

Shared String View

0.1

Generated by Doxygen 1.9.1

1 Class Index	1
1.1 Class List	1
2 File Index	3
2.1 File List	3
3 Class Documentation	5
3.1 std::hash< Ghoti::shared_string_view > Struct Reference	5
3.1.1 Detailed Description	5
3.2 Ghoti::shared_string_view Class Reference	6
3.2.1 Constructor & Destructor Documentation	7
3.2.1.1 shared_string_view() [1/5]	7
3.2.1.2 shared_string_view() [2/5]	7
3.2.1.3 shared_string_view() [3/5]	8
3.2.1.4 shared_string_view() [4/5]	8
3.2.1.5 shared_string_view() [5/5]	8
3.2.2 Member Function Documentation	8
3.2.2.1 begin()	9
3.2.2.2 end()	9
3.2.2.3 length()	9
3.2.2.4 operator std::string_view()	9
3.2.2.5 operator+()	9
3.2.2.6 operator+=()	10
3.2.2.7 operator<=>()	10
3.2.2.8 operator==()	11
3.2.2.9 operator[]()	11
3.2.2.10 rbegin()	11
3.2.2.11 rend()	12
3.2.2.12 substr()	12
4 File Documentation	15
4.1 include/shared_string_view.hpp File Reference	15
4.1.1 Detailed Description	16
4.1.2 Function Documentation	16
4.1.2.1 operator<<()	16
4.2 src/shared_string_view.cpp File Reference	17
4.2.1 Detailed Description	17
4.3 test/test.cpp File Reference	17
4.3.1 Detailed Description	18
Index	19

Chapter 1

Class Index

1.1 Class List

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

std::hash< Ghoti::shared_string_view >	
Hashing function, consistent with <code>std::string_view</code>	5
Ghoti::shared_string_view	6

Chapter 2

File Index

2.1 File List

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

include/ shared_string_view.hpp	
Header file containing the definitions of the shared_string_view class	15
src/ shared_string_view.cpp	
Define the shared_string_view class	17
test/ test.cpp	
Test the shared_string_view behavior	17

Chapter 3

Class Documentation

3.1 `std::hash< Ghoti::shared_string_view >` Struct Reference

Hashing function, consistent with `std::string_view`.

```
#include <shared_string_view.hpp>
```

Public Member Functions

- `std::size_t operator() (const Ghoti::shared_string_view &ssv) const` noexcept

3.1.1 Detailed Description

Hashing function, consistent with `std::string_view`.

Parameters

<code>ssv</code>	The <code>shared_string_view</code> to be hashed.
------------------	---

Returns

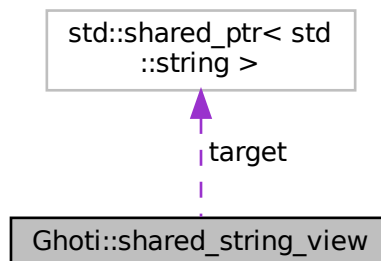
The hashed value.

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

- `include/shared_string_view.hpp`
- `src/shared_string_view.cpp`

3.2 Ghoti::shared_string_view Class Reference

Collaboration diagram for Ghoti::shared_string_view:



Public Member Functions

- [shared_string_view](#) ()
Default Constructor.
- [shared_string_view](#) (const char *s)
Constructor.
- [shared_string_view](#) (const char *s, size_t length)
Constructor.
- [shared_string_view](#) (const std::string &s)
Constructor.
- [operator std::string_view](#) () const
Provides a string_view variant of the [shared_string_view](#) object.
- size_t length () const
Return the length of the string represented by the view.
- [shared_string_view substr](#) (size_t offset, size_t length)
Calculate a substring based on the current [shared_string_view](#).
- bool [operator==](#) (const [Ghoti::shared_string_view](#) &ssv) const
Compare two [shared_string_view](#) objects.
- std::weak_ordering [operator<=>](#) (const [Ghoti::shared_string_view](#) &ssv) const
Perform a three-way comparison on two [shared_string_view](#) objects.
- [shared_string_view](#) & [operator+=](#) (const [Ghoti::shared_string_view](#) &rhs)
Perform a concatenation of the supplied string to the existing string view object and apply it to the existing string view object.
- [shared_string_view operator+](#) (const [Ghoti::shared_string_view](#) &rhs) const
Perform a concatenation of the supplied string to the existing string view object.
- std::string_view::const_iterator [begin](#) () const
Provide an iterator from the beginning of the view.
- std::string_view::const_iterator [end](#) () const
Provide an iterator pointing to the end of the view.
- std::string_view::const_reverse_iterator [rbegin](#) () const
Provide a reverse iterator from the end of the view.

- `std::string_view::const_reverse_iterator` [rend](#) () const
Provide a reverse iterator pointing to the beginning of the view.
- `char` [operator\[\]](#) (size_t pos) const
Perform an index operation into the target string.

Private Member Functions

- [shared_string_view](#) (bool)
Private constructor.

Private Attributes

- `std::shared_ptr< std::string >` [target](#)
The shared target string pointed to by this object.
- `size_t` [start](#)
The offset into the shared target string at which this view begins.
- `size_t` [len](#)
The length of the view.

3.2.1 Constructor & Destructor Documentation

3.2.1.1 [shared_string_view\(\)](#) [1/5]

```
shared_string_view::shared_string_view ( )
```

Default Constructor.

Initializes using an empty string.

3.2.1.2 [shared_string_view\(\)](#) [2/5]

```
shared_string_view::shared_string_view (
    const char * s )
```

Constructor.

Parameters

<code>s</code>	A C-string used to construct the string.
----------------	--

3.2.1.3 `shared_string_view()` [3/5]

```
shared_string_view::shared_string_view (
    const char * s,
    size_t length )
```

Constructor.

Parameters

<i>s</i>	A C-string used to construct the string.
<i>length</i>	The length of the C-string.

3.2.1.4 `shared_string_view()` [4/5]

```
shared_string_view::shared_string_view (
    const std::string & s )
```

Constructor.

Parameters

<i>s</i>	A String object used to construct the string.
----------	---

3.2.1.5 `shared_string_view()` [5/5]

```
Ghoti::shared_string_view::shared_string_view (
    bool ) [explicit], [private]
```

Private constructor.

This constructor is private because it will create an object whose target is not initialized, which should not be done in general.

Parameters

<i>ignore</i>	Only used to denote that this constructor is intentionally being called.
---------------	--

3.2.2 Member Function Documentation

3.2.2.1 begin()

```
string_view::const_iterator shared_string_view::begin ( ) const
```

Provide an iterator from the beginning of the view.

Returns

A forward iterator.

3.2.2.2 end()

```
string_view::const_iterator shared_string_view::end ( ) const
```

Provide an iterator pointing to the end of the view.

Returns

An ending iterator.

3.2.2.3 length()

```
size_t shared_string_view::length ( ) const
```

Return the length of the string represented by the view.

The shared string may be longer, but this is the length of the substring that this view represents.

Returns

The length of the string represented by the view.

3.2.2.4 operator std::string_view()

```
Ghoti::shared_string_view::operator std::string_view ( ) const
```

Provides a `string_view` variant of the [shared_string_view](#) object.

It is up to the programmer to ensure that the [shared_string_view](#) object remains in scope while the `string_view` is in use.

3.2.2.5 operator+()

```
shared_string_view shared_string_view::operator+ (
    const Ghoti::shared_string_view & rhs ) const
```

Perform a concatenation of the supplied string to the existing string view object.

Return a new string view.

Parameters

<i>rhs</i>	A string to be appended to the <code>shared_string_view</code> object.
------------	--

Returns

The new `shared_string_view` resulting from the concatenation.

3.2.2.6 `operator+=()`

```
shared_string_view & shared_string_view::operator+= (
    const Ghoti::shared_string_view & rhs )
```

Perform a concatenation of the supplied string to the existing string view object and apply it to the existing string view object.

If the target string can be appended to safely, then that will be done. Otherwise, a new internal string will be created.

Because this may modify the target string, all previously-provided `std::string_view` references will be invalidated. This is similar to the behavior of `std::string.cstr()`, in which modifying the string will invalidate the c-string pointer.

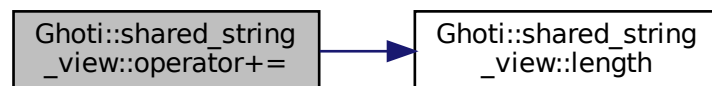
Parameters

<i>rhs</i>	A string to be appended to the <code>shared_string_view</code> object.
------------	--

Returns

The amended `shared_string_view` resulting from the concatenation.

Here is the call graph for this function:

3.2.2.7 `operator<=>()`

```
std::weak_ordering shared_string_view::operator<=> (
    const Ghoti::shared_string_view & ssv ) const
```

Perform a three-way comparison on two `shared_string_view` objects.

Parameters

<code>ssv</code>	The right hand side operator.
------------------	-------------------------------

Returns

A weak ordering indicator of the two objects.

3.2.2.8 operator==()

```
bool shared_string_view::operator== (
    const Ghoti::shared_string_view & ssv ) const
```

Compare two [shared_string_view](#) objects.

Parameters

<code>ssv</code>	The right hand side operator.
------------------	-------------------------------

Returns

True if the objects have equivalent values, false otherwise.

3.2.2.9 operator[]()

```
char shared_string_view::operator[] (
    size_t pos ) const
```

Perform an index operation into the target string.

Parameters

<code>pos</code>	The 0-based index position.
------------------	-----------------------------

Returns

The character at the position requested.

3.2.2.10 rbegin()

```
string_view::const_reverse_iterator shared_string_view::rbegin ( ) const
```

Provide a reverse iterator from the end of the view.

Returns

A reverse iterator.

3.2.2.11 `rend()`

```
string_view::const_reverse_iterator shared_string_view::rend ( ) const
```

Provide a reverse iterator pointing to the beginning of the view.

Returns

An reverse ending iterator.

3.2.2.12 `substr()`

```
shared_string_view shared_string_view::substr (
    size_t offset,
    size_t length )
```

Calculate a substring based on the current [shared_string_view](#).

If the substring is out of range, then an empty view will be provided. If the substring length requested is greater than what is available, then the returned substring will contain as many characters as possible, within the limits of the parent string view range.

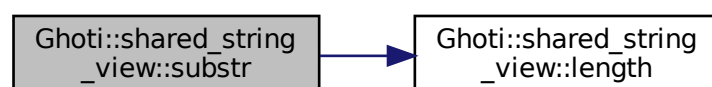
Parameters

<i>offset</i>	The 0-based offset from which the substring should start.
<i>length</i>	The length of the substring desired.

Returns

A new [shared_string_view](#) of the requested substring.

Here is the call graph for this function:



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

- [include/shared_string_view.hpp](#)
- [src/shared_string_view.cpp](#)

Chapter 4

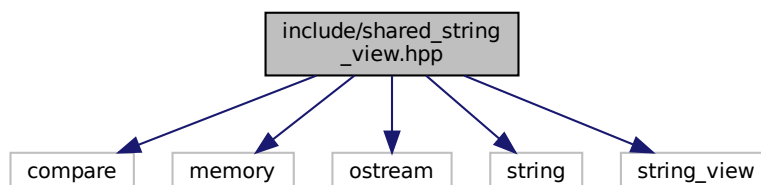
File Documentation

4.1 include/shared_string_view.hpp File Reference

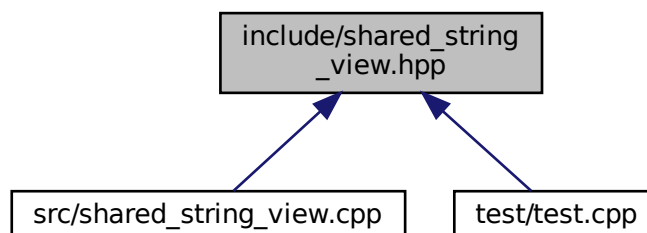
Header file containing the definitions of the shared_string_view class.

```
#include <compare>
#include <memory>
#include <ostream>
#include <string>
#include <string_view>
```

Include dependency graph for shared_string_view.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [Ghoti::shared_string_view](#)
- struct [std::hash< Ghoti::shared_string_view >](#)
Hashing function, consistent with `std::string_view`.

Functions

- `std::ostream & Ghoti::operator<< (std::ostream &out, const Ghoti::shared_string_view &ssv)`
Insertion operator.

4.1.1 Detailed Description

Header file containing the definitions of the `shared_string_view` class.

4.1.2 Function Documentation

4.1.2.1 `operator<<()`

```
ostream & Ghoti::operator<< (
    std::ostream & out,
    const Ghoti::shared\_string\_view & ssv )
```

Insertion operator.

Parameters

<i>out</i>	The output stream to be written to.
<i>ssv</i>	The <code>shared_string_view</code> to be inserted into the stream.

Returns

The output stream.

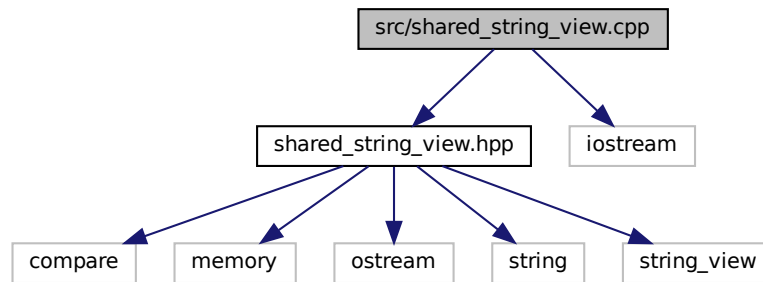
Here is the call graph for this function:



4.2 src/shared_string_view.cpp File Reference

Define the `shared_string_view` class.

```
#include "shared_string_view.hpp"
#include <iostream>
Include dependency graph for shared_string_view.cpp:
```



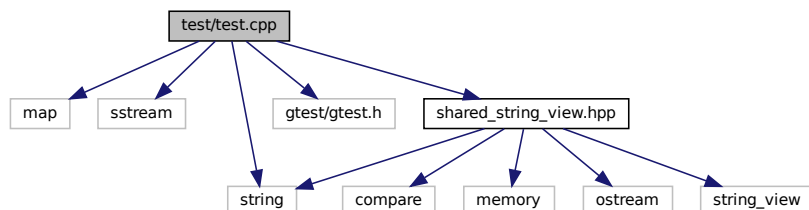
4.2.1 Detailed Description

Define the `shared_string_view` class.

4.3 test/test.cpp File Reference

Test the `shared_string_view` behavior.

```
#include <map>
#include <sstream>
#include <string>
#include <gtest/gtest.h>
#include "shared_string_view.hpp"
Include dependency graph for test.cpp:
```



Functions

- **TEST** (Constructor, Length)
- **TEST** (Operator, string_stream)
- **TEST** (Operator, Extraction)
- **TEST** (Operator, ThreeWayComparison)
- **TEST** (Operator, PlusEqual)
- **TEST** (Operator, Plus)
- **TEST** (Method, substr)
- **TEST** (Method, ForwardIterator)
- **TEST** (Method, ReverseIterator)
- **TEST** (Method, Index)
- **TEST** (Aux, Hash)
- int **main** (int argc, char **argv)

4.3.1 Detailed Description

Test the shared_string_view behavior.

Index

- begin
 - Ghoti::shared_string_view, [8](#)
- end
 - Ghoti::shared_string_view, [9](#)
- Ghoti::shared_string_view, [6](#)
 - begin, [8](#)
 - end, [9](#)
 - length, [9](#)
 - operator std::string_view, [9](#)
 - operator<=>, [10](#)
 - operator+, [9](#)
 - operator+=, [10](#)
 - operator==, [11](#)
 - operator[], [11](#)
 - rbegin, [11](#)
 - rend, [12](#)
 - shared_string_view, [7](#), [8](#)
 - substr, [12](#)
- include/shared_string_view.hpp, [15](#)
- length
 - Ghoti::shared_string_view, [9](#)
- operator std::string_view
 - Ghoti::shared_string_view, [9](#)
- operator<<
 - shared_string_view.hpp, [16](#)
- operator<=>
 - Ghoti::shared_string_view, [10](#)
- operator+
 - Ghoti::shared_string_view, [9](#)
- operator+=
 - Ghoti::shared_string_view, [10](#)
- operator==
 - Ghoti::shared_string_view, [11](#)
- operator[]
 - Ghoti::shared_string_view, [11](#)
- rbegin
 - Ghoti::shared_string_view, [11](#)
- rend
 - Ghoti::shared_string_view, [12](#)
- shared_string_view
 - Ghoti::shared_string_view, [7](#), [8](#)
- shared_string_view.hpp
 - operator<<, [16](#)
- src/shared_string_view.cpp, [17](#)
- std::hash< Ghoti::shared_string_view >, [5](#)
- substr
 - Ghoti::shared_string_view, [12](#)
- test/test.cpp, [17](#)