Ruby Starter Pack Cheatsheet

Ruby, the Magical Mistress

What do you think of Ruby so far? Fun right? Below you will find a grab bag of Ruby tips, tricks, and important methods.

Strings

Strings can be surrounded by either single quotes:

```
‘This is a string’
```

Or double quotes:

```
“This is also a string”
```

To use a contraction inside a string, always use double quotes:

```
“Isn’t Ruby fun?!”
```

To use a variable inside a string, always use double quotes and put the variable inside #{}:

```
myCat = ‘Reggie’
“My super cool cat’s name is #{myCat}”
```

To add strings and variables together, use the plus (+) symbol:

```
myDog = ‘Abbie’
puts “My dog’s name is ” + myDog + ‘.’
```

made with love by: skillcrush

Variables

The rules of Ruby variables:

- Variable names can only contain letters, numbers, and underscores
- Variables cannot start with a number
- Variables should not be all-caps; those are special variables called constants
- Variables are case sensitive

Good variable names:

✓ myVariable

✓ _another_variable

Bad variable names:

✗ 1_my_variable

✗ MYVARIABLE

To create a variable:

```
myCat = ‘Roxy’
```

```
_my_age = 32
```

Methods

capitalize

Replaces the first letter of each word in a string with uppercase, and the rest of the letters of each word with lowercase.

```
city = ‘BrOoKlYn’
```

```
city.capitalize # city is equal to ‘Brooklyn’
```

```
state = ‘NEW yoRK’
```

```
state.capitalize # state is equal to ‘New York’
```

chomp

Removes the new line from the end of a string, or a specified portion of a string, and returns a new string.

“\n” adds a new line to the end of a string and must be inside double quotes:

```
“hello there\n”
```

chomp removes the new line:

```
“hello there\n”.chomp # returns “hello there”
```

chomp can also remove portions of a string and return what’s left:

```
‘Hello World!’.chomp(‘ World!’) # returns “Hello”
```

chop

Removes the last character from a string and returns a new string. Also removes the new line from the end of a string.

“\n” adds a new line to the end of a string and must be inside double quotes:

```
“hello there\n”
```

chop removes the new line:

```
“hello there\n”.chop # returns “hello there”
```

chop also removes the last character from a string:

```
‘hello there’.chop # returns ‘hello ther’
```

downcase

Replaces all uppercase letters in a string with lowercase.

```
city = ‘BrOoKlYn’
```

```
city.downcase # returns ‘brooklyn’
```

empty?

Returns true or false depending on whether a string is empty or has contents.

```
‘Skillcrush’.empty? # returns false
```

```
“”.empty? # returns true
```

eql?

Returns true or false depending on how two strings compare. The strings are considered equal if their content and length is the same.

```
var1 = ‘apple’
var2 = ‘orange’
var1.eql?(var2) # returns false
var3 = ‘banana’
var4 = ‘banana’
var3.eql?(var4) # returns true
```

gets

Asks the user for input.

```
puts ‘What is your name?’
name = gets
puts “Hello, #{name}!”
```

insert

Inserts the given character(s) into a string at the given index. Ruby is zero-indexed!

```
city = ‘Brooklyn’
city.insert(0, ‘X’) # city is equal to ‘XBrooklyn’ - X is inserted at
the 0 position of the string ‘Brooklyn’, which is the beginning
state = ‘New York’
state.insert(8, ‘XYZ’) # state is equal to ‘BrooklynXYZ’ - X is inserted
at the 8th position of the string ‘Brooklyn’, which is the end
```

Ruby Starter Pack Cheatsheet

length

Returns the length of the string

```
‘hello’.length # returns 5
name = ‘Skillcrush’
name.length # returns 10
```

print

Prints to the screen with NO new line included at the end.

```
print ‘Hello, World!’
```

puts

Prints to the screen with a new line included at the end.

```
puts ‘Hello, World!’ # the same as print “Hello, World!\n”
```

strip

Removes extra white space (spaces) from the beginning and end of a string

```
text = ‘     Hello, World!     ’
text.strip # text is equal to ‘Hello, World!’
```

swapcase

Returns a string with the uppercase letters converted to lowercase, and lowercase letters converted to uppercase.

```
name = ‘SkIlLcRuSh’
name.swapcase # returns ‘sKiLlCrUsH’
```

upcase

Replaces all lowercase letters in a string with uppercase.

```
city = ‘BrOoKlYn’
```

```
city.upcase # returns ‘BROOKLYN’
```

More Information About Ruby

For more information about Ruby, check out the Ruby Documentation.