):
    for parser in self.get_parsers():
        if parser.can_unpack_value(item):
            key, source_time, value = parser.unpack_value(item)
            self.send_datachange(key, source_time, value)
```

value_as_string

Is primarily used by web interfaces to display value in table. Can be overridden to limit the display of large data. Example:

```

@property
def value_as_string(self):
    if self.value and isinstance(self.value, bytes):
        n = len(self.value)
        return "<{}...><data len:{}>".format(self.value[:10], n)
    else:
        return super().value_as_string

```

class netdef.Sources.BaseSource.**StatusCode**

Bases: `enum.Enum`

Used to indicate the quality of a value in `BaseSource.status_code`

NONE: Value is not set yet. INITIAL: First value. you might have to update caches with this value at application startup. GOOD: A normal value update. INVALID: A value update where the value is not to be trusted.

GOOD = 2

INITIAL = 1

INVALID = 3

NONE = 0

18.3 Built-in Interfaces

BytestringSource

class netdef.Sources.BytestringSource.**BytestringSource** (*args, **kwargs)

Bases: `netdef.Sources.BaseSource.BaseSource`

value_as_string
byte data as string

CommTestSource

class netdef.Sources.CommTestSource.**CommTestSource** (*args, **kwargs)

Bases: `netdef.Sources.FloatSource.FloatSource`

unpack_host_and_port ()

ConcurrentWebRequestSource

class netdef.Sources.ConcurrentWebRequestSource.**ConcurrentWebRequestSource** (*args, **kwargs)

Bases: `netdef.Sources.BaseSource.BaseSource`

DEFAULT_CLIENT_SESSION_TIMEOUT = 2

add_client_session (session)

build_url (url)

get_basic_auth ()

get_client_session ()

get_client_session_timeout ()

```

    get_commands_list ()
    get_connect_request ()
    get_poll_request ()
    get_poll_request_interval ()
    get_start_url ()
    has_basic_auth ()
    has_connect_request ()
    has_poll_request ()
    parse_url (url)
class netdef.Sources.ConcurrentWebRequestSource.Request (method, url,
                                                         params=None,
                                                         data=None)
    Bases: object
    data
    method
    params
    url
class netdef.Sources.ConcurrentWebRequestSource.Result (result)
    Bases: object
    result
netdef.Sources.ConcurrentWebRequestSource.setup (shared)

```

CrontabSource

```

class netdef.Sources.CrontabSource.CrontabSource (*args, **kwargs)
    Bases: netdef.Sources.BaseSource.BaseSource

```

DictSource

```

class netdef.Sources.DictSource.DictSource (*args, **kwargs)
    Bases: netdef.Sources.BaseSource.BaseSource

```

FloatSource

```

class netdef.Sources.FloatSource.FloatSource (*args, **kwargs)
    Bases: netdef.Sources.BaseSource.BaseSource

```

HoldingRegisterSource

```

class netdef.Sources.HoldingRegisterSource.HoldingRegisterSource (*args,
                                                                    **kwargs)
    Bases: netdef.Sources.IntegerSource.IntegerSource
    static pack_unit_and_address (unit, address)

```

`unpack_unit_and_address()`

InfluxDBLoggerSource

class `netdef.Sources.InfluxDBLoggerSource.InfluxDBLoggerSource` (**args*,
***kwargs*)

Bases: `netdef.Sources.BaseSource.BaseSource`

A dataholder class to be used with `InfluxDBLoggerController`

get_points (*data*, *source_time*, *status_code*)

Returns a list suitable as argument for `InfluxDBClient.write_points()`

Parameters

- **data** (`InfluxDBLoggerInterface.Value`) – an object with data to store in influxdb
- **source_time** (`datetime.datetime`) – measurement time
- **status_code** (`BaseSource.StatusCode`) – measurement field.status_code

Returns a list of dicts

static make_points (*interface*, *measurement*, *value*, *source_time*, *status_code*)

Make a list suitable as argument for `InfluxDBClient.write_points()`

Parameters

- **interface** (`BaseSource`, `InfluxDBLoggerInterface`) – an object with key, rule, source and controller attrs
- **measurement** (*str*) – influxdb measurement name
- **value** – measurement field.value
- **source_time** (`datetime.datetime`) – measurement time
- **status_code** (`BaseSource.StatusCode`) – measurement field.status_code

Returns a list of dicts

unpack_measurement ()

Returns self.key. Override to change measurement name.

IntegerSource

class `netdef.Sources.IntegerSource.IntegerSource` (**args*, ***kwargs*)

Bases: `netdef.Sources.BaseSource.BaseSource`

InternalSource

class `netdef.Sources.InternalSource.InternalSource` (**args*, ***kwargs*)

Bases: `netdef.Sources.DictSource.DictSource`

static can_unpack_value (*value*)

Function that confirms / determines if the input data is compatible with this class. If so, `unpack_value` should be used afterwards.

Example:

```

def parse_item(self, item):
    for parser in self.get_parsers():
        if parser.can_unpack_value(item):
            key, source_time, value = parser.unpack_value(item)
            self.send_datachange(key, source_time, value)

```

pack_value (*value*)

Function that converts key and values into a format that the source uses. Can be overridden and adapted to the controller it is to be used in.

Example:

```

def handle_write_source(self, incoming, value, source_time):
    data = incoming.pack_value(value, source_time)
    topic, payload = incoming.make_message(incoming.key, data)
    self.publish_data_item(topic, payload)

```

static unpack_value (*value*)

Function that parses response from source and returns following tuple: (key, source_time, value) Key can then be used to find the right instance and update values.

Can be overridden and adapted to the controller it is to be used in.

Returns tuple(key, source_time, value)

Return type tuple

Example:

```

def parse_item(self, item):
    for parser in self.get_parsers():
        if parser.can_unpack_value(item):
            key, source_time, value = parser.unpack_value(item)
            self.send_datachange(key, source_time, value)

```

MQTTDataMessageSource

```

class netdef.Sources.MQTTDataMessageSource.MQTTDataMessageSource(*args,
                                                                    **kwargs)

```

Bases: *netdef.Sources.BaseSource.BaseSource*

static can_unpack_value (*value*)

Check if it is possible to extract a value from the payload

static make_message (*topic, datamessage*)

Wraps given datamessage into a json-payload

Parameters

- **topic** (*str*) – mqtt topic
- **datamessage** (*DataMessage*) – a datamessage object

Returns tuple of topic and json payload

Return type tuple

pack_value (*value, stime, status_code, origin*)

pack the value and stime into a mqtt payload

static parse_message (*topic, payload*)
Parse given json-payload into a datamessage object

Parameters

- **topic** (*str*) – mqtt topic
- **payload** (*str*) – json payload

Returns a DataMessage object

Return type *DataMessage*

static unpack_value (*value*)
Return a tuple with key, time and value from the mqtt payload :param DataMessage value: datamessage
from mqtt payload :returns: tuple(key, source_time, value, origin) :rtype: tuple

SubprocessSource

class netdef.Sources.SubprocessSource.**SubprocessSource** (**args, **kwargs*)

Bases: *netdef.Sources.BaseSource.BaseSource*

DEFAULT_INTERVAL = 10

static can_unpack_subitems (*value*)
Returns False, cannot unpack subitems

get_command_and_args (*args=None*)
Get command and argument to run

get_poll_interval ()

has_initial_poll ()

has_poll_interval ()

parse_stdout_response (*value*)
Implement parsing function

static unpack_subitems (*value*)
Yields None, cannot unpack subitems

netdef.Sources.SubprocessSource.**setup** (*shared*)

SystemMonitorSource

class netdef.Sources.SystemMonitorSource.**SystemMonitorByteSource** (**args, **kwargs*)

Bases: *netdef.Sources.SystemMonitorSource.SystemMonitorSource*

static get_interface ()

class netdef.Sources.SystemMonitorSource.**SystemMonitorPercentSource** (**args, **kwargs*)

Bases: *netdef.Sources.SystemMonitorSource.SystemMonitorSource*

static get_interface ()

class netdef.Sources.SystemMonitorSource.**SystemMonitorSource** (**args, **kwargs*)

Bases: *netdef.Sources.BaseSource.BaseSource*

static get_interface ()

```
get_value_and_unit ()
value_as_string
```

TextSource

```
class netdef.Sources.TextSource.TextSource (*args, **kwargs)
    Bases: netdef.Sources.BaseSource.BaseSource
```

VariantSource

```
class netdef.Sources.VariantSource.VariantSource (*args, **kwargs)
    Bases: netdef.Sources.BaseSource.BaseSource
```

XmlRpcMethodCallSource

```
class netdef.Sources.XmlRpcMethodCallSource.XmlRpcMethodCallSource (*args,
                                                                    **kwargs)
    Bases: netdef.Sources.BaseSource.BaseSource
    static can_unpack_subitems (value)
        Returns False, cannot unpack subitems
    make_rpc_request (value)
    parse_rpc_response (value)
    poll_request ()
    static unpack_subitems (value)
        Yields None, cannot unpack subitems
```

ZmqDataAccessSource

```
class netdef.Sources.ZmqDataAccessSource.ZmqDataAccessSource (*args, **kwargs)
    Bases: netdef.Sources.BaseSource.BaseSource
    pack_address (addr)
    unpack_address ()
```

19 Indices and tables

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

Python Module Index

n

netdef.__main__, 54
netdef.Controllers.BaseAsyncController, 65
netdef.Controllers.BaseController, 62
netdef.Controllers.CommTestController, 66
netdef.Controllers.ConcurrentWebRequestController, 67
netdef.Controllers.Controllers, 62
netdef.Controllers.CrontabController, 68
netdef.Controllers.InfluxDBLoggerController, 68
netdef.Controllers.InternalController, 68
netdef.Controllers.ModbusClientController, 71
netdef.Controllers.ModbusServerController, 73
netdef.Controllers.MQTTDataMessageController, 75
netdef.Controllers.OPCUAClientController, 76
netdef.Controllers.OPCUAServerController, 76
netdef.Controllers.RESTJsonController, 79
netdef.Controllers.SubprocessController, 80
netdef.Controllers.SystemMonitorController, 80
netdef.Controllers.XmlRpcController, 82
netdef.Controllers.ZmqDataAccessController, 84
netdef.Engines.BaseEngine, 85
netdef.Engines.expression.Collector, 87
netdef.Engines.expression.Expression, 86
netdef.Engines.NginxWebGuiReverseProxy, 89
netdef.Engines.ThreadedEngine, 88
netdef.Engines.ThreadedWebGuiEngine, 88
netdef.Engines.webadmin.AdminIndex, 89
netdef.Engines.webadmin.ExpressionsView, 90
netdef.Engines.webadmin.FileModel, 92
netdef.Engines.webadmin.MyBaseView, 94
netdef.Engines.webadmin.SecurityCertificatesView, 102
netdef.Engines.webadmin.SecurityWebadminView, 101
netdef.Engines.webadmin.SettingsModel, 94
netdef.Engines.webadmin.SourcesModel, 96
netdef.Engines.webadmin.StatisticsModel, 99
netdef.Engines.webadmin.Tools, 100
netdef.Engines.webadmin.Views, 103
netdef.Interfaces.ByteStringInterface, 105
netdef.Interfaces.CommTestInterface, 106
netdef.Interfaces.ConcurrentWebRequestInterface, 106
netdef.Interfaces.datamessage, 105
netdef.Interfaces.DefaultInterface, 104
netdef.Interfaces.FloatInterface, 106
netdef.Interfaces.InfluxDBLoggerInterface, 106
netdef.Interfaces.IntegerInterface, 107
netdef.Interfaces.internal.tick, 105
netdef.Interfaces.StringInterface, 107
netdef.Interfaces.UnitOfValueInterface, 107
netdef.Rules.BaseRule, 109
netdef.Rules.CSVRule, 112
netdef.Rules.InfluxDBLoggerRule, 113
netdef.Rules.INIRule, 113
netdef.Rules.Rules, 108
netdef.Rules.utils, 109
netdef.Rules.YAMLRule, 113
netdef.service, 55
netdef.Shared.Internal, 114
netdef.Shared.Shared, 114
netdef.Shared.SharedConfig, 115
netdef.Shared.SharedExpressions, 115
netdef.Shared.SharedQueues, 116
netdef.Shared.SharedSources, 118
netdef.Sources.BaseSource, 120
netdef.Sources.ByteStringSource, 122
netdef.Sources.CommTestSource, 122
netdef.Sources.ConcurrentWebRequestSource, 122
netdef.Sources.CrontabSource, 123
netdef.Sources.DictSource, 123
netdef.Sources.FloatSource, 123
netdef.Sources.HoldingRegisterSource, 123
netdef.Sources.InfluxDBLoggerSource, 124

netdef.Sources.IntegerSource, 124
netdef.Sources.InternalSource, 124
netdef.Sources.MQTTDataMessageSource,
125
netdef.Sources.Sources, 119
netdef.Sources.SubprocessSource, 126
netdef.Sources.SystemMonitorSource, 126
netdef.Sources.TextSource, 127
netdef.Sources.VariantSource, 127
netdef.Sources.XmlRpcMethodCallSource,
127
netdef.Sources.ZmqDataAccessSource, 127
netdef.systemd_service, 58
netdef.testutils, 60
netdef.utils, 59
netdef.windows_service, 56

Index

Symbols

`__call__()` (*netdef.Engines.expression.Collector.Collector* method), 87

A

`access` (*netdef.Interfaces.datamessage.DataDefinition* attribute), 105

`action_view()` (*netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView* method), 91

`action_view()` (*netdef.Engines.webadmin.FileModel.Files* method), 92

`action_view()` (*netdef.Engines.webadmin.FileModel.InstallationRepo* method), 93

`action_view()` (*netdef.Engines.webadmin.SettingsModel.SettingsModelView* method), 95

`action_view()` (*netdef.Engines.webadmin.SourcesModel.SourcesModelView* method), 97

`action_view()` (*netdef.Engines.webadmin.StatisticsModel.StatisticsModelView* method), 99

`activate_session()` (*netdef.Controllers.OPCUAServerController.CustomInternalSession* method), 77

`add()` (*netdef.Controllers.ConcurrentWebRequestController.NextInterval* method), 67

`add()` (*netdef.Controllers.SubprocessController.NextInterval* method), 80

`add_arg()` (*netdef.Engines.expression.Expression.Expression* method), 87

`add_class_to_controller()` (*netdef.Rules.BaseRule.BaseRule* method), 109

`add_client_session()` (*netdef.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource* method), 122

`add_controller()` (*netdef.Shared.SharedQueues.SharedQueues* method), 116

`add_controller_classes()` (*netdef.Engines.BaseEngine.BaseEngine* method), 85

`add_expression()` (*netdef.Shared.SharedExpressions.ExpressionInstances* method), 115

`add_expression_in_source_ref()` (*netdef.Shared.SharedExpressions.ExpressionInstances* method), 115

`add_folder()` (*netdef.Controllers.OPCUAServerController.OPCUAServerController* method), 78

`add_instance_to_controller()` (*netdef.Rules.BaseRule.BaseRule* method), 109

`add_interrupt()` (*netdef.Controllers.BaseController.BaseController* method), 63

`add_interrupt()` (*netdef.Engines.BaseEngine.BaseExpressionExecutor* method), 85

`add_interrupt()` (*netdef.Rules.BaseRule.BaseRule* method), 109

`add_item()` (*netdef.Shared.SharedSources.SourceClasses* method), 118

`add_item()` (*netdef.Shared.SharedSources.SourceInstances* method), 118

`add_kwarg()` (*netdef.Engines.expression.Expression.Expression* method), 87

`add_logger()` (*netdef.Controllers.BaseController.BaseController* method), 63

`add_name()` (*netdef.Engines.BaseEngine.BaseExpressionExecutor* method), 85

`add_new_expression()` (*netdef.Rules.BaseRule.BaseRule* method), 109

`add_new_parser()` (*netdef.Rules.BaseRule.BaseRule* method), 109

`ADD_PARSER` (*netdef.Shared.SharedQueues.MessageType* attribute), 116

`add_parser()` (*netdef.Controllers.BaseController.BaseController* method), 63

`add_rule()` (*netdef.Shared.SharedQueues.SharedQueues* method), 116

`add_rule_classes()` (*netdef.Engines.BaseEngine.BaseEngine* method), 85

`add_section()` (*netdef.Shared.SharedConfig.Config* method), 115

`add_shared()` (*netdef.Engines.BaseEngine.BaseExpressionExecutor* method), 85

`add_shared_object()` (*netdef.Controllers.Controllers.Controllers* method), 62

`add_shared_object()` (*netdef.Engines.BaseEngine.BaseEngine* method), 85

add_shared_object () (netdef.Engines.webadmin.Views.Views method), 104
 add_shared_object () (netdef.Rules.Rules.Rules method), 108
 add_shared_object () (netdef.Sources.Sources.Sources method), 119
 ADD_SOURCE (netdef.Shared.SharedQueues.MessageType attribute), 116
 add_source () (netdef.Controllers.BaseController.BaseController method), 63
 add_source_classes () (netdef.Engines.BaseEngine.BaseEngine method), 85
 add_variablenode () (netdef.Controllers.OPCUAServerController.OPCUAServerController method), 78
 ajax_lookup () (netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView method), 91
 ajax_lookup () (netdef.Engines.webadmin.SettingsModel.SettingsModelView method), 95
 ajax_lookup () (netdef.Engines.webadmin.SourcesModel.SourcesModelView method), 97
 ajax_lookup () (netdef.Engines.webadmin.StatisticsModel.StatisticsModelView method), 99
 ajax_update () (netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView method), 91
 ajax_update () (netdef.Engines.webadmin.SettingsModel.SettingsModelView method), 95
 ajax_update () (netdef.Engines.webadmin.SourcesModel.SourcesModelView method), 97
 ajax_update () (netdef.Engines.webadmin.StatisticsModel.StatisticsModelView method), 99
 allowed_extensions (netdef.Engines.webadmin.FileModel.Files attribute), 92
 allowed_extensions (netdef.Engines.webadmin.FileModel.InstallationRepository attribute), 93
 app_callback (netdef.systemd_service.ApplicationService attribute), 58
 APP_STATE (netdef.Shared.SharedQueues.MessageType attribute), 116
 application (netdef.windows_service.GenericApplicationService attribute), 56
 ApplicationService (class in netdef.systemd_service), 58
 AppStateType (class in netdef.Shared.SharedQueues), 116
 args (netdef.Controllers.SystemMonitorController.DataItem attribute), 80
 Argument (class in netdef.Engines.expression.Expression), 86
 arguments (netdef.Rules.BaseRule.ExpressionInfo attribute), 112
 assert_any_call () (netdef.testutils.MockSource method), 60
 assert_called () (netdef.testutils.MockSource method), 60
 assert_called_once () (netdef.testutils.MockSource method), 60
 assert_called_once_with () (netdef.testutils.MockSource method), 60
 assert_called_with () (netdef.testutils.MockSource method), 61
 assert_not_called () (netdef.testutils.MockSource method), 61
 assert_value () (netdef.testutils.MockSource method), 61
 auto_upgrade () (netdef.Engines.webadmin.Tools.Tools method), 100
 auto_upgrade_upgrade () (netdef.Engines.webadmin.Tools.Tools method), 100
 auto_upgrade_upgrade () (netdef.Interfaces.CommTestInterface.CommTestInterface attribute), 106
 available (netdef.Interfaces.CommTestInterface.Value attribute), 106
 available (netdef.Interfaces.ConcurrentWebRequestInterface.ConcurrentWebRequestInterface attribute), 106
 available (netdef.Interfaces.ConcurrentWebRequestInterface.Value attribute), 106
 BaseAsyncController (class in netdef.Controllers.BaseAsyncController), 65
 BaseController (class in netdef.Controllers.BaseController), 62
 BaseEngine (class in netdef.Engines.BaseEngine), 85
 BaseExpressionExecutor (class in netdef.Engines.BaseEngine), 85
 BaseRule (class in netdef.Rules.BaseRule), 109
 BaseSource (class in netdef.Sources.BaseSource), 120
 basicConstraints (netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView attribute), 102

BasicSecurityForm (class in net-def.Engines.webadmin.SecurityWebadminView), 101
 can_download (net-def.Engines.webadmin.FileModel.InstallationRepo attribute), 93
 bit () (netdef.Interfaces.IntegerInterface.IntegerInterface method), 107
 can_edit (netdef.Engines.webadmin.ExpressionsView.ExpressionsModel attribute), 91
 bits () (netdef.Interfaces.IntegerInterface.IntegerInterface method), 107
 can_edit (netdef.Engines.webadmin.SettingsModel.SettingsModelView attribute), 95
 block () (netdef.Engines.BaseEngine.BaseEngine static method), 85
 can_edit (netdef.Engines.webadmin.SourcesModel.SourcesModelView attribute), 97
 block () (netdef.Engines.NginxWebGuiReverseProxy.NginxReverseProxy method), 89
 can_view_details (netdef.Engines.webadmin.StatisticsModel.StatisticsModelView attribute), 99
 block () (netdef.Engines.ThreadedEngine.ThreadedEngine static method), 88
 can_rename (netdef.Engines.webadmin.FileModel.InstallationRepo attribute), 93
 block () (netdef.Engines.ThreadedWebGuiEngine.ThreadedWebGuiEngine method), 88
 can_value_from_string () (net-def.Sources.BaseSource.BaseSource method), 120
 build_folders () (net-def.Controllers.OPCUAServerController.OPCUAServerController method), 78
 can_subitems () (net-def.Sources.BaseSource.BaseSource static method), 120
 build_url () (netdef.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource method), 122
 can_unpack_subitems () (net-def.Sources.SubprocessSource.SubprocessSource static method), 126
 bytes2human () (in module net-def.Interfaces.UnitOfValueInterface), 107
 can_unpack_subitems () (net-def.Sources.XmlRpcMethodCallSource.XmlRpcMethodCallSource static method), 127
 ByteStringInterface (class in net-def.Interfaces.ByteStringInterface), 105
 can_unpack_value () (net-def.Sources.BaseSource.BaseSource static method), 120
 ByteStringSource (class in net-def.Sources.ByteStringSource), 122
 can_unpack_value () (net-def.Sources.InternalSource.InternalSource static method), 124
 ByteUnitInterface (class in net-def.Interfaces.UnitOfValueInterface), 107
 can_unpack_value () (net-def.Sources.MQTTDataMessageSource.MQTTDataMessageSource static method), 125
 C
 call_args (netdef.testutils.MockSource attribute), 61
 can_view_details (netdef.Engines.webadmin.StatisticsModel.StatisticsModelView attribute), 99
 call_args_list (netdef.testutils.MockSource attribute), 61
 can_delete (netdef.Engines.webadmin.StatisticsModel.StatisticsModelView attribute), 102
 call_count (netdef.testutils.MockSource attribute), 61
 can_create (netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView attribute), 91
 can_create (netdef.Engines.webadmin.SourcesModel.SourcesModelView attribute), 97
 can_create (netdef.Engines.webadmin.StatisticsModel.StatisticsModelView attribute), 99
 can_delete (netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView attribute), 91
 choices_keys (net-def.Engines.webadmin.SecurityWebadminView.SecurityWebadminView attribute), 102
 can_delete (netdef.Engines.webadmin.SettingsModel.SettingsModelView attribute), 95
 can_delete (netdef.Engines.webadmin.SourcesModel.SourcesModelView attribute), 97
 can_delete (netdef.Engines.webadmin.StatisticsModel.StatisticsModelView attribute), 99
 can_delete (netdef.Engines.webadmin.StatisticsModel.StatisticsModelView attribute), 99
 can_download (net-def.Engines.webadmin.FileModel.Files attribute), 92
 clearbit () (netdef.Interfaces.IntegerInterface.IntegerInterface method), 107
 clearbits () (netdef.Interfaces.IntegerInterface.IntegerInterface method), 107
 classes (netdef.Shared.SharedSources.SharedSources attribute), 118
 clear_moving () (net-def.Controllers.BaseController.BaseController method), 63

method), 107
 cli () (in module netdef.__main__), 54
 cn (netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesViewRequestInterface (class in net-
 attribute), 102
 collect () (in module net-
 def.Engines.expression.Collector), 88
 Collector (class in net-
 def.Engines.expression.Collector), 87
 column_details_list (net-
 def.Engines.webadmin.SourcesModel.SourcesModelView (class in netdef.Shared.SharedConfig), 115
 attribute), 97
 column_list (netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView (netdef.Shared.SharedConfig.Config
 attribute), 91
 column_list (netdef.Engines.webadmin.SettingsModel.SettingsModelView (netdef.Engines.webadmin.SecurityWebadminView.SecurityForm
 attribute), 95
 column_list (netdef.Engines.webadmin.SourcesModel.SourcesModelView (netdef.Controllers.RESTJsonController.RESTJsonController
 attribute), 97
 column_list (netdef.Engines.webadmin.StatisticsModel.StatisticsModelView (netdef.Controllers.ZmqDataAccessController.ZmqDataAcces
 attribute), 99
 column_searchable_list (net-
 def.Engines.webadmin.ExpressionsView.ExpressionsModelView (netdef.Engines.expression.Expression.Argument
 attribute), 91
 column_searchable_list (net-
 def.Engines.webadmin.SettingsModel.SettingsModelView (netdef.Interfaces.InfluxDBLoggerInterface.Value
 attribute), 95
 column_searchable_list (net-
 def.Engines.webadmin.SourcesModel.SourcesModelView (netdef.Rules.BaseRule.SourceInfo
 attribute), 97
 column_searchable_list (net-
 def.Engines.webadmin.StatisticsModel.StatisticsModelView (netdef.Controllers (class in netdef.Controllers.Controllers),
 attribute), 99
 column_sortable_list (net-
 def.Engines.webadmin.ExpressionsView.ExpressionsModelView (netdef.Rules.BaseRule.BaseRule method), 110
 attribute), 91
 column_sortable_list (net-
 def.Engines.webadmin.SettingsModel.SettingsModelView (netdef.Sources.BaseSource.BaseSource method),
 attribute), 95
 column_sortable_list (net-
 def.Engines.webadmin.SourcesModel.SourcesModelView (netdef.Controllers (class in netdef.Controllers.Controllers),
 attribute), 97
 column_sortable_list (net-
 def.Engines.webadmin.StatisticsModel.StatisticsModelView (netdef.Controllers.OPCUAServerController.OPCUAServerControlle
 attribute), 99
 command_result_view () (net-
 def.Engines.webadmin.AdminIndex.MyAdminIndexView (netdef.Controllers.OPCUAServerController.OPCUAServerControlle
 method), 90
 method), 90
 CommTestController (class in net-
 def.Controllers.CommTestController), 66
 CommTestInterface (class in net-
 def.Interfaces.CommTestInterface), 106
 CommTestSource (class in net-
 def.Sources.CommTestSource), 122
 ConcurrentWebRequestController (class in net-
 def.Controllers.ConcurrentWebRequestController), 67
 ConcurrentWebRequestInterface (class in net-
 def.Interfaces.ConcurrentWebRequestInterface), 106
 ConcurrentWebRequestSource (class in net-
 def.Sources.ConcurrentWebRequestSource), 122
 Config (class in netdef.Shared.SharedConfig), 115
 Config () (netdef.Controllers.OPCUAClientController.OPCUAClientCon
 method), 76
 Config (class in netdef.Shared.SharedConfig), 115
 Config () (netdef.Controllers.OPCUAClientController.OPCUAClientCon
 method), 115
 SecurityWebadminView (class in netdef.Engines.webadmin.SecurityWebadminView.SecurityForm
 attribute), 101
 RESTJsonController (class in netdef.Controllers.RESTJsonController.RESTJsonController
 method), 79
 ZmqDataAccessController (class in netdef.Controllers.ZmqDataAccessController.ZmqDataAcces
 method), 84
 controller, 6
 controller (netdef.Engines.expression.Expression.Argument
 attribute), 86
 controller (netdef.Interfaces.InfluxDBLoggerInterface.Value
 attribute), 106
 controller (netdef.Rules.BaseRule.SourceInfo
 attribute), 112
 controllers (class in netdef.Controllers.Controllers),
 62
 convert_to_instance () (net-
 def.Rules.BaseRule.BaseRule method), 110
 copy_get_value () (net-
 def.Sources.BaseSource.BaseSource method),
 120
 copy_value () (net-
 def.Sources.BaseSource.BaseSource method),
 120
 create_datavalue () (net-
 def.Controllers.OPCUAServerController.OPCUAServerControlle
 method), 78
 create_interface () (net-
 def.Engines.expression.Expression.Argument
 method), 86
 create_monitored_items () (net-
 def.Controllers.OPCUAServerController.OPCUAServerControlle
 method), 78
 create_project () (in module netdef.__main__), 54
 create_view () (net-
 def.Engines.webadmin.ExpressionsView.ExpressionsModelView
 method), 91
 create_view () (net-
 def.Engines.webadmin.SettingsModel.SettingsModelView
 method), 95
 create_view () (net-
 def.Engines.webadmin.SourcesModel.SourcesModelView

method), 97

create_view() (netdef.Engines.webadmin.StatisticsModel.StatisticsModelView attribute), 126

method), 99

CrontabController (class in netdef.Controllers.CrontabController), 68

CrontabSource (class in netdef.Sources.CrontabSource), 123

CSVRule (class in netdef.Rules.CSVRule), 112

current_password (netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView attribute), 102

CustomAnonInternalSession (class in netdef.Controllers.OPCUAServerController), 76

CustomInternalSession (class in netdef.Controllers.OPCUAServerController), 76

CustomServer (class in netdef.Controllers.OPCUAServerController), 77

D

daemon_threads (netdef.Controllers.ModbusServerController.MyController attribute), 75

data (netdef.Interfaces.ConcurrentWebRequestInterface.ConcurrentWebRequestInterface attribute), 106

data (netdef.Interfaces.ConcurrentWebRequestInterface.Value attribute), 106

data (netdef.Sources.ConcurrentWebRequestSource.Request attribute), 123

datachange_notification() (netdef.Controllers.OPCUAClientController.SubHandler method), 76

datachange_notification() (netdef.Controllers.OPCUAServerController.SubHandler method), 79

DataDefinition (class in netdef.Interfaces.datamessage), 105

DataItem (class in netdef.Controllers.SystemMonitorController), 80

DataMessage (class in netdef.Interfaces.datamessage), 105

datatype (netdef.Interfaces.datamessage.DataDefinition attribute), 105

days (netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView attribute), 102

default (netdef.Interfaces.datamessage.DataDefinition attribute), 105

DEFAULT_CLIENT_SESSION_TIMEOUT (netdef.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource attribute), 122

DEFAULT_INTERVAL (netdef.Sources.SubprocessSource.SubprocessSource attribute), 126

DefaultInterface (class in netdef.Interfaces.DefaultInterface), 104

defaultvalue (netdef.Rules.BaseRule.SourceInfo attribute), 112

delay (netdef.Interfaces.CommTestInterface.CommTestInterface attribute), 106

delay (netdef.Interfaces.CommTestInterface.Value attribute), 106

delay (netdef.Interfaces.ConcurrentWebRequestInterface.ConcurrentWebRequestInterface attribute), 106

delay (netdef.Interfaces.ConcurrentWebRequestInterface.Value attribute), 106

delete() (netdef.Engines.webadmin.FileModel.Files method), 92

delete() (netdef.Engines.webadmin.FileModel.InstallationRepo method), 93

delete_view() (netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView method), 91

delete_view() (netdef.Engines.webadmin.SettingsModel.SettingsModelView method), 95

delete_view() (netdef.Engines.webadmin.SourcesModel.SourcesModelView method), 97

delete_view() (netdef.Engines.webadmin.StatisticsModel.StatisticsModelView method), 99

details_view() (netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView method), 91

details_view() (netdef.Engines.webadmin.SettingsModel.SettingsModelView method), 95

details_view() (netdef.Engines.webadmin.SourcesModel.SourcesModelView method), 97

details_view() (netdef.Engines.webadmin.StatisticsModel.StatisticsModelView method), 99

DictSource (class in netdef.Sources.DictSource), 123

disable() (netdef.Engines.expression.Expression.Expression method), 87

dns_1 (netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView attribute), 102

dns_2 (netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView attribute), 102

dns_3 (netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView attribute), 102

doView() (netdef.Engines.webadmin.FileModel.Files method), 92

download() (*netdef.Engines.webadmin.FileModel.InstallationRepo* attribute), 93

E

ExpressionsModel (class in *netdef.Engines.webadmin.ExpressionsView*), 90

ExpressionsModelForm (class in *netdef.Engines.webadmin.ExpressionsView*), 90

echo() (*netdef.Engines.webadmin.Tools.Tools* method), 100

edit() (*netdef.Engines.webadmin.FileModel.Files* attribute), 92

edit() (*netdef.Engines.webadmin.FileModel.InstallationRepo* attribute), 93

edit_template (*netdef.Engines.webadmin.FileModel.Files* attribute), 92

edit_view() (*netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView* attribute), 91

edit_view() (*netdef.Engines.webadmin.SettingsModel.SettingsModelView* attribute), 95

edit_view() (*netdef.Engines.webadmin.SourcesModel.SourcesModelView* attribute), 97

edit_view() (*netdef.Engines.webadmin.StatisticsModel.StatisticsModelView* attribute), 99

editable_extensions (*netdef.Engines.webadmin.FileModel.Files* attribute), 92

engine, 5

entrypoint() (in module *netdef.__main__*), 54

event_notification() (*netdef.Controllers.OPCUAClientController.SubHandler* method), 76

event_notification() (*netdef.Controllers.OPCUAServerController.SubHandler* method), 79

exe_name (*netdef.systemd_service.ApplicationService* attribute), 58

execute() (*netdef.Engines.expression.Expression.ExpressionForm* attribute), 87

export() (*netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView* attribute), 91

export() (*netdef.Engines.webadmin.SettingsModel.SettingsModelView* attribute), 95

export() (*netdef.Engines.webadmin.SourcesModel.SourcesModelView* attribute), 97

export() (*netdef.Engines.webadmin.StatisticsModel.StatisticsModelView* attribute), 100

export() (*netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView* attribute), 103

expression, 7

Expression (class in *netdef.Engines.expression.Expression*), 86

ExpressionExecutor (class in *netdef.Engines.ThreadedEngine*), 88

ExpressionInfo (class in *netdef.Rules.BaseRule*), 112

ExpressionInstances (class in *netdef.Shared.SharedExpressions*), 115

extendedKeyUsage (*netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView* attribute), 103

extension (*netdef.Interfaces.datamessage.DataDefinition* attribute), 105

extension (*netdef.Interfaces.datamessage.DataMessage* attribute), 105

F

fetch_one_incoming() (*netdef.Controllers.BaseController.BaseController* method), 63

Files (class in *netdef.Engines.webadmin.FileModel*), 92

FIRST (*netdef.Engines.expression.Collector.Mode* attribute), 87

FIRST_WITH_EVENT (*netdef.Engines.expression.Collector.Mode* attribute), 87

FloatInterface (class in *netdef.Interfaces.FloatInterface*), 106

FloatSource (class in *netdef.Sources.FloatSource*), 123

form (*netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView* attribute), 91

form (*netdef.Engines.webadmin.SettingsModel.SettingsModelView* attribute), 95

form (*netdef.Engines.webadmin.SourcesModel.SourcesModelView* attribute), 98

form (*netdef.Engines.webadmin.StatisticsModel.StatisticsModelView* attribute), 100

form (*netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView* attribute), 103

from_uri() (*netdef.Interfaces.datamessage.DataDefinition* class method), 105

from_uri() (*netdef.Interfaces.datamessage.DataMessage* class method), 105

func (*netdef.Controllers.SystemMonitorController.DataItem* attribute), 80

func (*netdef.Rules.BaseRule.ExpressionInfo* attribute), 112

function_arguments (*netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView* attribute), 105

<i>attribute</i>), 90	get_commands_list ()	(<i>net-</i>
function_arguments	(<i>net-</i>	<i>def.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource</i>
<i>def.Engines.webadmin.ExpressionsView.ExpressionsModelForm</i>	<i>method</i>), 122	
<i>attribute</i>), 90	get_connect_request ()	(<i>net-</i>
function_name	(<i>net-</i>	<i>def.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource</i>
<i>def.Engines.webadmin.ExpressionsView.ExpressionsModelForm</i>	<i>method</i>), 123	
<i>attribute</i>), 90	get_data_items_dict ()	(<i>in module net-</i>
function_name	(<i>net-</i>	<i>def.Controllers.SystemMonitorController</i>),
<i>def.Engines.webadmin.ExpressionsView.ExpressionsModelForm</i>	<i>method</i>	
<i>attribute</i>), 90	get_default_value ()	(<i>net-</i>
	<i>def.Controllers.OPCUAServerController.OPCUAServerController</i>	
	<i>method</i>), 78	
G		
gen_opcua (<i>netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesForm</i>	<i>static</i>	
<i>attribute</i>), 103	<i>method</i>), 114	
gen_webadmin	(<i>net-</i>	<i>def.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesForm</i>
<i>attribute</i>), 103	<i>method</i>), 115	
generate_certificate ()	(<i>in module net-</i>	<i>def.Engines.webadmin.FileModel.Files</i>
<i>def.__main__</i>), 55	<i>method</i>), 92	
generate_webadmin_auth ()	(<i>in module net-</i>	<i>def.Controllers.BaseAsyncController.BaseAsyncController</i>
<i>def.__main__</i>), 55	<i>method</i>), 65	
GenericApplicationService	(<i>class in net-</i>	<i>def.Rules.BaseRule.BaseRule</i>
<i>def.windows_service</i>), 56	<i>method</i>), 110	
get (<i>netdef.Engines.expression.Expression.Argument</i>	<i>attribute</i>), 86	
get (<i>netdef.Sources.BaseSource.BaseSource</i>	<i>attribute</i>), 120	
get ()	(<i>netdef.Shared.Internal.Statistics</i>	<i>static method</i>), 114
get_args ()	(<i>netdef.Engines.expression.Expression.Expression</i>	<i>method</i>), 87
get_basic_auth ()	(<i>net-</i>	<i>def.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource</i>
<i>method</i>), 122	<i>method</i>), 74	
get_cache_filename ()	(<i>net-</i>	<i>def.Controllers.InternalController.InternalController</i>
<i>method</i>), 70	<i>method</i>), 115	
get_callbacks ()	(<i>netdef.testutils.MockExpression</i>	<i>method</i>), 60
get_clean_mount_point_name ()	(<i>in module net-</i>	<i>def.Controllers.SystemMonitorController</i>), 81
get_client_session ()	(<i>net-</i>	<i>def.Controllers.ConcurrentWebRequestController.ConcurrentWebRequestController</i>
<i>method</i>), 67	<i>static method</i>), 126	
get_client_session ()	(<i>net-</i>	<i>def.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource</i>
<i>method</i>), 122	<i>static method</i>), 126	
get_client_session_timeout ()	(<i>net-</i>	<i>def.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource</i>
<i>method</i>), 122	<i>method</i>), 118	
get_command_and_args ()	(<i>net-</i>	<i>def.Shared.SharedSources.SourceInstances</i>
<i>def.Sources.SubprocessSource.SubprocessSource</i>	<i>method</i>), 118	
<i>method</i>), 126	get_key ()	(<i>netdef.Controllers.MQTTDataMessageController.MQTTDataMessageController</i>
	<i>method</i>), 75	

`get_kwargs()` (*netdef.Engines.expression.Expression.Expression* method), 87
`get_list()` (*netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView* method), 91
`get_list()` (*netdef.Engines.webadmin.SettingsModel.SettingsModelView* method), 95
`get_list()` (*netdef.Engines.webadmin.SourcesModel.SourcesModelView* method), 98
`get_list()` (*netdef.Engines.webadmin.StatisticsModel.StatisticsModelView* method), 100
`get_messages_to_controller()` (*netdef.Shared.SharedQueues.SharedQueues* method), 116
`get_messages_to_engine()` (*netdef.Shared.SharedQueues.SharedQueues* method), 116
`get_messages_to_rule()` (*netdef.Shared.SharedQueues.SharedQueues* method), 116
`get_modbus_server_context()` (*netdef.Controllers.ModbusServerController.ModbusServerController* method), 74
`get_module()` (*netdef.testutils.MockExpression* method), 60
`get_module_from_string()` (in module *netdef.Rules.utils*), 109
`get_module_from_string()` (*netdef.Rules.BaseRule.BaseRule* static method), 110
`get_nodeid()` (*netdef.Controllers.OPCUAServerController.OPCUAServerController* method), 79
`get_one()` (*netdef.Engines.webadmin.SourcesModel.SourcesModelView* method), 98
`get_parsers()` (*netdef.Controllers.BaseController.BaseController* method), 63
`get_pk_value()` (*netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView* method), 92
`get_pk_value()` (*netdef.Engines.webadmin.SettingsModel.SettingsModelView* method), 96
`get_pk_value()` (*netdef.Engines.webadmin.SourcesModel.SourcesModelView* method), 98
`get_pk_value()` (*netdef.Engines.webadmin.StatisticsModel.StatisticsModelView* method), 100
`get_points()` (*netdef.Sources.InfluxDBLoggerSource.InfluxDBLoggerSource* method), 124
`get_poll_interval()` (*netdef.Sources.SubprocessSource.SubprocessSource* method), 126
`get_poll_request()` (*netdef.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource* method), 123
`get_poll_request_interval()` (*netdef.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource* method), 123
`get_proc()` (in module *netdef.Controllers.SystemMonitorController*), 81
`get_reference()` (*netdef.Sources.BaseSource.BaseSource* method), 120
`get_service()` (in module *netdef.service*), 55
`get_service()` (in module *netdef.systemd_service*), 58
`get_service()` (in module *netdef.windows_service*), 56
`get_setup_func()` (*netdef.Rules.BaseRule.SourceInfo* method), 120
`get_source()` (*netdef.Controllers.BaseController.BaseController* method), 63
`get_sources()` (*netdef.Controllers.BaseController.BaseController* method), 63
`get_start_url()` (*netdef.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource* method), 123
`get_status()` (*netdef.Rules.BaseRule.BaseRule* method), 110
`get_ttl()` (*netdef.Controllers.MQTTDataMessageController.MQTTDataMessageController* method), 75
`get_update_cmd()` (in module *netdef.Engines.webadmin.Tools*), 101
`get_uri()` (in module *netdef.Engines.webadmin.SecurityCertificatesView*), 103
`get_user()` (*netdef.Engines.webadmin.AdminIndex.LoginForm* method), 89
`get_value()` (*netdef.Controllers.SystemMonitorController.DataItem* method), 80
`get_value_and_unit()` (*netdef.Interfaces.UnitOfValueInterface.ByteUnitInterface* method), 107
`get_value_and_unit()` (*netdef.Interfaces.UnitOfValueInterface.NoUnitInterface* method), 107
`get_value_and_unit()` (*netdef.Interfaces.UnitOfValueInterface.PercentUnitInterface* method), 107
`get_value_and_unit()` (*netdef.Interfaces.UnitOfValueInterface.PercentUnitInterface* method), 107

def.Sources.SystemMonitorSource.SystemMonitorSource.handle_app_state() (*net-*
method), 126 *def.Controllers.BaseController.BaseController*
get_varianttype() (*net-* *method*), 64
def.Controllers.OPCUAServerController.OPCUAServerController.state_running() (*net-*
method), 79 *def.Controllers.BaseAsyncController.BaseAsyncController*
get_vm() (*in module net-* *method*), 65
def.Controllers.SystemMonitorController), *handle_app_state_running()* (*net-*
82 *def.Controllers.BaseController.BaseController*
GOOD (*netdef.Sources.BaseSource.StatusCode* attribute), *method*), 64
122 *handle_app_state_setup()* (*net-*
def.Controllers.BaseController.BaseController
method), 64
H
handle_add_parser() (*net-* *handle_datachange()* (*net-*
def.Controllers.BaseController.BaseController *def.Controllers.ModbusServerController.ModbusServerController*
method), 63 *method*), 74
handle_add_source() (*net-* *handle_read_source()* (*net-*
def.Controllers.BaseController.BaseController *def.Controllers.BaseController.BaseController*
method), 64 *method*), 64
handle_add_source() (*net-* *handle_read_source()* (*net-*
def.Controllers.ConcurrentWebRequestController.ConcurrentWebRequestController.RESTJsonController.RESTJsonController
method), 67 *method*), 79
handle_add_source() (*net-* *handle_read_source()* (*net-*
def.Controllers.InternalController.InternalController *def.Controllers.XmlRpcController.XmlRpcController*
method), 71 *method*), 84
handle_add_source() (*net-* *handle_readall()* (*net-*
def.Controllers.ModbusClientController.ModbusClientController *def.Controllers.BaseController.BaseController*
method), 72 *method*), 64
handle_add_source() (*net-* *handle_readall()* (*net-*
def.Controllers.ModbusServerController.ModbusServerController *def.Controllers.RESTJsonController.RESTJsonController*
method), 74 *method*), 79
handle_add_source() (*net-* *handle_readall()* (*net-*
def.Controllers.MQTTDataMessageController.MQTTDataMessageController.XmlRpcController.XmlRpcController
method), 75 *method*), 84
handle_add_source() (*net-* *handle_restart()* (*in module netdef.utils*), 59
def.Controllers.OPCUAClientController.OPCUAClientController.expression() (*net-*
method), 76 *def.Engines.BaseEngine.BaseExpressionExecutor*
method), 85
handle_add_source() (*net-* *expression()* (*net-*
def.Controllers.OPCUAServerController.OPCUAServerController *def.Engines.ThreadedEngine.ExpressionExecutor*
method), 79 *method*), 88
handle_add_source() (*net-* *method*), 88
def.Controllers.RESTJsonController.RESTJsonController *handle_run_expression()* (*net-*
method), 79 *def.Rules.BaseRule.BaseRule* *method*), 110
handle_add_source() (*net-* *handle_run_expression()* (*net-*
def.Controllers.SubprocessController.SubprocessController *def.Rules.CSVRule.CSVRule* *method*), 112
method), 80 *handle_run_expression()* (*net-*
def.Rules.InfluxDBLoggerRule.InfluxDBLoggerRule
method), 113
handle_add_source() (*net-* *method*), 113
def.Controllers.SystemMonitorController.SystemMonitorController *handle_run_expression()* (*net-*
method), 81 *def.Rules.INIRule.INIRule* *method*), 113
handle_add_source() (*net-* *handle_run_expression()* (*net-*
def.Controllers.XmlRpcController.XmlRpcController *def.Rules.YAMLRule.YAMLRule* *method*),
method), 84 *method*),
113
handle_add_source() (*net-* *method*), 84
def.Controllers.ZmqDataAccessController.ZmqDataAccessController (*net-*
method), 84 *def.Controllers.BaseController.BaseController*

method), 64
 handle_write_source() (*net-def.Controllers.BaseController.BaseController method*), 64
 handle_write_source() (*net-def.Controllers.ConcurrentWebRequestController.ConcurrentWebRequestController method*), 67
 handle_write_source() (*net-def.Controllers.InfluxDBLoggerController.InfluxDBLoggerController method*), 68
 handle_write_source() (*net-def.Controllers.InternalController.InternalController method*), 71
 handle_write_source() (*net-def.Controllers.ModbusClientController.ModbusClientController method*), 72
 handle_write_source() (*net-def.Controllers.ModbusServerController.ModbusServerController method*), 74
 handle_write_source() (*net-def.Controllers.MQTTDataMessageController.MQTTDataMessageController method*), 75
 handle_write_source() (*net-def.Controllers.OPCUAClientController.OPCUAClientController method*), 76
 handle_write_source() (*net-def.Controllers.OPCUAServerController.OPCUAServerController method*), 79
 handle_write_source() (*net-def.Controllers.RESTJsonController.RESTJsonController method*), 79
 handle_write_source() (*net-def.Controllers.SubprocessController.SubprocessController method*), 80
 handle_write_source() (*net-def.Controllers.SystemMonitorController.SystemMonitorController method*), 81
 handle_write_source() (*net-def.Controllers.XmlRpcController.XmlRpcController method*), 84
 handle_write_source() (*net-def.Controllers.ZmqDataAccessController.ZmqDataAccessController method*), 84
 has_basic_auth() (*net-def.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource method*), 123
 has_connect_request() (*net-def.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource method*), 123
 has_existing_instance() (*net-def.Rules.BaseRule.BaseRule method*), 110
 has_expression_in_source_ref() (*net-def.Shared.SharedExpressions.ExpressionInstances method*), 115
 has_initial_poll() (*net-def.Sources.SubprocessSource.SubprocessSource method*), 126
 has_interrupt() (*net-def.Controllers.BaseController.BaseController method*)
 has_interrupt() (*net-def.Engines.BaseEngine.BaseExpressionExecutor method*)
 has_interrupt() (*net-def.Rules.BaseRule.BaseRule method*), 110
 has_interval() (*net-def.Controllers.ConcurrentWebRequestController.NextInterval method*), 67
 has_interval() (*net-def.Controllers.SubprocessController.NextInterval method*), 80
 has_ref() (*net-def.Shared.SharedSources.SourceInstances method*), 118
 has_request() (*net-def.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource method*), 123
 has_role() (*net-def.Engines.webadmin.AdminIndex.User method*), 90
 has_role() (*net-def.Engines.webadmin.MyBaseView.MyBaseView method*), 94
 has_source() (*net-def.Controllers.BaseController.BaseController method*), 64
 has_source_ref() (*net-def.Shared.SharedExpressions.ExpressionInstances method*), 115
 HoldingRegisterSource (class in *net-def.Sources.HoldingRegisterSource*), 123
 import_file() (in module *netdef.Rules.utils*), 109
 instance_callback() (*net-def.Engines.webadmin.MyBaseView.MyBaseView method*), 94
 index() (*net-def.Engines.webadmin.AdminIndex.MyAdminIndexView method*), 90
 index() (*net-def.Engines.webadmin.FileModel.Files method*)
 index() (*net-def.Engines.webadmin.FileModel.InstallationRepo method*), 93
 index() (*net-def.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView method*), 103
 index() (*net-def.Engines.webadmin.SecurityWebadminView.SecurityWebadminView method*), 102

index() (*netdef.Engines.webadmin.Tools.Tools method*), 64
index_view() (*netdef.Engines.BaseEngine.BaseExpressionExecutor method*), 86
index_view() (*netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView method*), 92
index_view() (*netdef.Engines.webadmin.FileModel.Files method*), 92
index_view() (*netdef.Engines.webadmin.FileModel.InstallationRepo method*), 93
index_view() (*netdef.Engines.webadmin.SettingsModel.SettingsModelView method*), 96
index_view() (*netdef.Engines.webadmin.SourcesModel.SourcesModelView method*), 98
index_view() (*netdef.Engines.webadmin.StatisticsModel.StatisticsModelView method*), 100
InfluxDBLoggerController (*class in netdef.Controllers.InfluxDBLoggerController*), 68
InfluxDBLoggerInterface (*class in netdef.Interfaces.InfluxDBLoggerInterface*), 106
InfluxDBLoggerRule (*class in netdef.Rules.InfluxDBLoggerRule*), 113
InfluxDBLoggerSource (*class in netdef.Sources.InfluxDBLoggerSource*), 124
INIRule (*class in netdef.Rules.INIRule*), 113
init() (*netdef.Controllers.Controllers.Controllers method*), 62
init() (*netdef.Engines.BaseEngine.BaseEngine method*), 85
init() (*netdef.Engines.ThreadedEngine.ThreadedEngine method*), 88
init() (*netdef.Engines.ThreadedWebGuiEngine.ThreadedWebGuiEngine method*), 89
init() (*netdef.Rules.Rules.Rules method*), 108
init() (*netdef.Sources.Sources.Sources method*), 119
init_app() (*in module netdef.Engines.ThreadedWebGuiEngine*), 89
init_asyncio() (*netdef.Controllers.BaseAsyncController.BaseAsyncController method*), 65
init_items() (*netdef.Shared.SharedSources.SourceClasses method*), 118
init_parsers() (*netdef.Controllers.BaseController.BaseController method*), 64
init_queue() (*netdef.Controllers.BaseController.BaseController method*), 64
init_queue() (*netdef.Rules.BaseRule.BaseRule method*), 110
init_search() (*netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView method*), 92
init_search() (*netdef.Engines.webadmin.SettingsModel.SettingsModelView method*), 96
init_search() (*netdef.Engines.webadmin.SourcesModel.SourcesModelView method*), 98
init_server() (*netdef.Controllers.ModbusServerController.ModbusServerController method*), 74
init_sources() (*netdef.Controllers.BaseController.BaseController method*), 64
init_task_limit() (*netdef.Controllers.ConcurrentWebRequestController.ConcurrentWebRequestController method*), 67
INITIAL (*netdef.Sources.BaseSource.StatusCode attribute*), 122
install_service() (*in module netdef.systemd_service*), 58
InstallationRepo (*class in netdef.Engines.webadmin.FileModel*), 93
instance (*netdef.Engines.expression.Expression.Argument attribute*), 86
instances (*netdef.Shared.SharedExpressions.SharedExpressions attribute*), 115
instances (*netdef.Shared.SharedSources.SharedSources attribute*), 118
IntegerInterface (*class in netdef.Interfaces.IntegerInterface*), 107
IntegerSource (*class in netdef.Sources.IntegerSource*), 124
InternalController (*class in netdef.Controllers.InternalController*), 68
InternalSource (*class in netdef.Sources.InternalSource*), 124
Internal (*netdef.Controllers.SystemMonitorController.DataItem attribute*), 80
INVALID (*netdef.Sources.BaseSource.StatusCode attribute*), 122
ip_1 (*netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView attribute*), 103
ip_2 (*netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView attribute*), 103
ip_3 (*netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView attribute*), 103

ip_4 (*netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView* attribute), 103
 ip_5 (*netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView* attribute), 103
 is_accessible() (*netdef.Engines.webadmin.StatisticsModel.StatisticsModelForm* attribute), 99
 is_accessible() (*netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView* attribute), 99
 is_accessible() (*netdef.Interfaces.datamessage.DataDefinition* attribute), 105
 is_accessible() (*netdef.Engines.webadmin.FileModel.Files* key (*netdef.Interfaces.datamessage.DataMessage* attribute), 105
 method), 93
 is_accessible() (*netdef.Interfaces.InfluxDBLoggerInterface.Value* attribute), 106
 is_accessible() (*netdef.Engines.webadmin.FileModel.InstallationRepo* key (*netdef.Interfaces.InfluxDBLoggerInterface.Value* attribute), 106
 method), 93
 is_accessible() (*netdef.Rules.BaseRule.SourceInfo* attribute), 112
 is_accessible() (*netdef.Engines.webadmin.MyBaseView.MyBaseView* keyUsage (*netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView* attribute), 103
 method), 94
 is_accessible() (*netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView* keyUsage (*netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView* attribute), 103
 method), 103
 is_accessible() (*netdef.Engines.expression.Collector.Mode* attribute), 87
 is_accessible() (*netdef.Engines.webadmin.SecurityWebadminView.SecurityWebadminView* LAST_WITH_EVENT (*netdef.Engines.expression.Collector.Mode* attribute), 88
 method), 102
 is_accessible() (*netdef.Engines.webadmin.SettingsModel.SettingsModelView* LIST_ALL (*netdef.Engines.expression.Collector.Mode* attribute), 88
 method), 96
 is_accessible() (*netdef.Engines.webadmin.FileModel.Files* list_template (*netdef.Engines.webadmin.FileModel.Files* attribute), 93
 method), 98
 is_accessible() (*netdef.Engines.webadmin.FileModel.InstallationRepo* list_template (*netdef.Engines.webadmin.FileModel.InstallationRepo* attribute), 94
 method), 100
 is_accessible_path() (*netdef.Controllers.Controllers.Controllers* load() (*netdef.Controllers.Controllers.Controllers* method), 62
 method), 93
 is_hidden_value() (*netdef.Engines.BaseEngine.BaseEngine* load() (*netdef.Engines.BaseEngine.BaseEngine* method), 85
 method), 115
 is_uri() (*netdef.Interfaces.datamessage.DataDefinition* load() (*netdef.Engines.ThreadedEngine.ThreadedEngine* method), 88
 static method), 105
 is_uri() (*netdef.Interfaces.datamessage.DataMessage* load() (*netdef.Engines.ThreadedWebGuiEngine.ThreadedWebGuiEngine* method), 89
 static method), 105
 is_writable() (*netdef.Engines.webadmin.Views.Views* load() (*netdef.Engines.webadmin.Views.Views* method), 104
 method), 79
 is_writable() (*netdef.Rules.Rules.Rules* load() (*netdef.Rules.Rules.Rules* method), 108
 method), 79
 is_writable() (*netdef.Sources.Sources.Sources* load() (*netdef.Sources.Sources.Sources* method), 119
 method), 79
 is_writable() (*netdef.Rules.utils* load_point() (in module *netdef.Rules.utils*), 109
 method), 79
 K
 key (*netdef.Controllers.SystemMonitorController.DataItem* login (*netdef.Engines.webadmin.AdminIndex.LoginForm* attribute), 80
 attribute), 86
 key (*netdef.Engines.expression.Expression.Argument* attribute), 86
 attribute), 86
 key (*netdef.Engines.webadmin.SettingsModel.SettingsModel* login (*netdef.Engines.webadmin.SecurityWebadminView.SecurityForm* attribute), 101
 attribute), 94
 login_view() (*netdef.Engines.webadmin.AdminIndex.MyAdminIndexView* method), 90
 key (*netdef.Engines.webadmin.SettingsModel.SettingsModelForm* attribute), 95
 attribute), 95

LoginForm (class in netdef.Engines.webadmin.AdminIndex), 89
 logout_view() (netdef.Engines.webadmin.AdminIndex.MyAdminIndexView method), 90
 loop_futures() (netdef.Engines.ThreadedEngine.ExpressionExecutor method), 88
 loop_incoming() (netdef.Controllers.BaseController.BaseController method), 64
 loop_incoming() (netdef.Engines.BaseEngine.BaseExpressionExecutor method), 86
 loop_incoming() (netdef.Rules.BaseRule.BaseRule method), 110
 loop_incoming_until_interrupt() (netdef.Controllers.BaseAsyncController.BaseAsyncController method), 65
 loop_mqtt() (netdef.Controllers.MQTTDataMessageController.MQTTDataMessageController method), 75
 loop_outgoing() (netdef.Controllers.BaseController.BaseController method), 64
 loop_outgoing() (netdef.Controllers.OPCUAClientController.OPCUAClientController method), 76
 loop_outgoing() (netdef.Controllers.RESTJsonController.RESTJsonController method), 79
 loop_outgoing_until_interrupt() (netdef.Controllers.CommTestController.CommTestController method), 67
 loop_outgoing_until_interrupt() (netdef.Controllers.ConcurrentWebRequestController.ConcurrentWebRequestController method), 67
 loop_subscribers() (netdef.Controllers.ZmqDataAccessController.ZmqDataAccessController method), 84
 loop_until_app_state_running() (netdef.Controllers.BaseController.BaseController method), 64

M

maintain_searches() (netdef.Rules.BaseRule.BaseRule method), 110
 make_admin_users_dict() (in module netdef.Engines.ThreadedWebGuiEngine), 89
 make_message() (netdef.Sources.MQTTDataMessageSource.MQTTDataMessageSource static method), 125
 make_points() (netdef.Sources.InfluxDBLoggerSource.InfluxDBLoggerSource static method), 124

make_rpc_request() (netdef.Sources.XmlRpcMethodCallSource.XmlRpcMethodCallSource method), 127
 MessageType (class in netdef.Shared.SharedQueues), 116
 method (netdef.Sources.ConcurrentWebRequestSource.Request attribute), 123
 mkdir() (netdef.Engines.webadmin.FileModel.Files method), 93
 mkdir() (netdef.Engines.webadmin.FileModel.InstallationRepo method), 94
 MockExpression (class in netdef.testutils), 60
 MockShared (class in netdef.testutils), 60
 MockSource (class in netdef.testutils), 60
 ModbusClientController (class in netdef.Controllers.ModbusClientController), 71
 ModbusServerController (class in netdef.Controllers.ModbusServerController), 71
 Mode (class in netdef.Engines.expression.Collector), 87
 modify_monitored_items() (netdef.Controllers.OPCUAServerController.OPCUAServerController method), 79
 module (netdef.Rules.BaseRule.ExpressionInfo attribute), 112
 module_filename (netdef.Engines.webadmin.ExpressionsView.ExpressionsModel attribute), 90
 module_filename (netdef.Engines.webadmin.ExpressionsView.ExpressionsModelForm attribute), 90
 mqtt_connect() (netdef.Controllers.MQTTDataMessageController.MQTTDataMessageController method), 75
 mqtt_safe_disconnect() (netdef.Controllers.MQTTDataMessageController.MQTTDataMessageController method), 75
 MQTTDataMessageController (class in netdef.Controllers.MQTTDataMessageController), 75
 MQTTDataMessageSource (class in netdef.Sources.MQTTDataMessageSource), 125
 MyAdminIndexView (class in netdef.Engines.webadmin.AdminIndex), 90
 MyBaseView (class in netdef.Engines.webadmin.MyBaseView), 94
 MyContext (class in netdef.Controllers.ModbusServerController), 75
 MyController (class in netdef.Controllers.ModbusServerController), 75

N

netdef.__main__ (module), 54
netdef.Controllers.BaseAsyncController (module), 65
netdef.Controllers.BaseController (module), 62
netdef.Controllers.CommTestController (module), 66
netdef.Controllers.ConcurrentWebRequestController (module), 67
netdef.Controllers.Controllers (module), 62
netdef.Controllers.CrontabController (module), 68
netdef.Controllers.InfluxDBLoggerController (module), 68
netdef.Controllers.InternalController (module), 68
netdef.Controllers.ModbusClientController (module), 71
netdef.Controllers.ModbusServerController (module), 73
netdef.Controllers.MQTTDataMessageController (module), 75
netdef.Controllers.OPCUAClientController (module), 76
netdef.Controllers.OPCUAServerController (module), 76
netdef.Controllers.RESTJsonController (module), 79
netdef.Controllers.SubprocessController (module), 80
netdef.Controllers.SystemMonitorController (module), 80
netdef.Controllers.XmlRpcController (module), 82
netdef.Controllers.ZmqDataAccessController (module), 84
netdef.Engines.BaseEngine (module), 85
netdef.Engines.expression.Collector (module), 87
netdef.Engines.expression.Expression (module), 86
netdef.Engines.NginxWebGuiReverseProxy (module), 89
netdef.Engines.ThreadedEngine (module), 88
netdef.Engines.ThreadedWebGuiEngine (module), 88
netdef.Engines.webadmin.AdminIndex (module), 89
netdef.Engines.webadmin.ExpressionsView (module), 90
netdef.Engines.webadmin.FileModel (module), 92
netdef.Engines.webadmin.MyBaseView (module), 94
netdef.Engines.webadmin.SecurityCertificatesView (module), 102
netdef.Engines.webadmin.SecurityWebadminView (module), 101
netdef.Engines.webadmin.SettingsModel (module), 94
netdef.Engines.webadmin.SourcesModel (module), 96
netdef.Engines.webadmin.StatisticsModel (module), 99
netdef.Engines.webadmin.Tools (module), 100
netdef.Engines.webadmin.Views (module), 103
netdef.Interfaces.ByteStringInterface (module), 105
netdef.Interfaces.CommTestInterface (module), 106
netdef.Interfaces.ConcurrentWebRequestInterface (module), 106
netdef.Interfaces.datamessage (module), 105
netdef.Interfaces.DefaultInterface (module), 104
netdef.Interfaces.FloatInterface (module), 106
netdef.Interfaces.InfluxDBLoggerInterface (module), 106
netdef.Interfaces.IntegerInterface (module), 107
netdef.Interfaces.internal.tick (module), 105
netdef.Interfaces.StringInterface (module), 107
netdef.Interfaces.UnitOfValueInterface (module), 107
netdef.Rules.BaseRule (module), 109
netdef.Rules.CSVRule (module), 112
netdef.Rules.InfluxDBLoggerRule (module), 113
netdef.Rules.INIRule (module), 113
netdef.Rules.Rules (module), 108
netdef.Rules.utils (module), 109
netdef.Rules.YAMLRule (module), 113
netdef.service (module), 55
netdef.Shared.Internal (module), 114
netdef.Shared.Shared (module), 114
netdef.Shared.SharedConfig (module), 115
netdef.Shared.SharedExpressions (module), 115
netdef.Shared.SharedQueues (module), 116
netdef.Shared.SharedSources (module), 118

netdef.Sources.BaseSource (module), 120
 netdef.Sources.BytestringSource (module), 122
 netdef.Sources.CommTestSource (module), 122
 netdef.Sources.ConcurrentWebRequestSource (module), 122
 netdef.Sources.CrontabSource (module), 123
 netdef.Sources.DictSource (module), 123
 netdef.Sources.FloatSource (module), 123
 netdef.Sources.HoldingRegisterSource (module), 123
 netdef.Sources.InfluxDBLoggerSource (module), 124
 netdef.Sources.IntegerSource (module), 124
 netdef.Sources.InternalSource (module), 124
 netdef.Sources.MQTTDataMessageSource (module), 125
 netdef.Sources.Sources (module), 119
 netdef.Sources.SubprocessSource (module), 126
 netdef.Sources.SystemMonitorSource (module), 126
 netdef.Sources.TextSource (module), 127
 netdef.Sources.VariantSource (module), 127
 netdef.Sources.XmlRpcMethodCallSource (module), 127
 netdef.Sources.ZmqDataAccessSource (module), 127
 netdef.systemd_service (module), 58
 netdef.testutils (module), 60
 netdef.utils (module), 59
 netdef.windows_service (module), 56
 new (netdef.Engines.expression.Expression.Argument attribute), 86
 new_flask_secret (netdef.Engines.webadmin.SecurityWebadminView.BasicSecurityForm attribute), 101
 new_flask_secret (netdef.Engines.webadmin.SecurityWebadminView.SecurityForm attribute), 101
 next (netdef.Controllers.SystemMonitorController.DataItem attribute), 81
 next () (netdef.Controllers.ConcurrentWebRequestController.NextInterval method), 68
 next () (netdef.Controllers.SubprocessController.NextInterval method), 80
 NextInterval (class in netdef.Controllers.ConcurrentWebRequestController), 67
 NextInterval (class in netdef.Controllers.SubprocessController), 80
 NginxReverseProxy (class in netdef.Engines.NginxWebGuiReverseProxy), 89
 NONE (netdef.Sources.BaseSource.StatusCode attribute), 122
 NoUnitInterface (class in netdef.Interfaces.UnitOfValueInterface), 107
O
 old_password (netdef.Engines.webadmin.SecurityWebadminView.BasicSecurityForm attribute), 101
 old_password (netdef.Engines.webadmin.SecurityWebadminView.SecurityForm attribute), 101
 on (netdef.Shared.Internal.Statistics attribute), 114
 on_connect () (netdef.Controllers.MQTTDataMessageController.MQTTDataMessage method), 75
 on_disconnect () (netdef.Controllers.MQTTDataMessageController.MQTTDataMessage method), 75
 on_message () (netdef.Controllers.MQTTDataMessageController.MQTTDataMessage method), 75
 OPCUAClientController (class in netdef.Controllers.OPCUAClientController), 76
 OPCUAServerController (class in netdef.Controllers.OPCUAServerController), 77
 origin (netdef.Interfaces.datamessage.DataMessage attribute), 105
P
 pack_add_source () (netdef.Sources.BaseSource.BaseSource method), 120
 pack_address () (netdef.Sources.ZmqDataAccessSource.ZmqDataAccessSource method), 127
 pack_subitems () (netdef.Sources.BaseSource.BaseSource static method), 120
 pack_unit_and_address () (netdef.Sources.HoldingRegisterSource.HoldingRegisterSource static method), 123
 pack_value () (netdef.Sources.BaseSource.BaseSource method), 120
 pack_value () (netdef.Sources.InternalSource.InternalSource method), 125
 pack_value () (netdef.Sources.MQTTDataMessageSource.MQTTDataMessageSource

method), 125
 params (*netdef.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource* attribute), 123
 parse_item() (*netdef.Controllers.RESTJsonController.RESTJsonController* method), 79
 parse_item() (*netdef.Controllers.SubprocessController.SubprocessController* method), 80
 parse_item() (*netdef.Controllers.XmlRpcController.XmlRpcController* method), 84
 parse_message() (*netdef.Sources.MQTTDataMessageSource.MQTTDataMessageSource* static method), 125
 parse_response() (*netdef.Controllers.SubprocessController.SubprocessController* method), 80
 parse_response() (*netdef.Controllers.XmlRpcController.XmlRpcController* method), 84
 parse_rpc_response() (*netdef.Sources.XmlRpcMethodCallSource.XmlRpcMethodCallSource* method), 127
 parse_stdout_response() (*netdef.Sources.SubprocessSource.SubprocessSource* method), 126
 parse_url() (*netdef.Sources.ConcurrentWebRequestSource.ConcurrentWebRequestSource* method), 123
 password (*netdef.Engines.webadmin.AdminIndex.LoginForm* attribute), 89
 password (*netdef.Engines.webadmin.SecurityWebadminView.SecurityWebadminView* attribute), 101
 PercentUnitInterface (class in *netdef.Interfaces.UnitOfValueInterface*), 107
 poll_data() (*netdef.Controllers.SystemMonitorController.SystemMonitorController* method), 81
 poll_outgoing_item() (*netdef.Controllers.BaseController.BaseController* method), 65
 poll_outgoing_item() (*netdef.Controllers.CrontabController.CrontabController* method), 68
 poll_outgoing_item() (*netdef.Controllers.InternalController.InternalController* method), 71
 poll_outgoing_item() (*netdef.Controllers.ModbusClientController.ModbusClientController* method), 73
 poll_outgoing_item() (*netdef.Controllers.SubprocessController.SubprocessController* method), 80
 poll_outgoing_item() (*netdef.Controllers.XmlRpcController.XmlRpcController* method), 84
 process() (*netdef.Engines.webadmin.SourcesModel.SourcesModelForm* method), 96
 process_ticks() (*netdef.Rules.BaseRule.BaseRule* method), 110
 publish_data_item() (*netdef.Controllers.MQTTDataMessageController.MQTTDataMessageController* method), 75

R

READ_ALL (*netdef.Shared.SharedQueues.MessageType* attribute), 116
 read_default() (*netdef.Shared.SharedConfig.Config* method), 115
 READ_SOURCE (*netdef.Shared.SharedQueues.MessageType* attribute), 116
 ready() (*netdef.Controllers.SystemMonitorController.DataItem* method), 81
 register() (in module *netdef.Controllers.Controllers*), 62
 register() (in module *netdef.Engines.webadmin.Views*), 104
 register() (in module *netdef.Rules.Rules*), 108
 register() (in module *netdef.Sources.Sources*), 119
 register_set_callback() (*netdef.Sources.BaseSource.BaseSource* method), 121
 REMOVE_SOURCE (*netdef.Shared.SharedQueues.MessageType* attribute), 116
 rename() (*netdef.Engines.webadmin.FileModel.Files* method), 93
 rename() (*netdef.Engines.webadmin.FileModel.InstallationRepo* method), 94
 request (class in *netdef.Sources.ConcurrentWebRequestSource*), 123
 request_new() (*netdef.Engines.webadmin.AdminIndex.MyAdminIndexView* method), 90
 requesting (*netdef.Engines.webadmin.AdminIndex.MyAdminIndexView* attribute), 90
 RESTJsonController (class in *netdef.Controllers.RESTJsonController*), 79

Result (class in netdef.Sources.ConcurrentWebRequestSource), 123
 result (netdef.Sources.ConcurrentWebRequestSource.Result attribute), 123
 rpc_call () (netdef.Controllers.XmlRpcController.XmlRpcController method), 84
 rule, 6
 rule (netdef.Interfaces.InfluxDBLoggerInterface.Value attribute), 106
 rule_name_from_key () (netdef.Rules.BaseRule.BaseRule method), 110
 Rules (class in netdef.Rules.Rules), 108
 run () (netdef.Controllers.BaseAsyncController.BaseAsyncController method), 65
 run () (netdef.Controllers.BaseController.BaseController method), 65
 run () (netdef.Controllers.CommTestController.CommTestController method), 67
 run () (netdef.Controllers.ConcurrentWebRequestController.ConcurrentWebRequestController method), 67
 run () (netdef.Controllers.CrontabController.CrontabController method), 68
 run () (netdef.Controllers.InfluxDBLoggerController.InfluxDBLoggerController method), 68
 run () (netdef.Controllers.InternalController.InternalController method), 71
 run () (netdef.Controllers.ModbusClientController.ModbusClientController method), 73
 run () (netdef.Controllers.ModbusServerController.ModbusServerController method), 74
 run () (netdef.Controllers.MQTTDataMessageController.MQTTDataMessageController method), 75
 run () (netdef.Controllers.OPCUAClientController.OPCUAClientController method), 76
 run () (netdef.Controllers.OPCUAServerController.OPCUAServerController method), 79
 run () (netdef.Controllers.RESTJsonController.RESTJsonController method), 79
 run () (netdef.Controllers.SubprocessController.SubprocessController method), 80
 run () (netdef.Controllers.SystemMonitorController.SystemMonitorController method), 81
 run () (netdef.Controllers.XmlRpcController.XmlRpcController method), 84
 run () (netdef.Controllers.ZmqDataAccessController.ZmqDataAccessController method), 84
 run () (netdef.Engines.BaseEngine.BaseExpressionExecutor method), 86
 run () (netdef.Engines.ThreadedEngine.ExpressionExecutor method), 88
 run () (netdef.Rules.BaseRule.BaseRule method), 111
 run () (netdef.Rules.CSVRule.CSVRule method), 112
 run () (netdef.Rules.InfluxDBLoggerRule.InfluxDBLoggerRule method), 113
 run () (netdef.Rules.INIRule.INIRule method), 113
 run () (netdef.Rules.YAMLRule.YAMLRule method), 113
 run_async_on_interrupt () (netdef.Controllers.BaseAsyncController.BaseAsyncController method), 66
 RUN_EXPRESSION (netdef.Shared.SharedQueues.MessageType attribute), 116
 run_expressions_in_engine () (netdef.Shared.SharedQueues.SharedQueues method), 116
 run_expressions_in_rule () (netdef.Shared.SharedQueues.SharedQueues method), 117
 run_service () (in module netdef.service), 56
 run_service () (in module netdef.systemd_service), 59
 run_service () (in module netdef.windows_service), 57
 RUNNING (netdef.Shared.SharedQueues.AppStateType attribute), 116

S

safe_disconnect () (netdef.Controllers.ModbusClientController.ModbusClientController method), 73
 safe_disconnect () (netdef.Controllers.OPCUAClientController.OPCUAClientController method), 76
 sample () (netdef.Engines.webadmin.ExpressionsView.ExpressionsModelView static method), 92
 sample () (netdef.Engines.webadmin.SettingsModel.SettingsModelView static method), 96
 sample_log () (netdef.Engines.webadmin.SourcesModel.SourcesModelView static method), 98
 selection (netdef.Engines.webadmin.SettingsModel.SettingsModel attribute), 94
 selection (netdef.Engines.webadmin.SettingsModel.SettingsModelForm attribute), 95
 SecurityCertificatesForm (class in netdef.Engines.webadmin.SecurityCertificatesView), 102
 SecurityCertificatesView (class in netdef.Engines.webadmin.SecurityCertificatesView), 103
 SecurityForm (class in netdef.Engines.webadmin.SecurityWebadminView), 101
 SecurityWebadminView (class in netdef.Engines.webadmin.SecurityWebadminView), 101

send_datachange () (net- set_init_values () (net-
 def:Controllers.OPCUAClientController.OPCUAClientController (netdef.testutils.MockExpression method), 60
 method), 76 set_none_values () (net-
 send_datachange () (net- def:testutils.MockExpression method), 60
 def:Controllers.OPCUAServerController.OPCUAServerController (netdef.Engines.webadmin.SourcesModel.SourcesModelForm
 method), 79 attribute), 97
 send_datachange () (net- set_source_time (net-
 def:Controllers.RESTJsonController.RESTJsonController def:Engines.webadmin.SourcesModel.SourcesModelForm
 method), 79 attribute), 97
 send_datachange () (net- set_status_code (net-
 def:Controllers.SubprocessController.SubprocessController def:Engines.webadmin.SourcesModel.SourcesModelForm
 method), 80 attribute), 97
 send_datachange () (net- set_value (netdef.Engines.webadmin.SourcesModel.SourcesModelForm
 def:Controllers.SystemMonitorController.SystemMonitorController attribute), 97
 method), 81 set_value_from_string () (net-
 send_datachange () (net- def:Sources.BaseSource.BaseSource method),
 def:Controllers.XmlRpcController.XmlRpcController 121
 method), 84 setbit () (netdef.Interfaces.IntegerInterface.IntegerInterface
 method), 107
 send_expressions_to_engine () (net- setbits () (netdef.Interfaces.IntegerInterface.IntegerInterface
 def:Rules.BaseRule.BaseRule method), 111 method), 107
 send_message_to_controller () (net- SettingsModel (class in net-
 def:Shared.SharedQueues.SharedQueues def:Engines.webadmin.SettingsModel), 94
 method), 117 SettingsModelForm (class in net-
 send_message_to_engine () (net- def:Engines.webadmin.SettingsModel), 94
 def:Shared.SharedQueues.SharedQueues SettingsModelView (class in net-
 method), 117 def:Engines.webadmin.SettingsModel), 95
 send_message_to_rule () (net- setup (netdef.Rules.BaseRule.ExpressionInfo attribute),
 def:Shared.SharedQueues.SharedQueues 112
 method), 117 setup (netdef.Rules.BaseRule.SourceInfo attribute), 112
 send_outgoing () (net- SETUP (netdef.Shared.SharedQueues.AppStateType at-
 def:Controllers.BaseController.BaseController tribute), 116
 method), 65 setup () (in module net-
 send_running_state_to_controller () (netdef.Shared.SharedQueues.SharedQueues
 method), 117 def:Engines.webadmin.ExpressionsView),
 92
 send_setup_state_to_controller () (net- setup () (in module net-
 def:Shared.SharedQueues.SharedQueues def:Engines.webadmin.FileModel), 94
 method), 117 setup () (in module net-
 send_ticks () (netdef.Rules.BaseRule.BaseRule def:Engines.webadmin.SecurityCertificatesView),
 method), 111 103
 service_actions () (net- setup () (in module net-
 def:Controllers.ModbusServerController.MyController def:Engines.webadmin.SecurityWebadminView),
 method), 75 102
 set (netdef.Engines.expression.Expression.Argument at- setup () (in module net-
 tribute), 86 def:Engines.webadmin.SettingsModel), 96
 set (netdef.Sources.BaseSource.BaseSource attribute), setup () (in module net-
 121 def:Engines.webadmin.SourcesModel), 98
 set () (netdef.Shared.Internal.Statistics static method), setup () (in module net-
 114 def:Engines.webadmin.StatisticsModel),
 100
 set_config () (netdef.Shared.SharedConfig.Config method), 115 setup () (in module netdef.Engines.webadmin.Tools),
 101
 set_hidden_value () (net- setup () (in module net-
 def:Shared.SharedConfig.Config method), 115 def:Sources.ConcurrentWebRequestSource),
 115

123
 setup () (in module netdef.Sources.SubprocessSource), 126
 setup () (netdef.Engines.webadmin.Views.Views method), 104
 setup () (netdef.Rules.BaseRule.BaseRule method), 111
 setup () (netdef.Rules.CSVRule.CSVRule method), 112
 setup () (netdef.Rules.InfluxDBLoggerRule.InfluxDBLoggerRule method), 113
 setup () (netdef.Rules.INIRule.INIRule method), 113
 setup () (netdef.Rules.YAMLRule.YAMLRule method), 113
 setup_auto_logging () (netdef.Rules.InfluxDBLoggerRule.InfluxDBLoggerRule method), 113
 setup_conf_secrets_and_https () (netdef.Engines.webadmin.SecurityWebadminView.SecurityWebadminView method), 102
 setup_conf_userdata () (netdef.Engines.webadmin.SecurityWebadminView.SecurityWebadminView method), 102
 setup_csv_rule () (netdef.Rules.CSVRule.CSVRule method), 112
 setup_done () (netdef.Rules.BaseRule.BaseRule method), 111
 setup_form_defaults () (netdef.Engines.webadmin.SecurityWebadminView.SecurityWebadminView method), 102
 setup_ini_rule () (netdef.Rules.INIRule.INIRule method), 113
 setup_interval_plan () (netdef.Controllers.SubprocessController.SubprocessController method), 80
 setup_logging () (in module netdef.utils), 59
 setup_ticks () (netdef.Rules.BaseRule.BaseRule method), 111
 setup_yaml_rule () (netdef.Rules.YAMLRule.YAMLRule method), 113
 setValues () (netdef.Controllers.ModbusServerController.ModbusServerController method), 75
 Shared (class in netdef.Shared.Shared), 114
 SharedExpressions (class in netdef.Shared.SharedExpressions), 115
 SharedQueues (class in netdef.Shared.SharedQueues), 116
 SharedSources (class in netdef.Shared.SharedSources), 118
 shutdown_server () (in module netdef.Engines.webadmin.AdminIndex), 90
 shutdown_view () (netdef.Engines.webadmin.AdminIndex.MyAdminIndexView method), 90
 shuttingdown (netdef.Engines.webadmin.AdminIndex.MyAdminIndexView attribute), 90
 sleep () (netdef.Controllers.BaseController.BaseController method), 65
 sleep () (netdef.Rules.BaseRule.BaseRule method), 111
 source, 6
 source (netdef.Engines.webadmin.SourcesModel.SourcesModelForm attribute), 97
 source (netdef.Interfaces.InfluxDBLoggerInterface.Value attribute), 106
 source_and_controller_from_key () (netdef.Rules.BaseRule.BaseRule method), 111
 source_datatype (netdef.Engines.webadmin.SourcesModel.SourcesModelForm attribute), 97
 source_time (netdef.Interfaces.InfluxDBLoggerInterface.Value attribute), 105
 source_type (netdef.Controllers.SystemMonitorController.DataItem attribute), 81
 SourceClasses (class in netdef.Shared.SharedSources), 118
 SourceInfo (class in netdef.Rules.BaseRule), 112
 SourceInstances (class in netdef.Shared.SharedSources), 118
 Sources (class in netdef.Sources.Sources), 119
 SourcesModelForm (class in netdef.Engines.webadmin.SourcesModel), 96
 SourcesModelView (class in netdef.Engines.webadmin.SourcesModel), 97
 spans (netdef.Controllers.ConcurrentWebRequestController.NextInterval attribute), 68
 spans (netdef.Controllers.SubprocessController.NextInterval attribute), 80
 ssl_certificate (netdef.Engines.webadmin.SecurityWebadminView.SecurityForm attribute), 101
 ssl_certificate_key (netdef.Engines.webadmin.SecurityWebadminView.SecurityForm attribute), 101
 ssl_on (netdef.Engines.webadmin.SecurityWebadminView.SecurityForm attribute), 101
 start (netdef.Controllers.ConcurrentWebRequestController.NextInterval attribute), 68
 start (netdef.Controllers.SubprocessController.NextInterval attribute), 80
 start () (netdef.Engines.BaseEngine.BaseEngine method), 85
 start () (netdef.Engines.ThreadedEngine.ThreadedEngine method), 88
 Statistics (class in netdef.Shared.Internal), 114

statistics (*netdef.Shared.Internal.Statistics attribute*), 114
 statistics_update() (in module *netdef.Controllers.SystemMonitorController*), 82
 statistics_update() (*netdef.Controllers.BaseController.BaseController method*), 65
 StatisticsModel (class in *netdef.Engines.webadmin.StatisticsModel*), 99
 StatisticsModelForm (class in *netdef.Engines.webadmin.StatisticsModel*), 99
 StatisticsModelView (class in *netdef.Engines.webadmin.StatisticsModel*), 99
 status_change_notification() (*netdef.Controllers.OPCUAClientController.SubHandler method*), 76
 status_code (*netdef.Interfaces.datamessage.DataMessage attribute*), 105
 status_code (*netdef.Interfaces.InfluxDBLoggerInterface.Value attribute*), 107
 status_ok (*netdef.Engines.expression.Expression.Argument attribute*), 86
 StatusCode (class in *netdef.Sources.BaseSource*), 122
 stdout_from_terminal() (in module *netdef.Controllers.SubprocessController*), 80
 stdout_from_terminal() (in module *netdef.Engines.webadmin.Tools*), 101
 stdout_from_terminal_as_generator() (in module *netdef.Engines.webadmin.Tools*), 101
 stop() (*netdef.Engines.BaseEngine.BaseEngine method*), 85
 stop() (*netdef.Engines.ThreadedEngine.ThreadedEngine method*), 88
 store_to_disk() (*netdef.Controllers.InternalController.InternalController method*), 71
 StringInterface (class in *netdef.Interfaces.StringInterface*), 107
 SubHandler (class in *netdef.Controllers.OPCUAClientController*), 76
 SubHandler (class in *netdef.Controllers.OPCUAServerController*), 79
 subjectAltName (*netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesForm attribute*), 103
 SubprocessController (class in *netdef.Controllers.SubprocessController*), 80
 SubprocessSource (class in *netdef.Sources.SubprocessSource*), 126
 svc_name (*netdef.systemd_service.ApplicationService attribute*), 58
 SvcDoRun() (*netdef.windows_service.GenericApplicationService method*), 56
 SvcStop() (*netdef.windows_service.GenericApplicationService method*), 56
 SystemMonitorByteSource (class in *netdef.Sources.SystemMonitorSource*), 126
 SystemMonitorController (class in *netdef.Controllers.SystemMonitorController*), 81
 SystemMonitorPercentSource (class in *netdef.Sources.SystemMonitorSource*), 126
 SystemMonitorSource (class in *netdef.Sources.SystemMonitorSource*), 126

T

Template_callback (*netdef.systemd_service.ApplicationService attribute*), 58
 TextSource (class in *netdef.Sources.TextSource*), 127
 ThreadedEngine (class in *netdef.Engines.ThreadedEngine*), 88
 ThreadedWebGuiEngine (class in *netdef.Engines.ThreadedWebGuiEngine*), 88
 Tick (class in *netdef.Interfaces.internal.tick*), 105
 TICK (*netdef.Shared.SharedQueues.MessageType attribute*), 116
 tick() (*netdef.Interfaces.internal.tick.Tick method*), 105
 timediff() (*netdef.Interfaces.internal.tick.Tick method*), 105
 Tools (class in *netdef.Engines.webadmin.Tools*), 100
 typename (*netdef.Rules.BaseRule.SourceInfo attribute*), 112

U

unpack_address() (*netdef.Sources.ZmqDataAccessSource.ZmqDataAccessSource method*), 127
 unpack_host_and_port() (*netdef.Sources.CommTestSource.CommTestSource method*), 122
 unpack_measurement() (*netdef.Sources.InfluxDBLoggerSource.InfluxDBLoggerSource method*), 124
 unpack_subitems() (*netdef.Sources.BaseSource.BaseSource static method*), 121
 unpack_subitems() (*netdef.Sources.SubprocessSource.SubprocessSource static method*), 126

unpack_subitems () (net- 90
 def.Sources.XmlRpcMethodCallSource.XmlRpcMethodCallSource.sources_empty () (net-
 static method), 127 def.Engines.webadmin.SecurityWebadminView.SecurityWebadminView
 unpack_unit_and_address () (net- method), 102
 def.Sources.HoldingRegisterSource.HoldingRegisterSource
 method), 123

V

unpack_value () (net- validate_current_password () (net-
 def.Sources.BaseSource.BaseSource static def.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesView
 method), 121 static method), 103
 unpack_value () (net- validate_login () (net-
 def.Sources.InternalSource.InternalSource static def.Engines.webadmin.AdminIndex.LoginForm
 static method), 125 method), 90
 unpack_value () (net- validate_old_password () (net-
 def.Sources.MQTTDataMessageSource.MQTTDataMessageSource static def.Engines.webadmin.SecurityWebadminView.SecurityForm
 static method), 126 static method), 101
 update (netdef.Engines.expression.Expression.Argument validate_password (net-
 attribute), 86 def.Engines.webadmin.SecurityWebadminView.BasicSecurityForm
 update_model () (net- attribute), 101
 def.Engines.webadmin.SourcesModel.SourcesModelView validate_password () (net-
 method), 98 def.Engines.webadmin.SecurityWebadminView.SecurityForm
 update_on (netdef.Engines.webadmin.SecurityWebadminView.SecurityForm static method), 101
 attribute), 101 Value (class in netdef.Interfaces.CommTestInterface),
 update_pre_release (net- 106
 def.Engines.webadmin.SecurityWebadminView.SecurityForm (class in net-
 attribute), 101 def.Interfaces.ConcurrentWebRequestInterface),
 update_source_instance_status () (net- 106
 def.Controllers.BaseController.BaseController Value (class in net-
 class method), 65 def.Interfaces.InfluxDBLoggerInterface),
 update_source_instance_value () (net- 106
 def.Controllers.BaseController.BaseController value (netdef.Engines.expression.Expression.Argument
 static method), 65 attribute), 86
 update_statistics () (net- value (netdef.Engines.webadmin.SettingsModel.SettingsModel
 def.Rules.BaseRule.BaseRule method), 112 attribute), 94
 update_usertable () (net- value (netdef.Engines.webadmin.SettingsModel.SettingsModelForm
 def.Engines.webadmin.SecurityWebadminView.SecurityWebadminView attribute), 95
 method), 102 value (netdef.Engines.webadmin.StatisticsModel.StatisticsModel
 attribute), 99
 update_value () (netdef.testutils.MockSource value (netdef.Engines.webadmin.StatisticsModel.StatisticsModelForm
 method), 61 attribute), 99
 upload () (netdef.Engines.webadmin.FileModel.Files value (netdef.Engines.webadmin.StatisticsModel.StatisticsModelForm
 method), 93 attribute), 99
 upload () (netdef.Engines.webadmin.FileModel.InstallationRepo value (netdef.Interfaces.datamessage.DataMessage at-
 method), 94 tribute), 105
 uri_1 (netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesForm value (netdef.Interfaces.InfluxDBLoggerInterface.Value
 attribute), 103 attribute), 107
 uri_2 (netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesForm value_as_string (net-
 attribute), 103 def.Sources.BaseSource attribute),
 121
 uri_3 (netdef.Engines.webadmin.SecurityCertificatesView.SecurityCertificatesForm (net-
 attribute), 103 def.Sources.BytestringSource.BytestringSource
 url (netdef.Sources.ConcurrentWebRequestSource.Request attribute), 122
 attribute), 123 value_as_string (net-
 urlerrorhandling () (net- def.Sources.SystemMonitorSource.SystemMonitorSource
 def.Controllers.RESTJsonController.RESTJsonController attribute), 127
 method), 79 VariantSource (class in net-
 User (class in netdef.Engines.webadmin.AdminIndex), def.Sources.VariantSource), 127

`verify()` (*netdef.Shared.SharedConfig.Config*
method), 115
`Views` (*class in netdef.Engines.webadmin.Views*), 103

W

`wait()` (*netdef.Engines.BaseEngine.BaseEngine*
method), 85
`wait()` (*netdef.Engines.ThreadedEngine.ThreadedEngine*
method), 88
`write()` (*netdef.Controllers.OPCUAServerController.CustomAnonInternalSession*
method), 76
`WRITE_SOURCE` (*net-*
def.Shared.SharedQueues.MessageType *at-*
tribute), 116
`write_value_to_controller()` (*net-*
def.Shared.SharedQueues.SharedQueues
method), 117

X

`XmlRpcController` (*class in net-*
def.Controllers.XmlRpcController), 82
`XmlRpcMethodCallSource` (*class in net-*
def.Sources.XmlRpcMethodCallSource),
127

Y

`YAMLRule` (*class in netdef.Rules.YAMLRule*), 113

Z

`ZmqDataAccessController` (*class in net-*
def.Controllers.ZmqDataAccessController),
84
`ZmqDataAccessSource` (*class in net-*
def.Sources.ZmqDataAccessSource), 127