Sinatra routes

```
# returning a simple string
get '/hello' do
  "Why hello there, friendly Web traveler!"
end

# returning a processed erb file (containing HTML)
get '/hello' do
  erb :hello
end
```

Ruby language

Variables

```
# local variable
magic_number = 42
# instance variable
@instance_variable = "My first instance variable!"
# global variable
$global_variable = "Be careful with global variables!"
```

Data types

Strings

```
# syntax
"This is a string."
"123456789abcde"
"!@#$%^&*()"
""
'A string can also be in single quotes'
# string methods
# string interpolation
dog_speak = "woof!"
puts "Dogs say #{dog_speak}"
# upcase
"quiet library voice".upcase # => "QUIET LIBRARY VOICE"
# reverse
"bolton".reverse             # => "notlob"
# split
"this was a string".split    # => ["this","was","a","string"]
```

Symbols

```
:a_symbol
:another_symbol
:name
```

Integers

```
42
10000
-256
```

Floats

```
1.1
3.141592653589793
99.999
-128.6
# methods for arithmetic
5 + 5       # addition
10 - 5      # subtraction
20 * 20     # multiplication
100 / 20    # division
104 % 5     # modulus (remainder)
# methods for comparison
10 == 10    # equality
5 != 6      # inequality
30 > 20     # greater than
10 < 15     # less than
5 >= 5      # greater than or equal
99 <= 100   # less than or equal
```

Booleans

```
true
false
```

Conditionals

```
number = 50
if number > 50
  puts "The number is greater than 50"
else
  puts "The number is 50 or less."
end
```

Methods

```
# defining methods
def create_greeting(name)
  return "Welcome to my website, #{name}!"
end
# calling methods
create_greeting("Sam")
```

Arrays

```
# an array of strings
["a","b","c","d","e","f","g","h"]
# an array of hashes
[{:language => "Ruby", usage: => "backend"},
{:language => "JavaScript", :usage => "frontend"},
{:language => "Swift", :usage => "mobile"}]

# array methods
# accessing
letters = ["a","b","c","d","e","f","g","h"]
letters[0] # => "a"
letters[7] # => "h"
# updating
# if we want to change "a" and "b":
letters[0] = "apples"
letters[1] = "bananas"
# inserting
letters << "elephant"    # adds "elephant" to the end of the array
letters.push("elephant") # adds "elephant" to the end of the array
letters.unshift("zebra") # adds "zebra" to the start of the array
# removing
letters.pop # removes and returns the last element
letters.shift # removes and returns the first element
# join
["this","was","an","array"].join # => "thiswasanarray"
["this","was","an","array"].join("-") # => "this-was-an-array"
# sample
["luck", "of", "the", "draw"].sample # => a randomly chosen element
```

Block syntax

```
# curly braces for one line
greetings.each { |word| puts "You can say hi by saying #{word}!" }

# do/end for multiple lines
greetings.each do |word|
  puts "Heres a new word: #{word}"
  puts "You can say hi by saying #{word}!"
end
```

Hashes

```
# symbols as keys, strings or integers as values
{:name => "Sam", :age => 28, :sex => "male"}
# accessing
person[:name]  # returns "Sam"
# updating
person[:name] = "Samuel" # the value in :name is now "Samuel"
# inserting
person[:gender] = "cis male"
# removing
person.delete(:sex) # removes the key :sex and returns its value
```

Loops

```
# times loop
10.times do
  puts "Hello!"
end
# times loops with counter
10.times do |counter|
  puts "Here we go. Let's count: #{counter}"
end
# each iterator
greetings = ["hello", "kia ora", "aloha", "talofa", "malo"]
greetings.each do |word|
  puts "You can say hi by saying #{word}!"
end
# map method
greetings = ["hello", "kia ora", "aloha", "talofa", "malo"]
shouty_greetings = greetings.map do |word|
  word.upcase
end
# returns ["HELLO", "KIA ORA", "ALOHA", "TALOFA", "MALO"]
```

devbootcamp