How to Get Started with Python and GitLab CI

Hello!

I’m Mario García

GitLab Hero

Twitter: @mariogmd

1.
Python + Rust

Web Apps with Python and Rust

Building apps with both technologies

- Using Rust for writing native Python modules
- Running Python code from a Rust binary

Rust crates available

- CPython
- PyO3

2.
Local Dev Environment

Install Rust and Python

Installing Python

- ▷ Download from **python.org/downloads**
- ▷ From the repositories (Linux)
- ▷ Using pyenv
  - ○ Build Python with --enable-shared → **github.com/pyenv/pyenv/wiki**

Install Rust and Python

Installing Rust

- Using rustup → **rustup.rs**
  - rustup install stable/beta/nightly

Using Docker

- Custom Docker image
  - Python, Rust and development tools
    - **gitlab.com/mattdark/docker-rust-python**

3.
Heroku

Python Support

- Official buildpack for Python
  - Dependencies
    - requirements.txt
    - Pipfile
  - Startup configuration
    - Procfile
  - Python runtime

      runtime.txt
- Buildpack with support for Poetry

Rust Support

- Community buildpack for Rust
  - Dependencies
    - Cargo.toml
  - Startup configuration
    - Procfile
  - Rust toolchain

      RustConfig

4.
Code

- A web app built with Rust that has access to a Firebase database
  - Using Python library to connect to Firebase

gitlab.com/mattdark/rust-python-demo

5.
CI for Python & Rust

CI Configuration - Python

- Procfile
  - web: gunicorn app:app
- Dependencies
  - pyproject.toml
  - requirements.txt
- runtime.txt
  - python-3.8.6
- .gitlab-ci.yml

gitlab.com/mattdark/python-blog

CI Configuration - Rust

- Procfile
  - web: ROCKET_PORT=$PORT ROCKET_ENV=prod ./target/release/app
- Dependencies
  - Cargo.toml
- RustConfig
  - VERSION=stable/beta/nightly
- Rocket.toml
- .gitlab-ci.yml

gitlab.com/mattdark/python-blog

6.
Continuous Integration

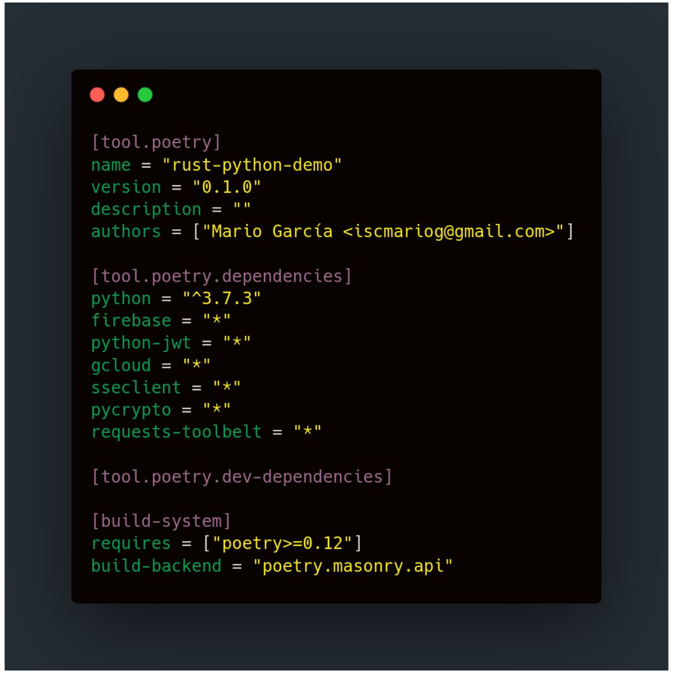
Repository - Configuration

- Bash script (Procfile)
  - ROCKET_PORT=$PORT ROCKET_ENV=prod ./target/release/rust-python-demo
- Dependencies
  - Cargo.toml
  - pyproject.toml
- Dockerfile

Repository - Configuration

pyproject.toml

- ▷ Specify Python runtime
- ▷ Firebase library and its dependencies

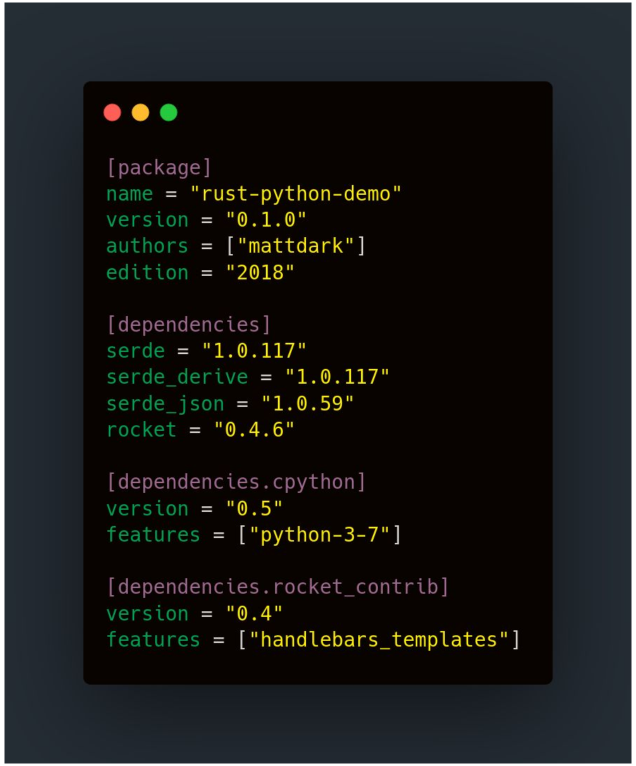


```
[tool.poetry]
name = "rust-python-demo"
version = "0.1.0"
description = ""
authors = ["Mario García <iscmariog@gmail.com>"]

[tool.poetry.dependencies]
python = "^3.7.3"
firebase = "*"
python-jwt = "*"
gcloud = "*"
sseclient = "*"
pycrypto = "*"
requests-toolbelt = "*"

[tool.poetry.dev-dependencies]

[build-system]
requires = ["poetry>=0.12"]
build-backend = "poetry.masonry.api"
```

Repository - Configuration

Cargo.toml

- Serde
- Rocket
- CPython



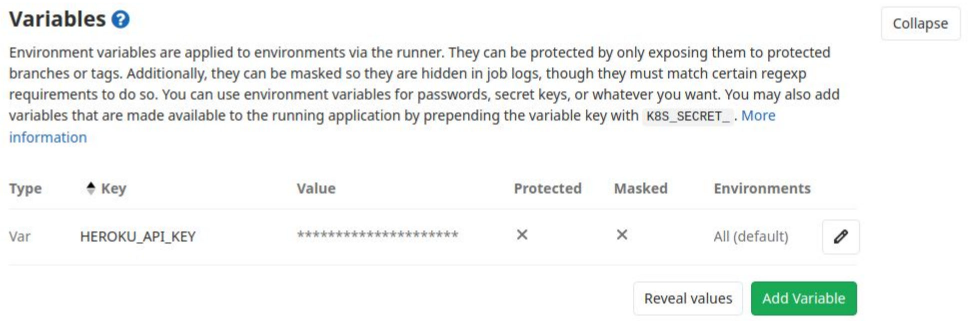
```
[package]
name = "rust-python-demo"
version = "0.1.0"
authors = ["mattdark"]
edition = "2018"

[dependencies]
serde = "1.0.117"
serde_derive = "1.0.117"
serde_json = "1.0.59"
rocket = "0.4.6"

[dependencies.cpython]
version = "0.5"
features = ["python-3-7"]

[dependencies.rocket_contrib]
version = "0.4"
features = ["handlebars_templates"]
```

Heroku - Configuration

- Create new app
- ~~Add Python buildpack~~
- ~~Add Rust buildpack~~
- Go to dashboard.heroku.com/account
  - Copy API Key

GitLab CI - Configuration

- Configure GitLab CI
  - Go to Settings → CI/CD
  - Click Expand in the Variables section
  - Add HEROKU_API_KEY variable and paste the API Key in the Value field

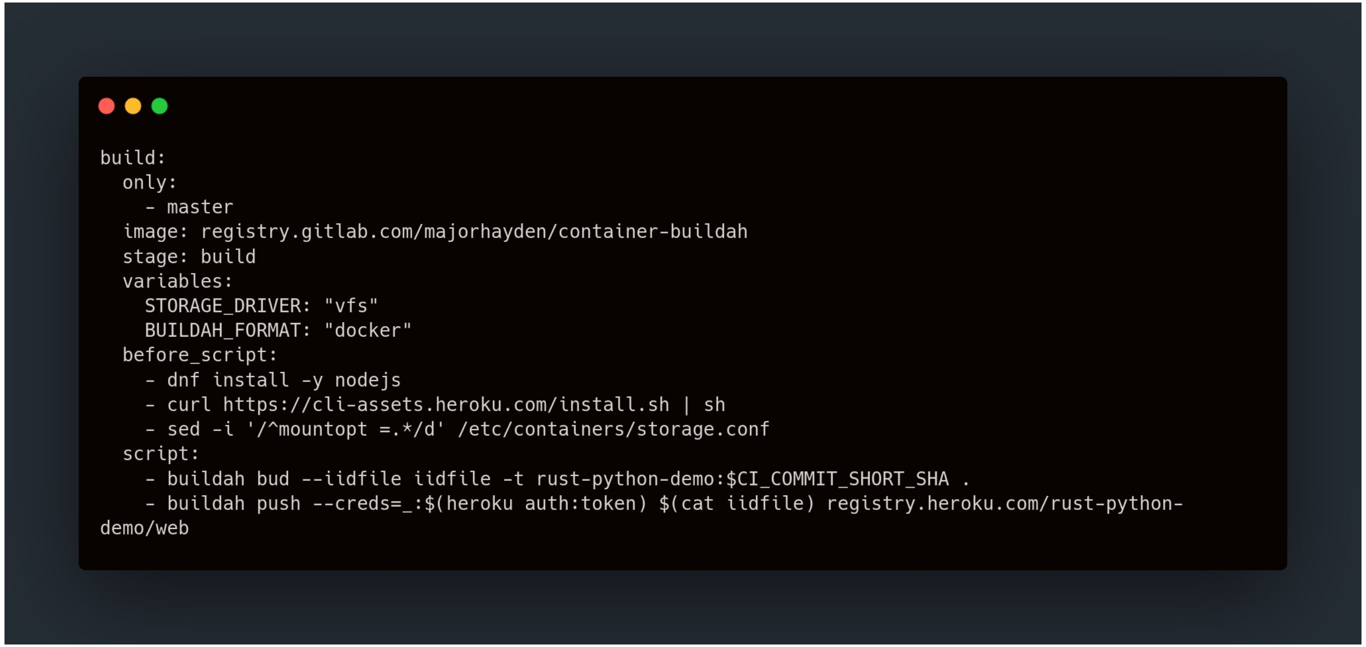


Environment variables are applied to environments via the runner. They can be protected by only exposing them to protected branches or tags. Additionally, they can be masked so they are hidden in job logs, though they must match certain regexp requirements to do so. You can use environment variables for passwords, secret keys, or whatever you want. You may also add variables that are made available to the running application by prepending the variable key with `K8S_SECRET_`. More information

Type	Key	Value	Protected	Masked	Environments
Var	HEROKU_API_KEY	********************	✕	✕	All (default)

GitLab CI - Configuration

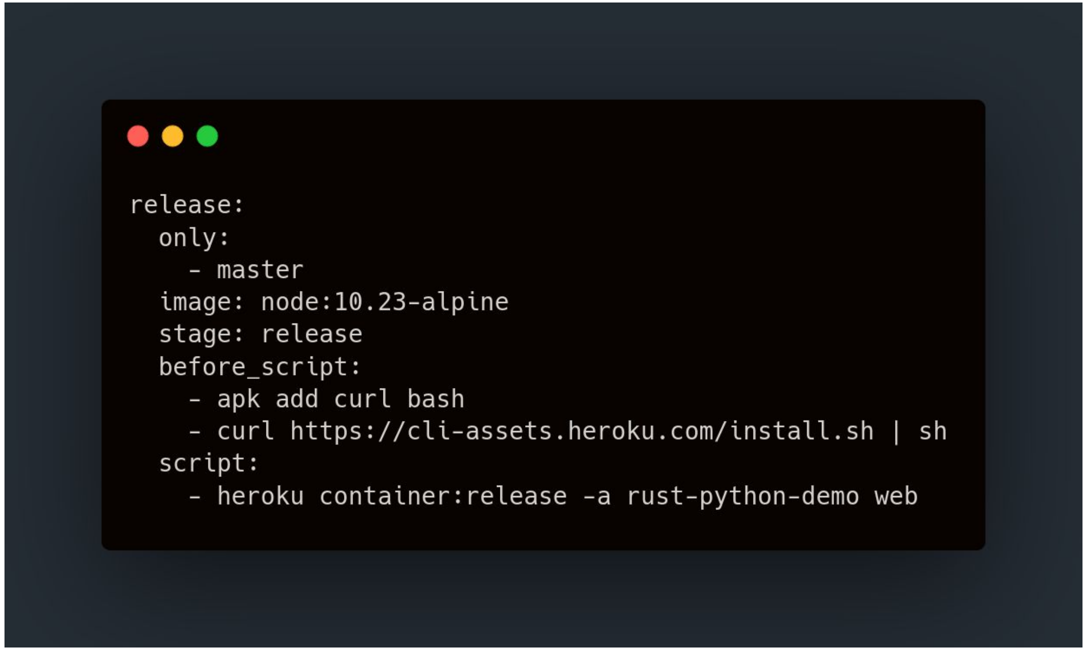
- Create '.gitlab-ci.yml'



```
build:
  only:
    - master
  image: registry.gitlab.com/majorhayden/container-buildah
  stage: build
  variables:
    STORAGE_DRIVER: "vfs"
    BUILDAH_FORMAT: "docker"
  before_script:
    - dnf install -y nodejs
    - curl https://cli-assets.heroku.com/install.sh | sh
    - sed -i '/^mountopt =.*/d' /etc/containers/storage.conf
  script:
    - buildah bud --iidfile iidfile -t rust-python-demo:$CI_COMMIT_SHORT_SHA .
    - buildah push --creds=_:$(heroku auth:token) $(cat iidfile) registry.heroku.com/rust-python-
demo/web
```

GitLab CI - Configuration

- Create '.gitlab-ci.yml'



```
release:
  only:
    - master
  image: node:10.23-alpine
  stage: release
  before_script:
    - apk add curl bash
    - curl https://cli-assets.heroku.com/install.sh | sh
  script:
    - heroku container:release -a rust-python-demo web
```

Thanks!

Questions?

@mariogmd

dev.to/mattdark

gitlab.com/mattdark/rust-python-demo

Credits

Special thanks to all the people who made and released these awesome resources for free:

- ▷ Presentation template by SlidesCarnival
- ▷ Photographs by Unsplash