JS EQUIVALENTS

#	//	is	===
and	&&	isnt	!==
or	||	not	!
true, yes, on	true	false, no, off	false
@	this	@prop	this.prop
v ||= x	v = v || x	**	exponential operator

ARRAYS

```
arr = [
   1
   2
   3
]
```

```
[1..6]
```

***creates a range of [1, 2, 3, 4, 5, 6]*

```
[1...6]
```

***non-inclusive: [1, 2, 3, 4, 5]*

```
myArr[2..3]
```

***selects the second and third items*

OBJECTS

```
customer = name:"Bob", age:34
```

```
client =
   name:"Anne"
   age:27
```

NEAT TRICKS

```
console.log "Hey #{name}, what's up?"
```

easy variable escaping with double quotes

```
elvis?
```

***true if not undefined or null*

```
default ?= 0
```

```
`
write.javascript();
`
```

CONDITIONALS

```
if name is “Steve”
   console.log name
```

```
if 4 is 4 then alert "The same!"
```

```
mood = “happy” unless monday
```

```
coat = if cold
   “parka”
else
   “jacket”
```

LOOPS

```
for number in array
   number + 1
```

***returns an array with each iteration’s result*

```
nums = (num + 1 for num in arr)
```

```
for key, value of myObject
   console.log key, value
```

FUNCTIONS

```
sum = (x, y) ->
   x + y
```

***automatically returns last logically executed line*

```
total = sum x, y
```

```
returnDefault = (default=0) ->
   return default
```

***can set defaults in the parameters list,*
and/or explicitly use a return statement

```
list = (first, others...) ->
   alert first, others
```

***splats (var...) equate to the end of JS’s arguments object*

```
thisFunc = =>
   console.log @
```

***fat arrow passes in ‘this’ from outer scope*

CLASSES

```
class Car
   drive: (mpg) ->
```

```
class Honda extends Car
   drive: ->
      super 50
```

***extend both parent classes and parent methods*
uses pseudoclassical class style