Standard Types

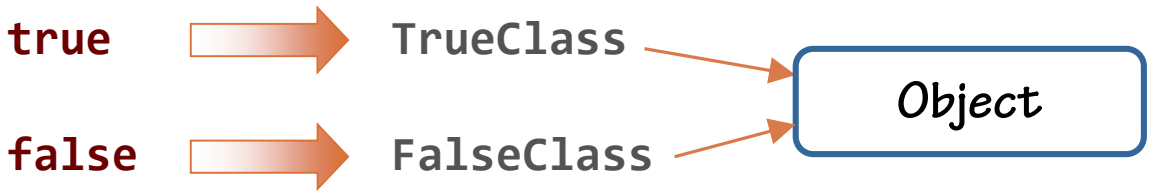
Ruby Fundamentals

pluralsight
hardcore developer training

Overview

- Booleans
- Integer and floating point numbers
- Regular expressions
- Strings
- Symbols
- Arrays and hashes
- Ranges
- Parallel assignment

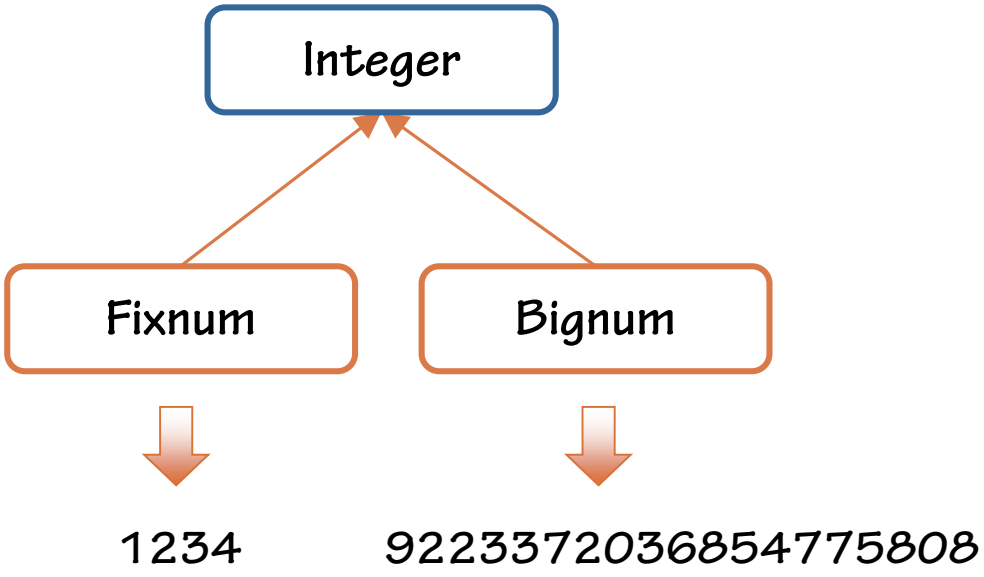
Booleans




```
true.to_s      # "true"

false.nil?     # false
```

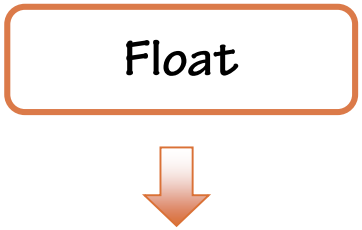
Integer Numbers




```
0xAF     # decimal 175
010      # decimal 8
0o10     # decimal 8
0d175    # decimal 175
0b1010   # decimal 10
```

```
100_999
0b1010_1111_0110
```

Floating Point Numbers



1.234
1.0e3

Strings

```
'Serenity'
```

Serenity

```
'\'Serenity\''
```

‘Serenity’

```
'Backslash: \\'
```

Backslash: \

```
%q('Serenity' transport)
%q['Serenity' transport]
%q{'Serenity' transport}
%q<'Serenity' transport>
```

‘Serenity’ transport

```
%q*'Serenity' transport*
%q/'Serenity' transport/
```

Strings

`\n`		newline
`\t`		tab
`\"`		double quote
`\100` `\x40`		@
`\u20ac`		€
`\u{20ac a3 e3f}`		€£฿

Strings

```
lander_count = 10
probe_count = 20
puts "Lander count: #{lander_count}"
puts "Total units: #{lander_count + probe_count}"


%Q(“Serenity” transport)
%*“Serenity” transport*
```

Strings

```
message = <<EOS
  There's no place I can be
  since I found serenity
EOS
```

```
message = <<-EOS
          There's no place I can be
          since I found serenity
          EOS
```

Regular Expressions

- Ruby Regexp class docs: http://ruby-doc.org/core-2.0/Regexp.html
- Regex test tool: http://rubular.com/

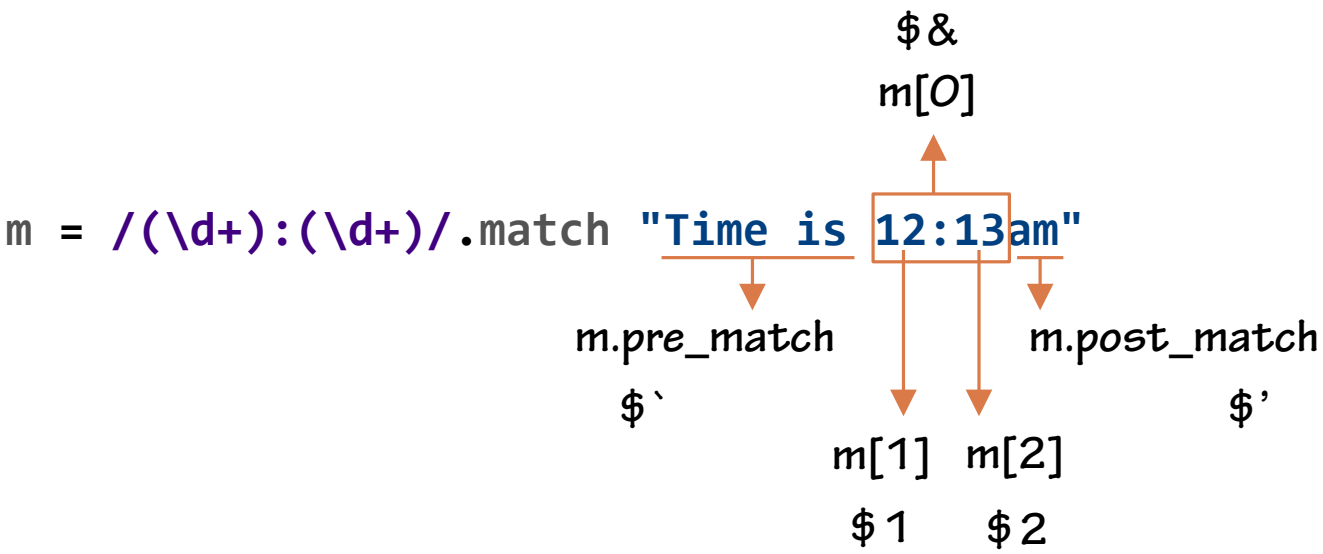
Regular Expressions

Regexp

```
/(\d+:\d+)(am|pm)/
```

```
%r(\d+/\d+)
```

Regular Expressions



Symbols

```
attr_accessor :destination
private :batten_hatches
```

```
Symbol
```

```
:"abc"
:"3"
```

```
direction = "west"
:"turn_#{direction}" # :turn_west
```

```
traverse_tree(:depth_first)
```

Arrays

```
Array

[1, 2, 3]

[1, "Z", Object.new]
```

Enumerable

Hashes

```
Hash

h = {}

h = {"min" => 0, "max" => 100}

h = {min: 0, max: 100}
```

Ranges

Range

```
1..5   # [1, 5]

1...5  # [1, 5)


puts case sample_reading
     when 0..100 then "below normal"
     when 101..150 then "normal"
     else "excessive"
     end
```

Parallel Assignment and the Splat Operator

```ruby
a, b = 1, 2           # a == 1, b == 2


a = 1, 2, 3, 4        # a == [1, 2, 3, 4]


def get_values
  [1, 2, 3, 4]
end

a, b = get_values     # a == 1, b == 2

first, dummy, dummy, last = get_values

first, _, _, last = get_values
```

Parallel Assignment and the Splat Operator

```
a, *b = get_values      # a == 1, b == [2, 3, 4]

a, *b, c = get_values   # a == 1, b == [2, 3], c == 4

a, b, c = *1..3

first, _, _, _, last = 1, 2, *[3, 4, 5]  # last == 5

first, *, last = 1, 2, *[3, 4, 5] # last == 5
```

Summary

- Booleans, numbers, strings
- Regular expressions
- Symbols
- Arrays, hashes, ranges
- Enumerable
- Parallel assignment and the splat operator