Ruby trunk - Feature #10254

Array#each and Array#map for nested arrays

09/18/2014 02:33 AM - sawa (Tsuyoshi Sawada)

Status:	Feedback
Priority:	Normal
Assignee:	
Target version:	

Description

In order to allow iteration over elements of arrays nested within an array, I propose to pass Array#each and Array#map an optional argument that expresses the depth to iterate over.

Conventionally, iterating over nested elements requires nested each or map:

```
[[1, 2], [3, 4], [5, 6]].map{|a| a.map{|e| e + 1}} #=> [[2, 3], [4, 5], [6, 7]]
[[[1, 2], [3, 4]], [[5, 6]]].map{|a| a.map{|a| a.map{|e| e + 1}}} #=> [[[2, 3], [4, 5]], [[6, 7]]]
```

With the proposed optional argument, this would be done by:

```
[[1, 2], [3, 4], [5, 6]].map(1){|e| e + 1} #=> [[2, 3], [4, 5], [6, 7]]
[[[1, 2], [3, 4]], [[5, 6]]].map(2){|e| e + 1} #=> [[[2, 3], [4, 5]], [[6, 7]]]
```

Absence of the parameter should be understood as the parameter being defaulted to 0.

```
[1, 2, 3, 4, 5, 6].map{|e| e + 1} #=> [2, 3, 4, 5, 6, 7]
[1, 2, 3, 4, 5, 6].map(0){|e| e + 1} #=> [2, 3, 4, 5, 6, 7]
```

History

#1 - 09/18/2014 04:33 PM - matz (Yukihiro Matsumoto)

- Status changed from Open to Feedback

It should be separated in different method, e.g. #nested_map in my opinion.

Matz.

#2 - 09/19/2014 06:58 PM - avit (Andrew Vit)

nested_map makes sense since there is a flat_map. This is similar but different:

```
[[[1, 2], [3, 4]], [[5, 6]]].flatten(2).map{|e| e + 1} #=> [2, 3, 4, 5, 6, 7]
```

#3 - 09/19/2014 07:38 PM - trans (Thomas Sawyer)

It's a shame it's not map_flat, and thus in this case map_nested, as it would organize documentation in a nicer fashion.