It's a common idiom to use a fiber-local variable, but return a default value if it hasn't been assigned. Here's an example of Ruby code I would expect to be able to write:

```ruby
def feature_enabled?
  Thread.current.fetch(:feature_enabled, true)
end
```

Unfortunately, Thread.current does not implement `fetch`, like Hash does. So I have to write the more verbose:

```ruby
def feature_enabled?
  Thread.current.key?(:feature_enabled) ? Thread.current[:feature_enabled] : true
end
```

Thanks for your consideration!

### Associated revisions

**Revision 7d5b4e4a - 02/22/2017 07:16 AM - nobu (Nobuyoshi Nakada)**

Thread#fetch

* thread.c (rb_thread_fetch): add new method Thread#fetch.
  [Feature #13009]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@57683 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

**Revision 57683 - 02/22/2017 07:16 AM - nobu (Nobuyoshi Nakada)**

Thread#fetch

* thread.c (rb_thread_fetch): add new method Thread#fetch.
  [Feature #13009]

**Revision 57683 - 02/22/2017 07:16 AM - nobu (Nobuyoshi Nakada)**

Thread#fetch

* thread.c (rb_thread_fetch): add new method Thread#fetch.
  [Feature #13009]

**Revision 57683 - 02/22/2017 07:16 AM - nobu (Nobuyoshi Nakada)**

Thread#fetch

* thread.c (rb_thread_fetch): add new method Thread#fetch.
  [Feature #13009]

### History

**#1 - 12/06/2016 01:26 PM - shevegen (Robert A. Heiler)**

I can not say whether this feature is important or not, but on the syntax suggestion - I like the above (on the premise that it behaves as .fetch on Hash does; I have not checked this either).

**#2 - 12/06/2016 03:27 PM - nobu (Nobuyoshi Nakada)**

Thread.current just returns a Thread instance, so it should be Thread#fetch method.

https://github.com/ruby/ruby/compare/trunk...nobu:feature/Thread%23fetch
Thread#fetch

- thread.c (rb_thread_fetch): add new method Thread#fetch. [Feature #13009]