SFML

* [Main Page](http://docs.google.com/index.htm)
* [Modules](http://docs.google.com/modules.htm)
* [Classes](http://docs.google.com/annotated.htm)
* [Files](http://docs.google.com/files.htm)
* [Class List](http://docs.google.com/annotated.htm)
* [Class Index](http://docs.google.com/classes.htm)
* [Class Hierarchy](http://docs.google.com/hierarchy.htm)
* [Class Members](http://docs.google.com/functions.htm)
* **sf**
* [Time](http://docs.google.com/classsf_1_1Time.htm)

[Public Member Functions](#_gjdgxs) | [Static Public Attributes](#_30j0zll) | [Friends](#_1fob9te) | [Related Functions](#_3znysh7) | [List of all members](http://docs.google.com/classsf_1_1Time-members.htm)

sf::Time Class Reference

[System module](http://docs.google.com/group__system.htm)

Represents a time value. [More...](http://docs.google.com/classsf_1_1Time.htm#details)

#include <[Time.hpp](http://docs.google.com/Time_8hpp_source.htm)>

| Public Member Functions | |
| --- | --- |
|  | [Time](http://docs.google.com/classsf_1_1Time.htm#acba0cfbc49e3a09a22a8e079eb67a05c) () |
|  | Default constructor. |
|  | |
| float | [asSeconds](http://docs.google.com/classsf_1_1Time.htm#a7538140d095e48da9d7eee015dd455a9) () const |
|  | Return the time value as a number of seconds. |
|  | |
| Int32 | [asMilliseconds](http://docs.google.com/classsf_1_1Time.htm#a85e6deb41fa71896508ce0f64059a6ae) () const |
|  | Return the time value as a number of milliseconds. |
|  | |
| Int64 | [asMicroseconds](http://docs.google.com/classsf_1_1Time.htm#ae41a7e0ca73ceea771b3c150c12abdd2) () const |
|  | Return the time value as a number of microseconds. |
|  | |

| Static Public Attributes | |
| --- | --- |
| static const [Time](http://docs.google.com/classsf_1_1Time.htm) | [Zero](http://docs.google.com/classsf_1_1Time.htm#a8db127b632fa8da21550e7282af11fa0) |
|  | Predefined "zero" time value. |
|  | |

| Friends | |
| --- | --- |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | **seconds** (float) |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | **milliseconds** (Int32) |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | **microseconds** (Int64) |
|  | |

| Related Functions | |
| --- | --- |
| (Note that these are not member functions.) | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | [seconds](http://docs.google.com/classsf_1_1Time.htm#af9fc40a6c0e687e3430da1cf296385b1) (float amount) |
|  | Construct a time value from a number of seconds. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | [milliseconds](http://docs.google.com/classsf_1_1Time.htm#a9231f886d925a24d181c8dcfa6448d87) (Int32 amount) |
|  | Construct a time value from a number of milliseconds. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | [microseconds](http://docs.google.com/classsf_1_1Time.htm#a8a6ae28a1962198a69b92355649c6aa0) (Int64 amount) |
|  | Construct a time value from a number of microseconds. |
|  | |
| bool | [operator==](http://docs.google.com/classsf_1_1Time.htm#a9bbb2368cf012149f1001535a20c664a) ([Time](http://docs.google.com/classsf_1_1Time.htm) left, [Time](http://docs.google.com/classsf_1_1Time.htm) right) |
|  | Overload of == operator to compare two time values. |
|  | |
| bool | [operator!=](http://docs.google.com/classsf_1_1Time.htm#a3a142729f295af8b1baf2d8762bc39ac) ([Time](http://docs.google.com/classsf_1_1Time.htm) left, [Time](http://docs.google.com/classsf_1_1Time.htm) right) |
|  | Overload of != operator to compare two time values. |
|  | |
| bool | [operator<](http://docs.google.com/classsf_1_1Time.htm#a3bad89721b8c026e80082a7aa539f244) ([Time](http://docs.google.com/classsf_1_1Time.htm) left, [Time](http://docs.google.com/classsf_1_1Time.htm) right) |
|  | Overload of < operator to compare two time values. |
|  | |
| bool | [operator>](http://docs.google.com/classsf_1_1Time.htm#a9a472ce6d82aa0caf8e20af4a4b309f2) ([Time](http://docs.google.com/classsf_1_1Time.htm) left, [Time](http://docs.google.com/classsf_1_1Time.htm) right) |
|  | Overload of > operator to compare two time values. |
|  | |
| bool | [operator<=](http://docs.google.com/classsf_1_1Time.htm#aafb9de87ed6047956cd9487ab807371f) ([Time](http://docs.google.com/classsf_1_1Time.htm) left, [Time](http://docs.google.com/classsf_1_1Time.htm) right) |
|  | Overload of <= operator to compare two time values. |
|  | |
| bool | [operator>=](http://docs.google.com/classsf_1_1Time.htm#a158c5f9a6abf575651b7b2f6af8aedaa) ([Time](http://docs.google.com/classsf_1_1Time.htm) left, [Time](http://docs.google.com/classsf_1_1Time.htm) right) |
|  | Overload of >= operator to compare two time values. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | [operator-](http://docs.google.com/classsf_1_1Time.htm#acaead0aa2de9f82a548fcd8208a40f70) ([Time](http://docs.google.com/classsf_1_1Time.htm) right) |
|  | Overload of unary - operator to negate a time value. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | [operator+](http://docs.google.com/classsf_1_1Time.htm#a8249d3a28c8062c7c46cc426186f76c8) ([Time](http://docs.google.com/classsf_1_1Time.htm) left, [Time](http://docs.google.com/classsf_1_1Time.htm) right) |
|  | Overload of binary + operator to add two time values. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) & | [operator+=](http://docs.google.com/classsf_1_1Time.htm#a34b983deefecaf2725131771d54631e0) ([Time](http://docs.google.com/classsf_1_1Time.htm) &left, [Time](http://docs.google.com/classsf_1_1Time.htm) right) |
|  | Overload of binary += operator to add/assign two time values. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | [operator-](http://docs.google.com/classsf_1_1Time.htm#aebd95ec0cd0b2dc5d858e70149ccd136) ([Time](http://docs.google.com/classsf_1_1Time.htm) left, [Time](http://docs.google.com/classsf_1_1Time.htm) right) |
|  | Overload of binary - operator to subtract two time values. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) & | [operator-=](http://docs.google.com/classsf_1_1Time.htm#ae0a16136d024a44bbaa4ca49ac172c8f) ([Time](http://docs.google.com/classsf_1_1Time.htm) &left, [Time](http://docs.google.com/classsf_1_1Time.htm) right) |
|  | Overload of binary -= operator to subtract/assign two time values. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | [operator\*](http://docs.google.com/classsf_1_1Time.htm#ab891d4f3dbb454f6c1c484a7844bb581) ([Time](http://docs.google.com/classsf_1_1Time.htm) left, float right) |
|  | Overload of binary \* operator to scale a time value. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | [operator\*](http://docs.google.com/classsf_1_1Time.htm#a667d1568893f4e2520a223fa4e2b6ee2) ([Time](http://docs.google.com/classsf_1_1Time.htm) left, Int64 right) |
|  | Overload of binary \* operator to scale a time value. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | [operator\*](http://docs.google.com/classsf_1_1Time.htm#a61e3255c79b3d98a1a04ed8968a87863) (float left, [Time](http://docs.google.com/classsf_1_1Time.htm) right) |
|  | Overload of binary \* operator to scale a time value. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | [operator\*](http://docs.google.com/classsf_1_1Time.htm#a998a2ae6bd79e753bf9f4dea5b06370c) (Int64 left, [Time](http://docs.google.com/classsf_1_1Time.htm) right) |
|  | Overload of binary \* operator to scale a time value. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) & | [operator\*=](http://docs.google.com/classsf_1_1Time.htm#a3f7baa961b8961fc5e6a37dea7de10e3) ([Time](http://docs.google.com/classsf_1_1Time.htm) &left, float right) |
|  | Overload of binary \*= operator to scale/assign a time value. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) & | [operator\*=](http://docs.google.com/classsf_1_1Time.htm#ac883749b4e0a72c32e166ad802220539) ([Time](http://docs.google.com/classsf_1_1Time.htm) &left, Int64 right) |
|  | Overload of binary \*= operator to scale/assign a time value. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | [operator/](http://docs.google.com/classsf_1_1Time.htm#a67510d018fd010819ee075db2cbd004f) ([Time](http://docs.google.com/classsf_1_1Time.htm) left, float right) |
|  | Overload of binary / operator to scale a time value. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) | [operator/](http://docs.google.com/classsf_1_1Time.htm#a5f7b24dd13c0068d5cba678e1d5db9a6) ([Time](http://docs.google.com/classsf_1_1Time.htm) left, Int64 right) |
|  | Overload of binary / operator to scale a time value. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) & | [operator/=](http://docs.google.com/classsf_1_1Time.htm#ad513a413be41bc66feb0ff2b29d5f947) ([Time](http://docs.google.com/classsf_1_1Time.htm) &left, float right) |
|  | Overload of binary /= operator to scale/assign a time value. |
|  | |
| [Time](http://docs.google.com/classsf_1_1Time.htm) & | [operator/=](http://docs.google.com/classsf_1_1Time.htm#ac4b8df6ef282ee71808fd185f91490aa) ([Time](http://docs.google.com/classsf_1_1Time.htm) &left, Int64 right) |
|  | Overload of binary /= operator to scale/assign a time value. |
|  | |

## Detailed Description

Represents a time value.

[sf::Time](http://docs.google.com/classsf_1_1Time.htm) encapsulates a time value in a flexible way.

It allows to define a time value either as a number of seconds, milliseconds or microseconds. It also works the other way round: you can read a time value as either a number of seconds, milliseconds or microseconds.

By using such a flexible interface, the API doesn't impose any fixed type or resolution for time values, and let the user choose its own favorite representation.

[Time](http://docs.google.com/classsf_1_1Time.htm) values support the usual mathematical operations: you can add or subtract two times, multiply or divide a time by a number, compare two times, etc.

Since they represent a time span and not an absolute time value, times can also be negative.

Usage example:

[sf::Time](http://docs.google.com/classsf_1_1Time.htm) t1 = sf::seconds(0.1f);

Int32 milli = t1.[asMilliseconds](http://docs.google.com/classsf_1_1Time.htm#a85e6deb41fa71896508ce0f64059a6ae)(); // 100

[sf::Time](http://docs.google.com/classsf_1_1Time.htm) t2 = sf::milliseconds(30);

Int64 micro = t2.[asMicroseconds](http://docs.google.com/classsf_1_1Time.htm#ae41a7e0ca73ceea771b3c150c12abdd2)(); // 30000

[sf::Time](http://docs.google.com/classsf_1_1Time.htm) t3 = sf::microseconds(-800000);

float sec = t3.[asSeconds](http://docs.google.com/classsf_1_1Time.htm#a7538140d095e48da9d7eee015dd455a9)(); // -0.8

void update([sf::Time](http://docs.google.com/classsf_1_1Time.htm) elapsed)

{

position += speed \* elapsed.[asSeconds](http://docs.google.com/classsf_1_1Time.htm#a7538140d095e48da9d7eee015dd455a9)();

}

update(sf::milliseconds(100));

See Also[sf::Clock](http://docs.google.com/classsf_1_1Clock.htm)

Definition at line [40](http://docs.google.com/Time_8hpp_source.htm#l00040) of file [Time.hpp](http://docs.google.com/Time_8hpp_source.htm).

## Constructor & Destructor Documentation

| sf::Time::Time | ( |  | ) |  |
| --- | --- | --- | --- | --- |

Default constructor.

Sets the time value to zero.

## Member Function Documentation

| Int64 sf::Time::asMicroseconds | ( |  | ) | const |
| --- | --- | --- | --- | --- |

Return the time value as a number of microseconds.

Returns[Time](http://docs.google.com/classsf_1_1Time.htm) in microseconds See Also[asSeconds](http://docs.google.com/classsf_1_1Time.htm#a7538140d095e48da9d7eee015dd455a9), [asMilliseconds](http://docs.google.com/classsf_1_1Time.htm#a85e6deb41fa71896508ce0f64059a6ae)

| Int32 sf::Time::asMilliseconds | ( |  | ) | const |
| --- | --- | --- | --- | --- |

Return the time value as a number of milliseconds.

Returns[Time](http://docs.google.com/classsf_1_1Time.htm) in milliseconds See Also[asSeconds](http://docs.google.com/classsf_1_1Time.htm#a7538140d095e48da9d7eee015dd455a9), [asMicroseconds](http://docs.google.com/classsf_1_1Time.htm#ae41a7e0ca73ceea771b3c150c12abdd2)

| float sf::Time::asSeconds | ( |  | ) | const |
| --- | --- | --- | --- | --- |

Return the time value as a number of seconds.

Returns[Time](http://docs.google.com/classsf_1_1Time.htm) in seconds See Also[asMilliseconds](http://docs.google.com/classsf_1_1Time.htm#a85e6deb41fa71896508ce0f64059a6ae), [asMicroseconds](http://docs.google.com/classsf_1_1Time.htm#ae41a7e0ca73ceea771b3c150c12abdd2)

## Friends And Related Function Documentation

| | [Time](http://docs.google.com/classsf_1_1Time.htm) microseconds | ( | Int64 | *amount* | ) |  | | --- | --- | --- | --- | --- | --- | | related |
| --- | --- | --- | --- | --- | --- | --- | --- |

Construct a time value from a number of microseconds.

Parameters

| amount | Number of microseconds |
| --- | --- |

Returns[Time](http://docs.google.com/classsf_1_1Time.htm) value constructed from the amount of microseconds See Also[seconds](http://docs.google.com/classsf_1_1Time.htm#af9fc40a6c0e687e3430da1cf296385b1), [milliseconds](http://docs.google.com/classsf_1_1Time.htm#a9231f886d925a24d181c8dcfa6448d87)

| | [Time](http://docs.google.com/classsf_1_1Time.htm) milliseconds | ( | Int32 | *amount* | ) |  | | --- | --- | --- | --- | --- | --- | | related |
| --- | --- | --- | --- | --- | --- | --- | --- |

Construct a time value from a number of milliseconds.

Parameters

| amount | Number of milliseconds |
| --- | --- |

Returns[Time](http://docs.google.com/classsf_1_1Time.htm) value constructed from the amount of milliseconds See Also[seconds](http://docs.google.com/classsf_1_1Time.htm#af9fc40a6c0e687e3430da1cf296385b1), [microseconds](http://docs.google.com/classsf_1_1Time.htm#a8a6ae28a1962198a69b92355649c6aa0)

| | bool operator!= | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) | *left*, | | --- | --- | --- | --- | |  |  | [Time](http://docs.google.com/classsf_1_1Time.htm) | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of != operator to compare two time values.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a time) |

ReturnsTrue if both time values are different

| | [Time](http://docs.google.com/classsf_1_1Time.htm) operator\* | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) | *left*, | | --- | --- | --- | --- | |  |  | float | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary \* operator to scale a time value.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a number) |

Returns*left* multiplied by *right*

| | [Time](http://docs.google.com/classsf_1_1Time.htm) operator\* | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) | *left*, | | --- | --- | --- | --- | |  |  | Int64 | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary \* operator to scale a time value.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a number) |

Returns*left* multiplied by *right*

| | [Time](http://docs.google.com/classsf_1_1Time.htm) operator\* | ( | float | *left*, | | --- | --- | --- | --- | |  |  | [Time](http://docs.google.com/classsf_1_1Time.htm) | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary \* operator to scale a time value.

Parameters

| left | Left operand (a number) |
| --- | --- |
| right | Right operand (a time) |

Returns*left* multiplied by *right*

| | [Time](http://docs.google.com/classsf_1_1Time.htm) operator\* | ( | Int64 | *left*, | | --- | --- | --- | --- | |  |  | [Time](http://docs.google.com/classsf_1_1Time.htm) | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary \* operator to scale a time value.

Parameters

| left | Left operand (a number) |
| --- | --- |
| right | Right operand (a time) |

Returns*left* multiplied by *right*

| | [Time](http://docs.google.com/classsf_1_1Time.htm) & operator\*= | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) & | *left*, | | --- | --- | --- | --- | |  |  | float | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary \*= operator to scale/assign a time value.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a number) |

Returns*left* multiplied by *right*

| | [Time](http://docs.google.com/classsf_1_1Time.htm) & operator\*= | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) & | *left*, | | --- | --- | --- | --- | |  |  | Int64 | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary \*= operator to scale/assign a time value.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a number) |

Returns*left* multiplied by *right*

| | [Time](http://docs.google.com/classsf_1_1Time.htm) operator+ | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) | *left*, | | --- | --- | --- | --- | |  |  | [Time](http://docs.google.com/classsf_1_1Time.htm) | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary + operator to add two time values.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a time) |

ReturnsSum of the two times values

| | [Time](http://docs.google.com/classsf_1_1Time.htm) & operator+= | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) & | *left*, | | --- | --- | --- | --- | |  |  | [Time](http://docs.google.com/classsf_1_1Time.htm) | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary += operator to add/assign two time values.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a time) |

ReturnsSum of the two times values

| | [Time](http://docs.google.com/classsf_1_1Time.htm) operator- | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) | *right* | ) |  | | --- | --- | --- | --- | --- | --- | | related |
| --- | --- | --- | --- | --- | --- | --- | --- |

Overload of unary - operator to negate a time value.

Parameters

| right | Right operand (a time) |
| --- | --- |

ReturnsOpposite of the time value

| | [Time](http://docs.google.com/classsf_1_1Time.htm) operator- | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) | *left*, | | --- | --- | --- | --- | |  |  | [Time](http://docs.google.com/classsf_1_1Time.htm) | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary - operator to subtract two time values.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a time) |

ReturnsDifference of the two times values

| | [Time](http://docs.google.com/classsf_1_1Time.htm) & operator-= | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) & | *left*, | | --- | --- | --- | --- | |  |  | [Time](http://docs.google.com/classsf_1_1Time.htm) | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary -= operator to subtract/assign two time values.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a time) |

ReturnsDifference of the two times values

| | [Time](http://docs.google.com/classsf_1_1Time.htm) operator/ | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) | *left*, | | --- | --- | --- | --- | |  |  | float | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary / operator to scale a time value.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a number) |

Returns*left* divided by *right*

| | [Time](http://docs.google.com/classsf_1_1Time.htm) operator/ | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) | *left*, | | --- | --- | --- | --- | |  |  | Int64 | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary / operator to scale a time value.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a number) |

Returns*left* divided by *right*

| | [Time](http://docs.google.com/classsf_1_1Time.htm) & operator/= | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) & | *left*, | | --- | --- | --- | --- | |  |  | float | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary /= operator to scale/assign a time value.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a number) |

Returns*left* divided by *right*

| | [Time](http://docs.google.com/classsf_1_1Time.htm) & operator/= | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) & | *left*, | | --- | --- | --- | --- | |  |  | Int64 | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of binary /= operator to scale/assign a time value.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a number) |

Returns*left* divided by *right*

| | bool operator< | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) | *left*, | | --- | --- | --- | --- | |  |  | [Time](http://docs.google.com/classsf_1_1Time.htm) | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of < operator to compare two time values.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a time) |

ReturnsTrue if *left* is lesser than *right*

| | bool operator<= | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) | *left*, | | --- | --- | --- | --- | |  |  | [Time](http://docs.google.com/classsf_1_1Time.htm) | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of <= operator to compare two time values.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a time) |

ReturnsTrue if *left* is lesser or equal than *right*

| | bool operator== | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) | *left*, | | --- | --- | --- | --- | |  |  | [Time](http://docs.google.com/classsf_1_1Time.htm) | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of == operator to compare two time values.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a time) |

ReturnsTrue if both time values are equal

| | bool operator> | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) | *left*, | | --- | --- | --- | --- | |  |  | [Time](http://docs.google.com/classsf_1_1Time.htm) | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of > operator to compare two time values.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a time) |

ReturnsTrue if *left* is greater than *right*

| | bool operator>= | ( | [Time](http://docs.google.com/classsf_1_1Time.htm) | *left*, | | --- | --- | --- | --- | |  |  | [Time](http://docs.google.com/classsf_1_1Time.htm) | *right* | |  | ) |  |  | | related |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Overload of >= operator to compare two time values.

Parameters

| left | Left operand (a time) |
| --- | --- |
| right | Right operand (a time) |

ReturnsTrue if *left* is greater or equal than *right*

| | [Time](http://docs.google.com/classsf_1_1Time.htm) seconds | ( | float | *amount* | ) |  | | --- | --- | --- | --- | --- | --- | | related |
| --- | --- | --- | --- | --- | --- | --- | --- |

Construct a time value from a number of seconds.

Parameters

| amount | Number of seconds |
| --- | --- |

Returns[Time](http://docs.google.com/classsf_1_1Time.htm) value constructed from the amount of seconds See Also[milliseconds](http://docs.google.com/classsf_1_1Time.htm#a9231f886d925a24d181c8dcfa6448d87), [microseconds](http://docs.google.com/classsf_1_1Time.htm#a8a6ae28a1962198a69b92355649c6aa0)

## Member Data Documentation

| | const [Time](http://docs.google.com/classsf_1_1Time.htm) sf::Time::Zero | | --- | | static |
| --- | --- | --- |

Predefined "zero" time value.

Definition at line [85](http://docs.google.com/Time_8hpp_source.htm#l00085) of file [Time.hpp](http://docs.google.com/Time_8hpp_source.htm).

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

* [Time.hpp](http://docs.google.com/Time_8hpp_source.htm)

Copyright � Laurent Gomila  ::  Documentation generated by [doxygen](http://www.doxygen.org/)  ::