Ruby 范围（Range） Ruby 文件的输入与输出

Ruby 迭代器

简单来说：迭代(iterate)指的是重复做相同的事，所以迭代器(iterator)就是用来重复多次相同的事。

迭代器是*集合*支持的方法。存储一组数据成员的对象称为集合。在 Ruby 中，数组(Array)和哈希(Hash)可以称之为集合。

迭代器返回集合的所有元素，一个接着一个。在这里我们将讨论两种迭代器，*each* 和 *collect*。

Ruby *each* 迭代器

each 迭代器返回数组或哈希的所有元素。

语法

```
collection.each do |variable|
code
end
```

为*集合*中的每个元素执行 *code*。在这里，集合可以是数组或哈希。

实例

```
#!/usr/bin/ruby
ary = [1,2,3,4,5]
ary.each do |i|
puts i
end
```

尝试一下 »

以上实例运行输出结果为：

```
1
2
3
4
5
```

each 迭代器总是与一个块关联。它向块返回数组的每个值，一个接着一个。值被存储在变量 **i** 中，然后显示在屏幕上。

Ruby *collect* 迭代器

collect 迭代器返回集合的所有元素。

语法

```
collection = collection.collect
```

collect 方法不需要总是与一个块关联。*collect* 方法返回整个集合，不管它是数组或者是哈希。

实例

实例

```
#!/usr/bin/ruby
a = [1,2,3,4,5]
b = Array.new
b = a.collect{ |x|x }
puts b
```

尝试一下 »

以上实例运行输出结果为：

```
1
2
3
4
5
```

注意：*collect* 方法不是数组间进行复制的正确方式。这里有另一个称为 *clone* 的方法，用于复制一个数组到另一个数组。

当您想要对每个值进行一些操作以便获得新的数组时，您通常使用 collect 方法。例如，下面的代码会生成一个数组，其值是 a 中每个值的 10 倍。

实例

```
#!/usr/bin/ruby
a = [1,2,3,4,5]
b = a.collect{|x| 10*x}
puts b
```

尝试一下 »

以上实例运行输出结果为：

```
10
20
30
40
50
```

← Ruby 范围（Range）

Ruby 文件的输入与输出 →

1 篇笔记

写笔记

Java需要把Map转化成List类型的容器才能使用迭代器，但Ruby有直接针对Map的迭代器：

```
sum = 0
cutcome = {"block1" => 1000, "book2" => 1000, "book3" => 4000}
cutcome.each{|item, price| sum += price}
print "sum = " + sum.to_s
```

甚至还可以这样：

```
sum = 0
cutcome = {"block1" => 1000, "book2" => 1000, "book3" => 4000}
cutcome.each{|pair| sum += pair[1]}
print "sum = " + sum.to_s
```

hjc132 1年前 (2017-12-09)