Schema config Documentation

Release 0.0.1

Cromefire_

Aug 23, 2019

Contents:

CHAPTER 1

Usage

1.1 Creation

The easiest way of using `schema_config` is to just load the schema from a string:

```
from schema_config import Configurator

schema = """
    {
      "type": "object",
      "properties": {
        "port": {
          "type": "number"
        }
      },
      required: ["port"]
    }
"""

config_gen = Configurator.from_string(schema)
```

Or combined with `resource_string()`:

```
from pkg_resources import resource_string

from schema_config import Configurator

schema = resource_string(__name__, "config.schema.json")

config_gen = Configurator.from_string(schema)
```

Or you can load the schema from a file:

```
from schema_config import Configurator

config_gen = Configurator.from_file("./config.schema.json")
```

The last option is to feed it a `dict` directly:

```
from schema_config import Configurator

schema = {
    "type": "object",
    "properties": {
        "port": {
            "type": "number"
        }
    },
    required: ["port"]
}

config_gen = Configurator(schema)
```

1.2 Plugins

Plugins can be added like this:

```
from schema_config import EnvPlugin

config_gen.add_plugin(EnvPlugin())
```

1.3 Config

Finally you can load the config like that:

```
config = config_gen.load_config()
```

Read about *Plugins* or the Configuration

CHAPTER 2

Plugins

2.1 YAML Plugin

Todo: Add description

2.2 JSON Plugin

Todo: Add description

2.3 Env Plugin

Todo: Add description

Todo: Description of interface

Schema config is an extensible configuration loader and validator, that works using jsonschema.

CHAPTER 3

Indices and tables

- genindex
- modindex
- search