

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

«Установка пакетов в Python. Виртуальные окружения»

**Отчет по лабораторной работе № 2.14
по дисциплине «Программирование на Python»**

Выполнил студент группы ИВТ-б-о-21-1

Артемьев Андрей Витальевич.

«3 » ноября 2022г.

Подпись студента _____

Работа защищена « » _____ 20__г.

Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2022

Цель работы: приобретение навыков по работе с менеджером пакетов `pip` и виртуальными окружениями при написании программ с помощью языка программирования Python версии 3.x.

Порядок выполнения работы:

1. Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python.

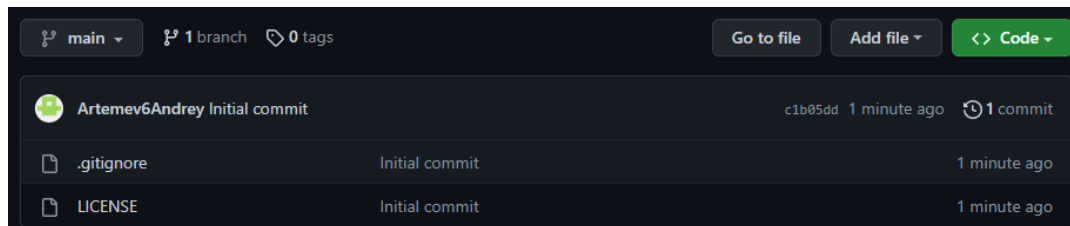


Рисунок 1 - Создание репозитория

2. Выполните клонирование созданного репозитория.

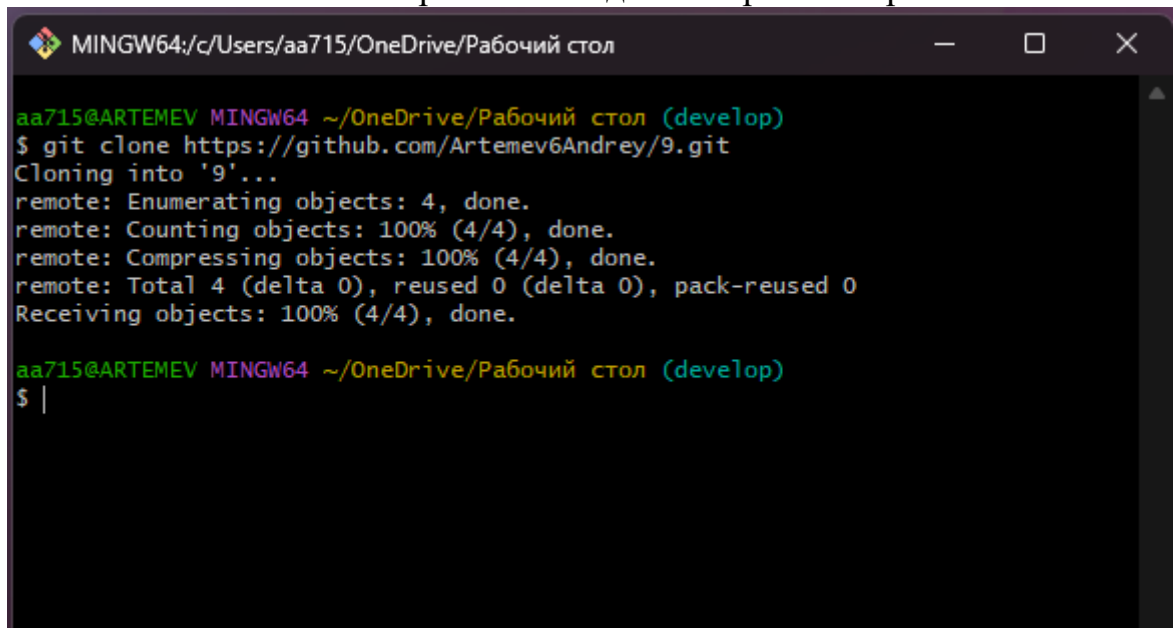


Рисунок 2 - Клонирование репозитория

3. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

```
aa715@ARTEMEV MINGW64 ~/OneDrive/Рабочий стол/9 (main)
$ git flow init

which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/] Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/aa715/OneDrive/Рабочий стол/9/.git/hooks]

aa715@ARTEMEV MINGW64 ~/OneDrive/Рабочий стол/9 (develop)
$
```

Рисунок 3 - Ветвление по модели git-flow

4. Создайте виртуальное окружение Anaconda с именем репозитория.

```
Anaconda Prompt (anaconda) x + v

(base) C:\Users\aa715>cd aa715\OneDrive\Рабочий стол\9
Системе не удается найти указанный путь.

(base) C:\Users\aa715>cd OneDrive\Рабочий стол\9

(base) C:\Users\aa715\OneDrive\Рабочий стол\9>conda create -n LL_2.14 python=3.10
Collecting package metadata (current_repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 22.9.0
  latest version: 22.11.1

Please update conda by running

  $ conda update -n base -c defaults conda

## Package Plan ##

environment location: C:\Users\aa715\.conda\envs\LL_2.14

added / updated specs:
- python=3.10

The following packages will be downloaded:
```

```

The following NEW packages will be INSTALLED:

bzip2                pkgs/main/win-64::bzip2-1.0.8-he774522_0 None
ca-certificates      pkgs/main/win-64::ca-certificates-2022.10.11-haa95532_0 None
certifi              pkgs/main/win-64::certifi-2022.9.24-py310haa95532_0 None
libffi               pkgs/main/win-64::libffi-3.4.2-hd77b12b_6 None
openssl              pkgs/main/win-64::openssl-1.1.1s-h2bbff1b_0 None
pip                  pkgs/main/win-64::pip-22.2.2-py310haa95532_0 None
python               pkgs/main/win-64::python-3.10.8-h966fe2a_1 None
setuptools           pkgs/main/win-64::setuptools-65.5.0-py310haa95532_0 None
sqlite               pkgs/main/win-64::sqlite-3.40.0-h2bbff1b_0 None
tk                   pkgs/main/win-64::tk-8.6.12-h2bbff1b_0 None
tzdata               pkgs/main/noarch::tzdata-2022g-h04d1e81_0 None
vc                   pkgs/main/win-64::vc-14.2-h21ff451_1 None
vs2015_runtime       pkgs/main/win-64::vs2015_runtime-14.27.29016-h5e58377_2 None
wheel                pkgs/main/noarch::wheel-0.37.1-pyhd3eb1b0_0 None
wincertstore         pkgs/main/win-64::wincertstore-0.2-py310haa95532_2 None
xz                   pkgs/main/win-64::xz-5.2.8-h8cc25b3_0 None
zlib                 pkgs/main/win-64::zlib-1.2.13-h8cc25b3_0 None

Proceed ([y]/n)?

```

Рисунок 4 - Создание виртуального пространства

```
(base) C:\Users\aa715\OneDrive\Рабочий стол\9>conda activate LL_2.14
(LL_2.14) C:\Users\aa715\OneDrive\Рабочий стол\9>
```

Рисунок 5 - Активация виртуального пространства

5. Установите в виртуальное окружение следующие пакеты: pip, NumPy, Pandas, SciPy.

```
(LL_2.14) C:\Users\aa715\OneDrive\Рабочий стол\9>conda install Pandas
Collecting package metadata (current_repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 22.9.0
  latest version: 22.11.1

Please update conda by running

  $ conda update -n base -c defaults conda

## Package Plan ##

  environment location: C:\Users\aa715\.conda\envs\LL_2.14

  added / updated specs:
    - pandas

The following packages will be downloaded:

  package | build | size
  -----|-----|-----
  bottleneck-1.3.5 | py310h9128911_0 | 106 KB
  numexpr-2.8.4 | py310hd213c9f_0 | 128 KB
  pandas-1.5.2 | py310h4ed8f06_0 | 10.5 MB
  pyparsing-3.0.9 | py310haa95532_0 | 154 KB
  pytz-2022.1 | py310haa95532_0 | 196 KB
  -----|-----|-----
  Total: | | 11.1 MB

The following NEW packages will be INSTALLED:

  bottleneck pkgs/main/win-64::bottleneck-1.3.5-py310h9128911_0 None
  numexpr pkgs/main/win-64::numexpr-2.8.4-py310hd213c9f_0 None
  packaging pkgs/main/noarch::packaging-21.3-pyhd3eb1b0_0 None
  pandas pkgs/main/win-64::pandas-1.5.2-py310h4ed8f06_0 None
  pyparsing pkgs/main/win-64::pyparsing-3.0.9-py310haa95532_0 None
  python-dateutil pkgs/main/noarch::python-dateutil-2.8.2-pyhd3eb1b0_0 None
  pytz pkgs/main/win-64::pytz-2022.1-py310haa95532_0 None

Proceed [y]/n)? |
```

```
(LL_2.14) C:\Users\aa715\OneDrive\Рабочий стол\9>conda install numpy
Collecting package metadata (current_repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 22.9.0
  latest version: 22.11.1

Please update conda by running

  $ conda update -n base -c defaults conda

## Package Plan ##

environment location: C:\Users\aa715\.conda\envs\LL_2.14

added / updated specs:
- numpy

The following packages will be downloaded:
```

package	build	
mkl-service-2.4.0	py310h2bbff1b_0	48 KB
mkl_fft-1.3.1	py310ha0764ea_0	136 KB
mkl_random-1.2.2	py310h4ed8f06_0	221 KB
numpy-1.23.4	py310h60c9a35_0	11 KB
numpy-base-1.23.4	py310h04254f7_0	6.0 MB
Total:		6.4 MB

```

The following NEW packages will be INSTALLED:

blas                pkgs/main/win-64::blas-1.0-mkl None
intel-openmp        pkgs/main/win-64::intel-openmp-2021.4.0-haa95532_3556 None
mkl                 pkgs/main/win-64::mkl-2021.4.0-haa95532_640 None
mkl-service         pkgs/main/win-64::mkl-service-2.4.0-py310h2bbff1b_0 None
mkl_fft             pkgs/main/win-64::mkl_fft-1.3.1-py310ha0764ea_0 None
mkl_random          pkgs/main/win-64::mkl_random-1.2.2-py310h4ed8f06_0 None
numpy               pkgs/main/win-64::numpy-1.23.4-py310h60c9a35_0 None
numpy-base         pkgs/main/win-64::numpy-base-1.23.4-py310h04254f7_0 None
six                 pkgs/main/noarch::six-1.16.0-pyhd3eb1b0_1 None

```

```
(LL_2.14) C:\Users\aa715\OneDrive\Рабочий стол\9>conda install SciPy
Collecting package metadata (current_repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 22.9.0
  latest version: 22.11.1

Please update conda by running

  $ conda update -n base -c defaults conda

## Package Plan ##

environment location: C:\Users\aa715\.conda\envs\LL_2.14

added / updated specs:
- scipy

The following packages will be downloaded:
```

package	build	
scipy-1.9.3	py310h86744a3_0	18.0 MB
Total:		18.0 MB

```

The following NEW packages will be INSTALLED:

fftw                pkgs/main/win-64::fftw-3.3.9-h2bbff1b_1 None
icc_rt              pkgs/main/win-64::icc_rt-2022.1.0-h6049295_2 None
scipy               pkgs/main/win-64::scipy-1.9.3-py310h86744a3_0 None

Proceed ([y]/n)? |

```

Рисунок 6 - Установка пакетов

6. Попробуйте установить менеджером пакетов conda пакет TensorFlow.

```
(LL_2.14) C:\Users\aa715\OneDrive\Рабочий стол\9>conda install tensorflow
Collecting package metadata (current_repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 22.9.0
  latest version: 22.11.1

Please update conda by running

  $ conda update -n base -c defaults conda

## Package Plan ##

environment location: C:\Users\aa715\.conda\envs\LL_2.14

added / updated specs:
- tensorflow

The following packages will be downloaded:
```

package	build	
_tfflow_select-2.3.0	mkl	3 KB
abseil-cpp-20211102.0	hd77b12b_0	1.7 MB
absl-py-1.3.0	py310haa95532_0	172 KB
aiohttp-3.8.3	py310h2bbff1b_0	418 KB
aiohttp-3.8.3	pyhd3eb1b0_0	12 KB
astunparse-1.6.3	py_0	17 KB
async-timeout-4.0.2	py310haa95532_0	12 KB
attrs-22.1.0	py310haa95532_0	85 KB
blinker-1.4	py310haa95532_0	22 KB
brotlipy-0.7.0	py310h2bbff1b_1002	335 KB
cachetools-4.2.2	pyhd3eb1b0_0	13 KB
cffi-1.15.1	py310h2bbff1b_3	239 KB
click-8.0.4	py310haa95532_0	157 KB
colorama-0.4.5	py310haa95532_0	28 KB
cryptography-38.0.1	py310h21b164f_0	996 KB
flatbuffers-2.0.0	h6c2663c_0	1.4 MB
flit-core-3.6.0	pyhd3eb1b0_0	42 KB
frozenset-1.3.3	py310h2bbff1b_0	40 KB
gast-0.5.3	pyhd3eb1b0_0	21 KB
google-auth-2.6.0	pyhd3eb1b0_0	83 KB
google-auth-oauthlib-0.4.4	pyhd3eb1b0_0	18 KB
google-pasta-0.2.0	pyhd3eb1b0_0	46 KB

Рисунок 7 - Установка Tensorflow

Пакет Tensorflow успешно установился.

7. Сформируйте файлы requirements.txt и environment.yml.
Проанализируйте содержимое этих файлов.

```
(LL_2.14) C:\Users\aa715\OneDrive\Рабочий стол\9>conda env export > environment.yml
(LL_2.14) C:\Users\aa715\OneDrive\Рабочий стол\9>
```

Рисунок 8 - Формирование файла environmental.yml

```
(LL_2.14) C:\Users\aa715\OneDrive\Рабочий стол\9>pip freeze > requirements.txt
(LL_2.14) C:\Users\aa715\OneDrive\Рабочий стол\9>
```

Рисунок 9 - Формирование файла requirements.txt

Все пакеты, которые были установлены перед выполнением команды и предположительно использованы в каком-либо проекте, будут перечислены в файлах с именем «requirements.txt» и «environmental.yml».

Контрольные вопросы:

1. Каким способом можно установить пакет Python, не входящий в стандартную библиотеку?

Необходимо узнать, какой пакет содержит функционал, который вам необходим, найти его, скачать, разместить в нужном каталоге и начать использовать.

Существует так называемый Python Package Index (PyPI) – это репозиторий, открытый для всех Python разработчиков, в нем вы можете найти пакеты для решения практически любых задач.

2. Как осуществить установку менеджера пакетов pip?

Будем считать, что Python у вас уже установлен, теперь необходимо установить pip. Для того, чтобы это сделать, скачайте скрипт get-pip.py

```
$ curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
```

и выполните его.

```
$ python get-pip.py
```

3. Откуда менеджер пакетов pip по умолчанию устанавливает пакеты?

По умолчанию pip устанавливает пакеты из Python Package Index (PyPI) – это репозиторий, открытый для всех Python разработчиков, в нем вы можете найти пакеты для решения практически любых задач.

4. Как установить последнюю версию пакета с помощью pip?

```
$ pip install ProjectName
```

5. Как установить заданную версию пакета с помощью pip?

```
$ pip install ProjectName==3.2
```

6. Как установить пакет из git репозитория (в том числе GitHub) с помощью pip?

```
$ pip install -e git+https://gitrepo.com/ProjectName.git
```


7. Как установить пакет из локальной директории с помощью pip?

```
$ pip install ./dist/ProjectName.tar.gz
```

8. Как удалить установленный пакет с помощью pip?

```
$ pip uninstall ProjectName
```

9. Как обновить установленный пакет с помощью pip?

```
$ pip install --upgrade ProjectName
```

10. Как отобразить список установленных пакетов с помощью pip?

```
$ pip list
```

11. Каковы причины появления виртуальных окружений в языке Python?

В системе для интерпретатора Python может быть установлена глобально только одна версия пакета. Это порождает ряд проблем.

1. Проблема обратной совместимости
2. Проблема коллективной разработки

Если вы уже сталкивались с этой проблемой, то уже задумались, что для каждого проекта нужна своя "песочница", которая изолирует зависимости. Такая "песочница" придумана и называется "виртуальным окружением" или "виртуальной средой".

12. Каковы основные этапы работы с виртуальными окружениями?

1. Создаём через утилиту новое виртуальное окружение в отдельной папке для выбранной версии интерпретатора Python.
2. Активируем ранее созданное виртуального окружения для работы.
3. Работаем в виртуальном окружении, а именно управляем пакетами используя pip и запускаем выполнение кода.
4. Деактивируем после окончания работы виртуальное окружение.
5. Удаляем папку с виртуальным окружением, если оно нам больше не нужно.

13. Как осуществляется работа с виртуальными окружениями с помощью venv?

Создание виртуального окружения

Для создания виртуального окружения достаточно дать команду в формате:

```
python3 -m venv <путь к папке виртуального окружения>
```

Чтобы активировать виртуальное окружение под Windows команда выглядит иначе:

```
> env\\Scripts\\activate
```

Чтобы переключиться с одного окружения на другое нам нужно выполнить команду деактивации команду активации другого виртуального окружения, например, так:

```
> C:\\Python38\\python -m venv env
```

14. Как осуществляется работа с виртуальными окружениями с помощью virtualenv?

Создание виртуального окружения с утилитой virtualenv отличается от стандартного. Например, создание в текущей папке виртуального окружения для интерпретатора доступного через команду python3 с названием папки окружения env:

```
virtualenv -p python3 env
```

Активация и деактивация такая же, как у стандартной утилиты Python.

```
> env\\Scripts\\activate
```

```
(env) > deactivate
```

15. Изучите работу с виртуальными окружениями pipenv. Как осуществляется работа с виртуальными окружениями pipenv?

Для формирования и развертывания пакетных зависимостей используется утилита pip. Основные возможности pipenv:

- Создание и управление виртуальным окружением;
- Синхронизация пакетов в Pipfile при установке и удалении пакетов;
- Автоматическая подгрузка переменных окружения из .env файла .

После установки `pipenv` начинается работа с окружением. Его можно создать в любой папке. Достаточно установить любой пакет внутри папки.

Используем `requests`, он автоматически установит окружение и создаст `Pipfile` и `Pipfile.lock`.

16. Каково назначение файла `requirements.txt` ? Как создать этот файл? Какой он имеет формат?

Все пакеты, которые вы установили перед выполнением команды и предположительно использовали в каком-либо проекте, будут перечислены в файле с именем «`requirements.txt`». Кроме того, будут указаны их точные версии.

```
pip freeze > requirements.txt
```

```
pip install -r requirements.txt
```

17. В чем преимущества пакетного менеджера `conda` по сравнению с пакетным менеджером `pip`?

Основная проблема заключается в том, что `pip`, `easy_install` и `virtualenv` ориентированы на Python. Эти инструменты игнорируют библиотеки зависимостей, реализованные с использованием других языков. Например, XSLT, HDF5, MKL и другие, которые не имеют `setup.py` в исходном коде и не устанавливают файлы в директорию `site-packages`.

`Conda` же способна управлять пакетами как для Python, так и для C/C++, R, Ruby, Lua, Scala и других. `Conda` устанавливает двоичные файлы, поэтому работу по компиляции пакета самостоятельно выполнять не требуется (по сравнению с `pip`).

18. В какие дистрибутивы Python входит пакетный менеджер `conda`?

Anaconda и Miniconda.

19. Как создать виртуальное окружение `conda`?

```
conda create -n $PROJ_NAME python=3.7
```

20. Как активировать и установить пакеты в виртуальное окружение `conda`?

```
conda activate %PROJ_NAME%
```

```
conda install django, pandas
```

21. Как деактивировать и удалить виртуальное окружение conda?

```
conda deactivate
```

Если вы хотите удалить только что созданное окружение, выполните:

```
conda remove -n $PROJ_NAME
```

22. Каково назначение файла environment.yml ? Как создать этот файл?

Файл environment.yml позволит воссоздать окружение в любой нужный момент.

```
conda env export > environment.yml
```

23. Как создать виртуальное окружение conda с помощью файла environment.yml ?

```
conda env create -f environment.yml
```

24. Самостоятельно изучите средства IDE PyCharm для работы с виртуальными окружениями conda. Опишите порядок работы с виртуальными окружениями conda в IDE PyCharm.

Для первого способа ход работы следующий: запускаем PyCharm и в окне приветствия выбираем Create New Project. В мастере создания проекта, указываем в поле Location путь расположения создаваемого проекта. Имя конечной директории также является именем проекта. Далее разворачиваем параметры окружения, щелкая по Project Interpreter. И выбираем New environment using Virtualenv. Путь расположения окружения генерируется автоматически. И нажимаем на Create. Теперь установим библиотеки, которые будем использовать в программе. С помощью главного меню переходим в настройки File → Settings. Где переходим в Project: project_name → Project Interpreter. Выходим из настроек. Для запуска программы, необходимо создать профиль с конфигурацией. Для этого в верхнем правом углу нажимаем на кнопку Add Configuration. Откроется окно Run/Debug

Configurations, где нажимаем на кнопку с плюсом (Add New Configuration) в правом верхнем углу и выбираем Python. Далее указываем в поле Name имя конфигурации и в поле Script path расположение Python файла с кодом программы. В завершение нажимаем на Apply, затем на ОК. Для второго способа необходимо сделать следующее: на экране приветствия в нижнем правом углу через Configure → Settings переходим в настройки. Затем переходим в раздел Project Interpreter. В верхнем правом углу есть кнопка с шестерёнкой, нажимаем на неё и выбираем Add, создавая новое окружение. И указываем расположение для нового окружения. Нажимаем на ОК. Далее в созданном окружении устанавливаем нужные пакеты. И выходим из настроек. В окне приветствия выбираем Create New Project. В мастере создания проекта, указываем имя расположения проекта в поле Location. Разворачиваем параметры окружения, щелкая по Project Interpreter, где выбираем Existing interpreter и указываем нужное нам окружение. Далее создаем конфигурацию запуска программы, также как создавали для раннее. После чего можно выполнить программу.

25. Почему файлы requirements.txt и environment.yml должны храниться в репозитории git?

Чтобы пользователи, которые скачивают какие-либо программы, скрипты, модули могли без проблем посмотреть, какие пакеты им нужно установить дополнительно для корректной работы. За описание о наличии какихлибо пакетов в среде как раз и отвечают файлы requirements.txt и environment.yml.

Вывод: были приобретены навыки по работе с менеджером пакетов pip и виртуальными окружениями при написании программ с помощью языка программирования Python версии 3.x.