# JSON Schema Validator Documentation

***Release 2.3.1***

**Zygmunt Krynicki**

February 06, 2016

Contents

CHAPTER 1

# About

This package contains an implementation of JSON Schema validator.

**Note:** This project is just getting started. While the code relatively feature-complete, rather well tested and used in production daily the *documentation* is lacking.

**Warning:** This implementation was based on the *second draft* of the specification A third draft was published on the 22nd Nov 2010. This draft introduced several important changes that are not yet implemented.

**Note:** Only a subset of schema features are currently supported. Unsupported features are detected and raise a NotImplementedError when you call `json_schema_validator.schema.Validator.validate()`

**See also:**

http://json-schema.org/ for details about the schema

CHAPTER 2

# Table of contents

## 2.1 Installation

### 2.1.1 Prerequisites

This package has the following prerequisites:

- simplejson
- versiontools

To run the test suite you will also need:

- testtools
- testscenarios

To build the documentation from source you will also need:

- sphinx

### 2.1.2 Installation Options

This package is being actively maintained and published in the Python Package Index. You can install it if you have pip tool using just one line:

```
pip install json-schema-validator
```

## 2.2 Usage

TODO

## 2.3 Code documentation

JSON Schema Validator

### 2.3.1 Errors module

Error classes used by this package.

**exception** `json_schema_validator.errors.`**SchemaError**
Exception raised when there is a problem with the schema itself.

**exception** `json_schema_validator.errors.`**ValidationError**(*message*, *new_message=None*, *object_expr=None*, *schema_expr=None*)
Exception raised on mismatch between the validated object and the schema.

**message**
Old and verbose message that contains less helpful message and lots of JSON data (deprecated).

**new_message**
Short and concise message about the problem.

**object_expr**
A JavaScript expression that evaluates to the object that failed to validate. The expression always starts with a root object called `'object'`.

**schema_expr**
A JavaScript expression that evaluates to the schema that was checked at the time validation failed. The expression always starts with a root object called `'schema'`.

### 2.3.2 Misc module

Stuff that does not belong anywhere else

### 2.3.3 Schema module

Helper module to work with raw JSON Schema

**class** `json_schema_validator.schema.`**Schema**(*json_obj*)
JSON schema object

**enum**
Enumeration of possible correct values.

*Must* be either `None` or a non-empty list of valid objects. The list *must not* contain duplicate elements.

**pattern**

---

**Note:** JSON schema specifications says that this value SHOULD follow the `EMCA 262/Perl 5` format. We cannot support this so we support python regular expressions instead. This is still valid but should be noted for clarity.

---

:returns None or compiled regular expression

**properties**
The properties property contains schema for each property in a dictionary.

The dictionary name is the property name. The dictionary value is the schema object itself.

**type**
Return the 'type' property of this schema.

The return value is always a list of correct JSON types. Correct JSON types are one of the pre-defined simple types or another schema object.

List of built-in simple types: * 'string' * 'number' * 'integer' * 'boolean' * 'object' * 'array' * 'any' (default)

### 2.3.4 Shortcuts module

One liners that make the code shorter

`json_schema_validator.shortcuts.`**`validate`**(*schema_text*, *data_text*)
Validate specified JSON text (data_text) with specified schema (schema text). Both are converted to JSON objects with `simplesjon.loads()`.

**Parameters**

- **schema_text** (`str`) – Text of the JSON schema to check against
- **data_text** (`str`) – Text of the JSON object to check

**Returns** Same as *json_schema_validator.validator.Validator.validate()*

**Raises** Whatever may be raised by simplejson (in particular `simplejson.decoder.JSONDecoderError`, a subclass of `ValueError`)

**Raises** Whatever may be raised by *json_schema_validator.validator.Validator.validate()*. In particular *json_schema_validator.errors.ValidationError* and *json_schema_validator.errors.SchemaError*

### 2.3.5 Validator module

Validator implementation

**class** `json_schema_validator.validator.`**`Validator`**
JSON Schema validator.

Can be used to validate any JSON document against a *json_schema_validator.schema.Schema*.

**classmethod** **validate**(*schema*, *obj*)
Validate specified JSON object obj with specified Schema instance schema.

**Parameters**

- **schema** (*json_schema_validator.schema.Schema*) – Schema to validate against
- **obj** – JSON object to validate

**Return type** bool

**Returns** True on success

**Raises**

- ***json_schema_validator.errors.ValidationError*** – if the object does not match schema.
- ***json_schema_validator.errors.SchemaError*** – if the schema itself is wrong.

## 2.4 Version History

### 2.4.1 Version 2.3.1

- Maintenance release

### 2.4.2 Version 2.3

- Add support for minimum, maximum, minLength, maxLength, minItems, maxItems validators

### 2.4.3 Version 2.2

- Maintenance release

### 2.4.4 Version 2.1

- Renamed from linaro-json to json-schema-validator and moved to github.
- Reorganized the package into more sub-modules
- Updated documentation a bit

### 2.4.5 Version 2.0.1

- Make the package installable via pip by using new versiontools
- Fix test suite to be agnostic to python's rounding of floats

### 2.4.6 Version 2.0

- New major release, incompatible with past releases
- Drop everything apart from schema validator as other elements have lost their significance
- Tweak requirements to support current Ubuntu LTS (10.04 aka Lucid)
- Refresh installation instructions to point to the new PPA, provide links to lp.net project page and pypi project page.

### 2.4.7 Version 1.2.3

- Change how setup.py finds the version of the code to make it pip install okay when simplejson is not installed yet.

### 2.4.8 Version 1.2.2

- Fix another problem with pip and versiontools (removed unconditional import of versiontools from __init__.py)

### 2.4.9 Version 1.2.1

- Fix installation problem with pip due to versiontools not being available when parsing initial setup.py

### 2.4.10 Version 1.2

- Change name to json-schema-validator
- Add dependency on versiontools
- Register on pypi
- Add ReST documentation

### 2.4.11 Version 1.1

- Add support for retrieving default values from the schema

### 2.4.12 Version 1.0

- Initial release

## 2.5 Hacking

The project is hosted on github (http://github.com/zyga/json-schema-validator/), feel free to fork it and propose a pull request.

### 2.5.1 Goals

The goal of this project is to construct a complete and fast implementation of the JSON Schema as defined by http://json-schema.org/.

JSON is powerful because of the simplicity. Unlike the baroque YAML it thrives on being easy to implement in any language, correctly, completely, with confidence and test. Python has a good built-in implementation of the serializer (decoder and encoder) but lacks more advanced tools. Working with JSON as a backend for your application, whether it's configuration, application data or envelope for your RPC system, almost always requires data validation and integrity checking.

### 2.5.2 Infrastructure

Github is used for:

- Hosting source code (in git)
- Reporting and tracking bugs

Launchpad.net is used for:

- Hosting source code (as bzr mirror)
- Packaging aid for Ubuntu

PyPi is used for:

- Hosting released tarballs
- Hosting built documentation

CHAPTER 3

# Indices and tables

- genindex
- modindex
- search

## j

## E



## J



## M



## N



## O



## P



## S



## T



## V