Description
=begin
Kernel.require on 1.9 is far saner than 1.8 with respect to normalizing path names before storing them in `$`. To use the example from the rdoc, require 'a' and require './a' are now correctly regarded as identical, causing 'a' to be loaded only once. (I have a separate ticket open to update the rdoc).

However, there are still a couple of edge cases when paths aren't normalised.

```ruby
>> require '/tmp/../tmp/c.rb'
c.rb
=> true
>> require '/tmp/c.rb'
c.rb
=> true
>> require '/../../..../tmp/c.rb'
c.rb
=> true
>> $_.grep /c.rb/
=> ['/tmp/tmp/c.rb', '/tmp/c.rb', '/../..../tmp/c.rb']
```

Another is that multiple consecutive separators are not collapsed:

```ruby
>> require '/tmp/c.rb'
c.rb
=> true
>> require '/tmp//c.rb'
c.rb
=> true
>> $_.grep /c.rb/
=> ['/tmp/c.rb', '/tmp//c.rb']
```

However, tilde expansion is handled correctly:

```ruby
>> require '/tmp/c'
c.rb
=> true
>> require '~/../../tmp/c'
=> false
>> $_.grep /c.rb/
=> ['/tmp/c.rb']
```

In all cases the path should be expanded and made absolute before being stored in `$`. This would presumably aid performance, avoid files being inadvertently loaded multiple times, and be more consistent.

(As an aside, all filenames in `$` on my system are absolute apart from 'enumerator.so', which is relative. It would be more consistent if all such paths were absolute.)
=end

History
#1 - 06/14/2009 10:57 AM - nobu (Nobuyoshi Nakada)
- Status changed from Open to Closed
In addition, require './a.rb' and require '../a.rb' should be regarded as identical on Windows.

Hi,

In message "Re: [ruby-core:23848] [Bug #1627] Kernel.require Should Canonicalise Paths" on Sun, 14 Jun 2009 11:00:48 +0900, Heesob Park redmine@ruby-lang.org writes:

|In addition, require './a.rb' and require '../a.rb' should be regarded as identical on Windows.

No, never use backslash path separator for #require.

matz.

(As an aside, all filenames in $* on my system are absolute apart from 'enumerator.so', which is relative. It would be more consistent if all such paths were absolute.)

enumerator.so in $* is a compatibility hack, which will eventually be removed in the future. It's now built-in, so it does not have any absolute path.

matz.