The pogoscript.js cheat sheet

Variables

Variables can contain spaces, numbers, $ and _.

Defining

```
small number = 8
```

Or, on a new line:

```
small number =
    8
```

Identifier Conjugation

`small number` is the same as `smallNumber`

Calling Functions

```
move file "logbook.txt" to dir "~/docs"
```

Variable Arguments

```
filename = "logbook.txt"
docs dir = "~/docs"
move file @filename to dir (docs dir)
```

Without arguments

```
current time?
```

Or, with a `!`

```
refresh account info!
```

With Optional Arguments

```
web server, port 80
```

With Splat Arguments

```
sum @numbers ...
```

Block Arguments

```
set timeout
    console: log "hi!"
1000
```

Or all on one line:

```
set timeout @{console: log "hi!"} 1000
```

Block Parameters

```
map each #name in @names into
    "<div class='name'>@name</div>"
```

Defining Functions

```
move file @filename to dir (dir name) =
    ...
```

Without Arguments

```
current time? = new @Date
```

Or with a `!`

```
refresh account info! = ...
```

With Optional Parameters

```
web server, port = ...
```

Or, with an default value

```
web server, port 80 = ...
```

With Splat Parameters

```
sum @numbers ... =
    ...
```

Objects

Calling Methods

```
date = new (Date 2011 4 5)
month = date: get month?
date: set minutes 5
```

Defining Methods and Properties

```
size = {}
size: x = 10
size: y = 20
size: area? = self: x * self: y
```

Or as a hash:

```
size = {
    x = 10
    y = 20
    area? = self: x * self: y
}
```

Self

Accessing Self

`self` is akin to `this` in JavaScript

```
self: x
```

Or you can omit `self`

```
:x
```

Blocks Preserve Self

Self is always preserved in blocks:

```
person = {
  name = 'Man Ray'
  say hi later =
    set timeout
      console: log "my name is @(:name)"
    1000
}
```

Self Blocks Redefine Self

use `=>` to allow self to be redefined by the caller of the block:

```
person, name = object =>
    :name = name
```

Array Indexes

```
fib = [0. 1. 1. 2. 3. 5]

fib: 0
fib: 5 = 5
```

Arrays

```
colours = ['red'. 'blue'. 'yellow']
```

Hashes

```
colour scheme = {bg 'red'. fg 'yellow'}
```

Or

```
colour scheme = {
    background = 'red'
    foreground = 'yellow'
}
```

If

```
if (wind speed > 20)
    console: log "gone kitesurfing!"
else
    console: log "no wind, programming"
```

While

```
set timeout
    finished = false
3000

while (!finished)
    console: log "still going"
```

For Each

```
for each #mountain in @moutains
    console: log @mountain
```

For

```
for (n = 0. n < 10. n = n + 1)
    console: log @n
```

Comments

```
// this is a comment

/* this
   is
   a
   comment */
```

Strings

Non-interpolating

```
'Sophie''s World'
```

Interpolating

```
"hi @name"
```

Or

```
"hi @(persons name)"
```

Special Characters

Only work in double-quoted strings:

```
"tab: \t
 linefeed: \n
 carriage return: \r"
```

Multi-line

Newlines are permitted, and indentation on subsequent lines is ignored:

```
some html = "line one
             line two
             line three"
```

Or with single quotes:

```
some html = 'line one
             line two
             line three'
```

Regexps

Regexps are between back-ticks (`` ` ``), and accept the usual sufixes, `g`, `i` and `m`.

```
`.*\.jpg`i: test 'geographe.jpg'
```