UEMF
الجامعة الأورومتوسطية بفاس
EUROMED UNIVERSITY OF FES
UNIVERSITÉ EUROMED DE FÈS

EIDIA
École d'Ingénierie Digitale et d'Intelligence Artificielle

Web Technologies

1st Year Full Stack Engineering Cycle

Academic Year: 2024-2025

Pr. CHRAA MESBAHI Soukaina

Course Objectives

Main Objective:

- Develop a solid foundation in web development through the mastery of HTML5, CSS3, and an understanding of web protocols.

Sub-Objectives:

- Understand Web Fundamentals
- Create Static Web Pages
- Style and Make Responsive Pages
- Introduction to JavaScript

Course Outline

1. Introduction to Web Technologies

2. HTML and CSS

3. JavaScript Basics

4. PHP for Server-Side Scripting

5. Performance for Server-Side Scripting

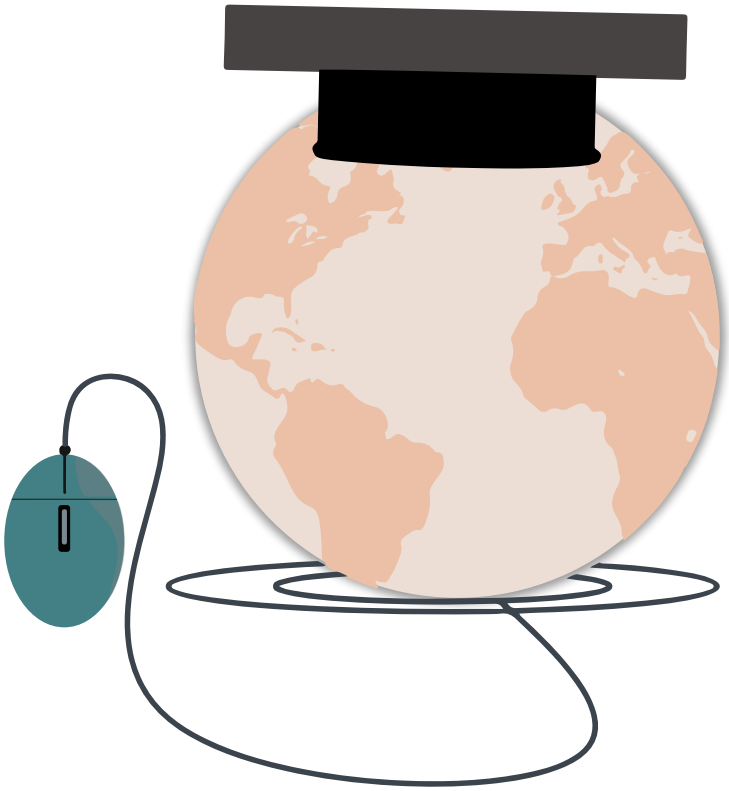
Introduction to Web Technologies

WEB TECHNOLOGY

What is Web Technology?

Collection of tools, methods, and software that are used to create, develop, and maintain websites and web applications

About building and delivering content and services over the Internet



COMPONENTS

Web Browsers

Primary interface through which users access the internet.

Web Servers

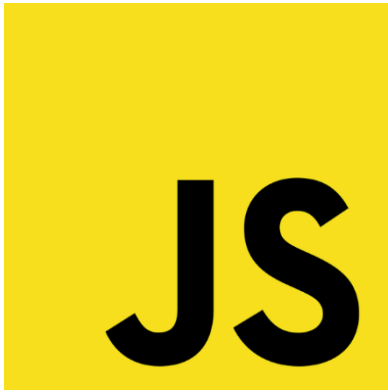
Powerful computers that store and serve web content to users when requested.

HTML

Markup language used to structure and present content on the web.

CSS

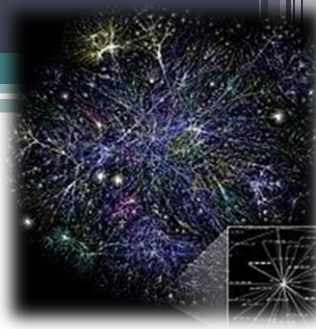
Visual presentation of web content



JavaScript

Dynamic programming language that adds interactivity and functionality to web pages

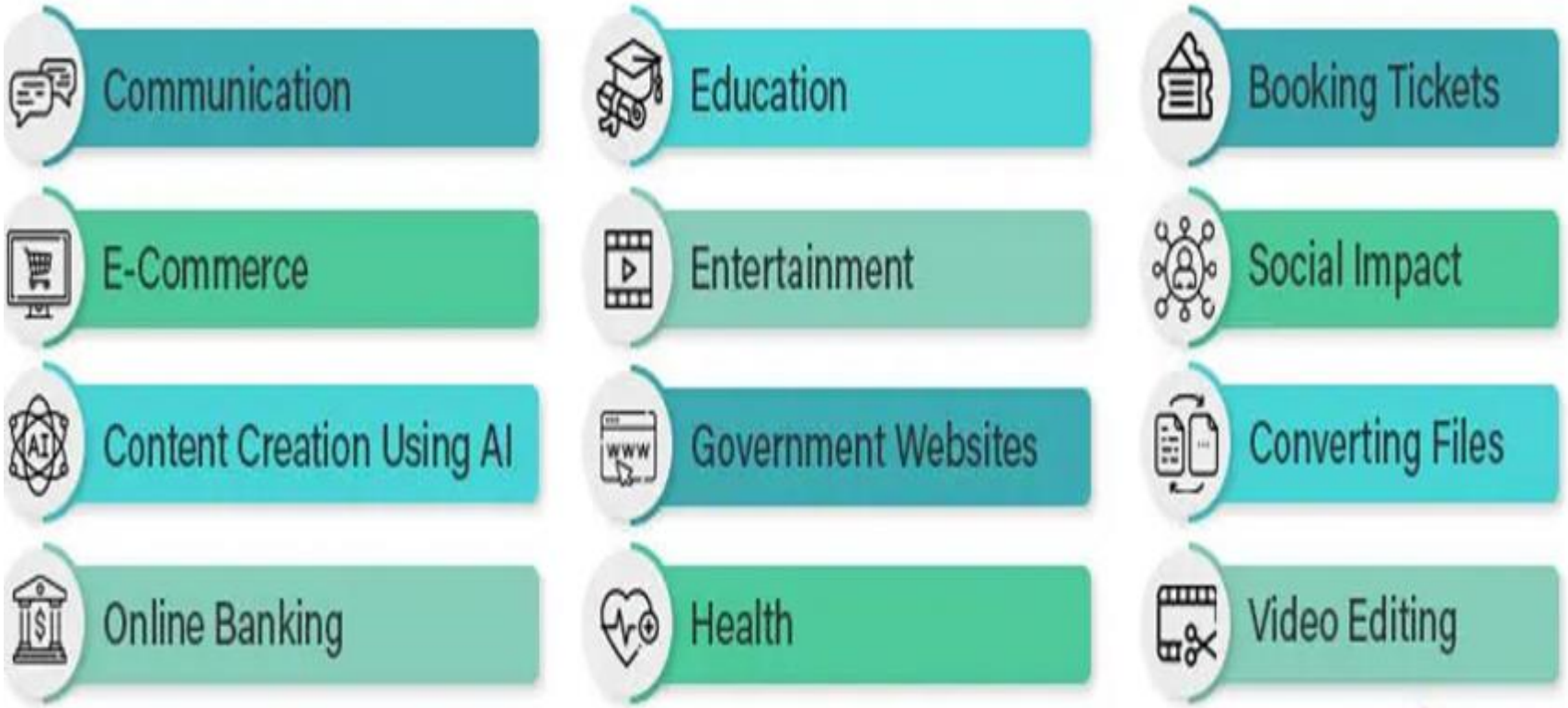
What is Internet ?



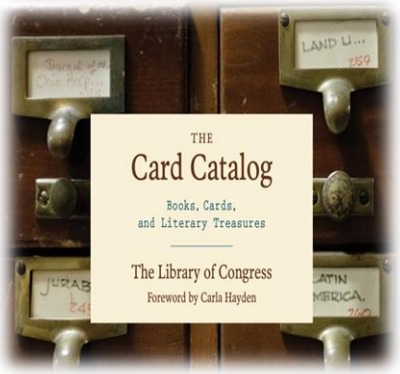
The Internet

- It is the largest network in the world that connects hundreds of thousands of individual networks all over the word.
- With the Internet, it's possible to access almost any information, communicate with anyone else in the world.
- Works using protocols (a set of rules) like **TCP/IP**, which enables the exchange of information between connected devices.

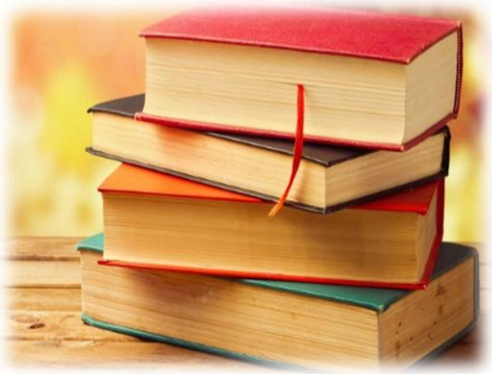
Uses of The Internet



What is the « Web » ?



URLs are like the reference numbers of books



Each book represents a website

The web is a vast library filled with books

Signs or maps in a library

Hyperlinks

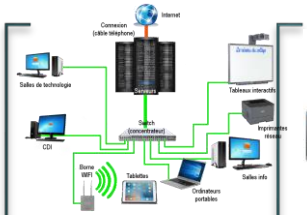
WEB

- The ***Web, or "World Wide Web",*** is a global system of interconnected information organized into web pages that contain text, images, and multimedia content.

- A website comprises multiple web pages linked by hyperlinks, facilitating navigation from one page to another.

- Through **hypertext**, URLs (addresses of web pages) can be embedded within pages to reference other resources, creating an immense interconnected structure on a global scale, which explains the term "World Wide Web."

WEB - History



ARPANET

Creation of the first computer network

1960s

(TCP/IP)

Protocol f or linking multiple networks together

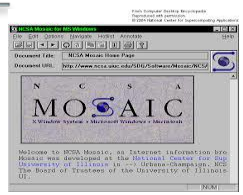
1970s

Invention of the Web

By Tim Berners Lee

Creation of a hypertext linking system

1989s



Web Browser 1.0

Appearan ce of the first browsers (Mosaic, Netscape).

1990s

Web 2.0 :

the rise and burst of the dot-com bubble . Social networks, blogs, content sharing (Facebook, YouTube).

2000s

Web 3.0 :

Semantic Web, Artificial Intelligen ce.

2010s

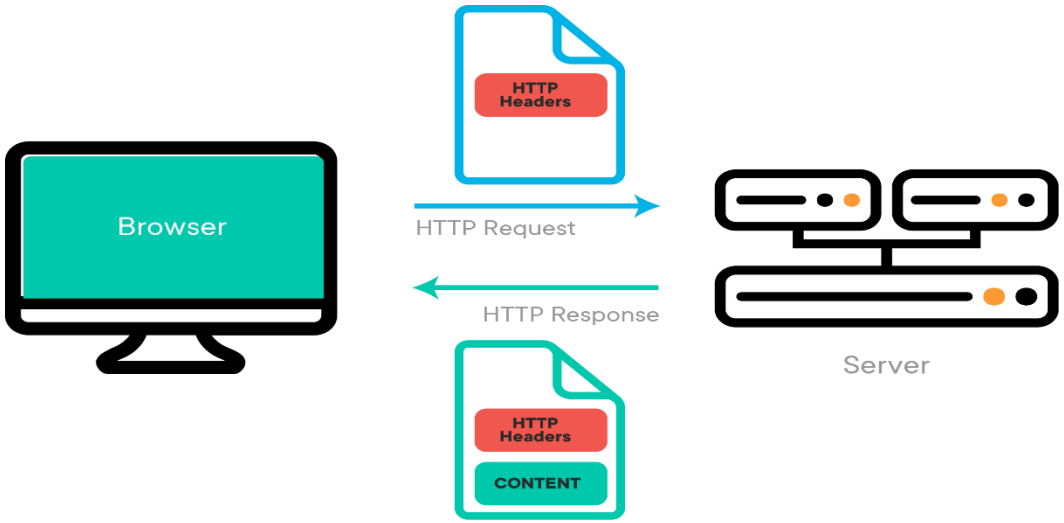
Web 4.0 :

Converge nce of IoT, augment ed reality, and AI.

2025 and beyond

How the WEB works !!

- Use of a **client/server** architecture
- When a user types the (**URL**) of a website into their browser (**client**), it sends a request to the **server** hosting that site.
- The **server** then sends the requested page to the browser, which displays it.
- To communicate effectively, the client and web server use a common protocol**: HTTP**.



How to access the Web?

- **Client**
  - It is the user's Internet browser.
  - Identifies servers on the Internet.
  - Requests resources from servers.
  - Displays resources to users.
- **Browser**
  - It is software that analyzes the (X)HTML and CSS code of web pages and produces a visual result that is easy for humans to read.
  - Examples of Web browser: Firefox, Internet Explorer, Google Chrome, Opera etc

Hyper Text

Transfer Protocol

“The Hypertext Transfer Protocol (HTTP) is the foundation of the World Wide Web, and is used to load webpages using hypertext links.”

HYPERTEXT TRANSFER PROTOCOL

HTTP is a ***Connectionless protocol*** which can deliver any sort of data

The basic structure of HTTP communication follows the ***request - response model.***

The purpose of the HTTP protocol is to enable file transfer, primarily in HTML format, located via a string of characters called a URL, between a browser (the client) and a web server.

URL : Uniform Resource Locator

- Universal form of URI and a standardized naming convention for designating resources accessible over the ***Internet***.
- In the case of the Web, resource = document or fragment

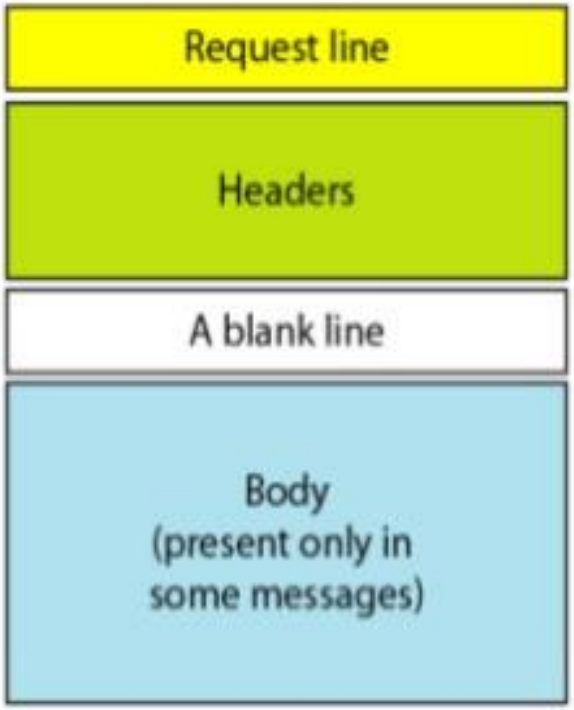
- Main protocols used in URLs:
  - ✓ **HTTP, HTTPS**: Two web protocols, with and without encryption and authentication.
  - ✓ **FTP**: File Transfer Protocol, sometimes used on the Web for downloading large files.
  - ✓ **MAILTO**: A pseudo-protocol denoting an email address.
- **Default Ports**: 80 for HTTP, 443 for HTTPS.

HTTP – REQUEST

HTTP requests are *messages sent by the client (such as a browser) to a server to request a resource.*

Every HTTP request message has the same basic structure:

- ✓ ***Request line***
- ✓ ***Header field(s)***
- ✓ ***Request body***



HTTP – REQUEST Line

A command line in an HTTP request is a line that specifies the type of document being requested, the method to be used, and the version of the protocol in use.

Method + Request URI + Protocol Version

- **HTTP Method** : Tell the server what action to perform
  - ***GET*** **:** Retrieve information from the server (web page, file, etc.)
  - ***POST*** **:** Send data to the server.
  - ***PUT*** **:** Update an existing resource.
  - ***DELETE*** **:** Remove a document from the server.

- **URL :** Specifies the resource path, or the absolute path of the protocol port, for example: **/index.html** or **/api/user.**

- **Protocol Version:** Indicates the version of the HTTP protocol used, for example: **HTTP/1.1** or **HTTP/2.**

Example : GET / index.html HTTP/1.1

HTTP – REQUEST Header fields

Header fields provide additional information about the request, such as the types of content the client can accept or information about the client itself.

Each header field consists of two parts: a **name** and a **value** separated by a colon **(:)**.

Examples of header fields:

- **User-Agent :** Describes the client making the request, namely the browser and the IOS operating.

  Example : **User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;x64)**

- **Host :** Indicates the server´s domain name which the request is sent.

  Example : **Host: www.example.com**

HTTP – REQUEST Header fields

- **Accept :** Specifies the content types that the client is willing to receive.

  Example : **Accept: text/html, application/json**

- **Content-Type :** (In the case of a POST or PUT request): Indicates the data type being sent in the request's body.

  Example : **Content-Type: application/x-www-form-urlencoded**

HTTP – REQUEST Body

The body of the request is optional and is typically present only in certain requests, particularly those using the **POST** or **PUT** method. It contains the data that the client wishes to send to the server, such as form information.

- **Body content:** The content depends on the request. For example, when submitting a form via POST, the body of the request contains the form data.
- **Separation :**The body of the request is separated from the headers by a blank line.

Example : **name=JohnDoe&email=johndoe@example.com**

HTTP – REQUEST EXAMPLE

Request Line

METHOD URL
VERSION<crlf>
HEADER : Value <crlf>
.
.

POST /submit-form HTTP/1.1
Host: www.example.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
Accept: text/html, application/json
Content-Type: application/x-www-form-urlencoded
Content-Length: 35
A blank line
name=JohnDoe&email=johndoe@example.com

Request Header

Request body

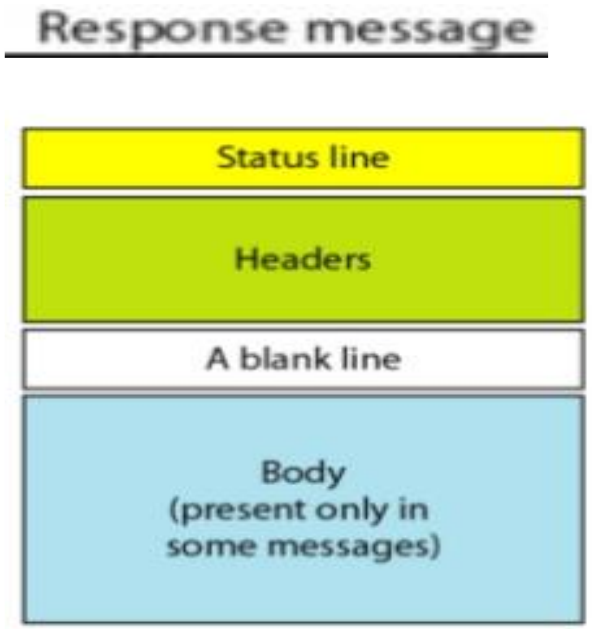
RESPONSE MESSAGE

- Each Request has a Response.
- Every HTTP Response message consists of the following:

✓ ***Status line***

✓ ***Header field(s) (one or more)***

✓ ***Message body (optional)***



RESPONSE MESSAGE -STATUS LINE

The start line of an HTTP response, called the **status line**, contains the following information:

- The protocol version, usually **HTTP/1.1**
- A status code, indicating **success** or **failure** of the request.
  Common status codes are 200, 404, or 302
- A status text. A brief, purely informational, textual description of the status code to help a human understand the HTTP message

A typical status line looks like: HTTP/1.1 404 Not Found.

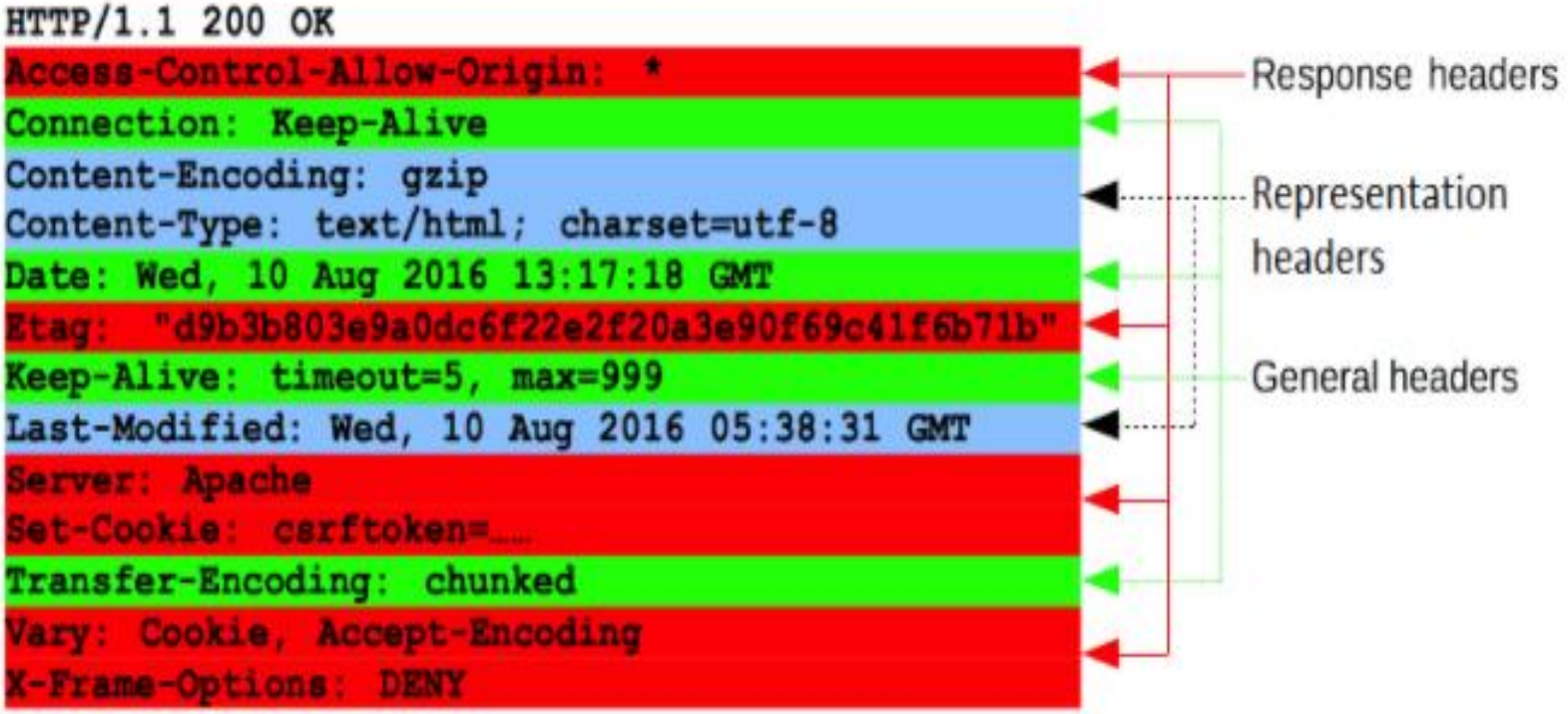
RESPONSE MESSAGE - HEADER FIELDS

HTTP headers for responses follow the same structure as any other header: a case-insensitive string followed by a colon (':') and a value whose structure depends upon the type of the header.

The whole header, including its value, presents as a single line.

- General headers, like Via, apply to the whole message.
- Response headers, like **Vary** and **Accept-Ranges**, give additional information about the server which doesn't fit in the status line.
- Representation metadata headers (formerly entity headers), like **Content-Length** that describe the encoding and format of the message body (only present if the message has a body).

RESPONSE MESSAGE - HEADER FIELDS



HTTP Status Codes

HTTP status codes are responses sent by the server to the client to indicate the result of processing a request. They help the client (such as a web browser) understand whether the request was successfully processed, if errors occurred, or if further actions are required.

Code	Message	Description
10x	Informational message	Indicate that the request is being processed.
20x	Success	These codes indicate the successful completion of the transaction.
200	OK	The request was successfully completed.
201	CREATED	It follows a POST request and indicates success. The body of the response is expected to provide the URL where the newly created document should be located.

HTTP Status Codes

Code	Message	Description
200	ACCEPTED	The request has been accepted, but the subsequent process has not been completed.
201	PARTIAL INFORMATION	When this code is received in response to a GET request, it indicates that the response is incomplete.
204	NO RESPONSE	The server has received the request, but there is no information to send back.
301	RESET CONTENT	The server instructs the browser to clear the contents of the form fields.
302	PARTIAL CONTENT	This is a response to a request that includes the Range header. The server must indicate the Content-Range header.

HTTP Status Codes

Code	Message	Description
30x	**Redirection**	These codes indicate that the resource is no longer at the specified location.
301	MOVED	The requested data is at a new URL, but it may have been moved since then…
302	FOUND	The server has received the request, but there is no information to return.
303	METHOD	This implies that the client should try a new address, preferably using a method other than GET.
304	NOT MODIFIED	If the client has made a conditional GET request (asking if the document has been modified since the last time) and the document has not been modified, it returns this code.

HTTP Status Codes

Code	Message	Description
40x	**Erreur due au client**	These codes indicate that the request is incorrect.
400	BAD REQUEST	The syntax of the request is malformed or cannot be fulfilled.
401	UNAUTHORIZED	The message parameter provides the specifications for acceptable authorization formats. The client must reformulate its request with the correct authorization data.
402	PAYMENT REQUIRED	The client must reformulate their request with the correct payment information.
403	FORBIDDEN	Access to the resource is simply forbidden.
404	NOT FOUND	Classic! The server found nothing at the specified address.

HTTP Status Codes

Code	Message	Description
50x	**Erreur due au serveur**	These codes indicate that there was an internal server error.
500	INTERNAL ERROR	The server encountered an unexpected condition that prevented it from fulfilling the request.
501	NOT IMPLEMENTED	The server does not support the requested service.
502	BAD GATEWAY	The server received an invalid response from the server it was trying to access while acting as a gateway or proxy.
503	SERVICE UNAVAILABLE	The server cannot respond to you at the moment because the traffic is too heavy (all lines to your correspondent are busy; please try again later).
504	GATEWAY TIMEOUT	The server's response was too long regarding the time the gateway was prepared to wait for it (the time allotted to you has now expired...).

Static web and et Dynamic Web

Static Web:

Static websites consist of pages with fixed, predefined content that does not change based on user actions.

Example : A company presentation site with information that does not change often.

Limitations :

- Any content modification requires a manual change to the HTML file.

Technologie :

HTTP/1.1 200 OK
Date: Tue, 22 Jun 2015 13:18:15 GMT
Server: Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.1.2 mod_ssl/2.8.9
OpenSSL/0.9.6
"63f3d-8e-40d8
timeout=15, ma
<html>
<body>
<h1> pa
 cor
</body>
<html>

ag:

GET /page1.html HTTP/1.1
Host: www.e-monsite.com
Connection: Keep-Alive
User-Agent: Mozilla/5.0 (compatible; Konqueror/3.1; Linux; fr)
Referer: http://www.e-monsite.com
Pragma: no-cache
Cache-control: no-cache
Accept: text/html, image/jpeg, image/png, text/*, image/*, */*
Accept-Encoding: x-gzip, x-deflate, gzip, deflate, identity
Accept-Charset: iso-8859-1, utf-8;q=0.5, *;q=0.5 Accept-Language: fr, en

2. Generation of the web page

WEB SERVER

Client (Web browser)

1. Sending an HTTP request

3. Response to the HTTP request
HTML, CSS, JS

http://www.e-monsite.com/page1.html

Static web and et Dynamic Web

Dynamic Web :

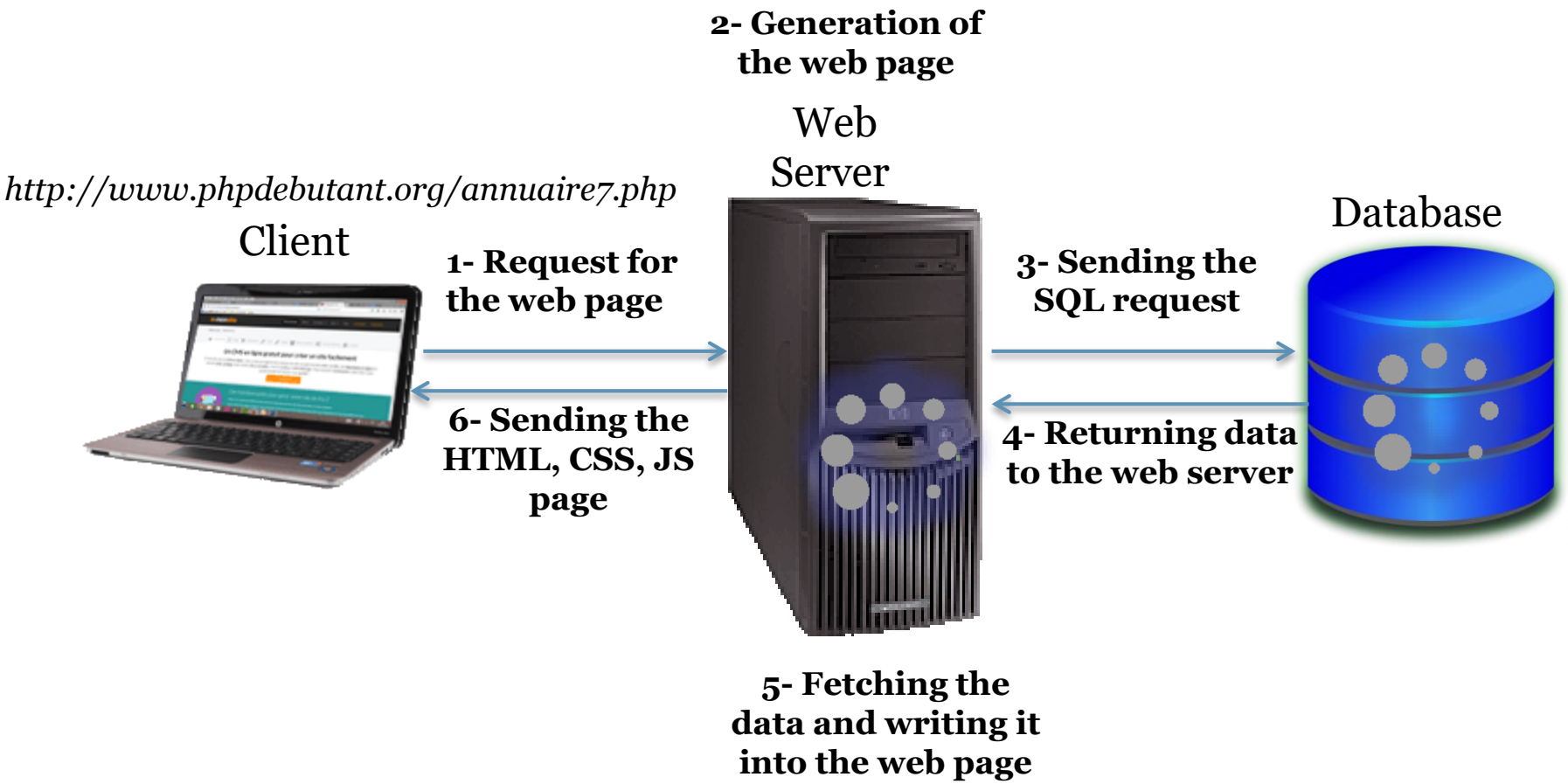
Dynamic websites consist of pages whose content can change based on user interactions or data from a database.

Characteristics :

- **Dynamic Content:** The server generates content on the fly based on the request parameters or user data.
- **Technologies :** Utilizes server-side languages like PHP, Python, Ruby, or frameworks like Node.js. Data can be stored and retrieved from databases.

Example : An e-commerce site where products and prices change based on inventory and user interactions.

Dynamic Web


2- Generation of the web page
Web Server
http://www.phpdebutant.org/annuaire7.php
Client
Database
1- Request for the web page
3- Sending the SQL request
6- Sending the HTML, CSS, JS page
4- Returning data to the web server
5- Fetching the data and writing it into the web page

HTML Version	Year
HTML 1.0	1991
HTML 2.0	1995
HTML 3.2	1997
HTML 4.01	1999
XHTML	2000
HTML 5	2014

Hyper Text Mark-up Language

Skeleton of all web pages.

Provides structure to the content appearing on a website

Created by Tim Berners-Lee in 1991

How is a website created?

HTML & CSS Code

Translation by the computer

The visible result in the browser

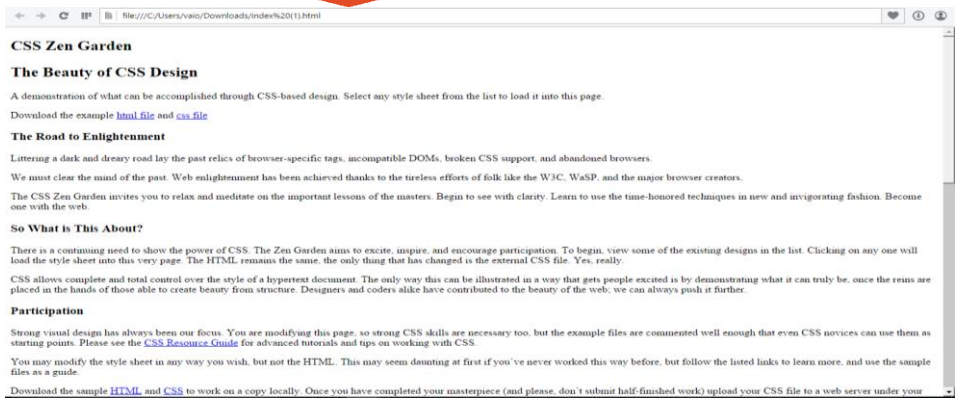
Web Language

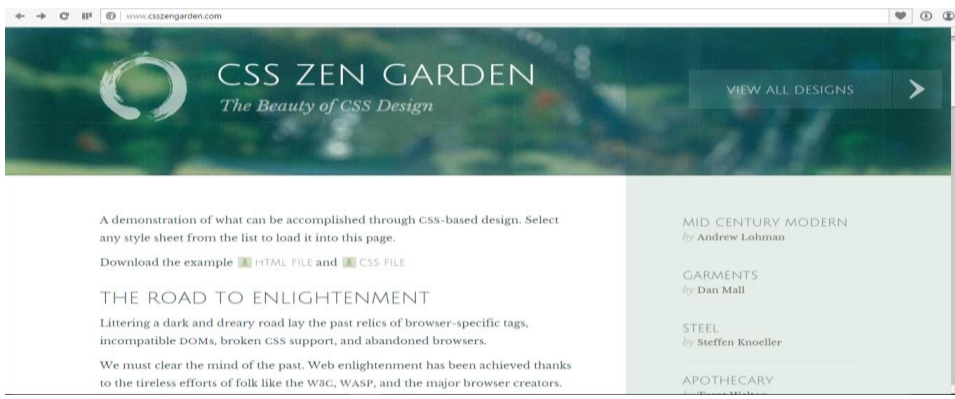
Two languages for creating a website

- **HTML (HyperText Markup Language):** Its role is to manage the content. So, you will write in HTML what should be displayed on the page: text, links, images.
- **CSS (Cascading Style Sheets):** The role of CSS is to manage the appearance of the web page (styling, colors, text size, etc.). This language was introduced to complement HTML in 1996.

Background of the web page

Design of the web page



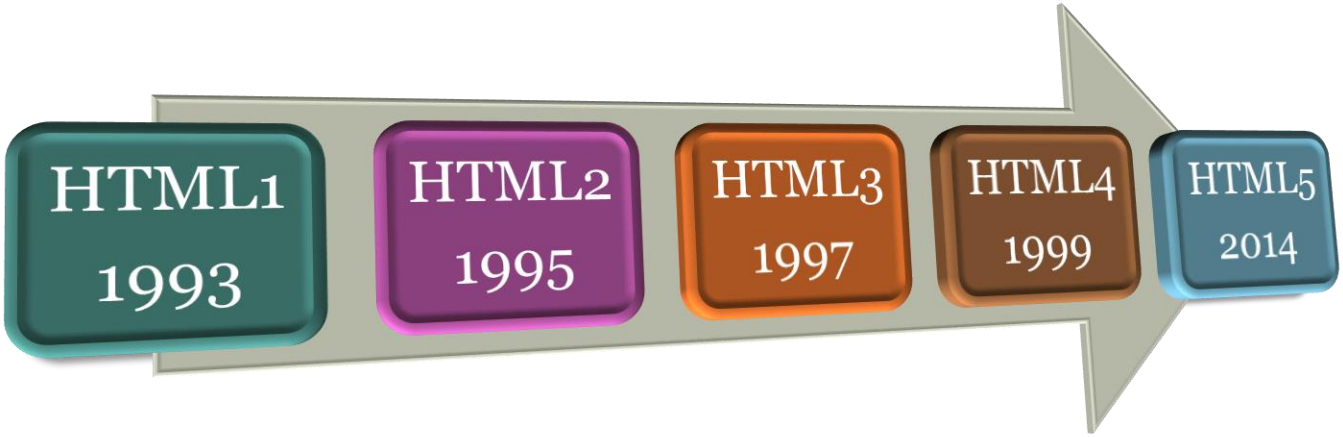


HTML -Language

- **HTML (HyperText Markup Language)** is the standard language used to create web pages.
- It allows structuring content with tags.
- Each content element is surrounded by an opening tag and a closing tag.
- HTML uses some ***predefined tags*** which tells the browser about content display property, that is how to display a particular given content.

- We do not describe the presentation (this will be the role of CSS).
- We do not describe dynamic behavior (this will be the role of JavaScript and server-side languages).

HTML -Language



Usefulness of HTML

The HTML language allows:

- Defining the logical structure of a web document;
- Composing a set of formatting commands;
- Relying on the concept of an environment with a beginning and an end;
- Using delimiters: these are tags or markers;
- To implement HTML, all you need is a text editor to type the page code and a web browser to display the formatted page.
- Tags are case-insensitive and can be written in lowercase, uppercase, or a mix of both.

HTML5 Tools

To learn the HTML language, we need

Text Editor

Modern Browser

HTML5 Tools: Text editor

To create your website, you need software.

There are two main categories:

- WYSIWYG (What You See Is What You Get): These behave like word processing software: no need to know HTML or CSS to create your website. The most well-known are Mozilla Composer, Microsoft Expression Web, Dreamweaver, KompoZer, etc.

  - **Problems: They often generate poor-quality HTML and CSS code, and there is no control over the code.**

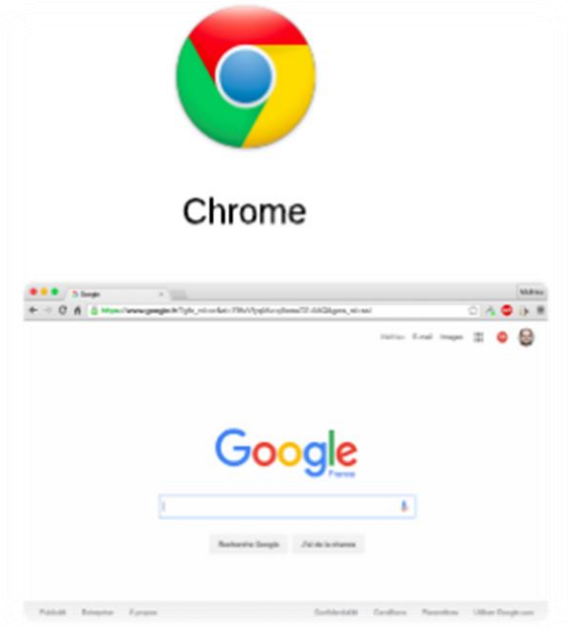
- Text Editors: Not specific to HTML and CSS but greatly facilitate development: Sublime Text, Notepad++, Kwrite, gedit, Visual Studio Code, etc.

In practical work, we will use the text editor **Visual Studio Code**.

HTML5 Tools: Browser

A browser allows you to view websites.

- There are many of them: Google Chrome, Mozilla Firefox, Internet Explorer, Edge, Opera, Safari, etc.....



HTML5 markup language

- Tags, also called elements, are commands intended for the browser, enclosed in angle brackets **(**< and >**)**. Thus, a tag is written as **<tag>.**
- There are two types of tags:
  1. Tags that open and then close, and surround content.

     Example : **** A little italic****
  2. Tags that open and close at the same time.

     Example :

```
<h1 >An example of a title</h1 >
<hr />
<div >
        <p>
              Text in <b> bold</b >, then <em > italic </em >,
             then <b> <em > bold and italic </em ></b>.
       </p >
       <i mg src= " my_image.png " alt= « An image " />
</div >
```

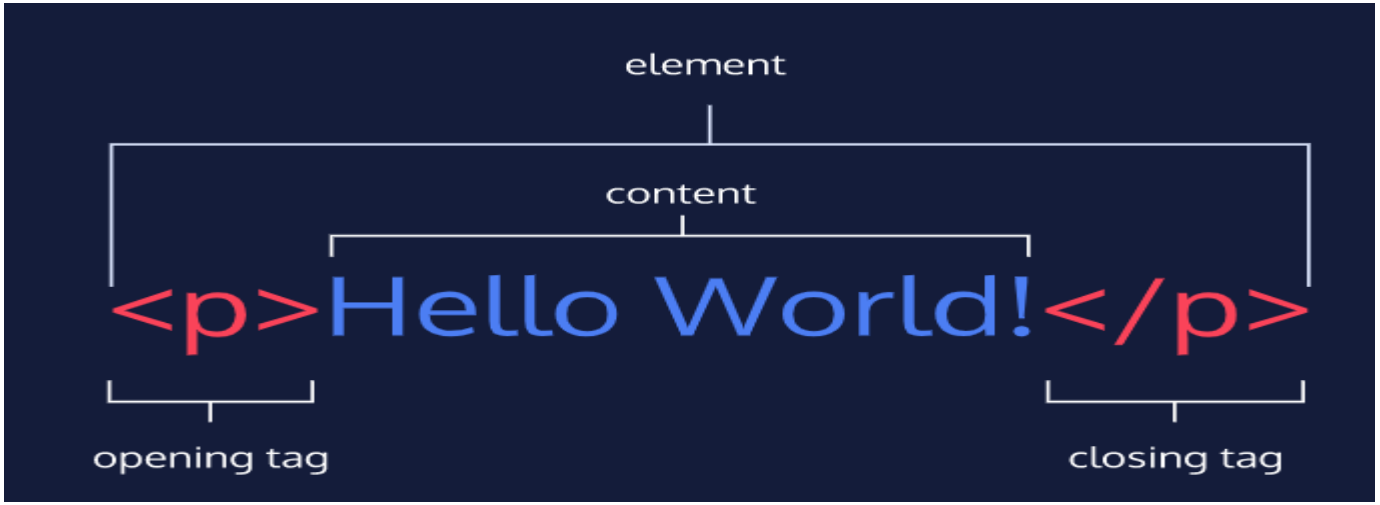
HTML5 markup language

Example :

<b>	b for bold, which means bold text.
<i>	i for italic, which means italic text.
<p>	p for paragraph, which means paragraph.
	div for division, which means division
<table>	table means table
<form>	form means form
<img />	img for image, which means image

HTML -Sturcture

For Example, to create a paragraph, one must use the paragraph tags***( )*** and to insert an image one must use the img tags***(<img />)***



Basic HTML Tags & Attributes

- **Paired tags:**

A tag requires an opening tag and a closing tag to function properly.

HTML Paragraph * * - to write paragraph statements in a webpage.

```
<p>I am a paragraph</p>
```

- **Self-closing(orphan)tags:**

Self-closing tags close themselves and do not need a closing tag.
Example of an image:

```
<img src="image.jpg" alt="Description of image">
```

HTML Images *<img src=”source_of_image“>* - to insert an image into our web page

Basic HTML Tags & Attributes

- **Attributes:**

Attributes provide additional information to tags. They are included in the opening tag and modify its behavior or appearance.

```
<img src="image.jpg" alt="Description of image">
```

Basic HTML Tags & Attributes

- ✓ ***Break </br> -*** to break line and act as a carriage return.
- ✓ ***Horizontal Rule <hr> -*** to break the page into various parts, creating horizontal margins
- ✓ ***HTML Attributes -*** to provide extra or additional information about an element
- ✓ ***HTML Comments -*** for inserting comments in the HTML code.

Tags & hierarchy

- Tags are structured **hierarchically**.
- Each tag inherits the properties of the higher-level tag—unless it redefines them.
- Tags and hierarchy:

```
<p>This text is <b>only bold</b>.</p>
<p>This text is <b>only bold
<i>and this part is bold and italic</i></b>.</p>
```

Closing order:

→ It is essential to close the tags in the reverse order in which they were opened.

Comments

Just like in programming languages, it is essential to comment on your HTML code. Comments allow you to:

- Separate and structure the different sections of the code.
- Add important information (such as the author's name, creation date, or version tracking) without displaying it in the browser.
- Provide explanations to clarify complex parts of the code.
- Facilitate maintenance, whether for the creator or for other developers.

→ Comments are HTML code that is not executed and thus invisible in the browser.

Syntax: **<!--Text within comments -->**

Note: ***Comments are visible when viewing the source code (right-click followed by "View page source").***

```
<h1 >a title (which will be displayed) </h1 >
<p> This sentence will be displayed in the browser.
    <!-- However, this sentence will not be displayed -->
</p >
```

HTML Structure

Every Webpage must contain the above code.

- ***<!DOCTYPE html> :*** This tag is used to tells the ***HTML version.*** This currently tells that the version is ***HTML 5***.
- **<html>:** This is called ***HTML root element*** and used to wrap all the code.

An HTML Document is mainly divided into two parts:

- ***< head >* :** This contains the information about the HTML document.

Head tag contains ***metadata, title, page CSS*** etc.

➡ These pieces of information are intended for machines (browsers, robots, etc.)

- ***<body >*** : This contains everything you want to display on the Web Page. Webpage content.

➡ Information intended for humans (and machines)

```
<!DOCTYPE html>
    <html>

        <head>
            <title> Title here </title>
        </head>

        <body>
            Web page content goes here.
        </body>

    </html>
```

DOCTYPE and its Importance

Document Type Definition :

- Every HTML document must start with a **DOCTYPE** declaration. HTML5 introduces a unique and simplified **DOCTYPE**, which is:

  <!DOCTYPE html >.

- This DOCTYPE informs the browser that the document is written in HTML5, making it easier to display and interpret the content correctly.
- The DOCTYPE declaration is not an HTML tag but an instruction for the browser.
- It is mandatory and must be placed at the very top of the document, before any other HTML tag.

HTML4	**<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" >**
HTML5	**<!DOCTYPE html>**

HTML Tag

The <html> tag :

- Indicates to the browser that the file is an HTML document.
- It serves as **the root element** of the HTML document, encompassing all the content.

```
<!DOCTYPE html >
<html lang= "fr">
      <! - - Document content - - >

</html >
```

- The <html> tag can include the attribute lang="fr", which indicates that the document is written in French. This information is particularly useful for search engines like Google, as well as for screen readers used by visually impaired individuals, to enhance accessibility.

Document Header

Header <head>...</head>

Role :

- Provides essential information about the document.
- Allows browsers to understand how to display the content and helps search engines properly index the page.

Position :

- Located at the beginning of the HTML document, immediately after the **<html>** tag and before the **<body>** tag .

Header: <head>.....</head>

The header of the HTML document contains a series of essential information:

- **Document title:**
  - Defined by the **<title>...</title>** tag, it appears in the browser's tab and is crucial for SEO (search engine optimization).

- **CSS Stylesheet declaration or links:**
  - Links to CSS files for styling the document, usually included with the **<link rel="stylesheet" href="style.css">** tag.

- **JavaScript functions or links to JavaScript files:**
  - Integration of JavaScript to add interactivity, often included with the **<script src="script.js"></script>** tag.

- **Information for browsers:**
  - Instructs Internet Explorer to render the page in the most recent compatibility mode.
  <meta http-equiv= « X-UA-Compatible" content= " IE=edge " >.

Header: <meta>

Meta tags (<meta>) :

Meta tags provide additional information about the HTML document. They are located in the <head> section and are presented in the form of attribute-value pairs.

- **Encoding :**

  <meta charset= " UTF-8 " >

  - Specifies the character encoding used in the document.
- **Description :**

  <meta name="description" content="description of page" >

  - Provides a brief description of the content, useful for search engine optimization.
- **Keywords :**

  <meta name= "keywords" content= " keyword1, keyword2" >

  - List of keywords associated with the content of the page.

Header: <meta>

- **Author :**

  <meta name="author" content= " author's name" >

  - Indicates the author of the document.
- **Copyright :**

  <meta name ="copyright" content="© Ýear Author's name" >

  - Mentions the copyright associated with the content.
- **Language :**

  < meta http_equiv ="Content-languaguage" content="fr">

  - Specifies the language of the page's content.

important

Filling in these fields is important for search engine optimization (SEO) in search engines.

Character Encoding in HTML

The <meta charset> tag is used to specify the character encoding in an HTML document.

- **Syntax :**

```
<meta charset="Name_of_encoding" />
```

- **Possible values :**

  - **Utf-8:**

  - Unicode encoding that supports most characters and symbols from all languages.

  - It is the most common and recommended encoding for compatibility and universality.

  - **iso-889-1 :**

  - Supports the Latin alphabet (characters from English and other Western languages).

  - Less common today, especially with the rise of UTF-8

  - **Other encodings:**

  - Several other encodings (e.g., windows-1252, utf-16) can be used depending on specific needs.

Always use <meta charset="utf-8"/> to ensure compatibility and avoid display issues.

Header example


```
<!DOCTYPE html >
<html lang= "fr">
    <head >
        <meta charset= " utf-8 " />
        <meta name= " description " content= "Une page Web
            vraiment intéressante " />
        <meta name= " keywords " content= "course ,HTML ,IUT
            deVilletaneuse , R109 " />
        <meta name= " author " content= "Prof " />
        <title >The title of the Webpage </title >

    </head >
    <body >
        ...
    </body >
</html >
```

Header:The <Title> tag

The <title> tag :

- **Location:** Found between the **<head>** and **</head>** tags, which form the header of the HTML document.
- **Mandatory:** It is the only mandatory tag within the header. Without it, the page may lack proper identification in browsers and search engines.
- **Syntax :**

```
…
<head >
      <title> The title of my page</title>
</head >
…
```

The text placed between the <title> and </title> tags does not appear on the page itself, but only in the browser interface and search engine results.

Header: The <Title> tag(continued)

The <title> tag :

- **Function:** The content of the **<title>** tag defines the title of the web page, which appears in the browser tab as well as in search engine results. It provides users with an overview of the page's content.
- **Display :**
  - **Before :** The title consistently appeared in the window title or in the browser tab.
  - **Today** : Its use in this position is becoming less common in some modern browsers, sometimes replaced by other elements such as icons.

Document Body: <body>

The <body> tag:

- The **<body>…</body>** tags define what is called the body of the HTML document.
- Everything visible on the page is found inside the **<body>** tag.
- All visible content of the webpage, such as text, images, videos, links, and other interactive elements, must be placed between these tags.
- It is within this section that HTML5 code is used to structure and organize the content that will be displayed in the browser window. The content between the **<body>** tags is what the end user will see and interact with when visiting the page.

➡ The body of the page is therefore essential for creating the user interface and enhancing the browsing experience.

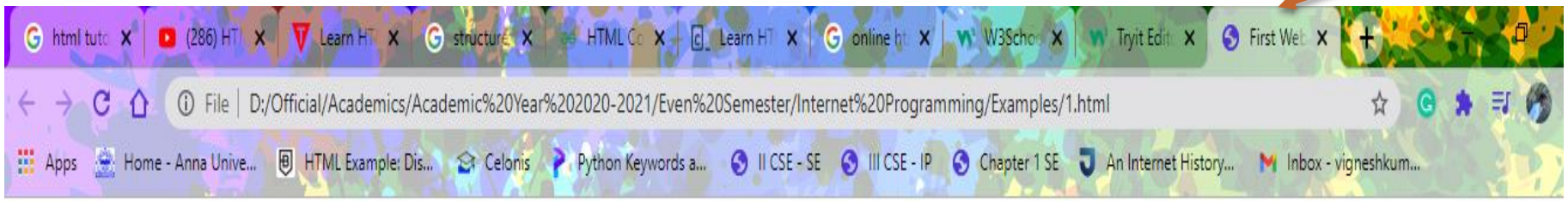
HTML –First web page

- ***Example:***

```
<html>
    <head>
        <title>
            First Web Page
        </title>
    <meta charset="UTF-8">
    </head>

    <body>
        Hello World!
    </body>
</html>
```

- ***Output:***



Exercises:

Exercise :

Q1. Is the DOCTYPE declaration mandatory? Where should it be placed? What information does it contain?

Q2. What must be done to validate HTML5 code?

Q3. Where is the character set used in the document defined?

Q4. Where is the root element of the document declared? What is its role and name?

Q5. What are the child elements of the <html> element?

Exercises:

Exercise 1:

Q1. Yes, the DOCTYPE declaration is mandatory in HTML5.
It must be placed at the very top of the HTML document, before the <html> tag. The DOCTYPE declaration tells the browser which version of HTML is being used.

Q2.

- To validate HTML5 code, you need to check your code against the HTML5 specifications. This can be done by:
- Using an HTML validator tool.
- Ensuring that all HTML elements and attributes are properly closed and nested.
- Following HTML5 syntax rules, like using lowercase tags, using the DOCTYPE declaration, and ensuring attributes have proper values.

Exercises:

- Q3. The character set used in the document is defined in the <meta> tag inside the <head> section of the HTML document. This specifies how characters are encoded. In HTML5, it is usually defined as UTF-8, like this:<meta charset="UTF-8">

- Q4. The root element of the document is declared with the <html> tag. It is the top-level element of the HTML document that encloses all the other elements. Its role is to contain the entire content of the webpage. It generally has two child elements: <head> and <body>.

- Q5. The <html> element typically has two main child elements:
- <head>: Contains meta-information about the document, such as the character set, title, and links to external resources like stylesheets.
- <body>: Contains the content of the webpage, including text, images, links, and other HTML elements that are visible on the page.

Text formatting tags in HTML5

Text formatting tags :

- Allow structuring and styling of web page content. They influence the appearance and semantics of the text, making it easier for users to read and for search engines to process.
- Help organize content into logical segments, making the text easier to read and understand.
- Enable highlighting important information, emphasizing certain words or phrases, and improving the user experience.
- Are placed inside the <body> tag because they relate to the visible content of the page.

Title Tags (<h1> to <h6>):

Title tags :

- are used to structure content into sections, making it easier to read and navigate.
- They are also important for **SEO** (Search Engine Optimization), as they help search engines understand the structure and topic of your page.

Important attributes:

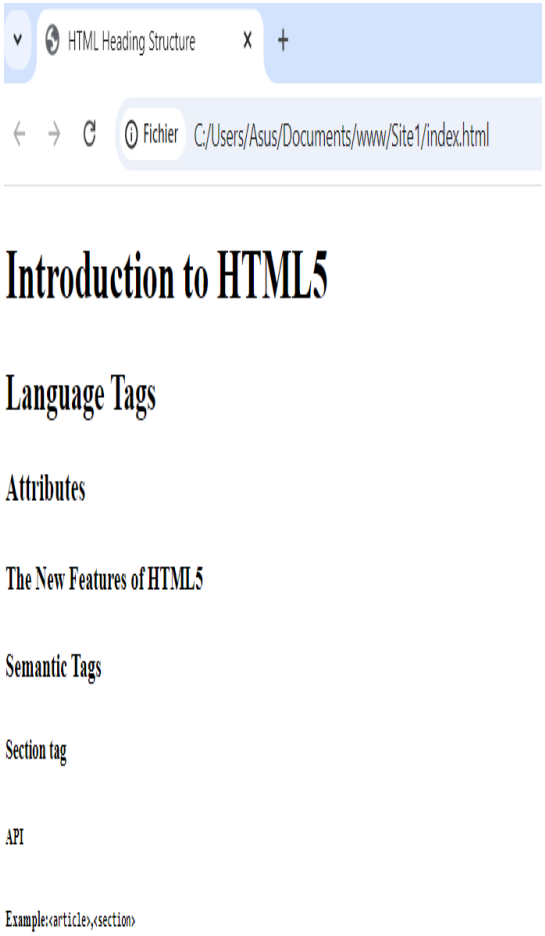
- **id**: Used to uniquely identify an element on the page and create anchor links.
- **class**: Used to apply specific CSS styles to the titles.

Note:

- Use headings in order (from <h1> to <h6>) for better hierarchy and accessibility.
- Use only one <h1> tag per page to indicate the main topic.

```
<h1>Title of level 1</h1>
<h2>Title of level 2</h2>
<h3>Title of level 3</h3>
<h4>Title of level 4</h4>
<h5>Title of level 5</h5>
<h6>Title of level 6</h6>
```

Title Tags (<h1> to <h6>):Example




```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title> HTML Heading Structure
    </title>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1 id="introduction">Introduction to HTML5</h1>
    <h2>Language Tags</h2>
    <h3>Attributes</h3>
    <h4>The New Features of HTML5</h4>
    <h4>Semantic Tags</h4>
    <h5>Section tag</h5>
    <h6>API</h6>
    <h6>Example:<code>&lt;article&gt;</code>,<code>&lt;section&gt;

    </code></h6>
    <p></p>

</body>
</html>
```

Text formatting tags in HTML5

Tag	Signification	example
	Puts the text in bold without indicating importance.	**** This is bold text. ****
	Indicates that the text is important, often in bold.	**** This is important text. ****
	Puts the text in italics without indicating emphasis.	**** This is italic text. ****
	Indicates that the text should be emphasized, often in italics.	**** This is emphasized text. ****
<u>	Underlines the text.	**<u>** This is underlined text. **</u>**
<mark>	Highlights the text to draw attention.	**<mark>** This is highlighted text. **</mark>**
<small>	Reduces the size of the text	**** This is small text. ****

Text formatting tags in HTML5

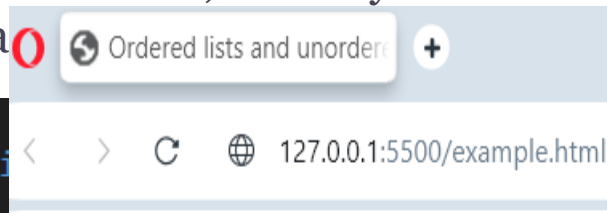
Tag	Signification	example
<del>	Indicates deleted text, usually strikethrough.	~~This is deleted text.~~
<ins>	Indicates added text, often underlined.	<ins>This is added text.</ins>
<sub>	Indicates subscript text	H₂O
<sup>	Indicates superscript text	x²
<code>	Indicates a portion of computer code	<code>console.log('Hello World');</code>
 	Line break	Text before the line break. Text after the line break.
<q>	Indicates a short quotation, often rendered in quotation marks.	<q>This is a short quote.</q>

Ordered lists and unordered lists

Ordered List :

- A list where the items are presented in a specific order, usually numbered, used when the order of items is importa

```
<ol>
    <li>Prepare the ingredients</li
    <li>cook on low heat</li>
    <li>serve hot</li>
</ol>
```



1. Prepare the ingredients
2. cook on low heat
3. serve hot

Unordered List <ul> :

- A list where the items are presented without a
usually in bullet points.

```
<ul>
    <li> Apple</li>
    <li>Banana</li>
    <li>Orange</li>
</ul>
```

- Apple
- Banana
- Orange

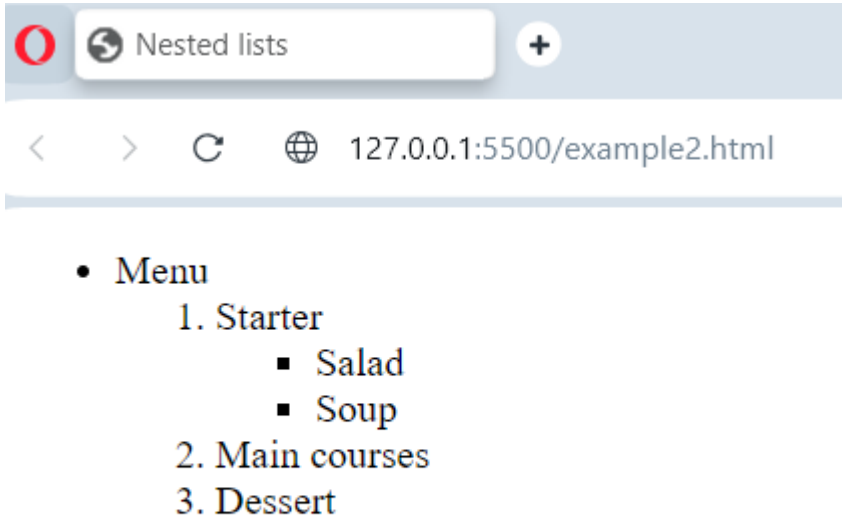
Tag <li> (list Item) :

- **The <li> tag must always be used inside <ul> or <ol> tags.**

Nested lists

- Nested lists allow for the creation of hierarchical structures within a main list. This means that a list item **(<li>)** can contain another ordered list **(<ol>)** or unordered list **(<ul>),** thereby creating sub-levels of items.

```
<ul>
    <li>Menu
        <ol>
            <li>Starter
                <ul>
                    <li>Salad </li>
                    <li>Soup </li>
                </ul>
            </li>
            <li> Main courses</li>
            <li>Dessert</li>
        </ol>
    </li>
</ul>
```



Internal and external links

Internal link:

- A link that redirects to another section or page of the same website.

To create an internal link:

- Define a target with an id: **<h2 id="section1">Section 1</h2>.**
- Create the internal link using the <a> tag with its href attribute (Hypertext Reference) containing # followed by the identifier (pointing to #id).

Example:

Go to the Section 1

Internal and external links

External Link :

- A link that redirects to a resource or page located on another domain or website.

To create an external link:

- Use the **href** attribute by specifying the full URL of the resource.
- Use **target="_blank"** (optional) to open the link in a new tab.

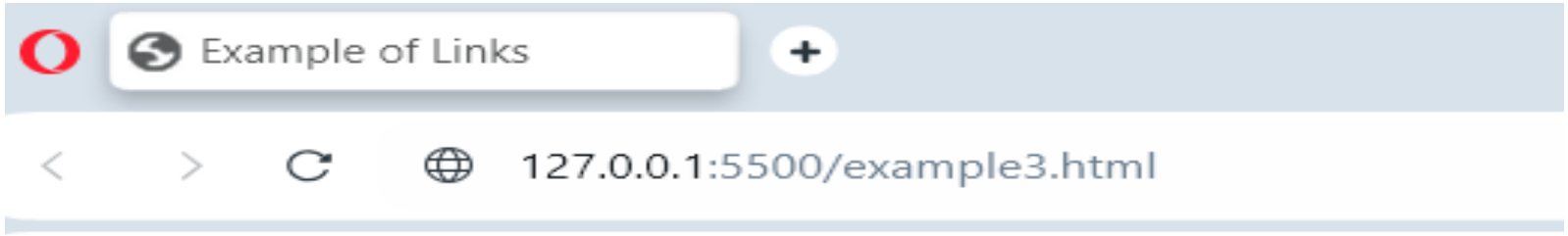
Example :

```
<a href="https://www.example.com" target="_blank">Visit Example.com</a>
```

Other types of links:

Type of links	Description	example
Download links	Allows the download of files	Download the Document PDF
Mailto Links	Opens the email client	Send an E-mail
Tel Links	Opens the dialer to make a phone call	 Call +123456789

Example:



Contact me

You can contact me via

- WebSite
- Email
- Phone

Return to the top of the page

HTML Tables

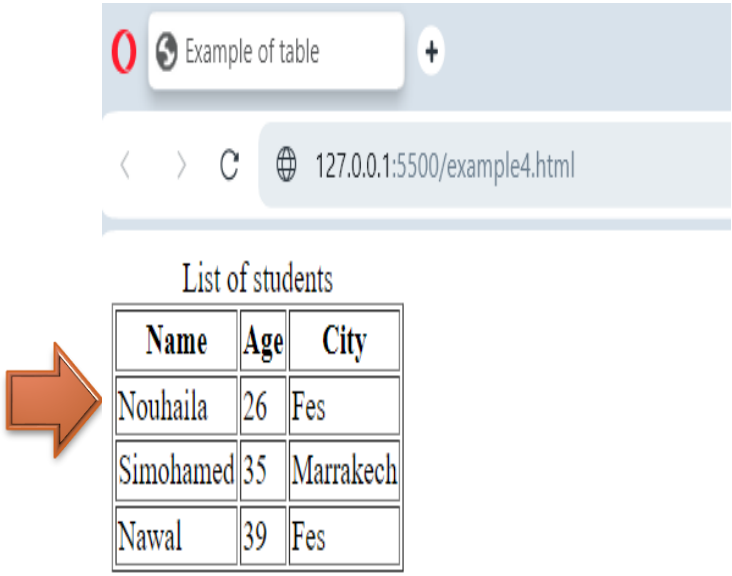
Tables :

- Thanks to the **<table> </table>**
- With the « **border** » attribute (<table border>), we define a grid.
- An HTML table consists of the following elements:
  - **<table>** : The container element for the table.
  - **<tr>** : (Table Row): Defines a row in the table.
  - **<th>** : (Table Header): Defines a header cell, often used for column titles.
  - **<td>** : (Table Data): Defines a data cell containing information.

  We can add a table title using the **<caption>** tag.

HTML Tables: EXAMPLE

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Example of table</title>
</head>
<body>
    <table border="1">
        <caption>List of students</caption>
        <tr>
            <th>Name</th>
            <th>Age</th>
            <th>City</th>
        </tr>
        <tr>
            <td>Nouhaila</td>
            <td>26</td>
            <td>Fes</td>
        </tr>
        <tr>
            <td>Simohamed</td>
            <td>35</td>
            <td>Marrakech</td>
        </tr>
        <tr>
            <td>Nawal</td>
            <td>39</td>
            <td>Fes</td>
        </tr>
    </table>
</body>
</html>
```

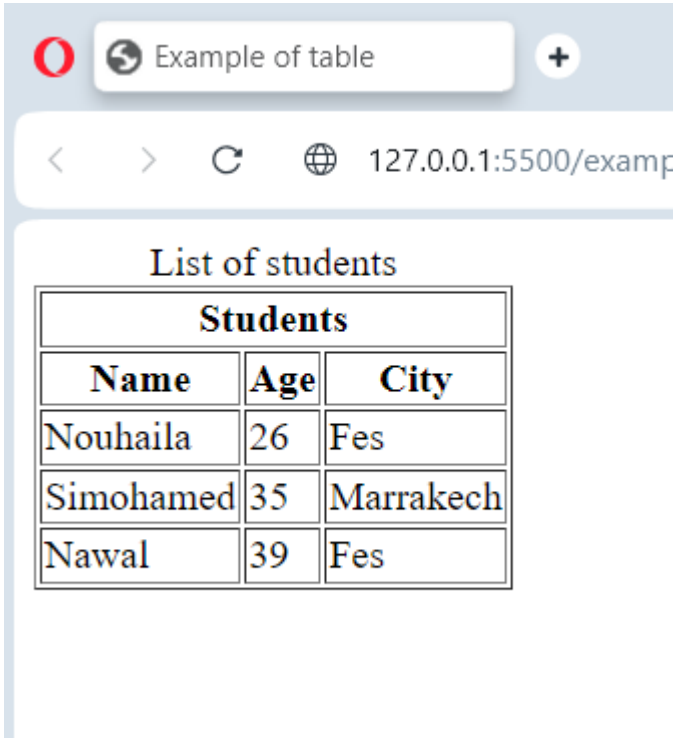


How to manipulate a table

- To delete a row, simply remove the corresponding **<tr>** element.
- To delete a column, remove the **<td>** element from each row of the column to be deleted .
- To merge cells horizontally, use the **colspan** attribute to span cells across multiple columns.
- To merge cells vertically, use the **rowspan** attribute to span cells across multiple rows.

How to manipulate a table

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Example of table</title>
</head>
<body>
    <table border="1">
        <caption>List of students</caption>
        <tr>
            <th colspan="3">Students</th> <!--Merging headers-->
        </tr>
        <tr>
            <th>Name</th>
            <th>Age</th>
            <th>City</th>
        </tr>
        <tr>
            <td>Nouhaila</td>
            <td>26</td>
            <td>Fes</td>
        </tr>
        <tr>
            <td>Simohamed</td>
            <td>35</td>
            <td>Marrakech</td>
        </tr>
        <tr>
            <td>Nawal</td>
            <td>39</td>
            <td>Fes</td>
        </tr>
    </table>
</body>
</html>
```



Forms

HTML Forms:

- Allow users to enter and submit data.
- Make web pages interactive.

How to create a Form

Main element : <form>

- A form is defined by the **<form>** element and can contain various types of input elements.

Important Attributes :

- Action : The URL to which the data is sent
- Method : The HTTP method used to send the data (GET or POST)

Example :

```
<form action="/submit" method="post">
    <!-- Contenu du formulaire ici -->
</form>
```

How to create a Form

Single-line Text Area:

- Create using the **<input>** element with the **type** attribute set to **text**.
- It allows the user to enter a single line of text.

```
<input type="text" name="pseudo" />
```

How to create a Form: Single-line Text Area

Adiitional Attributes :

- placeholder : Displays a hint text in the field before the user enters any data.

```
<input type="text" name="pseudo" placeholder="Entrez votre pseudonyme" />
```

- maxlength : Limits the number of characters that the user can enter.

```
<input type="text" name="pseudo" maxlength="20" />
```

- value : Sets a default value in the field.

```
<input type="text" name="pseudo" value="MonPseudo" />
```

- disabled : Makes the field non-editable.

```
<input type="text" name="pseudo" disabled />
```

- readonly : Allows the user to see the value but prevents them from modifying it.

```
<input type="text" name="pseudo" readonly />
```

How to create a Form: Single-line Text Area

Labels :

- A **<label>** tag is an HTML element that associates descriptive text with an input field, thus facilitating user interaction.
- To link the label to the field, it must have a for attribute with the same value as the **id** of the field.

Example :

```
<form action="/submit" method="post">
    <label for="pseudo">Pseudonyme :</label>
    <input type="text" id="pseudo" name="pseudo" />
</form>
```

How to create a Form: Single-line Text Area

Password Field:

- A password input field allows the user to enter a password securely by masking the characters typed.
- To create this type of input field, the **<input>** element is used with **type="password".**

Example :

```
<label for="password">Mot de passe :</label>
<input type="password" id="password" name="password" />
```

How to create a Form ?

Multiline Input Area:<textarea> </textarea>

- A **<textarea>** element allows the user to enter text over multiple lines.
- The **<textarea>** element is used with the **rows** and **cols** attributes.

- rows: Defines the number of visible lines in the text area.

For example: **rows="4"** creates a text area with 4 lines.

- **cols :** Defines the number of visible columns (width) in the text area.

For example: **cols="50"** creates a text area with a width of 50 characters.

Example:

```
<textarea name="zone" id="zone" rows="10" cols="50"></textarea>
```

How to create a Form ?

Enhanced Input Fields (HTML5):

- Enhanced input fields in HTML5 allow for gathering information in a more interactive and dynamic way by using specific input types.
  - **type="email": Allows the user to enter an email address.**
  - **type="url": Allows the user to enter a URL (starting with http://).**
  - **type="tel": Dedicated for entering phone numbers.**
  - **type="number": This field allows the entry of an integer.**
  - **type="range": Allows the selection of a number with a slider.**
  - **type="color": This field allows the user to input a color.**
  - **type="date": This field allows the selection of a date.**
  - **type="search": This field allows for creating a search input.**

How to create a Form ?

Option Elements:

Checkboxes:

- Checkboxes allow the user to make multiple selections. Each checkbox can be checked or unchecked independently of the others.

- To create a checkbox, use the **<input>** element with the attribute type="checkbox".

Example:

```
<input type="checkbox" name="conditions"
    value="accepte" required> J'accepte les conditions
```

How to create a Form ?

Option Elements:

Drop-down Lists:

- A drop-down list allows the user to choose an option from several available choices. It is useful for avoiding manual input and for making the form more organized.

- Use the **<select>** element to create a drop-down list and **<option>** for each option.

Example:

```
<select id="langue" name="langue"></select>
    <option value="fr">Français</option>
    <option value="en">Anglais</option>
    <option value="es">Espagnol</option>
    <option value="de">Allemand</option>
</select>
```

How to create a Form ?

Submit Button:
The submit button is created with the <input/> tag and allows users to interact with a form. It comes in four versions:

- **type="submit":** the main submit button for the form. This is the one used most often. The visitor will be directed to the page specified in the form's action attribute.
- **type="reset":** resets the form. It restores all the form fields to their default values.
- **type="image":** equivalent to the submit button, but presented as an image. Add the **src** attribute to indicate the URL of the image.
- **type="button":** a generic button that will have no effect by default. Generally, this button is handled in JavaScript to perform actions on the page.

Exercise

- The image provides instructions on how to create an HTML page that lists the modules you've studied, including details about each module and links.

- Here's a summary of the instructions:

1. Create an HTML file with a basic structure.
2. Add a title and an introduction paragraph about the modules you've studied in your program.
3. Use an unordered list (<ul>) to display the names of the modules.
4. Under each module, use an ordered list (<ol>) to list details (e.g., hours, professor).
5. Add internal links for navigation to different sections of the page and external links to resources or recommended books.
6. Create a table to summarize the module information.

My study modules

Here is a list of the main modules I have studied in my program

- Databases

1. Number of hours: 56 hours
2. professor: Dr.Amal
3. Online resources

- Web Technologies

1. Number of hours: 48 hours
2. professor: Dr. Soukaina
3. Online resources

- Python Programming

1. Number of hours: 60 hours
2. professor: Dr.Fadoua
3. Online resources

Summary table of modules

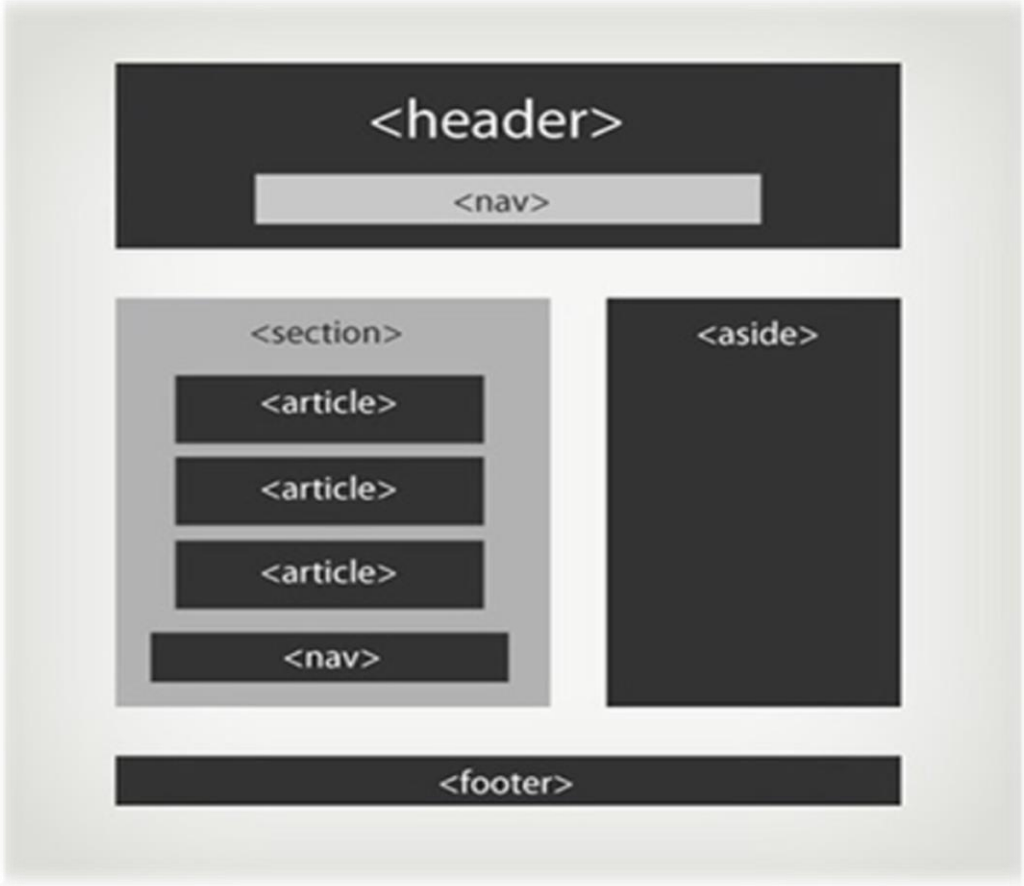
Modules and Professor

Module	Number of hours	Professor
Databases	56 Hours	Dr. Amal
Web Technologies	48 hours	Dr. Soukaina
Python Programming	60 hours	Dr.Fadoua

Return to the list of modules

Section, Article, Nav, Aside, Header, Footer Elements:

These are semantic tags in HTML5 that allow structuring the content of a web page in a more meaningful and accessible way.



<Header>

- Most websites generally have a header.
- It often contains a logo, a banner, the site's slogan, etc.
- These elements should be placed inside the <header></header> tag

```
<header>
        <!-- Placer ici le contenu de l'en-tête de votre page -- >
</header>
```

<footer>

- Unlike the header, the footer is generally located at the very bottom of the document.
- It contains information such as contact links, the author's name, legal notices, etc."

```
<footer>
        <!-- Placer ici le contenu du pied de page -- >
</header>
```

Example :

```
<footer>
    <ul>
    <li>droits auteur</li>
    <li>carte du site</li>
    <li>contactez-nous</li>
    <li>retour au début de la page</li>
    </ul>
</footer>
```

<nav>:Main navigation links

- The <nav> tag should group all the main navigation links of the website.
- For example, the main menu of the site is placed here.
- Generally, the menu is created as a bulleted list inside the tag:"

Example :

```
<nav>
    <ul>
    <li><a href="index.html">Accueil</a></li>
    <li><a href="forum.html">Forum</a></li>
    <li><a href="contact.html">Contact</a></li>
    </ul>
</nav>
```

<article>:an independent article

- The <article> tag is used to enclose a generally autonomous portion of the page.
- It represents a part of the page that could be reused on another site.
- This is the case, for example, with news items (newspaper or blog articles).
- Each article can have its own structure."

Example :

```
<article>
    <h1>Mon article</h1>
    <p>.............</p>
</article>
```

<section>:page section

- The <section> tag is used to group content based on its theme.

  A section distinguishes a logical block of content.

  It can be subdivided into multiple sections and consist of a header, a footer, and navigation.

Example :

```
<section>
    <h1>Ma section de page</h1>
    <p>.............</p>
</section>
```

```html
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Page Exemple</title>
</head>
<body>

    <header>
        <h1>Bienvenue sur notre site</h1>
        <nav>
            <ul>
                <li><a href="#about">À propos</a></li>
                <li><a href="#services">Services</a></li>
                <li><a href="#contact">Contact</a></li>
            </ul>
        </nav>
    </header>

    <section id="about">
        <header>
            <h2>À propos de nous</h2>
        </header>
        <p>Nous sommes une entreprise dédiée à la satisfaction de nos clients.</p>
    </section>

    <section id="services">
        <header>
            <h2>Nos Services</h2>
        </header>
        <p>Nous offrons une variété de services pour répondre à vos besoins.</p>
    </section>

    <footer>
        <p>&copy; 2024 Mon Entreprise. Tous droits réservés.</p>
    </footer>

</body>
</html>
```

Example

Bienvenue sur notre site

- À propos
- Services
- Contact

À propos de nous

Nous sommes une entreprise dédiée à la satisfaction de nos clients.

Nos Services

Nous offrons une variété de services pour répondre à vos besoins.

<aside> Additional information

- The <aside> tag is designed to contain complementary information related to the document being viewed.
- These pieces of information are generally placed on the side (though this is not mandatory).
- There can be several <aside> blocks on a page."

```
<aside>
    <!-- Placer ici des informations complémentaires -- >
</aside>
```

Validation and compability

Validation :

- What is it? It is the process of verifying that the HTML, CSS, and JavaScript code adheres to web standards..

Cross-browser compatibility :

- Different browsers may display the same code differently.

  → **Test your site on multiple browsers (Chrome, Firefox, Safari, Edge) and mobile versions.**

Validation and compability

W3C Validation Tools:

The W3C (World Wide Web Consortium) provides two essential tools to ensure the quality of your code:

1. **HTML Validator :**
   - http://validator.w3.org : Checks that your HTML code complies with current standards.
2. **CSS Validator :**
   - http://jigsaw.w3.org/css-validator : Ensures that your CSS code conforms to web standards.

→ **You should always strive to have valid code.**