

B PDFlib Quick Reference

General Functions

Function prototype	page
<code>void PDF_boot(void)</code>	195
<code>void PDF_shutdown(void)</code>	195
<code>PDF *PDF_new(void)</code>	196
<code>PDF *PDF_new2(void (*errorhandler)(PDF *p, int errortype, const char *msg), void* (*allocproc)(PDF *p, size_t size, const char *caller), void* (*reallocproc)(PDF *p, void *mem, size_t size, const char *caller), void (*freeproc)(PDF *p, void *mem), void *opaque)</code>	196
<code>void PDF_delete(PDF *p)</code>	197
<code>int PDF_begin_document(PDF *p, const char *filename, int len, const char *optlist)</code>	198
<code>void PDF_begin_document_callback(PDF *p, size_t (*writeproc)(PDF *p, void *data, size_t size), const char *optlist)</code>	198
<code>void PDF_end_document(PDF *p, const char *optlist)</code>	200
<code>const char *PDF_get_buffer(PDF *p, long *size)</code>	200
<code>void PDF_begin_page_ext(PDF *p, double width, double height, const char *optlist)</code>	203
<code>void PDF_end_page_ext(PDF *p, const char *optlist)</code>	204
<code>void PDF_suspend_page(PDF *p, const char *optlist)</code>	205
<code>void PDF_resume_page(PDF *p, const char *optlist)</code>	206
<code>double PDF_get_value(PDF *p, const char *key, double modifier)</code>	207
<code>void PDF_set_value(PDF *p, const char *key, double value)</code>	207
<code>const char *PDF_get_parameter(PDF *p, const char *key, double modifier)</code>	207
<code>void PDF_set_parameter(PDF *p, const char *key, const char *value)</code>	208
<code>void PDF_create_pvf(PDF *p, const char *filename, int len, const void *data, size_t size, const char *optlist)</code>	208
<code>int PDF_delete_pvf(PDF *p, const char *filename, int len)</code>	209
<code>int PDF_get_errnum(PDF *p)</code>	210
<code>const char *PDF_get_errmsg(PDF *p)</code>	210
<code>const char *PDF_get_apiname(PDF *p)</code>	210
<code>void *PDF_get_opaque(PDF *p)</code>	211
<code>const char *PDF_utf16_to_utf8(PDF *p, const char *utf16string, int len, int *size)</code>	211
<code>const char *PDF_utf8_to_utf16(PDF *p, const char *utf8string, const char *ordering, int *size)</code>	211

Font Functions

Function prototype	page
<code>int PDF_load_font(PDF *p, const char *fontname, int len, const char *encoding, const char *optlist)</code>	214
<code>void PDF_setfont(PDF *p, int font, double fontsize)</code>	216
<code>void PDF_begin_font(PDF *p, char *fontname, int reserved, double a, double b, double c, double d, double e, double f, const char *optlist)</code>	217
<code>void PDF_end_font(PDF *p)</code>	217
<code>void PDF_begin_glyph(PDF *p, char *glyphname, double wx, double llx, double lly, double urx, double ury)</code>	218
<code>void PDF_end_glyph(PDF *p)</code>	218
<code>void PDF_encoding_set_char(PDF *p, const char *encoding, int slot, const char *glyphname, int uv)</code>	219

Text Output Functions

Function prototype	page
<code>void PDF_set_text_pos(PDF *p, double x, double y)</code>	221
<code>void PDF_show(PDF *p, const char *text)</code>	221
<code>void PDF_xshow(PDF *p, const char *text, int len, const double *xadvancelist)</code>	222
<code>void PDF_show_xy(PDF *p, const char *text, double x, double y)</code>	222
<code>void PDF_continue_text(PDF *p, const char *text)</code>	223
<code>void PDF_fit_textline(PDF *p, const char *text, int len, double x, double y, const char *optlist)</code>	223
<code>double PDF_stringwidth(PDF *p, const char *text, int font, double fontsize)</code>	226
<code>int PDF_create_textflow(PDF *p, const char *text, int len, const char *optlist)</code>	227
<code>const char *PDF_fit_textflow(PDF *p, int textflow, double llx, double lly, double urx, double ury, const char *optlist)</code>	228
<code>double PDF_info_textflow(PDF *p, int textflow, const char *keyword)</code>	230
<code>void PDF_delete_textflow(PDF *p, int textflow)</code>	231

Graphics Functions

Function prototype	page
<code>void PDF_setdash(PDF *p, double b, double w)</code>	236
<code>void PDF_setdashpattern(PDF *p, const char *optlist)</code>	236
<code>void PDF_setflat(PDF *p, double flatness)</code>	237
<code>void PDF_setlinejoin(PDF *p, int linejoin)</code>	237
<code>void PDF_setlinecap(PDF *p, int linecap)</code>	237
<code>void PDF_setmiterlimit(PDF *p, double miter)</code>	238
<code>void PDF_setlinewidth(PDF *p, double width)</code>	238
<code>void PDF_initgraphics(PDF *p)</code>	238
<code>void PDF_save(PDF *p)</code>	239
<code>void PDF_restore(PDF *p)</code>	239
<code>void PDF_translate(PDF *p, double tx, double ty)</code>	240
<code>void PDF_scale(PDF *p, double sx, double sy)</code>	240
<code>void PDF_rotate(PDF *p, double phi)</code>	240
<code>void PDF_skew(PDF *p, double alpha, double beta)</code>	241
<code>void PDF_concat(PDF *p, double a, double b, double c, double d, double e, double f)</code>	241
<code>void PDF_setmatrix(PDF *p, double a, double b, double c, double d, double e, double f)</code>	241
<code>int PDF_create_gstate(PDF *p, const char *optlist)</code>	242
<code>void PDF_set_gstate(PDF *p, int gstate)</code>	243
<code>void PDF_moveto(PDF *p, double x, double y)</code>	243
<code>void PDF_lineto(PDF *p, double x, double y)</code>	243
<code>void PDF_curveto(PDF *p, double x1, double y1, double x2, double y2, double x3, double y3)</code>	244
<code>void PDF_circle(PDF *p, double x, double y, double r)</code>	244
<code>void PDF_arc(PDF *p, double x, double y, double r, double alpha, double beta)</code>	245
<code>void PDF_arcn(PDF *p, double x, double y, double r, double alpha, double beta)</code>	245
<code>void PDF_rect(PDF *p, double x, double y, double width, double height)</code>	245
<code>void PDF_closepath(PDF *p)</code>	246
<code>void PDF_stroke(PDF *p)</code>	246

Function prototype	page
<code>void PDF_closepath_stroke(PDF *p)</code>	246
<code>void PDF_fill(PDF *p)</code>	247
<code>void PDF_fill_stroke(PDF *p)</code>	247
<code>void PDF_closepath_fill_stroke(PDF *p)</code>	247
<code>void PDF_clip(PDF *p)</code>	247
<code>void PDF_endpath(PDF *p)</code>	248
<code>int PDF_define_layer(PDF *p, const char *name, int len, const char *optlist)</code>	248
<code>void PDF_set_layer_dependency(PDF *p, const char *type, const char *optlist)</code>	249
<code>void PDF_begin_layer(PDF *p, int layer)</code>	250
<code>void PDF_end_layer(PDF *p)</code>	250

Color Functions

Function prototype	page
<code>void PDF_setcolor(PDF *p, const char *fstype, const char *colorspace, double c1, double c2, double c3, double c4)</code> 251	
<code>int PDF_makespotcolor(PDF *p, const char *spotname, int reserved)</code>	252
<code>int PDF_load_iccprofile(PDF *p, const char *filename, int len, const char *optlist)</code>	253
<code>int PDF_begin_pattern(PDF *p, double width, double height, double xstep, double ystep, int painttype)</code>	255
<code>void PDF_end_pattern(PDF *p)</code>	255
<code>int PDF_shading_pattern(PDF *p, int shading, const char *optlist)</code>	255
<code>void PDF_shfill(PDF *p, int shading)</code>	256
<code>int PDF_shading(PDF *p, const char *shtype, double xo, double yo, double x1, double y1, double c1, double c2, double c3, double c4, const char *optlist)</code>	256

Image Functions

Function prototype	page
<code>int PDF_load_image(PDF *p, const char *imagetype, const char *filename, int len, const char *optlist)</code>	258
<code>void PDF_close_image(PDF *p, int image)</code>	262
<code>void PDF_fit_image(PDF *p, int image, double x, double y, const char *optlist)</code>	262
<code>int PDF_begin_template(PDF *p, double width, double height)</code>	264
<code>void PDF_end_template(PDF *p)</code>	265
<code>void PDF_add_thumbnail(PDF *p, int image)</code>	265

PDF Import Functions (PDI)

Function prototype	page
<code>int PDF_open_pdi(PDF *p, const char *filename, const char *optlist, int len)</code>	266
<code>int PDF_open_pdi_callback(PDF *p, void *opaque, size_t filesize, size_t (*readproc)(void *opaque, void *buffer, size_t size), int (*seekproc)(void *opaque, long offset), const char *optlist)</code>	267
<code>void PDF_close_pdi(PDF *p, int doc)</code>	268
<code>int PDF_open_pdi_page(PDF *p, int doc, int pagenumber, const char *optlist)</code>	268
<code>void PDF_close_pdi_page(PDF *p, int page)</code>	269

Function prototype	page
<code>void PDF_fit_pdi_page(PDF *p, int page, double x, double y, const char *optlist)</code>	270
<code>int PDF_process_pdi(PDF *p, int doc, int page, const char *optlist)</code>	270
<code>double PDF_get_pdi_value(PDF *p, const char *key, int doc, int page, int reserved)</code>	271
<code>const char *PDF_get_pdi_parameter(PDF *p, const char *key, int doc, int page, int reserved, int *len)</code>	272

Block Filling Functions (PPS)

Function prototype	page
<code>int PDF_fill_textblock(PDF *p, int page, const char *blockname, const char *text, int len, const char *optlist)</code>	274
<code>int PDF_fill_imageblock(PDF *p, int page, const char *blockname, int image, const char *optlist)</code>	275
<code>int PDF_fill_pdfblock(PDF *p, int page, const char *blockname, int contents, const char *optlist)</code>	275

Hypertext Functions

Function prototype	page
<code>int PDF_create_action(PDF *p, const char *type, const char *optlist)</code>	278
<code>void PDF_add_nameddest(PDF *p, const char *name, int len, const char *optlist)</code>	281
<code>void PDF_create_annotation(PDF *p, double llx, double lly, double urx, double ury, const char *type, const char *optlist)</code>	283
<code>void PDF_create_field(PDF *p, double llx, double lly, double urx, double ury, const char *name, int len, const char *type, const char *optlist)</code>	286
<code>void PDF_create_fieldgroup(PDF *p, const char *name, int len, const char *optlist)</code>	287
<code>int PDF_create_bookmark(PDF *p, const char *text, int len, const char *optlist)</code>	292
<code>void PDF_set_info(PDF *p, const char *key, const char *value)</code>	293

Tagged PDF and Structure Functions

Function prototype	page
<code>int PDF_begin_item(PDF *p, const char *tag, const char *optlist)</code>	296
<code>void PDF_end_item(PDF *p, int id)</code>	298
<code>void PDF_activate_item(PDF *p, int id)</code>	298