Ruby trunk - Bug #7310

URI::FTP API inconsistency

11/08/2012 08:13 PM - t3hk0d3 (Igor Yamolov)

Status:	Rejected		
Priority:	Normal		
Assignee:			
Target version:	2.6		
ruby -v:	1.9.3p194	**Backport:**	

Description

1.9.3p194 :012 > test = URI.parse("http://test/")
=> #
1.9.3p194 :013 > test.path
=> "/"
1.9.3p194 :014 > test = URI.parse("ftp://test/")
=> #
1.9.3p194 :015 > test.path
=> ""

History

#1 - 11/08/2012 08:18 PM - t3hk0d3 (Igor Yamolov)

In other words URI::HTTP returning/expecting absolute path, while URI::FTP returning/expecting relative path.

#2 - 11/08/2012 08:49 PM - knu (Akinori MUSHA)

- Status changed from Open to Rejected

Please read the RFCs, especially RFC 1738 and see how the path
part in FTP URI is defined.

tl;dr - URI('ftp://host/etc/motd').path == "etc/motd" and
URI('ftp://host/%2Fetc/motd').path == "/etc/motd"

To retrieve a file at "ftp://host/cwd1/cwd2/file", an FTP client
logs in to the server "host", issues "CWD cwd1", "CWD cwd2" and
then "RETR file". So, the path part of the URI should naturally
be taken as a relative path "cwd1/cwd2/file" not an absolute path
starting with a slash. To say that you want a file
at "/cwd1/cwd2/file" on the host, the URI must be composed
as "ftp://host/%2Fcwd1/cwd2/file" in which the starting slash is
percent encoded as required by RFC 1738.

This is what existing programs expect and rely on. We shouldn't
change that. It's just that each scheme has its own set of
rules, which sometimes looks inconsistent.

#3 - 11/09/2012 03:40 PM - naruse (Yui NARUSE)

knu (Akinori MUSHA) wrote:

> Please read the RFCs, especially RFC 1738 and see how the path
> part in FTP URI is defined.

RFC 1738 is obsoleted by RFC 2396 and no RFCs define FTP URI scheme now.

A latest FTP URI scheme draft defines ftp-path, but there is no path
http://tools.ietf.org/html/draft-yevstifeyev-ftp-uri-scheme-08

#4 - 11/09/2012 04:09 PM - knu (Akinori MUSHA)

naruse (Yui NARUSE) wrote:

> knu (Akinori MUSHA) wrote:

> > Please read the RFCs, especially RFC 1738 and see how the path
> > part in FTP URI is defined.
>
> RFC 1738 is obsoleted by RFC 2396 and no RFCs define FTP URI scheme now.

As far as I know, (part of) RFC 1738 was just "updated" by RFC

1. It seems that it was actually obsoleted by RFC 4248 and 4266, neither of which however is about FTP.

So, my understanding is that developers who deal with FTP scheme
URIs would still look into and use RFC 1738 as one of references.

> A latest FTP URI scheme draft defines ftp-path, but there is no path
> http://tools.ietf.org/html/draft-yevstifeyev-ftp-uri-scheme-08

Thanks for the pointer. It could be used as a source when it is
officially published.

#5 - 11/10/2012 04:33 PM - duerst (Martin Dürst)

knu (Akinori MUSHA) wrote:

> naruse (Yui NARUSE) wrote:
>
> > knu (Akinori MUSHA) wrote:
> >
> > > Please read the RFCs, especially RFC 1738 and see how the path
> > > part in FTP URI is defined.
> >
> > RFC 1738 is obsoleted by RFC 2396 and no RFCs define FTP URI scheme now.
>
> As far as I know, (part of) RFC 1738 was just "updated" by RFC
>
> 1. It seems that it was actually obsoleted by RFC 4248 and 4266, neither of which however is about FTP.

"obsolete" is strange here. ftp: and file: are still only defined in RFC 1738 (but see below).

> So, my understanding is that developers who deal with FTP scheme
> URIs would still look into and use RFC 1738 as one of references.

Most probably not! What's much more important is that the "U" in URI stands for "uniform". URI schemes have a lot of common syntax. http: and ftp: are a good example. For the common syntax, RFC 1738 is almost useless. This is very much improved in RFC 2396 and later RFC 3986. So using RFC 1738 for URI syntax is not really appropriate.

Also, please have a look at http://tantek.com/2011/238/b1/many-ways-slice-url-name-pieces. There are many differences between the various libraries, but one of the positive conclusions was the following:
"path has been used consistently for the past 10+ years and in a way consistent with its operating system roots." There's a good reason for this: Having a slash at the start means that the path is absolute. No slash means relative.

Now imagine Ruby being listed at http://tantek.com/2011/238/b1/many-ways-slice-url-name-pieces, and needing two rows, because http: and ftp: are treated differently. It would look really, really bad. So please fix this as early as possible!

> > A latest FTP URI scheme draft defines ftp-path, but there is no path
> > http://tools.ietf.org/html/draft-yevstifeyev-ftp-uri-scheme-08
>
> Thanks for the pointer. It could be used as a source when it is
> officially published.

Please don't wait for this. RFC 2396 (and RFC 3986) updated the general syntax already. Mykyta Yevstifeyev isn't currently active in the IETF, and nobody else is currently working on this document, but including the first "/" in the path is not something that has ever been questioned.

So the only problem with changing this may be backwards compatibility, but I think this can be covered by a warning.

#6 - 11/10/2012 06:50 PM - naruse (Yui NARUSE)

duerst (Martin Dürst) wrote:

> knu (Akinori MUSHA) wrote:
>
> > naruse (Yui NARUSE) wrote:
> >
> > > knu (Akinori MUSHA) wrote:
> > >
> > > > Please read the RFCs, especially RFC 1738 and see how the path
> > > > part in FTP URI is defined.
> > >
> > > RFC 1738 is obsoleted by RFC 2396 and no RFCs define FTP URI scheme now.
> >
> > As far as I know, (part of) RFC 1738 was just "updated" by RFC
>
> 1. It seems that it was actually obsoleted by RFC 4248 and 4266, neither of which however is about FTP.

Ah, yes. what you say is correct.

> "obsolete" is strange here. ftp: and file: are still only defined in RFC 1738 (but see below).

It is strange but it is true.
RFC 4248 and RFC 4266 says it obsoletes RFC 1738.

> So, my understanding is that developers who deal with FTP scheme
> URIs would still look into and use RFC 1738 as one of references.

> Most probably not! What's much more important is that the "U" in URI stands for "uniform". URI schemes have a lot of common syntax. http: and ftp: are a good example.

No, it is a bad example.
Real ftp: scheme shows that ftp URI is a bit different world from http's.

> For the common syntax, RFC 1738 is almost useless. This is very much improved in RFC 2396 and later RFC 3986. So using RFC 1738 for URI syntax is not really appropriate.

The clear one is RFC 3986 doesn't have typecode.
typecode is defined in RFC 1738 and many servers and clients support it.

> Also, please have a look at http://tantek.com/2011/238/b1/many-ways-slice-url-name-pieces. There are many differences between the various libraries, but one of the positive conclusions was the following:
> "path has been used consistently for the past 10+ years and in a way consistent with its operating system roots." There's a good reason for this: Having a slash at the start means that the path is absolute. No slash means relative.

It is http's consensus, not ftp's.

> Now imagine Ruby being listed at http://tantek.com/2011/238/b1/many-ways-slice-url-name-pieces, and needing two rows, because http: and ftp: are treated differently. It would look really, really bad. So please fix this as early as possible!

Where you live is the bad world.

> > A latest FTP URI scheme draft defines ftp-path, but there is no path
> > http://tools.ietf.org/html/draft-yevstifeyev-ftp-uri-scheme-08
>
> Thanks for the pointer. It could be used as a source when it is
> officially published.

> Please don't wait for this. RFC 2396 (and RFC 3986) updated the general syntax already. Mykyta Yevstifeyev isn't currently active in the IETF, and nobody else is currently working on this document, but including the first "/" in the path is not something that has ever been questioned.

Ah yes, the draft doesn't describe the real world..
What I should show is http://suika.fam.cx/~wakaba/wiki/sw/n/ftp%3A#anchor-1
and it still say there is a consensus while there is not such thing.

For example this "/" of path and cwd problem,
the server is vsftpd 3.0.0 on FreeBSD 9.0.
following format is "* browser name and version: related filesystem path.

ftp://foo/

- IE9: /
- Fx15: ~
- Opera10: ~
- Chrome 23.0.1271.64: ~

ftp://foo//

- IE9: /
- Fx15: /
- Opera10: /
- Chrome 23.0.1271.64: ~

ftp://foo/%2F

- IE9: /
- Fx15: /
- Opera10: /
- Chrome 23.0.1271.64: ~

ftp://foo//bin

- IE9: /bin
- Fx15: /bin
- Opera10: /bin
- Chrome 23.0.1271.64: ~/bin

ftp://foo/bar

- IE9: /bar
- Fx15: ~/bar
- Opera10: ~/bar
- Chrome 23.0.1271.64: ~/bar

> So the only problem with changing this may be backwards compatibility, but I think this can be covered by a warning.

So the fundamental problem is there is no standard and consensus.

#7 - 11/10/2012 07:35 PM - knu (Akinori MUSHA)

duerst (Martin Dürst) wrote:

> Most probably not! What's much more important is that the "U" in URI stands for "uniform". URI schemes have a lot of common syntax. http: and ftp: are a good example. For the common syntax, RFC 1738 is almost useless. This is very much improved in RFC 2396 and later RFC 3986. So using RFC 1738 for URI syntax is not really appropriate.

Good point.

> Also, please have a look at http://tantek.com/2011/238/b1/many-ways-slice-url-name-pieces. There are many differences between the various libraries, but one of the positive conclusions was the following:
> "path has been used consistently for the past 10+ years and in a way consistent with its operating system roots." There's a good reason for this: Having a slash at the start means that the path is absolute. No slash means relative.

Having a slash at the start is syntactically mandatory because there exists no separator defined between the authority part and the path part.
The FTP scheme is an example of how it is used being different from how it looks.

> Now imagine Ruby being listed at http://tantek.com/2011/238/b1/many-ways-slice-url-name-pieces, and needing two rows, because http: and ftp: are treated differently. It would look really, really bad. So please fix this as early as possible!

I'd recommend using Addressable::URI for those who value proactive standard conformance as well as fancy features.
It implements a #path method that always returns the part part with a leading slash.

> So the only problem with changing this may be backwards compatibility, but I think this can be covered by a warning.

This only problem is the most important. As I explained, the feature has been there for a long time and the behavior can be considered reasonable

from a usability standpoint.

I agree that there could be some other method like URI::FTP#local_path and #path could return the full path part regardless of the scheme, but how would you make that happen while taking good care of compatibility?