Redmine - Defect #30411

Filesystem adapter does not show correct size for large files

2019-01-11 23:44 - Felix Schäfer

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Jean-Philippe Lang	**% Done:**	0%
Category:	SCM	**Estimated time:**	0.00 hour
Target version:	4.0.2		
Resolution:	Fixed	**Affected version:**	4.0.0

Description

We have a couple of large files (~9GB) which size doesn't show correctly.

I suspect source:/tags/4.0.0/lib/redmine/scm/adapters/filesystem_adapter.rb#L84, the conversion to a signed 32-bit Integer (https://apidock.com/ruby/Array/pack), to introduce errors for file sizes that are a little over 2GB.

Changing l and L to q and Q should solve the problem.

Related issues:

Related to Redmine - Defect #12279: error with large files in filesystem repo	**Closed**

Associated revisions

Revision 17831 - 2019-01-20 08:39 - Jean-Philippe Lang

Filesystem adapter does not show correct size for large files (#30411).

Revision 17832 - 2019-01-20 08:40 - Jean-Philippe Lang

Merged r17831 to 4.0-stable (#30411).

Revision 17833 - 2019-01-20 08:40 - Jean-Philippe Lang

Merged r17831 to 3.4-stable (#30411).

History

#1 - 2019-01-12 07:13 - Go MAEDA

The expression "[File.size(target)].pack('l').unpack('L').first" was introduced in r1509.

According to the commit log, the purpose of the expression is to fix the issue that the adapter shows a negative value for large files under Win32. Maybe it returns -2147483648 for a file with 2GB size and -1 for a file with (4G - 1) bytes. The expression converts -2147483648 and -1 to 2147483648 and 4294967295.

However, if we change the 'l' and 'L' to 'q' and 'Q', -2147483648 and -1 are converted to 18446744071562067968 and 18446744073709551615 (correct values are 2147483648 and 4294967295). So, I think we have to fix this issue in a different way.

The following patch should fix the problem. It converts the value using pack and unpack only if File.size returns negative value.

```
Index: lib/redmine/scm/adapters/filesystem_adapter.rb
===================================================================
--- lib/redmine/scm/adapters/filesystem_adapter.rb    (revision 17791)
+++ lib/redmine/scm/adapters/filesystem_adapter.rb    (working copy)
@@ -76,12 +76,16 @@
                    not File.basename(e1).match(/^\.+$/) # avoid . and ..
                p1         = File.readable?(t1) ? relative_path : ""
                utf_8_path = scm_iconv('UTF-8', @path_encoding, p1)
+               size       = File.directory?(t1) ? nil : File.size(t1)
+               # File#size returns a negative value under Win32 when
+               # the file size is >= 2GB and < 4GB
+               size = [size].pack('l').unpack('L').first if size.to_i < 0
                entries <<
                  Entry.new({ :name => scm_iconv('UTF-8', @path_encoding, File.basename(e1)),
                              # below : list unreadable files, but dont link them.
                              :path => utf_8_path,
                              :kind => (File.directory?(t1) ? 'dir' : 'file'),
-                             :size => (File.directory?(t1) ? nil : [File.size(t1)].pack('l').unpack('L').first),
```

```
+                           :size => size,
                            :lastrev =>
                                Revision.new({:time => (File.mtime(t1)) })
                        })
```

#2 - 2019-01-12 07:14 - Go MAEDA

- Category set to SCM

- Target version set to 3.4.8

#3 - 2019-01-13 16:04 - Go MAEDA

- Related to Defect #12279: error with large files in filesystem repo added

#4 - 2019-01-19 11:54 - Jean-Philippe Lang

- Target version changed from 3.4.8 to Candidate for next minor release

Felix, can you confirm the fix is OK?

#5 - 2019-01-19 16:13 - Felix Schäfer

I can confirm the patch still shows the correct file size for the file that caused problems in our case.

The error the patch in r1509 addresses intrigued me though to I went looking if maybe this was due to a bug in Ruby that was since fixed. Searching on the ruby-lang bug tracker yielded the following result https://bugs.ruby-lang.org/issues/2671

My understanding is that the Ruby bug that caused files above 2GB to be shown as negative File.size values has been fixed in Ruby 1.9. I have searched the ruby git repository for clues but unfortunately I was not able to find a corresponding commit or commit message.

I have however tested the value of File.size for a file that has a size just under 4GB for Ruby 2.5, Ruby 1.9 and Ruby 1.8 on Windows 10. Ruby 1.8 will show a negative file size, while Ruby 1.9 and Ruby 2.5 show the correct file size.

As current Redmine versions require at least Ruby 1.9 I think we can remove this work-around altogether.

#6 - 2019-01-20 08:06 - Jean-Philippe Lang

- Target version changed from Candidate for next minor release to 4.0.2

Thanks!

#7 - 2019-01-20 08:40 - Jean-Philippe Lang

- Status changed from New to Closed

- Assignee set to Jean-Philippe Lang

- Resolution set to Fixed