Ruby master - Feature #9347

Accept non callable argument to detect

01/03/2014 07:37 AM - marcandre (Marc-Andre Lafortune)

Status:	Open
Priority:	Normal
Assignee:	marcandre (Marc-Andre Lafortune)
Target version:	

Description

Currently, the only argument that Enumerable#detect accepts is a callable object.

Shouldn't we accept non callable objects too?

```
[42].detect(:not_found){} # => NoMethodError: undefined method `call' for :not_found:Symbol
 # would return :not_found instead.
```

I'd suggest that if the given argument does not respond_to? :call, then it would be returned as is instead of raising an error as currently.
Wouldn't this be more flexible and possibly more performant?

Inspired by http://stackoverflow.com/questions/20883414/why-does-enumerabledetect-need-a-proc-lambda

History

#1 - 01/03/2014 11:38 AM - matz (Yukihiro Matsumoto)

I am afraid that kind of conditional behavior could error prone (e.g. accidental misbehavior when one forget having #call in argument object). I'd rather have #detect with keyword argument, like ary.detect(ifnone: :notfound).

Matz.

#2 - 01/03/2014 01:23 PM - fuadksd (Fuad Saud)

I think, if it was possible, a call like this ary.detect(:sym) would make more sense by using case equality for comparison instead of #call. It would be useful with regexes for example. There's a ticket proposing this but I can't seem to find it.

#3 - 03/18/2014 08:50 PM - marcandre (Marc-Andre Lafortune)

- Assignee changed from matz (Yukihiro Matsumoto) to marcandre (Marc-Andre Lafortune)

I agree, an optional argument would be best. I'll propose a patch. Could we use if_none though?

#4 - 03/19/2014 01:41 AM - nobu (Nobuyoshi Nakada)

[GH-561] https://github.com/ruby/ruby/pull/561 ?

#5 - 01/05/2018 09:00 PM - naruse (Yui NARUSE)

- Target version deleted (2.2.0)