WEEK1-HTML

Exercise1.1

What is the main functionality of the browser?

The main functionality of the browsers is to show the web resources which we are chosen, by requesting it from the server side and showing it with in the browser window. Basically the resource is a HTML document, Image, PDF or any other. The location of this resource is indicated by the client using Universal Resource Identifier.

Diagram

Description automatically generated

The way the browser stops and shows HTML files is indicated within the HTML and CSS specifications, and these specifications are maintained by the World Wide Web Consortium, This is the standard organization for web’s.

For a long time the browsers are established to only a part of the specifications and deployed their own extensions and this cause the serious issue for the web authors.

The common user interface elements are

* Address bar which is for inserting a URI
* Forward and backward buttons
* Bookmarking options
* Refresh and stop buttons which are used for the refreshing or stopping the loading of current documents
* Home button which takes you to the home page

Basically in the HTML5 specifications the author does not define UI elements that a browser must need, but they lists the some of the common elements which are the address bar, tool bar and status bar.

High Level Components of a browser

The high level components of the browser are

**User Interface:** In the user interface it includes the address bar, bookmarking menu, forward button, backward button, here in the user interface every part of the browser display expects the window, where you can see the requested page.

**Browser engine:** The browser engine acts as a bridge between the user interface and the rendering engine.

**Rendering engine:** It is used for the displaying the requested content

**Networking:** It is used for the network calls such as HTTP requested by using different platforms.

**UI backend:** This is used for drawing the basic applications like combo borus and windows. Undernet uses operating system user interface method

**JavaScript interpreter:** It is utilized to phase and execute the JavaScript code

**Data Storage:** This is often a persistence layer, the browser may got to spare all sorts of information locally, such as cookies.

Here the chrome run multiple instances of the rendering engine. Here each tab runs in a separate process.

Rendering engine and its use

It is the software that draws a text and image on the screen. A rendering engine is a software that draws text and pictures on the screen. The engine draws the contents from the document and groups it properly based on the given style declarations. The main core of the rendering engines are Edge HTML, WebKit.

There are different rendering engines for different browsers such as internet explorer uses trident, Firefox uses gecko, safari uses webKit.

WebKit is a open source rendering engine, it is basically started as an engine of Linux platform and later it was modified by the apple for the support of Mac and Windows.

The rendering engine will start getting the contents of the requested documents from the networking layer. This will usually be done in 8k chunks. After this the below block diagram is the basic flow of the rendering engine.

Graphical user interface, diagram

Description automatically generated

parsers (HTML, CSS, etc)

Parsing means and changing over a program into an internal format that a run time environment can really run.

parse - analysing a string of symbols, according to the rules of a formal grammar, generally breaking it down into a data structure.

HTML - hypertext markup language, the language used to describe web page content.

So, here it means reading through a body of HTML, and recognizing its content into its components… tags such as <p> or <head>, the content of its elements, etc.

The CSS parser takes the bytes and converts over then into characters at that point tokens, nodes and finally they are connected to CSSOM

Script Processor

The Script Processors uses the script cache to recompile the script for each appearing document. To perform the performance, ensure the script cache is sized while using a script processor in production.

Tree construction

The render tree was the combination of DOM and CSSOM trees which are build based on the HTML and CSS inputs, because both of them are the independent objects and different aspects of the document in which one is defines the content and the other is defines the style rules.

So by merging both DOM and CSSOM we get the render tree. So the process to merge the DOM and CSSOM.

Diagram

Description automatically generated

So, from the above diagram here while we combine DOM and CSSOM and then here we will create a green circle node as a render tree. Here the P will be the part of the render tree, and here the spam will not include in the render tree since there CSSOM related. Other common DOM nodes that doesn’t have place in render tree are head, meta and link tags since there’s nothing obvious data to show their to appear.

Order of script processing

* Before business rules with an order less than 1000
* Before engines: it means it will run further your workflows and the approval
* Before business rule with an order greater than or equal to 1000
* Database operation like insert, update or delete any crude operation if we are doing then that operation on the fourth stage it will run
* After business rules with an order less than 1000
* After engines: Basically, after rule whatever you have written then it will execute over like a workflow and approval again
* E mail notification, so if you can figure any email notification then it will run on the seventh position
* After business rules with an order greater than or equal to 1000
* So, in last it will run after business rule which is having order greater than and equal to 1000

So this is the common setup defined by service now execution order.

Layout and Painting

**Layout:** The layout will be in charge to calculate the positions and measurement of each node on the screen like if you rotate your phone or otherwise if you need to resize your browser the layout piece will be executed

**Paint:** Finally we know that which nodes are obvious and their computed styles and geometry. We are able to pass the information to the final step. Which converts each node with in the render tree in to actual pixels on the screen and this is know as painting

In the painting stage the render tree is traversed and the renders “paint()” method is called to display content on the screen. Painting uses the UI infrastructure component. The stacking order of a block render is

* Background colour
* Background image
* Border
* Children
* outline