Работа с сетью в Python: Socket и HTTP

№ урока: 1 **Курс:** Python Advanced

Средства обучения: Python3.6, PyCharm

Обзор, цель и назначение урока

Научить студентов писать сетевые приложения на языке Python. Дать базовые знания сетевой модели OSI и её уровней, а также научить создавать собственные socket-сервера/клиенты.
Изучить протокол HTTP. Дать базовые знания данного протокола и его особенностей, а также научить обрабатывать HTTP сообщения с использованием языка Python.

Изучив материал данного занятия, учащийся сможет:

- Понимать основы сетевой модели OSI.
- Разрабатывать UDP/TCP socket-сервера.
- Разрабатывать UDP/TCP socket-клиенты, как для собственных, так и для сторонних socket-серверов.
- Понимать протокол HTTP.
- Понимать типы запросов, их особенности и ограничения.
- Формировать запросы к HTTP серверам и обрабатывать ответы от них.
- Использовать стандартную библиотеку urllib.
- Использовать стороннюю библиотеку requests.
- Создавать программы на языке Python, позволяющие автоматизировать обработку ответов.

Содержание урока

1. Основы сетевой модели OSI и её уровней.
2. Понятие адресации (IP и Port) и что такое socket.
3. Описание протокола UDP.
4. Создание UDP клиента/сервера.
5. Создание TCP клиента/сервера и сравнение с UDP.
6. Блокирующие и неблокирующие режимы работы socket.
7. ООП подход при создании socket-серверов используя библиотеку socketserver.
8. Создание простого socket-сервера средствами фреймворка Twisted.
9. Что такое протокол HTTP, как его использовать и основные типы запросов.
10. Понятие заголовков и статус кодов.
11. Типы HTTP запросов и их особенности.
12. Создание socket для демонстрации заголовков и ответов сервера.
13. Стандартная библиотека языка Python- urllib.
14. Библиотека requests.
15. Конфигурация библиотеки urllib: размер, pull-а соединений и режимы работы.

Резюме

- Модель OSI подробно описывает работу сети. Каждый уровень данной модели решает свою конкретную задачу, будь то логическая адресация, шифрование или передача данных по оптоволокну. Каждый уровень является частью единой модели.
- Полный адрес компонента сети состоит из IP и порта и записывается следующим образом 127.0.0.1:80.

CyberBionic systematics

CyberBionic Systematics ® 2019
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- UDP протокол является ненадёжным протоколом передачи данных. Пакеты, отправленные с использованием данного протокола, могут быть потеряны или же нарушен их порядок при получении. Нет никакой гарантии 100%-ной доставки пакетов.
- TCP протокол является надёжным протоколом передачи данных. Данный протокол осуществляет так называемое рукопожатие и перед передачей/получением данных должен установить соединение с конечным узлом сети.
- В стандартной библиотеке языка Python существует библиотека socket, которая позволяет создавать socket-сервера и socket-клиенты.
- Для создания socket-серверов наиболее удобным подходом является ООП. Это необязательное требование, однако ООП подход предоставляет более удобный интерфейс для обработки запросов и новых подключений. Мы можем использовать стандартную библиотеку socketserver для создания сокет-серверов в ООП стиле.
- Используйте готовые библиотеки, проверенные временем. Такие библиотеки позволят вам создавать стабильные и надежные сокет-сервера/клиенты, написав всего лишь несколько строк кода. В то же время данные библиотеки предоставляют возможность удобной конфигурации ваших сокетов: порты, блокировки, таймауты и т.п.
- HTTP протокол является текстовым протоколом с определенной структурой. Данный протокол- это надстройка над socket, ведь по своей сути мы открываем socket-соединение и обмениваемся данными, закрывая его по завершению.
- HTTP протокол реализует 9 различных методов: OPTIONS, GET, HEAD, POST, PUT, PATCH, DELETE, TRACE, CONNECT. Каждый из методов предназначен для конкретной задачи, например, GET- для получения данных, а PUT для обновления данных. Каждый метод имеет свои особенности и отличия.
- Существует важное отличие POST метода от GET метода- размер передаваемых данных GET методом ограничен и составляет 8 килобайт. Объем данных, передаваемый методом POST исчисляется мегабайтами и как плюс, всегда можно увеличить это значение.
- Данные, передаваемые методом GET, передаются в URL строке браузера. Данные передаваемые методом POST, передаются в теле запроса (стандартный потов ввода).
- Чтобы понять, успешно ли завершился наш запрос, мы можем использовать статус коды. 200-ые статусы означают успешность запросов. Ошибки сервера- 500-ые. Ошибки данных передаваемых клиентами- 400-ые.
- Язык Python содержит в своей стандартной библиотеке инструменты для работы с HTTP-библиотека urllib.

Закрепление материала

- Что такое сетевая модель OSI и из чего она состоит?
- Почему в ряде задач лучше использовать протокол TCP вместо UDP?
- В чем отличие протоколов TCP и UDP?
- Какой тип сокетов можно использовать, если вы хотите наладить взаимодействие между двумя программами в рамках одного сервера?
- За что отвечает константа socket.AF_INET?
- За что отвечает константа socket.AF_UNIX?
- Что такое HTTP протокол и из чего он состоит?
- Что такое заголовки?
- Какие типы запросов бывают?
- Какой метод необходимо использовать, например, для изменения данных о товаре?
- Какие существуют различия между методами GET и POST?
- Какой метод необходимо использовать для передачи логина и пароля при авторизации пользователя?
- За что отвечают статус коды диапазона 2xx?
- Если пользователь отправил некорректные данные на сервер, какой статус код ему необходимо вернуть в ответе?

CyberBionic systematics
CyberBionic Systematics ® 2019
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- Какой статус код необходимо вернуть, если пользователь не прошел авторизацию?
- Что такое pull соединений и как его можно настроить?
- Что такое блокирующий режим, pull-а соединений и в чем его особенность?

Дополнительное задание

Задание 1
Создать простой чат, на основе TCP протокола, который позволит подключаться нескольким клиентам и обмениваться сообщениями.
Задание 2
Создайте HTTP клиента, который будет принимать URL ресурса, тип метода и словарь в качестве передаваемых данных (опциональный). Выполнять запрос с полученным методом на полученный ресурс, передавая данные соответствующим методом, и печатать на консоль статус код, заголовки и тело ответа.

Самостоятельная деятельность учащегося

Задание 1
Изучите основные понятия, рассмотренные в уроке и работу с TCP и UDP протоколами в Python.

Задание 2
Создайте UDP сервер, который ожидает сообщения о новых устройствах в сети. Он принимает сообщения определенного формата, в котором будет идентификатор устройства и печатает новые подключения в консоль. Создайте UDP клиента, который будет отправлять уникальный идентификатор устройства на сервер, уведомляя о своем присутствии.

Задание 3
Создайте UNIX сокет, который принимает сообщение с двумя числами, разделенными запятой. Сервер должен конвертировать строковое сообщения в два числа и вычислять его сумму. После успешного вычисления возвращать ответ клиенту.

Задание 4
Изучите основные понятия, рассмотренные в уроке и работу с HTTP протоколам в Python, используя библиотеки urllib и requests.

Задание 5
Изучите более подробно и попробуйте возможности настройки, pull-а соединений и его режимов. Используя утилиту ab протестируйте ваши наработки (https://ru.wikipedia.org/wiki/ApacheBench).

Задание 6
Используя сервис https://jsonplaceholder.typicode.com/ попробуйте построить различные типы запросов и обработать ответы. Необходимо попрактиковаться с urllib и библиотекой requests. Рекомендуется сначала попробовать выполнить запросы, используя urllib, а затем попробовать реализовать то же самое используя requests

Рекомендуемые ресурсы

Официальный сайт Python (3.6)
https://docs.python.org/3.6/library/socket.html
Официальный сайт Python (3.6):
https://docs.python.org/3.6/library/urllib.html
Официальная документация библиотеки requests:
http://docs.python-requests.org/en/master/

CyberBionic systematics
CyberBionic Systematics ® 2019
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com