Environmental Variable Programming

環境変数プログラミング

sanemat {AT} tachikoma.io

Environment variables in CI

CI での環境変数の扱い

Environment Variables - Travis CI

```
CI=true
TRAVIS=true
CONTINUOUS_INTEGRATION=true
DEBIAN_FRONTEND=noninteractive
HAS_JOSH_K_SEAL_OF_APPROVAL=true
USER=travis (do not depend on this value)
HOME=/home/travis (do not depend on this value)
LANG=en_US.UTF-8
LC_ALL=en_US.UTF-8
RAILS_ENV=test
RACK_ENV=test
MERB_ENV=test
(snip)
```

Environment variables - CircleCI

```
CIRCLECI=true
CI=true
CIRCLE_PROJECT_USERNAME=foo
  The username or organization name of the project being tested,
  i.e. "foo" in circleci.com/gh/foo/bar/123
CIRCLE_PROJECT_REPONAME=bar
  The repository name of the project being tested,
  i.e. "bar" in circleci.com/gh/foo/bar/123
CIRCLE_BRANCH=master
  The name of the branch being tested, e.g. 'master'.
```

`(snip)`

Many environment variables, they depend on CI.

いろんな環境変数、CI 環境によって違う。

There is no shared rules, I think.

特にルールはない (と思う)。あるのかも?

I want to know this later.

ちょっと後で聞いてみたいけど。

But there are similar rules.

結構なんとなく似通ったいろいろ。

For example, `CI=true`. Some tool sends code coverage to "Coveralls" if `CI=true`. I'll talk about troubles I met, tools I build.

`CI=true` など。ツールによっては、`CI=true` だとカバレッジを coveralls に送る、など。この分野で、ハマったことや作ったモジュール、もう他の人がはまらなくていいようにをまとめました。

Rules (Feeling)

なんとなく感じ取ったルール

Truthy case

truthy のとき

- Some string
  - e.g. `CI=true`

何か文字列が入る たとえば `CI=true`

Falsy case

falsy のとき

There are no key in environment variables　環境変数の key 自体がなくなるパターン

```
if ENV['SOME_VALUE']
  # your code
end
```

Environment variables' value is empty string　環境変数の value が空文字列のパターン

```
if !ENV['SOME_VALUE'].empty?
  # your code
end
```

Problems

問題

That moment when you use environment variables. part1

あるある 1

Some CI env has a document which shows the truthy examples of keys and values. But they does not show falsy one. "not provide the key" or "empty string" or else.

結構こういう、このキーである、という情報はどうにかドキュメント有るのだが、こういう値を取りうる、という記述が欠けていることが多い。

And keys and values does not share between CI envs, of course. And they have different behavior between keys on same CI env!

しかも、(当然だけど)CI 環境間で統一されていない。さらに、同じ CI 環境内でも、key によって違うことがある。

They say "document patch welcome!", yes I know, but…

"document patch welcome!" って言われるんだけど、それはツライ。

That moment when you use environment variables. part2

あるある 2

Ruby specific problem, The empty string means falsy in many CI envs, but the empty string means truthy in Ruby.

Ruby 固有のメンドイこととしては、CI 環境的には空文字列は falsy だけど、Ruby 的には空文字列は truthy

That moment when you use environment variables. part3

あるある 3

First I made the module with Travis-CI hard-coded, but after I want to use this with CircleCi. Many many times.

Travis CI 決め打ちで作って、CircleCI で使いたくなる よくある

That moment when you use environment variables. part4

あるある 4

The more CI envs, the more complexity in test. Test became a nested structure, because test and pull request themselves run on CI env. Sometimes I forget deleting related environment variables, restoring them.

テストでいちいち考えなくちゃいけないことが増える。 pull request やテスト自体が CI 環境上で動くので、二重構造になる。環境変数消し漏れたり、戻し漏れたり、で動かないはずのものが動く分岐の方に行ってしまったり。

env_branch

I build gem which get branch information from environment variables.

branch 情報を取り出したいことがよくあって、環境変数から取り出す部分を gem に切り出した。

Usage

```
require 'env_branch'

env_branch = EnvBranch.new
env_branch.branch? #=> true
env_branch.branch_name #=> 'your-branch-name'
```

Question

Q. branch 名って `git branch` コマンドで取れるのでは?

A. CI 環境によって違う

取れる CI 環境も有る。 Travis-CI だと、環境変数から取るのが良い。

何も設定せずに、リポジトリの中でブランチ作って、pull request を送ると
テストが 2 本走る。

pull request の test をするときに、

```
$ git clone --depth=50 https://github.com/packsaddle/rubocop-select.git packsaddle/ruboc
$ cd packsaddle/rubocop-select
$ git fetch origin +refs/pull/58/merge:
$ git checkout -qf FETCH_HEAD

$ git branch
* (detached from FETCH_HEAD)
  master
```

branch のpush の test をするときに、

```
git clone --depth=50 --branch=sanemat-patch-1 https://github.com/packsaddle/ruby-parse_g
$ cd packsaddle/ruby-parse_gemspec
$ git checkout -qf 1e185190162d8a3b021bbb27aa422c4b00272117

$ git branch
* (detached from 1e18519)
  sanemat-patch-1
```

なので、`git branch` しても current branch に branch 名はない。

helper

```
require 'env_branch/test_helper'

class TestExample < Test::Unit::TestCase
  extend ::EnvBranch::TestHelper

  def self.startup
    stash_env_branch
  end

  def self.shutdown
    restore_env_branch
  end
end
```

各 CI 環境での branch に関する環境変数を、いったん退避して、最後書き戻す。便利。

env_pull_request

pull request id を取り出したい。`https://github.com/sanemat/node-boolify-string/pull/16` だとしたら、'16' これ。 GitHub の pull request に対して hook なりで何かをしたい場合、これを使ってリクエストする必要がある。

pull_request 番号を取り出したいことがよくあって、環境変数から取り出す部分を gem に切り出した。

```
require 'env_pull_request'

env_pull = EnvPullRequest.new
env_pull.pull_request? #=> true
env_pull.pull_request_id #=> 800
```

```
require 'env_pull_request/test_helper'

class TestExample < Test::Unit::TestCase
  extend ::EnvPullRequest::TestHelper
```

```
  def self.startup
    stash_env_pull_request
  end

  def self.shutdown
    restore_env_pull_request
  end
end
```

便利なので使ってください

Supported CI env

対応している CI 環境

- env_branch
  - Travis-ci
  - CircleCI
- env_pull_request
  - Travis-ci
  - CircleCI
  - Jenkins GitHub pull request builder plugin

余談

pull request

falsy のとき、環境変数の key 自体がない場合と、value が空文字列の場合があるといった。

引用

> TRAVIS_PULL_REQUEST: The pull request number if the current job is a pull request, “false” if it’s not a pull request.

“false”

!???

CI 環境の環境変数、基本的には

- Truthy case
  - Some string
- Falsy case
  - There are no key in environment variables
  - Environment variables' value is empty string
  - Environment variables' value is string "false"
- truthy のとき
  - 何か文字列が入る
- falsy のとき
  - 環境変数の key 自体がなくなるパターン
  - 環境変数の value が空文字列のパターン
  - 環境変数の value が "false" のパターン

そういうのにも env_branch や env_pull_request は対応済みです。なのでぜひ使って。コレを使うと、余計なことに悩まされなくて良い。その他 ci 環境は pull request ください。 drone や wercker など。使う人が対応しようってことで。

楽しい環境変数プログラミングを。