Coffeescript

like Javascript but *more awesomer*

- **alternative syntax for Javascript**
- compiles to plain Javascript
- 100% boilerplate-free*
- fixes things you hate
- feels like FP
- works with all runtimes*
- compatible with hype.js
- reads like a good book

* mostly

Basic Style & Syntax

- indentation matters (sorry)
- use comprehensions whenever possible
- functions are first-class citizens; treat them nicely
- one line is better than two
- high-level things use JS best practices*

* except self-executing anonymous functions, we'll get to that

Arrays

```coffee
days = [0..6] # where Sunday is 0

days = [0, 1, 2, 3, 4, 5, 6]

weekdays = days[1..5]

days = days[..]

days = (num for num in [0..6] by 1)
```

Comprehensions

```
doubles = (num + num for num in [1..10])

doubles
# => [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
```

```
keys = ["tag", "rating", "pop"]
keys = ("play_" + k for k in keys)

keys
# => "play_tag", "play_rating", "play_pop"
```

Functions

```
days = [0..6] # where Sunday is 0

is_weekend = (day) ->
  if day in [0, 6] then yes else no

is_weekday = (day) ->
  if day in [1..5] then yes else no

is_day = (day) ->
  is_weekend(day) or is_weekday(day)
```

Functions

```coffee
days = [0..6] # where Sunday is 0

is_weekend = (day) ->
  day in [0, 6]

is_weekday = (day) ->
  day in [1..5]

is_day = (day) ->
  is_weekend(day) or is_weekday(day)
```

Syntactic Diabetes

```coffee
is_weekend(sunday) is true
is_weekday(sunday) is false
is_weekday(sunday) isnt true

7 not in days
3 in days

message = "Sunday has day index #{sunday}."

do (local_three = 3, local_one = 1) ->
  console.log local_three
  console.log local_one

console.log local_three # ReferenceError
```

===	is
!==	isnt
!	not
&&	and
||	or
true	true, yes, on
false	false, no, off
this	@
in	of
	in
	**
	//

```coffee
(3 is 4) isnt true

2 not in [3..8]

turn_light off
#turn_light false

is_good = (drink) ->
  if "coffee" in drink
    yes
  else
    no

is_good = (drink) ->
  "coffee" in drink
```

Unique Tests & Loops

- unless x ↔ if not x
- until x ↔ while not x
- loop ↔ while true

```
unless time > 2100
  ride bike

until time > 2100
  ride bike

loop
  pedal()
```

Existential Operator

```coffeescript
language = "Coffeescript"

# prints only if defined
console.log language if language?

# assigns unless already defined
language ?= "not Coffeescript"

wrong_language =
  language.toUpperCase?().split?(" ")
```

Tangent: Resources

- coffeescript.org's *Overview*
- coffeescript.org's *Annotated Source*
- Ricardo Tomasi's *10 Coffeescript One-Liners to Impress Your Friends*
- the somewhat corny *Coffeescript Ristretto*

“Objects”

```
norfolk:
  name: "Norfolk"
  state: "Virginia"
  stats:
    area_km2: 250
    elev_m: 2.13

norfolk.stats.elev_m
# => 2.13
```

```
light =
  state: off

  flick_switch: ->
    @state = not @state

  turn_on: ->
    @state = on

  turn_off: ->
    @state = off

# light.state is off
light.turn_on
# light.state is on
```

Do you need a class?

```
mower =
  fuel_L: 0.8
  wear: 0.23
  fuel_rate: 0.05
  wear_rate: 0.12

  repair: -> @wear = 0

  ...
```

```
...

status: ->
  console.log "fuel: #{@fuel_L}L, wear: #{@wear}"

mow: (minutes) ->
   @wear += @wear_rate
   newfuel = @fuel_L - (@fuel_rate * minutes)
   if newfuel < 0
     dead_after = @fuel_L / @fuel_rate
     console.log "Dead after #{dead_after} min!"
     @fuel_L = 0

   else
     @fuel_L = newfuel
```

Maybe not.

```
# let's mow the lawn
mower.mow(22.3)

# refuel and try again
mower.fuel_L += 3.0
mower.repair()
mower.mow(22.3)
```

Okay fine, you want classes.

The Good News

- built-in classes are *super great*
- you never have to touch a prototype
- no extra dependencies
  - ~~base2~~
  - ~~prototype.js~~
  - ~~jsclass~~

Classes

```coffeescript
class Animal
  constructor: (name) ->
    @name = name
  move: (meters) ->
    alert @name + " moved #{meters}m."

class Snake extends Animal
  move: ->
    alert "Slithering..."
    super 5

sam = new Snake "Sammy the Python"
sam.move()
```

(modified from class demo at coffeescript.org)

Classes

```coffeescript
class Animal
  constructor: (@name) ->

  move: (meters) ->
    alert @name + " moved #{meters}m."

class Snake extends Animal
  move: ->
    alert "Slithering..."
    super 5

sam = new Snake "Sammy the Python"
sam.move()
```

(modified from class demo at coffeescript.org)

Tangent: `coffee`

```
# interpret coffeescript source
coffee hello_world.coffee

# compile to javascript
# --compile, --bare
coffee -cb hello_world.coffee

# a pretty good REPL
coffee

# you're all smart
coffee --help
```

(modified from class demo at coffeescript.org)

Destructuring Assignment

```coffeescript
last = 2
current = 3
[last, current] = [current, current + last]

futurists =
  sculptor: "Umberto Boccioni"
  painter:  "Vladimir Burliuk"
  poet:
    name:   "F.T. Marinetti"
    address: [
      "Via Roma 42R"
      "Bellagio, Italy 22021"
    ]

{poet: {name, address: [street, city]}} = futurists
```

(object example taken directly from coffeescript.org)

Splats...

```
cli_handler = (subcommand, args...) ->

  if "--help" in args
    console.log usage
    exit_with_status 0

  else if subcommand is "commit"
    make_commit get_head()
    exit_with_status 0

  else
    console.log usage
    exit_with_status 2
```

Splats...

```
days = [0..6]
days = [0...7]

[sunday, weekdays..., saturday] = days

weekends = [sunday, saturday]

sdn = [0..9]
[..., last_sdn] = sdn
[first_sdn, ...] = sdn

[first_sdn, ..., last_sdn] = sdn
```

…so much more

- function binding to deal with scope of `this`
- better comparisons
- even fancier splats…
- source maps
- block syntax for arrays, strings, comments, regex
- native switches
- native try/catch
- cake (like make)

```
$ sudo npm install -g coffee-script
```