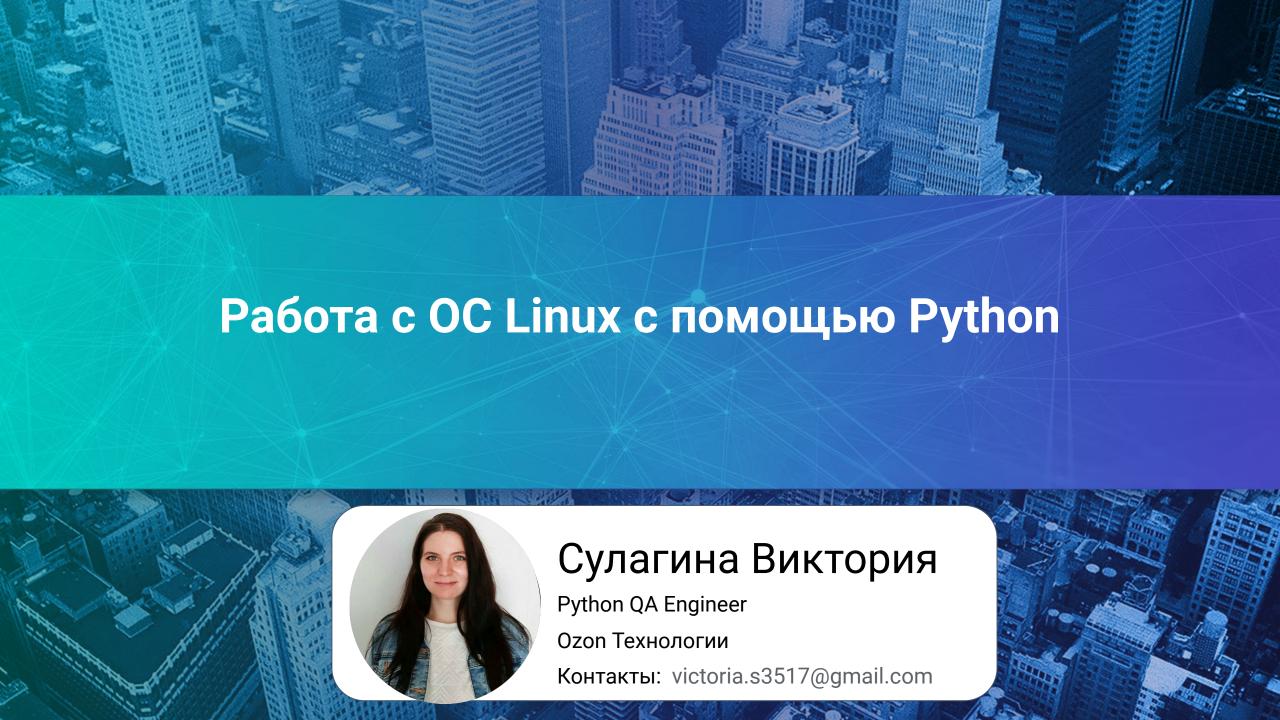




Проверить, идет ли запись!





Преподаватель



Сулагина Виктория

- Ozon Технологии Python QA Engineer
- Epam, Positive Technologies Software Engineer in Test
- Занимаюсь автоматизацией тестирования на Python более 3-х лет, опыт работы с Python более 5 лет
- Пишу тесты на API (Rest, GraphQI) и UI
- Провожу обучение коллег практикам тестирования и автоматизации

Правила вебинара



Активно участвуем



Задаем вопрос в чат или голосом

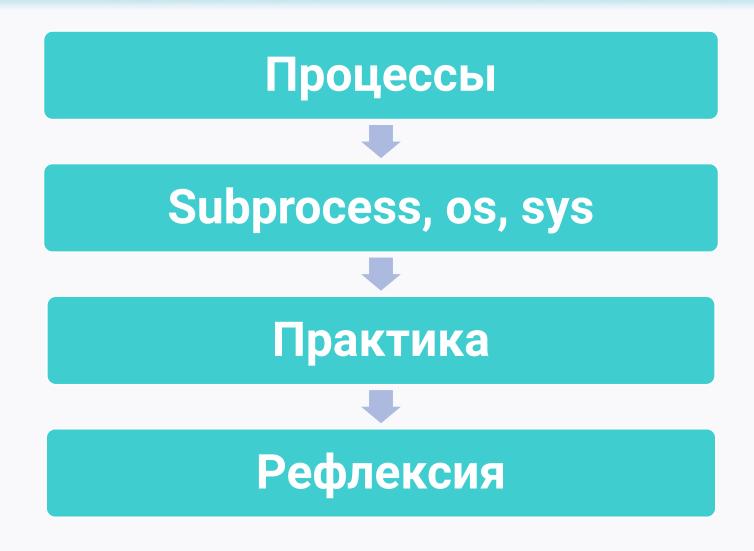


Off-topic обсуждаем в Slack #канал группы или #general



Вопросы вижу в чате, могу ответить не сразу

Маршрут вебинара



Цели вебинара | После занятия вы сможете

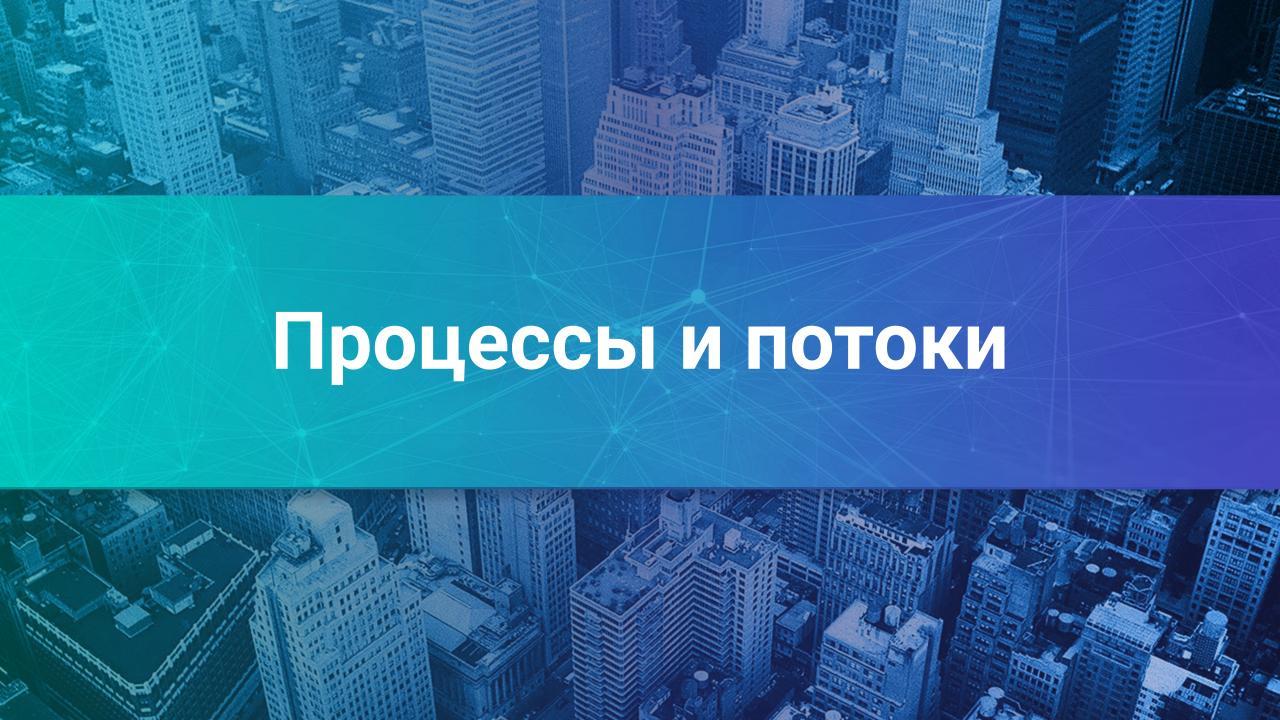
Работать с процессами os из Python кода

Работать с файлами и директориями внутри фреймворка и на сервере

Смысл | Зачем вам это уметь

Подготовка, поддержка и отладка окружения для автотестов

Подготовка вспомогательных скриптов для тестирования систем и мониторинга приложений



Процессы

Процесс - запущенная программа, приложение или команда.

Некоторые приложения могут иметь несколько процессов (объединены в группы, группа имеет свой идентификатор, обычно - это идентификатор первого процесса в группе)

Процесс имеет идентификатор - PID(PPID - идентификатор родительского процесса)

Посмотреть полный список процессов: **ps -ef**

Процессы

Какие способы межпроцессного взаимодействия вы знаете?

Способы межпроцессного взаимодействия

- PIPE
- Сокеты
- Коды возврата
- Сигналы
- Переменные окружения

Сигналы

- SIGKILL
- SIGTERM
- SIGSTOP

Посмотреть полный список сигналов: kill -L

Коды ответа

- 0 -- Bce OK
- 1 -- Разные ошибки
- 126 -- Невозможно выполнить
- 127 -- Не найдено
- 128 -- Неверный аргумент
- 130 -- Завершение по control-c
- 255+ -- За пределами диапазона

Процессы

- fork -- создает новый процесс из переданного.
- ехес -- запускает процесс.
- sleep -- усыпляет.
- kill -- посылает сигнал (SIGTERM, SIGKILL, etc)
- рѕ -- список процессов

Поток - это процесс, который использует некоторые ресурсы совместно с другими процессами. Поток имеет определенную структуру и для ядра представляется обычным процессом.

7

Какие потоки ввода и вывода вам известны и как их можно перенаправить?

- stdout
- stderr
- stdin

- > -- перенаправить вывод
- < -- перенаправить ввод
- 2> -- перенаправить ошибки
- 2>&1 -- вывод и ошибки
- >> \ << -- перенаправление с перезаписью

dev/null



subprocess дает разработчику возможность запускать процессы из Python кода, получать соединение с потоками стандартного ввода, стандартного вывода, и получение кодов возврата от этих процессов.

subprocess.call (args, *, stdin=None, stdout=None, stderr=None,
shell=False, cwd=None, timeout=None)

- Выполняет команду переданную в args
- Блокирует выполнёние скрипта и ждёт выполнения команды Возвращает только код возврата

subprocess.check_call (args, *, stdin=None, stdout=None, stderr=None,
shell=False, cwd=None, timeout=None)

- Выполняет команду переданную в args
- Блокирует выполнёние скрипта и ждёт выполнения команды Если команда вернула не 0, бросит CalledProcessError

subprocess.check_output (args, *, stdin=None, stderr=None, shell=False, cwd=None, encoding=None, errors=None, universal_newlines=None, timeout=None, text=None)

- Выполняет команду переданную в args
- Блокирует выполнение скрипта и ждет выполнения команды Если команда вернула не 0, бросит CalledProcessError Если команда вернула 0 вернет stdout

subprocess.Popen — класс для гибкого выполнения команд. Не блокирует выполнение скрипта.

Popen.poll -- проверяет статус процесса запущенного Popen.

Popen.wait -- устанавливает таймаут для запущенного процесса, блокирует выполнение скрипта, если процесс не уложился, бросает TimeoutExpired.

Popen.communicate -- передает данные в stdin процесса, возвращает stdout, stderr.

Popen.send_signal -- отправляет сигнал процессу

Popen.terminate -- отправляет SIGTERM процессу

Popen.kill -- отправляет SIGKILL процессу



sys

sys -- предоставляет доступ к некоторым переменным окружения используемых интерпретатором.

sys

sys.argv -- список аргументов командной строки, переданных скрипту (первый имя скрипта).

sys.executable -- строка, полный путь до интерпретатора Python.

sys.path -- список путей по которым питон ищет модули. Изменяемая копия PYTHONPATH.

sys.platform -- информация об ос

sys.exit -- завершит выполнение с переданным кодом. Бросит SystemExit (finally блоки будут выполнены)

sys

Управление потоками ввода/вывода:

sys.stdin sys.stdout sys.stderr



os -- модуль представляет множество функций для работы с ос, причем их поведение не зависит от ос

- os.name -- семейство ОС (posix, nt, java)
- os.environ -- словарь с переменными окружения
- os.environb -- байтовая версия os.environ
- os.getcwd -- текущая директория (откуда запущен скрипт)
- os.getlogin -- текущий пользователь
- os.path -- модуль для безопасной работы с путями
- os.chmod -- установить права на файл
- os.listdir -- список файлов в папке
- os.mkdir создать папку

Слайд с тезисами

Работа с процессами через Python - не всегда самое рациональное решение

Нужно учитывать универсальность написанных скриптов для запуска на разных системах

> Одна мысль на слайде

без картинок

Использование цифр для опроса

)-9



Домашнее задание

Написать Python скрипт, который выводит:

- 1. Список всех процессов
- 2. Информацию о конкретном процессе
- 3. Список в файлов в директории (указать директорию)
- 4. Текущую директорию и список в файлов в ней
- 5. Версию ядра
- 6. Версию операционной системы

Условия:

Пользоваться только модулями subprocess, os, sys и bash внутри subprocess



Пройдите, пожалуйста, опрос в Чате с преподавателем после приёмки вашего ДЗ

Рефлексия



С какими основными мыслями и инсайтами уходите с вебинара?



Список материалов для изучения

- процессы в Linux:
 http://www.linuxcenter.ru/lib/articles/system/linux_processes.phtml
- subprocess: https://docs.python.org/3/library/subprocess.html
- os: https://docs.python.org/3/library/os.html
- sys: https://docs.python.org/3/library/sys.html

