

Tachikoma next

sanemat {AT} tachikoma.io

Bundle update regularly, frequently

2013-05-31 - Kensuke Nagae @kyanny

[Continuous gem dependency updating with Jenkins and Pull Request](#)

2013-06-23 - Kenichi Murahashi @sanemat

[Tachikoma gem first version v2.0.0 \(not published\) commit](#)

2014-09-17 - Kenichi Murahashi @sanemat

[When was the build passing? / Travis CI Meetup 2014-09-17](#)

Motivation

- shell script で書くのがそもそもツライ
- ほぼ同じスクリプトがプロジェクトごとに散らばるのでツライ

Tachikoma gem

pros

- yaml の設定を置くと、gem とコマンドラインで完結する

cons

- カスタマイズ性低い
- 実行環境は自分で作成する必要がある
- 設計思想として、自前 jenkins で動かすのを第一に考えていた
 - CI as a web service の発展

tachikoma.io

pros

- github OAuth で連携, toggle 一発
- .tachikoma.yml で設定

cons

- リポジトリの読み書き権限もらうのが非常に苦戦
- public 版も、ユーザー権限で push 出来るようにするものはあるが、300+ のユーザー中利用者 1(自分のみ)
- ユーザー権限で push できると、bot のcommit の上に自分のコミット 詰めるので、すごい便利なのだが...
- カスタマイズ性低い

揺り戻し

- shell script や ruby script で書けばいいじゃん

しかし

やっぱりつらい

shell script or Ruby script で書くの何がツライのか直に Octokit 使えばいい だけなのわかるけど、いちいち API 調べてとかダルい

対象を絞ってより使いやすくするアプローチ

Ruby gem の場合

- <https://www.deppbot.com/>
- circleci-bundle-update-pr
- ci-bundle-update

Node.js npm の場合

- <http://greenkeeper.io/>

ここから知見 + 私見 + 主張

カスタマイズ欲求がプロジェクト・プロダクトの特性によって無限に出てくる (はず)

- mongodb 使いたい
- mecab 使いたい
- 頻度は平日の朝に来て欲しい
- プロジェクト構成が、トップレベルに Ruby や Rails 来てない
- 複数 Ruby プロジェクトが入っている
- Node.js npm もやりたい

これは、travis-ci, CircleCI を再発明することになるのでは??

CI native のアプリケーションへ

CI 上で動けばいいじゃん

先回り では、**bundle update as a service** は不要?

では、tachikoma.io や deppbot は不要なのか? tachikoma.io 止めちゃうの?

やっぱり、スクリプト書くのめんどい... 設定メンドイな...わかってる自分でやるのもメンドイぐらいなので、他の人はたぶんどきないなやっぱtachikoma.io 便利だな

自分には必要!

結論

bin/bundle-update.sh あるいは bin/bundle-update.rb を [Cron for GitHub](#) あるいは [AWS Lambda Scheduled Event](#) で kick する。

bin/bundle-update.sh

```
#!/usr/bin/env bash
```

```
set -ev
```

```
# only sunday
```

```
if [[ -n "${TRAVIS_PULL_REQUEST}" && "${TRAVIS_PULL_REQUEST}" == "false" && "${TRAVIS_BRANCH}" == "master" ]]
```

```

# gem prepare
gem install --no-document git_httpsable-push pull_request-create

# git prepare
git config user.name sanemat
git config user.email foo@example.com
HEAD_DATE=$(date +%Y%m%d_%H-%M-%S)
HEAD="tachikoma/update-`${HEAD_DATE}`"

# checkout (for TravisCI)
git checkout -b "${HEAD}" "${TRAVIS_BRANCH}"

# bundle install
bundle --no-deployment --without nothing --jobs 4

# bundle update
bundle update

git add Gemfile.lock
git commit -m "Bundle update `${HEAD_DATE}`"

# git push
git httpsable-push origin "${HEAD}"

# pull request
pull-request-create
fi

exit 0

bin/bundle-update.rb bundle-udpate/Gemfile

bundle-udpate/bin/bundle-update.rb

cd bundle-update && bundle exec bin/bundle-update.rb && cd ..

```

Example [sanemat/ruby-example-rails-banana](#)

compare-linker

web サービスなら同時にやってくれる方がいいかもだけど、ライブラリなら tachikoma に内蔵するより、別コマンドの方が良い あと、github api 依存の部分を外した

アプリの依存に載せたくないが、.lock は取りたい、場合なども `bundle install --gemfile=Gemfile.bundle-update`

にしておけばいいんじゃないですかね (試してない) なお、そうするとこっちの依存部分についても、定期的に `bundle update` できる。(ように自分で書けば良い)

`compare-linker-wrapper` で出力して、`saddler` で pull request のコメントに載せる

課題定期的にライブラリの依存関係をアップデートして Pull Request する – [Saddler - checkstyle to anywhere](#)

`gem install` がコケて定期実行に失敗することがあるらしい `travis_retry gem install xxx` とすればよいのでは (travis なら)

tachikoma gem は発展的解消できたなでも、コレ設定メンドイな...わかってる自分でやるのもメンドイぐらいなので、他の人はできないな やっぱ、tachikoma.io 便利だな

shell script なら言語中立にいけるやん! パイプ最高や! 期にノッて書いたライブラリ 群

あれ、shell script って環境依存激しいし微妙に方言が有る windows? なにそれうまいの? ユーザー多いし、どうせなら多くの人に使ってもらわないと。

このツールチェーンでは、C extension を使わない。nokogiri, rugged を避ける標準モジュールを使う nokogiri ではなく rexml ビルドするの大変になるから rugged 使わなかったけど、それでは windows で動かなそう、などなど

golang で書き換えたいから golang の勉強はじめようかな うーん n 度目の golang やりたい期 golang だとインストールが curl とかになるのでそれはそれでどうなのよ