Создание и использование виртуальных сред в Python3

Мо ержку создания "*облегченных виртуальных сред*". Каждая виртуальная среда имеет свой со hon и может иметь собственный независимый набор установленных пакетов Python в своих ка

Из создаются новые виртуальные среды Python, то для для определения путей внутри среды ve овки sysconfig. Когда Python работает в виртуальной среде, по умолчанию используется та же схема установки. Это означает, что нижестоящие дистрибьюторы могут изменить схему установки по умолчанию, без изменения поведения самих виртуальных сред. Сторонний код, который также может создавать новые виртуальные среды, должен делать то же самое.

Если в операционной системе по умолчанию используется Python 3 и не нравится функциональность встроенного модуля venv, то модуль virtualenv можно установить как скомпилированный пакет операционной системы.

```
# Для пользователей Ubuntu, Debian
$ sudo apt install python3-virtualenv
$ virtualenv --version
virtualenv 20.0.17 from /usr/lib/python3/dist-packages/virtualenv/__init__.py
```

Виртуальная среда - это среда Python, в которой установленный в ней интерпретатор Python, библиотеки и скрипты изолированы от других виртуальных сред и изолированы от любых библиотек, установленных в Python как часть операционной системы.

Обычные инструменты установки, такие как setuptools и pip работают в виртуальных средах как и ожидается. Другими словами, когда виртуальная среда активна, то пакеты Python устанавливаются в виртуальную среду без необходимости явно указывать это.

Когда виртуальная среда активна, т. е. работает интерпретатор Python виртуальной среды, атрибуты sys.prefix и sys.exec_prefix указывают на базовый каталог виртуальной среды, тогда как sys.base_prefix и sys.base_exec_prefix останутся указывать на базовую установку Python, ту, из которой была создана виртуальная среда. Если виртуальная среда не активна, то sys.prefix совпадает с sys.base_prefix, а sys.exec_prefix будет совпадать с sys.base_exec_prefix, все они указывают на установку Python не в виртуальной среде.

Когда виртуальная среда активна, любые параметры, которые изменяют путь установки, будут игнорироваться во всех файлах конфигурации distutils, чтобы предотвратить случайную установку проектов вне виртуальной среды.

Работая в командной оболочке, пользователи могут активизировать виртуальную среду, запустив сценарий активации в каталоге исполняемых файлов виртуальной среды source <venv>/bin/activate, который добавляет каталог виртуальной среды для исполняемых файлов в переменную окружения PATH для работающей оболочки.

Создание виртуальной среды в OS Linux/Windows.

Создание виртуальной среды в OS Linux выполняется с помощью команды venv:

```
# в виртуальную среду установится системная версия Python3
# т.е. версия установленная по умолчанию в вашей системе
$ python3 -m venv /path/to/new/virtual/environment --prompt Python3.9
```

Вверх

Представленная выше команда создает целевой каталог environment, создавая все не существующие родительские каталоги. После этого python поместит в него файл pyvenv.cfg с ключом, который будет указывать на ту версию Python, для которой запущена эта команда. Она также создает подкаталог bin в Unix или Scripts в Windows, содержащий копию/символическую ссылку двоичных файлов Python в зависимости от платформы или аргументов, используемых во время создания среды. Вышеуказанная команда также создает изначально пустой подкаталог lib/pythonX.Y/site-packages в Unix, в Windows это Lib\\site-packages. Если указан существующий каталог виртуальной среды, то он будет использован повторно.

Если, например, вы собрали Python3.11 из исходников в папку /opt/python-3.11.0/ и хотите именно его поставить в виртуальную среду, то команда будет выглядеть следующим образом:

```
# в виртуальную среду установится версия Python
# расположенная по пути /opt/python-3.11.0/bin/python3.11
$ /opt/python-3.11.0/bin/python3.11 -m venv ~/.python3.11.0 --prompt Python3.11
```

Представленная выше команда установит версию Python, который был собран из исходников в каталоге /opt/python-3.11.0/, в папку виртуальной среды ~/.python3.11.0. Папка виртуальной среды будет расположена в скрытой (точка впереди названия папки) домашней (~/) директории пользователя. Параметр --prompt Python3.11 создаст, после активации, дополнительную подсказку о том, что вы находитесь в виртуальной среде Python3.11.

Активировать установленный Python в такой виртуальной среде можно командой:

```
$ source ~/.python3.11.0/bin/activate
# в скобках - та самая подсказка, о которой говорилось выше
(Python3.11) :~$
```

Установить дополнительные модули/пакеты для этой виртуальной среды можно только если она (виртуальная среда) активирована. Например, установка фреймворка pytest будет выглядеть следующим образом:

```
# команда установит модуль `pytest`
# в папку виртуальной среды ~/.python3.11.0
(Python3.11) :~$ python3 -m pip install -U pytest
```

Для **создания виртуальной среды в OS Windows**, необходимо вызвать команду venv следующим образом:

```
c:\>c:\Python39\python -m venv c:\path\to\myenv --prompt Python3.9
```

В качестве альтернативы, если настроены переменные PATH и PATHEXT для установки Python:

```
c:\>python3 -m venv c:\path\to\myenv
```

Для venv может быть задано несколько путей, и в этом случае будет создана идентичная виртуальная среда в соответствии с заданными параметрами для каждого указанного пути.

После создания виртуальной среды ее можно "активировать" с помощью сценария, расположенном каталоге виртуальной среды. Вызов сценария зависит от платформы:

Платформа	Оболочка	Команда активации виртуальной среды
POSIX	bash/zsh	$ source venv/bin/activate
	fish	$. venv/bin/activate.fish
	csh/tcsh	$ source venv/bin/activate.csh
	PowerShell Core	$ venv/bin/Activate.ps1
Windows	cmd.exe	C:> venv\Scripts\activate.bat
	PowerShell	PS C:> venv\Scripts\Activate.ps1

Примечание к таблице: venv должен быть заменен путем к каталогу, содержащему виртуальную среду:

Не нужно специально активировать среду если сценарий запускается с указанием полного пути до интерпретатора Python, ус[Вверх]нного в виртуальную среду. Активация просто добавляет каталог виртуальной среды к пути интерпретатора Python тем самым давая возможность запускать сценарии не используя полный путь.

Вы можете "деактивировать" виртуальную среду, набрав команду deactivate в своей оболочке. Точный механизм зависит от платформы и является внутренней деталью реализации. Обычно используется скрипт или функция оболочки.

Примеры:

Установка виртуальной среды в директорию env:

```
$ python3 -m venv ~/env
```

Активация виртуальной среды env:

```
$ source ~/env/bin/activate
(env) docs-python@IdeaCentre:~$
```

Установка пакетов в виртуальную среду env:

```
(env) $ python3 -m pip install pymysql

# или

(env) $ pip install pymysql
```

Выполнение сценариев в виртуальной среде env:

```
(env) $ python ~/env/test.py
```

Деактивация виртуальной среды:

```
(env) $ deactivate
# произойдет выход из виртуального окружения
$
```

Не нужно специально активировать среду если сценарий запускается с указанием полного пути до интерпретатора Python:

```
$ ~/env/bin/python3 ~/env/test.py
```

Если команда python3 -m venv запускается с опцией -h, то можно увидеть доступные опции:

```
$ python3 -m venv -h
usage: venv [-h] [--system-site-packages] [--symlinks | --copies] [--clear]
            [--upgrade] [--without-pip] [--prompt PROMPT]
            ENV_DIR [ENV_DIR ...]
...
...
```

Позиционные аргументы:

- ENV_DIR - Каталог для создания виртуальной среды.

Необязательные аргументы:

- -h, --help - показать справочное сообщение и выйти,
- --system-site-packages - предоставляет виртуальной среде доступ к системным пакетам site-packages,
- --symlinks - при установке виртуальной среды создает символические ссылки вместо копирования файлов,
- --copies - при установке виртуальной среды копирует файлы вместо создания символических ссылок,
- --clear - перед созданием виртуальной среды удаляет содержимое каталога, если он уже существует,
- --upgrade - Обновите каталог виртуальной среды, чтобы использовать установленную версию Python,
- --without-pip - пропускает установку или обновление pip в виртуальной среде (pip загружается по умолчанию),
- --prompt PROMPT - предоставляет альтернативный префикс приглашения для этой среды.

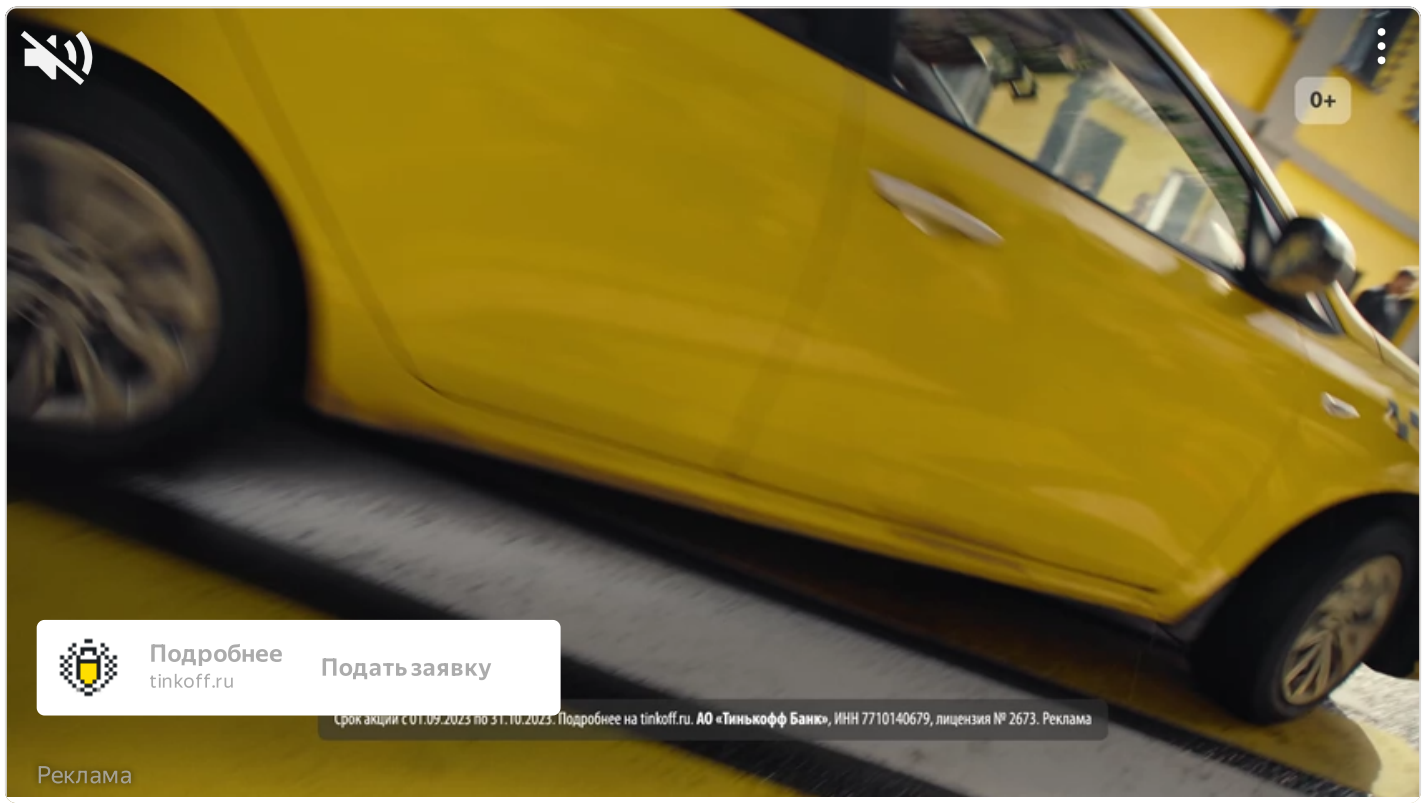
Примечания:

- Хотя в Windows поддерживаются символические ссылки, они не рекомендуются. Особо следует отметить, что двойной щелчок по python.exe в проводнике файлов разрешит символическую ссылку и проигнорирует виртуальную среду.

- В Microsoft Windows может потребоваться включить сценарий Activate.ps1, установив политику выполнения для пользователя. Вы можете сделать это, введя следующую команду PowerShell:PS C: Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser

Созданный файл pyvenv.cfg также содержит ключ include-system-site-packages, для которого установлено значение true, если venv запускается с параметром --system-site-packages, в противном случае значение будет false.

Если не указана опция --without-pip, то будет вызываться [модуль ensurepip](#) для начальной загрузки pip в виртуальную среду.



Вверх