

# The `aeb_mlink` Package

## A member of the AeB Pro family

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

processed July 14, 2020

### Contents

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

## 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 than 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@dvipfalse
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 }

```

**urlOpts** The options of the url package may be passed through the value of this key; for example, `urlOpts={hyphens}`.

```

59 \define@key{aeb_mlink.sty}{urlOpts}[]{\def\url@0pts{[#1]}}
60 \let\url@0pts\empty

```

Undefined options are passed to eforms.

```

61 \DeclareOptionX*{\PassOptionsToPackage{\CurrentOption}{eforms}}

```

**dbllevel** Sets the debug level.

```

62 \@ifundefined{mldblevel}{\newcount\mldblevel\mldblevel=0 }{}
63 \define@key{aeb_mlink.sty}{dbllevel}[0]{\mldblevel=#1 }

```

Package error message when not dvips (or dvispone)

```

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 urlOpts.

```

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\space@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}

```

`\pboxRect` is defined in `eforms`

```

101 \def\pboxRect{mlRect }
102 \if@ml@dvips

```

**dvips** driver: Code for the dvips driver This next `\special` defines some standard conversion formulas,  $\text{T}_{\text{E}}\text{X}$  to PDF and PDF to  $\text{T}_{\text{E}}\text{X}$  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,  $\text{T}_{\text{E}}\text{X}$  to PDF and PDF to  $\text{T}_{\text{E}}\text{X}$  in the  $\text{YandY}$   $\text{T}_{\text{E}}\text{X}$  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 /bhook{ \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\ae@b@bbox
152 \newcount\ae@arrayIndx \ae@arrayIndx=0
153 \newcount\ae@mLinkCnt \ae@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 \iffixmlinks (defined in eforms is true.

156 \newif\ifSmallRect \SmallRecttrue
157 \AtEndDocument{\wrtmllinktot@1\ckchngmllinktot@1\wrt@linksnotformed}
158 \def\wrt@linksnotformed{\iflinknotformed
159   \PackageWarningNoLine{ae@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\wrtmllinktot@1{\immediate\write\@auxout{\string\gdef
164   \string\mllinkstotal{\the\ae@mLinkCnt}}}
165 \def\ckchngmllinktot@1{\ifundefined{mllinkstotal}{
166   {\ml@mllinktot@1@changed}}
167 \def\ml@mllinktot@1@changed{%
168   \ifnum\mllinkstotal=\the\ae@mLinkCnt\relax\else
169   \PackageWarningNoLine{ae@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{ae@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{\mlfix0ff\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<\mldblevel#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}%
```

if comparison 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 inert 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 exh 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 mLinkFxp(*num*), we declare the next array of length gCount just computed.

```

325 /sliceOfLinkfxup gCount array def^^J%

```

Populate this array with a slice of mLinkFxp(*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 \mldb1{(/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=Opt`. 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=Opt
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] []{\hglueOpt\beginngroup
441   \global\ml@displaytrue
442   \toks@=\expandafter{#2}%
443   \edef\ml@HytexArg{\the\toks@}}%
444   \global\ae@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@mlcrackat` is empty or not. If not-empty we break this link to two parts.

```

451   \ifx\eq@mlignore\ef@YES
452     \global\advance\ae@mLinkCnt\@ne\relax
453     \def\ml@next{\mlhypertext@i{#1}}\else
454     \ifx\eq@mlcrackat\@empty

```

A ‘normal’ link, continue

```

455     \global\advance\ae@mLinkCnt\@ne\relax
456     \def\ml@next{\mlhypertext@i{#1}}%
457     \else % \eq@mlcrackat 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@mlcrackinsat
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@i#1{%
469   \ifundefined{mLinkLngh\the\aeB@mLinkCnt}{\global
470     \linknotformedtrue\def\ml@lngh{17}}
471   {\edef\ml@lngh{\@nameuse{mLinkLngh\the\aeB@mLinkCnt}}}%
472   \ml@start@link{\the\aeB@mLinkCnt}{\ml@lngh}% Step 1
473   \def\mlh@preambleCmdInsert{%
474     \ml@MrkLnk{\the\aeB@mLinkCnt}\ml@earlyExecProps{#1}}%
475   \def\mlh@postambleCmd{\endgroup}%
476   Start soul on \mlhypertext (\aeB@mlh)
477   \expandafter\aeB@mlh\ml@HytexArg
478   \ml@finish@link{\the\aeB@mLinkCnt}{\ml@lngh}%
479   \ml@setlink{\the\aeB@mLinkCnt}%
480   \ifoldstylequads\else
481     \iffixmlinks\literalps@out{restore}\fi\fi
482     \@ifundefined{mLinkLngh\the\aeB@mLinkCnt}{%
483       \immediate\write\@auxout{\string\mlcsarg
484         \string\gdef{mLinkLngh\the\aeB@mLinkCnt}{17}}%
485       }{\immediate\write\@auxout{\string\mlcsarg
486         \string\gdef{mLinkLngh\the\aeB@mLinkCnt}%
487         {\the\aeB@arrayIndx}}}\endgroup
488 }
489 \def\mlh@setQuadSyllable#1#2#3#4{%
490 % #1 = current array Index
491 % #2 = quad total
492 % #3 = link cnt
493 % #4 = content
494 \setbox\aeB@bbox=\hbox{\ml@strut#4}%
495 {%
496   \setbox\@tempboxa\hbox{\ml@strut}%
497   \ifx\itsunderline\ef@YES\@tempdima1bp\relax\else
498     \@tempdima\dp\@tempboxa \ifdim\@tempdima>2bp
499       \advance\@tempdima-2bp\fi
500   \fi
501   \ifx\isstrikeout\ef@YES\advance\@tempdima-2bp\fi
502   \dp\aeB@bbox\@tempdima
503   \@tempdima\ht\@tempboxa \advance\@tempdima\dp\aeB@bbox
504   \advance\@tempdima1bp
505   \ht\aeB@bbox\@tempdima\def\x{\the\count\z@}%
506   \count\z@=\ht\aeB@bbox\xdef\aeB@bbox@ht{\x}%
507   \count\z@=\wd\aeB@bbox\xdef\aeB@bbox@wd{\x}%
508   \count\z@=\dp\aeB@bbox\xdef\aeB@bbox@dp{\x}%
509 }%

```



```

509 \ifoldstylequads
510   \literalps@out{%
511     bCreateLink {^^J%
512       \mDict\space/mLinkFxp#3\space known {^^J%
513       \ps@mark{mLink#3}
514       \the\aeB@arrayIndx\space [\setQuadBox]
515       \space /PUTINTERVAL pdfmark}{^^J%
516     xMsgB {
517       /xMsgB false def
518       (\mL@nnotName\the\aeB@arrayIndx)
519       #3\space
520       xMsg % dpsa08
521     } if^^J% xMsgB
522     } ifelse } if
523   }%
524 \else
525   \literalps@out{%
526     bCreateLink {^^J%
527       \mDict\space/mLinkFxp#3\space known {^^J%
528       mLinkFxp#3\space
529       #1\space[\setQuadBox] putinterval^^J%
530       Initiate fix up of little rectangles
531       #2\space %
532       #3\space
533       mLinkFxp#3\space
534       #1\space
535       smallquadpointsfixup }{^^J%
536     xMsgB {
537       /xMsgB false def
538       (\mL@nnotName#3)
539       #1\space
540       xMsg % dpsa08
541     } if^^J% xMsgB
542     } ifelse } if
543   }%
544 \fi
545 \global\advance\aeB@arrayIndx8\relax
546 }

```

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

```

546 \def\aeB@exiii{\expandafter\expandafter\expandafter}
547 \def\aeBnameref#1{\@ifundefined{r#1}{??}
548   {\aeB@exiii\@thirdoffive\csname r#1\endcsname}}
549 \def\labelRef#1{\@ifundefined{r#1}{Doc-Start}
550   {\aeB@exiii\@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@temp{\mlhypertext[#1\A{/S/GoTo/D (\labelRef{#2})}]}%
558   \expandafter\ml@temp\expandafter{\ml@temp}}
```

We use a work around to a `\relax` problem encountered in the `\mlNameRef` command. L<sup>A</sup>T<sub>E</sub>X inserts a `relax` at the end of label titles, which stops soul. We insert `\let\SOU@stop\ml@SOUL@stop`, this seems to work, no guarantees.

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

```
559 \newcommand\mlNameRef[2] [] {\let\SOU@stop\ml@SOUL@stop
560   \protected@edef\ml@temp{'\aebnameref{#2}' on page~\atPage{#2}}%
561   \def\ml@temp{\mlhypertext[#1\A{/S/GoTo/D (\labelRef{#2})}]}%
562   \expandafter\ml@temp\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\ae@saved@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   \begingroup
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   \ae@saved@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@dossyllable`, 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       \the\SOUL@word}%

```

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@dossyllable rather than \SOUL@dossyllable.

```

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

```

Use \ml@SOUL@dossyllable not \SOUL@dossyllable.

```

658 \ml@SOUL@dossyllable
659 \fi
660 \fi
661 }}

```

```

662 \newif\ifml@display \ml@displaytrue

```

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

```

663 \def\ml@SOUL@dossyllable{%
664 \SOUL@gettoken
665 \SOUL@eventuallyexhyphen{\the\SOUL@token}%
666 \edef\x{\the\SOUL@token}%
667 \ifx\x\SOUL@hyphenhintM
668 \let\SOUL@n\ml@SOUL@dossyllable
669 \else\ifx\x\SOUL@lowerthanM
670 \SOUL@gettoken
671 \SOUL@getkern{\the\SOUL@lasttoken}{\SOUL@charkern}
672 {\the\SOUL@token}%
673 \SOUL@everylowerthan
674 \SOUL@puttoken
675 \let\SOUL@n\ml@SOUL@dossyllable
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\@eb@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     \@eb@arrayIndx=\z@

```

```

start first link
709     \ml@start@link{\the\ae@mLinkCnt}{\ml@qlngthchunki}%
710     \ml@bld@quadchunks{\the\ae@mLinkCnt}
711         {\ml@qlngthchunki}{\ml@urlbuildi}%
712     \else
Continue with first link
713     \ifnum\syllableCnt>\revCrackAt\relax
714         \ml@bld@quadchunks{\the\ae@mLinkCnt}
715             {\ml@qlngthchunki}{\ml@urlbuildi}%
716     \else
717         \ifnum\syllableCnt=\@ne
start second link
718         \ae@arrayIndx=\z@
Everything is in reverse order, this is the last syllable of the second chunk
719         \ml@start@link{\ae@mLinkCnt}{\ml@qlngthchunkii}%
720         \ml@bld@quadchunks{\ae@mLinkCnt}
721             {\ml@qlngthchunkii}{\ml@urlbuildii}%
722     \else
continue with second link
723         \ml@bld@quadchunks{\ae@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 \@ifundefined{mLinkLngth\the\ae@mLinkCnt}
735 {\edef\@indx{\the\ae@arrayIndx}}
736 {\@tempcnta#2\relax
737 \advance\@tempcnta-8\relax
738 \advance\@tempcnta-\ae@arrayIndx
739 \def\@indx{\the\@tempcnta}%
740 }%
741 \@tempcntb=\@indx \divide\@tempcntb by 8
742 \advance\@tempcntb by 1
743 \ifx#3\ml@urlbuildii \advance\@tempcntb by \eq@mlcrackat\relax\fi
744 \global\setbox#3=\hbox{%
745 \mlh@setQuadSyllable{\@indx}{#2}{#1}{\unhcopy\@ne}%
746 \hbox{\unhcopy\@ne}\relax
747 \ml@typeset@@syl{\@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 `\ml@url` 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{S}\W{0}#1\end@nil
759 \ifx\eq@S@value\ml@underlined
760   \let\itsunderline\ef@YES\else\let\itsunderline\ef@NO\fi
```

`\ml@url` is the specified url to create a link around

```
761 \global\ae@arrayIndx=0\relax
762 \global\syllableCnt=0\relax
763 \global\advance\ae@mLinkCnt\@ne\relax
764 \ifundefined{mLinkLngh\the\ae@mLinkCnt}{%
765   \global\linknotformedtrue\def\ml@lngh{17}%
766   \def\ml@qlnghchunki{17}}
767   {\@tempcnta\@nameuse{mLinkLngh\the\ae@mLinkCnt}\relax
768   \edef\ml@lngh{\the\@tempcnta}%\multiply\@tempcnta8\relax
769   \edef\ml@qlnghchunki{\the\@tempcnta}}%
```

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\ae@mLinkCnt}
771   {\def\mLM@XCNT{0}\def\ml@lnghchunki{0}\def\ml@lnghchunkii{0}%
772   \def\ml@qlnghchunki{0}\def\ml@qlnghchunkii{0}\def\revCrackAt{0}}
773   {\edef\mLM@XCNT{\@nameuse{mLinkSyCnt\the\ae@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     \@tempcnta\mLM@XCNT\relax\advance\@tempcnta-\eq@mlcrackat\relax
776     \edef\ml@lnghchunki{\eq@mlcrackat}%
777     \edef\ml@lnghchunkii{\the\@tempcnta}%
778     \@tempcnta\ml@lnghchunki\relax\multiply\@tempcnta8\relax
779     \edef\ml@qlnghchunki{\the\@tempcnta}%
780     \@tempcnta\ml@lnghchunkii\relax\multiply\@tempcnta8\relax
781     \edef\ml@qlnghchunkii{\the\@tempcnta}}%
```



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\@eB@mLinkCnt\advance\@tempcnta\@ne
788     \edef\@eB@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\@eB@mLinkCnt}{\ml@length}\fi
798     \def\mlh@preambleCmdInsert{\ml@MrkLnk{\the\@eB@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 `\@eB@mlh` which starts soul with `\SOUL@`. This does this analysis, the custom command build the url in `\ml@urlbuild`, which we then unbox.

```

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

```

```

815 \else
816 \ml@finish@link{\the\aeB@mLinkCnt}{\ml@qlngthchunki}%
817 \ml@setlink{\the\aeB@mLinkCnt}%
818 \iffixmlinks\literalps@out{restore}\fi
819 \eq@mlcrackinsat\penalty-100
820 \ml@start@link{\aeB@mLinkCnt@}{\ml@qlngthchunkii}%
821 \penalty0\@ifundefined{mLinkLngh\the\aeB@mLinkCnt}{%
822 {\ml@MrkLnk{\aeB@mLinkCnt@}}\unhcopy\ml@urlbuildii
823 \ml@finish@link{\aeB@mLinkCnt@}{\ml@qlngthchunkii}%
824 \ml@setlink{\aeB@mLinkCnt@}\iffixmlinks
825 \literalps@out{restore}\fi
826 \global\advance\aeB@mLinkCnt@\@ne
827 \fi\aftergroup\normalcolor\endgroup
828 }
829 \def\ml@start@link#1#2{% Step 1
830 % #1=link number
831 % #2 = final quad length
832 \literalps@out{%
833 /xMsgB 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 mLIbld #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{% Step 4 and 5
847 % #1=link number
848 % #2 = final quad length
849 \ifoldstylequads\else
850 \ifnum\aeB@arrayIndx=0\relax
851 \PackageWarning{aeB_mlink}{%
852 Problem with mLink\the\aeB@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\aeB@arrayIndx>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   \mDict\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 `mlnameref` and `mlNameref` 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@mlheverysyllable{% dpsaug16
885   \global\advance\syllableCnt\@ne
886   \ifx\eq@mlcrackat\@empty
887     \expandafter\SOUL@mlheverysyllable@i
888   \else
889     \expandafter\SOUL@mlheverysyllable@ii
890   \fi
891 }
892 \let\eq@mlhyph\@empty
893 \def\ml@typeset@syl#1{\raisebox{\ml@raiseamt}
894   {\smash{\normalfont\normalcolor\tiny\strut\llap{\the#1}}}}

```

```

\turnSyllbCntOn Turns on the counting of the syllables, while \turnSyllbCntOff counting marks
\turnSyllbCntOff off.
895 \def\turnSyllbCntOn{\mlMarksOn\let\ml@typeset@@syl\ml@typeset@syl}
896 \def\turnSyllbCntOff{\let\ml@typeset@@syl\@gobble}
897 \turnSyllbCntOff

```

```

898 \def\SOUL@mlheverysyllable@i{%
899   \mlh@setQuadSyllable{\the\ae@arrayIndx}{\ml@lngth}%
900   {\the\ae@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@mlheverysyllable@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@mlheverysyllable@i
918     \fi
919   \else
920     \ifnum\syllableCnt>\eq@mlcrackat\relax
921       \global\ml@displaytrue
922       \SOUL@mlheverysyllable@i
923     \else
924       \global\ml@displayfalse
925     \fi
926   \fi
927   \ml@dynamicsetup
928 }
929 %\def\SOUL@mlheveryspace#1{#1\space\hskip\spaceskip}
930 \let\ml@space\space
931 \def\SOUL@mlheveryspace#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\ae@arrayIndx}{\ml@lngth}%
943   {\the\ae@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@postambleCmd{\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@postambleCmd}
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	
<code>\%</code> .....	203
<code>\@auxout</code> .....	163, 172, 482, 484, 807, 809
<code>\@backslashchar</code> .....	41, 202
<code>\@filebordercolor</code> .....	571
<code>\@filecolor</code> .....	570, 584
<code>\@fourthoffive</code> .....	550
<code>\@hyper@launch</code> .....	600
<code>\@indx</code> .....	735, 739, 741, 745
<code>\@linkbordercolor</code> .....	179
<code>\@makeoother</code> .....	203, 749
<code>\@ml@dvi<sup>s</sup>false</code> .....	51
<code>\@ml@dvi<sup>s</sup>true</code> .....	49, 55
<code>\@onlypreamble</code> .....	199, 200
<code>\@pdfstartview</code> .....	572, 574
<code>\@plus</code> .....	623
<code>\@runbordercolor</code> .....	585
<code>\@tempboxa</code> .....	495, 497, 502
<code>\@thirdoffive</code> .....	548
<code>\@urlbordercolor</code> .....	565, 793
<code>\@urlcolor</code> .....	564, 792
<code>\~</code> .....	749
<b>A</b>	
<code>\A</code> .....	553, 555, 557, 561, 566, 579, 590, 755
<code>\aeb@arrayIndx</code> .....	152, 444, 486, 514, 518, 544, 708, 718, 735, 738, 761, 810, 850, 856, 899, 942
<code>\aeb@bbox</code> .....	151, 493, 501, 502, 504–507
<code>\aeb@bbox@dp</code> .....	118, 138, 507
<code>\aeb@bbox@ht</code> .....	122, 142, 505
<code>\aeb@bbox@wd</code> .....	121, 123, 141, 143, 506
<code>\aeb@exiii</code> .....	546, 548, 550
<code>\aeb@mlh</code> .....	476, 804, 969
<code>\aeb@mLinkCnt</code> .....	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
<code>\aeb@mLinkCnt@</code> .....	719, 720, 723, 788, 820, 822–824
<code>\aeb@saved@href</code> .....	594, 601
<code>\aebnameref</code> .....	547, 556, 560
<code>\aftergroup</code> .....	462, 827
<code>\AtBeginDocument</code> .....	178
<code>\AtBeginDvi</code> .....	212
<code>\AtEndDocument</code> .....	157
<code>\atPage</code> .....	551, 560
<b>B</b>	
<code>\baselineskip</code> .....	431, 433
<code>\bWebCustomize</code> .....	66
<b>C</b>	
<code>\ckchngm<sup>l</sup>inktot@1</code> .....	157, 165
<code>\CMT</code> .....	204
<code>\Color</code> .....	186
<code>\color</code> .....	877
<code>\cr@ckitfalse</code> .....	437
<code>\cr@ckittrue</code> .....	460
<code>\CurrentBorderColor</code> ...	179, 186, 565, 571, 585, 793
<code>\CurrentOption</code> .....	7, 61
<b>D</b>	
<code>dbllevel</code> (option) .....	4
<code>\DeclareOptionX</code> .....	7, 51, 55, 61
<code>\DeclareRobustCommand</code> .....	969
<code>\discretionary</code> .....	935, 945
<code>\divide</code> .....	741
<code>dvips</code> (option) .....	3
<code>dvipsone</code> (option) .....	3
<b>E</b>	
<code>\ef@colorthislink</code> .....	791, 804, 877, 879, 883
<code>\ef@NO</code> .....	450, 760
<code>\ef@sanitize@toks</code> .....	750
<code>\ef@thislinkcolor</code> .....	564, 570, 584, 752, 754, 792, 802, 803, 805, 877
<code>\ef@thislinkcolor@SAVE</code> .....	753, 806
<code>\ef@YES</code> .....	450, 451, 496, 500, 760
<code>\egroup</code> .....	204, 435
<code>\empty</code> .....	609
<code>\endinput</code> .....	17, 20, 25, 48, 66
<code>\eq@m<sup>l</sup>hyph</code> .....	892, 900, 902, 911, 914
<code>\eq@drivernum</code> .....	51, 55
<code>\eq@m<sup>l</sup>chunk</code> .....	905
<code>\eq@m<sup>l</sup>crackat</code> .....	454, 457, 703, 743, 774–776, 783, 796, 811, 886, 906, 910, 920
<code>\eq@m<sup>l</sup>crackinsat</code> .....	459, 819
<code>\eq@m<sup>l</sup>hyph</code> .....	911
<code>\eq@m<sup>l</sup>ignore</code> .....	451
<code>\eq@S@value</code> .....	449, 759

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

<code>\ml@strut</code>	493, 495		
<code>\ml@temp</code>	556, 558, 560, 562		
<code>\ml@temp_i</code>	557, 558, 561, 562		
<code>\ml@typeset@syl</code>	747, 895, 896, 902		
<code>\ml@typset@syl</code>	893, 895		
<code>\ml@underlined</code>	438, 449, 759		
<code>\ml@url</code>	789, 804		
<code>\ml@urlbuild</code>	733		
<code>\ml@urlbuild_i</code>	603, 606, 705, 711, 715, 804		
<code>\ml@urlbuild_ii</code>	604, 607, 721, 724, 743, 822		
<code>\mlcrackinsat</code>	210		
<code>\mlcs</code>	41, 202		
<code>\mlcsarg</code>	50, 482, 484, 807, 809		
<code>\mldb</code>	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		
<code>\mldblevel</code>	62, 63, 84, 207		
<code>\mldbModeOff</code>	206		
<code>\mldbModeOn</code>	205		
<code>\mlDict</code>	103, 131, 512, 527, 859, 870		
<code>\mlfixOff</code>	197		
<code>\mlh@postambleCmd</code>	475, 800, 949, 959		
<code>\mlh@preambleCmdInsert</code>	201, 473, 791, 798, 883		
<code>\mlh@setQuadSyllable</code>	488, 745, 899, 942		
<code>\mlhref</code>	35, <a href="#">595</a>		
<code>\mlhref@args</code>	566, 579, 590, 597		
<code>\mlhyperlink</code>	31, <a href="#">552</a>		
<code>\mlhyperref</code>	32, <a href="#">552</a>		
<code>\mlhypertext</code>	30, <a href="#">439</a> , 553, 555, 557, 561, 566, 579, 590		
<code>\mlhypertext@i</code>	453, 456, 468		
<code>\mlignore</code>	459, 462		
<code>\mlinkstotal</code>	164, 168		
<code>\mlinktotalchangedfalse</code>	155		
<code>\mlinktotalchangedtrue</code>	172		
<code>\mllnkcontainer</code>	98		
<code>\mlM@XCNT</code>	771, 773, 775, 782		
<code>\mlmarksfalse</code>	427, 429		
<code>\mlMarksOff</code>	14, 38, 429		
<code>\mlMarksOn</code>	14, 37, 428, 895		
<code>\mlmarkstrue</code>	428		
<code>\mlMaxNSylls</code>	42, 209, 298		
<code>\mlNameref</code>	33, <a href="#">552</a>		
<code>\mlnameref</code>	34, <a href="#">552</a>		
<code>\mlpgMsg</code>	85, 88, 92		
<code>\mlurl</code>	23, 36, <a href="#">603</a>		
<code>\mlurl@</code>	750, 751		
<code>\MrkLnkLtr</code>	430, 435		
<code>\multiply</code>	768, 778, 780		
		<b>N</b>	
<code>\n</code>	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		
<code>\NeedsTeXFormat</code>	3		
<code>\newbox</code>	151, 603, 604		
<code>\nolinkurl</code>	789		
<code>\normalcolor</code>	434, 463, 752, 827, 894		
<code>\normalfont</code>	433, 894		
		<b>O</b>	
<code>\OldStyleBoxesOff</code>	198, 200		
<code>\OldStyleBoxesOn</code>	197, 199		
<code>\oldstylequadsfalse</code>	196, 198		
<code>\oldstylequadstrue</code>	197		
options:			
<code>dblevel</code>	4		
<code>dvips</code>	3		
<code>dvipsone</code>	3		
<code>urlOpts</code>	4		
		<b>P</b>	
<code>\PackageError</code>	68, 69		
<code>\PackageWarning</code>	851		
<code>\PackageWarningNoLine</code>	43, 159, 169, 175		
<code>\par@Rect</code>	125, 145, 416		
<code>\parfillskip</code>	623		
<code>\PassOptionsToPackage</code>	7, 52, 53, 56, 57, 61		
<code>\pboxRect</code>	101		
<code>\penalty</code>	460, 747, 819, 821		
<code>\pgmonitoring</code>	89, 115, 135		
<code>\presets</code>	448, 566, 579, 590, 758		
<code>\pretolerance</code>	617		
<code>\processAppArgs</code>	447, 757		
<code>\ProcessOptionsX</code>	8, 77		
<code>\protected@edef</code>	556, 560		
<code>\ProvidesPackage</code>	5		
<code>\ps@mark</code>	83, 513, 834, 864, 871		
		<b>Q</b>	
<code>\QuadPoints</code>	446, 756		
		<b>R</b>	
<code>\raisebox</code>	435, 893		
<code>\removeatend</code>	210, 211		
<code>\RequirePackage</code>	4, 9, 13–15, 26–28, 78–82		
<code>\revCrackAt</code>	707, 713, 772, 785		
<code>\rightskip</code>	620		
<code>\rlap</code>	433		



S	
<code>\S</code>	185, 448, 758
<code>\set@LinkPboxDefaults</code>	447, 757
<code>\setbox</code>	432, 493, 495, 606, 607, 612, 638, 642, 686, 702, 744
<code>\setLinkPbox</code>	445, 755
<code>\setQuadBox</code>	117, 137, 514, 529
<code>\smallRectTF</code>	94, 99
<code>\SmallRecttrue</code>	156
<code>\smash</code>	433, 894
<code>\SOUL@</code>	970
<code>\SOUL@buffer</code>	611
<code>\SOUL@charkern</code>	671, 690, 901, 946
<code>\SOUL@doword</code>	794, 795
<code>\SOUL@errmsg</code>	629, 630, 682, 683
<code>\SOUL@error</code>	630
<code>\SOUL@eventuallyexhyphen</code>	665
<code>\SOUL@everyexhyphen</code>	953, 956, 966
<code>\SOUL@everyhyphen</code>	656, 965
<code>\SOUL@everylowerthan</code>	673
<code>\SOUL@everyspace</code>	952, 955, 964
<code>\SOUL@everysyllable</code>	677, 962
<code>\SOUL@everytoken</code>	694
<code>\SOUL@getkern</code>	654, 671, 690
<code>\SOUL@gettoken</code>	664, 670, 689
<code>\SOUL@hyphenhintM</code>	667
<code>\SOUL@hyphkern</code>	654, 937, 943, 944
<code>\SOUL@lasttoken</code>	654, 671, 688, 690, 693
<code>\SOUL@lowerthanM</code>	669
<code>\SOUL@mlleverysyllable@ii</code>	889, 904
<code>\SOUL@mlheveryexhyphen</code>	941, 953, 966
<code>\SOUL@mlheveryhyphen</code>	934, 965
<code>\SOUL@mlheveryspace</code>	929, 931, 952, 964
<code>\SOUL@mlheverysyllable</code>	884, 962
<code>\SOUL@mlheverysyllable@i</code>	887, 898, 917, 922
<code>\SOUL@mlhpostamble</code>	959, 967
<code>\SOUL@mlhpreamble</code>	790, 882, 961
<code>\SOUL@mlhsetup</code>	960, 970
<code>\SOUL@n</code>	668, 675, 679, 684, 696, 698
<code>\SOUL@postamble</code>	967
<code>\SOUL@preamble</code>	961
<code>\SOUL@puttoken</code>	674, 678, 692
<code>\SOUL@sethyphenchar</code>	655, 938
<code>\SOUL@setkern</code>	901, 937, 943, 944, 946
<code>\SOUL@setup</code>	960
<code>\SOUL@stop</code>	559
<code>\SOUL@stopM</code>	681
<code>\SOUL@syllable</code>	650, 695, 900, 901
<code>\SOUL@syllgoal</code>	646, 652, 676
<code>\SOUL@syllwidth</code>	649, 676, 687
<code>\SOUL@token</code>	665, 666, 672, 686, 688, 691, 693, 695
<code>\SOUL@tt</code>	613, 686
<code>\SOUL@ttwidth</code>	652
<code>\SOUL@word</code>	608, 628, 634
<code>\spaceskip</code>	929
<code>\strut</code>	435, 894
<code>\syllableCnt</code>	154, 701, 707, 713, 717, 762, 808, 885, 902, 906, 910, 920, 941, 969
T	
<code>\texttt</code>	41, 202
<code>\theView</code>	573, 574, 580
<code>\tiny</code>	435, 894
<code>\tolerance</code>	618
<code>\turnSyllbCntOff</code>	27, 40, 896, 897
<code>\turnSyllbCntOn</code>	27, 39, 895
U	
<code>\unhbox</code>	702
<code>\unhcopy</code>	745–747, 804, 822
<code>\unkern</code>	936
<code>\unpenalty</code>	641
<code>\unvcopy</code>	639
<code>\URI</code>	755
<code>\url@Opts</code>	59, 60, 78
<code>\urldef</code>	789
<code>urlOpts (option)</code>	4
W	
<code>\W</code>	185, 448, 758
<code>\write</code>	163, 172, 482, 484, 807, 809
<code>\wrt@linksnotformed</code>	157, 158
<code>\wrtmlinktot@l</code>	157, 163
X	
<code>\x</code>	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 <code>aeb-mlink</code> 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 <code>\hg1ueOpt</code> 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 <code>/Rect</code> to minimal rectangle containing text . . . . .	13	General: <code>\mlMaxNSylls</code> now sets the array size of <code>gAryL</code> . . . . .	11
v2.1.17 (2018/04/19)		Form the array <code>aFixup</code> for <code>aeb-mlink</code> . . . . .	12
\mlhypertext: added conditional command definition for <code>\mlhypertext</code> . . . . .	15	\mlurl: We make <code>gFixup</code> into a macro name <code>\FixupProc</code> . . . . .	26
v2.1.18 (2018/04/20)		v2.3.5 (2020/01/06)	
\mlurl: Added <code>bCreateLink</code> , etc., to protect against distiller/ps2pdf from crashing . . . . .	27	General: Added <code>urlOpts</code> . . . . .	4
v2.1.2 (2018/03/09)		Conform to the new <code>web.cfg</code> format . . . . .	4
General: Added PS proc <code>smallquadpointsfixup</code>	13	Now require <code>url</code> package and pass options to <code>url</code> through <code>urlOpts</code> . . . . .	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 <code>dvips</code> . . . . .	2
v2.1.8 (2018/03/19)			
General: Added tracking marks . . . . .	14		
v2.1.9 (2018/03/22)			
General: Changed the definition of <code>\ml@MrkLnk</code>	14		