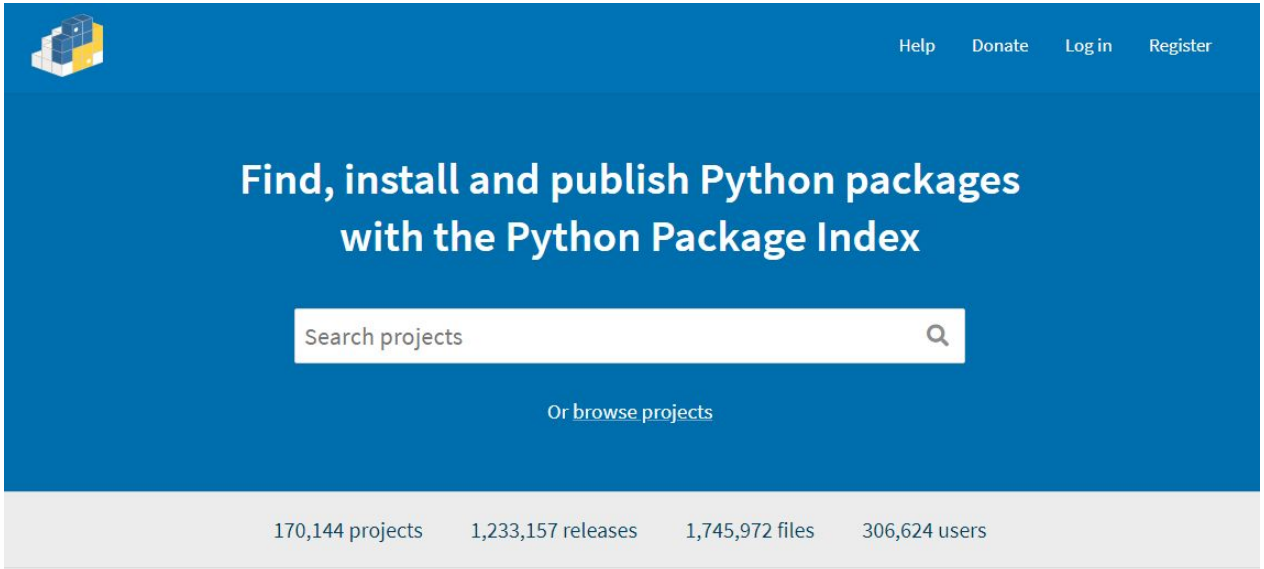
Wirtualne środowisko

PyPI, venv

Adam Chyła
chyla.org

PyPI

The Python Package Index (PyPI) to repozytorium pakietów (bibliotek i programów) dla języka Python dostępne pod adresem https://pypi.org/.



PyPI

Przykładowe narzędzia dostępne w repozytorium:

- pycodestyle - https://pypi.org/project/pycodestyle/,
- pytest - https://pypi.org/project/pytest/,
- django - https://pypi.org/project/Django/,
- flask - https://pypi.org/project/Flask/.

Częste pytanie: *Jak dodać własną bibliotekę/program do repozytorium?*

Instrukcja: https://packaging.python.org/tutorials/packaging-projects/

PyPI

Zarządzanie pakietami (wyszukiwanie, instalacja, aktualizacja, dezinstalacja, wypisanie listy zainstalowanych pakietów) odbywa się za pomocą, uruchamianego z konsoli, programu **`pip`**.

Najważniejsze polecenie:

`pip`

Wyświetla dostępne polecenia i ich opis.

PyPI

Zarządzanie pakietami (wyszukiwanie, instalacja, aktualizacja, dezinstalacja, wypisanie listy zainstalowanych pakietów) odbywa się za pomocą, uruchamianego z konsoli, programu **`pip`**.

Przykłady:

Wyszukiwanie pakietów powiązanych z biblioteką pytest:

```
pip search pytest
```

PyPI

Zarządzanie pakietami (wyszukiwanie, instalacja, aktualizacja, dezinstalacja, wypisanie listy zainstalowanych pakietów) odbywa się za pomocą, uruchamianego z konsoli, programu **`pip`**.

Przykłady:

Instalacja pakietu:

```
pip install pytest
```

PyPI

Zarządzanie pakietami (wyszukiwanie, instalacja, aktualizacja, dezinstalacja, wypisanie listy zainstalowanych pakietów) odbywa się za pomocą, uruchamianego z konsoli, programu **`pip`**.

Przykłady:

Instalacja konkretnej wersji pakietu:

```
pip install pytest==4.3.0
```

PyPI

Zarządzanie pakietami (wyszukiwanie, instalacja, aktualizacja, dezinstalacja, wypisanie listy zainstalowanych pakietów) odbywa się za pomocą, uruchamianego z konsoli, programu **`pip`**.

Przykłady:

Aktualizacja pakietu do najnowszej wersji:

```
pip install --upgrade pytest
```

PyPI

Zarządzanie pakietami (wyszukiwanie, instalacja, aktualizacja, dezinstalacja, wypisanie listy zainstalowanych pakietów) odbywa się za pomocą, uruchamianego z konsoli, programu **`pip`**.

Przykłady:

Dezinstalacja pakietu:

```
pip uninstall pytest
```

PyPI

Zarządzanie pakietami (wyszukiwanie, instalacja, aktualizacja, dezinstalacja, wypisanie listy zainstalowanych pakietów) odbywa się za pomocą, uruchamianego z konsoli, programu **`pip`**.

Przykłady:

Wypisanie listy zainstalowanych pakietów:

```
pip list
```

PyPI

Zarządzanie pakietami (wyszukiwanie, instalacja, aktualizacja, dezinstalacja, wypisanie listy zainstalowanych pakietów) odbywa się za pomocą, uruchamianego z konsoli, programu **`pip`**.

Więcej informacji na temat zarządzania pakietami:
https://packaging.python.org/tutorials/installing-packages/

PyPI

Problemy?

PyPI

Problemy?

Przykładowy problem:

Którą wersję biblioteki zainstalować, jeśli chcę korzystać z dwóch programów w tej samej chwili, a korzystają one z tych samych bibliotek, ale w różnych wersjach? Czy różne wersje tej samej biblioteki będą ze sobą współpracować?

Program A	**Program B**
Wymaga Django w wersji **2.1**.	Wymaga Django w wersji **1.11** (LTS).

Wirtualne środowisko

venv

Wirtualne środowisko pozwala zarządzać pakietami bez ingerencji w systemową instalację Pythona.

Moduł venv odpowiedzialny za tworzenie wirtualnego środowiska dostarczany jest wraz z domyślną instalacją Pythona 3.7. W niektórych dystrybucjach Linuksa (np. Debian/Ubuntu) może być konieczne doinstalowanie dodatkowego pakietu zawierającego moduł venv.

venv

Utworzenie wirtualnego środowiska jest możliwe za pomocą polecenia `python -m venv`, argumentem do modułu jest nazwa środowiska (w poniższym przykładzie webenv):

```
python -m venv webenv
```

venv

W przypadku powłoki BASH, **do aktywacji środowiska służy polecenie source**. Parametrem jest ścieżka do skryptu, który odpowiednio zmienia zmienne środowiskowe (np. PATH, PS1):

```
source webenv/bin/activate
```

W przypadku powłoki systemu Windows **aktywację środowiska przeprowadza się poprzez uruchomienie skryptu activate**.

```
webenv\Scripts\activate
```

venv

Do **dezaktywacji środowiska** służy polecenie:

```
deactivate
```

venv

Po aktywacji możemy używać poleceń związanych z pythonem. Przyładowo, instalację Django w wersji 1.8.13 wykonujemy poleceniem:

```
pip install Django==1.8.13
```

venv

Innymi komendami, które warto znać, są:

```
pip freeze > nazwa_pliku
```

```
pip install -r nazwa_pliku
```

Pierwsze polecenie polecenie tworzy plik, którego zawartością jest lista zainstalowanych w środowisku pakietów, wraz z ich wersjami. Ogólnie przyjęta nazwa pliku to **`requirements.txt`**. Tworząc nowe, czyste środowisko możemy wczytać taką listę i zainstalować pakiety w niej zawarte (drugie polecenie).

Pytania

1. Jakie znasz polecenia programu PIP?
2. Jakie znasz cele stosowania wirtualnego środowiska?
3. Jak aktywować/dezaktywować wirtualne środowisko?
4. Jaką nazwę nosi plik zawierający listę wymaganych pakietów? W jaki sposób zainstalować wymienione w pliku pakiety?

Literatura

1. Obsługa polecenia PIP i informacje o wirtualnych środowiskach, https://packaging.python.org/tutorials/installing-packages/
2. Opis innych narzędzi do zarządania wirtualnym środowiskiem, https://docs.python-guide.org/dev/virtualenvs/

Tekst podlega ochronie na podstawie ustawy z dnia 4 lutego 1994 roku o prawie autorskim i prawach pokrewnych (tekst jednolity Dz.U. z 2000 r. Nr 80, poz. 904 z późn. zm.). Jakiekolwiek kopiowanie, odtwarzanie, przegrywanie, zwielokrotnianie zawartości jest zabronione. Kopiowanie treści możliwe jest tylko po udzieleniu zgody przez autora.