CoffeeScript

introduction

大綱

- **CoffeeScript 介紹**
- 語法
- 類別

CoffeeScript? Why?

- 語法較為簡潔
- 避免掉一些js常見錯誤
- => , @ 的使用: ES6
- 字串代換
- 支援class
- 不少套件的是以coffeescript寫成
- http://www.netmagazine.com/features/10-good-reasons-use-coffeescript

使用 CoffeeScript

```
#coffeescript命令列
~ $ npm install -g coffee-script
```

```
#編譯coffee檔成js檔
~ $ coffee --compile my-script.coffee
```

注解 & 變數

```
# 單行注解

###
   多行注解
###
```

(不會出現在編譯後的程式碼)

```
/*
   多行注解
*/
```

```
# variable
myVariable = "test"
```

```
var myVariable;

myVariable = "test";
```

operator & alias

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

函示宣告

```
# function
func = ->
  # An extra line
  "bar"
```

```
var func;

func = function() {
  return "bar";
};
```

```
times = (a, b) -> a * b
```

```
var times;

times = function(a, b) {
  return a * b;
};
```

```
times = (a = 1, b = 2) -> a * b
```

```
var times;

times = function(a, b) {
  if (a == null) {
    a = 1;
  }
  if (b == null) {
    b = 2;
  }
  return a * b;
};
```

函式

```
sum = (nums...) ->
  result = 0
  nums.forEach (n) -> result += n
  result
```

```
var sum,
  __slice = [].slice;

sum = function() {
  var nums, result;
  nums = 1 <= arguments.length ?
__slice.call(arguments, 0) : [];
  result = 0;
  nums.forEach(function(n) {
    return result += n;
  });
  return result;
};
```

```
a = "Howdy!"

console.log a
# Equivalent to:
console.log(a)

console.log foo a
# Equivalent to:
console.log(foo(a))
```

```
var a;

a = "Howdy!";

console.log(a);

console.log(a);

console.log(foo(a));

console.log(foo(a));
```

context binding

```
this.clickHandler = -> alert "clicked"
element.addEventListener "click", (e) => this.clickHandler(e)
```

```
var _this = this;

this.clickHandler = function() {
  return alert("clicked");
};

element.addEventListener("click", function(e) {
  return _this.clickHandler(e);
});
```

物件

```
object1 = {one: 1, two: 2}

# Without braces
object2 = one: 1, two: 2

# Using new lines instead of commas
object3 =
  one: 1
  two: 2

User.create(name: "John Smith")
```

```
var object1, object2, object3;

object1 = { one: 1, two: 2 };

object2 = { one: 1, two: 2 };

object3 = { one: 1, two: 2 };

User.create({
  name: "John Smith"
});
```

陣列

```
array1 = [1, 2, 3]

array2 = [
  1
  2
  3
]

array3 = [1,2,3,]
```

```
var array1, array2, array3;

array1 = [1, 2, 3];

array2 = [1, 2, 3];

array3 = [1, 2, 3];
```

if else

```
alert "It's cold!" if heat < 5

if not true then "Panic"

unless true
  "Panic"

if true is 1
  "Type coercion fail!"

if true isnt true
  alert "Opposite day!"
```

```
if (heat < 5) { alert("It's cold!"); }

if (!true) { "Panic"; }

if (!true) { "Panic"; }

if (true === 1) { "Type coercion fail!"; }

if (true !== true) { alert("Opposite day!"); }
```

字串代換

```
color = "Blue"
out1 = "My favorite color is: #{favourite_color}"
out2 = 'My favorite color is: #{favourite_color}'
```

```
var color, out1, out2;

color = "Blue";

out1 = "My favorite color is: " + favourite_color;

out2 = 'My favorite color is: #{favourite_color}';
```

迴圈

```
for name in ["Roger", "Roderick", "Brian"]
  alert "Release #{name}"
```

```
var name, _i, _len, _ref;
_ref = ["Roger", "Roderick", "Brian"];
for (_i = 0, _len = _ref.length; _i < _len; _i++) {
  name = _ref[_i];
  alert("Release " + name);
}
```

迴圈

```
#第二個參數為index
for name, i in ["Roger the pickpocket", "Roderick the robber"]
  alert "#{i} - Release #{name}"
```

```
var i, name, _len, _ref;
_ref = ["Roger the pickpocket", "Roderick the robber"];
for (i = 0, _len = _ref.length; i < _len; i++) {
  name = _ref[i];
  alert("" + i + " - Release " + name);
}
```

迴圈

```
release prisoner for prisoner in ["Roger", "Roderick", "Brian"]
```

```
var prisoner, _i, _len, _ref;
_ref = ["Roger", "Roderick", "Brian"];
for (_i = 0, _len = _ref.length; _i < _len; _i++) {
  prisoner = _ref[_i];
  release(prisoner);
}
```

迴圈 - filter

```
prisoners = ["Roger", "Roderick", "Brian"]
release prisoner for prisoner in prisoners when prisoner[0] is "R"
```

```
var prisoner, prisoners, _i, _len;
prisoners = ["Roger", "Roderick", "Brian"];
for (_i = 0, _len = prisoners.length; _i < _len; _i++) {
  prisoner = prisoners[_i];
  if (prisoner[0] === "R") {
    release(prisoner);
  }
}
```

陣列操作

```
range = [1..5]
```

```
var range;
range = [1, 2, 3, 4, 5];
```

```
firstTwo = ["one", "two", "three"][0..1]
```

```
var firstTwo;
firstTwo = ["one", "two", "three"].slice(0, 2);
```

```
numbers = [0..9]
numbers[3..5] = [-3, -4, -5]
```

```
var numbers, _ref;
numbers = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9];
[].splice.apply(numbers, [3, 3].concat(_ref
= [-3, -4, -5])), _ref;
```

```
#字串也可以操作
my = "my string"[0..2]
```

```
var my;
my = "my string".slice(0, 3);
```

陣列查找（跨版本）

```
words = ["rattled", "roudy", "rebbles", "ranks"]
alert "Stop wagging me" if "ranks" in words
```

```
var words;
var __indexOf = Array.prototype.indexOf || function(item) {
  for (var i = 0, l = this.length; i < l; i++) {
    if (this[i] === item) return i;
  }
  return -1;
};
words = ["rattled", "roudy", "rebbles", "ranks"];
if (__indexOf.call(words, "ranks") >= 0) {
  alert("Stop wagging me");
}
```

@, ::, ?

```
@saviour = true
```

```
this.saviour = true
```

```
User::first = -> @records[0]
```

```
User.prototype.first = function() {
  return this.records[0];
};
```

?

```
praise if brian?
```

```
if (typeof brian !== "undefined" && brian !
== null) {
  praise;
}
```

```
velocity = southern ? 40
```

```
var velocity;
velocity = typeof southern !==
"undefined" && southern !== null ?
southern : 40;
```

```
blackKnight.getLegs()?.kick()
```

```
var _ref;
if ((_ref = blackKnight.getLegs()) !=
null) {
  _ref.kick();
}
```

```
blackKnight.getLegs().kick?()
```

```
var _base;
if (typeof (_base =
blackKnight.getLegs()).kick ===
"function") {
  _base.kick();
```

Class

```
class Animal
  price: 5

  sell: =>
    alert "Give me #{@price} shillings!"

animal = new Animal
$("#sell").click(animal.sell)
```

```
var Animal, animal;
var __bind = function(fn, me){ return function(){ return fn.apply(me,
arguments); }; };
Animal = (function() {
  function Animal() {
    this.sell = __bind(this.sell, this);
  }
  Animal.prototype.price = 5;
  Animal.prototype.sell = function() {
    return alert("Give me " + this.price + " shillings!");
  };
  return Animal;
})();
animal = new Animal;
$("#sell").click(animal.sell);
```

Class method

```
class Animal
  @find: (name) ->

Animal.find("Parrot")
```

```
var Animal;
Animal = (function() {
  function Animal() {}
  Animal.find = function(name) {};
  return Animal;
})();
Animal.find("Parrot");
```

Class inherit

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

  alive: ->
    false

class Parrot extends Animal
  constructor: ->
    super("Parrot")

  dead: ->
    not @alive()
```

Resources

- http://coffeescript.org/
- http://coffeescriptcookbook.com/
- The Little Book on CoffeeScript
- http://js2coffee.org/