semantic-version-check

Release 0.0.1

Konstantinos Lampridis

May 18, 2022

CONTENTS:

SEMANTIC-VERSION-CHECKER

Check if a ‘version’ is a valid Semantic Version.

Code: https://github.com/boromir674/semantic-version-check
Docs: https://semantic-version-check.readthedocs.io/en/master/
PyPI: https://pypi.org/project/semantic-version-check/
CI: https://github.com/boromir674/semantic-version-check/actions/

CHAPTER ONE

FEATURES

1. **semantic_version_check** *python package*
   a. CLI that can be usefull for quick checks, a script or in a CI pipeline
   b. Semantic Version format check, using Regular Expressions
2. Tested against multiple *platforms* and *python* versions

1.1 Development

Here are some useful notes related to doing development on this project.

1. **Test Suite**, using pytest, located in *tests* dir
2. **Parallel Execution** of Unit Tests, on multiple cpu's
3. **Documentation Pages**, hosted on *readthedocs* server, located in *docs* dir
4. **Automation**, using tox, driven by single *tox.ini* file
   a. **Code Coverage** measuring
   b. **Build Command**, using the build python package
   c. **Pypi Deploy Command**, supporting upload to both pypi.org and test.pypi.org servers
   d. **Type Check Command**, using mypy
   e. **Lint** *Check* and *Apply* commands, using isort and black
5. **CI Pipeline**, running on Github Actions, defined in *.github/*
   a. **Job Matrix**, spanning different *platform*'s and *python version*'s
      1. Platforms: *ubuntu-latest*, *macos-latest*
      2. Python Interpreters: *3.6*, *3.7*, *3.8*, *3.9*, *3.10*
   b. **Parallel Job** execution, generated from the *matrix*, that runs the *Test Suite*

CHAPTER

TWO

PREREQUISITES

You need to have *Python* installed.

CHAPTER

THREE

QUICKSTART

Using *pip* is the approved way for installing *semantic_version_check*.

```
python3 -m pip install semantic_version_check
```

One Use Case for the semantic_version_check is to invoke its cli, through a console
and do SemVer check on a single input string.

Open a console and run:

```
check-semantic-version 1.0.0
echo $?
echo "Exit code is 0 meaning operation succeeded"

check-semantic-version 1.3
echo $?
echo "Exit code is 1, meaning operation failed"
```

CHAPTER

FOUR

LICENSE

- GNU Affero General Public License v3.0

CHAPTER

FIVE

LICENSE

- Free software: GNU Affero General Public License v3.0

5.1 Introduction

This is **semantic-version-check**, a *Python Package* desinged to check whether
a given string is formatted according to Semantic Version.

This documentation aims to help people understand what are the package's features and to demonstrate
how to leverage them for their use cases.

5.2 Why this Package?

So, why would one opt for this Package?

It is useful in automation scenarios, since it can be run through its cli (eg using a *bash* shell) and "exits" with 0 or 1 accordingly.

It is useful whenever you are writing python and want to have a single source of truth for parsing and checking a string according to Semantic Version.

Specifically, the package exposes a regular expression (*from semantic_version_check import regex*) that can serve as a single source of truth to your python projects, whenever you need to do some SemVer parsing of string.

Also, it is easy to *install* using *pip*.

Finally, the package is well-tested against multiple Python Interpreter versions (from 3.6 to 3.10), tested on both *Linux* (Ubuntu) and *Darwin* (Macos) platforms.

This package's releases follow **Semantic Versioning** too :)

5.3 Usage

5.3.1 Installation

semantic_version_check is available on PyPI hence you can use *pip* to install it.

It is recommended to perform the installation in an isolated *python virtual environment* (env). You can create and activate an *env* using any tool of your preference (ie *virtualenv*, *venv*, *pyenv*).

Assuming you have 'activated' a *python virtual environment*:

```
python -m pip install semantic-version-check
```

5.3.2 Simple Use Case

One Use Case for the semantic_version_check is to invoke its cli, through a console
and do SemVer check on a single input string.

Open a console and run:

```
check-semantic-version 1.0.0
echo $?
echo "Exit code is 0 meaning operation succeeded"

check-semantic-version 1.3
echo $?
echo "Exit code is 1, meaning operation failed"
```

Note: this use case may be useful for a CI pipeline.

5.4 semantic_version_check

5.4.1 semantic_version_check package

Module contents

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX