

The **aeb_mlink** Package

A member of the AeB Pro family

D. P. Story
 Email: dpstory@uakron.edu

processed July 14, 2020

Contents

1	Introduction	1
2	The aeb-mlink Package	2
3	Package Requirements and Options	2
4	Driver Dependent Code	5
5	The Multi-line Linking Commands	6
6	Macros used by the SOUL Interface	27
7	Index	30
8	Change History	34

1 Introduction

This package creates multiline-links. The package `hyperref` does create links, but generally these links cannot be broken across lines, unless `pdflatex` is used to create a PDF.

This package uses the `QuadPoints` entry in the link annotation to create a bounding region; consequently, this package requires **Acrobat Distiller** to create a PDF. `QuadPoints` is a PDF 1.6 feature, so these multiline links will work in Adobe Reader 7.0 or later. If viewed in a version of Adobe Reader previous to 7.0, the viewer will use the underlying bounding box.

LaTeX package requirements are the `eForms` and `hyperref`. Only the use of `dvips` and `dvipsone` is supported.

The key to creating a multi-line is contained in Table 8.24 of the PDF Reference. The description of `QuadPoints` in the PDF Reference is as follows:

(Optional; PDF 1.6) An array of $8 \times n$ numbers specifying the coordinates of n quadrilaterals in default user space that comprise the region in which the link should be activated. The coordinates for each quadrilateral are given in the order

$$x_1 \ y_1 \ x_2 \ y_2 \ x_3 \ y_3 \ x_4 \ y_4$$

specifying the four vertices of the quadrilateral in counterclockwise order. For orientation purposes, such as when applying an underline border style, the bottom of a quadrilateral is the line formed by (x_1, y_1) and (x_2, y_2) . If this entry is not present or the viewer application does not recognize it, the region specified by the `Rect` entry should be used. `QuadPoints` should be ignored if any coordinate in the array lies outside the region specified by `Rect`.

2 The `aeb-mlink` Package

The `aeb_mlink` package is listed on CTAN as `aeb-mlink`; there was, in fact, no `aeb-mlink`. Here, we provide a ‘dummy’ package by that name which passes everything on to `aeb_mlink`.

```

1 % Begin Alt pkg
2 <*altpkgname>
3 \NeedsTeXFormat{LaTeX2e}
4 \RequirePackage{xkeyval}
5 \ProvidesPackage{aeb-mlink}
6 [2018/04/26 v1.0 AeB MLink Alt-name (dps)]
7 \DeclareOptionX*{\PassOptionsToPackage{\CurrentOption}{aeb_mlink}}
8 \ProcessOptionsX
9 \RequirePackage{aeb_mlink}[2018/08/18]
10 </altpkgname>
```

3 Package Requirements and Options

After having established the alternate of this package, we now work on the package itself.

```

11 % Begin Package
12 <*package>
13 \RequirePackage{xkeyval}
14 \RequirePackage{ifpdf}[2006/02/20]
15 \RequirePackage{ifxetex}[2006/08/21]
```

(2020/07/12) We test for non-pdfmark drivers, if present, we make minimal package definitions, define all relevant commands to display their `\text{}` argument. In this way, `pdflatex`, `lualatex`, and `xelatex` can be used to preview the document, perhaps viewing the results in SumatraPDF.

```

16 \ifpdf
17   \let\ML@action\endinput
18 \else
```

```

19  \ifxetex
20    \let\ML@action\endinput
21  \else
22    \let\ML@action\relax
23  \fi
24 \fi
25 \ifx\ML@action\endinput

```

Begin the minimal version of the package. Designed for when a non-pdfmark driver is used: `latex -> dvips -> (distiller | ps2pdf)`

```

26 \RequirePackage{hyperref}
27 \% \RequirePackage{refcount}
28 \RequirePackage{eforms}[2018/08/16]

```

Make all commands of this package to do nothing other then to reproduce their `\text` argument.

```

29 \@ifundefined{mlhypertext}{\newcommand{\renewcommand}%
30 \mlhypertext[2] []{#2}}
31 \newcommand{\mlhyperlink[3] []{#3}}
32 \newcommand{\mlhyperref[3] []{#3}}
33 \newcommand{\mlNameref[2] []{#2}}
34 \newcommand{\mlnameref[2] []{#2}}
35 \newcommand{\mlhref[3] []{#3}}
36 \newcommand{\mlurl[2] []{\expandafter\Hurl\expandafter{#2}}}
37 \let\mlMarksOn\relax
38 \let\mlMarksOff\relax
39 \let\turnSyllbCntOn\relax
40 \let\turnSyllbCntOff\relax
41 \def\mlcs#1{\texttt{@backslashchar#1}}
42 \def\mlMaxNSylls{30}
43 \PackageWarningNoLine{aeb_mlink}
44 {PDF creation requires Adobe Distiller.\MessageBreak
45 Workflow is latex > dvips > distiller; otherwise,\MessageBreak
46 this package does nothing}
47 \fi
48 \ML@action % \endinput or \relax

```

Begin the pdfmark version of the package

Set the driver for dvips

```

49 \newif\if@ml@dvips \!@ml@dvipstrue
50 \def\mlcsarg#1#2{\expandafter#1\csname#2\endcsname}

```

dvipsone Set the driver for dvipsone

```

51 \DeclareOptionX{dvipsone}{\def\eq@drivernum{0}\!@ml@dvipsfalse
52 \PassOptionsToPackage{dvipsone}{eforms}
53 \PassOptionsToPackage{dvipsone}{hyperref}
54 }

```

dvips Set the driver for dvips

```

55 \DeclareOptionX{dvips}{\def\eq@drivernum{0}\@ml@dvipstrue
56   \PassOptionsToPackage{dvips}{eforms}
57   \PassOptionsToPackage{dvips}{hyperref}
58 }

url0pts The options of the url package may be passed through the value of this key; for example, url0pts={hyphens}.
59 \define@key{aeb_mlink.sty}{url0pts}[]{\def\url@0pts{[#1]}}
60 \let\url@0pts\empty
Undefined options are passed to eforms.
61 \DeclareOptionX*{\PassOptionsToPackage{\CurrentOption}{eforms}}

dblevel Sets the debug level.
62 \ifundefined{mldblevel}{\newcount\mldblevel\mldblevel=0 }{}
63 \define@key{aeb_mlink.sty}{dblevel}[0]{\mldblevel=#1 }
Package error message when not dvips (or dvipsone)
64 \def\ml@err@msg{This package requires the driver dvips and\MessageBreak
65   Adobe Distiller as the PDF creator}
(2020/01/06) Conform to the new web.cfg format.
66 \let\bWebCustomize\endinput
67 \let\eWebCustomize\relax
68 \ifpdf\PackageError{aeb_mlink}{\ml@err@msg}\else
69   \ifxetex\PackageError{aeb_mlink}{\ml@err@msg}\else
70     \let\ExecuteOptions@SAVE\ExecuteOptions
71     \let\ExecuteOptions\ExecuteOptionsX
72     \InputIfFileExists{web.cfg}{}%
73     {\ifundefined{l@tex@@@driver}{\ExecuteOptionsX{dvips}}%
74      {\ExecuteOptionsX{dvipsone}}}%
75     \let\ExecuteOptions\ExecuteOptions@SAVE
76 \fi\fi
(2020/01/06) Now require url package and pass options to url through url0pts.
77 \ProcessOptionsX
78 \expandafter\RequirePackage\url@0pts{url}
We require hyperref, eforms and soul. For eforms, a recent version is needed,
2008/03/14 or later.
79 \RequirePackage{hyperref}
80 \RequirePackage{refcount}

Beginning with the version of eforms dated 2018/03/22 or later, there are several
link options that are defined. These are \mlfix, \mlstrut, \mlcrackat and
\mlhyph.
81 \RequirePackage{eforms}[2018/08/16]
82 \RequirePackage{soul}

```

4 Driver Dependent Code

Driver dependent definitions for dvipsone and dvips.

```

83 \def\ps@mark{[\space]
84 \ifnum\mldblevel>0
85   \def\mlpgMsg{(\string\n Beginning of page: ) pf
86     PhysicalPage 20 string cvs
87     pf(\string\n)pf}\else
88   \def\mlpgMsg{}\fi
89 \def\pgmonitoring{\if@ml@dvips
90   dup /PhysicalPage exch 1 add def
91   /PhysicalPage PhysicalPage def^J\fi
92   \mlpgMsg
93 }
```

Redefining `\mllnkcontainer`, defined in eforms dated 2018/03/14 or later. If `\ifoldstylequads` is true, we do not set `/Rect` to the minimal rectangle, rather it is set to the region defining the whole page. This was the default rectangle in the past.

```

94 \def\smallRectTF{\ifoldstylequads false\else
95   \iffixmlinks true\else false\fi\fi\space
96   \ifSmallRect true\else false\fi\space and}
97 \def\ml@nnotName{mLink} % dps
98 \def\mllnkcontainer#1{bCreateLink { xoMsgB {
99   \smallRectTF\space mlRectFix^J%
100  #1}if}{(\ml@nnotName\the\aeB@mLinkCnt) mlIsBldMsg}ifelse}
101 \pboxRect is defined in eforms
102 \def\pboxRect{mlRect }
103 \if@ml@dvips
```

dvips driver: Code for the dvips driver. This next `\special` defines some standard conversion formulas, `TeX` to PDF and PDF to `TeX` for dvips.

```

103 \def\mlDict{SDict}
104 \special{!userdict begin
105   /TeXtoPDF {65536 div DVImag mul} def      % sp to pts
106   /PDFtoDvips {72.27 div Resolution mul} def    % points to dots
107   /PDFtoVDvips {72.27 div VResolution mul} def    % points to dots
108   /DvipstoPDF {72.27 mul Resolution div} def      % dots to points
109   /HTeXtoDvips {TeXtoPDF PDFtoDvips} def        % sp to dots
110   /VTeXtoDvips {TeXtoPDF PDFtoVDvips} def^J%      % sp to dots
111   /PhysicalPage 0 def^J%
112   /PageHeight {vsize} def^J%
113   /PDFtoTeX {PDFtoDvips} def^J%
114   /pf{print flush}def^J%
115   /bop-hook{ \pgmonitoring\space } def
116 end}
```

This command calculates the `\QuadPoints` array when we are using the dvips driver.

```
117 \def\setQuadBox{%
```

```

118      currentpoint DvipstoPDF \aeb@bbox@dp\space TeXtoPDF add
119      neg vsize add 72 sub                                % y1
120      exch DvipstoPDF 72 add exch                      % x1
121      2 copy exch \aeb@bbox@wd\space TeXtoPDF add exch    % x2
122      2 copy \aeb@bbox@ht\space TeXtoPDF add             % y3
123      2 copy exch \aeb@bbox@wd\space TeXtoPDF sub exch    % x4
124  }

```

For the bounding rectangle, we just enclose the entire page. This simplifies things greatly.

```

125  \def\par@@Rect
126  {%
127      72 neg PDFtoDvips vsize 72 sub PDFtoVDvips
128      hsize 72 sub PDFtoDvips 72 neg PDFtoVDvips
129  }

```

dvipsone driver: Code for the dvipsone driver This next `\special` defines some standard conversion formulas, \TeX to PDF and PDF to \TeX in the YandY \TeX System.

```

130 \else
131  \def\mlDict{dvidict}
132  \special{!{/TeXtoPDF {65536 div mag 1000 div mul} def
133      /PDFtoTeX {65536 mul mag 1000 div div} def^~J%
134      /pf{print flush}def^~J%
135      /bphook{ \pgmonitoring\space } def^~J%
136  }

```

This command calculates the `\QuadPoints` array when we are using the dvipsone driver.

```

137  \def\setQuadBox{%
138      currentpoint \aeb@bbox@dp\space add TeXtoPDF
139      neg PageHeight add 72 sub                                % y1
140      exch TeXtoPDF 72 add exch                      % x1
141      2 copy exch \aeb@bbox@wd\space TeXtoPDF add exch    % x2
142      2 copy \aeb@bbox@ht\space TeXtoPDF add             % y3
143      2 copy exch \aeb@bbox@wd\space TeXtoPDF sub exch    % x4
144  }

```

For the bounding rectangle, we just enclose the entire page. This simplifies things greatly.

```

145  \def\par@@Rect
146  {%
147      72 neg PDFtoTeX PageHeight 72 sub PDFtoTeX
148      PageWidth 72 sub PDFtoTeX 72 neg PDFtoTeX
149  }
150 \fi

```

5 The Multi-line Linking Commands

We use a box, and two counters for this package.

```

151 \newbox\aebo@bbox
152 \newcount\aebo@arrayIndx \aebo@arrayIndx=0
153 \newcount\aebo@mLinkCnt \aebo@mLinkCnt=0
154 \newcount\syllableCnt \syllableCnt=0
155 \newif\ifmllinktotalchanged\mllinktotalchangedfalse

When \ifSmallRect is true (the default), the smallest possible Rect is constructed
to enclose the text; obeyed only when \iffixxmlinks (defined in eforms is true.

156 \newif\ifSmallRect \SmallRecttrue
157 \AtEndDocument{\wrtmlinktot@l\ckchngmlinktot@l\wrt@linksnotformed}
158 \def\wrt@linksnotformed{\iflinknotformed
159   \PackageWarningNoLine{aebo_mlink}{Some link calculations are not
160   complete.\MessageBreak
161   DO NOT CONVERT TO PDF at this time. Compile at \MessageBreak
162   least twice more}\fi}
163 \def\wrtmlinktot@l{\immediate\write\auxout{\string\gdef
164   \string\mlinkstotal{\the\aebo@mLinkCnt}}}
165 \def\ckchngmlinktot@l{\@ifundefined{mlinkstotal}{}{
166   {\ml@mlinktot@l@changed}}
167 \def\ml@mlinktot@l@changed{%
168   \ifnum\mlinkstotal=\the\aebo@mLinkCnt\relax\else
169     \PackageWarningNoLine{aebo_mlink}{The number of links has
170     changed. Compile again\MessageBreak until this message
171     does not appear}\immediate
172   \write\auxout{\string\mllinktotalchangedtrue}\fi
173 }
174 \def\ml@mllinktotalchanged{\ifmllinktotalchanged
175   \PackageWarningNoLine{aebo_mlink}
176   {The number of links has changed, continue\MessageBreak
177   to compile}\fi}
178 \AtBeginDocument{\ml@mllinktotalchanged}
179 \def\CurrentBorderColor{@linkbordercolor}
180 \def\ml@nocolorHighlight{I}
181 \def\ml@nocolorLineStyle{S}
182 \def\ml@nocolorLineWidth{1}
183 \def\ml@setnocolorDefaults{%
184 \def\ml@nocolor@defaults{\H{\ml@nocolorHighlight}%
185   \S{\ml@nocolorLineStyle}\W{\ml@nocolorLineWidth}%
186   \Color{\CurrentBorderColor}}%
187 }
188 \ifHy@colorlinks
189   \let\ml@nocolor@defaults\empty
190 \else
191   \ml@setnocolorDefaults
192 \fi
193 \def\ml@earlyExecProps#1{%
194   \eq@setWidgetProps\relax{#1}%
195 }

```

The new scheme of fixing up the quad points write information to the AUX file, which then requires multiple compilations to bring that information up to

date in the document. When working on document development, you can declare `\OldStyleBoxesOn` in the preamble to revert to the old style boxes that do not require AUX info.

```

196 \newif\ifoldstylequads \oldstylequadsfalse
197 \def\OldStyleBoxesOn{\mlfixOff\oldstylequadstrue}
198 \def\OldStyleBoxesOff{\oldstylequadsfalse}
199 \onlypreamble\OldStyleBoxesOn
200 \onlypreamble\OldStyleBoxesOff
201 \let\mlh@preambleCmdInsert\relax
202 \def\mlcs#1{\texttt{\{\\backslashchar#1\}}}
203 \bgroup\makeother\%
204 \gdef\CMT#1{ \%space #1}\egroup
205 \def\mldbModeOn{\def\mldb##1##2{##2}}
206 \def\mldbModeOff{\def\mldb##1##2{}}
207 \def\mldb#1#2{\ifnum#1<\mlblevel#2\fi}
208 \def\ml@adj@x{2}\def\ml@adj@y{2}
209 \def\mlMaxNSylls{30}
210 % usage \mlcrackinsat{\removelastspace}
211 \def\removelastspace{\hskip-\fontdimen2\font}
212 \AtBeginDvi{\special{!%

```

Begin Postscript code This code is executed as the PDF is created by either Adobe Distiller (preferred) or `ps2pdf`. Information is written to the distiller (`ps2pdf`) log.

A switch to determine if the link should be created or now; it may not be fully formed.

```
213 /bCreateLink true def
```

The length of the arrays we deal with are multiples of eight (8), we use a value of 17 as a way of testing whether an array size has been updated. All arrays are set to 17 in length initially.

```

214 /mlIsBld 17 def^^J%
215 /mlIsBldMsg {^^J%
216   /sName exch def^^J%
217   (\string\n!! )pf
218   sName 20 string cvs pf
219   ( is not completely formed,
220   compile again!!\string\n)pf^^J%
221 } def^^J%
222 /xoMsgB true def^^J%
223 /xoMsg {^^J%
224   /Indx exch def^^J%
225   /sName exch def^^J%
226   /nSyllable Indx 8 div def^^J% dpsa08
227   (!-----%
228   \string\n Warning:\string\n
229   The text of )pf
230   sName 20 string cvs pf

```

```

231  ( has crossed a page boundary from page )pf
232  PhysicalPage 1 sub 10 string cvs pf
233  ( to ) pf PhysicalPage 10 string cvs pf
234  sName 0 1 getinterval (m) eq {
235    (.\string\n Cross page links are not supported by the
236    PDF Specification)pf
237    (.\string\n This link is not constructed,
238    please fix it.\string\n)pf
239    (Break point is after syllable number )pf
240    nSyllable cvi 20 string cvs pf (.\string\n)pf
241    (Use the \string\\mlcrackat{})pf
242    nSyllable cvi 20 string cvs pf
243    {} option with this link.\string\n)pf
244  }{
245    (.\string\n Cross page annotations are not supported by the
246    PDF Specification)pf
247    (.\string\n This annotation is not constructed,
248    please fix it.\string\n)pf
249    (Break point is after syllable number )pf
250    nSyllable cvi 20 string cvs pf (.\string\n)pf
251    (Use the mlcrackat=)pf
252    nSyllable cvi 20 string cvs pf
253    ( option with this annotation.\string\n)pf
254  } ifelse
255  (!-----%  

256  \string\n)pf^^J%
257 } def^^J%

```

quadpointsfixup is the major procedure for combining all ‘rectangles’ that are on the same line.

```

258 /quadpointsfixup {^^J%
259  /ary exch def^^J%
260  /quadL exch def^^J%
261  /sName exch def^^J%
262 \mldb0{(Processing )pf sName pf (: OK\string\n)pf^^J}%
263 %\mldb0{lnkCnt 20 string cvs pf (: OK\string\n) pf^^J}%
264 quadL 0 eq {
265 (Problems with this link, length=0,
266 will skip the creation of this link)pf^^J%
267 }{

```

Begin by defining some variables needed for this operation.

```
268 /gOffset 0 def^^J%
```

gY holds the y-coordinate of the lower-left corner of the first entry of a quad. We use it and others like it to determine at which syllable the line is broken.

```

269 /gY ary\space 1 gOffset add get def^^J%
270 \mldb1{(gY is ) pf gY 20 string cvs print^^J}%
271 \mldb1{flush (\string\n) pf^^J}%
272 /gN 0 def^^J%

```

`gMrk` is an array that will load the offsets into `mLinkFxup(<num>)`. Each offset is the beginning of a line. We initially set the length of the array to 10, though this may not be correct. That is, we assume the hypertext will not exceed 10 lines in length.

```
273 /gMrk 10 array def^^J% limitation
```

The first entry is always 0, because marks the location of the quad corresponding to the beginning of the first syllable.

```
274 gMrk 0 0 put^^J%
```

`gMrkL` is the length of the array, we set it to 1, as we've already inserted the first entry into `gMrkL`.

```
275 /gMrkL 1 gOffset add def^^J%
276 \mldb2{Begin first for\string\n} pf^^J}%
```

for loop Begin a for loop. The purpose of this loop is to search through the quad points of `mLinkFxup(<num>)` and find the offsets into the structure where the line breaks occur.

```
277 0 8 quadL 8 sub {^^J%
```

`gIndx` is the loop counter, which we use below. Since we are dealing with quads, we increment by 8 each time.

```
278 /gIndx exch def^^J%
279 \mldb2{Outside gt if with gIndx=} pf^^J}%
280 \mldb2{gIndx 20 string cvs pf^^J}%
281 \mldb2{(\string\n) pf^^J}%
```

`getEntry` is the y-coordinate of the lower-left corner of the quad we are currently examining. We will compare its value to that of `\gY`.

```
282 /getEntry ary\space 1 gOffset add get def^^J%
283 \mldb2{(getEntry=) pf getEntry 20 string cvs pf^^J}%
284 \mldb2{(\string\n) pf^^J}%
```

Here's where we compare `\gY` with `getEntry`. If they differ by more than 2 points then the line has changed. (We do it this way since these are decimal numbers and they may have been rounded in unpredictable ways.)

```
285 \gY getEntry sub abs 2 gt {^^J%
286 \mldb2{Inside gt if\string\n} pf^^J}%
```

The two entries differ, so a line break must have occurred. We insert value of `gIndx` into `gMrk`. `gIndx` is essentially the offset into `mLinkFxup(<num>)` where the line break occurs.

```
287 gMrk gMrkL gIndx put^^J%
```

Increment `gMrkL` accordingly.

```
288 /gMrkL gMrkL 1 add def^^J%
```

Place the new y-value in `\gY` before we look back and look for another line break.

```
289 /gY getEntry def^^J%
290 \mldb2{(Updating gY to )pf \gY 20 string cvs pf^^J}%
291 \mldb2{(\string\n) pf^^J}%
292 \mldb2{(gMrkL=)pf gMrkL 20 string cvs pf^^J}%
293 \mldb2{(\string\n) pf^^J}%
```

```

end if End of the if
294 } if^^J%
295 /gOffset gOffset 8 add def^^J%
end for End of the for loop
296 } for^^J%
297 \mldb2{(end first for\string\n) pf^^J}%

```

We finished searching for line breaks and are now going to work on combining contiguous quads. Contiguous quads are the ones between the offsets recorded in the `gMrk` array. Now, if there are now line breaks, the length of `gMrk` is one.

`gAry` is a temporary array that holds all the quads *corresponding to one line*. Each syllable generates an quad of length 8. Here, we assume any given line has at most `\mlMaxNSylls` syllables, (currently set to `\mlMaxNSylls`, but may be revised). The array `gAry` is declared inside the next loop, so it is redeclared at each iteration of the loop. If Distiller of ps2pdf fails, it may be due to `\mlMaxNSylls` being too small for some of your sentences; in this case, redefine `\mlMaxNSylls` to a larger value.

```

298 /gAryL 8 \mlMaxNSylls\space mul def^^J% limitation
299 \mldb2{(gAryL=) pf gAryL 20 string cvs pf^^J}%
300 \mldb2{(\string\n)pf^^J}%

```

`gFixup` is an array that will hold the final combined quad points, this array is the one that will be referenced by the `QuadPoints` entry of the link annotation. It's length is set to 8 times the number of lines over which the hypertext is broken (`gMrkL`) One quad for each line, rather than many quads, one for each syllable.

```

301 /gFixup 8 gMrkL mul array def^^J% links
302 /aFixup 8 gMrkL mul array def^^J% text markup annotations
303 /gOffset 0 def^^J%
304 \mldb2{(for loop: gMrkL=)pf gMrkL 20 string cvs pf^^J}%
305 \mldb2{(\string\n) pf^^J}%

```

The last loop. In this loop, for each line of hypertext, we build its contiguous quad and load it into `gFixup`.

```

306 0 1 gMrkL 1 sub {^^J%
307 \mldb2{(After gAry\string\n) pf^^J}%
308 \mldb2{(Top of for loop\string\n) pf^^J}%
309 /gIndx exch def^^J%
310 \mldb2{(gIndx=)pf gIndx 20 string cvs pf^^J}%
311 \mldb2{(\string\n)pf^^J}%
312 \mldb2{(gMrk=)pf gMrk gIndx get 20 string cvs pf^^J}%
313 \mldb2{(\string\n) pf^^J}%
314 \mldb2{(mLinkFxup<num> length = )^^J}%
315 \mldb2{pf ary\space length 20 string cvs^^J}%
316 \mldb2{pf (\string\n) pf^^J}%

```

We need to determine if we are examining the quads for the last line, or not. The calculation of `gCount` is dependent on this.

```

317 gIndx 1 add gMrkL eq {^^J%
318 /gCount ary\space length gMrk gIndx get sub def^^J%

```

```

319 }{^^J%
320 /gCount gMrk gIndx 1 add get gMrk gIndx get sub def^^J%
321 } ifelse^^J%

```

Declare the `gAry` array

```

322 /gAry gAryL array def^^J%
323 \mldb2{(gCount=)pf gCount 20 string cvs pf^^J}%
324 \mldb2{(\string\n)pf^^J}%

```

We want to copy a slice of `mLinkFxup<num>`, we declare the next array of length `gCount` just computed.

```
325 /sliceOfLinkfxup gCount array def^^J%
```

Populate this array with a slice of `mLinkFxup<num>` beginning at `gMrk[gIndx]` and including the subsequent `gCount` entries. array of length `gCount` just computed.

```

326 sliceOfLinkfxup 0 ary gMrk gIndx get^^J%
327   gCount getinterval putinterval^^J%
328     gAry 0 sliceOfLinkfxup putinterval^^J%
329 \mldb1{(Listing elements of gFixup\string\n) pf^^J}%
330 \mldb1{gAry 0 get 20 string cvs pf (\string\n)pf^^J}%
331 \mldb1{gAry 1 get 20 string cvs pf (\string\n)pf^^J}%
332 \mldb1{gAry gCount 1 sub 5 sub get
333   20 string cvs pf (\string\n)pf^^J}%
334 \mldb1{gAry gCount 1 sub 4 sub get
335   20 string cvs pf (\string\n)pf^^J}%
336 \mldb1{gAry gCount 1 sub 3 sub get
337   20 string cvs pf (\string\n)pf^^J}%
338 \mldb1{gAry gCount 1 sub 2 sub get
339   20 string cvs pf (\string\n)pf^^J}%
340 \mldb1{gAry 6 get 20 string cvs pf (\string\n)pf^^J}%
341 \mldb1{gAry 7 get 20 string cvs pf (\string\n)pf^^J}%
342   gFixup gOffset [^^J%
343     gAry 0 get ^^J%                                x1 ll 1
344     gAry 1 get^^J%                                y1 ll 1
345     gAry gCount 1 sub 5 sub get^^J%                x2 lr 2
346     gAry gCount 1 sub 4 sub get^^J%                y2 lr 2
347     gAry gCount 1 sub 3 sub get^^J%                x3 ur 3
348     gAry gCount 1 sub 2 sub get^^J%                y3 ur 3
349     gAry 6 get^^J%                                x4 ul 4
350     gAry 7 get^^J%                                y4 ul 4
351   ] putinterval^^J%

```

When forming quad points for text markup annotations, the PDF reference is not followed. We have to reorder the array `gFixup` to conform to how Acrobat/Reader expect it. Entries 0 and 4 are increased by the same amount that was subtracted out earlier in

```

352   aFixup gOffset [^^J%
353     gAry 6 get gOffset 0 eq {\ml@adj@x\space add}if^^J% x4 ul 4
354     gAry 7 get^^J%                                y4 ul 4
355     gAry gCount 1 sub 3 sub get^^J%                x3 ur 3
356     gAry gCount 1 sub 2 sub get^^J%                y3 ur 3

```

```

357      gAry 0 get gOffset 0 eq {\ml@adj@x\space add}if^^J% x1 ll 1
358      gAry 1 get 1 sub^^J%                                y1 ll 1
359      gAry gCount 1 sub 5 sub get^^J%                  x2 lr 2
360      gAry gCount 1 sub 4 sub get 1 sub^^J%          y2 lr 2
361      ] putinterval^^J%
362      /gOffset gOffset 8 add def^^J%
363  } for^^J%
364 \mldb2{(End of second for\string\n) pf^^J}%
365 } ifelse
366 } def^^J%

```

smallquadpointsfixup fixes up the bounding boxes. Raises their heights by \ml@adj@y (removed), and moves the left boundary \ml@adj@x to the left.

```

367 /smallquadpointsfixup {^^J%
368 /gIndx exch def^^J%
369 /ary exch def^^J%
370 /lnkCnt exch def^^J%
371 /quadL exch def^^J%
372 /gSFup 8 array def^^J%
373   gSFup 0 ary 0 gIndx add get
374   gIndx 0 eq {\ml@adj@x\space sub} if put^^J% x1
375   gSFup 1 ary 1 gIndx add get put^^J%           y1
376   gSFup 2 ary 2 gIndx add get put^^J%           x2
377   gSFup 3 ary 3 gIndx add get put^^J%           y2
378   gSFup 4 ary 4 gIndx add get put^^J%           x3
379   gSFup 5 ary 5 gIndx add get put^^J%
380   gSFup 6 ary 6 gIndx add get
381   gIndx 0 eq {\ml@adj@x\space sub} if put^^J% x4
382   gSFup 7 ary 7 gIndx add get put^^J%
383   ary gIndx gSFup putinterval^^J%
384 } def^^J%

```

The mlRectFix procedure creates the minimum sized rectangle that encloses the text, and we use this as the dimensions of /Rect

```

385 /mlRectFix {^^J%
386 /ifRectFix exch def^^J%
387 ifRectFix {
388   /nL gFixup length 8 sub def^^J% number of lines
389   /xMin gFixup 0 get def^^J%
390   0 8 nL {^^J%
391   /Indx exch def^^J%
392   gFixup Indx get xMin lt {/xMin gFixup Indx get def}if } for^^J%
393   /xMin xMin 2 sub def^^J%
394   /xMax gFixup 2 get def^^J%
395   2 8 nL 2 add {^^J%
396   /Indx exch def^^J%
397   gFixup Indx get xMax gt {/xMax gFixup Indx get def}if } for^^J%
398   /xMax xMax 2 add def^^J%
399   /yMin gFixup 1 get def^^J%
400   1 8 nL 1 add {^^J%

```

```

401 /Indx exch def^^J%
402 gFixup Indx get yMin lt {/yMin gFixup Indx get def}if } for^^J%
403 /yMin yMin 4 sub def^^J%
404 /yMax gFixup 5 get def^^J%
405 5 8 nL 5 add{^^J%
406 /Indx exch def^^J%
407 gFixup Indx get yMax gt {/yMax gFixup Indx get def}if } for^^J%
408 /yMax yMax 2 add def^^J%
409 /mlRect {/Rect [^^J%
410     xMin 72 sub PDFtoTeX^^J%
411     PageHeight 72 sub yMax sub PDFtoTeX^^J%
412     xMax 72 sub PDFtoTeX^^J%
413     PageHeight yMin sub 72 sub PDFtoTeX ]^^J%
414 }def^^J%
415 }{^^J%
416 /mlRect{/Rect [ \par@@Rect ] }def^^J%
417 }ifelse^^J%
418 ifRectFix {
419 \mldbi{(/Rect [)pf^^J%
420 xMin 20 string cvs pf( )pf^^J%
421 yMax 20 string cvs pf( )pf^^J%
422 xMax 20 string cvs pf( )pf^^J%
423 yMin 20 string cvs pf( \string\n)pf^^J%}
424 } if^^J%
425 } def
426 }

```

\mlMarksOn Added tracking marks. Turn them on with \mlMarksOn and off again with \mlMarksOff.

```

427 \newif\ifmlmarks\mlmarksfalse
428 \def\mlMarksOn{\mlmarkstrue}
429 \def\mlMarksOff{\mlmarksfalse}

```

\ml@MrkLnk{\num} The internal command \ml@MrkLnk typesets the tracking mark; it may be redefined. When the link is within a `tabular` environment (and perhaps others), in this case, \baselineskip=0pt. We raise instead the height of the capital letter ‘T’, plus a little.

```

430 \def\MrkLnkLtr[L]
431 \def\ml@MrkLnk#1{\ifmlmarks\bgroup\ifdim\baselineskip=0pt
432   \setbox\z@\hbox{T}\gdef\ml@raiseamt{\ht\z@+.4pt}\else
433   \gdef\ml@raiseamt{.6\baselineskip}\fi\smash{\rlap{\normalfont
434     \normalcolor\bfseries
435     \raisebox{\ml@raiseamt}{\tiny\strut{\MrkLnkLtr#1}}}}\egroup\fi}
436 \newif\iflinknotformed \linknotformedfalse
437 \newif\ifcr@ckit \cr@ckitfalse
438 \def\ml@underlinded{U}

```

\mlhypertext[*opts*]{*text*} This is a general purpose hypertext link. Not only is it a fine stand-alone linking command, but it also serves as a building block to some convenience commands that follow.

The commands takes two arguments, the first an optional one the second one requires.

<opts> (Optional) A standard optional argument for eforms to change the appearance of the link and/or to include actions.

<text> The text to be enclosed by the link.

The most recently, eforms defines \mlhypertext to a warning message. So if \mlhypertext is already defined, we \renewcommand else we \newcommand.

```
439 \ifundefined{\mlhypertext}{\newcommand{\renewcommand}%
440 {\mlhypertext}[2] []{\hglue0pt\begingroup
441 \global\ml@displaytrue
442 \toks@=\expandafter{\#2}%
443 \edef\ml@HytextArg{{\the\toks@}}%
444 \global\aebl@arrayIndx=z@
445 \def\ml@setlink##1{\setLinkPbox{%
446 \QuadPoints{mLink##1}#1}%
447 \expandafter\processAppArgs\set@LinkPboxDefaults
448 \presets{\ml@nocolor@defaults}\S{S}\W{0}#1\end@nil
449 \ifx\eq@S@value\ml@underlined
450 \let\itsunderline\ef@YES\else\let\itsunderline\ef@NO\fi}
```

Now we test whether \eq@mocrackat is empty or not. If not-empty we break this link to two parts.

```
451 \ifx\eq@mignore\ef@YES
452 \global\advance\aebl@mLinkCnt@ne\relax
453 \def\ml@next{\mlhypertext@if{#1}}\else
454 \ifx\eq@mocrackat\empty
```

A ‘normal’ link, continue

```
455 \global\advance\aebl@mLinkCnt@ne\relax
456 \def\ml@next{\mlhypertext@if{#1}}%
457 \else % \eq@mocrackat not \empty
```

Crack it up. We define \ml@next to call \mlhypertext consecutively; the first time we specify \mlignore{0} to indicate this link is the first part, and then \mlignore{1} to indicate the second part. For the first part, we proceed as usual until we get the the syllable number specified by \mlcrackAt, then we continue with the next \mlhypertext command, we remove all content up to the syllable number \mlcrackAt, and typeset the rest; for example,

This is a test sentence that we want to break across pages. (1)

This is a test sentence that we want to break across pages. (2)

The first link produces the typeset material in (1), where grayed-out text are removed; while the second link produces line (2), again, with the grayed-out material removed.

```
458 \def\ml@next{\global\ml@displaytrue\let\ml@space\space
459 \mlhypertext[#1\mlignore{0}]{#2}\eq@mocrackinsat
460 \penalty-100 \cr@ckittrue
```

```

461      \global\ml@displayfalse\let\ml@space\space
462      \mlhypertext[#1\mlignore{1}]{#2}\aftergroup
463      \normalcolor\endgroup}%
464  \fi
465 \fi
466 \ml@next
467 }
468 \def\mlhypertext@#1{%
469   @ifundefined{mLinkLngth}{\the\aebo@mLinkCnt}{\global
470     \linknotformedtrue\def\ml@lngth{17}}
471   {\edef\ml@lngth{\cnameuse{mLinkLngth}{\the\aebo@mLinkCnt}}}\%
472   \ml@start@link{\the\aebo@mLinkCnt}{\ml@lngth}\% Step 1
473   \def\mlh@preambleCmdInsert{%
474     \ml@MrkLnk{\the\aebo@mLinkCnt}\ml@earlyExecProps{#1}}\%
475   \def\mlh@postambleCmd{\endgroup}

Start soul on \mlhypertext (\aebo@mlh)
476   \expandafter\aebo@mlh\ml@HytextArg
477   \ml@finish@link{\the\aebo@mLinkCnt}{\ml@lngth}\%
478   \ml@setlink{\the\aebo@mLinkCnt}\%
479   \ifoldstylequads\else
480     \iffixxmlinks\literalps@out{restore}\fi\fi
481   @ifundefined{mLinkLngth}{\the\aebo@mLinkCnt}{\%
482     \immediate\write\@auxout{\string\mlcsarg
483       \string\gdef{mLinkLngth}{\the\aebo@mLinkCnt}{17}}\%
484   }{\immediate\write\@auxout{\string\mlcsarg
485     \string\gdef{mLinkLngth}{\the\aebo@mLinkCnt}\%
486     {\the\aebo@arrayIndx}}}\endgroup
487 }
488 \def\mlh@setQuadSyllable#1#2#3#4{%
489 % #1 = current array Index
490 % #2 = quad total
491 % #3 = link cnt
492 % #4 = content
493   \setbox\aebo@bbox=\hbox{\ml@strut#4}\%
494   \setbox\aebo@bbox=\hbox{\ml@strut}\%
495   \ifx\itsunderline\ef@YES\@tempdima1bp\relax\else
496     \@tempdima\dp\@tempboxa \ifdim\@tempdima>2bp
497       \advance\@tempdima-2bp\fi
498     \fi
499   \ifx\isstrikeout\ef@YES\advance\@tempdima-2bp\fi
500   \dp\aebo@bbox\@tempdima
501   \@tempdima\ht\@tempboxa \advance\@tempdima\dp\aebo@bbox
502   \advance\@tempdima1bp
503   \ht\aebo@bbox\@tempdima\def\x{\the\count\z@\%
504   \count\z@=\ht\aebo@bbox\xdef\aebo@bbox@ht{\x}\%
505   \count\z@=\wd\aebo@bbox\xdef\aebo@bbox@wd{\x}\%
506   \count\z@=\dp\aebo@bbox\xdef\aebo@bbox@dp{\x}\%
507 }%

```

```

509  \ifoldstylequads
510   \literalps@out{%
511     bCreateLink {^^J%
512       \mlDict\space\mLinkFxup#3\space known {^^J%
513       \ps@mark{mLink#3}
514       \the\aeboarrayIndx\space [\setQuadBox]
515         \space /PUTINTERVAL pdfmark}{^^J%
516       xoMsgB {
517         /xoMsgB false def
518         (\ml@onnotName\the\aeboarrayIndx)
519         #3\space
520         xoMsg % dpsa08
521       } if^^J% xoMsgB
522     } ifelse } if
523   }%
524 \else
525   \literalps@out{%
526     bCreateLink {^^J%
527       \mlDict\space\mLinkFxup#3\space known {^^J%
528       \mLinkFxup#3\space
529       #1\space[\setQuadBox] putinterval^^J%
Initiate fix up of little rectangles
530       #2\space %
531       #3\space
532       \mLinkFxup#3\space
533       #1\space
534       smallquadpointsfixup }{^^J%
535       xoMsgB {
536         /xoMsgB false def
537         (\ml@onnotName#3)
538         #1\space
539         xoMsg % dpsa08
540       } if^^J% xoMsgB
541     } ifelse } if
542   }%
543 \fi
544 \global\advance\aeboarrayIndx8\relax
545 }

```

The next four commands are used internally, though `\aebonameref`, `\labelRef` and `\atPage` are public, and can be used.

```

546 \def\aeboexiii{\expandafter\expandafter\expandafter}
547 \def\aebonameref#1{\@ifundefined{r@#1}{??}
548   {\aeboexiii\@thirdoffive\csname r@#1\endcsname}}
549 \def\labelRef#1{\@ifundefined{r@#1}{Doc-Start}
550   {\aeboexiii\@fourthoffive\csname r@#1\endcsname}}
551 \def\atPage#1{\getrefbykeydefault{#1}{page}{??}}

```

- `\mlhyperlink` These four commands mimic the `hyperref` commands of the same root name. The
`\mlhyperref` commands `\mlhyperlink` and `\mlhyperref` take three parameters (the first one
`\mlnameref`
`\mlNameref`

optional). The optional parameter modifies the appearance of the link, the second is the target/destination of the link, the third is the text the link is wrapped around. In the case of `\mlhyperlink` that target is a defined by `\hypertarget`; for `\mlhyperref` the target is a latex label.

The commands `\mlnameref` and `\mlNameref` take two parameters (the first is optional). As before, the first modifies the appearance of the link, the second is the target, a latex label.

Syntax: `\mlhyperlink[<opts>]{<named-dest>}{<text>}`

```
552 \newcommand\mlhyperlink[3][]{%
553   \mlhypertext[#1\A{/S/GoTo/D (#2)}]{#3}}
```

Syntax: `\mlhyperref[<opts>]{<label>}{<text>}`

```
554 \newcommand\mlhyperref[3][]{%
555   \mlhypertext[#1\A{/S/GoTo/D (\labelRef{#2})}]{#3}}
```

Syntax: `\mlnameref[<opts>]{<label>}`

```
556 \newcommand\mlnameref[2][]{\protected@edef\ml@temp{\aebnameref{#2}}%
557   \def\ml@tempi{\mlhypertext[#1\A{/S/GoTo/D (\labelRef{#2})}]}%
558   \expandafter\ml@tempi\expandafter{\ml@temp}}
```

We use a work around to a `\relax` problem encountered in the `\mlNameRef` command. L^AT_EX inserts a `relax` at the end of label titles, which stops soul. We insert `\let\SOUL@stop\ml@SOUL@stop`, this seems to work, no guarantees.

Syntax: `\mlNameref[<opts>]{<label>}`

```
559 \newcommand\mlNameref[2][]{\let\SOUL@stop\ml@SOUL@stop
560   \protected@edef\ml@temp{\aebnameref{#2} on page~\atPage{#2}}%
561   \def\ml@tempi{\mlhypertext[#1\A{/S/GoTo/D (\labelRef{#2})}]}%
562   \expandafter\ml@tempi\expandafter{\ml@temp}}
```

The next three commands are modifications some low hyperref commands found in the `pdfmark.def` file. Depending on the parsing, `\href` calls one of these three; we intercept them, and insert our own command `\mlhypertext` so the link string gets wrapped around if needed. These are in preparation for the definition of `\mlhref`.

```
563 \def\ml@hyper@linkurl#1#2{\hyper@chars
564   \let\ef@thislinkcolor\@urlcolor
565   \let\CurrentBorderColor\@urlbordercolor
566   \mlhypertext[\presets{\mlhref@args}\A{/S/URI/URI(#2)}]{#1}%
567   \endgroup
568 }
569 \def\ml@hyper@linkfile#1#2#3{%
570   \let\ef@thislinkcolor\@filecolor
571   \let\CurrentBorderColor\@filebordercolor
572   \ifx\@pdfstartview\empty
573     \def\theView{[0 /Fit]}\else
```

```

574     \def\theView{[0 \@pdfstartview]}\fi
575     \@ifundefined{ifHy@pdfnewwindow}
576     {\ifHy@newwindow}{\ifHy@pdfnewwindow}%
577     \def\isWindow{/NewWindow true}\else
578     \let\isWindow\@empty\fi
579     \mlhypertext[\presets{\mlhref@args}\A{/S/GoToR \isWindow
580     /F (#2) /D \ifx\\#3\\\theView\else(#3)\fi}]{#1}%
581     \endgroup
582 }
583 \def\ml@hyper@launch run:#1\\#2#3{%
584   \let\ef@thislinkcolor\@filecolor
585   \let\CurrentBorderColor\@runbordercolor
586   \@ifundefined{ifHy@pdfnewwindow}
587   {\ifHy@newwindow}{\ifHy@pdfnewwindow}%
588   \def\isWindow{/NewWindow true}\else
589   \let\isWindow\@empty\fi
590   \mlhypertext[\presets{\mlhref@args}\A{/S/Launch\isWindow
591     /F (#1) \ifx\\#3\\\else /Win << /P (#3) /F (#1) >> \fi}]{#2}%
592   \endgroup
593 }

```

Below is the code for `\mlhref`. We first let the old commands found in `pdfmark.def` equal to the new versions, then we call `\href` to do all the parsing. Things eventually comes back to the above three commands.

```
594 \let\aebsaved@href\href
```

\mlhref This command is similar to `\href`. This command also takes three arguments, one optional. The first is usual optional argument that allow one to modify the appearance of the link, the second one is the URL that we are linking to, the third is the text that we are wrapping this link around.

Syntax: `\mlhref [<opts>]{<url>}{{<text>}}`

```

595 \newcommand{\mlhref}[1][]{%
596   \begin{group}
597     \def\mlhref@args{#1}%
598     \let\hyper@linkurl\ml@hyper@linkurl
599     \let\hyper@linkfile\ml@hyper@linkfile
600     \let\@hyper@launch\ml@hyper@launch
601     \aebsaved@href
602 }

```

\mlurl The multi-line version of Donald Arseneau's url package. There are problems with this one, will continue to work on it.

The problem is not as "easy" as the previous cases. Arseneau places the URL in math mode and it does not reconstruct (`soul` terminology) as it should. Our solution is to hijack three commands of `soul`, these are `\SOUL@doword`, `\SOUL@analyze`, and `\SOUL@dosyllable`, and modify them to do the work on an URL.

```
603 \newbox\ml@urlbuildi
```

```

604 \newbox\ml@urlbuildii
We modify \SOUL@doword and name it \ml@SOUL@doword.
605 \def\ml@SOUL@doword{%
606   \global\setbox\ml@urlbuildi\hbox{}%
607   \global\setbox\ml@urlbuildii\hbox{}%
608   \edef\x{\the\SOUL@word}%
609   \ifx\x\empty
610   \else
611     \SOUL@buffer={}
612     \setbox\z@\vbox{%
613       \SOUL@tt
614       \hyphenchar\font`-
615       \hfuzz\maxdimen
616       \hbadness\@M
617       \pretolerance\m@ne
618       \tolerance\@M
619       \leftskip\z@
620       \rightskip\z@
621       \hsize1sp
622       \everypar{%
623         \parfillskip\z@\@plus1fil
624         \hyphenpenalty-\@M
625         \noindent
626         \hskip\z@
627         \relax
628       }%
}

```

We don't do the reconstruction, so no need for the message.

```

629   \let\SOUL@errmsg\relax
630 % \let\SOUL@errmsg\SOUL@error
631 \let\-\relax
632 \count@\m@ne

```

Here is the first major change, rather than splitting off to \SOUL@analyze, we go to our modified version, \ml@SOUL@analyze.

```

633 \ml@SOUL@analyze
634 \SOUL@word={}
635 \fi
636 }

```

We modify \SOUL@analyze and name it \ml@SOUL@analyze.

```

637 \def\ml@SOUL@analyze{%
638   \setbox\z@\vbox{%
639     \unvcopy\z@
640     \unskip
641     \unpenalty
642     \global\setbox\@ne=\lastbox}%
643 \ifvoid\@ne
644 \else

```

We encapsulate our changes at this point in \ml@interface@analyze

```

645      \ml@interface@analyze
646      \SOUL@syllgoal=\wd\@ne
647      \advance\count@\@ne

```

We get the tokens recursively, but we jump back to `\ml@SOUL@analyze` not `\SOUL@analyze`.

```

648      \ml@SOUL@analyze
649      \SOUL@syllwidth\z@
650      \SOUL@syllable={}
651      \ifnum\count@>\z@
652          \advance\SOUL@syllgoal-\SOUL@ttwidth

```

At this point, we jump to `\ml@SOUL@dosyllable` rather than `\SOUL@dosyllable`.

```

653      \ml@SOUL@dosyllable
654      \SOUL@gettken{\the\SOUL@lasttoken}{\SOUL@hyphkern}%
655      {\SOUL@sethyphenchar}%
656      \SOUL@everyhyphen
657      \else

```

Use `\ml@SOUL@dosyllable` not `\SOUL@dosyllable`.

```

658          \ml@SOUL@dosyllable
659          \fi
660      \fi
661 }
662 \newif\ifml@display \ml@displaytrue

```

We modify `\SOUL@dosyllable` and name it `\ml@SOUL@dosyllable`.

```

663 \def\ml@SOUL@dosyllable{%
664     \SOUL@gettken
665     \SOUL@eventuallyexhyphen{\the\SOUL@token}%
666     \edef\x{\the\SOUL@token}%
667     \ifx\x\SOUL@hyphenhintM
668         \let\SOUL@n\ml@SOUL@dosyllable
669     \else\ifx\x\SOUL@lowerthanM
670         \SOUL@gettken
671         \SOUL@gettken{\the\SOUL@lasttoken}{\SOUL@charkern}%
672         {\the\SOUL@token}%
673     \SOUL@everylowerthan
674     \SOUL@puttoken
675     \let\SOUL@n\ml@SOUL@dosyllable
676     \else\ifdim\SOUL@syllwidth=\SOUL@syllgoal
677         \SOUL@everysyllable
678         \SOUL@puttoken
679         \let\SOUL@n\relax
680     \else
681         \ifx\x\SOUL@stopM
682             \SOUL@errmsg
683             \global\let\SOUL@errmsg\relax
684             \let\SOUL@n\relax
685     \else
686         \setbox\tw@\hbox{\SOUL@tt\the\SOUL@token}%

```

```

687      \advance\SOUL@syllwidth\wd\tw@
688      \global\SOUL@lasttoken=\SOUL@token
689      \SOUL@gettoken
690      \SOUL@getkern{\the\SOUL@lasttoken}{\SOUL@charkern}
691      {\the\SOUL@token}%
692      \SOUL@puttoken
693      \global\SOUL@token=\SOUL@lasttoken
694      \SOUL@everytoken
695      \edef\x{\SOUL@syllable={\the\SOUL@syllable\the\SOUL@token}}\x

```

Here is the only change, we direct flow back to `\ml@SOUL@dosyllable`

```

696      \let\SOUL@n\ml@SOUL@dosyllable
697      \fi\fi\fi\fi
698      \SOUL@n
699 }

```

`\ml@interface@analyze` We put most of our changes to `\ml@SOUL@analyze` in `\ml@interface@analyze`

```

700 \def\ml@interface@analyze{%
701   \global\advance\syllableCnt\@ne % dpsa11
702   \setbox\@ne\hbox{\unhbox\@ne}%

```

If we say `\mlurl{http://www.math.uakron.edu/~dpstory}`, then the `\box\z@` above contains the following tokens, listed at their breakpoints:

```

http:
\\
www.
math.
uakron.
edu/
~dpstory

```

The idea is to get each of these using `\global\setbox\@ne=\lastbox` (bottom to top) and to build the URL with the quad points calculated. Each new token is added in front of the URL as we build it. Results are held in `\ml@urlbuild`. We insert `\penalty0` to promote a break point between components, as each component is enclosed in an `\hbox` now.

We'll try cracking the url here, if requested.

```
703 \ifx\eq@\mlcrackat\empty
```

Ordinary link, we don't crack it apart, continue with the old code.

```

704 \ml@bld@quadchunks{\the\aebo@mLinkCnt}
705   {\ml@qlngthchunki}{\ml@urlbuildi}%
706 \else
707   \ifnum\syllableCnt=\revCrackAt\relax

```

Everything is in reverse order, this is the last syllable of the first chunk

```
708   \aebo@arrayIdx=\z@
```

```

start first link
709      \ml@start@link{\the\aebo@mLinkCnt}{\ml@qlngthchunki}%
710      \ml@bld@quadchunks{\the\aebo@mLinkCnt}
711      {\ml@qlngthchunki}{\ml@urlbuildi}%
712 \else

    Continue with first link
713      \ifnum\syllableCnt>\revCrackAt\relax
714          \ml@bld@quadchunks{\the\aebo@mLinkCnt}
715          {\ml@qlngthchunki}{\ml@urlbuildi}%
716 \else
717      \ifnum\syllableCnt=\@ne

start second link
718          \aebo@arrayIndx=\z@

Everything is in reverse order, this is the last syllable of the second chunk
719      \ml@start@link{\aebo@mLinkCnt@}{\ml@qlngthchunkii}%
720      \ml@bld@quadchunks{\aebo@mLinkCnt@}
721      {\ml@qlngthchunkii}{\ml@urlbuildii}%
722 \else

continue with second link
723      \ml@bld@quadchunks{\aebo@mLinkCnt@}
724      {\ml@qlngthchunkii}{\ml@urlbuildii}%
725      \fi
726      \fi
727      \fi
728      \fi
729 }

730 \def\ml@bld@quadchunks#1#2#3{%
731 % #1 = link cnt
732 % #2 = chunk size
733 % #3 = \ml@urlbuild to i or ii
734 \cifundefined{mLinkLngth}{\the\aebo@mLinkCnt}
735     {\edef\@indx{\the\aebo@arrayIndx}}
736     {\@tempcnta#2\relax
737         \advance\@tempcnta-8\relax
738         \advance\@tempcnta-\aebo@arrayIndx
739         \def\@indx{\the\@tempcnta}%
740     }%
741     \global\setbox\@tempcntb=\@indx \divide\@tempcntb by 8
742     \advance\@tempcntb by 1
743 \ifx#3\ml@urlbuildi \advance\@tempcntb by \eq@mlcrackat\relax\fi
744     \global\setbox\@tempcntb=\hbox{%
745         \mlh@setQuadSyllable{\@indx}{#2}{#1}{\unhcopy\@ne}%
746         \hbox{\unhcopy\@ne}\relax
747         \ml@typeset@syll{\@tempcntb}\penalty0\unhcopy#3}\hbox
748 }

```

\mlurl After the above preliminaries, we finally define \mlurl.

Syntax: \mlurl[*<opts>*]{*<url>*}

```
749 \newcommand{\mlurl}{\begingroup\@makeother\^\relax\def~{\string~}%
750     \ef@sanitize@toks\mlurl@}
```

After sanitizing, we save the URL (#2) as a macro \mlcurl using the \urldef command, defined in the url package.

```
751 \newcommand{\mlurl@}[2][]{\@ifundefined{ef@thislinkcolor}%
752     {\let\ef@thislinkcolor\normalcolor}\expandafter
753     \def\expandafter{\ef@thislinkcolor@SAVE
754         \expandafter{\ef@thislinkcolor}}%
755     \def\ml@setlink##1{\setLinkPbox{\A{\URI{##2}}}}%
756     \QuadPoints{mLink##1#1}}%
```

We get the link options early to determine if this link is to be underlined.

```
757     \expandafter\processAppArgs\set@LinkPboxDefaults
758     \presets{\ml@nocolor@defaults}\S\$}\W{0}#1\end\@nil
759     \ifx\eq@S@value\ml@underlined
760         \let\itsunderline\ef@YES\else\let\itsunderline\ef@NO\fi
```

\mlcurl is the specified url to create a link around

```
761     \global\aeboarrayIdx=0\relax
762     \global\syllableCnt=0\relax
763     \global\advance\aebo@mLinkCnt\@ne\relax
764     \@ifundefined{mLinkLngh}{\the\aebo@mLinkCnt}{%
765         \global\linknotformedtrue\def\ml@lngh{17}%
766         \def\ml@qlnghchunki{17}%
767         {\@tempcpta@\nameuse{mLinkLngh}\the\aebo@mLinkCnt}\relax
768         \edef\ml@lngh{\the\@tempcpta}\multiply\@tempcpta8\relax
769         \edef\ml@qlnghchunki{\the\@tempcpta}}%
```

We make some calculations in preparation to a link that will be cracked apart.

\mlM@XCNT is the number of syllables of the un-cracked URL.

```
770     \@ifundefined{mLinkSyCnt}{\the\aebo@mLinkCnt}{%
771         {\def\mlM@XCNT{0}\def\ml@lnghchunki{0}\def\ml@lnghchunki{0}}%
772         \def\ml@qlnghchunki{0}\def\ml@qlnghchunki{0}\def\revCrackAt{0}%
773         {\edef\mlM@XCNT{\nameuse{mLinkSyCnt}\the\aebo@mLinkCnt}}}%
```

When \mlM@XCNT is known, and \eq@mlcrackat is known, we can calculate the number of syllables of each chunk of the URL, for the first chunk, it is \eq@mlcrackat, for the second it is \mlM@XCNT - \eq@mlcrackat.

```
774     \ifx\eq@mlcrackat\empty\else
775         \@tempcpta\mlM@XCNT\relax\advance\@tempcpta-\eq@mlcrackat\relax
776         \edef\ml@lnghchunki{\eq@mlcrackat}%
777         \edef\ml@lnghchunki{\the\@tempcpta}%
778         \@tempcpta\ml@lnghchunki\relax\multiply\@tempcpta8\relax
779         \edef\ml@qlnghchunki{\the\@tempcpta}%
780         \@tempcpta\ml@lnghchunki\relax\multiply\@tempcpta8\relax
781         \edef\ml@qlnghchunki{\the\@tempcpta}}%
```

We take these chunks off in reverse order so we measure each from the end of the url. This calculation can be move to earlier code where it is not executed for each syllable (chunk). \revCrackAt is the value of \eq@mlcrackat as measure from the end of the url.

```

782      \@tempcnta\mlM@XCNT\relax
783      \advance\@tempcnta-\eq@mlcrackat\relax
784      \advance\@tempcnta\@ne
785      \edef\revCrackAt{\the\@tempcnta}%
786      \fi}%
787      \@tempcnta\aeB@mLinkCnt\advance\@tempcnta\@ne
788      \edef\aeB@mLinkCnt{\the\@tempcnta}%
789      \urldef\ml@url\nolinkurl{\#2}%
790      \def\SOUL@mlhpreamble{\begingroup
791      \mlh@preambleCmdInsert{\ef@colorthislink}\hyper@chars
792      \let\ef@thislinkcolor\urlcolor
793      \let\CurrentBorderColor\urlbordercolor

```

Within this group, we direct the `soul` package to our customized versions of the commands.

```

794      \let\ml@SOUL@doword@SAVE\SOUL@doword
795      \let\SOUL@doword\ml@SOUL@doword

```

The next several lines are taken from the definition of `\mlhypertext`, the basic command for construction many of the ‘`\ml`’ commands of this package.

```

796      \ifx\eq@mlcrackat\empty
797      \ml@start@link{\the\aeB@mLinkCnt}{\ml@length}\fi
798      \def\mlh@preambleCmdInsert{\ml@MrkLnk{\the\aeB@mLinkCnt}%
799      \ml@earlyExecProps{\#1}}%
800      \def\mlh@postambleCmd{%
801      \expandafter

```

The coloring of the hypertext does not work unless we make the definition global, so we do so and hope this does not mess other things up. Save the incoming link color so we can globally change it. After the link is formed, change it back again.

```

802      \def\expandafter{\ef@thislinkcolor
803      \expandafter{\ef@thislinkcolor}}%

```

Finally, we call `\aeB@mlh` which starts `soul` with `\SOUL@`. This does this analysis, the custom command build the url in `\ml@urlbuild`, which we then unbox.

```

804      \aeB@mlh\ml@url\ef@colorthislink\unhc@py\ml@urlbuildi
805      \expandafter\gdef\expandafter{\ef@thislinkcolor
806      \expandafter{\ef@thislinkcolor@SAVE}}%
807      \immediate\write\auxout{\string\mlcsarg\string
808      \gdef{\mlLinkSyCnt\the\aeB@mLinkCnt}{\the\syllableCnt}}% dpsa11
809      \immediate\write\auxout{\string\mlcsarg
810      \string\gdef{\mlLinkLngth\the\aeB@mLinkCnt}{\the\aeB@arrayIndx}}%
811      \ifx\eq@mlcrackat\empty
812      \ml@finish@link{\the\aeB@mLinkCnt}{\ml@length}%
813      \ml@setlink{\the\aeB@mLinkCnt}%
814      \iifxmlinks\literalps@out{restore}\fi

```

```

815     \else
816         \ml@finish@link{\the\aebo@mLinkCnt}{\ml@qlngthchunki}%
817         \ml@setlink{\the\aebo@mLinkCnt}%
818         \iffixmlinks\literalps@out{restore}\fi
819         \eq@mlcrackinsat\penalty-100
820         \ml@start@link{\aebo@mLinkCnt@}{\ml@qlngthchunki}%
821         \penalty0\@ifundefined{mLinkLngth}{\the\aebo@mLinkCnt}{}%
822         {\ml@MrkLnk{\aebo@mLinkCnt@}}\unhcopy\ml@urlbuildii
823         \ml@finish@link{\aebo@mLinkCnt@}{\ml@qlngthchunki}%
824         \ml@setlink{\aebo@mLinkCnt@}\iffixmlinks
825             \literalps@out{restore}\fi
826         \global\advance\aebo@mLinkCnt@ne
827     \fi\aftergroup\normalcolor\endgroup
828 }
829 \def\ml@start@link#1#2{%
830 % #1=link number
831 % #2 = final quad length
832     \literalps@out{%
833         /xoMsgB true def^J%
834         \ps@mark/_objdef {mLink#1}%
835         /type /array /OBJ pdfmark^J%
836         \ifoldstylequads
837         /mLinkFxup#1\space0 array def
838     \else
839         /bCreateLink mLIIsBld #2\space eq not def^J%
840         bCreateLink{ /mLinkFxup#1\space
841             #2\space array def }if
842     \fi
843 }%
844 }

```

Make gFixup into a macro, so other packages (aeb_mlink can intercept and insert its own procedure here.

```

845 \def\FixupProc{gFixup}
846 \def\ml@finish@link#1#2{%
847 % #1=link number
848 % #2 = final quad length
849     \ifoldstylequads\else
850         \ifnum\aebo@arrayIdx=0\relax
851             \PackageWarning{aeb_mlink}{%
852                 Problem with mLink\the\aebo@mLinkCnt, Will skip the \MessageBreak
853                 creation of this link}\fi
854     \literalps@out{%

```

If \iffixmlinks is true, we fix the links, otherwise, we no not.

```

855     \iffixmlinks
856         \ifnum\aebo@arrayIdx>0
857             save^J%
858             bCreateLink {^J%
859             \mlDict\space/mLinkFxup#1\space known {^J%
860             (\ml@nnotName#1)

```

```

861      #2\space
862      mLinkFxup#1\space
863      quadpointsfixup^^J%
864      \ps@mark{mLink#1} 0 \FixupProc\space
865      /PUTINTERVAL pdfmark^^J%
866      }if }if
867      \fi
868  \else
(2018/04/20) Added bCreateLink, etc., to protect against distiller/ps2pdf from
crashing when \mlfix{n}.
869  bCreateLink {^^J%
870  \mlDict\space/mLinkFxup#1\space known {^^J%
871  \ps@mark{mLink#1} 0 mLinkFxup#1
872  /PUTINTERVAL pdfmark^^J%
873  }if }if
874  \fi}%
875 \fi}

```

6 Macros used by the SOUL Interface

```

876 \ifHy@colorlinks
877   \def\ef@colorthislink{\color{\ef@thislinkcolor}}
878 \else
879   \let\ef@colorthislink\relax
880 \fi

```

I've inserted \let\protect\@empty to make `\lnameref` and `\lNameref` work.

```

881 \def\ml@SOUL@stop{\relax}
882 \def\SOUL@mlhpreamble{\begingroup

```

(2011/12/27) Originally, I had \let\protect\@empty here, but removing this seems to do no harm, so, we'll go for it.

```

883   \mlh@preambleCmdInsert\ef@colorthislink}
884 \def\SOUL@mlheversyllable{%
885   \global\advance\syllableCnt@ne
886   \ifx\eq@\mlcrackat\@empty
887     \expandafter\SOUL@mlheversyllable@i
888   \else
889     \expandafter\SOUL@mleversyllable@ii
890   \fi
891 }
892 \let\eq@\mlhyph\@empty
893 \def\ml@typset@syl#1{\raisebox{\ml@raiseamt}{%
894   {\smash{\normalfont\normalcolor\tiny\strut\llap{\the#1}}}}}

```

<code>\turnSyllbCntOn</code>	Turns on the counting of the syllables, while <code>\turnSyllbCntOff</code> counting marks off.
<code>\turnSyllbCntOff</code>	
	892 \def\turnSyllbCntOn{\mlMarksOn\let\ml@typeset@@syl\ml@typset@syl}
	893 \def\turnSyllbCntOff{\let\ml@typeset@@syl\@gobble}
	894 \turnSyllbCntOff

```

898 \def\SOUL@mlheversyllable@i{%
899   \mlh@setQuadSyllable{\the\aeB@arrayIdx}{\ml@lngth}%
900   {\the\aeB@mLinkCnt}{\the\SOUL@syllable\eq@mlhyph}%
901   \the\SOUL@syllable \% \SOUL@setkern\SOUL@charkern
902   \ml@typeset@@syl{\syllableCnt}\eq@mlhyph
903 }%
904 \def\SOUL@mleversyllable@ii{%
905   \ifnum\eq@mlchunk=0\relax
906     \ifnum\syllableCnt>\eq@mlcrackat\relax
907       \global\ml@displayfalse
908     \else
909       \global\ml@displaytrue
910       \ifnum\syllableCnt=\eq@mlcrackat\relax
911         \let\eq@mlhyph\eq@mlhyph
912         \global\let\ml@space\relax
913       \else
914         \let\eq@mlhyph\empty
915         \global\let\ml@space\space
916       \fi
917       \SOUL@mlheversyllable@i
918     \fi
919   \else
920     \ifnum\syllableCnt>\eq@mlcrackat\relax
921       \global\ml@displaytrue
922       \SOUL@mlheversyllable@i
923     \else
924       \global\ml@displayfalse
925     \fi
926   \fi
927   \ml@dynamicsetup
928 }%
929 %\def\SOUL@mlheverspace#1{#1\space\hskip\spaceskip}
930 \let\ml@space\space
931 \def\SOUL@mlheverspace#1{%
932   #1\ml@space\global\let\ml@space\space
933 }%
934 \def\SOUL@mlheveryhyphen{%
935   \discretionary{%
936     \unkern
937     \SOUL@setkern\SOUL@hyphkern
938     \SOUL@sethyphenchar
939   }{}{%
940 }%
941 \def\SOUL@mlheveryexhyphen#1{\global\advance\syllableCnt@ne
942   \mlh@setQuadSyllable{\the\aeB@arrayIdx}{\ml@lngth}%
943   {\the\aeB@mLinkCnt}{\SOUL@setkern\SOUL@hyphkern#1}%
944   \SOUL@setkern\SOUL@hyphkern\hbox{#1}%
945   \discretionary{}{}{%
946     \SOUL@setkern\SOUL@charkern
947   }%

```

```
948 }
949 \def\mlh@postamble{\relax}
950 \def\ml@dynamicsetup{\dpsaug16
951   \ifml@display
952     \global\let\SOUL@everyspace\SOUL@mlheveryspace
953     \global\let\SOUL@everyexhyphen\SOUL@mlheveryexhyphen
954   \else
955     \gdef\SOUL@everyspace##1{}%
956     \gdef\SOUL@everyexhyphen##1{}%
957   \fi
958 }
959 \def\SOUL@mlhpostamble{\mlh@postamble}
960 \def\SOUL@mlhsetup{\SOUL@setup
961   \let\SOUL@preamble\SOUL@mlhpreamble
962   \let\SOUL@everysyllable\SOUL@mlheverysyllable
963   \ml@dynamicsetup
964 %   \let\SOUL@everyspace\SOUL@mlheveryspace
965   \let\SOUL@everyhyphen\SOUL@mlheveryhyphen
966 %   \let\SOUL@everyexhyphen\SOUL@mlheveryexhyphen
967   \def\SOUL@postamble{\SOUL@mlhpostamble}%
968 }
969 \DeclareRobustCommand*\aeb@mlh{\syllableCnt=0
970   \SOUL@mlhsetup\SOUL@}
971 % End of Package
972 </package>
```

7 Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Symbols	
\%	203
\@auxout	163, 172, 482, 484, 807, 809
\@backslashchar	41, 202
\@filebordercolor	571
\@filecolor	570, 584
\@fourthoffive	550
\@hyper@launch	600
\@indx	735, 739, 741, 745
\@linkbordercolor	179
\@makeother	203, 749
\@ml@dvipsfalse	51
\@ml@dvipstrue	49, 55
\@onlypreamble	199, 200
\@pdfstartview	572, 574
\@plus	623
\@runbordercolor	585
\@tempboxa	495, 497, 502
\@thirdoffive	548
\@urlbordercolor	565, 793
\@urlcolor	564, 792
\`	749
A	
\A	553, 555, 557, 561, 566, 579, 590, 755
\aeb@arrayIdx	152, 444, 486, 514, 518, 544, 708, 718, 735, 738, 761, 810, 850, 856, 899, 942
\aeb@bbox	151, 493, 501, 502, 504–507
\aeb@bbox@dp	118, 138, 507
\aeb@bbox@ht	122, 142, 505
\aeb@bbox@wd	121, 123, 141, 143, 506
\aeb@exiii	546, 548, 550
\aeb@mjh	476, 804, 969
\aeb@mLinkCnt	100, 153, 164, 168, 452, 455, 469, 471, 472, 474, 477, 478, 481, 483, 485, 704, 709, 710, 714, 734, 763, 764, 767, 770, 773, 787, 797, 798, 808, 810, 812, 813, 816, 817, 821, 826, 852, 900, 943
\aeb@mLinkCnt@	719, 720, 723, 788, 820, 822–824
\aeb@saved@href	594, 601
\aebnameref	547, 556, 560
\aftergroup	462, 827
\AtBeginDocument	178
\AtBeginDvi	212
\AtEndDocument	157
\atPage	551, 560
B	
\baselineskip	431, 433
\bWebCustomize	66
C	
\ckchngmlinktot@l	157, 165
\CMT	204
\Color	186
\color	877
\cr@ckitfalse	437
\cr@ckittrue	460
\CurrentBorderColor	179, 186, 565, 571, 585, 793
\CurrentOption	7, 61
D	
dblevel (option)	4
\DeclareOptionX	7, 51, 55, 61
\DeclareRobustCommand	969
\discretionary	935, 945
\divide	741
dvips (option)	3
dvipson (option)	3
E	
\ef@colorthislink	791, 804, 877, 879, 883
\ef@NO	450, 760
\ef@sanitize@toks	750
\ef@thislinkcolor	564, 570, 584, 752, 754, 792, 802, 803, 805, 877
\ef@thislinkcolor@SAVE	753, 806
\ef@YES	450, 451, 496, 500, 760
\egroup	204, 435
\empty	609
\endinput	17, 20, 25, 48, 66
\eq@mlhyph	892, 900, 902, 911, 914
\eq@drivernum	51, 55
\eq@mlchunk	905
\eq@mlcrackat	454, 457, 703, 743, 774–776, 783, 796, 811, 886, 906, 910, 920
\eq@mlcrackinsat	459, 819
\eq@mlhyph	911
\eq@mlignore	451
\eq@S@value	449, 759

\eq@setWidgetProps	194	\lastbox	642
\everypar	622	\leftskip	619
\eWebCustomize	67	\linknotformedfalse	436
\ExecuteOptions	70, 71, 75	\linknotformedtrue	470, 765
\ExecuteOptions@SAVE	70, 75	\literalps@out	480, 510, 525, 814, 818, 825, 832, 854
\ExecuteOptionsX	71, 73, 74	\llap	894
F			
\FixupProc	845, 864		
\font	211, 614		
\fontdimen	211		
G			
\getrefbykeydefault	551		
H			
\H	184	\m@ne	617, 632
\hbadness	616	\maxdimen	615
\hglue	440	\ml@nnotName	97, 100, 518, 537, 860
\href	594	\ML@action	17, 20, 22, 25, 48
\Hurl	36	\ml@adj@x	208, 353, 357, 374, 381
\hyper@chars	563, 791	\ml@adj@y	208
\hyper@linkfile	599	\ml@bld@quadchunks	704, 710, 714, 720, 723, 730
\hyper@linkurl	598	\ml@displayfalse	461, 907, 924
\hyphenchar	614	\ml@displaytrue	441, 458, 662, 909, 921
\hyphenpenalty	624	\ml@dynamicsetup	927, 950, 963
I			
\if@ml@dvips	49, 89, 102	\ml@earlyExecProps	193, 474, 799
\ifcr@ckit	437	\ml@err@msg	64, 68, 69
\ifdim	431, 497, 676	\ml@finish@link	477, 812, 816, 823, 846
\ifixxmlinks	95, 480, 814, 818, 824, 855	\ml@hyper@launch	583, 600
\ifHy@colorlinks	188, 876	\ml@hyper@linkfile	569, 599
\ifHy@newwindow	576, 587	\ml@hyper@linkurl	563, 598
\ifHy@pdfnewwindow	576, 587	\ml@HytextArg	443, 476
\iflinknotformed	158, 436	\ml@interface@analyze	22, 645, 700
\ifml@display	662, 951	\ml@lngth	470–472, 477, 765, 768, 797, 812, 899, 942
\ifml@linktotalchanged	155, 174	\ml@lngthchunki	771, 776, 778
\ifmlmarks	427, 431	\ml@lngthchunkii	771, 777, 780
\ifoldstylequads	94, 196, 479, 509, 836, 849	\ml@mlinktot@l@changed	166, 167
\ifpdf	16, 68	\ml@mlinktotalchanged	174, 178
\ifSmallRect	96, 156	\ml@MrkLnk	14, 431, 474, 798, 822
\ifvoid	643	\ml@next	453, 456, 458, 466
\ifxetex	19, 69	\ml@nocolor@defaults	184, 189, 448, 758
\InputIfFileExists	72	\ml@nocolorHighlight	180, 184
\isstrikeout	500	\ml@nocolorLineStyle	181, 185
\isWindow	577–579, 588–590	\ml@nocolorLineWidth	182, 185
\itsunderline	450, 496, 760	\ml@qlngthchunki	
L			
\labelRef	549, 555, 557, 561	705, 709, 711, 715, 766, 769, 772, 779, 816
M			
\ml@qlngthchunkii	719, 721, 724, 772, 781, 820, 823	\ml@raiseamt	432, 433, 435, 893
\ml@setlink	445, 478, 755, 813, 817, 824	\ml@setnocolorDefaults	183, 191
\ml@SOUL@analyze	633, 637, 648	\ml@SOUL@dosyllable	653, 658, 663, 668, 675, 696
\ml@SOUL@doword	605, 795	\ml@SOUL@doword@SAVE	794
\ml@SOUL@stop	559, 881	\ml@space	458, 461, 912, 915, 930, 932
\ml@start@link	472, 709, 719, 797, 820, 829		

\ml@strut	493, 495
\ml@temp	556, 558, 560, 562
\ml@tempi	557, 558, 561, 562
\ml@typeset@syl	747, 895, 896, 902
\ml@typset@syl	893, 895
\ml@underlined	438, 449, 759
\ml@url	789, 804
\ml@urlbuild	733
\ml@urlbuildi	603, 606, 705, 711, 715, 804
\ml@urlbuildii	604, 607, 721, 724, 743, 822
\mlcrackinsat	210
\mlcs	41, 202
\mlcsarg	50, 482, 484, 807, 809
\mldb	205–207, 262, 263, 270, 271, 276, 279–281, 283, 284, 286, 290–293, 297, 299, 300, 304, 305, 307, 308, 310–316, 323, 324, 329–332, 334, 336, 338, 340, 341, 364, 419
\mldblevel	62, 63, 84, 207
\mldbModeOff	206
\mldbModeOn	205
\mlDict	103, 131, 512, 527, 859, 870
\mlfixOff	197
\mlh@postambleCmd	475, 800, 949, 959
\mlh@preambleCmdInsert	201, 473, 791, 798, 883
\mlh@setQuadSyllable	488, 745, 899, 942
\mlhref	35, 595
\mlhref@args	566, 579, 590, 597
\mlhyperlink	31, 552
\mlhyperref	32, 552
\mlhypertext	30, 439, 553, 555, 557, 561, 566, 579, 590
\mlhypertext@i	453, 456, 468
\mlignore	459, 462
\mlinkstotal	164, 168
\mlinktotalchangedfalse	155
\mlinktotalchangedtrue	172
\mllnkcontainer	98
\mlM@XCNT	771, 773, 775, 782
\mlmarksfalse	427, 429
\mlMarksOff	14, 38, 429
\mlMarksOn	14, 37, 428, 895
\mlmarkstrue	428
\mlMaxNSylls	42, 209, 298
\mlNameref	33, 552
\mlnameref	34, 552
\mlpgMsg	85, 88, 92
\mlurl	23, 36, 603
\mlurl@	750, 751
\MrkLnkLtr	430, 435
\multiply	768, 778, 780
N	
\n	85, 87, 217, 220, 228, 235, 237, 238, 240, 243, 245, 247, 248, 250, 253, 256, 262, 263, 271, 276, 281, 284, 286, 291, 293, 297, 300, 305, 307, 308, 311, 313, 316, 324, 329–331, 333, 335, 337, 339–341, 364, 423
\NeedsTeXFormat	3
\newbox	151, 603, 604
\nolinkurl	789
\normalcolor	434, 463, 752, 827, 894
\normalfont	433, 894
O	
\OldStyleBoxesOff	198, 200
\OldStyleBoxesOn	197, 199
\oldstylequadsfalse	196, 198
\oldstylequadstrue	197
options:	
\dbevel	4
\dvips	3
\dvipsone	3
\urlOpts	4
P	
\PackageError	68, 69
\PackageWarning	851
\PackageWarningNoLine	43, 159, 169, 175
\par@QRect	125, 145, 416
\parfillskip	623
\PassOptionsToPackage	7, 52, 53, 56, 57, 61
\pboxRect	101
\penalty	460, 747, 819, 821
\pgmonitoring	89, 115, 135
\presets	448, 566, 579, 590, 758
\pretolerance	617
\processAppArgs	447, 757
\ProcessOptionsX	8, 77
\protected@edef	556, 560
\ProvidesPackage	5
\ps@mark	83, 513, 834, 864, 871
Q	
\QuadPoints	446, 756
R	
\raisebox	435, 893
\removelastspace	210, 211
\RequirePackage	4, 9, 13–15, 26–28, 78–82
\revCrackAt	707, 713, 772, 785
\rightskip	620
\rlap	433

S	
\S	185, 448, 758
\set@LinkPboxDefaults	447, 757
\setbox	432, 493, 495, 606, 607, 612, 638, 642, 686, 702, 744
\setLinkPbox	445, 755
\setQuadBox	117, 137, 514, 529
\smallRectTF	94, 99
\SmallRecttrue	156
\smash	433, 894
\SOUL@	970
\SOUL@buffer	611
\SOUL@charkern	671, 690, 901, 946
\SOUL@doword	794, 795
\SOUL@errormsg	629, 630, 682, 683
\SOUL@error	630
\SOUL@eventuallyexhyphen	665
\SOUL@everyexhyphen	953, 956, 966
\SOUL@everyhyphen	656, 965
\SOUL@everylowerthan	673
\SOUL@everyspace	952, 955, 964
\SOUL@evensyllable	677, 962
\SOUL@everytoken	694
\SOUL@getkern	654, 671, 690
\SOUL@gettoken	664, 670, 689
\SOUL@hyphenhintM	667
\SOUL@hyphkern	654, 937, 943, 944
\SOUL@lasttoken	654, 671, 688, 690, 693
\SOUL@lowerthanM	669
\SOUL@mleverysyllable@ii	889, 904
\SOUL@cmlheveryexhyphen	941, 953, 966
\SOUL@cmlheveryhyphen	934, 965
\SOUL@cmlheveryspace	929, 931, 952, 964
\SOUL@cmlheverysyllable	884, 962
\SOUL@cmlheverysyllable@i	887, 898, 917, 922
\SOUL@cmlhpostamble	959, 967
\SOUL@cmlhpreamble	790, 882, 961
\SOUL@cmlhsetup	960, 970
\SOUL@n	668, 675, 679, 684, 696, 698
\SOUL@postamble	967
\SOUL@preamble	961
\SOUL@puttoken	674, 678, 692
\SOUL@sethyphenchar	655, 938
\SOUL@setkern	901, 937, 943, 944, 946
\SOUL@setup	960
\SOUL@stop	559
\SOUL@stopM	681
\SOUL@syllable	650, 695, 900, 901
\SOUL@syllgoal	646, 652, 676
\SOUL@syllwidth	649, 676, 687
\SOUL@token	665, 666, 672, 686, 688, 691, 693, 695
\SOUL@tt	613, 686
\SOUL@ttwidth	652
\SOUL@word	608, 628, 634
\spaceskip	929
\strut	435, 894
\syllableCnt	154, 701, 707, 713, 717, 762, 808, 885, 902, 906, 910, 920, 941, 969
T	
\texttt	41, 202
\theView	573, 574, 580
\tiny	435, 894
\tolerance	618
\turnSyllbCntOff	27, 40, 896, 897
\turnSyllbCntOn	27, 39, 895
U	
\unhbox	702
\unhcopy	745–747, 804, 822
\unkern	936
\unpenalty	641
\unvcopy	639
\URI	755
\url@Opts	59, 60, 78
\urldef	789
urlOpts (option)	4
W	
\W	185, 448, 758
\write	163, 172, 482, 484, 807, 809
\wrt@linksnotformed	157, 158
\wrtmlinktot@l	157, 163
X	
\x	504–507, 608, 609, 666, 667, 669, 681, 695

8 Change History

v2.0 (2016/02/16)		v2.3 (2018/04/26)	
\mlurl: Added support for the \url command.	19	General: Added aeb-mlink as an alternate name for this package	2
v2.0.1 (2017/09/19)		v2.3.1 (2018/07/18)	
\mlurl: Save the incoming link color	25	\mlhypertext: added \hglue0pt to add some hard glue just before the beginning of \mlhypertext	15
v2.1.10 (2018/03/25)		v2.3.2 (2018/08/09)	
General: Set /Rect to minimal rectangle containing text	13	General: \mlMaxNSylls now sets the array size of gAryL	11
v2.1.17 (2018/04/19)		Form the array aFixup for aeb_mlink	12
\mlhypertext: added conditional command definition for \mlhypertext	15	\mlurl: We make gFixup into a macro name \FixupProc	26
v2.1.18 (2018/04/20)		v2.3.5 (2020/01/06)	
\mlurl: Added bCreateLink, etc., to protect against distiller/ps2pdf from crashing	27	General: Added url0pts	4
v2.1.2 (2018/03/09)		Conform to the new web.cfg format	4
General: Added PS proc smallquadpointsfixup	13	Now require url package and pass options to url through url0pts	4
v2.1.3 (2018/03/10)		v2.3.6 (2020/07/12)	
General: Reinstate old style link	8	General: Added some safeguards against using a driver other than dvips	2
v2.1.8 (2018/03/19)			
General: Added tracking marks	14		
v2.1.9 (2018/03/22)			
General: Changed the definition of \ml@MrkLnk	14		