Компьютерная лингвистика.
Подготовка среды обработки данных

1. Установка PyCharm

Для установки PyCharm Professional необходимо перейти на официальный сайт JetBrains по ссылке https://www.jetbrains.com/ru-ru/pycharm/download/#section=windows и выбрать данную версию для скачивания.

2. Установка VirtualBox

VirtualBox (Oracle VM VirtualBox) — программный продукт виртуализации для операционных систем Microsoft Windows, Linux, FreeBSD, macOS. Установщик VirtualBox по ссылке https://download.virtualbox.org/virtualbox/6.1.2/VirtualBox-6.1.2-135663-Win.exe.

3. Установка Vagrant в ОС Windows

3.1. Установка Vagrant и загрузка окружения.

Vagrant свободное и открытое программное обеспечение для создания и конфигурирования виртуальной среды разработки

Внимание! На время установки Vagrant лучше отключить антивирус, так как возможны ошибки.

Для установки Vagrant необходимо скачать установщик https://releases.hashicorp.com/vagrant/2.2.7/vagrant_2.2.7_x86_64.msi с официального сайта и запустить его.

После того, как Vagrant установлен, мы переходим к созданию рабочего окружения. Основа его построения — «коробка» Vagrant (box). «Коробка» Vagrant представляет собой файл с расширением box (архив tar). Внутри архива хранятся образ виртуальной машины и файлы, необходимые для ее корректного запуска.

На официальном сайте Vagrant есть значительное число готовых «коробок» с различными версиями Linux и установленным ПО. Их список можно посмотреть на www.vagrantbox.es.

Для наших целей подойдет ubuntu/bionic64 из официального репозитория. Перед тем как устанавливать окружение, создадим директорию, где будет храниться файл с конфигурацией для нее. Назовем папку «Vagrant». Переходим в эту директорию и выполняем в консоли:

```
\Vagrant> vagrant init ubuntu/bionic64
```

После этого в директории появится файл Vagrantfile, в котором содержится описание конфигурации данного окружения. Например, строчка config.vm.box = «ubuntu/bionic64» определяет имя нашего окружения. Теперь все готово к тому, чтобы приступить к запуску окружения. Выполним

```
\Vagrant> vagrant up
```

и увидим сообщение о загрузке соответствующей «коробки». Т.к. это первый запуск данного окружения, перед стартом его надо скачать из репозитория на локальный диск.

Внимание! В BIOS необходимо включить «Intel Virtualization Technology».

После того как мы выполнили «vagrant up» и дождались загрузки окружения, оно готово к использованию, в чем можно убедиться, проверив его статус:

```
\Vagrant> vagrant status
```

Для завершения работы окружения используется команда

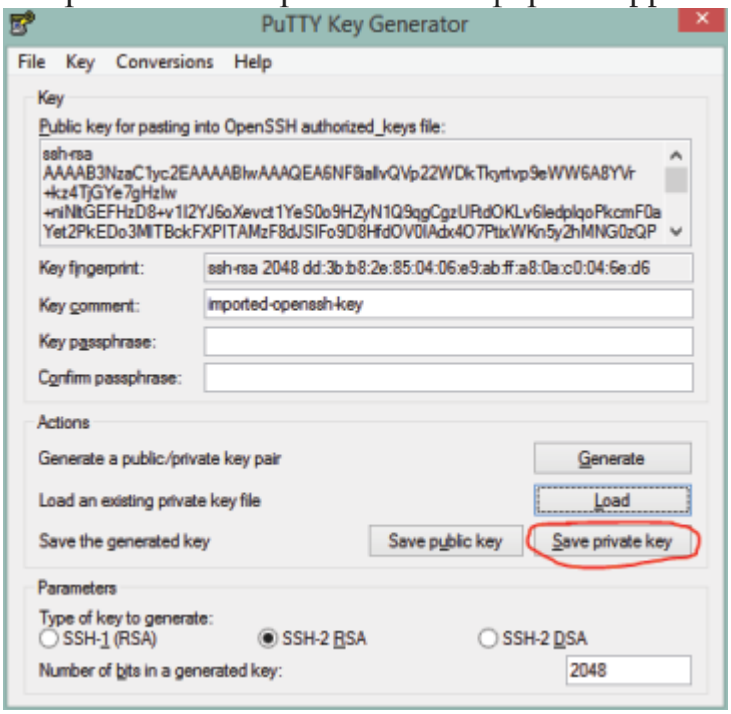
```
\Vagrant> vagrant halt
```

Все установленные окружения Vagrant можно просмотреть с помощью vagrant box list или непосредственно в VirtualBox (VMware).

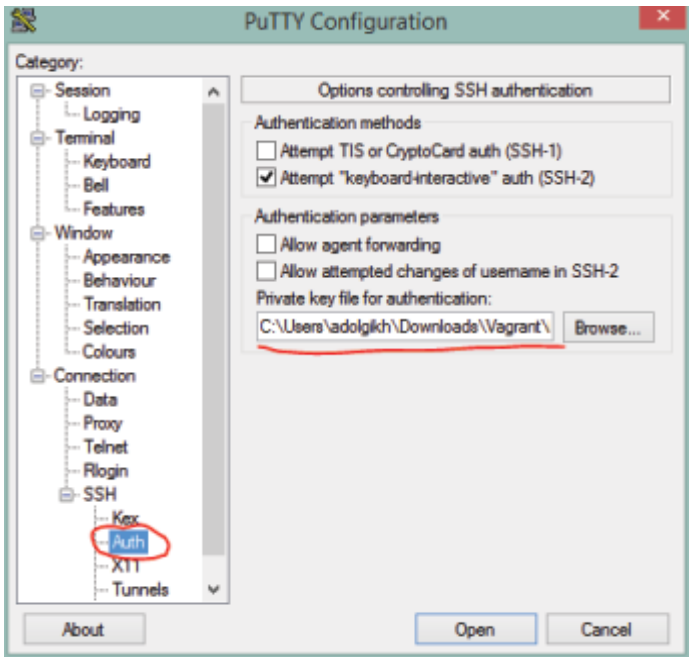
Также отметим, что окружение Vagrant может быть добавлено с помощью команды vagrant box add. Подробнее о ней и о других консольных командах Vagrant можно прочитать на странице официальной документации http://docs.vagrantup.com/v2/cli/index.html.

3.2. Подключение к окружению Vagrant

При старте окружения (vagrant up) в папке .vagrant автоматически создается ключ private_key для подключения к окружению по протоколу ssh. Для использования протокола ssh в Windows нам потребуются программы PuTTY (https://putty.org.ru/download.html) и PuTTYgen (для конвертирования ключа в формат, который PuTTY понимает). Запускаем PuTTYgen, открываем insecure_private_key и видим сообщение, что ключ был успешно импортирован. Теперь можно сохранить его в формате ppk как private key.



Из соображений удобства ключ можно сохранить в папку с файлом Vagrantfile для данного окружения. Указываем PuTTY, какой ключ использовать, и подключаемся через порт 2222 (номер порта отображается при выполнении vagrant up). По умолчанию для подключения используется имя (login) — vagrant.



Если подключение прошло удачно, то окажемся в консоли нашего Vagrant-окружения, откуда можно устанавливать библиотеки Python или системные приложения.

3.3. Установка Python

Теперь, когда рабочие окружения скачаны и запущены, все готово к установке Python. Хорошая практика — использование виртуальных окружений для работы с Python. В этом случае риск конфликтов с уже установленными в системе версиями интерпретатора и прочего ПО сводится к минимуму и появляется возможность одновременной работы с разными версиями интерпретатора Python, Django или любого другого пакета.

Для установки виртуального окружения выполняем в консоли Vagrant:

```
sudo apt-get update
sudo apt-get install python-virtualenv
```

После завершения установки создаем новое виртуальное окружение

```
virtualenv venv
```

активируем его

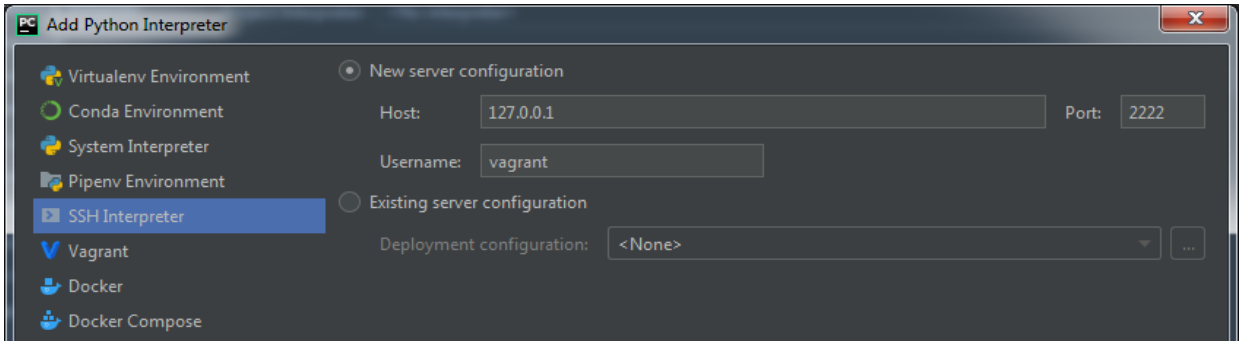
```
source /home/vagrant/venv/bin/activate
```

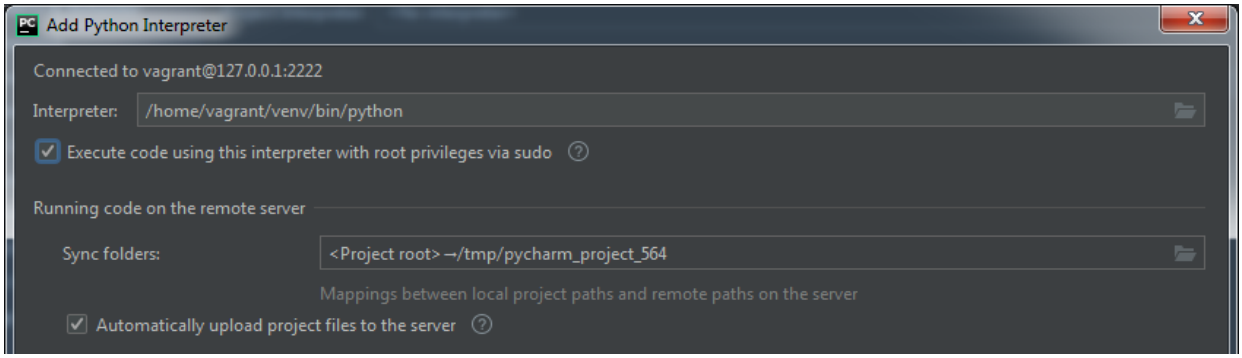
Теперь необходимо научить PyCharm работать с интерпретатором Python, установленным в окружении Vagrant.

3.4. Настройка PyCharm

Запустим PyCharm и создадим новый проект. По умолчанию он создается в папке ~/PycharmProjects/. Пусть это будет проект Test. Далее в домашней директории рабочего окружения Vagrant создадим папку PycharmProjects (совпадение имен случайно и выбрано из соображений удобства).

Переходим в меню настроек проекта: File->Settings->Project Interpeter. Здесь добавляем новый интерпретатор: указываем его адрес (127.0.0.1), номер порта (2222), имя пользователя (vagrant), путь к файлу с ключом для ssh и, наконец, указываем путь к интерпретатору /home/vagrant/venv/bin/python.





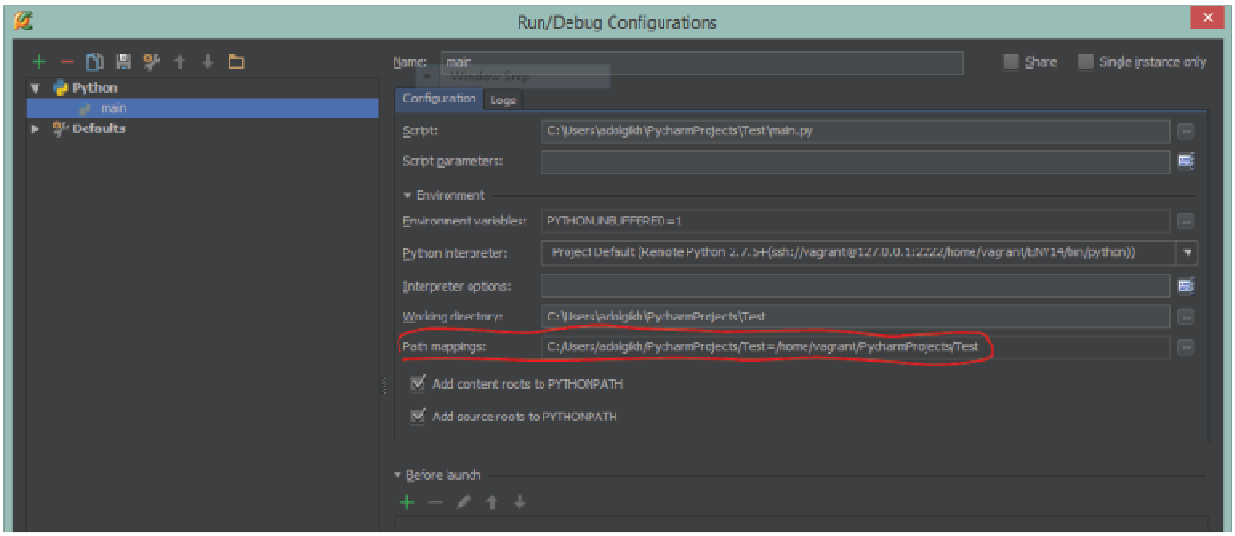
Теперь необходимо настроить синхронизацию локальной папки с проектами и аналогичной папки в окружении Vagrant. Открываем для редактирования файл Vagrant и раскомментируем в нем строку с config.vm.synced_folder, исправляя ее на config.vm.synced_folder C:/Users/ИмяПользователя/PycharmProjects, /home/vagrant/PycharmProjects.

Теперь локальные файлы будут синхронизироваться с окружением, в котором установлен интерпретатор. Перезапускаем Vagrant

```
$  vagrant reload
```

и видим, что наша папка с проектом Test была успешно добавлена в папку с проектами в удаленном рабочем окружении.

Чтобы текущие изменения, сделанные по ходу работы над проектом, сразу отправлялись в удаленную среду, необходимо сделать дополнительные настройки в проекте Pycharm. Заходим в меню Run->Edit Configuration и корректируем Path Mapping.



Упоминания заслуживает возможность запуска ssh-консоли в Pycharm. Она запускается из меню Tools и подключается по ssh к окружению, которое указано в настройках удаленного интерпретатора. Таким образом, нам нет необходимости использовать PuTTY с корректно настроенным проектом в Pycharm.