Ruby master - Feature #8172

IndexError-returning counterparts to destructive Array methods

03/27/2013 10:51 AM - sawa (Tsuyoshi Sawada)

Status:	Feedback
Priority:	Normal
Assignee:	
Target version:	

Description

=begin
There are a few desctructive (({Array})) methods that take an index as an argument and silently insert (({nil})) if the index is out of range:

```
a = []; a[1] = :foo; a # => [nil, :foo]
[].insert(1, :foo) # => [nil, :foo]
[].fill(:foo, 1, 1) # => [nil, :foo]
```

Among them, (({Array#[]})) has a counterpart that returns an (({IndexError})) when the index is out of range:

```
[].fetch(1) # => IndexError
```

and this is useful to avoid bugs that would be difficult to find if (({Array#[]})) were used. However for (({Array#insert})) and (({Array#fill})), there are no such counterparts, and that fact that these methods silently insert (({nil})) is often the cause of a bug that is difficult to find. I suggest there should be some versions of these methods that return (({IndexError})) when index is out of range:

```
[].insert!(1, :foo) # => IndexError
[].fill!(:foo, 1, 1) # => IndexError
```

I believe this would make debugging easier.
=end

History

#1 - 03/27/2013 11:00 AM - sawa (Tsuyoshi Sawada)

=begin
In the above, I missed to say that there is no counterpart for (({Array#[]=})). There should be one for it as well, but I cannot think of a good method name.
=end

#2 - 03/27/2013 11:24 AM - nobu (Nobuyoshi Nakada)

=begin
(({Hash})) has (({#store})) as an alias of (({#[]=})).
=end

#3 - 03/27/2013 11:53 AM - duerst (Martin Dürst)

This may be just an issue of wording: You say "index is out of range".
By definition, Ruby arrays don't have a range. They can grow
dynamically. In many cases, this is a big feature.

Also, you complain about inserting nil. So what about the following case:

a = [1, 2, 3]; a[3] = :foo; a # => [1, 2, 3, :foo]

There is no nil, so maybe this is okay. But the array is expanded.
Should there be an error?

Regards, Martin.

On 2013/03/27 10:51, sawa (Tsuyoshi Sawada) wrote:

> Issue #8172 has been reported by sawa (Tsuyoshi Sawada).
>
> ---
>
> Feature #8172: IndexError-returning counterparts to destructive Array methods

#4 - 03/27/2013 11:55 AM - sawa (Tsuyoshi Sawada)

=begin

> Martin
> For clarification, I meant to have it return an error only in cases where (({nil})) needs to be inserted otherwise. So cases like the following should not return an error:

```
a = [1, 2, 3]; a[3] = :foo; a # => (actually it should be a different method name) [1, 2, 3, :foo]
[1, 2, 3].insert!(3, :foo) # => [1, 2, 3, :foo]
[1, 2, 3].fill!(:foo, 3) # => [1, 2, 3, :foo]
```

=end

#5 - 03/27/2013 12:26 PM - matz (Yukihiro Matsumoto)

- Status changed from Open to Feedback

I am not against the idea itself, but using bang (!) for the names is not consistent with other bang methods.

Matz.