
Outpak Documentation

Release 1.0.0

Chris Maillefaud

Feb 02, 2018

Contents:

1	What is Outpak?	3
1.1	Tutorial	4
1.2	Pak.yml Reference	5
2	Indices and tables	9

This document will guide you how to install, configure and use [Outpak](#) in your projects.

CHAPTER 1

What is Outpak?

Outpak is a tool for installing packages inside `requirements.txt` using [Git Personal Tokens](#) or [Bitbucket App Passwords](#), instead of using *SSH keys*. This is specially important on [Docker](#) projects, if you don't want to copy the *SSH keys* inside the containers.

For example, if you have on `requirements.txt` the following lines:

```
-e git+git@git.myproject.org:MyProject#egg=MyProject
-e git://git.myproject.org/MyProject.git@da39a3ee5e6b4b0d3255bfef95601890afd80709
↪#egg=MyProject
```

Outpak will:

1. Parse the urls:

```
from: git+git@git.myproject.org:MyProject or git://git.myproject.org/MyProject.git
to: https://git.myproject.org/myproject
```

2. Clone the repositories using the token/password and directory informed in `pak.yml` file:

```
$ git clone https://my_git_token@git.myproject.org/myproject /tmp/myproject
```

3. Run `git reset` to correct commit if informed:

```
$ cd /tmp/myproject && git reset --hard da39a3ee5e6b4b0d3255bfef95601890afd80709
```

4. And installing package using the `“pip install -e .“` command:

```
$ cd /tmp/myproject && pip install -e .
```

Note: [Outpak](#) are tested for Bitbucket and Github services. For other DVCS services please check our [issues page](#) on github.

1.1 Tutorial

This document will explain how to install [Outpak](#) and use it in your projects.

1.1.1 Installing Outpak

First, install Outpak using the command:

```
$ pip install outpak
```

1.1.2 Creating the pak.yml file

For a simple example, let's consider the following environment for your project, loaded in the `.bashrc` file:

```
$ export MY_ENVIRONMENT="docker"
$ export MY_GIT_TOKEN="12345abcde"
```

Based on these values, we can create the `pak.yml` configuration file:

```
version: "1"
github_key: MY_GIT_TOKEN
env_key: MY_ENVIRONMENT
envs:
  Docker:
    key_value: docker
    clone_dir: /opt/src
    files:
      - requirements.txt
      - requirements_test.txt
```

Note: Save the `pak.yml` on the same directory where the `requirements.txt` files are located.

The `github_key`

The `github_key` points to the environment variable you use to store your [Git Personal Token](#). (For [Bitbucket App Password](#), use the key `bitbucket_key`). On our example is the `MY_GIT_TOKEN` env.

The `env_key`

The `env_key` points to the environment variable which you use to indicate what is the project current *working environment* (*development, stage, etc...*). In our example is the `MY_ENVIRONMENT` env.

The `envs` key

The `envs` list can hold one entry per possible value the `MY_ENVIRONMENT` (the `env_key`) holds. In our example, `MY_ENVIRONMENT` was set to “docker”, so we need a “Docker” entry in this key:

- The `key_value` must be the same value stored in the `MY_ENVIRONMENT` var: in our example “docker”
- The full path for cloning projects will be `/opt/src` as indicated in `clone_dir` key.

- The list of files which will be processed are: `requirements.txt` and `requirements_test.txt` as indicated in key files.

Note: Check the [Pak.yml Reference](#) page to the complete reference for `pak.yml` files.

1.1.3 Running Outpak

After create the configuration file, you can start install packages with the command:

```
$ pak install --config /path/to/pak/file
```

If you do not inform the path for the `pak.yml` file, Outpak will attempt to find it in the current directory.

Note: Also you can set the `OUTPAK_FILE` environment variable for where the `pak.yml` file is located.

1.2 Pak.yml Reference

This is the reference documentation for the `pak.yml` file.

1.2.1 Reference List

- *bitbucket_key*
- *clone_dir*
- *env_key*
- *envs*
- *files*
- *github_key*
- *key_value*
- *token_key*
- *use_virtual*
- *version*

bitbucket_key

Set the environment variable which holds your Bitbucket App Password

```
bitbucket_key: MY_BITBUCKET_APP_PASSWORD
```

In the environment key you set the app password: `MY_BITBUCKET_APP_PASSWORD="username:password"`

Note: The format for the bitbucket app password in the environment key must be: `username:password`.

clone_dir

Set the base path where the projects will be cloned:

```
envs:  
  virtualenv:  
    clone_dir: /tmp
```

Outpak will generate a full path for each project, using the base path provided and the project name found in url:

For example, if url is `git+git@git.myproject.org:MyProject` and `clone_dir` is `/tmp` the cloning path will be `/tmp/myproject`.

You need to inform a full path, do not use relative paths.

Note: Make sure the current user can be the right permissions to save in this directory.

env_key

Set the environment variable which control your Project environment.

```
env_key: MY_ENVIRONMENT_KEY
```

envs

Returns a list of possible values for the environment key defined `env_key`:

```
env_key: MY_ENVIRONMENT_KEY  
envs:  
  Virtualenv:  
    key_value: development  
  Docker:  
    key_value: docker  
  Staging:  
    key_value: stage  
  Production:  
    key_value: prod
```

At least one environment must be set.

Note: Make sure you have create entries for all possible values for your environment key.

files

Returns a list of `requirements.txt` files must be processed for each environment defined:

```
env_key: MY_ENVIRONMENT_KEY  
envs:  
  Dev:  
    key_value: development  
    files:  
      - requirements.txt
```

```

    - requirements_test.txt
  Prod:
    key_value: prod
    files:
      - requirements.txt

```

github_key

Set the environment variable which holds your [Git Personal Token](#)

```
github_key: MY_GIT_PERSONAL_TOKEN
```

key_value

OutPak will get value found inside the environment variable you define in `env_key` to find the correct env to process the `requirement.txt` files.

```

env_key: MY_ENVIRONMENT_KEY
envs:
  Virtualenv:
    key_value: development
    clone_dir: /tmp
  Docker:
    key_value: docker
    clone_dir: /opt/src
  Staging:
    key_value: stage
    clone_dir: /opt/src
  Production:
    key_value: prod
    clone_dir: /opt/src

```

For example, if the env `MY_ENVIRONMENT_KEY="development"`, then Outpak will use the `/tmp` as base path for cloning projects.

token_key

Same as *github_key*.

Note: This key is deprecated and will be removed in next version.

use_virtual

Set if Outpak need to check if a virtualenv was activated, before start processing the `requirement.txt` files:

```

version: "1"
env_key: MY_ENVIRONMENT_KEY
envs:
  Prod:
    key_value: production
    clone_dir: /opt/src

```

```
Dev:  
  key_value: development  
  use_virtual: true  
  clone_dir: /tmp
```

version

Set the version for this file. Current version is: "1"

```
version: "1"
```

CHAPTER 2

Indices and tables

- `genindex`
- `modindex`
- `search`