Table of Contents

Objectives 1
- General Objectives 1
- Specific Objectives 1

Introduction 2
Application 3
Literature survey 3
Existing System 4
Methodology 4
1. The input byte stream: 4
2. Input stream preprocessor: 5
3. Tokenization: 5
4. Tree construction: 5
5. Layout Calculation 6
6. Paint: 6

Implementation 7
- Block Diagram 7

Results & Demonstrations 8
Problems Faced and solutions 10
Limitations and future enhancements 11
Conclusion and recommendations 12
References 13

Objectives

General Objectives

1. To get acquainted with the fundamentals of object-oriented programming paradigm.
2. To understand and implement object-oriented concepts like polymorphism, inheritance, encapsulation and abstraction at its core.
3. To gain experience with coding in C++ to build applications as per necessity.
4. To develop problem solving skills for real world problems with the help of code.

Specific Objectives

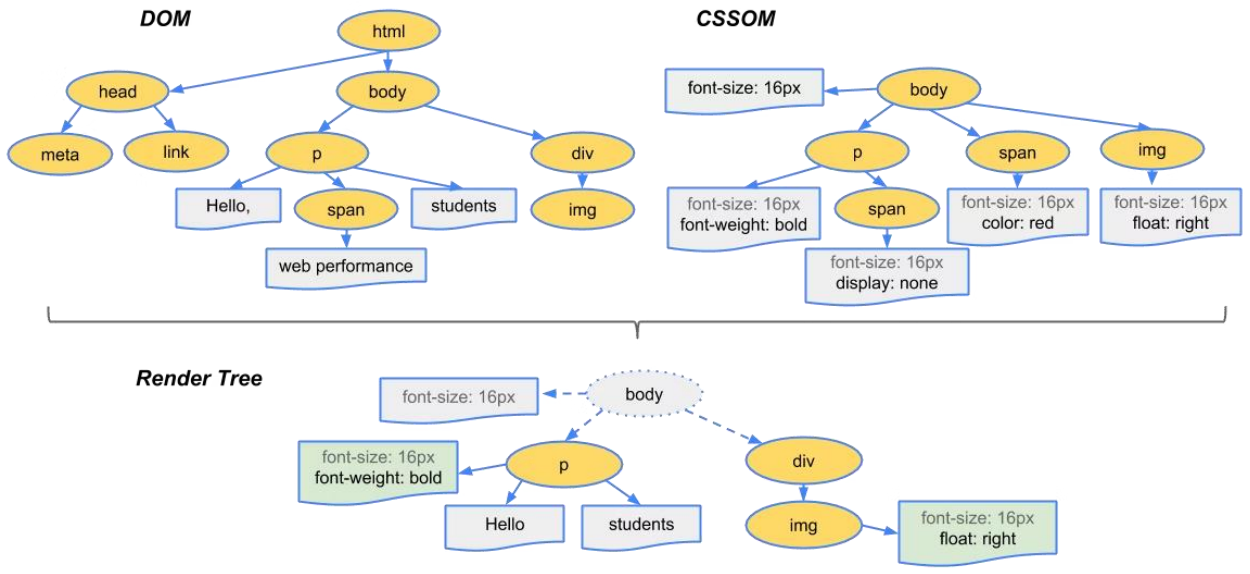
1. To get acquainted with the concept of parsing, tokenization and syntax tree construction.
2. To create the Document Object Model (DOM) and render the layout accordingly.
3. To implement basic styling features of CSS using CSS Object Model (CSSOM).
4. To get to know the basics of a GUI application like painting, rendering and updating the window.
5. Be familiar with calculating the required height and width of various tags available in the Standard HTML.

Introduction

“HTMLER” is a group project completed by the undergraduate students of Pulchowk Campus. In this project, we have created a basic prototype of HyperText Markup Language (HTML) parser, basically used to render HTML file with Cascading Style Sheets (CSS) in the web browser. HTML is simply a standard markup language for displaying our desired contents in an organized form in a web browser and parsing, in general, means breaking down something into its constituent parts and components particularly for the study and analysis of those individual parts. An example of parse is to break down a sentence to explain each element to someone. So, HTML parser extracts each and every element/tag of the entire HTML document and renders and displays it exactly as the webpage, as the browser does.

The main objective of our project is not to optimize the processing of our HTML file since all the browsers do it perfectly. Rather, it is just our curiosity to learn and implement how browsers do all this functionality to process the HTML tags and attributes to provide the user data and information flawlessly. And finally, we aspire to implement our proposition into action using object-oriented programming to make the best use of inheritance, encapsulation, polymorphism concepts.

Graphical Representation of DOM, CSSOM and Render tree:



DOM
html
head
body
meta
link
p
div
Hello,
span
students
img
web performance
CSSOM
font-size: 16px
body
p
span
img
font-size: 16px font-weight: bold
span
font-size: 16px color: red
font-size: 16px float: right
font-size: 16px display: none
Render Tree
font-size: 16px
body
font-size: 16px font-weight: bold
p
div
Hello
students
img
font-size: 16px float: right

Application

HTMLer is simply an implementation of HTML & CSS tags and stylings in limited scope which the browser performs in the backend to render and display different sorts of websites. The scope of this project is to display the equivalent webpage by rendering HTML, CSS tags and attributes which we have implemented in our project. This project aims to those who have keen interest in knowing the basic mechanism at the ground level of how HTML and CSS, functions and processes to show the required webpage to the user. Since websites have become common in our day to day life to solve minor to complex tasks in this era, this project can be useful to make a simply designed web page and give some information through posts, news and blogs.

Literature survey

Since this project was about utilizing and understanding the concept of Object Oriented Programming (OOP) there are many different books considered best out there among them the one we referred to get the clear concept of OOP paradigm was “A Tour of C++” by ”Bjarne Stroustrup”. For implementing the graphics in our project we have used SDL and for this different e-books were referred and different tutorials were viewed.

Similarly the parsing part was quite challenging and on top of it interesting too. Various books and online documentation were consulted for this purpose. The official documentation of HTML and CSS proved to be very useful for clear understanding of the concept and utilizing it.

Existing System

As the main objective of our project was not to optimize the processing of our HTML file, since all the browsers do it perfectly, all the browser engines can be regarded as the existing system for our project.

Methodology

Research methodology refers to the various sequential steps to adopt by a researcher in studying a problem with certain objectives in view. In other words researcher methodology refers to the various methods of practices applied by the researcher in the entire aspect of the study. The main methods that will be used for the study preparing this project are as follows:

1. The input byte stream:

The stream of code points that will be the input for the tokenization stage will be initially seen by the user agent as a stream of byte typically coming from a network or a from a local file system. The bytes encode the actual characters as per a particular character encoding, which the user agent uses to decode the bytes into characters.

Given a character encoding, the bytes in the input byte stream must be converted to characters for using them with the tokenizer as its input stream, bypassing the input byte stream and character encoding to decode.

When the HTML parser is decoding an input byte stream, it uses a character encoding and a confidence that is either tentative, certain, or irrelevant. The encoding used, and the type of confidence in that encoding is employed during the parsing to determine whether to change the encoding. If no encoding is necessary, e.g. because the parser is operating on a Unicode stream and doesn’t have to use a character encoding at all, then the confidence is irrelevant.

2. Input stream preprocessor:

The input stream is made of the characters pushed into it as the input byte stream is decoded or from the various APIs that directly manipulate the input stream. Before the tokenization stage, the newlines are normalized in the input stream. Initially, the next input character is the first character in the input that is yet to be consumed and the current input character is the last character to have been consumed. The insertion point is the position where content inserted using () is actually inserted. The insertion point is not an absolute offset into the input stream rather it is relative to the position of the character immediately after it. Initially, the insertion point is undefined.

3. Tokenization:

Implementations are expected to act as if they are using the following state machine to tokenize HTML. The state machine is expected to start in a data state. Most states take a single character, which either switches the state machine to a new state to re-consume the current input character or switches it to a new state to consume the next character. Some states have more complicated behavior and can take in several characters before switching to another state. In some cases, the tokenizer state is also affected by the tree construction stage.

The output generated in this step is either a series of zero or more of the following tokens: DOCTYPE, start tag, end tag, comment, character, end-of-file. Also creating and emitting tokens are two completely different concepts. When a token is emitted, it must immediately be attended by the tree construction stage. The tree construction stage can affect the state of the tokenization stage and is even allowed to insert additional characters into the stream.

4. Tree construction:

The sequence of tokens from the tokenization state form the input for the Tree construction stage. Once the parser is created, the tree construction stage is associated with the Document Object Model (DOM). The output of this stage consists of dynamically modifying or extending that document's DOM tree. As each token is dispatched from the tokenizer the user agent is expected to follow a certain algorithm in order to deal with them.

5. Layout Calculation

Layout is the process by which the width, height, and location of all the nodes in the render tree are determined, plus the determination of the size and position of each object on the page. render

The render tree identified which nodes are displayed (even if invisible) along with their computed styles, but not the dimensions or location of each node. To determine the exact size and location of each object, the browser starts at the root of the render tree and traverses it.

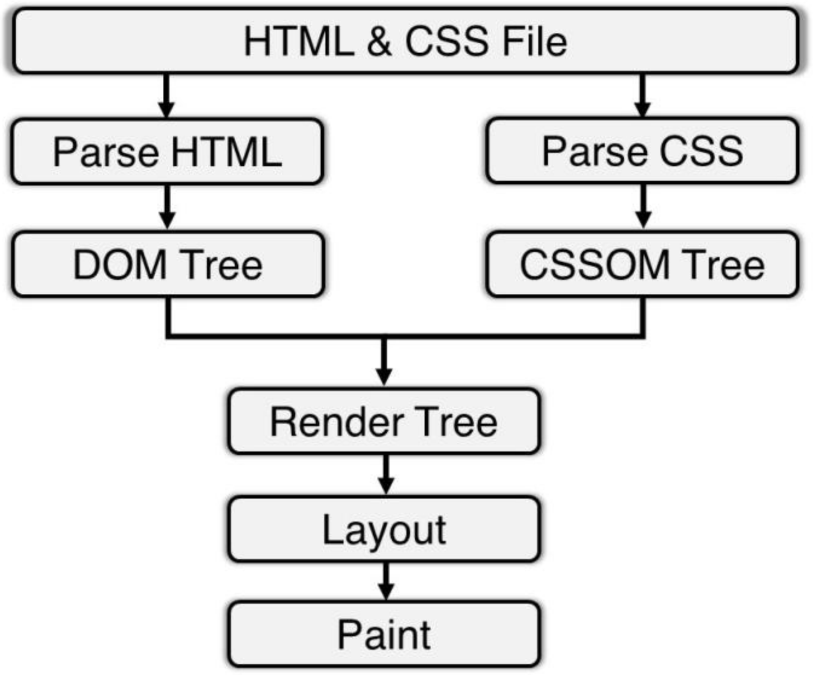
6. Paint:

The last step in the rendering webpage is painting the individual nodes to the screen. In the painting or rasterization phase, the program converts each box calculated in the layout phase to actual pixels on the screen. Painting involves drawing every visual part of an element to the screen, including text, colors, borders, shadows, and replaced elements like buttons and images.

Implementation

As this project is completely dependent on C++ language, we tried to cover almost each and every concepts of object oriented programming like encapsulation, abstraction, inheritance, polymorphism, dynamic binding and so on. Not only this, we also used a cross-platform software development library named SDL(Simple DirectMedia Layer) which renders the graphics which are necessary to parse the HTML and CSS contents.

Block Diagram

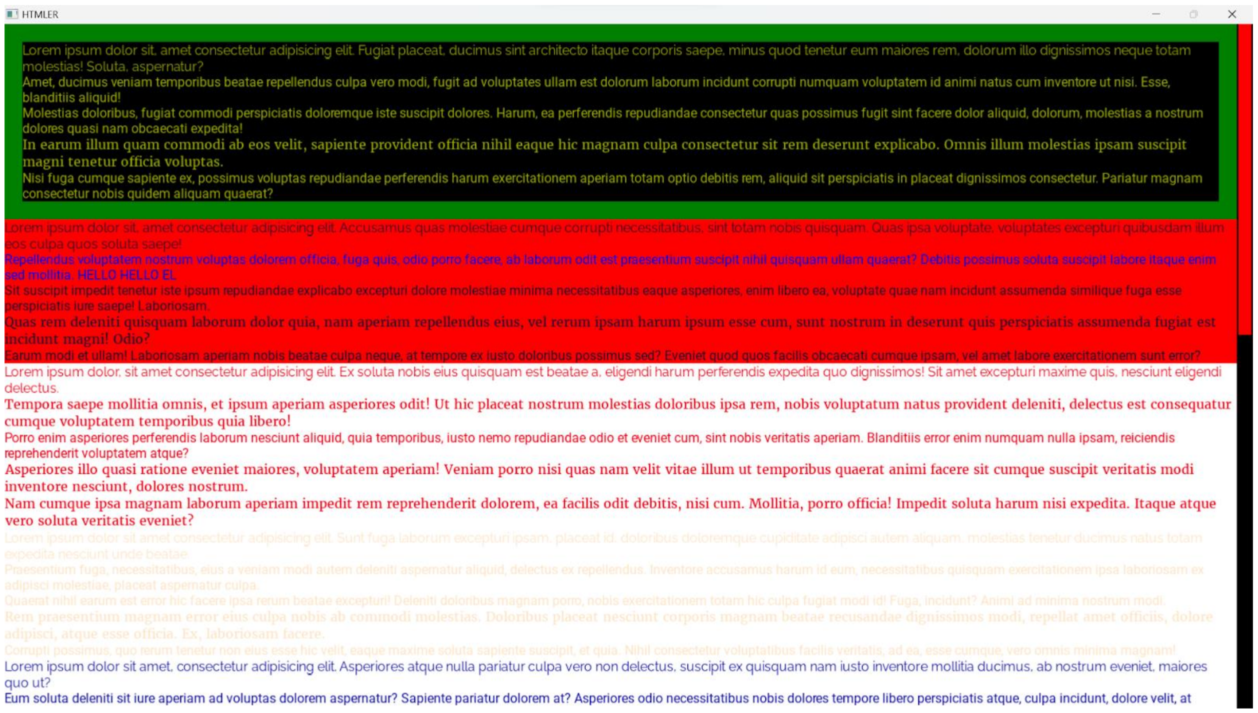


Results & Demonstrations

After the completion of this project, the main objective of this project was fulfilled i.e. HTML and CSS files were successfully parsed and rendered.

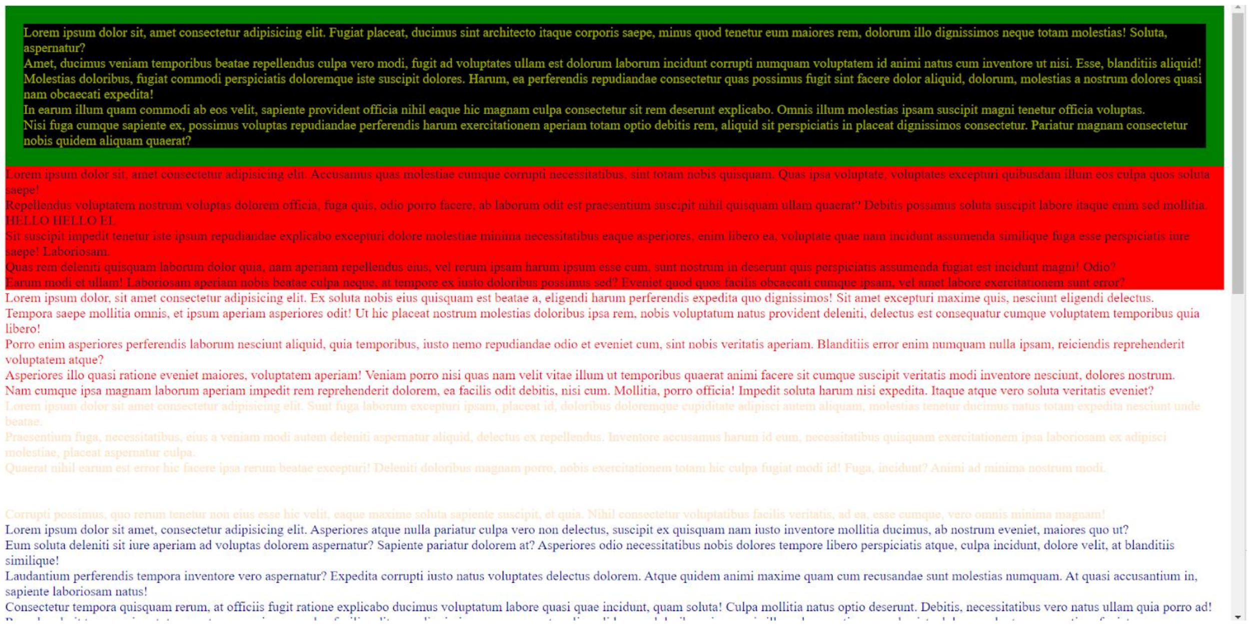
Let us have a look at the output of our project and that rendered by the browser:

Rendered in our Project



Lorem ipsum dolor sit. amet consectetur adipisicing elit. Fugiat placeat. ducimus sint architecto itaque corporis saepe. minus quod tenetur eum maiores rem. dolorum illo dignissimos neque totam molestias! Soluta. aspernatur?
Amet, ducimus veniam temporibus beatae repellendus culpa vero modi, fugit ad voluptates ullam est dolorum laborum incidunt corrupti numquam voluptatem id animi natus cum inventore ut nisi. Esse, blanditiis aliquid!
Molestias doloribus, fugiat commodi perspiciatis doloremque iste suscipit dolores. Harum, ea perferendis repudiandae consectetur quas possimus fugit sint facere dolor aliquid, dolorum, molestias a nostrum dolores quasi nam obcaecati expedita!
In earum illum quam commodi ab eos velit, sapiente provident officia nihil eaque hic magnam culpa consectetur sit rem deserunt explicabo. Omnis illum molestias ipsam suscipit magni tenetur officia voluptas.
Nisi fuga cumque sapiente ex, possimus voluptas repudiandae perferendis harum exercitationem aperiam totam optio debitis rem, aliquid sit perspiciatis in placeat dignissimos consectetur. Pariatur magnam consectetur nobis quidem aliquam quaerat?
Lorem ipsum dolor sit. amet consectetur adipisicing elit. Accusamus quas molestiae cumque corrupti necessitatibus. sint totam nobis quisquam. Quas ipsa voluptate. voluptates excepturi quibusdam illum eos culpa quos soluta saepe!
Repellendus voluptatem nostrum voluptas dolorem officia, fuga quis, odio porro facere, ab laborum odit est praesentium suscipit nihil quisquam ullam quaerat? Debitis possimus soluta suscipit labore itaque enim sed mollitia. HELLO HELLO EL
Sit suscipit impedit tenetur iste ipsum repudiandae explicabo excepturi dolore molestiae minima necessitatibus eaque asperiores, enim libero ea, voluptate quae nam incidunt assumenda similique fuga esse perspiciatis iure saepe! Laboriosam.
Quas rem deleniti quisquam laborum dolor quia, nam aperiam repellendus eius, vel rerum ipsam harum ipsum esse cum, sunt nostrum in deserunt quis perspiciatis assumenda fugiat est incidunt magni! Odio?
Earum modi et ullam! Laboriosam aperiam nobis beatae culpa neque, at tempore ex iusto doloribus possimus sed? Eveniet quod quos facilis obcaecati cumque ipsam, vel amet labore exercitationem sunt error?
Lorem ipsum dolor. sit amet consectetur adipisicing elit. Ex soluta nobis eius quisquam est beatae a. eligendi harum perferendis expedita quo dignissimos! Sit amet excepturi maxime quis. nesciunt eligendi delectus.
Tempora saepe mollitia omnis, et ipsum aperiam asperiores odit! Ut hic placeat nostrum molestias doloribus ipsa rem, nobis voluptatum natus provident deleniti, delectus est consequatur cumque voluptatem temporibus quia libero!
Porro enim asperiores perferendis laborum nesciunt aliquid, quia temporibus, iusto nemo repudiandae odio et eveniet cum, sint nobis veritatis aperiam. Blanditiis error enim numquam nulla ipsam, reiciendis reprehenderit voluptatem atque?
Asperiores illo quasi ratione eveniet maiores, voluptatem aperiam! Veniam porro nisi quas nam velit vitae illum ut temporibus quaerat animi facere sit cumque suscipit veritatis modi inventore nesciunt, dolores nostrum.
Nam cumque ipsa magnam laborum aperiam impedit rem reprehenderit dolorem, ea facilis odit debitis, nisi cum. Mollitia, porro officia! Impedit soluta harum nisi expedita. Itaque atque vero soluta veritatis eveniet?
Lorem ipsum dolor sit amet consectetur adipisicing elit. Sunt fuga laborum excepturi ipsam. placeat id. doloribus doloremque cupiditate adipisci autem aliquam. molestias tenetur ducimus natus totam expedita nesciunt unde beatae.
Praesentium fuga, necessitatibus, eius a veniam modi autem deleniti aspernatur aliquid, delectus ex repellendus. Inventore accusamus harum id eum, necessitatibus quisquam exercitationem ipsa laboriosam ex adipisci molestiae, placeat aspernatur culpa.
Quaerat nihil earum est error hic facere ipsa rerum beatae excepturi! Deleniti doloribus magnam porro, nobis exercitationem totam hic culpa fugiat modi id! Fuga, incidunt? Animi ad minima nostrum modi.
Rem praesentium magnam error eius culpa nobis ab commodi molestias. Doloribus placeat nesciunt corporis magnam beatae recusandae dignissimos modi, repellat amet officiis, dolore adipisci, atque esse officia. Ex, laboriosam facere.
Corrupti possimus, quo rerum tenetur non eius esse hic velit, eaque maxime soluta sapiente suscipit, et quia. Nihil consectetur voluptatibus facilis veritatis, ad ea, esse cumque, vero omnis minima magnam!
Lorem ipsum dolor sit amet. consectetur adipisicing elit. Asperiores atque nulla pariatur culpa vero non delectus. suscipit ex quisquam nam iusto inventore mollitia ducimus. ab nostrum eveniet. maiores quo ut?
Eum soluta deleniti sit iure aperiam ad voluptas dolorem aspernatur? Sapiente pariatur dolorem at? Asperiores odio necessitatibus nobis dolores tempore libero perspiciatis atque, culpa incidunt, dolore velit, at

Rendered in Browser



Lorem ipsum dolor sit, amet consectetur adipisicing elit. Fugiat placeat, ducimus sint architecto itaque corporis saepe, minus quod tenetur eum maiores rem, dolorum illo dignissimos neque totam molestias! Soluta, aspernatur?
Amet, ducimus veniam temporibus beatae repellendus culpa vero modi, fugit ad voluptates ullam est dolorum laborum incidunt corrupti numquam voluptatem id animi natus cum inventore ut nisi. Esse, blanditiis aliquid!
Molestias doloribus, fugiat commodi perspiciatis doloremque iste suscipit dolores. Harum, ea perferendis repudiandae consectetur quas possimus fugit sint facere dolor aliquid, dolorum, molestias a nostrum dolores quasi nam obcaecati expedita!
In earum illum quam commodi ab eos velit, sapiente provident officia nihil eaque hic magnam culpa consectetur sit rem deserunt explicabo. Omnis illum molestias ipsam suscipit magni tenetur officia voluptas.
Nisi fuga cumque sapiente ex, possimus voluptas repudiandae perferendis harum exercitationem aperiam totam optio debitis rem, aliquid sit perspiciatis in placeat dignissimos consectetur. Pariatur magnam consectetur nobis quidem aliquam quaerat?

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Accusamus quas molestiae cumque corrupti necessitatibus, sint totam nobis quisquam. Quas ipsa voluptate, voluptates excepturi quibusdam illum eos culpa quos soluta saepe!
Repellendus voluptatem nostrum voluptas dolorem officia, fuga quis, odio porro facere, ab laborum odit est praesentium suscipit nihil quisquam ullam quaerat? Debitis possimus soluta suscipit labore itaque enim sed mollitia.
HELLO HELLO EL
Sit suscipit impedit tenetur iste ipsum repudiandae explicabo excepturi dolore molestiae minima necessitatibus eaque asperiores, enim libero ea, voluptate quae nam incidunt assumenda similique fuga esse perspiciatis iure saepe! Laboriosam.
Quas rem deleniti quisquam laborum dolor quia, nam aperiam repellendus eius, vel rerum ipsam harum ipsum esse cum, sunt nostrum in deserunt quis perspiciatis assumenda fugiat est incidunt magni! Odio?
Earum modi et ullam! Laboriosam aperiam nobis beatae culpa neque, at tempore ex iusto doloribus possimus sed? Eveniet quod quos facilis obcaecati cumque ipsam, vel amet labore exercitationem sunt error?
Lorem ipsum dolor, sit amet consectetur adipisicing elit. Ex soluta nobis eius quisquam est beatae a, eligendi harum perferendis expedita quo dignissimos! Sit amet excepturi maxime quis, nesciunt eligendi delectus.
Tempora saepe mollitia omnis, et ipsum aperiam asperiores odit! Ut hic placeat nostrum molestias doloribus ipsa rem, nobis voluptatum natus provident deleniti, delectus est consequatur cumque voluptatem temporibus quia libero!
Porro enim asperiores perferendis laborum nesciunt aliquid, quia temporibus, iusto nemo repudiandae odio et eveniet cum, sint nobis veritatis aperiam. Blanditiis error enim numquam nulla ipsam, reiciendis reprehenderit voluptatem atque?
Asperiores illo quasi ratione eveniet maiores, voluptatem aperiam! Veniam porro nisi quas nam velit vitae illum ut temporibus quaerat animi facere sit cumque suscipit veritatis modi inventore nesciunt, dolores nostrum.
Nam cumque ipsa magnam laborum aperiam impedit rem reprehenderit dolorem, ea facilis odit debitis, nisi cum. Mollitia, porro officia! Impedit soluta harum nisi expedita. Itaque atque vero soluta veritatis eveniet?
Lorem ipsum dolor sit amet consectetur adipisicing elit. Sunt fuga laborum excepturi ipsam, placeat id, doloribus doloremque cupiditate adipisci autem aliquam, molestias tenetur ducimus natus totam expedita nesciunt unde beatae.
Praesentium fuga, necessitatibus, eius a veniam modi autem deleniti aspernatur aliquid, delectus ex repellendus. Inventore accusamus harum id eum, necessitatibus quisquam exercitationem ipsa laboriosam ex adipisci molestiae, placeat aspernatur culpa.
Quaerat nihil earum est error hic facere ipsa rerum beatae excepturi! Deleniti doloribus magnam porro, nobis exercitationem totam hic culpa fugiat modi id! Fuga, incidunt? Animi ad minima nostrum modi.

Corrupti possimus, quo rerum tenetur non eius esse hic velit, eaque maxime soluta sapiente suscipit, et quia. Nihil consectetur voluptatibus facilis veritatis, ad ea, esse cumque, vero omnis minima magnam!
Lorem ipsum dolor sit amet, consectetur adipisicing elit. Asperiores atque nulla pariatur culpa vero non delectus, suscipit ex quisquam nam iusto inventore mollitia ducimus, ab nostrum eveniet, maiores quo ut?
Eum soluta deleniti sit iure aperiam ad voluptas dolorem aspernatur? Sapiente pariatur dolorem at? Asperiores odio necessitatibus nobis dolores tempore libero perspiciatis atque, culpa incidunt, dolore velit, at blanditiis similique!
Laudantium perferendis tempora inventore vero aspernatur? Expedita corrupti iusto natus voluptates delectus dolorem. Atque quidem animi maxime quam cum recusandae sunt molestias numquam. At quasi accusantium in, sapiente laboriosam natus!
Consectetur tempora quisquam rerum, at officiis fugit ratione explicabo ducimus voluptatum labore quasi quae incidunt, quam soluta! Culpa mollitia natus optio deserunt. Debitis, necessitatibus vero natus ullam quia porro ad!

Problems Faced and solutions

This project was completed on time with lots of effort. Although the project was completed on time, many problems and errors were faced during the development of this project and solutions to the problems were obtained by discussing with friends and teachers and thorough study of the codes finally made the project work successfully. The major problems that occurred during the development of project can be listed as follows:

- As the SDL graphics library was completely new for us, we had to go through the e-books and video tutorials related to this library and check the codes time and again.
- We faced the problem while calculating the geometry of various elements and font rendering which was solved by consulting different materials available on the internet.
- Tokenization, Parsing and Rendering were a new topic for us. So, a lot of time was consumed on their algorithm.
- We used Visual Studio Code a lot but the user interface of Visual Studio was new to us.
- Problems occurred while merging the code from different members and a lot of conflict occurred in merging the code through version control.
- The code readability was poor in the beginning, but we enhanced the code by adding comments and meaningful variable names.
- We had a slight complexity in parallel time management for the project as well as academic activities like assignment, lab report submission.

Limitations and future enhancements

Due to the busy schedule of classes and the limited timeframe for project completion we are unable to make this project as we desired. Following are the limitations of our project:

- Various tags like image, span, input and other inline elements are not supported in our project due to the difficulty in layout calculation of inline and inline block elements.
- ASCII characters are only supported (UTF-8 not supported).
- Implementation of advanced CSS like animation, pseudo selectors, and media query are not implemented.

Following Enhancements can be made in this project to make it more similar to other popular parsers:

As our project is a minimal HTML parser and Renderer, we can enhance it a lot. Mainly, we can add the support for inline and inline block elements so that most of the tags can be supported by the project. We can add animations and make our project to support responsive design. Further, to make fully fledged parser and renderer Javascript can be implemented which is the heart of any webpages.

Conclusion and recommendations

Our project “HTMLer” is a representation of a page/window just as the replica of a webpage by rendering some of the common HTML tags & CSS contents. Though our project does not fulfill other major requisites for a wonderful and awesome website, it can definitely become a backbone for understanding the core mechanism of how these codes perform in the backend to execute and display the required result.

So, this project is not a practical implementation to render websites commercially, it can just render plain ASCII text and some CSS stylings to modify the contents. Since we could add more to make the use of various tags in HTML & stylings in CSS along with the implementation of web programming languages like JavaScript, it was found to be more complex and time consuming in the given time frame.

References

The following are the resources and references we have used:

1. https://html.spec.whatwg.org/multipage/parsing.html#overview-of-the-parsing-model
2. https://www.w3.org/TR/css-syntax-3/#tokenizing-and-parsing
3. https://html.spec.whatwg.org/multipage/parsing.html
4. https://dom.spec.whatwg.org/
5. https://www.w3.org/TR/CSS2/visudet.html#containing-block-details
6. Robert Lafore, "Object Oriented Programming with C++", Sams Publication
7. Daya Sagar Baral and Diwakar Baral, "The Secrets of Object oriented Programming", Bhundipuran Prakasan
8. Bjarne Stroustrup, "A Tour of C++", Addison-Wesley Professional .
9. https://www.learncpp.com
10. https://stackoverflow.com
11. https://developer.mozilla.org/en-US/docs/Web/Performance/How_browsers_work
12. https://www.libsdl.org/