slides.first

slides[0]

```
{ :agenda =>
    [ [ 'Arrays','Hashes' ],
      'Break',
      [ 'Gists','File Structure' ],
      [ 'Exercise' ] ]
```

slides[1]

slides[1][:agenda][0][0]

slides[2]

‘Arrays’

slides[3]

Array.new

slides[4]

~~Array.new~~

slides[5]

[ ]

slides[6]

[1,2,3]

slides[7]

[:name,2,OpenStruct.new]

slides[8]

variable = [1,2,3]

slides[9]

variable[0] # 1
variable[1] # 2
variable[2] # 3

slides[10]

variable[-3] # 1
variable[-2] # 2
variable[-1] # 3

slides[11]

```
variable.first # 1

variable.last # 3
```

slides[12]

```
variable[0..0] #=> [1]
variable[0..1] # => [1,2]
variable[0..-2] # => [1,2]
```

slides[13]

```
[1,2,3][0] # 1
[1,2,3][1] # 2
[1,2,3][2] # 3
```

slides[14]

var = [1,2,2,3,3]

var.uniq # [1,2,3]
var # [1,2,2,3,3]

slides[15]

WTF!

slides[16]

var = [1,2,2,3,3]

var.uniq! # [1,2,3]
var # [1,2,3]

slides[17]

! Operation is performed on itself

Result *is sometimes* nil

slides[18]

?

slides[19]

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

var.include?(1) # true
```

slides[20]

? Result is expected to be:

truthy or **falsy**

slides[21]

? falsy: **nil or false**
truthy: **anything but**

slides[22]

%w[good bad ugly].sample

"good" or "bad" or sometimes "ugly"

slides[23]

slides[1][:agenda][0][1]

slides[24]

‘Hashes’

slides[25]

Hash.new

slides[26]

~~Hash.new~~

slides[27]

{ }

slides[28]

May be confused with ~~blocks~~

```
[1,2,3].each{|item| puts item }
```

slides[29]

{ ‘key’ => ‘value’ }

slides[30]

{ :key => [1,:b,’hash’] }

slides[31]

colors =
{ :red => 0,
:green => 128,
:blue => 255 }

slides[32]

```
colors[:red]  # 0
colors[:green] # 128
colors[:blue] # 256
```

slides[33]

```
colors.keys
[ :red, :green, :blue ]
```

slides[34]

```
colors.values
[ 0, 128, 255 ]
```

slides[35]

colors.include?
colors.key?
colors.has_key?

slides[36]

colors.default = 255
colors[:alpha] # 255

slides[37]

‘Enumerable’

slides[38]

My Favorite!

slides[39]

NOT RUBY!

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

for (x = 0; x < array.length; x++) {
      puts array[x]
}
```

slides[40]

```
[1,2,3].each do |item|
    puts item
end
```

slides[41]

```
[1,2,3].each do |1|
    puts 1
end
```

slides[42]

```
[1,2,3].each do |2|
    puts 2
end
```

slides[43]

```
[1,2,3].each do |3|
    puts 3
end
```

slides[44]

```
[1,2,3].each {|item| puts item }
```

slides[45]

```
[1,2,3].each {|item| puts item }
# [1,2,3]
```

slides[46]

```
[1,2,3].each {|item| item * 2 }
# [1,2,3]
```

slides[47]

```
[1,2,3].map{|item| item * 2 }

# [2,4,6]
```

slides[48]

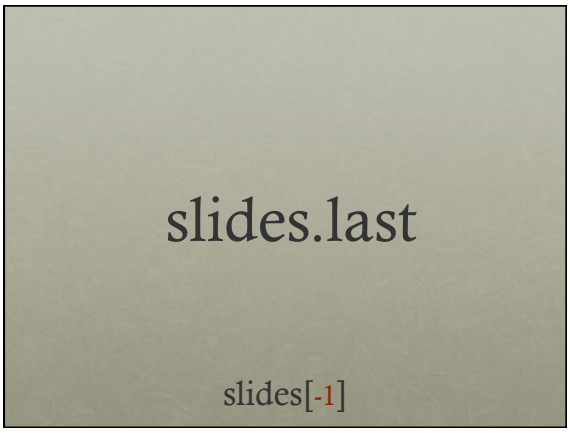
```
doubled = [1,2,3].map{|item| item * 2 }

# [2,4,6]
```

slides[-3]

Questions?

slides[-2]


slides.last
slides[-1]