## CommonRuby - Feature #8571

**Refinement inheritance by Module#include**

06/27/2013 06:19 PM - shugo (Shugo Maeda)

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>shugo (Shugo Maeda)</td>
</tr>
<tr>
<td>Target version:</td>
<td></td>
</tr>
</tbody>
</table>

### Description

Currently, refinements are not inherited by `Module#include`. How about to make using to activate refinements in the ancestors of the specified module?

For example,

```ruby
module A
  refine X do ... end
  refine Y do ... end
end

module B
  refine Z do ... end
end

module C
  include A
  include B
end

using C
# Refinements in A and B are activated here.
```

In this code,

```ruby
using C
```

is equivalent to

```ruby
C.ancestors.reverse_each { |m| using m }
```

This feature enables you to distribute refinements into different modules, and then package the refinements into one module.

Please note that `Module#include` does't activate refinements in the argument module.

For example,

```ruby
module C
  include A
  include B

# Refinements in A and B are not activated here.
end
```

I've implemented this feature, and have attached the patch.

### History

**#1 - 07/01/2013 07:10 AM - matz (Yukihiro Matsumoto)**

- Assignee changed from matz (Yukihiro Matsumoto) to shugo (Shugo Maeda)

Agreed. It also conforms the original design.

Matz.
matz (Yukihiro Matsumoto) wrote:

Agreed. It also conforms the original design.

I've committed in r41719. Thanks.