**Websockets and Real-Time Web communication**

If you follow recent developments in web technologies you are likely to come across

the phrase real-time web.

This is not about so-called hard real-time where the delivery of computation results within a specified time interval is guaranteed.

The request/response design of HTTP alone makes that highly problematic, as evidenced by the fact that none of the approaches devised in the past have provided a satisfactory solution.

The *real-time web* is a network web using technologies and practices that enable users to receive information as soon as it is published by its authors, rather than requiring that they or their software check a source periodically for updates.

**Achieving duplex communication between clients and servers**

* Polling
* Long Polling
* Streaming
* AJAX
* Websockets

**Polling**

Polling can be defined as a method, which performs periodic requests regardless of the data that exists in the transmission.

The periodic requests are sent in a synchronous way.

The client makes a periodic request in a specified time interval to the Server. The response of the server includes available data or some warning message in it.

Constant requests slows the process.

**Long Polling**

Long polling, as the name suggests, includes similar technique like polling.

The client and the server keep the connection active until some data is fetched or timeout occurs.

If the connection is lost due to some reasons, the client can start over and perform sequential request.

Long polling is nothing but performance improvement over polling process, but constant requests may slow down the process.

**Streaming**

It is considered as the best option for real-time data transmission.

See WebRTC later

The server keeps the connection open and active with the client until and unless the required data is being fetched.

In this case, the connection is said to be open indefinitely.

Streaming includes HTTP headers which increases the file size, increasing delay.

This can be considered as a major drawback.

**AJAX**

AJAX is based on Javascript's XmlHttpRequest Object.

It is an abbreviated form of Asynchronous Javascript and XML.

XmlHttpRequest Object allows execution of the Javascript without reloading the complete web page.

AJAX sends and receives only a portion of the web page.

The major drawbacks of AJAX in comparison with Web Sockets are −

They send HTTP headers, which makes total size larger.

The communication is half-duplex.

The web server consumes more resources.

**Why Do We Need Web Sockets?**

The Internet was conceived to be a collection of Hypertext Mark-up Language (HTML) pages.

Richer items were added later, such as images, which are now also a part of the web fabric.

We need real-time bi-directional communication between a server and a client to avoid page reloads and to speed up data delivery of images etc.

This gave rise to the need of Web Sockets Protocol.

It gave rise to full-duplex communication bringing desktop-rich functionality to all web browsers.

**Websockets**

The WebSocket protocol was designed from the ground up to provide a practical solution to the problem of bidirectional data transmission on the web, allowing client and server to transmit messages at any time and, consequently, requiring them to handle message receipt asynchronously. (Most recent browsers support WebSocket as the client-side API of HTML5.)

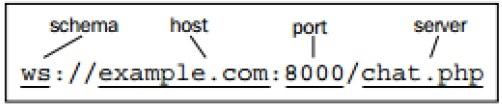
A two-way communication over a single TCP connection between the servers and the clients, which mean both the parties, communicate and exchange data at the same time - (not resquest/response like HTTP)

The key points of Web Sockets are true concurrency and optimization of performance, resulting in more responsive and rich web applications.

**Description of Web Socket Protocol**

URL

The following image shows the Web Socket URL in tokens.



**Browser Support**

The latest specification of Web Socket protocol is defined as RFC 6455 – a proposed standard.

RFC 6455 is supported by various browsers like Internet Explorer, Mozilla Firefox, Google Chrome, Safari, and Opera.

**Web Socket for Developers and Architects**

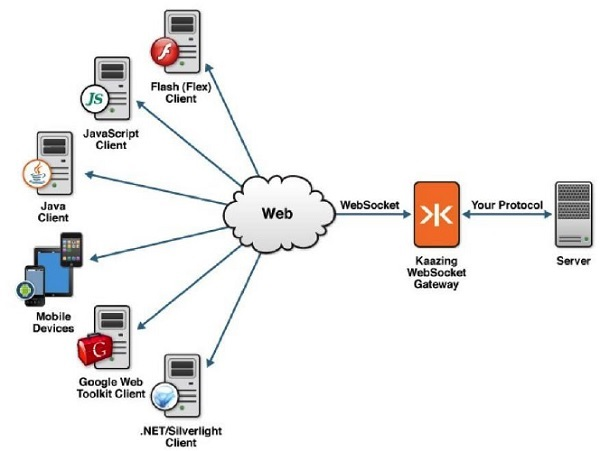
Web Socket is an independent TCP-based protocol.

Web Socket is a transport layer on top of which any other protocol can run.

Examples of such protocols include XMPP, STOMP, and AMQP

The only requirement on the browser-side is to run a JavaScript library that can interpret the Web Socket handshake, establish and maintain a Web Socket connection.

On the server side, the industry standard is to use existing protocol libraries that run on top of TCP and leverage a Web Socket Gateway.



Web Socket connections are initiated via HTTP; HTTP servers typically interpret Web Socket handshakes as an Upgrade request.

The Upgrade header field is an HTTP header field introduced in HTTP/1.1.

In the exchange, the client begins by making a cleartext request, which is later upgraded to a newer HTTP protocol version or switched to a different protocol like websockets.

(<https://en.wikipedia.org/wiki/HTTP/1.1_Upgrade_header#Use_with_WebSocket>)

**Functions of Web Sockets**

Web Sockets provide a connection between the web server and a client such that both the parties can start sending the data.

The steps for establishing the connection of Web Socket are as follows −

* The client establishes a connection through a process known as Web Socket handshake.
* The process begins with the client sending a regular HTTP request to the server.
* An Upgrade header is requested. In this request, it informs the server that request is for Web Socket connection.
* Web Socket URLs use the ws scheme. They are also used for secure Web Socket connections, which are the equivalent to HTTPs.

A simple example of initial request headers is as follows −

GET ws://websocket.example.com/ HTTP/1.1

Origin: <http://example.com>

Connection: Upgrade

Host: websocket.example.com

Upgrade: websocket

It is necessary to initialize the connection to the server from client for communication between them.

For initializing the connection, creation of Javascript object with the URL with the remote or local server is required.

var socket = new WebSocket(“ ws://echo.websocket.org ”);

The URL mentioned above is a public address that can be used for testing and experiments. The websocket.org server is always up and when it receives the message and sends it back to the client.

**Web Sockets – Events**

There are four main Web Socket API events −

* Open
* Message
* Close
* Error

Each of the events are handled by implementing the functions like onopen, onmessage, onclose and onerror functions respectively. It can also be implemented with the help of addEventListener method.

The protocol is designed to achieve the following benefits, which can be considered as the key points −

Reduce unnecessary network traffic and latency using full duplex through a single connection (instead of two).

Streaming through proxies and firewalls, with the support of upstream and downstream communication simultaneously.

**WebSocket Example**

Code from Netty in Action chapter 12

Github link: <https://github.com/normanmaurer/netty-in-action>

Instructions at end of document for running the code.

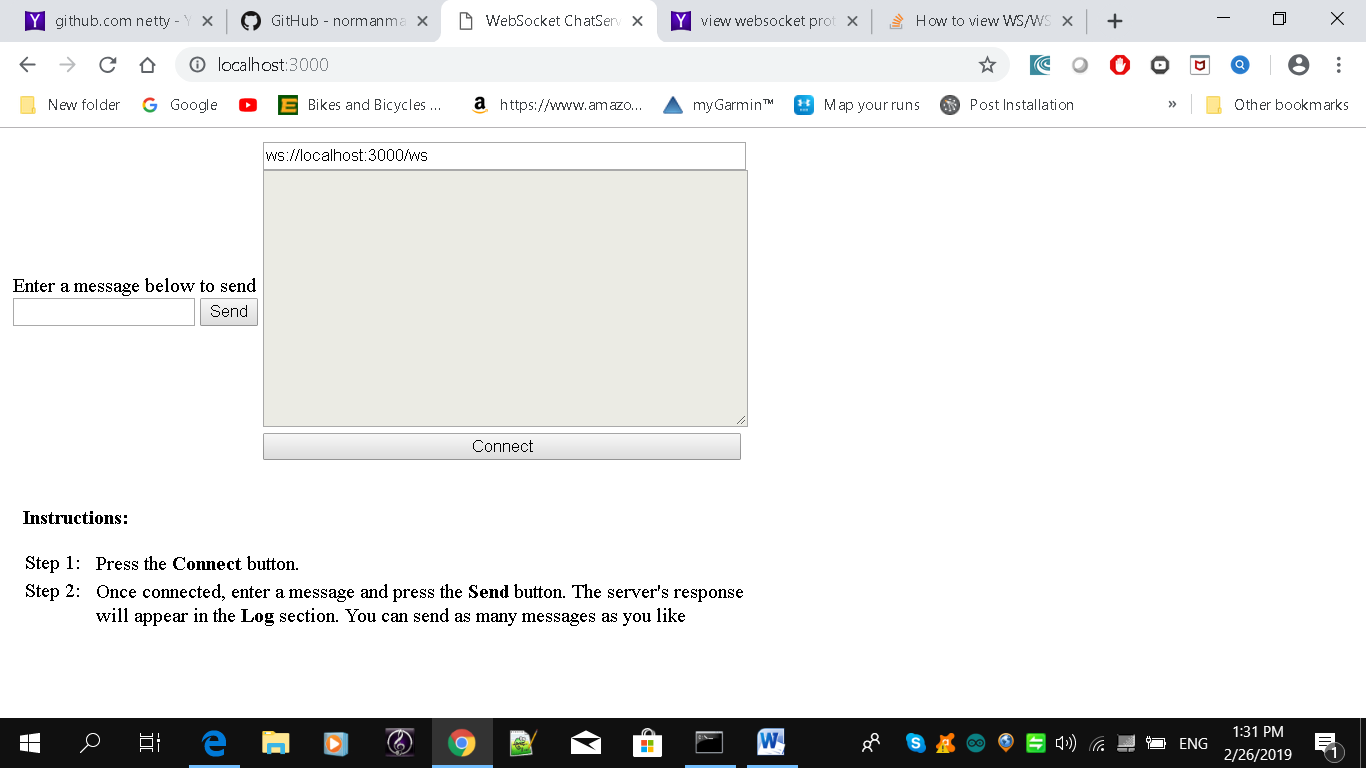
**A chat Server and Client**

HTTP uses a request/response where each round requires a new connection which incurs overhead. You could poll but still this will not achieve real-time connection.

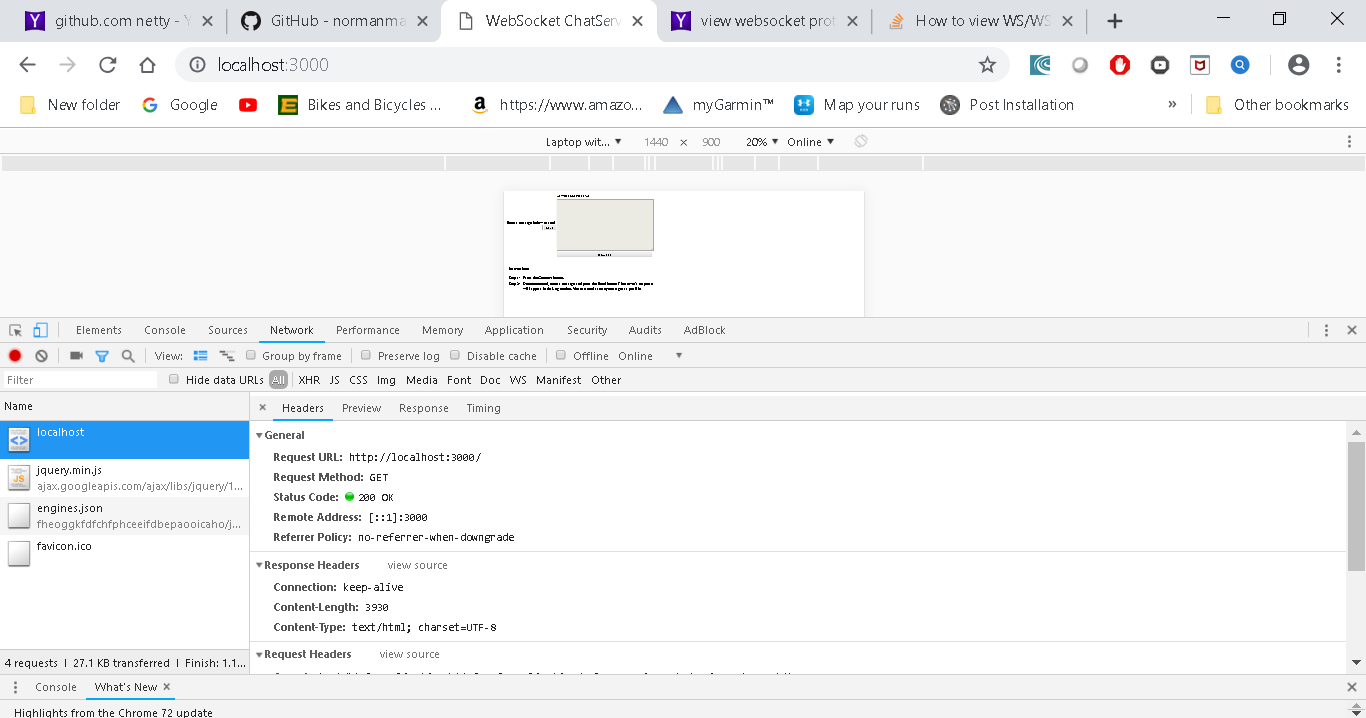
The definition or real-time for web applications essentially means that data is received immediately it is published rather than requiring that the software check periodically for updates (polling or long polling)

The server written to initially respond to HTTP requests

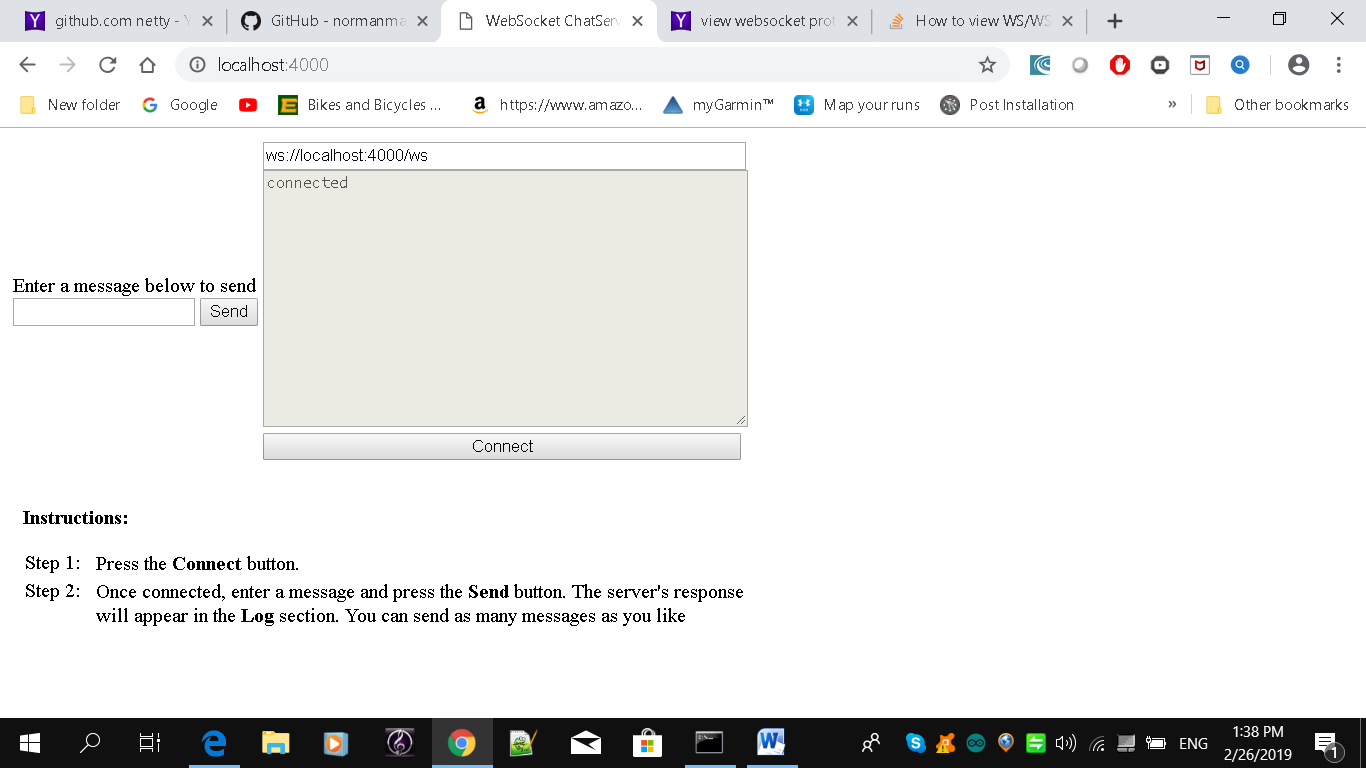
Here is the initial response to the client



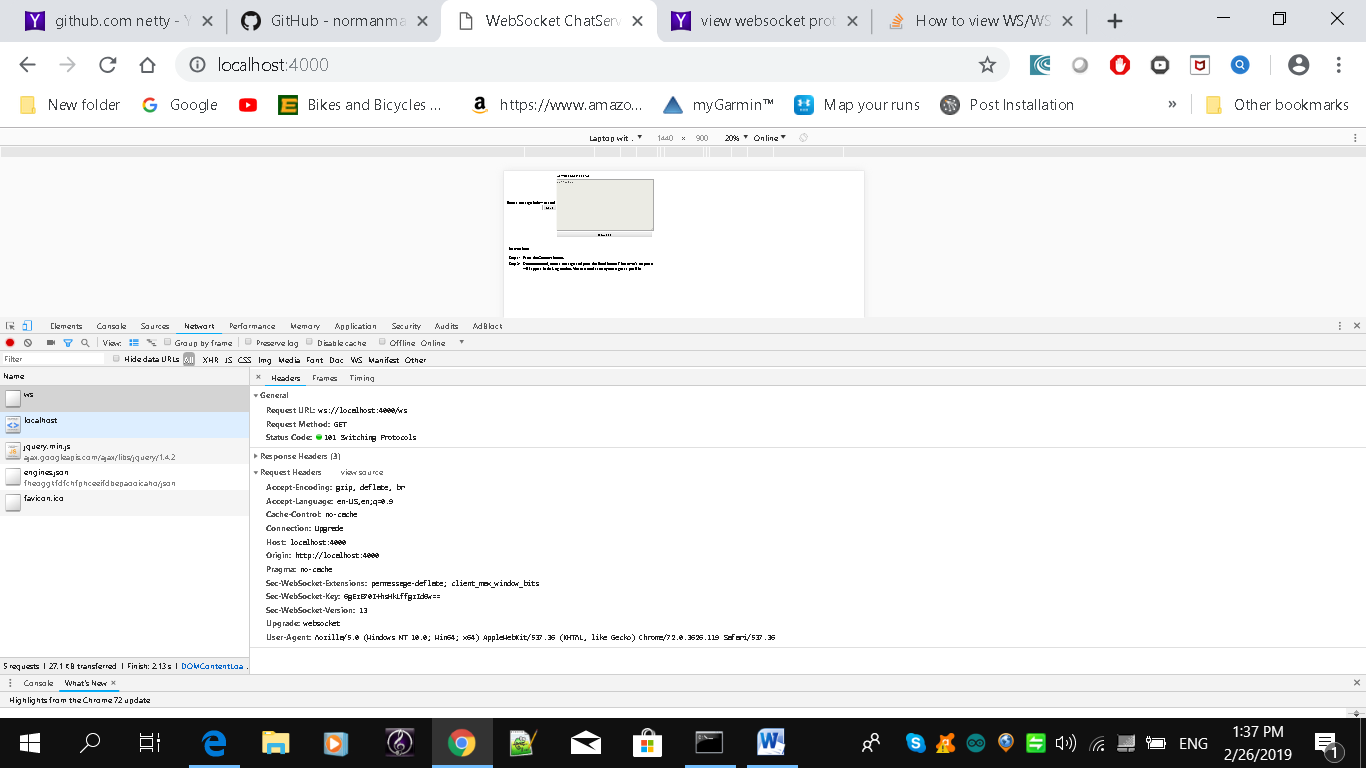
The initial GET message returns a text/html page above



However when we connect the protocol upgrades to websockets



The ws tab shows websocket messages (web sockets send data in frames)

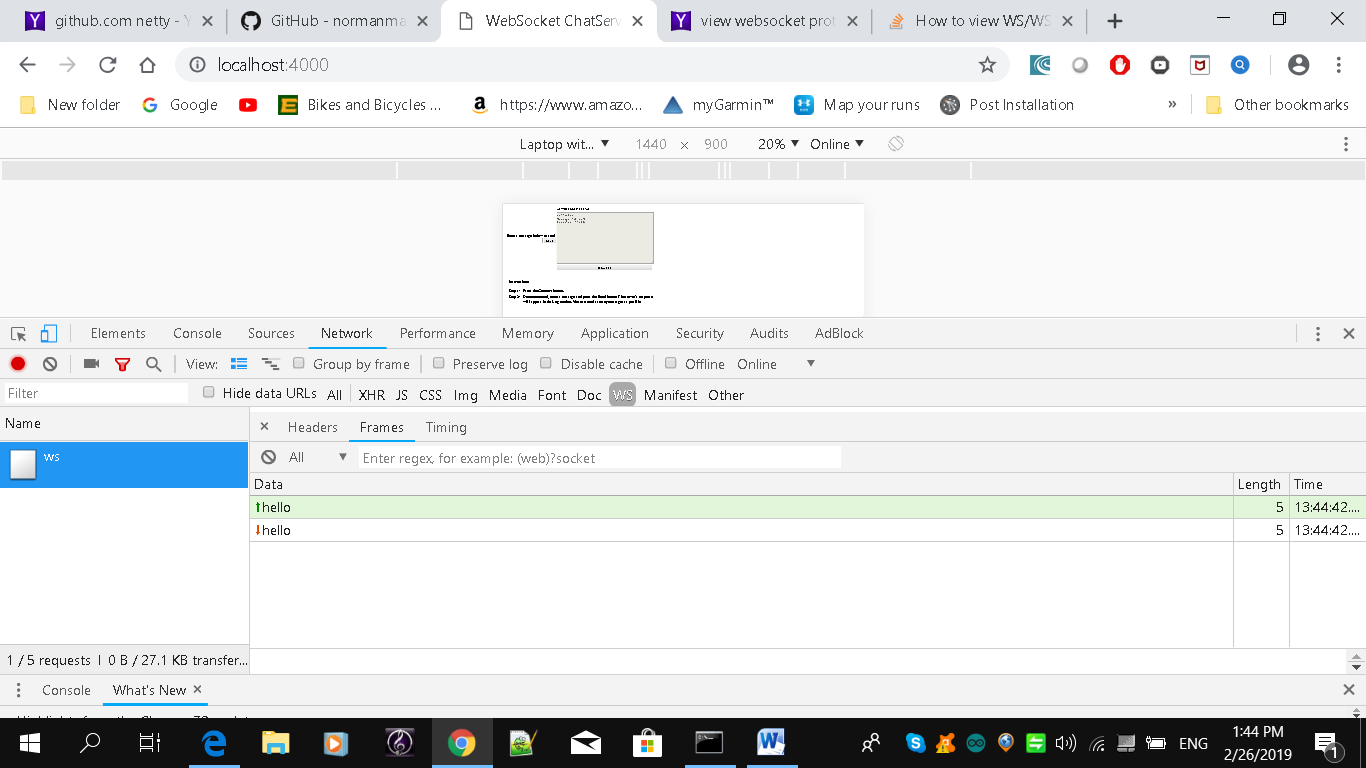


Note you can see the status code: switching protocols

You can see in the response header WebSocket-Version:13

Also Upgrade: websocket

The message “hello” is sent in a websocket frame. Here we can see the frames containing hello.



Frames allowed in websockets include binary frame, text frame, close frame, ping frame, pong frame, continuation frame.

**Running the example**

Download the book code

(Note on the githb site there were updates to the index.html and pom files in chapter 12 so update the application with these files first)

Goto chapter 12 and use maven

mvn –PChatServer –Dport=3000 clean package exec:exec

Access localhost:3000 from a browser

To write your own WebSocket application in Java/C# see: <https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API>

Reference:

<https://en.wikipedia.org/wiki/WebSocket>

<https://codeburst.io/polling-vs-sse-vs-websocket-how-to-choose-the-right-one-1859e4e13bd9>

(Note WebRTC Signalling Server exercise makes good use of Websockets)