← Ruby 多线程 Ruby JSON →

Ruby 数据类型

本章节我们将为大家介绍 Ruby 的基本数据类型。

Ruby支持的数据类型包括基本的Number、String、Ranges、Symbols，以及true、false和nil这几个特殊值，同时还有两种重要的数据结构——Array和Hash。

数值类型(Number)

1、整型(Integer)

整型分两种，如果在31位以内（四字节），那为Fixnum实例。如果超过，即为Bignum实例。

整数范围从 -2^{30} 到 $2^{30}-1$，在这个范围内的整数是类 *Fixnum* 的对象， 当整数值大于或等于2的30次方时（ -2^{62} 到 $2^{62}-1$），会自动转化为 Bignum 类型。

您可以在整数前使用一个可选的前导符号，一个可选的基础指标（0 对应 octal，0x 对应 hex，0b 对应 binary），后跟一串数字。下划线字符在数字字符串中被忽略。

您可以获取一个 ASCII 字符或一个用问号标记的转义序列的整数值。

实例

```
123 # Fixnum 十进制
1_234 # Fixnum 带有下划线的十进制
-500 # 负的 Fixnum
0377 # 八进制
0xff # 十六进制
0b1011 # 二进制
"a".ord # "a" 的字符编码
?\n # 换行符（0x0a）的编码
12345678901234567890 # 大数
#整型 Integer 以下是一些整型字面量
#字面量（literal）：代码中能见到的值，数值，bool值，字符串等都叫字面量
#如以下的0,1_000_000,0xa等
a1=0
#带千分符的整型
a2=1_000_000
#其它进制的表示
a3=0xa
puts a1,a2
puts a3
#puts print 都是向控制台打印字符，其中puts带回车换行符
=begin
这是注释，称作：嵌入式文档注释
类似C#中的/**/
=end
```

浮点型

Ruby 支持浮点数。它们是带有小数的数字。浮点数是类 *Float* 的对象，且可以是下列中任意一个。

实例

实例

```
123.4 # 浮点值
1.0e6 # 科学记数法
4E20 # 不是必需的
4e+20 # 指数前的符号
#浮点型
f1=0.0
f2=2.1
f3=1000000.1
puts f3
```

算术操作

加减乘除操作符：+-*/；指数操作符为**

指数不必是整数，例如

实例

```
#指数算术
puts 2**(1/4)#1与4的商为0，然后2的0次方为1
puts 16**(1/4.0)#1与4.0的商为0.25（四分之一），然后开四次方根
```

字符串类型

Ruby 字符串简单地说是一个 8 位字节序列，它们是类 String 的对象。

双引号标记的字符串允许替换和使用反斜线符号，单引号标记的字符串不允许替换，且只允许使用 \\ 和 \' 两个反斜线符号。

实例

```
#!/usr/bin/ruby -w
puts 'escape using "\\"';
puts 'That\'s right';
```

尝试一下 »

这将产生以下结果：

```
escape using "\"
That's right
```

您可以使用序列 **#{ expr }** 替换任意 Ruby 表达式的值为一个字符串。在这里，expr 可以是任意的 Ruby 表达式。

实例

```
#!/usr/bin/ruby -w
puts "相乘 : #{24*60*60}";
```

这将产生以下结果：

```
相乘 : 86400
```

实例

```
#!/usr/bin/ruby -w
name="Ruby"
puts name
puts "#{name+",ok"}"
```

输出结果为：

```
Ruby
Ruby,ok
```

反斜线符号

下表列出了 Ruby 支持的反斜线符号：

符号	表示的字符
\n	换行符 (0x0a)
\r	回车符 (0x0d)
\f	换页符 (0x0c)
\b	退格键 (0x08)
\a	报警符 Bell (0x07)
\e	转义符 (0x1b)
\s	空格符 (0x20)
\nnn	八进制表示法 (n 是 0-7)
\xnn	十六进制表示法 (n 是 0-9、a-f 或 A-F)
\cx, \C-x	Control-x
\M-x	Meta-x (c | 0x80)
\M-\C-x	Meta-Control-x
\x	字符 x

如需了解更多有关 Ruby 字符串的细节，请查看 Ruby 字符串（String）。

数组

数组字面量通过[]中以逗号分隔定义，且支持range定义。

- （1）数组通过[]索引访问
- （2）通过赋值操作插入、删除、替换元素
- （3）通过+，-号进行合并和删除元素，且集合做为新集合出现
- （4）通过<<号向原数据追加元素
- （5）通过*号重复数组元素
- （6）通过|和&符号做并集和交集操作（注意顺序）

实例

```
#!/usr/bin/ruby
ary = [ "fred", 10, 3.14, "This is a string", "last element", ]
ary.each do |i|
puts i
end
```

尝试一下 »

这将产生以下结果：

```
fred
10
3.14
This is a string
last element
```

如需了解更多有关 Ruby 数组的细节，请查看 Ruby 数组（Array）。

哈希类型

Ruby 哈希是在大括号内放置一系列键/值对，键和值之间使用逗号和序列 => 分隔。尾部的逗号会被忽略。

实例

实例

```
#!/usr/bin/ruby
hsh = colors = { "red" => 0xf00, "green" => 0x0f0, "blue" => 0x00f }
hsh.each do |key, value|
print key, " is ", value, "\n"
end
```

尝试一下 »

这将产生以下结果：

```
red is 3840
green is 240
blue is 15
```

如需了解更多有关 Ruby 哈希的细节，请查看 Ruby 哈希（Hash）。

范围类型

一个范围表示一个区间。

范围是通过设置一个开始值和一个结束值来表示。范围可使用 s..e 和 s...e 来构造，或者通过 Range.new 来构造。

使用 .. 构造的范围从开始值运行到结束值（包含结束值）。使用 ... 构造的范围从开始值运行到结束值（不包含结束值）。当作为一个迭代器使用时，范围会返回序列中的每个值。

范围 (1..5) 意味着它包含值 1, 2, 3, 4, 5，范围 (1...5) 意味着它包含值 1, 2, 3, 4 。

实例

```
#!/usr/bin/ruby
(10..15).each do |n|
print n, ' '
end
```

尝试一下 »

这将产生以下结果：

```
10 11 12 13 14 15
```

如需了解更多有关 Ruby 范围的细节，请查看 Ruby 范围（Range）。

← Ruby 多线程 Ruby JSON →