Ruby trunk - Feature #15363

Case insensitive file systems - add info to CONFIG or somewhere?

11/30/2018 11:34 PM - MSP-Greg (Greg L)

Status:	Open
Priority:	Normal
Assignee:	
Target version:	

Description

I was under the impression that Windows was the only case insensitive file system. That is not the case. In RubyGems, there are places in both code and tests where this needs to be accounted for.

Although none come to mind, the same may exist here. I haven't seen anything defining it's state. If I'm not mistaken, could something like the following be added, maybe as an additional CONFIG key or constant somewhere?

```
if __FILE__ != __FILE__.downcase
  FS_CASE_INSENS = File.exists?(__FILE__.downcase)
elsif __FILE__ != __FILE.upcase
  FS_CASE_INSENS = File.exists?(__FILE__.upcase)
else
  FS_CASE_INSENS = true # indeterminate? assume true?
end
p "FS_CASE_INSENS #{FS_CASE_INSENS}"
```

Thanks, Greg

History

#1 - 12/01/2018 03:51 PM - Hanmac (Hans Mackowiak)

i don't think that works that easy because it depends on the filesystem which depends on the location of the file.

For example it might be that it's case insensitive in one location and sensitive on another.

#2 - 12/01/2018 04:27 PM - ahorek (Pavel Rosický)

it looks like Ruby checks only a platform for the case-(in)sensivity

for example on WSL I can create a file in my home folder which is case-sensitive

```
touch aAa.txt
touch aaa.txt
```

this creates two separate files

if I do the same in /mnt/c/* which is a remote case-insensitive NTFS filesystem. It creates only one.

on Windows

```
irb(main):002:0> Dir['c:/aaa.txt']
=> ["c:/aAa.txt"]
```

on WSL

```
irb(main):002:0> Dir['/mnt/c/aaa.txt']
=> [] (wrong)

irb(main):002:0> Dir['/mnt/c/aAa.txt']
=> ['/mnt/c/aAa.txt']
```

note that Ruby has a significant perf impact on case-insensitive filesystem already. In order to support this feature, Ruby would have to check filesystem capabilities on the fly and on every single FS operation that matters. For that reason I don't care much about fixing it.

#3 - 12/01/2018 04:36 PM - MSP-Greg (Greg L)

ahorek (Pavel Rosický) & Hans,

Thanks for the feedback. Both of you are right, there isn't a trivial solution to this.

Ok to close. The code had an error in it also...

#4 - 12/01/2018 04:41 PM - ahorek (Pavel Rosický)

unlike ruby, shell's glob finds a file, but it doesn't return a real filename as a result.

```
ls /mnt/c/aaa.txt
=> /mnt/c/aaa.txt (wrong)

ls /mnt/c/aAa.txt
=> /mnt/c/aAa.txt
```

https://ruby-doc.org/core-2.5.0/Dir.html
Note that the pattern is not a regexp, it's closer to a shell glob. See File.fnmatch for the meaning of the flags parameter. Case sensitivity depends on your system (File::FNM_CASEFOLD is ignored), as does the order in which the results are returned.

#5 - 12/02/2018 12:37 PM - nobu (Nobuyoshi Nakada)

Case-sensitiveness is runtime environment property, and may differ per mounted volumes.

#6 - 12/02/2018 01:01 PM - nobu (Nobuyoshi Nakada)

Now I can't remember why File::FNM_SYSCASE is 0 on macOS..., maybe it can select case-sensitive filesystems.

#7 - 12/02/2018 03:10 PM - MSP-Greg (Greg L)

Now, I'm pretty much only windows, but previously the network I used had Windows, MacOS/OSX, and a few of *nix NAS devices. I vaguely recall some casing issues. Unfortunately, I can't test anything like that now.

Maybe the questions are:

1. Should Ruby attempt to account for case insensitive file systems? The main issue I've come across is in comparisons using ENV variables that contain paths.

2. If #1 is yes, is that (reasonably) possible?

#8 - 12/03/2018 12:00 AM - nobu (Nobuyoshi Nakada)

You can't compare path names just by case-insensitiveness, on Windows.
Consider short file names.
You can compare them by File.identical? instead, or should use File.expand_path before comparison at least.

#9 - 12/03/2018 03:54 AM - MSP-Greg (Greg L)

nobu (Nobuyoshi Nakada) wrote:

> You can't compare path names just by case-insensitiveness, on Windows.
> Consider short file names.
> You can compare them by File.identical? instead, or should use File.expand_path before comparison at least.

Thank you for the mention of File.indentical?, as I have never noticed that method. Using File.exist? does correctly identify Windows as being case sensitive, but will give a false positive if, for instance, one has folders /Ruby and /ruby on a case sensitive file system. Unlikely, but possible...

I'm not sure about the effect of short file names. If test is a string, and File.exist? test is true, then the following should work?

```
fs_case_insens =
  (test != test.upcase   && File.identical?(test, test.upcase  )) ||
  (test != test.downcase && File.identical?(test, test.downcase)
```

#10 - 12/03/2018 04:30 AM - nobu (Nobuyoshi Nakada)

MSP-Greg (Greg L) wrote:

> I'm not sure about the effect of short file names. If test is a string, and File.exist? test is true, then the following should work?
>
> ```
> fs_case_insens =
>   (test != test.upcase   && File.identical?(test, test.upcase  )) ||
>   (test != test.downcase && File.identical?(test, test.downcase)
> ```

As far as test contains alphabets, yes, probably.
Or maybe File.identical?(test, test.swapcase).