

Текстовая расшифровка видео:

## КАК ОБУСТРОИТЬ РАБОЧЕЕ ОКРУЖЕНИЕ ДЛЯ PYTHON-ПРОЕКТА

План:

- Правила и ошибки;
- Virtualenv;
- Pyenv + virtualenv;
- Ipython/Jupyter [lab].

### Правила и ошибки

Никогда не ставьте пакеты через `~$ sudo pip install`, даже если в документации к проекту так написано.

Причины:

- Pip – это собственный встроенный пакетный менеджер питона. Иначе говоря, в современных системах есть свои пакетные менеджеры и pip с ними несовместим (используя его, можно сломать систему или ее компоненты, которые будет сложно восстановить).
- Существует понятие «Дополнительные виды атак» (тайпсквоттинг). Например, когда вы набираете пакет, вы ставите популярный requests-пакет. Часто при наборе команд люди могут допустить опечатку, чем могут воспользоваться мошенники. Они специально «заливают» пакеты с вирусным содержанием, которые отличаются от названий базовых пакетов (например, на одну букву). При наборе `sudo pip install` запускается этот произвольный код на машине с правами суперпользователя. Такой код непредсказуем.
- Установка пакетов на уровне системы мешает отслеживать зависимости конкретного проекта.

Исключение: использование `sudo pip install` не на физической машине, а внутри Docker-контейнера. В этом случае можно сделать все по правилам: создать внутри образа пользователя, ставить пакеты от этого пользователя. Также вы можете в любой момент удалить этот образ и пересобрать его заново, учтя все прошлые ошибки.





```
Alacrity meow-nofer - Thunar
├── __init__.py
├── _parser.py
├── __pycache__
│   ├── __init__.cpython-311.pyc
│   ├── _parser.cpython-311.pyc
│   ├── _re.cpython-311.pyc
│   └── _types.cpython-311.pyc
├── _re.py
├── _types.py
├── typing_extensions.py
├── zipp.py
├── version.py
├── wheel.py
├── windows_support.py
├── setuptools-65.5.0.dist-info
│   ├── entry_points.txt
│   ├── INSTALLER
│   ├── LICENSE
│   ├── METADATA
│   ├── RECORD
│   ├── REQUESTED
│   ├── top_level.txt
│   └── WHEEL
├── lib64 -> lib
└── pyvenv.cfg

180 directories, 1466 files
meow-nofer@pikachu ~/Experiments/slurm source s
[0] 0:zsh*
```

```
Alacrity meow-nofer - Thunar
meow-nofer@pikachu ~/Experiments/slurm source slurm_venv/bin/activate
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm pip freeze
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm
```

Например, если запустить эту команду и дать ей название, создастся локальный каталог, в котором будет много скопированных элементов из интерпретатора Python. Мы можем активировать виртуальное окружение с помощью команды «Source». Внутри можно посмотреть наличие пакетов по умолчанию и в случае чего поставить что-либо.

**Рекомендация:** в первую очередь команду `~$ pip install -U pip wheel setuptools` запускать внизу, тем самым обновляя встроенные специальные пакеты Python для установки других пакетов. Это уменьшит риск попадания зависимостей, собранных с более новыми версиями инсталляторов и т.д.

Пример:

```
meow-nofer@pikachu ~/Experiments/slurm source slurm_venv/bin/activate
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm pip freeze
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm pip install -U pip wheel setuptools
Requirement already satisfied: pip in ./slurm_venv/lib/python3.11/site-packages (22.3.1)
Collecting pip
  Using cached pip-23.1.2-py3-none-any.whl (2.1 MB)
Collecting wheel
  Using cached wheel-0.40.0-py3-none-any.whl (64 kB)
Requirement already satisfied: setuptools in ./slurm_venv/lib/python3.11/site-packages (65.5.0)
Collecting setuptools
  Using cached setuptools-67.7.2-py3-none-any.whl (1.1 MB)
Installing collected packages: wheel, setuptools, pip
  Attempting uninstall: setuptools
    Found existing installation: setuptools 65.5.0
    Uninstalling setuptools-65.5.0:
      Successfully uninstalled setuptools-65.5.0
  Attempting uninstall: pip
    Found existing installation: pip 22.3.1
    Uninstalling pip-22.3.1:
      Successfully uninstalled pip-22.3.1
Successfully installed pip-23.1.2 setuptools-67.7.2 wheel-0.40.0
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm pip freeze
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm pip install requests
```

```
Successfully installed pip-23.1.2 setuptools-67.7.2 wheel-0.40.0
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm pip freeze
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm pip install requests
Collecting requests
  Using cached requests-2.30.0-py3-none-any.whl (62 kB)
Collecting charset-normalizer<4,>=2 (from requests)
  Using cached charset_normalizer-3.1.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (197
Collecting idna<4,>=2.5 (from requests)
  Using cached idna-3.4-py3-none-any.whl (61 kB)
Collecting urllib3<3,>=1.21.1 (from requests)
  Using cached urllib3-2.0.2-py3-none-any.whl (123 kB)
Collecting certifi>=2017.4.17 (from requests)
  Using cached certifi-2023.5.7-py3-none-any.whl (156 kB)
Installing collected packages: urllib3, idna, charset-normalizer, certifi, requests
Successfully installed certifi-2023.5.7 charset-normalizer-3.1.0 idna-3.4 requests-2.30.0 urllib3-2.0.2
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm pip freeze
certifi==2023.5.7
charset-normalizer==3.1.0
idna==3.4
requests==2.30.0
urllib3==2.0.2
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm which python
/home/meow-nofer/Experiments/slurm/slurm_venv/bin/python
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm source deactivate
pyenv-virtualenv: deactivate slurm_venv
meow-nofer@pikachu ~/Experiments/slurm ls
paperclip slurm_venv
meow-nofer@pikachu ~/Experiments/slurm
[0] 0:zsh*
```

Важный нюанс: поскольку это метапакеты, они не появятся в списке установленных пакетов, так как это зависимости для развертывания, а не зависимости проекта.

Далее, мы можем установить, например, requests как зависимость, так у нас появятся pip freeze, requests и дополнительные пакеты, от которых он сам зависит. Таким образом все зависимости проекта будут в одном месте. Они будут существовать до тех пор, пока существует данный каталог. При этом мы всегда можем проверить какой именно Python будет запускаться. Если все сделано правильно и mkvirtualenv активирован, то команда which python покажет путь до бинарника Python, который находится внутри нашего виртуального окружения.

С таким подходом возникает ряд вопросов, например, нужно ли заливать каталог в систему контроля версий? Рекомендация: никогда так не делайте по причине разных проблем, которые могут возникнуть (например, совместимость).

**Экспертное мнение:** используйте инструменты, которые позволяют создавать mkvirtualenv в отдельном месте и обращаться к ним поименно, а не механизм из локальной папки.

#### **Инструменты:**

**Virtualenvwrapper** – классический инструмент, который заменяет наборы команд (~\$ python3 -m venv my\_venv; ~\$ source my\_venv/bin/activate; ~\$ source deactivate; ~\$ pip install -U pip setuptools) на свои упрощенные версии:

- Mkvirtualenv my\_venv;
- workon my\_venv;
- lsvirtualenv;
- Rmvirtualenv my\_venv.

### **Pyenv + virtualenv**

#### **Функции:**

- Управление разными версиями интерпретатора;
- Плагин позволяет создавать виртуальные окружения под конкретной версией интерпретатора;

Пример:

```
Successfully installed pip-23.1.2 setuptools-67.7.2 wheel-0.40.0
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm pip freeze
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm pip install requests
Collecting requests
  Using cached requests-2.30.0-py3-none-any.whl (62 kB)
Collecting charset-normalizer<4,>=2 (from requests)
  Using cached charset_normalizer-3.1.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (197 kB)
Collecting idna<4,>=2.5 (from requests)
  Using cached idna-3.4-py3-none-any.whl (61 kB)
Collecting urllib3<3,>=1.21.1 (from requests)
  Using cached urllib3-2.0.2-py3-none-any.whl (123 kB)
Collecting certifi>=2017.4.17 (from requests)
  Using cached certifi-2023.5.7-py3-none-any.whl (156 kB)
Installing collected packages: urllib3, idna, charset-normalizer, certifi, requests
Successfully installed certifi-2023.5.7 charset-normalizer-3.1.0 idna-3.4 requests-2.30.0 urllib3-2.0.2
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm pip freeze
certifi==2023.5.7
charset-normalizer==3.1.0
idna==3.4
requests==2.30.0
urllib3==2.0.2
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm which python
/home/meow-nofer/Experiments/slurm/slurm_venv/bin/python
(slurm_venv) meow-nofer@pikachu ~/Experiments/slurm source deactivate
pyenv-virtualenv: deactivate slurm_venv
meow-nofer@pikachu ~/Experiments/slurm ls
paperclip slurm_venv
meow-nofer@pikachu ~/Experiments/slurm vim ~/.zshrc
[0] 0:zsh*
```

```
~/ .zshrc
10
11 export WINEPREFIX=$HOME/.wine
12
13 export AIRFLOW_HOME=$HOME/Dropbox/Research/Udemy/Airflow
14
15 export VAULT_ADDR='http://127.0.0.1:8200'
16
17 export JAVA_HOME=/usr/lib/jvm/default
18 export HADOOP_HOME=/opt/hadoop3
19 export HADOOP_MAPRED_HOME=$HADOOP_HOME
20 export HADOOP_COMMON_HOME=$HADOOP_HOME
21 export HADOOP_HDFS_HOME=$HADOOP_HOME
22 export HADOOP_OPTS="$HADOOP_OPTS -Djava.library.path=$HADOOP_HOME/lib/native"
23 export YARN_HOME=$HADOOP_HOME
24 export SQOOP_HOME=/opt/sqoop
25 export KAFKA_HOME=/opt/kafka
26 export FLUME_HOME=/opt/flume
27 export HIVE_HOME=/opt/hive2
28 export HADOOP_OPTS="$HADOOP_OPTS -Djava.library.path=$HADOOP_HOME/lib/native"
29 export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
30 export PATH=$HOME/.pyenv/shims:$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin:$SQOOP_HOME/bin:$KAFKA_HOME/bin:$FLUME
31
32
33 alias vim=nvim
34 kelpie() {
NORMAL ~/ .zshrc zsh utf-8 4% 10
"~/ .zshrc" 210L, 6994B
[0] 0:nvim*
```

```
Alacritty meow-rofer - Thunar 66GB 18% 13% 59%
~/ .zshrc
34 kelpie() {
35     cmd="source /home/s-radex/.bashrc; /home/s-radex/bin/kelpie "
36     cmd+="$@"
37     docker exec -u s-radex:s-radex -t kelpie /bin/bash -c "${cmd}"
38 }
39 alias swagger="docker run --rm -it -e GOPATH=$HOME/Projects/go:/go -v $HOME:$HOME -w $(pwd) quay.io/goswagger
40 alias pbcopy='xclip -selection clipboard'
41 alias pbpaste='xclip -selection clipboard -o'
42
43 export LIBVA_DRIVER_NAME=vdpau
44 export SPARK_HOME=/opt/spark
45
46 mkvirtualenv(){ pyenv virtualenv $@; pyenv activate "${@: -1}" }
47 alias rmvirtualenv='pyenv virtualenv-delete'
48 alias lsvirtualenv='pyenv versions'
49 alias workon='pyenv activate'
50
51 lfcd () {
52     tmp="$(mktemp)"
53     lf -last-dir-path="$tmp" "$@"
54     if [ -f "$tmp" ]; then
55         dir="$(cat "$tmp")"
56         rm -f "$tmp"
57         [ -d "$dir" ] && [ "$dir" != "$(pwd)" ] && cd "$dir"
58     fi
59 }
NORMAL ~/ .zshrc zsh utf-8 22% 48/
[0] 0:nvim*
```

Можно использовать обертки `mkvirtualenv`. На примере в `zshrc` прописана `alias`, которая заменяет команды на команды `mkvirtualenv`. Не обязательно делать так же. Однако с помощью этого способа можно создавать окружения с нужной версией интерпретатора, удалять, активировать и т.д.

**Как это выглядит:**

```
meow-nofer@pikachu ~/Experiments/slurm ls  
paperclip slurm_venv  
meow-nofer@pikachu ~/Experiments/slurm vim ~/.zshrc  
meow-nofer@pikachu ~/Experiments/slurm lsvirtualenv
```

```
[0] 0:zsh*
```

```
Alacritty
```

```
3.10.9/envs/ans  
3.10.9/envs/bots  
3.10.9/envs/debugpy  
3.10.9/envs/dj  
3.10.9/envs/fast  
3.10.9/envs/geo  
3.10.9/envs/geodev  
3.10.9/envs/jup  
3.11.1  
aioes  
ans  
bots  
cookie  
dbtenv  
debugpy  
dj  
fake  
fast  
geo  
geodev  
hub  
i3  
jup  
neovim2  
neovim3  
pipe  
smtp
```

```
[0] 0:bash*
```

```
Alacritty meow-nofer - Thun
3.10.9/envs/ans
3.10.9/envs/bots
3.10.9/envs/debugpy
3.10.9/envs/dj
3.10.9/envs/fast
3.10.9/envs/geo
3.10.9/envs/geodev
3.10.9/envs/jup
3.11.1
aioes
ans
bots
cookie
dbtenv
debugpy
dj
fake
fast
geo
geodev
hub
i3
jup
neovim2
neovim3
pipe
smtp
meow-nofer@pikachu ~/Experiments/slurm mkvirtualenv 3.10.5 slurm_3
[0] 0:zsh*
smtp
meow-nofer@pikachu ~/Experiments/slurm mkvirtualenv 3.10.5 slurm_3
(slurm_3) meow-nofer@pikachu ~/Experiments/slurm python
Python 3.10.5 (main, Jul 19 2022, 13:06:39) [GCC 12.1.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> ^C
(slurm_3) meow-nofer@pikachu ~/Experiments/slurm source deactivate
pyenv-virtualenv: deactivate 3.10.5/envs/slurm_3
meow-nofer@pikachu ~/Experiments/slurm
[0] 0:zsh*
```

Мы можем посмотреть, какое окружение и версии Python есть в системе. Если необходимо использовать одну из многочисленных версий, то благодаря alias можно поставить нужную: `mkvirtualenv 3.10.5` (дать название). Pyenv проверит существование запрашиваемой версии и создаст виртуальное окружение, активировав его с нужной версией Python.

В остальных случаях работы все выглядит идентично.

- Можно настроить алиасы и использовать что-то наподобие `direnv`.

Пример:

```

1 # use a certain pyenv version
2 use_python() {
3     if [ -n "$(which pyenv)" ]; then
4         local pyversion=$1
5         pyenv local ${pyversion}
6     fi
7 }
8
9 layout_virtualenv() {
10    local pyversion=$1
11    local pvenv=$2
12    if [ -n "$(which pyenv-virtualenv)" ]; then
13        pyenv virtualenv --force --quiet ${pyversion} ${pvenv}-${pyversion}
14    fi
15    pyenv local --unset
16 }
17
18 layout_activate() {
19    if [ -n "$(which pyenv)" ]; then
20        source $PYENV_ROOT/versions/$1/bin/activate
21    fi
22 }

```

## In use

1. Create a project directory.

```
mkdir project_1
```

2. Create virtual environment:

```
pyenv virtualenv 3.6.7 venv-name
```

3. Create `.python-version` under `./project_1` with the virtual environment name created in step 2.

```
venv-name
```

4. Create `.envrc` under `./project_1` with environment variables declaration like: `export ENV_VARIBALE_NAME=bla`

5. Enter the project directory, run this command to trust current environment, you only need to run this at this first time and once `.envrc` is updated.

```
cd projects_1/
direnv allow
```

Now anytime you enter `./project_1/` or a directory under it, the virtualenv will automatically be activated, and environment variables will be setup.

6. For IDE virtualenv setup, the virtualenv is under `~/.pyenv/versions/virtualenv_name/`

**Direnv** – элемент, в котором вы создаете файл с конфигом. При заходе в каталог выполняются функции, автоматически активирующие нужное окружение.

Говоря об окружении, можно вспомнить «научный» интерпретатор Python, который называется «Anaconda».

**Anaconda** – «научный» дистрибутив Python с платной поддержкой и своими репозиториями пакетов.

Не всегда рекомендуется использовать, так как часто собранные пакеты могут быть несовместимы с версиями, собранными извне. Также проблемы, связанные с Anaconda не всегда легки в решении.

ipython/Jupyter [lab]

Инструментом для исследований аналитики чаще всего предпочитают Jupyter.

```
>>>
meow-nofer@pikachu ~/Experiments/slurm ipython
Python 3.11.3 (main, Apr 5 2023, 15:52:25) [GCC 12.2.1 20230201]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.13.2 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import os

In [2]: def foo(a, b):
...:     return a + b
...:

In [3]: def foo(a, b, c=5):
...:     return a + b + c
...:

In [4]: foo(1, 2)
Out[4]: 8

In [5]:
Do you really want to exit ([y]/n)? y
meow-nofer@pikachu ~/Experiments/slurm jupyter
[0] 0:zsh*
```

```
Alacritty meow-nofer - Thur
neovim2
neovim3
pipe
smtp
meow-nofer@pikachu ~/Experiments/slurm mkvirtualenv 3.10.5 slurm_3
(slurm_3) meow-nofer@pikachu ~/Experiments/slurm python
Python 3.10.5 (main, Jul 19 2022, 13:06:39) [GCC 12.1.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> ^C

(slurm_3) meow-nofer@pikachu ~/Experiments/slurm source deactivate
pyenv-virtualenv: deactivate 3.10.5/envs/slurm_3
meow-nofer@pikachu ~/Experiments/slurm python
Python 3.11.3 (main, Apr 5 2023, 15:52:25) [GCC 12.2.1 20230201] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import os
>>> def foo():
...     return 1 + 2
...
>>> import paperclip
KeyboardInterrupt
>>>
meow-nofer@pikachu ~/Experiments/slurm ipython
Python 3.11.3 (main, Apr 5 2023, 15:52:25) [GCC 12.2.1 20230201]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.13.2 -- An enhanced Interactive Python. Type '?' for help.

In [1]:
[0] 0:python*
```

Если посмотреть на Python интерактивный Shell, можно заметить целый ряд недостатков, связанных с отсутствием подсветки синтаксиса, с невозможностью многострочного редактирования и т.д. С этим помогает справиться **Ipython**. Он включает в себя подсветку синтаксиса, многострочное редактирование функций.

**Jupyter** более развесистый элемент, который позволяет запускать Ipython в браузере, в интерактивном режиме с возможностью визуальных представлений.

Jupyter существует в двух версиях: старой и новой.

**Первая версия** имеет название «Jupyter notebook». Запустим его в виртуальном окружении:

```
In [4]: foo(1, 2)
Out[4]: 8

In [5]:
Do you really want to exit ([y]/n)? y
meow-nofer@pikachu ~/Experiments/slurm jupyter notebook
^CTraceback (most recent call last):
  File "/usr/bin/jupyter-notebook", line 5, in <module>
    from notebook.notebookapp import main
  File "/usr/lib/python3.11/site-packages/notebook/notebookapp.py", line 84, in <module>
    from .gateway.managers import GatewayKernelManager, GatewayKernelSpecManager, GatewaySessionManager, GatewayClient
  File "<frozen importlib._bootstrap>", line 1178, in _find_and_load
  File "<frozen importlib._bootstrap>", line 1140, in _find_and_load_unlocked
  File "<frozen importlib._bootstrap>", line 1080, in _find_spec
  File "<frozen importlib._bootstrap_external>", line 1504, in find_spec
  File "<frozen importlib._bootstrap_external>", line 1476, in _get_spec
  File "<frozen importlib._bootstrap_external>", line 1631, in find_spec
KeyboardInterrupt

✖ meow-nofer@pikachu ~/Experiments/slurm workon jup
(jup) meow-nofer@pikachu ~/Experiments/slurm jupyter notebook

[0] 0:zsh*

from .gateway.managers import GatewayKernelManager, GatewayKernelSpecManager, GatewaySessionManager, GatewayClient
File "<frozen importlib._bootstrap>", line 1178, in _find_and_load
File "<frozen importlib._bootstrap>", line 1140, in _find_and_load_unlocked
File "<frozen importlib._bootstrap>", line 1080, in _find_spec
File "<frozen importlib._bootstrap_external>", line 1504, in find_spec
File "<frozen importlib._bootstrap_external>", line 1476, in _get_spec
File "<frozen importlib._bootstrap_external>", line 1631, in find_spec
KeyboardInterrupt

✖ meow-nofer@pikachu ~/Experiments/slurm workon jup
(jup) meow-nofer@pikachu ~/Experiments/slurm jupyter notebook
[W 13:47:38.648 NotebookApp] Loading JupyterLab as a classic notebook (v6) extension.
[C 13:47:38.648 NotebookApp] You must use Jupyter Server v1 to load JupyterLab as notebook extension. You have v2.1.0
You can fix this by executing:
  pip install -U "jupyter-server<2.0.0"
[I 13:47:38.682 NotebookApp] Serving notebooks from local directory: /home/meow-nofer/Experiments/slurm
[I 13:47:38.682 NotebookApp] Jupyter Notebook 6.5.2 is running at:
[I 13:47:38.682 NotebookApp] http://localhost:8888/?token=292ee25a8464adf3aa9c808d8ee4f6aace390f1a52d46c1c
[I 13:47:38.682 NotebookApp] or http://127.0.0.1:8888/?token=292ee25a8464adf3aa9c808d8ee4f6aace390f1a52d46c1c
[I 13:47:38.683 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 13:47:38.871 NotebookApp]

To access the notebook, open this file in a browser:
  file:///home/meow-nofer/.local/share/jupyter/runtime/nbserver-241953-open.html
Or copy and paste one of these URLs:
  http://localhost:8888/?token=292ee25a8464adf3aa9c808d8ee4f6aace390f1a52d46c1c
  or http://127.0.0.1:8888/?token=292ee25a8464adf3aa9c808d8ee4f6aace390f1a52d46c1c

[0] 0:python3.10*
```

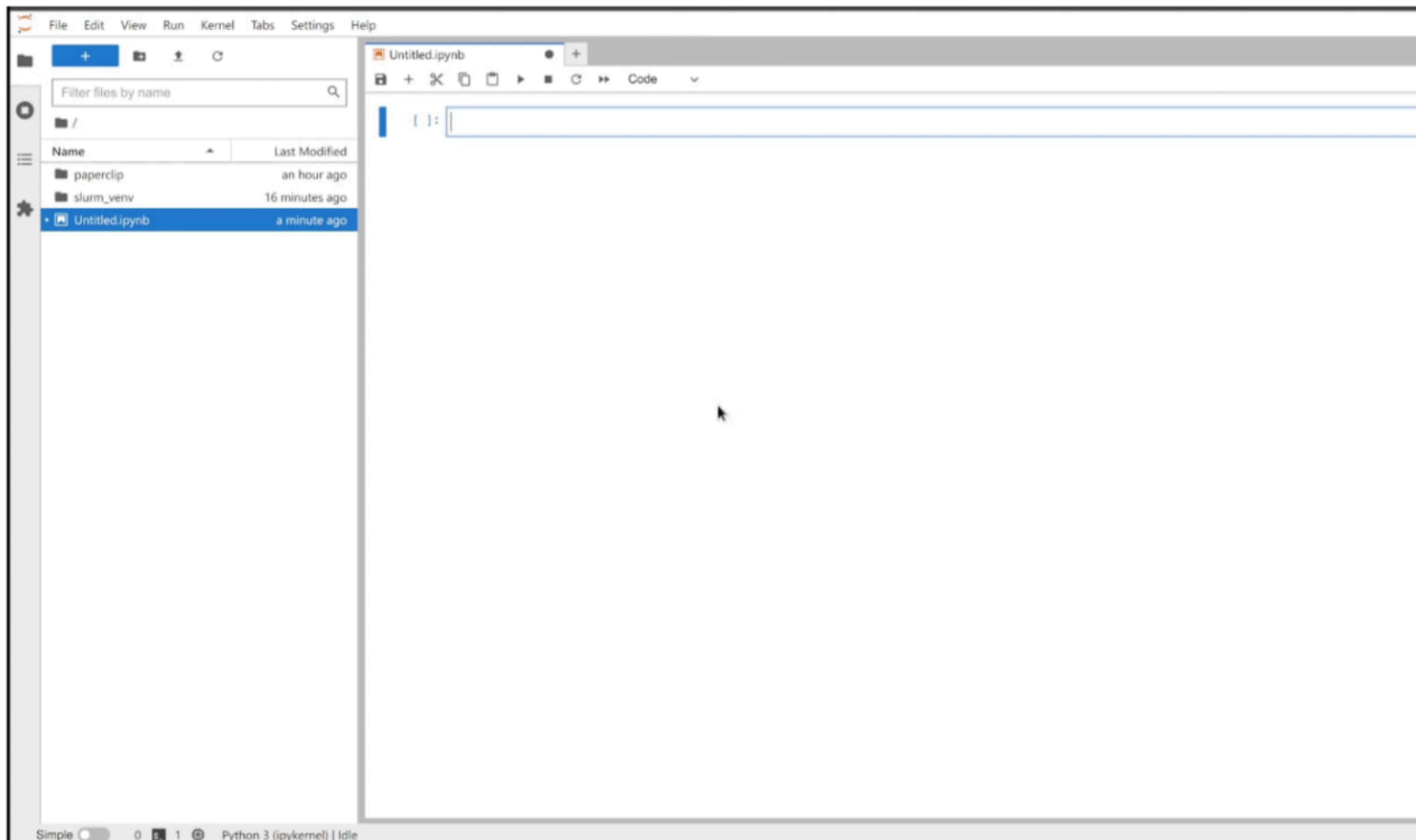
Jupyter notebook запустит сервер, интерпретатор Python, после чего откроется окно в браузере, в котором будет следующий интерфейс:



Далее, мы можем создать интерактивный блокнот (Ipython), где будут ячейки, которые можно выполнять.

**Вторая версия** (более новая) называется «Jupyter lab». Данная версия тоже ставится как дополнительный пакет.

Несмотря на фактически идентичную функциональность, Jupyter lab более удобна для работы:



**Нюанс:** существует большая разница между IDE разработки продуктовых решений на Python и исследованием проведения аналитики.

**Проблема:** код, находящийся в notebook, тяжело «вытащить».

Jupyter поддерживает разные плагины.

**Важно запомнить:** Jupyter notebook и Jupyter lab кардинально по-разному реализованы с точки зрения фронтенда. Реализовать один и тот же плагин, работающий везде одинаково, нереально. Необходимо будет создать две версии плагина и определять, где произошел запуск.

Как вам урок?



Далее >

Слёрм ©

[+7\(495\)248-05-80](tel:+7(495)248-05-80)

[Лицензия №ДЛ-1368 от 22.08.2019](#)

[Политика конфиденциальности](#)

[Публичная оферта](#)