

As funções FPDF sempre retornam um valor lógico. O primeiro parâmetro é sempre o pointer para o documento.

`<l>=fpdf_addlink (<p>, <iden>)` - cria um link interno e retorna seu número em <iden>

`<l>=fpdf_addpage (<p> [,orient>])` - cria uma nova página com a orientação "P" ou "L"

`<l>=fpdf_aliasnbpages (<p> [,<cad>])` - define a cadeia para substituir pelo número da página; default é "{nb}"

`<l>=fpdf_cell (<p>, <w> [, <h> [, <text> [, <bord> [, <salt> [, <alin> [, <ench> [, <link>]]]]]]])` - apresenta uma célula (área retangular) com bordas opcionais, cor de fundo e uma cadeia de caracteres; o canto superior da célula corresponde a posição corrente; o texto pode ser alinhado ou centrado; após a chamada a posição corrente move-se para a direita ou para a próxima linha; é possível colocar um link no texto; se a quebra de página automática esta ligada e a célula se estende após o limite, uma quebra de página é realizada antes da exibição; <w> é a largura da célula; se 0, a célula se estende até a margem direita; <h> é altura da célula; o valor default é 0; <text> é a cadeia a imprimir; o valor default é uma cadeia vazia; <bord> indica se uma borda deve ser desenhada em volta da célula; o valor pode ser : "0" sem borda (valor default), "1" com borda, ou uma cadeia contendo algum ou todas as letras em qualquer ordem : "L" borda esquerda, "T" borda superior, "R" borda da direita, "B" borda inferior; <salt> indica para onde a posição corrente deve ir; valores possíveis são: "0" para a direita (default), "1" para o inicio da próxima linha, "2" abaixo; especificar "1" é equivalente a especificar "0" e chamar fpdf_ln a seguir; <alin> permite centralizar ou alinhar o texto; valores possíveis são: "L" ou cadeia vazia para alinhar à esquerda (valor default), "C": centralizar ou "R" alinhar à direita; <ench> indica se o fundo da célula sera pintado "1" ou transparente: "0" (default); <link> é a URL ou identificador retornado por fpdf_addlink.

`<l>=fpdf_close (<p>)` - termina o documento

`<l>=fpdf_setfooter (<p>, <rot>)` - indica a rotina de rodapé

`<l>=fpdf_new (<p>, [, <orient> [, <unid> [, <form>]])` - cria a estrutura para o PDF. <orient> pode ser "P" (default) ou "L". <unid> é a unidade de medida, pode ser "pt" point, "mm": milimetro (default), "cm": centimetro ou "in": polegada. <form> pode ser "A3", "A4", "A5", "letter" ou "legal".

`<l>=fpdf_getstringwidth (<p>, <cad>)` - calcula o tamanho da cadeia

`<l>=fpdf_getx (<p>, <num>)` - devolve a posição x corrente em <num>

`<l>=fpdf_gety (<p>, <num>)` - devolve a posição y corrente em <num>

`<l>=fpdf_setheader (<p>, <rot>)` - indica a rotina de cabeçalho

`<l>=fpdf_image (<p>, <arq>, <x>, <y> [, <w> [, <h> [, <tipo> [, <link>]]]])` - coloca uma imagem na página. <arq> é o nome do arquivo contendo a imagem. <x> é a abcissa do canto superior esquerdo <y> é a ordenada do canto superior esquerdo; <w> é a largura da imagem; se não especificada ou zero, é automaticamente calculada. <h> é a altura da imagem na página; se não especificada ou zero, é automaticamente calculada; se nenhuma das duas <w> e <h> for especificada, a imagem é exibida em 72dpi. <tipo> é o formato da imagem; pode ser JPG, JPEG ou PNG; se não especificada o tipo é obtido do extensão do arquivo. <link> é a URL ou o identificador retornado por fpdf_addlink.

`<l>=fpdf_line (<p>, <x1>, <y1>, <x2>, <y2>)` - <x1> é a abcissa do 1º. ponto, <y1> é ordenada do 1º. ponto. <x2> é a abcissa do 2º. ponto, <y2> é ordenada do 2º. ponto.

`<l>=fpdf_link (<p>, <x>, <y>, <w>, <h>, <link>)` - põe um link numa área retangular de uma página;; links de texto ou imagem são geralmente postos via fpdf_cell, fpdf_write ou fpdf_image, mas esta função pode ser útil por exemplo, para definir uma área clicável dentro de uma imagem

`<l>=fpdf_ln (<p> [, <h>])` - executa uma quebra de linha; a abcissa corrente volta para a margem esquerda e a ordenada aumenta pelo valor passado pelo parâmetro `<h>` ; por default o valor de `<h>` é o valor da altura da última célula impressa.

`<l>=fpdf_multicell (<p>, <w>, <h>, <texto> [, <bord> [, <alinh> [, <ench.>]])` - esta função permite a impressão de texto com quebras de linha que podem ser automáticas (logo que o texto alcança a borda direita da célula) ou explícita (via o caracter `\n`); tantas células são geradas quanto necessárias, uma abaixo da outra; `<w>` indica a largura das células; se 0, a célula se estende até a margem direita da página; `<h>` indica a altura das células; `<texto>` especifica o texto a imprimir; `<bord>` Indica se bordas serão desenhadas ao redor da célula; o valor pode ser "0": sem borda (default), "1" com borda, ou uma cadeia contendo uma ou mais das letras em qualquer ordem : "L" esquerda, "T" superior, "R" direita, "B" inferior; `<alinh>` especifica o alinhamento do texto; pode ser "L" alinha à esquerda, "C" alinha ao centro, "R" alinha à direita ou "J" justifica o texto (default); `<ench.>` Indica se o fundo deve ser pintado ("1") ou transparente ("0") que é o default.

`<l>=fpdf_output (<p>, <arq>)` - grava o documento no arquivo `<arq>`

`<l>=fpdf_pageno (<p>, <num>)` - retorna em `<num>` o número da página corrente

`<l>=fpdf_rect (<p>, <x>, <y>, <w>, <h> [, <esti>])` - Gera um retângulo; pode ser desenhado (só a borda), cheio (sem borda) ou ambos; `<esti>` pode ser "D" ou cadeia vazia (default), "F" preencha, "DF" ou "FD" : desenha e preencha

`<l>=fpdf_setacceptpagebreak (<p>, <rot>)` - indica a rotina a ser chamada no caso de quebra de página

`<l>=fpdf_setauthor (<p>, <autor>)` - indica o autor do documento

`<l>=fpdf_setautopagebreak (<p>, <auto> [, <marg>])` - Habilita ou desabilita o modo de quebra de página automático `<auto.>` é "0" ou "1". Quando habilita o segundo parametro é a distância do fundo da página que define o limite de quebra; por default o modo está habilitado e a margem é 2 cm.

`<l>=fpdf_setcompression (<p>, <op>)` - ativa ou desativa a compressão; `<op>` e igual a "0" ou "1".

`<l>=fpdf_setcreator (<p>, <criador>)` - indica o criador do documento

`<l>=fpdf_setdisplaymode (<p> , <zoom> [, <layout>])` - define o modo que o documento será apresentado. `<zoom>` pode ser "fullpage", "fullwidth", "real" ou "default" (do Reader) ou um numero indicando o fator de zoom a utilizar. O `<layout>` da pagina pode ser "single", "continuous" (default), "two" ou "default" (do Reader). Por default, os documentos usam o "fullwidth" da janela e apresentação "continuous".

`<l>=fpdf_setdrawcolor (<p>, <r> [, <g> ,])` - define a cor usada para todas as operações de desenho (linhas, retangulos e bordas de células). Pode ser expressa em componentes RGB ou escala de cinza; pode ser chamado antes da primeira página ser criada; o valor é retido de página para página.; se `<g>` e `` são dados `<r>` indica o componente vermelho, se não indica o nível de cinza; valor entre 0 e 255.

`<l>=fpdf_setfillcolor (<p>, <r> [, <g> ,])` - define a cor usada para todas as operações de preenchimento (retângulos cheios e fundo de células). Pode ser expressa em componentes RGB ou escala de cinza; pode ser chamado antes da primeira página ser criada; o valor é retido de página para página.; se `<g>` e `` são dados `<r>` indica o componente vermelho, se não indica o nível de cinza; valor entre 0 e 255.

`<l>=fpdf_setfont (<p>, <familia> [, <estilo> [tamanho]])` - define a fonte para impressão de texto; é obrigatório chamar esta função pelo menos uma vez antes de imprimir texto; `<familia>` deve ser uma das famílias padrão: "Courier" (largura fixa) "Helvetica" ou "Arial" (sinônmos, sans serif), "Times" (serif), Symbol (simbolica) ou "ZapfDingbats" (simbolica); é possível passar uma cadeia vazia; neste caso a família corrente é mantida; `<estilo>` pode ser: cadeia vazia :regular, "B" : bold , "I": itálica, "U" : sublinhada

ou qualquer combinação; o valor default é regular; bold e italico não se aplicam a Symbol e ZapfDingbats; <tamanho> é tamanho da fonte em points; o valor default é o tamanho corrente; se nenhum tamanho foi especificado desde o inicio do documento, o tamanho é 12.

<l>=**fpdf_setfontsize** (<p>, <tamanho>) - define o tamanho da fonte corrente

<l>=**fpdf_setkeywords** (<p>, <palavrs>) - associa uma lista de palavras chaves ao documento; geralmente na forma "palavra1 palavra2 ..."

<l>=**fpdf_setleftmargin** (<p>, <margem>) - define a margem esquerda; pode ser chamado antes de criar a primeira página.

<l>=**fpdf_setlinewidth** (<p>, <tamanho>) - define o tamanho de linha por default o tamanho é 0.2 mm; a função deve ser chamada antes da criação da primeira página e é retido de página para página.

<l>=**fpdf_setlink** (<p>, <link> [, <y> [, <pag>]]) - define a página e posição para o qual um link aponta. <link> É o identificador retornado por fpdf_addlink; <y> é a ordenada da posição alvo; -1 indica a posição corrente; o valor default é 0 (alto da página); <pag> indica o número da página alvo ; -1 indica a página corrente (default)

<l>=**fpdf_setmargins** (<p>, <esq>, <sup> [, <dir>]) - define as margens esquerda, superior e direita; por default são iguais a 1 cm; se <dir> não for especificada fica igual a <esq>.

<l>=**fpdf_setrightmargin** (<p>, <margem>) - define a margem direita; pode ser chamado antes de criar a primeira página.

<l>=**fpdf_setsubject** (<p>, <assunto>) - indica o assunto do documento

<l>=**fpdf_settextcolor** (<p>, <r> [, <g> ,]) - define a cor usada para o texto. Pode ser expressa em componentes RGB ou escala de cinza; pode ser chamado antes da primeira página ser criada; o valor é retido de página para página.; se <g> e são dados <r> indica o componente vermelho, se não indica o nivel de cinza; valor entre 0 e 255.

<l>=**fpdf_settitle** (<p>, <titulo>) - indica o titulo do documento

<l>=**fpdf_settopmargin** (<p>, <margem>) - define a margem superior; pode ser chamado antes de criar a primeira página.

<l>=**fpdf_setx** (<p>, <x>) - define a abcissa da posição corrente; se o valor passado for negativo, é relativo a margem direita da página

<l>=**fpdf_setxy** (<p>, <x>, <y>) - define a abcissa <x> e ordenada <y> para a posição corrente. Se os valores passados são negativos, são relativos a direita e fundo da página

<l>=**fpdf_sety** (<p>, <y>) - Move a abcissa corrente para a margem esquerda e estabelece a ordenada <y>. Se o valor passado é negativo, é relativo ao fundo da página

<l>=**fpdf_text** (<p>, <x>, <y>, <texto>) - exibe uma cadeia; <x> é a abcissa da origem, <y> é a ordenada da origem e <texto> a cadeia a exibir

<l>=**fpdf_write** (<p>, <altura>, <texto> [, <link>]) - exibe o texto indicado; <link> é uma URL ou o identificador retornado em fpdf_addlink

Exemplo 1

prog

```

pointer p
x=fpdf_new(p)
x=fpdf_setcompression(p,"0")
x=fpdf_AddPage(p)
x=fpdf_SetFont(p,"Courier","B",16)
x=fpdf_Cell(p,40,10,"Hello World!")
x=fpdf_Output(p, "tuto1.pdf")

```

Exemplo 2

```

$noLib
prog
pointer p
external cabec
external rodape
x=fpdf_new(p)
x=fpdf_setcompression(p,"0")
x=fpdf_setheader(p,cabec)
x=fpdf_setfooter(p,rodape)
x=fpdf_AliasNbPages(p)
x=fpdf_AddPage(p)
x=fpdf_SetFont(p, "Times","",12)
for i=1 to 40
  x=fpdf_Cell(p, 0, 10, "Imprimindo a linha numero "+str(i), "0", 1)
next
x=fpdf_Output(p, "fpdf2.pdf" )

```

```

proc cabec
parameter p(P)
  &&Logo
  x=fpdf_image(p, "logo_pb.png",10,8,33)
  &&Arial bold 15
  x=fpdf_SetFont(p, "Arial","B",15)
  &&Move to the right
  x=fpdf_cell(p, 80)
  &&Title
  x=fpdf_cell(p, 80,10,"Titulo do Documento","1",0,"C")
  &&Line break
  x=fpdf_Ln(p, 20)
return

```

```

proc rodape
parameter p(P)
  && posiciona a 1.5 cm do fundo
  x=fpdf_SetY(p, -15)
  &&Arial italic 8
  x=fpdf_SetFont(p, "Arial", "I", 8)
  &&Page number
  pg=0
  x=fpdf_PageNo(p, pg)
  cad = "Página " + str(pg) + "/{nb}"
  x=fpdf_Cell(p, 0, 10, cad , "0", 0, "C")
return

```

Exemplo 3

```

$noLib
prog

```

```

pointer p
public title
external header, footer
x=fpdf_new(p)
x=fpdf_SetCompression(p,"0")
x=fpdf_setheader(p,header)
x=fpdf_setfooter(p,footer)
title="Documento PDF com OPUS"
x=fpdf_SetTitle(p,title)
x=fpdf_SetAuthor(p,"Suporte OpenBase")
PrintChapter(p,1,"Relação de Nomes:", "nome.txt")
PrintChapter(p,2,"Profissionais Qualificados", "nome2.txt")
x=fpdf_Output(p,"fpdf3.pdf")
return

```

```

proc header
parameter p(p)
public title
  x=fpdf_SetFont(p,"Arial","B",15)
  w=0
  x=fpdf_GetStringWidth(p,title,w)
  w=w+6
  x=fpdf_SetX(p,(210-w)/2)
  x=fpdf_SetDrawColor(p,0,80,180)
  x=fpdf_SetFillColor(p,230,230,0)
  x=fpdf_SetTextColor(p,220,50,50)
  x=fpdf_SetLineWidth(p,1)
  x=fpdf_Cell(p,w,9,title,1,1,"c",1)
  x=fpdf_Ln(p,10)
return

```

```

proc footer
parameter p(p)
  x=fpdf_SetY(p,-15)
  x=fpdf_SetFont(p,"Arial","I",8)
  x=fpdf_SetTextColor(p,128)
  pg=0
  x=fpdf_PageNo(p,pg)
  x=fpdf_Cell(p,0,10,"Pagina" + str(pg),0,0,"c")
return

```

```

proc ChapterTitle
parameters p(p), num(n), labell
  &&Arial 12
  x=fpdf_SetFont(p,"Arial","",12)
  &&&Background color
  x=fpdf_SetFillColor(p,200,220,255)
  &&Title
  x=fpdf_Cell(p,0,6,"Pagina" + str(num) + " : " + labell,0,1,"L",1)
  &&Line break
  x=fpdf_Ln(p,4)
return

```

```

proc ChapterBody
parameters p(p), file1

```

```

    &&Read text file
private txt(30000)
x=memoread(file1,txt)
    &&Times 12
    x=fpdf_SetFont(p,"Times","",12)
    &&Output justified text
    x=fpdf_MultiCell(p,0,5,txt)
    &&Line break
    x=fpdf_Ln(p)
    &&Mention in italics
    x=fpdf_SetFont(p,"","I")
    x=fpdf_Cell(p,0,5,"(Fim)")
return

proc PrintChapter
parameters p(p), num(n), title, file
    x=fpdf_AddPage(p)
    ChapterTitle(p,num,title)
    ChapterBody(p,file)
return

```

Exemplo 4

```

$noLib
prog
public scol(n),y0(n)
public title
pointer p
external header, footer, quebra
x=fpdf_new(p)
x=fpdf_SetCompression(p,"0")
x=fpdf_setheader(p,header)
x=fpdf_setfooter(p,footer)
x=fpdf_setfooter(p,quebra)
x=fpdf_setacceptpagebreak(p,quebra)
title="Documento PDF com OPUS"
x=fpdf_SetTitle(p,title)
x=fpdf_SetAuthor(p,"Suporte Banco de Dados OpenBase")
PrintChapter(p,1,"Relação de Nomes:", "nome.txt")
PrintChapter(p,2,"Profissionais Qualificados", "nome2.txt")
x=fpdf_Output(p,"fpdf4.pdf")
quit

proc header
parameter p(p)
public title
    x=fpdf_SetFont(p,"Arial","B",15)
    t=0
    x=fpdf_GetStringWidth(p,title,t)
    w=t+6
    x=fpdf_SetX(p,(210-w)/2)
    x=fpdf_SetDrawColor(p,0,80,180)
    x=fpdf_SetFillColor(p,230,230,0)
    x=fpdf_SetTextColor(p,220,50,50)

```

```

x=fpdf_SetLineWidth(p,1)
x=fpdf_Cell(p,w,9,title,1,1,"c",1)
y=0
x=fpdf_Gety(p,y)
y0=y
return

```

```

proc footer
parameter p(p)
x=fpdf_SetY(p,-15)
x=fpdf_SetFont(p,"Arial","I",8)
x=fpdf_SetTextColor(p,128)
pg=0
x=fpdf_PageNo(p,pg)
x=fpdf_Cell(p,0,10,"Pagina" + str(pg),0,0,"c")
return

```

```

proc SetCol
parameters p(p), coll(n)
public scol(n)
x1=10+scol*65
x=fpdf_SetLeftMargin(p,x1)
x=fpdf_SetX(p,x1)
return

```

```

function quebra
parameter p(p)
public scol(n), y0(n)
external SetCol
if (scol < 2)
SetCol(p,scol+1)
x=fpdf_SetY(p,y0)
return .F.
else
SetCol(p,0)
return .T.
endif
return .T.

```

```

proc ChapterTitle
parameters p(p), num(n), labell
public y0(n)
&& Title
x=fpdf_SetFont(p,"Arial","",12)
&&&Background color
x=fpdf_SetFillColor(p,200,220,255)
x=fpdf_Cell(p,0,6,"Pagina" + str(num) + " : " + labell,0,1,"L",1)
&&Line break
x=fpdf_Ln(p,4)
y=0
x=fpdf_GeY(p,y)
y0=y
return

```

```

proc ChapterBody
parameters p(p), file1
  &&Read text file
private txt(20000)
x=memoread(file1,txt)
  &&Times 12
x=fpdf_SetFont(p,"Times","",12)
  &&Output justified text
x=fpdf_MultiCell(p,60,5,txt)
x=fpdf_Ln(p)
  &&Mention in italics
x=fpdf_SetFont(p,"","I")
x=fpdf_Cell(p,0,5,"(Fim)")
SetCol(p,0)
return

proc PrintChapter
parameters p(p), num(n), title, file
x=fpdf_AddPage(p)
ChapterTitle(p,num,title)
ChapterBody(p,file)
return

```

Exemplo 5

```

$noLib
prog
pointer p
external LoadData, BasicTable, ImprovedTable
x=fpdf_new(p)
x=fpdf_SetCompression(p, "0")
public nh(n)
nh=4
decl header[nh]=space(20)
header[1]='País '
header[2]='Capital '
header[3]='Área (sq km) '
header[4]='População '
public np(n)
np=15
decl data[15]=space(40)
LoadData(p, 'paises.txt',data)
x=fpdf_SetFont(p, 'Arial','',14)
x=fpdf_AddPage(p)
BasicTable(p, header, data)
x=fpdf_AddPage(p, )
ImprovedTable(p, header,data)
x=fpdf_AddPage(p, )
FancyTable(p, header,data)

```

```

x=fpdf_Output(p, 'fpdf5.pdf')
return

proc LoadData
parameters p(p),file1, data[]
use *
$ 1 len(40)
a u40
enduse
FILE=file1
locate
i=0
do while .not. eof
    ++i
    data[i]=a
    continue
enddo
close file
return

proc BasicTable
parameters p(p),header[],data[]
public nh(n), np(n)
    for i=1 to nh
        x=fpdf_Cell(p, 40,7,header[i],1)
    next
x=fpdf_Ln(p)
for i=1 to np
    dat=data[i]
    coll=word(dat,";",1)
    x=fpdf_Cell(p,40,6,coll,1)
    for j=2 to nh
        coll=word(dat,";",0)
        x=fpdf_Cell(p,40,6,coll,1)
    next
    x=fpdf_Ln(p)
next
return

proc ImprovedTable
parameters p(p),header[],data[]
public nh(n)
    decl w[4]=0
    w[1]=40
    w[2]=35
    w[3]=40
    w[4]=45
    t=0
    for i=1 to 4
        t=t+w[i]
    next
    for i=1 to nh
        x=fpdf_Cell(p, w[i],7,header[i],1,0,'C')
    next
x=fpdf_Ln(p)
for i=1 to np
    dat=data[i]

```

```

coll=word(dat,";",1)
x=fpdf_Cell(p, w[1],6,col1,'LR')
coll=word(dat,";",0)
x=fpdf_Cell(p, w[2],6,col1,'LR')
coll=word(dat,";",0)
l=len(coll)
if l>3
    col2=left(coll,l-3)+", "+right(coll,3)
else
    col2=coll
endif
x=fpdf_Cell(p, w[3],6,col2,'LR',0,'R')
coll=word(dat,";",0)
l=len(coll)
if l>3
    col2=left(coll,l-3)+", "+right(coll,3)
else
    col2=coll
endif
x=fpdf_Cell(p, w[4], 6, col2, 'LR', 0, 'R')
x=fpdf_Ln(p)
next
&& Closure line
x=fpdf_Cell(p, t, 0, '', 'T')
return

```

&& Colored table

```

proc FancyTable
parameters p(p),header[],data[]
public nh(n)
    && Colors, line width and bold font
    x=fpdf_SetFillColor(p, 255,0,0)
    x=fpdf_SetTextColor(p, 255)
    x=fpdf_SetDrawColor(p, 128,0,0)
    x=fpdf_SetLineWidth(p, .3)
    x=fpdf_SetFont(p, '', 'B')
    && Header
    decl w[4]=0
    w[1]=40
    w[2]=35
    w[3]=40
    w[4]=45
    t=0
    for i=1 to 4
        t=t+w[i]
    next
    for i=1 to nh
        x=fpdf_Cell(p, w[i],7,header[i],1,0,'C',1)
    next
    x=fpdf_Ln(p)
    x=fpdf_SetFillColor(p, 224,235,255)
    x=fpdf_SetTextColor(p, 0)
    x=fpdf_SetFont(p, '')
    && Data
    fill1=0
    for i=1 to np
        dat=data[i]
    next
endproc

```

```

col2=word(dat,";",1)
x=fpdf_Cell(p, w[1],6,col2,'LR',0,'L',fill1)
col2=word(dat,";",0)
x=fpdf_Cell(p, w[2],6,col2,'LR',0,'L',fill1)
col1=word(dat,";",0)
l=len(col1)
if l>3
    col2=left(col1,l-3)+", "+right(col1,3)
else
    col2=col1
endif
x=fpdf_Cell(p, w[3],6,col2,'LR',0,'R',fill1)
col1=word(dat,";",0)
l=len(col1)
if l>3
    col2=left(col1,l-3)+", "+right(col1,3)
else
    col2=col1
endif
x=fpdf_Cell(p, w[4], 6, col2, 'LR', 0, 'R',fill1)
x=fpdf_Ln(p)
if fill1=0
    fill1=1
else
    fill1=0
endif
next
x=fpdf_Cell(p, t, 0, '', 'T')
return

```

Exemplo 6

```

$noLib
$lentmp=400
prog
pointer p
public B(n)
public I(n)
public U(n)
public HREF
private html(400)
    && Inicializacao
    B=0
    I=0
    U=0
    HREF=''
external WriteHTML
html='Você pode acessar agora e saber mais sobre os diferentes formatos e
tipos de fontes. <BR>'

```

```
html=html+' Exemplo: <B> bold </B>, <I> italic </I>, <U> underlined </U>
<BR>'
html=html+' Consulte agora <A HREF="http://www.fpdf.org"> www.fpdf.org </A>,'
html=html+' .'
```

```
x=fpdf_new(p)
x=fpdf_SetCompression(p, "0")
x=fpdf_AddPage(p)
x=fpdf_SetFont(p, 'Arial','',20)
x=fpdf_Write(p, 5,[Para saber mais, consulte o tutorial, clique ])
x=fpdf_SetFont(p, '', 'U')
link=0
x=fpdf_AddLink(p, link)
x=fpdf_Write(p, 5,'Aqui',link)
x=fpdf_SetFont(p, '')
&& Second page
x=fpdf_AddPage(p)
x=fpdf_SetLink(p, link)
x=fpdf_image(p, "logo_pb.png",10,8,33)
x=fpdf_SetLeftMargin(p, 45)
x=fpdf_SetFontSize(p, 14)
WriteHTML(p, html)
x=fpdf_Output(p, 'fpdf6.pdf')
```

```
proc WriteHTML
parameters p(p),html
public HREF
private texto(400)
  i=at(chr(10),html)
  do while i>0
    html=stuff(html,i,1," ")
    i=at(chr(10),html)
  enddo
  sep=[ >"]=]
  p1=word(html,sep,1)
  l=len(p1)
  texto=""
  do while l > 0
    if left(p1,1) <> "<"
      if .not. empty(HREF)
        PutLink(p, HREF, p1)
      else
        texto=texto+p1+" "
      endif
    else
      if .not. empty(texto)
        x=fpdf_Write(p, 5, texto)
        texto=""
      endif
      tag=substr(p1,2,1)
      if substr(p1,2,1) = "/"
        CloseTag(p, substr(p1,3,l-1))
      else
        if "BR" $ p1
          tag="BR"
        endif
        attr=""
      endif
    endif
  enddo
endproc
```

```

        if tag="A"
            p1=word(html,sep,0)
            if left(p1,4)="HREF"
                attr=word(html,sep,0)
            endif
        endif
        OpenTag(p, tag, attr)
    endif
    endif
    p1=word(html,sep,0)
    l=len(p1)
enddo
if .not. empty(texto)
    x=fpdf_Write(p, 5, texto)
endif
return

```

```

proc OpenTag
parameters p(p),tag,attr
public HREF
    && Opening tag
    if tag="B" .or. tag="I" .or. tag="U"
        SetStyle(p, tag, "1")
    endif
    if tag="A"
        HREF=attr
    endif
    if tag="BR"
        x=fpdf_Ln(p, 5)
    endif
return

```

```

proc CloseTag
parameters p(p),tag
public HREF
    && Closing tag
    if tag="B" .or. tag="I" .or. tag="U"
        SetStyle(p, tag, "0")
    endif
    if tag="A"
        HREF=''
    endif
return

```

```

proc SetStyle
parameters p(p),tag,enable
public B(n)
public I(n)
public U(n)
    && Modify style and select corresponding font
    if enable="1"
        if tag="B"
            B=1
        endif
        if tag="I"
            I=1
        endif
    endif

```

```

        if tag="U"
            U=1
        endif
    else
        if tag="B"
            B=-1
        endif
        if tag="I"
            I=-1
        endif
        if tag="U"
            U=-11
        endif
    endif
    style=''
    if B>0
        style=style+"B"
    endif
    if I>0
        style=style+"I"
    endif
    if U>0
        style=style+"U"
    endif
    x=fpdf_SetFont(p, '', style)
return

proc PutLink
parameters p(p), URL, txt
    && Put a hyperlink
    x=fpdf_SetTextColor(p, 0,0,255)
    SetStyle(p, 'U', "1")
    x=fpdf_Write(p, 5, txt, URL)
    SetStyle(p, 'U', "0")
    x=fpdf_SetTextColor(p, 0)
return

```