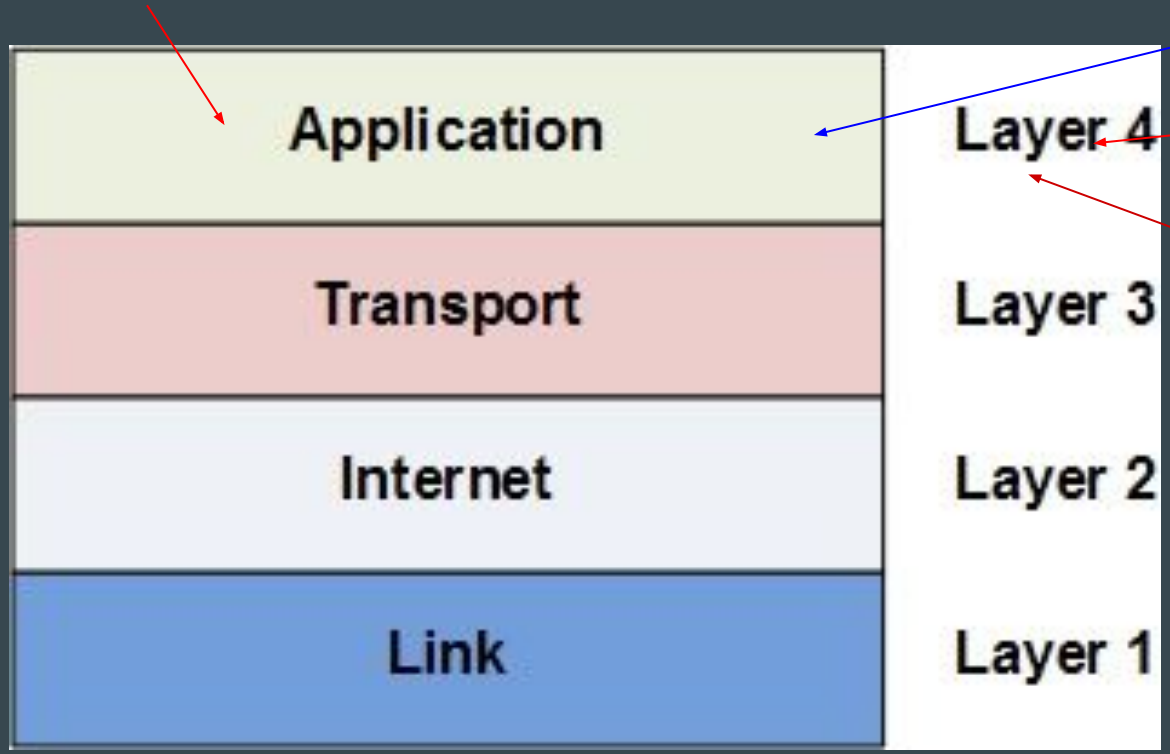
WebSockets

Team Socket Rocket: Anish Shenoy and Jasper Cheung


Application
Transport
Internet
Link
Layer 4
Layer 3
Layer 2
Layer 1

TCP/IP Model

PROTOCOL

=

1. Handshake
2. Data Transfer

HTTP:

- 1990s
- Client Driven Request-Response
- Headers are Bulky
- Calls a new connection for each request/response exchange.*
- :(

*long polling and other alternatives exist

- :(

WebSockets:

- 2008 - Michael Carter
- Hand Shake
  - Client sends a http request to server
  - Server sends http response back and upgrades
- Constant connection is opened and the client and server can send any number of messages to each other.

- Messages are mostly payload = Nice!

```
GET ws://websocket.example.com/ HTTP/1.1
Origin: http://example.com
Connection: Upgrade
Host: websocket.example.com
Upgrade: websocket
```

```
HTTP/1.1 101 WebSocket Protocol Handshake
Date: Wed, 16 Oct 2013 10:07:34 GMT
Connection: Upgrade
Upgrade: WebSocket
```

HTTP	Websockets
Half-Duplex/Request-Response	Full Duplex/Bi-Directional
A Lot of overhead per request	Moderate Overhead for the handshake then minimal for messages
Broad Support	Modern Languages and Clients

OMG WEBSOCKETS ARE AMAZING
...How Do I Use Them?

Socket.IO : A WebSocket API

- Uses Feature Detection to decide if the connection will be established with WebSocket, AJAX long polling, Flash, etc.
- Event Listeners :)
- User Friendly!

Socket.IO : A WebSocket API

The Client Can...

1) Create a Socket and bind it to an address
2) Add Event Listeners for messages from the server
3) Send messages to the server

What About the Server?

Socket.IO : A WebSocket API

There are Socket.IO libraries for nearly all major programming languages.

For Python Servers, there is Flask-SocketIO

```
$ pip install flask-socketio
```

Socket.IO : A WebSocket API

The Server Can...:

1) Create a Socket and wait for a client to connect

2) Add Event Listeners for messages from the client

3) Send Messages to the Client