

**Acknowledgement**

The **“Linky-The Text Based Browser”** is created by the efforts and motivation of many people. Major impact is made by our Professor. Hereby we would like to extend my thanks and gratitude to our project guide **Dr. Honghua Dai.** They help us to each phase of our project and we get continuous feedback from our professor and timely assistance throughout the development of this project.

We are very glad to take this opportunity to acknowledge all those who helped us in designing, developing, and successful execution of our Project **“Linky-The Text Based Browser”.**

Table of Contents:

**1: Description of System Developed**

1.1 What is the system Developed?

1.2 Functionalities and the features of your system

1.3 State what was completed and what has not and why?

1.3.1 Completed Features

1.3.2 in-Complete Features

1.4 Scope of the Linky

1.5 A web browser Building Challenges

1.6 Brief Summary on Achievement overall and each individual

**2: System Implementation**

2.1 What has been implemented?

2.2 How was it Implemented?

2.2.1 Techniques, languages, tools?

2.2.2 Algorithms

2.2.3 Tools

2.3 Pros and cons of Linky browser

**3. Systems Testing and Quality assurance**

3.1 Test Plan

3.2 List of test cases/Items

3.3 Designed test data

3.4 Test results

3.5 Description of system’s quality, reliability and accuracy etc.

3.6 Assumptions

3.7 Feasibility Study

3.8 Hardware and Software Requirements

**4. Systems Documentation**

4.1 User Manual

* 1. Downloading and installing instructions

## 4.3 System Online help

**5. Conclusion**

**6. Future Enhancement of Text based browser**

**7. Source code Attachments**

**8. References**

# 1: Description of System Developed

## What is the system Developed?

The system being developed is a **Text-based web browser**

**About Web Browser:**

A web browser engine (here and there called design engine or rendering engine) is a program that renders increased substance, (for example, HTML, XML, picture records, and so forth.) and organizing data, (for example, CSS, XSL, and so forth.) in well-formatted document.

A design layout engine is a run of the main segment of web programs, email customers, digital book pursuers, on-line help frameworks or different applications that require the showing (and altering) of web substance. Web browser engine may sit tight for all information to be gotten before rendering a page, or may start rendering before all information are gotten. This can bring about pages changing as more information is gotten, and adds dynamicity to the web pages for example, pictures being filled in or a blaze of unstyled substance if rendering starts before designing data is gotten.

**Technical Operation:**

The previous web browsers were solid and monolithic. They utilized different methods acquired from content preparing and then they use to parse. For example, general expressions, to parse HTML into a visual portrayal. Later they embraced a more particular approach and were part into a host application and a motor.

* The **web engine** does the greater part of the work. It takes a URL and an arrangement of window (substance region) rectangle facilitates as contentions. It then recovers the report relating to the URL and paints a graphical portrayal of it in the given rectangle. It additionally handles joins, frames, treats, customer side scripting, module stacking, and different matters.
* The **host application** gives the menu bar, address bar, status bar, bookmark chief, history and inclinations usefulness (in addition to other things). It combines the engine and server in as an interface between the client, the motor, and the basic working framework. Since it gives the graphical components encompassing the region, in which the motor paints records, software engineers now and then utilize the term chrome to allude to its UI (like the chrome encompassing an auto).

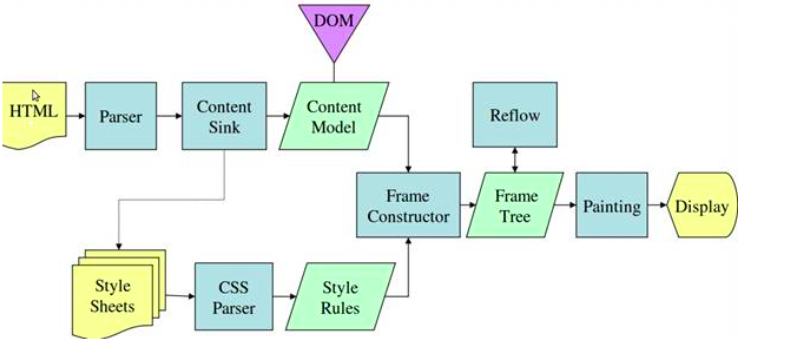


Fig1. Document Object Modelling tree

## 1.2 Functionalities and the features of your system?

* Text-based web browser.
* HTTP Client.
* HTML Parser.
* HTML Renderer.
* Mouse Event Listener.
* Interactive Graphical User-Interface.

## 1.3 Comparison to your proposal. State what was completed and what has not and why?

### 1.3.1 Completed Features

* + HTTP Client: it is the module that is responsible for fetching the
  + HTML document form the remote web-server. The browser works on HTTP GET Requests.
  + Mouse Event Listener: the module listens was built to intercept users inputs and trigger corresponding events.
  + Interactive Graphical User-Interface: the user interface is very simple, with an input for web URl. Once when the user enters the URL and hits enter the HTML is rendered to the screen.
  + HTML Parser: the parser parses the HTML from the HTTP response into a parse tree.
  + HTML Renderer: the renderer parses the HTML into the ﬁnal display.
  + Although, only a small but the most popular HTML Tags are handled.
  + Tags handled COMMENTS, HTML, HEAD, BODY, DIV, TITLE, P and H1 - H6.

Although, all the features mentioned above were implemented, certain features were developed to a point to demonstrate its functionality. Modern browsers have been around from 1990 and have built up very sophisticated features. The aim for the current project was to demonstrate a skeleton implementation within a semester.

### 1.3.2 in-Complete Features

* There are 120 HTML tags deﬁned by the W3c.
* Out of the above 10 Tags were depreciated.
* There are about 25 HTML Tags most commonly used.
* Out of 25 of the most popular TAGS used 10 were supported.
* The 10 Tags supported demonstrated the key principles used in constructing a web browser.

### 1.4 Scope of the Linky

* To reduce the server load
* Reduces the lagging from various websites.
* Reduces the loading time of webpage
* Increases the response time.
* Provide a hassle free work procedure

### 1.5 Challenges faced

Building a web browser from scratch requires large technical challenges. Some of the challenges were:

* Limited to 10 development hours per week for the entire project.
* Lack of design documentation for existing projects and literature in the subject.
* Most of the programming involved was experimental.
* Compilers were studied to understand parsing and rendering.

### 1.6 Brief Summary on Achievement overall and each individual

While the main aim of this project is to build the Text based web browser and in parallel learn how to build it and technologies involved in the completion of this project, so as to keep the force up to industry level.

While building this project, basically all the Team involved in building and coding some of the parts of the project, but we ourselves majorly divide the tasks as below:

1. **Arvind(Team Leader)**
   * Design and Analysis of the project.
   * Build HTML Parser and Renderer
2. **Nupura**

* **Built frontend fully-functional model/prototype of the browser**
* **Designed UI for the browser**
* **Functional testing for the browser**

1. **Tejas**

* Building Browser component and Designing parser

1. **Mohammad**

* Testing the browser and supporting other browsing components

# 2: System Implementation

## 2.1 What has been implemented?

* + HTTP Client.
  + HTML Parser.
  + HTML Renderer.
  + Mouse Event Listener.
  + Interactive Graphical User-Interface.

## 2.2 How was it Implemented?

### 2.2.1 Techniques, languages, tools?

* Java
* Gradle
* JSoup Library
* JavaFx
* JavaFx Scene Builder
* Jquery
* CSS
* Webkit

### 2.2.2 Algorithms

* + Custom Model-View-Controller design pattern.
  + Singleton Design Pattern for the renderer.
  + Factory Design Pattern for the Renderer Object.
  + Tree based algorithms for the Parser and Renderer.

### 2.2.3 Tools

* + Gradle was used for build management.
  + Trello was used for project management.
  + Git was used to version code.

## 2.3 Pros and cons of Linky browser

Advantages of Web based Text Browser: -

* Reduces Server Load
* Quick response time to client.
* Centralize Database Access Helps in Universal Accessibility
* Processing and Execution of URL Becomes Faster
* Usually more powerful than UI tools
* Easy to stand on the shoulders of giants (aliases, piping,)
* You get to know and handle the tool better

Disadvantages: **-**

* Rendering becomes difficult in terms complex and heavy websites.
* Not useful where message need to be conveyed with the help of pictures.
* Steeper learning curve (mostly, although there are UI with even steeper curves)
* Worse overview
* Conventions not as well-known as UI conventions
* Sometimes cryptic parameters
* Bad for certain tasks that involve selecting or graphical tasks

# 3. Systems Testing and Quality assurance

## 3.1 Test plan

Test Objective/ Requirement

* Iterative and experimental development required that tests be performed directly on the implementation.
* Testing involved White, Black and Grey box testing with minimal or no Unit tests.

The plan is to test this browser to test its functionality with regards to text fetching and rendering. For this Lot of test cases has been implemented in this testing,

* A Sample HTML page has been created
* Different tags has been created and formulated
* Negative and positive test cases has been developed

## 3.2 List of test cases/Items

We have made a list of all test cases that we have d.The list is in the format of EXCEL sheet. All the important tags has been tested and made in a list format.



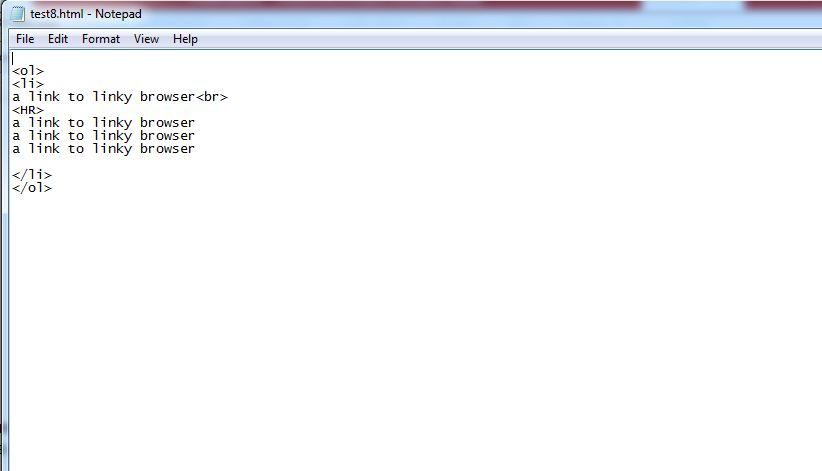
## 3.3 Designed test data

Test data is the list of tags in HTML and made them in different formats and check in the browser. By doing this we can measure the fetching capacity, of the browser that we have built.

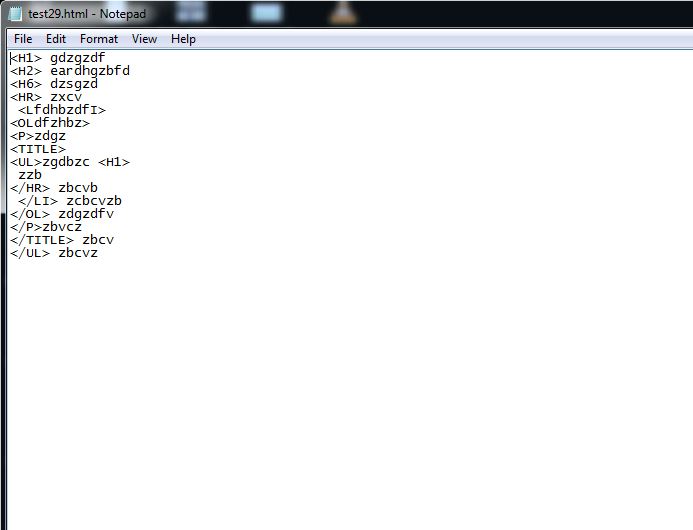
There are two types of data test, positive and negative:

Positive test data is to test the functionality of browser in a normal case without any faults faced by the browser.

Example:



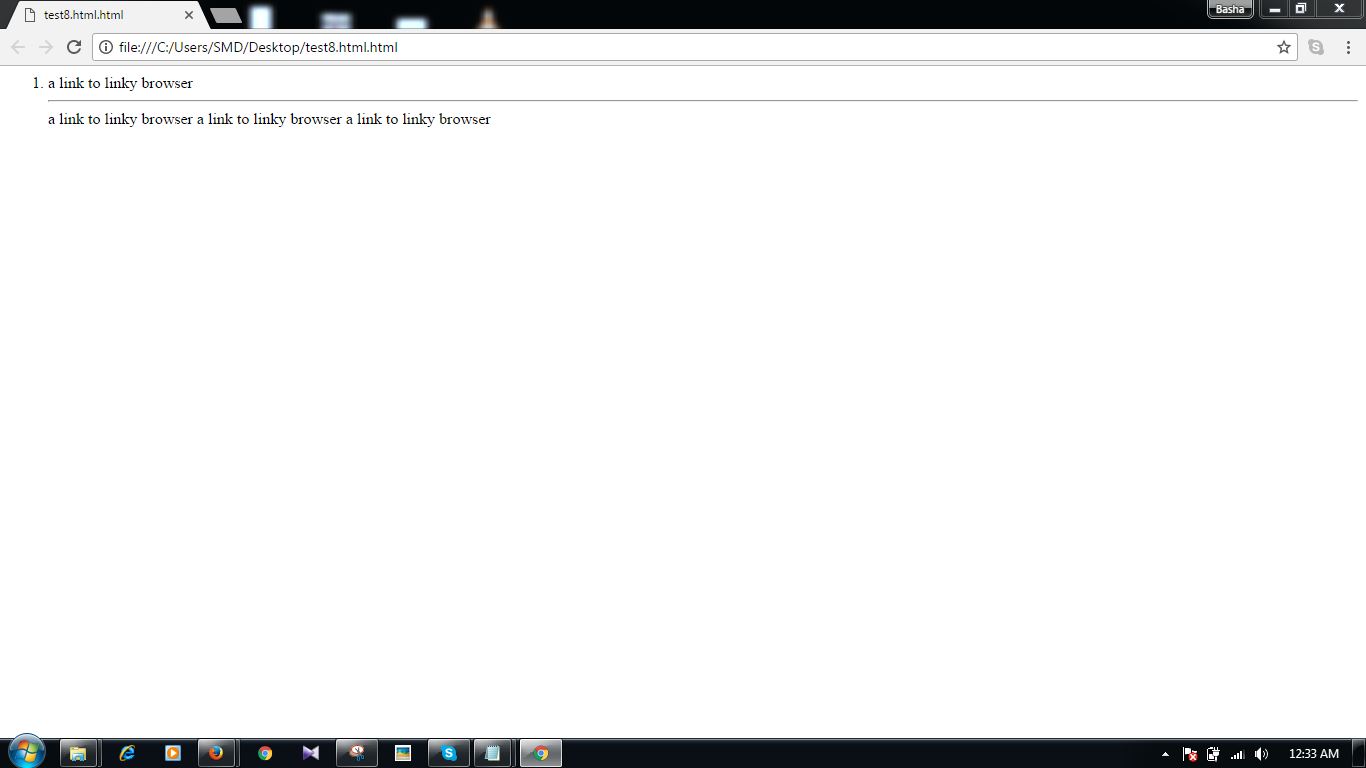
Negative testing is the bizarre combination of code that browser may have to have in the real world scenario.

Example:

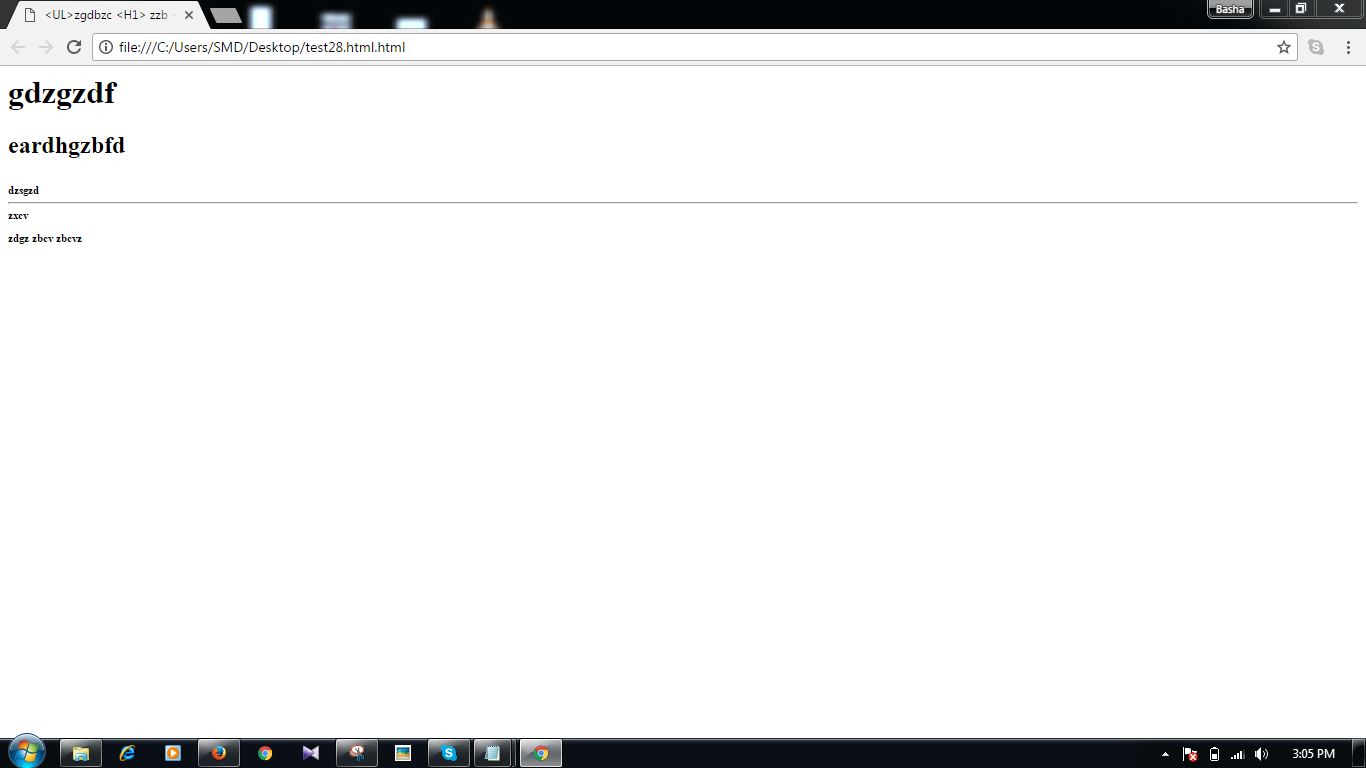
## 3.4 Test results

The browser has been rendered every text that was given to it. The test results were positive and the browser has been working perfectly to text only the TEXT from the web page.

Positive test result



Negative Test Result



3.5 Description of system’s quality, reliability and accuracy etc.

As this system is in the initial stages of development the quality of this system is not so great but the functionality is good

## 3.6 Assumptions in Test cases

* Performance and Security Testing Out-of-scope.
* No automated tests were considered as the project was experimental.

## 3.7 Feasibility Study

It is a test of system proposal according to its workability, impact of the organization, ability to meet needs and effective use of the resources.

* System provides GUI interface with online server, just go to our link and login there and select the branch.
* Available resources are java, MySQL languages.
* Our system is capable of solving the all kind of customer’s problems using help section on the web page.

## 3.8 Hardware and Software Requirements

Hardware Requirements:-

Processor: - Intel Core Duo 1.6 GHz Onwards

RAM: - Minimum 1 GB

Monitor: - CRT/LCD Monitor

Keyboard: - Normal or multimedia

Mouse: - Compatible mouse

Software Requirements:-

Operating System: - Windows Xp And Above

* Front End: - Java SE Development Kit 7

# 4. Systems Documentation

## 4.1 User Manual

* Using Browser to navigate or surf the web

1. Make sure to type the address of the website correctly in the address bar at the top
2. Address must always start with “http://”. For example: “http://www.abc.com”
3. Click on the Load Button next to the address bar once you have typed the address

* Accessing Browser History

1. The browser history drop menu is located at the bottom of the browser window
2. Click on the drop menu and select the previous page that you want to navigate to

* Navigating Back and Forward

1. Right click anywhere on the browsing window screen to access the “Go Back” or “Go Forward” Feature

* Accessing Help Section

1. Click on the Help button at the bottom of the browser window to access the Help section
2. To exit the help section click the “Exit the Application” Button

* Accessing Documentation

Click on the Documentation Button at the bottom of the browser window to access the Documentation for the browser

## 4.2 Downloading and installing instructions

There are two projects for presentation “Code.Zip” and “Browser.zip”

**Code.ZIp**

This projects demonstrates the implementation of custom MVC, Factory design pattern, Singleton design pattern, parser and renderer.

* Unzip the project.
* Install Gradle (<https://gradle.org/)>
* Run gradle run within the project.
* Test cases under (src/test/data/testcases)

**Browser.Zip**

This browser is the normal package of browser where one can download it as usual from the hosting website and install it in the PC like any normal browser installation.

The installation will follow some basic installation guide and show the user

* Install Eclipse IDE
* Download and Install java sdk 1.8
* After installation, check if java jdk and jre are successfully installed through command prompt/open terminal. Restart computer, if required.
* Download and extract Browser.zip file as per the following path: “C:/Browser/src/textilate”
* For additional instructions, go through the README.txt file located in Browser.zip
* Build and run the TestBrowser.java file using Eclipse IDE

## System Online help

Accessing Help Section in the Browser

* + Click on the Help button at the bottom of the browser window to access the Help section
  + To exit the help section click the “Exit the Application” Button

# 5. Conclusion

In our project entitled “**Linky-The Text Based Browser**” we have tried our best to fulfils all the requirements from clients prospective. The project being simple and flexible, it is running smoothly. The main advantage of our project is that its simplicity attracts a lot of users. It can be easily run by a novice user.

This project when implemented, it will remove all the security issues. Also there will be speedy and secured authentication procedure for the maintenance of records. Data display is fast and easy, because it automatically filters the images which are coming from the server side. Therefore, our software will definitely prove to be a successful steppingstone in replacing the outdated manual method of browsing. The work plan also includes the detailed features of the technology used in the project defining the front end and the back end. The objectives and scope of the project in future have been elaborated.

# 6. Future Enhancement of Text based browser

Although the automobile industry is very competitive, the lifestyle changes created by the modern living continue to fuel its steady growth. Trends are important and our software is well help to generalize the work flow of the system. So while we keep in mind that the fact there is no software that is perfect, which would also mean there is perfect tower management software. We put in an endless effort to improve our tower management software.

There are abundant features that are not included in this project, so we are mentioning those features that can be added in our project in the future. Those features are listed below:

* The software can be used with a Web Based Interface.
* The future version of this software could also be “multi-lingual” and “self-translating” as well.
* We can improve the UI management system by keeping a track of every component utilized for a particular site.

# 7. Source Code Attachments

* Browser.zip

# 8. References

Kendall, K.E., Kendall, J.E., Kendall, E.J. and Kendall, J.A., 1992. Systems analysis and design (Vol. 4). New Jersey: Prentice Hall.

Satzinger, J.W., Jackson, R.B. and Burd, S.D., 2011. Systems analysis and design in a changing world. Cengage learning

MJG International Free Conference at Google HQ,California <http://dayofjs.com/videos/22158462/web-browsers_alex-russel>

<https://crash-stats.mozilla.com/home/product/Firefox>

[www.google.com](http://www.google.com)

[www.tutorialspoint.com](http://www.tutorialspoint.com)

[www.youtube.com](http://www.youtube.com)

[www.w3school.com](http://www.w3school.com)

END OF REPORT