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)

Shape.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\_SHAPE\_HPP

26 #define SFML\_SHAPE\_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/VertexArray.hpp>

35 #include <SFML/System/Vector2.hpp>

36

37

38 namespace sf

39 {

[44](http://docs.google.com/classsf_1_1Shape.htm) class SFML\_GRAPHICS\_API [Shape](http://docs.google.com/classsf_1_1Shape.htm) : public [Drawable](http://docs.google.com/classsf_1_1Drawable.htm), public [Transformable](http://docs.google.com/classsf_1_1Transformable.htm)

45 {

46 public :

47

52  virtual ~[Shape](http://docs.google.com/classsf_1_1Shape.htm)();

53

74  void setTexture(const [Texture](http://docs.google.com/classsf_1_1Texture.htm)\* texture, bool resetRect = false);

75

88  void setTextureRect(const [IntRect](http://docs.google.com/classsf_1_1Rect.htm)& rect);

89

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

106

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

118

132  void setOutlineThickness(float thickness);

133

146  const [Texture](http://docs.google.com/classsf_1_1Texture.htm)\* getTexture() const;

147

156  const [IntRect](http://docs.google.com/classsf_1_1Rect.htm)& getTextureRect() const;

157

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

167

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

177

186  float getOutlineThickness() const;

187

196  virtual unsigned int getPointCount() const = 0;

197

210  virtual [Vector2f](http://docs.google.com/classsf_1_1Vector2.htm) getPoint(unsigned int index) const = 0;

211

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

225

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

239

240 protected :

241

246  [Shape](http://docs.google.com/classsf_1_1Shape.htm)();

247

256  void update();

257

258 private :

259

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

268

273  void updateFillColors();

274

279  void updateTexCoords();

280

285  void updateOutline();

286

291  void updateOutlineColors();

292

293 private :

294

296  // Member data

298  const [Texture](http://docs.google.com/classsf_1_1Texture.htm)\* m\_texture;

299  [IntRect](http://docs.google.com/classsf_1_1Rect.htm) m\_textureRect;

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

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

302  float m\_outlineThickness;

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

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

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

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

307 };

308

309 } // namespace sf

310

311

312 #endif // SFML\_SHAPE\_HPP

313

314

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