Структура Python проекта

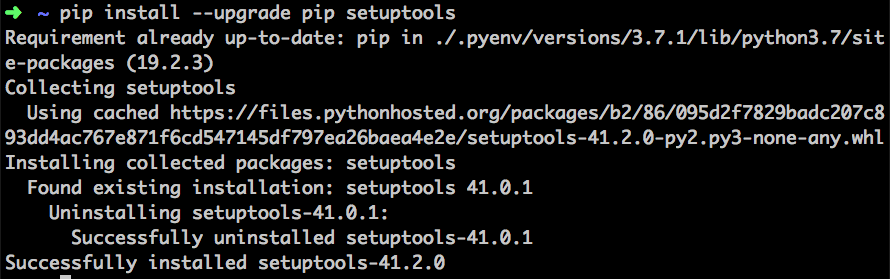
# Настройка виртуального окружения для проекта

## Виртуальное окружение

Поскольку в Вашей системе уже есть python и используется некоторыми службами (например, графической оболочкой), то ставить на системный питон какие-то пакеты, нужные нам для задач - не очень, т.к. может произойти конфликт версий и что-нибудь в системе не будет работать корректно (личный опыт - для запуска модели на моём ноутбуке мне требовалась более новая версия scikit-image, но после установки у меня перестали работать настройки экрана, пришлось откатываться). Для решения этой проблемы было придумано виртуальное окружение - выбранная нами версия python и требующиеся для проекта библиотеки теперь лежат в проекте, и в каждом они свои.

1. Открываем папку с проектом, открываем в ней консоль
2. Прописываем pyenv local 3.7.1 (можно любую другую понравившуюся версию, главное, чтобы 3.7)

Так мы указали, что в этой папке мы используем строго эту версию

1. Далее создаём виртуальное окружение командой python -m venv venv (второе venv это название папки с окружением)
2. Подключаемся к окружению командой source venv/bin/activate
3. Там по умолчанию будет pip. Вызываем pip install --upgrade pip setuptools и видим следующее:

Обновили инструменты для установки библиотек.

1. Далее нужно установить в наше виртуальное окружение нужные библиотеки. Варианта 2:
2. pip install перечисление всех библиотек через пробел
3. pip install -r requirements.txt

С первым всё понятно, и это не удобно.

## requirements.txt

* requirements.txt - файл с перечисленными через перевод строки требующимися библиотеками (иногда с указанием версий)
* должен лежать в репозитории
* позволяет одной командой установить все нужные зависимости

Заполнять этот файл нужно вручную в соответствии с тем, какие библиотеки вы устанавливали в процессе создания проекта.

Пример наполнения:

numpy

pandas

scikit-learn

scipy

scikit-image

vk\_api

!ВАЖНО!

requirements.txt не всегда должен совпадать с pip list, закидывайте только то, что нужно непосредственно Вам (pip list отражает всё установленное, некоторые библиотеки оттуда нужны для работы нужных нам библиотек или для обеспечения безопасности системы, мы к ним напрямую обращаться не будем, соответственно и указывать их не надо)

# Git

## .gitignore

Можно сгенерировать с помощью <https://www.gitignore.io/>

* также должен лежать в репозитории
* содержит только то, что нам не нужно в репозитории, но нужно в проекте
* то, что не нужно в репозитории и не важно в проекте из папки проекта убираем

Например,

.idea

venv

\*.pyc

\_\_pycache\_\_/

### Linux ###

\*~

.Trash-\*

.nfs\*

### чек-лист:

* в репозитории остаются:
  + исходники
  + тесты (если есть)
  + конфиги
  + служебные файлы типа .gitignore, .python-version и т.п.
* пишутся в .gitignore:
  + .idea
  + venv
  + прочие файлы связанные с IDE, т.к. вынести из папки проекта мы не можем, а удалять и создавать каждый раз затратно
* выносятся из папки проекта наружу:
  + данные
  + сохранённые модели нейросетей(т.н. чекпоинты)

# Среда разработки

## [.editorconfig](https://editorconfig.org/)

Удобный способ пользоваться одинаковыми настройками IDE на разных машинах ничего не перенастраивая.

Также лежит в репозитории.

Пример:

root = true

[\*]

indent\_style = space

indent\_size = 4

end\_of\_line = lf

charset = utf-8

trim\_trailing\_whitespace = true

insert\_final\_newline = true

[\*.md]

trim\_trailing\_whitespace = false

[Makefile]

indent\_style = tab

[\*.{cpp,c,hpp,h}]

indent\_size = 2

## Makefile

Невероятно полезная вещь, призванная автоматизировать все предыдущие шаги для быстрого развёртывания новых проектов.

Бессмысленно описывать его здесь, когда есть замечательная [инструкция](https://makefile.site/).

Пример: (отступы табуляцией! Гугл доки там сделали пробелы. Когда будете копировать, проверьте это. Editorconfig должен автоматически поправить)

all: pyenv-activate create-venv deps

run:

. venv/bin/activate;PYTHONPATH=src python src/main.py

deps:

. venv/bin/activate;pip install --upgrade pip

. venv/bin/activate;pip install --upgrade setuptools

. venv/bin/activate;pip install -r requirements.txt

create-venv: pyenv-activate

python -m venv venv

pyenv-activate:

pyenv local # it get info from .python-version file

.PHONY: deps create-venv pyenv pyenv-activate