

# pathsuris.sty: Paths and URIs for $\text{\TeX}$ \*

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

October 2, 2020

## Abstract

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

## Contents

<b>1</b>	<b>User Interface</b>	<b>2</b>
1.1	Base URIs . . . . .	2
1.2	Using Absolute Paths . . . . .	2
1.3	Path Canonicalization . . . . .	2
1.4	URI splitting . . . . .	2
<b>2</b>	<b>The Implementation</b>	<b>4</b>
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`<sup>1</sup>

## 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  $\text{\TeX/L\TeX}$  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  $\text{\LaTeXML}$  transformation, which (if `\langle baseURI \rangle` is specified) resolves `\langle path \rangle` to an absolute URI according to [BerFieMas:05].

## 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 URI splitting

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

---

<sup>1</sup>EdNOTE: document it

and `\macronamefragment`, as in the following example. If the optional `macroname` is not provided, the default name is `pathsuris@curruri@`.

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

yields:

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

## 2 The Implementation

```

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

```

### 2.1 Base URIs

`\baseURI` On the L<sup>A</sup>T<sub>E</sub>X 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\@ToHere{.}
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 \let\pathsuris@temp\pathsuris@curruri%
140 \makeuri@setempty\pathsuris@curruri@scheme%
141 \makeuri@setempty\pathsuris@curruri@authority%
142 \makeuri@setempty\pathsuris@curruri@path%
143 \makeuri@setempty\pathsuris@curruri@query%
144 \makeuri@setempty\pathsuris@curruri@fragment%
145 % scheme
146 \IfSubStr{\pathsuris@temp}{\@Colon}{%
147 % TODO check for valid scheme
148 \StrBefore{\pathsuris@temp}{\@Colon}[\pathsuris@curruri@scheme]%
149 \StrBehind{\pathsuris@temp}{\@Colon}[\pathsuris@temp]%
150 }{}%
151 % authority
152 \IfBeginWith{\pathsuris@temp}{\@Slash\@Slash}{%
153 \StrBehind{\pathsuris@temp}{\@Slash\@Slash}[\pathsuris@temp]%

```

```

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

```



```

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

\seturi

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

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



223
224 \def\asuri#1{%
225 \pathsuris@setcatcodes%
226 \expandafter\pathsuris@resetcatcodes\asuri@{#1}%
227 }
228
229 \def\@asuri[#1]#2{
230 \cpath{#2}

```

```

231 \expandafter\def\csname #1\endcsname{
232 \expandafter\edef\csname #1uri\endcsname{\@CanPath}
233 \seturi[#1]{\@CanPath}
234 \expandafter\renewcommand\csname #1\endcsname[1][]{%
235 \pathsuris@setcatcodes%
236 \@asuri@[##1]{#1}%
237 }%
238 }
239
240 \protected\def\@asuri@[#1]#2#3{
241 \pathsuris@resetcatcodes
242 \@asuri@[#1]{#2}{#3}
243 }
244
245 \newif\if@asuri@changed@
246 \protected\def\@asuri@[#1]#2#3{
247 \@asuri@changed@false
248 \edef\@asuri@command{#3}
249 \trimstring\@asuri@command
250 \IfBeginWith\@asuri@command{drop}{
251 \StrBehind{\@asuri@command}{drop}[\@asuri@command]
252 \trimstring\@asuri@command
253 \IfStrEq\@asuri@command{query}{
254 \makeuri{\csname #2scheme\endcsname}%
255 {\csname #2authority\endcsname}%
256 {\csname #2path\endcsname}%
257 \makeuri@empty%
258 {\csname #2fragment\endcsname}%
259 \@asuri@changed@true
260 }{
261 \IfStrEq\@asuri@command{fragment}{
262 \makeuri{\csname #2scheme\endcsname}%
263 {\csname #2authority\endcsname}%
264 {\csname #2path\endcsname}%
265 {\csname #2query\endcsname}%
266 \makeuri@empty%
267 \@asuri@changed@true
268 }{}}
269 }{
270 \IfBeginWith\@asuri@command{\@Slash}{
271 \@cpath{\csname #2path\endcsname\@asuri@command}
272 \makeuri{\csname #2scheme\endcsname}%
273 {\csname #2authority\endcsname}%
274 {\@CanPath}%
275 \makeuri@empty%
276 \makeuri@empty%
277 \@asuri@changed@true
278 }{
279 \IfBeginWith\@asuri@command{\@QuestionMark}{
280 \expandafter\ifx\csname #2query\endcsname\makeuri@empty

```

```

281         \StrBehind\@asuri@command\@QuestionMark[\@asuri@command]
282         \edef\@asuri@nquery{\@asuri@command}
283     \else
284         \edef\@asuri@nquery{\csname #2query\endcsname\@asuri@command}
285     \fi
286     \makeuri{\csname #2scheme\endcsname}%
287         {\csname #2authority\endcsname}%
288         {\csname #2path\endcsname}%
289         {\@asuri@nquery}%
290     \makeuri@empty%
291     \@asuri@changed@true
292 }{
293 \IfBeginWith\@asuri@command{\@Fragment}{
294     \expandafter\ifx\csname #2fragment\endcsname\makeuri@empty
295     \StrBehind\@asuri@command\@Fragment[\@asuri@command]
296     \edef\@asuri@nfrag{\@asuri@command}
297 \else
298     \edef\@asuri@nfrag{\csname #2fragment\endcsname\@asuri@command}
299 \fi
300     \makeuri{\csname #2scheme\endcsname}%
301         {\csname #2authority\endcsname}%
302         {\csname #2path\endcsname}%
303         {\csname #2query\endcsname}%
304         {\@asuri@nfrag}%
305     \@asuri@changed@true
306 }{}
307 }}}
308 \edef\@asuri@ncs{#1}
309 \if@asuri@changed@
310     \ifx\@asuri@ncs\@empty
311         \asuri{#2}\makeuri@uri
312     \else
313         \asuri\@asuri@ncs\makeuri@uri
314     \fi
315 \fi
316 }
317

```

auxiliary code:

```

318 \def\@Space{ }
319 \def\trimstring#1{
320     \edef\pathsuris@trim@temp{#1}
321     \IfBeginWith\pathsuris@trim@temp\@Space{
322         \StrGobbleLeft\pathsuris@trim@temp1[#1]
323         \trimstring{#1}
324     }{
325         \IfEndWith\pathsuris@trim@temp\@Space{
326             \StrGobbleRight\pathsuris@trim@temp1[#1]
327             \trimstring{#1}
328         }{

```

```

329         \edef#1{\pathsuris@trim@temp}
330     }
331 }
332 }
333
334 % windows paths
335
336 \catcode'\.=0
337 .catcode'\.=12
338 .let.\@BackSlash\
339 .catcode'\.=0
340 \catcode'\.=12
341
342 \newif\if@windowstopath@inpath@
343 \def\windows@to@path#1{
344     \@windowstopath@inpath@false
345     \def\windows@temp{ }
346     \edef\windows@path{#1}
347     \ifx\windows@path\empty\else
348         \expandafter\windows@path@loop\windows@path\windows@path@end
349     \fi
350     \let#1\windows@temp
351 }
352 \def\windows@path@loop#1#2\windows@path@end{
353     \def\windows@temp@b{#2}
354     \ifx\windows@temp@b\empty
355         \def\windows@continue{ }
356     \else
357         \def\windows@continue{\windows@path@loop#2\windows@path@end}
358     \fi
359     \if@windowstopath@inpath@
360         \ifx#1\@BackSlash
361             \edef\windows@temp{\windows@temp\@Slash}
362         \else
363             \edef\windows@temp{\windows@temp#1}
364         \fi
365     \else
366         \ifx#1:
367             \edef\windows@temp{\@Slash\windows@temp}
368             \@windowstopath@inpath@true
369         \else
370             \edef\windows@temp{\windows@temp#1}
371         \fi
372     \fi
373     \windows@continue
374 }
375
376 \def\path@to@windows#1{
377     \@windowstopath@inpath@false
378     \def\windows@temp{ }

```

```

379 \edef\windows@path{#1}
380 \edef\windows@path{\expandafter\@gobble\windows@path}
381 \ifx\windows@path\@empty\else
382 \expandafter\path@windows@loop\windows@path\windows@path@end
383 \fi
384 \let#1\windows@temp
385 }
386 \def\path@windows@loop#1#2\windows@path@end{
387 \def\windows@temp@b{#2}
388 \ifx\windows@temp@b\@empty
389 \def\windows@continue{}
390 \else
391 \def\windows@continue{\path@windows@loop#2\windows@path@end}
392 \fi
393 \if@windowstopath@inpath@
394 \ifx#1/
395 \edef\windows@temp{\windows@temp\@BackSlash}
396 \else
397 \edef\windows@temp{\windows@temp#1}
398 \fi
399 \else
400 \ifx#1/
401 \edef\windows@temp{\windows@temp:\@BackSlash}
402 \@windowstopath@inpath@true
403 \else
404 \edef\windows@temp{\windows@temp#1}
405 \fi
406 \fi
407 \windows@continue
408 }
409
410 \end{package}

```