Полное руководство по сетевому программированию для разработчиков игр. Часть 2.

Автор: **x84**

Гы... я просто поражаюсь твоему упорству, дружище.
Как можно читать этот бред? :) (c) x84

Так-с... Мы успешно преодолели первую часть пути. Осталось еще 194 части. Шутка. Мы познакомились с сокетами, теперь нам надо знать, почему два сокета, открытых на двух разных машинах, находящихся в разных частях света, позволяют нам передавать через них данные. В этой части мы построим свою (пока еще примитивную) иерархию классов для работы с сокетами и обмена данными. Это будет наш каркас - полигон для проведения дальнейших испытаний.

Американские вояки

Пожалуй, настал момент, когда следует немного отвлечься от программирования и совершить небольшой экскурс в историю...

Все, связаное с Интернет, начиналось с проекта ARPA, финансируемого министерством обороны США. Им надо было обеспечить себя быстрой связью, однако они хотели сделать связь не только быстрой, но и надежной. Посему, они сказали своим ученым дядькам: "Ми хотеть, чтобы сеть работать, даже если ее большой часть уничтожайть Советский Союз!". То есть они хотели, чтобы если какие-то узлы сети перестанут функционировать ("Русский нападать на нас! Тревога!"), это не сказалось на возможности обмена данными между другими узлами. Американские ученые почесали свои американские затылки и нашли-таки выход из ситуации. Они просто решили использовать не потоковую, а пакетную передачу данных. Это позволило сети адекватно реагировать на такие внештатные ситуации, как, например, выпадение целых сегментов сети, и позволило ввести механизмы для поиска альтернативных маршрутов движения данных. К чему я все это написал? Да просто такая динамическая структура сети накладывает на наше приложение определенные ограничения, а также создает немало проблем. Например, мы не можем полагаться на стабильную работу компьютера по ту стороны соединения и промежуточных компьютеров,

Стек TCP/IP
Адреса и порты

Стек TCP/IP

Мы будем рассматривать только стек протоколов, под названием TCP/IP. Стек протоколов - это набор, содержащий в себе разные протоколы, каждый из которых предназначен для решения отдельных типов задач. Однако стек - это не просто набор, стек - это некоторая иерархическая структура.

IP — Internet Protocol (Протокол Интернет)

Этот протокол находится на самом нижнем уровне стека. При помощи него мы можем передавать пакеты от отправителя адресату. IP пакет логически похож на письмо. Каждый пакет содержит в себе заголовок со служебной информацией (конверт) и сами данные (письмо), предназначенные адресату. Что такое заголовок пакета? Это некоторая служебная информация, находящаяся в начале пакета (на лицевой стороне конверта), которая определяет как этот пакет будет обрабатываться на выходе от отправителя, на промежуточных узлах сети, и на входе у получателя. Заголовок IP пакета содержит в себе адреса отправителя и получателя. Протокол IP не дает гарантии, что пакет достигнет адресата или что данные, содержащиеся в нем, не будут искажены (лично я сталкивался с потерей пакетов, но никогда не сталкивался с искажением данных на пути следования пакета (если только они не были искажены кем-то намерено)).

Сам по себе протокол IP не очень-то функционален. Поэтому были изобретены протоколы TCP и UDP, которые добавляют к IP новые возможности и являются более удобным для программиста средством передачи данных, нежели IP. Однако это не отдельные протоколы. Они основываются на IP. То есть в стеке TCP/IP они находятся уровнем выше.

UDP — User Datagram Protocol (Протокол Пользовательских Дэйтаграмм) - SOCK_DGRAM

Этот протокол позволяет передавать данные без установления соединения, то есть, чтобы послать кому-то данные, его не надо предварительно оповещать об этом. Просто посылаем дэйтаграмму и забываем о ней. Дэйтаграмма представляет из себя связку "заголовок + пользовательские данные". Заголовок дэйтаграммы включает в себя средства контроля за целостностью данных, то есть этот протокол не гарантирует доставку данных адресату, но если уж они доставлены, то доставлены полностью и не были искажены во время путешествия по сети.

TCP — Transmission Control Protocol (Протокол Контроля за Передачей Данных) - SOCK_STREAM

соединение, и уже только после этого они смогут обмениваться данными. TCP также включает в себя заголовок и сами данные. Его недостаток по сравнению с UDP - это более медленный обмен данными и повышенное количество байт служебных данных.

ICMP — Internet Control Message Protocol (Протокол Служебных Сообщений Интернет)

Этот протокол используется для обмена служебными сообщениями. Он используется в основном для диагностики работы сети и сетевых устройств и редко применяется для обмена пользовательскими данными, поэтому сейчас мы рассматривать его не будем, а сделаем это позже.

Как же все это работает? Очень просто. IP состоит из заголовка и данных. А что, если нам нужна дополнительная функциональность и мы хотим использовать TCP? Мы просто берем пакет TCP и вкладываем его в пакет IP. То есть получается, что IP пакет будет состоять из IP-заголовка и поля данных, которое в свою очередь будет содержать TCP-заголовок и данные пользователя. То есть физически IP-пакет содержит в себе TCP-пакет, но логически протокол TCP находится уровнем выше IP. С UDP ситуация абсолютно идентичная. Получается "два в одном". Причем, что самое приятное, упаковкой TCP или UDP в пакет IP занимается сетевая подсистема ОС, то есть она делает эту процедуру абсолютно прозрачной для нас и мы не заботимся об этом. Мы просто говорим: "Послать данные нафиг!" - все остальное сделает ОС. Согласно OSI мы работаем на 4 уровне, который скрывает реализацию процесса отправки и приема пакетов. В этом и заключается вся прелесть этой модели. Нам важен результат и операционная система нам гарантирует, что результат будет получен оптимальным образом.

Страницы: **1** 2 Следующая »

#TCP, #TCP/IP, #UDP

9 июня 2003 (Обновление: 30 сен 2009)

Комментарии [1]

Контакт
Сообщества
Участники
Каталог сайтов
Категории
Архив новостей