What is web socket and how it is different from the HTTP?

Difficulty Level : Easy • Last Updated : 21 Feb, 2022

HTTP and WebSocket both are communication protocols used in client-server communication.

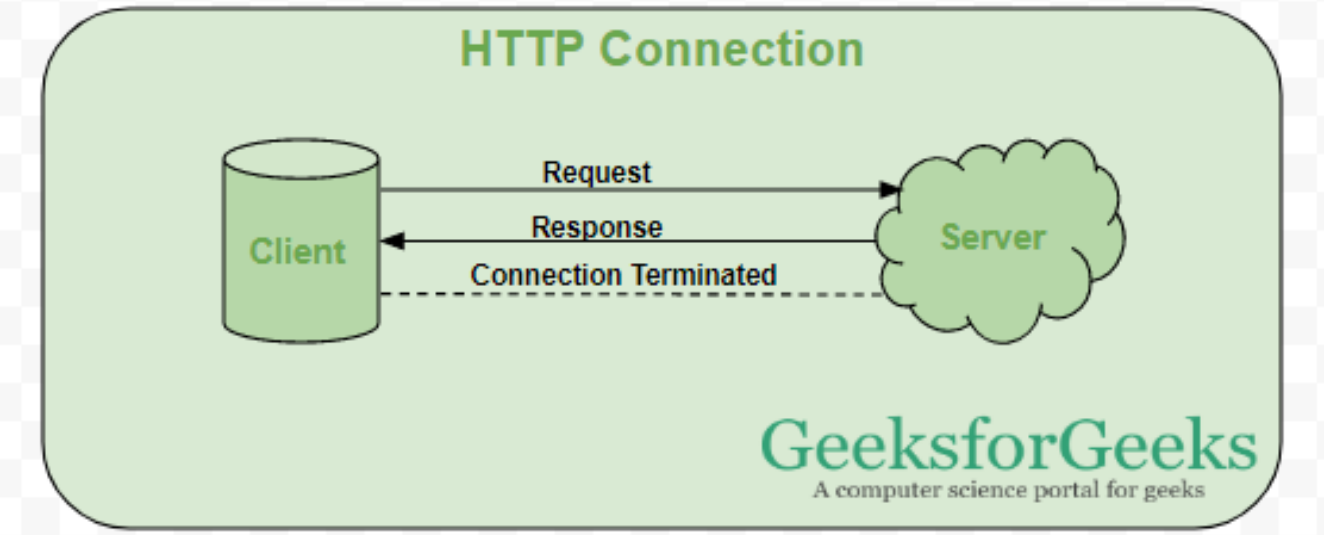
HTTP protocol: HTTP is unidirectional where the client sends the request and the server sends the response. Let's take an example when a user sends a request to the server this request goes in the form of HTTP or HTTPS, after receiving a request server send the response to the client, each request is associated with a corresponding response, after sending the response the connection gets closed, each HTTP or HTTPS request establish the new connection to the server every time and after getting the response the connection gets terminated by itself.
HTTP is a stateless protocol that runs on top of TCP which is a connection-oriented protocol it guarantees the delivery of data packet transfer using the three-way handshaking methods and re-transmits the lost packets.

HTTP can run on top of any reliable connection-oriented protocol such as TCP, SCTP. When a client sends an HTTP request to the server, a TCP connection is open between the client and server and after getting the response the TCP connection gets terminated, each HTTP request opens a separate TCP connection to the server, for e.g. if the client sends 10

getting the response/fallback.

HTTP message information encoded in ASCII, each HTTP request message composed HTTP protocol version(HTTP/1.1, HTTP/2), HTTP methods (GET/POST, etc.), HTTP headers (content type, content length), host information, etc. and the body which contain the actual message which is being transferred to the server. HTTP headers varied from 200 bytes to 2 KB in size, the common size of HTTP header is 700-800 bytes. When a web

features of the agent it reduces the HTTP header payload.

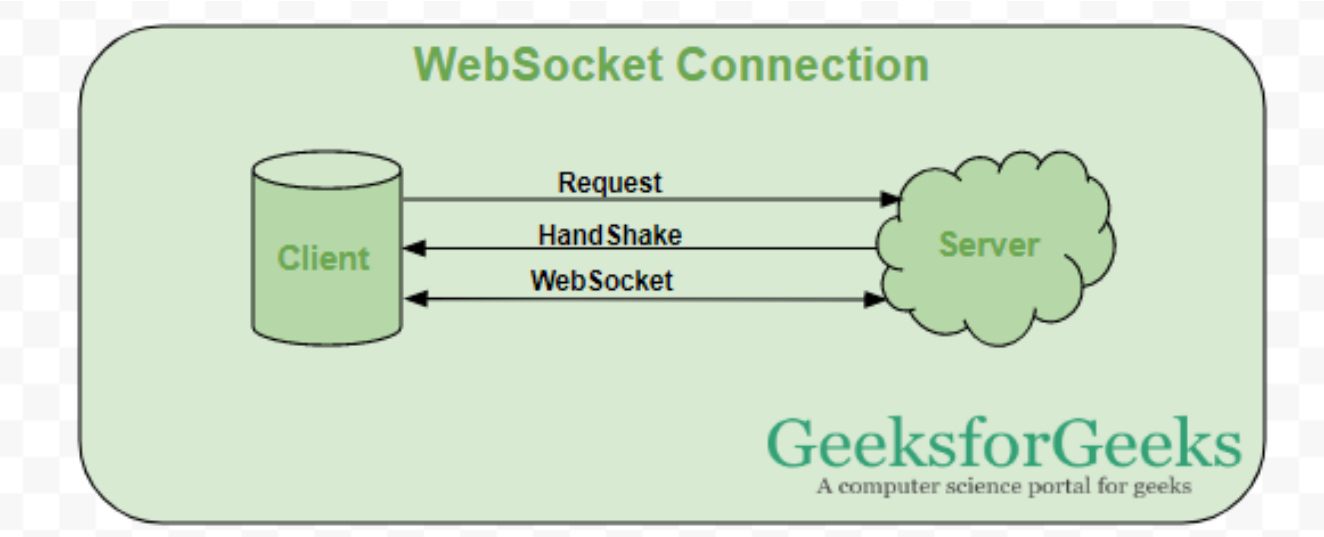


WebSocket: WebSocket is bidirectional, a full-duplex protocol that is used in the same scenario of client-server communication, unlike HTTP it starts from **ws://** or **wss://**. It is a stateful protocol, which means the connection between client and server will keep alive until it is terminated by either party (client or server). After closing the connection by either of the client and server, the connection is terminated from both ends.

Let's take an example of client-server communication, there is the client which is a web browser and a server, whenever we initiate the connection between client and server, the client-server made the handshaking and decide to create a new connection and this connection will keep alive until terminated by any of them. When the connection is established and alive the communication takes place using the same connection channel until it is terminated.

This is how after client-server handshaking, the client-server decide on a new connection to keep it alive, this new connection will be known as WebSocket. Once the communication link establishment and the connection are opened, message exchange will take place in bidirectional mode until connection persists between client-server. If anyone of them

way in which socket works is slightly different from how HTTP works, the status code 101 denotes the switching protocol in WebSocket.



When can a web socket be used:

- **Real-time web application:** Real-time web application uses a web socket to show the data at the client end, which is continuously being sent by the backend server. In WebSocket, data is continuously pushed/transmitted into the same connection which is already open, that is why WebSocket is faster and improves the application performance.
  For e.g. in a trading website or bitcoin trading, for displaying the price fluctuation and movement data is continuously pushed by the backend server to the client end by using a WebSocket channel.

- **Gaming application:** In a Gaming application, you might focus on that, data is continuously received by the server, and without refreshing the UI, it will take effect on the screen, UI gets automatically refreshed without even establishing the new connection, so it is very helpful in a Gaming application.

- **Chat application:** Chat applications use WebSockets to establish the connection only once for exchange, publishing, and broadcasting the message among the subscribers. It reuses the same WebSocket connection, for sending and receiving the message and for one-to-one message transfer.

When not to use WebSocket: WebSocket can be used if we want any real-time updated or continuous streams of data that are being transmitted over the network. If we want to fetch

with **HTTP protocol**, old data which is not required very frequently or fetched only once can be queried by the simple HTTP request, so in this scenario, it's better not use WebSocket.

Note: RESTful web services are sufficient to get the data from the server if we are loading the data only once.

Differences between HTTP and WebSocket Connection:

WebSocket Connection	HTTP Connection
WebSocket is a bidirectional communication protocol that can send the data from the client to the server or from the server to the client by reusing the established connection channel. The connection is kept alive until terminated by either the client or the server.	The HTTP protocol is a unidirectional protocol that works on top of TCP protocol which is a connection-oriented transport layer protocol, we can create the connection by using HTTP request methods after getting the response HTTP connection get closed.
Almost all the real-time applications like (trading, monitoring, notification) services use WebSocket to receive the data on a single communication channel.	Simple RESTful application uses HTTP protocol which is stateless.
All the frequently updated applications used WebSocket because it is faster than HTTP Connection.	When we do not want to retain a connection for a particular amount of time or reuse the connection for transmitting data; An HTTP connection is slower than WebSockets.

Note: Depending on your project you have to choose where it will be WebSocket or HTTP Connection.

Related Articles

1. Node.js http.ClientRequest.socket Property
2. Node.js http.ServerResponse.socket Api

3. Web-Socket in Node.js
4. Why are HTTP cookies used by Node.js for sending and receiving HTTP cookies?
5. Difference between HTTP/2 and HTTP/1.1
6. Comparison Between Web 1.0, Web 2.0 and Web 3.0
7. Will the 'error' event ever be emitted on 'http.IncomingMessage' in a node.js http.request ?
8. Different types of module used for performing HTTP Request and Response in Node.js
9. How to Intercept HTTP requests in web extension ?
10. User to User private Chat App using ReactJS and Firebase | Without socket programming

Next

Socket Programming in C/C++

Article Contributed By :

ArpitAsati
@ArpitAsati

Vote for difficulty

Current difficulty : Easy

Easy | Normal | Medium | Hard | Expert

Improved By : harshalrathore2014, sagar0719kumar, simmytarika5, tarunpai

Article Tags : GBlog, Web Technologies

GeeksforGeeks
A-143, 9th Floor, Sovereign Corporate Tower,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

Database Management System

Software Engineering

Digital Logic Design

Computer Graphics

Engineering Maths

Machine Learning Tutorial

Maths For Machine Learning

Pandas Tutorial

NumPy Tutorial

OpenCV Python Tutorial

Interview Corner

Company Preparation

Preparation for SDE

Company Interview Corner

Experienced Interview

Internship Interview

Competitive Programming

Python

Python Tutorial

Python Programming Examples

Django Tutorial

Python Projects

Python Tkinter

OpenCV Python Tutorial

School [Class 6-12]

CBSE Notes for Class 8

CBSE Notes for Class 9

CBSE Notes for Class 10

CBSE Notes for Class 11

CBSE Notes for Class 12

English Grammar

UPSC/SSC/BANKING

SSC CGL Syllabus

SBI PO Syllabus

IBPS PO Syllabus

UPSC Ethics Notes

UPSC Economics Notes

UPSC History Notes