

The luamplib package

Hans Hagen, Taco Hoekwater, Elie Roux, Philipp Gesang and Kim Dohyun
Maintainer: LuaLaTeX Maintainers — Support: <lualatex-dev@tug.org>

2013/12/11 v2.03

Abstract

Package to have metapost code typeset directly in a document with LuaTeX.

1 Documentation

This package aims at providing a simple way to typeset directly metapost code in a document with LuaTeX. LuaTeX is built with the lua mplib library, that runs metapost code. This package is basically a wrapper (in Lua) for the Lua mplib functions and some TeX functions to have the output of the mplib functions in the pdf.

The package needs to be in PDF mode in order to output something, as PDF specials are not supported by the DVI format and tools.

The metapost figures are put in a TeX hbox with dimensions adjusted to the metapost code.

The code is from the luatex-mplib.lua and luatex-mplib.tex files from ConTeXt, they have been adapted to L^AT_EX and Plain by Elie Roux and Philipp Gesang, new functionalities have been added by Kim Dohyun. The changes are:

- a L^AT_EX environment
- all TeX macros start by mplib
- use of luatexbase for errors, warnings and declaration
- possibility to use btex ... etex to typeset TeX. texttext() is a more versatile macro equivalent to TEX() from TEX.mp. TEX() is also allowed unless TEX.mp is loaded, which should be always avoided.

Using this package is easy: in Plain, type your metapost code between the macros mplibcode and endmplibcode, and in L^AT_EX in the mplibcode environment.

There are (basically) two formats for metapost: *plain* and *mpfun*. By default, the *plain* format is used, but you can set the format to be used by future figures at any time using \mplibsetformat{<format name>}.

2 Implementation

2.1 Lua module

Use the `luamplib` namespace, since `mplib` is for the metapost library itself. ConT_EXt uses metapost.

```
1
2 luamplib          = luamplib or { }
3
```

Identification.

```
4
5 local luamplib    = luamplib
6 luamplib.showlog  = luamplib.showlog or false
7 luamplib.lastlog  = ""
8
9 local err, warn, info, log = luatexbase.provides_module({
10   name           = "luamplib",
11   version        = 2.03,
12   date           = "2013/12/11",
13   description    = "Lua package to typeset Metapost with LuaTeX's MPLib.",
14 })
15
16
```

This module is a stripped down version of libraries that are used by ConT_EXt. Provide a few “shortcuts” expected by the imported code.

```
17
18 local format, abs = string.format, math.abs
19
20 local stringgsub   = string.gsub
21 local stringfind   = string.find
22 local stringmatch  = string.match
23 local stringgmatch = string.gmatch
24 local tableconcat  = table.concat
25 local textsprint   = tex.sprint
26
27 local mplib = require ('mplib')
28 local kpse  = require ('kpse')
29
30 local file = file
31 if not file then
32
```

This is a small trick for L^AT_EX. In L^AT_EX we read the metapost code line by line, but it needs to be passed entirely to `process()`, so we simply add the lines in `data` and at the end we call `process(data)`.

A few helpers, taken from `l-file.lua`.

```
33
```

```

34 file = { }
35
36 function file.replacesuffix(filename, suffix)
37     return (stringgsub(filename,"%.[%a%d]+$","") .. "." .. suffix)
38 end
39
40 function file.stripsuffix(filename)
41     return (stringgsub(filename,"%.[%a%d]+$",""))
42 end
43 end

```

As the finder function for mplib, use the kpse library and make it behave like as if MetaPost was used (or almost, since the engine name is not set this way—not sure if this is a problem).

```

44
45 local mpkpse = kpse.new("luatex", "mpost")
46
47 local function finder(name, mode, ftype)
48     if mode == "w" then
49         return name
50     else
51         return mpkpse.find_file(name, ftype)
52     end
53 end
54 luamplib.finder = finder
55

```

The rest of this module is not documented. More info can be found in the LuaTeX manual, articles in user group journals and the files that ship with ConTeXt.

```

56
57 function luamplib.resetlastlog()
58     luamplib.lastlog = ""
59 end
60

```

Below included is section that defines fallbacks for older versions of mplib.

```

61 local mplibone = tonumber(mplib.version()) <= 1.50
62
63 if mplibone then
64
65     luamplib.make = luamplib.make or function(name, mem_name, dump)
66         local t = os.clock()
67         local mpx = mplib.new {
68             ini_version = true,
69             find_file = luamplib.finder,
70             job_name = file.stripsuffix(name)
71         }
72         mpx:execute(format("input %s ;", name))
73         if dump then
74             mpx:execute("dump ;")
75         end
76     end
77 end

```

```

75         info("format %s made and dumped for %s in %0.3f seconds",mem_name,name,os.clock()-t)
76     else
77         info("%s read in %0.3f seconds",name,os.clock()-t)
78     end
79     return mpx
80 end
81
82 function luamplib.load(name)
83     local mem_name = file.replacesuffix(name,"mem")
84     local mpx = mplib.new {
85         ini_version = false,
86         mem_name = mem_name,
87         find_file = luamplib.finder
88     }
89     if not mpx and type(luamplib.make) == "function" then
90         -- when i have time i'll locate the format and dump
91         mpx = luamplib.make(name,mem_name)
92     end
93     if mpx then
94         info("using format %s",mem_name,false)
95         return mpx, nil
96     else
97         return nil, { status = 99, error = "out of memory or invalid format" }
98     end
99 end
100
101 else
102

```

These are the versions called with sufficiently recent mplib.

```

103
104     local preamble = [[
105         boolean mplib ; mplib := true ;
106         let dump = endinput ;
107         input %s ;
108     ]]
109
110     luamplib.make = luamplib.make or function()
111     end
112
113     function luamplib.load(name)
114         local mpx = mplib.new {
115             ini_version = true,
116             find_file = luamplib.finder,
117         }
118         local result
119         if not mpx then
120             result = { status = 99, error = "out of memory"}
121         else
122             result = mpx:execute(format(preamble, file.replacesuffix(name,"mp")))

```

```

123         end
124         luamplib.reporterror(result)
125         return mpx, result
126     end
127
128 end
129
130 local currentformat = "plain"
131
132 local function setformat (name) --- used in .sty
133     currentformat = name
134 end
135 luamplib.setformat = setformat
136
137
138 luamplib.reporterror = function (result)
139     if not result then
140         err("no result object returned")
141     elseif result.status > 0 then
142         local t, e, l = result.term, result.error, result.log
143         if t then
144             info(t)
145         end
146         if e then
147             err(e)
148         end
149         if not t and not e and l then
150             luamplib.lastlog = luamplib.lastlog .. "\n " .. l
151             log(l)
152         else
153             err("unknown, no error, terminal or log messages")
154         end
155     else
156         return false
157     end
158     return true
159 end
160
161 local function process_indeed (mpx, data)
162     local converted, result = false, {}
163     local mpx = luamplib.load(mpx)
164     if mpx and data then
165         local result = mpx:execute(data)
166         if not result then
167             err("no result object returned")
168         elseif result.status > 0 then
169             err("%s", (result.term or "no-term") .. "\n" .. (result.error or "no-error"))
170         elseif luamplib.showlog then
171             luamplib.lastlog = luamplib.lastlog .. "\n" .. result.term
172             info("%s", result.term or "no-term")

```

```

173     elseif result.fig then
174         converted = luamplib.convert(result)
175     else
176         err("unknown error, maybe no beginfig/endfig")
177     end
178 else
179     err("Mem file unloadable. Maybe generated with a different version of mplib?")
180 end
181 return converted, result
182 end
183 local process = function (data)
184     return process_indeed(currentformat, data)
185 end
186 luamplib.process = process
187
188 local function getobjects(result, figure, f)
189     return figure:objects()
190 end
191
192 local function convert(result, flusher)
193     luamplib.flush(result, flusher)
194     return true -- done
195 end
196 luamplib.convert = convert
197
198 local function pdf_startfigure(n, llx, lly, urx, ury)

```

The following line has been slightly modified by Kim.

```

199     texsprint(format("\mplibstarttoPDF{%f}{%f}{%f}{%f}", llx, lly, urx, ury))
200 end
201
202 local function pdf_stopfigure()
203     texsprint("\mplibstoptoPDF")
204 end
205
206 local function pdf_literalcode(fmt, ...) -- table
207     texsprint(format("\mplibtoPDF{%s}", format(fmt, ...)))
208 end
209 luamplib.pdf_literalcode = pdf_literalcode
210
211 local function pdf_textfigure(font, size, text, width, height, depth)
212     text = text:gsub(".", "\hbox{%1}") -- kerning happens in metapost

```

The following line has been slightly modified by Kim.

```

213     texsprint(format("\mplibtexttext{%s}{%f}{%s}{%s}{%f}", font, size, text, 0, -( 7200/ 7227)/65536*depth))
214 end
215 luamplib.pdf_textfigure = pdf_textfigure
216
217 local bend_tolerance = 131/65536
218

```

```

219 local rx, sx, sy, ry, tx, ty, divider = 1, 0, 0, 1, 0, 0, 1
220
221 local function pen_characteristics(object)
222     local t = mplib.pen_info(object)
223     rx, ry, sx, sy, tx, ty = t.rx, t.ry, t.sx, t.sy, t.tx, t.ty
224     divider = sx*sy - rx*ry
225     return not (sx==1 and rx==0 and ry==0 and sy==1 and tx==0 and ty==0), t.width
226 end
227
228 local function concat(px, py) -- no tx, ty here
229     return (sy*px-ry*py)/divider, (sx*py-rx*px)/divider
230 end
231
232 local function curved(ith,pth)
233     local d = pth.left_x - ith.right_x
234     if abs(ith.right_x - ith.x_coord - d) <= bend_tolerance and abs(pth.x_coord - pth.left_x - d) <= bend_tolerance then
235         d = pth.left_y - ith.right_y
236         if abs(ith.right_y - ith.y_coord - d) <= bend_tolerance and abs(pth.y_coord - pth.left_y - d) <= bend_tolerance then
237             return false
238         end
239     end
240     return true
241 end
242
243 local function flushnormalpath(path,open)
244     local pth, ith
245     for i=1,#path do
246         pth = path[i]
247         if not ith then
248             pdf_literalcode("%f %f m",pth.x_coord,pth.y_coord)
249         elseif curved(ith,pth) then
250             pdf_literalcode("%f %f %f %f %f c",ith.right_x,ith.right_y,pth.left_x,pth.left_y,pth.x_coord,pth.y_coord)
251         else
252             pdf_literalcode("%f %f l",pth.x_coord,pth.y_coord)
253         end
254         ith = pth
255     end
256     if not open then
257         local one = path[1]
258         if curved(pth,one) then
259             pdf_literalcode("%f %f %f %f %f %f c",pth.right_x,pth.right_y,one.left_x,one.left_y,one.x_coord,one.y_coord)
260         else
261             pdf_literalcode("%f %f l",one.x_coord,one.y_coord)
262         end
263     elseif #path == 1 then
264         -- special case .. draw point
265         local one = path[1]
266         pdf_literalcode("%f %f l",one.x_coord,one.y_coord)

```

```

267     end
268     return t
269 end
270
271 local function flushconcatpath(path,open)
272     pdf_literalcode("%f %f %f %f %f %f cm", sx, rx, ry, sy, tx ,ty)
273     local pth, ith
274     for i=1,#path do
275         pth = path[i]
276         if not ith then
277             pdf_literalcode("%f %f m",concat(pth.x_coord,pth.y_coord))
278         elseif curved(ith,pth) then
279             local a, b = concat(ith.right_x,ith.right_y)
280             local c, d = concat(pth.left_x,pth.left_y)
281             pdf_literalcode("%f %f %f %f %f %f c",a,b,c,d,concat(pth.x_coord, pth.y_co-
ord))
282         else
283             pdf_literalcode("%f %f l",concat(pth.x_coord, pth.y_coord))
284         end
285         ith = pth
286     end
287     if not open then
288         local one = path[1]
289         if curved(pth,one) then
290             local a, b = concat(pth.right_x,pth.right_y)
291             local c, d = concat(one.left_x,one.left_y)
292             pdf_literalcode("%f %f %f %f %f %f c",a,b,c,d,concat(one.x_coord, one.y_co-
ord))
293         else
294             pdf_literalcode("%f %f l",concat(one.x_coord,one.y_coord))
295         end
296     elseif #path == 1 then
297         -- special case .. draw point
298         local one = path[1]
299         pdf_literalcode("%f %f l",concat(one.x_coord,one.y_coord))
300     end
301     return t
302 end
303

```

Below code has been contributed by Dohyun Kim. It implements btex / etex functions.

v2.2: `texttext()` is now available, which is equivalent to `TEX()` macro from `TEX.mp`.
`TEX()` is synonym of `texttext()` unless `TEX.mp` is loaded.

```

304
305 local mplibcodepreamblefirst = [[
306 def texttext (expr t) =
307     image( special "%mkTEXbox:"&t; )
308 enddef;
309 let TEX = texttext;
310 ]]

```



```

311
312 local mplibcodepreamblesecond = [[
313 vardef texttext (expr t) =
314     TEXBOX := TEXBOX + 1;
315     image (
316         addto currentpicture doublepath unitsquare
317         xscaled TEXBOXwd[TEXBOX]
318         yscaled (TEXBOXht[TEXBOX] + TEXBOXdp[TEXBOX])
319         shifted (0, -TEXBOXdp[TEXBOX])
320         withprescript "%TEXttxbox:" &
321             decimal TEXBOX & ":" &
322             decimal TEXBOXwd[TEXBOX] & ":" &
323             decimal(TEXBOXht[TEXBOX]+TEXBOXdp[TEXBOX]);
324 )
325 enddef;
326 def TEX (expr t) = texttext (t) enddef;
327 ]]
328
329 local factor = 65536*(7227/7200)
330
331 local function putTEXboxes (object)
332     local n,tw,th = stringmatch(object.prescript,
333         "%TEXttxbox: (%d+): ([%d%.%+%-]+): ([%d%.%+%-]+)")
334     if n and tw and th then
335         local op = object.path
336         local first, second, fourth = op[1], op[2], op[4]
337         local tx, ty = first.x_coord, first.y_coord
338         local sx, sy = (second.x_coord - tx)/tw, (fourth.y_coord - ty)/th
339         local rx, ry = (second.y_coord - ty)/tw, (fourth.x_coord - tx)/th
340         if sx == 0 then sx = 0.00001 end
341         if sy == 0 then sy = 0.00001 end
342         pdf_literalcode("q %f %f %f %f %f cm",sx,rx,ry,sy,tx,ty)
343         texpstr(format("\mplibputtextbox{%i}",n))
344         pdf_literalcode("Q")
345     end
346 end
347
348 local function domakeTEXboxes (data)
349     local num = tex.count[14] -- newbox register
350     if data and data.fig then
351         local figures = data.fig
352         for f=1, #figures do
353             local figure = figures[f]
354             local objects = getobjects(data,figure,f)
355             if objects then
356                 for o=1,#objects do
357                     local object = objects[o]
358                     local prescript = object.prescript
359                     if prescript and stringfind(prescript,"%mkTEXbox:") then
360                         num = num + 1

```

```

361             stringgsub(prescript, "%%mkTEXbox:(.)",
362             function(str)
363                 texpstr(format("\\setbox%i\\hbox{%s}", num, str))
364             end)
365         end
366     end
367 end
368 end
369 end
370 end
371
372 local function makeTEXboxes (data)
373     data = stringgsub(data, "(%A)btex(%A.-%A)etex(%A)",
374     function(pre, texcode, post)
375         texcode = stringgsub(texcode, "%^%s*(.)%s*$", "%1")
376         return pre .. "texttext(\"" .. texcode .. "\")" .. post
377     end)
378     local mpx = luamplib.load(currentformat)
379     if mpx and data then
380         local result = mpx:execute(mplibcodepreamblefirst .. data)
381         domakeTEXboxes(result)
382     end
383     return data
384 end
385
386 luamplib.makeTEXboxes = makeTEXboxes
387
388 local function processwithTEXboxes (data)
389     local num = tex.count[14] -- the same newbox register
390     local prepreamble = "numeric TEXBOX, TEXBOXwd[], TEXBOXht[], TEXBOXdp[];\n" ..
391     "TEXBOX := \"..num..\";\n"
392     while true do
393         num = num + 1
394         local box = tex.box[num]
395         if not box then break end
396         prepreamble = prepreamble ..
397         "TEXBOXwd[\"..num..\"] := \"..box.width/factor ..\";\n" ..
398         "TEXBOXht[\"..num..\"] := \"..box.height/factor ..\";\n" ..
399         "TEXBOXdp[\"..num..\"] := \"..box.depth /factor ..\";\n"
400     end
401     process(prepreamble .. mplibcodepreamblesecond .. data)
402 end
403
404 luamplib.processwithTEXboxes = processwithTEXboxes
405
End of btex – etex patch.
406
407 local function flush(result, flusher)
408     if result then

```

```

409     local figures = result.fig
410     if figures then
411         for f=1, #figures do
412             info("flushing figure %s",f)
413             local figure = figures[f]
414             local objects = getobjects(result,figure,f)
415             local fignum = tonumber(stringmatch(figure:filename(),"([%d]+)$") or figure:charcode() or 0)
416             local miterlimit, linecap, linejoin, dashed = -1, -1, -1, false
417             local bbox = figure:boundingbox()
418             local llx, lly, urx, ury = bbox[1], bbox[2], bbox[3], bbox[4] -- faster than unpack
419             if urx < llx then
420                 -- invalid
421                 pdf_startfigure(fignum,0,0,0,0)
422                 pdf_stopfigure()
423             else
424                 pdf_startfigure(fignum,llx,lly,urx,ury)
425                 pdf_literalcode("q")
426                 if objects then
427                     for o=1,#objects do
428                         local object = objects[o]
429                         local objecttype = object.type

```

Change from ConT_EXt code: the following 3 lines are part of the btex...etex patch.

```

430                     local prescript = object.prescript --- [be]tex patch
431                     if prescript and stringfind(prescript,"%%%TEXTxtbox:") then
432                         putTEXboxes(object)
433                     elseif objecttype == "start_bounds" or objecttype == "stop_bounds" then
434                         -- skip
435                     elseif objecttype == "start_clip" then
436                         pdf_literalcode("q")
437                         flushnormalpath(object.path,t,false)
438                         pdf_literalcode("W n")
439                     elseif objecttype == "stop_clip" then
440                         pdf_literalcode("Q")
441                         miterlimit, linecap, linejoin, dashed = -1, -1, -1, false
442                     elseif objecttype == "special" then
443                         -- not supported
444                     elseif objecttype == "text" then
445                         local ot = object.transform -- 3,4,5,6,1,2
446                         pdf_literalcode("q %f %f %f %f %f %f cm",ot[3],ot[4],ot[5],ot[6],ot[1],ot[2])
447                         pdf_textfigure(object.font,object.dsize,object.text,object.width,object.height)
448                         pdf_literalcode("Q")
449                     else
450                         local cs = object.color

```

Change from ConT_EXt code: the following 5 lines are part of the btex...etex patch.

```

451                     local cr = nil
452                     if cs and #cs > 0 then

```

```

453         local start, stop = luamplib.colorconverter(cs)
454         pdf_literalcode(start)
455         cr = stop
456     end
457     local ml = object.miterlimit
458     if ml and ml ~= miterlimit then
459         miterlimit = ml
460         pdf_literalcode("%f M", ml)
461     end
462     local lj = object.linejoin
463     if lj and lj ~= linejoin then
464         linejoin = lj
465         pdf_literalcode("%i j", lj)
466     end
467     local lc = object.linecap
468     if lc and lc ~= linecap then
469         linecap = lc
470         pdf_literalcode("%i J", lc)
471     end
472     local dl = object.dash
473     if dl then
474         local d = format("[%s] %i d", tableconcat(dl.dashes or {}, " "), dl.offset)
475         if d ~= dashed then
476             dashed = d
477             pdf_literalcode(dashed)
478         end
479     elseif dashed then
480         pdf_literalcode("[] 0 d")
481         dashed = false
482     end
483     local path = object.path
484     local transformed, penwidth = false, 1
485     local open = path and path[1].left_type and path[#path].right_type
486     local pen = object.pen
487     if pen then
488         if pen.type == 'elliptical' then
489             transformed, penwidth = pen.characteris-
tics(object) -- boolean, value
490             pdf_literalcode("%f w", penwidth)
491             if objecttype == 'fill' then
492                 objecttype = 'both'
493             end
494             else -- calculated by mplib itself
495                 objecttype = 'fill'
496             end
497         end
498         if transformed then
499             pdf_literalcode("q")
500         end
501         if path then

```

```

502         if transformed then
503             flushconcatpath(path,open)
504         else
505             flushnormalpath(path,open)
506         end
507         if objecttype == "fill" then
508             pdf_literalcode("h f")
509         elseif objecttype == "outline" then
510             pdf_literalcode((open and "S") or "h S")
511         elseif objecttype == "both" then
512             pdf_literalcode("h B")
513         end
514     end
515     if transformed then
516         pdf_literalcode("Q")
517     end
518     local path = object.htap
519     if path then
520         if transformed then
521             pdf_literalcode("q")
522         end
523         if transformed then
524             flushconcatpath(path,open)
525         else
526             flushnormalpath(path,open)
527         end
528         if objecttype == "fill" then
529             pdf_literalcode("h f")
530         elseif objecttype == "outline" then
531             pdf_literalcode((open and "S") or "h S")
532         elseif objecttype == "both" then
533             pdf_literalcode("h B")
534         end
535         if transformed then
536             pdf_literalcode("Q")
537         end
538     end
539     if cr then
540         pdf_literalcode(cr)
541     end
542 end
543     end
544 end
545 pdf_literalcode("Q")
546 pdf_stopfigure()
547 end
548     end
549 end
550 end
551 end

```

```

552 luamplib.flush = flush
553
554 local function colorconverter(cr)
555     local n = #cr
556     if n == 4 then
557         local c, m, y, k = cr[1], cr[2], cr[3], cr[4]
558         return format("%.3f %.3f %.3f %.3f k %.3f %.3f %.3f %.3f K", c, m, y, k, c, m, y, k), "0 g 0 G"
559     elseif n == 3 then
560         local r, g, b = cr[1], cr[2], cr[3]
561         return format("%.3f %.3f %.3f rg %.3f %.3f %.3f RG", r, g, b, r, g, b), "0 g 0 G"
562     else
563         local s = cr[1]
564         return format("%.3f g %.3f G", s, s), "0 g 0 G"
565     end
566 end
567 luamplib.colorconverter = colorconverter

```

2.2 T_EX package

```

568 <*package>

```

First we need to load some packages.

```

569 \bgroup\expandafter\expandafter\expandafter\egroup
570 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
571     \input luatexbase-modutils.sty
572 \else
573     \NeedsTeXFormat{LaTeX2e}
574     \ProvidesPackage{luamplib}
575     [2013/12/11 v2.03 mplib package for LuaTeX]
576     \RequirePackage{luatexbase-modutils}
577     \RequirePackage{pdftexcmds}
578 \fi

```

Loading of lua code.

```

579 \RequireLuaModule{luamplib}

```

Set the format for metapost.

```

580 \def\mplibsetformat#1{%
581     \directlua{luamplib.setformat("\luatexluaescapestring{#1}")}}

```

MPLib only works in PDF mode, we don't do anything if we are in DVI mode, and we output a warning.

```

582 \ifnum\pdfoutput>0
583     \let\mplibtoPDF\pdfliteral
584 \else
585     %\def\MPLIBtoPDF#1{\special{pdf:literal direct #1}} % not ok yet
586     \def\mplibtoPDF#1{}
587     \expandafter\ifx\csname PackageWarning\endcsname\relax
588         \write16{}
589         \write16{Warning: MPLib only works in PDF mode, no figure will be output.}
590         \write16{}

```



```

635 }%
636 \directlua{
637   luamplib.processwithTEXboxes(luamplib.tempdata)
638 }%
639 \end{mplibcode}%
640 \else
641   \toks@\expandafter{\the\toks@\end{#1}}\expandafter\ltxdomplibcode
642 \fi%
643 }
644 \fi

```

We use a dedicated scratchbox.

```

645 \ifx\mplibscratchbox\undefined \newbox\mplibscratchbox \fi

```

We encapsulate the literals.

```

646 \def\mplibstarttoPDF#1#2#3#4{%
647   \hbox\bgroup
648   \xdef\MPllx{#1}\xdef\MPlly{#2}%
649   \xdef\MPurx{#3}\xdef\MPury{#4}%
650   \xdef\MPwidth{\the\dimexpr#3bp-#1bp\relax}%
651   \xdef\MPheight{\the\dimexpr#4bp-#2bp\relax}%
652   \parskip0pt%
653   \leftskip0pt%
654   \parindent0pt%
655   \everypar{}%
656   \setbox\mplibscratchbox\vbox\bgroup
657   \noindent
658 }

659 \def\mplibstoptoPDF{%
660   \egroup %
661   \setbox\mplibscratchbox\hbox %
662     {\hskip-\MPllx bp%
663      \raise-\MPlly bp%
664      \box\mplibscratchbox}%
665   \setbox\mplibscratchbox\vbox to \MPheight
666     {\vfill
667      \hsize\MPwidth
668      \wd\mplibscratchbox0pt%
669      \ht\mplibscratchbox0pt%
670      \dp\mplibscratchbox0pt%
671      \box\mplibscratchbox}%
672   \wd\mplibscratchbox\MPwidth
673   \ht\mplibscratchbox\MPheight
674   \box\mplibscratchbox
675   \egroup
676 }

```

Text items have a special handler.

```

677 \def\mplibtexttext#1#2#3#4#5{%
678   \begingroup
679   \setbox\mplibscratchbox\hbox

```



```

680     {\font\temp=#1 at #2bp%
681       \temp
682       #3}%
683 \setbox\mplibscratchbox\hbox
684   {\hskip#4 bp%
685     \raise#5 bp%
686     \box\mplibscratchbox}%
687 \wd\mplibscratchbox0pt%
688 \ht\mplibscratchbox0pt%
689 \dp\mplibscratchbox0pt%
690 \box\mplibscratchbox
691 \endgroup
692 }

    That's all folks!
693 \</package>

```

3 The GNU GPL License v2

The GPL requires the complete license text to be distributed along with the code. I recommend the canonical source, instead: <http://www.gnu.org/licenses/old-licenses/gpl-2.0.html>. But if you insist on an included copy, here it is. You might want to zoom in.

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright © 1989, 1991 Free Software Foundation, Inc.

51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you wish it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original author's reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

1. This License applies to any program or other work which contains a notice placed by the copyright holder stating it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law, that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification.") Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

2. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty, keep intact all the notices that refer to this License and to the absence of any warranty, and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

3. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

(a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

(b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole or no charge to all third parties under the terms of this License.

(c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be

on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

4. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

(a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

(b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

(c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

5. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

6. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program for any work based on the Program, you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

7. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

8. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims. This section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

9. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

10. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

11. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

No Warranty

12. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

13. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR RE-DISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the program's name and a brief idea of what it does.
Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) yyyy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands show w and show c should show the appropriate parts of the General Public License. Of course, the commands you may use may be called something other than show w and show c; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
"Gnomovision" (which makes passes at compilers) written by James
Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

18