МИНОБРНАУКИ РФ

ФГБОУ ВПО Тверской государственный технический университет

Кафедра “Программное обеспечение”

Курсовая работа

дисциплина “Операционные системы и сети”

Тема: “ Протокол TCP. Сокеты. ”

Выполнил: студент группы

ПИН 17.06

Середавкин А.С

Проверил:

Прохныч А.Н

Тверь 2019

Оглавление

[Введение 3](#_Toc27591476)

[Аналитическая часть 3](#_Toc27591477)

[Теория 3](#_Toc27591478)

[Проектная часть 5](#_Toc27591479)

[Архитектура 5](#_Toc27591480)

[Пример использования 8](#_Toc27591481)

[Заключение 9](#_Toc27591482)

# Введение

Цель работы: ознакомится со стеком протоколов TCP/IP, в частности, TCP протоколом. Понять принцип работы сокетов.

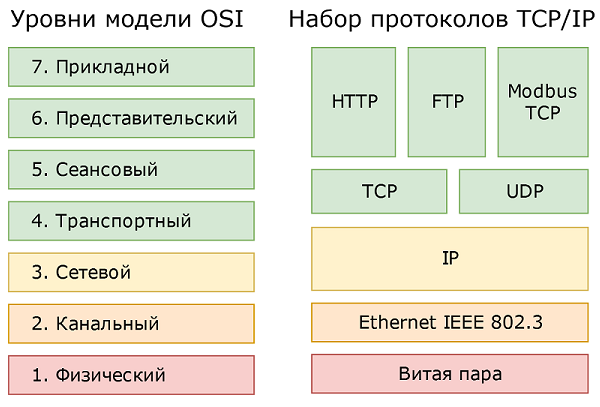
Задача: закрепить теоретические знания, разработав приложение, которое позволяет общаться нескольким пользователям. Использовать сокеты для связи между сервером и клиентами.

# Аналитическая часть

## Теория

**Transmission Control Protocol** (TCP, протокол управления передачей) — один из основных протоколов передачи данных интернета, предназначенный для управления передачей данных.

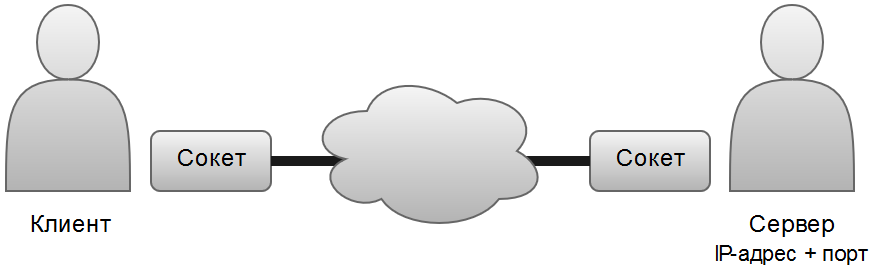
В стеке протоколов TCP/IP выполняет функции транспортного уровня модели OSI.



Механизм TCP предоставляет поток данных с предварительной установкой соединения, осуществляет повторный запрос данных в случае потери данных и устраняет дублирование при получении двух копий одного пакета, гарантируя целостность передаваемых данных и уведомление отправителя о результатах передачи.

Когда осуществляется передача от компьютера к компьютеру через Интернет, TCP работает на верхнем уровне между двумя конечными системами, например, браузером и веб-сервером. TCP осуществляет надёжную передачу потока байтов от одного процесса к другому. TCP реализует управление потоком, управление перегрузкой, рукопожатие, надёжную передачу.

Взаимодействие между устройствами в рамках стека TCP/IP осуществляется с помощью связки IP адреса и порта.  
  
Для задания IP адресов в настоящее время чаще всего используется протокол IPv4. Для него IP-адрес записывается в виде 32-битной формы, представляемой в символьной форме *mmm.nnn.ppp.qqq*: адрес, разбитый на четыре поля, разделённых точками, по одному байту в поле, например, 192.168.102.101. Номер порта задается в диапазоне от 0 до 65535.  
  
Пара адрес и порт образует ***сокет***(с английского socket – «гнездо»). Сокет является программным интерфейсом, который обеспечивает обмен данными между устройствами.



**Принципы сокетов**

Каждый процесс может создать слушающий сокет (серверный сокет) и привязать его к какому-нибудь порту операционной системы (в UNIX непривилегированные процессы не могут использовать порты меньше 1024). Слушающий процесс обычно находится в цикле ожидания, то есть просыпается при появлении нового соединения. При этом сохраняется возможность проверить наличие соединений на данный момент, установить тайм-аут для операции и т.д.

Обычно клиент явно подсоединяется к слушателю, после чего любое чтение или запись через его файловый дескриптор будут передавать данные между ним и сервером.

**Основные функции**

|  |  |
| --- | --- |
| **Общие** |  |
| Socket | Создать новый сокет и вернуть файловый дескриптор |
| Send | Отправить данные по сети |
| Receive | Получить данные из сети |
| Close | Закрыть соединение |
|  |  |
| **Серверные** |  |
| Bind | Связать сокет с IP-адресом и портом |
| Listen | Объявить о желании принимать соединения. Слушает порт и ждет когда будет установлено соединение |
| Accept | Принять запрос на установку соединения |
|  |  |
| **Клиентские** |  |
| Connect | Установить соединение |

# 

# Проектная часть

## Архитектура

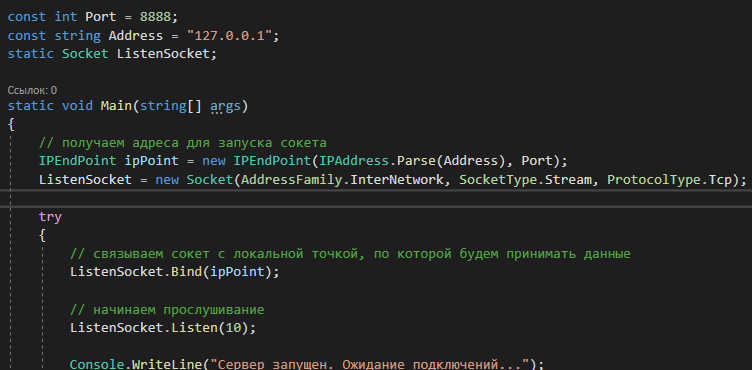
Приложение состоит из серверной и клиентской части.

В клиентской части классы **FirstClient** и **SecondClient**

* получают введенное пользователем имя
* устанавливают соединение с сервером, “регистрируя” это имя
* Отправляют сообщения от данного клиента на сервер и получают ответ.

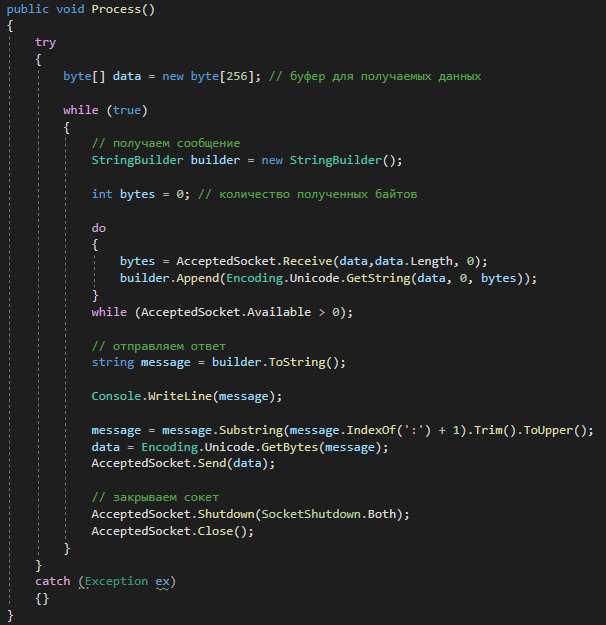


Класс **App** создает сокет, который слушает входящие подключения.



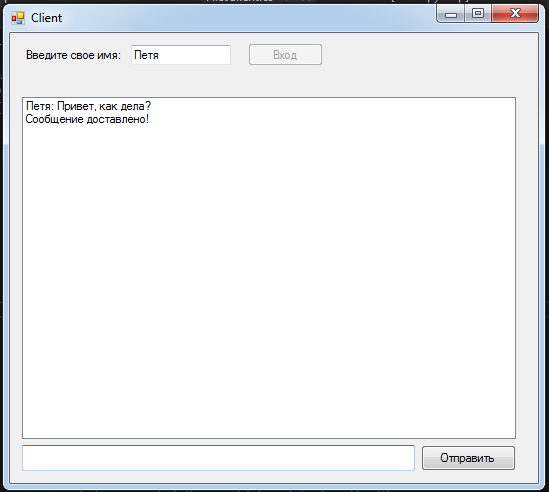
Когда клиент подключается, класс создает новый поток, который выполняет метод из класса **ClientHandler**.

Этот метод работает с данным клиентом: получает сообщение, выводя в консоль и возвращает ему ответ.

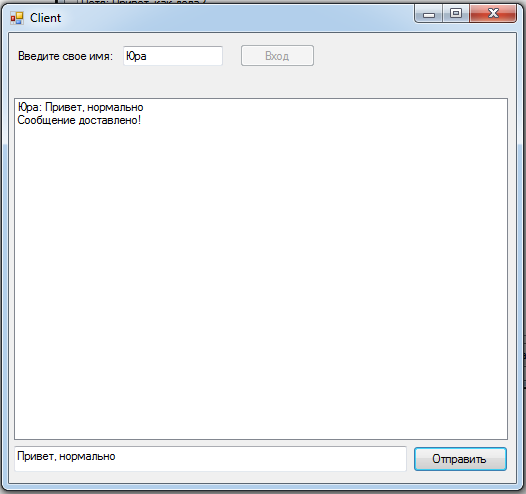


## Пример использования

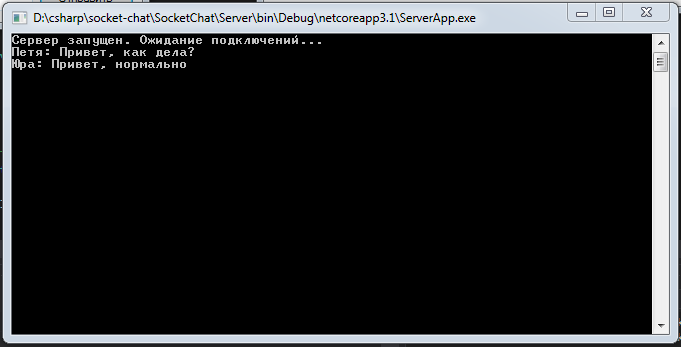
1. Пользователь вводит свое имя и сообщение



1. Второй пользователь делает то же самое



1. Их диалог отображается в консоли



# Заключение

В ходе работы над данной курсовой я понял принципы протокола TCP и реализовал чат между двумя пользователями, используя сокеты.