Lesson 1: How the Web Works

Lesson Overview

Every time you open a website, several technologies work together behind the scenes to deliver the page to your screen. Understanding these technologies will help you become a better web developer because you'll know not only how to build websites, but also how they work.

In this lesson, you will learn about web browsers, the internet, the World Wide Web, domain names, DNS, HTTP/HTTPS, and web hosting.

Learning Objectives

By the end of this lesson, you should be able to:

- Explain the difference between the internet and the World Wide Web.
- Describe the role of a web browser.
- Explain what domain names and DNS are.
- Describe how HTTP and HTTPS enable communication between browsers and servers.
- Explain what web hosting is.
- Describe the basic process that occurs when someone visits a website.

1. Understanding the Internet and the World Wide Web

What is the Internet?

The internet is a global network of interconnected computer networks and devices that communicate using agreed-upon protocols.

It enables services such as email, online gaming, video calls, file transfers, and the World Wide Web.

What is the World Wide Web?

The World Wide Web, commonly called the web, is a system of interconnected resources that we access over the internet using web browsers.

These resources include websites, webpages, images, videos, and web applications.

Internet vs. World Wide Web

The internet is the underlying network infrastructure, while the web is one of the services that uses it.

For example, when you visit an online learning platform in your browser, you are using the web over the internet.

2. Web Browsers

A web browser is a software application that allows users to access and interact with websites and web applications.

Examples include:

- Google Chrome
- Mozilla Firefox
- Microsoft Edge
- Apple Safari

What does a browser do?

A browser can:

- Accept a URL entered by the user.
- Locate and request web resources.
- Receive responses from web servers.
- Interpret HTML to structure a webpage.
- Apply CSS to style the page.
- Execute JavaScript to provide interactivity.
- Display the resulting webpage to the user.

For example, when you visit a webpage containing HTML, CSS, and JavaScript, the browser processes these resources to present the website as an interactive interface.

3. URLs and Domain Names

What is a URL?

URL stands for Uniform Resource Locator. It is an address used to identify a resource on the web.

Example:

```
https://www.example.com/about
```

This URL contains several parts:

- `https` — the scheme or protocol.
- `www.example.com` — the hostname.
- `/about` — the path to a resource on the website.

A URL may also contain a port, query parameters, or a fragment.

What is a domain name?

A domain name is a human-readable name used to identify an internet resource.

Examples include:

- `google.com`
- `github.com`
- `example.com`

Domain names are easier for people to remember than numerical IP addresses.

A domain name is not the website itself. It provides a convenient name that can be configured to direct users to the appropriate internet services.

4. Domain Name System (DNS)

DNS stands for Domain Name System.

It is a distributed naming system that helps translate domain names into information needed to locate internet services, commonly including IP addresses.

What is an IP address?

An IP address is a numerical or alphanumeric address assigned to a network interface and used to identify it for internet protocol communication.

Examples:

- IPv4: `192.0.2.1`
- IPv6: `2001:db8::1`

These are documentation examples, not necessarily the addresses of real websites.

How does DNS work?

When you enter a domain name into your browser, the browser or operating system may first check previously cached DNS information. If a suitable answer is unavailable, a DNS lookup may be performed.

A simplified DNS process is:

1. You enter a domain name.
2. Your device checks available cached information.
3. A DNS resolver obtains the required DNS information if necessary.
4. The resolver returns an appropriate answer, often including an IP address.
5. The browser uses the available connection information to contact the destination server.

DNS does not transfer the webpage itself. Its role is to help the device find the appropriate internet service.

5. HTTP and HTTPS

HTTP stands for Hypertext Transfer Protocol.

It defines how clients, such as web browsers, and servers communicate through requests and responses.

How HTTP works

A simplified request-response cycle looks like this:

1. The browser sends an HTTP request.
2. The server processes the request.
3. The server returns an HTTP response.
4. The browser processes the response and displays the appropriate content.

For example, a browser might request an HTML document, and the server may respond with that document and a status code.

Common HTTP status codes

- `200 OK` — the request succeeded.
- `301 Moved Permanently` — the resource has a permanent redirect.
- `404 Not Found` — the requested resource could not be found.
- `500 Internal Server Error` — the server encountered an unexpected error.

What is HTTPS?

HTTPS is HTTP secured using encryption through TLS (Transport Layer Security).

It helps protect data exchanged between the browser and server against interception and tampering and allows the browser to authenticate the server's identity.

HTTPS is especially important when users log in, submit forms, or exchange sensitive information.

Remember: HTTPS does not automatically mean that a website is trustworthy. It protects the connection, not the honesty of the website operator.

6. Web Servers and Web Hosting

What is a web server?

The term web server can refer to software that handles HTTP requests or the computer on which that software runs.

When a browser requests a resource, the server may return a file, generate a response using application code, or retrieve information from a database.

What is web hosting?

Web hosting is a service or arrangement that makes website files and application services available online.

Hosting providers supply infrastructure and services that allow websites to be accessed over a network.

Examples of hosting platforms and services include:

- Vercel
- Netlify

- Amazon Web Services (AWS)
- DigitalOcean

Different websites have different hosting needs. A simple static website may require only static hosting, while a dynamic application may need a backend server, a database, and additional infrastructure.

Domain names vs. hosting

A domain name provides a human-readable address. Hosting provides the infrastructure that serves the website or application.

They work together, but they are different things.

You can register a domain with one provider and host your website with another.

7. What Happens When You Visit a Website?

Let's bring everything together.

Suppose you enter `https://example.com` into your browser.

A simplified sequence is:

1. **URL processing:** The browser interprets the address you entered.
2. **DNS resolution:** The browser or operating system obtains the destination's IP address or other required connection information.
3. **Connection establishment:** The browser establishes a suitable network connection. For HTTPS, this includes setting up a secure TLS connection, unless an existing connection can be reused.
4. **HTTP request:** The browser sends a request for the resource.
5. **Server processing:** The server handles the request and prepares a response.
6. **HTTP response:** The server sends back the requested resource or another appropriate response.
7. **Resource loading:** The browser may request additional files, such as CSS, JavaScript, images, and fonts.
8. **Rendering:** The browser processes the resources and renders the webpage.

The exact sequence can vary because browsers cache resources, reuse connections, follow redirects, and load some resources in parallel.

8. Why This Matters to Frontend Developers

As a frontend developer, you will build the parts of websites that users see and interact with.

Understanding how the web works helps you:

- Understand why a website might fail to load.
- Recognize common HTTP status codes.
- Understand why domain and DNS configuration matters during deployment.
- Know how browsers load HTML, CSS, JavaScript, and other resources.
- Understand how frontend applications communicate with backend services.
- Troubleshoot problems using browser developer tools.

You do not need to master every networking detail immediately. Build a strong foundation and deepen your knowledge as you develop more complex applications.

Key Terms

Term	Definition
Internet	A global network of interconnected networks.
World Wide Web	A system of interconnected resources accessed over the internet.
Browser	Software used to access and interact with web content.
URL	An address identifying a resource on the web.
Domain name	A human-readable name used to identify an internet resource.
DNS	A distributed system that resolves domain names into relevant information.
IP address	An address used for communication through the Internet Protocol.
HTTP	A protocol for exchanging requests and responses between clients and servers.
HTTPS	HTTP secured using TLS.
Web server	Software or a computer that serves web resources.
Web hosting	Infrastructure or a service that makes a website or application available online.
Rendering	The process by which a browser turns web resources into a displayed page.

Knowledge Check

Answer these questions in your own words.

1. What is the difference between the internet and the World Wide Web?
2. What are three things a web browser does?
3. What is the purpose of a domain name?
4. What does DNS do, and why is it needed?
5. What is the difference between HTTP and HTTPS?
6. What is the difference between a domain name and web hosting?
7. What happens after a browser sends a request to a web server?
8. Why should a frontend developer understand HTTP status codes?

Practical Activity

Explore a website using your browser

1. Open Google Chrome or another modern browser.
2. Visit a website you use regularly.
3. Identify the domain name in the address bar.
4. Check whether the website uses HTTPS.
5. Open Developer Tools by pressing `F12`.
6. Select the **Network** tab.
7. Reload the webpage.
8. Select a request and inspect its status code, request URL, and resource type.

Write down your observations and identify at least three different resources the browser loads.

Lesson Summary

The web depends on multiple technologies working together. Browsers request resources, DNS helps locate internet services, domain names provide human-readable addresses, HTTP and HTTPS enable client-server communication, and hosting makes website resources available.

Understanding these fundamentals prepares you for the next lesson, where you will explore frontend development, backend development, and how APIs connect different parts of an application.