Ruby trunk - Feature #855

HTTP/1.1 fixes and other enhancements to webrick

12/11/2008 07:08 PM - candlerb (Brian Candler)

Status:	Closed	
Priority:	Normal	
Assignee:	normalperson (Eric Wong)	
Target version:	2.6	

Description

=begin
I raised the following issues on ruby-core:

1. When returning an open IO object (without Content-Length or chunking), Webrick fails to close the HTTP/1.1 connection, and hence the client waits forever for the end of the data.
   http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/18454
   http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/18565

2. Webrick makes it very difficult to send a '100 continue' response when a HTTP/1.1 client requests one, and yet the RFC2616 says it *must* do so.
   http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/18459

3. It would be convenient to be able to stream not just real IO objects, but other objects which duck-type like them (such as an open zip file entry from rubyzip)
   http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/18460

4. It would be convenient to be able to provide a proc object to generate a streamed response:
   proc { |out| out << "data"; out << "more data"; etc }

5. The default block size of 4K is too small to be efficient when serving large files. This was already fixed for 1.9 in
   http://redmine.ruby-lang.org/repositories/revision/ruby-19?rev=10167
   (default now 64K and tunable). Please consider this for 1.8, especially since a similar improvement has been backported for Net::HTTP in
   http://redmine.ruby-lang.org/repositories/revision/ruby-18?rev=12092

The attached file contains small monkey-patches to address these issues. If there is interest these could be rewritten as actual patches against webrick.
=end

Associated revisions

Revision a4fa58f9ab9883a03bf7ed95655501acfb2fa554 - 09/10/2010 10:20 AM - nahi (Hiroshi Nakamura)

- lib/webrick/httprequest.rb (WEBrick::HTTPRequest#continue): add method for generating HTTP/1.1 100 continue response if the client expects it, otherwise does nothing. Patch by Brian Candler.
  ref #855.

```
* test/webrick/test_httprequest.rb: test added.
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@29218 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision a4fa58f9 - 09/10/2010 10:20 AM - nahi (Hiroshi Nakamura)

- lib/webrick/httprequest.rb (WEBrick::HTTPRequest#continue): add method for generating HTTP/1.1 100 continue response if the client expects it, otherwise does nothing. Patch by Brian Candler.
  ref #855.

`* test/webrick/test_httprequest.rb: test added.`

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@29218 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 29218 - 09/10/2010 10:20 AM - nahi (Hiroshi Nakamura)

- lib/webrick/httprequest.rb (WEBrick::HTTPRequest#continue): add method for generating HTTP/1.1 100 continue response if the client expects it, otherwise does nothing. Patch by Brian Candler. ref #855.

  `* test/webrick/test_httprequest.rb: test added.`

Revision 29218 - 09/10/2010 10:20 AM - nahi (Hiroshi Nakamura)

- lib/webrick/httprequest.rb (WEBrick::HTTPRequest#continue): add method for generating HTTP/1.1 100 continue response if the client expects it, otherwise does nothing. Patch by Brian Candler. ref #855.

  `* test/webrick/test_httprequest.rb: test added.`

Revision 29218 - 09/10/2010 10:20 AM - nahi (Hiroshi Nakamura)

- lib/webrick/httprequest.rb (WEBrick::HTTPRequest#continue): add method for generating HTTP/1.1 100 continue response if the client expects it, otherwise does nothing. Patch by Brian Candler. ref #855.

  `* test/webrick/test_httprequest.rb: test added.`

Revision 29218 - 09/10/2010 10:20 AM - nahi (Hiroshi Nakamura)

- lib/webrick/httprequest.rb (WEBrick::HTTPRequest#continue): add method for generating HTTP/1.1 100 continue response if the client expects it, otherwise does nothing. Patch by Brian Candler. ref #855.

  `* test/webrick/test_httprequest.rb: test added.`

Revision 29218 - 09/10/2010 10:20 AM - nahi (Hiroshi Nakamura)

- lib/webrick/httprequest.rb (WEBrick::HTTPRequest#continue): add method for generating HTTP/1.1 100 continue response if the client expects it, otherwise does nothing. Patch by Brian Candler. ref #855.

  `* test/webrick/test_httprequest.rb: test added.`

Revision 29218 - 09/10/2010 10:20 AM - nahi (Hiroshi Nakamura)

- lib/webrick/httprequest.rb (WEBrick::HTTPRequest#continue): add method for generating HTTP/1.1 100 continue response if the client expects it, otherwise does nothing. Patch by Brian Candler. ref #855.

```
* test/webrick/test_httprequest.rb: test added.
```

Revision 4ce15814 - 06/21/2011 12:58 PM - nahi (Hiroshi Nakamura)

- lib/webrick/httpresponse.rb (HTTPResponse#setup_header): Close HTTP/1.1 connection when returning an IO object as response body without setting HTTPResponse#chunked to true. See #855 no.1.
- test/webrick/test_httpserver.rb: Test it.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@32188 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 32188 - 06/21/2011 12:58 PM - nahi (Hiroshi Nakamura)

- lib/webrick/httpresponse.rb (HTTPResponse#setup_header): Close HTTP/1.1 connection when returning an IO object as response body without setting HTTPResponse#chunked to true. See #855 no.1.
- test/webrick/test_httpserver.rb: Test it.

Revision 32188 - 06/21/2011 12:58 PM - nahi (Hiroshi Nakamura)

- lib/webrick/httpresponse.rb (HTTPResponse#setup_header): Close HTTP/1.1 connection when returning an IO object as response body without setting HTTPResponse#chunked to true. See #855 no.1.
- test/webrick/test_httpserver.rb: Test it.

Revision 32188 - 06/21/2011 12:58 PM - nahi (Hiroshi Nakamura)

- lib/webrick/httpresponse.rb (HTTPResponse#setup_header): Close HTTP/1.1 connection when returning an IO object as response body without setting HTTPResponse#chunked to true. See #855 no.1.
- test/webrick/test_httpserver.rb: Test it.

Revision 32188 - 06/21/2011 12:58 PM - nahi (Hiroshi Nakamura)

- lib/webrick/httpresponse.rb (HTTPResponse#setup_header): Close HTTP/1.1 connection when returning an IO object as response body without setting HTTPResponse#chunked to true. See #855 no.1.

- test/webrick/test_httpserver.rb: Test it.

Revision 32188 - 06/21/2011 12:58 PM - nahi (Hiroshi Nakamura)

- lib/webrick/httpresponse.rb (HTTPResponse#setup_header): Close HTTP/1.1 connection when returning an IO object as response body without setting HTTPResponse#chunked to true. See #855 no.1.
- test/webrick/test_httpserver.rb: Test it.

Revision 32188 - 06/21/2011 12:58 PM - nahi (Hiroshi Nakamura)

- lib/webrick/httpresponse.rb (HTTPResponse#setup_header): Close HTTP/1.1 connection when returning an IO object as response body without setting HTTPResponse#chunked to true. See #855 no.1.
- test/webrick/test_httpserver.rb: Test it.

Revision bb88b1aa - 10/30/2017 11:56 PM - normal

webrick: support Proc objects as body responses

- lib/webrick/httpresponse.rb (send_body): call send_body_proc
  (send_body_proc): new method
  (class ChunkedWrapper): new class
- test/webrick/test_httpresponse.rb (test_send_body_proc): new test
  (test_send_body_proc_chunked): ditto
  [Feature #855]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@60584 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 60584 - 10/30/2017 11:56 PM - normalperson (Eric Wong)

webrick: support Proc objects as body responses

- lib/webrick/httpresponse.rb (send_body): call send_body_proc
  (send_body_proc): new method
  (class ChunkedWrapper): new class
- test/webrick/test_httpresponse.rb (test_send_body_proc): new test
  (test_send_body_proc_chunked): ditto
  [Feature #855]

Revision 60584 - 10/30/2017 11:56 PM - normal

webrick: support Proc objects as body responses

- lib/webrick/httpresponse.rb (send_body): call send_body_proc
  (send_body_proc): new method
  (class ChunkedWrapper): new class
- test/webrick/test_httpresponse.rb (test_send_body_proc): new test
  (test_send_body_proc_chunked): ditto
  [Feature #855]

Revision 60584 - 10/30/2017 11:56 PM - normal

webrick: support Proc objects as body responses

- lib/webrick/httpresponse.rb (send_body): call send_body_proc
  (send_body_proc): new method
  (class ChunkedWrapper): new class

- test/webrick/test_httpresponse.rb (test_send_body_proc): new test
  (test_send_body_proc_chunked): ditto
  [Feature #855]

Revision d02f2996 - 12/15/2017 09:55 PM - normal

NEWS: update for WEBrick Proc body responses

Better late than never :x [Feature #855]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@61288 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 61288 - 12/15/2017 09:55 PM - normalperson (Eric Wong)

NEWS: update for WEBrick Proc body responses

Better late than never :x [Feature #855]

Revision 61288 - 12/15/2017 09:55 PM - normal

NEWS: update for WEBrick Proc body responses

Better late than never :x [Feature #855]

Revision 61288 - 12/15/2017 09:55 PM - normal

NEWS: update for WEBrick Proc body responses

Better late than never :x [Feature #855]

Revision c461bdc3 - 03/28/2018 01:54 PM - nagachika (Tomoyuki Chikanaga)

merge revision(s) 60584,62954,62955,62956,62957,62958,62959,63008:

```
webrick: support Proc objects as body responses

* lib/webrick/httpresponse.rb (send_body): call send_body_proc
  (send_body_proc): new method
  (class ChunkedWrapper): new class

* test/webrick/test_httpresponse.rb (test_send_body_proc): new test
  (test_send_body_proc_chunked): ditto
  [Feature #855]

webrick/httpresponse: IO.copy_stream for regular files

Remove the redundant _send_file method since its functionality
is unnecessary with IO.copy_stream.  IO.copy_stream also allows
the use of sendfile under some OSes to speed up copies to
non-TLS sockets.

Testing with "curl >/dev/null" and "ruby -run -e httpd" to
read a 1G file over Linux loopback reveals a reduction from
around ~0.770 to ~0.490 seconds on the client side.

* lib/webrick/httpresponse.rb (send_body_io): use IO.copy_stream
  (_send_file): remove
  [Feature #14237]

webrick: use IO.copy_stream for single range response

This is also compatible with range responses generated
by Rack::File (tested with rack 2.0.3).

* lib/webrick/httpresponse.rb (send_body_io): use Content-Range
* lib/webrick/httpservlet/filehandler.rb (make_partial_content):
  use File object for the single range case
```

```
* test/webrick/test_filehandler.rb (get_res_body): use send_body
  to test result

test/webrick/test_filehandler.rb: stricter multipart range test

We need to ensure we generate compatibile output in
the face of future changes

* test/webrick/test_filehandler.rb (test_make_partial_content):
  check response body

webrick: quiet warning for multi-part ranges

Content-Length is ignored by WEBrick::HTTPResponse even if we
calculate it, so instead we chunk responses to HTTP/1.1 clients
and terminate HTTP/1.0 connections.

* lib/webrick/httpservlet/filehandler.rb (make_partial_content):
  quiet warning

webrick/httpresponse: make ChunkedWrapper copy_stream-compatible

The .write method needs to return the number of bytes written
to avoid confusing IO.copy_stream.

* lib/webrick/httpresponse.rb (ChunkedWrapper#write): return bytes written
  (ChunkedWrapper#<<): return self

webrick: use IO.copy_stream for multipart response

Use the new Proc response body feature to generate a multipart
range response dynamically.  We use a flat array to minimize
object overhead as much as possible; as many ranges may fit
into an HTTP request header.

* lib/webrick/httpservlet/filehandler.rb (multipart_body): new method
  (make_partial_content): use multipart_body

get rid of test error/failure on Windows introduced at r62955

* lib/webrick/httpresponse.rb (send_body_io): use seek if NotImplementedError
  is raised in IO.copy_stream with offset.

* lib/webrick/httpservlet/filehandler.rb (multipart_body): ditto.
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_4@63012 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 63012 - 03/28/2018 01:54 PM - nagachika (Tomoyuki Chikanaga)

merge revision(s) 60584,62954,62955,62956,62957,62958,62959,63008:

```
webrick: support Proc objects as body responses

* lib/webrick/httpresponse.rb (send_body): call send_body_proc
  (send_body_proc): new method
  (class ChunkedWrapper): new class

* test/webrick/test_httpresponse.rb (test_send_body_proc): new test
  (test_send_body_proc_chunked): ditto
  [Feature #855]

webrick/httpresponse: IO.copy_stream for regular files

Remove the redundant _send_file method since its functionality
is unnecessary with IO.copy_stream.  IO.copy_stream also allows
the use of sendfile under some OSes to speed up copies to
non-TLS sockets.

Testing with "curl >/dev/null" and "ruby -run -e httpd" to
read a 1G file over Linux loopback reveals a reduction from
around ~0.770 to ~0.490 seconds on the client side.

* lib/webrick/httpresponse.rb (send_body_io): use IO.copy_stream
  (_send_file): remove
  [Feature #14237]
```

```
webrick: use IO.copy_stream for single range response

This is also compatible with range responses generated
by Rack::File (tested with rack 2.0.3).

* lib/webrick/httpresponse.rb (send_body_io): use Content-Range
* lib/webrick/httpservlet/filehandler.rb (make_partial_content):
  use File object for the single range case
* test/webrick/test_filehandler.rb (get_res_body): use send_body
  to test result

test/webrick/test_filehandler.rb: stricter multipart range test

We need to ensure we generate compatibile output in
the face of future changes

* test/webrick/test_filehandler.rb (test_make_partial_content):
  check response body

webrick: quiet warning for multi-part ranges

Content-Length is ignored by WEBrick::HTTPResponse even if we
calculate it, so instead we chunk responses to HTTP/1.1 clients
and terminate HTTP/1.0 connections.

* lib/webrick/httpservlet/filehandler.rb (make_partial_content):
  quiet warning

webrick/httpresponse: make ChunkedWrapper copy_stream-compatible

The .write method needs to return the number of bytes written
to avoid confusing IO.copy_stream.

* lib/webrick/httpresponse.rb (ChunkedWrapper#write): return bytes written
  (ChunkedWrapper#<<): return self

webrick: use IO.copy_stream for multipart response

Use the new Proc response body feature to generate a multipart
range response dynamically.  We use a flat array to minimize
object overhead as much as possible; as many ranges may fit
into an HTTP request header.

* lib/webrick/httpservlet/filehandler.rb (multipart_body): new method
  (make_partial_content): use multipart_body

get rid of test error/failure on Windows introduced at r62955

* lib/webrick/httpresponse.rb (send_body_io): use seek if NotImplementedError
  is raised in IO.copy_stream with offset.

* lib/webrick/httpservlet/filehandler.rb (multipart_body): ditto.
```

Revision d32a6d37 - 03/28/2018 02:13 PM - usa (Usaku NAKAMURA)

merge revision(s) 60584,62954-62959,63008:

```
webrick: support Proc objects as body responses

* lib/webrick/httpresponse.rb (send_body): call send_body_proc
  (send_body_proc): new method
  (class ChunkedWrapper): new class

* test/webrick/test_httpresponse.rb (test_send_body_proc): new test
  (test_send_body_proc_chunked): ditto
  [Feature #855]

webrick: favor .write over << method

This will make the next change to use IO.copy_stream
easier-to-read.  When we can drop Ruby 2.4 support in a few
years, this will allow us to use writev(2) with multiple
arguments for headers and chunked responses.
```

```
* lib/webrick/cgi.rb (write): new wrapper method
  lib/webrick/httpresponse.rb: (send_header): use socket.write
  (send_body_io): ditto
  (send_body_string): ditto
  (send_body_proc): ditto
  (_write_data): ditto
  (ChunkedWrapper#write): ditto
  (_send_file): ditto
------------------------------------------------------------------------
r62954 | normal | 2018-03-28 17:05:52 +0900 (□, 28 3 2018) | 14 lines

webrick/httpresponse: IO.copy_stream for regular files

Remove the redundant _send_file method since its functionality
is unnecessary with IO.copy_stream.  IO.copy_stream also allows
the use of sendfile under some OSes to speed up copies to
non-TLS sockets.

Testing with "curl >/dev/null" and "ruby -run -e httpd" to
read a 1G file over Linux loopback reveals a reduction from
around ~0.770 to ~0.490 seconds on the client side.

* lib/webrick/httpresponse.rb (send_body_io): use IO.copy_stream
  (_send_file): remove
  [Feature #14237]
------------------------------------------------------------------------
r62955 | normal | 2018-03-28 17:05:57 +0900 (□, 28 3 2018) | 10 lines

webrick: use IO.copy_stream for single range response

This is also compatible with range responses generated
by Rack::File (tested with rack 2.0.3).

* lib/webrick/httpresponse.rb (send_body_io): use Content-Range
* lib/webrick/httpservlet/filehandler.rb (make_partial_content):
  use File object for the single range case
* test/webrick/test_filehandler.rb (get_res_body): use send_body
  to test result
------------------------------------------------------------------------
r62956 | normal | 2018-03-28 17:06:02 +0900 (□, 28 3 2018) | 7 lines

test/webrick/test_filehandler.rb: stricter multipart range test

We need to ensure we generate compatibile output in
the face of future changes

* test/webrick/test_filehandler.rb (test_make_partial_content):
  check response body
------------------------------------------------------------------------
r62957 | normal | 2018-03-28 17:06:08 +0900 (□, 28 3 2018) | 8 lines

webrick: quiet warning for multi-part ranges

Content-Length is ignored by WEBrick::HTTPResponse even if we
calculate it, so instead we chunk responses to HTTP/1.1 clients
and terminate HTTP/1.0 connections.

* lib/webrick/httpservlet/filehandler.rb (make_partial_content):
  quiet warning
------------------------------------------------------------------------
r62958 | normal | 2018-03-28 17:06:13 +0900 (□, 28 3 2018) | 7 lines

webrick/httpresponse: make ChunkedWrapper copy_stream-compatible

The .write method needs to return the number of bytes written
to avoid confusing IO.copy_stream.

* lib/webrick/httpresponse.rb (ChunkedWrapper#write): return bytes written
  (ChunkedWrapper#<<): return self
------------------------------------------------------------------------
r62959 | normal | 2018-03-28 17:06:18 +0900 (□, 28 3 2018) | 9 lines

webrick: use IO.copy_stream for multipart response

Use the new Proc response body feature to generate a multipart
```

```
range response dynamically.  We use a flat array to minimize
object overhead as much as possible; as many ranges may fit
into an HTTP request header.

* lib/webrick/httpservlet/filehandler.rb (multipart_body): new method
  (make_partial_content): use multipart_body

get rid of test error/failure on Windows introduced at r62955

* lib/webrick/httpresponse.rb (send_body_io): use seek if NotImplementedError
  is raised in IO.copy_stream with offset.

* lib/webrick/httpservlet/filehandler.rb (multipart_body): ditto.
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_3@63014 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 63014 - 03/28/2018 02:13 PM - usa (Usaku NAKAMURA)

merge revision(s) 60584,62954-62959,63008:

```
webrick: support Proc objects as body responses

* lib/webrick/httpresponse.rb (send_body): call send_body_proc
  (send_body_proc): new method
  (class ChunkedWrapper): new class

* test/webrick/test_httpresponse.rb (test_send_body_proc): new test
  (test_send_body_proc_chunked): ditto
  [Feature #855]

webrick: favor .write over << method

This will make the next change to use IO.copy_stream
easier-to-read.  When we can drop Ruby 2.4 support in a few
years, this will allow us to use writev(2) with multiple
arguments for headers and chunked responses.

* lib/webrick/cgi.rb (write): new wrapper method
  lib/webrick/httpresponse.rb: (send_header): use socket.write
  (send_body_io): ditto
  (send_body_string): ditto
  (send_body_proc): ditto
  (_write_data): ditto
  (ChunkedWrapper#write): ditto
  (_send_file): ditto
------------------------------------------------------------------------
r62954 | normal | 2018-03-28 17:05:52 +0900 (□, 28 3 2018) | 14 lines

webrick/httpresponse: IO.copy_stream for regular files

Remove the redundant _send_file method since its functionality
is unnecessary with IO.copy_stream.  IO.copy_stream also allows
the use of sendfile under some OSes to speed up copies to
non-TLS sockets.

Testing with "curl >/dev/null" and "ruby -run -e httpd" to
read a 1G file over Linux loopback reveals a reduction from
around ~0.770 to ~0.490 seconds on the client side.

* lib/webrick/httpresponse.rb (send_body_io): use IO.copy_stream
  (_send_file): remove
  [Feature #14237]
------------------------------------------------------------------------
r62955 | normal | 2018-03-28 17:05:57 +0900 (□, 28 3 2018) | 10 lines

webrick: use IO.copy_stream for single range response

This is also compatible with range responses generated
by Rack::File (tested with rack 2.0.3).

* lib/webrick/httpresponse.rb (send_body_io): use Content-Range
* lib/webrick/httpservlet/filehandler.rb (make_partial_content):
  use File object for the single range case
* test/webrick/test_filehandler.rb (get_res_body): use send_body
  to test result
```

```
------------------------------------------------------------------------
r62956 | normal | 2018-03-28 17:06:02 +0900 (□, 28 3 2018) | 7 lines

test/webrick/test_filehandler.rb: stricter multipart range test

We need to ensure we generate compatibile output in
the face of future changes

* test/webrick/test_filehandler.rb (test_make_partial_content):
  check response body
------------------------------------------------------------------------
r62957 | normal | 2018-03-28 17:06:08 +0900 (□, 28 3 2018) | 8 lines

webrick: quiet warning for multi-part ranges

Content-Length is ignored by WEBrick::HTTPResponse even if we
calculate it, so instead we chunk responses to HTTP/1.1 clients
and terminate HTTP/1.0 connections.

* lib/webrick/httpservlet/filehandler.rb (make_partial_content):
  quiet warning
------------------------------------------------------------------------
r62958 | normal | 2018-03-28 17:06:13 +0900 (□, 28 3 2018) | 7 lines

webrick/httpresponse: make ChunkedWrapper copy_stream-compatible

The .write method needs to return the number of bytes written
to avoid confusing IO.copy_stream.

* lib/webrick/httpresponse.rb (ChunkedWrapper#write): return bytes written
  (ChunkedWrapper#<<): return self
------------------------------------------------------------------------
r62959 | normal | 2018-03-28 17:06:18 +0900 (□, 28 3 2018) | 9 lines

webrick: use IO.copy_stream for multipart response

Use the new Proc response body feature to generate a multipart
range response dynamically.  We use a flat array to minimize
object overhead as much as possible; as many ranges may fit
into an HTTP request header.

* lib/webrick/httpservlet/filehandler.rb (multipart_body): new method
  (make_partial_content): use multipart_body

get rid of test error/failure on Windows introduced at r62955

* lib/webrick/httpresponse.rb (send_body_io): use seek if NotImplementedError
  is raised in IO.copy_stream with offset.

* lib/webrick/httpservlet/filehandler.rb (multipart_body): ditto.
```

Revision 19cb3fa9 - 03/28/2018 02:44 PM - usa (Usaku NAKAMURA)

merge revision(s) 60584,62954-62959,63008:

```
webrick: support Proc objects as body responses

* lib/webrick/httpresponse.rb (send_body): call send_body_proc
  (send_body_proc): new method
  (class ChunkedWrapper): new class

* test/webrick/test_httpresponse.rb (test_send_body_proc): new test
  (test_send_body_proc_chunked): ditto
  [Feature #855]

webrick: favor .write over << method

This will make the next change to use IO.copy_stream
easier-to-read.  When we can drop Ruby 2.4 support in a few
years, this will allow us to use writev(2) with multiple
arguments for headers and chunked responses.

* lib/webrick/cgi.rb (write): new wrapper method
  lib/webrick/httpresponse.rb: (send_header): use socket.write
  (send_body_io): ditto
```

```
  (send_body_string): ditto
  (send_body_proc): ditto
  (_write_data): ditto
  (ChunkedWrapper#write): ditto
  (_send_file): ditto
------------------------------------------------------------------------
r62954 | normal | 2018-03-28 17:05:52 +0900 (□, 28 3 2018) | 14 lines

webrick/httpresponse: IO.copy_stream for regular files

Remove the redundant _send_file method since its functionality
is unnecessary with IO.copy_stream.  IO.copy_stream also allows
the use of sendfile under some OSes to speed up copies to
non-TLS sockets.

Testing with "curl >/dev/null" and "ruby -run -e httpd" to
read a 1G file over Linux loopback reveals a reduction from
around ~0.770 to ~0.490 seconds on the client side.

* lib/webrick/httpresponse.rb (send_body_io): use IO.copy_stream
  (_send_file): remove
  [Feature #14237]
------------------------------------------------------------------------
r62955 | normal | 2018-03-28 17:05:57 +0900 (□, 28 3 2018) | 10 lines

webrick: use IO.copy_stream for single range response

This is also compatible with range responses generated
by Rack::File (tested with rack 2.0.3).

* lib/webrick/httpresponse.rb (send_body_io): use Content-Range
* lib/webrick/httpservlet/filehandler.rb (make_partial_content):
  use File object for the single range case
* test/webrick/test_filehandler.rb (get_res_body): use send_body
  to test result
------------------------------------------------------------------------
r62956 | normal | 2018-03-28 17:06:02 +0900 (□, 28 3 2018) | 7 lines

test/webrick/test_filehandler.rb: stricter multipart range test

We need to ensure we generate compatibile output in
the face of future changes

* test/webrick/test_filehandler.rb (test_make_partial_content):
  check response body
------------------------------------------------------------------------
r62957 | normal | 2018-03-28 17:06:08 +0900 (□, 28 3 2018) | 8 lines

webrick: quiet warning for multi-part ranges

Content-Length is ignored by WEBrick::HTTPResponse even if we
calculate it, so instead we chunk responses to HTTP/1.1 clients
and terminate HTTP/1.0 connections.

* lib/webrick/httpservlet/filehandler.rb (make_partial_content):
  quiet warning
------------------------------------------------------------------------
r62958 | normal | 2018-03-28 17:06:13 +0900 (□, 28 3 2018) | 7 lines

webrick/httpresponse: make ChunkedWrapper copy_stream-compatible

The .write method needs to return the number of bytes written
to avoid confusing IO.copy_stream.

* lib/webrick/httpresponse.rb (ChunkedWrapper#write): return bytes written
  (ChunkedWrapper#<<): return self
------------------------------------------------------------------------
r62959 | normal | 2018-03-28 17:06:18 +0900 (□, 28 3 2018) | 9 lines

webrick: use IO.copy_stream for multipart response

Use the new Proc response body feature to generate a multipart
range response dynamically.  We use a flat array to minimize
object overhead as much as possible; as many ranges may fit
into an HTTP request header.
```

```
* lib/webrick/httpservlet/filehandler.rb (multipart_body): new method
  (make_partial_content): use multipart_body

get rid of test error/failure on Windows introduced at r62955

* lib/webrick/httpresponse.rb (send_body_io): use seek if NotImplementedError
  is raised in IO.copy_stream with offset.

* lib/webrick/httpservlet/filehandler.rb (multipart_body): ditto.
```

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_2_2@63020 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 02/03/2009 11:00 AM - shyouhei (Shyouhei Urabe)

- Assignee set to gotoyuzo (GOTOU Yuuzou)

=begin

=end

#2 - 03/19/2009 06:52 PM - candlerb (Brian Candler)

=begin
There is an additional problem in setup_header, which prevents use of send_body_proc to stream bodies to HTTP/1.0 clients where no Content-Length has been set.

```
   elsif @header['content-length'].nil?
     unless @body.is_a?(IO)  # <<<<< PROBLEM: Proc is not an IO!
       @header['content-length'] = @body ? @body.size : 0
     end
   end
```

I suggest something like this:

```
   elsif @header['content-length'].nil?
     if @body.nil?
       @header['content-length'] = 0
     elsif @body.respond_to?(:size)
       @header['content-length'] = @body.size
     else
       @header['connection'] = 'close'
     end
   end
```

=end

#3 - 03/19/2009 11:41 PM - candlerb (Brian Candler)

=begin
I have made individual patches for all these issues and posted them onto github. There are three separate branches:

http://github.com/candlerb/webrick/commits/master
http://github.com/candlerb/webrick/commits/ruby18
http://github.com/candlerb/webrick/commits/ruby186

Most of my testing has been on the ruby186 branch, although I have checked that the test suites still pass for ruby18 (with 1.8.7p72) and master (with 1.9.1p0)

The main difference between these branches is that the master branch uses 'bytesize' instead of 'size'

I hope this work will make it easier for people to test the changes, and/or for them to be committed into subversion.

=end

#4 - 09/08/2010 11:39 AM - nahi (Hiroshi Nakamura)

- Category set to lib

- Assignee changed from gotoyuzo (GOTOU Yuuzou) to nahi (Hiroshi Nakamura)

=begin

=end

#5 - 09/10/2010 07:47 PM - nahi (Hiroshi Nakamura)

=begin
Sorry for late response.
Applied the patch No. 2 for '100-continue' to ruby_1_9: http://redmine.ruby-lang.org/repositories/revision/ruby-19?rev=29218
Thank you!

I'll give a look to other changes later. For now, No.3 sounds good and No.1 and No.4 could be OK.
=end

#6 - 09/14/2010 04:47 PM - shyouhei (Shyouhei Urabe)

- Status changed from Open to Assigned

=begin

=end

#7 - 06/22/2011 01:42 AM - nahi (Hiroshi Nakamura)

- Project changed from Ruby 1.8 to Ruby trunk

- Category changed from lib to lib

#8 - 06/23/2011 01:14 PM - nahi (Hiroshi Nakamura)

- Target version set to 2.0.0

r32192 is a fix for No.1. User can set HTTPResponse#chunked = true but it might not be a choice (Some client does not support chunked encoding for example.)

No.3 and No.4 should be discussed in the future...

#9 - 02/13/2012 08:56 PM - mame (Yusuke Endoh)

NaHi, are you still willing to discuss about No.3 and 4?
If not, could you please close this?

--
Yusuke Endoh mame@tsg.ne.jp

#10 - 02/14/2012 05:29 PM - nahi (Hiroshi Nakamura)

- Priority changed from 3 to Normal

Sorry that I kept this open long time.

Yes, the feature request for streaming (No.3 and No.4) is reasonable. I'll re-evaluate the patch before 2.0.0

I keep the assignee to me and raise the priority.

#11 - 11/20/2012 09:12 PM - mame (Yusuke Endoh)

- Target version changed from 2.0.0 to 2.6

#12 - 10/19/2017 12:51 PM - mame (Yusuke Endoh)

- Assignee changed from nahi (Hiroshi Nakamura) to normalperson (Eric Wong)

Eric Wong,

Could you handle this very old ticket about webrick?

#13 - 10/20/2017 08:41 AM - normalperson (Eric Wong)

mame@ruby-lang.org wrote:

> Could you handle this very old ticket about webrick?

Sure; it looks like a lot are already done except 4. Will take

a closer look tomorrow or next week. Thanks for the ping.

#14 - 10/30/2017 11:56 PM - Anonymous

- Status changed from Assigned to Closed

Applied in changeset trunk|r60584.

---

webrick: support Proc objects as body responses

- lib/webrick/httpresponse.rb (send_body): call send_body_proc
  (send_body_proc): new method
  (class ChunkedWrapper): new class

- test/webrick/test_httpresponse.rb (test_send_body_proc): new test
  (test_send_body_proc_chunked): ditto
  [Feature #855]

Files

webrick-patches.rb	2.69 KB	12/11/2008	candlerb (Brian Candler)