Communication Over the Internet

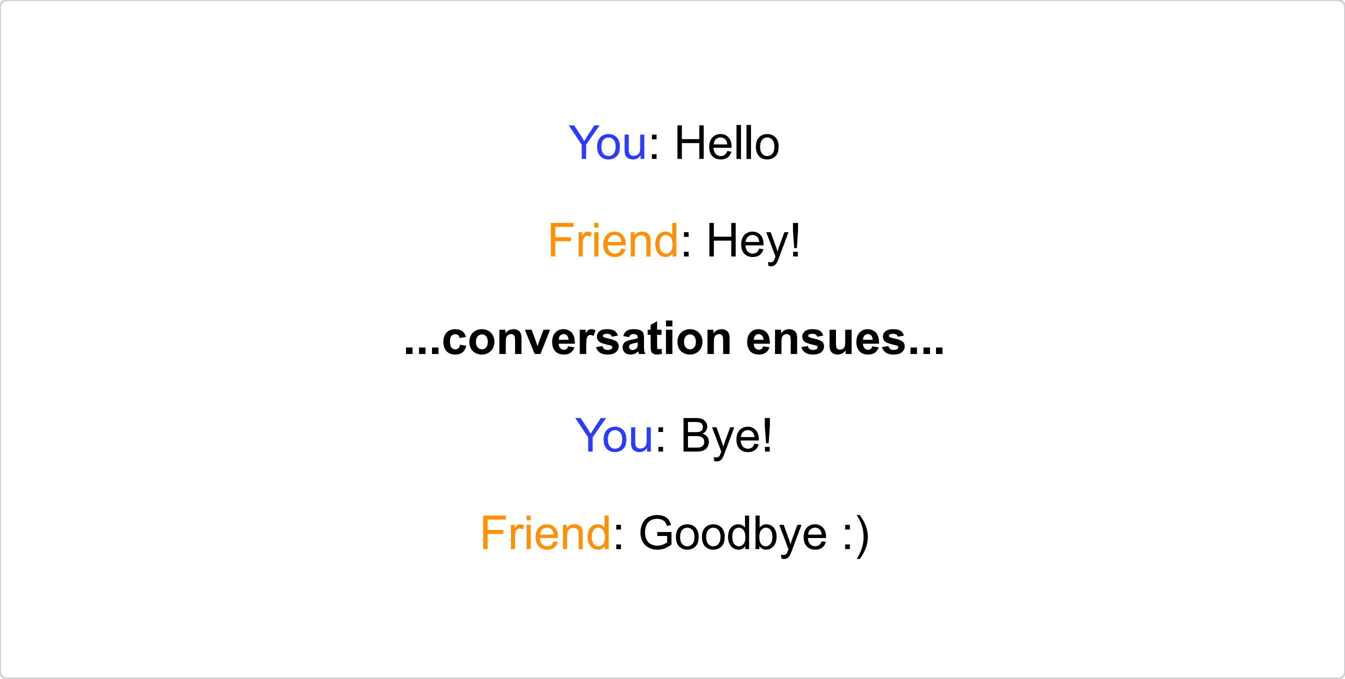
Before we dive deeper into the course, let's study some key concepts to understand how communication over the Internet works

We'll cover the following

What Is a Protocol?

An Analogy

Let’s start with an analogy. Think of your routine conversations. They usually **follow a general pattern** dictated by predefined rules. For example, most conversations start with greetings and end with goodbyes. They probably go something like this:



You: Hello

Friend: Hey!

...conversation ensues...

You: Bye!

Friend: Goodbye :)

Turns out that end systems also follow such **protocols to communicate with each other effectively** on the network.

Formally, according to the Oxford Dictionary, a protocol is “a set of rules governing the exchange or transmission of data between devices.” In the next few chapters, we’ll study several network protocols in detail.

TCP

The **Transmission Control Protocol (TCP)** is one such protocol. It was created to allow end systems to communicate effectively. The distinguishing feature of TCP is that it ensures that data reaches the intended destination and is not corrupted along the way.

UDP

The **User Datagram Protocol (UDP)** is also one such key protocol. However, it **does not ensure** that data reaches the destination and that it remains incorrupt.

HTTP

HyperText Transfer Protocol (HTTP) is a web protocol that defines the format of messages to be exchanged between web clients, e.g., web browsers and web servers and what action is to be taken in response to the message. The World Wide Web uses this as its underlying protocol.

Packets

Now that we've established that end systems communicate with each other based on set protocols, let's discuss *how* they actually communicate. Computers send messages to each other that are made up of ones and zeros (bits).

However, instead of sending messages of possibly *trillions* of bits all in one go, they're broken down into smaller units called **packets** to make transmission more manageable. These smaller sizes make transmission more manageable because most links are shared by a few end-systems. Sending smaller units in succession instead of one big file all in one go makes usage of the network fairer amongst end-systems.

We'll talk about the exact technical definition of a packet in a future chapter.

Addressing

So, applications communicate with each other by sending messages based on protocols. However, packets have to be addressed to a certain application on a certain end system. How do you do that out of potentially millions of end systems and hundreds of applications on each of them? The answer lies in addressing.

An address identifies a sending entity and a receiving entity.

IP Addresses

Every device that is connected to the Internet has an address called an 'IP Address' which is much like a mailing address.

- IP addresses are 32 bit numbers (in IP version 4).
- The human readable way for looking at these numbers is the **dotted decimal notation**, whereby the number is considered one octet of bits (8 bits) at a time. Those octets are read out in decimals, then separated by dots.
    - Hence, each number can be from 0 to 255. For example, `1.2.3.4`.

- Some IP addresses are reserved for specific functions. We'll discuss them in more depth in later lessons.

Check yours by running the following command on a shell on your local setup.

```
curl ifconfig.me -s
```

> All of the code on our platform is run on one of our servers, and the output is returned and printed on your screen. Hence, **the IP address here belongs to an Educative server**!

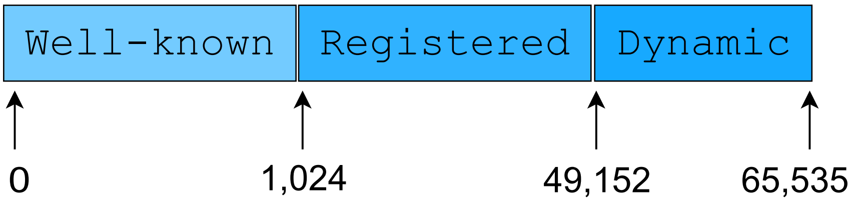
Ports

Any host connected to the Internet could be running many network applications. In order to distinguish these applications, all bound to the same IP address, from one another, another form of addressing, known as **port numbers**, is used. Each endpoint in a communication session is identified with a unique IP address and port combination. This combination is also known as a **socket**. So in essence, ports help to address the packet to **specific applications** on hosts.

- IP addresses identify end systems but ports identify an application on the end system.
- Every application has a 16-bit port number. So the port number could range from 0 to $2^{16} = 65535$.
- The ports $0 - 1023$ are reserved for specific applications and are called **well-known ports**.
    - For instance, port 80 is reserved for HTTP traffic.
- The ports $1024 - 49152$ are known as **registered** ports and they are used by specific, potentially proprietary, applications that are known but not system defined.
    - SQL server for example, uses port 1433

  - It is generally considered best practice not to use these ports for any user defined applications although there is no technical restriction on using them.
- The ports 49152–65535 can be used by user applications or for other purposes (dynamic port allocation for instance, but more on that later).

Here is a visual of these ports.



Quick Quiz!

1

Which of the following is a valid IP version 4 address?

○ A) 3.-1.33.67

○ B) 3456.3.17.8

○ C) a.e0.55.2

○ D) 203.128.22.10

COMPLETED 0%

1 of 4 < >

In the next lesson, we’ll study some physical and hardware aspects of computer networks.