**JS vs Ruby Syntax**

**===============**

* **Classes (class Frame)**
  + **Private methods**
  + **initialize**
* **Variables (a = 5)**

JAVASCRIPT: let

=============

let name = “Chloe”; *// mutable*

RUBY: variable

===========

name = “Chloe”

* **Constants (CONSTANT = 5)**

JAVASCRIPT: const

===============

const NAME = “Chloe”; *// convention states to use uppercase*

RUBY: const

==========

NAME = “Chloe” # *convention states to use uppercase*

* **Private: in classes**

JAVASCRIPT: Private functions

========================

\_methodName *// treat it as if it’s private*

#privateMethodName *// is private and only accessible inside a class*

RUBY: Private methods

==================

private *# written above the methods within the class*

* **Addition ( 6 + 4 )**

JAVASCRIPT & RUBY: Same

=======================

* **Methods (def some\_method)**

JAVASCRIPT: Function

==================

let sayHello = (name) => {

return `Hello ${name}`;

}

sayHello(‘Chloe’);

OR: one line syntax

===============

let sayHello = name => `Hello ${name}`; *// note the use of backticks!*

console.log(sayHello(“Chloe”); *// need to use console.log as there is no return*

RUBY: Methods

============

def say\_hello (name)

Puts “Hello #{name}” *# note the use of double quotation marks!*

end

puts say\_hello(“Chloe”) *# need to use ‘puts’ or ‘print’*

* **If else (if x then some\_method end)**

JAVASCRIPT: If, else if, else

======================

if (num === 0) { *// note the ‘===’ for strict equality*

return “Hello”;

} else if (num > 10) {

return “Bye”;

} else {

return “Hello, bye”;

}

RUBY: if, elsif, end

==============

if num == 0 *# note the ‘==’ for strict equality*

puts “Hello”

elsif num > 10

puts “Bye”

else

puts “Hello, bye”

end

* Arrays (my\_array = []) (my\_array.push etc.)
* Loops (while, loop do, 5.times etc.)

JAVASCRIPT: For

==============

for (let i = 0; i < 10; i++) {

return i;

}

RUBY: For

========

for num in 1..10 do

puts num

end

OR

[1, 3, 9].each { |num| puts num }

OR *# can use curly braces or do/end*

[1, 3, 9].each do |num|

puts num

end

—----------------

JAVASCRIPT: While

===============

RUBY: While loop

==============

num = 10

while num < 20

puts num

num += 1

end

* Instance, Class variables (@instance\_variable, @@class\_variable)
* IO, (gets, chomp, puts, print, p etc)
* Testing (rspec)
* Data types (strings, numbers, booleans etc.)
* TDD/rspec
* Everything is an object in ruby.
* Hashes (my\_hash = Hash.new, my\_hash = {:key => ‘value’}
* **Mathematical operators ( +, - , \*, /)**

JAVASCRIPT & RUBY: Same

=======================

* Procs, blocks, lambdas (do..end, Proc.new)
* Gems/bundler
* Inheritance
* Polymorphism
* Struct
* **irb**

JAVASCRIPT: Node.js

=================

>> node *// opens the node REPL in the command line*

>> node file.js *// opens that specific file in the node REPL*

RUBY: irb or ruby

=============

>> irb *# opens the irb REPL in the command line*

>> irb -r “./file.rb” # *opens that specific file in the irb REPL*

>> ruby file.rb # *opens that specific file in the ruby REPL*

* **Comparators (==, <=, > etc.)**

JAVASCRIPT: Strict equality operator

============================

num === 0 // *three ‘=’ signs*

RUBY: Strict equality operator

=======================

num == 0 # *two ‘=’ signs*

* Ternary (x > 5 ? do\_something : do\_something\_else)
* Built in methods
  + Time
  + Random
* Errors - (raise)
* **Comments (# Look here)**

JAVASCRIPT: Comments

===================

// single line comment

/\* multi

line comment \*/

RUBY: Comments

==============

# single line comment

=begin multi

line comment

=end

* **String interpolation**

JAVASCRIPT: String template literals

============================

let name = “Chloe”

console.log(`Hello, ${name}`) ; *// note the use of backticks and $*

OR

return `Hello, ${name}`;

RUBY: String interpolation

====================

name = “Chloe”

puts “Hello #{name}”

OR

name = “Chloe”

print “Hello #{name}”