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.hpp

1

2 //

3 // SFML - Simple and Fast Multimedia Library

4 // Copyright (C) 2007-2013 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 #ifndef SFML\_VECTOR2\_HPP

26 #define SFML\_VECTOR2\_HPP

27

28

29 namespace sf

30 {

36 template <typename T>

[37](http://docs.google.com/classsf_1_1Vector2.htm) class [Vector2](http://docs.google.com/classsf_1_1Vector2.htm)

38 {

39 public :

40

47  [Vector2](http://docs.google.com/classsf_1_1Vector2.htm#a58c32383b5291380db4b43a289f75988)();

48

56  [Vector2](http://docs.google.com/classsf_1_1Vector2.htm#a58c32383b5291380db4b43a289f75988)(T X, T Y);

57

69  template <typename U>

70  explicit [Vector2](http://docs.google.com/classsf_1_1Vector2.htm#a58c32383b5291380db4b43a289f75988)(const [Vector2<U>](http://docs.google.com/classsf_1_1Vector2.htm)& vector);

71

73  // Member data

[75](http://docs.google.com/classsf_1_1Vector2.htm#a1e6ad77fa155f3753bfb92699bd28141)  T [x](http://docs.google.com/classsf_1_1Vector2.htm#a1e6ad77fa155f3753bfb92699bd28141);

[76](http://docs.google.com/classsf_1_1Vector2.htm#a420f2481b015f4eb929c75f2af564299)  T [y](http://docs.google.com/classsf_1_1Vector2.htm#a420f2481b015f4eb929c75f2af564299);

77 };

78

88 template <typename T>

89 [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm) operator -(const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& right);

90

104 template <typename T>

105 [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& operator +=([Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& left, const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& right);

106

120 template <typename T>

121 [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& operator -=([Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& left, const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& right);

122

133 template <typename T>

134 [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm) operator +(const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& left, const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& right);

135

146 template <typename T>

147 [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm) operator -(const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& left, const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& right);

148

159 template <typename T>

160 [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm) operator \*(const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& left, T right);

161

172 template <typename T>

173 [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm) operator \*(T left, const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& right);

174

188 template <typename T>

189 [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& operator \*=([Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& left, T right);

190

201 template <typename T>

202 [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm) operator /(const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& left, T right);

203

217 template <typename T>

218 [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& operator /=([Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& left, T right);

219

232 template <typename T>

233 bool operator ==(const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& left, const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& right);

234

247 template <typename T>

248 bool operator !=(const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& left, const [Vector2<T>](http://docs.google.com/classsf_1_1Vector2.htm)& right);

249

250 #include <SFML/System/Vector2.inl>

251

252 // Define the most common types

253 typedef [Vector2<int>](http://docs.google.com/classsf_1_1Vector2.htm) [Vector2i](http://docs.google.com/classsf_1_1Vector2.htm);

254 typedef [Vector2<unsigned int>](http://docs.google.com/classsf_1_1Vector2.htm) [Vector2u](http://docs.google.com/classsf_1_1Vector2.htm);

255 typedef [Vector2<float>](http://docs.google.com/classsf_1_1Vector2.htm) [Vector2f](http://docs.google.com/classsf_1_1Vector2.htm);

256

257 } // namespace sf

258

259

260 #endif // SFML\_VECTOR2\_HPP

261

262

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