FTGL

2.1.3∼rc5

Generated by Doxygen 1.8.1.1

Sat Oct 13 2012 20:48:25

Contents

ı	FIG	GL User Guide							
	1.1	Introdu	ction	3					
	1.2	Docum	entation	3					
	1.3	1.3 Additional information							
2	Freq	uently A	Asked Questions	5					
	2.1	FAQ .		5					
		2.1.1	When I try to compile %FTGL it complains about a missing file from the include: #include <ft2build.h></ft2build.h>	5					
		2.1.2	Is it possible to map a font to a "unit" size? My application relies on the fonts being a certain "physical" height (in OpenGL coordinate space) rather than a point size in display space. Any thoughts/suggestions?	5					
3	Proj	ects usi	ng FTGL	7					
	3.1	%FTGI	 L language bindings	7					
		3.1.1	%FTGL#	7					
		3.1.2	GIGuiA	7					
		3.1.3	Ruby %FTGL	7					
		3.1.4	PyFTGL	7					
	3.2	Project	s currently using %FTGL	7					
		3.2.1	Agent World	7					
		3.2.2	Amaltheia	8					
		3.2.3	Armagetron Advanced	8					
		3.2.4	Audicle	8					
		3.2.5	Battlestar T.U.X	8					
		3.2.6	BJS	8					
		3.2.7	Blender	8					
		3.2.8	Breve	8					
		3.2.9	BZFlag	8					
		3.2.10	Capture The Flag	9					
		3.2.11	Cello	9					
		3.2.12	Chimera	9					
		3.2.13	Cinepaint	9					

		pt3em		
		3.2.14	Duel	9
		3.2.15	Empty Clip	9
		3.2.16	Freebox	9
		3.2.17	Gem	9
		3.2.18	GLMayan	10
		3.2.19	Glover	10
		3.2.20	lvf++	10
		3.2.21	Jahshaka	10
		3.2.22	Karaoke FX	10
		3.2.23	Libinstrudeo	10
		3.2.24	Light Speed!	10
		3.2.25	MySQL GUI Tools	10
		3.2.26	OctPlot	11
		3.2.27	Open ActiveWrl	11
		3.2.28	OpenEaagles	11
		3.2.29	OpenGC	11
		3.2.30	OpenSG	11
		3.2.31	Panthera	11
		3.2.32	Planet Penguin Racer	11
		3.2.33	projectM	11
		3.2.34	Puzzle Bobble 3D	12
		3.2.35	ROOT	12
		3.2.36	SCIRun	12
		3.2.37	TINE	12
		3.2.38	Tiny Planet	12
		3.2.39	Truevision	12
		3.2.40	Tulip	12
		3.2.41	Ubit	12
		3.2.42	VRS	12
		3.2.43	VTK	13
		3.2.44	XLock	13
	3.3	Projects	s that used to use %FTGL	13
		3.3.1	GNU Backgammon	13
		3.3.2	OpenSceneGraph	13
		3.3.3	Teddy	13
		3.3.4	VigiPac	13
4	FTGI	L tutoria	al	15
4			ato use %FTGL	15
	4.1	oiai III (A TO USE 701 TOL	- 10

4.1

		pt3em				
	4.2	Choos	ing a font t	type	15	
		4.2.1	Raster fo	onts	15	
		4.2.2	Vector fo	onts	15	
		4.2.3	Textured	fonts	16	
	4.3	Create	font object	cts	16	
		4.3.1	in C		16	
		4.3.2	in C++ .		16	
	4.4	More for	ont comma	ands	17	
		4.4.1	Font met	trics	17	
		4.4.2	Specifyir	ng a character map encoding	17	
	4.5	Sample	e font man	nager class	18	
5	Nam	espace	Docume	ntation	21	
•	5.1	•		ce Reference	21	
	0	5.1.1	•	ation Type Documentation	21	
			5.1.1.1	RenderMode	21	
			5.1.1.2	TextAlignment	21	
6	Data	Struct	ure Docur	mentation	23	
	6.1	FTBBc		deference	23	
		6.1.1		Description	23	
		6.1.2	Construc	ctor & Destructor Documentation	24	
			6.1.2.1	FTBBox	24	
			6.1.2.2	FTBBox	24	
			6.1.2.3	FTBBox	24	
			6.1.2.4	FTBBox	24	
			6.1.2.5	~FTBBox	24	
		6.1.3		Function Documentation	24	
			6.1.3.1	Invalidate	24	
			6.1.3.2	IsValid	24	
			6.1.3.3	Lower	25	
			6.1.3.4	operator+=	25	
			6.1.3.5	operator =	25	
			6.1.3.6	SetDepth	25	
			6.1.3.7	Upper	25	
	6.2		•	lass Reference	26	
		6.2.1		Description	26	
		6.2.2		ctor & Destructor Documentation	26	
			6.2.2.1	FTBitmapFont	26	
			6.2.2.2	FTBitmapFont	27	

	pt3em		
		6.2.2.3 ~FTBitmapFont	27
	6.2.3	Member Function Documentation	27
		6.2.3.1 MakeGlyph	27
6.3	FTBitm	apGlyph Class Reference	27
	6.3.1	Detailed Description	28
	6.3.2	Constructor & Destructor Documentation	28
		6.3.2.1 FTBitmapGlyph	28
		6.3.2.2 \sim FTBitmapGlyph	28
	6.3.3	Member Function Documentation	28
		6.3.3.1 Render	28
6.4	FTBuff	er Class Reference	29
	6.4.1	Detailed Description	29
	6.4.2	Constructor & Destructor Documentation	29
		6.4.2.1 FTBuffer	29
		6.4.2.2 ~FTBuffer	30
	6.4.3	Member Function Documentation	30
		6.4.3.1 Height	30
		6.4.3.2 Pixels	30
		6.4.3.3 Pos	30
		6.4.3.4 Pos	30
		6.4.3.5 Size	30
		6.4.3.6 Width	31
6.5	FTBuff	erFont Class Reference	31
	6.5.1	Detailed Description	32
	6.5.2	Constructor & Destructor Documentation	32
		6.5.2.1 FTBufferFont	32
		6.5.2.2 FTBufferFont	32
		6.5.2.3 ~FTBufferFont	32
	6.5.3	Member Function Documentation	32
		6.5.3.1 MakeGlyph	32
6.6	FTBuff	erGlyph Class Reference	33
	6.6.1	Detailed Description	33
	6.6.2	Constructor & Destructor Documentation	33
		6.6.2.1 FTBufferGlyph	33
		6.6.2.2 \sim FTBufferGlyph	34
	6.6.3	Member Function Documentation	34
		6.6.3.1 Render	34
6.7	FTExtr	udeFont Class Reference	34
	6.7.1	Detailed Description	35

		pt3em		
	6.7.2	Construct	tor & Destructor Documentation	35
		6.7.2.1	FTExtrudeFont	35
		6.7.2.2	FTExtrudeFont	35
		6.7.2.3	\sim FTExtrudeFont	35
	6.7.3	Member I	Function Documentation	35
		6.7.3.1	MakeGlyph	35
6.8	FTExtr	udeGlyph (Class Reference	36
	6.8.1	Detailed I	Description	36
	6.8.2	Construct	tor & Destructor Documentation	36
		6.8.2.1	FTExtrudeGlyph	36
		6.8.2.2	~FTExtrudeGlyph	37
	6.8.3	Member I	Function Documentation	37
		6.8.3.1	Render	37
6.9	FTFont	t Class Ref	ference	37
	6.9.1	Detailed I	Description	39
	6.9.2	Construct	tor & Destructor Documentation	39
		6.9.2.1	FTFont	39
		6.9.2.2	FTFont	40
		6.9.2.3	\sim FTFont	40
	6.9.3	Member I	Function Documentation	40
		6.9.3.1	Advance	40
		6.9.3.2	Advance	40
		6.9.3.3	Ascender	41
		6.9.3.4	Attach	41
		6.9.3.5	Attach	41
		6.9.3.6	BBox	41
		6.9.3.7	BBox	42
		6.9.3.8	BBox	42
		6.9.3.9	BBox	42
		6.9.3.10	CharMap	43
		6.9.3.11	CharMapCount	43
		6.9.3.12	CharMapList	43
		6.9.3.13	Depth	43
		6.9.3.14	Descender	44
		6.9.3.15	Error	44
		6.9.3.16	FaceSize	44
		6.9.3.17	FaceSize	44
		6.9.3.18	GlyphLoadFlags	44
		6.9.3.19	LineHeight	45

		pt3em		
		6.9.3.20	MakeGlyph	45
		6.9.3.21	Outset	45
		6.9.3.22	Outset	45
		6.9.3.23	Render	45
		6.9.3.24	Render	46
		6.9.3.25	UseDisplayList	46
	6.9.4	Friends A	and Related Function Documentation	46
		6.9.4.1	FTBitmapFont	46
		6.9.4.2	FTBufferFont	47
		6.9.4.3	FTExtrudeFont	47
		6.9.4.4	FTFontImpl	47
		6.9.4.5	FTOutlineFont	47
		6.9.4.6	FTPixmapFont	47
		6.9.4.7	FTPolygonFont	47
		6.9.4.8	FTTextureFont	47
6.10	FTGlyp	h Class R	eference	47
	6.10.1	Detailed	Description	48
	6.10.2	Construc	tor & Destructor Documentation	48
		6.10.2.1	FTGlyph	48
		6.10.2.2	\sim FTGlyph	48
	6.10.3	Member	Function Documentation	49
		6.10.3.1	Advance	49
		6.10.3.2	BBox	49
		6.10.3.3	Error	49
		6.10.3.4	Render	49
	6.10.4	Friends A	and Related Function Documentation	49
		6.10.4.1	FTBitmapGlyph	49
		6.10.4.2	FTBufferGlyph	50
		6.10.4.3	FTExtrudeGlyph	50
		6.10.4.4	FTOutlineGlyph	50
		6.10.4.5	FTPixmapGlyph	50
		6.10.4.6	FTPolygonGlyph	50
		6.10.4.7	FTTextureGlyph	50
6.11	FTLayo	out Class F	Reference	50
	6.11.1	Detailed	Description	51
	6.11.2	Construc	tor & Destructor Documentation	51
		6.11.2.1	FTLayout	51
		6.11.2.2	\sim FTLayout	51
	6.11.3	Member	Function Documentation	51

		pt3em	
		6.11.3.1 BBox	51
		6.11.3.2 BBox	52
		6.11.3.3 Error	52
		6.11.3.4 Render	52
		6.11.3.5 Render	53
	6.11.4	Friends And Related Function Documentation	53
		6.11.4.1 FTSimpleLayout	53
6.12	FTOutli	ineFont Class Reference	53
	6.12.1	Detailed Description	54
	6.12.2	Constructor & Destructor Documentation	54
		6.12.2.1 FTOutlineFont	54
		6.12.2.2 FTOutlineFont	54
		6.12.2.3 ~FTOutlineFont	54
	6.12.3	Member Function Documentation	54
		6.12.3.1 MakeGlyph	54
6.13	FTOutli	ineGlyph Class Reference	55
	6.13.1	Detailed Description	55
	6.13.2	Constructor & Destructor Documentation	55
		6.13.2.1 FTOutlineGlyph	55
		6.13.2.2 ~FTOutlineGlyph	56
	6.13.3	Member Function Documentation	56
		6.13.3.1 Render	56
6.14	FTPixm	napFont Class Reference	56
	6.14.1	Detailed Description	57
	6.14.2	Constructor & Destructor Documentation	57
		6.14.2.1 FTPixmapFont	57
		6.14.2.2 FTPixmapFont	57
		6.14.2.3 ~FTPixmapFont	57
	6.14.3	Member Function Documentation	57
		6.14.3.1 MakeGlyph	58
6.15	FTPixm	napGlyph Class Reference	58
	6.15.1	Detailed Description	58
	6.15.2	Constructor & Destructor Documentation	59
		6.15.2.1 FTPixmapGlyph	59
		6.15.2.2 ~FTPixmapGlyph	59
	6.15.3	Member Function Documentation	59
		6.15.3.1 Render	59
6.16	FTPoin	t Class Reference	59
	6.16.1	Detailed Description	60

		pt3em	
	6.16.2	Constructor & Destructor Documentation	1
		6.16.2.1 FTPoint	1
		6.16.2.2 FTPoint	1
		6.16.2.3 FTPoint	1
	6.16.3	Member Function Documentation	1
		6.16.3.1 Normalise	1
		6.16.3.2 operator const FTGL_DOUBLE *	1
		6.16.3.3 operator*	31
		6.16.3.4 operator+	2
		6.16.3.5 operator+=	2
		6.16.3.6 operator	2
		6.16.3.7 operator-=	3
		6.16.3.8 operator $^{\wedge}$	3
		6.16.3.9 X	3
		6.16.3.10 X	3
		6.16.3.11 Xf	3
		6.16.3.12 Y	64
		6.16.3.13 Y	64
		6.16.3.14 Yf	64
		6.16.3.15 Z	64
		6.16.3.16 Z	64
		6.16.3.17 Zf	64
	6.16.4	Friends And Related Function Documentation	64
		6.16.4.1 operator!=	4
		6.16.4.2 operator*	64
		6.16.4.3 operator*	35
		6.16.4.4 operator==	35
6.17	FTPoly	gonFont Class Reference	35
	6.17.1	Detailed Description	6
	6.17.2	Constructor & Destructor Documentation	6
		6.17.2.1 FTPolygonFont	6
		6.17.2.2 FTPolygonFont	6
		6.17.2.3 \sim FTPolygonFont	67
	6.17.3	Member Function Documentation	67
		6.17.3.1 MakeGlyph	67
6.18	FTPoly	gonGlyph Class Reference	67
	6.18.1	Detailed Description	8
	6.18.2	Constructor & Destructor Documentation	8
		6.18.2.1 FTPolygonGlyph	8

		pt3em		
			6.18.2.2 \sim FTPolygonGlyph	68
		6.18.3	Member Function Documentation	68
			6.18.3.1 Render	68
	6.19	FTSimp	oleLayout Class Reference	69
		6.19.1	Detailed Description	69
		6.19.2	Constructor & Destructor Documentation	70
			6.19.2.1 FTSimpleLayout	70
			6.19.2.2 \sim FTSimpleLayout	70
		6.19.3	Member Function Documentation	70
			6.19.3.1 BBox	70
			6.19.3.2 BBox	70
			6.19.3.3 GetAlignment	71
			6.19.3.4 GetFont	71
			6.19.3.5 GetLineLength	71
			6.19.3.6 GetLineSpacing	71
			6.19.3.7 Render	71
			6.19.3.8 Render	71
			6.19.3.9 SetAlignment	72
			6.19.3.10 SetFont	72
			6.19.3.11 SetLineLength	72
			6.19.3.12 SetLineSpacing	72
	6.20	FTText	ureFont Class Reference	73
		6.20.1	Detailed Description	73
		6.20.2	Constructor & Destructor Documentation	73
			6.20.2.1 FTTextureFont	73
			6.20.2.2 FTTextureFont	74
			6.20.2.3 ~FTTextureFont	74
		6.20.3	Member Function Documentation	74
			6.20.3.1 MakeGlyph	74
	6.21	FTText	ureGlyph Class Reference	74
		6.21.1	Detailed Description	75
		6.21.2	Constructor & Destructor Documentation	75
			6.21.2.1 FTTextureGlyph	75
			6.21.2.2 ~FTTextureGlyph	75
		6.21.3	Member Function Documentation	75
			6.21.3.1 Render	75
7	File [Docume	entation	77
				77

pt3e	pt3em					
7.2	FTBBox.h File Reference					
7.3	FTBitm	apGlyph.h	File Reference	77		
	7.3.1	Function	Documentation	77		
		7.3.1.1	ftglCreateBitmapGlyph	77		
7.4	FTBuff	er.h File R	eference	78		
7.5	FTBuff	erFont.h F	ile Reference	78		
	7.5.1	Function	Documentation	78		
		7.5.1.1	ftglCreateBufferFont	78		
7.6	FTBuff	erGlyph.h	File Reference	79		
7.7	FTExtr	dGlyph.h F	File Reference	79		
	7.7.1	Macro De	efinition Documentation	79		
		7.7.1.1	FTExtrdGlyph	79		
	7.7.2	Function	Documentation	79		
		7.7.2.1	ftglCreateExtrudeGlyph	79		
7.8	FTFont	t.h File Ret	ference	80		
	7.8.1	Typedef [Documentation	81		
		7.8.1.1	FTGLfont	81		
	7.8.2	Function	Documentation	81		
		7.8.2.1	ftglAttachData	81		
		7.8.2.2	ftglAttachFile	81		
		7.8.2.3	ftglCreateCustomFont	82		
		7.8.2.4	ftglDestroyFont	82		
		7.8.2.5	ftglGetFontAdvance	82		
		7.8.2.6	ftglGetFontAscender	83		
		7.8.2.7	ftglGetFontBBox	83		
		7.8.2.8	ftglGetFontCharMapCount	83		
		7.8.2.9	ftglGetFontCharMapList	83		
		7.8.2.10	ftglGetFontDescender	84		
		7.8.2.11	ftglGetFontError	84		
		7.8.2.12	ftglGetFontFaceSize	84		
		7.8.2.13	ftglGetFontLineHeight	84		
		7.8.2.14	ftglRenderFont	85		
		7.8.2.15	ftglSetFontCharMap	85		
		7.8.2.16	ftglSetFontDepth	85		
		7.8.2.17	ftglSetFontDisplayList	85		
		7.8.2.18	ftglSetFontFaceSize	86		
		7.8.2.19	ftglSetFontOutset	86		
7.9	ftgl.dox	File Refe	rence	86		
7.10	ftgl.h F	ile Referer	nce	86		

		pt3em	
	7.10.1	Macro Definition Documentation	87
		7.10.1.1 FTGL_BEGIN_C_DECLS	87
		7.10.1.2 FTGL_END_C_DECLS	88
		7.10.1.3 FTGL_EXPORT	88
	7.10.2	Typedef Documentation	88
		7.10.2.1 FTGL_DOUBLE	88
		7.10.2.2 FTGL_FLOAT	88
7.11	FTGLB	itmapFont.h File Reference	88
	7.11.1	Macro Definition Documentation	88
		7.11.1.1 FTGLBitmapFont	88
	7.11.2	Function Documentation	89
		7.11.2.1 ftglCreateBitmapFont	89
7.12	FTGLE	xtrdFont.h File Reference	89
	7.12.1	Macro Definition Documentation	89
		7.12.1.1 FTGLExtrdFont	89
	7.12.2	Function Documentation	89
		7.12.2.1 ftglCreateExtrudeFont	89
7.13	FTGLC	OutlineFont.h File Reference	90
	7.13.1	Macro Definition Documentation	90
		7.13.1.1 FTGLOutlineFont	90
	7.13.2	Function Documentation	90
		7.13.2.1 ftglCreateOutlineFont	90
7.14	FTGLP	ixmapFont.h File Reference	91
	7.14.1	Macro Definition Documentation	91
		7.14.1.1 FTGLPixmapFont	91
	7.14.2	Function Documentation	91
		7.14.2.1 ftglCreatePixmapFont	91
7.15	FTGLP	olygonFont.h File Reference	92
	7.15.1	Macro Definition Documentation	92
		7.15.1.1 FTGLPolygonFont	92
	7.15.2	Function Documentation	92
		7.15.2.1 ftglCreatePolygonFont	92
7.16	FTGLT	extureFont.h File Reference	93
	7.16.1	Macro Definition Documentation	93
		7.16.1.1 FTGLTextureFont	93
	7.16.2	Function Documentation	93
		7.16.2.1 ftglCreateTextureFont	93
7.17	FTGlyp	h.h File Reference	93
	7.17.1	Typedef Documentation	94

	pt3em			
		7.17.1.1	FTGLglyph	94
	7.17.2	Function	Documentation	94
		7.17.2.1	ftglCreateCustomGlyph	94
		7.17.2.2	ftglDestroyGlyph	95
		7.17.2.3	ftglGetGlyphAdvance	95
		7.17.2.4	ftglGetGlyphBBox	95
		7.17.2.5	ftglGetGlyphError	95
		7.17.2.6	ftglRenderGlyph	96
7.18	FTLayo	ut.h File F	Reference	96
	7.18.1	Typedef [Documentation	96
		7.18.1.1	FTGLlayout	96
	7.18.2	Function	Documentation	97
		7.18.2.1	ftglDestroyLayout	97
		7.18.2.2	ftglGetLayoutBBox	97
		7.18.2.3	ftglGetLayoutError	97
		7.18.2.4	ftglRenderLayout	97
7.19	FTOutli	ineGlyph.h	n File Reference	98
	7.19.1	Function	Documentation	98
		7.19.1.1	ftglCreateOutlineGlyph	98
7.20	FTPixm	napGlyph.l	h File Reference	98
	7.20.1	Function	Documentation	99
			ftglCreatePixmapGlyph	
7.21	FTPoin	t.h File Re	eference	99
7.22	_		ile Reference	99
	7.22.1	Macro De	efinition Documentation	99
		7.22.1.1	FTPolyGlyph	99
	7.22.2	Function	Documentation	100
		7.22.2.1	ftglCreatePolygonGlyph	100
7.23	FTSimp	oleLayout.l	h File Reference	100
	7.23.1	Function	Documentation	100
		7.23.1.1	ftglCreateSimpleLayout	100
		7.23.1.2	ftglGetLayoutAlignement	100
		7.23.1.3	ftglGetLayoutFont	100
		7.23.1.4	ftglGetLayoutLineLength	100
		7.23.1.5	ftglGetLayoutLineSpacing	. 101
		7.23.1.6	ftglSetLayoutAlignment	. 101
		7.23.1.7	ftglSetLayoutFont	101
		7.23.1.8	ftglSetLayoutLineLength	101
		7.23.1.9	ftglSetLayoutLineSpacing	101

	pt3em																
7.24	FTTextureGlyph.h File Reference											101					
	7.24.1	Function	Documentation									 					101
		7.24.1.1	ftglCreateTextu	ıreGlyph								 					101
7.25	project	s_using_ft	gl.txt File Refere	ence								 					101
7.26	tutorial	.dox File R	leference									 					101

Chapter 1

FTGL User Guide



1.1 Introduction

OpenGL doesn't provide direct font support, so the application must use any of OpenGL's other features for font rendering, such as drawing bitmaps or pixmaps, creating texture maps containing an entire character set, drawing character outlines, or creating a 3D geometry for each character.

More information can be found on the OpenGL website:

- http://www.opengl.org/resources/faq/technical/fonts.htm
- http://www.opengl.org/resources/features/fontsurvey/

Most of these systems require a pre-processing stage to take the native fonts and convert them into a proprietary format.

FTGL was born out of the need to treat fonts in OpenGL applications just like any other application. For example when using Adobe Photoshop or Microsoft Word you don't need an intermediate pre-processing step to use high quality scalable fonts.

1.2 Documentation

- FTGL tutorial (p. ??)
- · C API reference:
 - FTGlyph.h (p. 93)
 - FTFont.h (p. 80)
 - FTLayout.h (p. 96)

- C++ API reference:
 - class FTGlyph (p. 47)
 - class FTFont (p. 37)
 - class FTLayout (p. 50)

1.3 Additional information

- Frequently Asked Questions (p. ??)
- Projects using FTGL (p. ??)

Chapter 2

Frequently Asked Questions

2.1 FAQ

2.1.1 When I try to compile %FTGL it complains about a missing file from the include: #include <ft2build.h>

FTGL relies on FreeType 2 for opening and decoding font files. This include is the main include for FreeType. You will need to download Freetype 2 and install it. Then make sure that the FTGL project that you are using points to your FreeType installation.

2.1.2 Is it possible to map a font to a "unit" size? My application relies on the fonts being a certain "physical" height (in OpenGL coordinate space) rather than a point size in display space. Any thoughts/suggestions?

We can do anything:) It would be easy to allow you to set the size in pixels, though I'm not sure this is what you want. Setting the size to 'OpenGL units' may be a bit harder. What does 1.0 in opengl space mean and how does that relate to point size? For one person it might mean scaling the font up, for someone else it may mean scaling down. Plus bitmaps and pixmaps have a pixel to pixel relationship that you can't change.

Here's some guidelines for vector and texture fonts. Take note that I say 'should' a lot :)

- One point in pixel space maps to 1 unit in OpenGL space, so a glyph that is 18 points high should be 18.0 units high.
- If you set an ortho projection to the window size and draw a glyph it's screen size should be the correct physical size ie a 72 point glyph on a 72dpi screen will be 1 inch high. Also if you set a perspective projection that maps 0.0 in the z axis to screen size you will get the same eg.

```
gluPerspective(90, window_height / 2 , small_number, large_number);
```

So basically it all depends on your projection matrix. Obviously you can use glScale but I understand if you don't want to.

Couple of extra things to note:

- The quality of vector glyphs will not change when you change the size, ie. a really small polygon glyph up close will look exactly the same as a big one from far away. They both contain the same amount of data. This doesn't apply to texture fonts.
- Secondly, there is a bug in the advance/kerning code that will cause ugliness at really small point sizes. This
 is because the advance and kerning use ints so an advance of 0.4 will become zero. If this is going to be a
 probelm, I can fix this.

Early on I did a lot of head scratching over the OpenGL unit to font size thing because when I was first integrating FTGL into my engine the fonts weren't the size I was expecting. I was tempted to build in some scaling but I decided doing nothing was the best approach because you can't please everyone. Plus it's 'correct' as it is.

Chapter 3

Projects using FTGL

To add your project to this list, please contact one of the FTGL developers at http://sf.net/projects/ftgl Projects are listed in alphabetical order.

3.1 %FTGL language bindings

3.1.1 %FTGL#

FTGL# (http://www.paskaluk.com/projects.php) is a collection of .NET bindings for FTGL.

3.1.2 GIGuiA

GlGuiA (http://sourceforge.net/projects/glguia/) is a set of packages for Ada 2006 that can be used to create Graphical User Interfaces, relaying (almost) only on OpenGl. Hence should be rather platform-independent.

3.1.3 Ruby %FTGL

Ruby FTGL# (http://rubyforge.org/projects/ruby-ftgl/) is a collection of Ruby bindings for FT-GL

3.1.4 PyFTGL

PyFTGL (http://code.google.com/p/pyftgl/) wraps the functionality of FTGL into a Python module so that it can be used in conjunction with PyOpenGL.

3.2 Projects currently using %FTGL

3.2.1 Agent World

Agent World (http://code.google.com/p/agentw/) provides tools for simulating and visualizing multiagent systems and is specially designed for testing machine learning applications (and specially focused on Case Based Reasoning ones). It includes support for representing information using the Feature Term formalism, and provides a series of relational machine learning algorithms that can deal with them. The whole project is created in C++ to maximize efficiency, and uses OpenGL as the visualization library to ensure cross-platformness.

3.2.2 Amaltheia

Amaltheia (http://home.gna.org/amaltheia/) is a cross-platform game programming API that supports two backends, OpenGL and DirectX. The aim of the Amaltheia project is to create an intuitive and simple to use library, providing core 3d and 2d functionality in a platform independent manner. It also provides platform independence regarding basic network functions, input handling, threads and sound. Currently the GNU/Linux and the Windows OSes are supported.

3.2.3 Armagetron Advanced

Armagetron Advanced (http://www.armagetronad.net/) is a multiplayer game in 3d that attempts to emulate and expand on the lightcycle sequence from the movie Tron. It's an old school arcade game slung into the 21st century. Highlights include a customizable playing arena, HUD, unique graphics, and AI bots. For the more advanced player there are new game modes and a wide variety of physics settings to tweak as well.

3.2.4 Audicle

Audicle (http://audicle.cs.princeton.edu/) is an audio programming environment that integrates the programmability of the development environment with elements of the runtime environment. The result is a duct-taped intersection of a concurrent smart editor, compiler, virtual machine, and debugger.

3.2.5 Battlestar T.U.X.

Battlestar T.U.X. (http://code.google.com/p/battlestar-tux/) is a top-down scrolling shooter project.

3.2.6 BJS

BJS (http://bjs.sourceforge.net/) is a funny arcade 3D multiplayer tank battle. It is fully playable and very fun in multiplayer. Of course the single player is also possible. There is no story. You just get a tank and go shoot other players. Currently there are 5 different tanks, 6 maps, 9 powerups and 4 weapons.

3.2.7 Blender

Blender (http://blender.org/) is an integrated 3d suite for modelling, animation, rendering, post-production, interactive creation and playback (games).

3.2.8 Breve

Breve (http://www.spiderland.org/) is a free, open-source software package which makes it easy to build 3D simulations of multi-agent systems and artificial life. Using Python, or using a simple scripting language called steve, you can define the behaviors of agents in a 3D world and observe how they interact. breve includes physical simulation and collision detection so you can simulate realistic creatures, and an OpenGL display engine so you can visualize your simulated worlds.

3.2.9 BZFlag

BZFlag (http://BZFlag.org/) is a 3D multi-player multiplatform tank battle game that allows users to play against each other in a network environment.

BZFlag uses FTGL as of version 2.99.

3.2.10 Capture The Flag

Capture The Flag (http://capturetf.sourceforge.net/) is an open source, multi-platform, network game project.

pt3em

3.2.11 Cello

Cello (http://common-lisp.net/project/cello/) is a project to create an open-source, industrial-strength, portable GUI toolkit for Common Lisp. Its features include anti-aliased fonts, accelerated 2d- and 3d-graphics, a standard set of GUI widgets, easy construction of new widgets, and much more. Cello heavily utilizes Cells (a sister project on common-lisp.net), in addition to industry-standard technologies such as OpenGL, Free-Type, and ImageMagick.

3.2.12 Chimera

Chimera (http://www.cgl.ucsf.edu/chimera/) is a highly extensible program for interactive visualization and analysis of molecular structures and related data, including density maps, supramolecular assemblies, sequence alignments, docking results, trajectories, and conformational ensembles. High-quality images and animations can be generated.

3.2.13 Cinepaint

Cinepaint (http://www.cinepaint.org/) is a deep paint image retouching tool that supports higher color fidelity than ordinary painting tools.

3.2.14 Duel

Duel (http://www.personal.rdg.ac.uk/ \sim sir03me/play/code.html) is a small overhead perspective spaceship game.

3.2.15 Empty Clip

Empty Clip (http://emptyclip.sourceforge.net/) is a top-down 2D Action RPG.

3.2.16 Freebox

Freebox (http://freebox.sourceforge.net/) is designed for use in a special type of computer called an 'HTPC', which is connected to a home-theatre system to watch XviD/DivX/DVD movies, play music (MP3, CD, whatever), play some emulated games, or whatever else you want to do with it.

3.2.17 Gem

Gem (http://gem.iem.at/) is a loadable library for puredata, which adds OpenGL graphics rendering and animation to Pd. Pd is a graphical programming language and computer music system.

3.2.18 GLMayan

GLMayan (http://glmayan.sourceforge.net/) is an OpenGL screensaver.

3.2.19 Glover

Glover (http://code.google.com/p/glover/) is a movie player that renders the content using openGL allowing all kinds of special effects using fragment shaders. The movie decoding is done using ffmpeg.

3.2.20 lvf++

lvf++ (http://ivfplusplus.sourceforge.net/) is a C++ library encapsulating OpenGL functionality. The primary goal is to make it easier to use the OpenGL library in interactive 3D applications. The second goal is extendibility, providing a set of well defined base classes for different object types to build new classes on. The third goal is portability, primarily between Linux and Windows, but the library should also be easily ported to Mac OS X.

3.2.21 Jahshaka

Jashaka (http://jahshaka.org/) is an advanced video editing, animation, visual effects, painting and music tool.

3.2.22 Karaoke FX

Karaoke FX (http://jeanchristophe.duber.free.fr/karaokefx/) is a midifile player that can display lyrics in synch whith the sound so as it can be used for karaoke. It relies on plugins for midi output devices as for lyrics display.

3.2.23 Libinstrudeo

Libinstrudeo (http://sourceforge.net/projects/libinstrudeo), initially written for the Screen-Kast program, provides the necessary logic to capture screen recordings and to process them. Includes a soapclient for the webservice at captorials.com that enables you to share your recordings.

3.2.24 Light Speed!

Light Speed! (http://lightspeed.sourceforge.net/) is an OpenGL-based program which illustrates the effects of special relativity on the appearance of moving objects. When an object accelerates past a few million meters per second, these effects begin to grow noticeable, becoming more and more pronounced as the speed of light is approached. These relativistic effects are viewpoint-dependent, and include shifts in length, object hue, brightness and shape.

3.2.25 MySQL GUI Tools

MySQL GUI Tools (http://dev.mysql.com/downloads/gui-tools/5.0.html) is a collection of tools for the MySQL database. It consists of MySQL Administrator, MySQL Query Browser and MySQL Migration Toolkit.

3.2.26 OctPlot

OctPlot (http://octplot.sourceforge.net/) is a graphics package for Octave, the free alternative to MATLAB. It provides high quality PostScript and on-screen graphics.

3.2.27 Open ActiveWrl

Open ActiveWrl (http://open-activewrl.sourceforge.net/) is a software development toolkit based on a generic software development approach that allows the implementation VRML/X3D browser componentes. These browser components can run within an conventional application or can be linked together for the implementation of parallel immersive VR setups.

3.2.28 OpenEaagles

OpenEaagles (http://www.openeaagles.org/) is a multi-platform simulation framework targeted to help simulation engineers and software developers build robust, scalable, virtual, constructive, stand-alone, and distributed simulation applications. It has been used extensively to build applications that demand real-time performance. This includes applications to conduct human factor studies, operator training, and the development of complete distributed virtual simulation systems. OpenEaagles has also been used to build stand-alone and distributed constructive applications oriented at system analysis.

3.2.29 OpenGC

OpenGC (http://www.opengc.org/) is a multi-platform, multi-simulator, open-source C++ tool for developing and implementing high quality glass cockpit displays for simulated flightdecks.

3.2.30 OpenSG

OpenSG (http://www.opensg.org/) is a portable scenegraph system to create realtime graphics programs, e.g. for virtual reality applications.

3.2.31 Panthera

Panthera (http://sourceforge.net/projects/panthera) is a C++ framework for interactive visualization, manipulation, and editing of volume data. Applications developed on top of Panthera can utilize both desktop and immersive user interface devices, such as position trackers and haptic displays.

3.2.32 Planet Penguin Racer

PlanetPenguin Racer (http://developer.berlios.de/projects/ppracer/) is a simple OpenGL racing game featuring Tux, the Linux mascot. The goal of the game is to slide down a snow- and ice-covered mountain as quickly as possible, avoiding the trees and rocks that will slow you down.

3.2.33 projectM

projectM (http://projectm.sourceforge.net/) is a music visualizer which uses OpenGL for hardware acceleration. It is compatible with Milkdrop presets.

3.2.34 Puzzle Bobble 3D

Puzzle Bobble 3D (http://homepage.mac.com/eric.lee/puzzle/) is a 3D video game for Linux. The game is similar to Tetris/Connect 4: connect balls of the same colour to make them disappear. Puzzle Bobble 3D is based on an already popular arcade game of the same name by Taito Corporation (see links section at the bottom of this page), but this particular variant is played in a 3D environment (hence the name).

3.2.35 ROOT

ROOT (http://root.cern.ch/) is an object-oriented data analysis framework.

3.2.36 SCIRun

SCIRun (http://software.sci.utah.edu/scirun.html) is a Problem Solving Environment (PSE), for modeling, simulation and visualization of scientific problems. It is available for free and open source.

3.2.37 TINE

TINE, or TINE is Not ELITE (http://tine.sunsite.dk/en/index.html) is an open source cross-platform remake of the classic space adventure game ELITE.

3.2.38 Tiny Planet

Tiny Planet (http://www.duberga.net/tinyplanet/) is a real-time OpenGL viewer of detailled earth texture such as BlueMarble from Earth Observatory (NASA) or any other planet texture. Vectorial data such as points of interest, boundaries, rivers can be superimposed to the texture.

3.2.39 Truevision

Truevision (http://truevision.sourceforge.net/) is a 3D modeler for GNOME.

3.2.40 Tulip

Tulip (http://tulip.labri.fr/) is a system dedicated to the visualization of huge graphs. It is capable of managing graphs with up to 500,000 nodes and edges on relatively modest hardware (eg. 600MHz Pentium III, 256MB RAM).

3.2.41 Ubit

Ubit (http://www.infres.enst.fr/ \sim elc/ubit/) Ubit is a new GUI toolkit that combines the advantages of scene graph and widget based toolkits. The Ubit3D extension makes it possible to display 2D GUIs in a 3D space.

3.2.42 VRS

The Virtual Rendering System (http://www.hpi.uni-potsdam.de/vrs/) is a computer graphics software library for constructing interactive 3D applications. It provides a large collection of 3D rendering components which facilitate implementing 3D graphics applications and experimenting with 3D graphics and imaging algorithms.

3.2.43 VTK

VTK, the Visualization Toolkit (http://www.vtk.org/), is an object oriented, high level library that allows one to easily write C++ programs, Tcl, Python and Java scripts that do 3D visualization.

3.2.44 XLock

XLock (http://www.tux.org/ \sim bagleyd/xlockmore.html) is a screensaver and screen locking utility with additional OpenGL and XPM modes.

3.3 Projects that used to use %FTGL

3.3.1 GNU Backgammon

GNU Backgammon (http://www.gnubg.org/) was using FTGL until version 0.14.3+20060520-1.

3.3.2 OpenSceneGraph

OpenSceneGraph (http://www.openscenegraph.org/projects/osg) is an open source high performance 3D graphics toolkit, used by application developers in fields such as visual simulation, games, virtual reality, scientific visualization and modelling. Written entirely in Standard C++ and OpenGL it runs on all Windows platforms, OSX, GNU/Linux, IRIX, Solaris, HP-Ux, AIX and FreeBSD operating systems.

3.3.3 Teddy

Teddy (http://teddy.sourceforge.net/) was a 3D graphics library. The main purpose was to be a simple scene graph manager.

3.3.4 VigiPac

 $\label{lem:vigipac.sourceforge.net/} \begin{tabular}{ll} Vigipac.sourceforge.net/) was a three-dimensional Pacman clone with multiplayer support, written in the C++ language. \end{tabular}$

Chapter 4

FTGL tutorial

4.1 Starting to use %FTGL

Only one header is required to use FTGL:

#include <FTGL/ftgl.h>

4.2 Choosing a font type

FTGL supports 6 font output types among 3 groups: raster fonts, vector fonts, and texture fonts which are a mixture of both. Each font type has its advantages and disadvantages.

4.2.1 Raster fonts

Raster fonts are made of pixels painted directly on the viewport's framebuffer. They cannot be directly rotated or scaled.

- · Bitmap fonts use 1-bit (2-colour) rasterised glyphs.
- Pixmap fonts use 8-bit (256 levels) rasterised glyphs.

This is a GLBitmapFont object. This is a GLPixmapFont object.

4.2.2 Vector fonts

Vector fonts are 3D objects that are rendered at the current matrix location. All position, scale, texture and material effects apply to vector fonts.

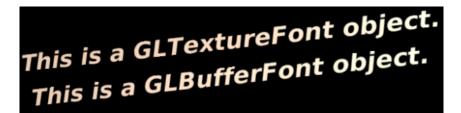
- · Polygon fonts use planar triangle meshes and can be texture-mapped.
- · Outline fonts use OpenGL lines.
- Extruded fonts are extruded polygon fonts, with the front, back and side meshes renderable separately to apply different effects and materials.



4.2.3 Textured fonts

Textured fonts are probably the most versatile types. They are fast, antialiased, and can be transformed just like any OpenGL primitive.

- Texture fonts use one texture per glyph. They are fast because glyphs are stored permanently in the video card's memory.
- Buffer fonts use one texture per line of text. They tend to be faster than texture fonts when the same line of text needs to be rendered for more than one frame.



4.3 Create font objects

Creating a font and displaying some text is really straightforward, be it in C or in C++.

4.3.1 in C

```
/* Create a pixmap font from a TrueType file. */
FTGLfont *font = ftglCreatePixmapFont("/home/user/Arial.ttf");
/* If something went wrong, bail out. */
if(!font)
    return -1;

/* Set the font size and render a small text. */
ftglSetFontFaceSize(font, 72, 72);
ftglRenderFont(font, "Hello World!", FTGL_RENDER_ALL);
/* Destroy the font object. */
ftglDestroyFont(font);

4.3.2 in C++

// Create a pixmap font from a TrueType file.
FTGLPixmapFont font("/home/user/Arial.ttf");
// If something went wrong, bail out.
```

if(font.Error())

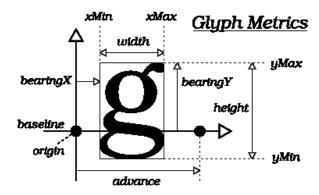
```
pt3em
    return -1;

// Set the font size and render a small text.
font.FaceSize(72);
font.Render("Hello World!"):
```

The first 128 glyphs of the font (generally corresponding to the ASCII set) are preloaded. This means that usual text is rendered fast enough, but no memory is wasted loading glyphs that will not be used.

4.4 More font commands

4.4.1 Font metrics



If you ask a font to render at 0.0, 0.0 the bottom left most pixel or polygon may not be aligned to 0.0, 0.0. With F-TFont::Ascender() (p. 41), FTFont::Descender() (p. 44) and FTFont::Advance() (p. 40) an approximate bounding box can be calculated.

For an exact bounding box, use the **FTFont::BBox()** (p. 41) function. This function returns the extent of the volume containing 'string'. 0.0 on the y axis will be aligned with the font baseline.

4.4.2 Specifying a character map encoding

From the FreeType documentation:

"By default, when a new face object is created, (FreeType) lists all the charmaps contained in the font face and selects the one that supports Unicode character codes if it finds one. Otherwise, it tries to find support for Latin-1, then ASCII."

It then gives up. In this case FTGL will set the charmap to the first it finds in the fonts charmap list. You can expilcitly set the char encoding with **FTFont::CharMap()** (p. 43).

Valid encodings as of FreeType 2.0.4 are:

- · ft_encoding_none
- ft_encoding_unicode
- · ft_encoding_symbol
- · ft_encoding_latin_1
- ft_encoding_latin_2
- · ft_encoding_sjis
- ft_encoding_gb2312

- · ft_encoding_big5
- · ft encoding wansung
- · ft encoding johab
- · ft_encoding_adobe_standard
- · ft_encoding_adobe_expert
- · ft_encoding_adobe_custom
- ft_encoding_apple_roman

For instance:

```
font.CharMap(ft_encoding_apple_roman);
```

This will return an error if the requested encoding can't be found in the font.

If your application uses Latin-1 characters, you can preload this character set using the following code:

```
// Create a pixmap font from a TrueType file.
FTGLPixmapFont font("/home/user/Arial.ttf");

// If something went wrong, bail out.
if(font.Error())
    return -1;

// Set the face size and the character map. If something went wrong, bail out.
font.FaceSize(72);
if(!font.CharMap(ft_encoding_latin_1))
    return -1;

// Create a string containing all characters between 128 and 255
// and preload the Latin-1 chars without rendering them.
char buf[129];
for(int i = 128; i < 256; i++)
{
    buf[i] = (char) (unsigned char)i;
}
buf[128] = '\0';
font.Advance(buf);
}</pre>
```

4.5 Sample font manager class

```
FTTextureFont* myFont = FTGLFontManager::Instance().GetFont("arial.ttf", 72);
#include <map>
#include <string>
#include <FTGL/ftgl.h>
using namespace std;
typedef map<string, FTFont*> FontList;
typedef FontList::const_iterator FontIter;
class FTGLFontManager
    public:
       // NOTE
        \ensuremath{//} This is shown here for brevity. The implementation should be in the
       source
        // file otherwise your compiler may inline the function resulting in
        // multiple instances of FTGLFontManager
        static FTGLFontManager& Instance()
            static FTGLFontManager tm;
            return tm:
```

```
pt3em
        ~FTGLFontManager()
        {
            FontIter font;
             for(font = fonts.begin(); font != fonts.end(); font++)
                delete (*font).second;
             fonts.clear();
        FTFont* GetFont(const char *filename, int size)
            char buf[256];
sprintf(buf, "%s%i", filename, size);
            string fontKey = string(buf);
            FontIter result = fonts.find(fontKey);
            if(result != fonts.end())
                 LOGMSG("Found font %s in list", filename);
                 return result->second;
             }
            FTFont* font = new FTTextureFont;
            string fullname = path + string(filename);
            if(!font->Open(fullname.c_str()))
                 LOGERROR("Font %s failed to open", fullname.c_str());
                 delete font;
                 return NULL;
             if(!font->FaceSize(size))
                 LOGERROR ("Font %s failed to set size %i", filename, size);
                 delete font;
                 return NULL;
            fonts[fontKey] = font;
            return font;
    private:
        // Hide these 'cause this is a singleton.
        FTGLFontManager(){}
        FTGLFontManager(const FTGLFontManager&){};
        FTGLFontManager& operator = (const FTGLFontManager&) { return *this; };
        // container for fonts
FontList fonts;
};
```

Chapter 5

Namespace Documentation

5.1 FTGL Namespace Reference

Enumerations

- enum RenderMode { RENDER_FRONT = 0x0001, RENDER_BACK = 0x0002, RENDER_SIDE = 0x0004, RENDER_ALL = 0xffff }
- enum TextAlignment { ALIGN_LEFT = 0, ALIGN_CENTER = 1, ALIGN_RIGHT = 2, ALIGN_JUSTIFY = 3 }

5.1.1 Enumeration Type Documentation

5.1.1.1 enum FTGL::RenderMode

pt3emEnumerator:

RENDER_FRONT RENDER_BACK RENDER_SIDE RENDER_ALL

Definition at line 53 of file ftgl.h.

5.1.1.2 enum FTGL::TextAlignment

pt3emEnumerator:

ALIGN_LEFT
ALIGN_CENTER
ALIGN_RIGHT
ALIGN_JUSTIFY

Definition at line 61 of file ftgl.h.

Chapter 6

Data Structure Documentation

6.1 FTBBox Class Reference

FTBBox (p. 23) is a convenience class for handling bounding boxes.

```
#include <FTBBox.h>
```

Public Member Functions

• FTBBox ()

Default constructor.

FTBBox (float lx, float ly, float lz, float ux, float uy, float uz)

Constructor.

• FTBBox (FTPoint I, FTPoint u)

Constructor.

• FTBBox (FT_GlyphSlot glyph)

Constructor.

• \sim FTBBox ()

Destructor.

· void Invalidate ()

Mark the bounds invalid by setting all lower dimensions greater than the upper dimensions.

· bool IsValid ()

Determines if this bounding box is valid.

FTBBox & operator+= (const FTPoint vector)

Move the Bounding Box by a vector.

• FTBBox & operator = (const FTBBox &bbox)

Combine two bounding boxes.

- void SetDepth (float depth)
- FTPoint const Upper () const
- · FTPoint const Lower () const

6.1.1 Detailed Description

FTBBox (p. 23) is a convenience class for handling bounding boxes.

Definition at line 42 of file FTBBox.h.

6.1.2 Constructor & Destructor Documentation

```
6.1.2.1 FTBBox::FTBBox() [inline]
```

Default constructor.

Bounding box is set to zero.

Definition at line 48 of file FTBBox.h.

6.1.2.2 FTBBox::FTBBox (float lx, float ly, float lz, float ux, float uy, float uz) [inline]

Constructor.

Definition at line 56 of file FTBBox.h.

6.1.2.3 FTBBox::FTBBox (FTPoint *I*, FTPoint *u*) [inline]

Constructor.

Definition at line 64 of file FTBBox.h.

6.1.2.4 FTBBox::FTBBox (FT_GlyphSlot glyph) [inline]

Constructor.

Extracts a bounding box from a freetype glyph. Uses the control box for the glyph. $FT_Glyph_Get_CBox()$

pt3emParameters

pt3em pt3emglyph | pt3emA freetype glyph

Definition at line 75 of file FTBBox.h.

6.1.2.5 FTBBox::∼FTBBox() [inline]

Destructor.

Definition at line 93 of file FTBBox.h.

6.1.3 Member Function Documentation

```
6.1.3.1 void FTBBox::Invalidate() [inline]
```

Mark the bounds invalid by setting all lower dimensions greater than the upper dimensions.

Definition at line 100 of file FTBBox.h.

6.1.3.2 bool FTBBox::IsValid() [inline]

Determines if this bounding box is valid.

pt3emReturns

True if all lower values are <= the corresponding upper values.

Definition at line 112 of file FTBBox.h.

6.1.3.3 FTPoint const FTBBox::Lower() const [inline]

Definition at line 165 of file FTBBox.h.

Referenced by FTFont::BBox().

6.1.3.4 FTBBox& FTBBox::operator+= (const FTPoint vector) [inline]

Move the Bounding Box by a vector.

pt3emParameters

pt3empt3emvector | pt3emThe vector to move the bbox in 3D space.

Definition at line 124 of file FTBBox.h.

6.1.3.5 FTBBox& FTBBox::operator = (const FTBBox & bbox) [inline]

Combine two bounding boxes.

The result is the smallest bounding box containing the two original boxes.

pt3emParameters

pt3em pt3embbox | pt3emThe bounding box to merge with the second one.

Definition at line 138 of file FTBBox.h.

References FTPoint::X(), FTPoint::Y(), and FTPoint::Z().

6.1.3.6 void FTBBox::SetDepth (float depth) [inline]

Definition at line 150 of file FTBBox.h.

6.1.3.7 FTPoint const FTBBox::Upper() const [inline]

Definition at line 159 of file FTBBox.h.

Referenced by FTFont::BBox().

The documentation for this class was generated from the following file:

• FTBBox.h

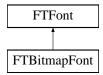
6.2 FTBitmapFont Class Reference

FTBitmapFont (p. 26) is a specialisation of the FTFont (p. 37) class for handling Bitmap fonts.

#include <FTGLBitmapFont.h>

Inheritance diagram for FTBitmapFont:

pt3em



Public Member Functions

• FTBitmapFont (const char *fontFilePath)

Open and read a font file.

• FTBitmapFont (const unsigned char *pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

• ∼FTBitmapFont ()

Destructor.

Protected Member Functions

virtual FTGlyph * MakeGlyph (FT_GlyphSlot slot)

Construct a glyph of the correct type.

6.2.1 Detailed Description

FTBitmapFont (p. 26) is a specialisation of the FTFont (p. 37) class for handling Bitmap fonts.

pt3emSee also

FTFont (p. 37)

Definition at line 45 of file FTGLBitmapFont.h.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 FTBitmapFont::FTBitmapFont (const char * fontFilePath)

Open and read a font file.

Sets Error flag.

@param fontFilePath font file path.

6.2.2.2 FTBitmapFont::FTBitmapFont (const unsigned char * pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

Sets Error flag. The buffer is owned by the client and is NOT copied by **FTGL** (p. 21). The pointer must be valid while using **FTGL** (p. 21).

pt3emParameters

pt3em	pt3em	pt3emthe in-memory buffer
рΒι	ufferBytes	
pt3emb	ufferSize-	pt3emthe length of the buffer in bytes
	InBytes	

6.2.2.3 FTBitmapFont::~FTBitmapFont()

Destructor.

6.2.3 Member Function Documentation

6.2.3.1 virtual FTGlyph* FTBitmapFont::MakeGlyph (FT_GlyphSlot *slot*) [protected], [virtual]

Construct a glyph of the correct type.

Clients must override the function and return their specialised FTGlyph (p. 47).

pt3emParameters

pt3em	pt3em <i>slot</i>	pt3emA FreeType glyph slot.
-------	-------------------	-----------------------------

pt3emReturns

An FT****Glyph or null on failure.

Implements FTFont (p. 45).

The documentation for this class was generated from the following file:

FTGLBitmapFont.h

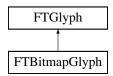
6.3 FTBitmapGlyph Class Reference

FTBitmapGlyph (p. 27) is a specialisation of FTGlyph (p. 47) for creating bitmaps.

#include <FTBitmapGlyph.h>

Inheritance diagram for FTBitmapGlyph:

pt3em



Public Member Functions

• FTBitmapGlyph (FT_GlyphSlot glyph)

Constructor.

virtual ∼FTBitmapGlyph ()

Destructor.

• virtual const FTPoint & Render (const FTPoint &pen, int renderMode)

Render this glyph at the current pen position.

Additional Inherited Members

6.3.1 Detailed Description

FTBitmapGlyph (p. 27) is a specialisation of FTGlyph (p. 47) for creating bitmaps.

Definition at line 42 of file FTBitmapGlyph.h.

6.3.2 Constructor & Destructor Documentation

 $6.3.2.1 \quad \mathsf{FTBitmapGlyph::} \mathsf{FTBitmapGlyph} \left(\begin{array}{c} \mathsf{FT_GlyphSlot} \ \mathit{glyph} \end{array} \right)$

Constructor.

pt3emParameters

pt3em pt3em <i>glyph</i>	pt3emThe Freetype glyph to be processed

6.3.2.2 virtual FTBitmapGlyph:: \sim FTBitmapGlyph () [virtual]

Destructor.

6.3.3 Member Function Documentation

6.3.3.1 virtual const FTPoint& FTBitmapGlyph::Render (const FTPoint & pen, int renderMode) [virtual]

Render this glyph at the current pen position.

pt3em	pt3em <i>pen</i>	pt3emThe current pen position.
	pt3em	pt3emRender mode to display
r	renderMode	

pt3emReturns

The advance distance for this glyph.

Implements FTGlyph (p. 49).

The documentation for this class was generated from the following file:

· FTBitmapGlyph.h

6.4 FTBuffer Class Reference

FTBuffer (p. 29) is a helper class for pixel buffers.

```
#include <FTBuffer.h>
```

Public Member Functions

• FTBuffer ()

Default constructor.

• \sim FTBuffer ()

Destructor.

• FTPoint Pos () const

Get the pen's position in the buffer.

• void Pos (FTPoint arg)

Set the pen's position in the buffer.

void Size (int w, int h)

Set the buffer's size.

· int Width () const

Get the buffer's width.

• int Height () const

Get the buffer's height.

• unsigned char * Pixels () const

Get the buffer's direct pixel buffer.

6.4.1 Detailed Description

FTBuffer (p. 29) is a helper class for pixel buffers.

It provides the interface between FTBufferFont (p. 31) and FTBufferGlyph (p. 33) to optimise rendering operations.

pt3emSee also

```
FTBufferGlyph (p. 33)
FTBufferFont (p. 31)
```

Definition at line 45 of file FTBuffer.h.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 FTBuffer::FTBuffer()

Default constructor.

```
pt3em
6.4.2.2 FTBuffer::~FTBuffer()
Destructor.
6.4.3
       Member Function Documentation
6.4.3.1 int FTBuffer::Height ( ) const [inline]
Get the buffer's height.
pt3emReturns
    The buffer's height, in pixels.
Definition at line 98 of file FTBuffer.h.
6.4.3.2 unsigned char* FTBuffer::Pixels ( ) const [inline]
Get the buffer's direct pixel buffer.
pt3emReturns
    A read-write pointer to the buffer's pixels.
Definition at line 105 of file FTBuffer.h.
6.4.3.3 FTPoint FTBuffer::Pos ( ) const [inline]
Get the pen's position in the buffer.
pt3emReturns
    The pen's position as an FTPoint (p. 59) object.
Definition at line 63 of file FTBuffer.h.
6.4.3.4 void FTBuffer::Pos ( FTPoint arg ) [inline]
Set the pen's position in the buffer.
pt3emParameters
         pt3emarg | pt3emAn FTPoint (p. 59) object with the desired pen's position.
Definition at line 73 of file FTBuffer.h.
```

6.4.3.5 void FTBuffer::Size (int w, int h)

Set the buffer's size.

pt3emParameters

pt3em	pt3emw	pt3emThe buffer's desired width, in pixels.
	pt3em <i>h</i>	pt3emThe buffer's desired height, in pixels.

pt3em

6.4.3.6 int FTBuffer::Width () const [inline]

Get the buffer's width.

pt3emReturns

The buffer's width, in pixels.

Definition at line 91 of file FTBuffer.h.

The documentation for this class was generated from the following file:

· FTBuffer.h

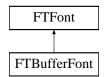
6.5 FTBufferFont Class Reference

FTBufferFont (p. 31) is a specialisation of the FTFont (p. 37) class for handling memory buffer fonts.

#include <FTBufferFont.h>

Inheritance diagram for FTBufferFont:

pt3em



Public Member Functions

• FTBufferFont (const char *fontFilePath)

Open and read a font file.

• FTBufferFont (const unsigned char *pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

• ∼FTBufferFont ()

Destructor.

Protected Member Functions

• virtual FTGlyph * MakeGlyph (FT_GlyphSlot slot)

Construct a glyph of the correct type.

6.5.1 Detailed Description

FTBufferFont (p. 31) is a specialisation of the FTFont (p. 37) class for handling memory buffer fonts.

pt3emSee also

FTFont (p. 37)

Definition at line 43 of file FTBufferFont.h.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 FTBufferFont::FTBufferFont (const char * fontFilePath)

Open and read a font file.

Sets Error flag.

```
@param fontFilePath font file path.
```

6.5.2.2 FTBufferFont::FTBufferFont (const unsigned char * pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

Sets Error flag. The buffer is owned by the client and is NOT copied by **FTGL** (p. 21). The pointer must be valid while using **FTGL** (p. 21).

pt3emParameters

1	t3em pt3em	pt3emthe in-memory buffer
	pBufferBytes	
Ī	pt3embufferSize-	pt3emthe length of the buffer in bytes
	InBytes	

6.5.2.3 FTBufferFont::∼FTBufferFont ()

Destructor.

6.5.3 Member Function Documentation

6.5.3.1 virtual FTGlyph* FTBufferFont::MakeGlyph (FT_GlyphSlot slot) [protected], [virtual]

Construct a glyph of the correct type.

Clients must override the function and return their specialised FTGlyph (p. 47).

pt3em	pt3em <i>slot</i>	pt3emA FreeType glyph slot.

pt3emReturns

An FT****Glyph or null on failure.

Implements FTFont (p. 45).

The documentation for this class was generated from the following file:

FTBufferFont.h

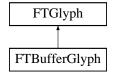
6.6 FTBufferGlyph Class Reference

FTBufferGlyph (p. 33) is a specialisation of FTGlyph (p. 47) for memory buffer rendering.

#include <FTBufferGlyph.h>

Inheritance diagram for FTBufferGlyph:

pt3em



Public Member Functions

• FTBufferGlyph (FT_GlyphSlot glyph, FTBuffer *buffer)

Constructor.

virtual ∼FTBufferGlyph ()

Destructor.

• virtual const FTPoint & Render (const FTPoint &pen, int renderMode)

Render this glyph at the current pen position.

Additional Inherited Members

6.6.1 Detailed Description

FTBufferGlyph (p. 33) is a specialisation of FTGlyph (p. 47) for memory buffer rendering.

Definition at line 40 of file FTBufferGlyph.h.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 FTBufferGlyph::FTBufferGlyph (FT_GlyphSlot glyph, FTBuffer * buffer)

Constructor.

pt3em pt3em <i>glyph</i>	pt3emThe Freetype glyph to be processed
pt3em <i>buffer</i>	pt3emAn FTBuffer (p. 29) object in which to render the glyph.

6.6.2.2 virtual FTBufferGlyph::∼**FTBufferGlyph()** [virtual]

Destructor.

6.6.3 Member Function Documentation

6.6.3.1 virtual const FTPoint& FTBufferGlyph::Render (const FTPoint & pen, int renderMode) [virtual]

Render this glyph at the current pen position.

pt3emParameters

pt3em pt3em <i>pen</i>	pt3emThe current pen position.
pt3em	pt3emRender mode to display
renderMode	

pt3emReturns

The advance distance for this glyph.

Implements FTGlyph (p. 49).

The documentation for this class was generated from the following file:

· FTBufferGlyph.h

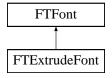
6.7 FTExtrudeFont Class Reference

FTExtrudeFont (p. 34) is a specialisation of the FTFont (p. 37) class for handling extruded Polygon fonts.

#include <FTGLExtrdFont.h>

Inheritance diagram for FTExtrudeFont:

pt3em



Public Member Functions

• FTExtrudeFont (const char *fontFilePath)

Open and read a font file.

• FTExtrudeFont (const unsigned char *pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

 $\bullet \ \sim \! \textbf{FTExtrudeFont} \ ()$

Destructor.

Protected Member Functions

virtual FTGlyph * MakeGlyph (FT_GlyphSlot slot)
 Construct a glyph of the correct type.

6.7.1 Detailed Description

FTExtrudeFont (p. 34) is a specialisation of the FTFont (p. 37) class for handling extruded Polygon fonts.

pt3emSee also

```
FTFont (p. 37)
FTPolygonFont (p. 65)
```

Definition at line 46 of file FTGLExtrdFont.h.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 FTExtrudeFont::FTExtrudeFont (const char * fontFilePath)

Open and read a font file.

Sets Error flag.

```
@param fontFilePath font file path.
```

6.7.2.2 FTExtrudeFont::FTExtrudeFont (const unsigned char * pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

Sets Error flag. The buffer is owned by the client and is NOT copied by **FTGL** (p. 21). The pointer must be valid while using **FTGL** (p. 21).

pt3emParameters

pt3em	pt3em	pt3emthe in-memory buffer
рΒι	ıfferBytes	
pt3emb	ufferSize-	pt3emthe length of the buffer in bytes
	InBytes	

6.7.2.3 FTExtrudeFont::~FTExtrudeFont()

Destructor.

6.7.3 Member Function Documentation

6.7.3.1 virtual FTGlyph* FTExtrudeFont::MakeGlyph (FT_GlyphSlot slot) [protected], [virtual]

Construct a glyph of the correct type.

Clients must override the function and return their specialised FTGlyph (p. 47).

pt3emParameters

pt3em pt3emslot pt3emA FreeType glyph slot.

pt3emReturns

An FT****Glyph or null on failure.

Implements FTFont (p. 45).

The documentation for this class was generated from the following file:

FTGLExtrdFont.h

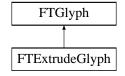
6.8 FTExtrudeGlyph Class Reference

FTExtrudeGlyph (p. 36) is a specialisation of FTGlyph (p. 47) for creating tessellated extruded polygon glyphs.

#include <FTExtrdGlyph.h>

Inheritance diagram for FTExtrudeGlyph:

pt3em



Public Member Functions

- FTExtrudeGlyph (FT_GlyphSlot glyph, float depth, float frontOutset, float backOutset, bool useDisplayList)
 Constructor.
- virtual ∼FTExtrudeGlyph ()

Destructor.

• virtual const **FTPoint** & **Render** (const **FTPoint** &pen, int renderMode)

Render this glyph at the current pen position.

Additional Inherited Members

6.8.1 Detailed Description

FTExtrudeGlyph (p. 36) is a specialisation of **FTGlyph** (p. 47) for creating tessellated extruded polygon glyphs. Definition at line 43 of file FTExtrdGlyph.h.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 FTExtrudeGlyph::FTExtrudeGlyph (FT_GlyphSlot *glyph*, float *depth*, float *frontOutset*, float *backOutset*, bool *useDisplayList*)

Constructor.

Sets the Error to Invalid_Outline if the glyph isn't an outline.

pt3emParameters

pt3em pt3emglyph	pt3emThe Freetype glyph to be processed
pt3em <i>depth</i>	pt3emThe distance along the z axis to extrude the glyph
pt3em	pt3emoutset contour size
frontOutset	
pt3em	pt3emoutset contour size
backOutset	
pt3em	pt3emEnable or disable the use of Display Lists for this glyph true turns ON display lists.
useDisplayList	false turns OFF display lists.

6.8.2.2 virtual FTExtrudeGlyph::~FTExtrudeGlyph() [virtual]

Destructor.

6.8.3 Member Function Documentation

6.8.3.1 virtual const FTPoint& FTExtrudeGlyph::Render (const FTPoint & pen, int renderMode) [virtual]

Render this glyph at the current pen position.

pt3emParameters

pt3em	pt3em <i>pen</i>	pt3emThe current pen position.
	pt3em	pt3emRender mode to display
	renderMode	

pt3emReturns

The advance distance for this glyph.

Implements FTGlyph (p. 49).

The documentation for this class was generated from the following file:

· FTExtrdGlyph.h

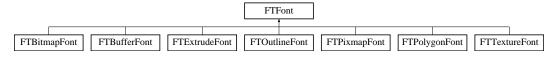
6.9 FTFont Class Reference

FTFont (p. 37) is the public interface for the FTGL (p. 21) library.

#include <FTFont.h>

Inheritance diagram for FTFont:

pt3em



Public Member Functions

- virtual ∼FTFont ()
- virtual bool Attach (const char *fontFilePath)

Attach auxilliary file to font e.g font metrics.

virtual bool Attach (const unsigned char *pBufferBytes, size_t bufferSizeInBytes)

Attach auxilliary data to font e.g font metrics, from memory.

virtual void GlyphLoadFlags (FT Int flags)

Set the glyph loading flags.

virtual bool CharMap (FT_Encoding encoding)

Set the character map for the face.

· virtual unsigned int CharMapCount () const

Get the number of character maps in this face.

virtual FT_Encoding * CharMapList ()

Get a list of character maps in this face.

virtual bool FaceSize (const unsigned int size, const unsigned int res=72)

Set the char size for the current face.

· virtual unsigned int FaceSize () const

Get the current face size in points (1/72 inch).

· virtual void Depth (float depth)

Set the extrusion distance for the font.

· virtual void Outset (float outset)

Set the outset distance for the font.

· virtual void Outset (float front, float back)

Set the front and back outset distances for the font.

virtual void UseDisplayList (bool useList)

Enable or disable the use of Display Lists inside FTGL (p. 21).

virtual float Ascender () const

Get the global ascender height for the face.

· virtual float Descender () const

Gets the global descender height for the face.

virtual float LineHeight () const

Gets the line spacing for the font.

virtual FTBBox BBox (const char *string, const int len=-1, FTPoint position=FTPoint(), FTPoint spacing=F-TPoint())

Get the bounding box for a string.

• void **BBox** (const char *string, float &llx, float &llx, float &llx, float &urx, float &ury, float &urz)

Get the bounding box for a string (deprecated).

virtual FTBBox BBox (const wchar_t *string, const int len=-1, FTPoint position=FTPoint(), FTPoint spacing=FTPoint())

Get the bounding box for a string.

• void BBox (const wchar_t *string, float &llx, float &llx, float &llx, float &urx, float &ury, float &urz)

Get the bounding box for a string (deprecated).

• virtual float **Advance** (const char *string, const int len=-1, **FTPoint** spacing=**FTPoint**())

Get the advance for a string.

• virtual float Advance (const wchar t *string, const int len=-1, FTPoint spacing=FTPoint())

Get the advance for a string.

 virtual FTPoint Render (const char *string, const int len=-1, FTPoint position=FTPoint(), FTPoint spacing=F-TPoint(), int renderMode=FTGL::RENDER_ALL)

Render a string of characters.

virtual FTPoint Render (const wchar_t *string, const int len=-1, FTPoint position=FTPoint(), FTPoint spacing=FTPoint(), int renderMode=FTGL::RENDER ALL)

Render a string of characters.

· virtual FT_Error Error () const

Queries the Font for errors.

Protected Member Functions

FTFont (char const *fontFilePath)

Open and read a font file.

• FTFont (const unsigned char *pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

virtual FTGlyph * MakeGlyph (FT GlyphSlot slot)=0

Construct a glyph of the correct type.

Friends

- class FTBitmapFont
- · class FTBufferFont
- · class FTExtrudeFont
- · class FTOutlineFont
- class FTPixmapFont
- · class FTPolygonFont
- class FTTextureFont
- class FTFontImpl

6.9.1 Detailed Description

FTFont (p. 37) is the public interface for the FTGL (p. 21) library.

Specific font classes are derived from this class. It uses the helper classes FTFace and FTSize to access the Freetype library. This class is abstract and deriving classes must implement the protected MakeGlyph function to create glyphs of the appropriate type.

It is good practice after using these functions to test the error code returned. FT_Error **Error()** (p. 44). Check the freetype file fterrdef.h for error definitions.

pt3emSee also

FTFace

FTSize

Definition at line 56 of file FTFont.h.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 FTFont::FTFont (char const * fontFilePath) [protected]

Open and read a font file.

Sets Error flag.

@param fontFilePath font file path.

6.9.2.2 FTFont::FTFont (const unsigned char * pBufferBytes, size_t bufferSizeInBytes) [protected]

Open and read a font from a buffer in memory.

Sets Error flag. The buffer is owned by the client and is NOT copied by **FTGL** (p. 21). The pointer must be valid while using **FTGL** (p. 21).

pt3emParameters

pt3em	pt3em	pt3emthe in-memory buffer
рΒι	ufferBytes	
pt3emb	ufferSize-	pt3emthe length of the buffer in bytes
	InBytes	

6.9.2.3 virtual FTFont::∼**FTFont()** [virtual]

6.9.3 Member Function Documentation

6.9.3.1 virtual float FTFont::Advance (const char * string, const int len = -1, FTPoint spacing = FTPoint ()) [virtual]

Get the advance for a string.

pt3emParameters

pt3em pt3em <i>string</i>	pt3em'C' style string to be checked.
pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be checked until a null character
	is encountered (optional).
pt3em <i>spacing</i>	pt3emA displacement vector to add after each character has been checked (optional).

pt3emReturns

The string's advance width.

6.9.3.2 virtual float FTFont::Advance (const wchar_t * string, const int len = -1, FTPoint spacing = FTPoint ()) [virtual]

Get the advance for a string.

pt3emParameters

pt3em pt3em <i>string</i>	pt3emA wchar_t string
pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be checked until a null character
	is encountered (optional).
pt3em <i>spacing</i>	pt3emA displacement vector to add after each character has been checked (optional).

pt3emReturns

The string's advance width.

6.9.3.3 virtual float FTFont::Ascender () const [virtual]

Get the global ascender height for the face.

pt3emReturns

Ascender height

6.9.3.4 virtual bool FTFont::Attach (const char * *fontFilePath* **)** [virtual]

Attach auxilliary file to font e.g font metrics.

Note: not all font formats implement this function.

pt3emParameters

pt3em	pt3em	pt3emauxilliary font file path.
	fontFilePath	

pt3emReturns

true if file has been attached successfully.

6.9.3.5 virtual bool FTFont::Attach (const unsigned char * pBufferBytes, size_t bufferSizeInBytes) [virtual]

Attach auxilliary data to font e.g font metrics, from memory.

Note: not all font formats implement this function.

pt3emParameters

pt3em	pt3em	pt3emthe in-memory buffer.
pBufi	ferBytes	
pt3em <i>but</i>	fferSize-	pt3emthe length of the buffer in bytes.
	InBytes	

pt3emReturns

true if file has been attached successfully.

6.9.3.6 virtual FTBBox FTFont::BBox (const char * string, const int len = -1, FTPoint position = FTPoint (), FTPoint spacing = FTPoint () | [virtual]

Get the bounding box for a string.

pt3em pt3em <i>string</i>	pt3emA char buffer.
pt3em <i>len</i>	pt3emThe length of the string. If < 0 then all characters will be checked until a null character
	is encountered (optional).
pt3emposition	pt3emThe pen position of the first character (optional).
pt3em <i>spacing</i>	pt3emA displacement vector to add after each character has been checked (optional).

pt3em

pt3emReturns

The corresponding bounding box.

Referenced by BBox().

6.9.3.7 void FTFont::BBox (const char * string, float & llx, float & llx, float & llx, float & urx, float & urx, float & urx, float & urx)
[inline]

Get the bounding box for a string (deprecated).

pt3emParameters

pt3em pt3em <i>string</i>	pt3emA char buffer.
pt3em/lx	pt3emLower left near x coordinate.
pt3em <i>lly</i>	pt3emLower left near y coordinate.
pt3em/lz	pt3emLower left near z coordinate.
pt3em <i>urx</i>	pt3emUpper right far x coordinate.
pt3em <i>ury</i>	pt3emUpper right far y coordinate.
pt3emurz	pt3emUpper right far z coordinate.

Definition at line 251 of file FTFont.h.

References BBox(), FTBBox::Lower(), FTBBox::Upper(), FTPoint::Xf(), FTPoint::Yf(), and FTPoint::Zf().

6.9.3.8 virtual FTBBox FTFont::BBox (const wchar_t * string, const int len = -1, FTPoint position = FTPoint (), FTPoint spacing = FTPoint () | [virtual]

Get the bounding box for a string.

pt3emParameters

pt3em pt3em <i>string</i>	pt3emA wchar_t buffer.
pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be checked until a null character
	is encountered (optional).
pt3emposition	pt3emThe pen position of the first character (optional).
pt3em <i>spacing</i>	pt3emA displacement vector to add after each character has been checked (optional).

pt3emReturns

The corresponding bounding box.

6.9.3.9 void FTFont::BBox (const wchar_t * string, float & llx, float & llx, float & llx, float & urx, float & urx, float & urx, float & urx)
[inline]

Get the bounding box for a string (deprecated).

pt3em pt3em <i>string</i>	pt3emA wchar_t buffer.
pt3em/lx	pt3emLower left near x coordinate.

ptoonii	
pt3em <i>lly</i>	pt3emLower left near y coordinate.
pt3em/lz	pt3emLower left near z coordinate.
pt3em <i>urx</i>	pt3emUpper right far x coordinate.
pt3em <i>ury</i>	pt3emUpper right far y coordinate.
pt3em <i>urz</i>	pt3emUpper right far z coordinate.

Definition at line 286 of file FTFont.h.

References BBox(), FTBBox::Lower(), FTBBox::Upper(), FTPoint::Xf(), FTPoint::Yf(), and FTPoint::Zf().

6.9.3.10 virtual bool FTFont::CharMap (FT_Encoding encoding) [virtual]

Set the character map for the face.

pt3emParameters

pt3 pt3 em <i>encoding</i>	pt3emFreetype enumerate for char map code.
-----------------------------------	--

pt3emReturns

true if charmap was valid and set correctly.

6.9.3.11 virtual unsigned int FTFont::CharMapCount() const [virtual]

Get the number of character maps in this face.

pt3emReturns

character map count.

6.9.3.12 virtual FT_Encoding* FTFont::CharMapList() [virtual]

Get a list of character maps in this face.

pt3emReturns

pointer to the first encoding.

6.9.3.13 virtual void FTFont::Depth (float depth) [virtual]

Set the extrusion distance for the font.

Only implemented by FTExtrudeFont (p. 34)

pt3emParameters

pt3em pt3em*depth* pt3emThe extrusion distance.

```
6.9.3.14 virtual float FTFont::Descender ( ) const [virtual]
```

Gets the global descender height for the face.

pt3emReturns

Descender height

```
6.9.3.15 virtual FT_Error FTFont::Error() const [virtual]
```

Queries the Font for errors.

pt3emReturns

The current error code.

6.9.3.16 virtual bool FTFont::FaceSize (const unsigned int *size*, const unsigned int *res* = 72) [virtual]

Set the char size for the current face.

pt3emParameters

pt3em	pt3em <i>size</i>	pt3emthe face size in points (1/72 inch)
	pt3em <i>res</i>	pt3emthe resolution of the target device.

pt3emReturns

true if size was set correctly

```
6.9.3.17 virtual unsigned int FTFont::FaceSize ( ) const [virtual]
```

Get the current face size in points (1/72 inch).

pt3emReturns

face size

6.9.3.18 virtual void FTFont::GlyphLoadFlags (FT_Int *flags* **)** [virtual]

Set the glyph loading flags.

By default, fonts use the most sensible flags when loading a font's glyph using FT_Load_Glyph(). This function allows to override the default flags.

```
pt3em pt3emflags pt3emThe glyph loading flags.
```

6.9.3.19 virtual float FTFont::LineHeight() const [virtual]

Gets the line spacing for the font.

pt3emReturns

Line height

6.9.3.20 virtual FTGlyph* FTFont::MakeGlyph (FT_GlyphSlot slot) [protected], [pure virtual]

Construct a glyph of the correct type.

Clients must override the function and return their specialised FTGlyph (p. 47).

pt3emParameters

pt3em pt3em*slot* pt3emA FreeType glyph slot.

pt3emReturns

An FT****Glyph or null on failure.

Implemented in FTExtrudeFont (p. 35), FTBitmapFont (p. 27), FTOutlineFont (p. 54), FTPixmapFont (p. 58), FTPolygonFont (p. 67), FTTextureFont (p. 74), and FTBufferFont (p. 32).

6.9.3.21 virtual void FTFont::Outset (float outset) [virtual]

Set the outset distance for the font.

Only implemented by FTOutlineFont (p. 53), FTPolygonFont (p. 65) and FTExtrudeFont (p. 34)

pt3emParameters

pt3empt3emoutset pt3emThe outset distance.

6.9.3.22 virtual void FTFont::Outset (float front, float back) [virtual]

Set the front and back outset distances for the font.

Only implemented by FTExtrudeFont (p. 34)

pt3emParameters

pt3em pt3emfr	ont pt3emThe front outset distance.	
pt3em <i>ba</i>	ack pt3emThe back outset distance.	

6.9.3.23 virtual FTPoint FTFont::Render (const char * string, const int len = -1, FTPoint position = FTPoint (), FTPoint spacing = FTPoint (), int renderMode = FTGL::RENDER_ALL) [virtual]

Render a string of characters.

pt3emParameters

pt3em pt3em <i>string</i>	pt3em'C' style string to be output.
pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be displayed until a null character
	is encountered (optional).
pt3emposition	pt3emThe pen position of the first character (optional).
pt3em <i>spacing</i>	pt3emA displacement vector to add after each character has been displayed (optional).
pt3em	pt3emRender mode to use for display (optional).
renderMode	

pt3emReturns

The new pen position after the last character was output.

```
6.9.3.24 virtual FTPoint FTFont::Render ( const wchar_t * string, const int len = -1, FTPoint position = FTPoint (), FTPoint spacing = FTPoint (), int renderMode = FTGL::RENDER_ALL ) [virtual]
```

Render a string of characters.

pt3emParameters

pt3em pt3em <i>string</i>	pt3emwchar_t string to be output.
pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be displayed until a null character
	is encountered (optional).
pt3emposition	pt3emThe pen position of the first character (optional).
pt3em <i>spacing</i>	pt3emA displacement vector to add after each character has been displayed (optional).
pt3em	pt3emRender mode to use for display (optional).
renderMode	

pt3emReturns

The new pen position after the last character was output.

```
6.9.3.25 virtual void FTFont::UseDisplayList (bool useList) [virtual]
```

Enable or disable the use of Display Lists inside FTGL (p. 21).

pt3emParameters

pt3empt3emuseList pt3emtrue turns ON display lists. false turns OFF display lists.	
--	--

6.9.4 Friends And Related Function Documentation

6.9.4.1 friend class FTBitmapFont [friend]

Definition at line 78 of file FTFont.h.

6.9.4.2 friend class FTBufferFont [friend]

Definition at line 79 of file FTFont.h.

6.9.4.3 friend class FTExtrudeFont [friend]

Definition at line 80 of file FTFont.h.

6.9.4.4 friend class FTFontImpl [friend]

Definition at line 367 of file FTFont.h.

6.9.4.5 friend class FTOutlineFont [friend]

Definition at line 81 of file FTFont.h.

6.9.4.6 friend class FTPixmapFont [friend]

Definition at line 82 of file FTFont.h.

6.9.4.7 friend class FTPolygonFont [friend]

Definition at line 83 of file FTFont.h.

6.9.4.8 friend class FTTextureFont [friend]

Definition at line 84 of file FTFont.h.

The documentation for this class was generated from the following file:

• FTFont.h

6.10 FTGlyph Class Reference

FTGlyph (p. 47) is the base class for FTGL (p. 21) glyphs.

#include <FTGlyph.h>

Inheritance diagram for FTGlyph:

pt3em



Public Member Functions

- virtual \sim FTGlyph ()
 - Destructor.
- virtual const FTPoint & Render (const FTPoint &pen, int renderMode)=0

Renders this glyph at the current pen position.

· virtual float Advance () const

Return the advance width for this glyph.

virtual const FTBBox & BBox () const

Return the bounding box for this glyph.

• virtual FT_Error Error () const

Queries for errors.

Protected Member Functions

• FTGlyph (FT_GlyphSlot glyph)

Create a glyph.

Friends

- · class FTBitmapGlyph
- class FTBufferGlyph
- class FTExtrudeGlyph
- · class FTOutlineGlyph
- class FTPixmapGlyph
- · class FTPolygonGlyph
- class FTTextureGlyph

6.10.1 Detailed Description

FTGlyph (p. 47) is the base class for FTGL (p. 21) glyphs.

It provides the interface between Freetype glyphs and their openGL renderable counterparts. This is an abstract class and derived classes must implement the Render function.

pt3emSee also

```
FTBBox (p. 23)
FTPoint (p. 59)
```

Definition at line 50 of file FTGlyph.h.

6.10.2 Constructor & Destructor Documentation

```
6.10.2.1 FTGlyph::FTGlyph (FT_GlyphSlot glyph ) [protected]
```

Create a glyph.

pt3emParameters

```
pt3em pt3emglyph | pt3emThe Freetype glyph to be processed
```

```
6.10.2.2 virtual FTGlyph::∼FTGlyph() [virtual]
```

Destructor.

6.10.3 Member Function Documentation

6.10.3.1 virtual float FTGlyph::Advance () const [virtual]

Return the advance width for this glyph.

pt3emReturns

advance width.

6.10.3.2 virtual const FTBBox& FTGlyph::BBox () const [virtual]

Return the bounding box for this glyph.

pt3emReturns

bounding box.

6.10.3.3 virtual FT_Error FTGlyph::Error () const [virtual]

Queries for errors.

pt3emReturns

The current error code.

6.10.3.4 virtual const FTPoint& FTGlyph::Render (const FTPoint & pen, int renderMode) [pure virtual]

Renders this glyph at the current pen position.

pt3emParameters

pt	3em	pt3em <i>pen</i>	pt3emThe current pen position.
		pt3em	pt3emRender mode to display
	re	enderMode	

pt3emReturns

The advance distance for this glyph.

Implemented in FTExtrudeGlyph (p. 37), FTTextureGlyph (p. 75), FTPolygonGlyph (p. 68), FTOutlineGlyph (p. 56), FTBitmapGlyph (p. 28), FTPixmapGlyph (p. 59), and FTBufferGlyph (p. 34).

6.10.4 Friends And Related Function Documentation

6.10.4.1 friend class FTBitmapGlyph [friend]

Definition at line 70 of file FTGlyph.h.

6.10.4.2 friend class FTBufferGlyph [friend]

Definition at line 71 of file FTGlyph.h.

6.10.4.3 friend class FTExtrudeGlyph [friend]

Definition at line 72 of file FTGlyph.h.

6.10.4.4 friend class FTOutlineGlyph [friend]

Definition at line 73 of file FTGlyph.h.

6.10.4.5 friend class FTPixmapGlyph [friend]

Definition at line 74 of file FTGlyph.h.

6.10.4.6 friend class FTPolygonGlyph [friend]

Definition at line 75 of file FTGlyph.h.

6.10.4.7 friend class FTTextureGlyph [friend]

Definition at line 76 of file FTGlyph.h.

The documentation for this class was generated from the following file:

• FTGlyph.h

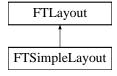
6.11 FTLayout Class Reference

FTLayout (p. 50) is the interface for layout managers that render text.

#include <FTLayout.h>

Inheritance diagram for FTLayout:

pt3em



Public Member Functions

• virtual \sim FTLayout ()

Destructor.

• virtual **FTBBox BBox** (const char *string, const int len=-1, **FTPoint** position=**FTPoint**())=0 Get the bounding box for a formatted string.

- virtual FTBBox BBox (const wchar_t *string, const int len=-1, FTPoint position=FTPoint())=0
 Get the bounding box for a formatted string.
- virtual void Render (const char *string, const int len=-1, FTPoint position=FTPoint(), int renderMode=FTG-L::RENDER ALL)=0

Render a string of characters.

virtual void Render (const wchar_t *string, const int len=-1, FTPoint position=FTPoint(), int renderMode=F-TGL::RENDER ALL)=0

Render a string of characters.

virtual FT_Error Error () const

Queries the Layout for errors.

Protected Member Functions

• FTLayout ()

Friends

· class FTSimpleLayout

6.11.1 Detailed Description

FTLayout (p. 50) is the interface for layout managers that render text.

Specific layout manager classes are derived from this class. This class is abstract and deriving classes must implement the protected Render methods to render formatted text and BBox methods to determine the bounding box of output text.

pt3emSee also

```
FTFont (p. 37)
FTBBox (p. 23)
```

Definition at line 52 of file FTLayout.h.

6.11.2 Constructor & Destructor Documentation

```
6.11.2.1 FTLayout::FTLayout( ) [protected]
6.11.2.2 virtual FTLayout::~FTLayout( ) [virtual]
```

Destructor.

6.11.3 Member Function Documentation

```
6.11.3.1 virtual FTBBox FTLayout::BBox ( const char * string, const int len = -1, FTPoint position = FTPoint () )

[pure virtual]
```

Get the bounding box for a formatted string.

pt3emParameters

pt3em pt3em <i>string</i>	pt3emA char string.
pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be checked until a null character
	is encountered (optional).
pt3emposition	pt3emThe pen position of the first character (optional).

pt3emReturns

The corresponding bounding box.

Implemented in FTSimpleLayout (p. 70).

6.11.3.2 virtual FTBBox FTLayout::BBox (const wchar_t * string, const int len = -1, FTPoint position = FTPoint ())

[pure virtual]

Get the bounding box for a formatted string.

pt3emParameters

pt3em pt3em <i>string</i>	pt3emA wchar_t string.
pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be checked until a null character
	is encountered (optional).
pt3emposition	pt3emThe pen position of the first character (optional).

pt3emReturns

The corresponding bounding box.

Implemented in FTSimpleLayout (p. 70).

6.11.3.3 virtual FT_Error FTLayout::Error () const [virtual]

Queries the Layout for errors.

pt3emReturns

The current error code.

6.11.3.4 virtual void FTLayout::Render (const char * string, const int len = -1, FTPoint position = FTPoint (), int renderMode = FTGL::RENDER_ALL) [pure virtual]

Render a string of characters.

pt3em pt3em <i>string</i>	pt3em'C' style string to be output.
pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be displayed until a null character
	is encountered (optional).
pt3emposition	pt3emThe pen position of the first character (optional).
pt3em	pt3emRender mode to display (optional)
renderMode	

Implemented in FTSimpleLayout (p. 71).

6.11.3.5 virtual void FTLayout::Render (const wchar_t * string, const int len = -1, FTPoint position = FTPoint (), int renderMode = FTGL::RENDER_ALL) [pure virtual]

Render a string of characters.

pt3emParameters

pt3em pt3em <i>string</i>	pt3emwchar_t string to be output.
pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be displayed until a null character
	is encountered (optional).
pt3emposition	pt3emThe pen position of the first character (optional).
pt3em	pt3emRender mode to display (optional)
renderMode	

Implemented in FTSimpleLayout (p. 71).

6.11.4 Friends And Related Function Documentation

6.11.4.1 friend class FTSimpleLayout [friend]

Definition at line 67 of file FTLayout.h.

The documentation for this class was generated from the following file:

FTLayout.h

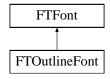
6.12 FTOutlineFont Class Reference

FTOutlineFont (p. 53) is a specialisation of the FTFont (p. 37) class for handling Vector Outline fonts.

#include <FTGLOutlineFont.h>

Inheritance diagram for FTOutlineFont:

pt3em



Public Member Functions

• FTOutlineFont (const char *fontFilePath)

Open and read a font file.

• FTOutlineFont (const unsigned char *pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

• ∼FTOutlineFont ()

Destructor.

Protected Member Functions

virtual FTGlyph * MakeGlyph (FT_GlyphSlot slot)
 Construct a glyph of the correct type.

6.12.1 Detailed Description

FTOutlineFont (p. 53) is a specialisation of the FTFont (p. 37) class for handling Vector Outline fonts.

pt3emSee also

FTFont (p. 37)

Definition at line 45 of file FTGLOutlineFont.h.

6.12.2 Constructor & Destructor Documentation

```
6.12.2.1 FTOutlineFont::FTOutlineFont ( const char * fontFilePath )
```

Open and read a font file.

Sets Error flag.

```
@param fontFilePath font file path.
```

6.12.2.2 FTOutlineFont::FTOutlineFont (const unsigned char * pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

Sets Error flag. The buffer is owned by the client and is NOT copied by **FTGL** (p. 21). The pointer must be valid while using **FTGL** (p. 21).

pt3emParameters

pt3em	pt3em	pt3emthe in-memory buffer
pВ	ufferBytes	
pt3em <i>b</i>	oufferSize-	pt3emthe length of the buffer in bytes
	InBytes	

```
6.12.2.3 FTOutlineFont::~FTOutlineFont()
```

Destructor.

6.12.3 Member Function Documentation

```
6.12.3.1 virtual FTGlyph* FTOutlineFont::MakeGlyph(FT_GlyphSlot slot) [protected], [virtual]
```

Construct a glyph of the correct type.

Clients must override the function and return their specialised FTGlyph (p. 47).

pt3emParameters

pt3em pt3em*slot* pt3emA FreeType glyph slot.

pt3emReturns

An FT****Glyph or null on failure.

Implements FTFont (p. 45).

The documentation for this class was generated from the following file:

· FTGLOutlineFont.h

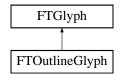
6.13 FTOutlineGlyph Class Reference

FTOutlineGlyph (p. 55) is a specialisation of FTGlyph (p. 47) for creating outlines.

#include <FTOutlineGlyph.h>

Inheritance diagram for FTOutlineGlyph:

pt3em



Public Member Functions

- FTOutlineGlyph (FT_GlyphSlot glyph, float outset, bool useDisplayList)
 - Constructor.
- virtual ∼FTOutlineGlyph ()

Destructor.

• virtual const FTPoint & Render (const FTPoint &pen, int renderMode)

Render this glyph at the current pen position.

Additional Inherited Members

6.13.1 Detailed Description

FTOutlineGlyph (p. 55) is a specialisation of FTGlyph (p. 47) for creating outlines.

Definition at line 42 of file FTOutlineGlyph.h.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 FTOutlineGlyph::FTOutlineGlyph (FT_GlyphSlot glyph, float outset, bool useDisplayList)

Constructor.

Sets the Error to Invalid_Outline if the glyphs isn't an outline.

pt3emParameters

ot3em pt3em <i>glyph</i>	pt3emThe Freetype glyph to be processed
pt3em <i>outset</i>	pt3emoutset distance
pt3em	pt3emEnable or disable the use of Display Lists for this glyph true turns ON display lists.
useDisplayList	false turns OFF display lists.

6.13.2.2 virtual FTOutlineGlyph::~FTOutlineGlyph() [virtual]

Destructor.

6.13.3 Member Function Documentation

6.13.3.1 virtual const FTPoint& FTOutlineGlyph::Render (const FTPoint & pen, int renderMode) [virtual]

Render this glyph at the current pen position.

pt3emParameters

þ	t3em pt3em <i>pen</i>	pt3emThe current pen position.
	pt3em	pt3emRender mode to display
	renderMode	

pt3emReturns

The advance distance for this glyph.

Implements FTGlyph (p. 49).

The documentation for this class was generated from the following file:

· FTOutlineGlyph.h

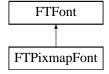
6.14 FTPixmapFont Class Reference

FTPixmapFont (p. 56) is a specialisation of the FTFont (p. 37) class for handling Pixmap (Grey Scale) fonts.

#include <FTGLPixmapFont.h>

Inheritance diagram for FTPixmapFont:

pt3em



Public Member Functions

FTPixmapFont (const char *fontFilePath)

Open and read a font file.

• FTPixmapFont (const unsigned char *pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

• ∼FTPixmapFont ()

Destructor.

Protected Member Functions

virtual FTGlyph * MakeGlyph (FT_GlyphSlot slot)

Construct a glyph of the correct type.

6.14.1 Detailed Description

FTPixmapFont (p. 56) is a specialisation of the FTFont (p. 37) class for handling Pixmap (Grey Scale) fonts.

pt3emSee also

FTFont (p. 37)

Definition at line 45 of file FTGLPixmapFont.h.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 FTPixmapFont::FTPixmapFont (const char * fontFilePath)

Open and read a font file.

Sets Error flag.

```
@param fontFilePath font file path.
```

6.14.2.2 FTPixmapFont::FTPixmapFont (const unsigned char * pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

Sets Error flag. The buffer is owned by the client and is NOT copied by **FTGL** (p. 21). The pointer must be valid while using **FTGL** (p. 21).

pt3emParameters

pt3em	pt3em	pt3emthe in-memory buffer
рВ	ufferBytes	
pt3emb	ufferSize-	pt3emthe length of the buffer in bytes
	InBytes	

6.14.2.3 FTPixmapFont::~FTPixmapFont()

Destructor.

6.14.3 Member Function Documentation

6.14.3.1 virtual FTGlyph* FTPixmapFont::MakeGlyph (FT_GlyphSlot slot) [protected], [virtual]

Construct a glyph of the correct type.

Clients must override the function and return their specialised FTGlyph (p. 47).

pt3emParameters

pt3em pt3em*slot* pt3emA FreeType glyph slot.

pt3emReturns

An FT****Glyph or null on failure.

Implements FTFont (p. 45).

The documentation for this class was generated from the following file:

FTGLPixmapFont.h

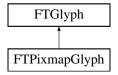
6.15 FTPixmapGlyph Class Reference

FTPixmapGlyph (p. 58) is a specialisation of FTGlyph (p. 47) for creating pixmaps.

#include <FTPixmapGlyph.h>

Inheritance diagram for FTPixmapGlyph:

pt3em



Public Member Functions

• FTPixmapGlyph (FT_GlyphSlot glyph)

Constructor.

• virtual \sim FTPixmapGlyph ()

Destructor.

• virtual const FTPoint & Render (const FTPoint &pen, int renderMode)

Render this glyph at the current pen position.

Additional Inherited Members

6.15.1 Detailed Description

FTPixmapGlyph (p. 58) is a specialisation of FTGlyph (p. 47) for creating pixmaps.

Definition at line 42 of file FTPixmapGlyph.h.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 FTPixmapGlyph::FTPixmapGlyph (FT_GlyphSlot glyph)

Constructor.

pt3emParameters

pt3em pt3emglyph | pt3emThe Freetype glyph to be processed

6.15.2.2 virtual FTPixmapGlyph::~FTPixmapGlyph() [virtual]

Destructor.

6.15.3 Member Function Documentation

6.15.3.1 virtual const FTPoint& FTPixmapGlyph::Render (const FTPoint & pen, int renderMode) [virtual]

Render this glyph at the current pen position.

pt3emParameters

pt3em pt3	3em <i>pen</i>	pt3emThe current pen position.
	pt3em	pt3emRender mode to display
rend	derMode	

pt3emReturns

The advance distance for this glyph.

Implements FTGlyph (p. 49).

The documentation for this class was generated from the following file:

· FTPixmapGlyph.h

6.16 FTPoint Class Reference

FTPoint (p. 59) class is a basic 3-dimensional point or vector.

#include <FTPoint.h>

Public Member Functions

• FTPoint ()

Default constructor.

• FTPoint (const FTGL_DOUBLE x, const FTGL_DOUBLE y, const FTGL_DOUBLE z=0)

Constructor.

• FTPoint (const FT_Vector &ft_vector)

Constructor.

FTPoint Normalise ()

Normalise a point's coordinates.

FTPoint & operator+= (const FTPoint &point)

Operator += In Place Addition.

• FTPoint operator+ (const FTPoint &point) const

Operator +.

FTPoint & operator-= (const FTPoint &point)

Operator -= In Place Substraction.

• FTPoint operator- (const FTPoint &point) const

Operator -.

FTPoint operator* (double multiplier) const

Operator * Scalar multiplication.

FTPoint operator^{\(\chi\)} (const FTPoint &point)

Operator \(^\) Vector product.

• operator const FTGL_DOUBLE * () const

Cast to FTGL_DOUBLE*.

• void X (FTGL_DOUBLE x)

Setters.

- void Y (FTGL_DOUBLE y)
- void **Z** (**FTGL_DOUBLE** z)
- FTGL_DOUBLE X () const

Getters.

- FTGL_DOUBLE Y () const
- FTGL_DOUBLE Z () const
- FTGL_FLOAT Xf () const
- FTGL_FLOAT Yf () const
- FTGL FLOAT Zf () const

Friends

• FTPoint operator* (double multiplier, FTPoint &point)

Operator * Scalar multiplication.

double operator* (FTPoint &a, FTPoint &b)

Operator * Scalar product.

• bool operator== (const FTPoint &a, const FTPoint &b)

Operator == Tests for equality.

bool operator!= (const FTPoint &a, const FTPoint &b)

Operator != Tests for non equality.

6.16.1 Detailed Description

FTPoint (p. 59) class is a basic 3-dimensional point or vector.

Definition at line 42 of file FTPoint.h.

```
6.16.2 Constructor & Destructor Documentation
```

```
6.16.2.1 FTPoint::FTPoint( ) [inline]
```

Default constructor.

Point is set to zero.

Definition at line 48 of file FTPoint.h.

```
6.16.2.2 FTPoint::FTPoint ( const FTGL_DOUBLE x, const FTGL_DOUBLE z = 0 )
[inline]
```

Constructor.

Z coordinate is set to zero if unspecified.

```
@param x First component
@param y Second component
@param z Third component
```

Definition at line 62 of file FTPoint.h.

```
6.16.2.3 FTPoint::FTPoint (const FT_Vector & ft_vector) [inline]
```

Constructor.

This converts an FT_Vector to an FTPoint (p. 59)

```
@param ft_vector A freetype vector
```

Definition at line 75 of file FTPoint.h.

6.16.3 Member Function Documentation

```
6.16.3.1 FTPoint FTPoint::Normalise ( )
```

Normalise a point's coordinates.

If the coordinates are zero, the point is left untouched.

pt3emReturns

A vector of norm one.

```
6.16.3.2 FTPoint::operator const FTGL_DOUBLE * ( ) const [inline]
```

Cast to FTGL_DOUBLE*.

Definition at line 240 of file FTPoint.h.

6.16.3.3 FTPoint FTPoint::operator* (double multiplier) const [inline]

Operator * Scalar multiplication.

pt3emParameters

pt3**ett3**em*multiplier* pt3em

pt3emReturns

this multiplied by multiplier.

Definition at line 159 of file FTPoint.h.

6.16.3.4 FTPoint FTPoint::operator+ (const FTPoint & point) const [inline]

Operator +.

pt3emParameters

pt3em pt3empoint pt3em

pt3emReturns

this plus point.

Definition at line 112 of file FTPoint.h.

6.16.3.5 FTPoint& FTPoint::operator+= (const FTPoint & point) [inline]

Operator += In Place Addition.

pt3emParameters

pt3em pt3empoint pt3em

pt3emReturns

this plus point.

Definition at line 97 of file FTPoint.h.

6.16.3.6 FTPoint FTPoint::operator-(const FTPoint & point) const [inline]

Operator -.

pt3emParameters

pt3em pt3empoint pt3em

```
pt3em6.16 FTPoint Class Reference
                                                    pt3em
                                                                                                     pt3em63
pt3em
pt3emReturns
    this minus point.
Definition at line 143 of file FTPoint.h.
6.16.3.7 FTPoint& FTPoint::operator-= ( const FTPoint & point ) [inline]
Operator -= In Place Substraction.
pt3emParameters
pt3em pt3empoint pt3em
pt3emReturns
    this minus point.
Definition at line 128 of file FTPoint.h.
6.16.3.8 FTPoint FTPoint::operator (const FTPoint & point) [inline]
Operator \(^\) Vector product.
pt3emParameters
pt3em pt3empoint | pt3emSecond point
pt3emReturns
    this vector point.
Definition at line 204 of file FTPoint.h.
6.16.3.9 void FTPoint::X(FTGL_DOUBLE x) [inline]
Setters.
Definition at line 249 of file FTPoint.h.
Referenced by FTBBox::operator =().
6.16.3.10 FTGL_DOUBLE FTPoint::X() const [inline]
Getters.
```

6.16.3.11 FTGL_FLOAT FTPoint::Xf() const [inline]

Referenced by FTFont::BBox().

Definition at line 257 of file FTPoint.h.

6.16.3.12 void FTPoint::Y(FTGL_DOUBLE y) [inline]

Definition at line 250 of file FTPoint.h.

Referenced by FTBBox::operator =().

6.16.3.13 FTGL_DOUBLE FTPoint::Y() const [inline]

Definition at line 258 of file FTPoint.h.

6.16.3.14 FTGL_FLOAT FTPoint::Yf()const [inline]

Definition at line 261 of file FTPoint.h.

Referenced by FTFont::BBox().

6.16.3.15 void FTPoint::Z(FTGL_DOUBLEz) [inline]

Definition at line 251 of file FTPoint.h.

Referenced by FTBBox::operator =().

6.16.3.16 FTGL_DOUBLE FTPoint::Z() const [inline]

Definition at line 259 of file FTPoint.h.

6.16.3.17 FTGL_FLOAT FTPoint::Zf() const [inline]

Definition at line 262 of file FTPoint.h.

Referenced by FTFont::BBox().

6.16.4 Friends And Related Function Documentation

6.16.4.1 bool operator!= (const FTPoint & a, const FTPoint & b) [friend]

Operator != Tests for non equality.

pt3emParameters

pt3er	m pt3em <i>a</i>	pt3em
	pt3em <i>b</i>	pt3em

pt3emReturns

true if a & b are not equal

6.16.4.2 FTPoint operator* (double multiplier, FTPoint & point) [friend]

Operator * Scalar multiplication.

pt3emParameters

pt3em pt3empoint	pt3em
pt3em <i>multiplier</i>	pt3em

pt3emReturns

multiplier multiplied by point.

Definition at line 177 of file FTPoint.h.

6.16.4.3 double operator* (FTPoint & a, FTPoint & b) [friend]

Operator * Scalar product.

pt3emParameters

pt3em	pt3em <i>a</i>	pt3emFirst vector.
	pt3em <i>b</i>	pt3emSecond vector.

pt3emReturns

a.b scalar product.

Definition at line 190 of file FTPoint.h.

6.16.4.4 bool operator== (const FTPoint & a, const FTPoint & b) [friend]

Operator == Tests for equality.

pt3emParameters

pt3em	pt3em <i>a</i>	pt3em
	pt3em <i>b</i>	pt3em

pt3emReturns

true if a & b are equal

The documentation for this class was generated from the following file:

· FTPoint.h

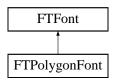
6.17 FTPolygonFont Class Reference

FTPolygonFont (p. 65) is a specialisation of the FTFont (p. 37) class for handling tesselated Polygon Mesh fonts.

#include <FTGLPolygonFont.h>

Inheritance diagram for FTPolygonFont:

pt3em



Public Member Functions

• FTPolygonFont (const char *fontFilePath)

Open and read a font file.

• FTPolygonFont (const unsigned char *pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

• \sim FTPolygonFont ()

Destructor.

Protected Member Functions

virtual FTGlyph * MakeGlyph (FT_GlyphSlot slot)
 Construct a glyph of the correct type.

6.17.1 Detailed Description

FTPolygonFont (p. 65) is a specialisation of the FTFont (p. 37) class for handling tesselated Polygon Mesh fonts.

pt3emSee also

FTFont (p. 37)

Definition at line 45 of file FTGLPolygonFont.h.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 FTPolygonFont::FTPolygonFont (const char * fontFilePath)

Open and read a font file.

Sets Error flag.

@param fontFilePath font file path.

6.17.2.2 FTPolygonFont::FTPolygonFont (const unsigned char * pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

Sets Error flag. The buffer is owned by the client and is NOT copied by **FTGL** (p. 21). The pointer must be valid while using **FTGL** (p. 21).

pt3emParameters

pt3em	pt3em	pt3emthe in-memory buffer	
рΒι	ıfferBytes		
ptdemb	ufferSize-	pt3emthe length of the buffer in byteem	pt3emGenerated on Sat Oct 13 2012 20:48:20 for FTGL by Doxygen
	InBytes	•	, ,,

6.17.2.3 FTPolygonFont::∼FTPolygonFont ()

Destructor.

6.17.3 Member Function Documentation

6.17.3.1 virtual FTGlyph* FTPolygonFont::MakeGlyph (FT_GlyphSlot slot) [protected], [virtual]

Construct a glyph of the correct type.

Clients must override the function and return their specialised **FTGlyph** (p. 47).

pt3emParameters

```
pt3em pt3emslot pt3emA FreeType glyph slot.
```

pt3emReturns

An FT****Glyph or null on failure.

Implements FTFont (p. 45).

The documentation for this class was generated from the following file:

· FTGLPolygonFont.h

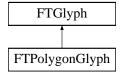
6.18 FTPolygonGlyph Class Reference

FTPolygonGlyph (p. 67) is a specialisation of FTGlyph (p. 47) for creating tessellated polygon glyphs.

#include <FTPolyGlyph.h>

Inheritance diagram for FTPolygonGlyph:

pt3em



Public Member Functions

• FTPolygonGlyph (FT_GlyphSlot glyph, float outset, bool useDisplayList)

Constructor.

virtual ~FTPolygonGlyph ()

Destructor.

• virtual const FTPoint & Render (const FTPoint &pen, int renderMode)

Render this glyph at the current pen position.

Additional Inherited Members

6.18.1 Detailed Description

FTPolygonGlyph (p. 67) is a specialisation of FTGlyph (p. 47) for creating tessellated polygon glyphs.

Definition at line 43 of file FTPolyGlyph.h.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 FTPolygonGlyph::FTPolygonGlyph (FT_GlyphSlot glyph, float outset, bool useDisplayList)

Constructor.

Sets the Error to Invalid_Outline if the glyphs isn't an outline.

pt3emParameters

-	ot3em pt3em <i>glyph</i>	pt3emThe Freetype glyph to be processed
	pt3em <i>outset</i>	pt3emThe outset distance
	pt3em	pt3emEnable or disable the use of Display Lists for this glyph true turns ON display lists.
	useDisplayList	false turns OFF display lists.

6.18.2.2 virtual FTPolygonGlyph:: \sim FTPolygonGlyph() [virtual]

Destructor.

6.18.3 Member Function Documentation

6.18.3.1 virtual const FTPoint& FTPoint& FTPoint& FTPoint & pen, int renderMode) [virtual]

Render this glyph at the current pen position.

pt3emParameters

pt3ei	n pt3em <i>pen</i>	pt3emThe current pen position.
	pt3em	pt3emRender mode to display
	renderMode	

pt3emReturns

The advance distance for this glyph.

Implements FTGlyph (p. 49).

The documentation for this class was generated from the following file:

· FTPolyGlyph.h

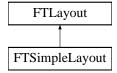
6.19 FTSimpleLayout Class Reference

FTSimpleLayout (p. 69) is a specialisation of FTLayout (p. 50) for simple text boxes.

#include <FTSimpleLayout.h>

Inheritance diagram for FTSimpleLayout:

pt3em



Public Member Functions

• FTSimpleLayout ()

Initializes line spacing to 1.0, alignment to ALIGN_LEFT and wrap to 100.0.

∼FTSimpleLayout ()

Destructor.

virtual FTBBox BBox (const char *string, const int len=-1, FTPoint position=FTPoint())

Get the bounding box for a formatted string.

virtual FTBBox BBox (const wchar_t *string, const int len=-1, FTPoint position=FTPoint())

Get the bounding box for a formatted string.

 virtual void Render (const char *string, const int len=-1, FTPoint position=FTPoint(), int renderMode=FTG-L::RENDER_ALL)

Render a string of characters.

virtual void Render (const wchar_t *string, const int len=-1, FTPoint position=FTPoint(), int renderMode=F-TGL::RENDER ALL)

Render a string of characters.

void SetFont (FTFont *fontInit)

Set the font to use for rendering the text.

- FTFont * GetFont ()
- void SetLineLength (const float LineLength)

The maximum line length for formatting text.

- · float GetLineLength () const
- · void SetAlignment (const FTGL::TextAlignment Alignment)

The text alignment mode used to distribute space within a line or rendered text.

- FTGL::TextAlignment GetAlignment () const
- · void SetLineSpacing (const float LineSpacing)

Sets the line height.

float GetLineSpacing () const

Additional Inherited Members

6.19.1 Detailed Description

FTSimpleLayout (p. 69) is a specialisation of FTLayout (p. 50) for simple text boxes.

This class has basic support for text wrapping, left, right and centered alignment, and text justification.

pt3emSee also

FTLayout (p. 50)

Definition at line 49 of file FTSimpleLayout.h.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 FTSimpleLayout::FTSimpleLayout()

Initializes line spacing to 1.0, alignment to ALIGN_LEFT and wrap to 100.0.

6.19.2.2 FTSimpleLayout:: \sim FTSimpleLayout ()

Destructor.

6.19.3 Member Function Documentation

6.19.3.1 virtual FTBBox FTSimpleLayout::BBox (const char * string, const int len = -1, FTPoint position = FTPoint ())
[virtual]

Get the bounding box for a formatted string.

pt3emParameters

pt3em pt3em <i>string</i>	pt3emA char string.
pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be checked until a null character
	is encountered (optional).
pt3emposition	pt3emThe pen position of the first character (optional).

pt3emReturns

The corresponding bounding box.

Implements FTLayout (p. 51).

6.19.3.2 virtual FTBBox FTSimpleLayout::BBox (const wchar_t * string, const int len = -1, FTPoint position = FTPoint ()) [virtual]

Get the bounding box for a formatted string.

pt3emParameters

pt3em pt3em <i>string</i>	pt3emA wchar_t string.
pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be checked until a null character
	is encountered (optional).
pt3emposition	pt3emThe pen position of the first character (optional).

pt3emReturns

The corresponding bounding box.

Implements FTLayout (p. 52).

6.19.3.3 FTGL::TextAlignment FTSimpleLayout::GetAlignment () const

pt3emReturns

The text alignment mode.

6.19.3.4 FTFont* FTSimpleLayout::GetFont ()

pt3emReturns

The current font.

6.19.3.5 float FTSimpleLayout::GetLineLength () const

pt3emReturns

The current line length.

6.19.3.6 float FTSimpleLayout::GetLineSpacing () const

pt3emReturns

The line spacing.

6.19.3.7 virtual void FTSimpleLayout::Render (const char * string, const int len = -1, FTPoint position = FTPoint (), int renderMode = FTGL::RENDER_ALL) [virtual]

Render a string of characters.

pt3emParameters

pt3em pt3em <i>string</i>	pt3em'C' style string to be output.
pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be displayed until a null character
	is encountered (optional).
pt3emposition	pt3emThe pen position of the first character (optional).
pt3em	pt3emRender mode to display (optional)
renderMode	

Implements FTLayout (p. 52).

6.19.3.8 virtual void FTSimpleLayout::Render (const wchar_t * string, const int len = -1, FTPoint position = FTPoint (), int renderMode = FTGL::RENDER_ALL) [virtual]

Render a string of characters.

pt3emParameters

pt3em pt3em <i>string</i>	pt3emwchar_t string to be output.
pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be displayed until a null character
	is encountered (optional).
pt3emposition	pt3emThe pen position of the first character (optional).
pt3em	pt3emRender mode to display (optional)
renderMode	

Implements FTLayout (p. 53).

6.19.3.9 void FTSimpleLayout::SetAlignment (const FTGL::TextAlignment Alignment)

The text alignment mode used to distribute space within a line or rendered text.

pt3emParameters

pt 3 mem Alignment	pt3emThe new alignment mode.
--------------------	------------------------------

6.19.3.10 void FTSimpleLayout::SetFont (FTFont * fontInit)

Set the font to use for rendering the text.

pt3emParameters

pt3empt3emfontInit	pt3emA pointer to the new font. The font is referenced by this but will not be disposed of when
	this is deleted.

6.19.3.11 void FTSimpleLayout::SetLineLength (const float LineLength)

The maximum line length for formatting text.

pt3emParameters

pt3em	nt3em	pt3emThe new line length.
pioo	ptociii	produttie new line length.
	LineLength	
	LineLengin	

6.19.3.12 void FTSimpleLayout::SetLineSpacing (const float LineSpacing)

Sets the line height.

pt3emParameters

pt3em	pt3em	pt3emThe height of each line of text expressed as a percentage of the current fonts line height.
Line	Spacing	

The documentation for this class was generated from the following file:

· FTSimpleLayout.h

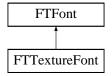
6.20 FTTextureFont Class Reference

FTTextureFont (p. 73) is a specialisation of the FTFont (p. 37) class for handling Texture mapped fonts.

#include <FTGLTextureFont.h>

Inheritance diagram for FTTextureFont:

pt3em



Public Member Functions

• FTTextureFont (const char *fontFilePath)

Open and read a font file.

• FTTextureFont (const unsigned char *pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

virtual ∼FTTextureFont ()

Destructor.

Protected Member Functions

• virtual FTGlyph * MakeGlyph (FT_GlyphSlot slot)

Construct a glyph of the correct type.

6.20.1 Detailed Description

FTTextureFont (p. 73) is a specialisation of the FTFont (p. 37) class for handling Texture mapped fonts.

pt3emSee also

FTFont (p. 37)

Definition at line 45 of file FTGLTextureFont.h.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 FTTextureFont::FTTextureFont (const char * fontFilePath)

Open and read a font file.

Sets Error flag.

@param fontFilePath font file path.

6.20.2.2 FTTextureFont::FTTextureFont (const unsigned char * pBufferBytes, size_t bufferSizeInBytes)

Open and read a font from a buffer in memory.

Sets Error flag. The buffer is owned by the client and is NOT copied by **FTGL** (p. 21). The pointer must be valid while using **FTGL** (p. 21).

pt3emParameters

pt3en	n pt3em	pt3emthe in-memory buffer
	pBufferBytes	
pt3e	em <i>bufferSize-</i>	pt3emthe length of the buffer in bytes
	InBytes	

6.20.2.3 virtual FTTextureFont::~FTTextureFont() [virtual]

Destructor.

6.20.3 Member Function Documentation

6.20.3.1 virtual FTGlyph* FTTextureFont::MakeGlyph (FT_GlyphSlot slot) [protected], [virtual]

Construct a glyph of the correct type.

Clients must override the function and return their specialised FTGlyph (p. 47).

pt3emParameters

pt3em	pt3em <i>slot</i>	pt3emA FreeType glyph slot.	
-------	-------------------	-----------------------------	--

pt3emReturns

An FT****Glyph or null on failure.

Implements FTFont (p. 45).

The documentation for this class was generated from the following file:

· FTGLTextureFont.h

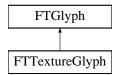
6.21 FTTextureGlyph Class Reference

FTTextureGlyph (p. 74) is a specialisation of FTGlyph (p. 47) for creating texture glyphs.

#include <FTTextureGlyph.h>

Inheritance diagram for FTTextureGlyph:

pt3em



Public Member Functions

- FTTextureGlyph (FT_GlyphSlot glyph, int id, int xOffset, int yOffset, int width, int height)
 - Constructor
- virtual ∼FTTextureGlyph ()

Destructor.

• virtual const FTPoint & Render (const FTPoint &pen, int renderMode)

Render this glyph at the current pen position.

Additional Inherited Members

6.21.1 Detailed Description

FTTextureGlyph (p. 74) is a specialisation of FTGlyph (p. 47) for creating texture glyphs.

Definition at line 43 of file FTTextureGlyph.h.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 FTTextureGlyph::FTTextureGlyph (FT_GlyphSlot glyph, int id, int xOffset, int yOffset, int width, int height)

Constructor.

pt3emParameters

pt3em pt3em <i>glyph</i>	pt3emThe Freetype glyph to be processed
pt3em <i>id</i>	pt3emThe id of the texture that this glyph will be drawn in
pt3emxOffset	pt3emThe x offset into the parent texture to draw this glyph
pt3emyOffset	pt3emThe y offset into the parent texture to draw this glyph
pt3em <i>width</i>	pt3emThe width of the parent texture
pt3em <i>height</i>	pt3emThe height (number of rows) of the parent texture

6.21.2.2 virtual FTTextureGlyph::~FTTextureGlyph() [virtual]

Destructor.

6.21.3 Member Function Documentation

6.21.3.1 virtual const FTPoint& FTTextureGlyph::Render (const FTPoint & pen, int renderMode) [virtual]

Render this glyph at the current pen position.

pt3emParameters

pt3en	n pt3em <i>pen</i>	pt3emThe current pen position.
	pt3em	pt3emRender mode to display
	renderMode	

pt3emReturns

The advance distance for this glyph.

Implements FTGlyph (p. 49).

The documentation for this class was generated from the following file:

• FTTextureGlyph.h

Chapter 7

File Documentation

7.1 faq.dox File Reference

7.2 FTBBox.h File Reference

```
#include <FTGL/ftgl.h>
```

Data Structures

· class FTBBox

FTBBox (p. 23) is a convenience class for handling bounding boxes.

7.3 FTBitmapGlyph.h File Reference

```
#include <FTGL/ftgl.h>
```

Data Structures

class FTBitmapGlyph

FTBitmapGlyph (p. 27) is a specialisation of FTGlyph (p. 47) for creating bitmaps.

Functions

• FTGLglyph * ftglCreateBitmapGlyph (FT_GlyphSlot glyph)

Create a specialisation of FTGLglyph for creating bitmaps.

7.3.1 Function Documentation

7.3.1.1 FTGLglyph* ftglCreateBitmapGlyph (FT_GlyphSlot glyph)

Create a specialisation of FTGLglyph for creating bitmaps.

pt3emParameters

pt3em pt3emglyph | pt3emThe Freetype glyph to be processed

pt3emReturns

An FTGLglyph* object.

7.4 FTBuffer.h File Reference

```
#include <FTGL/ftgl.h>
```

Data Structures

· class FTBuffer

FTBuffer (p. 29) is a helper class for pixel buffers.

7.5 FTBufferFont.h File Reference

```
#include <FTGL/ftgl.h>
```

Data Structures

· class FTBufferFont

FTBufferFont (p. 31) is a specialisation of the FTFont (p. 37) class for handling memory buffer fonts.

Functions

• FTGLfont * ftglCreateBufferFont (const char *file)

Create a specialised FTGLfont object for handling memory buffer fonts.

7.5.1 Function Documentation

7.5.1.1 FTGLfont* ftglCreateBufferFont (const char * file)

Create a specialised FTGLfont object for handling memory buffer fonts.

pt3emParameters

pt3em	pt3em <i>file</i>	10 TI (10)
DIJEM	nt'4am <i>tila</i>	pt3emThe font file name.
PLOCITI	DIOCITIE	Dideffi file folit file flattle.

pt3emReturns

An FTGLfont* object.

pt3emSee also

FTGLfont (p. 81)

7.6 FTBufferGlyph.h File Reference

#include <FTGL/ftgl.h>

Data Structures

· class FTBufferGlyph

FTBufferGlyph (p. 33) is a specialisation of FTGlyph (p. 47) for memory buffer rendering.

7.7 FTExtrdGlyph.h File Reference

#include <FTGL/ftgl.h>

Data Structures

· class FTExtrudeGlyph

FTExtrudeGlyph (p. 36) is a specialisation of FTGlyph (p. 47) for creating tessellated extruded polygon glyphs.

Macros

• #define FTExtrdGlyph FTExtrudeGlyph

Functions

• FTGLglyph * ftglCreateExtrudeGlyph (FT_GlyphSlot glyph, float depth, float frontOutset, float backOutset, int useDisplayList)

Create a specialisation of FTGLglyph for creating tessellated extruded polygon glyphs.

7.7.1 Macro Definition Documentation

7.7.1.1 #define FTExtrdGlyph FTExtrudeGlyph

Definition at line 77 of file FTExtrdGlyph.h.

7.7.2 Function Documentation

7.7.2.1 FTGLglyph* ftglCreateExtrudeGlyph (FT_GlyphSlot glyph, float depth, float frontOutset, float backOutset, int useDisplayList)

Create a specialisation of FTGLglyph for creating tessellated extruded polygon glyphs.

pt3emParameters

pt3em pt3em <i>glyph</i>	pt3emThe Freetype glyph to be processed
pt3em <i>depth</i>	pt3emThe distance along the z axis to extrude the glyph
pt3em	pt3emoutset contour size
frontOutset	
pt3em	pt3emoutset contour size
backOutset	
pt3em	pt3emEnable or disable the use of Display Lists for this glyph true turns ON display lists.
useDisplayList	false turns OFF display lists.

pt3emReturns

An FTGLglyph* object.

7.8 FTFont.h File Reference

#include <FTGL/ftgl.h>

Data Structures

· class FTFont

FTFont (p. 37) is the public interface for the FTGL (p. 21) library.

Typedefs

• typedef struct _FTGLfont FTGLfont

Functions

• FTGLfont * ftglCreateCustomFont (char const *fontFilePath, void *data, FTGLglyph *(*makeglyph-Callback)(FT_GlyphSlot, void *))

Create a custom FTGL (p. 21) font object.

void ftglDestroyFont (FTGLfont *font)

Destroy an FTGL (p. 21) font object.

int ftglAttachFile (FTGLfont *font, const char *path)

Attach auxilliary file to font e.g.

• int ftglAttachData (FTGLfont *font, const unsigned char *data, size_t size)

Attach auxilliary data to font, e.g.

• int ftglSetFontCharMap (FTGLfont *font, FT_Encoding encoding)

Set the character map for the face.

unsigned int ftglGetFontCharMapCount (FTGLfont *font)

Get the number of character maps in this face.

FT_Encoding * ftglGetFontCharMapList (FTGLfont *font)

Get a list of character maps in this face.

int ftglSetFontFaceSize (FTGLfont *font, unsigned int size, unsigned int res)

Set the char size for the current face.

unsigned int ftglGetFontFaceSize (FTGLfont *font)

Get the current face size in points (1/72 inch).

• void ftglSetFontDepth (FTGLfont *font, float depth)

Set the extrusion distance for the font.

• void ftglSetFontOutset (FTGLfont *font, float front, float back)

Set the outset distance for the font.

void ftglSetFontDisplayList (FTGLfont *font, int useList)

Enable or disable the use of Display Lists inside FTGL (p. 21).

float ftglGetFontAscender (FTGLfont *font)

Get the global ascender height for the face.

float ftglGetFontDescender (FTGLfont *font)

Gets the global descender height for the face.

float ftglGetFontLineHeight (FTGLfont *font)

Gets the line spacing for the font.

void ftglGetFontBBox (FTGLfont *font, const char *string, int len, float bounds[6])

Get the bounding box for a string.

• float ftglGetFontAdvance (FTGLfont *font, const char *string)

Get the advance width for a string.

void ftglRenderFont (FTGLfont *font, const char *string, int mode)

Render a string of characters.

• FT_Error ftglGetFontError (FTGLfont *font)

Query a font for errors.

7.8.1 Typedef Documentation

7.8.1.1 typedef struct _FTGLfont FTGLfont

Definition at line 399 of file FTFont.h.

7.8.2 Function Documentation

7.8.2.1 int ftglAttachData (FTGLfont * font, const unsigned char * data, size_t size)

Attach auxilliary data to font, e.g.

font metrics, from memory.

Note: not all font formats implement this function.

pt3emParameters

pt3em	pt3emfont	pt3emAn FTGLfont* object.
	pt3em <i>data</i>	pt3emThe in-memory buffer.
	pt3em <i>size</i>	pt3emThe length of the buffer in bytes.

pt3emReturns

1 if file has been attached successfully.

7.8.2.2 int ftglAttachFile (FTGLfont * font, const char * path)

Attach auxilliary file to font e.g.

font metrics.

Note: not all font formats implement this function.

pt3emParameters

pt3em	pt3emfont	pt3emAn FTGLfont* object.
pt3em <i>path</i>		pt3emAuxilliary font file path.

pt3emReturns

1 if file has been attached successfully.

7.8.2.3 FTGLfont* ftglCreateCustomFont (char const * fontFilePath, void * data, FTGLglyph *(*)(FT_GlyphSlot, void *) makeglyphCallback)

Create a custom FTGL (p. 21) font object.

pt3emParameters

pt3em	n pt3em	pt3emThe font file name.
	fontFilePath	
	pt3em <i>data</i>	pt3emA pointer to private data that will be passed to callbacks.
	pt3em	pt3emA glyph-making callback function.
	makeglyph-	
	Callback	

pt3emReturns

An FTGLfont* object.

7.8.2.4 void ftglDestroyFont (FTGLfont * font)

Destroy an FTGL (p. 21) font object.

pt3emParameters

pt3em	pt3em <i>font</i>	pt3emAn FTGLfont* object.

7.8.2.5 float ftglGetFontAdvance (FTGLfont * font, const char * string)

Get the advance width for a string.

pt3emParameters

pt3em	pt3emfont	pt3emAn FTGLfont* object.
pt3em <i>string</i>		pt3emA char string.

pt3emReturns

Advance width

7.8.2.6 float ftglGetFontAscender (FTGLfont * font)

Get the global ascender height for the face.

pt3emParameters

pt3em	pt3emfont	pt3emAn FTGLfont* object.

pt3emReturns

Ascender height

7.8.2.7 void ftglGetFontBBox (FTGLfont * font, const char * string, int len, float bounds[6])

Get the bounding box for a string.

pt3emParameters

pt3em	pt3emfont	pt3emAn FTGLfont∗ object.
I	pt3em <i>string</i>	pt3emA char buffer
	pt3em <i>len</i>	pt3emThe length of the string. If $<$ 0 then all characters will be checked until a null character
		is encountered (optional).
pt:	3em <i>bounds</i>	pt3emAn array of 6 float values where the bounding box's lower left near and upper right far
		3D coordinates will be stored.

7.8.2.8 unsigned int ftglGetFontCharMapCount (FTGLfont * font)

Get the number of character maps in this face.

pt3emParameters

pt3em	pt3emfont	pt3emAn FTGLfont∗ object.

pt3emReturns

character map count.

 $7.8.2.9 \quad \textbf{FT_Encoding}*\ \textbf{ftglGetFontCharMapList}\ (\ \ \textbf{FTGLfont}*\ \textbf{\textit{font}}\)$

Get a list of character maps in this face.

pt3emParameters

pt3em	pt3em <i>font</i>	pt3emAn FTGLfont∗ object.

pt3emReturns

pointer to the first encoding.

7.8.2.10 float ftglGetFontDescender (FTGLfont * font)

Gets the global descender height for the face.

pt3emParameters

pt3em pt3emfont pt3emAn FTGLfont* object.

pt3emReturns

Descender height

7.8.2.11 FT_Error ftglGetFontError (FTGLfont * font)

Query a font for errors.

pt3emParameters

pt3em pt3emfont pt3emAn FTGLfont* object.

pt3emReturns

The current error code.

7.8.2.12 unsigned int ftglGetFontFaceSize (FTGLfont * font)

Get the current face size in points (1/72 inch).

pt3emParameters

pt3em pt3emfont pt3emAn FTGLfont* object.

pt3emReturns

face size

7.8.2.13 float ftglGetFontLineHeight (FTGLfont * font)

Gets the line spacing for the font.

pt3emParameters

pt3em pt3emfont pt3emAn FTGLfont* object.

pt3emReturns

Line height

7.8.2.14 void ftglRenderFont (FTGLfont * font, const char * string, int mode)

Render a string of characters.

pt3emParameters

pt3en	n pt3emfont	pt3emAn FTGLfont∗ object.
	pt3em <i>string</i>	pt3emChar string to be output.
	pt3em <i>mode</i>	pt3emRender mode to display.

7.8.2.15 int ftglSetFontCharMap (FTGLfont * font, FT_Encoding encoding)

Set the character map for the face.

pt3emParameters

pt3em	pt3emfont	pt3emAn FTGLfont* object.
pt3em <i>encoding</i>		pt3emFreetype enumerate for char map code.

pt3emReturns

1 if charmap was valid and set correctly.

7.8.2.16 void ftglSetFontDepth (FTGLfont * font, float depth)

Set the extrusion distance for the font.

Only implemented by **FTExtrudeFont** (p. 34).

pt3emParameters

pt3em	pt3emfont	pt3emAn FTGLfont* object.
pt3em <i>depth</i>		pt3emThe extrusion distance.

7.8.2.17 void ftglSetFontDisplayList (FTGLfont * font, int useList)

Enable or disable the use of Display Lists inside **FTGL** (p. 21).

pt3emParameters

pt3em pt3emfa	nt pt3emAn FTGLfont* object.
pt3em <i>useL</i>	st pt3em1 turns ON display lists. 0 turns OFF display lists.

7.8.2.18 int ftglSetFontFaceSize (FTGLfont * font, unsigned int size, unsigned int res)

Set the char size for the current face.

pt3emParameters

pt3em	pt3emfont	pt3emAn FTGLfont* object.
	pt3em <i>size</i>	pt3emThe face size in points (1/72 inch).
	pt3em <i>res</i>	pt3emThe resolution of the target device, or 0 to use the default value of 72.

pt3emReturns

1 if size was set correctly.

7.8.2.19 void ftglSetFontOutset (FTGLfont * font, float front, float back)

Set the outset distance for the font.

Only **FTOutlineFont** (p. 53), **FTPolygonFont** (p. 65) and **FTExtrudeFont** (p. 34) implement front outset. Only **FT-ExtrudeFont** (p. 34) implements back outset.

pt3emParameters

pt3em	pt3emfont	pt3emAn FTGLfont* object.
	pt3em <i>front</i>	pt3emThe front outset distance.
	pt3em <i>back</i>	pt3emThe back outset distance.

7.9 ftgl.dox File Reference

7.10 ftgl.h File Reference

#include <ft2build.h>

```
#include <FT_FREETYPE_H>
#include <FT_GLYPH_H>
#include <FT_OUTLINE_H>
#include <FTGL/FTPoint.h>
#include <FTGL/FTBBox.h>
#include <FTGL/FTBuffer.h>
#include <FTGL/FTGlyph.h>
#include <FTGL/FTBitmapGlyph.h>
#include <FTGL/FTBufferGlyph.h>
#include <FTGL/FTExtrdGlyph.h>
#include <FTGL/FTOutlineGlyph.h>
#include <FTGL/FTPixmapGlyph.h>
#include <FTGL/FTPolyGlyph.h>
#include <FTGL/FTTextureGlyph.h>
#include <FTGL/FTFont.h>
#include <FTGL/FTGLBitmapFont.h>
#include <FTGL/FTBufferFont.h>
#include <FTGL/FTGLExtrdFont.h>
#include <FTGL/FTGLOutlineFont.h>
#include <FTGL/FTGLPixmapFont.h>
#include <FTGL/FTGLPolygonFont.h>
#include <FTGL/FTGLTextureFont.h>
#include <FTGL/FTLayout.h>
#include <FTGL/FTSimpleLayout.h>
```

Namespaces

· namespace FTGL

Macros

- #define FTGL_BEGIN_C_DECLS extern "C" { namespace FTGL {
- #define FTGL_END_C_DECLS } }
- #define FTGL_EXPORT

Typedefs

- typedef double FTGL DOUBLE
- typedef float FTGL FLOAT

Enumerations

- enum FTGL::RenderMode { FTGL::RENDER_FRONT = 0x0001, FTGL::RENDER_BACK = 0x0002, FTG-L::RENDER_SIDE = 0x0004, FTGL::RENDER_ALL = 0xffff }
- enum FTGL::TextAlignment { FTGL::ALIGN_LEFT = 0, FTGL::ALIGN_CENTER = 1, FTGL::ALIGN_RIG-HT = 2, FTGL::ALIGN_JUSTIFY = 3 }

7.10.1 Macro Definition Documentation

7.10.1.1 #define FTGL_BEGIN_C_DECLS extern "C" { namespace FTGL {

Definition at line 43 of file ftgl.h.

7.10.1.2 #define FTGL_END_C_DECLS } }

Definition at line 44 of file ftgl.h.

7.10.1.3 #define FTGL_EXPORT

Definition at line 107 of file ftgl.h.

7.10.2 Typedef Documentation

7.10.2.1 typedef double FTGL_DOUBLE

Definition at line 38 of file ftgl.h.

7.10.2.2 typedef float FTGL_FLOAT

Definition at line 39 of file ftgl.h.

7.11 FTGLBitmapFont.h File Reference

#include <FTGL/ftgl.h>

Data Structures

· class FTBitmapFont

FTBitmapFont (p. 26) is a specialisation of the FTFont (p. 37) class for handling Bitmap fonts.

Macros

• #define FTGLBitmapFont FTBitmapFont

Functions

• FTGLfont * ftglCreateBitmapFont (const char *file)

Create a specialised FTGLfont object for handling bitmap fonts.

7.11.1 Macro Definition Documentation

7.11.1.1 #define FTGLBitmapFont FTBitmapFont

Definition at line 84 of file FTGLBitmapFont.h.

7.11.2 Function Documentation

7.11.2.1 FTGLfont* ftglCreateBitmapFont (const char * file)

Create a specialised FTGLfont object for handling bitmap fonts.

pt3emParameters

pt3em pt3emfile pt3emThe font file name.

pt3emReturns

An FTGLfont* object.

pt3emSee also

FTGLfont (p. 81)

7.12 FTGLExtrdFont.h File Reference

#include <FTGL/ftgl.h>

Data Structures

· class FTExtrudeFont

FTExtrudeFont (p. 34) is a specialisation of the FTFont (p. 37) class for handling extruded Polygon fonts.

Macros

• #define FTGLExtrdFont FTExtrudeFont

Functions

• FTGLfont * ftglCreateExtrudeFont (const char *file)

Create a specialised FTGLfont object for handling extruded poygon fonts.

7.12.1 Macro Definition Documentation

7.12.1.1 #define FTGLExtrdFont FTExtrudeFont

Definition at line 85 of file FTGLExtrdFont.h.

7.12.2 Function Documentation

7.12.2.1 FTGLfont* ftglCreateExtrudeFont (const char * file)

Create a specialised FTGLfont object for handling extruded poygon fonts.

pt3emParameters

pt3em pt3emfile pt3emThe font file name.

pt3emReturns

An FTGLfont* object.

pt3emSee also

FTGLfont (p. 81) ftglCreatePolygonFont (p. 92)

7.13 FTGLOutlineFont.h File Reference

#include <FTGL/ftgl.h>

Data Structures

· class FTOutlineFont

FTOutlineFont (p. 53) is a specialisation of the FTFont (p. 37) class for handling Vector Outline fonts.

Macros

• #define FTGLOutlineFont FTOutlineFont

Functions

• FTGLfont * ftglCreateOutlineFont (const char *file)

Create a specialised FTGLfont object for handling vector outline fonts.

7.13.1 Macro Definition Documentation

7.13.1.1 #define FTGLOutlineFont FTOutlineFont

Definition at line 84 of file FTGLOutlineFont.h.

7.13.2 Function Documentation

7.13.2.1 FTGLfont* ftglCreateOutlineFont (const char * file)

Create a specialised FTGLfont object for handling vector outline fonts.

pt3emParameters

pt3em pt3emfile pt3emThe font file name.

pt3emReturns

An FTGLfont* object.

pt3emSee also

FTGLfont (p. 81)

7.14 FTGLPixmapFont.h File Reference

```
#include <FTGL/ftgl.h>
```

Data Structures

· class FTPixmapFont

FTPixmapFont (p. 56) is a specialisation of the FTFont (p. 37) class for handling Pixmap (Grey Scale) fonts.

Macros

• #define FTGLPixmapFont FTPixmapFont

Functions

• FTGLfont * ftglCreatePixmapFont (const char *file)

Create a specialised FTGLfont object for handling pixmap (grey scale) fonts.

7.14.1 Macro Definition Documentation

7.14.1.1 #define FTGLPixmapFont FTPixmapFont

Definition at line 84 of file FTGLPixmapFont.h.

7.14.2 Function Documentation

7.14.2.1 FTGLfont* ftglCreatePixmapFont (const char * file)

Create a specialised FTGLfont object for handling pixmap (grey scale) fonts.

pt3emParameters

pt3em pt3emfile pt3emThe font file name.

pt3emReturns

An FTGLfont* object.

pt3emSee also

FTGLfont (p. 81)

7.15 FTGLPolygonFont.h File Reference

```
#include <FTGL/ftgl.h>
```

Data Structures

· class FTPolygonFont

FTPolygonFont (p. 65) is a specialisation of the FTFont (p. 37) class for handling tesselated Polygon Mesh fonts.

Macros

• #define FTGLPolygonFont FTPolygonFont

Functions

• FTGLfont * ftglCreatePolygonFont (const char *file)

Create a specialised FTGLfont object for handling tesselated polygon mesh fonts.

7.15.1 Macro Definition Documentation

7.15.1.1 #define FTGLPolygonFont FTPolygonFont

Definition at line 84 of file FTGLPolygonFont.h.

7.15.2 Function Documentation

7.15.2.1 FTGLfont* ftglCreatePolygonFont (const char * file)

Create a specialised FTGLfont object for handling tesselated polygon mesh fonts.

pt3emParameters

pt3em pt3emfile pt3emThe font file name.

pt3emReturns

An FTGLfont* object.

pt3emSee also

FTGLfont (p. 81)

7.16 FTGLTextureFont.h File Reference

#include <FTGL/ftgl.h>

Data Structures

• class FTTextureFont

FTTextureFont (p. 73) is a specialisation of the FTFont (p. 37) class for handling Texture mapped fonts.

Macros

#define FTGLTextureFont

Functions

FTGLfont * ftglCreateTextureFont (const char *file)

Create a specialised FTGLfont object for handling texture-mapped fonts.

7.16.1 Macro Definition Documentation

7.16.1.1 #define FTGLTextureFont FTTextureFont

Definition at line 84 of file FTGLTextureFont.h.

7.16.2 Function Documentation

7.16.2.1 FTGLfont* ftglCreateTextureFont (const char * file)

Create a specialised FTGLfont object for handling texture-mapped fonts.

pt3emParameters

pt3em pt3emfile pt3emThe font file name.

pt3emReturns

An FTGLfont* object.

pt3emSee also

FTGLfont (p. 81)

7.17 FTGlyph.h File Reference

#include <FTGL/ftgl.h>

Data Structures

· class FTGlyph

FTGlyph (p. 47) is the base class for FTGL (p. 21) glyphs.

Typedefs

typedef struct _FTGLglyph

Functions

FTGLglyph * ftglCreateCustomGlyph (FTGLglyph *base, void *data, void(*renderCallback)(FTGLglyph *, void *, FTGL_DOUBLE, FTGL_DOUBLE, int, FTGL_DOUBLE *, FTGL_DOUBLE *), void(*destroy-Callback)(FTGLglyph *, void *))

Create a custom FTGL (p. 21) glyph object.

void ftglDestroyGlyph (FTGLglyph *glyph)

Destroy an FTGL (p. 21) glyph object.

 void ftglRenderGlyph (FTGLglyph *glyph, FTGL_DOUBLE penx, FTGL_DOUBLE peny, int renderMode, FTGL_DOUBLE *advancex, FTGL_DOUBLE *advancey)

Render a glyph at the current pen position and compute the corresponding advance.

float ftglGetGlyphAdvance (FTGLglyph *glyph)

Return the advance for a glyph.

void ftglGetGlyphBBox (FTGLglyph *glyph, float bounds[6])

Return the bounding box for a glyph.

• FT_Error ftglGetGlyphError (FTGLglyph *glyph)

Query a glyph for errors.

7.17.1 Typedef Documentation

7.17.1.1 typedef struct _FTGLglyph FTGLglyph

Definition at line 133 of file FTGlyph.h.

7.17.2 Function Documentation

7.17.2.1 FTGLglyph* ftglCreateCustomGlyph (FTGLglyph* base, void * data, void(*)(FTGLglyph*, void *, FTGL_DOUBLE, FTGL_DOUBLE, int, FTGL_DOUBLE *, FTGL_DOUBLE *) renderCallback, void(*)(FTGLglyph *, void *) destroyCallback)

Create a custom FTGL (p. 21) glyph object.

FIXME: maybe get rid of "base" and have advanceCallback etc. functions

pt3emParameters

pt3em pt3embase	pt3emThe base FTGLglyph∗ to subclass.
pt3em <i>data</i>	pt3emA pointer to private data that will be passed to callbacks.
pt3em	pt3emA rendering callback function.
renderCallback	
pt3em	pt3emA callback function to be called upon destruction.
destroyCallback	

pt3emReturns

An FTGLglyph* object.

7.17.2.2 void ftglDestroyGlyph (FTGLglyph * glyph)

Destroy an FTGL (p. 21) glyph object.

pt3emParameters

pt3em pt3emglyph | pt3emAn FTGLglyph* object.

7.17.2.3 float ftglGetGlyphAdvance (FTGLglyph * glyph)

Return the advance for a glyph.

pt3emParameters

pt3em pt3emglyph | pt3emAn FTGLglyph* object.

pt3emReturns

The advance's X component.

7.17.2.4 void ftglGetGlyphBBox (FTGLglyph * glyph, float bounds[6])

Return the bounding box for a glyph.

pt3emParameters

pt3em pt3em <i>glyph</i>	pt3emAn FTGLglyph* object.
pt3em <i>bounds</i>	pt3emAn array of 6 float values where the bounding box's lower left near and upper right far
	3D coordinates will be stored.

7.17.2.5 FT_Error ftglGetGlyphError (FTGLglyph * glyph)

Query a glyph for errors.

pt3emParameters

pt3em pt3emglyph | pt3emAn FTGLglyph* object.

pt3emReturns

The current error code.

7.17.2.6 void ftglRenderGlyph (FTGLglyph * glyph, FTGL_DOUBLE penx, FTGL_DOUBLE peny, int renderMode, FTGL_DOUBLE * advancex, FTGL_DOUBLE * advancey)

Render a glyph at the current pen position and compute the corresponding advance.

pt3emParameters

pt3em pt3em <i>glyph</i>	pt3emAn FTGLglyph* object.
pt3em <i>penx</i>	pt3emThe current pen's X position.
pt3em <i>peny</i>	pt3emThe current pen's Y position.
pt3em	pt3emRender mode to display
renderMode	
pt3em <i>advancex</i>	pt3emA pointer to an FTGL_DOUBLE where to write the advance's X component.
pt3em <i>advancey</i>	pt3emA pointer to an FTGL_DOUBLE where to write the advance's Y component.

7.18 FTLayout.h File Reference

#include <FTGL/ftgl.h>

Data Structures

· class FTLayout

FTLayout (p. 50) is the interface for layout managers that render text.

Typedefs

• typedef struct _FTGLlayout FTGLlayout

Functions

• void ftglDestroyLayout (FTGLlayout *layout)

Destroy an FTGL (p. 21) layout object.

• void ftglGetLayoutBBox (FTGLlayout *layout, const char *string, float bounds[6])

Get the bounding box for a string.

• void ftglRenderLayout (FTGLlayout *layout, const char *string, int mode)

Render a string of characters.

• FT_Error ftglGetLayoutError (FTGLlayout *layout)

Query a layout for errors.

7.18.1 Typedef Documentation

7.18.1.1 typedef struct _FTGLlayout FTGLlayout

Definition at line 151 of file FTLayout.h.

7.18.2 Function Documentation

7.18.2.1 void ftglDestroyLayout (FTGLlayout * layout)

Destroy an FTGL (p. 21) layout object.

pt3emParameters

pt3empt3em/ayout	pt3emAn FTGLlayout* object.

7.18.2.2 void ftglGetLayoutBBox (FTGLlayout * layout, const char * string, float bounds[6])

Get the bounding box for a string.

pt3emParameters

pt3empt3em <i>layout</i>	pt3emAn FTGLlayout* object.
pt3em <i>string</i>	pt3emA char buffer
pt3em <i>bounds</i>	pt3emAn array of 6 float values where the bounding box's lower left near and upper right far
	3D coordinates will be stored.

7.18.2.3 FT_Error ftglGetLayoutError (FTGLlayout * layout)

Query a layout for errors.

pt3emParameters

-10 10 1	io A FTOLL : I' :
pt3empt3em <i>lavout</i>	pt3emAn FTGLlayout* object.
procriptocimayour	ploenian i delayout object.

pt3emReturns

The current error code.

7.18.2.4 void ftglRenderLayout (FTGLlayout * layout, const char * string, int mode)

Render a string of characters.

pt3emParameters

pt3empt3em <i>layout</i>	pt3emAn FTGLlayout* object.
pt3em <i>string</i>	pt3emChar string to be output.
pt3em <i>mode</i>	pt3emRender mode to display.

7.19 FTOutlineGlyph.h File Reference

#include <FTGL/ftgl.h>

Data Structures

· class FTOutlineGlyph

FTOutlineGlyph (p. 55) is a specialisation of FTGlyph (p. 47) for creating outlines.

Functions

• FTGLglyph * ftglCreateOutlineGlyph (FT_GlyphSlot glyph, float outset, int useDisplayList)

Create a specialisation of FTGLglyph for creating outlines.

7.19.1 Function Documentation

7.19.1.1 FTGLglyph* ftglCreateOutlineGlyph (FT_GlyphSlot glyph, float outset, int useDisplayList)

Create a specialisation of FTGLglyph for creating outlines.

pt3emParameters

р	t3em pt3em <i>glyph</i>	pt3emThe Freetype glyph to be processed
	pt3em <i>outset</i>	pt3emoutset contour size
	pt3em	pt3emEnable or disable the use of Display Lists for this glyph true turns ON display lists.
	useDisplayList	false turns OFF display lists.

pt3emReturns

An FTGLglyph* object.

7.20 FTPixmapGlyph.h File Reference

#include <FTGL/ftgl.h>

Data Structures

class FTPixmapGlyph

FTPixmapGlyph (p. 58) is a specialisation of FTGlyph (p. 47) for creating pixmaps.

Functions

• FTGLglyph * ftglCreatePixmapGlyph (FT_GlyphSlot glyph)

Create a specialisation of FTGLglyph for creating pixmaps.

7.20.1 Function Documentation

7.20.1.1 FTGLglyph* ftglCreatePixmapGlyph (FT_GlyphSlot glyph)

Create a specialisation of FTGLglyph for creating pixmaps.

pt3emParameters

pt3em pt3emglyph | pt3emThe Freetype glyph to be processed

pt3emReturns

An FTGLglyph* object.

7.21 FTPoint.h File Reference

```
#include <FTGL/ftgl.h>
```

Data Structures

· class FTPoint

FTPoint (p. 59) class is a basic 3-dimensional point or vector.

7.22 FTPolyGlyph.h File Reference

```
#include <FTGL/ftgl.h>
```

Data Structures

class FTPolygonGlyph

FTPolygonGlyph (p. 67) is a specialisation of FTGlyph (p. 47) for creating tessellated polygon glyphs.

Macros

#define FTPolyGlyph FTPolygonGlyph

Functions

• FTGLglyph * ftglCreatePolygonGlyph (FT_GlyphSlot glyph, float outset, int useDisplayList)

Create a specialisation of FTGLglyph for creating tessellated polygon glyphs.

7.22.1 Macro Definition Documentation

7.22.1.1 #define FTPolyGlyph FTPolygonGlyph

Definition at line 74 of file FTPolyGlyph.h.

7.22.2 Function Documentation

7.22.2.1 FTGLglyph* ftglCreatePolygonGlyph (FT_GlyphSlot glyph, float outset, int useDisplayList)

Create a specialisation of FTGLglyph for creating tessellated polygon glyphs.

pt3emParameters

pt3em pt3em <i>glyph</i>	pt3emThe Freetype glyph to be processed
pt3em <i>outset</i>	pt3emoutset contour size
pt3em	pt3emEnable or disable the use of Display Lists for this glyph true turns ON display lists.
useDisplayList	false turns OFF display lists.

pt3emReturns

An FTGLglyph* object.

7.23 FTSimpleLayout.h File Reference

```
#include <FTGL/ftgl.h>
```

Data Structures

· class FTSimpleLayout

FTSimpleLayout (p. 69) is a specialisation of FTLayout (p. 50) for simple text boxes.

Functions

- FTGLlayout * ftglCreateSimpleLayout (void)
- void ftglSetLayoutFont (FTGLlayout *, FTGLfont *)
- FTGLfont * ftglGetLayoutFont (FTGLlayout *)
- void ftglSetLayoutLineLength (FTGLlayout *, const float)
- float ftglGetLayoutLineLength (FTGLlayout *)
- void ftglSetLayoutAlignment (FTGLlayout *, const int)
- int ftglGetLayoutAlignement (FTGLlayout *)
- void ftglSetLayoutLineSpacing (FTGLlayout *, const float)
- float ftglGetLayoutLineSpacing (FTGLlayout *)

7.23.1 Function Documentation

```
    7.23.1.1 FTGLlayout* ftglCreateSimpleLayout ( void )
    7.23.1.2 int ftglGetLayoutAlignement ( FTGLlayout * )
    7.23.1.3 FTGLfont* ftglGetLayoutFont ( FTGLlayout * )
    7.23.1.4 float ftglGetLayoutLineLength ( FTGLlayout * )
```

7.23.1.5 float ftglGetLayoutLineSpacing (FTGLlayout *)

```
7.23.1.6 void ftglSetLayoutAlignment ( FTGLlayout * , const int )
```

7.23.1.7 void ftglSetLayoutFont (FTGLlayout * , FTGLfont *)

7.23.1.8 void ftglSetLayoutLineLength (FTGLlayout * , const float)

7.23.1.9 void ftglSetLayoutLineSpacing (FTGLlayout * , const float)

7.24 FTTextureGlyph.h File Reference

```
#include <FTGL/ftgl.h>
```

Data Structures

· class FTTextureGlyph

FTTextureGlyph (p. 74) is a specialisation of FTGlyph (p. 47) for creating texture glyphs.

Functions

• FTGLglyph * ftglCreateTextureGlyph (FT_GlyphSlot glyph, int id, int xOffset, int yOffset, int width, int height)

Create a specialisation of FTGLglyph for creating pixmaps.

7.24.1 Function Documentation

7.24.1.1 FTGLglyph* ftglCreateTextureGlyph (FT_GlyphSlot glyph, int id, int xOffset, int yOffset, int width, int height)

Create a specialisation of FTGLglyph for creating pixmaps.

pt3emParameters

pt3em pt3em <i>glyph</i>	pt3emThe Freetype glyph to be processed.
pt3em <i>id</i>	pt3emThe id of the texture that this glyph will be drawn in.
pt3emxOffset	pt3emThe x offset into the parent texture to draw this glyph.
pt3emyOffset pt3emThe y offset into the parent texture to draw this glyph.	
pt3em <i>width</i>	pt3emThe width of the parent texture.
pt3em <i>height</i>	pt3emThe height (number of rows) of the parent texture.

pt3emReturns

An FTGLglyph* object.

7.25 projects_using_ftgl.txt File Reference

7.26 tutorial.dox File Reference

pt3emIndex

Ascender

FTFont, 40

pt3em FTFont, 41, 42 FTBBX FTGlyph, 49 FTBBX FTBitmapFont, 27 FTBitmapGlyph FTBitmapGlyph FTBitmapGlyph, 28 FTBuffer, 29 FTBuffer, 29 FTBufferFont FTBuffer, 29 FTBufferGlyph FTBufferGlyph FTBufferGlyph FTBufferGlyph FTBufferGlyph FTExtrudeFont, 35 FTExtrudeGlyph FTExtrudeGlyph, 37 FTFont, 40 FTGlyph, 48 FTGlyph, 48 FTLayout, 51 FTOutlineFont, 54 FTOutlineFont, 54 FTOutlineGlyph, 56 FTDutlineGlyph, 56 FTPixmapGlyph FTDixmapFont FTPixmapGlyph FTPixmapFont FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTFoutlineGlyph, 68 FTFpolygonGlyph FTFolygonGlyph FTExtureGoth, 74 FTTextureGlyph FTExtureGlyph FTFoxtureFont FTPolygonGlyph FTFixmapGlyph FTFoxtureFont FTFoxtureFont, 74 FTTextureGlyph FTTextureGlyph FTTextureGlyph FTTextureGlyph FTTextureGlyph FTFoxtureFont, 74 FTTextureGlyph FTGL, 21 FTBuffer, 29		
FTBBox, 24 FTBBox, 24 FTBitmapFont, 27 FTBitmapFont, 27 FTBitmapGlyph FTBitmapGlyph, 28 FTBuffer, 29 FTBuffer, 29 FTBuffer, 32 FTBufferGlyph FTBufferGlyph FTBufferGlyph FTBufferGlyph FTBufferGlyph FTBufferGlyph, 34 FTExtrudeFont, 35 FTExtrudeFont, 35 FTExtrudeGlyph FTExtrudeGlyph, 37 FTFont, 40 FTGlyph, 48 FTLayout, 51 FTClyph, 48 FTLayout, 51 FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTPixmapFont, 57 FTPixmapFont, 57 FTPixmapFont FTPixmapGlyph, 59 FTPixmapGlyph FTPolygonGlyph FTPixmapGlyph FTPixmapGlyph FTFBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapGlyph, 28 FTBitmapGlyph, 29 FTBitmapGlyph, 29 FTBitmapGlyph, 29 FTBitmapGlyph, 29 FTBitmapGlyph, 29 FTBitmapGlyph, 39 FTBitmapGlyph FT	pt3em	FTFont, 41, 42
FTBBox, 24 FTBitmapFont FTBitmapFont, 27 FTBitmapGlyph, 28 FTBuffer FTBuffer, 29 FTBuffer, 29 FTBuffer CharMapLount FTBuffer, 29 FTBufferFont FTBufferGlyph FTBufferGlyph, 34 FTExtrudeFont, 35 FTExtrudeFont, 35 FTExtrudeGlyph FTExtrudeGlyph, 37 FTExtrudeGlyph FTExtrudeGlyph, 37 FTFont, 40 FTGlyph, 48 FTGuyth, 49 FTGutlineFont, 54 FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTPixmapFont, 57 FTPixmapFont, 57 FTPixmapFont, 57 FTPixmapGlyph, 59 FTPolygonFont FTPolygonGlyph FTPixmapGlyph, 68 FTSimpleLayout, 70 FTTExtureGlyph FTExtureGlyph FTExtureGlyph FTExtureGlyph FTExtureGlyph FTExtureFont FTFont, 40 FTBbox, 24 FTBbox, 25 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29		
~FTBitmapFont, 27 FTBitmapGlyph FTBitmapGlyph, 28 FTBuffer FTBuffer FTBuffer, 29 FTBuffer, 29 FTBufferFont FTBufferFont FTBufferGlyph FTBufferGlyph FTBufferGlyph, 34 FTExtrudeFont, 35 FTExtrudeGlyph FTExtrudeGlyph FTExtrudeGlyph FTExtrudeGlyph FTExtrudeGlyph, 37 FTFont, 40 FTGlyph FTGlyph, 48 FTLayout FTLayout, 51 FTOutlineFont, 54 FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTDutlineGlyph FTDutlineGlyph FTPixmapGnt, 57 FTPixmapFont, 57 FTPixmapFont, 57 FTPixmapGlyph, 59 FTPolygonFont FTPolygonGlyph, 68 FTSimpleLayout FTIButureFont, 74 FTTExtureGlyph, 75 FTBExtrudeGlyph, 75 FTBExtrudeGlyph, 75 ALIGN_LEFT FTGL, 21 ALIGN_LEFT FTGL, 21 Advance pt3enfTFont, 40 FTBLyout FTBLyout, 50 FTBLyout FTBLyout, 51 FTBLyout, 51 FTBLyout, 52 FTBLyout FTBLyout, 51 FTBLyout, 52 FTBLyout, 52 FTBLyout, 53 FTBLyout, 54 FTBLyout, 54 FTBLyout, 55 FTBLyout, 56 FTBLyout, 57 FTBLyout, 57 FTBLyout, 58 FTBLyout, 59 FTBLyout, 50 FTBLyout, 70 FTBLyout, 50 FTBLyout, 40 FTSLyout, 40 FTSLyout, 40 FTBLyout, 40 FTSLyout, 40 F	-	
FTBitmapGlyph FTBitmapGlyph, 28 FTBuffer FTBuffer, 29 FTBuffer, 29 FTBufferFont, 32 FTBufferFont, 32 FTBufferGlyph FTBufferGlyph FTBufferGlyph, 34 FTExtrudeFont FTExtrudeFont, 35 FTExtrudeGlyph FTExtrudeGlyph FTExtrudeGlyph FTEyht	,	
~FTBitmapGlyph FTBitmapGlyph, 28 FTBuffer FTBuffer, 29 FTBuffer, 29 FTBufferFont FTBufferFont, 32 FTBufferGlyph FTBufferGlyph FTBufferGlyph FTBufferGlyph, 34 FTExtrudeFont, 35 FTExtrudeGlyph, 37 FTExtrudeGlyph, 37 FTFont, 40 FTGlyph FTGlyph, 48 FTLayout, 51 FTOutlineFont, 54 FTOutlineFont, 54 FTOutlineGlyph FTOutlineGlyph FTDufferGlyph, 36 FTPixmapFont, 57 FTPixmapGlyph FTOutlineGlyph FTOutlineGlyph FTBufferGlyph, 36 FTFibymapFont, 57 FTPixmapGlyph FTExtrudeGlyph FTExtrudeGlyph FTUutlineGlyph FTUutlineGlyph FTUutlineGlyph FTUutlineGlyph FTUutlineGlyph FTExtrughGlyph FTExtrughGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPolygonGlyph FTPolygonGlyph FTFIphylapout, 70 FTTExtureGlyph FTTextureGlyph FTTExtureGlyph FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3effFont, 40 Pt3em FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29	•	i i ompiozajout, i o
FTBuffer CharMapCount FTBuffer, 29 FTBuffer CoharMapCount FTBufferFont FTBufferFont FTBufferFont, 32 FTBufferGlyph, 34 FTExtrudeFont, 35 FTExtrudeFont, 35 FTExtrudeFont, 35 FTExtrudeGlyph FTBufferGlyph, 37 FTFont, 40 FTFont, 40 FTGlyph FTGlyph, 48 FTLayout, 51 FTOutlineFont FTUutlineFont FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTDixmapGlyph, 56 FTPixmapGlyph FTPixmapFont FTPixmapGlyph FTPixmapFont FTPixmapGlyph FTPixmapFont FTPixmapGlyph FTFixmapGlyph FTFixmapGlyph FTFixmapGlyph FTFixmapGlyph FTFixmapGlyph FTFixmapGlyph FTFixmapGlyph FTFixmapGlyph, 28 FTBitmapGlyph, 29 FTBuffer, 29 Atlandam Advance p33effFont, 40 pt3em	•	CharMan
~FTBuffer CharMapCount FTFont, 43 ~FTBuffer, 29 ~FTBufferFont, 32 ~FTBufferGlyph FTBufferGlyph, 34 ~FTExtrudeFont, 35 ~FTExtrudeGlyph FTExtrudeGlyph, 37 ~FTExtrudeGlyph, 37 ~FTFont, 40 ~FTGlyph FTGlyph, 48 ~FTLayout, 51 ~FTOutlineFont, 54 ~FTOutlineGlyph FTOutlineGlyph FTDutfineGlyph, 56 ~FTPixmapFont FTPixmapFont FTPixmapFont FTPixmapGlyph FTRygonGlyph FTRygonGlyph FTRygonGlyph FTRygonGlyph FTRygonGlyph FTRygonGlyph FTRytureGlyph, 58 ~FTExtureGlyph, 68 ~FTSimpleLayout FTSimpleLayout FTSimpleLayout FTSimpleLayout FTSimpleLayout FTSimpleLayout FTSimpleLayout FTSimpleLayout FTSimpleLayout FTRytureGlyph FTRytureGlyph, 75 ALIGN_CENTER FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pl3effFont, 40 pt3em		•
FTBuffer, 29 FTBufferFont FTBufferFont, 32 FTBufferGlyph FTBufferGlyph, 34 FTExtrudeFont FTExtrudeFont, 35 FTExtrudeGlyph FTExtrudeGlyph FTExtrudeGlyph, 37 FTFont, 40 FTGlyph FTGlyph, 48 FTLayout, 51 FTOutlineFont FTOutlineFont FTOutlineFont FTOutlineGlyph, 56 FTPixmapFont, 57 FTPixmapGlyph FTPixmapGlyph, 59 FTPolygonFont FTPolygonGlyph FTPolygonGlyph FTRimpleLayout, 70 FTTExtureGlyph FTExtrudeGlyph, 75 ALIGN_CENTER FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erffer, 29 FTBuffer, 29		
~FTBufferFont FTBufferFont, 32 ~FTBufferGlyph, 34 ~FTExtrudeFont, 35 ~FTExtrudeFont, 35 ~FTExtrudeGlyph FTExtrudeGlyph FTExtrudeGlyph, 37 ~FTFont FTFont, 40 ~FTGlyph FTGlyph, 48 ~FTLayout FTLayout, 51 ~FTOutlineFont, 54 ~FTOutlineGlyph FTOutlineGlyph FTDutlineGlyph FTPixmapFont, 57 ~FTPixmapFont FTPixmapGlyph, 59 ~FTPolygonFont, 67 ~FTPolygonGlyph FTPolygonGlyph, 68 ~FTSimpleLayout, 70 ~FTTextureGlyph, 75 ALIGN_CENTER FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALVAN AVAINA Depth FTFont, 43 PTFont, 43 PETFOnt, 44 FTGlyph, 49 FTGlyph, 49 FTBBox, 24 FTBBox, 24 FTBBox, 24 FTBBox, 24 FTBBox, 24 Invalidate, 24 IsValid, 24 Lower, 25 Operator+=, 25 SetDepth, 25 Upper, 25 SetDepth, 25 Upper, 25 FTBBront, 26 ~FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapGlyph, 27 ~FTBitmapGlyph, 28 FTBitmapGlyph, 29 FTBuffer, 29 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTBL, 21 ALIGN_RIGHT FTBLIFET FTBLIFET		•
FTBufferFont, 32 FTBufferGlyph FTBufferGlyph, 34 FTExtrudeFont FTExtrudeFont, 35 FTExtrudeGlyph FTExtrudeGlyph FTExtrudeGlyph FTExtrudeGlyph, 37 FTFont, 40 FTGlyph, 48 FTLayout FTLayout, 51 FTOutlineFont FTOutlineFont FTOutlineGlyph, 56 FTPixmapGlyph, 59 FTPixmapGlyph, 59 FTPixmapGlyph, 59 FTPixmapGlyph, 59 FTPolygonFont FTPolygonFont FTPolygonFont FTFolygonFont FTFolygonFont FTFolygonFont FTFolygonFont FTFIxmapGlyph FTFIxmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPolygonFont FTFOlygonFont FTFOlygonFont FTFolygonFont FTFOlygonFont FTFIextureFont FTFIextureFont FTFixtureGlyph FTFTextureGlyph, 75 ALIGN_CENTER FTGL, 21 ALIGN_JUSTIFY FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3emferver Pt3emfer, 29 FTBuffer, 29 FTBuffer, 29 Advance pt3emfer Depth FTFont, 43 Depth FTFont, 43 PETFont, 43 Depth FTFont, 43 PETFont, 43 Descender FTFont, 43 PETFont, 43 PETFont, 43 PETFont, 43 PETFont, 43 Descender FTFont, 43 PETFont, 43 PETGN, 49 FTBBox, 27 FTBBox, 23 FTBBox, 24 FTBBox, 25 SetDepth, 25 FTBIttmapFont, 26 FTBittmapFont, 26 FTBittmapGlyph, 27 FTBittmapGlyph, 28 FTBittmapGlyph, 28 FTBittmapGlyph, 28 FTBittmapGlyph, 28 FTBittmapGlyph, 28 FTBittmapGlyph, 28 FTBittmapGlyph, 27 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29	•	
~FTBufferGlyph, 34 ~FTExtrudeFont FTExtrudeFont, 35 ~FTExtrudeGlyph FTExtrudeGlyph FTGlyph FTGlyph FTGlyph, 48 ~FTLayout, 51 ~FTOutlineFont FTOutlineFont, 54 ~FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTPixmapFont FTPixmapFont FTPixmapFont, 57 ~FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph FTExtureFont FTExtureFont FTExtureGlyph FTTextureGlyph FTTextureGlyph FTTextureGlyph FTTextureGlyph FTTextureGlyph FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pl3erfirFont, 40 PtSeender FTFont, 43 Descender FTFont, 44 FTBont, 49 FTBBox, 23 ~FTBBox, 24 FTBBox, 24 FTBBox, 24 Invalidate, 24 IsValid, 24 Lower, 25 Operator+=, 25 SetDepth, 25 SetDepth, 25 SetDepth, 25 SetDepth, 25 SetDepth, 26 FTBBbox, 27 FTBItmapFont, 26 FTBItmapFont, 26 FTBItmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 77 FTBitmapGlyph,		•
FTBufferGlyph, 34 FTExtrudeFont FTExtrudeFont, 35 FFExtrudeGlyph FTExtrudeGlyph FTExtrudeGlyph FTExtrudeGlyph, 37 FTFont, 40 FTGlyph FTGlyph, 48 FTLayout, 51 FTOutlineFont FTOutlineFont FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTPixmapFont FTPixmapFont FTPixmapFont, 57 FTFibxmapGlyph FTPixmapGlyph FTPolygonGlyph FTFibylogonGlyph FTFetyperGlyph FTFixtureFont FTExtureFont FTTextureFont FTTextureGlyph FTTextureGlyph FTTextureGlyph FTGL, 21 ALIGN_USTIFY FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pl3erfiTFont, 40 PSECRATE FTENTALY FTBurfer, 29 FTBuffer, 29	FTBufferFont, 32	FIFont, 43
~FTExtrudeFont, 35 ~FTExtrudeGlyph FTExtrudeGlyph, 37 ~FTEnt, 40 ~FTGlyph FTGlyph, 48 ~FTLayout, 51 ~FTOutlineFont FTOutlineGlyph FTOutlineGlyph FTPixmapFont, 56 ~FTPixmapFont, 57 ~FTPixmapGlyph FTPixmapGlyph FTPolygonFont FTPolygonFont FTPolygonGlyph FTPolygonGlyph FTSimpleLayout, 70 ~FTTExtureGlyph FTTExtureGlyph FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_LEFT FTGL, 21 Advance pi3erfTFont, 40 Descender FTFont, 43 Descender FTFont, 44 FTGL, 44 FTGL, 21 FTBDN, 23 ~FTBInt, 44 FTGL, 21 FTBBox, 23 ~FTBBox, 24 FTBBox, 24 FTBBox, 24 FTBBox, 24 Invalidate, 24 Invalidate, 24 Invalidate, 24 Invalidate, 24 Invalidate, 24 FTBBox, 25 FTBBox, 27 FTBItmapFont, 26 FTFont, 46 MakeGlyph, 27 FTBitmapFont, 26 FTBitmapGlyph, 28 FTBitmapGlyph, 29 FTBuffer, 29	\sim FTBufferGlyph	5
FTExtrudeFont, 35 FFExtrudeGlyph FTExtrudeGlyph, 37 FFFont, 40 FFFont, 40 FFGlyph FTGlyph, 48 FFLayout, 51 FTOutlineFont FTOutlineFont, 54 FFTOutlineGlyph FTOutlineGlyph FTPixmapFont, 57 FFTPixmapGlyph, 59 FFTPixmapGlyph, 59 FFTPolygonFont FTPolygonFont FTPolygonGlyph FTSimpleLayout FTSimpleLayout, 70 FFTExtureFont, 74 FFTExtureGlyph FTExtureGlyph FTGxtureGlyph FTGxtureFont, 27 FFTExtureGlyph FTGxtureGlyph FTGxtureGlyph FTGxtureFont, 27 FFTGL, 21 ALIGN_JUSTIFY FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pi3erfTFont, 40 PSECENDER FTFON, 43 FTFont, 43 FTFont, 44 FTFont, 44 FTFont, 44 FTFont, 44 FTEND, 49 FTBbox, 23 FTBBox, 23 FTBBox, 24 FTBBox, 24 FTBBox, 24 FTBBox, 24 Invalidate, 24 IsValid, 24 Lower, 25 operator+=, 25 SetDepth, 25 Upper, 25 FTBBox.h, 77 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 29 FTBitm	FTBufferGlyph, 34	•
FTExtrudeGlyph FTExtrudeGlyph, 37 FFFont, 40 FFTGolyph, 48 FFTLayout, 51 FTLayout, 51 FTOutlineFont FTOutlineFont, 54 FFTOutlineGlyph FTDutlineGlyph FTDutlineGlyph FTDutlineGlyph FTOutlineGlyph FTPixmapFont, 57 FFTPixmapFont FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPolygonFont FTPolygonFont FTPolygonGlyph FTPolygonGlyph FTSimpleLayout FTSimpleLayout FTSimpleLayout FTSimpleLayout, 70 FFTExtureFont FTTextureFont, 74 FFTExtureGlyph FTTextureGlyph FTGxt, 21 ALIGN_USTIFY FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erfitFont, 40 PTSimpleFont, 40 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29	\sim FTExtrudeFont	FTFont, 43
~FTExtrudeGlyph FTExtrudeGlyph, 37 ~FTFont FTFont, 40 ~FTGlyph, 48 ~FTLayout, 51 ~FTLayout, 51 ~FTOutlineFont, 54 ~FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTPixmapFont FTPixmapFont FTPixmapGlyph, 59 ~FTPolygonFont FTPolygonGlyph FTPolygonGlyph FTSimpleLayout FTSimpleLayout FTSimpleLayout, 70 ~FTTextureGlyph FTTextureGlyph FTTextureGlyph FTGL, 21 ALIGN_USTIFY FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erfirFont, 40 FTSort, 46 FTSort, 46 FTBout, 43 Error FTFont, 44 FTEND, 44 FTGlyph, 49 FTClyph, 49 FTClyph, 49 FTClyph, 49 FTBout, 27 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapGlyph, 28 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 77 FTBitmap	FTExtrudeFont, 35	Descender
FTExtrudeGlyph, 37 FFFont FFFont, 40 FFTGott, 40 FFTGlyph FFTGlyph, 48 FFLayout FFLayout, 51 FFLoutlineFont FFOutlineFont, 54 FFOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTPixmapFont FTPixmapFont, 57 FFTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPolygonFont FTPolygonFont FTPolygonGlyph, 68 FFTSimpleLayout FTSimpleLayout FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 29 FTBitmapGlyph, 49 Render, 28 FTBitmapGlyph, 77 FTBuffer, 29		FTFont, 43
FTFont FTFont, 40 FTGlyph FTGlyph, 48 FTLayout, 51 FTDutlineFont FTBbox, 23 FTDutlineFont, 54 FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTPixmapFont, 57 FTPixmapFont, 57 FTPixmapGlyph FTPixmapGlyph, 59 FTPolygonFont FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph FTTextureFont, 74 FTTextureFont, 74 FTTextureFont, 74 FTTextureGlyph FTTextureGlyph, 75 ALIGN_CENTER FTGL, 21 ALIGN_UEFT FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erfitFont, 40 FTSimpleLayou pt3em		
FTFont, 40 FTGlyph FTGlyph, 48 FTLayout, 51 FTLayout, 51 FTDutlineFont FTOutlineFont, 54 FTDutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTOutlineGlyph FTPixmapFont FTPixmapFont FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPolygonFont FTPolygonFont FTPolygonGlyph FTTPolygonGlyph FTSimpleLayout FTSimpleLayout FTSimpleLayout FTExtureFont FTTextureGlyph FTTextureGlyph FTTextureGlyph FTGL, 21 ALIGN_USTIFY FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erffTFont, 40 FTBlyput, 52 FTBuffer, 29	••	Error
~FTGlyph, 48 FTGlyph, 48 ~FTLayout FTLayout, 51 ~FTOutlineFont FTOutlineFont, 54 ~FTOutlineGlyph FTOutlineGlyph, 56 ~FTPixmapFont, 57 ~FTPixmapFont, 57 ~FTPixmapGlyph FTOutlygonFont FTPolygonFont FTPolygonGlyph FTOutlyph FTOutlineGlyph FTPixmapGlyph, 59 ~FTPolygonFont FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph FTSimpleLayout FTSimpleLayout FTSimpleLayout, 70 ~FTTextureFont FTTextureGlyph FTTextureGlyph FTTextureGlyph FTTextureGlyph FTGL, 21 ALIGN_USTIFY FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erfitFont, 40 FTBitmapGlyph, 28 FTBuffer, 29 FTBuffer, 29 Advance pt3erfitFont, 40 FTBitmapGlyph, 77 FTBuffer, 29		FTFont, 44
FTGlyph, 48 FFLayout FFLayout, 51 FFOutlineFont FTOutlineFont, 54 FFDutlineGlyph FTOutlineGlyph, 56 FFIPixmapFont FTPixmapFont, 57 FFIPixmapGlyph, 59 FFPolygonFont FTPolygonFont FTPolygonGlyph FTOutlineGlyph, 68 FFISimpleLayout, 70 FFTSimpleLayout, 70 FFTExtureFont FTTextureGlyph FTTextureGlyph, 75 ALIGN_CENTER FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erffTFont, 40 FTButhapCont, 52 FFTBux, 24 FTBbox, 25 SetDepth, 25 Upper, 25 FTBbox, 27 FTBbox, 24 FTBbox, 26 FTBbo	•	FTGlyph, 49
FTLayout FTLayout, 51 FTOutlineFont FTOutlineFont, 54 FTOutlineGlyph FTOutlineGlyph, 56 FTPixmapFont FTPixmapFont, 57 FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPolygonFont FTPolygonFont FTPolygonGlyph FTPolygonGlyph FTSimpleLayout FTSimpleLayout FTSimpleLayout, 70 FTTextureFont FTTextureGlyph FTTextureGlyph FTTextureGlyph FTTextureGlyph FTGL, 21 ALIGN_CENTER FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erffTFont, 40 FTBoutlineEdx, 24 Invalidate, 24 Inval	,,	FTLayout, 52
FTLayout, 51 FTOutlineFont FTOutlineFont, 54 FTOutlineGlyph FTOutlineGlyph, 56 FTPixmapFont FTPixmapFont, 57 FTPixmapGlyph FTPixmapGlyph, 59 FTPolygonFont FTPolygonGlyph FTPolygonGlyph FTSimpleLayout FTSimpleLayout FTSimpleLayout, 70 FTTextureFont, 74 FTTextureGlyph FTTextureGlyph FTTextureGlyph FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3eriffTent, 40 FTBoutlineGlyph, 54 FTBBox, 24 FTBBox, 24 Invalidate, 24 IsValid, 24 Lower, 25 operator+=, 25 SetDepth, 25 Upper, 25 SetDepth, 25 FTBBox.h, 77 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTFont, 46 MakeGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 77 ftglCreateBitmapGlyph, 77 FTBuffer, 29 FTBuffer, 29 Advance pt3erifTFont, 40	**	•
~FTOutlineFont FTOutlineFont, 54 ~FTOutlineGlyph FTOutlineGlyph, 56 ~FTPixmapFont FTPixmapFont FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPolygonFont FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph FTSimpleLayout FTSimpleLayout FTSimpleLayout FTTextureFont FTTextureFont FTTextureGlyph FTGL, 21 ALIGN_USTIFY FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3eriffFont, 40 ~FTBuffar, 29 FTBuffar, 29	•	FTBBox, 23
FTOutlineFont, 54 ~FTOutlineGlyph FTOutlineGlyph, 56 ~FTPixmapFont FTPixmapFont, 57 ~FTPixmapGlyph FTPixmapGlyph, 59 ~FTPolygonFont FTPolygonFont, 67 ~FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph FTFextureFont FTTextureFont FTTextureFont FTTextureGlyph FTTextureGlyph FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3eriffFont, 40 FTButmapFont, 54 FTBBox, 24 Invalidate, 26 Invalidate, 26 Invalidate, 26 Invalidate, 26 In		\sim FTBBox, 24
FTOutlineFont, 54 FTOutlineGlyph FTOutlineGlyph, 56 FFIPixmapFont FTPixmapFont, 57 FTPixmapGlyph FTPixmapGlyph FTPixmapGlyph FTPolygonFont FTPolygonFont FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph FTSimpleLayout FTSimpleLayout FTSimpleLayout, 70 FTTextureFont FTTextureGlyph FTTextureGlyph FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3eriffFont, 40 FTBudiate, 24 Invalidate, 24 Invalidat		,
FTOutlineGlyph, 56 FTPixmapFont FTPixmapFont, 57 FTPixmapGlyph FTPixmapGlyph, 59 FTPolygonFont FTPolygonFont FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph FTSimpleLayout FTSimpleLayout FTSimpleLayout, 70 FTTextureFont FTTextureGlyph FTTextureGlyph FTGL, 21 ALIGN_LEFT FTGL, 21 AVance pt3erifTFont, 40 Invalidate, 24 IsValid, 24 Lower, 25 operator+=, 25 SetDepth, 25 Upper, 25 FTBBox.h, 77 FTBstmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 77 ftglCreateBitmapGlyph, 77 FTBuffer, 29 Advance pt3erifTFont, 40		· ·
FTOutlineGlyph, 56 FTPixmapFont FTPixmapFont, 57 FTPixmapGlyph FTPixmapGlyph, 59 FTPolygonFont FTPolygonFont, 67 FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph, 68 FTSimpleLayout FTSimpleLayout FTSimpleLayout FTSimpleLayout FTSimpleLayout FTSimpleLayout FTFextureFont FTExtureFont FTTextureGlyph FTTextureGlyph FTTextureGlyph FTGL, 21 ALIGN_CENTER FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 AVance pt3erffTFont, 40 IsValid, 24 Lower, 25 operator+=, 25 SetDepth, 25 Upper, 25 FTBBox.h, 77 FTBBox.h, 77 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTFont, 46 MakeGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTGlyph, 49 Render, 28 FTBitmapGlyph, 77 FTGL, 21 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 Advance pt3erffTFont, 40	\sim FTOutlineGlyph	· ·
FTPixmapFont, 57 FTPixmapGlyph FTPixmapGlyph, 59 FTPolygonFont FTPolygonFont FTPolygonGlyph FTPolygonGlyph FTPolygonGlyph, 68 FTSimpleLayout FTSimpleLayout, 70 FTTextureFont FTTextureFont, 74 FTTextureGlyph FTTextureGlyph, 75 ALIGN_CENTER FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 AVance pt3erifTFont, 40 Lower, 25 operator+=, 25 operator+=, 25 SetDepth, 25 Upper, 25 FTBbmx.h, 77 FTBbmx.h, 77 FTBbmx.h, 77 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTFont, 46 MakeGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTGL, 21 FTBitmapGlyph, 49 Render, 28 FTBitmapGlyph, 77 FTGL, 21 FTBuffer, 29 Advance pt3erifTFont, 40	FTOutlineGlyph, 56	
FTPixmapFont, 57 FTPixmapGlyph FTPixmapGlyph, 59 FTPolygonFont FTPolygonFont, 67 FTPolygonGlyph FTPolygonGlyph, 68 FTSimpleLayout FTSimpleLayout FTSimpleLayout, 70 FTTextureFont FTTextureGlyph FTTextureGlyph FTTextureGlyph FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3eriffTFont, 40 poperator+=, 25 SetDepth, 25 SetDepth, 25 Upper, 25 FTBBmx.h, 77 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapGlyph, 26 FTFont, 46 MakeGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 77 FTGL, 21 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 Advance pt3erifTFont, 40	\sim FTPixmapFont	
FTPixmapGlyph, 59 FTPolygonFont FTPolygonFont, 67 FTPolygonGlyph FTPolygonGlyph, 68 FTSimpleLayout FTSimpleLayout, 70 FTTextureFont FTTextureFont FTTextureGlyph FTTextureGlyph FTTextureGlyph FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 AUdvance pt3eriftTFont, 40 SetDepth, 25 Upper, 25 FTBbmx.h, 77 FTBbmx.h, 77 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapGlyph, 26 FTFont, 46 MakeGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTGl, 21 FTGL, 21 FTBitmapGlyph, 77 FTGL, 21 FTBuffer, 29	FTPixmapFont, 57	•
FTPixmapGlyph, 59 ~FTPolygonFont FTPolygonFont, 67 ~FTPolygonGlyph FTPolygonGlyph, 68 ~FTSimpleLayout FTSimpleLayout, 70 ~FTTextureFont FTTextureGlyph FTTextureGlyph, 75 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3eriftTent, 40 SetDeptn, 25 Upper, 25 FTBbox.h, 77 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 49 Render, 28 FTBitmapGlyph, 77 FTBitmapGlyph, 77 FTBitmapGlyph, 77 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 Advance pt3em	~FTPixmapGlyph	•
~FTPolygonFont FTPolygonGlyph FTPolygonGlyph, 68 ~FTSimpleLayout FTSimpleLayout, 70 ~FTTextureFont FTTextureGlyph FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 AVAnce pt3erfiTFont, 40 Cpper, 25 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTFont, 46 MakeGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTGlyph, 49 Render, 28 FTGlyph, 49 FTBitmapGlyph, 77 FTGL, 21 FTBUffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 Advance pt3erfiTFont, 40		•
FTPolygonFont, 67 ~FTPolygonGlyph FTPolygonGlyph, 68 ~FTSimpleLayout FTSimpleLayout, 70 ~FTTextureFont FTTextureFont, 74 ~FTTextureGlyph FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erfiTFont, 40 FTBitmapFont, 26 FT		• •
~FTPolygonGlyph FTPolygonGlyph, 68 ~FTSimpleLayout FTSimpleLayout, 70 ~FTTextureFont FTTextureFont, 74 ~FTTextureGlyph FTTextureGlyph, 75 ALIGN_CENTER FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erfiTFont, 40 FTBitmapFont, 26 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTGlyph, 49 Render, 28 FTBitmapGlyph, 77 ftglCreateBitmapGlyph, 77 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29	• •	FTBBox.h, 77
FTPolygonGlyph, 68 FTSimpleLayout FTSimpleLayout, 70 FTTextureFont FTTextureFont, 74 FTTextureGlyph FTTextureGlyph, 75 ALIGN_USTIFY FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 AVance pt3erfiTFont, 40 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapFont, 26 FTBitmapGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTGlyph, 49 Render, 28 FTBitmapGlyph.h, 77 ftglCreateBitmapGlyph, 77 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 Advance pt3erfiTFont, 40 Pt3em	· ·	FTBitmapFont, 26
~FTSimpleLayout, 70 FTSimpleLayout, 70 FTTextureFont FTTextureGlyph FTTextureGlyph, 75 ALIGN_JUSTIFY FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erfiTFont, 40 FTBitmapFont, 26 FTBitmapFont, 26 FTFont, 46 MakeGlyph, 27 FTBitmapGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTGlyph, 49 Render, 28 FTBitmapGlyph.h, 77 ftglCreateBitmapGlyph, 77 FTBL, 21 FTBuffer, 29		\sim FTBitmapFont, 27
FTSimpleLayout, 70 ~FTTextureFont FTTextureFont, 74 ~FTTextureGlyph FTTextureGlyph, 75 ALIGN_CENTER FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 AVance pt3erfiTFont, 40 FTBitmapFont, 26 FTFont, 46 MakeGlyph, 27 FTBitmapGlyph, 27 ~FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTGlyph, 49 Render, 28 FTGlyph, 49 Render, 28 FTBitmapGlyph.h, 77 ftglCreateBitmapGlyph, 77 FTGL, 21 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29		FTBitmapFont, 26
~FTTextureFont FTTextureGlyph FTTextureGlyph FTTextureGlyph, 75 ALIGN_CENTER FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erfiTFont, 40 MakeGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 49 Render, 28 FTBitmapGlyph.h, 77 ftglCreateBitmapGlyph.h, 77 FTBuffer, 29 ~FTBuffer, 29 Advance pt3erfiTFont, 40 Pt3em		FTBitmapFont, 26
FT TextureFont, 74 FTTextureGlyph FTTextureGlyph, 75 ALIGN_CENTER FTGL, 21 ALIGN_JUSTIFY FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erfiTFont, 40 MakeGlyph, 27 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 49 Render, 28 FTBitmapGlyph.h, 77 ftglCreateBitmapGlyph, 77 FTBL, 21 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29		FTFont, 46
FTTextureGlyph FTTextureGlyph, 75 FTExtureGlyph, 75 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTGL, 21 FTGL, 21 FTGL, 21 FTGL, 21 FTBitmapGlyph, 49 Render, 28 FTGL, 21 FTBitmapGlyph, 49 Render, 28 FTGL, 21 FTBitmapGlyph, 77 FTGL, 21 FTBitmapGlyph, 77 FTGL, 21 FTBitmapGlyph, 77 FTGL, 21 FTBitmapGlyph, 28 FTGlyph, 49 Render, 28 FTBitmapGlyph, 77 FTBitmapGlyph, 77 FTBitmapGlyph, 77 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTGlyph, 49 Render, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTGlyph, 49 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTGlyph, 49 FTBitmapGlyph, 28 FTB		
FTTextureGlyph, 75 FTTextureGlyph, 75 ALIGN_CENTER FTGL, 21 ALIGN_JUSTIFY FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 AVance pt3erfiTFont, 40 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTBitmapGlyph, 28 FTGlyph, 49 Render, 28 FTBitmapGlyph.h, 77 ftglCreateBitmapGlyph.h, 77 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 PTBuffer, 29		
FT lextureGlyph, 75 FTBitmapGlyph, 28 ALIGN_CENTER FTGL, 21 ALIGN_JUSTIFY FTGL, 21 ALIGN_LEFT FTGL, 21 FTGL, 21 FTBitmapGlyph, 28 FTGlyph, 49 Render, 28 FTBitmapGlyph.h, 77 FTGL, 21 FTBitmapGlyph.h, 77 ftglCreateBitmapGlyph, 77 FTGL, 21 FTBuffer, 29 ALIGN_RIGHT FTGL, 21 FTBuffer, 29 Advance pt3erfiTFont, 40 pt3em		
$\begin{array}{llllllllllllllllllllllllllllllllllll$	FTTextureGlyph, 75	
FTGL, 21 FTGlyph, 49 ALIGN_JUSTIFY Render, 28 FTGL, 21 FTBitmapGlyph.h, 77 ALIGN_LEFT ftglCreateBitmapGlyph, 77 FTGL, 21 FTBuffer, 29 ALIGN_RIGHT ~FTBuffer, 29 FTGL, 21 FTBuffer, 29 Advance pt3erfiTFont, 40	ALICN CENTED	
ALIGN_JUSTIFY Render, 28 FTGL, 21 FTBitmapGlyph.h, 77 ALIGN_LEFT ftglCreateBitmapGlyph, 77 FTGL, 21 FTBuffer, 29 ALIGN_RIGHT ~FTBuffer, 29 FTGL, 21 FTBuffer, 29 Advance pt3erfiTFont, 40 pt3em	-	
FTGL, 21 ALIGN_LEFT FTGL, 21 ALIGN_RIGHT FTGL, 21 Advance pt3erfiTFont, 40 FTBitmapGlyph.h, 77 ftglCreateBitmapGlyph, 77 FTBuffer, 29 ~FTBuffer, 29 FTBuffer, 29 FTBuffer, 29 FTBuffer, 29		- ·
ALIGN_LEFT ftglCreateBitmapGlyph, 77 FTGL, 21 FTBuffer, 29 ALIGN_RIGHT ~FTBuffer, 29 FTGL, 21 FTBuffer, 29 Advance pt3erfiTFont, 40 pt3em	_	•
FTGL, 21 FTBuffer, 29 ALIGN_RIGHT ~FTBuffer, 29 FTGL, 21 FTBuffer, 29 Advance pt3erfiTFont, 40 pt3em	,	
ALIGN_RIGHT \sim FTBuffer, 29 FTBuffer, 29 Advance pt3erfiTFont, 40 pt3em	-	
FTGL, 21 FTBuffer, 29 Advance pt3erfiTFont, 40 pt3em		FTBuffer, 29
Advance pt3erfiTFont, 40 pt3em	-	\sim FTBuffer, 29
pt3erfiTFont, 40 pt3em	FTGL, 21	FTBuffer, 29
	Advance	
	pt3erfiTFont, 40	pt3em
	FTGlyph, 49	•

pt3em

pt3em	FTPixmapFont, 47
FTBuffer, 29	FTPolygonFont, 47
	FTTextureFont, 47
Height, 30	
Pixels, 30	FaceSize, 44
Pos, 30	FTFont, 39
Size, 30	GlyphLoadFlags, 44
Width, 31	LineHeight, 44
FTBuffer.h, 78	MakeGlyph, 45
FTBufferFont, 31	Outset, 45
\sim FTBufferFont, 32	Render, 45, 46
FTBufferFont, 32	UseDisplayList, 46
FTBufferFont, 32	FTFont.h, 80
FTFont, 46	FTGLfont, 81
MakeGlyph, 32	ftglAttachData, 81
FTBufferFont.h, 78	ftglAttachFile, 81
ftglCreateBufferFont, 78	ftglCreateCustomFont, 82
FTBufferGlyph, 33	ftglDestroyFont, 82
∼FTBufferGlyph, 34	ftglGetFontAdvance, 82
	ftglGetFontAscender, 83
FTBufferGlyph, 33	_
FTBufferGlyph, 33	ftglGetFontBBox, 83
FTGlyph, 49	ftglGetFontCharMapCount, 83
Render, 34	ftglGetFontCharMapList, 83
FTBufferGlyph.h, 79	ftglGetFontDescender, 84
FTExtrdGlyph	ftglGetFontError, 84
FTExtrdGlyph.h, 79	ftglGetFontFaceSize, 84
FTExtrdGlyph.h, 79	ftglGetFontLineHeight, 84
FTExtrdGlyph, 79	ftglRenderFont, 85
ftglCreateExtrudeGlyph, 79	ftglSetFontCharMap, 85
FTExtrudeFont, 34	ftglSetFontDepth, 85
\sim FTExtrudeFont, 35	ftglSetFontDisplayList, 85
FTExtrudeFont, 35	ftglSetFontFaceSize, 85
FTExtrudeFont, 35	ftglSetFontOutset, 86
FTFont, 47	FTFontImpl
MakeGlyph, 35	FTFont, 47
FTExtrudeGlyph, 36	FTGL, 21
~FTExtrudeGlyph, 37	ALIGN_CENTER, 21
FTExtrudeGlyph, 36	ALIGN_JUSTIFY, 21
FTExtrudeGlyph, 36	ALIGN_LEFT, 21
FTGlyph, 50	ALIGN_RIGHT, 21
Render, 37	RENDER_ALL, 21
FTFont, 37	RENDER_BACK, 21
\sim FTFont, 40	RENDER_FRONT, 21
Advance, 40	RENDER_SIDE, 21
Ascender, 40	RenderMode, 21
Attach, 41	TextAlignment, 21
BBox, 41, 42	FTGL_BEGIN_C_DECLS
CharMap, 43	ftgl.h, 87
CharMapCount, 43	FTGL DOUBLE
CharMapList, 43	ftgl.h, 88
Depth, 43	FTGL_END_C_DECLS
Descender, 43	ftgl.h, 87
Error, 44	FTGL EXPORT
	-
FTB://fasFant. 46	ftgl.h, 88
FTBufferFont, 46	FTGL_FLOAT
FTExtrudeFont, 47	ftgl.h, 88
FTFont, 39	FTGLBitmapFont
FTFontImpl, 47	FTGLBitmapFont.h, 88
FTOutlineFont, 47	FTGLBitmapFont.h, 88

pt3em	∼FTLayout, 51
FTGLBitmapFont, 88	BBox, 51, 52
ftglCreateBitmapFont, 89	Error, 52
FTGLExtrdFont	FTLayout, 51
FTGLExtrdFont.h, 89	FTSimpleLayout, 53
FTGLExtrdFont.h, 89	FTLayout, 51
	-
FTGLExtrdFont, 89	Render, 52, 53
ftglCreateExtrudeFont, 89	FTLayout.h, 96
FTGLOutlineFont	FTGLlayout, 96
FTGLOutlineFont.h, 90	ftglDestroyLayout, 97
FTGLOutlineFont.h, 90	ftglGetLayoutBBox, 97
FTGLOutlineFont, 90	ftglGetLayoutError, 97
ftglCreateOutlineFont, 90	ftglRenderLayout, 97
FTGLPixmapFont	FTOutlineFont, 53
FTGLPixmapFont.h, 91	~FTOutlineFont, 54
FTGLPixmapFont.h, 91	FTOutlineFont, 54
FTGLPixmapFont, 91	FTFont, 47
ftglCreatePixmapFont, 91	FTOutlineFont, 54
FTGLPolygonFont	MakeGlyph, 54
FTGLPolygonFont.h, 92	FTOutlineGlyph, 55
FTGLPolygonFont.h, 92	\sim FTOutlineGlyph, 56
FTGLPolygonFont, 92	FTOutlineGlyph, 55
ftglCreatePolygonFont, 92	FTGlyph, 50
FTGLTextureFont	FTOutlineGlyph, 55
FTGLTextureFont.h, 93	Render, 56
FTGLTextureFont.h, 93	FTOutlineGlyph.h, 98
FTGLTextureFont, 93	ftglCreateOutlineGlyph, 98
ftglCreateTextureFont, 93	FTPixmapFont, 56
FTGLfont	\sim FTPixmapFont, 57
FTFont.h, 81	FTPixmapFont, 57
FTGLglyph	FTFont, 47
FTGlyph.h, 94	FTPixmapFont, 57
FTGLlayout	MakeGlyph, 57
FTLayout.h, 96	FTPixmapGlyph, 58
FTGlyph, 47	\sim FTPixmapGlyph, 59
\sim FTGlyph, 48	FTPixmapGlyph, 59
Advance, 49	FTGlyph, 50
BBox, 49	FTPixmapGlyph, 59
Error, 49	Render, 59
FTBitmapGlyph, 49	FTPixmapGlyph.h, 98
FTBufferGlyph, 49	ftglCreatePixmapGlyph, 99
FTExtrudeGlyph, 50	FTPoint, 59
FTGlyph, 48	FTPoint, 61
FTOutlineGlyph, 50	FTPoint, 61
FTPixmapGlyph, 50	Normalise, 61
FTPolygonGlyph, 50	operator const FTGL_DOUBLE $*$, 61
FTTextureGlyph, 50	operator*, 61, 64, 65
FTGlyph, 48	operator $^{\wedge}$, 63
Render, 49	operator+, 62
FTGlyph.h, 93	operator+=, 62
FTGLglyph, 94	operator-, 62
ftglCreateCustomGlyph, 94	operator-=, 63
ftglDestroyGlyph, 95	operator==, 65
ftglGetGlyphAdvance, 95	X, 63
ftglGetGlyphBBox, 95	Xf, 63
ftglGetGlyphError, 95	Y, 63, 64
ftglRenderGlyph, 95	Yf, 64
FTLayout, 50	Z, 64

pt3em	ftglCreateTextureGlyph, 101
Zf, 64	FaceSize
FTPoint.h, 99	FTFont, 44
FTPolyGlyph	faq.dox, 77
FTPolyGlyph.h, 99	ftgl.dox, 86
FTPolyGlyph.h, 99	ftgl.h, 86
FTPolyGlyph, 99	FTGL_BEGIN_C_DECLS, 87
ftglCreatePolygonGlyph, 100	FTGL DOUBLE, 88
FTPolygonFont, 65	FTGL_END_C_DECLS, 87
~FTPolygonFont, 67	FTGL EXPORT, 88
FTPolygonFont, 66	FTGL_FLOAT, 88
FTFont, 47	ftglAttachData
FTPolygonFont, 66	FTFont.h, 81
MakeGlyph, 67	ftglAttachFile
FTPolygonGlyph, 67	FTFont.h, 81
~FTPolygonGlyph, 68	ftglCreateBitmapFont
FTPolygonGlyph, 68	FTGLBitmapFont.h, 89
FTGlyph, 50	ftglCreateBitmapGlyph
FTPolygonGlyph, 68	FTBitmapGlyph.h, 77
Render, 68	ftglCreateBufferFont
FTSimpleLayout, 69	FTBufferFont.h, 78
~FTSimpleLayout, 70	ftglCreateCustomFont
BBox, 70	FTFont.h, 82
FTSimpleLayout, 70	ftglCreateCustomGlyph
FTLayout, 53	FTGlyph.h, 94
FTSimpleLayout, 70	ftglCreateExtrudeFont
GetAlignment, 71	FTGLExtrdFont.h, 89
GetFont, 71	ftglCreateExtrudeGlyph
GetLineLength, 71	FTExtrdGlyph.h, 79
GetLineSpacing, 71	ftglCreateOutlineFont
Render, 71	FTGLOutlineFont.h, 90
SetAlignment, 72	ftglCreateOutlineGlyph
SetFont, 72	FTOutlineGlyph.h, 98
SetLineLength, 72	ftglCreatePixmapFont
SetLineSpacing, 72	FTGLPixmapFont.h, 91
FTSimpleLayout.h, 100	ftglCreatePixmapGlyph
ftglCreateSimpleLayout, 100	FTPixmapGlyph.h, 99
ftglGetLayoutAlignement, 100	ftglCreatePolygonFont
ftglGetLayoutFont, 100	FTGLPolygonFont.h, 92
ftglGetLayoutLineLength, 100	ftglCreatePolygonGlyph
ftglGetLayoutLineSpacing, 100	FTPolyGlyph.h, 100
ftglSetLayoutAlignment, 101	ftglCreateSimpleLayout
ftglSetLayoutFont, 101	FTSimpleLayout.h, 100
ftglSetLayoutLineLength, 101	ftglCreateTextureFont
ftglSetLayoutLineSpacing, 101	FTGLTextureFont.h, 93
FTTextureFont, 73	ftglCreateTextureGlyph
~FTTextureFont, 74	FTTextureGlyph.h, 101
FTTextureFont, 73	ftglDestroyFont
FTFont, 47	FTFont.h, 82
FTTextureFont, 73	ftglDestroyGlyph
MakeGlyph, 74 FTTextureGlyph, 74	FTGlyph.h, 95
~FTTextureGlyph, 75	ftglDestroyLayout FTLayout.h, 97
	-
FTTextureGlyph, 75 FTGlyph, 50	ftglGetFontAdvance FTFont.h, 82
FTTextureGlyph, 75	ftglGetFontAscender
Render, 75	FTFont.h, 83
FTTextureGlyph.h, 101	ftglGetFontBBox
Trontarodiypinii, 101	ng. don one box

nt2om	CotFont
pt3em	GetFont
FTFont.h, 83	FTSimpleLayout, 71
ftglGetFontCharMapCount	GetLineLength
FTFont.h, 83	FTSimpleLayout, 71
ftglGetFontCharMapList	GetLineSpacing
FTFont.h, 83	FTSimpleLayout, 71
ftglGetFontDescender	GlyphLoadFlags
FTFont.h, 84	FTFont, 44
ftglGetFontError	
FTFont.h, 84	Height
ftglGetFontFaceSize	FTBuffer, 30
FTFont.h, 84	
ftglGetFontLineHeight	Invalidate
FTFont.h, 84	FTBBox, 24
	IsValid
ftglGetGlyphAdvance	FTBBox, 24
FTGlyph.h, 95	- ,
ftglGetGlyphBBox	LineHeight
FTGlyph.h, 95	FTFont, 44
ftglGetGlyphError	Lower
FTGlyph.h, 95	FTBBox, 25
ftglGetLayoutAlignement	1 1 BBOX, 23
FTSimpleLayout.h, 100	MakeGlyph
ftglGetLayoutBBox	FTBitmapFont, 27
FTLayout.h, 97	FTBufferFont, 32
ftglGetLayoutError	·
FTLayout.h, 97	FTExtrudeFont, 35
ftglGetLayoutFont	FTFont, 45
FTSimpleLayout.h, 100	FTOutlineFont, 54
ftglGetLayoutLineLength	FTPixmapFont, 57
- · · · · · · · · · · · · · · · · · · ·	FTPolygonFont, 67
FTSimpleLayout.h, 100	FTTextureFont, 74
ftglGetLayoutLineSpacing	
FTSimpleLayout.h, 100	Normalise
•	
ftglRenderFont	FTPoint, 61
ftglRenderFont FTFont.h, 85	FTPoint, 61
ftglRenderFont	FTPoint, 61 operator const FTGL_DOUBLE *
ftglRenderFont FTFont.h, 85	FTPoint, 61
ftglRenderFont FTFont.h, 85 ftglRenderGlyph	FTPoint, 61 operator const FTGL_DOUBLE *
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator*
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+=
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85	perator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator-
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85 ftglSetFontOutset	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator- FTPoint, 62
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85 ftglSetFontOutset FTFont.h, 86	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 62 operator-
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85 ftglSetFontOutset FTFont.h, 86 ftglSetLayoutAlignment	preparation operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 63
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85 ftglSetFontOutset FTFont.h, 86 ftglSetLayoutAlignment FTSimpleLayout.h, 101	preparation operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 63 operator- FTPoint, 63 operator-= FTPoint, 63 operator-= FTPoint, 63 operator-=
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85 ftglSetFontOutset FTFont.h, 86 ftglSetLayoutAlignment FTSimpleLayout.h, 101 ftglSetLayoutFont	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 63 operator-= FTPoint, 63 operator== FTPoint, 65
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85 ftglSetFontOutset FTFont.h, 86 ftglSetLayoutAlignment FTSimpleLayout.h, 101 ftglSetLayoutFont FTSimpleLayout.h, 101	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 63 operator- FTPoint, 65 Outset
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85 ftglSetFontOutset FTFont.h, 86 ftglSetLayoutAlignment FTSimpleLayout.h, 101 ftglSetLayoutFont	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 63 operator-= FTPoint, 63 operator== FTPoint, 65
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85 ftglSetFontOutset FTFont.h, 86 ftglSetLayoutAlignment FTSimpleLayout.h, 101 ftglSetLayoutFont FTSimpleLayout.h, 101	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 63 operator- FTPoint, 65 Outset FTFont, 45
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85 ftglSetFontOutset FTFont.h, 86 ftglSetLayoutAlignment FTSimpleLayout.h, 101 ftglSetLayoutFont FTSimpleLayout.h, 101 ftglSetLayoutLineLength	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 63 operator-= FTPoint, 63 operator-= FTPoint, 65 Outset FTFont, 45 Pixels
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85 ftglSetFontOutset FTFont.h, 86 ftglSetLayoutAlignment FTSimpleLayout.h, 101 ftglSetLayoutFont FTSimpleLayout.h, 101 ftglSetLayoutLineLength FTSimpleLayout.h, 101 ftglSetLayoutLineSpacing	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 63 operator- FTPoint, 65 Outset FTFont, 45
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85 ftglSetFontOutset FTFont.h, 86 ftglSetLayoutAlignment FTSimpleLayout.h, 101 ftglSetLayoutFont FTSimpleLayout.h, 101 ftglSetLayoutLineLength FTSimpleLayout.h, 101	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 63 operator-= FTPoint, 63 operator-= FTPoint, 65 Outset FTFont, 45 Pixels
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85 ftglSetFontOutset FTFont.h, 86 ftglSetLayoutAlignment FTSimpleLayout.h, 101 ftglSetLayoutFont FTSimpleLayout.h, 101 ftglSetLayoutLineLength FTSimpleLayout.h, 101 ftglSetLayoutLineSpacing	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 63 operator-= FTPoint, 63 operator== FTPoint, 65 Outset FTFont, 45 Pixels FTBuffer, 30
ftglRenderFont FTFont.h, 85 ftglRenderGlyph FTGlyph.h, 95 ftglRenderLayout FTLayout.h, 97 ftglSetFontCharMap FTFont.h, 85 ftglSetFontDepth FTFont.h, 85 ftglSetFontDisplayList FTFont.h, 85 ftglSetFontFaceSize FTFont.h, 85 ftglSetFontOutset FTFont.h, 86 ftglSetLayoutAlignment FTSimpleLayout.h, 101 ftglSetLayoutFont FTSimpleLayout.h, 101 ftglSetLayoutLineLength FTSimpleLayout.h, 101 ftglSetLayoutLineSpacing FTSimpleLayout.h, 101	FTPoint, 61 operator const FTGL_DOUBLE * FTPoint, 61 operator* FTPoint, 61, 64, 65 operator^ FTPoint, 63 operator+ FTPoint, 62 operator+= FTBBox, 25 FTPoint, 62 operator- FTPoint, 62 operator- FTPoint, 63 operator-= FTPoint, 63 operator== FTPoint, 65 Outset FTFont, 45 Pixels FTBuffer, 30 Pos

Ζ pt3em FTPoint, 64 RENDER_ALL Zf FTGL, 21 FTPoint, 64 RENDER_BACK FTGL, 21 RENDER FRONT FTGL, 21 RENDER_SIDE FTGL, 21 Render FTBitmapGlyph, 28 FTBufferGlyph, 34 FTExtrudeGlyph, 37 FTFont, 45, 46 FTGlyph, 49 FTLayout, 52, 53 FTOutlineGlyph, 56 FTPixmapGlyph, 59 FTPolygonGlyph, 68 FTSimpleLayout, 71 FTTextureGlyph, 75 RenderMode FTGL, 21 SetAlignment FTSimpleLayout, 72 SetDepth FTBBox, 25 SetFont FTSimpleLayout, 72 SetLineLength FTSimpleLayout, 72 SetLineSpacing FTSimpleLayout, 72 Size FTBuffer, 30 TextAlignment FTGL, 21 tutorial.dox, 101 Upper FTBBox, 25 UseDisplayList FTFont, 46 Width FTBuffer, 31 Χ FTPoint, 63 Χf FTPoint, 63 Υ FTPoint, 63, 64 Υf

FTPoint, 64