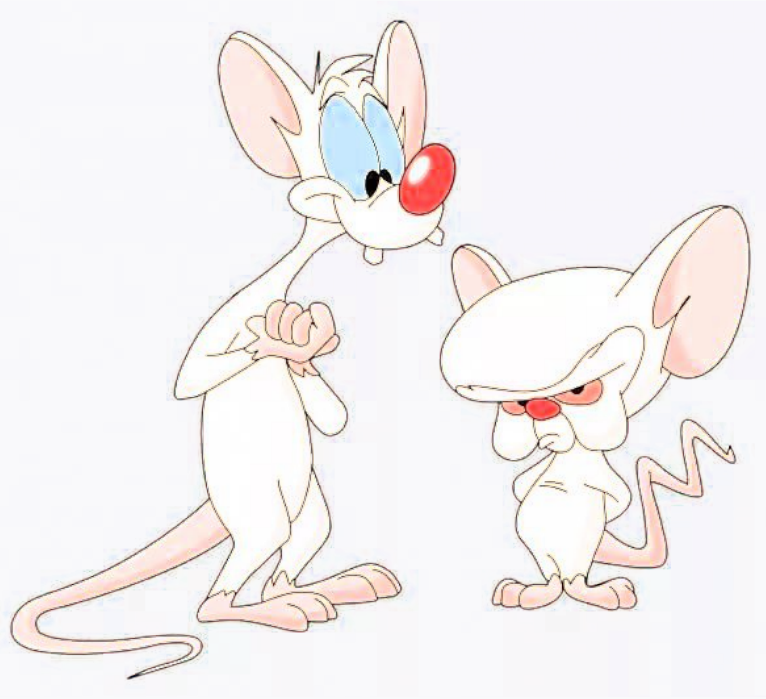
Клиент-серверные приложения на Python

Урок 5

Логирование

Журналирование событий и модуль logging.

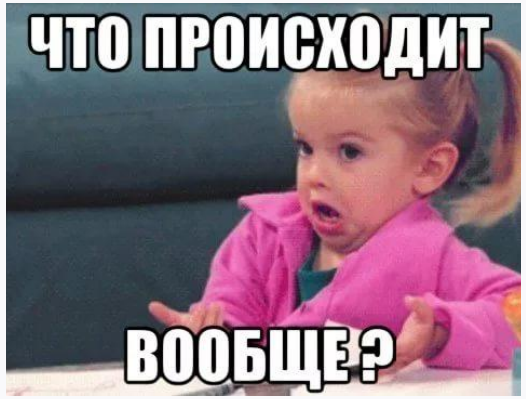
Цели урока



1. Понять, какие задачи решает логирование;
2. Как логирование реализуется на практике;
3. Модуль logging в Python и особенности его использования.

Журналирование (логирование)

Журналирование. Зачем?



- показать, что система делает прямо сейчас, не прибегая к помощи отладчика;

- выяснение обстоятельств, которые привели к определенному состоянию системы (отказ, ошибка);

- анализ расхода времени/ресурсов (профилирование).

Модуль `logging`.
Уровни логирования (важности)

Уровень	Значение	Описание	Метод
`CRITICAL`	50	Критические ошибки/сообщения	`log.critical(fmt [, *args [, exc_info [, extra]]])`
`ERROR`	40	Ошибки	`log.error(fmt [, *args [, exc_info [, extra]]])`
`WARNING`	30	Предупреждения	`log.warning(fmt [, *args [, exc_info [, extra]]])`
`INFO`	20	Информационные сообщения	`log.info(fmt [, *args [, exc_info [, extra]]])`
`DEBUG`	10	Отладочная информация	`log.debug(fmt [, *args [, exc_info [, extra]]])`
`NOTSET`	0	Не установлен	

Модуль logging. Настройка

1. Создать логгер (**getLogger()**), установить уровень важности.
2. Создать обработчик (**FileHandler/SocketHandler/...**), установить уровень важности.
3. Создать объект **Formatter**, подключить его к обработчику (**setFormatter()**).
4. Подключить обработчик **Handler** к объекту **Logger** (**addHandler()**).

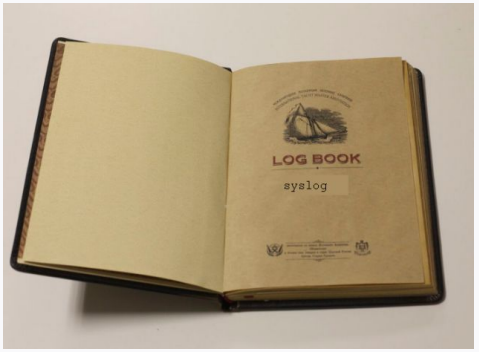
Модуль logging. Настройка

```
import logging

logger = logging.getLogger('app.main')

formatter = logging.Formatter("%(asctime)s - %(levelname)s - %(message)s ")

# Создать файловый обработчик логгирования (можно задать кодировку):
fh = logging.FileHandler("app.log", encoding='utf-8')
fh.setLevel(logging.DEBUG)
fh.setFormatter(formatter)

# Добавить в логгер новый обработчик и установить уровень логгирования
logger.addHandler(fh)
logger.setLevel(logging.DEBUG)

logger.info('Замечательный день для релиза!')
```

Текущий этап

Практическое задание

Практическое задание



Для проекта «Мессенджер» реализовать логирование с использованием модуля **logging**:

1. В директории проекта создать каталог **log**, в котором для клиентской и серверной сторон в отдельных модулях формата **client_log_config.py** и **server_log_config.py** создать логгеры;
2. В каждом модуле выполнить настройку соответствующего логгера по алгоритму,подробно описанному в практическом задании в методичке к уроку 5.
3. Реализовать применение созданных логгеров для решения двух задач:
    a. Журналирование обработки исключений **try/except**. Вместо функции **print()** использовать журналирование и обеспечить вывод служебных сообщений в лог-файл;
    b. Журналирование функций, исполняемых на серверной и клиентской сторонах при работе мессенджера.

Дополнительные материалы

1. Logging Cookbook:

   https://docs.python.org/3/howto/logging-cookbook.html.

2. Логгирование в Python: https://python-scripts.com/logging-python.