Neat (Stupid?) Ruby Tricks

take_while/drop_while

```
[1,3,7,4,8].take_while{ |x| x < 5 }
=>[1,3]

[1,3,7,4,8].drop_while{ |x| x < 5 }
=>[7,4,8]
```

Why?

Super fast on sorted data

```
(1..100000].select{ |x| x < 5 }
```

1000000 comparisons

```
[1..10000].take_while{ |x| x < 5 }
```

5 comparisons

Hashes with Default Value

```
some_hash = {}
some_hash[ :key ] # nil


some_hash = Hash.new do |key|
  “What is #{key}?”
end

some_hash[ :key ] # “What is key?”
```

Why?

Hashes that throw exception with unknown key

```
some_hash = Hash.new do |key|
   raise( “#{ key } not found!” )
end

some_hash[ :key ] # Boom!
```

Hash Syntax Sugar

```
{ :a => 1, :b => 2 }
{ a: 1, b: 2 }
```

Only works for symbols!

```
{ 'a': 1 }
{ 1: 'one' }
```

Why?

```
Book.new( :author => 'DHH',
          :title  => 'AWDWR')
```

```
Book.new( author: 'DHH', title: 'AWDWR' )
```

```
Book.new author: 'DHH', title: 'AWDWR'
```

Proc#=== alias to Proc#call

```
multiple_of_3 = lambda do |number|
  number.modulo( 3 ).zero?
end

multiple_of_3.call( 6 )

multiple_of_3 === 6
```

Why?

Case uses ===

```
case number
  when multiple_of_3: puts “3!”
  when multiple_of_2: puts “2!”
end
```

Dynamic!

```
def multiple_of( x )
  lambda{ |n| n.modulo( x ).zero? }
end

case number
  when multiple_of( 3 ): puts “3!”
  when multiple_of( 2 ): puts “2!”
end
```