

pathsuris.sty: Paths and URIs for \TeX *

Jinbo Zhang, Michael Kohlhase, Dennis Müller
FAU Erlangen-Nürnberg

October 20, 2020

Abstract

This package provides macros to deal with paths and base URIs for \TeX . In particular, it offers a path canonicalizer, which is used in package `modules`, in order to support modules specified with relative path.

Contents

1	User Interface	2
1.1	Base URIs	2
1.2	Using Absolute Paths	2
1.3	Path Canonicalization	2
1.4	URI splitting	2
2	The Implementation	4
2.1	Base URIs	4
2.2	Using Absolute Paths	4
2.3	Path Canonicalization	4
2.4	URI splitting	6

*Version v2.1 (last revised 2020/09/30)

1 User Interface

1.1 Base URIs

`\baseURI` `\baseURI`¹

1.2 Using Absolute Paths

Finally, the separation of documents into multiple modules often profits from a symbolic management of file paths. To simplify this, the `modules` package supplies the `\defpath` macro: `\defpath[\langle baseURI \rangle]{\langle cname \rangle}{\langle path \rangle}` defines a command, so that `\langle cname \rangle{\langle name \rangle}` expands to `\langle path \rangle/\langle name \rangle`. So we could have used

```
\defpath{OPaths}{../other}
\importmodule[load=\OPahts{bar}]{bar}
```

instead of the second line in Example ???. The variant `\OPaths` has the big advantage that we can get around the fact that $\text{\TeX}/\text{\LaTeX}$ does not set the current directory in `\input`, so that we can use systematically deployed `\defpath`-defined path macros to make modules relocatable by defining the path macros locally. The optional parameter `\langle baseURI \rangle` is for the L^AT_EXML transformation, which (if `\langle baseURI \rangle` is specified) resolves `\langle path \rangle` to an absolute URI according to [BFM05, section 5.2].

1.3 Path Canonicalization

By calling `\@cpath{\langle path \rangle}`, the canonicalized path will be stored in `\@CanPath`. To print a canonicalized path, simply use `\cpath{\langle path \rangle}`. Here is a set of examples with their canonizalized paths for testing.

path	canonicalized path	expected
aaa	aaa	aaa
../.. /aaa	../.. /aaa	../.. /aaa
aaa/bbb	aaa/bbb	aaa/bbb
aaa/..		
../.. /aaa/bbb	../.. /aaa/bbb	../.. /aaa/bbb
../aaa/.. /bbb	../bbb	../bbb
../aaa/bbb	../aaa/bbb	../aaa/bbb
aaa/bbb/.. /ddd	aaa/ddd	aaa/ddd
aaa/bbb/.. /..		

1.4 URIs

By calling `\seturi[\meta{macroname}]{\langle path \rangle}`, the URI will be split into its components `\macronamescheme`, `\macronameauthority`, `\macronamepath`,

¹EdNOTE: document it

`\macronamequery` and `\macronamefragment`, and the resolved URI itself is stored in `\macronameuri`, as in the following example. If the optional `macroname` is not provided, the default name is `pathsuris@curruri@`.

In order to differentiate between empty and missing components, a missing component will be equal to `\makeuri@empty`, whose *expansion* is `\relax`.

```
\seturi[myuri]{http://this.isatest/foo/bar/?query#fragment}
```

yields:

macro	value
<code>\myuriuri</code>	<code>http://this.isatest/foo/bar?query#fragment</code>
<code>\myurischeme</code>	<code>http</code>
<code>\myuriauthority</code>	<code>this.isatest</code>
<code>\myuripath</code>	<code>foo/bar</code>
<code>\myuriquery</code>	<code>query</code>
<code>\myurifragment</code>	<code>fragment</code>

`\makeuri{\meta{scheme}}{\meta{authority}}{\meta{path}}{\meta{query}}{\meta{fragment}}` constructs a URI from its individual components. The (expanded and resolved) individual components will be stored in `\makeuri@scheme`, `\makeuri@authority`, etc.; the resolved URI will be stored in `\makeuri@uri`.

`\asuri{\meta{macroname}}{\meta{uri}}`, similarly to `setpath`, defines a new macro `\macroname[\meta{newmacroname}]{\meta{command}}` that allows manipulating `uri` in various ways. `\asuri` calls `\seturi[\meta{macroname}]{\meta{uri}}`, so the individual components and full `uri` (as string) are subsequently stored in `\macronamescheme`, `\macronameauthority`, etc.

If an optional new macro name is given in `\macroname`, then the result of the modification is stored in that new macro, as if defined via `\asuri`; otherwise, the macro is modified “in place”.

- `\macroname{drop query}` drops the query component.
- `\macroname{drop fragment}` drops the fragment component.
- `\macroname{/other/path}` drops query and fragment, appends `other/path` to the path and resolves the URI.
- `\macroname{?newquery}` drops the fragment and either declares `newquery` as a new query component, or appends `?newquery` to the existing query component, if it is not `\makeuri@empty`. Note, that this behaviour diverges from the official URI specification, but it conforms to MMT URI’s, which use `?` as separator between DPaths, modules names and declaration names.
- `\macroname{#newfragment}` analogously to `{?newquery}`.

2 The Implementation

```

1 <*package>
2 \RequirePackage{stex-base}
3 \RequirePackage{xstring}
4 \RequirePackage{etoolbox}

```

2.1 Base URIs

`\baseURI` On the L^AT_EX side we do nothing (for the moment).

```

5 \newcommand\baseURI[2] [] {}

```

2.2 Using Absolute Paths

`\defpath` `\defpath[optional argument]{macro name}{base path}` defines a new macro which can take another path to form one integrated path. For example, `\MathHub` in every `localpaths.tex` is defined as:

```

\defpath{MathHub}{/path/to/localmh/MathHub}

```

then we can use `\MathHub` to form other paths, for example,

```

\MathHub{source/smgglom/sets}

```

will generate `/path/to/localmh/MathHub/source/smgglom/sets`.

```

6 \newrobustcmd\defpath[3] [] {%
7   \expandafter\newcommand\csname #2\endcsname[1] {#3/##1}%
8 }%

```

2.3 Path Canonicalization

We define two macros for changing the category codes of common characters in URIs, in particular `#`.

```

9 \def\pathsuris@setcatcodes{%
10   \edef\pathsuris@oldcatcode@hash{\the\catcode'\#}%
11   \catcode'\#=12\relax%
12   \edef\pathsuris@oldcatcode@slash{\the\catcode'\/%}
13   \catcode'\/=12\relax%
14   \edef\pathsuris@oldcatcode@colon{\the\catcode'\:%}
15   \catcode'\:=12\relax%
16   \edef\pathsuris@oldcatcode@qm{\the\catcode'\?}%
17   \catcode'\?=12\relax%
18 }
19 \def\pathsuris@resetcatcodes{%
20   \catcode'\#\pathsuris@oldcatcode@hash\relax%
21   \catcode'\/>\pathsuris@oldcatcode@slash\relax%
22   \catcode'\:\pathsuris@oldcatcode@colon\relax%
23   \catcode'\?\pathsuris@oldcatcode@qm\relax%
24 }

```

We define some macros for later comparison.

```

25 \def\@ToTop{..}
26 \def\@Slash{/}
27 \def\@Colon{:}
28 \def\@QuestionMark{?}
29 \def\@Dot{.}

```

30

```

31 \pathsuris@setcatcodes
32 \def\@Fragment{#}
33 \pathsuris@resetcatcodes

```

Implement \@cpath.

\@cpath

```

34 \def\@cpath#1{%
35   \edef\pathsuris@cpath@temp{#1}%
36   \def\@CanPath{}%
37   \IfBeginWith\pathsuris@cpath@temp\@Slash{%
38     \@cpath@loop%
39     \edef\@CanPath{\@Slash\@CanPath}%
40   }{%
41     \@cpath@loop%
42   }%
43   \IfEndWith\@CanPath\@Slash{%
44     \ifx\@CanPath\@Slash\else%
45       \StrGobbleRight\@CanPath1[\@CanPath]%
46     \fi%
47   }{}%
48 }
49
50 \def\@cpath@loop{%
51   \IfSubStr\pathsuris@cpath@temp\@Slash{%
52     \StrCut\pathsuris@cpath@temp\@Slash\pathsuris@cpath@temp@a\pathsuris@cpath@temp%
53     \ifx\pathsuris@cpath@temp@a\@ToTop%
54       \ifx\@CanPath\empty%
55         \edef\@CanPath{\@ToTop}%
56       \else%
57         \edef\@CanPath{\@CanPath\@Slash\@ToTop}%
58       \fi%
59       \@cpath@loop%
60     \else%
61       \IfBeginWith\pathsuris@cpath@temp\@ToTop{%
62         \StrBehind{\pathsuris@cpath@temp}{\@ToTop}[\pathsuris@cpath@temp]%
63         \IfBeginWith\pathsuris@cpath@temp\@Slash{%
64           \edef\pathsuris@cpath@temp{\@CanPath\pathsuris@cpath@temp}%
65         }{%
66           \ifx\@CanPath\empty\else%
67             \edef\pathsuris@cpath@temp{\@CanPath\@Slash\pathsuris@cpath@temp}
68           \fi%
69         }%

```

```

70         \def\@CanPath{}%
71         \@cpath@loop%
72     }{%
73         \ifx\@CanPath\@empty%
74             \edef\@CanPath{\pathsuris@cpath@temp@a}%
75         \else%
76             \edef\@CanPath{\@CanPath\@Slash\pathsuris@cpath@temp@a}%
77         \fi%
78         \@cpath@loop
79     }%
80 \fi%
81 }{
82     \ifx\@CanPath\@empty%
83         \edef\@CanPath{\pathsuris@cpath@temp}%
84     \else%
85         \edef\@CanPath{\@CanPath\@Slash\pathsuris@cpath@temp}%
86     \fi%
87 }%
88 }

```

Implement `\cpath` to print the canonicalized path.

`\cpath`

```

89 \newcommand\cpath[1]{%
90     \@cpath{#1}%
91     \@CanPath%
92 }

```

2.4 URIs

Various macros for dealing with URIs. To deal with empty URI components (scheme, authority, etc.), we use `\relax` to signify a non-existent component as opposed to an empty one.

```

93 \def\makeuri@setempty#1{\def#1{\relax}}
94 \def\makeuri@empty{\relax}
95 \def\makeuri@test#1{%
96     \ifx#1\makeuri@empty\else#1\fi%
97 }

```

`\makeuri` `\makeuri` constructs a URI from scheme, authority, path, query and fragment separately.

```

98 \def\makeuri@uri{}
99 \def\makeuri#1#2#3#4#5{
100     \edef\makeuri@scheme{#1}
101     \edef\makeuri@authority{#2}
102     \edef\makeuri@path{#3}
103     \ifx\makeuri@path\makeuri@empty\else
104         \@cpath{#3}
105     \edef\makeuri@path{\@CanPath}

```

```

106 \fi
107 \edef\makeuri@query{#4}
108 \edef\makeuri@fragment{#5}
109 \ifx\makeuri@scheme\makeuri@empty\else
110 \edef\makeuri@scheme{\makeuri@scheme\@Colon}
111 \fi
112 \ifx\makeuri@authority\makeuri@empty\else
113 \edef\makeuri@authority{\@Slash\@Slash\makeuri@authority}
114 \ifx\makeuri@path\makeuri@empty\else
115 \IfBeginWith\makeuri@path\@Slash{}\{
116 \edef\makeuri@path{\@Slash\makeuri@path}
117 }
118 \fi
119 \fi
120 \ifx\makeuri@query\makeuri@empty\else
121 \edef\makeuri@query{\@QuestionMark\makeuri@query}
122 \fi
123 \ifx\makeuri@fragment\makeuri@empty\else
124 \edef\makeuri@fragment{\@Fragment\makeuri@fragment}
125 \fi
126 \edef\makeuri@uri{%
127 \makeuri@test\makeuri@scheme%
128 \makeuri@test\makeuri@authority%
129 \makeuri@test\makeuri@path%
130 \makeuri@test\makeuri@query%
131 \makeuri@test\makeuri@fragment%
132 }
133 }

```

\seturi@

```

134 \newif\if@pathsuris@done@
135 \def\seturi@[#1]#2{%
136 \@pathsuris@done@false%
137 \def\pathsuris@prefix@temp{#1}
138 \edef\pathsuris@curruri{#2}%
139 \edef\pathsuris@curruri{\expandafter\detokenize\expandafter{\pathsuris@curruri}}
140 \let\pathsuris@temp\pathsuris@curruri%
141 \makeuri@setempty\pathsuris@curruri@scheme%
142 \makeuri@setempty\pathsuris@curruri@authority%
143 \makeuri@setempty\pathsuris@curruri@path%
144 \makeuri@setempty\pathsuris@curruri@query%
145 \makeuri@setempty\pathsuris@curruri@fragment%
146 % scheme
147 \IfSubStr{\pathsuris@temp}{\@Colon}{%
148 % TODO check for valid scheme
149 \StrBefore{\pathsuris@temp}{\@Colon}[\pathsuris@curruri@scheme]%
150 \StrBehind{\pathsuris@temp}{\@Colon}[\pathsuris@temp]%
151 }{}%
152 % authority
153 \IfBeginWith{\pathsuris@temp}{\@Slash\@Slash}{%

```

```

154 \StrBehind{\pathsuris@temp}{\@Slash\@Slash}[\pathsuris@temp]%
155 \IfSubStr{\pathsuris@temp}{\@Slash}{%
156 \StrBefore{\pathsuris@temp}{\@Slash}[\pathsuris@curruri@authority]%
157 \StrBehind{\pathsuris@temp}{\@Slash}[\pathsuris@temp]%
158 % TODO userinfo,host,port
159 }{%
160 \IfSubStr\pathsuris@temp\@QuestionMark{
161 \StrBefore{\pathsuris@temp}{\@QuestionMark}[\pathsuris@curruri@authority]%
162 \StrBehind{\pathsuris@temp}{\@QuestionMark}[\pathsuris@temp]%
163 \edef\pathsuris@temp{\@QuestionMark\pathsuris@temp}%
164 }{
165 \IfSubStr\pathsuris@temp\@Fragment{
166 \StrBefore{\pathsuris@temp}{\@Fragment}[\pathsuris@curruri@authority]%
167 \StrBehind{\pathsuris@temp}{\@Fragment}[\pathsuris@temp]%
168 \edef\pathsuris@temp{\@Fragment\pathsuris@temp}%
169 }{
170 \edef\pathsuris@curruri@authority{\pathsuris@temp}%
171 \@pathsuris@done@true%
172 }
173 }
174 }%
175 }{}%
176 % path, query, fragment
177 \if@pathsuris@done@else%
178 \IfSubStr{\pathsuris@temp}{\@QuestionMark}{%
179 % path
180 \StrBefore{\pathsuris@temp}{\@QuestionMark}[\pathsuris@curruri@path]%
181 \@cpath\pathsuris@curruri@path%
182 \edef\pathsuris@curruri@path{\@CanPath}%
183 \StrBehind{\pathsuris@temp}{\@QuestionMark}[\pathsuris@temp]%
184 % query, fragment
185 \IfSubStr{\pathsuris@temp}{\@Fragment}{%
186 \StrBefore{\pathsuris@temp}{\@Fragment}[\pathsuris@curruri@query]%
187 \StrBehind{\pathsuris@temp}{\@Fragment}[\pathsuris@curruri@fragment]%
188 }{%
189 \edef\pathsuris@curruri@query{\pathsuris@temp}%
190 }%
191 }{%
192 % path, fragment
193 \IfSubStr{\pathsuris@temp}{\@Fragment}{%
194 \StrBefore{\pathsuris@temp}{\@Fragment}[\pathsuris@curruri@path]%
195 \@cpath\pathsuris@curruri@path%
196 \edef\pathsuris@curruri@path{\@CanPath}%
197 \StrBehind{\pathsuris@temp}{\@Fragment}[\pathsuris@curruri@fragment]%
198 }{%
199 \edef\pathsuris@curruri@path{\pathsuris@temp}%
200 }%
201 }%
202 \fi%
203 \makeuri\pathsuris@curruri@scheme\pathsuris@curruri@authority\pathsuris@curruri@path\pathsuris@curruri@query\pathsuris@curruri@fragment

```



```

204 \let\pathsuris@curruri@uri\makeuri@uri
205 %drop trailing slash of path
206 %\IfEndWith{\pathsuris@curruri@path}{\@Slash}{%
207 % \StrGobbleRight{\pathsuris@curruri@path}{1}[\pathsuris@curruri@path]
208 %}{}%
209 %
210 %\edef\pathsuris@curruri@path{\cpath{\pathsuris@curruri@path}}%
211 \ifx\pathsuris@prefix@temp\empty\else%
212 \expandafter\let\csname \pathsuris@prefix@temp scheme\endcsname\pathsuris@curruri@schem
213 \expandafter\let\csname \pathsuris@prefix@temp authority\endcsname\pathsuris@curruri@au
214 \expandafter\let\csname \pathsuris@prefix@temp path\endcsname\pathsuris@curruri@path%
215 \expandafter\let\csname \pathsuris@prefix@temp query\endcsname\pathsuris@curruri@query%
216 \expandafter\let\csname \pathsuris@prefix@temp fragment\endcsname\pathsuris@curruri@fra
217 \expandafter\let\csname \pathsuris@prefix@temp uri\endcsname\pathsuris@curruri@uri%
218 \fi%
219 }

\seturi

220 \newrobustcmd\seturi[1][]{%
221 \pathsuris@setcatcodes%
222 \expandafter\pathsuris@resetcatcodes\seturi@[#1]%
223 }

\asuri \asuri{macroname}{uri} generates \macroname[optional new macro name]{action},
that allows for modifying uri in various ways.

224
225 \def\asuri#1{%
226 \pathsuris@setcatcodes%
227 \expandafter\pathsuris@resetcatcodes\asuri[#1]%
228 }
229
230 \def\@asuri[#1]#2{
231 \cpath{#2}
232 \expandafter\def\csname #1\endcsname{
233 \expandafter\edef\csname #1uri\endcsname{\@CanPath}
234 \seturi[#1]{\@CanPath}
235 \expandafter\renewcommand\csname #1\endcsname[1][]{%
236 \pathsuris@setcatcodes%
237 \@asuri@[##1]{#1}%
238 }%
239 }
240
241 \protected\def\@asuri@[#1]#2#3{
242 \pathsuris@resetcatcodes
243 \@asuri[#1]{#2}{#3}
244 }
245
246 \newif\if@asuri@changed@
247 \protected\def\@asuri@[#1]#2#3{

```

```

248 \@asuri@changed@false
249 \edef\@asuri@command{#3}
250 \trimstring\@asuri@command
251 \IfBeginWith\@asuri@command{drop}{
252   \StrBehind{\@asuri@command}{drop}[\@asuri@command]
253   \trimstring\@asuri@command
254   \IfStrEq\@asuri@command{query}{
255     \makeuri{\csname #2scheme\endcsname}%
256     {\csname #2authority\endcsname}%
257     {\csname #2path\endcsname}%
258     \makeuri@empty%
259     {\csname #2fragment\endcsname}%
260     \@asuri@changed@true
261   }{
262     \IfStrEq\@asuri@command{fragment}{
263       \makeuri{\csname #2scheme\endcsname}%
264       {\csname #2authority\endcsname}%
265       {\csname #2path\endcsname}%
266       {\csname #2query\endcsname}%
267       \makeuri@empty%
268       \@asuri@changed@true
269     }{
270       \IfStrEq\@asuri@command{extension}{
271         \edef\@asuri@oldpath{\csname #2path\endcsname}
272         \StrCount\@asuri@oldpath.[\@asuri@lastdot]
273         \ifnum\@asuri@lastdot>0
274           \StrBehind[\@asuri@lastdot]\@asuri@oldpath.[\@asuri@extension]
275           \IfSubStr\@asuri@extension\@Slash{{
276             \StrBefore[\@asuri@lastdot]\@asuri@oldpath.[\@asuri@oldpath]
277           }}
278           \fi
279           \makeuri{\csname #2scheme\endcsname}%
280           {\csname #2authority\endcsname}%
281           \@asuri@oldpath%
282           \makeuri@empty%
283           \makeuri@empty%
284           \@asuri@changed@true
285         }{}}
286   }{
287     \IfBeginWith\@asuri@command{\@Slash}{
288       \@cpath{\csname #2path\endcsname\@asuri@command}
289       \makeuri{\csname #2scheme\endcsname}%
290       {\csname #2authority\endcsname}%
291       {\@CanPath}%
292       \makeuri@empty%
293       \makeuri@empty%
294       \@asuri@changed@true
295     }{
296       \IfBeginWith\@asuri@command{\@QuestionMark}{
297         \expandafter\ifx\csname #2query\endcsname\makeuri@empty

```

```

298         \StrBehind\@@asuri@command\@QuestionMark[\@@asuri@command]
299         \edef\@@asuri@nquery{\@@asuri@command}
300     \else
301         \edef\@@asuri@nquery{\csname #2query\endcsname\@@asuri@command}
302     \fi
303     \makeuri{\csname #2scheme\endcsname}%
304         {\csname #2authority\endcsname}%
305         {\csname #2path\endcsname}%
306         {\@@asuri@nquery}%
307     \makeuri@empty%
308     \@asuri@changed@true
309 }{
310 \IfBeginWith\@@asuri@command{\@Fragment}{
311     \expandafter\ifx\csname #2fragment\endcsname\makeuri@empty
312     \StrBehind\@@asuri@command\@Fragment[\@@asuri@command]
313     \edef\@@asuri@nfrag{\@@asuri@command}
314 \else
315     \edef\@@asuri@nfrag{\csname #2fragment\endcsname\@@asuri@command}
316 \fi
317     \makeuri{\csname #2scheme\endcsname}%
318         {\csname #2authority\endcsname}%
319         {\csname #2path\endcsname}%
320         {\csname #2query\endcsname}%
321         {\@@asuri@nfrag}%
322     \@asuri@changed@true
323 }{}
324 }}}
325 \edef\@@asuri@ncs{#1}
326 \if@asuri@changed@
327     \ifx\@@asuri@ncs\@empty
328         \asuri{#2}\makeuri@uri
329     \else
330         \asuri\@@asuri@ncs\makeuri@uri
331     \fi
332 \fi
333 }
334

```

auxiliary code:

```

335 \def\@Space{ }
336 \def\trimstring#1{
337     \edef\pathsuris@trim@temp{#1}
338     \IfBeginWith\pathsuris@trim@temp\@Space{
339         \StrGobbleLeft\pathsuris@trim@temp1[#1]
340         \trimstring{#1}
341     }{
342         \IfEndWith\pathsuris@trim@temp\@Space{
343             \StrGobbleRight\pathsuris@trim@temp1[#1]
344             \trimstring{#1}
345         }{

```

```

346         \edef#1{\pathsuris@trim@temp}
347     }
348 }
349 }
350
351 % windows paths
352
353 \catcode'\.=0
354 .catcode'\.=12
355 .let.\@BackSlash\
356 .catcode'\.=0
357 \catcode'\.=12
358
359 \newif\if@windowstopath@inpath@
360 \def\windows@to@path#1{
361     \@windowstopath@inpath@false
362     \def\windows@temp{}
363     \edef\windows@path{#1}
364     \ifx\windows@path\empty\else
365         \expandafter\windows@path@loop\windows@path\windows@path@end
366     \fi
367     \let#1\windows@temp
368 }
369 \def\windows@path@loop#1#2\windows@path@end{
370     \def\windows@temp@b{#2}
371     \ifx\windows@temp@b\empty
372         \def\windows@continue{}
373     \else
374         \def\windows@continue{\windows@path@loop#2\windows@path@end}
375     \fi
376     \if@windowstopath@inpath@
377         \ifx#1\@BackSlash
378             \edef\windows@temp{\windows@temp\@Slash}
379         \else
380             \edef\windows@temp{\windows@temp#1}
381         \fi
382     \else
383         \ifx#1:
384             \edef\windows@temp{\@Slash\windows@temp}
385             \@windowstopath@inpath@true
386         \else
387             \edef\windows@temp{\windows@temp#1}
388         \fi
389     \fi
390     \windows@continue
391 }
392
393 \def\path@to@windows#1{
394     \@windowstopath@inpath@false
395     \def\windows@temp{}

```

```

396 \edef\windows@path{#1}
397 \edef\windows@path{\expandafter\@gobble\windows@path}
398 \ifx\windows@path\@empty\else
399 \expandafter\path@windows@loop\windows@path\windows@path@end
400 \fi
401 \let#1\windows@temp
402 }
403 \def\path@windows@loop#1#2\windows@path@end{
404 \def\windows@temp@b{#2}
405 \ifx\windows@temp@b\@empty
406 \def\windows@continue{}
407 \else
408 \def\windows@continue{\path@windows@loop#2\windows@path@end}
409 \fi
410 \if@windowstopath@inpath@
411 \ifx#1/
412 \edef\windows@temp{\windows@temp\@BackSlash}
413 \else
414 \edef\windows@temp{\windows@temp#1}
415 \fi
416 \else
417 \ifx#1/
418 \edef\windows@temp{\windows@temp:\@BackSlash}
419 \@windowstopath@inpath@true
420 \else
421 \edef\windows@temp{\windows@temp#1}
422 \fi
423 \fi
424 \windows@continue
425 }
426
427 % kpsewhich
428
429 \newif\if@iswindows@\@iswindows@false
430 \IfFileExists{nul:}{\IfFileExists{/dev/null}}{\@iswindows@true}}{}
431
432 \def\kpsewhich#1#2{\begingroup
433 \def\@Space{ }
434 \edef\kpsewhich@cmd{"|kpsewhich #2"}
435 \everyeof{\noexpand}
436 \catcode'\=12
437 \edef#1{\@input\kpsewhich@cmd\@Space}
438 \trimstring#1
439 \global\let#1#1
440 \endgroup}
441
442 % main directory
443
444 \edef\oldpercentcatcode{\the\catcode'\%}
445 \catcode'\%=12

```

```

446 \let\percent%
447 \catcode'\%=\oldpercentcatcode
448
449 \edef\pwd@cmd{\if@iswindows@ -expand-var \percent CD\percent\else -var-value PWD\fi}
450 \kpsewhich\stex@maindir\pwd@cmd
451 \if@iswindows@\windows@to@path\stex@maindir\fi
452
453 \def\path@filename#1#2{
454     \edef\filename@oldpath{#1}
455     \StrCount\filename@oldpath\@Slash[\filename@lastslash]
456     \ifnum\filename@lastslash>0
457         \StrBehind[\filename@lastslash]\filename@oldpath\@Slash[\filename@oldpath]
458         \csedef{#2}{\filename@oldpath}
459     \else
460         \csedef{#2}{\filename@oldpath}
461     \fi
462 }
463
464 \def\path@droextension#1#2{
465     \path@filename{#1}{droextension@temp}
466     \StrCount\droextension@temp\@Dot[\droextension@lastdot]
467     \ifnum\droextension@lastdot>0
468         \StrBehind[\droextension@lastdot]\droextension@temp\@Dot[\droextension@ext]
469         \StrLen\droextension@ext[\droextension@lastdot]
470         \StrGobbleRight{#1}{\the\numexpr\droextension@lastdot+1\@Space}[\droextension@temp]
471         \trimstring\droextension@temp
472         \csedef{#2}{\droextension@temp}
473     \else
474         \csedef{#2}{#1}
475     \fi
476 }
477
478 \</package>

```

Change History

v1.0

General: First Version with
Documentation 1

v1.1

General: adding `\baseURI` from
`omdoc.sty` and `\defpath` from
`modules.sty` 1

References

- [BFM05] Tim Berners-Lee, Roy T. Fielding, and Larry Masinter. *Uniform Resource Identifier (URI): Generic Syntax*. RFC 3986. Internet Engineering Task Force (IETF), 2005. URL: <http://www.ietf.org/rfc/rfc3986.txt>.