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)
* [Graphics](http://docs.google.com/dir_aaa96c3797a59111c2945d0d638ce5cf.htm)

Text.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\_TEXT\_HPP

26 #define SFML\_TEXT\_HPP

27

29 // Headers

31 #include <SFML/Graphics/Export.hpp>

32 #include <SFML/Graphics/Drawable.hpp>

33 #include <SFML/Graphics/Transformable.hpp>

34 #include <SFML/Graphics/Font.hpp>

35 #include <SFML/Graphics/Rect.hpp>

36 #include <SFML/Graphics/VertexArray.hpp>

37 #include <SFML/System/String.hpp>

38 #include <string>

39 #include <vector>

40

41

42 namespace sf

43 {

[48](http://docs.google.com/classsf_1_1Text.htm) class SFML\_GRAPHICS\_API [Text](http://docs.google.com/classsf_1_1Text.htm) : public [Drawable](http://docs.google.com/classsf_1_1Drawable.htm), public [Transformable](http://docs.google.com/classsf_1_1Transformable.htm)

49 {

50 public :

51

[56](http://docs.google.com/classsf_1_1Text.htm#aa8add4aef484c6e6b20faff07452bd82)  enum [Style](http://docs.google.com/classsf_1_1Text.htm#aa8add4aef484c6e6b20faff07452bd82)

57  {

[58](http://docs.google.com/classsf_1_1Text.htm#aa8add4aef484c6e6b20faff07452bd82a2af9ae5e1cda126570f744448e0caa32)  Regular = 0,

[59](http://docs.google.com/classsf_1_1Text.htm#aa8add4aef484c6e6b20faff07452bd82af1b47f98fb1e10509ba930a596987171)  Bold = 1 << 0,

[60](http://docs.google.com/classsf_1_1Text.htm#aa8add4aef484c6e6b20faff07452bd82aee249eb803848723c542c2062ebe69d8)  Italic = 1 << 1,

[61](http://docs.google.com/classsf_1_1Text.htm#aa8add4aef484c6e6b20faff07452bd82a664bd143f92b6e8c709d7f788e8b20df)  Underlined = 1 << 2

62  };

63

70  [Text](http://docs.google.com/classsf_1_1Text.htm)();

71

80  [Text](http://docs.google.com/classsf_1_1Text.htm)(const [String](http://docs.google.com/classsf_1_1String.htm)& string, const [Font](http://docs.google.com/classsf_1_1Font.htm)& font, unsigned int characterSize = 30);

81

101  void setString(const [String](http://docs.google.com/classsf_1_1String.htm)& string);

102

118  void setFont(const [Font](http://docs.google.com/classsf_1_1Font.htm)& font);

119

130  void setCharacterSize(unsigned int size);

131

144  void setStyle(Uint32 style);

145

156  void setColor(const [Color](http://docs.google.com/classsf_1_1Color.htm)& color);

157

175  const [String](http://docs.google.com/classsf_1_1String.htm)& getString() const;

176

189  const [Font](http://docs.google.com/classsf_1_1Font.htm)\* getFont() const;

190

199  unsigned int getCharacterSize() const;

200

209  Uint32 getStyle() const;

210

219  const [Color](http://docs.google.com/classsf_1_1Color.htm)& getColor() const;

220

236  [Vector2f](http://docs.google.com/classsf_1_1Vector2.htm) findCharacterPos(std::size\_t index) const;

237

250  [FloatRect](http://docs.google.com/classsf_1_1Rect.htm) getLocalBounds() const;

251

264  [FloatRect](http://docs.google.com/classsf_1_1Rect.htm) getGlobalBounds() const;

265

266 private :

267

275  virtual void draw([RenderTarget](http://docs.google.com/classsf_1_1RenderTarget.htm)& target, [RenderStates](http://docs.google.com/classsf_1_1RenderStates.htm) states) const;

276

281  void updateGeometry();

282

284  // Member data

286  [String](http://docs.google.com/classsf_1_1String.htm) m\_string;

287  const [Font](http://docs.google.com/classsf_1_1Font.htm)\* m\_font;

288  unsigned int m\_characterSize;

289  Uint32 m\_style;

290  [Color](http://docs.google.com/classsf_1_1Color.htm) m\_color;

291  [VertexArray](http://docs.google.com/classsf_1_1VertexArray.htm) m\_vertices;

292  [FloatRect](http://docs.google.com/classsf_1_1Rect.htm) m\_bounds;

293 };

294

295 } // namespace sf

296

297

298 #endif // SFML\_TEXT\_HPP

299

300

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