

.deb から Python パッケージへの変遷

まえだこうへい

2015年4月18日

自己紹介 (2014年4月の時)

Debian: blockdiag シリーズ、django REST framework、
yrmcds などのメンテナ

言語: 主に Python。
最近 Golang も使おうと勉強中

家庭: 猫一匹 (♀)、娘二人、妻一人

勉強会: ほぼ Debian 勉強会のみ。でも今日は約半年ぶり

仕事: Python で LDAP ごによごによとか、
django REST framework で API 開発とか

自己紹介 (今回)

Debian: blockdiag シリーズ、django REST framework、
yrmcds などのメンテナ

言語: 仕事は Python。Private は Golang。
最近 C++勉強始めました

家庭: 猫一匹 (♀)、娘二人、妻一人

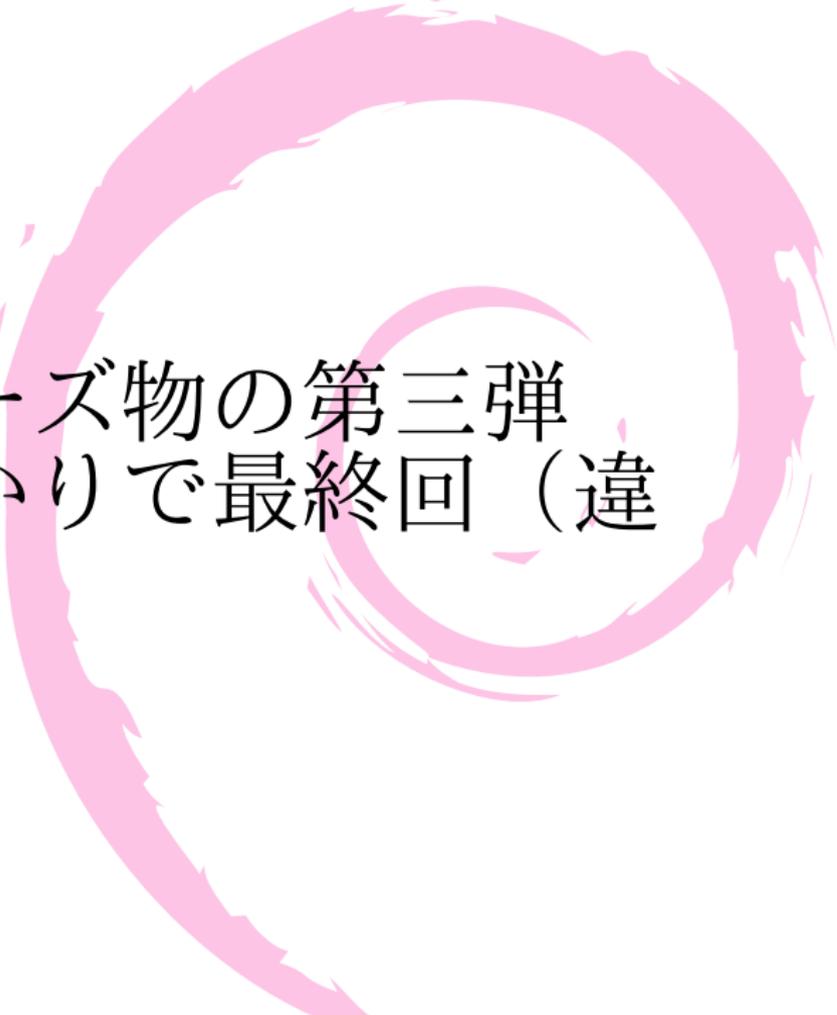
勉強会: ほぼ Debian 勉強会のみ。でも今回も約 9ヶ月ぶり¹

仕事: Private Cloud の Controller(IaaS)
のメンテを引き継ぎ中

¹前回の時はプレゼン資料作り忘れしました

今日の話の流れ

- 今回のお話の背景
- ローカル Debian CI のおさらい
- ローカル Python CI (New!)
- before / after
- Python パッケージは Debian パッケージにすべきか？

A large, stylized pink brushstroke graphic that forms a circular shape with a smaller circle inside, resembling a swirl or a spiral. The brushstroke has a textured, hand-painted appearance with varying shades of pink and some white highlights.

シリーズ物の第三弾
三年がかりで最終回（違

開発、Python と Django に統一するってよ

え、今頃…？(2014年9月現在)²

- プロジェクトの開始当初から2年半
- Python 以外で Java とか Golang で開発したユーティリティも増えました
- Python の Web フレームワークも開発初期は Django 以外に Flask も使っていました
- 最初のリリース後増員された人員も、減っていく一方

²本日現在諸事情により更に OpenStack 導入に方針が変わっています

組織人員の構成

属性	おおよその比率
プログラミングできない人	6割 ³
Java や Ruby では開発できる	3割
Django を使ったことがない	8割
django REST framework を使ったことが無い	9割
Python でユニットテスト書いたことが無い	私以外全員
Python パッケージを作ったことがない	私以外全員

なんらかの方法で下駄を履かせないといけませんでした

³うち Python お勉強中の人は5割

下駄を履かせるために行ったこと

- 開発標準の方針の文書化
- Python パッケージ用テンプレートの作成
文法チェック/ユニットテスト/カバレッジ計測を強制実行, ドキュメント自動生成, Python パッケージ作成, Git リポジトリ作成・設定など
- Django 用 Python パッケージのテンプレート作成
各種ベストプラクティス適用, django REST framework の適用, サンプルアプリの提供など
- 独自認証システム⁴用のライブラリ開発
本番環境と開発用のモックモードの切替機能, Django 認証モジュールの開発, 前述のテンプレートへの組み込みなど
- Python パッケージ用のローカルリポジトリ (ローカル PyPI) の構築
- Jenkins での関連システムの統合 (ローカル PyPI CI)
GitHub:e, テスト自動実行, HipChat への通知, ローカル PyPI へのアップロードなど
- 説明会、アップデートの通知、個別フォロー

⁴この時の開発スコープの一コンポーネント

その前におさらい。ローカル Debian CI

Jenkins で、ソースパッケージからビルド、テスト、ローカルアーカイブにアップロードする仕組み

ローカル PyPI CI

Jenkins で、Git リポジトリからユニットテスト、ローカル PyPI にアップロードする仕組み

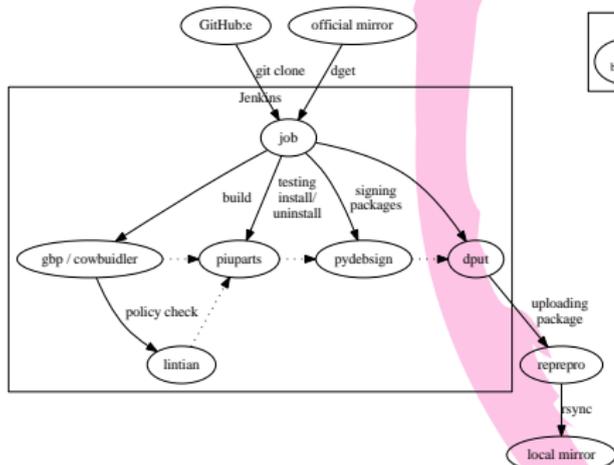
比較

機能的にほぼ同じ+ α

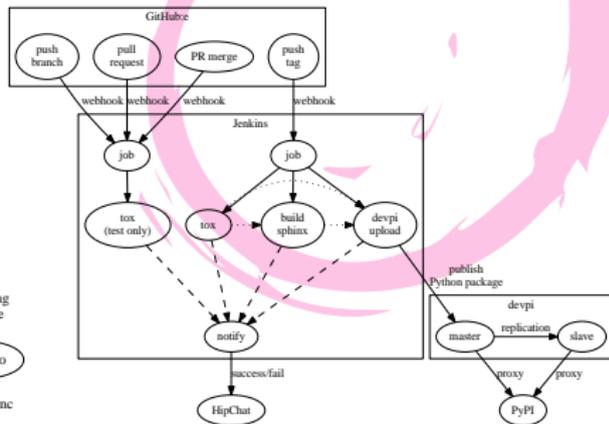
機能	ローカル Debian CI	ローカル PyPI CI
ミラー	apt-mirror	bandersnatch
リポジトリ	GitHub:e	GitHub:e or 公式ミラーなど
ジョブスクリプト	Python で実装	Python で実装
クリーン (ビルド) 環境	cowbuilder	virtualenv
テスト	lintian, piuparts	tox
署名	pydebsign	n/a
アップロード	dput	devpi client
ローカルアーカイブ/リポジトリ	reprepro	devpi server
パッケージテンプレート	n/a	シェルスクリプトで実装
Git push /pull request hook	n/a	GitHub:e webhook
テスト結果通知	n/a	HipChat

比較

ローカル Debian CI



ローカル PyPI CI

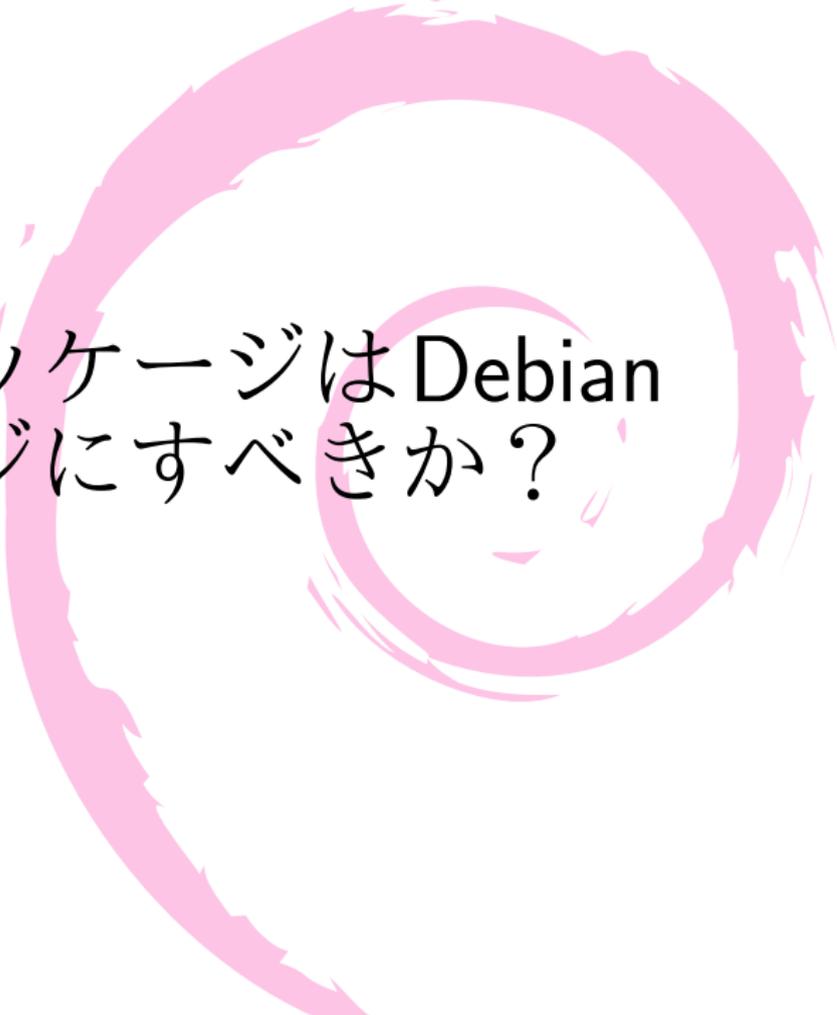


.deb から Python パッケージへの Before/After

Debian のカスタムビルドパッケージとの比較の話。

- パッケージ化を行えるメンバーの増加
- 開発時のリポジトリ切替の簡易化
- パッケージ化のコスト削減

※開発標準化の before/after の話ではありません



Python パッケージは Debian
パッケージにすべきか？

したほうが良い

- システムグローバルでインストールするツール⁵
- PyPI で公開されていない C binding のライブラリ

⁵sudo pip install とするケース

しても良い

- コマンドラインツール
- デーモン
- PyPI で公開されている C binding のライブラリ

しなくても良い

- Debian パッケージではない Python パッケージに依存されないパッケージ

すると辛い

- 開発が活発で、コマンドラインやデーモンにも依存されないライブラリ

ケースバイケース

