Ruby monstas Ruby cheat sheet

Data types and how to use them

Name	Description	Structure	Examples
Integer literal	A whole number		3 -552
Floating point literal	A decimal number		42.23 -0.133742
Addition		*a + b*	5.2 + 6.34 # => 11.54
Subtraction		*a - b*	2.59 - 4.89 # => -2.3
Multiplication		*a * b*	5 * 3.7 # => 18.5
Division	Mind the difference between integer and float divisions!	*a / b*	6 / 4 # => 1 (integer division) 6 / 4.0 # => 1.5 (float division)
Modulo	Returns the remainder of a division	*a % b*	13 % 6 # => 1
String literal	A string of characters, text		"this is a string"
String interpolation	Text with Ruby code embedded in it		"another string with an #{interpolation}"

Arrays

Name	Description	Structure	Examples
Array literal	Creates a new array	*[item1, item2, ...]*	`my_array = [1, 2, 3]`
Length	Returns the length of the array (the number of items it contains)	*array.length*	`my_array.length         # => 3`
Index operator	Lets you access the item at a given position within an array	*array[index]*	`my_array[1]             # => 2`
delete_at	Deletes the item at a given index and returns it	*array.delete_at(index)*	`my_array.delete_at(1)   # => 2`

each	Lets you iterate over all elements in an array	*array.each do |item|*
		
		end
			`my_array.each do |item|`
			`   puts item`
			`end`

first	Returns the very first item of the array	*array.first*	`my_array.first          # => 1`
last	Returns the very last item of the array	*array.last*	`my_array.last           # => 3`
include?	Returns a boolean, whether the array contains a certain element or not	*array.include?(item)*	`my_array.include?(4)    # => false`
pop	Removes the last item of the array and returns it	*array.pop*	`my_array.pop            # => 3`
push or <<	Adds an item to the end of the array	*array.push(item)* *array << item*	`my_array.push(4)` `my_array << 4`
reverse	Returns a copy of the array with the elements in reverse order	*array.reverse*	`my_array.reverse        # [3, 2, 1]`
sort	Returns a sorted copy of the array	*array.sort*	`[5, 2, 4].sort          # [2, 4, 5]`
uniq	Returns a copy of the array with duplicates removed	*array.uniq*	`[1, 1, 2, 2].uniq       # [1, 2]`

Hashes

Name	Description	Structure	Examples

Hash literal	Create a hash	*{ "key" => "value" }*	```hash = {}```
			```hash = { "key" => "value", "other_key"```
			```=> 42 }```

Hash access	Access a value by its key	*hash[key]*	```hash["key"] # => "value"```
Key deletion	Delete a key-value pair by its key	*hash.delete(key)*	```hash.delete("key") # => "value"```
Empty hash	Remove all pairs from the hash	*hash.clear*	```hash.clear```

Iterate over hash	Iterate over all the pairs in the hash	*hash.each do |key, value|*
		end
			```hash.each do |key, value|```
			``` puts "#{key} has value: #{value}"```
			```end```

Iterate over pairs	Iterate over all the pairs in the hash	*hash.each_pair do |key, value|*
		end
			```hash.each_pair do |key, value|```
			``` puts "#{key} has value: #{value}"```
			```end```

Get key	Get a value for a key, with default value if the key does not exist.	*hash.fetch(key, default)*	```hash.fetch("key") # => "value"```
			```hash.fetch("xy", "default")```
			```=> "default"```

Key existence	Ask the hash if it has a certain key	*hash.has_key?(key)*	```hash.has_key?("key") # => true```
Value existence	Ask the hash if it has a certain value	*hash.has_value?(value)*	```hash.has_value?("xy") # => false```
All keys	Get all the keys stored in the hash	*hash.keys*	```hash.keys # => ["key", "other_key"]```
All values	Get all the values stored in the hash	*hash.values*	```hash.values # => ["value", 42]```
Merge	Merge two hashes	*hash.merge(other_hash)*	```hash.merge({"a_key" => 23})``` ```# => { "key" => "value", "a_key" => 23 }```