

Лабораторная работа 2.14

Тема: Установка пакетов в Python. Виртуальные окружения.

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

Порядок выполнения работы

1. Создаём аккаунт в GitHub. Затем создаём новый общедоступный репозиторий, в котором будет использована лицензия MIT и язык программирования Python.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner *

 RomanGorchakov ▾

Repository name *

/ Test

✔ Test is available.

Great repository names are short and memorable. Need inspiration? How about [automatic-octo-chainsaw](#) ?

Description (optional)

☒  **Public**

Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**

You choose who can see and commit to this repository.

Initialize this repository with:

☐ Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: Python ▾

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

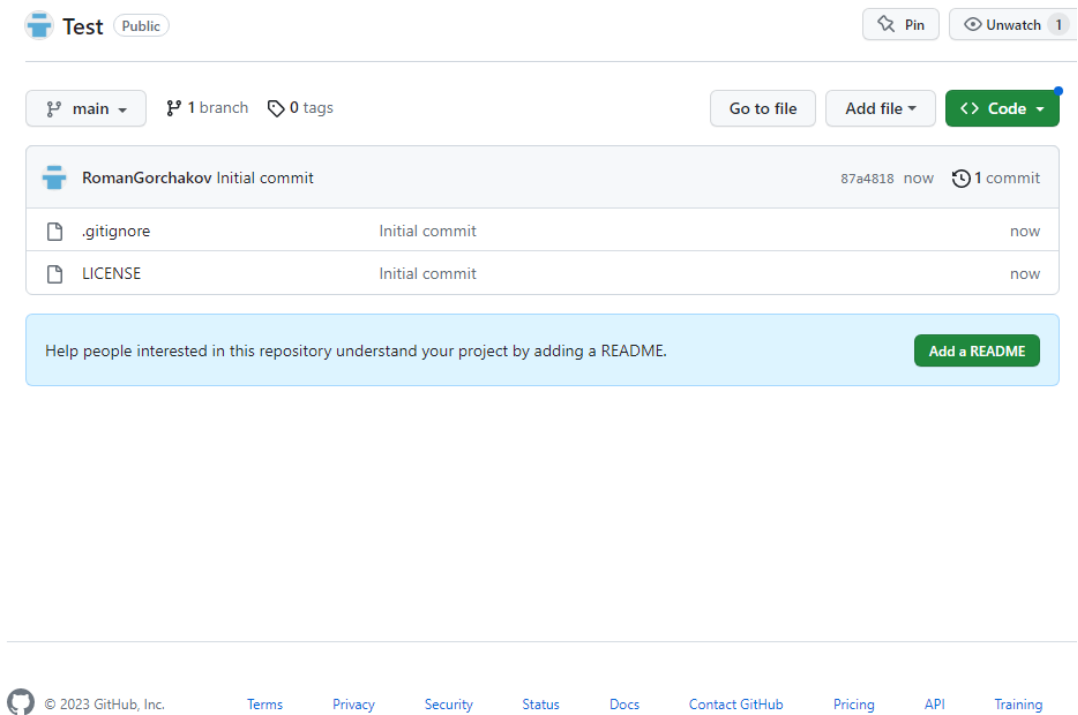
Choose a license

License: MIT License ▾


















A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

ⓘ You are creating a public repository in your personal account.

Create repository



2. Теперь необходимо дополнить файл `.gitignore` с необходимыми правилами для языка программирования Python. Для этого переходим по ссылке «<https://github.com/github/gitignore>» и скачиваем оттуда файл «Python.gitignore».

 Phalcon.gitignore	Remove trailing asterisks in Phalcon rules	10 years ago
 PlayFramework.gitignore	Added ./project/project to PlayFramework.gitignore	7 years ago
 Plone.gitignore	Covered by global vim template	10 years ago
 Prestashop.gitignore	Update for Prestashop 1.7 (#3261)	3 years ago
 Processing.gitignore	Ignore transpiled .java and .class files (#3016)	4 years ago
 PureScript.gitignore	Update PureScript adding .spago (#3278)	3 years ago
 Python.gitignore	Update Python.gitignore	last year
 Qooxdoo.gitignore	Add gitignore for qooxdoo apps	13 years ago
 Qt.gitignore	Remove trailing whitespace	2 years ago
 R.gitignore	Merge pull request #3792 from jl5000/patch-1	2 years ago
 README.md	Merge pull request #3854 from AnilSeervi/patch-1	2 years ago
 ROS.gitignore	Added ignore for files created by <code>catkin_make_isolated</code>	6 years ago
 Racket.gitignore	Update Racket.gitignore	2 years ago
 Rails.gitignore	Ignore Rails .env according recommendations	2 years ago
 Raku.gitignore	Changes the name of Perl 6 to Raku (#3312)	3 years ago
 RhodesRhomobile.gitignore	Add Rhodes mobile application framework gitignore	13 years ago
 Ruby.gitignore	Ruby: ignore RuboCop remote inherited config files (#3197)	4 years ago

```
1  # Byte-compiled / optimized / DLL files
2  __pycache__/
3  *.py[cod]
4  *$py.class
5
6  # C extensions
7  *.so
8
9  # Distribution / packaging
10 .Python
11 build/
12 develop-eggs/
13 dist/
14 downloads/
15 eggs/
16 .eggs/
17 lib/
18 lib64/
19 parts/
20 sdist/
21 var/
22 wheels/
23 share/python-wheels/
24 *.egg-info/
25 .installed.cfg
26 *.egg
27 MANIFEST
28
```

3. Теперь создаём файл «README.md», где вносим информацию о своей группе и ФИО. Сохраняем набранный текст через кнопку «Commit changes».



4. В окне «Codespace» выбираем опцию «Create codespace on main». Откроется терминал, куда мы введём команду «git clone», чтобы клонировать свой репозиторий. После этого организуем репозиторий в соответствии с моделью ветвления Git-flow. Для этого введём в терминал команды: «git checkout –b develop» для создания ветки разработки; «git branch feature_branch» для создания ветки функций; «git branch release/1.0.0» для создания ветки релиза; «git checkout main» и «git branch hotfix» для создания веток hotfix.

```
@RomanGorchakov →/workspaces/Py14 (main) $ git clone https://github.com/RomanGorchakov/Py14.git ClonedPy.git
Cloning into 'ClonedPy.git'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 9 (delta 1), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (9/9), 4.81 KiB | 4.81 MiB/s, done.
Resolving deltas: 100% (1/1), done.
@RomanGorchakov →/workspaces/Py14 (main) $ git checkout -b develop
Switched to a new branch 'develop'
@RomanGorchakov →/workspaces/Py14 (develop) $ git branch feature_branch
@RomanGorchakov →/workspaces/Py14 (develop) $ git branch release/1.0.0
@RomanGorchakov →/workspaces/Py14 (develop) $ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
@RomanGorchakov →/workspaces/Py14 (main) $ git branch hotfix
@RomanGorchakov →/workspaces/Py14 (main) $ git checkout develop
Switched to branch 'develop'
@RomanGorchakov →/workspaces/Py14 (develop) $
```

5. Создаём виртуальное окружение с помощью команды «python3 -m venv env». Теперь нужно установить pip. Для этого вводим команду «curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py» для получения скрипта get-pip.py и «python get-pip.py» для его выполнения. Устанавливаем пакеты NumPy, Pandas и SciPy с помощью команд «pip install».

```
@RomanGorchakov →/workspaces/Py14 (develop) $ curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           %             %             Dload  Upload  Total  Spent    Left   Speed
100 2574k  100 2574k    0     0  24.1M      0  --:--:-- --:--:-- --:--:-- 24.1M
@RomanGorchakov →/workspaces/Py14 (develop) $ python get-pip.py

Collecting pip
  Downloading pip-24.0-py3-none-any.whl.metadata (3.6 kB)
Collecting wheel
  Downloading wheel-0.42.0-py3-none-any.whl.metadata (2.2 kB)
  Downloading pip-24.0-py3-none-any.whl (2.1 MB)
    2.1/2.1 MB 24.9 MB/s eta 0:00:00
  Downloading wheel-0.42.0-py3-none-any.whl (65 kB)
    65.4/65.4 kB 1.9 MB/s eta 0:00:00
Installing collected packages: wheel, pip
  WARNING: The script wheel is installed in '/usr/local/python/3.10.13/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
  Attempting uninstall: pip
    Found existing installation: pip 23.3.2
    Uninstalling pip-23.3.2:
      Successfully uninstalled pip-23.3.2
  WARNING: The scripts pip, pip3 and pip3.10 are installed in '/usr/local/python/3.10.13/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
  Successfully installed pip-24.0 wheel-0.42.0
@RomanGorchakov →/workspaces/Py14 (develop) $
```

```
@RomanGorchakov →/workspaces/Py14 (develop) $ pip install numpy
Requirement already satisfied: numpy in /home/codespace/.local/lib/python3.10/site-packages (1.26.3)
@RomanGorchakov →/workspaces/Py14 (develop) $ pip install pandas
Requirement already satisfied: pandas in /home/codespace/.local/lib/python3.10/site-packages (2.1.4)
Requirement already satisfied: numpy<2,>=1.22.4 in /home/codespace/.local/lib/python3.10/site-packages (from pandas) (1.26.3)
Requirement already satisfied: python-dateutil>=2.8.2 in /home/codespace/.local/lib/python3.10/site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /home/codespace/.local/lib/python3.10/site-packages (from pandas) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in /home/codespace/.local/lib/python3.10/site-packages (from pandas) (2023.4)
Requirement already satisfied: six>=1.5 in /home/codespace/.local/lib/python3.10/site-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
@RomanGorchakov →/workspaces/Py14 (develop) $ pip install scipy
Requirement already satisfied: scipy in /home/codespace/.local/lib/python3.10/site-packages (1.11.4)
Requirement already satisfied: numpy<1.28.0,>=1.21.6 in /home/codespace/.local/lib/python3.10/site-packages (from scipy) (1.26.3)
@RomanGorchakov →/workspaces/Py14 (develop) $
```

6. Устанавливаем пакет Tensorflow с помощью менеджера пакетов conda, затем с помощью менеджера пакетов pip. Получаем доступ к файлу «requirements.txt» с помощью команды «pip freeze > requirements.txt» и к файлу «environment.yml» с помощью команды «conda env export > environment.yml».

```
requests-oauthlib pkgs/main/noarch::requests-oauthlib-1.3.0-py_0
rsa pkgs/main/noarch::rsa-4.7.2-pyhd3eb1b0_1
scipy pkgs/main/linux-64::scipy-1.11.4-py311h08b1b3b_0
six pkgs/main/noarch::six-1.16.0-pyhd3eb1b0_1
snappy pkgs/main/linux-64::snappy-1.1.10-h6a678d5_1
tbb pkgs/main/linux-64::tbb-2021.8.0-hdb19cb5_0
tensorboard pkgs/main/linux-64::tensorboard-2.12.1-py311h06a4308_0
tensorboard-data-server pkgs/main/linux-64::tensorboard-data-server-0.7.0-py311h52d8a92_0
tensorboard-plugin-wit pkgs/main/noarch::tensorboard-plugin-wit-1.6.0-py_0
tensorflow pkgs/main/linux-64::tensorflow-2.12.0-mkl_py311h34a0fa1_0
tensorflow-base pkgs/main/linux-64::tensorflow-base-2.12.0-mkl_py311he5f8e37_0
tensorflow-estimator pkgs/main/linux-64::tensorflow-estimator-2.12.0-py311h06a4308_0
termcolor pkgs/main/linux-64::termcolor-2.1.0-py311h06a4308_0
typing_extensions pkgs/main/linux-64::typing_extensions-4.9.0-py311h06a4308_1
werkzeug pkgs/main/linux-64::werkzeug-2.3.8-py311h06a4308_0
wrapt pkgs/main/linux-64::wrapt-1.14.1-py311h5eee18b_0
yarl pkgs/main/linux-64::yarl-1.9.3-py311h5eee18b_0
```

The following packages will be UPDATED:

```
certifi 2023.11.17-py311h06a4308_0 --> 2024.2.2-py311h06a4308_0
```

The following packages will be DOWNGRADED:

```
conda 23.11.0-py311h06a4308_0 --> 23.9.0-py311h06a4308_0
conda-libmamba-solver 23.11.1-py311h06a4308_0 --> 23.9.3-py311h06a4308_0
cryptography 41.0.3-py311hdda0065_0 --> 41.0.3-py311h130f0dd_0
icu 73.1-h6a678d5_0 --> 58.2-he6710b0_3
krb5 1.20.1-h143b758_1 --> 1.20.1-h568e23c_1
libarchive 3.6.2-h6ac8c49_2 --> 3.6.2-hb4bd9a0_1
libcurl 8.4.0-h251f7ec_1 --> 8.2.1-h91b91d3_0
libmamba 1.5.3-haf1ee3a_0 --> 1.5.1-hba0046a_0
libmambapy 1.5.3-py311h2dafd23_0 --> 1.5.1-py311ha06983f_0
libnghttp2 1.57.0-h2d74bed_0 --> 1.52.0-ha637b67_1
libssh2 1.10.0-hbdb6064_2 --> 1.10.0-h37d81fd_2
libxml2 2.10.4-hf1b16e4_1 --> 2.10.4-hcbfbd50_0
openssl 3.0.12-h7f8727e_0 --> 1.1.1w-h7f8727e_0
python 3.11.5-h955ad1f_0 --> 3.11.5-h7a1cb2a_0
yaml-cpp 0.8.0-h6a678d5_0 --> 0.7.0-h295c915_1
```

Proceed ([y]/n)? y

Downloading and Extracting Packages:

Preparing transaction: done

Verifying transaction: done

Executing transaction: done

@RomanGorchakov →/workspaces/Py14 (develop) \$

```
133.7/133.7 kB 4.7 MB/s eta 0:00:00
Downloading flatbuffers-23.5.26-py2.py3-none-any.whl (26 kB)
Downloading grpcio-1.60.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (5.4 MB)
5.4/5.4 MB 52.8 MB/s eta 0:00:00
Downloading h5py-3.10.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.8 MB)
4.8/4.8 MB 50.7 MB/s eta 0:00:00
Downloading keras-2.15.0-py3-none-any.whl (1.7 MB)
1.7/1.7 MB 34.0 MB/s eta 0:00:00
Downloading libclang-16.0.6-py2.py3-none-manylinux2010_x86_64.whl (22.9 MB)
22.9/22.9 MB 38.9 MB/s eta 0:00:00
Downloading ml_dtypes-0.2.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.0 MB)
1.0/1.0 MB 23.4 MB/s eta 0:00:00
Downloading protobuf-4.25.2-cp37-abi3-manylinux2014_x86_64.whl (294 kB)
294.6/294.6 kB 9.7 MB/s eta 0:00:00
Downloading tensorboard-2.15.2-py3-none-any.whl (5.5 MB)
5.5/5.5 MB 51.8 MB/s eta 0:00:00
Downloading tensorflow_estimator-2.15.0-py2.py3-none-any.whl (441 kB)
442.0/442.0 kB 13.8 MB/s eta 0:00:00
Downloading tensorflow_io_gcs_filesystem-0.36.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (5.1 MB)
5.1/5.1 MB 52.0 MB/s eta 0:00:00
Downloading termcolor-2.4.0-py3-none-any.whl (7.7 kB)
186.8/186.8 kB 6.6 MB/s eta 0:00:00
Downloading google_auth-2.27.0-py2.py3-none-any.whl (186 kB)
186.8/186.8 kB 6.6 MB/s eta 0:00:00
Downloading google_auth_oauthlib-1.2.0-py2.py3-none-any.whl (24 kB)
103.9/103.9 kB 3.6 MB/s eta 0:00:00
Downloading Markdown-3.5.2-py3-none-any.whl (103 kB)
103.9/103.9 kB 3.6 MB/s eta 0:00:00
Downloading tensorboard_data_server-0.7.2-py3-none-manylinux_2_31_x86_64.whl (6.6 MB)
6.6/6.6 MB 48.5 MB/s eta 0:00:00
Downloading werkzeug-3.0.1-py3-none-any.whl (226 kB)
226.7/226.7 kB 7.1 MB/s eta 0:00:00
Downloading cachetools-5.3.2-py3-none-any.whl (9.3 kB)
84.9/84.9 kB 2.9 MB/s eta 0:00:00
Downloading pyasn1-0.5.1-py2.py3-none-any.whl (84 kB)
84.9/84.9 kB 2.9 MB/s eta 0:00:00
Installing collected packages: libclang, flatbuffers, wrapt, werkzeug, termcolor, tensorflow-io-gcs-filesystem, tensorflow-estimator, tensorboard-data-server, pyasn1, protobuf, opt-einsum, oauthlib, ml-dtypes, markdown, keras, h5py, grpcio, google-pasta, gast, cachetools, astunparse, absl-py, rsa, requests-oauthlib, pyasn1-modules, google-auth, google-auth-oauthlib, tensorboard, tensorflow
WARNING: The script markdown.py is installed in '/usr/local/python/3.10.13/bin' which is not on PATH.
Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
WARNING: The scripts pyrsa-decrypt, pyrsa-encrypt, pyrsa-keygen, pyrsa-priv2pub, pyrsa-sign and pyrsa-verify are installed in '/usr/local/python/3.10.13/bin' which is not on PATH.
Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
WARNING: The script google-oauthlib-tool is installed in '/usr/local/python/3.10.13/bin' which is not on PATH.
Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
WARNING: The script tensorboard is installed in '/usr/local/python/3.10.13/bin' which is not on PATH.
Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
WARNING: The scripts estimator_cpkt_converter, import_pb_to_tensorboard, saved_model_cli, tensorboard, tf_upgrade_v2, tfite_convert, toco and toco_from_protos are installed in '/usr/local/python/3.10.13/bin' which is not on PATH.
Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed absl-py-2.1.0 astunparse-1.6.3 cachetools-5.3.2 flatbuffers-23.5.26 gast-0.5.4 google-auth-2.27.0 google-auth-oauthlib-1.2.0 google-pasta-0.2.0 grpcio-1.60.1 h5py-3.10.0 keras-2.15.0 libclang-16.0.6 markdown-3.5.2 ml-dtypes-0.2.0 oauthlib-3.2.2 opt-einsum-3.3.0 protobuf-4.25.2 pyasn1-0.5.1 pyasn1-modules-0.3.0 requests-oauthlib-1.3.1 rsa-4.9 tensorboard-2.15.2 tensorboard-data-server-0.7.2 tensorflow-2.15.0.py311 tensorflow-estimator-2.15.0 tensorflow-io-gcs-filesystem-0.36.0 termcolor-2.4.0 werkzeug-3.0.1 wrapt-1.14.1
```

```
pyparsing==3.1.1
python-dateutil==2.8.2
python-json-logger==2.0.7
pytz==2023.3.post1
PyYAML==6.0.1
pyzmq==25.1.2
referencing==0.32.1
requests==2.31.0
requests-oauthlib==1.3.1
rfc3339-validator==0.1.4
rfc3986-validator==0.1.1
rpds-py==0.16.2
rsa==4.9
scikit-learn==1.3.2
scipy==1.11.4
seaborn==0.13.1
Send2Trash==1.8.2
six==1.16.0
smmap==5.0.1
sniffio==1.3.0
soupsieve==2.5
stack-data==0.6.3
sympy==1.12
tenacity==8.2.3
tensorboard==2.15.2
tensorboard-data-server==0.7.2
tensorflow==2.15.0.post1
tensorflow-estimator==2.15.0
tensorflow-io-gcs-filesystem==0.36.0
termcolor==2.4.0
terminado==0.18.0
threadpoolctl==3.2.0
tinycss2==1.2.1
tomli==2.0.1
torch==2.1.2
tornado==6.4
traitlets==5.14.1
triton==2.1.0
types-python-dateutil==2.8.19.20240106
typing_extensions==4.9.0
tzdata==2023.4
uri-template==1.3.0
urllib3==2.0.7
wcwidth==0.2.13
webcolors==1.13
webencodings==0.5.1
websocket-client==1.7.0
Werkzeug==3.0.1
wrapt==1.14.1
```

● @RomanGorchakov → /workspaces/Py14 (develop) \$ pip freeze > requirements.txt

```

- pyasn1-modules=0.2.8=py_0
- pybind11-abi=4=hd3eb1b0_1
- pycosat=0.6.6=py311h5eee18b_0
- pycparser=2.21=pyhd3eb1b0_0
- pyjwt=2.4.0=py311h06a4308_0
- pyopenssl=23.2.0=py311h06a4308_0
- pysocks=1.7.1=py311h06a4308_0
- python=3.11.5=h7a1cb2a_0
- python-flatbuffers=2.0=pyhd3eb1b0_0
- re2=2022.04.01=h295c915_0
- readline=8.2=h5eee18b_0
- reproc=14.2.4=h295c915_1
- reproc-cpp=14.2.4=h295c915_1
- requests=2.31.0=py311h06a4308_0
- requests-oauthlib=1.3.0=py_0
- rsa=4.7.2=pyhd3eb1b0_1
- ruamel.yaml=0.17.21=py311h5eee18b_0
- scipy=1.11.4=py311h08b1b3b_0
- setuptools=68.0.0=py311h06a4308_0
- six=1.16.0=pyhd3eb1b0_1
- snappy=1.1.10=h6a678d5_1
- sqlite=3.41.2=h5eee18b_0
- tbb=2021.8.0=hdb19cb5_0
- tensorboard=2.12.1=py311h06a4308_0
- tensorboard-data-server=0.7.0=py311h52d8a92_0
- tensorboard-plugin-wit=1.6.0=py_0
- tensorflow=2.12.0=mkl_py311h34a0fa1_0
- tensorflow-base=2.12.0=mkl_py311he5f8e37_0
- tensorflow-estimator=2.12.0=py311h06a4308_0
- termcolor=2.1.0=py311h06a4308_0
- tk=8.6.12=h1ccaba5_0
- tqdm=4.65.0=py311h92b7b1e_0
- truststore=0.8.0=py311h06a4308_0
- typing_extensions=4.9.0=py311h06a4308_1
- tzdata=2023c=h04d1e81_0
- urllib3=1.26.18=py311h06a4308_0
- werkzeug=2.3.8=py311h06a4308_0
- wheel=0.41.2=py311h06a4308_0
- wrapt=1.14.1=py311h5eee18b_0
- xz=5.4.5=h5eee18b_0
- yaml-cpp=0.7.0=h295c915_1
- yar1=1.9.3=py311h5eee18b_0
- zlib=1.2.13=h5eee18b_0
- zstandard=0.19.0=py311h5eee18b_0
- zstd=1.5.5=hc292b87_0
- pip:
  - cryptography==41.0.4
  - pip==23.3.2
prefix: /opt/conda
@RomanGorchakov → /workspaces/Py14 (develop) $ conda env export > environment.yml

```

7. Выполняем коммит файлов в репозиторий Git в ветку разработки, сливаем её с веткой main и отправляем изменения на сервер GitHub.


```
● @RomanGorchakov → /workspaces/Py14 (develop) $ git add .
warning: adding embedded git repository: ClonedPy.git
hint: You've added another git repository inside your current repository.
hint: Clones of the outer repository will not contain the contents of
hint: the embedded repository and will not know how to obtain it.
hint: If you meant to add a submodule, use:
hint:
hint:   git submodule add <url> ClonedPy.git
hint:
hint: If you added this path by mistake, you can remove it from the
hint: index with:
hint:
hint:   git rm --cached ClonedPy.git
hint:
hint: See "git help submodule" for more information.
● @RomanGorchakov → /workspaces/Py14 (develop) $ git commit -m "Pip"
[develop ad999b5] Pip
4 files changed, 33340 insertions(+)
create mode 160000 ClonedPy.git
create mode 100644 environment.yml
create mode 100644 get-pip.py
create mode 100644 requirements.txt
● @RomanGorchakov → /workspaces/Py14 (develop) $ git checkout main
warning: unable to rmdir 'ClonedPy.git': Directory not empty
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
● @RomanGorchakov → /workspaces/Py14 (main) $ git merge develop
Updating daa232b..ad999b5
Fast-forward
 ClonedPy.git      |      1 +
 environment.yml   |     142 +
 get-pip.py        |    33036 ++++++
 requirements.txt   |      161 +
 4 files changed, 33340 insertions(+)
 create mode 160000 ClonedPy.git
 create mode 100644 environment.yml
 create mode 100644 get-pip.py
 create mode 100644 requirements.txt
● @RomanGorchakov → /workspaces/Py14 (main) $ git push -u
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 2 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 2.02 MiB | 6.33 MiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/RomanGorchakov/Py14
 daa232b..ad999b5  main -> main
branch 'main' set up to track 'origin/main'.
○ @RomanGorchakov → /workspaces/Py14 (main) $ ll
```

main

1 Branch

0 Tags

Go to file

Add file

Code

RomanGorchakov

Add files via upload

16c0f24 · 1 minute ago

5 Commits

ClonedPy.git	Pip	15 minutes ago
.gitignore	Create .gitignore	last week
LICENSE	Create LICENSE	last week
README.md	Create README.md	last week
environment.yml	Pip	15 minutes ago
get-pip.py	Pip	15 minutes ago
requirements.txt	Pip	15 minutes ago
ЛР2.14_ГорчаковРВ.pdf	Add files via upload	1 minute ago

README

MIT license

Информация обо мне

Имя: Роман.

Фамилия: Горчаков.

Отчество: Владимирович.

Название группы: ПИЖ-6-о-22-1.

Контрольные вопросы

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

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. Там также есть возможность выкладывать свои пакеты. Для скачивания и установки используется специальная утилита, которая называется `pip`.

2. Как осуществить установку менеджера пакетов `pip`?

Для установки `pip`, нужно скачать скрипт `get-pip.py` и выполнить его. 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?
`pip` устанавливает пакеты из основного индекса — PyPI.

4. Как установить последнюю версию пакета с помощью `pip`?

С помощью команды «`pip install ProjectName`».

5. Как установить заданную версию пакета с помощью `pip`?

С помощью команды «`pip install ProjectName==3.2`».

6. 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`».

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

С помощью команды «`pip install ./dist/ProjectName.tar.gz`».

8. Как удалить установленный пакет с помощью `pip`?

С помощью команды «`pip uninstall ProjectName`».

9. Как обновить установленный пакет с помощью `pip`?

С помощью команды «`pip install --upgrade ProjectName`».

10. Как отобразить список установленных пакетов с помощью `pip`?

С помощью команды «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». Виртуальные окружения были созданы с целью решить проблему обратной совместимости и проблему коллективной разработки.

12. Каковы основные этапы работы с виртуальными окружениями?

- Создание через утилиту нового виртуального окружения в отдельной папке для выбранной версии интерпретатора Python.
- Активация ранее созданного виртуального окружения для работы.
- Работа в виртуальном окружении, а именно управление пакетами, используя pip, и запуск выполнения кода.
- Деактивация виртуального окружения по окончании его работы.
- Удаление папки с виртуальным окружением, если оно нам больше не нужно.

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

Для создания виртуального окружения достаточно дать команду в формате «python3 -m venv <путь к папке виртуального окружения>». Чтобы активировать окружение под Linux и macOS нужно дать команду «source env/bin/activate». Чтобы переключиться с одного окружения на другое нам нужно выполнить команду деактивации «deactivate» и команду активации другого виртуального окружения «source <путь к папке виртуального окружения>/activate».

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

Для создания виртуального окружения достаточно дать команду в формате «virtualenv -p python3 env». Активация и деактивация такая же, как у стандартной утилиты Python.

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

Грубо говоря, `pipenv` можно рассматривать как симбиоз утилит `pip` и `venv` (или `virtualenv`), которые работают вместе, пряча многие неудобные детали от конечного пользователя.

Помимо этого, `pipenv` умеет:

- автоматически находить интерпретатор Python нужной версии (находит даже интерпретаторы, установленные через `pyenv` и `asdf`!);
- запускать вспомогательные скрипты для разработки;
- загружать переменные окружения из файла `.env`;
- проверять зависимости на наличие известных уязвимостей.

Для создания виртуального окружения достаточно дать команду в формате «`pipenv sync --dev`». Чтобы удалить виртуальное окружение, нужно дать команду «`pipenv --rm`».

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. В нём сохранены все пакетные зависимости виртуального окружения. Создать его можно с помощью команды «`pip freeze > 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?

Менеджер пакетов conda включен в состав дистрибутивов Anaconda и Miniconda. JetBrains включил этот инструмент в состав PyCharm.

19. Как создать виртуальное окружение conda?

С помощью команды «conda create -n %PROJ_NAME% python=3.7».

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

С помощью команд «conda activate %PROJ_NAME%» и «conda install <название пакета>» соответственно.

21. Как деактивировать и удалить виртуальное окружение conda?

С помощью команд «conda deactivate» и «conda remove -n %PROJ_NAME%» соответственно.

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

Файл environment.yml экспортирует параметры окружения, что позволяет воссоздать это самое окружение в любой нужный момент. Создать его можно с помощью команды «conda env export > environment.yml».

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

С помощью команды «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.

Поддержка Pip и Virtualenv в PyCharm появилась уже довольно давно. Иногда конечно возникают проблемы, но взаимодействие работает в основном стабильно.

Рассмотрим два варианта работы с виртуальными окружениями:

1) Создаём проект со своим собственным виртуальным окружением, куда затем будут устанавливаться необходимые библиотеки;

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

Убедитесь, что Anaconda или Miniconda загружена и установлена на ваш компьютер, и вам известен путь к ее исполняемому файлу. Для получения дополнительной информации обратитесь к инструкциям по установке. Нажмите кнопку выбора интерпретатора Python и выберите «Добавить новый интерпретатор». Выберите «Добавить локального переводчика». На левой панели диалогового окна «Добавить интерпретатор Python» выберите «Среда conda».

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

Для формирования и развертывания пакетных зависимостей.