Refinement inheritance by Module#include

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
```

```ruby
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 doesn'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.