МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

## Факультет информационных технологий и робототехники

**Кафедра программного обеспечения информационных технологий и систем**

**Отчет по лабораторной работе № 5**

”Компьютерные системы и сети”

”ОСНОВЫ ТЕХНОЛОГИИ СОКЕТОВ”

|  |  |
| --- | --- |
| Выполнил: | Студент группы 10701117  Гутыра А. А. |
| Принял**:** | Белова С. В. |

Минск 2019

**Цели работы.**

Изучить:

* понятие сокета, типы сокетов TCP/IP;
* принципы реализации архитектуры клиент-сервер.
* сервер на основе интерфейса сокетов;
* основные классы и методы .NET Framework для разработки сетевых приложений с использованием Winsoc.;

**Постановка задачи.**

1. Изучить методические указания к лабораторной работе, материалы лекций и рекомендуемую литературу.
2. Разработать консольное клиент-серверное приложение, демонстрирующее взаимодействие на основе потоковых сокетов.
3. Разработать консольное клиент-серверное приложение, демонстрирующее взаимодействие на основе дейтаграммных сокетов.
4. Ответить на контрольные вопросы.

**Задание 1:** Используя класс Socket пространства имен System.Net.Sockets .Net Framework, разработать синхронное консольное клиент-серверное приложение. Клиент и сервер должны осуществлять взаимодействие по протоколу TCP.

**Задание 2:** Разработать клиент-серверное приложение, аналогичное предыдущему заданию, но для взаимодействия клиента и севера использовать протокол UDP и дейтаграммные сокеты.

**Задание 3:** Разработать клиент-серверное приложение для получения списка открытых TCP портов удаленного хоста. Взаимодействие клиента и сервера осуществлять на основе  
дейтаграммных сокетов. Для получения портов использовать класс SocketException. Сканирование портов выполнять в цикле по номерам портов.

**Результаты выполнения заданий.**

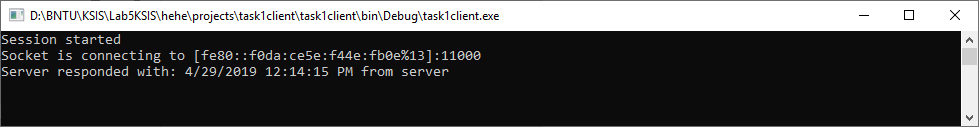


Рисунок 1 – клиент (задание 1).

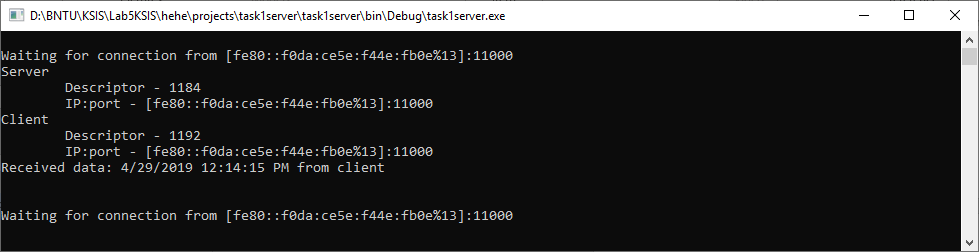


Рисунок 2 – сервер (задание 1).

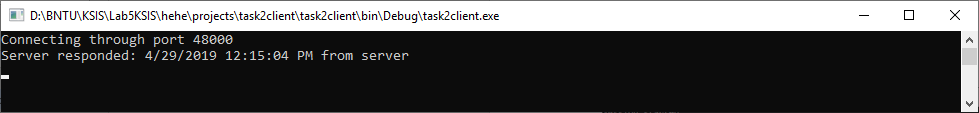


Рисунок 3 – клиент (задание 2).

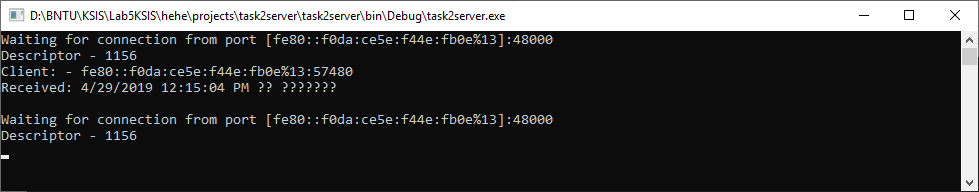


Рисунок 4 – сервер (задание 2).

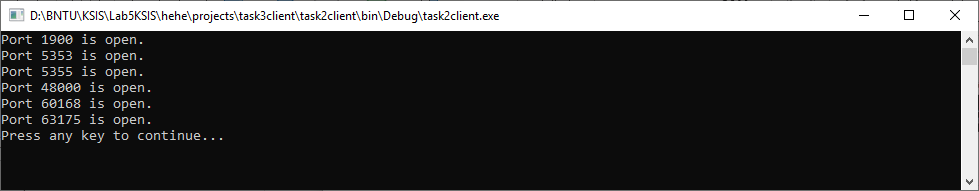


Рисунок 5 – клиент (задание 3).

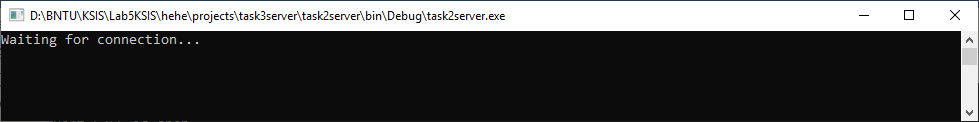


Рисунок 6 – сервер (задание 3).

**Контрольные вопросы.**

1. **Что такое сокет? Как по отношению к уровням стека TCP/IP расположен интерфейс сокетов?**

Сокет (Socket - гнездо, разъем) - абстрактное программное понятие, используемое для обозначения в прикладной программе конечной точки сетевого соединения.

Интерфейс сокетов расположен над транспортным уровнем стека TCP/IP, но в некоторых случаях может взаимодействовать напрямую с сетевым уровнем в обход транспортного.

1. **Сколько сокетов необходимо для взаимодействия клиента и сервера? Что представляет собой адрес сокета?**

Два сокета (один на клиент и один на сервер).

Адрес сокета при использовании протоколов TCP/IP – это IP-адрес и номер порта прикладной службы.

1. **Назовите типы сокетов. В каком случае предпочтительнее использовать тот или иной тип сокетов? Какой тип сокетов обеспечивает надежность и порядок доставки?**

Существуют три основных типа сокетов: потоковые, дейтаграммые и сырые.

*Потоковые сокеты* – это сокеты с установлением соединения, состоящие из потока байтов, который может быть двунаправленным. Т.е. через такую конечную точку приложение может и передавать, и получать данные. Потоковый сокет гарантирует обнаружение и исправление ошибок, обрабатывает доставку и сохраняет последовательность данных. Он подходит для передачи больших объемов данных, поскольку в этом случае накладные расходы, связанные с установлением соединения, незначительны по сравнению со временем передачи самого сообщения. Качество передачи достигается за счет использования протокола TCP.

*Дейтаграммные сокеты* – это сокеты без установления соединения, не обеспечивающие надежность при передаче. Применяются для приложений, когда неприемлемы затраты времени, связанные с установлением явного соединения. Для передачи данных используется протокол UDP.

*Сырые сокеты* (raw sockets - необрабатываемые, простые) – это сокеты, которые взаимодействуют с протоколами сетевого уровня в обход протоколов транспортного уровня. Используются для непосредственного доступа приложения к IP-пакетам сетевого уровня. Использование сырых сокетов возможно при разработке низкоуровневого системного ПО. Например, сырые сокеты используют различные программы-анализаторы пакетов, сниферы, утилиты TCP/IP ping, tracert и т д.

Потоковые сокеты.

1. **Чем различаются потоковые протоколы и протоколы, ориентированные на передачу сообщений? Как это отражается на коде при написании программ?**

Потоковые протоколы ориентированы на установление логической связи и надёжность передачи (контроль ошибок), а протоколы, ориентированные на передачу сообщений, не контролируют установление соединения и правильность передачи сообщений.

При использовании дейтаграмных протоколов необходимо программно контролировать установление соединения и ошибки при передаче.

1. **Чем отличается процесс получения и отправки данных на сокете, не требующем соединения?**

При использовании дейтаграммных сокетов сначала создают сокет. Затем выполняют привязку сокета к интерфейсу, на котором будут принимать данные, методом Bind (как и в случае протоколов, ориентированных на соединения). Разница в том, что вместо использования методов Listen или Accept нужно просто ожидать приема входящих данных. Поскольку в этом случае соединения нет, принимающий сокет может получать дейтаграммы от любой машины в сети.

Простейший метод приема — ReceiveFrom:

Другой способ приема (отправки) данных на сокетах, не требующих соединения, — установление соединения (хоть это и звучит странно).

После создания сокета можно вызвать метод Connect. Фактически никакого соединения не происходит. Адрес сокета, переданный в функцию соединения, ассоциируется с сокетом, чтобы было можно использовать метод Receive вместо ReceiveFrom (поскольку источник данных известен).

Есть два способа отправки данных через сокет, не требующий соединения. Первый и самый простой — создать сокет и вызвать метод SendTo.

Как и при получении данных, сокет, не требующий соединения, можно подключать к адресу конечной точки и отправлять данные методом Send. После создания этой привязки использовать для обмена данными метод SendTo с другим адресом нельзя — будет выдана ошибка. Отменить привязку сокета можно, лишь вызвав метод Close с описателем этого сокета, после чего следует создать новый сокет. Поскольку соединение не устанавливается, его формального разрыва или корректного закрытия не требуется. После прекращения отправки или получения данных отправителем или получателем просто вызывается метод Close с описателем требуемого сокета, в результате чего освобождаются все выделенные ему ресурсы.

1. **Какой адрес можно использовать, чтобы прослушивать все сетевые интерфейсы локального узла?**

0.0.0.0

1. **Какой порядок байтов используется в Intel-совместимых процессорах? Какой порядок байтов применяется для работы с сокетами?**

Для Intel-совместимых процессоров это порядок от менее значимого к более значимому байту. Его называют обратный или little endian (младший байт числа расположен в младшем адресе памяти).

Сетевые стандарты требуют, чтобы многобайтные значения передавались в сетевом порядке (network-byte order).

1. **Какие номера портов могут задействовать прикладные программы при работе с сокетами?**

Номера портов разделяются на 3 категории (стандартные, зарегистрированные и динамические и/или частные):

- Номера от 0 до 1023 зарезервированы для стандартных служб.

- Порты с номерами от 1024 до 49151 являются регистрируемыми. - Порты с номерами от 49152 до 65535 - динамические и частные порты.

Во избежание накладок с портами, уже занятыми системой или другим приложением, ваша программа должна выбирать порты, начиная с 1024. Можно вместо конкретного номера порта задать 0, тогда система сама выберет произвольный неиспользуемый в данный момент номер.

1. **Как можно узнать номера портов, занятых стандартными службами? В каком файле хранится эта информация?**

Запустив утилиту netstat –a, можно увидеть перечень всех используемых в данный момент на компьютере номеров портов.

В файле services из каталога <windir>\system32\drivers\ets перечислены предопределенные пользовательские и системные номера портов, используемых стандартными службами.

1. **Как осуществляется корректное завершение сеанса работы с потоковым сокетом, с дейтаграммным сокетом?**

Чтобы корректно инициировать закрытие потокового сокета, вызывается метод Shutdown.

Поскольку при работе с дейтаграммным соединение не устанавливается, его формального разрыва или корректного закрытия не требуется. После прекращения отправки или получения данных отправителем или получателем просто вызывается метод Close с описателем требуемого сокета, в результате чего освобождаются все выделенные ему ресурсы.

1. **Какие пространства имен и классы в .NET Framework обеспечивают поддержку сокетов?**

Поддержку сокетов в .NET обеспечивают классы в пространстве имен System.Net.Sockets.

NetworkStream реализует базовый класс потока, из которого данные отправляются и в котором они получаются. Это абстракция высокого уровня, представляющая соединение с каналом связи TCP/IP.

Socket – класс, обеспечивающий базовую функциональность приложения на основе сокетов.

TcpClient строится на классе Socket, чтобы обеспечить TCP-обслуживание на более высоком уровне. TcpClient предоставляет несколько методов для отправки и получения данных через сеть. TcpListener построен на низкоуровневом классе Socket. Предназначен для создания серверных приложений. Он ожидает входящие запросы на соединения от клиентов и уведомляет приложение о любых соединениях.

UdpClient предназначен для реализации обслуживания по протоколу UDP.

SocketException - это исключение порождается, когда в сокете возникает ошибка.

**Выводы:** в результате лаборатороной работы были изучены принципы реализации архитектуры клиент-сервер, понятие сокета, типы сокетов TCP/IP, основные классы и методы .NET Framework для разработки сетевых приложений.