

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №4
дисциплины «Языки программирования»

Выполнила:
Иващенко Олеся Игорьевна
2 курс, группа ИТС-б-о-21-1,
11.03.02 «Инфокоммуникационные
технологии и системы связи»,
направленность (профиль)
«Инфокоммуникационные системы и
сети», очная форма обучения

(подпись)

Руководитель практики:
Воронкин Р. А., доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2022 г.

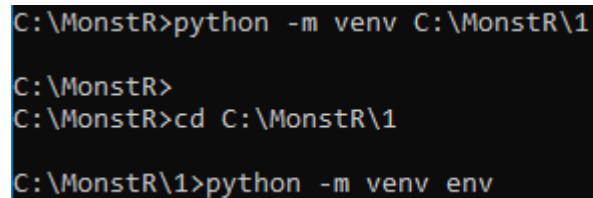
Тема: Установка пакетов в Python. Виртуальные окружения

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

Ход работы:

1. Виртуальное окружение с venv.

1. Для создания виртуального окружения достаточно дать команду.
2. Создадим виртуальное окружение в папке проекта. Для этого необходимо перейти в корень любого проекта на Python ≥ 3.3 и дать команду:

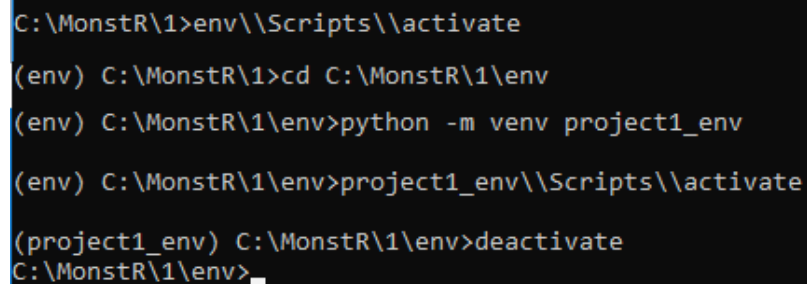


```
C:\MonstR>python -m venv C:\MonstR\1
C:\MonstR>
C:\MonstR>cd C:\MonstR\1
C:\MonstR\1>python -m venv env
```

Рисунок 1. Создание виртуального окружения

3. Чтобы активировать виртуальное окружение под Windows вводим команду.

4. Чтобы переключиться с одного окружения на другое нам нужно выполнить команду деактивации и команду активации другого виртуального окружения.



```
C:\MonstR\1>env\Scripts\activate
(env) C:\MonstR\1>cd C:\MonstR\1\env
(env) C:\MonstR\1\env>python -m venv project1_env
(env) C:\MonstR\1\env>project1_env\Scripts\activate
(project1_env) C:\MonstR\1\env>deactivate
C:\MonstR\1\env>_
```

Рисунок 2. Активация и деактивация виртуального окружения

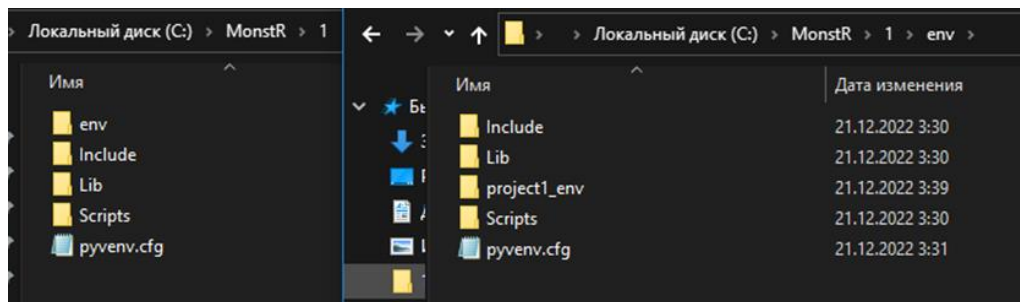


Рисунок 3. Файлы виртуального окружения

2. Виртуальное окружение с virtualenv.

1. Установка пакета. Установку можно выполнить командой:

```
C:\Users\MonstR>python -m pip install virtualenv
Collecting virtualenv
  Downloading virtualenv-20.17.1-py3-none-any.whl (8.8 MB)
----- 8.8/8.8 MB 6.4 MB/s eta 0:00:00
Collecting distlib<1,>=0.3.6
  Downloading distlib-0.3.6-py2.py3-none-any.whl (468 kB)
----- 468.5/468.5 kB 9.8 MB/s eta 0:00:00
Collecting filelock<4,>=3.4.1
  Downloading filelock-3.8.2-py3-none-any.whl (10 kB)
Collecting platformdirs<3,>=2.4
  Downloading platformdirs-2.6.0-py3-none-any.whl (14 kB)
Installing collected packages: distlib, platformdirs, filelock, virtualenv
Successfully installed distlib-0.3.6 filelock-3.8.2 platformdirs-2.6.0 virtualenv-20.17.1
C:\Users\MonstR>
```

Рисунок 4. Установка пакета

2. 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:

```
C:\Users\MonstR>virtualenv -p python3 env
created virtual environment CPython3.11.1.final.0-64 in 2328ms
creator CPython3Windows(dest=C:\Users\MonstR\env, clear=False, no_vcs_ignore=False, global=False)
seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=C:\Users\MonstR\AppData\Local\pypa\virtualenv)
added seed packages: pip==22.3.1, setuptools==65.6.3, wheel==0.38.4
activators BashActivator,BatchActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator
```

Рисунок 5. Создание виртуального окружения

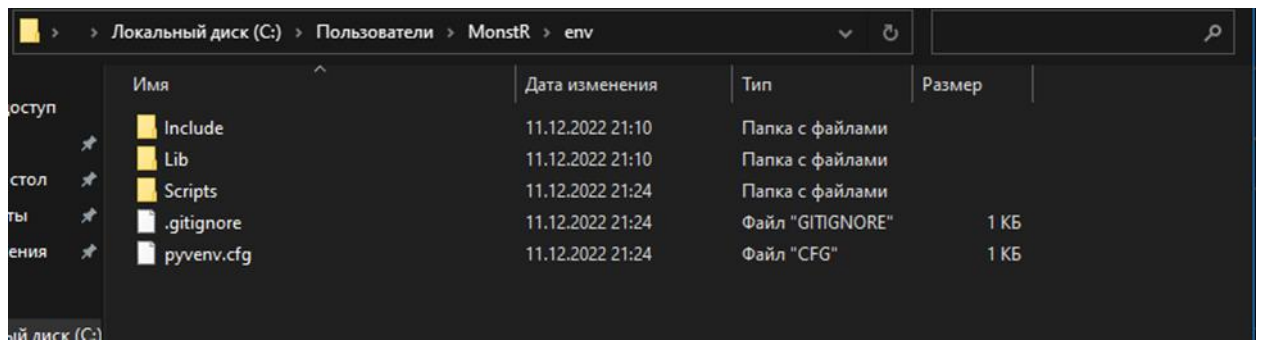


Рисунок 6. Файлы виртуального окружения

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

```
C:\Users\MonstR>env\Scripts\activate
(env) C:\Users\MonstR>deactivate
C:\Users\MonstR>
```

Рисунок 7. Активация и деактивация виртуального окружения

3. Перенос виртуального окружения.

1. Просмотреть список зависимостей можно командой:

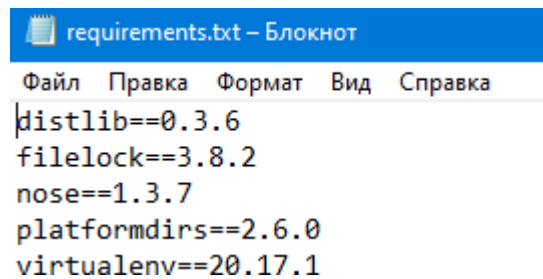
```
C:\Users\MonstR>pip freeze
distlib==0.3.6
filelock==3.8.2
nose==1.3.7
platformdirs==2.6.0
virtualenv==20.17.1
```

Рисунок 8. Список зависимостей

2. Чтобы его сохранить, нужно перенаправить вывод команды в файл:

```
C:\Users\MonstR>pip freeze > requirements.txt
```

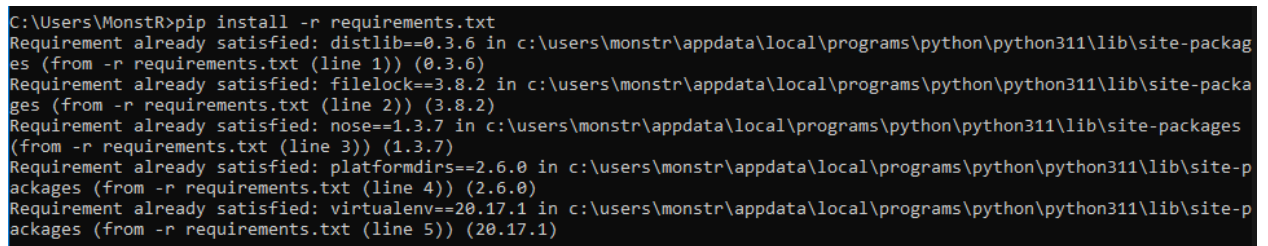
Рисунок 9. Команда перенаправления в файл



```
requirements.txt - Блокнот
Файл  Правка  Формат  Вид  Справка
distlib==0.3.6
filelock==3.8.2
nose==1.3.7
platformdirs==2.6.0
virtualenv==20.17.1
```

Рисунок 10. Файл со списком зависимостей

3. Установка пакетов из файла зависимостей в новом виртуальном окружении так же выполняется одной командой:



```
C:\Users\Monstr>pip install -r requirements.txt
Requirement already satisfied: distlib==0.3.6 in c:\users\monstr\appdata\local\programs\python\python311\lib\site-packages (from -r requirements.txt (line 1)) (0.3.6)
Requirement already satisfied: filelock==3.8.2 in c:\users\monstr\appdata\local\programs\python\python311\lib\site-packages (from -r requirements.txt (line 2)) (3.8.2)
Requirement already satisfied: nose==1.3.7 in c:\users\monstr\appdata\local\programs\python\python311\lib\site-packages (from -r requirements.txt (line 3)) (1.3.7)
Requirement already satisfied: platformdirs==2.6.0 in c:\users\monstr\appdata\local\programs\python\python311\lib\site-packages (from -r requirements.txt (line 4)) (2.6.0)
Requirement already satisfied: virtualenv==20.17.1 in c:\users\monstr\appdata\local\programs\python\python311\lib\site-packages (from -r requirements.txt (line 5)) (20.17.1)
```

Рисунок 11. Установка файла зависимостей

4. Виртуальное окружение с conda.

1. Начиная проект, создайте чистую директорию и дайте ей понятное короткое имя. Для Windows это будет соответствовать набору команд:



```
(base) PS C:\Users\Monstr> mkdir Monya

Каталог: C:\Users\Monstr

Mode                LastWriteTime         Length Name
----                -
d-----           11.12.2022     22:16         Monya

(base) PS C:\Users\Monstr> cd Monya
(base) PS C:\Users\Monstr\Monya> copy NUL > main.py
```

Рисунок 12. Установка файла зависимостей

2. Создайте чистое conda окружение с таким же именем, как директория проекта, и затем активируйте его.

```

(base) PS C:\Users\MonstR\Monya> conda create -n Monya python=3.7
WARNING: A conda environment already exists at 'C:\MonstR\Anaconda\envs\Monya'
Remove existing environment (y/[n])? y

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:\MonstR\Anaconda\envs\Monya

  added / updated specs:
    - python=3.7

The following NEW packages will be INSTALLED:

ca-certificates      pkgs/main/win-64::ca-certificates-2022.10.11-haa95532_0 None
certifi              pkgs/main/win-64::certifi-2022.9.24-py37haa95532_0 None
openssl              pkgs/main/win-64::openssl-1.1.1s-h2bbff1b_0 None
pip                  pkgs/main/win-64::pip-22.3.1-py37haa95532_0 None
python               pkgs/main/win-64::python-3.7.15-h6244533_1 None
setuptools           pkgs/main/win-64::setuptools-65.5.0-py37haa95532_0 None
sqlite               pkgs/main/win-64::sqlite-3.40.0-h2bbff1b_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-py37haa95532_2 None

Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#   $ conda activate Monya
#
# To deactivate an active environment, use
#
#   $ conda deactivate

Retrieving notices: ...working... done
(base) PS C:\Users\MonstR\Monya> conda activate Monya

```

Рисунок 13. Создание окружения и его активация

3. Установите пакеты, необходимые для реализации проекта.

```
(Monya) PS C:\Users\Monstr\Monya> conda install django, pandas
Collecting package metadata (current_repodata.json): done
Solving environment: failed with initial frozen solve. Retrying with flexible solve.
Solving environment: failed with repodata from current_repodata.json, will retry with next repodata source.
Collecting package metadata (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:\Monstr\Anaconda\envs\Monya

  added / updated specs:
    - django
    - pandas

The following NEW packages will be INSTALLED:

  asgiref                pkgs/main/win-64::asgiref-3.5.2-py37haa95532_0 None
  blas                  pkgs/main/win-64::blas-1.0-mkl None
  bottleneck             pkgs/main/win-64::bottleneck-1.3.5-py37h080aedc_0 None
  django                pkgs/main/win-64::django-3.2.15-py37haa95532_0 None
  flit-core              pkgs/main/noarch::flit-core-3.6.0-pyhd3eb1b0_0 None
  intel-openmp           pkgs/main/win-64::intel-openmp-2021.4.0-haa95532_3556 None
  krb5                   pkgs/main/win-64::krb5-1.19.2-h5b6d351_0 None
  libpq                 pkgs/main/win-64::libpq-12.9-hb652d5d_3 None
  mkl                   pkgs/main/win-64::mkl-2021.4.0-haa95532_640 None
  mkl-service           pkgs/main/win-64::mkl-service-2.4.0-py37h2bbff1b_0 None
  mkl_fft               pkgs/main/win-64::mkl_fft-1.3.1-py37h277e83a_0 None
  mkl_random             pkgs/main/win-64::mkl_random-1.2.2-py37hf11a4ad_0 None
  numexpr               pkgs/main/win-64::numexpr-2.8.4-py37h5b0cc5e_0 None
  numpy                 pkgs/main/win-64::numpy-1.21.5-py37h7a0a035_3 None
  numpy-base            pkgs/main/win-64::numpy-base-1.21.5-py37hca35cd5_3 None
  packaging             pkgs/main/noarch::packaging-21.3-pyhd3eb1b0_0 None
  pandas                pkgs/main/win-64::pandas-1.3.5-py37h6214cd6_0 None
  pycopg2               pkgs/main/win-64::pycopg2-2.9.3-py37hcd4344a_0 None
  pyparsing             pkgs/main/win-64::pyparsing-3.0.9-py37haa95532_0 None
  python-dateutil       pkgs/main/noarch::python-dateutil-2.8.2-pyhd3eb1b0_0 None
  pytz                  pkgs/main/win-64::pytz-2022.1-py37haa95532_0 None
  six                   pkgs/main/noarch::six-1.16.0-pyhd3eb1b0_1 None
  sqlparse              pkgs/main/win-64::sqlparse-0.4.3-py37haa95532_0 None
  typing_extensions     pkgs/main/win-64::typing_extensions-4.4.0-py37haa95532_0 None
  zlib                  pkgs/main/win-64::zlib-1.2.13-h8cc25b3_0 None

Proceed ([y]/n)? y
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
Retrieving notices: ...working... done
```

Рисунок 14. Установка пакетов

4. Периодически экспортируйте параметры окружения. Экспортируйте после установки, перед каждым большим или маленьким КОММИТОМ:

```
(Monya) PS C:\Users\Monstr\Monya> conda env export > environment.yml
```

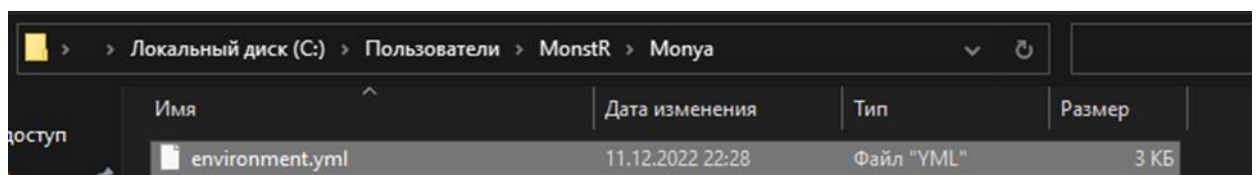


Рисунок 15. Экспорт параметров окружения

5. Когда пришло время прекратить разработку или переключиться на

новый проект, отключите среду.

6. Если вы хотите удалить только что созданное окружение, выполните:

```
(Monya) PS C:\Users\MonstR\Monya> conda deactivate
(base) PS C:\Users\MonstR\Monya> conda remove --name Monya --all

Remove all packages in environment C:\MonstR\Anaconda\envs\Monya:

## Package Plan ##

  environment location: C:\MonstR\Anaconda\envs\Monya

The following packages will be REMOVED:

asgiref-3.5.2-py37haa95532_0
blas-1.0-mkl
bottleneck-1.3.5-py37h080aedc_0
ca-certificates-2022.10.11-haa95532_0
certifi-2022.9.24-py37haa95532_0
django-3.2.15-py37haa95532_0
flit-core-3.6.0-pyhd3eb1b0_0
intel-openmp-2021.4.0-haa95532_3556
krb5-1.19.2-h5b6d351_0
libpq-12.9-hb652d5d_3
mkl-2021.4.0-haa95532_640
mkl-service-2.4.0-py37h2bbff1b_0
mkl_fft-1.3.1-py37h277e83a_0
mkl_random-1.2.2-py37hf11a4ad_0
numexpr-2.8.4-py37h5b0cc5e_0
numpy-1.21.5-py37h7a0a035_3
numpy-base-1.21.5-py37hca35cd5_3
openssl-1.1.1s-h2bbff1b_0
packaging-21.3-pyhd3eb1b0_0
pandas-1.3.5-py37h6214cd6_0
pip-22.3.1-py37haa95532_0
psycopg2-2.9.3-py37hcd4344a_0
pyparsing-3.0.9-py37haa95532_0
python-3.7.15-h6244533_1
python-dateutil-2.8.2-pyhd3eb1b0_0
pytz-2022.1-py37haa95532_0
setuptools-65.5.0-py37haa95532_0
six-1.16.0-pyhd3eb1b0_1
sqlite-3.40.0-h2bbff1b_0
sqlparse-0.4.3-py37haa95532_0
typing_extensions-4.4.0-py37haa95532_0
vc-14.2-h21ff451_1
vs2015_runtime-14.27.29016-h5e58377_2
wheel-0.37.1-pyhd3eb1b0_0
wincertstore-0.2-py37haa95532_2
zlib-1.2.13-h8cc25b3_0
```

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

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

Рисунок 16. Экспорт параметров окружения

7. Файл `environment.yml` позволит воссоздать окружение в любой нужный момент. Достаточно набрать:


```
(base) PS C:\Users\MonstR\Monya> conda env create -f environment.yml
Collecting package metadata (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

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate Monya
#
# To deactivate an active environment, use
#
#     $ conda deactivate

Retrieving notices: ...working... done
(base) PS C:\Users\MonstR\Monya> conda activate Monya
(Monya) PS C:\Users\MonstR\Monya>
```

Рисунок 17. Экспорт параметров окружения

Ответы на контрольные вопросы:

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

При помощи созданного виртуального окружения с conda, командой conda install, diango, pandas.

2. Как осуществить установку менеджера пакетов pip?

Для того, чтобы это сделать, нужно скачать скрипт get-pip.py \$ curl <https://bootstrap.pypa.io/get-pip.py> -o get-pip.py и выполнить его. \$ python get - pip.py. При этом, вместе с pip будут установлены setuptools и wheels. Setuptools – это набор инструментов для построения пакетов Python. Wheels – это формат дистрибутива для пакета Python.

3. Откуда менеджер пакетов pip по умолчанию устанавливает пакеты?

Если вы пользуетесь Python 2.7.9 (и выше) или Python 3.4 (и выше), PIP устанавливается вместе с Python по умолчанию.

4. Как установить последнюю версию пакета с помощью pip?

Ввести в cmd: `$ pip install <Название пакета>`.

5. Как установить заданную версию пакета с помощью pip?

Ввести в cmd: `$ pip install <Название пакета> ==<Версия>`.

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

Ввести в cmd: `$ pip install -e git+https://gitrepo.com/<Название пакета>.git`.

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

Ввести в cmd: `$ pip install ./dist/<Название пакета>.tar.gz`.

8. Как удалить установленный пакет с помощью pip?

Ввести в cmd: `$ pip uninstall <Название пакета>`.

9. Как обновить установленный пакет с помощью pip?

Ввести в cmd: `$ pip install --upgrade <Название пакета>`.

10. Как отобразить список установленных пакетов с помощью pip?

Ввести в cmd: `$ pip list`.

11. Каковы причины появления виртуальных окружений в языке Python?

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?

Создаем, активируем, работаем, деактивируем, удаляем.

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

Создание виртуального окружения с помощью cmd, активируется, производится работа и в дальнейшем деактивация.

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

Также как и с `venv`, но `virtualenv` очень распространён и поддерживает большее число вариантов и версий интерпретатора Python, например, PyPy и CPython.

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

После установки `pipenv` начинается работа с окружением. Просто установите любой пакет внутри папки. Используем `requests`. Он автоматически установит окружение и создаст `Pipfile` и `Pipfile.lock`. В директории будут созданы два файла. `Pipfile`, включающий список пакетов, версию Python и прочую информацию. Файл блокировки (lock file) генерируется в качестве `Pipfile.lock`. Чтобы заглянуть внутрь Python и проверить наличие пакетов с помощью `import requests`, используется команда `python`. Если ошибокнет, это значит, что пакет успешно установлен.

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

Имя файла хранения зависимостей `requirements.txt` выбрано не зря. Оно является стандартной договоренностью и используется некоторыми утилитами автоматически. Установка пакетов из файла зависимостей в новом виртуальном окружении так же выполняется одной командой:

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

Данный формат является обычным текстовым файлом, где указано название пакета `python`, его версия и условие, больше, меньше, равно.

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).

Существуют также некоторые различия, если вы заинтересованы в создании собственных пакетов. Например, pip создан на основе setuptools, тогда как conda использует свой собственный формат, который имеет некоторые преимущества (например, статическая компиляция пакета).

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

Все чаще среди Python-разработчиков заходит речь о менеджере пакетов conda, включенный в состав дистрибутивов Anaconda и Miniconda. JetBrains включил этот инструмент в состав PyCharm.

19. Как создать виртуальное окружение conda?

Создайте чистое conda-окружение с таким же именем, как директория проекта, и затем активируйте его.

```
conda create -n %PROJ_NAME% python=3.7
conda activate %PROJ_NAME%
```

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

```
conda activate %PROJ_NAME%
```

```
conda install diango, pandas
```

21. Как деактивировать и удалить виртуальное окружение conda?

```
conda deactivate
```

```
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. Создание файла по команде: conda env create -f environment.yml.

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

Необходимо создать новое виртуальное окружение и в нем воссоздать

окружение.

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

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

Чтобы в удобный момент можно было воссоздать виртуальное окружение или установить его.

Вывод: Приобрела навыки по работе с менеджером пакетов pip и виртуальными окружениями с помощью языка программирования Python версии 3.x.