
pyvlog

Release 0.1

Sep 05, 2019

Contents:

1	pyvlog package	1
1.1	pyvlog.parsers module	1
1.2	pyvlog.converters module	2
1.3	pyvlog.utils module	3
2	Indices and tables	5
	Python Module Index	7
	Index	9

1.1 pyvlog.parsers module

Classes for parsing V-Log messages and logging statuses.

class `pyvlog.parsers.VLogParser` (*logged_types=['detectie', 'externeSignaalgroep'], **kwargs*)
Bases: `object`

Base class for parsing v-log messages. Does not log statuses.

Parameters `logged_types` (*list*) – Message types (should match keys of `messagesypes.MESSAGE_TYPE_DICT`) to be logged. If empty list all types are logged.

log_status (*status*)
Placeholder function, does nothing with the status.

Parameters `status` (*dict*) – V-log status to be logged.

parse_message (*message*)
Parse a v-log message and update the status. If the status is complete `log_status` is called.

Parameters `message` (*str*) – V-log message.

class `pyvlog.parsers.VLogParserToJson` (*path_to_json, logged_types=['detectie', 'externeSignaalgroep']*)
Bases: `pyvlog.parsers.VLogParser`

Class for parsing v-log messages to a json of statuses. Appends each logged status to a json file.

Parameters

- **path_to_json** (*str*) – Path to json file.
- **logged_types** (*list*) – Message types (should match keys of `messagesypes.MESSAGE_TYPE_DICT`) to be logged. If empty list all types are logged.

log_status (*status, path_to_json*)
Append the status to a json file.

Parameters

- **status** (*dict*) – V-log status to be logged.
- **path_to_json** (*str*) – Path to json file.

class pyvlog.parsers.VLogParserToList (*status_list*, *logged_types*=['detectie', 'externeSignaalgroep'])

Bases: `pyvlog.parsers.VLogParser`

Class for parsing v-log messages to a list of statuses. Appends each logged status to a list object.

Parameters

- **status_list** (*list*) – List to be appended to.
- **logged_types** (*list*) – Message types (should match keys of `messagetypes.MESSAGE_TYPE_DICT`) to be logged. If empty list all types are logged.

log_status (*status*, *status_list*)

Append the status to a list.

Parameters

- **status** (*dict*) – V-log status to be logged.
- **status_list** (*list*) – List to be appended to.

1.2 pyvlog.converters module

Functions for converting V-Log data into statuses.

pyvlog.converters.file_to_dataframe (*path_to_vlg*, *logged_types*=['detectie', 'externeSignaalgroep'])

Convert a file of v-log messages (each on a new line) to a dataframe of statuses.

Parameters

- **path_to_vlg** (*str*) – Path to file containing vlog messages.
- **logged_types** (*list*) – Message types (should match keys of `messagetypes.MESSAGE_TYPE_DICT`) to be logged. If empty list all types are logged.

Returns **df** – Dataframe of statuses.

Return type `pd.DataFrame`

pyvlog.converters.file_to_json (*path_to_vlg*, *path_to_json*, *logged_types*=['detectie', 'externeSignaalgroep'])

Convert a file of v-log messages (each on a new line) to a json file of statuses.

Parameters

- **path_to_vlg** (*str*) – Path to file containing vlog messages.
- **path_to_json** (*str*) – Path to json file to write to.
- **logged_types** (*list*) – Message types (should match keys of `messagetypes.MESSAGE_TYPE_DICT`) to be logged. If empty list all types are logged.

pyvlog.converters.file_to_list (*path_to_vlg*, *logged_types*=['detectie', 'externeSignaalgroep'])

Convert a file of v-log messages (each on a new line) to a list of statuses.

Parameters

- **path_to_vlg** (*str*) – Path to file containing v-log messages.
- **logged_types** (*list*) – Message types (should match keys of `messagetypes.MESSAGE_TYPE_DICT`) to be logged. If empty list all types are logged.

Returns **status_list** – List of statuses.

Return type `list`

```
pyvlog.converters.list_to_dataframe(messages, logged_types=['detectie', 'externeSignaalgroep'])
```

Convert a list of v-log messages to a dataframe of statuses.

Parameters

- **messages** (*list*) – List of vlog messages.
- **logged_types** (*list*) – Message types (should match keys of `messagetypes.MESSAGE_TYPE_DICT`) to be logged. If empty list all types are logged.

Returns **df** – Dataframe of statuses.

Return type `pd.DataFrame`

```
pyvlog.converters.list_to_json(messages, path_to_json, logged_types=['detectie', 'externeSignaalgroep'])
```

Convert a list of v-log messages to a json file of statuses.

Parameters

- **messages** (*list*) – List of v-log messages.
- **path_to_json** (*str*) – Path to json file to write to.
- **logged_types** (*list*) – Message types (should match keys of `messagetypes.MESSAGE_TYPE_DICT`) to be logged. If empty list all types are logged.

```
pyvlog.converters.list_to_list(messages, logged_types=['detectie', 'externeSignaalgroep'])
```

Convert a list of v-log messages to a list of statuses.

Parameters

- **messages** (*list*) – List of v-log messages.
- **logged_types** (*list*) – Message types (should match keys of `messagetypes.MESSAGE_TYPE_DICT`) to be logged. If empty list all types are logged.

Returns **status_list** – List of statuses.

Return type `list`

1.3 pyvlog.utils module

Utility functions for converting hex messages and logging statuses.

```
pyvlog.utils.flatten(d, parent_key="", sep='_')
```

Flatten a nested dict.

Parameters

- **d** (*dict*) – Dictionary to flatten.
- **parent_key** (*str*) – Parent key to prefix new key.
- **sep** (*str*) – Separation character(s) for combined keys.

Returns Flattened dict.

Return type dict

`pyvlog.utils.hex_string_to_bits(string)`

Convert a string of hex characters to bits.

Parameters `string` (*str*) – String of hex characters.

Returns `bits` – Number of sensors in status.

Return type str

`pyvlog.utils.parse_detection_data(string)`

Parse an element of detection data.

Parameters `string` (*str*) – Element of detection data.

Returns `out_concise` – Dictionary of internal phase status.

Return type dict

`pyvlog.utils.parse_instruction_data(string)`

Parse an element of instruction variable status.

Parameters `string` (*str*) – Element of instruction data.

Returns `out_concise` – Dictionary of internal phase status.

Return type dict

`pyvlog.utils.parse_internal_data(string)`

Parse an element of internal phase data.

Parameters `string` (*str*) – Element of internal phase data.

Returns `out_concise` – Dictionary of internal phase status.

Return type dict

`pyvlog.utils.parse_ovhd_data(string)`

Parse an element of ov/hulpdienst data.

Parameters `string` (*str*) – Element of internal phase data.

Returns `out_concise` – Dictionary of internal phase status.

Return type dict

pyvlog is a Python package for working with traffic device data from smart intersections that adhere to the [V-Log](#) messaging protocol. It can be used to convert sequences of device messages into time-stamped statuses for any combination of traffic devices at a smart intersection. [UltraJSON](#) provides fast writing of the status data to JSON format.

See the README in the [GitHub repo](#) for usage examples.

CHAPTER 2

Indices and tables

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

p

- `pyvlog.converters`, [2](#)
- `pyvlog.parsers`, [1](#)
- `pyvlog.utils`, [3](#)

F

`file_to_dataframe()` (in module `pyvlog.converters`), 2
`file_to_json()` (in module `pyvlog.converters`), 2
`file_to_list()` (in module `pyvlog.converters`), 2
`flatten()` (in module `pyvlog.utils`), 3

H

`hex_string_to_bits()` (in module `pyvlog.utils`), 4

L

`list_to_dataframe()` (in module `pyvlog.converters`), 3
`list_to_json()` (in module `pyvlog.converters`), 3
`list_to_list()` (in module `pyvlog.converters`), 3
`log_status()` (`pyvlog.parsers.VLogParser` method), 1
`log_status()` (`pyvlog.parsers.VLogParserToJson` method), 1
`log_status()` (`pyvlog.parsers.VLogParserToList` method), 2

P

`parse_detection_data()` (in module `pyvlog.utils`), 4
`parse_instruction_data()` (in module `pyvlog.utils`), 4
`parse_internal_data()` (in module `pyvlog.utils`), 4
`parse_message()` (`pyvlog.parsers.VLogParser` method), 1
`parse_ovhd_data()` (in module `pyvlog.utils`), 4
`pyvlog.converters` (module), 2
`pyvlog.parsers` (module), 1
`pyvlog.utils` (module), 3

V

`VLogParser` (class in `pyvlog.parsers`), 1
`VLogParserToJson` (class in `pyvlog.parsers`), 1
`VLogParserToList` (class in `pyvlog.parsers`), 2