

# The `aeb_mlink` Package

## A member of the AeB Pro family

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

processed January 9, 2020

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>The <code>aeb-mlink</code> Package</b>	<b>2</b>
<b>3</b>	<b>Package Requirements and Options</b>	<b>2</b>
<b>4</b>	<b>Driver Dependent Code</b>	<b>4</b>
<b>5</b>	<b>The Multi-line Linking Commands</b>	<b>5</b>
<b>6</b>	<b>Macros used by the SOUL Interface</b>	<b>26</b>
<b>7</b>	<b>Index</b>	<b>29</b>
<b>8</b>	<b>Change History</b>	<b>33</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]
16 \newif\if@ml@dvi ps \@ml@dvi ps true
17 \def\mlcsarg#1#2{\expandafter#1\csname#2\endcsname}

dvipsone Set the driver for dvipsone

18 \DeclareOptionX{dvipsone}{\def\eq@drivernum{0}\@ml@dvi ps false}
19 \PassOptionsToPackage{dvipsone}{eforms}
20 \PassOptionsToPackage{dvipsone}{hyperref}
21 }
```

**dvips** Set the driver for dvips

```

22 \DeclareOptionX{dvips}{\def\eq@drivernum{0}\@ml@dvipstrue
23 \PassOptionsToPackage{dvips}{eforms}
24 \PassOptionsToPackage{dvips}{hyperref}
25 }

```

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

```

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

```

Undefined options are passed to eforms.

```

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

```

**dblevel** Sets the debug level.

```

29 \@ifundefined{mldblevel}{\newcount\mldblevel\mldblevel=0 }{}
30 \define@key{aeb_mlink.sty}{dblevel}[0]{\mldblevel=#1 }

```

Package error message when not dvips (or dvipsone)

```

31 \def\ml@err@msg{This package requires the driver dvips and\MessageBreak
32 Adobe Distiller as the PDF creator}

```

(2020/01/06) Conform to the new web.cfg format.

```

33 \let\bWebCustomize\endinput
34 \let\eWebCustomize\relax
35 \ifpdf\PackageError{aeb_mlink}{\ml@err@msg}\else
36 \ifxetex\PackageError{aeb_mlink}{\ml@err@msg}\else
37 \let\ExecuteOptions@SAVE\ExecuteOptions
38 \let\ExecuteOptions\ExecuteOptionsX
39 \InputIfFileExists{web.cfg}{}
40 {\@ifundefined{!tex@@@driver}{\ExecuteOptionsX{dvips}}
41 {\ExecuteOptionsX{dvipsone}}}%
42 \let\ExecuteOptions\ExecuteOptions@SAVE
43 \fi\fi

```

(2020/01/06) Now require url package and pass options to url through urlOpts.

```

44 \ProcessOptionsX
45 \expandafter\RequirePackage\url@0pts{url}

```

We require hyperref, eforms and soul. For eforms, a recent version is needed, 2008/03/14 or later.

```

46 \RequirePackage{hyperref}
47 \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.

```

48 \RequirePackage{eforms}[2018/08/16]
49 \RequirePackage{soul}

```

## 4 Driver Dependent Code

Driver dependent definitions for dvipsone and dvips.

```

50 \def\ps@mark{[\space}
51 \ifnum\mldblevel>0
52   \def\mlpgMsg{(\string\n Beginning of page: ) pf
53     PhysicalPage 20 string cvs
54     pf(\string\n)pf}\else
55   \def\mlpgMsg{}\fi
56 \def\pgmonitoring{\if@ml@dvips
57   dup /PhysicalPage exch 1 add def
58   /PhysicalPage PhysicalPage def^^J\fi
59   \mlpgMsg
60 }

```

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.

```

61 \def\smallRectTF{\ifoldstylequads false\else
62   \iffixmlinks true\else false\fi\fi\space
63   \ifSmallRect true\else false\fi\space and}
64 \def\ml@@nnotName{mLink} % dps
65 \def\mllnkcontainer#1{bCreateLink { xoMsgB {
66   \smallRectTF\space mlRectFix^^J%
67   #1}if}{(\ml@@nnotName\the\ae b@mLinkCnt) mlIsBldMsg}ifelse}
\pboxRect is defined in eforms
68 \def\pboxRect{mlRect }
69 \if@ml@dvips

```

**dvips driver:** Code for the dvips driver This next \special defines some standard conversion formulas, T<sub>E</sub>X to PDF and PDF to T<sub>E</sub>X for dvips.

```

70 \def\mlDict{SDict}
71 \special{!userdict begin
72   /TeXtoPDF {65536 div DVImag mul} def % sp to pts
73   /PDFtoDvips {72.27 div Resolution mul} def % points to dots
74   /PDFtoVDvips {72.27 div VResolution mul} def % points to dots
75   /DvipstoPDF {72.27 mul Resolution div} def % dots to points
76   /HTeXtoDvips {TeXtoPDF PDFtoDvips} def % sp to dots
77   /VTeXtoDvips {TeXtoPDF PDFtoVDvips} def^^J% % sp to dots
78   /PhysicalPage 0 def^^J%
79   /PageHeight {vsize} def^^J%
80   /PDFtoTeX {PDFtoDvips} def^^J%
81   /pf{print flush}def^^J%
82   /bop-hook{ \pgmonitoring\space } def
83   end}

```

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

```

84 \def\setQuadBox{%

```

```

85     currentpoint DvipstoPDF \aeb@bbox@dp\space TeXtoPDF add
86     neg vsize add 72 sub                                % y1
87     exch DvipstoPDF 72 add exch                          % x1
88     2 copy exch \aeb@bbox@wd\space TeXtoPDF add exch    % x2
89     2 copy \aeb@bbox@ht\space TeXtoPDF add              % y3
90     2 copy exch \aeb@bbox@wd\space TeXtoPDF sub exch    % x4
91 }

```

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

```

92 \def\par@@Rect
93 {%
94     72 neg PDFtoDvips vsize 72 sub PDFtoVDvips
95     hsize 72 sub PDFtoDvips 72 neg PDFtoVDvips
96 }

```

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

```

97 \else
98   \def\mDict{dvidict}
99   \special{!/TeXtoPDF {65536 div mag 1000 div mul} def
100     /PDFtoTeX {65536 mul mag 1000 div div} def^^J%
101     /pf{print flush}def^^J%
102     /bhook{ \pgmonitoring\space } def^^J%
103 }

```

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

```

104 \def\setQuadBox{%
105     currentpoint \aeb@bbox@dp\space add TeXtoPDF
106     neg PageHeight add 72 sub                                % y1
107     exch TeXtoPDF 72 add exch                                % x1
108     2 copy exch \aeb@bbox@wd\space TeXtoPDF add exch        % x2
109     2 copy \aeb@bbox@ht\space TeXtoPDF add                  % y3
110     2 copy exch \aeb@bbox@wd\space TeXtoPDF sub exch        % x4
111 }

```

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

```

112 \def\par@@Rect
113 {%
114     72 neg PDFtoTeX PageHeight 72 sub PDFtoTeX
115     PageWidth 72 sub PDFtoTeX 72 neg PDFtoTeX
116 }
117 \fi

```

## 5 The Multi-line Linking Commands

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

```

118 \newbox\ae@b@b@box
119 \newcount\ae@arrayIndx \ae@arrayIndx=0
120 \newcount\ae@mLinkCnt \ae@mLinkCnt=0
121 \newcount\syllableCnt \syllableCnt=0
122 \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.

123 \newif\ifSmallRect \SmallRecttrue
124 \AtEndDocument{\wrtmllinktot@l\ckchngmllinktot@l\wrt@linksnotformed}
125 \def\wrt@linksnotformed{\iflinknotformed
126   \PackageWarningNoLine{ae@b@link}{Some link calculations are not
127   complete.\MessageBreak
128   DO NOT CONVERT TO PDF at this time. Compile at \MessageBreak
129   least twice more}\fi}
130 \def\wrtmllinktot@l{\immediate\write\@auxout{\string\gdef
131   \string\mllinkstotal{\the\ae@mLinkCnt}}
132 \def\ckchngmllinktot@l{\ifundefined{mllinkstotal}{%
133   {\m@mllinktot@l@changed}}
134 \def\m@mllinktot@l@changed{%
135   \ifnum\mllinkstotal=\the\ae@mLinkCnt\relax\else
136   \PackageWarningNoLine{ae@b@link}{The number of links has
137   changed. Compile again\MessageBreak until this message
138   does not appear}\immediate
139   \write\@auxout{\string\mllinktotalchangedtrue}\fi
140 }
141 \def\m@mllinktotalchanged{\ifmllinktotalchanged
142   \PackageWarningNoLine{ae@b@link}
143   {The number of links has changed, continue\MessageBreak
144   to compile}\fi}
145 \AtBeginDocument{\m@mllinktotalchanged}
146 \def\CurrentBorderColor{\@linkbordercolor}
147 \def\ml@nocolorHighlight{I}
148 \def\ml@nocolorLineStyle{S}
149 \def\ml@nocolorLineWidth{1}
150 \def\ml@setnocolorDefaults{%
151 \def\ml@nocolor@defaults{\H{\ml@nocolorHighlight}}%
152   \S{\ml@nocolorLineStyle}\W{\ml@nocolorLineWidth}%
153   \Color{\CurrentBorderColor}}%
154 }
155 \ifHy@colorlinks
156   \let\ml@nocolor@defaults\@empty
157 \else
158   \ml@setnocolorDefaults
159 \fi
160 \def\ml@earlyExecProps#1{%
161   \eq@setWidgetProps\relax{#1}%
162 }

```

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.

```

163 \newif\ifoldstylequads \oldstylequadsfalse
164 \def\OldStyleBoxesOn{\mlfix0ff\oldstylequadstrue}
165 \def\OldStyleBoxesOff{\oldstylequadsfalse}
166 \@onlypreamble\OldStyleBoxesOn
167 \@onlypreamble\OldStyleBoxesOff
168 \let\mlh@preambleCmdInsert\relax
169 \def\mlcs#1{\texttt{\@backslashchar#1}}
170 \bgroup\@makeother\%
171 \gdef\CMT#1{ %\space #1}\egroup
172 \def\mldbModeOn{\def\mldb##1##2{##2}}
173 \def\mldbModeOff{\def\mldb##1##2{}}
174 \def\mldb#1#2{\ifnum#1<\mldblevel#2\fi}
175 \def\ml@adj@x{2}\def\ml@adj@y{2}
176 \def\mlMaxNSylls{30}
177 % usage \mlcrackinsat{\removelastspace}
178 \def\removelastspace{\hskip-\fontdimen2\font}
179 \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.

```

180 /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.

181 /mlIsBld 17 def^^J%
182 /mlIsBldMsg {^^J%
183   /sName exch def^^J%
184   (\string\n!! )pf
185   sName 20 string cvs pf
186   ( is not completely formed,
187   compile again!!\string\n)pf^^J%
188 } def^^J%
189 /xoMsgB true def^^J%
190 /xoMsg {^^J%
191   /Indx exch def^^J%
192   /sName exch def^^J%
193   /nSyllable Indx 8 div def^^J% dpsa08
194   (!!-----%
195   \string\n Warning:\string\n
196   The text of )pf
197   sName 20 string cvs pf

```

```

198 ( has crossed a page boundary from page )pf
199 PhysicalPage 1 sub 10 string cvs pf
200 ( to ) pf PhysicalPage 10 string cvs pf
201 sName 0 1 getinterval (m) eq {
202   (. \string\n Cross page links are not supported by the
203   PDF Specification)pf
204   (. \string\n This link is not constructed,
205   please fix it.\string\n)pf
206   (Break point is after syllable number )pf
207   nSyllable cvi 20 string cvs pf (. \string\n)pf
208   (Use the \string\mlcrackat{)pf
209   nSyllable cvi 20 string cvs pf
210   (} option with this link.\string\n)pf
211 }{
212   (. \string\n Cross page annotations are not supported by the
213   PDF Specification)pf
214   (. \string\n This annotation is not constructed,
215   please fix it.\string\n)pf
216   (Break point is after syllable number )pf
217   nSyllable cvi 20 string cvs pf (. \string\n)pf
218   (Use the mlcrackat=)pf
219   nSyllable cvi 20 string cvs pf
220   ( option with this annotation.\string\n)pf
221 } ifelse
222 (!!-----%
223 \string\n)pf^^J%
224 } def^^J%

```

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

```

225 /quadpointsfixup {^^J%
226   /ary exch def^^J%
227   /quadL exch def^^J%
228   /sName exch def^^J%
229   \mldb0{(Processing )pf sName pf (: OK\string\n)pf^^J}%
230   %\mldb0{lnkCnt 20 string cvs pf (: OK\string\n) pf^^J}%
231   quadL 0 eq {
232     (Problems with this link, length=0,
233     will skip the creation of this link)pf^^J%
234   }{

```

Begin by defining some variables needed for this operation.

```

235 /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.

236 /gY ary\space 1 gOffset add get def^^J%
237 \mldb1{(gY is ) pf gY 20 string cvs print^^J}%
238 \mldb1{flush (\string\n) pf^^J}%
239 /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.

```
240 /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.

```
241 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`.

```
242 /gMrkL 1 gOffset add def^^J%
```

```
243 \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.

```
244 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.

```
245 /gIndx exch def^^J%
```

```
246 \mldb2{(Outside gt if with gIndx=) pf^^J}%
```

```
247 \mldb2{gIndx 20 string cvs pf^^J}%
```

```
248 \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`.

```
249 /getEntry ary\space 1 gOffset add get def^^J%
```

```
250 \mldb2{(getEntry=) pf getEntry 20 string cvs pf^^J}%
```

```
251 \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.

```
252 gY getEntry sub abs 2 gt {^^J%
```

```
253 \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.

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

Increment `gMrkL` accordingly.

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

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

```
256 /gY getEntry def^^J%
```

```
257 \mldb2{(Updating gY to )pf gY 20 string cvs pf^^J}%
```

```
258 \mldb2{(\string\n) pf^^J}%
```

```
259 \mldb2{(gMrkL=)pf gMrkL 20 string cvs pf^^J}%
```

```
260 \mldb2{(\string\n) pf^^J}%
```

```

end if    End of the if
261    } if^^J%
262    /gOffset gOffset 8 add def^^J%
end for   End of the for loop
263 } for^^J%
264 \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 `\m1MaxNSylls` syllables, (currently set to 30, 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 `\m1MaxNSylls` being too small for some of your sentences; in this case, redefine `\m1MaxNSylls` to a larger value.

```

265    /gAryL 8 \m1MaxNSylls\space mul def^^J% limitation
266 \mldb2{(gAryL=) pf gAryL 20 string cvs pf^^J}%
267 \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.

```

268    /gFixup 8 gMrkL mul array def^^J% links
269    /aFixup 8 gMrkL mul array def^^J% text markup annotations
270    /gOffset 0 def^^J%
271 \mldb2{(for loop: gMrkL=)pf gMrkL 20 string cvs pf^^J}%
272 \mldb2{(\string\n) pf^^J}%

```

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

```

273 0 1 gMrkL 1 sub {^^J%
274 \mldb2{(After gAry\string\n) pf^^J}%
275 \mldb2{(Top of for loop\string\n) pf^^J}%
276    /gIndx exch def^^J%
277 \mldb2{(gIndx=)pf gIndx 20 string cvs pf^^J}%
278 \mldb2{(\string\n)pf^^J}%
279 \mldb2{(gMrk=)pf gMrk gIndx get 20 string cvs pf^^J}%
280 \mldb2{(\string\n) pf^^J}%
281 \mldb2{(mLinkFxup<num> length = )^^J}%
282 \mldb2{pf ary\space length 20 string cvs^^J}%
283 \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.

```

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

```

```

286 }{^^J%
287 /gCount gMrk gIndx 1 add get gMrk gIndx get sub def^^J%
288 } ifelse^^J%

```

Declare the gAry array

```

289 /gAry gAryL array def^^J%
290 \mldb2{(gCount=)pf gCount 20 string cvs pf^^J}%
291 \mldb2{(\string\n)pf^^J}%

```

We want to copy a slice of mLinkFxup(*num*), we declare the next array of length gCount just computed.

```

292 /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.

```

293 sliceOfLinkfxup 0 ary gMrk gIndx get^^J%
294 gCount getinterval putinterval^^J%
295 gAry 0 sliceOfLinkfxup putinterval^^J%
296 \mldb1{(Listing elements of gFixup\string\n) pf^^J}%
297 \mldb1{gAry 0 get 20 string cvs pf (\string\n)pf^^J}%
298 \mldb1{gAry 1 get 20 string cvs pf (\string\n)pf^^J}%
299 \mldb1{gAry gCount 1 sub 5 sub get
300 20 string cvs pf (\string\n)pf^^J}%
301 \mldb1{gAry gCount 1 sub 4 sub get
302 20 string cvs pf (\string\n)pf^^J}%
303 \mldb1{gAry gCount 1 sub 3 sub get
304 20 string cvs pf (\string\n)pf^^J}%
305 \mldb1{gAry gCount 1 sub 2 sub get
306 20 string cvs pf (\string\n)pf^^J}%
307 \mldb1{gAry 6 get 20 string cvs pf (\string\n)pf^^J}%
308 \mldb1{gAry 7 get 20 string cvs pf (\string\n)pf^^J}%
309 gFixup gOffset [^^J%
310 gAry 0 get ^^J% x1 ll 1
311 gAry 1 get^^J% y1 ll 1
312 gAry gCount 1 sub 5 sub get^^J% x2 lr 2
313 gAry gCount 1 sub 4 sub get^^J% y2 lr 2
314 gAry gCount 1 sub 3 sub get^^J% x3 ur 3
315 gAry gCount 1 sub 2 sub get^^J% y3 ur 3
316 gAry 6 get^^J% x4 ul 4
317 gAry 7 get^^J% y4 ul 4
318 ] 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

```

319 aFixup gOffset [^^J%
320 gAry 6 get gOffset 0 eq {\ml@adj@x\space add}if^^J% x4 ul 4
321 gAry 7 get^^J% y4 ul 4
322 gAry gCount 1 sub 3 sub get^^J% x3 ur 3
323 gAry gCount 1 sub 2 sub get^^J% y3 ur 3

```

```

324      gAry 0 get gOffset 0 eq {\ml@adj@x\space add}if^^J% x1 ll 1
325      gAry 1 get 1 sub^^J%                                y1 ll 1
326      gAry gCount 1 sub 5 sub get^^J%                    x2 lr 2
327      gAry gCount 1 sub 4 sub get 1 sub^^J%              y2 lr 2
328    ] putinterval^^J%
329    /gOffset gOffset 8 add def^^J%
330  } for^^J%
331 \mldb2{(End of second for\string\n) pf^^J}%
332  } ifelse
333 } 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.

```

334 /smallquadpointsfixup {^^J%
335 /gIndx exch def^^J%
336 /ary exch def^^J%
337 /lnkCnt exch def^^J%
338 /quadL exch def^^J%
339 /gSFup 8 array def^^J%
340 gSFup 0 ary 0 gIndx add get
341 gIndx 0 eq {\ml@adj@x\space sub} if put^^J% x1
342 gSFup 1 ary 1 gIndx add get put^^J%          y1
343 gSFup 2 ary 2 gIndx add get put^^J%          x2
344 gSFup 3 ary 3 gIndx add get put^^J%          y2
345 gSFup 4 ary 4 gIndx add get put^^J%          x3
346 gSFup 5 ary 5 gIndx add get put^^J%
347 gSFup 6 ary 6 gIndx add get
348 gIndx 0 eq {\ml@adj@x\space sub} if put^^J% x4
349 gSFup 7 ary 7 gIndx add get put^^J%
350 ary gIndx gSFup putinterval^^J%
351 } def^^J%

```

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

```

352 /mlRectFix {^^J%
353 /ifRectFix exch def^^J%
354 ifRectFix {
355   /nL gFixup length 8 sub def^^J% number of lines
356   /xMin gFixup 0 get def^^J%
357   0 8 nL {^^J%
358     /Indx exch def^^J%
359     gFixup Indx get xMin lt {/xMin gFixup Indx get def}if } for^^J%
360     /xMin xMin 2 sub def^^J%
361     /xMax gFixup 2 get def^^J%
362     2 8 nL 2 add {^^J%
363       /Indx exch def^^J%
364       gFixup Indx get xMax gt {/xMax gFixup Indx get def}if } for^^J%
365       /xMax xMax 2 add def^^J%
366       /yMin gFixup 1 get def^^J%
367       1 8 nL 1 add {^^J%

```

```

368 /Indx exch def^^J%
369 gFixup Indx get yMin lt {/yMin gFixup Indx get def}if } for^^J%
370 /yMin yMin 4 sub def^^J%
371 /yMax gFixup 5 get def^^J%
372 5 8 nL 5 add{^^J%
373 /Indx exch def^^J%
374 gFixup Indx get yMax gt {/yMax gFixup Indx get def}if } for^^J%
375 /yMax yMax 2 add def^^J%
376 /mlRect {/Rect [^^J%
377   xMin 72 sub PDFtoTeX^^J%
378   PageHeight 72 sub yMax sub PDFtoTeX^^J%
379   xMax 72 sub PDFtoTeX^^J%
380   PageHeight yMin sub 72 sub PDFtoTeX ]^^J%
381 }def^^J%
382 }{^^J%
383 /mlRect{/Rect [ \par@@Rect ] }def^^J%
384 }ifelse^^J%
385 ifRectFix {
386 \mldb1{(/Rect [])pf^^J%
387 xMin 20 string cvs pf( )pf^^J%
388 yMax 20 string cvs pf( )pf^^J%
389 xMax 20 string cvs pf( )pf^^J%
390 yMin 20 string cvs pf(]\string\n)pf^^J}%
391 } if^^J%
392 } def
393 }}

```

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

```

394 \newif\ifmlmarks\mlmarksfalse
395 \def\mlMarksOn{\mlmarkstrue}
396 \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.

```

397 \def\MrkLnkLtr{L}
398 \def\ml@MrkLnk#1{\ifmlmarks\bgroup\ifdim\baselineskip=0pt
399   \setbox\z@\hbox{T}\gdef\ml@raiseamt{\ht\z@+.4pt}\else
400   \gdef\ml@raiseamt{.6\baselineskip}\fi\smash{\rlap{\normalfont
401     \normalcolor\bfseries
402     \raisebox{\ml@raiseamt}{\tiny\strut{\MrkLnkLtr#1}}}}\egroup\fi}
403 \newif\iflinknotformed\linknotformedfalse
404 \newif\ifcr@ckit\cr@ckitfalse
405 \def\ml@underlined{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.

$\langle \textit{opts} \rangle$  (Optional) A standard optional argument for `eforms` to change the appearance of the link and/or to include actions.

$\langle \textit{text} \rangle$  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`.

```

406 \ifundefined{mlhypertext}{\newcommand}{\renewcommand}%
407 {\mlhypertext}[2][\hglueOpt\beginngroup
408   \global\ml@displaytrue
409   \toks@=\expandafter{#2}%
410   \edef\ml@HytexArg{\the\toks@}%
411   \global\ae@arrayIndx=\z@
412   \def\ml@setlink##1{\setLinkPbox{%
413     \QuadPoints{mLink##1}#1}}%
414   \expandafter\processAppArgs\set@LinkPboxDefaults
415   \presets{\ml@nocolor@defaults}\S{S}\W{0}#1\end\@nil
416   \ifx\eq@S@value\ml@underlined
417     \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.

```

418   \ifx\eq@mignore\ef@YES
419     \global\advance\ae@mLinkCnt\@ne\relax
420     \def\ml@next{\mlhypertext@i{#1}}\else
421     \ifx\eq@mlcrackat\@empty
422       \global\advance\ae@mLinkCnt\@ne\relax
423       \def\ml@next{\mlhypertext@i{#1}}%
424     \else % \eq@mlcrackat not \@empty

```

A ‘normal’ link, continue

**Crack it up.** We define `\ml@next` to call `\mlhypertext` consecutively; the first time we specify `\mignore{0}` to indicate this link is the first part, and then `\mignore{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.

```

425   \def\ml@next{\global\ml@displaytrue\let\ml@space\space
426   \mlhypertext[#1\mignore{0}]{#2}\eq@mlcrackinsat
427   \penalty-100 \cr@ckittrue

```

```

428     \global\ml@displayfalse\let\ml@space\space
429     \mlhypertext[#1\mlignore{1}]{#2}\aftergroup
430     \normalcolor\endgroup}%
431   \fi
432 \fi
433 \ml@next
434 }
435 \def\mlhypertext@i#1{%
436   \ifundefined{mLinkLngh\the\aeB@mLinkCnt}{\global
437     \linknotformedtrue\def\ml@lngh{17}}
438   {\edef\ml@lngh{\@nameuse{mLinkLngh\the\aeB@mLinkCnt}}}%
439   \ml@start@link{\the\aeB@mLinkCnt}{\ml@lngh}% Step 1
440   \def\mlh@preambleCmdInsert{%
441     \ml@MrkLnk{\the\aeB@mLinkCnt}\ml@earlyExecProps{#1}}%
442   \def\mlh@postambleCmd{\endgroup}%
443   Start soul on \mlhypertext (\aeB@mlh)
444   \expandafter\aeB@mlh\ml@HytexArg
445   \ml@finish@link{\the\aeB@mLinkCnt}{\ml@lngh}%
446   \ml@setlink{\the\aeB@mLinkCnt}%
447   \ifoldstylequads\else
448     \iffixmlinks\literalps@out{restore}\fi\fi
449     \ifundefined{mLinkLngh\the\aeB@mLinkCnt}{%
450       \immediate\write\@auxout{\string\mlcsarg
451         \string\gdef{mLinkLngh\the\aeB@mLinkCnt}{17}}%
452       }{\immediate\write\@auxout{\string\mlcsarg
453         \string\gdef{mLinkLngh\the\aeB@mLinkCnt}%
454         {\the\aeB@arrayIndx}}}\endgroup
455     }
456   \def\mlh@setQuadSyllable#1#2#3#4{%
457     % #1 = current array Index
458     % #2 = quad total
459     % #3 = link cnt
460     % #4 = content
461     \setbox\aeB@bbox=\hbox{\ml@strut#4}%
462     {%
463       \setbox\@tempboxa\hbox{\ml@strut}%
464       \ifx\itsunderline\ef@YES\@tempdima1bp\relax\else
465         \@tempdima\dp\@tempboxa \ifdim\@tempdima>2bp
466           \advance\@tempdima-2bp\fi
467         \fi
468         \ifx\isstrikeout\ef@YES\advance\@tempdima-2bp\fi
469         \dp\aeB@bbox\@tempdima
470         \@tempdima\ht\@tempboxa \advance\@tempdima\dp\aeB@bbox
471         \advance\@tempdima1bp
472         \ht\aeB@bbox\@tempdima\def\x{\the\count\z@}%
473         \count\z@=\ht\aeB@bbox\xdef\aeB@bbox@ht{\x}%
474         \count\z@=\wd\aeB@bbox\xdef\aeB@bbox@wd{\x}%
475         \count\z@=\dp\aeB@bbox\xdef\aeB@bbox@dp{\x}%

```

```

476 \ifoldstylequads
477 \literalps@out{%
478   bCreateLink {^^J%
479     \mDict\space/mLinkFxup#3\space known {^^J%
480     \ps@mark{mLink#3}
481     \the\aeB@arrayIndx\space [\setQuadBox]
482     \space /PUTINTERVAL pdfmark}{^^J%
483     xMsgB {
484       /xMsgB false def
485       (\mL@nnotName\the\aeB@arrayIndx)
486       #3\space
487       xMsg % dpsa08
488     } if^^J% xMsgB
489     } ifelse } if
490   }%
491 \else
492   \literalps@out{%
493     bCreateLink {^^J%
494       \mDict\space/mLinkFxup#3\space known {^^J%
495       mLinkFxup#3\space
496       #1\space[\setQuadBox] putinterval^^J%
497       Initiate fix up of little rectangles
498       #2\space %
499       #3\space
500       mLinkFxup#3\space
501       #1\space
502       smallquadpointsfixup }{^^J%
503       xMsgB {
504         /xMsgB false def
505         (\mL@nnotName#3)
506         #1\space
507         xMsg % dpsa08
508       } if^^J% xMsgB
509     } ifelse } if
510   }%
511 \fi
512 \global\advance\aeB@arrayIndx8\relax

```

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

```

513 \def\aeB@exiii{\expandafter\expandafter\expandafter}
514 \def\aeBnameref#1{\@ifundefined{r@#1}{??}
515   {\aeB@exiii\@thirdoffive\csname r@#1\endcsname}}
516 \def\labelRef#1{\@ifundefined{r@#1}{Doc-Start}
517   {\aeB@exiii\@fourthoffive\csname r@#1\endcsname}}
518 \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⟩}`

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

Syntax: `\mlhyperref[⟨opts⟩]{⟨label⟩}{⟨text⟩}`

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

Syntax: `\mlnameref[⟨opts⟩]{⟨label⟩}`

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

We use a work around to a `\relax` problem encountered in the `\mlNameRef` command. `LATEX` 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⟩}`

```
526 \newcommand\mlNameref[2] [] {\let\SOUL@stop\ml\SOUL@stop
527   \protected@edef\ml@temp{'\aebnameref{#2}' on page~\atPage{#2}}%
528   \def\ml@tempi{\mlhypertext[#1\A{/S/GoTo/D (\labelRef{#2})}]}%
529   \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`.

```
530 \def\ml@hyper@linkurl#1#2{\hyper@chars
531   \let\ef@thislinkcolor\@urlcolor
532   \let\CurrentBorderColor\@urlbordercolor
533   \mlhypertext[\presets{\mlhref@args}\A{/S/URI/URI(#2)}]{#1}%
534   \endgroup
535 }
536 \def\ml@hyper@linkfile#1#2#3{%
537   \let\ef@thislinkcolor\@filecolor
538   \let\CurrentBorderColor\@filebordercolor
539   \ifx\@pdfstartview\@empty
540     \def\theView{[0 /Fit]} \else
```

```

541 \def\theView{[0 \@pdfstartview]}\fi
542 \@ifundefined{ifHy@pdfnewwindow}
543 {\ifHy@newwindow}{\ifHy@pdfnewwindow}%
544 \def\isWindow{/NewWindow true}\else
545 \let\isWindow\empty\fi
546 \mlhypertext[\presets{\mlhref@args}\A{/S/GoToR \isWindow
547 /F (#2) /D \ifx\#3\theView\else(#3)\fi}]{#1}%
548 \endgroup
549 }
550 \def\ml@hyper@launch run:#1\#2#3{%
551 \let\ef@thislinkcolor\@filecolor
552 \let\CurrentBorderColor\@runbordercolor
553 \@ifundefined{ifHy@pdfnewwindow}
554 {\ifHy@newwindow}{\ifHy@pdfnewwindow}%
555 \def\isWindow{/NewWindow true}\else
556 \let\isWindow\empty\fi
557 \mlhypertext[\presets{\mlhref@args}\A{/S/Launch\isWindow
558 /F (#1) \ifx\#3\else /Win << /P (#3) /F (#1) >> \fi}]{#2}%
559 \endgroup
560 }

```

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.

```
561 \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>}`

```

562 \newcommand{\mlhref}[1][{}]{%
563 \begingroup
564 \def\mlhref@args{#1}%
565 \let\hyper@linkurl\ml@hyper@linkurl
566 \let\hyper@linkfile\ml@hyper@linkfile
567 \let\@hyper@launch\ml@hyper@launch
568 \ae@saved@href
569 }

```

**\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.

```
570 \newbox\ml@urlbuildi
```

571 \newbox\ml@urlbuildii

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

```

572 \def\ml@SOUL@doword{%
573   \global\setbox\ml@urlbuildi\hbox{}%
574   \global\setbox\ml@urlbuildii\hbox{}%
575   \edef\x{\the\SOUL@word}%
576   \ifx\x\empty
577   \else
578     \SOUL@buffer={}%
579     \setbox\z@\vbox{%
580       \SOUL@tt
581       \hyphenchar\font'\-
582       \hfuzz\maxdimen
583       \hbadness\@M
584       \pretolerance\m@ne
585       \tolerance\@M
586       \leftskip\z@
587       \rightskip\z@
588       \hsize1sp
589       \everypar{}%
590       \parfillskip\z@\@plus1fil
591       \hyphenpenalty-\@M
592       \noindent
593       \hskip\z@
594       \relax
595       \the\SOUL@word}%

```

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

```

596   \let\SOUL@errmsg\relax
597 %   \let\SOUL@errmsg\SOUL@error
598   \let\-\relax
599   \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.

```

600   \ml@SOUL@analyze
601   \SOUL@word={}%
602   \fi
603 }

```

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

```

604 \def\ml@SOUL@analyze{%
605   \setbox\z@\vbox{%
606     \unvcopy\z@
607     \unskip
608     \unpenalty
609     \global\setbox\@ne=\lastbox}%
610   \ifvoid\@ne
611   \else

```

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

```

612 \ml@interface@analyze
613 \SOUL@syllgoal=\wd\@ne
614 \advance\count@\@ne

```

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

```

615 \ml@SOUL@analyze
616 \SOUL@syllwidth\z@
617 \SOUL@syllable={}%
618 \ifnum\count@>\z@
619 \advance\SOUL@syllgoal-\SOUL@ttwidth

```

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

```

620 \ml@SOUL@dosyllable
621 \SOUL@getkern{\the\SOUL@lasttoken}{\SOUL@hyphkern}%
622 {\SOUL@sethyphenchar}%
623 \SOUL@everyhyphen
624 \else

```

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

```

625 \ml@SOUL@dosyllable
626 \fi
627 \fi
628 }}

```

```

629 \newif\ifml@display \ml@displaytrue

```

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

```

630 \def\ml@SOUL@dosyllable{%
631 \SOUL@gettoken
632 \SOUL@eventuallyexhyphen{\the\SOUL@token}%
633 \edef\x{\the\SOUL@token}%
634 \ifx\x\SOUL@hyphenhintM
635 \let\SOUL@n\ml@SOUL@dosyllable
636 \else\ifx\x\SOUL@lowerthanM
637 \SOUL@gettoken
638 \SOUL@getkern{\the\SOUL@lasttoken}{\SOUL@charkern}
639 {\the\SOUL@token}%
640 \SOUL@everylowerthan
641 \SOUL@puttoken
642 \let\SOUL@n\ml@SOUL@dosyllable
643 \else\ifdim\SOUL@syllwidth=\SOUL@syllgoal
644 \SOUL@every syllable
645 \SOUL@puttoken
646 \let\SOUL@n\relax
647 \else
648 \ifx\x\SOUL@stopM
649 \SOUL@errmsg
650 \global\let\SOUL@errmsg\relax
651 \let\SOUL@n\relax
652 \else
653 \setbox\tw@\hbox{\SOUL@tt\the\SOUL@token}%

```

```

654      \advance\SOUL@syllwidth\wd\tw@
655      \global\SOUL@lasttoken=\SOUL@token
656      \SOUL@gettoken
657      \SOUL@getkern{\the\SOUL@lasttoken}{\SOUL@charkern}
658      {\the\SOUL@token}%
659      \SOUL@puttoken
660      \global\SOUL@token=\SOUL@lasttoken
661      \SOUL@everytoken
662      \edef\x{\SOUL@syllable={\the\SOUL@syllable\the\SOUL@token}}\x

  Here is the only change, we direct flow back to \ml@SOUL@dosyllable
663      \let\SOUL@n\ml@SOUL@dosyllable
664      \fi\fi\fi\fi
665      \SOUL@n
666 }

```

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

```

667 \def\ml@interface@analyze{%
668   \global\advance\syllableCnt\@ne % dpsa11
669   \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.

```

670   \ifx\eq@mlcrackat\@empty

  Ordinary link, we don't crack it apart, continue with the old code.
671     \ml@bld@quadchunks{\the\@eb@mLinkCnt}
672     {\ml@qlngthchunki}{\ml@urlbuildi}%
673   \else
674     \ifnum\syllableCnt=\revCrackAt\relax

```

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

```

675     \@eb@arrayIndx=\z@

```

```

start first link
676      \ml@start@link{\the\aeB@mLinkCnt}{\ml@qlngthchunki}%
677      \ml@bld@quadchunks{\the\aeB@mLinkCnt}
678      {\ml@qlngthchunki}{\ml@urlbuildi}%
679      \else
Continue with first link
680      \ifnum\syllableCnt>\revCrackAt\relax
681      \ml@bld@quadchunks{\the\aeB@mLinkCnt}
682      {\ml@qlngthchunki}{\ml@urlbuildi}%
683      \else
684      \ifnum\syllableCnt=\@ne
start second link
685      \aeB@arrayIndx=\z@
Everything is in reverse order, this is the last syllable of the second chunk
686      \ml@start@link{\aeB@mLinkCnt@}{\ml@qlngthchunkii}%
687      \ml@bld@quadchunks{\aeB@mLinkCnt@}
688      {\ml@qlngthchunkii}{\ml@urlbuildii}%
689      \else
continue with second link
690      \ml@bld@quadchunks{\aeB@mLinkCnt@}
691      {\ml@qlngthchunkii}{\ml@urlbuildii}%
692      \fi
693      \fi
694      \fi
695      \fi
696  }
697  \def\ml@bld@quadchunks#1#2#3{%
698  % #1 = link cnt
699  % #2 = chunk size
700  % #3 = \ml@urlbuild to i or ii
701  \@ifundefined{mLinkLngth\the\aeB@mLinkCnt}
702  {\edef\@indx{\the\aeB@arrayIndx}}
703  {\@tempcnta#2\relax
704   \advance\@tempcnta-8\relax
705   \advance\@tempcnta-\aeB@arrayIndx
706   \def\@indx{\the\@tempcnta}%
707  }%
708  \@tempcntb=\@indx \divide\@tempcntb by 8
709  \advance\@tempcntb by 1
710  \ifx#3\ml@urlbuildii \advance\@tempcntb by \eq@mlcrackat\relax\fi
711  \global\setbox#3=\hbox{%
712   \mlh@setQuadSyllable{\@indx}{#2}{#1}{\unhcopy\@ne}%
713   \hbox{\unhcopy\@ne}\relax
714   \ml@typeset@csyl{\@tempcntb}\penalty0\unhcopy#3}%hbox
715  }}

```

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

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

```
716 \newcommand{\mlurl}{\begingroup\makeother\~\relax% \def~{\string~}%
717   \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.

```
718 \newcommand{\mlurl@}[2][\ifundefined{ef@thislinkcolor}
719   {\let\ef@thislinkcolor\normalcolor}\expandafter
720   \def\expandafter\ef@thislinkcolor@SAVE
721   \expandafter{\ef@thislinkcolor}%
722   \def\ml@setlink##1{\setLinkPbox{\A{\URI{#2}}}%
723   \QuadPoints{mLink##1}#1}}%
```

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

```
724   \expandafter\processAppArgs\set@LinkPboxDefaults
725   \presets{\ml@nocolor@defaults}\S{S}\W{0}#1\end\@nil
726   \ifx\eq@S@value\ml@underlined
727     \let\itsunderline\ef@YES\else\let\itsunderline\ef@NO\fi
```

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

```
728   \global\ae@arrayIndx=0\relax
729   \global\syllableCnt=0\relax
730   \global\advance\ae@mLinkCnt\@ne\relax
731   \ifundefined{mLinkLngth\the\ae@mLinkCnt}{%
732     \global\linknotformedtrue\def\ml@lngth{17}%
733     \def\ml@qlngthchunki{17}}
734   {\@tempcnta\@nameuse{mLinkLngth\the\ae@mLinkCnt}\relax
735   \edef\ml@lngth{\the\@tempcnta}%\multiply\@tempcnta8\relax
736   \edef\ml@qlngthchunki{\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.

```
737   \ifundefined{mLinkSyCnt\the\ae@mLinkCnt}
738     {\def\mlM@XCNT{0}\def\ml@lngthchunki{0}\def\ml@lngthchunkii{0}%
739     \def\ml@qlngthchunki{0}\def\ml@qlngthchunkii{0}\def\revCrackAt{0}}
740     {\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`.

```
741   \ifx\eq@mlcrackat\empty\else
742     \@tempcnta\mlM@XCNT\relax\advance\@tempcnta-\eq@mlcrackat\relax
743     \edef\ml@lngthchunki{\eq@mlcrackat}%
744     \edef\ml@lngthchunkii{\the\@tempcnta}%
745     \@tempcnta\ml@lngthchunki\relax\multiply\@tempcnta8\relax
746     \edef\ml@qlngthchunki{\the\@tempcnta}%
747     \@tempcnta\ml@lngthchunkii\relax\multiply\@tempcnta8\relax
748     \edef\ml@qlngthchunkii{\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.

```

749      \@tempcnta\mLM@XCNT\relax
750      \advance\@tempcnta-\eq@mlcrackat\relax
751      \advance\@tempcnta\@ne
752      \edef\revCrackAt{\the\@tempcnta}%
753      \fi}%
754      \@tempcnta\@eb@mlinkCnt\advance\@tempcnta\@ne
755      \edef\@eb@mlinkCnt{\the\@tempcnta}%
756      \urldef\ml@url\nolinkurl{#2}%
757      \def\SOUL@mlhpreamble{\% \begingroup
758      \mlh@preambleCmdInsert\ef@colorthislink}\hyper@chars
759      \let\ef@thislinkcolor\@urlcolor
760      \let\CurrentBorderColor\@urlbordercolor

```

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

```

761      \let\ml@SOUL@doword@SAVE\SOUL@doword
762      \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.

```

763      \ifx\eq@mlcrackat\@empty
764      \ml@start@link{\the\@eb@mlinkCnt}{\ml@length}\fi
765      \def\mlh@preambleCmdInsert{\ml@MrkLnk{\the\@eb@mlinkCnt}%
766      \ml@earlyExecProps{#1}}%
767      \def\mlh@postambleCmd{%
768      \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.

```

769      \def\expandafter\ef@thislinkcolor
770      \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.

```

771      \@eb@mlh\ml@url\ef@colorthislink\unhcopy\ml@urlbuildi
772      \expandafter\gdef\expandafter\ef@thislinkcolor
773      \expandafter{\ef@thislinkcolor@SAVE}%
774      \immediate\write\@auxout{\string\mlcsarg\string
775      \gdef{mLinkSyCnt\the\@eb@mlinkCnt}{\the\syllableCnt}}% dpsa11
776      \immediate\write\@auxout{\string\mlcsarg
777      \string\gdef{mLinkLength\the\@eb@mlinkCnt}{\the\@eb@arrayIndx}}%
778      \ifx\eq@mlcrackat\@empty
779      \ml@finish@link{\the\@eb@mlinkCnt}{\ml@length}%
780      \ml@setlink{\the\@eb@mlinkCnt}%
781      \iffixmlinks\literalps@out{restore}\fi

```



```

782 \else
783 \ml@finish@link{\the\aeB@mLinkCnt}{\ml@qlngthchunki}%
784 \ml@setlink{\the\aeB@mLinkCnt}%
785 \iffixmlinks\literalps@out{restore}\fi
786 \eq@mlcrackinsat\penalty-100
787 \ml@start@link{\aeB@mLinkCnt@}{\ml@qlngthchunkii}%
788 \penalty0\@ifundefined{mLinkLngth\the\aeB@mLinkCnt}{%
789 {\ml@MrkLnk{\aeB@mLinkCnt@}}\unhcopy\ml@urlbuildii
790 \ml@finish@link{\aeB@mLinkCnt@}{\ml@qlngthchunkii}%
791 \ml@setlink{\aeB@mLinkCnt@}\iffixmlinks
792 \literalps@out{restore}\fi
793 \global\advance\aeB@mLinkCnt\@ne
794 \fi\aftergroup\normalcolor\endgroup
795 }
796 \def\ml@start@link#1#2{% Step 1
797 % #1=link number
798 % #2 = final quad length
799 \literalps@out{%
800 /xMsgB true def^^J%
801 \ps@mark/_objdef {mLink#1}%
802 /type /array /OBJ pdfmark^^J%
803 \ifoldstylequads
804 /mLinkFxup#1\space0 array def
805 \else
806 /bCreateLink mLinkBld #2\space eq not def^^J%
807 bCreateLink{ /mLinkFxup#1\space
808 #2\space array def }if
809 \fi
810 }%
811 }

```

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

```

812 \def\FixupProc{gFixup}
813 \def\ml@finish@link#1#2{% Step 4 and 5
814 % #1=link number
815 % #2 = final quad length
816 \ifoldstylequads\else
817 \ifnum\aeB@arrayIndx=0\relax
818 \PackageWarning{aeB_mlink}{%
819 Problem with mLink\the\aeB@mLinkCnt, Will skip the \MessageBreak
820 creation of this link}\fi
821 \literalps@out{%
822 \iffixmlinks
823 \ifnum\aeB@arrayIndx>0
824 save^^J%
825 bCreateLink {^^J%
826 \mlDict\space/mLinkFxup#1\space known {^^J%
827 (\ml@nnotName#1)

```

```

828      #2\space
829      mLinkFxup#1\space
830      quadpointsfixup^^J%
831      \ps@mark{mLink#1} 0 \FixupProc\space
832      /PUTINTERVAL pdfmark^^J%
833      }if }if
834      \fi
835      \else

(2018/04/20) Added bCreateLink, etc., to protect against distiller/ps2pdf from
crashing when \mlfix{n}.

836      bCreateLink {^^J%
837      \mDict\space/mLinkFxup#1\space known {^^J%
838      \ps@mark{mLink#1} 0 mLinkFxup#1
839      /PUTINTERVAL pdfmark^^J%
840      }if }if
841      \fi}%
842 \fi}

```

## 6 Macros used by the SOUL Interface

```

843 \ifHy@colorlinks
844   \def\ef@colorthislink{\color{\ef@thislinkcolor}}
845 \else
846   \let\ef@colorthislink\relax
847 \fi

```

I've inserted `\let\protect\@empty` to make `mlnameref` and `mlNameref` work.

```

848 \def\ml@SOUL@stop{\relax}
849 \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.

850   \mlh@preambleCmdInsert\ef@colorthislink}
851 \def\SOUL@mlheverysyllable{% dpsaug16
852   \global\advance\syllableCnt\@ne
853   \ifx\eq@mlcrackat\@empty
854     \expandafter\SOUL@mlheverysyllable@i
855   \else
856     \expandafter\SOUL@mlheverysyllable@ii
857   \fi
858 }
859 \let\eq@mlhyph\@empty
860 \def\ml@typeset@syl#1{\raisebox{\ml@raiseamt}
861   {\smash{\normalfont\normalcolor\tiny\strut\llap{\the#1}}}}
862 \def\turnSyllbCntOn{\mlMarksOn\let\ml@typeset@syl\ml@typeset@syl}
863 \def\turnSyllbCntOff{\let\ml@typeset@syl\@gobble}
864 \turnSyllbCntOff
865 \def\SOUL@mlheverysyllable@i{%
866   \mlh@setQuadSyllable{\the\aeB@arrayIndx}{\ml@lngth}%

```

```

867     {\the\aeB@mLinkCnt}{\the\SOUL@syllable\eq@@mlhyph}%
868     \the\SOUL@syllable %\SOUL@setkern\SOUL@charkern
869     \ml@typeset@@syl{\syllableCnt}\eq@@mlhyph
870 }%
871 \def\SOUL@mleveryssyllable@ii{%
872     \ifnum\eq@mlchunk=0\relax
873     \ifnum\syllableCnt>\eq@mlcrackat\relax
874         \global\ml@displayfalse
875     \else
876         \global\ml@displaytrue
877         \ifnum\syllableCnt=\eq@mlcrackat\relax
878             \let\eq@@mlhyph\eq@mlhyph
879             \global\let\ml@space\relax
880         \else
881             \let\eq@@mlhyph\@empty
882             \global\let\ml@space\space
883         \fi
884         \SOUL@mleveryssyllable@i
885     \fi
886 \else
887     \ifnum\syllableCnt>\eq@mlcrackat\relax
888         \global\ml@displaytrue
889         \SOUL@mleveryssyllable@i
890     \else
891         \global\ml@displayfalse
892     \fi
893 \fi
894 \ml@dynamicsetup
895 }
896 %\def\SOUL@mleverysspace#1{#1\space\hskip\spaceskip}
897 \let\ml@space\space
898 \def\SOUL@mleverysspace#1{%
899     #1\ml@space\global\let\ml@space\space
900 }
901 \def\SOUL@mleveryhyphen{%
902     \discretionary{%
903         \unkern
904         \SOUL@setkern\SOUL@hyphkern
905         \SOUL@sethyphenchar
906     }{}{}%
907 }
908 \def\SOUL@mleveryexhyphen#1{\global\advance\syllableCnt\@ne
909     \mlh@setQuadSyllable{\the\aeB@arrayIndx}{\ml@length}%
910     {\the\aeB@mLinkCnt}{\SOUL@setkern\SOUL@hyphkern#1}%
911     \SOUL@setkern\SOUL@hyphkern\hbox{#1}%
912     \discretionary{}{}{}%
913     \SOUL@setkern\SOUL@charkern
914 }%
915 }
916 \def\mlh@postambleCmd{\relax}

```

```

917 \def\ml@dynamicsetup{%dpsaug16
918   \ifml@display
919     \global\let\SOUL@everySPACE\SOUL@mlheverySPACE
920     \global\let\SOUL@everyexhyphen\SOUL@mlheveryexhyphen
921   \else
922     \gdef\SOUL@everySPACE##1{%
923     \gdef\SOUL@everyexhyphen##1{%
924   \fi
925 }
926 \def\SOUL@mlhpostamble{\mlh@postambleCmd}
927 \def\SOUL@mlhsetup{\SOUL@setup
928   \let\SOUL@preamble\SOUL@mlhpreamble
929   \let\SOUL@everySYLLABLE\SOUL@mlheverySYLLABLE
930   \ml@dynamicsetup
931 %   \let\SOUL@everySPACE\SOUL@mlheverySPACE
932   \let\SOUL@everyhyphen\SOUL@mlheveryhyphen
933 %   \let\SOUL@everyexhyphen\SOUL@mlheveryexhyphen
934   \def\SOUL@postamble{\SOUL@mlhpostamble}%
935 }
936 \DeclareRobustCommand*\aeb@mlh{\syllableCnt=0
937   \SOUL@mlhsetup\SOUL@}
938 % End of Package
939 \end{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> .....	170
<code>\@auxout</code> .....	130, 139, 449, 451, 774, 776
<code>\@backslashchar</code> .....	169
<code>\@filebordercolor</code> .....	538
<code>\@filecolor</code> .....	537, 551
<code>\@fourthoffive</code> .....	517
<code>\@hyper@launch</code> .....	567
<code>\@indx</code> .....	702, 706, 708, 712
<code>\@linkbordercolor</code> .....	146
<code>\@makeoother</code> .....	170, 716
<code>\@mldvipsfalse</code> .....	18
<code>\@mldvipstrue</code> .....	16, 22
<code>\@onlypreamble</code> .....	166, 167
<code>\@pdfstartview</code> .....	539, 541
<code>\@plus</code> .....	590
<code>\@runbordercolor</code> .....	552
<code>\@tempboxa</code> .....	462, 464, 469
<code>\@thirdoffive</code> .....	515
<code>\@urlbordercolor</code> .....	532, 760
<code>\@urlcolor</code> .....	531, 759
<code>\~</code> .....	716
<b>A</b>	
<code>\A</code> .....	520, 522, 524, 528, 533, 546, 557, 722
<code>\aeb@arrayIndx</code> ....	119, 411, 453, 481, 485, 511, 675, 685, 702, 705, 728, 777, 817, 823, 866, 909
<code>\aeb@bbox</code> .....	118, 460, 468, 469, 471–474
<code>\aeb@bbox@dp</code> .....	85, 105, 474
<code>\aeb@bbox@ht</code> .....	89, 109, 472
<code>\aeb@bbox@wd</code> .....	88, 90, 108, 110, 473
<code>\aeb@exiii</code> .....	513, 515, 517
<code>\aeb@mlh</code> .....	443, 771, 936
<code>\aeb@mLinkCnt</code> .....	67, 120, 131, 135, 419, 422, 436, 438, 439, 441, 444, 445, 448, 450, 452, 671, 676, 677, 681, 701, 730, 731, 734, 737, 740, 754, 764, 765, 775, 777, 779, 780, 783, 784, 788, 793, 819, 867, 910
<code>\aeb@mLinkCnt@</code> ....	686, 687, 690, 755, 787, 789–791
<code>\aeb@saved@href</code> .....	561, 568
<code>\aebnameref</code> .....	514, 523, 527
<code>\aftergroup</code> .....	429, 794
<code>\AtBeginDocument</code> .....	145
<code>\AtBeginDvi</code> .....	179
<code>\AtEndDocument</code> .....	124
<code>\atPage</code> .....	518, 527
<b>B</b>	
<code>\baselineskip</code> .....	398, 400
<code>\bWebCustomize</code> .....	33
<b>C</b>	
<code>\ckchnngmlinktot@l</code> .....	124, 132
<code>\CMT</code> .....	171
<code>\Color</code> .....	153
<code>\color</code> .....	844
<code>\cr@ckitfalse</code> .....	404
<code>\cr@ckittrue</code> .....	427
<code>\CurrentBorderColor</code> ...	146, 153, 532, 538, 552, 760
<code>\CurrentOption</code> .....	7, 28
<b>D</b>	
<code>dbllevel</code> (option) .....	3
<code>\DeclareOptionX</code> .....	7, 18, 22, 28
<code>\DeclareRobustCommand</code> .....	936
<code>\discretionary</code> .....	902, 912
<code>\divide</code> .....	708
<code>dvips</code> (option) .....	3
<code>dvipsone</code> (option) .....	2
<b>E</b>	
<code>\ef@colorthislink</code> .....	758, 771, 844, 846, 850
<code>\ef@NO</code> .....	417, 727
<code>\ef@sanitize@toks</code> .....	717
<code>\ef@thislinkcolor</code> .....	531, 537, 551, 719, 721, 759, 769, 770, 772, 844
<code>\ef@thislinkcolor@SAVE</code> .....	720, 773
<code>\ef@YES</code> .....	417, 418, 463, 467, 727
<code>\egroup</code> .....	171, 402
<code>\empty</code> .....	576
<code>\endinput</code> .....	33
<code>\eq@mhlhyph</code> .....	859, 867, 869, 878, 881
<code>\eq@drivernum</code> .....	18, 22
<code>\eq@mlchunk</code> .....	872
<code>\eq@mlcrackat</code> .....	421, 424, 670, 710, 741–743, 750, 763, 778, 853, 873, 877, 887
<code>\eq@mlcrackinsat</code> .....	426, 786
<code>\eq@mhlhyph</code> .....	878
<code>\eq@mignore</code> .....	418
<code>\eq@S@value</code> .....	416, 726

\eq@setWidgetProps .....	161	\lastbox .....	609
\everypar .....	589	\leftskip .....	586
\eWebCustomize .....	34	\linknotformedfalse .....	403
\ExecuteOptions .....	37, 38, 42	\linknotformedtrue .....	437, 732
\ExecuteOptions@SAVE .....	37, 42	\literalps@out .....	447, 477, 492, 781, 785, 792, 799, 821
\ExecuteOptionsX .....	38, 40, 41	\llap .....	861
<b>F</b>		<b>M</b>	
\FixupProc .....	812, 831	\m@ne .....	584, 599
\font .....	178, 581	\maxdimen .....	582
\fontdimen .....	178	\ml@@nnotName .....	64, 67, 485, 504, 827
<b>G</b>		\ml@adj@x .....	175, 320, 324, 341, 348
\getrefbykeydefault .....	518	\ml@adj@y .....	175
<b>H</b>		\ml@bld@quadchunks .....	671, 677, 681, 687, 690, 697
\H .....	151	\ml@displayfalse .....	428, 874, 891
\hbadness .....	583	\ml@displaytrue .....	408, 425, 629, 876, 888
\hglue .....	407	\ml@dynamicsetup .....	894, 917, 930
\href .....	561	\ml@earlyExecProps .....	160, 441, 766
\hyper@chars .....	530, 758	\ml@err@msg .....	31, 35, 36
\hyper@linkfile .....	566	\ml@finish@link .....	444, 779, 783, 790, 813
\hyper@linkurl .....	565	\ml@hyper@launch .....	550, 567
\hyphenchar .....	581	\ml@hyper@linkfile .....	536, 566
\hyphenpenalty .....	591	\ml@hyper@linkurl .....	530, 565
<b>I</b>		\ml@HytextArg .....	410, 443
\if@ml@dvijs .....	16, 56, 69	\ml@interface@analyze .....	21, 612, 667
\ifcr@ckit .....	404	\ml@length .....	437–439, 444, 732, 735, 764, 779, 866, 909
\ifdim .....	398, 464, 643	\ml@lengthchunki .....	738, 743, 745
\iffixmlinks .....	62, 447, 781, 785, 791, 822	\ml@lengthchunkii .....	738, 744, 747
\ifHy@colorlinks .....	155, 843	\ml@minktoto@changed .....	133, 134
\ifHy@newwindow .....	543, 554	\ml@mllinkttotalchanged .....	141, 145
\ifHy@pdfnewwindow .....	543, 554	\ml@MrkLnk .....	13, 398, 441, 765, 789
\iflinknotformed .....	125, 403	\ml@next .....	420, 423, 425, 433
\ifml@display .....	629, 918	\ml@nocolor@defaults .....	151, 156, 415, 725
\ifmllinkttotalchanged .....	122, 141	\ml@nocolorHighlight .....	147, 151
\ifmlmarks .....	394, 398	\ml@nocolorLineStyle .....	148, 152
\ifoldstylequads .....	61, 163, 446, 476, 803, 816	\ml@nocolorLineWidth .....	149, 152
\ifpdf .....	35	\ml@qlngthchunki .....	672, 676, 678, 682, 733, 736, 739, 746, 783
\ifSmallRect .....	63, 123	\ml@qlngthchunkii .....	686, 688, 691, 739, 748, 787, 790
\ifvoid .....	610	\ml@raiseamt .....	399, 400, 402, 860
\ifxetex .....	36	\ml@setlink .....	412, 445, 722, 780, 784, 791
\InputIfFileExists .....	39	\ml@setnocolorDefaults .....	150, 158
\isstrikeout .....	467	\ml@SOUL@analyze .....	600, 604, 615
\isWindow .....	544–546, 555–557	\ml@SOUL@dosyllable .....	620, 625, 630, 635, 642, 663
\itsunderline .....	417, 463, 727	\ml@SOUL@doword .....	572, 762
<b>L</b>		\ml@SOUL@doword@SAVE .....	761
\labelRef .....	516, 522, 524, 528	\ml@SOUL@stop .....	526, 848
		\ml@space .....	425, 428, 879, 882, 897, 899
		\ml@start@link .....	439, 676, 686, 764, 787, 796
		\ml@strut .....	460, 462

`\ml@temp` ..... 523, 525, 527, 529  
`\ml@tempi` ..... 524, 525, 528, 529  
`\ml@typeset@syl` ..... 714, 862, 863, 869  
`\ml@typset@syl` ..... 860, 862  
`\ml@underlinded` ..... 405, 416, 726  
`\ml@url` ..... 756, 771  
`\ml@urlbuild` ..... 700  
`\ml@urlbuildi` ..... 570, 573, 672, 678, 682, 771  
`\ml@urlbuildii` ..... 571, 574, 688, 691, 710, 789  
`\mlcrackinsat` ..... 177  
`\mlcs` ..... 169  
`\mlcsarg` ..... 17, 449, 451, 774, 776  
`\mlldb` ..... 172–174, 229, 230, 237, 238, 243, 246–248, 250, 251, 253, 257–260, 264, 266, 267, 271, 272, 274, 275, 277–283, 290, 291, 296–299, 301, 303, 305, 307, 308, 331, 386  
`\mlldblevel` ..... 29, 30, 51, 174  
`\mlldbModeOff` ..... 173  
`\mlldbModeOn` ..... 172  
`\mlDict` ..... 70, 98, 479, 494, 826, 837  
`\mlfixOff` ..... 164  
`\mlh@postambleCmd` ..... 442, 767, 916, 926  
`\mlh@preambleCmdInsert` .... 168, 440, 758, 765, 850  
`\mlh@setQuadSyllable` ..... 455, 712, 866, 909  
`\mlhref` ..... 562  
`\mlhref@args` ..... 533, 546, 557, 564  
`\mlhyperlink` ..... 519  
`\mlhyperref` ..... 519  
`\mlhypertext` .. 406, 520, 522, 524, 528, 533, 546, 557  
`\mlhypertext@i` ..... 420, 423, 435  
`\mlignore` ..... 426, 429  
`\mllinkstotal` ..... 131, 135  
`\mllinktotalchangedfalse` ..... 122  
`\mllinktotalchangedtrue` ..... 139  
`\mllnkcontainer` ..... 65  
`\mlM@XCNT` ..... 738, 740, 742, 749  
`\mlmarksfalse` ..... 394, 396  
`\mlMarksOff` ..... 13, 396  
`\mlMarksOn` ..... 13, 395, 862  
`\mlmarkstrue` ..... 395  
`\mlMaxNSylls` ..... 176, 265  
`\mlNameref` ..... 519  
`\mlnameref` ..... 519  
`\mlpgMsg` ..... 52, 55, 59  
`\mlurl` ..... 22, 570  
`\mlurl@` ..... 717, 718  
`\MrkLnkLtr` ..... 397, 402  
`\multiply` ..... 735, 745, 747

## N

`\n` ..... 52, 54, 184, 187, 195, 202, 204, 205, 207, 210, 212, 214, 215, 217, 220, 223, 229, 230, 238, 243, 248, 251, 253, 258, 260, 264, 267, 272, 274, 275, 278, 280, 283, 291, 296–298, 300, 302, 304, 306–308, 331, 390  
`\NeedsTeXFormat` ..... 3  
`\newbox` ..... 118, 570, 571  
`\nolinkurl` ..... 756  
`\normalcolor` ..... 401, 430, 719, 794, 861  
`\normalfont` ..... 400, 861

## O

`\OldStyleBoxesOff` ..... 165, 167  
`\OldStyleBoxesOn` ..... 164, 166  
`\oldstylequadsfalse` ..... 163, 165  
`\oldstylequadstrue` ..... 164  
options:  
`dblevel` ..... 3  
`dvips` ..... 3  
`dvipsone` ..... 2  
`urlOpts` ..... 3

## P

`\PackageError` ..... 35, 36  
`\PackageWarning` ..... 818  
`\PackageWarningNoLine` ..... 126, 136, 142  
`\par@Rect` ..... 92, 112, 383  
`\parfillskip` ..... 590  
`\PassOptionsToPackage` ..... 7, 19, 20, 23, 24, 28  
`\pboxRect` ..... 68  
`\penalty` ..... 427, 714, 786, 788  
`\pgmonitoring` ..... 56, 82, 102  
`\presets` ..... 415, 533, 546, 557, 725  
`\pretolerance` ..... 584  
`\processAppArgs` ..... 414, 724  
`\ProcessOptionsX` ..... 8, 44  
`\protected@edef` ..... 523, 527  
`\ProvidesPackage` ..... 5  
`\ps@mark` ..... 50, 480, 801, 831, 838

## Q

`\QuadPoints` ..... 413, 723

## R

`\raisebox` ..... 402, 860  
`\removelastspace` ..... 177, 178  
`\RequirePackage` ..... 4, 9, 13–15, 45–49  
`\revCrackAt` ..... 674, 680, 739, 752  
`\rightskip` ..... 587

\rlap .....	400	\SOUL@sethyphenchar .....	622, 905
<b>S</b>		\SOUL@setkern .....	868, 904, 910, 911, 913
\S .....	152, 415, 725	\SOUL@setup .....	927
\set@LinkPboxDefaults .....	414, 724	\SOUL@stop .....	526
\setbox .....	399,	\SOUL@stopM .....	648
460, 462, 573, 574, 579, 605, 609, 653, 669, 711		\SOUL@syllable .....	617, 662, 867, 868
\setLinkPbox .....	412, 722	\SOUL@syllgoal .....	613, 619, 643
\setQuadBox .....	84, 104, 481, 496	\SOUL@syllwidth .....	616, 643, 654
\smallRectTF .....	61, 66	\SOUL@token ...	632, 633, 639, 653, 655, 658, 660, 662
\SmallRecttrue .....	123	\SOUL@tt .....	580, 653
\smash .....	400, 861	\SOUL@ttwidth .....	619
\SOUL@ .....	937	\SOUL@word .....	575, 595, 601
\SOUL@buffer .....	578	\spaceskip .....	896
\SOUL@charkern .....	638, 657, 868, 913	\strut .....	402, 861
\SOUL@doword .....	761, 762	\syllableCnt .....	121, 668, 674, 680,
\SOUL@errmsg .....	596, 597, 649, 650	684, 729, 775, 852, 869, 873, 877, 887, 908, 936	
\SOUL@error .....	597	<b>T</b>	
\SOUL@eventuallyexhyphen .....	632	\texttt .....	169
\SOUL@everyexhyphen .....	920, 923, 933	\theView .....	540, 541, 547
\SOUL@everyhyphen .....	623, 932	\tiny .....	402, 861
\SOUL@everylowerthan .....	640	\tolerance .....	585
\SOUL@everyspace .....	919, 922, 931	\turnSyllbCntOff .....	863, 864
\SOUL@everysyllable .....	644, 929	\turnSyllbCntOn .....	862
\SOUL@everytoken .....	661	<b>U</b>	
\SOUL@getkern .....	621, 638, 657	\unhbox .....	669
\SOUL@getttoken .....	631, 637, 656	\unhcopy .....	712–714, 771, 789
\SOUL@hyphenhintM .....	634	\unkern .....	903
\SOUL@hyphkern .....	621, 904, 910, 911	\unpenalty .....	608
\SOUL@lasttoken .....	621, 638, 655, 657, 660	\unvcopy .....	606
\SOUL@lowerthanM .....	636	\URI .....	722
\SOUL@mleveryysyllable@ii .....	856, 871	\url@Opts .....	26, 27, 45
\SOUL@mlheveryexhyphen .....	908, 920, 933	\urldef .....	756
\SOUL@mlheveryhyphen .....	901, 932	urlOpts (option) .....	3
\SOUL@mlheveryspace .....	896, 898, 919, 931	<b>W</b>	
\SOUL@mlheverysyllable .....	851, 929	\W .....	152, 415, 725
\SOUL@mlheverysyllable@i .....	854, 865, 884, 889	\write .....	130, 139, 449, 451, 774, 776
\SOUL@mlhpostamble .....	926, 934	\wrt@linksnotformed .....	124, 125
\SOUL@mlhpreamble .....	757, 849, 928	\wrtmlinktot@l .....	124, 130
\SOUL@mlhsetup .....	927, 937	<b>X</b>	
\SOUL@n .....	635, 642, 646, 651, 663, 665	\x .....	471–474, 575, 576, 633, 634, 636, 648, 662
\SOUL@postamble .....	934		
\SOUL@preamble .....	928		
\SOUL@puttoken .....	641, 645, 659		



## 8 Change History

v2.0 (2016/02/16)	v2.1.9 (2018/03/22)
\mlurl: Added support for the \url command. . 18	General: Changed the definition of \ml@MrkLnk . 13
v2.0.1 (2017/09/19)	v2.3 (2018/04/26)
\mlurl: Save the incoming link color . . . . . 24	General: Added aeb-mlink as an alternate name
v2.1.10 (2018/03/25)	for this package . . . . . 2
General: Set /Rect to minimal rectangle	v2.3.1 (2018/07/18)
containing text . . . . . 12	\mlhypertext: added \hglue0pt to add some
v2.1.17 (2018/04/19)	hard glue just before the beginning of
\mlhypertext: added conditional command	\mlhypertext . . . . . 14
definition for \mlhypertext . . . . . 14	v2.3.2 (2018/08/09)
v2.1.18 (2018/04/20)	General: \mlMaxNSylls now sets the array size of
\mlurl: Added bCreateLink, etc., to protect	gAryL . . . . . 10
against distiller/ps2pdf from crashing . . . . . 26	Form the array aFixup for aeb_mlink . . . . . 11
v2.1.2 (2018/03/09)	\mlurl: We make gFixup into a macro name
General: Added PS proc smallquadpointsfixup 12	\FixupProc . . . . . 25
v2.1.3 (2018/03/10)	v2.3.5 (2020/01/06)
General: Reinstate old style link . . . . . 7	General: Added url0pts . . . . . 3
v2.1.8 (2018/03/19)	Conform to the new web.cfg format . . . . . 3
General: Added tracking marks . . . . . 13	Now require url package and pass options to
	url through url0pts . . . . . 3