# Ruby - Bug #1685

## Some windows unicode path issues remain

06/24/2009 07:36 PM - spatulasnout (B Kelly)

| | | | |
|---|---|---|---|
| **Status:** | Closed | | |
| **Priority:** | Normal | | |
| **Assignee:** | usa (Usaku NAKAMURA) | | |
| **Target version:** | 2.6 | | |
| **ruby -v:** | ruby 1.9.2dev (2009-06-24) [i386-mswin32_71] | **Backport:** | |

**Description**

=begin
Hi,

I see some nice progress has been made in unicode path handling on windows.

The following tests are not exhaustive, but do reveal some remaining issues.

Everything below "NOT WORKING" fails in one way or another.

Regards,

Bill

```
# encoding: UTF-8

# Test unicode path/dir handling on windows

require 'test/unit'

class TestUnicodeFilenamesAndPaths < Test::Unit::TestCase
  def setup
    tmpdir = ENV['TEMP'] || "C:/TEMP"
    Dir.chdir tmpdir
    puts Dir.pwd
    testdir = "ruby_unicode_test"
    Dir.mkdir testdir unless test ?d, testdir
    Dir.chdir testdir
    puts Dir.pwd
  end

  def test_unicode_paths
    fname_resume = "R\xC3\xA9sum\xC3\xA9".force_encoding("UTF-8")
    fname_chinese = "\u52ec\u52ee\u52f1\u52f2.txt"
    dname_chinese = "\u52ec\u52ee\u52f1\u52f2"

    assert_equal( "UTF-8", fname_resume.encoding.name )
    File.open(fname_resume, "w") {|io| io.puts "Hello, World"}

    assert_equal( "UTF-8", fname_chinese.encoding.name )
    File.open(fname_chinese, "w") {|io| io.puts "Hello, World"}

    dat = File.read(fname_chinese)
    assert_equal( "Hello, World\n", dat )

    files = Dir["*"]
    assert( files.include? fname_resume )
    assert( files.include? fname_chinese )

# NOT WORKING:
```

```
    Dir.rmdir dname_chinese rescue nil
    Dir.mkdir dname_chinese
    test ?d, dname_chinese
    Dir.chdir dname_chinese
    cwd = Dir.pwd
    assert( cwd[(-dname_chinese.length)..-1] == dname_chinese )
    Dir.chdir ".."

    x = File.stat(fname_resume)
    x = File.stat(fname_chinese)
    x = File.stat(dname_chinese)

    assert( File.exist? fname_resume )
    assert( File.exist? fname_chinese )
    assert( test(?f, fname_resume) )
    assert( test(?f, fname_chinese) )

    files = Dir[fname_resume]
    assert_equal( fname_resume, files.first )
    files = Dir[fname_chinese]
    assert_equal( fname_chinese, files.first )
    files = Dir[dname_chinese]
    assert_equal( dname_chinese, files.first )
  end
end
=end
```

**Related issues:**

| | | |
|---|---|---|
| Related to Ruby - Feature #2255: unicode parameters cannot be passed to ruby | **Closed** | |
| Related to Ruby - Bug #2332: Ruby doesn't run properly from unicode folder on... | **Closed** | **11/04/2009** |
| Related to Ruby - Bug #1771: system()/popen()/popen3() & windows & unicode is... | **Closed** | **07/13/2009** |
| Has duplicate Ruby - Bug #2137: Dir.glob does not support unicode on Windows | **Closed** | **09/23/2009** |

### Associated revisions

**Revision 6c28f99d8894b9f9a3c1394d120115f69012f2c3 - 04/30/2010 05:56 PM - usa (Usaku NAKAMURA)**

- merge some patches from win32-uncode-test branch.
  see #1685.
- file.c, include/ruby/intern.h (rb_str_encode_ospath): new function
  to convert encoding for pathname.
- win32.c, include/ruby/win32.h (rb_w32_ulink, rb_w32_urename,
  rb_w32_ustati64, rb_w32_uopen, rb_w32_uutime, rb_w32_uchdir,
  rb_w32_umkdir, rb_w32_urmdir, rb_w32_uunlink): new functions to
  accept UTF-8 path.
- win32/win32.c (rb_w32_opendir, link, rb_w32_stati64, rb_w32_utime,
  rb_w32_unlink): use WCHAR path internally.
- file.c (rb_stat, eaccess, access_internal, rb_file_s_ftype,
  chmod_internal, rb_file_chmod, rb_file_chown, utime_internal,
  rb_file_s_link, unlink_internal, rb_file_s_rename): use UTF-8 version
  functions on Win32.
- file.c (apply2files, rb_stat, rb_file_s_lstat, rb_file_symlink_p,
  rb_file_readable_p, rb_file_writable_p, rb_file_executable_p,
  check3rdbyte, rb_file_identical_p, rb_file_chmod, rb_file_chown,
  rb_file_s_link, rb_file_s_symlink, rb_file_s_rename): call
  rb_str_encode_ospath() before passing the path to system.
- io.c (rb_sysopen): ditto.
- dir.c (dir_chdir, dir_s_mkdir, dir_s_rmdir): ditto.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@27570 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

**Revision 6c28f99d8894b9f9a3c1394d120115f69012f2c3 - 04/30/2010 05:56 PM - usa (Usaku NAKAMURA)**

- merge some patches from win32-uncode-test branch.
  see #1685.
- file.c, include/ruby/intern.h (rb_str_encode_ospath): new function
  to convert encoding for pathname.
- win32.c, include/ruby/win32.h (rb_w32_ulink, rb_w32_urename,
  rb_w32_ustati64, rb_w32_uopen, rb_w32_uutime, rb_w32_uchdir,
  rb_w32_umkdir, rb_w32_urmdir, rb_w32_uunlink): new functions to
  accept UTF-8 path.
- win32/win32.c (rb_w32_opendir, link, rb_w32_stati64, rb_w32_utime,
  rb_w32_unlink): use WCHAR path internally.
- file.c (rb_stat, eaccess, access_internal, rb_file_s_ftype,
  chmod_internal, rb_file_chmod, rb_file_chown, utime_internal,
  rb_file_s_link, unlink_internal, rb_file_s_rename): use UTF-8 version
  functions on Win32.
- file.c (apply2files, rb_stat, rb_file_s_lstat, rb_file_symlink_p,
  rb_file_readable_p, rb_file_writable_p, rb_file_executable_p,
  check3rdbyte, rb_file_identical_p, rb_file_chmod, rb_file_chown,
  rb_file_s_link, rb_file_s_symlink, rb_file_s_rename): call
  rb_str_encode_ospath() before passing the path to system.
- io.c (rb_sysopen): ditto.
- dir.c (dir_chdir, dir_s_mkdir, dir_s_rmdir): ditto.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@27570 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

**Revision 6c28f99d - 04/30/2010 05:56 PM - U.Nakamura**

- merge some patches from win32-uncode-test branch.
  see #1685.
- file.c, include/ruby/intern.h (rb_str_encode_ospath): new function
  to convert encoding for pathname.
- win32.c, include/ruby/win32.h (rb_w32_ulink, rb_w32_urename,
  rb_w32_ustati64, rb_w32_uopen, rb_w32_uutime, rb_w32_uchdir,
  rb_w32_umkdir, rb_w32_urmdir, rb_w32_uunlink): new functions to
  accept UTF-8 path.
- win32/win32.c (rb_w32_opendir, link, rb_w32_stati64, rb_w32_utime,
  rb_w32_unlink): use WCHAR path internally.
- file.c (rb_stat, eaccess, access_internal, rb_file_s_ftype,
  chmod_internal, rb_file_chmod, rb_file_chown, utime_internal,
  rb_file_s_link, unlink_internal, rb_file_s_rename): use UTF-8 version
  functions on Win32.
- file.c (apply2files, rb_stat, rb_file_s_lstat, rb_file_symlink_p,
  rb_file_readable_p, rb_file_writable_p, rb_file_executable_p,
  check3rdbyte, rb_file_identical_p, rb_file_chmod, rb_file_chown,
  rb_file_s_link, rb_file_s_symlink, rb_file_s_rename): call
  rb_str_encode_ospath() before passing the path to system.
- io.c (rb_sysopen): ditto.
- dir.c (dir_chdir, dir_s_mkdir, dir_s_rmdir): ditto.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@27570 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

## History

#### #1 - 07/16/2009 04:10 PM - yugui (Yuki Sonoda)

*- Status changed from Open to Assigned*

*- Assignee set to usa (Usaku NAKAMURA)*

=begin

=end

#### #2 - 08/12/2009 03:48 PM - usa (Usaku NAKAMURA)

*- Priority changed from Normal to 5*

=begin

=end

#### #3 - 01/22/2010 10:01 PM - vo.x (Vit Ondruch)

=begin
Is there any progress regarding this issue(s)?
=end

#### #4 - 03/25/2010 10:13 AM - spatulasnout (B Kelly)

*- File spatulasnout-unicode-mkdir-diffs.txt added*

*- File test_io_unicode_paths.rb added*

=begin
Hi,

I'll be needing win32 unicode path support for my current project, so I would like to try to tackle the remaining issues.

I started with a relatively easy one, Dir.mkdir

For Dir.mkdir, I took an approach similar to what was already in place for rb_sysopen(), which is that it tries to call w32_conv_to_utf16() on the path, and if it succeeds calls the new rb_w32_wmkdir() with the wide path; otherwise it falls back to calling the old rb_w32_mkdir().

Attached files should include the diffs, and a new file adding a bootstrap test for unicode paths.  (The tests currently fail, because they need a working unicode stat and unlink in order to function.)

I'm planning to attempt File.stat next, but I have some questions about it so I'll post separately.

Regards,

Bill

=end

#### #5 - 03/25/2010 05:01 PM - vo.x (Vit Ondruch)

=begin
Hello Bill,

Are you aware of win32_unicode_branch? Its not up-to-date as far as I know, but there is lot of Unicode functionality covered. There is missing mainly Dir.glob functionality.

Vit
=end

#### #6 - 03/25/2010 07:10 PM - spatulasnout (B Kelly)

=begin
Hi Vit,

Thanks.  Wow.  Good to know.

win32/win32.c:rb_w32_uchown(const char *path, int owner, int group)
win32/win32.c:rb_w32_ulink(const char *from, const char *to)
win32/win32.c:rb_w32_urename(const char *from, const char *to)
win32/win32.c:rb_w32_ustati64(const char *path, struct stati64 *st)
win32/win32.c:rb_w32_uaccess(const char *path, int mode)
win32/win32.c:rb_w32_uopen(const char *file, int oflag, ...)

win32/win32.c:rb_w32_uutime(const char *path, const struct utimbuf *times)
win32/win32.c:rb_w32_utime(const char *path, const struct utimbuf *times)
win32/win32.c:rb_w32_uchdir(const char *path)
win32/win32.c:rb_w32_umkdir(const char *path, int mode)
win32/win32.c:rb_w32_urmdir(const char *path)
win32/win32.c:rb_w32_uunlink(const char *path)
win32/win32.c:rb_w32_unlink(const char *path)
win32/win32.c:rb_w32_uchmod(const char *path, int mode)

And it looks like a much cleaner implementation than what
is currently in the 1.9.2 trunk.  No more 'wchar' in
sysopen_struct, no more #ifdef _WIN32 surrounding
w32_conv_to_utf16 logic, just some defines at the top:

dir.c:#define chdir(p) rb_w32_uchdir(p)
dir.c:#define mkdir(p, m) rb_w32_umkdir(p, m)
dir.c:#define rmdir(p) rb_w32_urmdir(p)
file.c:#define STAT(p, s)       rb_w32_ustati64(p, s)
file.c:#define lstat(p, s)      rb_w32_ustati64(p, s)
file.c:#define access(p, m)     rb_w32_uaccess(p, m)
file.c:#define chmod(p, m)      rb_w32_uchmod(p, m)
file.c:#define chown(p, o, g)   rb_w32_uchown(p, o, g)
file.c:#define utime(p, t)      rb_w32_uutime(p, t)
file.c:#define link(f, t)       rb_w32_ulink(f, t)
file.c:#define unlink(p)        rb_w32_uunlink(p)
file.c:#define rename(f, t)     rb_w32_urename(f, t)
io.c:#define open       rb_w32_uopen

I wonder if there is a reason this should not be merged
into trunk ASAP?

Regards,

Bill

=end

#### #7 - 03/25/2010 10:28 PM - usa (Usaku NAKAMURA)

=begin
Hello,

In message "[ruby-core:28979] [Bug #1685] Some windows unicode path issues remain"
on Mar.25,2010 19:10:35, redmine@ruby-lang.org wrote:

> I wonder if there is a reason this should not be merged
> into trunk ASAP?

Because I'm too busy to test this branch well :(

Endoh-san says that the feature freeze is March 31.
Then, it is necessary to complete merging it until then,
if we want to include it in 1.9.2 release...

win32-unicode-branch has not contained the globbing features
yet, as Vit pointed in [ruby-core:28977] (thank you, Vit).
However, because it relates to the command line interpretation,
it might be difficult to implement until March 31.
Should we wait until all functions are covered, or merge the
current one?

Summary:
(1) need the decision whether merging it or not
(2) need testers :)
(3) need the worker(s) to make the patch to trunk

## Regards,

U.Nakamura usa@garbagecollect.jp

=end

#### #8 - 03/25/2010 10:57 PM - vo.x (Vit Ondruch)

=begin
For me, it would be helpful to merge what we have now. I am not aware of any problematic parts with methods which are already implemented. However, I am aware that this can lead in confusion when not everything will work with unicode :(

Vit
=end

#### #9 - 03/28/2010 04:51 PM - spatulasnout (B Kelly)

=begin
Hi,

U.Nakamura wrote:

> Hello,
>
> In message "[ruby-core:28979] [Bug #1685] Some windows unicode path issues remain"
> on Mar.25,2010 19:10:35, redmine@ruby-lang.org wrote:
>
> > I wonder if there is a reason this should not be merged
> > into trunk ASAP?
>
> Because I'm too busy to test this branch well :(
>
> Endoh-san says that the feature freeze is March 31.
> Then, it is necessary to complete merging it until then,
> if we want to include it in 1.9.2 release...
>
> win32-unicode-branch has not contained the globbing features
> yet, as Vit pointed in [ruby-core:28977] (thank you, Vit).
> However, because it relates to the command line interpretation,
> it might be difficult to implement until March 31.

I understand how this might be considered a 'feature', but
I think it is also possible to consider it a bug fix.

1.9.1 was supposed to support unicode path on win32, but
this was deferred to 1.9.2.

Nevertheless, I quote matz from November, 2008:

Yukihiro Matsumoto wrote:

> Hi,
>
> In message "Re: [ruby-core:20109] Re: 1.9, encoding & win32 wide char support"
> on Wed, 26 Nov 2008 12:26:53 +0900, "Bill Kelly" billk@cts.com writes:
>
> |> Does anyone have information as to the current status of
> |> adding Unicode-savvy path handling to 1.9 ruby?
> |
> |Ugh.  Sorry, I mean of course: Unicode-savvy path handling
> |on *win32* ruby 1.9.
>
> Every path encoding is UTF-8 and converted to UTF-16 internally.  If
> there's something still use *A functions, it will eventually replaced
> by *W functions.  In short, if you're using UTF-8 for your program
> encoding, you should not see any problem  (if you do, it's a bug).
>
> ```
>                                                 matz.
> ```

I don't know if matz has changed his mind, but; personally I would
like to consider it a bug that ruby 1.9.x fails for unicode paths
on windows.

> Should we wait until all functions are covered, or merge the
> current one?
>
> Summary:
> (1) need the decision whether merging it or not
> (2) need testers :)
> (3) need the worker(s) to make the patch to trunk

(1) Please, yes. Let us merge. 93.75% is better than current 6.25% coverage.
(2) I hope to contribute unicode_path unit-tests. (such as in bootstraptest/)
(3) I would like to contribute to the patch if my efforts can be useful.
(diffs on io.c, file.c, and dir.c look pretty straightforward.)
(diffs on win32/win32.c look more difficult, but I can attempt.)

Regards,

Bill

=end

#### #10 - 03/28/2010 11:57 PM - usa (Usaku NAKAMURA)

=begin
Hello,

In message "[ruby-core:29082] Re: [Bug #1685] Some windows unicode path issues remain"
on Mar.28,2010 16:51:26, billk@cts.com wrote:

> I understand how this might be considered a 'feature', but
> I think it is also possible to consider it a bug fix.

Hmm, it has a point in it.
The branch manager should judge whether this change is a bug fix or
feature change.
How do you think, Yugui-san?

> Summary:
> (1) need the decision whether merging it or not
> (2) need testers :)
> (3) need the worker(s) to make the patch to trunk

> (1) Please, yes. Let us merge. 93.75% is better than current 6.25% coverage.
> (2) I hope to contribute unicode_path unit-tests. (such as in bootstraptest/)
> (3) I would like to contribute to the patch if my efforts can be useful.
> (diffs on io.c, file.c, and dir.c look pretty straightforward.)
> (diffs on win32/win32.c look more difficult, but I can attempt.)

I'm very glad to hear your offer of cooperation.
Thank you!

## Regards,

U.Nakamura usa@garbagecollect.jp

=end

#### #11 - 04/01/2010 03:16 AM - spatulasnout (B Kelly)

=begin
U.Nakamura wrote:

> Hello,
>
> In message "[ruby-core:29082] Re: [Bug #1685] Some windows unicode path issues remain"
> on Mar.28,2010 16:51:26, billk@cts.com wrote:
>
> > I understand how this might be considered a 'feature', but
> > I think it is also possible to consider it a bug fix.
>
> Hmm, it has a point in it.
> The branch manager should judge whether this change is a bug fix or
> feature change.
> How do you think, Yugui-san?

Any word?

Regards,

Bill

=end

#### #12 - 04/01/2010 10:30 AM - yugui (Yuki Sonoda)

=begin

> The branch manager should judge whether this change is a bug fix or
> feature change.
> How do you think, Yugui-san?

It's a bug fix.
=end

#### #13 - 04/01/2010 02:52 PM - spatulasnout (B Kelly)

=begin
Yuki Sonoda wrote:

> Issue #1685 has been updated by Yuki Sonoda.
>
> > The branch manager should judge whether this change is a bug fix or
> > feature change.
> > How do you think, Yugui-san?
>
> It's a bug fix.

Wonderful news!

Thank you,

Bill

=end

#### #14 - 04/30/2010 08:12 AM - spatulasnout (B Kelly)

=begin
Hi,

Bill Kelly wrote:

> Yuki Sonoda wrote:
>
> > Issue #1685 has been updated by Yuki Sonoda.
> >
> > > The branch manager should judge whether this change is a bug fix or
> > > feature change.
> > > How do you think, Yugui-san?
> >
> > It's a bug fix.
>
> Wonderful news!

In order to avoid duplication of effort, I wanted to inquire
whether anyone else may currently be working on Windows
Unicode related code?

U.Nakamura wrote:

> (2) need testers :)
> (3) need the worker(s) to make the patch to trunk

If there is no conflict with others' work, I would like to
attempt merging the win32-unicode branch into trunk within
the next week or two.

Regards,

Bill

=end

#### #15 - 05/05/2010 03:49 AM - usa (Usaku NAKAMURA)

=begin
Hello,

In message "[ruby-core:29892] Re: [Bug #1685] Some windows unicode path issues remain"
on Apr.30,2010 08:12:33, billk@cts.com wrote:
| In order to avoid duplication of effort, I wanted to inquire
| whether anyone else may currently be working on Windows
| Unicode related code?
|
|
| U.Nakamura wrote:
| >
| >  (2) need testers :)
| >  (3) need the worker(s) to make the patch to trunk
|
|
| If there is no conflict with others' work, I would like to
| attempt merging the win32-unicode branch into trunk within
| the next week or two.

Ah, I've merged most parts of win32-unicode-test branch because
the time limit of code freeze (Apr.30) has come.

# See r27570

Of course, test cases and bug reports are welcomed.

## Regards

U.Nakamura usa@garbagecollect.jp

=end

#### #16 - 05/05/2010 03:35 PM - spatulasnout (B Kelly)

=begin
U.Nakamura wrote:

> Hello,
>
> In message "[ruby-core:29892] Re: [Bug #1685] Some windows unicode path issues remain"
> on Apr.30,2010 08:12:33, billk@cts.com wrote:
> |
> | If there is no conflict with others' work, I would like to
> | attempt merging the win32-unicode branch into trunk within
> | the next week or two.
>
> Ah, I've merged most parts of win32-unicode-test branch because
> the time limit of code freeze (Apr.30) has come.
>
> # See r27570

Oh!  Thank you very much!

(I had thought the code freeze applied to new features, rather
than bug fixes.)

> Of course, test cases and bug reports are welcomed.

My initial attempt at a bootstraptest for unicode path
support is failing.

It is incomplete, but I uploaded the current version:

http://redmine.ruby-lang.org/attachments/download/910

It is failing at:

DNAME_CHINESE = "\u52ec\u52ee\u52f1\u52f2"
Dir.mkdir DNAME_CHINESE
test(?d, DNAME_CHINESE) or raise "test ?d fail"

It seems rb_stat in file.c calls stat(), but stat does
not map to the unicode version.

win32.h:

#define stat(path,st)  rb_w32_stat(path,st)

file.c:

static int
rb_stat(VALUE file, struct stat *st)
{
VALUE tmp;

```
 rb_secure(2);
 tmp = rb_check_convert_type(file, T_FILE, "IO", "to_io");
 if (!NIL_P(tmp)) {
rb_io_t *fptr;

GetOpenFile(tmp, fptr);
return fstat(fptr->fd, st);
 }
 FilePathValue(file);
 file = rb_str_encode_ospath(file);
 return stat(StringValueCStr(file), st);
```

}

Regards,

Bill

=end

#### #17 - 05/05/2010 03:57 PM - usa (Usaku NAKAMURA)

=begin
Hello,

In message "[ruby-core:30012] Re: [Bug #1685] Some windows unicode path issues remain"
on May.05,2010 15:35:11, billk@cts.com wrote:
| My initial attempt at a bootstraptest for unicode path
| support is failing.
|
| It is incomplete, but I uploaded the current version:
|
| http://redmine.ruby-lang.org/attachments/download/910
|
| It is failing at:
|
|   DNAME_CHINESE = "\u52ec\u52ee\u52f1\u52f2"
|   Dir.mkdir DNAME_CHINESE
|   test(?d, DNAME_CHINESE) or raise "test ?d fail"
|
|
| It seems rb_stat in file.c calls stat(), but stat does
| not map to the unicode version.

Oops, thank you!

## Regards

U.Nakamura usa@garbagecollect.jp

=end

#### #18 - 05/06/2010 07:39 PM - spatulasnout (B Kelly)

=begin
U.Nakamura wrote:

> In message "[ruby-core:30012] Re: [Bug #1685] Some windows unicode path issues remain"
> on May.05,2010 15:35:11, billk@cts.com wrote:
> |
> | It seems rb_stat in file.c calls stat(), but stat does
> | not map to the unicode version.
>
> Oops, thank you!

Thanks, the test gets much further now.

It now fails at the last line:

Dir.chdir DNAME_CHINESE
cwd = Dir.pwd
( cwd[(-DNAME_CHINESE.length)..-1] == DNAME_CHINESE ) or raise "cwd check fail"

Currently there was only rb_w32_getcwd. I have added a unicode
rb_w32_ugetcwd:

```
Index: include/ruby/win32.h
===================================================================
--- include/ruby/win32.h (revision 27644)
+++ include/ruby/win32.h (working copy)
@@ -254,6 +254,7 @@
 extern struct servent  *WSAAPI rb_w32_getservbyport(int, const char *);
 extern int    rb_w32_socketpair(int, int, int, int *);
 extern char * rb_w32_getcwd(char *, int);
+extern char * rb_w32_ugetcwd(char *, int);
 extern char * rb_w32_getenv(const char *);
 extern int    rb_w32_rename(const char *, const char *);
 extern int    rb_w32_urename(const char *, const char *);
@@ -611,7 +612,7 @@
 #define get_osfhandle(h) rb_w32_get_osfhandle(h)

 #undef getcwd
-#define getcwd(b, s)  rb_w32_getcwd(b, s)
+#define getcwd(b, s)  rb_w32_ugetcwd(b, s)

 #undef getenv
 #define getenv(n)  rb_w32_getenv(n)
Index: win32/win32.c
===================================================================
--- win32/win32.c (revision 27644)
+++ win32/win32.c (working copy)
@@ -3692,6 +3692,57 @@
     return p;
 }

+char *
+rb_w32_ugetcwd(char *buffer, int size)
+{
+    char *p;
+    WCHAR *wp;
+    long len, wlen;
+
+    wlen = GetCurrentDirectoryW(0, NULL);  // wlen includes null terminating character
+    if (!wlen) {
+ errno = map_errno(GetLastError());
+ return NULL;
+    }
+
+    wp = malloc(wlen * sizeof(WCHAR));
+    if (!wp) {
+ errno = ENOMEM;
+ return NULL;
+    }
+
+    if (!GetCurrentDirectoryW(wlen, wp)) {
+ errno = map_errno(GetLastError());
```

```
+ free(wp);
+ 	return NULL;
+     }
+
+     p = wstr_to_utf8(wp, &len);
+     free(wp);
+     len += 1;  // len now includes null terminating character
+
+     if (!p) {
+ errno = ENOMEM;
+ return NULL;
+     }
+
+     if (buffer) {
+ if (size < len) {
+ 	free(p);
+ 	errno = ERANGE;
+ 	return NULL;
+ }
+
+ memcpy(buffer, p, len);
+ free(p);
+ p = buffer;
+     }
+
+     translate_char(p, '\\', '/');
+
+     return p;
+}
+
 int
 chown(const char *path, int owner, int group)
 {
```

This works, in terms of returning a UTF-8 path string; however, rb_dir_getwd calls rb_enc_associate(cwd, rb_filesystem_encoding()) on the result, associating the WINDOWS-1252 encoding instead of UTF-8.

So, I would like to ask: is there a reason enc_set_filesystem_encoding() should not return UTF-8 now for Windows?

```
static int
enc_set_filesystem_encoding(void)
{
int idx;
#if defined NO_LOCALE_CHARMAP
idx = rb_enc_to_index(rb_default_external_encoding());
#elif defined _WIN32 || defined CYGWIN
char cp[sizeof(int) * 8 / 3 + 4];
snprintf(cp, sizeof cp, "CP%d", AreFileApisANSI() ? GetACP() : GetOEMCP());
idx = rb_enc_find_index(cp);
if (idx < 0) idx = rb_ascii8bit_encindex();
#else
idx = rb_enc_to_index(rb_default_external_encoding());
#endif

 enc_alias_internal("filesystem", idx);
 return idx;

}
```

It seems strange that it still selects non-unicode encodings.

---

Also, my bootstraptest encountered one more problem. The mktmpdir can't delete the unicode directory entries created by my test:

P:/code/ruby-svn/trunk/lib/fileutils.rb:1307:in unlink': Invalid argument - C:/temp/bootstraptest20100505-1016-1lvss6a.tmpwd/???? (Errno::EINVAL)
from P:/code/ruby-svn/trunk/lib/fileutils.rb:1307:in block in remove_file'
from P:/code/ruby-svn/trunk/lib/fileutils.rb:1315:in platform_support' from P:/code/ruby-svn/trunk/lib/fileutils.rb:1306:in remove_file'
from P:/code/ruby-svn/trunk/lib/fileutils.rb:1295:in remove' from P:/code/ruby-svn/trunk/lib/fileutils.rb:761:in block in remove_entry'
from P:/code/ruby-svn/trunk/lib/fileutils.rb:1345:in block (2 levels) in postorder_traverse' from P:/code/ruby-svn/trunk/lib/fileutils.rb:1349:in postorder_traverse'

from P:/code/ruby-svn/trunk/lib/fileutils.rb:1344:in block in postorder_traverse' from P:/code/ruby-svn/trunk/lib/fileutils.rb:1343:in each'
from P:/code/ruby-svn/trunk/lib/fileutils.rb:1343:in postorder_traverse' from P:/code/ruby-svn/trunk/lib/fileutils.rb:759:in remove_entry'
from P:/code/ruby-svn/trunk/lib/fileutils.rb:688:in remove_entry_secure' from P:/code/ruby-svn/trunk/lib/tmpdir.rb:85:in ensure in mktmpdir'
from P:/code/ruby-svn/trunk/lib/tmpdir.rb:85:in mktmpdir' from ./bootstraptest/runner.rb:375:in in_temporary_working_directory'
from ./bootstraptest/runner.rb:126:in main' from ./bootstraptest/runner.rb:398:in '

I don't have a patch for this yet.  However, it looks like
in win32.c, routines such as rb_w32_opendir and rb_w32_readdir_with_enc
are already using WCHAR internally!

For example:

DIR *
rb_w32_opendir(const char *filename)
{
struct stati64 sbuf;
WIN32_FIND_DATAW fd;
HANDLE fh;
WCHAR *wpath;

```
 if (!(wpath = filecp_to_wstr(filename, NULL)))
return NULL;
```

... so it seems if filesystem encoding were considered UTF-8
instead of WINDOWS-1252, then opendir might just work.

Similarly (somewhat) with rb_w32_readdir_with_enc.  (At least,
it does call readdir_internal, which uses WCHAR.)

So I *think* these are very close to working UTF-8, but, again,
I don't understand why enc_set_filesystem_encoding() uses
WINDOWS-1252 still.

Thanks,

Regards,

Bill

=end

#### #19 - 05/06/2010 07:58 PM - usa (Usaku NAKAMURA)

=begin
Hello,

In message "[ruby-core:30052] Re: [Bug #1685] Some windows unicode path issues remain"
on May.06,2010 19:39:27, billk@cts.com wrote:

> This works, in terms of returning a UTF-8 path string; however,
> rb_dir_getwd calls rb_enc_associate(cwd, rb_filesystem_encoding())
> on the result, associating the WINDOWS-1252 encoding instead of
> UTF-8.
>
> So, I would like to ask: is there a reason
> enc_set_filesystem_encoding() should not return UTF-8 now for
> Windows?

For compatibility.

I will not change filesystem encoding in Windows in 1.9 series.
In all methods which returns filenames, the default encoding
of returned value must be filesystem encoding.
So, if someone want to get filename with another encoding, he/she
should specify the encoding by some way.
Of course, it is necessary to decide the "some way" of each
methods.

> Also, my bootstraptest encountered one more problem.  The mktmpdir
> can't delete the unicode directory entries created by my test:

Yes, I know it.
This is the problem of globbing.

I've already decided to solve this problem 1.9.3 or later.

## Regards,

U.Nakamura usa@garbagecollect.jp

=end

#### #20 - 05/06/2010 09:38 PM - spatulasnout (B Kelly)

=begin
Hi,

U.Nakamura wrote:

> In message "[ruby-core:30052] Re: [Bug #1685] Some windows unicode path issues remain"
> on May.06,2010 19:39:27, billk@cts.com wrote:
>
> > This works, in terms of returning a UTF-8 path string; however,
> > rb_dir_getwd calls rb_enc_associate(cwd, rb_filesystem_encoding())
> > on the result, associating the WINDOWS-1252 encoding instead of
> > UTF-8.
> >
> > So, I would like to ask: is there a reason
> > enc_set_filesystem_encoding() should not return UTF-8 now for
> > Windows?
>
> For compatibility.
>
> I will not change filesystem encoding in Windows in 1.9 series.
> In all methods which returns filenames, the default encoding
> of returned value must be filesystem encoding.
> So, if someone want to get filename with another encoding, he/she
> should specify the encoding by some way.
> Of course, it is necessary to decide the "some way" of each
> methods.

Ah.

So my rb_w32_ugetcwd patch is not very useful, at present,
since there is no "some way" to specify the encoding via
Dir.pwd.

May I suggest a new command line flag for this purpose:

ruby --DEAR_GOD_WORK_WITH_UTF_8_DAMN_IT

;)

Well then, this becomes a philosophical question at this point,
but in an effort to better understand, I am wondering:

How does it break compatibility, if we allow filesystem encoding
to become UTF-8 when rb_default_external_encoding is UTF-8?

Do we have evidence that anyone has written scripts that will
break in such a case?  (And if so, can we agree to summon the
fleas of a thousand camels to infest their undergarments?)

> > Also, my bootstraptest encountered one more problem.  The mktmpdir
> > can't delete the unicode directory entries created by my test:
>
> Yes, I know it.
> This is the problem of globbing.
> I've already decided to solve this problem 1.9.3 or later.

OK.

I admit I don't understand why it's considered a globbing problem.
Does the UTF-8 support somehow make the globbing more difficult?
I thought it was just the same situation as above: a filesystem

encoding problem?

Regards,

Bill

=end

#### #21 - 05/06/2010 10:27 PM - usa (Usaku NAKAMURA)

=begin
Hello,

In message "[ruby-core:30054] Re: [Bug #1685] Some windows unicode path issues remain"
on May.06,2010 21:38:19, billk@cts.com wrote:

> For compatibility.
>
> I will not change filesystem encoding in Windows in 1.9 series.
> In all methods which returns filenames, the default encoding
> of returned value must be filesystem encoding.
> So, if someone want to get filename with another encoding, he/she
> should specify the encoding by some way.
> Of course, it is necessary to decide the "some way" of each
> methods.

Ah.

So my rb_w32_ugetcwd patch is not very useful, at present,
since there is no "some way" to specify the encoding via
Dir.pwd.

Unfortunately...

> Well then, this becomes a philosophical question at this point,
> but in an effort to better understand, I am wondering:
>
> How does it break compatibility, if we allow filesystem encoding
> to become UTF-8 when rb_default_external_encoding is UTF-8?

You should advocate using default_internal instead of
default_external :)
It's acceptable for me.

> I admit I don't understand why it's considered a globbing problem.

FileUtils uses Dir.entries to get filenames to remove.

> Does the UTF-8 support somehow make the globbing more difficult?
> I thought it was just the same situation as above: a filesystem
> encoding problem?

Yes, you are right.

## Regards,

U.Nakamura usa@garbagecollect.jp

=end

#### #22 - 05/07/2010 06:11 AM - spatulasnout (B Kelly)

=begin
Hi,

U.Nakamura wrote:

> How does it break compatibility, if we allow filesystem encoding
> to become UTF-8 when rb_default_external_encoding is UTF-8?

> You should advocate using default_internal instead of
> default_external :)
> It's acceptable for me.

Ah, thanks, default_internal does make more sense. :)

Regarding advocacy: apart from yourself, who are the people
who need to comment on this? Is this a question for Matz?
Or... ?

Thanks,

Bill

=end

#### #23 - 05/07/2010 11:56 AM - usa (Usaku NAKAMURA)

=begin
Hello,

In message "[ruby-core:30071] Re: [Bug #1685] Some windows unicode path issues remain"
on May.07,2010 06:11:00, billk@cts.com wrote:

> Regarding advocacy: apart from yourself, who are the people
> who need to comment on this? Is this a question for Matz?
> Or... ?

I assume,

1. matz: the grand designer of Ruby
2. naruse: an authority of Ruby M17N
3. me: main maintainer of Ruby on Windows
4. all users of Ruby, of course, especially people using
   non-unicode (multibyte) environment

## Regards,

U.Nakamura usa@garbagecollect.jp

=end

#### #24 - 05/07/2010 01:12 PM - spatulasnout (B Kelly)

=begin
Hi,

U.Nakamura wrote:

> I assume,
>
> 1. matz: the grand designer of Ruby
> 2. naruse: an authority of Ruby M17N
> 3. me: main maintainer of Ruby on Windows
> 4. all users of Ruby, of course, especially people using
>    non-unicode (multibyte) environment

For #1 (sorry, I can't resist :)

http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/20110

For #4, wouldn't we expect people using a non-unicode environment
to *not* set their default_internal encoding to UTF-8 ? So I
would think they would not be affected.

---

As an aside, from the point of view of writing ruby software to
be installed on arbitrary user's machines, I don't think there's
any such thing as a non-unicode environment anymore.

The minute any of my users (even if they are English speaking)

downloads a file from their web browser called
[redacted].mpg
my application which uses Dir.entries to locate and catalog
media files, is now broken on their system.

(Of course, since I distribute the Ruby interpreter with my
application, I have the luxury of working around the problem,
by installing a non-standard Ruby.  But I still believe it's
important for standard Ruby to have full Unicode support on
Windows.)

Regards,

Bill

=end

#### #25 - 05/18/2010 07:31 PM - spatulasnout (B Kelly)

=begin
Hi,

Bill Kelly wrote:

> 1. matz: the grand designer of Ruby
> 2. naruse: an authority of Ruby M17N
> 3. me: main maintainer of Ruby on Windows
> 4. all users of Ruby, of course, especially people using
>    non-unicode (multibyte) environment
>
> For #4, wouldn't we expect people using a non-unicode environment
> to *not* set their default_internal encoding to UTF-8 ?  So I
> would think they would not be affected.

Noticed an interesting ChangeLog entry from yesterday on
ruby_1_9_2 branch:

Mon May 17 11:09:58 2010  NAKAMURA Usaku  usa@ruby-lang.org

```
    merge from trunk (r27856, r27857)

    * lib/fileutils.rb (FileUtils::Entry_#entries): returns pathname in
      UTF-8 on Windows to allow FileUtils accessing all pathnames
      internally.
```

# Index: lib/fileutils.rb

--- lib/fileutils.rb    (revision 27657)
+++ lib/fileutils.rb    (working copy)
@@ -1176,7 +1176,9 @@
end

```
  def entries
```

- [redacted]
- [redacted]
- [redacted]
- [redacted]

```
      .reject {|n| n == '.' or n == '..' }\
      .map {|n| Entry_.new(prefix(), join(rel(), n.untaint)) }
```

    end
    ==========

Would this approach also be considered for Dir.pwd:

result = Dir.pwd(:encoding => "UTF-8")

?

If so, I already have the rb_w32_ugetcwd implementation (presented in [ruby-core:30052] ).

I would be happy to provide a patch for Dir.pwd if this is acceptable.

Regards,

Bill

=end

#### #26 - 05/18/2010 10:07 PM - usa (Usaku NAKAMURA)

=begin
Hello,

In message "[ruby-core:30296] Re: [Bug #1685] Some windows unicode path issues remain"
on May.18,2010 19:30:53, billk@cts.com wrote:

> Noticed an interesting ChangeLog entry from yesterday on
> ruby_1_9_2 branch:
>
> Mon May 17 11:09:58 2010  NAKAMURA Usaku  usa@ruby-lang.org
>
> ```
>     merge from trunk (r27856, r27857)
>
>     * lib/fileutils.rb (FileUtils::Entry_#entries): returns pathname in
>       UTF-8 on Windows to allow FileUtils accessing all pathnames
>       internally.
> ```

In this case, Dir.entries already has its own "some way".
So I can use it.

> Would this approach also be considered for Dir.pwd:
>
> result = Dir.pwd(:encoding => "UTF-8")
>
> ?

This might be a moot point.
For instance, there might be an insistence that Dir.pwd
should accept only the encoding because there is no possiblity
that it takes other arguments.

## Regards,

U.Nakamura usa@garbagecollect.jp

=end

#### #27 - 05/19/2010 04:49 PM - spatulasnout (B Kelly)

=begin
Hi,

U.Nakamura wrote:

> In message "[ruby-core:30296] Re: [Bug #1685] Some windows unicode path issues remain"
> on May.18,2010 19:30:53, billk@cts.com wrote:
>
> > Would this approach also be considered for Dir.pwd:
> >
> > result = Dir.pwd(:encoding => "UTF-8")
> >
> > ?
>
> This might be a moot point.
> For instance, there might be an insistence that Dir.pwd
> should accept only the encoding because there is no possiblity

> that it takes other arguments.

Any solution would be fine with me.  :)

Thanks to your finding a solution for Dir.entries, it seems
we are approaching nearly 100% unicode path capability for
win32!

Do you anticipate it will be difficult to reach a decision
regarding:

result = Dir.pwd(:encoding => "UTF-8")
vs.
result = Dir.pwd("UTF-8")
vs.
(some other way)

?

Thanks,

Regards,

Bill

=end

#### #28 - 06/03/2010 10:29 AM - usa (Usaku NAKAMURA)

*- Category changed from core to M17N*

*- Priority changed from 5 to Normal*

*- Target version changed from 1.9.2 to 2.0.0*

=begin

=end

#### #29 - 12/09/2012 09:40 PM - mame (Yusuke Endoh)

*- Description updated*

Usa-san, what's the status?

--
Yusuke Endoh mame@tsg.ne.jp

#### #30 - 02/18/2013 09:07 PM - mame (Yusuke Endoh)

*- Target version changed from 2.0.0 to 2.6*

Usa-san, what's the status?

--
Yusuke Endoh mame@tsg.ne.jp

#### #31 - 04/03/2014 02:07 PM - thomthom (Thomas Thomassen)

B Kelly wrote:

> =begin
> Thanks to your finding a solution for Dir.entries, it seems
> we are approaching nearly 100% unicode path capability for
> win32!
> =end

In Ruby 2.0 there appear to still be several issues with Ruby and Unicode characters in filenames. Dir.entries fail, load and require fail. **FILE** has the wrong encoding. I see some things slated for Ruby 2.2, but not everything.

#### #32 - 04/04/2014 10:38 AM - duerst (Martin Dürst)

On 2014/04/03 23:07, thomas@thomthom.net wrote:

> Issue #1685 has been updated by Thomas Thomassen.
>
> In Ruby 2.0 there appear to still be several issues with Ruby and Unicode characters in filenames. Dir.entries fail, load and require fail. **FILE** has the wrong encoding. I see some things slated for Ruby 2.2, but not everything.

If you know of anything that's not yet in Ruby 2.2, please tell us, best by opening a bug for each issue.

Regards, Martin.

---

> Bug #1685: Some windows unicode path issues remain
> https://bugs.ruby-lang.org/issues/1685#change-46061
>
> - Author: B Kelly
> - Status: Assigned
> - Priority: Normal
> - Assignee: Usaku NAKAMURA
> - Category: M17N
> - Target version: next minor
> - ruby -v: ruby 1.9.2dev (2009-06-24) [i386-mswin32_71]
> - Backport:

---

> =begin
> Hi,
>
> I see some nice progress has been made in unicode path handling on windows.
>
> The following tests are not exhaustive, but do reveal some remaining issues.
>
> Everything below "NOT WORKING" fails in one way or another.
>
> Regards,
>
> Bill

```
# encoding: UTF-8

# Test unicode path/dir handling on windows

require 'test/unit'

class TestUnicodeFilenamesAndPaths < Test::Unit::TestCase
  def setup
    tmpdir = ENV['TEMP'] || "C:/TEMP"
    Dir.chdir tmpdir
    puts Dir.pwd
    testdir = "ruby_unicode_test"
    Dir.mkdir testdir unless test ?d, testdir
    Dir.chdir testdir
    puts Dir.pwd
  end

  def test_unicode_paths
    fname_resume = "R\xC3\xA9sum\xC3\xA9".force_encoding("UTF-8")
    fname_chinese = "\u52ec\u52ee\u52f1\u52f2.txt"
    dname_chinese = "\u52ec\u52ee\u52f1\u52f2"

    assert_equal( "UTF-8", fname_resume.encoding.name )
    File.open(fname_resume, "w") {|io| io.puts "Hello, World"}

    assert_equal( "UTF-8", fname_chinese.encoding.name )
    File.open(fname_chinese, "w") {|io| io.puts "Hello, World"}

    dat = File.read(fname_chinese)
    assert_equal( "Hello, World\n", dat )

    files = Dir["*"]
```

```
    assert( files.include? fname_resume )
    assert( files.include? fname_chinese )

# NOT WORKING:
    Dir.rmdir dname_chinese rescue nil
    Dir.mkdir dname_chinese
    test ?d, dname_chinese
    Dir.chdir dname_chinese
    cwd = Dir.pwd
    assert( cwd[(-dname_chinese.length)..-1] == dname_chinese )
    Dir.chdir ".."

    x = File.stat(fname_resume)
    x = File.stat(fname_chinese)
    x = File.stat(dname_chinese)

    assert( File.exist? fname_resume )
    assert( File.exist? fname_chinese )
    assert( test(?f, fname_resume) )
    assert( test(?f, fname_chinese) )

    files = Dir[fname_resume]
    assert_equal( fname_resume, files.first )
    files = Dir[fname_chinese]
    assert_equal( fname_chinese, files.first )
    files = Dir[dname_chinese]
    assert_equal( dname_chinese, files.first )
  end
end
=end


---Files--------------------------------
spatulasnout-unicode-mkdir-diffs.txt (3.56 KB)
test_io_unicode_paths.rb (925 Bytes)
```

#### #33 - 04/04/2014 11:45 AM - thomthom (Thomas Thomassen)

Martin Dürst wrote:

> If you know of anything that's not yet in Ruby 2.2, please tell us, best
> by opening a bug for each issue.

I've been setting up tests and running them through Ruby 2.2 I find some are fixed but there is still several issues related to file handling. We'll be filing issues for what we have uncovered.

#### #34 - 04/07/2014 06:50 PM - usa (Usaku NAKAMURA)

*- Status changed from Assigned to Closed*

This ticket is too old and too various problems.
Now Thomas investigates many things and is making some new tickets. (Thank you!)
Please refer to them from now on.

### Files

| | | | |
|---|---|---|---|
| spatulasnout-unicode-mkdir-diffs.txt | 3.56 KB | 03/25/2010 | spatulasnout (B Kelly) |
| test_io_unicode_paths.rb | 925 Bytes | 03/25/2010 | spatulasnout (B Kelly) |