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)
* [File List](http://docs.google.com/files.htm)
* [include](http://docs.google.com/dir_f3190241575fd2bd132a392ae6942f4a.htm)
* [SFML](http://docs.google.com/dir_692f376662c82a26cfe4cfa3aceebe24.htm)
* [System](http://docs.google.com/dir_60c5c649f8df3b69a45a020d59f81335.htm)

Vector2.inl

1

2 //

3 // SFML - Simple and Fast Multimedia Library

4 // Copyright (C) 2007-2012 Laurent Gomila (laurent.gom@gmail.com)

5 //

6 // This software is provided 'as-is', without any express or implied warranty.

7 // In no event will the authors be held liable for any damages arising from the use of this software.

8 //

9 // Permission is granted to anyone to use this software for any purpose,

10 // including commercial applications, and to alter it and redistribute it freely,

11 // subject to the following restrictions:

12 //

13 // 1. The origin of this software must not be misrepresented;

14 // you must not claim that you wrote the original software.

15 // If you use this software in a product, an acknowledgment

16 // in the product documentation would be appreciated but is not required.

17 //

18 // 2. Altered source versions must be plainly marked as such,

19 // and must not be misrepresented as being the original software.

20 //

21 // 3. This notice may not be removed or altered from any source distribution.

22 //

24

25

27 template <typename T>

28 inline Vector2<T>::Vector2() :

29 x(0),

30 y(0)

31 {

32

33 }

34

35

37 template <typename T>

38 inline Vector2<T>::Vector2(T X, T Y) :

39 x(X),

40 y(Y)

41 {

42

43 }

44

45

47 template <typename T>

48 template <typename U>

49 inline Vector2<T>::Vector2(const Vector2<U>& vector) :

50 x(static\_cast<T>(vector.x)),

51 y(static\_cast<T>(vector.y))

52 {

53 }

54

55

57 template <typename T>

58 inline Vector2<T> [operator -](http://docs.google.com/classsf_1_1Time.htm#acaead0aa2de9f82a548fcd8208a40f70)(const Vector2<T>& right)

59 {

60  return Vector2<T>(-right.x, -right.y);

61 }

62

63

65 template <typename T>

66 inline Vector2<T>& [operator +=](http://docs.google.com/classsf_1_1Color.htm#af39790b2e677c9ab418787f5ff4583ef)(Vector2<T>& left, const Vector2<T>& right)

67 {

68  left.x += right.x;

69  left.y += right.y;

70

71  return left;

72 }

73

74

76 template <typename T>

77 inline Vector2<T>& [operator -=](http://docs.google.com/classsf_1_1Time.htm#ae0a16136d024a44bbaa4ca49ac172c8f)(Vector2<T>& left, const Vector2<T>& right)

78 {

79  left.x -= right.x;

80  left.y -= right.y;

81

82  return left;

83 }

84

85

87 template <typename T>

88 inline Vector2<T> [operator +](http://docs.google.com/classsf_1_1Color.htm#a0355ba6bfd2f83ffd8f8fafdca26cdd0)(const Vector2<T>& left, const Vector2<T>& right)

89 {

90  return Vector2<T>(left.x + right.x, left.y + right.y);

91 }

92

93

95 template <typename T>

96 inline Vector2<T> [operator -](http://docs.google.com/classsf_1_1Time.htm#acaead0aa2de9f82a548fcd8208a40f70)(const Vector2<T>& left, const Vector2<T>& right)

97 {

98  return Vector2<T>(left.x - right.x, left.y - right.y);

99 }

100

101

103 template <typename T>

104 inline Vector2<T> [operator \*](http://docs.google.com/classsf_1_1Color.htm#a1bae779fb49bb92dbf820a65e45a6602)(const Vector2<T>& left, T right)

105 {

106  return Vector2<T>(left.x \* right, left.y \* right);

107 }

108

109

111 template <typename T>

112 inline Vector2<T> [operator \*](http://docs.google.com/classsf_1_1Color.htm#a1bae779fb49bb92dbf820a65e45a6602)(T left, const Vector2<T>& right)

113 {

114  return Vector2<T>(right.x \* left, right.y \* left);

115 }

116

117

119 template <typename T>

120 inline Vector2<T>& [operator \*=](http://docs.google.com/classsf_1_1Color.htm#a7d1ea2b9bd5dbe29bb2e54feba9b4b38)(Vector2<T>& left, T right)

121 {

122  left.x \*= right;

123  left.y \*= right;

124

125  return left;

126 }

127

128

130 template <typename T>

131 inline Vector2<T> [operator /](http://docs.google.com/classsf_1_1Time.htm#a67510d018fd010819ee075db2cbd004f)(const Vector2<T>& left, T right)

132 {

133  return Vector2<T>(left.x / right, left.y / right);

134 }

135

136

138 template <typename T>

139 inline Vector2<T>& [operator /=](http://docs.google.com/classsf_1_1Time.htm#ad513a413be41bc66feb0ff2b29d5f947)(Vector2<T>& left, T right)

140 {

141  left.x /= right;

142  left.y /= right;

143

144  return left;

145 }

146

147

149 template <typename T>

150 inline bool [operator ==](http://docs.google.com/classsf_1_1Color.htm#a2adc3f68860f7aa5e4d7c79dcbb31d30)(const Vector2<T>& left, const Vector2<T>& right)

151 {

152  return (left.x == right.x) && (left.y == right.y);

153 }

154

155

157 template <typename T>

158 inline bool [operator !=](http://docs.google.com/classsf_1_1Color.htm#a394c3495753c4b17f9cd45556ef00b8c)(const Vector2<T>& left, const Vector2<T>& right)

159 {

160  return (left.x != right.x) || (left.y != right.y);

161 }

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