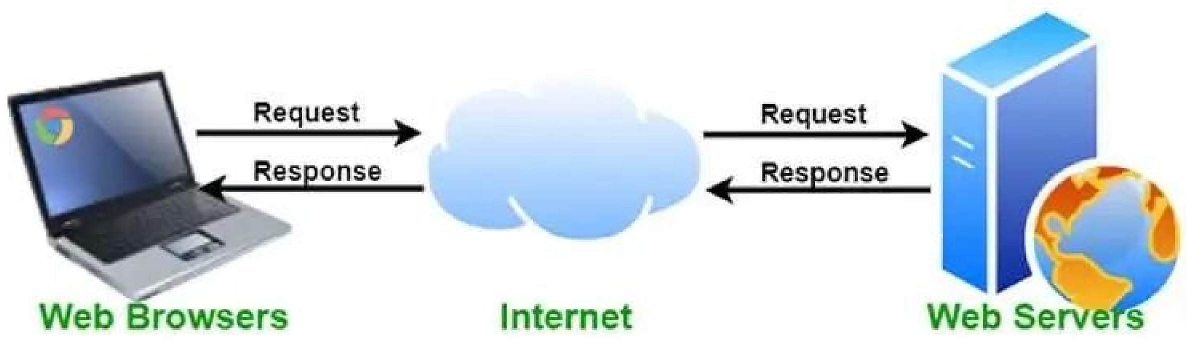
1- On Web Server-

What Does a Web Server Do?

So what does a web server do? It’s software or hardware (or both together) that stores and delivers content to a web browser at a basic level. The servers communicate with browsers using Hypertext Transfer Protocol (HTTP). Web servers can also support SMTP (Simple Mail Transfer Protocol) and FTP (File Transfer Protocol).

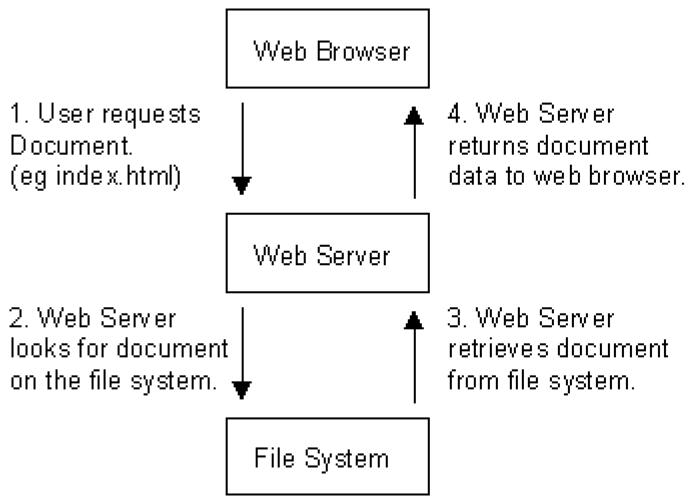
Web servers are also used for hosting websites and data for web applications. They can host single websites and multiple websites using virtualization.



How Does a Web Server Work Internally?

When the webserver receives a request for content for a web page, such as
http://www.Webcompare.com/index.html
and maps that URL to a local file on the host server.
In this case, the file

index.html
is somewhere on the host file system. The server then loads this file from disk and serves it out across the network to the user's Web browser. This entire exchange is mediated by the browser and server talking to each other using HTTP. This workflow is shown in the figure below.


Web Browser
1. User requests Document. (eg index.html)
4. Web Server returns document data to web browser.
Web Server
2. Web Server looks for document on the file system.
3. Web Server retrieves document from file system.
File System

This allows the serving of static content such as HyperText Markup Language (HTML) and images files to a Web browser was the initial concept behind what we now call the World Wide Web. The beauty of its simplicity is that it has led to much more complex information exchanges being possible between browsers and Web servers.

Perhaps the most important expansion on this was the concept of dynamic content (i.e., Web pages created in response to a user's input, whether directly or indirectly). The oldest and most used standard for doing this is Common Gateway Interface (CGI). This is a pretty meaningless name, but it basically defines how a Web server should run programs locally and transmit their output through the Web server to the user's Web browser that is requesting the dynamic content.
For all intents and purposes the user's Web browser never really has to know that the content is dynamic because CGI is basically a Web server extension protocol. The figure below shows what happens when a browser requests a page dynamically generated from a CGI program.
The second important advance, and the one that makes e-commerce possible, was the introduction of HyperText Transmission Protocol, Secure (HTTPS). This protocol allows secure communication to go on between the browser and Web server.
In a nutshell, this means that it is safe for user and server to transmit sensitive data to each another across what might be considered an insecure network. What happens when the data arrives at either end is another matter, however, and should not be ignored. We will discuss this

a bit later.

The simplicity of the above arrangements is deceptive, and underestimating its complexities often leads to bad decisions being made about the design of a Web-hosting infrastructure. It is too easy to focus on the design of the Web pages themselves and the technologies used to create dynamic content, such as Java, Javascript, Perl, C/C++, and ASP, and to subsequently miss the fact that each of these technologies can be aided, or hindered, by the platform on which they are to be run — the Web server itself.

In other words, explaining how a Web server works involve discussing more than just how a Web server serves documents. We will go over the following topics in our quest to finding out what and how a modern Web server goes about doing its activities.

We will start by explaining the low-level details underlying what a Web server must do. Then, we will discuss the issues surrounding the use of a Web server and how it fits within the scope of other infrastructural elements of the Internet. We will then close with a discussion of the relationship that applications have with Web servers.

2-HTML element-

An HTML element is a piece of content on a web page. It is the basic building block of a website and is defined by a pair of tags.

For example, the <p> tag is used to define a paragraph element, and the <img> tag is used to define an image element.

HTML elements can have attributes, which are extra pieces of information that provide additional details about the element. For example, the src attribute of an <img> element specifies the URL of the image to display.

HTML elements can also have content, which is the text or other elements that are between the opening and closing tags. For example, the content of an <p> element would be the text of the paragraph.

Here is an example of an HTML element:

```
<p>This is a paragraph element.</p>
```

In this example, the <p> element defines a paragraph, and the text "This is a paragraph element." is the content of the element.

3- Input Elements:

Input elements refer to the various form fields and controls that users can interact with to enter and submit data in a form. Examples of input elements include text fields, radio buttons, checkboxes, and dropdown menus. These elements are typically used in web forms to collect information from users, such as their names, email address, and preferences.

4-Audio and Video Tag:

The <audio> and <video> HTML tags are used to add audio and video content to a web page. These tags allow developers to embed audio and video files, such as MP3s and MP4s, on their websites. They also provide a range of attributes and options that can be used to customize the playback of the audio or video, such as controls for play, pause, volume, and seeking.

5-Flex Box (CSS)

The CSS Flexible Box Layout, commonly known as flexbox, is a layout mode that provides a simple and flexible way to lay out items in a one-dimensional space. It allows elements to be aligned and distributed within a container in various ways and is designed to support direction-agnostic layouts and responsive resizing of elements. Flexbox is often used for page layout, as well as for arranging items within a container, such as a navigation menu or a grid of cards.

Some of the CSS properties that are used with flexbox include:

- display: flex: This property is used to turn an element into a flex container, which allows its child elements to be laid out using flexbox.
- flex-direction: This property determines the direction in which flex items are laid out within the flex container. It can have the values row, row-reverse, column, and column-reverse.
- justify-content: This property aligns flex items along the main axis of the flex container. It can have the values flex-start, flex-end, center, space-between, space-around, and space-evenly.
- align-items: This property aligns flex items along the cross-axis of the flex container. It can have the values flex-start, flex-end, center, baseline, and stretch.
- flex-wrap: This property determines whether flex items are allowed to wrap onto multiple lines, or whether they should be forced onto a single line. It can have the values nowrap, wrap, and wrap-reverse.

6-Positioning (CSS)

In CSS, the positioning of an element refers to how the element is positioned within the document and relative to other elements on the page. There are several positioning schemes available in CSS, each with its own set of properties and behavior. These include:

- static: This is the default positioning scheme, in which elements are positioned according to the normal flow of the document.
- relative: In this positioning scheme, elements are positioned relative to their normal position in the document flow. This allows elements to be shifted from their default position, but they still take up space in the layout and can affect the positioning of other elements.
- absolute: In this positioning scheme, elements are positioned relative to their closest positioned ancestor (or the initial containing block if no ancestor is positioned), and they are removed from the normal document flow. This allows elements to be positioned precisely, but they do not take up space and do not affect the positioning of other elements.
- fixed: In this positioning scheme, elements are positioned relative to the initial containing block, and they are fixed in place even when the page is scrolled. This allows elements to be positioned precisely and remain visible, but they do not take up space and do not affect the positioning of other elements.

Each positioning scheme has its own set of CSS properties that can be used to control the position and behavior of elements. For example, the top, right, bottom, and left properties are used to specify the distance of an element from the corresponding edge of its containing block when using the absolute or fixed positioning schemes.

7-CSS Box Model (Padding, Margin, Border)

The CSS box model is a fundamental concept in web design and layout. It describes the rectangular boxes that are generated for elements in the document and the way that these boxes are spaced and positioned on the page.

The box model consists of the following elements:

- Content: This is the innermost part of the box, and it contains the element's actual content, such as text or images.
- Padding: This is the space around the content, and it separates the content from the border. The padding can be customized using the padding property in CSS.
- Border: This is the line around the padding, and it separates the padding from the margin. The border can be customized using the border property in CSS.
- Margin: This is the space outside the border, and it separates the border from other elements on the page. The margin can be customized using the margin property in CSS.

Together, these elements form a rectangular box around the element's content, and they determine the element's size and position on the page. The size and position of the box can be controlled using a combination of these properties, as well as other layout-related CSS properties.

8-Media Query

A media query is a CSS feature that allows developers to apply different styles to a document based on the characteristics of the device displaying it. Media queries use the @media rule to specify the conditions under which a certain set of styles should be applied, such as the width of the viewport, the device's orientation, or the resolution of the screen.
For example, a media query might be used to apply a different set of styles to a web page when it is viewed on a small screen, such as a smartphone, compared to when it is viewed on a larger screen, such as a desktop computer. This allows the page to be optimized for the specific capabilities and constraints of each device, and can improve the user experience by making the page more readable and user-friendly on different devices.

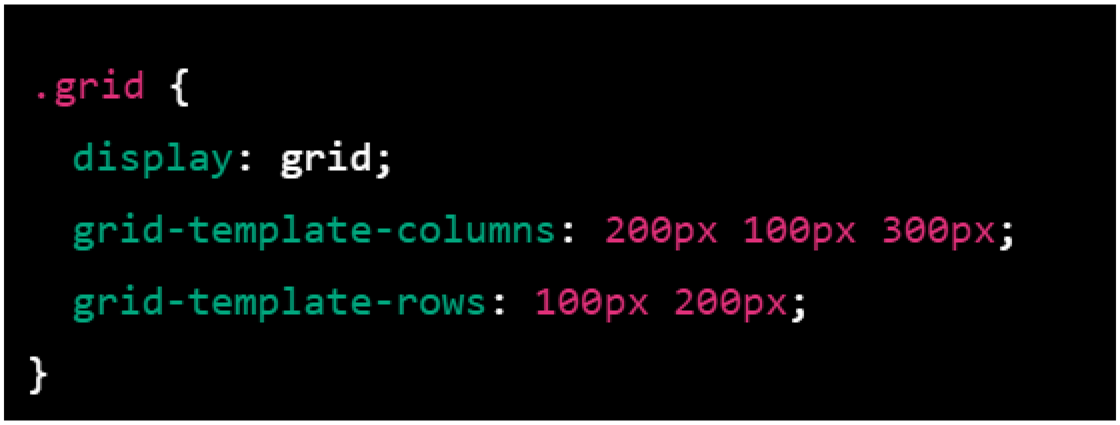
Here is an example of a media query that applies a different set of styles to a page when the viewport is less than 600 pixels wide:

```
@media only screen and (max-width: 600px) { /* Styles go here */ }
```

This media query uses the @media rule to specify that the styles inside it should only be applied when the screen is being used, and when the width of the viewport is less than 600 pixels. This allows the developer to define a separate set of styles that will be applied when the page is viewed on a small screen, such as a smartphone.

9-CSS Grid

The CSS Grid Layout is a two-dimensional grid-based layout system that allows developers to create complex and responsive web page layouts easily and efficiently. It provides a set of CSS classes and properties that can be used to define the structure of a grid, and to control the size and position of the elements within the grid.
The grid layout is based on a set of horizontal and vertical grid lines that divide the grid into a series of rows and columns. Elements in the grid can be placed onto these grid lines, and can span multiple rows and columns. The grid layout also allows elements to be aligned and justified within the grid, and to be resized and rearranged automatically as the viewport changes.



```
.grid {
  display: grid;
  grid-template-columns: 200px 100px 300px;
  grid-template-rows: 100px 200px;
}
```

In this example, the .grid class defines a grid container with three columns and two rows. The width of the columns and the height of the rows are specified using the grid-template-columns and grid-template-rows properties, respectively. Elements can then be placed onto the grid using the grid-column and grid-row properties.

Reference

Properties

- display
- grid-template-columns
- grid-template-rows
- grid-template-areas
- grid-template
- grid-auto-columns
- grid-auto-rows
- grid-auto-flow
- grid
- grid-row-start
- grid-column-start
- grid-row-end
- grid-column-end
- grid-row
- grid-column
- grid-area
- row-gap
- column-gap

- gap
- masonry-auto-flow
- Experimental
-
- align-tracks
- Experimental
-
- justify-tracks
- Experimental
-

Functions

- repeat()
- minmax()
- fit-content()

Data types

- <flex>