Логирование. Реализация api первого источника валют

№ урока:	4	**Курс:**	Python Practice

Средства обучения:	Интерпретатор Python, virtualenv, текстовый редактор

Обзор, цель и назначение урока

Цель урока: ознакомить студентов необходимостью логирования в проектах, изучить с возможности модуля logging, его настройку. Добавить логирование в проект golden-eye. Познакомить студентов с библиотекой requests для взаимодействия с удаленными api, научить отправлять запрос и обрабатывать ответ, полученный по api.

Изучив материал данного занятия, учащийся сможет:

- добавлять логирование в проекты, настраивать модуль logging, анализировать информацию из файла логов
- реализовывать взаимодействия с удаленными api по протоколу HTTP с помощью библиотеки requests
- анализировать ответ api, проверять его на корректность
- парсить ответ для получения необходимой информации в нужном виде

Содержание урока

1. Что такое логирование и для чего оно нужно и даже необходимо?
2. Настройка логирования в файл с помощью встроенного модуля logging, конфигурация логирования
3. Вынесение данных о настройках - путь к файлу базы данных, путь и настройки файла логов - в отдельный модуль, config.py. Внесение изменений в затронутые модули, тестирование и анализ сформированных файлов логов.
4. Реализация модуля privat_api.py - отправка запроса, анализ ответа, извлечение информации о курсе, сохранение в базу обновленной информации.
5. Что такое парсинг - мы его уже осуществляем!
6. Написание и запуск теста для проверки работы нового модуля.
7. Добавление проверок ответа api
8. Полученная структура проекта, обзор

Резюме

- Логирование - это процесс фиксации действий, совершаемых в коде скрипта, приложения, проекта. Фиксация может осуществляться в консоль, файл, базу данных.
- Логирование преследует определенные цели:

CyberBionic systematics

CyberBionic Systematics ® 2018
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

t. +380 (44) 361-8937
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

    - отражение хода выполнения программы, независимо от того, идет все штатно или возникла ошибка
    - источник информации для дальнейшего анализа работы приложения - количество ошибок, время выполнения того или действия, общая нагрузка на приложение и так далее, анализировать можно в различных направлениях и плоскостях, админы будут благодарны за грамотные логи
    - сохранение максимальной информации для анализа причины ошибки

  Грамотно построенное логирование позволяет достигать все эти цели.
- Принципы грамотного логирования:
    - Отображение основных событий приложения - запуск-остановка, начало-конец обработки запросов, возникновение ошибок
    - Правильное использование различных уровней логирования, DEBUG, INFO, WARNING, ERROR, FATAL
    - Все логи должны соответствовать определенному формату
    - Должны быть легко анализируемыми, давать понимание об источнике записи лога
    - Одна строка — одно сообщение.
    - Логи должны быть лаконичными, стоит избегать избыточности, особенно для уровня логирование выше DEBUG.
    - Стоит избегать сложной логики, выполнения каких-либо действий при логировании - основная задача просто отобразить (значение переменной, ее тип, текст ответа и тп)
- Существует огромное количество вариантов организации логирования на данный момент - от библиотек конкретного языка программирования до программных комплексов, независимых от выбранного языка - ElasticSearch, Logstash и тп
- Для организации логирования с помощью модуля logging необходимо создать объект методом getLogger, указав имя логгера, которое будет отображаться в файле логов, и привязать к созданному объекту логгера обработчик логов - получатель, в данном случае FileHandler(так как мы хотим писать логи в файл). У объекта FileHandler указывается путь к файлу логов, который будет создан на диске, если еще не существует.
- Необходимо установить уровень логирования с помощью метода setLevel, а также указать, в каком формате нужно сохранять логи. Для управления форматом логов используется объект Formatter.
- Параметры для конфигурации логгера, а также файла базы данных необходимо выносить в отдельный модуль, для удобства управления ими. Обычно такой модуль с конфигурационной информацией называют config.py
- Существует множество библиотек для отправки запросов по протоколу HTTP, начиная от встроенной библиотеки urllib2 или httplib, но в настоящее время наиболее популярной и удобной библиотекой является requests.
- Для отправки GET запроса с помощью requests необходимо просто вызвать метод requests.get(), указав в качестве параметра - url удаленного api. Результатом выполнения будет объект Response, который содержит текст ответа, хедеры, HTTP код ответа и много другой полезной информации.
- Парсинг - это процесс разбора текстовой информации разнообразного формата (под каждый формат - свой парсер), получение необходимых данных и сохранение их в унифицированном формате.
- При работе с удаленными api необходимо всегда помнить одну истину - API могут отвечать неожидаемым форматом или вовсе быть недоступными. Поэтому необходимо всегда проверять полученный ответ во избежание необработанных ошибок.

Закрепление материала

- Что такое логирование? Зачем нужно добавлять логирование в проекты?

CyberBionic Systematics ® 2018
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

t. +380 (44) 361-8937
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- Какова последовательность действий для подключения логирования в файл с помощью библиотеки logging?
- Какие уровни логирования существуют? Для логирования какой информации подходит каждый из них?
- Зачем выносить некоторые константы в файл config.py? Какие параметры логично добавлять в это модуль?
- Какие библиотеки для отправки запросов по протоколу HTTP вы знаете?
- Какие основные атрибуты и методы объекта Response библиотеки requests?

Дополнительное задание

Задание
Заключить код отправки запроса по api в конструкцию try-except для того, чтоб в случае сетевой ошибки (таймаута, например) скрипт не «падал» с ошибкой, а отрабатывал без изменения данных в БД,

Самостоятельная деятельность учащегося

Задание 1
Добавить еще один получатель для логирования - логирование в консоль. Запустить тесты и проверить, что будет происходить

Задание 2
Расширить формат файла логов - добавить номер потока и его имя. Запустить тесты и проверить файл логов

Задание 3
Напишите python скрипт, который будет отправлять GET запрос на url http://google.com и выводить в консоль и файл логов текст полученного ответа

Задание 4
Дополните метод проверки ответа, полученного от api Приватбанка - добавить проверку на наличие ожидаемых параметоров в json ответе.

Рекомендуемые ресурсы

https://dou.ua/lenta/articles/logs-beat-debug/
https://gist.github.com/swvitaliy/8730998
https://docs.python.org/2/howto/logging-cookbook.html#logging-cookbook
https://docs.python.org/2/library/logging.html#logging-levels
http://requests.readthedocs.io/en/master/

CyberBionic systematics
CyberBionic Systematics ® 2018
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
t. +380 (44) 361-8937
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com