Просветление хипстера

...или что такое хлопок одной ладонью на ruby

Слияние двух хешей

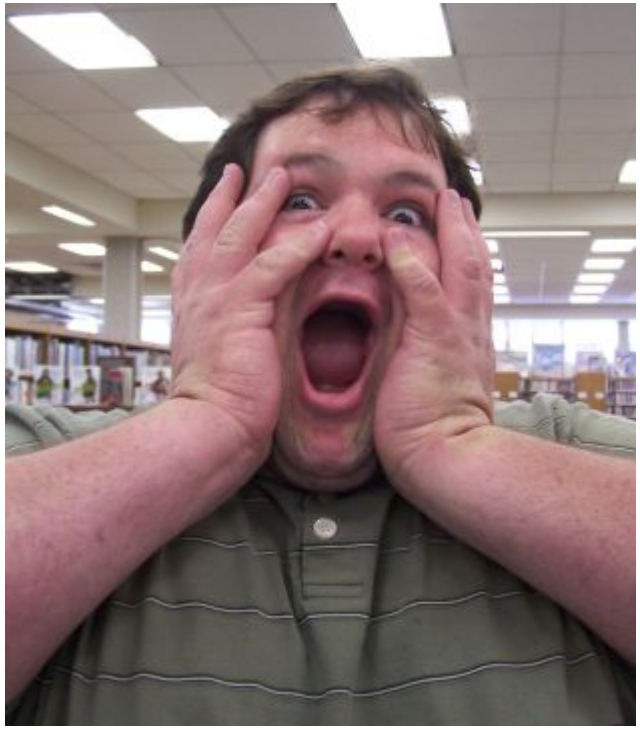

```
first_hash = { "first" => { "second" => {
"third" => "payload" }}}

second_hash = { "first" => { "another" => {
"zen" => "eats our brains"} }}

p first_hash.merge(second_hash)
```

```
p first_hash.merge(second_hash)

# => {"first"=>{"another"=>{"zen"=>"eats our brains"}}}
# Oh, no!
```



Включаем суперсилу

_why

hsh.merge(other_hash) → new_hash
hsh.merge(other_hash){ |key, oldval, newval| block } → new_hash

Returns a new hash containing the contents of *other_hash* and the contents of *hsh*. If no block is specified, the value for entries with duplicate keys will be that of *other_hash*. Otherwise the value for each duplicate key is determined by calling the block with the key, its value in *hsh* and its value in *other_hash*.

```
h1 = { "a" => 100, "b" => 200 }
h2 = { "b" => 254, "c" => 300 }
h1.merge(h2)   #=> {"a"=>100, "b"=>254, "c"=>300}
h1.merge(h2){ |key, oldval, newval| newval - oldval }
               # => { "a"=>100, "b"=>54,  "c"=>300 }
h1             # => { "a"=>100, "b"=>200 }
```

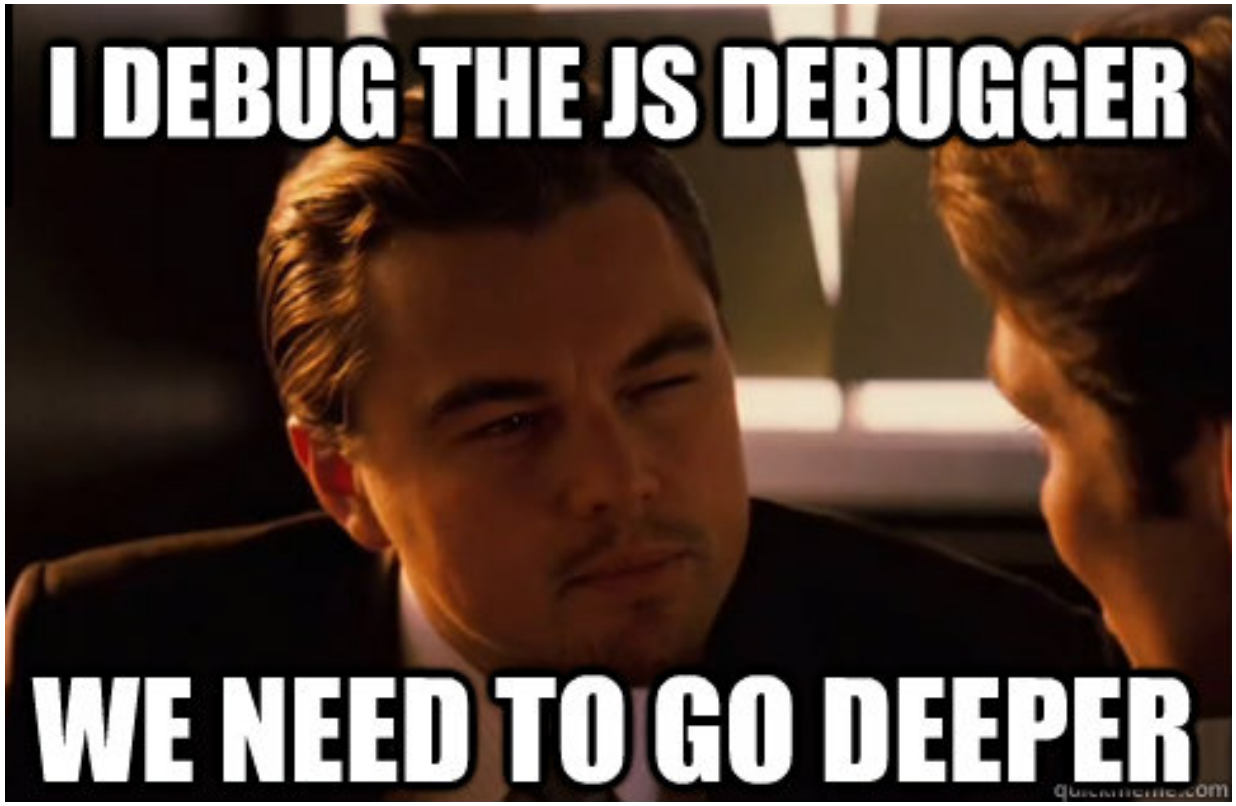
```ruby
first_hash = { "first" => { "second" => { "third" =>
"payload" }}}
second_hash = { "first" => { "another" => { "zen" => "eats
our brains"} }}

m = proc{|_, o, n| o.merge(n, &m)}

p first_hash.merge(second_hash, &m)

# => {"first"=>{"second"=>{"third"=>"
payload"}, "another"=>{"zen"=>"eats our
brains"}}}
# Wow!
```

но...



...что насчет вложенных хешей?

```ruby
first_hash = { "first" => { "second" => { "third" =>
"payload" }}}

m = proc{|_, o, n| o.merge(n, &m)}


third_hash = { "first" => { "second" => { "mother" =>
"kill me, plz!" }}}

# We should go deeper!
p first_hash.merge(third_hash, &m)

# => {"first"=>{"second"=>{"third"=>"payload", "mother"=>"
kill me, plz!"}}}

# Enlightment was reached with easy
```