Week 10 - Monday

COMP 2000

Last time

- What did we talk about last time?
- Reading and writing binary files
- Serialization

Questions?

Project 3

Reading and Writing Whole Objects

What if I wanted to read or write a whole object?

- An object has data inside of it
- Each piece of data is either a reference to an object or is primitive data
- When reading or writing whole objects, we could read or write each piece of data separately
- But doing so is challenging because we could forget some data
- And because there could be circular references:
  - Object A might have a reference to object B which might have a reference to object A again…

Serialization

- Serialization takes a reference to an object and dumps it into a file
- It writes representations to primitive types pretty much the same way that a `DataOutputStream` does
- And if there're objects inside of the object you're serializing, it serializes them too
- **And!** Serialization makes a note of all the objects that are getting serialized, so if it sees an object a second time, it just writes down a serial number for it instead of the whole thing

Example Serializable class

- Here's a class we might want to be able to dump into a file

```
public class Troll implements Serializable {
  private String name;
  private int age;
  private Object hatedThing; // All trolls hate something
  public Troll(String name, int age, Object hatedThing) {
     this.name = name;
     this.age = age;
     this.hatedThing = hatedThing;
  }
  public Object getHatedThing() {
     return hatedThing;
  }
}
```

Example of writing

- Here's some code that creates a couple of **`Troll`** objects and then writes them to a file called **`trolls.dat`**

```
Troll tom = new Troll("Tom", 351, "Bilbo Baggins");
Troll bert = new Troll("Bert", 417, tom);
ObjectOutputStream out = null;
try {
    out = new ObjectOutputStream(new FileOutputStream("trolls.dat"));
    out.writeObject(tom);
    out.writeObject(bert);
}
catch(IOException e) {
    System.out.println("Serialization failed.");
}
finally { try{ out.close(); } catch(Exception e){} }
```

Example of reading

- Here's some code that reads in the **`Troll`** objects we serialized in the previous example

```
Troll tom = null;
Troll bert = null;
ObjectInputStream in = null;
try {
    in = new ObjectInputStream(new FileInputStream("trolls.dat"));
    tom = (Troll)in.readObject();
    bert = (Troll)in.readObject();
}
catch(IOException e) {
    System.out.println("Deserialization failed.");
}
finally { try{ in.close(); } catch(Exception e){} }
```

The good

- Serialization allows you to read or write objects (even complex objects) or arrays of objects in a single line of code
- It's an impressive achievement of Java
- To make your own classes serializable, all you have to do is mark them with the **`Serializable`** interface
  - An interface with no methods!
- It more or less works like magic!

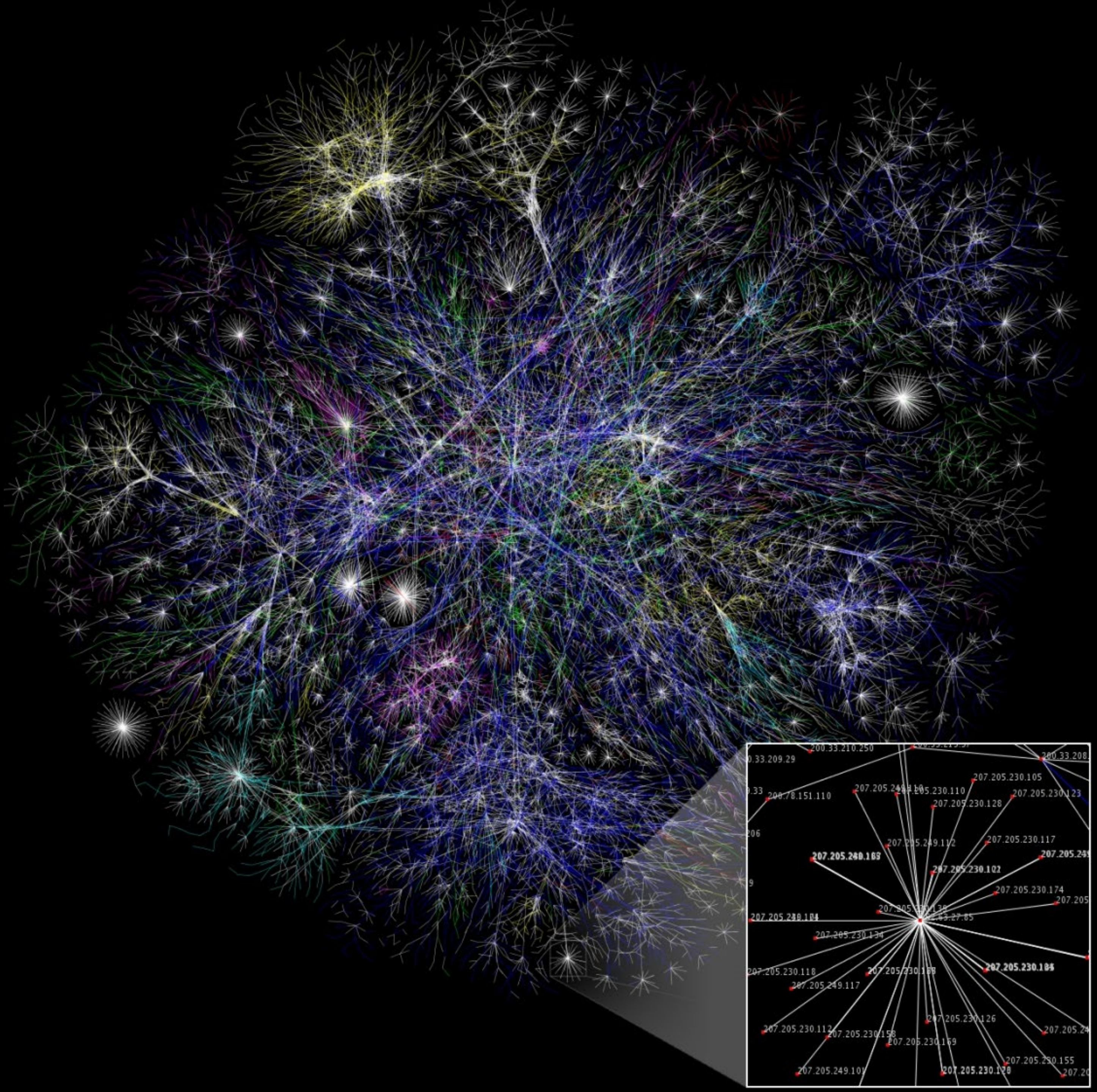
The bad

- Some objects are not serializable, but they are comparatively rare
- An example is the `Thread` class, which encapsulates the state of a currently running thread…so how could you store it on disk?
- Serialization does have storage overhead needed to keep track of the size of arrays and type information about classes
  - You might be able to use less space if you stored the data directly

The ugly

- If you forget to mark one of your classes **`Serializable`**, it will crash your code when you try to write it out, even indirectly
- If you serialize objects to a file but later change the class, adding or removing members or methods, you will no longer be able to read those objects back from the file
- Their data in the file will no longer match what the class is supposed to look like
- This problem can happen with different versions of the same program

Internet



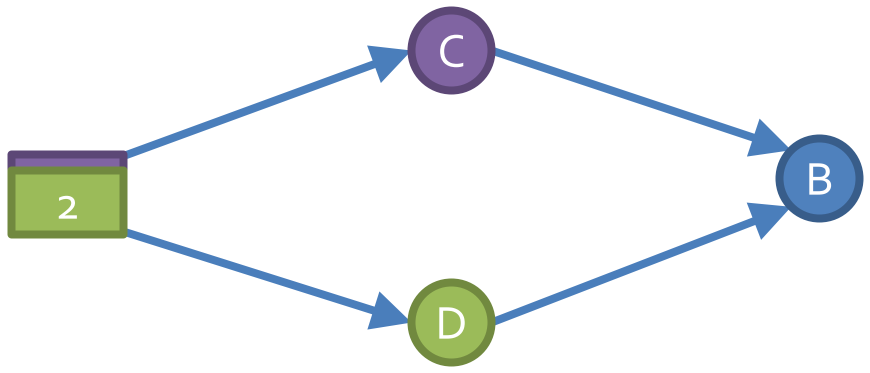
200.33.210.250
207.205.230.105
207.205.230.123
207.205.230.128
207.205.249.112
207.205.230.117
207.205.230.174
207.205.230.134
207.205.230.118
207.205.249.117
207.205.230.126
207.205.230.112
207.205.230.155

What is the Internet?

- The network of hardware and software systems that connects many of the world's computers
- Typically, people say the Internet and capitalize the "I" because there is only one
  - Until we meet aliens
  - Or decide to break off from the rest of the world
- The World Wide Web is the part of the Internet that is concerned with webpages
- The Internet also includes:
  - FTP
  - VOIP
  - Bittorrent
  - Multiplayer video games
  - Much, much more…

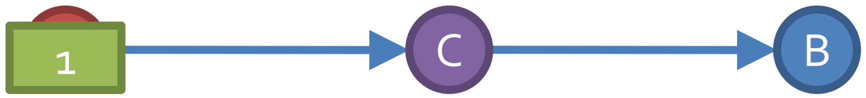
Packet switched

- The Internet is a packet switched system
- Individual pieces of data (called packets) are sent on the network
  - Each packet knows where it is going
  - A collection of packets going from point A to point B might not all travel the same route



Circuit switched

- Traditionally, phone lines have been circuit switched
- A specific circuit is set up for a specific communication
- Operators used to do this by hand
- Now it is done automatically
- Only one path for data



Circuit vs. packet switching

- Which one is faster?
  - Circuit switching
- Which one is more predictable?
  - Circuit switching
- So, why is the Internet packet switched?
  - More adaptable

Birth of the Internet

- The Advanced Research Projects Agency was created in 1958 to respond to the Russians launching Sputnik
- The ARPANET connected its first two major nodes over 10 years later
- Packet switching was used so that the network could still communicate after a nuclear strike

IP addresses

- Computers on the Internet have addresses, not names
- **Google.com** is actually [`74.125.67.100`]
- **Google.com** is called a **domain**
- The Domain Name System or DNS turns the name into an address

IPv4

- Old-style IP addresses are in this form:
  - **`74.125.67.100`**
- 4 numbers between 0 and 255, separated by dots
- That's a total of 256^4 = 4,294,967,296 addresses
- But there are 7 billion people on earth…

IPv6

- IPv6 are the new IP addresses that are beginning to be used by modern hardware
  - 8 groups of 4 hexadecimal digits each
  - **`2001:0db8:85a3:0000:0000:8a2e:0370:7334`**
  - 1 hexadecimal digit has 16 possibilities
  - How many different addresses is this?
  - $16^{32} = 2^{128} \approx 3.4 \times 10^{38}$ is enough to have 500 trillion addresses for every cell of every person's body on Earth
  - Will it be enough?!

Other failures in design

- Y2K bug
  - 2 bytes for the date is not enough
  - It's all just going to get messed up in Y10K
- Y2038 bug
  - Unix and Linux machines often use a signed 32-bit integer to represent seconds since January 1, 1970
- Zip codes
- Vehicle identification numbers

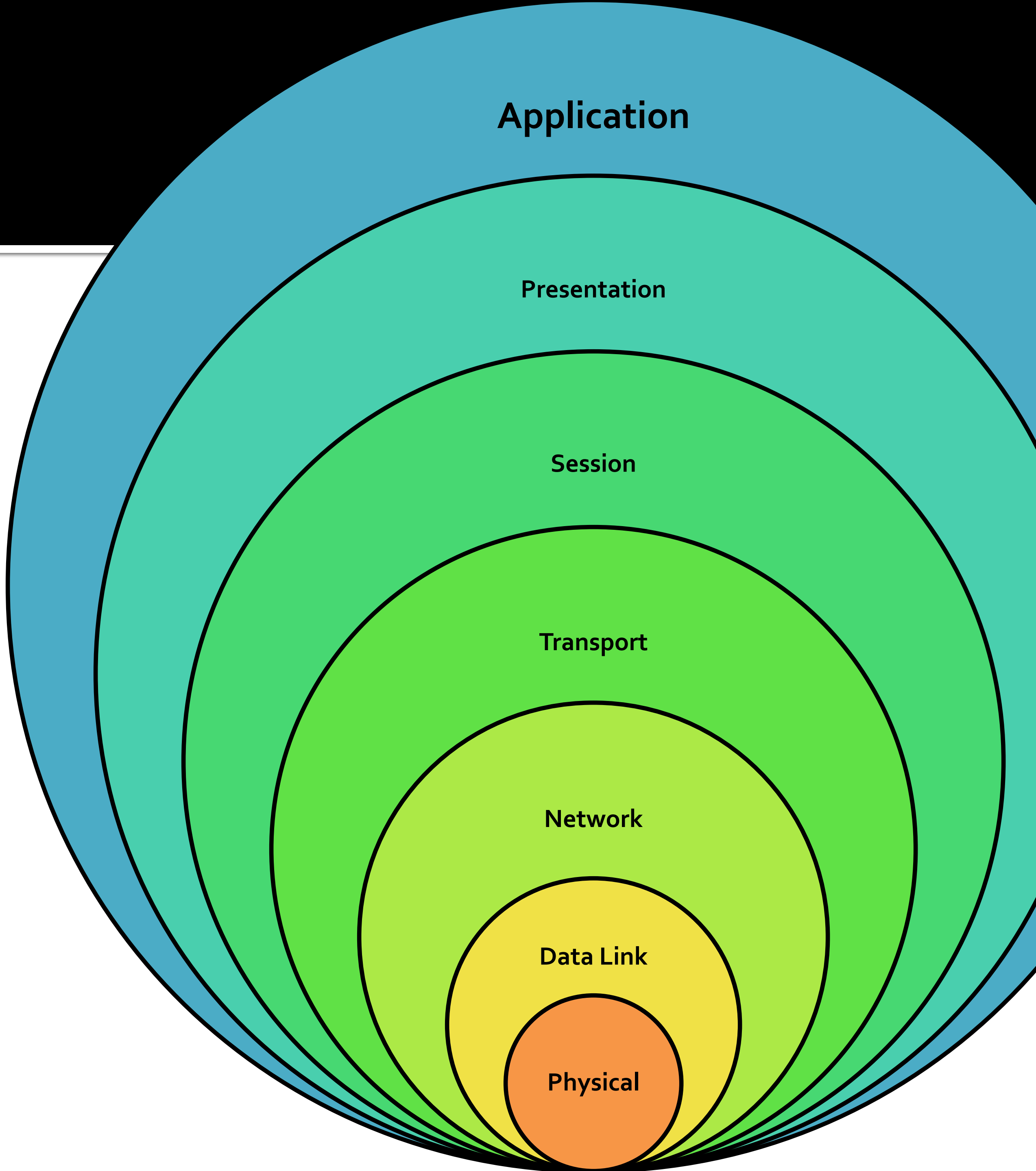
Networking

OSI seven layer model

- You can build layers of I/O on top of other layers
  - **`System.out.println()`** is built on top of low-level calls, eventually some C system call to the OS
- One standard networking model is called the Open Systems Interconnection Reference Model
  - Also called the OSI model
  - Or the 7 layer model

Protocols

- There are many different communication protocols
- The OSI reference model is an idealized model of how different parts of communication can be abstracted into 7 layers
- Imagine that each layer is talking to another parallel layer called a **peer** on another computer
- Only the physical layer is a real connection between the two



Layers

- Not every layer is always used
- Sometimes user errors are referred to as Layer 8 problems

Layer	Name	Mnemonic	Activity	Example
7	**Application**	Away	User-level data	HTTP
6	**Presentation**	Pretzels	Data appearance, some encryption	SSL
5	**Session**	Salty	Sessions, sequencing, recovery	IPC and part of TCP
4	**Transport**	Throw	Flow control, end-to-end error detection	TCP
3	**Network**	Not	Routing, blocking into packets	IP
2	**Data Link**	Dare	Data delivery, packets into frames, transmission error recovery	Ethernet
1	**Physical**	Programmers	Physical communication, bit transmission	Electrons in copper

Physical layer

- There is where the rubber meets the road
- The actual protocols for exchanging bits as electronic signals happen at the physical layer
- At this level are things like RJ45 jacks and rules for interpreting voltages sent over copper
  - Or light pulses over fiber

Data link layer

- Ethernet is the most widely used example of the data layer
- Machines at this layer are identified by a 48-bit Media Access Control (MAC) address
- The Address Resolution Protocol (ARP) can be used for one machine to ask another for its MAC address
- Some routers allow a MAC address to be spoofed, but MAC addresses are intended to be unique and unchanging for a particular piece of hardware

Network layer

- The most common network layer protocol is Internet Protocol (IP)
- Each computer connected to the Internet should have a unique IP address
  - IPv4 is 32 bits written as four numbers from 0 – 255, separated by dots
  - IPv6 is 128 bits written as 8 groups of 4 hexadecimal digits
- We can use **`tracert`** on Windows to see the path of hosts leading to some IP address

Transport layer

- There are two popular possibilities for the transport layer
- Transmission Control Protocol (TCP) provides reliability
  - Sequence numbers for out of order packets
  - Retransmission for packets that never arrive
- User Datagram Protocol (UDP) is simpler
  - Packets can arrive out of order or never show up
  - Many online games use UDP because speed is more important

Session layer

- This layer doesn't necessarily exist in the TCP/IP model
- Transport Layer Security (TLS) uses the session layer
- TLS is the end-to-end encryption that HTTPS uses
- You know you're using TLS if there's a little lock showing on your browser
- Google is pushing for all websites to be HTTPS
- HTTPS is safer, but there's some overhead for the encryption, and websites have to have certificates for their public keys

Presentation layer

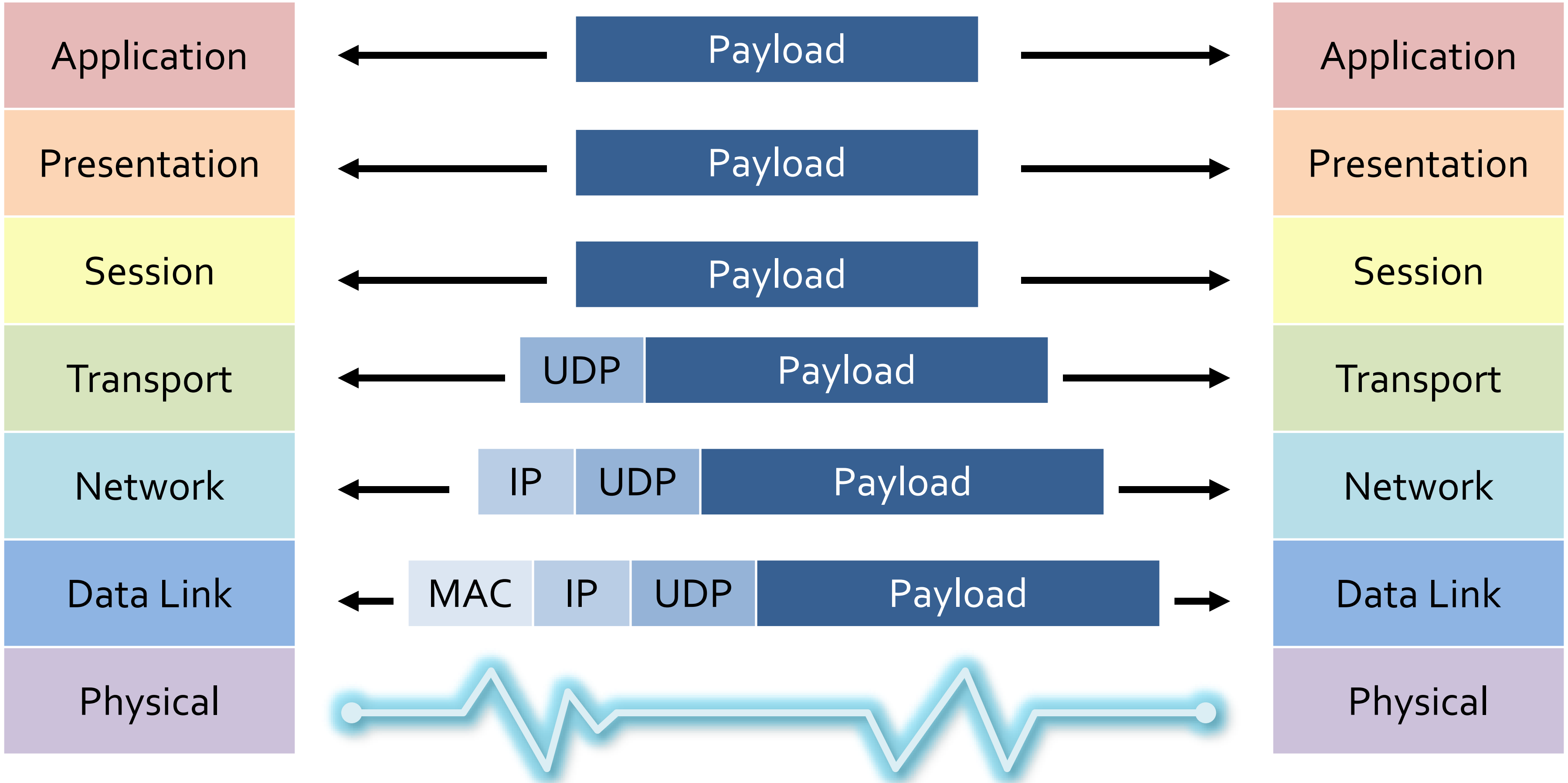
- The presentation layer is often optional
- It specifies how the data should appear
- This layer is responsible for character encoding (ASCII, UTF-8, etc.)
- MIME types are sometimes considered presentation layer issues

Application layer

- This is where the data is interpreted and used
- HTTP is an example of an application layer protocol
- A web browser takes the information delivered via HTTP and renders it
- Code you write deals significantly with the application layer

Transparency

- The goal of the OSI model is to make lower layers transparent to upper ones



Mnemonics

- Seven layers is a lot to remember
- Mnemonics have been developed to help

Application	All	All	A	Away
Presentation	Pros	People	Powered-Down	Pretzels
Session	Search	Seem	System	Salty
Transport	Top	To	Transmits	Throw
Network	Notch	Need	No	Not
Data Link	Donut	Data	Data	Dare
Physical	Places	Processing	Packets	Programmers

TCP/IP

- The OSI model is sort of a sham
  - It was invented after the Internet was already in use
  - You don't need all layers
  - Some people think this categorization is not useful
- Most network communication uses TCP/IP
- We can view TCP/IP as four layers:

Layer	Action	Responsibilities	Protocol
Application	Prepare messages	User interaction	HTTP, FTP, etc.
Transport	Convert messages to packets	Sequencing, reliability, error correction	TCP or UDP
Internet	Convert packets to datagrams	Flow control, routing	IP
Physical	Transmit datagrams as bits	Data communication	

TCP/IP

- A TCP/IP connection between two hosts (computers) is defined by four things
  - Source IP
  - Source port
  - Destination IP
  - Destination port
- One machine can be connected to many other machines, but the port numbers keep the different connections straight

Common port numbers

- Certain kinds of network communication are usually done on specific ports
  - **20** and **21**: File Transfer Protocol (FTP)
  - **22**: Secure Shell (SSH)
  - **23**: Telnet
  - **25**: Simple Mail Transfer Protocol (SMTP)
  - **53**: Domain Name System (DNS) service
  - **80**: Hypertext Transfer Protocol (HTTP)
  - **110**: Post Office Protocol (POP3)
  - **443**: HTTP Secure (HTTPS)

Upcoming

Next time…

- Socket communication

Reminders

- **Work on Project 3**
  - **Project 3 is now due on April 3**
- Keep reading Chapter 21