Programming for Everybody

4. Arrays & Hashes

le wagon

Arrays

A **collection** of Ruby data (or list of values) called **elements,** separated by commas, which may be stored in a variable

An Array is defined with square brackets [ ]

Arrays may contain: **numbers** (in any order, repeated or not), **strings**, **booleans, symbols** and even… **other arrays**! :) (arrays of arrays are called *multidimensional arrays*)

```
an_array = [“Bob”, “Joe”, “Zack”]

another_array = [1, 7, 16]
```

Arrays (cont.)

Creating new arrays

```
my_array = [ ]
my_other_array = [1, 2, 3, 7, 10]
# Index           0  1  2  3  4
```

Each element in the array is located at what is called an **index**

The first element is at index **0**, the next is at index **1**, the following is at index **2**, and so on.

Arrays (cont.)

access / **read** an element from an Array
- end: `array.last`
- beginning: `array.first`
- chosen index: `array[index]`

add an element to an Array
- end: `array.push(new element)`
- beginning: `array.unshift(new element)`
- chosen index: `array.insert(index, new element)`

Arrays (cont.)

delete an element from an Array

- end: `array.pop`
- beginning: `array.shift`
- by index: `array.delete_at(index)`
- by value: `array.delete(value)`

Arrays (cont.)

update an element from an Array

```
my_array = [“Mariana”, “Zoe”, “Maria”, “Lucas”]
my_array[0] = “João”

p my_array
# p shows us the true nature of the Object we are inspecting

# Displays in the terminal:
# [“João”, “Zoe”, “Maria”, “Lucas”]
```

Hashes

A **collection** of Ruby data, stored as a list of **key-value pairs**
The values may appear moe than once, but the **keys** are **unique!**

A Hash is defined with curly braces {}

We can use **any** Ruby object as a key or value

Values are assigned to keys using the Hash Rocket **=>**

```
hash_name = {
  key1 => value1,
  key2 => value2,
  key3 => value3
}
```

Hashes (cont.)

Creating a new Hash

```
my_hash = {
  “cat" => “Garfield”,
  “dog” => “Snoopy”
}
```

or

```
my_hash = Hash.new
my_hash[“cat”] = “Garfield”
my_hash[“dog”] = “Snoopy”
```

Hashes (cont.)

access / **read** a key-value pair

```
my_hash[my_key]
# => The value associated to the key my_key
```

add a key-value pair

```
my_hash[my_new_key] = my_new_value
```

Hashes (cont.)

delete a key-value pair

```
my_hash.delete(key)
```

update a key-value pair

```
my_hash = {
  “cat" => “Garfield”,
  “dog” => “Snoopy”
}
my_hash[“cat”] = “Kitty”
```

Iterating... again!

We can loop over an Array or a Hash, in which case we say we’re **iterating** over them

Iterating… again!

1. Iterating over an Array

```
my_array = [“Bob, “Joe”, “Zack”]

my_array.each do | name |
   puts name
end
```

or

```
my_array.each { | name | puts name }

# Both will print out Bob, Joe, Zack
```

Iterating… again!

2. Iterating over a Multidimensional Array

```
my_array = [ [“Bob, “Joe”, “Zack”], [“Zoe”,
“Nina", “Chloe”] ]

my_array.each do | sub_array |
   sub_array.each do | name |
       puts name
    end
end

# Will print out Bob, Joe, Zack, Zoe, Nina, Chloe
```

Iterating... again!

3. Iterating over a Hash

We need **two placeholders** to represent each key/value pair:

```
students_grades = {
  "Zack" => 7,
  "Zoe" => 10
}

students_grades.each do | student, grade |
  puts "#{student}: #{grade}"
end

# Will print out Zack: 7, Zoe: 10
```

Thank you.