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)

Shader.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\_SHADER\_HPP

26 #define SFML\_SHADER\_HPP

27

29 // Headers

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

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

33 #include <SFML/Graphics/Color.hpp>

34 #include <SFML/Window/GlResource.hpp>

35 #include <SFML/System/NonCopyable.hpp>

36 #include <SFML/System/Vector2.hpp>

37 #include <SFML/System/Vector3.hpp>

38 #include <map>

39 #include <string>

40

41

42 namespace sf

43 {

44 class InputStream;

45 class Texture;

46

[51](http://docs.google.com/classsf_1_1Shader.htm) class SFML\_GRAPHICS\_API [Shader](http://docs.google.com/classsf_1_1Shader.htm) : [GlResource](http://docs.google.com/classsf_1_1GlResource.htm), [NonCopyable](http://docs.google.com/classsf_1_1NonCopyable.htm)

52 {

53 public :

54

[59](http://docs.google.com/classsf_1_1Shader.htm#afaa1aa65e5de37b74d047da9def9f9b3)  enum [Type](http://docs.google.com/classsf_1_1Shader.htm#afaa1aa65e5de37b74d047da9def9f9b3)

60  {

[61](http://docs.google.com/classsf_1_1Shader.htm#afaa1aa65e5de37b74d047da9def9f9b3a8718008f827eb32e29bbdd1791c62dce)  [Vertex](http://docs.google.com/classsf_1_1Shader.htm#afaa1aa65e5de37b74d047da9def9f9b3a8718008f827eb32e29bbdd1791c62dce),

[62](http://docs.google.com/classsf_1_1Shader.htm#afaa1aa65e5de37b74d047da9def9f9b3ace6e88eec3a56b2e55ee3c8e64e9b89a)  Fragment

63  };

64

[70](http://docs.google.com/structsf_1_1Shader_1_1CurrentTextureType.htm)  struct [CurrentTextureType](http://docs.google.com/structsf_1_1Shader_1_1CurrentTextureType.htm) {};

71  static [CurrentTextureType](http://docs.google.com/structsf_1_1Shader_1_1CurrentTextureType.htm) CurrentTexture;

72

73 public :

74

81  [Shader](http://docs.google.com/classsf_1_1Shader.htm)();

82

87  ~[Shader](http://docs.google.com/classsf_1_1Shader.htm)();

88

108  bool loadFromFile(const std::string& filename, [Type](http://docs.google.com/classsf_1_1Shader.htm#afaa1aa65e5de37b74d047da9def9f9b3) type);

109

129  bool loadFromFile(const std::string& vertexShaderFilename, const std::string& fragmentShaderFilename);

130

149  bool loadFromMemory(const std::string& shader, [Type](http://docs.google.com/classsf_1_1Shader.htm#afaa1aa65e5de37b74d047da9def9f9b3) type);

150

170  bool loadFromMemory(const std::string& vertexShader, const std::string& fragmentShader);

171

190  bool loadFromStream([InputStream](http://docs.google.com/classsf_1_1InputStream.htm)& stream, [Type](http://docs.google.com/classsf_1_1Shader.htm#afaa1aa65e5de37b74d047da9def9f9b3) type);

191

211  bool loadFromStream([InputStream](http://docs.google.com/classsf_1_1InputStream.htm)& vertexShaderStream, [InputStream](http://docs.google.com/classsf_1_1InputStream.htm)& fragmentShaderStream);

212

232  void setParameter(const std::string& name, float x);

233

254  void setParameter(const std::string& name, float x, float y);

255

277  void setParameter(const std::string& name, float x, float y, float z);

278

301  void setParameter(const std::string& name, float x, float y, float z, float w);

302

322  void setParameter(const std::string& name, const [Vector2f](http://docs.google.com/classsf_1_1Vector2.htm)& vector);

323

343  void setParameter(const std::string& name, const [Vector3f](http://docs.google.com/classsf_1_1Vector3.htm)& vector);

344

370  void setParameter(const std::string& name, const [Color](http://docs.google.com/classsf_1_1Color.htm)& color);

371

393  void setParameter(const std::string& name, const [sf::Transform](http://docs.google.com/classsf_1_1Transform.htm)& transform);

394

425  void setParameter(const std::string& name, const [Texture](http://docs.google.com/classsf_1_1Texture.htm)& texture);

426

448  void setParameter(const std::string& name, [CurrentTextureType](http://docs.google.com/structsf_1_1Shader_1_1CurrentTextureType.htm));

449

471  static void bind(const [Shader](http://docs.google.com/classsf_1_1Shader.htm)\* shader);

472

483  static bool isAvailable();

484

485 private :

486

499  bool compile(const char\* vertexShaderCode, const char\* fragmentShaderCode);

500

508  void bindTextures() const;

509

511  // Types

513  typedef std::map<int, const Texture\*> TextureTable;

514

516  // Member data

518  unsigned int m\_shaderProgram;

519  int m\_currentTexture;

520  TextureTable m\_textures;

521 };

522

523 } // namespace sf

524

525

526 #endif // SFML\_SHADER\_HPP

527

528

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