ARBA MINCH UNIVERSITY

INSTITUTE OF TECHNOLOGY

FACULTY OF COMPUTING AND SOFTWARE ENGINEETING

Web design and Programming Lecture Notes

By: Chala Simon

ARBA MINCH UNIVERSITY

INSTITUTE OF TECHNOLOGY

FACULTY OF COMPUTING AND SOFTWARE ENGINEERING

Course Title Web Design and Programming

Course Code SEng7131

CP 5 (2hr Lecture, 3hr Laboratory)

Module Title Internet and Web Technology

Learning Outcomes:

At the end of this course the students will be able to:

- Build websites, web frameworks using basic HTML, CSS and JavaScript
- Build client-side application that responsive
- Build web application that consist AJAX and Server-Side
- Build Web APIs that are read for many client-side applications
- Build applications with database and ORM/ODM integrations
- Build fully integrated enterprise application

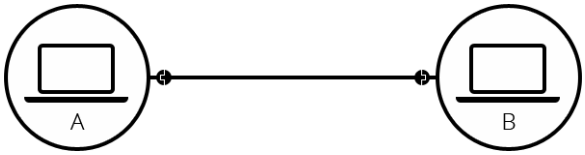
Table of Contents

OVERVIEW OF THE INTERNET AND WWW

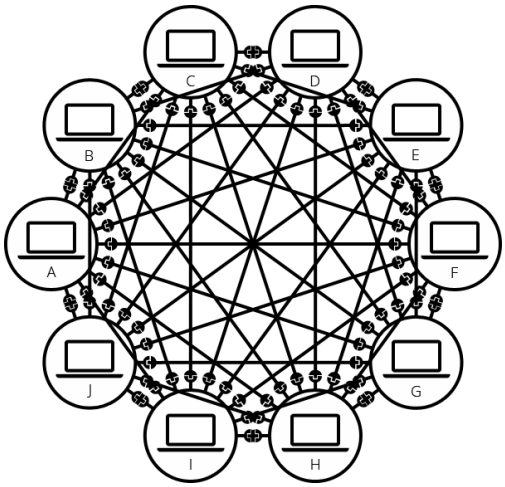
Introduction to Internet

What is Network?

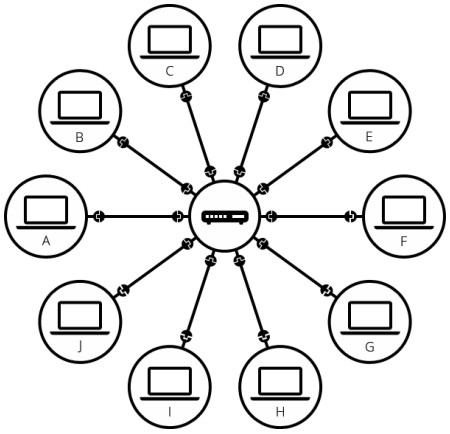
- A computer network is a set of computers that share resources and communicate with each other using common protocols over digital connections.
- When two or devices need to communicate, you have to link them, either physically (usually with an Ethernet cable) or wirelessly (for example with WiFi or Bluetooth systems).
- All modern computers can sustain any of those connections.



- You can connect as many computers as you wish. But it gets complicated quickly. If you're trying to connect, say, ten computers, you need 45 cables, with nine plugs per computer!

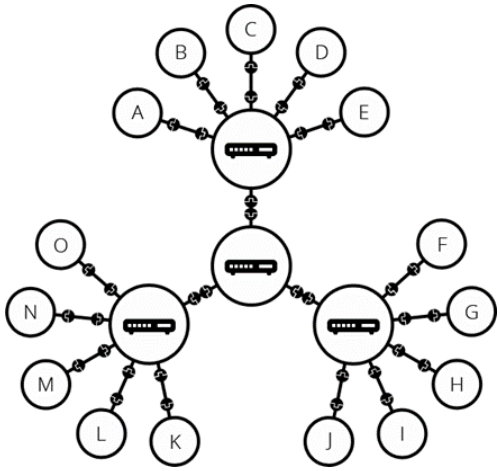


- To solve this problem, each computer on a network is connected to a special tiny computer called a **router**. This router has only one job: like a signaler at a railway station, it makes sure that a message sent from a given computer arrives at the right destination computer. To send a message to computer B, computer A must send the message to the router, which in turn forwards the message to computer B and makes sure the message is not delivered to computer C.
- Once we add a router to the system, our network of 10 computers only requires 10 cables: a single plug for each computer and a router with 10 plugs.

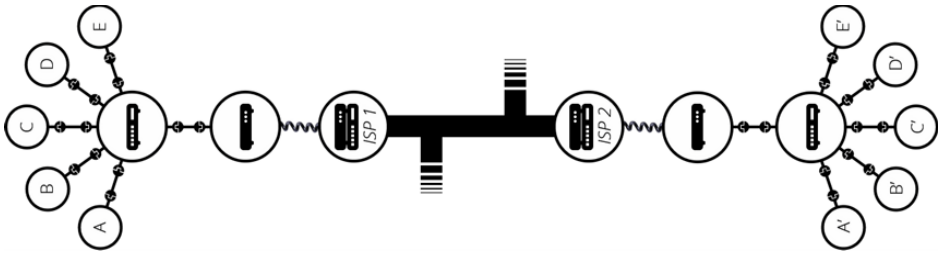


A network of networks

By connecting computers to routers, then routers to routers, we are able to scale infinitely.



- Such a network comes very close to what we call the Internet, but we're missing something. How can we connect and handle the connection between our network and rest of world? Therefore, we need more infrastructure.
- To do that, we will connect our network to an Internet Service Provider (ISP). An ISP is a company that manages some special routers that are all linked together and can also access other ISPs' routers. So, the message from our network is carried through the network of ISP networks to the destination network. The Internet consists of this whole infrastructure of networks.



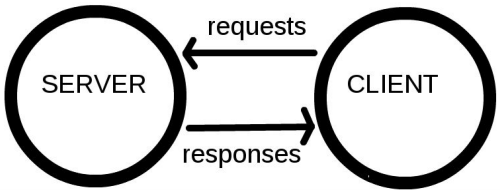
- The Internet (or internet) is the global system of interconnected computer networks that uses the Internet protocol suite (TCP/IP) to communicate between networks and devices.
- It is a network of networks that consists of private, public, academic, business, and government networks of local to global scope, linked by a broad array of electronic, wireless, and optical networking technologies.
- The Internet carries a vast range of information resources and services, such as the inter-linked hypertext documents and applications of the World Wide Web (WWW), electronic mail, telephony, and file sharing.

Internet and the web

The Internet is a technical infrastructure which allows billions of computers to be connected all together. Among those computers, some computers (called Web servers) can send messages intelligible to web browsers. The Internet is an infrastructure, whereas the Web is a service built on top of the infrastructure. It is worth noting there are several other services built on top of the Internet, such as email.

How the Web works

Computers connected to the web are called clients and servers. A simplified diagram of how they interact might look like this:



- Clients are the typical web user's internet-connected devices (for example, your computer connected to your Wi-Fi, or your phone connected to your mobile network) and web-accessing software available on those devices (usually a web browser like Firefox or Chrome).

- Servers are computers that store webpages, sites, or apps. When a client device wants to access a webpage, a copy of the webpage is downloaded from the server onto the client machine to be displayed in the user's web browser.

Terms and technologies

- ***web page***

  A document which can be displayed in a web browser such as Firefox, Google Chrome, Opera, Microsoft Internet Explorer or Edge, or Apple's Safari. These are also often called just "pages."

- ***Website***

  A **website** (also written as **web site**) is a collection of web pages and related content that is identified by a common domain name and published on at least one web server. The first page of a website is called home page.

  - Static websites:

  ones that are fixed and display the same content for every user, usually written exclusively in HTML.

  - Dynamic websites:

  one that can display different content and provide user interaction, by making use of advanced programming and databases in addition to HTML

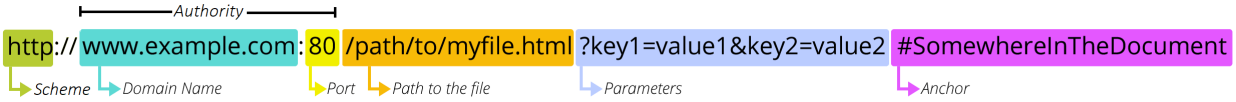
- ***web browser***

  A piece of software such as Mozilla Firefox, Google Chrome and Microsoft Edge that allows a computer to access and display documents, view pictures, hear sound, and view video clips from the World Wide Web.

- ***web server***

  A web server is software and hardware that uses HTTP (Hypertext Transfer Protocol) and other protocols to respond to client requests made over the World Wide Web.
  The main job of a web server is to display web content through storing, processing and delivering webpages to users.

- ***search engine***
  A web service that helps you find other web pages, such as Google, Bing, Yahoo, or DuckDuckGo. Search engines are normally accessed through a web browser (e.g., you can perform search engine searches directly in the address bar of Firefox, Chrome, etc.) or through a web page (e.g., bing.com or duckduckgo.com).
- ***Hypertext Transfer Protocol (HTTP)***
  It is the set of rules by which Web pages are transferred across the Internet.
- ***Hypertext***
  It's a text which contains links to other texts. Hypertext documents are interconnected by hyperlinks, which are typically activated by a mouse click, keypress set, or screen touch.
- ***Hyperlink***
  It's a text, images, graphics that, when clicked with a mouse (or activated by keystrokes) will connect the user to a new Web site. The link is usually obvious, such as underlined text or a "button" of some type, but not always.
- ***HTML***
  HTML is standard Markup Language for documents designed to be displayed in a web-browser. It is the most basic building block of the Web. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/behavior (JavaScript). Markup Language is a computer language that uses tags to define elements within a document.
- ***Uniform Resource Locator (URL)***
  URL is one of the key concepts of the Web. It is the mechanism used by browsers to retrieve any published resource on the web.



A URL for HTTP (or HTTPS) is normally made up of *three* or *four* components:

- A scheme - identifies the protocol to be used to access the resource on the Internet. It can be HTTP (without SSL) or HTTPS (with SSL).
- Host name - is the domain name which identifies the host that holds the resource.
- A path – is the path name which identifies the specific resource in the host that the web client wants to access.
- A query string - If a query string is used, it follows the path component, and provides a string of information that the resource can use for some purpose (for example, as parameters for a search or as data to be processed).

```
scheme://host:port/path?query
```

Computer programming and web language

Computer programming languages allow us to give instructions to a computer in a language the computer understands. Any of various languages for expressing a set of detailed instructions for a digital computer. A programming language is a computer language programmers use to develop software programs, scripts, or other sets of instructions for computers to execute.

Web development is coding or programming, which enables website functionality based on client requirements and deals with building websites' non-design aspect. Web development can be used for complex web-based applications, social network applications and electronic business applications based on the requirement.

Web development is as follows:

- Client-side coding
- Server-side coding
- Database technology

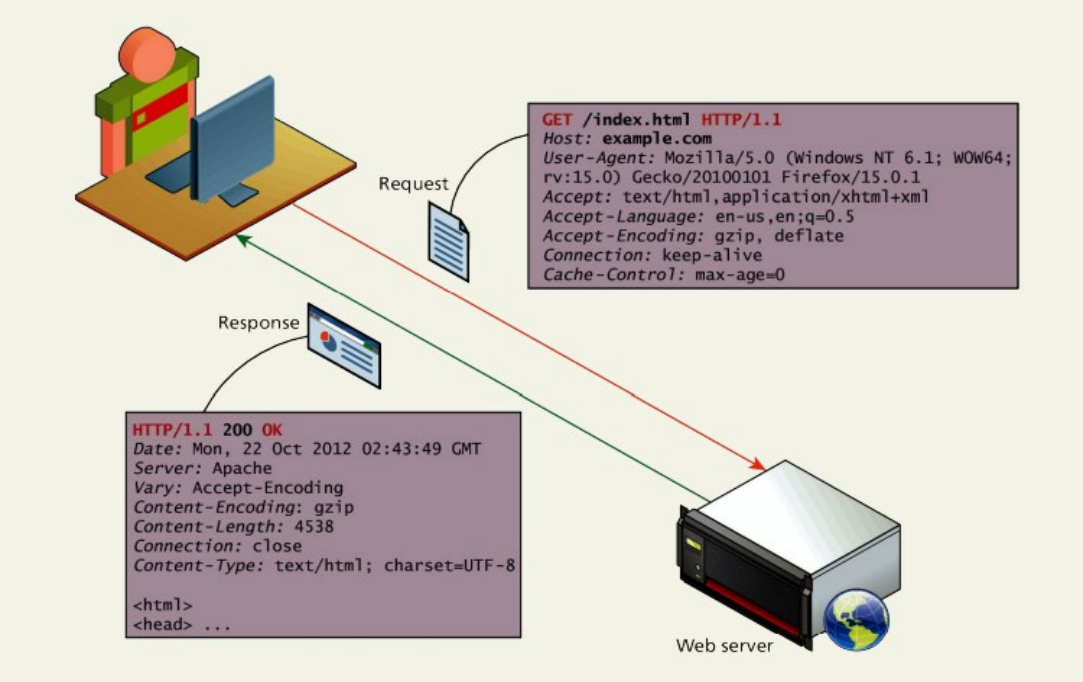
- Front-end development is of constructing what a user sees when they load a web application.
- Back-end development mainly controls on the backend of a web application. A back-end often uses a database to generate the front-end.

Programming vs Scripting

Programming Languages	Scripting Languages
A programming language is a formal language that specifies a set of instructions that can be used to produce various kinds of output. It Consists of instructions for a computer.	**A programming language for a runtime system** that automates the execution of tasks that would otherwise be performed individually by a human operator.
compiler- based language	Interpreter based Language
compressed into smaller packages that do not need to be interpreted by another language or application.	written in one language and interpreted within another program. For instance, JavaScript has to be incorporated within HTML, which the Internet browser will then interpret.
Hosting is not required	Require hosting

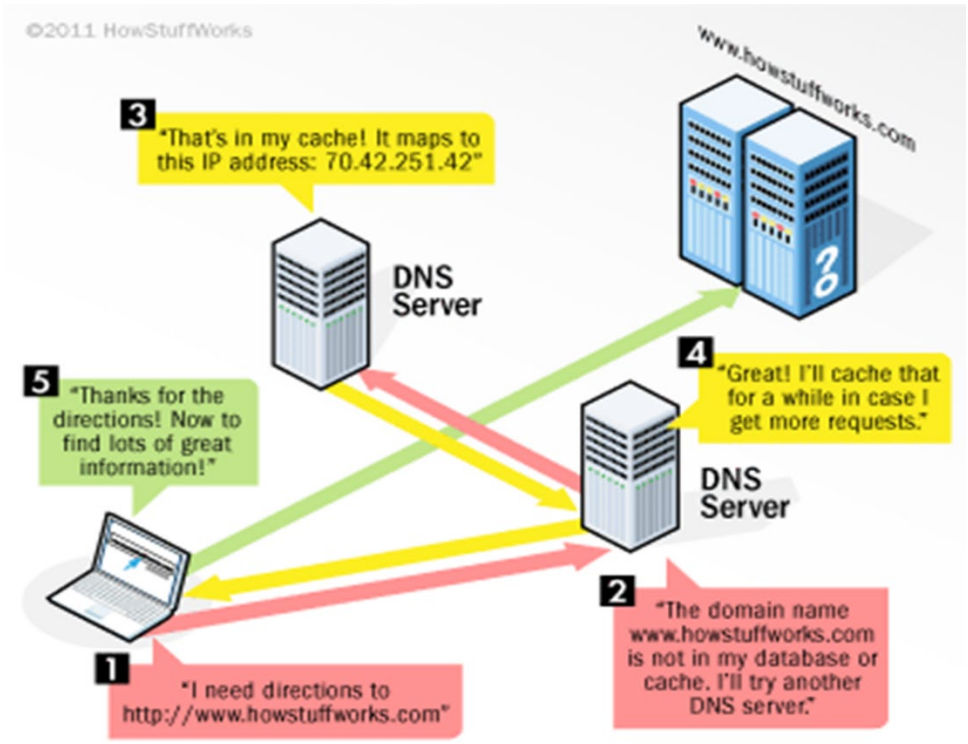
e.g. C, C++, C#, Java, VC++, VB, BASIC, Pascal...	e.g. JavaScript, VB script, PHP, Python, Lua. … etc.

How does HTTP work?



Domain Name System

- The Domain Name System (DNS) is the phonebook of the Internet. Humans access information online through domain names, like *amu.edu.et* or *ethiotelecom.et*.
- Web browsers interact through Internet Protocol (IP) addresses. DNS translates domain names to IP addresses so browsers can load Internet resources.
- Each device connected to the Internet has a unique IP address which other machines use to find the device.
- DNS servers eliminate the need for humans to memorize IP addresses such as 192.168.1.1 (in IPv4), or more complex newer alphanumeric IP addresses such as 2001:db8:3333:4444:CCCC:DDDD:EEEE:FFFF (in IPv6).



Protocols

A communications protocol or network protocol is the specification of a *set of rules* for a particular type of communication.

Common Internet protocols include TCP/IP (Transmission Control Protocol/Internet Protocol), UDP/IP (User Datagram Protocol/Internet Protocol), HTTP (HyperText Transfer Protocol) and FTP (File Transfer Protocol).

- ***TCP/IP***: is a stream protocol. This means that a connection is negotiated between a client and a server. Any data transmitted between these two endpoints is guaranteed to arrive, thus it is a so-called lossless protocol. Since the TCP protocol (as it is also referred to in short form) can only connect two endpoints, it is also called a peer-to-peer protocol.
- ***HTTP***: is the protocol used to transmit all data present on the World Wide Web. This includes text, multimedia and graphics. It is the protocol used to transmit HTML, the language that makes all the fancy decorations in your browser. It works upon TCP/IP.
- ***FTP***: is the protocol used to transmit files between computers connected to each other by a TCP/IP network, such as the Internet.

WEB DESIGN AND DEVELOPMENT FUNDAMENTALS

Getting started with Web application development

Getting started with the web is a concise series introducing you to the practicalities of web development. You'll set up the tools you need to construct a simple webpage and publish your own simple code.

Installing basic software

- A text editor – Sublime, vs-code, Notepad++ …
- Client software (browser) - Chrome, Firefox, Safari…
- Server-side stack software - WAMP, XAMPP, LAMP…

What will your website look like?

- Before you start writing the code for your website, you should *plan it first.*
    - What information are you showcasing?
    - What fonts and colors are you using?
    - Outline a simple method that you can follow to plan out your site's content and design.

Dealing with files

- A website consists of many files: *text content, code, stylesheets, media content*, and *so on*.
- When you're building a website, you *need to assemble these files into a sensible structure* and make sure they can talk to one another.

Client-Side Scripting and Server-Side Scripting

The scripts may be created in **two** ways: on the **client side** or the **server side**, where the server-side scripts are processed on a server. In contrast, client-side scripting needs browsers to execute scripts on the client system but doesn't connect with the server executing the client-side scripts.

What is Client-Side Scripting?

Client-Side Scripting refers to the output which is requested to the server by the end-users. The majority of this page is written in HTML. With client-side scripting, JavaScript is the primary language used. It is the most widely used language in this area, and it works with all programs.

Client-side scripting may be utilized to check the user's form for problems before submitting it and to change the content based on the user input. The web needs three components to function: client, database, and server.

The client-side scripting may significantly reduce server demand. It is intended to be utilized as a scripting language with a web browser as the host program. The HTML and CSS are delivered as plain text when a user uses a browser to request a webpage from the server, and the browser understands and renders the web content at the client end.

Client-side scripting language

HTML

It is not a scripting language; it is a markup language. However, it serves as the basic language for client-side web development, also referred to as front-end. The presence of hypertext on a page denotes its hyperlinks. The markup language uses tags to define the structure and layout. It is a programming language that is mainly used to design a web page's structure and layout.

CSS

CSS is an abbreviation for Cascading Style Sheets. It provides a technique for creating graphic elements that help a web application's appearance look more appealing. A style tag in a web page defines all the specifics regarding the web page's presentation, including its border styles, image styles, colour, font styles, borders, format, font size, margins, padding, etc.

JavaScript

It is a client-side scripting language designed for a specific purpose, but several JavaScript frameworks are already utilized as server-side scripting technologies.

VBScript

VBScript is based on Visual Basic, which was created by Microsoft in 1996. It is a scripting programming language that is lightweight, fast, and easy to learn. It is not a OOPs language but is similar to JavaScript.

Features of Client-side Scripting

There are various features of client-side scripting. Some main features of the client-side scripting are as follows:

1. It is intended to *execute code* on *which a web browser runs*, and the results of the inputs are *delivered to an accessible user*.
2. Client-side scripting *enables greater involvement with clients via the browser* and is used to *validate programs and functionality based on the request*.
3. The *client does not include any contact with the server in client-side scripting*; the only interaction is receiving the requested data.

What is Server-Side Scripting?

Server-side scripting is a programming technique for creating code that may ***run software on the server side***. In other words, server-side scripting is any scripting method that may operate on a web server. At the server end, actions such as website customization, dynamic changes in website content, response creation to user requests, database access, and many more are carried out.

Server-side scripting creates a communication channel between a server and a client. Previously, ***CGI (Common Gateway Interface)*** scripts were used to implement server-side scripting, and CGI was created to execute scripts written in computer languages such as C++ or Perl on websites.

The server-side is made up of three parts: the ***database***, ***the server***, the ***APIs***, and the ***backend web software*** written in the server-side scripting language. When a browser requests a page with server-side scripting, the web server evaluates the script before delivering the page to the browser. In this case, script processing may entail collecting information from a database, performing simple computations, or selecting the relevant material to be shown on the client end. The output is provided to the web browser when the script is processed. The web server hides the scripts from the end user until the content is delivered, making the data and source code safer.

Server-side scripting languages

There are various server-side scripting languages. Some main server-side scripting languages are as follows:

Python

It is an open-source language that is very powerful and easy to learn. It is suitable for beginners because it is simple to learn and read. It is believed to be used by Google and YouTube. It is a OOPs language with dynamic typing and data structures. It has grown to be one of the most popular languages for both quick application development and web development.

PHP

It is an open-source server-side scripting programming language mainly designed for web apps and is the most utilized scripting language. It allows you to retrieve and manipulate data from a database and is utilized along with SQL to query the database. It is a fast and simple language to learn and develop, and Facebook, Wikipedia, and WordPress utilize it.

Ruby

It is a free and open-source programming language that was developed and firstly introduced in the early 1990s. It is a dynamic language that is simple to read and write and an OOPs language that is interpreted as it runs. It has evolved continuously since its development and is one of the most utilized web development languages.

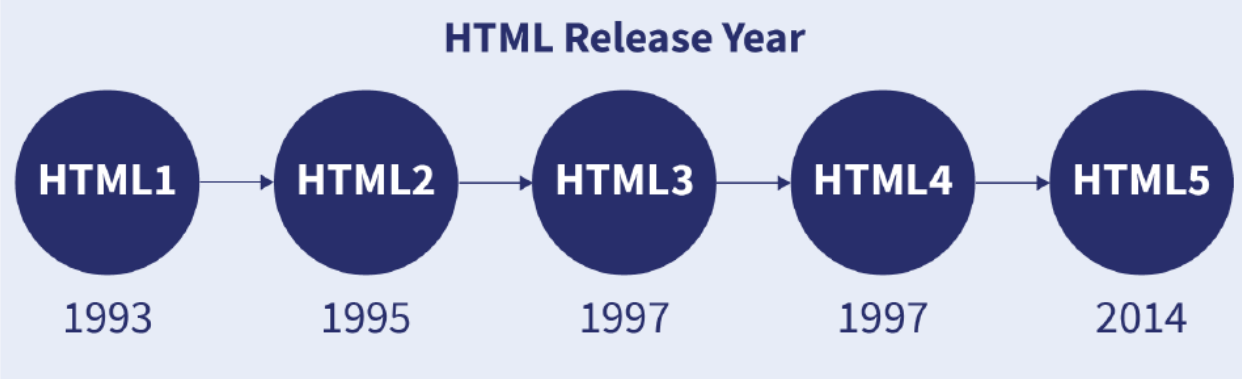
Features of Server-side Scripting

There are various features of server-side scripting. Some main features of the server-side scripting are as follows:

1. It is connected with *data access, error handling, and data processing speed.*
2. It is *processed and interacts* with the server.
3. Using a *highly integrated programming language* makes it *more secure than client-side scripting.*

HYPERTEXT MARKUP LANGUAGE (HTML)

- HTML stands for *Hyper Text Markup Language*
- HTML is the standard markup language for creating Web pages
- HTML describes the ***structure*** of a Web page
- HTML consists of a ***series of elements***
- HTML ***elements tell*** the browser ***how to display the content***
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

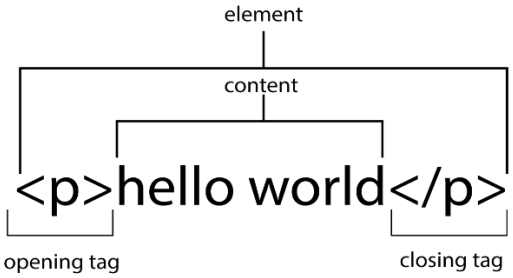


- HTML documents are simply text documents with a specific form
- Documents comprised of content and markup tags
    - ***Content***: actual information being conveyed
    - ***Markup tags***: tell the Web browser how to display the page
- An HTML file must have an .htm or .html file extension
- An HTML file can be created using a simple text editor.

Why do we use HTML?

- HTML is the language used to tell your web browser what each part of a website is. So, using HTML, you can define headers, paragraphs, links, images, and more, so your browser knows how to structure the web page you're looking at.
- HTML is the skeleton of just about any website. It provides the bones that underpin everything else on site. Common things that HTML is used to define are:
    - ***Paragraphs***
      The HTML paragraph element is one of the most common elements and as you might have guessed it defines a paragraph.
    - ***Line Breaks***
      As with print media, a paragraph creates a line break below it to visually separate it from other paragraphs. This is used to emphasize a semantic separation of content. The same structure is used in a novel or a magazine.
    - ***Block Elements***
      Elements that create the spacing below themselves on a page are called block elements. Block elements appear vertically down the left-hand side of a page at least until they are styled by CSS. Examples of block elements are , <article>, <table>, and many more. This feature allows HTML to start separating a webpage into different sections
    - ***Headings***
      Paragraphs and headings work in concert to create the majority of the text content of a web page and its structure. HTML has six heading elements, which are numbered **1** through **6**. **<h1>** is the most significant and usually contains the title of the content – Not to be confused with the title that appears in the browser tab. <h2> represents a subsection. <h3> and so on represent identifiers of further subjects in subsections until we get to <h6>.
- To give a clearer understanding of how HTML works to create the final version of a webpage, if HTML is our skeleton, CSS (Cascading Style Sheets) would be what gives us our features, like the colour of our eyes, skin and hair. JavaScript would be to do with our movements and how we interact with people.

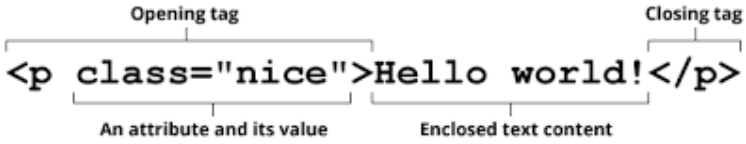
Anatomy of an HTML element



The main parts of the elements are as follows:

- ***The opening tag:*** This consists of the name of the element (in this case, p), wrapped in opening and closing angle brackets. This states where the element begins or starts to take effect — in this case where the paragraph begins.
- ***The closing tag:*** This is the same as the opening tag, except that it includes a forward slash before the element name. This states where the element ends — in this case where the paragraph ends. Failing to add a closing tag is one of the standard beginner errors and can lead to strange results.
- ***The content:*** This is the content of the element, which in this case, is just text.
- ***The element:*** The opening tag, the closing tag, and the content together comprise the element.

Elements can also have attributes that look like the following:



Attributes contain extra information about the element that you don't want to appear in the actual content. Here, ***class*** is the *attribute name* and "**nice**" is the ***attribute value***. The class attribute allows you to give the element a non-unique identifier that can be used to target it (and any other elements with the same class value) with style information and other things. Some attributes have no value, such as "**required**".

Attributes that set a value always have:

- A space between it and the element name (or the previous attribute, if the element already has one or more attributes).
- The attribute name followed by an equal sign.
- The attribute value wrapped by opening and closing quotation marks.

Anatomy of an HTML document

1. It's a document type declaration (also called DOCTYPE declaration) that identifies this document as an HTML5 document. It lets modern browsers know they should interpret the document as written according to the HTML5 specification.

2. The entire document is contained within an html element. The html element is called the root element because it contains all the elements in the document, and it may not be contained within any other element. It is used for both HTML and XHTML documents.

3. Within the html element, the document is divided into a head and a body. The head element contains descriptive information about the document itself, such as its title, the style sheet(s) it uses, scripts, and other types of "meta" information.

4. The meta elements within the head element provide information about the document itself. A meta element can be used to provide all sorts of information, but in this case, it specifies the character encoding (the standardized collection of letters, numbers, and symbols) used in the document.

5. Also in the head is the mandatory title element. According to the HTML specification, every document must contain a descriptive title.

6. Finally, the body element contains everything that we want to show up in the browser window.

Example:

```
<!DOCTYPE html>
<html>
<head>
<title>My code academy</title>
</head>
      <body>
            <h1> My-code-acad</h1>
            <p>Programming and Web Development</p>
      </body>
</html>
```

Inline HTML elements

An "element" is a single part of a webpage. Some elements are large and hold smaller elements like containers. Some elements are small and are "nested" inside larger ones. By default, "inline elements" appear next to one another in a webpage. They take up only as much width as they need in a page and fit together horizontally like words in a sentence or books shelved side-by-side in a row. All inline elements can be placed within the <body> element.

Usage	Element	Example
A link	<a>	 A link to example.org .
An image	<img>	
An inline container		Used to group elements: for example, to style them.
Emphasize text		I'm posh.
Italic text		Mark a phrase in italics.
Bold text		Bold a word or phrase.
Important text		I'm important!
Highlight text	<mark>	<mark>Notice me! </mark>
Strikethrough text	<s>	<s>I'm irrelevant. </s>
Subscript	<sub>	H₂O
Small text	<small>	Used to represent the small print of a document.
Address	<address>	<address>Main Street 67</address>
Textual citation	<cite>	For more monsters, see <cite>The Monster Book of Monsters</cite>.
Superscript	<sup>	x²
Inline quotation	<q>	<q>Me? </q>, she said.
A line break	 	Line 1 Line 2
A possible line break	<wbr>	 Llanfair<wbr>pwllgwyngyllgogerychwyrngogogoch.
Date	<time>	Used to format the date. For example: <time datetime="2020-05-24" pubdate> published on 23-05-2020</time>.
Code format	<code>	This text is in normal format, but <code>this text is in code format</code>.
Audio	<audio>	<audio controls> <source src="https://interactive-examples.mdn.mozilla.net/media/cc0-audio/t-rex-roar.mp3" type="audio/mpeg"> </audio>
Video	<video>	<video controls width="250" src="https://archive.org/download/ElephantsDream/ed_hd.ogv" >

```
 <a
href="https://archive.org/download/ElephantsDream/ed_hd.ogv">Dow
nload OGV video</a>
</video>
```

Block HTML elements

"Block elements," on the other hand, take up the entire width of a webpage. They also take up a full line of a webpage; they do not fit together side-by-side. Instead, they stack like paragraphs in an essay or toy blocks in a tower.

Usage	Element	Example
A simple paragraph	<p>	`<p>I'm a paragraph</p>` `<p>I'm another paragraph</p>`

An extended quotation	<blockquote>	`They said:`
		`<blockquote>The blockquote element indicates`
		`an extended quotation. </blockquote>`

Additional information	<details>	`<details>`
		`  <summary>Html Cheat Sheet</summary>`
		`  <p>Inline elements</p>`
		`  <p>Block elements</p>`
		`</details>`

An unordered list		`<ul>`
		`  <li>I'm an item</li>`
		`  <li>I'm another item</li>`
		`</ul>`

An ordered list		`<ol>`
		`  <li>I'm the first item</li>`
		`  <li>I'm the second item</li>`
		`</ol>`

A definition list	<dl>	`<dl>`
		`  <dt>A Term</dt>`
		`  <dd>Definition of a term</dd>`
		`  <dt>Another Term</dt>`
		`  <dd>Definition of another term</dd>`
		`</dl>`

A horizontal rule	<hr>	`before<hr>after`

Text Heading	<h1>-<h6>	`<h1> This is Heading 1 </h1>`
		`<h2> This is Heading 2 </h2>`
		`<h3> This is Heading 3 </h3>`
		`<h4> This is Heading 4 </h4>`
		`<h5> This is Heading 5 </h5>`
		`<h6> This is Heading 6 </h6>`

CASCADED STYLE SHEETS (CSS)

What is CSS?

CSS stands for ***Cascading Style Sheets***, which is a style sheet language used to describe the look and formatting of a document written in HTML, XHTML, or XML. CSS enables web developers to separate the ***presentation*** of a web page from its content, making it easier to maintain and update the ***design and layout*** of a website.

CSS allows web designers to control various aspects of a ***web page's appearance***, including its *color scheme, typography, layout, and visual effects* such as animations and transitions. It provides a wide range of styling options, including basic properties such as *font-size, color, and background, as well as more advanced techniques such as positioning, responsiveness, and media queries*.

Why use style sheets?

Style sheets provide a number of benefits for web developers:

- Separation of concerns: Separating the presentation and layout of a web page from its content makes it easier to maintain and update the site's design. With CSS, developers can make changes to the style of a site without having to modify the content of each individual page.
- Consistency: Using style sheets across a site ensures that the site's visual appearance is consistent across all pages, which can help establish a sense of brand identity.
- Efficiency: CSS can improve web page loading times by reducing the amount of code needed in HTML. With CSS, developers can define the style of a page using fewer lines of code than would be required using HTML alone.
- Accessibility: Separating the style of a web page from its content can make it easier for assistive technologies (such as screen readers) to navigate and read the page.

HTML vs. CSS

HTML provides the structure and content of a web page, while CSS provides the style and layout. Here are some examples of what each language is responsible for:

- HTML: HTML is used to create the structure of a web page, including headings, paragraphs, lists, images, and links. Here's an example of what HTML might look like for a simple web page:

```
<!DOCTYPE html>
<html>
	<head>
	<title>My Web Page</title>
	</head>
	<body>
		<h1>Welcome to my web page!</h1>
		<p>This is a paragraph of text.</p>
		<img src="image.jpg" alt="An image">
		<a href="http://www.example.com">Click here to visit Example.com</a>
	</body>
</html>
```

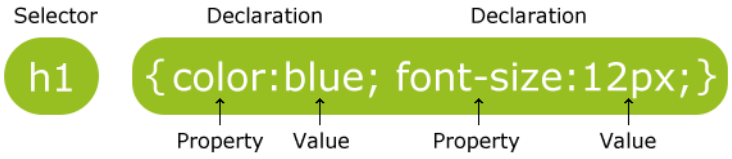
- CSS: CSS is used to define the visual appearance of a web page, including the colors, fonts, layout, and other design elements. Here's an example of what CSS might look like for the HTML above:

```
body {
  font-family: Arial, sans-serif;
  background-color: #f0f0f0;
}

h1 {
  color: blue;
}

p {
  font-size: 18px;
  line-height: 1.5;
}

img {
  width: 100%;
}

a {
  color: red;
  text-decoration: none;
}
```

In this example, we've defined the ***font-family*** and ***background-color*** for the '**body**' element, the color for the '**h1**' element, the ***font-size*** and ***line-height*** for the '**p**' element, the ***width*** for the ***img*** element, and the ***color*** and ***text-decoration*** for the '**a**' element.

Anatomy of a CSS Rule

A CSS rule consists of a selector and a declaration block.



- The selector points to the HTML element you want to style.
- The declaration block contains one or more declarations separated by semicolons.
- Each declaration includes a *CSS property name* and a *value*, separated by a colon.
- A CSS declaration always ***ends with a semicolon***, and declaration blocks are surrounded by ***curly braces.***

Here's an example of what a CSS rule might look like:

```
h1 {
  font-size: 36px;
  color: red;
}
```

Linking HTML and CSS

There are three main ways to insert CSS into an HTML document:

- **External Style Sheet:** In this method, the CSS code is stored in a separate file with a .css extension, and then linked to the HTML document using the <link> element in the <head> section of the HTML document. Here's an example:

```
<!DOCTYPE html>
<html>
        <head>
          <title>My Web Page</title>
          <link rel="stylesheet" type="text/css" href="styles.css">
         </head>
         <body>
                <h1>Welcome to my web page!</h1>
                <p>This is a paragraph of text.</p>
         </body>
</html>
```

  In this example, the href attribute specifies the location of the external style sheet file, which contains the CSS rules for the web page.

- **Internal Style Sheet:** In this method, the CSS code is included within the <style> element in the <head> section of the HTML document. Here's an example:

```
<!DOCTYPE html>
<html>
 <head>
        <title>My Web Page</title>
        <style>
                body {
                        background-color: #f0f0f0;
                }
                h1 {
                        color: blue;
                }
        </style>
 </head>
 <body>
        <h1>Welcome to my web page! </h1>
        <p>This is a paragraph of text.</p>
 </body>
</html>
```

In this example, the CSS rules are contained within the <style> element, which is located in the <head> section of the HTML document.

- **Inline Style:** In this method, the CSS code is included directly within the HTML elements using the style attribute. Here's an example:

```
<!DOCTYPE html>
<html>
 <head>
  <title>My Web Page</title>
 </head>
 <body>
  <h1 style="color: blue;">Welcome to my web page!</h1>
  <p style="font-size: 18px;">This is a paragraph of text.</p>
 </body>
</html>
```

In this example, the CSS rules are included within the style attribute of each HTML element. This method is generally not recommended, as it can be difficult to maintain and update the CSS code when it is scattered throughout the HTML document.

CSS Selectors

CSS selectors are used to target specific HTML elements on a web page and apply styling rules to them. There are several types of selectors in CSS, each with its own syntax and level of specificity. Understanding CSS selectors is a crucial part of mastering CSS and creating effective styles for your web pages.

Here are some of the most commonly used CSS selectors:

1. **Element Selectors:** Element selectors target specific HTML elements on a page, such as <p> or <h1>. They are the simplest and most common type of selector in CSS. Here's an example:

```
p {
  color: red;
}
```

In this example, the **'p' selector** targets all **<p>** elements on the page and applies the ***color*** property with a value of "***red***".

2. **Class Selectors:** Class selectors target HTML elements that have a specific class attribute. They are indicated by a period (.) followed by the class name. Here's an example:

```
.my-class {
  font-size: 18px;
}
```

In this example, the **.my-class selector** targets all HTML elements that have the class attribute set to "**my-class**" and applies the **font-size** property with a value of "***18px***".

3. **ID Selectors:** ID selectors target a specific HTML element that has a specific ID attribute. They are indicated by a hash (#) followed by the ID name. Here's an example:

```
#my-element {
  background-color: blue;
}
```

In this example, the **#my-element selector** targets the HTML element with the ID attribute set to "**my-element**" and applies the ***background-color*** property with a value of "***blue***".

A combinator selector

A combinator selector in CSS is used to select elements based on their relationship to other elements on the page. There are several types of combinator selectors in CSS, each with its own syntax and purpose.

Here are some of the most commonly used combinator selectors:

Descendant Selector (Space): This selector targets elements that are descendants of a specific parent element. It is indicated by a space between two or more selectors. For example:

```
ul li {
  color: red;
}
```

In this example, the **li** element will be selected only if it is a descendant of a **ul** element. The **ul** element may have other parent elements as well.

Child Selector (>): This selector targets elements that are direct children of a specific parent element. It is indicated by a greater than symbol (>) between two selectors. For example:

```
ul > li {
  color: red;
}
```

In this example, the **li** element will be selected only if it is a direct child of a **ul** element. The **ul** element cannot have any other elements between it and the **li** element.

Adjacent Sibling Selector (+): This selector targets an element that comes immediately after another specific element. It is indicated by a plus (+) sign between two selectors. For example:

```
h1 + p {
  color: red;
}
```

In this example, the **p** element will be selected only if it immediately follows an **h1** element.

General Sibling Selector (~): This selector targets elements that are siblings of a specific element. It is indicated by a tilde (~) between two selectors. For example:

```
h1 ~ p {
  color: red;
}
```

In this example, all **p** elements that follow an **h1** element will be selected, regardless of whether they immediately follow it or not.

Combinator selectors are a powerful tool in CSS that allow you to target specific elements on the page based on their relationship to other elements. By mastering the use of combinator selectors, you can create complex and precise styles that enhance the overall user experience.

Selector	Example	Example description
element element	div p	Selects all <p> elements inside elements
element>element	div > p	Selects all <p> elements where the parent is a element
element+element	div + p	Selects the first <p> element that are placed immediately after elements
element1~element2	p ~ ul	Selects every element that are preceded by a <p> element

Cascading Order

- What style will be used when there is more than one style specified for an HTML element?
- Order:
    - Inline style (inside an HTML element)
    - External and internal style sheets (in the head section)
    - Browser default

- So, an inline style (inside a specific HTML element) has the highest priority, which means that it will override a style defined inside the <head> tag, or in an external style sheet, or a *browser default value*.

Here is a CSS cheatsheet that includes some of the most commonly used CSS properties:

Selector

Selector	Description
*****	Selects all elements
element	Selects all elements of the specified type
.class	Selects all elements with the specified class
#id	Selects the element with the specified ID
element, element	Selects all elements that match either selector
element element	Selects all elements that are descendants of the specified element
element > element	Selects all elements that are direct children of the specified element

Box Model

Property	Description
width	Sets the width of an element
height	Sets the height of an element
margin	Sets the margin around an element
padding	Sets the padding inside an element
border	Sets the border around an element

Typography

Property	Description
font-family	Sets the font family for text
font-size	Sets the font size for text
font-weight	Sets the font weight for text
font-style	Sets the font style for text
line-height	Sets the line height for text

Colors

Property	Description
color	Sets the color of text
background-color	Sets the background color of an element
opacity	Sets the opacity of an element

Display

Property	Description
display	Sets the display style for an element
visibility	Sets the visibility of an element

Positioning

Property	Description
position	Sets the positioning method for an element
top, right, bottom, left	Sets the offset for a positioned element
float	Sets the floating behavior for an element
clear	Sets the clear behavior for an element

Flexbox

Property	Description
display: flex	Creates a flex container
flex-direction	Sets the direction of the flex container
justify-content	Aligns items along the main axis
align-items	Aligns items along the cross axis
flex-wrap	Sets whether items should wrap or not

Grid

Property	Description
display: grid	Creates a grid container
grid-template-columns	Defines the columns of the grid
grid-template-rows	Defines the rows of the grid
grid-column-gap	Sets the gap between columns
grid-row-gap	Sets the gap between rows
grid-template-areas	Defines the grid areas
grid-area	Assigns a grid area to an item

CLIENT-SIDE PROGRAMMING –JAVASCRIPT

What is Client-side programming?

Client-side programming refers to the practice of writing code that executes within a user's web browser. When a user visits a website, the web server sends the HTML, CSS, and JavaScript files to the user's browser. The browser then interprets the code and renders the web page. Client-side programming is used to add interactivity, validate form inputs, perform animations, and make AJAX calls to fetch data from a server without reloading the page.

What is JavaScript?

- JavaScript is a high-level, interpreted programming language that is commonly used for creating interactive web applications. It was first introduced in 1995 by Netscape, and it has since become one of the most popular programming languages on the web. JavaScript is often used in conjunction with HTML and CSS to create dynamic and responsive web pages.
- Javascript is a lightweight programming language ("scripting language")
    - used to make web pages interactive
    - insert text into HTML
        - document.getElementById("demo").innerHTML = "Hello JavaScript";
    - Can Change HTML Attribute Values
        - <button onclick="document.getElementById('myImage').src='dog.gif'"> Change</button>
    - Can change HTML Styles (CSS)
        - document.getElementById("demo").style.fontSize = "35px";
    - react to events (ex: page load user click)
    - get information about a user's computer (ex: browser type)
    - perform calculations on user's computer (ex: form validation)
    - a web standard (but not supported identically by all browsers)
    - NOT related to Java other than by name and some syntactic similarities

Common scripting tasks

JavaScript can be used to perform a wide variety of scripting tasks on the client-side, including:

- ***Validating form inputs***: JavaScript can be used to ensure that users enter valid information into forms, such as email addresses, phone numbers, and credit card numbers.
- ***Creating animations***: JavaScript can be used to animate web page elements, such as buttons, menus, and images.

- ***Performing calculations:*** JavaScript can be used to perform calculations on the client-side, such as adding up prices in a shopping cart or calculating the total distance traveled on a map.
- ***Making AJAX calls:*** JavaScript can be used to make asynchronous HTTP requests to fetch data from a server without reloading the page.

Limitations of client-side scripting

There are several limitations to client-side scripting that developers should be aware of:

- ***Security***: Because client-side scripts execute within the user's browser, they can be vulnerable to attacks such as cross-site scripting (XSS) and code injection.
- ***Performance***: JavaScript can slow down web page load times if it is not optimized properly. Additionally, not all users may have the same level of JavaScript support in their browsers.
- ***Accessibility***: Users who have disabilities or who use assistive technologies may have difficulty using websites that rely heavily on JavaScript.

JavaScript Terminology

Here are some common JavaScript terms that developers should be familiar with:

- ***Variable***: A container for storing a value, such as a number, string, or boolean.
- ***Function***: A reusable block of code that performs a specific task.
- ***Event***: An action that occurs on a web page, such as a button click or a page load.
- ***DOM***: The Document Object Model is a tree-like structure that represents the HTML elements on a web page.
- ***Callback function***: A function that is passed as an argument to another function and is executed when a certain event occurs.
- ***Object***: A collection of related data and functions that represent a real-world entity.
- ***Array***: An ordered collection of values, such as numbers, strings, or objects.
- ***JSON***: JavaScript Object Notation is a lightweight data interchange format that is often used to send data between a server and a web page.
- ***AJAX***: Asynchronous JavaScript and XML is a technique for making HTTP requests from a web page without reloading the entire page.

Javascript Language format

JavaScript code is typically written in a text editor or integrated development environment (IDE). Here is an example of the basic structure of a JavaScript program:

```
// This is a single-line comment in JavaScript

/*
This is a
multi-line
comment in JavaScript
*/

// Declare a variable called "message" and assign it the value "Hello, world!"
var message = "Hello, world!";

// Display the value of the "message" variable in the console
console.log(message);

// Declare a function called "greet" that takes a parameter called "name"
function greet(name) {
  // Display a personalized greeting in the console
  console.log("Hello, " + name + "!");
}

// Call the "greet" function with the argument "John"
greet("John");
```

JavaScript code can be embedded directly into an HTML file using the ***<script>*** tag:

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page</title>
  </head>
  <body>
    <h1>My Web Page</h1>
    <script>
      // This is a JavaScript program embedded in an HTML file
      var message = "Hello, world!";
      alert(message);
    </script>
  </body>
</html>
```

JavaScript code can also be included in a separate file and linked to an HTML file using the <script> tag:

```
<!DOCTYPE html>
<html>
 <head>
  <title>My Web Page</title>
  <script src="my-script.js"></script>
 </head>
 <body>
  <h1>My Web Page</h1>
 </body>
</html>
```

In this example, the JavaScript code is stored in a file called **"my-script.js"**, which is linked to the HTML file using the src attribute of the **<script>** tag.

- Placing scripts in external files has some advantages:
  - It separates HTML and code
  - It makes HTML and JavaScript easier to read and maintain
  - Cached JavaScript files can speed up page loads
  - To add several script files to one page use several script tags

  Example
  - <script src="myScript1.js"></script>
  - <script src="myScript2.js"></script>

Displaying message in Javascript

In JavaScript, there are several ways to display messages to the user or to the developer for debugging purposes. Here are some common methods:

- **alert():** This method displays a message in a dialog box that requires the user to click "OK" to dismiss it. Here's an example:

```
alert("Hello, world!");
```

- **console.log():** This method outputs a message to the console, which can be opened in the browser's developer tools. This is useful for debugging purposes or for displaying messages that the user doesn't need to see. Here's an example:

```
console.log("Hello, world!");
```

- **document.write():** This method writes a message directly to the web page. However, this method is not commonly used anymore because it can overwrite the existing content of the page. Here's an example:

```
document.write("Hello, world!");
```

- **innerHTML**: This property can be used to change the content of an HTML element on the page, such as a or a <p> tag. Here's an example:

```
document.getElementById("my-element").innerHTML = "Hello, world!";
```

  In this example, the message "Hello, world!" is displayed inside an HTML element with the ID "my-element". Note that this method requires an existing HTML element on the page to work.
- **prompt():** This method displays a dialog box that prompts the user to enter a value. The value entered by the user is returned as a string. Here's an example:

```
var name = prompt("What is your name?");
console.log("Hello, " + name + "!");
```

  In this example, the user is prompted to enter their name. The name is then stored in a variable called "name", which is used to display a personalized message in the console.
- **confirm():** Generates a message on pop up window with two options , **yes or no**

```
var conf = confirm("depositing 100 ETB, are you sure?");
```

Javascript variable

In JavaScript, variables are used to store values that can be used later in the program. Here are the basic syntax and rules for declaring and using variables in JavaScript:

1. **Declaration**: To declare a variable in JavaScript, use the var, let, or const keyword, followed by the variable name and an optional initial value:

```
var myVariable = 10;
let anotherVariable = "Hello, world!";
const PI = 3.14;
```

   - ***'var'*** is used to declare a variable that can be reassigned later in the program.
   - ***'let'*** is used to declare a variable that can be reassigned, but is limited to the block scope.
   - ***'const'*** is used to declare a variable that cannot be reassigned and is also limited to the block scope.
2. **Naming rules:** Variable names can contain letters, numbers, underscores, or dollar signs. They cannot start with a number, and they cannot be a reserved keyword, such as ***if, else, while***, etc.

3. **Assignment**: To assign a value to a variable, use the assignment operator ('='):

```
myVariable = 20;
```

4. **Access**: To access the value of a variable, simply refer to its name:

```
console.log(myVariable); // Output: 20
```

5. **Scope**: JavaScript has two types of scope: global scope and local scope. Variables declared outside a function have global scope and can be accessed from anywhere in the program. Variables declared inside a function have local scope and can only be accessed from within the function.

```
var globalVariable = "I am global";

function myFunction() {
  var localVariable = "I am local";
  console.log(globalVariable); // Output: I am global
  console.log(localVariable); // Output: I am local
}

myFunction();

console.log(globalVariable); // Output: I am global

console.log(localVariable); // Output: Uncaught ReferenceError: localVariable is not defined
```

In this example, the variable '**globalVariable**' has global scope and can be accessed from within the function 'myFunction' and outside of it. The variable '**localVariable**' has local scope and can only be accessed from within 'myFunction'.

Document Object Model (DOM)

DOM stands for Document Object Model, which is a programming interface for web documents. It represents the web page as a structured tree of HTML elements and allows JavaScript to interact with the web page by manipulating its elements and attributes.

Here are the basic concepts of using the DOM in JavaScript:

1. Accessing Elements: JavaScript can access the HTML elements on a page using methods like **document.getElementById(), document.querySelector(),** and **document.querySelectorAll().** These methods allow us to retrieve elements based on their ID, class, or tag name.

```
// Get an element by its ID
var myElement = document.getElementById("my-element");
// Get the first element with a certain class
var myClass = document.querySelector(".my-class");
// Get all elements with a certain tag name
var myTags = document.querySelectorAll("p");
```

2. Manipulating Elements: Once we have access to an HTML element, we can modify its content, attributes, and styles using JavaScript.

```
// Change the text content of an element
myElement.textContent = "Hello, world!";
// Change the background color of an element
myElement.style.backgroundColor = "red";
// Add a new attribute to an element
myElement.setAttribute("data-name", "my-element");
// Remove an attribute from an element
myElement.removeAttribute("data-name");
```

3. Events: JavaScript can also listen for events on the web page, such as clicks, mouse movements, and keyboard inputs. We can use event listeners to execute code when an event occurs.

```
// Add a click event listener to an element
myElement.addEventListener("click", function() {
  alert("You clicked me!");
});
// Add a mouseover event listener to an element
myElement.addEventListener("mouseover", function() {
  myElement.style.backgroundColor = "blue";
});
// Add a keydown event listener to the document
document.addEventListener("keydown", function(event) {
  console.log("You pressed the " + event.key + " key!");
});
```

In this example, we add event listeners to an element and to the document, and execute code when events occur.

4. Creating and Removing Elements: JavaScript can also create new HTML elements and remove existing ones from the page.

```
// Create a new element
var newElement = document.createElement("div");
newElement.textContent = "I am a new element!";

// Add the new element to the page
document.body.appendChild(newElement);

// Remove an existing element from the page
myElement.remove();
```

Javascript cheat sheet

Topic	Syntax

Variables	var variableName = value;
	let variableName = value;
	const CONSTANT_NAME = value;

Data Types	var myString = "Hello, world!"
	var myNumber = 10;
	var myBoolean = true;
	var myUndefined = undefined;
	var myNull = null;
	var myArray = [1, 2, 3];
	var myObject = {name: "John", age: 30};

Operators	**Arithmetic**:
	var sum = a + b;
	var difference = a - b;
	var product = a * b;
	var quotient = a / b;
	var remainder = a % b;
	Comparison:
	var isEqual = a == b;
	var isNotEqual = a != b;
	var isStrictEqual = a === b;
	var isGreaterThan = a > b;
	var isLessThan = a < b;
	var isGreaterThanOrEqual = a >= b;
	var isLessThanOrEqual = a <= b;
	Logical:
	var isTrue = true && false;
	var isFalse = true

Conditional Statements	***If -condition***
	if (condition) {
	// code to execute if condition is true
	}else if (anotherCondition) {
	// code to execute if anotherCondition is true
	}else {
	// code to execute if both conditions are false
	}
	Switch case
	switch (variableName) {
	case value1:
	//code to execute if variableName equals value1
	break;
	case value2:
	// code to execute if variableName equals value2
	break;
	default:
	// code to execute if variableName does not equal any case
	}

Loops	***For***:
	for (var i = 0; i < arrayName.length; i++) {
	// code to execute for each element in arrayName
	}
	
	While:
	var i = 0
	while (i < arrayName.length) {
	//code to execute for each element in arrayName
	i++;
	}

Functions	**function** functionname(parameter1,parameter2…)
	{
	//function body
	}

SERVER- SIDE PROGRAMMING-PHP

Introduction to server-side programming

What is server-side programming?

Server-side programming refers to the execution of code on a web server to generate dynamic web pages. It involves processing user requests, interacting with databases, and generating output that is sent back to the client's web browser. Server-side programming is used to create interactive web applications that require dynamic content.

Advantages of server-side programming

- ***Increased security:*** Server-side programming can help prevent malicious attacks by restricting access to certain parts of the code.
- ***Better performance:*** Server-side programming can provide better performance as the server can handle multiple requests simultaneously.
- ***Scalability:*** Server-side programming allows for scaling the application as needed to accommodate an increasing number of users.

Introduction to PHP

PHP (Hypertext Preprocessor) is a popular server-side scripting language used to create dynamic web pages. It was originally created in 1994 by **Rasmus Lerdorf** and has since become one of the most widely used programming languages on the web.

PHP is an open-source language and is constantly evolving, with new features and improvements being added regularly. Some of the key features of PHP include:

- Compatibility with various web servers and operating systems
- Support for a wide range of databases
- Easy integration with HTML, CSS, and JavaScript
- Large community of developers
- Components Required to Set up a PHP Development Environment:

To develop PHP applications, you will need the following components:

- ***Web Server:*** A web server is software that allows you to serve web pages to clients over the internet. Some of the popular web servers include Apache, Nginx, and IIS.
- ***PHP Interpreter:*** The PHP interpreter is the software that reads and executes PHP code. You can download the latest version of the PHP interpreter from the official PHP website.
- ***Database Management System:*** PHP is commonly used to interact with databases, so you will need a database management system such as MySQL or PostgreSQL.
- ***Text Editor or Integrated Development Environment (IDE):*** A text editor or IDE is used to write and edit PHP code. Some popular options include *Visual Studio Code, Sublime Text, and PhpStorm*.

PHP: PHP Basic syntax

PHP Syntax

PHP code is executed on the server-side and can be embedded within an HTML document. PHP code is enclosed in PHP tags, which are typically "<?php" and "?>". The code within the tags is executed by the server and the output is sent to the web browser.

Writing PHP Code in an HTML Document

To write PHP code in an HTML document, you must enclose the PHP code within the PHP tags. For example, to display the current date in a web page, you can use the following PHP code:

```
<html>
<head>
	<title>PHP Date Example</title>
</head>
<body>
	<h1>Today's Date</h1>
	<?php
		echo date("Y/m/d");
	?>
</body>
</html>
```

In the above example, the PHP code to display the current date is embedded within the HTML tags. The "echo" statement is used to output the result to the web page.

Declaring Variables in PHP

Variables are used to store data in PHP. In PHP, variables are declared using the "$" symbol followed by the variable name. For example, to declare a variable called "name", you would use the following syntax:

```
$name = "John";
```

In the above example, the variable "name" is assigned the value "John". You can also declare variables with a specific data type, such as integer or float. For example:

```
$age = 30;

$price = 9.99;
```

PHP is a loosely typed language, which means that you do not need to declare the data type of a variable before using it. The data type is automatically determined based on the value assigned to the variable.

Send Data to the Web Browser

Outputting Text and HTML with PHP

PHP can be used to output text and HTML to the web browser. The "echo" statement is used to output text or HTML to the web page. For example, to output the text "Hello, World!" to the web page, you can use the following PHP code:

```
<?php
	echo "Hello, World!";
?>
```

In the above example, the "echo" statement outputs the text "Hello, World!" to the web page.

You can also use PHP to output HTML to the web page. For example, to output a heading and a paragraph of text, you can use the following PHP code:

```
<?php
    echo "<h1>Welcome to My Website</h1>";
    echo "<p>Thank you for visiting! </p>";
?>
```

In the above example, the "echo" statements output HTML code to the web page, which is then rendered by the web browser.

Generating Dynamic Content with PHP

PHP can be used to generate dynamic content based on user input or other variables. For example, you can use PHP to generate a personalized greeting based on the user's name. To do this, you would first need to capture the user's name using a form or other input method. Once you have the user's name, you can use PHP to generate the personalized greeting. For example:

```
<?php
    $name = $_POST["name"];
    echo "Hello, $name!";
?>
```

In the above example, the $_POST superglobal variable is used to capture the user's name from a form. The variable $name is then used to generate the personalized greeting using the "echo" statement.

Writing Comments in PHP

Comments are used to explain what a particular section of code does or to provide additional information about the code. In PHP, comments can be written using two different methods. Single-line comments can be written using two forward slashes ("//"), while multi-line comments can be written using /* and */.

```
// This is a single-line comment
/*
This is a
multi-line comment
*/
```

Comments should be used to make code more readable and easier to understand, both for yourself and for other developers who may be working with your code in the future.

Utilizing Variables in PHP

Variables are used to store data in PHP. You can use variables to store strings, numbers, and other data types. To declare a variable, you use the "$" symbol followed by the variable name. For example:

```
$name = "John";

$age = 30;
```

You can then use the variables in your code, either by referencing the variable name directly or by concatenating the variable with other strings or variables. For example:

```
echo "My name is " . $name . " and I am " . $age . " years old.";
```

In the above example, the variables $name and $age are concatenated with other strings to create a sentence that is output to the web page.

Concatenating Variables and Strings in PHP

To concatenate variables and strings in PHP, you use the "." operator. For example:

```
$name = "John";

echo "Hello, " . $name . "!";
```

In the above example, the variable $name is concatenated with the string "Hello, " to create a personalized greeting that is output to the web page.

Manipulating Numbers and Working with Constants in PHP

Manipulating Numbers in PHP

PHP provides a wide range of functions for manipulating numbers. You can perform mathematical operations such as addition, subtraction, multiplication, and division using basic arithmetic operators. For example:

```
$x = 10;
$y = 5;
$sum = $x + $y; // adds x and y together
$difference = $x - $y; // subtracts y from x
$product = $x * $y; // multiplies x and y together
$quotient = $x / $y; // divides x by y
```

You can also use functions such as round(), ceil(), and floor() to round numbers to the nearest whole number, to the next highest whole number, or to the next lowest whole number, respectively.

Working with Constants in PHP:

Constants are values that cannot be changed once they are defined. In PHP, you can define constants using the define() function. For example:

```
define("PI", 3.14159);
echo PI; // outputs 3.14159
```

In the above example, the constant "PI" is defined as 3.14159 using the define() function. Once a constant is defined, it can be used throughout your code without the risk of accidentally changing its value.

You can also define constants with arrays:

```
define("COLORS", ['red', 'green', 'blue']);
echo COLORS[0]; // outputs "red"
```

In the above example, the constant "COLORS" is defined as an array of three colors. The first color in the array, "red," is output to the web page using the index notation.

Flow Control in PHP

Flow Control in PHP:

Flow control refers to the ability to control the order in which statements are executed in your code. In PHP, you can use conditional statements and loops to control the flow of your code.

Conditional Statements in PHP

Conditional statements allow you to execute code only if a certain condition is met. The most common conditional statement in PHP is the if statement. The basic syntax of an if statement is as follows:

```
if (condition) {
  // execute this code if the condition is true
}
```

For example:

```
$x = 10;
$y = 5;
if ($x > $y) {
  echo "x is greater than y";
}
```

In the above example, the if statement checks whether the variable $x is greater than the variable $y. If the condition is true, the message "x is greater than y" is output to the web page.

Loops in PHP

Loops allow you to execute a block of code repeatedly. The most common loops in PHP are the for loop, the while loop, and the do-while loop.

The for loop executes a block of code a specified number of times. The basic syntax of a for loop is as follows:

```
for (initialization; condition; increment/decrement) {
  // execute this code
}
```

For example:

```
for ($i = 0; $i < 5; $i++) {
  echo "The value of i is " . $i . "<br>";
}
```

In the above example, the for loop executes the code block five times, incrementing the variable $i each time.

The while loop executes a block of code as long as a certain condition is true. The basic syntax of a while loop is as follows:

```
while (condition) {
  // execute this code
}
```

For example:

```
$i = 0;
while ($i < 5) {
  echo "The value of i is " . $i . "<br>";
  $i++;
}
```

In the above example, the while loop executes the code block as long as the variable $i is less than 5, incrementing $i each time.

The do-while loop is similar to the while loop, but it always executes the code block at least once. The basic syntax of a do-while loop is as follows:

```
do {
  // execute this code
} while (condition);
```

For example:

```
$i = 0;
do {
  echo "The value of i is " . $i . "<br>";
  $i++;
} while ($i < 5);
```

In the above example, the do-while loop executes the code block at least once, and then continues to execute the block as long as the variable $i is less than 5.

Manipulating Arrays in PHP

Arrays in PHP

An array is a collection of variables that are stored together under a single name. In PHP, you can create arrays that store different types of data, including numbers, strings, and even other arrays.

Creating Arrays in PHP

You can create an array in PHP by using the array() function. The basic syntax of the array() function is as follows:

```
$array_name = array(value1, value2, value3, ...);
```

For example:

```
$fruits = array("apple", "banana", "orange");
```

In the above example, an array called $fruits is created, which contains the values "apple", "banana", and "orange".

Accessing Array Elements in PHP

You can access the elements of an array in PHP using their index values. The index of an array starts at 0, which means that the first element of an array has an index of 0, the second element has an index of 1, and so on.

For example:

```
$fruits = array("apple", "banana", "orange");
echo $fruits[0]; // outputs "apple"
echo $fruits[1]; // outputs "banana"
echo $fruits[2]; // outputs "orange"
```

Adding Elements to an Array in PHP

You can add new elements to an array in PHP using the array_push() function. The basic syntax of the array_push() function is as follows:

```
array_push($array_name, value1, value2, ...);
```

For example:

```
$fruits = array("apple", "banana", "orange");
array_push($fruits, "grape", "kiwi");
print_r($fruits); // outputs Array ( [0] => apple [1] => banana [2] => orange [3] =>
grape [4] => kiwi )
```

In the above example, the array_push() function adds the values "grape" and "kiwi" to the $fruits array.

Removing Elements from an Array in PHP

You can remove elements from an array in PHP using the unset() function. The basic syntax of the unset() function is as follows:

```
unset($array_name[index]);
```

For example:

```
$fruits = array("apple", "banana", "orange");
unset($fruits[1]);
print_r($fruits); // outputs Array ( [0] => apple [2] => orange )
```

In the above example, the unset() function removes the element at index 1 from the $fruits array, which is the value "banana".

String Manipulation in PHP

Strings in PHP

A string is a sequence of characters that are used to represent text. In PHP, you can create strings using single quotes (') or double quotes ("). Single quotes are used to create literal strings, while double quotes allow you to include variables and special characters in a string.

Creating Strings in PHP

You can create a string in PHP by enclosing it in either single or double quotes. For example:

```
$name = 'John';
$message = "Hello $name!";
```

In the above example, the variable $name contains the string 'John', and the variable $message contains the string "Hello John!".

Concatenating Strings in PHP

You can concatenate (join) two or more strings in PHP using the dot (.) operator. For example:

```
$first_name = 'John';
$last_name = 'Doe';
$name = $first_name . ' ' . $last_name;
echo $name; // outputs "John Doe"
```

In the above example, the dot (.) operator is used to concatenate the $first_name and $last_name variables, separated by a space.

String Functions in PHP

PHP provides a number of built-in functions for manipulating strings. Here are some examples:

- ***strlen():*** Returns the length of a string.

```
$str = 'Hello World!';
echo strlen($str); // outputs 12
```

- ***strpos():*** Returns the position of the first occurrence of a substring in a string.

```
$str = 'Hello World!';
echo strpos($str, 'World'); // outputs 6
```

- ***str_replace():*** Replaces all occurrences of a substring in a string with another substring.

```
$str = 'Hello World!';
echo str_replace('World', 'Universe', $str); // outputs "Hello Universe!"
```

Working with Functions

Functions in PHP

A function is a block of code that performs a specific task. Functions are used to organize code and make it more reusable. In PHP, you can create your own functions or use built-in functions that are provided by the PHP language.

Creating Functions in PHP

You can create a function in PHP using the function keyword, followed by the name of the function and any arguments that it takes. For example:

```
function sayHello($name) {
   echo "Hello, $name!";
}
```

In the above example, the function sayHello takes one argument, $name, and echoes out a message that includes the value of the $name argument.

Calling Functions in PHP

You can call a function in PHP by using its name and passing any required arguments. For example:

```
$name = 'John';
sayHello($name); // outputs "Hello, John!"
```

In the above example, the function sayHello is called with the argument $name, which is set to the value 'John'.

Passing Arguments to Functions:

You can pass one or more arguments to a function in PHP. When you call a function, you can pass any required arguments inside parentheses, separated by commas. For example:

```
function multiply($a, $b) {
   return $a * $b;
}
$result = multiply(3, 5); // returns 15
```

In the above example, the function multiply takes two arguments, $a and $b, and returns their product. The function is called with the arguments 3 and 5, and the result is stored in the variable $result.

Returning Values from Functions:

A function in PHP can also return a value. To return a value from a function, use the return keyword followed by the value that you want to return. For example:

```
function add($a, $b) {
   $sum = $a + $b;
   return $sum;
}
$result = add(3, 5); // returns 8
```

In the above example, the function add takes two arguments, $a and $b, and returns their sum. The function is called with the arguments 3 and 5, and the result is stored in the variable $result.

Manipulating MySQL Databases with PHP

Connecting to a MySQL Database:

To connect to a MySQL database using PHP, you need to use the mysqli or PDO extension. Here's an example using the mysqli extension:

```
// Database configuration
$host = 'localhost';
$user = 'username';
$password = 'password';
$database = 'database';

// Create a new mysqli object
$mysqli = new mysqli($host, $user, $password, $database);

// Check for errors
if ($mysqli->connect_error) {
   die('Connect Error (' . $mysqli->connect_errno . ') ' . $mysqli->connect_error);
}
echo 'Connected successfully!';
```

In the above example, we first define the database configuration variables ($host, $user, $password, and $database). Then, we create a new mysqli object and pass in the database configuration variables. If there's an error, we use the die() function to display an error message.

Sending Data to a MySQL Database:

To send data to a MySQL database using PHP, you need to use SQL statements such as INSERT.

Here's an example:

```
// SQL query to insert data
$sql = "INSERT INTO users (name, email) VALUES ('John Doe', 'johndoe@example.com')";

// Execute the query
if ($mysqli->query($sql) === TRUE) {
    echo 'Data inserted successfully!';
} else {
    echo 'Error inserting data: ' . $mysqli->error;
}
```

In the above example, we use the INSERT statement to insert data into the users table. We execute the query using the query() method of the mysqli object. If there's an error, we display an error message.

Retrieving Data from a MySQL Database

To retrieve data from a MySQL database using PHP, you need to use SQL statements such as SELECT. Here's an example:

```
// SQL query to select data
$sql = "SELECT * FROM users";
// Execute the query and fetch the results
$result = $mysqli->query($sql);
if ($result->num_rows > 0) {
    while ($row = $result->fetch_assoc()) {
        echo 'Name: ' . $row['name'] . '<br>';
        echo 'Email: ' . $row['email'] . '<br><br>';
    }
} else {
    echo 'No data found!';
}
```

In the above example, we use the SELECT statement to select all data from the users table. We execute the query using the query() method of the mysqli object, and then loop through the results using a while loop. We use the fetch_assoc() method to fetch each row of data as an associative array, and then display the data on the screen.

Modifying and Removing Data from a MySQL Database:

To modify or remove data from a MySQL database using PHP, you need to use SQL statements such as UPDATE and DELETE. Here's an example:

```
// SQL query to update data
$sql = "UPDATE users SET email='johndoe_updated@example.com' WHERE id=1";


// Execute the query
if ($mysqli->query($sql) === TRUE) {
    echo 'Data updated successfully!';
} else {
    echo 'Error updating data: ' . $mysqli->error;
}
```

References

- Harvey M. Deitel. 2nd and 4 thed. Internet and World Wide Web: How to Program.
- Goodman: Java script bible (3rd edition) Gage Publishing
- Deitel and Deitel. "Java - How to Program", Addison-Wesley Press, Reading, Mass.1998,
- David Flanagan. "Java in a Nutshell (Java 1.1)", Second Edition, O'Reilly and
- Associates Publishing, Sebastopol, CA, 1997.
- David Flanagan. "Java Examples in a Nutshell (Java 1.1)",
- O'Reilly and Associates Publishing, Sebastopol, CA, 1997.
- Larry Wall and Randall Schariz. "Programming Perl", O'Reilly and Associates Publishing. Sebastopol, CA, 1994
- Scott Oaks and Henry Wong. "Java Threads", O'Reilly and Associates Publishing. Sebastopol, CA, 1997.
- Gary Cornell, Cay Horstmann. "Core Java", SUN Soft Press Publishing, Mountain View, 1996.
- S. Gundavaram. "CGI Programming on the World Wide Web", O'Reilly

STUDY QUESTIONS

1. What is the primary purpose of web design?
    a. To communicate information to users
    b. To create a visually appealing website
    c. To provide an interactive experience
    d. To make the website easier to use
2. What does HTML stand for?
    a. Hypertext Markup Language
    b. Hyperlink Markup Language
    c. High Markup Language
    d. Hyper Markup Language
3. Which of the following is NOT a CSS property?
    a. font-size
    b. background-color
    c. text-wrap
    d. margin
4. Which of the following is NOT a semantic HTML tag?
    a. <nav>
    b. <article>
    c.
    d. <footer>
5. Which of the following is a JavaScript data type?
    a. Boolean
    b. String
    c. Array
    d. All of the above
6. Which of the following is NOT a web framework?
    a. React
    b. Angular
    c. Django
    d. jQuery
7. What is the purpose of CSS?
    a. To create dynamic web pages
    b. To add interactivity to web pages
    c. To style and format web pages
    d. To provide server-side functionality
8. Which of the following is NOT an event in JavaScript?
    a. click
    b. mouseover
    c. hover
    d. scroll

9. Which of the following is used to add interactivity to web pages?
    a. HTML
    b. CSS
    c. JavaScript
    d. PHP
10. Which of the following is a CSS unit of measurement?
    a. pixels
    b. points
    c. ems
    d. All of the above
11. What is the difference between a web page and a website?
    a. A web page is a single page, while a website is a collection of web pages
    b. A web page contains text, while a website contains images
    c. A web page is used to store data, while a website is used to display data
    d. A web page is used to display information, while a website is used to store information
12. Which of the following is a key aspect of building a client-side application that is responsive?
    a. Using a fixed layout for all devices
    b. Not considering different screen sizes and device types
    c. Implementing media queries to adjust styling and layout based on device characteristics
    d. Not optimizing images and other assets for faster load times
13. What is the primary benefit of using AJAX in a web application?
    a. It allows the application to load faster and respond more quickly to user input.
    b. It enables the application to perform complex calculations and data processing on the client-side.
    c. It reduces the need for server-side scripting and programming.
    d. Improved security
14. Which programming language is commonly used for building web applications that utilize AJAX and server-side processing?
    a. Python
    b. PHP
    c. Ruby
    d. JavaScript
15. What is the primary purpose of building Web APIs that are ready for many client-side applications?
    a. To restrict access to the API to specific client-side applications
    b. To provide a single client-side application access to the API
    c. To allow multiple client-side applications to access the API
    d. None of the above

16. Which of the following is NOT a valid character entity in HTML?
    a. <
    b. >
    c. &
    d. &backslash;
17. Which of the following is used to specify the character encoding for an HTML document?
    a. <meta charset="">
    b. <html charset="">
    c. <head charset="">
    d. <title charset="">
18. What does the CSS box-sizing property do?
    a. Specifies whether an element is visible or hidden
    b. Specifies the width and height of an element
    c. Specifies the position of an element
    d. Specifies how the total width and height of an element is calculated
19. Which property is used to change the font family in CSS?
    a. font-size
    b. font-style
    c. font-weight
    d. font-family
20. Which property is used to add spacing between lines of text in CSS?
    a. line-height
    b. letter-spacing
    c. word-spacing
    d. text-indent
21. Which property is used to set the background color of an element in CSS?
    a. color
    b. background-color
    c. border-color
    d. text-color
22. Which property is used to specify the size of an element in CSS?
    a. width
    b. height
    c. size
    d. dimension
23. Which property is used to add a border around an element in CSS?
    a. border-style
    b. border-width
    c. border-color
    d. All of the above
24. What is the difference between '==' and '===' operators in JavaScript?

a. '==' compares values, while '===' compares both values and types.
b. '==' compares types, while '===' compares values.
c. Both '==' and '===' compare values and types, but '===' is stricter.
d. There is no difference between '==' and '==='.

25. Which of the following is **not** a valid way to declare a variable in JavaScript?
    a. var x = 10;
    b. let y = 20;
    c. const z = 30;
    d. variable a = 40;
26. What does the typeof operator in JavaScript do?
    a. It returns the type of the given variable or expression.
    b. It converts a value to a boolean.
    c. It compares two values for equality.
    d. It is not a valid operator in JavaScript.
27. What is the difference between null and undefined in JavaScript?
    a. They are the same thing
    b. Null is an object, while undefined is a primitive value
    c. Undefined is an object, while null is a primitive value
    d. Null and undefined are both primitive values
28. What is the purpose of a conditional statement in JavaScript?
    a. To repeat a block of code multiple times
    b. To define a function
    c. To execute a block of code only if a certain condition is true
    d. To assign a value to a variable
29. What is the difference between a for loop and a while loop in JavaScript?
    a. There is no difference
    b. A for loop is used for iterating over a fixed number of elements, while a while loop is used for iterating until a certain condition is met
    c. A while loop is used for iterating over a fixed number of elements, while a for loop is used for iterating until a certain condition is met
    d. A for loop is used for executing a block of code only once, while a while loop is used for executing it multiple times
30. Which function is used to retrieve the value of a specified property from an object?
    a. getValue()
    b. getProperty()
    c. retrieveProperty()
    d. Object.values()
31. Which of the following is not a benefit of using PHP?
    a. Cross-platform compatibility
    b. Open source and free to use
    c. Large developer community
    d. Limited database integration capabilities

32. What is the difference between ORM and ODM?
    a. ORM is used for SQL databases, while ODM is used for NoSQL databases.
    b. ORM is used for NoSQL databases, while ODM is used for SQL databases.
    c. ORM and ODM are interchangeable terms.
    d. ORM is used for data modeling, while ODM is used for object modeling.
33. Which of the following methods is used to connect PHP with a MySQL database?
    a. mysqli_connect()
    b. mysql_connect()
    c. pdo_connect()
    d. db_connect()
34. What is the difference between GET and POST methods in PHP?
    a. GET method sends data through the HTTP request header, while POST method sends data through the HTTP request body.
    b. GET method sends data through the HTTP request body, while POST method sends data through the HTTP request header.
    c. There is no difference between GET and POST methods in PHP.
    d. GET and POST methods are used for different programming languages, not PHP.
35. What is the purpose of the mysqli_fetch_array() function in PHP?
    a. To execute a SELECT query on a database
    b. To fetch a row from a result set as an array
    c. To retrieve the number of rows in a result set
    d. To return the last inserted ID of a query

ANSWER KEY

1. ***Answer: d. to make the website easier to use***. While all the options mentioned are important aspects of web design, the primary purpose is to make the website easier to use. This includes designing a clear and easy-to-use navigation, ensuring that content is easy to read and understand, and optimizing the website for different devices and screen sizes.

2. ***Answer a. Hypertext Markup Language.*** HTML stands for Hypertext Markup Language. HTML is a markup language used to create and structure content on the web.

3. ***Answer: c. Text-wrap.*** This is NOT a CSS property. font-size, background-color, and margin are all valid CSS properties used to style and format content on the web.

4. ***Answer: c. .*** The div tag is NOT a semantic HTML tag. The nav, article, and footer tags are all examples of semantic HTML tags, which are used to add meaning and structure to the content on a web page. The div tag, on the other hand, is a generic container used for grouping and styling content.

5. ***Answer d. All of the above.*** JavaScript supports a range of data types, including booleans, strings, and arrays.

6. ***Answer: d. jQuery.*** jQuery is not a web framework, but a JavaScript library designed to simplify client-side scripting of HTML. The other options are all web frameworks: React is a JavaScript library for building user interfaces, Angular is a TypeScript-based open-source web application framework, and Django is a high-level Python web framework that enables rapid development of secure and maintainable websites.

7. ***Answer: c. to style and format web pages,*** CSS is used to separate the presentation of a web page from its content, allowing developers to control the layout, design, and visual appearance of web pages. CSS can be used to control various aspects of a web page's styling, including font size and type, color, background images, layout, and

positioning of elements. While CSS can add some interactivity to web pages, its primary purpose is to control the presentation of the content. It does not provide server-side functionality.

8. ***Answer: c. hover,*** the "hover" event is not a valid event in JavaScript. The other events listed, "click", "mouseover", and "scroll", are all valid events that can be used to trigger JavaScript code.

9. ***JavaScript is used to add interactivity to web pages***. While HTML is used for the content and structure of a web page and CSS is used for styling and layout, JavaScript is used to make the page interactive by allowing users to interact with elements on the page, trigger actions and events, and dynamically update content without having to refresh the page. PHP, on the other hand, is a server-side scripting language used for web development that allows for the creation of dynamic web pages, but it is not typically used for client-side interactivity.

10. ***All of the above options are CSS units of measurement***. Pixels (px) and points (pt) are absolute units that represent fixed sizes, while ems (em) is a relative unit that represents the font-size of the element. These units are commonly used in CSS to specify measurements for various properties such as width, height, font-size, margin, padding, etc.

11. ***Answer: a. A web page is a single page, while a website is a collection of web pages,*** A webpage is a single page, while a website is a collection of related pages with a common homepage. Website includes navigation, search, and forms, while a webpage is focused on displaying specific content.

12. ***Answer: c. Implementing media queries to adjust styling and layout based on device characteristics.*** Implementing media queries is key to building a responsive client-side application. Media queries allow developers to create device-specific stylesheets, ensuring the application adapts to different viewing environments. A fixed layout for all devices and not optimizing assets can lead to poor performance on various devices.

13. ***Answer: a. It allows the application to load faster and respond more quickly to user input,*** Using AJAX in web applications allows faster loading and more responsive user input. It enables web pages to make server requests in the background, without reloading or refreshing the page. This improves the user experience with faster feedback and reduced waiting time for new content. AJAX can perform complex client-side calculations, but that is not its primary benefit. It can also reduce the need for server-side scripting, but not eliminate it entirely. AJAX does not provide improved security as security concerns are addressed through other mechanisms.
14. ***Answer: d. JavaScript,*** JavaScript is the programming language commonly used for building web applications that utilize AJAX and server-side processing. JavaScript is a versatile programming language that can be used on both the client-side and server-side of web applications. It is particularly well-suited for building interactive, responsive user interfaces, as well as for implementing server-side logic using frameworks like Node.js. While Python, PHP, and Ruby are also commonly used in web development, they are not typically associated specifically with AJAX and server-side processing in the same way that JavaScript is.

15. ***Answer: c. To allow multiple client-side applications to access the API,*** the main purpose of building web APIs is to enable many client-side apps to access the API, providing a standard interface for different types of apps like web browsers, mobile apps, and software programs. This widens the availability of data and services, increasing the potential value of the application. Limiting access to specific apps or a single app would restrict the API's usefulness and accessibility.

16. ***Answer: d. &backslash;*** This is not a valid character entity in HTML. The correct escape sequence for a backslash is `&#92;`

17. ***Answer: a. <meta charset="">.*** This tag is used in the head section of an HTML document to specify the character encoding. The charset attribute specifies the character encoding used for the document.

18. ***Answer: d.*** Specifies how the total width and height of an element is calculated. The box-sizing property is used to specify how the total width and height of an element is calculated. The default value is content-box, which includes only the content in the calculation, while border-box includes the content, padding, and border in the calculation.

19. ***Answer: d. font-family,*** the font-family property is used to change the font family in CSS. It specifies the name of the font family, such as "Arial" or "Times New Roman".

20. ***Answer: a. line-height:*** The line-height property is used to add spacing between lines of text in CSS. It specifies the height of each line of text, which includes the space between the lines.

21. ***Answer: b. background-color***: The background-color property is used to set the background color of an element in CSS. It specifies the color to be used for the background of the element.

22. ***Answer: a. width***: The width property is used to specify the size of an element in CSS. It specifies the width of the element, which can be specified in pixels, percentages, or other units.

23. ***Answer: d. All of the above***: The border-style, border-width, and border-color properties are used together to add a border around an element in CSS. The border-style property specifies the style of the border, such as "solid" or "dotted", the border-width property specifies the width of the border, and the border-color property specifies the color of the border.

24. ***Answer: a. '==' compares values, while '===' compares both values and types***. The '==' operator compares values without considering their data types, while '===' compares both values and data types. For example, 5 == "5" would be true, but 5 === "5" would be false because 5 is a number and "5" is a string.

25. ***Answer: d. variable a = 40***. In JavaScript, variables can be declared using the 'var', 'let', or 'const' keywords. The syntax for declaring a variable using any of these keywords is <keyword> <variable_name> = <value>;. Option d is not a valid way to declare a variable in JavaScript because 'variable' is not a valid keyword.

26. ***Answer: a. It returns the type of the given variable or expression.*** The **typeof** operator in JavaScript returns a string indicating the type of the operand

27. ***Answer: b. Null is an object, while undefined is a primitive value***. In JavaScript, undefined means that a variable has been declared but has not been assigned a value, while null represents the intentional absence of any object value.

28. ***Answer c. To execute a block of code only if a certain condition is true***. Conditional statements, such as if/else and switch, are used in JavaScript to control the flow of execution based on whether a certain condition is true or false.

29. ***Answer: b. A for loop is used for iterating over a fixed number of elements, while a while loop is used for iterating until a certain condition is met***. A for loop is used when you know the number of times you need to loop through a block of code, while a while loop is used when you don't know the number of times you need to loop through a block of code.

30. ***Answer: b. getProperty().*** There is no built-in function called "getValue()" or "retrieveProperty()" in JavaScript. To retrieve the value of a specified property from an object, you can use the dot notation or bracket notation along with the name of the property.
31. ***Answer: d. Limited database integration capabilities.*** This is not a benefit of using PHP. In fact, PHP has extensive database integration capabilities and can easily connect to various databases such as MySQL, PostgreSQL, Oracle, and more.

32. ***Answer: a. ORM is used for SQL databases, while ODM is used for NoSQL databases.*** ORM (Object Relational Mapping) maps objects to tables in SQL databases, while ODM (Object Data Mapping) maps objects to documents in NoSQL

databases. Both allow developers to work with objects and classes in code and automatically translate them to the database.

33. ***Answer: a. mysqli_connect().*** To connect PHP with a MySQL database, developers use the mysqli_connect() function from the MySQL Improved Extension (mysqli). This replaces the deprecated mysql_connect() function in newer versions of PHP. The pdo_connect () and db_connect () functions are not valid options.

34. ***Answer: a. The GET method s ends data through the URL in the HTTP request header, while the POST method sends data through the HTTP request body***. In PHP, the GET method is commonly used for retrieving data from a server, while the POST method is commonly used for submitting data to a server.

35. ***Answer: b. To fetch a row from a result set as an array.*** The purpose of the mysqli_fetch_array() function in PHP is to fetch a row from a result set as an array. This function is used to retrieve a row of data from a result set returned by a SELECT query, and it returns an array that corresponds to the fetched row. The array can be indexed both numerically and by field name, which allows for easy access to the values of the returned columns. The mysqli_fetch_array() function can be used with the procedural or object-oriented style of the mysqli extension in PHP, which provides access to the MySQL database server.