# МИНИCTEPCTBO НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙ- СКОЙ ФЕДЕРАЦИИ

**Федеральное государственное автономное образовательное учреждение высшего образования**

# «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

**Кафедра инфокоммуникаций**

# Основы кроссплатформенного программирования Отчет по лабораторной работе №2.14

Тема: «Установка пакетов в Python. Виртуальные окружения»

|  |
| --- |
| Выполнил студент группы  ИВТ-б-о-21-1 |
| Назаров Н.Ю. « » 20 г. |
| Подпись студента |
| Работа защищена « » 20 г. |
| Проверил доцент  Кафедры инфокоммуникаций, старший преподаватель  Воронкин Р.А.  (подпись) |

Ставрополь 2022

**Выполнение работы**

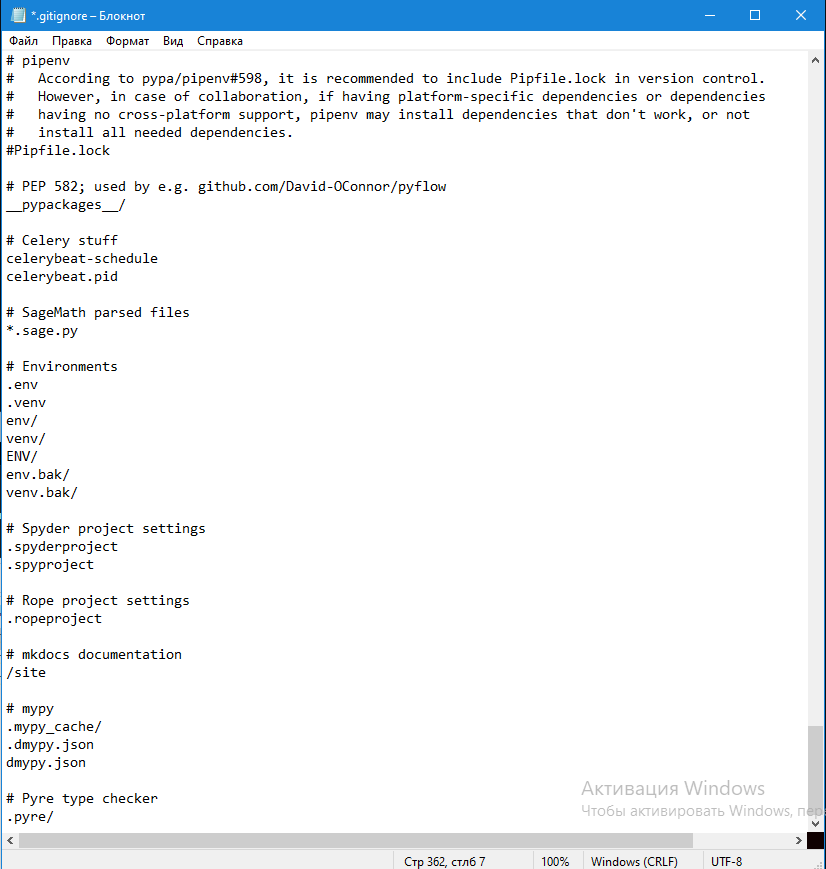


Рисунок 1 – редактирование gitignore

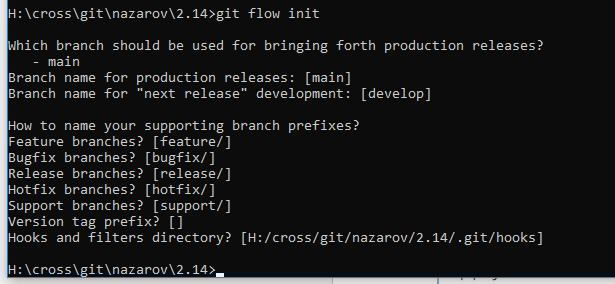


Рисунок 2 – организация репозитория в соответствии с git-flow

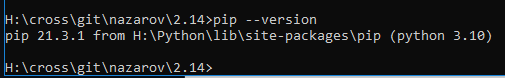


Рисунок 3 – проверка, установлен ли pip



Рисунок 4 – установка виртуального окружения

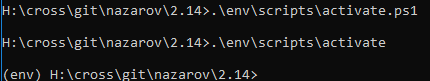


Рисунок 5 – активация виртуального окружения

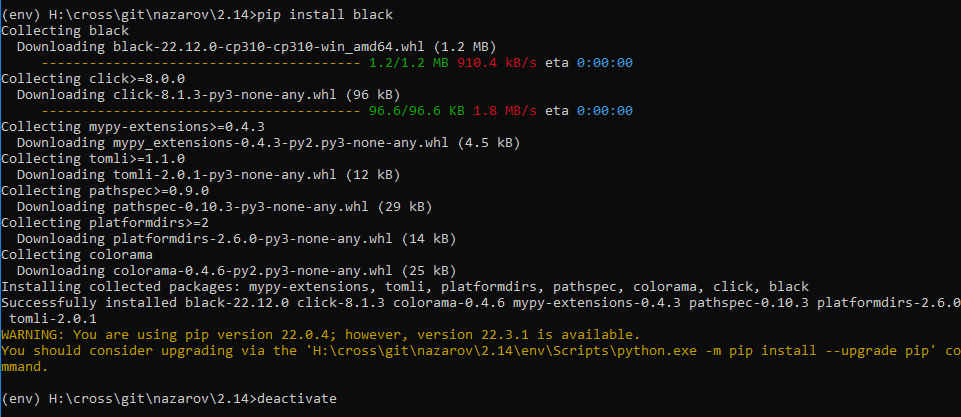


Рисунок 6 – установка пакета black и деактивация виртуального окружения



Рисунок 7 – установка virtualenv

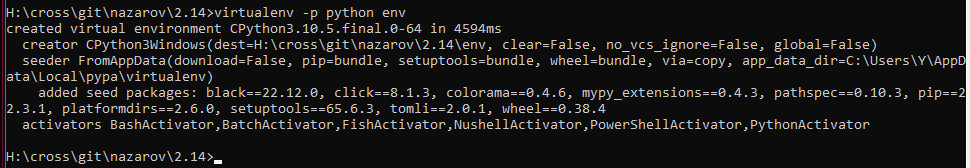


Рисунок 8 – установка виртуального окружения

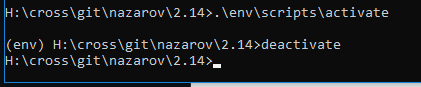


Рисунок 9 – активация и деактивация

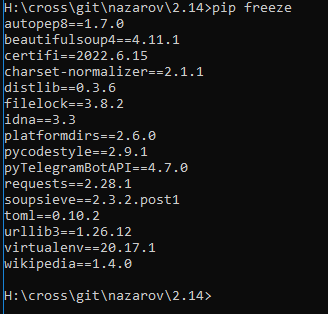


Рисунок 10 – список пакетных зависимостей

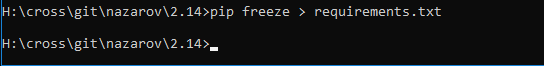


Рисунок 11 – сохранение пакетных зависимостей в отдельный файл

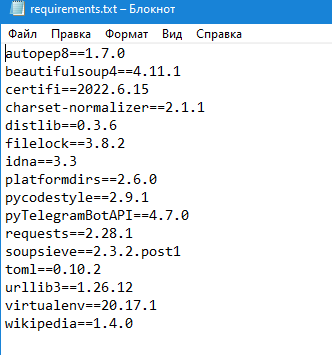


Рисунок 12 – файл requirements.txt

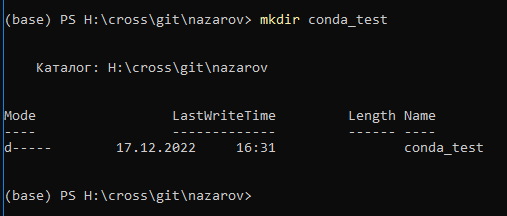


Рисунок 13 – создание чистой директории и виртуального окружения

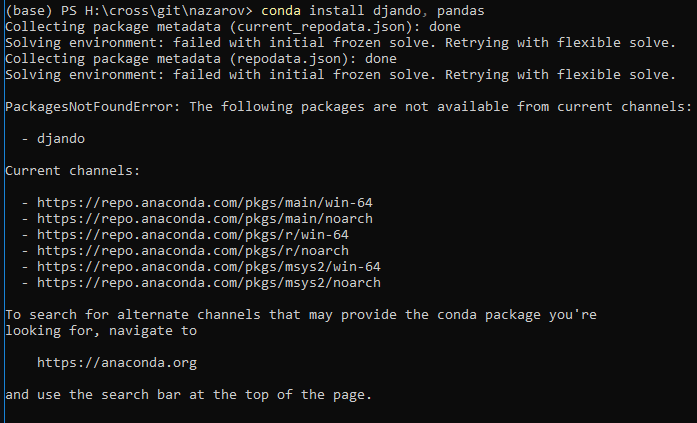


Рисунок 15 – установка django и pandas



Рисунок 16 – создание файла конфигурации

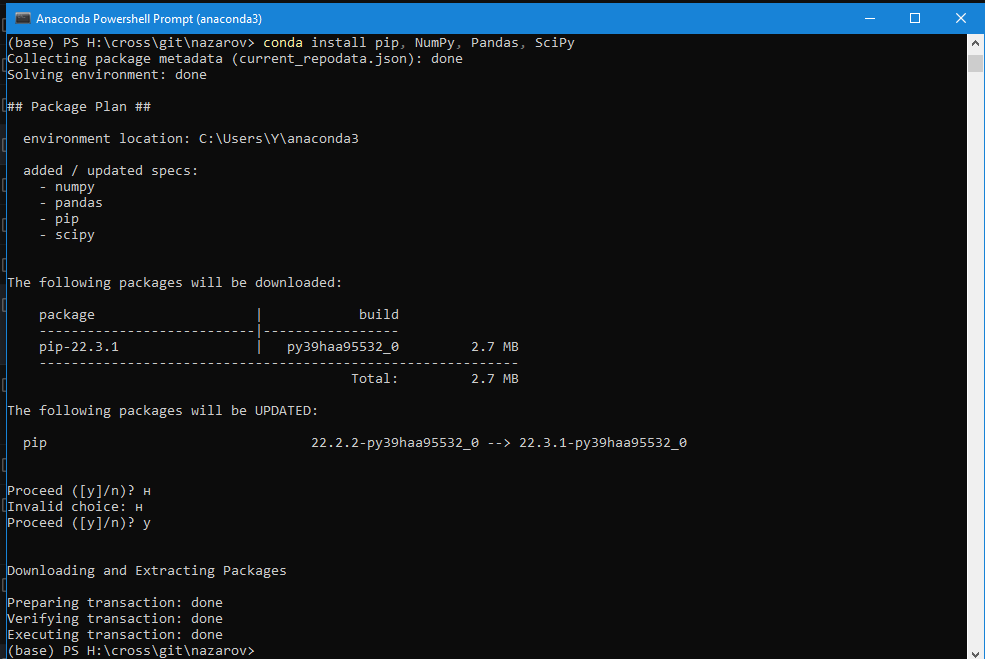


Рисунок 17 – установка необходимых пакетов

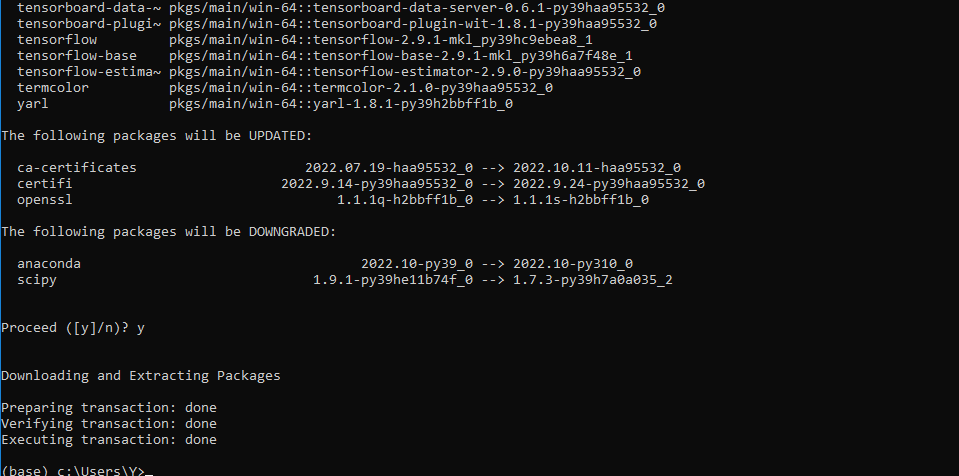


Рисунок 18 – установка TensorFlow

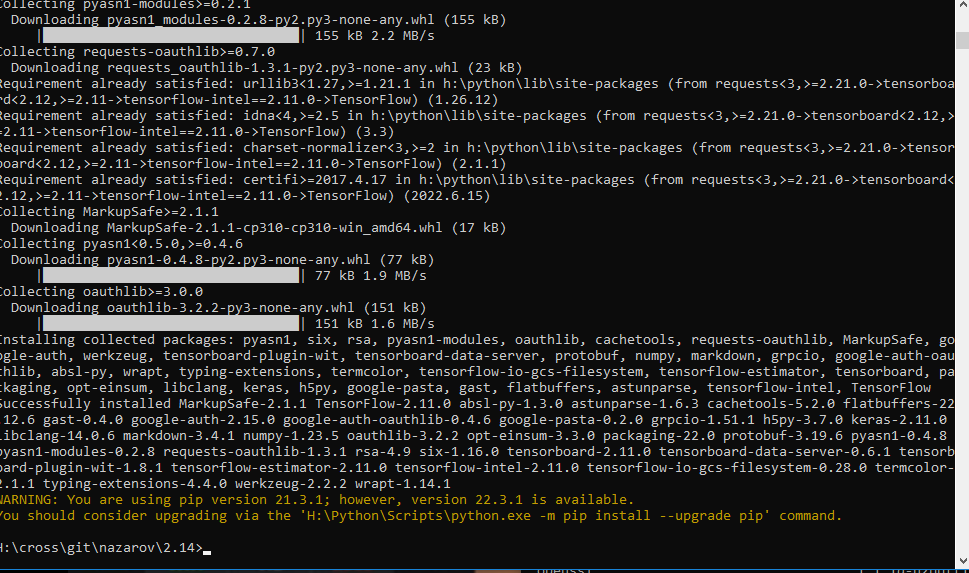


Рисунок 19 – установка TensorFlow через pip

# Ответы на контрольные вопросы:

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

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

# Как осуществить установку менеджера пакетов pip?

При развертывании современной версии Python, pip устанавливается ав- томатически. Но если, по какой-то причине, pip не установлен на вашем ПК, то сделать это можно вручную. Чтобы установить pip, нужно скачать скрипт get-pip.py и выполнить его.

# Откуда менеджер пакетов pip по умолчанию устанавливает па- кеты?

По умолчанию менеджер пакетов pip скачивает пакеты из Python Package Index (PyPI).

# Как установить последнюю версию пакета с помощью pip?

С помощью команды $ pip install ProjectName.

# Как установить заданную версию пакета с помощью pip?

С помощью команды $ pip install ProjectName==3.2, где вместо 3.2 необ- ходимо указать нужную версию пакета.

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

С помощью команды $ pip install e git+https://gitrepo.com/ ProjectName.git

# Как установить пакет из локальной директории с помощью pip?

С помощью команды $ pip install ./dist/ProjectName.tar.gz

# Как удалить установленный пакет с помощью pip?

С помощью команды $ pip uninstall ProjectName можно удалить установ- ленный пакет.

# Как обновить установленный пакет с помощью pip?

С помощью команды $ pip install --upgrade ProjectName можно обновить необходимый пакет.

# Как отобразить список установленных пакетов с помощью pip?

Командой $ pip list можно отобразить список установленных пакетов.

# Каковы причины появления виртуальных окружений в языке Python?

Существует несколько причин появления виртуальных окружений в языке Python - проблема обратной совместимости и проблема коллективной разработки. Проблема обратной совмести - некоторые операционные системы, например, Linux и MacOs используют содержащиеся в них предустановлен- ные интерпретаторы Python. Обновив или изменив самостоятельно версию ка- когото установленного глобально пакета, мы можем непреднамеренно сло- мать работу утилит и приложений из дистрибутива операционной системы.

Проблема коллективной разработки - Если разработчик работает над проектом не один, а с командой, ему нужно передавать и получать список за- висимостей, а также обновлять их на своем компьютере таким образом, чтобы не нарушалась работа других его проектов. Значит нам нужен механизм, кото- рый вместе с обменом проектами быстро устанавливал бы локально и все не- обходимые для них пакеты, при этом не мешая работе других проектов.

# Каковы основные этапы работы с виртуальными окружениями?

Основные этапы:

Создаём через утилиту новое виртуальное окружение в отдельной папк для выбранной версии интерпретатора Python.

Активируем ранее созданное виртуального окружения для работы.

Работаем в виртуальном окружении, а именно управляем пакетами ис- пользуя pip и запускаем выполнение кода.

Деактивируем после окончания работы виртуальное окружение.

Удаляем папку с виртуальным окружением, если оно нам больше не нужно.

# Как осуществляется работа с виртуальными окружениями с по- мощью venv?

С его помощью можно создать виртуальную среду, в которую можно устанавливать пакеты независимо от основной среды или других виртуальных окружений. Основные действия с виртуальными окружениями с помощью venv: создание виртуального окружения, его активация и деактивация.

# Как осуществляется работа с виртуальными окружениями с по- мощью virtualenv?

Для начала пакет нужно установить. Установку можно выполнить ко- мандой: python3 -m pip install virtualenv Virtualenv позволяет создать абсо- лютно изолированное виртуальное окружение для каждой из программ.

Окружением является обычная директория, которая содержит копию всего необходимого для запуска определенной программы, включая копию са- мого интерпретатора, полной стандартной библиотеки, pip, и, что самое глав- ное, копии всех необходимых пакетов.

# Изучите работу с виртуальными окружениями pipenv. Как осу- ществляется работа с виртуальными окружениями pipenv?

Для формирования и развертывания пакетных зависимостей использу- ется 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 со-

здать в любой папке. Достаточно установить любой пакет внутри папки.

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

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

Установить пакеты можно с помощью команды: pip install –r requirements.txt. Также можно использовать команду pip freeze > requirements.txt, которая создаст requirements.txt наполнив его названиями и версиями тех пакетов что используются вами в текущем окружении. Это удобно если вы разработали проект и в текущем окружении все работает, но вы хотите перенести проект в иное окружением (например, заказчику или на сервер). С помощью закрепления зависимостей мы можем быть уверены, что пакеты, установленные в нашей производственной среде, будут точно соот- ветствовать пакетам в нашей среде разработки, чтобы ваш проект неожиданно не ломался.

# В чем преимущества пакетного менеджера conda по сравнению с пакетным менеджером pip?

Conda способна управлять пакетами как для Python, так и для C/ C++, R, Ruby, Lua, Scala и других. Conda устанавливает двоичные файлы, поэтому ра- боту по компиляции пакета самостоятельно выполнять не требуется (по срав- нению с pip).

# 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.

# Как создать виртуальное окружение conda?

С помощью команды: conda create -n %PROJ\_NAME% python=3.7

# Как активировать и установить пакеты в виртуальное окруже- ние conda?

Чтобы установить пакеты, необходимо воспользоваться командой: – conda install А для активации: conda activate %PROJ\_NAME%

# Как деактивировать и удалить виртуальное окружение conda?

Для деактивации использовать команду: conda deactivate, а для удале- ния: conda remove -n $PROJ\_NAME.

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

Создание файла: conda env export > environment.yml

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

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

Достаточно набрать: conda env create -f environment.yml

# Самостоятельно изучите средства IDE PyCharm для работы с виртуальными окружениями conda. Опишите порядок работы с вирту- альными окружениями conda в IDE PyCharm.

Работа с виртуальными окружениями в PyCharm зависит от способа вза- имодействия с виртуальным окружением:

Создаём проект со своим собственным виртуальным окружением, куда затем будут устанавливаться необходимые библиотеки.

Предварительно создаём виртуальное окружение, куда установим нуж- ные библиотеки. И затем при создании проекта в PyCharm можно будет его выбирать, т.е. использовать для нескольких проектов.