

libsemver

1.0.0

Generated by Doxygen 1.8.10

Sat Apr 30 2016 02:53:15

Contents

1	Main Page	1
1.1	Introduction	1
1.2	Changelog	1
1.3	Available Bindings	1
1.4	libtool's versioning scheme	1
1.5	Reporting Bugs and Suggestions	2
2	History	3
2.1	1:0:0	3
3	Namespace Index	5
3.1	Namespace List	5
4	Class Index	7
4.1	Class List	7
5	File Index	9
5.1	File List	9
6	Namespace Documentation	11
6.1	semver Namespace Reference	11
6.1.1	Detailed Description	11
7	Class Documentation	13
7.1	semver_t Struct Reference	13
7.2	semver::version Class Reference	13
7.2.1	Detailed Description	14
7.2.2	Constructor & Destructor Documentation	14
7.2.2.1	version(const std::vector< unsigned int > versions, const std::string prerelease="","", const std::string metadata="")	14
7.2.3	Member Function Documentation	15
7.2.3.1	bump(unsigned int index) const	15
7.2.3.2	bump_major() const	15
7.2.3.3	bump_minor() const	15

7.2.3.4	bump_patch() const	15
7.2.3.5	from_string(std::string v)	15
7.2.3.6	get_metadata() const	16
7.2.3.7	get_prerelease() const	16
7.2.3.8	get_version() const	16
7.2.3.9	get_version(unsigned int index) const	16
7.2.3.10	is_release() const	16
7.2.3.11	operator<(const version &rh) const	17
7.2.3.12	operator==(const version &rh) const	17
7.2.3.13	operator>(const version &rh) const	17
7.2.3.14	str() const	17
7.2.3.15	strip_metadata() const	18
7.2.3.16	strip_prerelease() const	18
8	File Documentation	19
8.1	libsemver/c++/version.hpp File Reference	19
8.1.1	Detailed Description	19
8.2	libsemver/c/libsemver.cpp File Reference	20
8.2.1	Detailed Description	20
8.3	libsemver/c/libsemver.h File Reference	21
8.3.1	Detailed Description	21
Index		23

Chapter 1

Main Page

1.1 Introduction

`libsemver` is a C++ library with C bindings that provides the following functionality:

- Parsing a version number into an object.
- Comparing versions.
- Modifying versions object.

Versions are checked against *Semantic Versioning 2.0.0* (<http://semver.org/>).

1.2 Changelog

See the [History](#) page.

1.3 Available Bindings

`libsemver` is a C++ library with C bindings which makes it available to a wide range of programming languages. If a programming language has C bindings, then `libsemver` can be used from it. The C binding provides all the functionality provided by the C++ implementation and it can be used as a fallback solution when the C++ API cannot be used.

1.4 libtool's versioning scheme

`libtool`'s versioning scheme is described by three integers: `current:revision:age` where:

- `current` is the most recent interface number implemented by the library.
- `revision` is the implementation number of the current interface.
- `age` is the difference between the newest and the oldest interface that the library implements.

1.5 Reporting Bugs and Suggestions

If you find problems or have suggestions about this program or this manual, please report them as new issues in the official GitHub repository at <https://github.com/emcrisostomo/semver-utils>. Please, read the `CONTRIBUTING.md` file for detailed instructions on how to contribute to `fswatch`.

Chapter 2

History

2.1 1:0:0

- Initial release.

Chapter 3

Namespace Index

3.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

semver	Main namespace of <code>libsemver</code>	11
------------------------	--	----

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

semver_t	13
semver::version Class that represents a version number	13

Chapter 5

File Index

5.1 File List

Here is a list of all documented files with brief descriptions:

libsemver/ gettext.h	??
libsemver/ gettext_defs.h	??
libsemver/c++/ version.hpp	
Header of the semver::version class	19
libsemver/c/ errors.h	??
libsemver/c/ libsemver.cpp	
Main libsemver source file	20
libsemver/c/ libsemver.h	
Header of the libsemver library	21

Chapter 6

Namespace Documentation

6.1 semver Namespace Reference

Main namespace of `libsemver`.

Classes

- class [version](#)
Class that represents a version number.

Functions

- static `std::vector< unsigned int >` **parse_version** (const `std::string` &v)
- static void **match_prerelease** (const `std::string` &s)
- static void **match_metadata** (const `std::string` &s)
- static void **check_prerelease** (const `std::string` &s)
- static void **check_identifier** (const `std::string` &s)
- static const `std::string` **PRERELEASE_PATTERN** ("`[0-9A-Za-z-](\\.[0-9A-Za-z-]*)`")
- static const `std::string` **METADATA_PATTERN** ("`[0-9A-Za-z-](\\.[0-9A-Za-z-]*)`")

6.1.1 Detailed Description

Main namespace of `libsemver`.

Chapter 7

Class Documentation

7.1 semver_t Struct Reference

Public Attributes

- void * **ptr**

The documentation for this struct was generated from the following file:

- [libsemver/c/libsemver.h](#)

7.2 semver::version Class Reference

Class that represents a version number.

```
#include <version.hpp>
```

Public Member Functions

- **version** (const std::vector< unsigned int > versions, const std::string prerelease="", const std::string metadata="")
Constructs a [semver::version](#) instance with the specified parameters. The parameters must comply with Semantic Versioning 2.0.0.
- std::string **str** () const
Converts a version to its string representation.
- std::vector< unsigned int > **get_version** () const
Gets the version components.
- unsigned int **get_version** (unsigned int index) const
Gets the specified version component.
- std::string **get_prerelease** () const
Gets the prerelease string.
- std::string **get_metadata** () const
Gets the metadata string.
- **version bump_major** () const
Bumps the major version component.
- **version bump_minor** () const
Bumps the minor version component.

- [version bump_patch](#) () const
Bumps the patch version component.
- [version bump](#) (unsigned int index) const
Bumps the specified version component.
- [version strip_prerelease](#) () const
Strips the prerelease component.
- [version strip_metadata](#) () const
Strips the metadata component.
- bool [is_release](#) () const
Checks whether the instance is a release version.
- bool [operator==](#) (const [version](#) &rh) const
Checks two instances for equality.
- bool [operator<](#) (const [version](#) &rh) const
Compares two instances.
- bool [operator>](#) (const [version](#) &rh) const
Compares two instances.

Static Public Member Functions

- static [version from_string](#) (std::string v)
Constructs a [semver::version](#) instance from a std::string.

7.2.1 Detailed Description

Class that represents a version number.

This class represents a version number complying with [_Semantic Versioning 2.0.0](http://semver.org/) (<http://semver.org/>). As a supported extension this class allows version numbers to contain any number of components greater than 2.

Instances of this class are designed to be immutable.

7.2.2 Constructor & Destructor Documentation

- 7.2.2.1 `semver::version::version (const std::vector< unsigned int > versions, const std::string prerelease = "", const std::string metadata = "")`

Constructs a [semver::version](#) instance with the specified parameters. The parameters must comply with *Semantic Versioning 2.0.0*.

Parameters

<i>versions</i>	The version components. As a supported extension this class allows versions numbers to contain any number of components greater than 2.
<i>prerelease</i>	An optional prerelease string.
<i>metadata</i>	An optional metadata string.

Returns

A [semver::version](#) instance.

Exceptions

<code>std::invalid_argument</code>	if the parameters do not comply with <i>Semantic Versioning 2.0.0</i> .
------------------------------------	---

7.2.3 Member Function Documentation

7.2.3.1 `version semver::version::bump (unsigned int index) const`

Bumps the specified version component.

The component `index` is bumped and all components whose index is greater than `index` are set to 0. If `index` is greater than the current size `s` of the version number, then `index + 1 - s` components are added and set to 0.

Returns

A [semver::version](#) object representing the bumped version.

7.2.3.2 `version semver::version::bump_major () const`

Bumps the major version component.

This method uses `::bump()`;

Returns

A [semver::version](#) object containing the bumped version.

7.2.3.3 `version semver::version::bump_minor () const`

Bumps the minor version component.

This method uses `::bump()`;

Returns

A [semver::version](#) object representing the bumped version.

7.2.3.4 `version semver::version::bump_patch () const`

Bumps the patch version component.

This method uses `::bump()`;

Returns

A [semver::version](#) object representing the bumped version.

7.2.3.5 `version semver::version::from_string (std::string v) [static]`

Constructs a [semver::version](#) instance from a `std::string`.

The version number `v` must comply with *Semantic Versioning 2.0.0*. As a supported extension this class allows version numbers to contain any number of components greater than 2.

Parameters

<code>v</code>	The version number to parse.
----------------	------------------------------

Returns

A `semver::version` instance.

Exceptions

<code>std::invalid_argument</code>	if <code>v</code> is not a valid version number.
------------------------------------	--

7.2.3.6 `std::string semver::version::get_metadata () const`

Gets the metadata string.

Returns

The metadata string.

7.2.3.7 `std::string semver::version::get_prerelease () const`

Gets the prerelease string.

Returns

The prerelease string.

7.2.3.8 `std::vector< unsigned int > semver::version::get_version () const`

Gets the version components.

Returns

The version components.

7.2.3.9 `unsigned int semver::version::get_version (unsigned int index) const`

Gets the specified version component.

Returns

The specified version component, or 0 if the specified `index` does not exist.

7.2.3.10 `bool semver::version::is_release () const`

Checks whether the instance is a release version.

Returns

`true` if the instance represents a release version, `false` otherwise.

7.2.3.11 `bool semver::version::operator< (const version & rh) const`

Compares two instances.

An instance is less than another if its version number is less than the other's. Components are compared one by one, starting from the first. If the two components are equal, then the following component is considered. If the first component is less than the second, then `true` is returned, otherwise `false` is returned.

Parameters

<code>v</code>	The instance to be compared with.
----------------	-----------------------------------

Returns

`true` if this instance is less than `rh`, `false` otherwise.

7.2.3.12 `bool semver::version::operator==(const version & rh) const`

Checks two instances for equality.

Two instances are equal if and only if all its components are equal.

Parameters

<code>v</code>	The instance to be compared with.
----------------	-----------------------------------

Returns

`true` if `rh` is equal to this instance, `false` otherwise.

7.2.3.13 `bool semver::version::operator> (const version & rh) const`

Compares two instances.

An instance is greater than another if its version number is less than the other's. Components are compared one by one, starting from the first. If the two components are equal, then the following component is considered. If the first component is greater than the second, then `true` is returned, otherwise `false` is returned.

Parameters

<code>v</code>	The instance to be compared with.
----------------	-----------------------------------

Returns

`true` if this instance is greater than `rh`, `false` otherwise.

7.2.3.14 `std::string semver::version::str () const`

Converts a version to its string representation.

Returns

The string representation of this instance.

7.2.3.15 `version semver::version::strip_metadata () const`

Strips the metadata component.

Returns

A [semver::version](#) object representing the modified version.

7.2.3.16 `version semver::version::strip_prerelease () const`

Strips the prerelease component.

Returns

A [semver::version](#) object representing the modified version.

The documentation for this class was generated from the following files:

- [libsemver/c++/version.hpp](#)
- [libsemver/c++/version.cpp](#)

Chapter 8

File Documentation

8.1 libsemver/c++/version.hpp File Reference

Header of the [semver::version](#) class.

```
#include <vector>
#include <string>
```

Classes

- class [semver::version](#)
Class that represents a version number.

Namespaces

- [semver](#)
Main namespace of libsemver.

8.1.1 Detailed Description

Header of the [semver::version](#) class.

This header defines the [semver::version](#) class, the central type of the C++ API of libsemver library.

Copyright

Copyright (c) 2016 Enrico M. Crisostomo

License:

GNU General Public License v. 3.0

Author

Enrico M. Crisostomo

Version

1.0.0

8.2 libsemver/c/libsemver.cpp File Reference

Main `libsemver` source file.

```
#include "libsemver.h"
#include "errors.h"
#include "../c++/version.hpp"
#include "../gettext_defs.h"
#include <exception>
#include <cstdlib>
```

Functions

- static void **semver_set_last_error** (int err)
- static void **semver_reset_last_error** ()
- int **semver_last_error** ()
- [semver_t](#) * **semver_from_string** (const char *str)
- [semver_t](#) * **semver_create** (const unsigned int *c_vers, const unsigned long c_vers_num, const char *prerelease, const char *metadata)
- void **semver_free** ([semver_t](#) *ver)
- const char * **semver_str** ([semver_t](#) *ver)
- unsigned int * **semver_get_versions** ([semver_t](#) *ver)
- unsigned int **semver_get_version** ([semver_t](#) *ver, unsigned int index)
- const char * **semver_get_prerelease** ([semver_t](#) *ver)
- const char * **semver_get_metadata** ([semver_t](#) *ver)
- [semver_t](#) * **semver_bump_major** ([semver_t](#) *ver)
- [semver_t](#) * **semver_bump_minor** ([semver_t](#) *ver)
- [semver_t](#) * **semver_bump_patch** ([semver_t](#) *ver)
- [semver_t](#) * **semver_bump** ([semver_t](#) *ver, unsigned int index)
- [semver_t](#) * **semver_strip_prerelease** ([semver_t](#) *ver)
- [semver_t](#) * **semver_strip_metadata** ([semver_t](#) *ver)
- bool **semver_is_release** ([semver_t](#) *ver)
- bool **semver_equals** ([semver_t](#) *lh, [semver_t](#) *rh)
- bool **semver_is_less** ([semver_t](#) *lh, [semver_t](#) *rh)
- bool **semver_is_greater** ([semver_t](#) *lh, [semver_t](#) *rh)

Variables

- static `THREAD_LOCAL` int **last_error**

8.2.1 Detailed Description

Main `libsemver` source file.

Copyright

Copyright (c) 2016 Enrico M. Crisostomo

License:

GNU General Public License v. 3.0

Author

Enrico M. Crisostomo

Version

1.0.0

8.3 libsemver/c/libsemver.h File Reference

Header of the `libsemver` library.

```
#include <stdbool.h>
```

Classes

- struct [semver_t](#)

Typedefs

- typedef struct [semver_t](#) **semver_t**

Functions

- int **semver_last_error** ()
- [semver_t](#) * **semver_from_string** (const char *str)
- [semver_t](#) * **semver_create** (const unsigned int *c_vers, const unsigned long c_vers_num, const char *prerelease, const char *metadata)
- void **semver_free** ([semver_t](#) *ver)
- const char * **semver_str** ([semver_t](#) *ver)
- unsigned int * **semver_get_versions** ([semver_t](#) *ver)
- unsigned int **semver_get_version** ([semver_t](#) *ver, unsigned int index)
- const char * **semver_get_prerelease** ([semver_t](#) *ver)
- const char * **semver_get_metadata** ([semver_t](#) *ver)
- [semver_t](#) * **semver_bump_major** ([semver_t](#) *ver)
- [semver_t](#) * **semver_bump_minor** ([semver_t](#) *ver)
- [semver_t](#) * **semver_bump_patch** ([semver_t](#) *ver)
- [semver_t](#) * **semver_bump** ([semver_t](#) *ver, unsigned int index)
- [semver_t](#) * **semver_strip_prerelease** ([semver_t](#) *ver)
- [semver_t](#) * **semver_strip_metadata** ([semver_t](#) *ver)
- bool **semver_is_release** ([semver_t](#) *ver)
- bool **semver_equals** ([semver_t](#) *lh, [semver_t](#) *rh)
- bool **semver_is_less** ([semver_t](#) *lh, [semver_t](#) *rh)
- bool **semver_is_greater** ([semver_t](#) *lh, [semver_t](#) *rh)

8.3.1 Detailed Description

Header of the `libsemver` library.

This header file defines the API of the `libsemver` library.

Copyright

Copyright (c) 2016 Enrico M. Crisostomo

License:

GNU General Public License v. 3.0

Author

Enrico M. Crisostomo

Version

1.0.0

Index

- bump
 - semver::version, 15
- bump_major
 - semver::version, 15
- bump_minor
 - semver::version, 15
- bump_patch
 - semver::version, 15
- from_string
 - semver::version, 15
- get_metadata
 - semver::version, 16
- get_prerelease
 - semver::version, 16
- get_version
 - semver::version, 16
- is_release
 - semver::version, 16
- libsemver/c++/version.hpp, 19
- libsemver/c/libsemver.cpp, 20
- libsemver/c/libsemver.h, 21
- operator<
 - semver::version, 16
- operator>
 - semver::version, 17
- operator==
 - semver::version, 17
- semver, 11
- semver::version, 13
 - bump, 15
 - bump_major, 15
 - bump_minor, 15
 - bump_patch, 15
 - from_string, 15
 - get_metadata, 16
 - get_prerelease, 16
 - get_version, 16
 - is_release, 16
 - operator<, 16
 - operator>, 17
 - operator==, 17
 - str, 17
 - strip_metadata, 17
 - strip_prerelease, 18
 - version, 14
- semver_t, 13
- str
 - semver::version, 17
- strip_metadata
 - semver::version, 17
- strip_prerelease
 - semver::version, 18
- version
 - semver::version, 14