Ruby master - Feature #5628

Module#basename

11/14/2011 11:17 AM - trans (Thomas Sawyer)

Status:	Feedback	
Priority:	Normal	
Assignee:	matz (Yukihiro Matsumoto)	
Target version:		

Description

Something I use fairly often:

```
some_module.name.split("::").last
```

It's useful for things like factory methods. It would be much nicer if we had:

```
class Module
  def basename
    name.split("::").last
  end
end
```

History

#1 - 11/14/2011 02:53 PM - wycats (Yehuda Katz)

Totally agreed. This pattern is fairly common. It would also be nice to have Module#modules or something, such that Foo::Bar would return [Foo, Bar].

Yehuda Katz
(ph) 718.877.1325

On Sun, Nov 13, 2011 at 6:17 PM, Thomas Sawyer transfire@gmail.com wrote:

> Issue #5628 has been reported by Thomas Sawyer.
>
> ---
>
> Feature #5628: Module#basename
> http://redmine.ruby-lang.org/issues/5628
>
> Author: Thomas Sawyer
> Status: Open
> Priority: Normal
> Assignee:
> Category:
> Target version: 1.9.4
>
> Something I use fairly often:
>
> some_module.name.split("::").last
>
> It's useful for things like factory methods. It would be much nicer if we had:
>
> class Module
> def basename
> name.split("::").last
> end
> end
>
> --
> http://redmine.ruby-lang.org

#2 - 11/14/2011 03:32 PM - agrimm (Andrew Grimm)

I'm not sure about the method name. What should happen with File::basename ? Should it call super when there's no arguments, and do its own thing

when called with an argument? I'd prefer Ruby to tell me that I forgot to pass in a string!

#3 - 11/14/2011 07:23 PM - Eregon (Benoit Daloze)

On 14 November 2011 07:32, Andrew Grimm andrew.j.grimm@gmail.com wrote:

> Issue #5628 has been updated by Andrew Grimm.
>
> I'm not sure about the method name. What should happen with File::basename ? Should it call super when there's no arguments, and do its own thing when called with an argument? I'd prefer Ruby to tell me that I forgot to pass in a string!

I also agree this would be useful.
But the name 'basename' is probably not the best.
I remember I used simple_name for this, but it's not appealing.

#4 - 11/14/2011 07:59 PM - regularfry (Alex Young)

On 14/11/11 10:19, Benoit Daloze wrote:

> On 14 November 2011 07:32, Andrew Grimm andrew.j.grimm@gmail.com wrote:
>
> > Issue #5628 has been updated by Andrew Grimm.
> >
> > I'm not sure about the method name. What should happen with File::basename ? Should it call super when there's no arguments, and do its own thing when called with an argument? I'd prefer Ruby to tell me that I forgot to pass in a string!
>
> I also agree this would be useful.
> But the name 'basename' is probably not the best.
> I remember I used simple_name for this, but it's not appealing.
>
> I like this - I use .split("::").last all over the place.

A few alternatives, since I agree it would be unfortunate to overload
.basename:

.inner_name
.last_name
.primary_name
.name(false)
.name(:full => false)

--
Alex

#5 - 11/14/2011 08:58 PM - trans (Thomas Sawyer)

@Andrew File::basename is a class method, where as Module#basename is an instance method, so there isn't really any "polymorphic confliction".

I think we'd also be hard pressed to find a better name. I've thought about it quite a bit. And nothing else seems to fit. Ironically #firstname makes the most semantic sense for English speakers. Since Ruby is Japanese in origin, then #lastname is a fun option to confound Americans ;)

I think #basename is best b/c there is a common correspondence between files and classes, eg. lib/foo/some_class will usually contain Foo::SomeClass. There is also Pathname#basename.

#6 - 11/14/2011 11:53 PM - Eregon (Benoit Daloze)

On 14 November 2011 12:58, Thomas Sawyer transfire@gmail.com wrote:

> Issue #5628 has been updated by Thomas Sawyer.
>
> @Andrew File::basename is a class method, where as Module#basename is an instance method, so there isn't really any "polymorphic confliction".

Are you so sure? ;-)
A class method is an instance method on the class, so yes
File::basename conflicts with Module#basename:

```
class Module
def basename
name.split("::").last
end
```

end
Enumerator::Generator.basename
=> "Generator"
File.basename
ArgumentError: wrong number of arguments (0 for 1..2)
from (irb):7:in `basename'
File.basename 'file.ext'
=> "file.ext"

So one calling 'basename' on a Module can expect either a String or an ArgumentError.
And changing File::basename to be Module#basename when no arguments are given does not seem a good design at all.

#7 - 11/15/2011 12:10 AM - trans (Thomas Sawyer)

You're right. File is also Module (subclass of Class). So, yes, another name is needed, or ::File.basename accepted as an exception.

In my defense, I always thought File class methods for file handling were bad mojo! It would be much better if Pathname were core and File class methods non-existent (IMHO).

#8 - 11/15/2011 05:29 AM - wycats (Yehuda Katz)

I would probably be ok with mod.modules.last.name to be honest.

Yehuda Katz
(ph) 718.877.1325

On Mon, Nov 14, 2011 at 7:12 AM, Thomas Sawyer transfire@gmail.com wrote:

> Issue #5628 has been updated by Thomas Sawyer.
>
> You're right. File is also Module (subclass of Class). So, yes, another name is needed, or ::File.basename accepted as an exception.
>
> In my defense, I always thought File class methods for file handling were bad mojo! It would be much better if Pathname were core and File class methods non-existent (IMHO).
>
> ---
>
> Feature #5628: Module#basename
> http://redmine.ruby-lang.org/issues/5628
>
> Author: Thomas Sawyer
> Status: Open
> Priority: Normal
> Assignee:
> Category:
> Target version: 1.9.4
>
> Something I use fairly often:
>
> some_module.name.split("::").last
>
> It's useful for things like factory methods. It would be much nicer if we had:
>
> class Module
> def basename
> name.split("::").last
> end
> end
>
> --
> http://redmine.ruby-lang.org

#9 - 11/15/2011 05:59 AM - trans (Thomas Sawyer)

> I would probably be ok with mod.modules.last.name to be honest.

Then might as well add mod.lastname too. If the implementation is anything like the pure-Ruby one, it is more efficient to have a dedicated method. And I much prefer the shorter syntax. If not #lastname then maybe #modname.

#10 - 03/28/2012 12:17 AM - mame (Yusuke Endoh)

- Assignee set to matz (Yukihiro Matsumoto)

#11 - 03/31/2012 11:09 AM - mame (Yusuke Endoh)

- Status changed from Open to Assigned

#12 - 03/31/2012 11:15 AM - mame (Yusuke Endoh)

- Target version changed from 1.9.4 to 2.0.0

Hello,

2011/11/14 Thomas Sawyer transfire@gmail.com:

> Something I use fairly often:
>
> some_module.name.split("::").last

Personally, I've never used such a hack. When do you use?

--
Yusuke Endoh mame@tsg.ne.jp

#13 - 03/31/2012 11:29 AM - trans (Thomas Sawyer)

Common case (for me) is when user selects a "plug-in" via a command line option. For example a pseudo test framework:

```
$ mytest --reporter progress
```

Then in code:

```
module MyTestFramework
  def self.reporters
    @reporters ||= {}
  end

  module Reporters
    class Base
      def self.inherited(subclass)
        MyTestFramework.reporters[subclass.basename.downcase] = subclass
      end
    end

    class Progress < Base
      ...
    end
  end
end
```

Then reporters are very easy to lookup with command line option.

```
MyTestFramework.reporters[reporter]   # i.e. reporter = 'progress'
```

That's just one example, but I have found the basic pattern to be useful in many such "pluggable" designs.

#14 - 03/31/2012 06:24 PM - regularfry (Alex Young)

=begin
I'm doing something remarkably similar to this for mapping command-line subcommand selection into a module's namespace. It's very handy.
=end

#15 - 04/01/2012 01:29 AM - matz (Yukihiro Matsumoto)

- Status changed from Assigned to Feedback

I am not sure if it's worth adding to the core. It is so easy to add by third-party lib.
Besides that, even though the term "basename" is understandable from analogy to UNIX command name, but the term includes "base" might cause

confusion, since "base" in object class has different meaning (especially C++ and other languages).

Matz.

#16 - 04/01/2012 03:36 AM - trans (Thomas Sawyer)

You are right about name, it would have to be something else besides #basename.

Yehuda Katz said he would probably be ok with mod.modules.last.name, to which I commented, we may as well add mod.lastname.

#17 - 10/25/2012 05:49 PM - yhara (Yutaka HARA)

- Target version changed from 2.0.0 to 2.6

#18 - 12/25/2017 06:15 PM - naruse (Yui NARUSE)

- Target version deleted (2.6)