



INDUSTRIAL PROGRAMMING

Coursework 1



OCTOBER 27, 2015

MOHAMED SHARIF
H00158013

Table of Contents

Introduction.....	3
Requirements' Checklist.....	3
Logic	3
User Interface.....	3
Design Considerations.....	4
Class Design.....	4
GUI Design	5
Data Structures Used	5
Advanced Programming Features	5
User Guide.....	6
Installation	6
Using the Application	7
Navigation	7
History.....	9
Tabs	9
Window.....	10
Homepage.....	10
Favourites.....	10
Developer Guide.....	14
Functions	14
Program Class	14
WindowGUI Class.....	14
Tab Class.....	16
WebPage Class.....	17
XMLManager Class	17
AddControl Class.....	17
History Class	17
Favourite Class.....	18
ExtObject Class.....	18
ExtString Class.....	18
Testing.....	19
Functionalities	19
Web Exceptions	19
Adding an Existing Favourite.....	19
Using an Invalid URI.....	19

Conclusion	19
------------------	----

Introduction

The assignment was to develop a simple web browser that can browse and load webpages in html text format, while having favourites, history and tabs functionalities. This report will show the web browser features, design considerations, developers/user guide and testing the browser in different conditions. The assumptions made during the development was not to use Web Browser class and to use windows form to create the browser GUI.

Requirements' Checklist

Logic

- **Sending HTTP request messages** for URLs typed by the user. [\(Completed\)](#)
- **Receiving HTTP response messages** and display the contents of the messages on the interface. Note that you are only required to display the HTML code returned to the web browser from the web server (HTML parsing and graphical display are not required). In addition to the 200 (OK) HTTP response status code, the following HTTP response status error codes and the display of their corresponding error messages should be supported: [\(Completed\)](#)
 - ❖ 400 Bad Request [\(Completed\)](#)
 - ❖ 403 Forbidden [\(Completed\)](#)
 - ❖ 404 Not Found [\(Completed\)](#)
- **Home page:** The user should be able to create and edit a Home page URL. The Home page URL should be loaded on the browser's start up. [\(Completed\)](#)
- **Favourites:** The user should be able to add a URL for a web page requested to a list of favourite web pages. The user should also be able to associate a name with each favourite URL. Support for favourite items modification and deletion is required. The user should be able to request a favourite web page by clicking its name on the Favourites list. On the browser's start up, the Favourites list should be loaded to the browser. [\(Completed\)](#)
- **History:** The browser should maintain a list of URLs, corresponding to the web pages requested by the user. The user should be able to navigate to previous and next pages, and jump to a page by clicking on the links in the History list. On the browser's start up, the History list should be loaded to the browser. [\(Completed\)](#)
- **Multi-threading:** The web browser should be designed in such a way that the browser-server communication is separated from the GUI support, using different threads. This way the user can request more than one page at a time. This should be implemented by supporting separate Tabs in the browser, with one thread for each Tab, and with each Tab containing information about its local state. [\(Completed\)](#)

User Interface

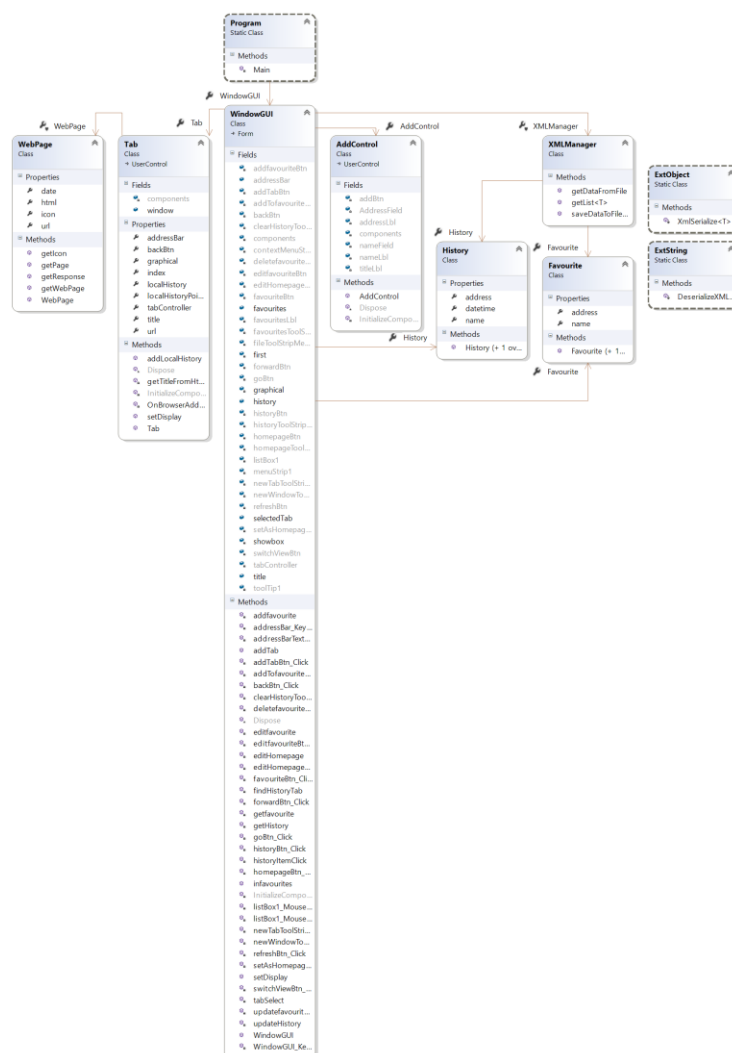
- Using the **GUI**, the user should be able to perform the operations discussed in the previous section. [\(Completed\)](#)
- **Make use of menus** (with appropriate shortcut keys) as well as buttons to increase accessibility. [\(Completed\)](#)

Design Considerations

Class Design

Classes have an appropriate identifier for the elements that make up class shown below along with adequate commentary within the code and xml style comments. Access modifiers are made private to help implement encapsulation so only the current class will have access to the field or method unless necessary to help faster development. A class for each feature is implemented displayed below.

- Favourite (Favourite.cs)
- History (History.cs)
- HTTP Communication (WebPage.cs)
- Tab (Tab.cs)
- XML Management (XMLManager.cs)
- Add Control Template (AddControl.cs)



GUI Design

Consideration was taken while designing the Graphic user interface of the browser. The aim of the GUI design was to create a browser that is easy to use and be readable.

The overall design used flat simple style to give it the feel of a modern browser and to make it more appealing to use.

Navigation button were implemented shown in bellow was to allow the user to move forward and back through the webpages and be able to refresh the page. The buttons are large to increase the accuracy of clicking and keyboard shortcuts were added such as the return key so the user can type in their website and press their return key to request the website. Images were added to the navigation buttons for easier readability.

Data Structures Used

To be able to move forward and back pages a string list was implemented which stored the URL's of the website. When the button is pushed it moves the current tab's pointer and displays the webpage.

The history and favourites are saved into the xml file which use the xmlmanager to convert to xml data

Advanced Programming Features

Delegates is used with multi-threading in the development of the browser. A delegate is a reference type variable that holds the reference to a method, in this case delegates are used with threading in which a new thread is created for a new tab and it is then invoked at the http request/response and the setting of the display.

Exceptions is used in handling errors such as websites returning 404's and exception handler then deals with the error case.

Generics is used in the implementation of the XMLManager to save generic lists to files and convert xml into list objects.

Extensions is used in the ExtString and ExtObject language classes to extend there functionalities to serialise and deserialise to and from xml.

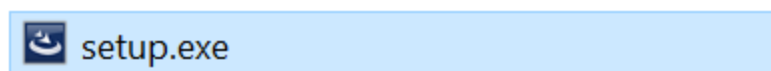
User Guide

The features this browser supports are

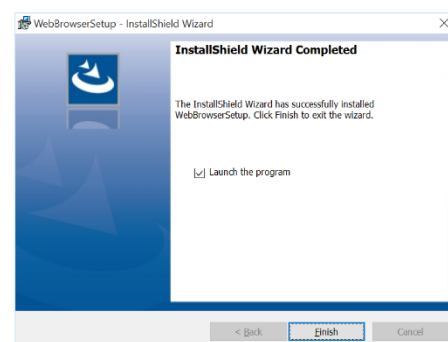
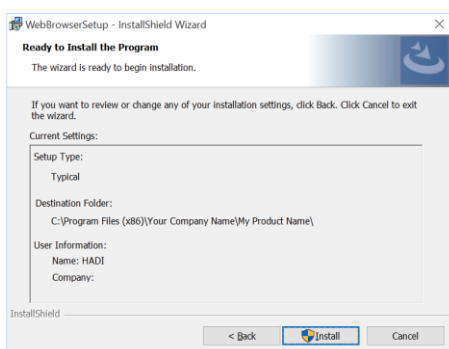
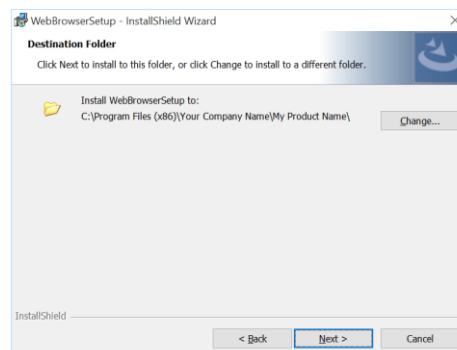
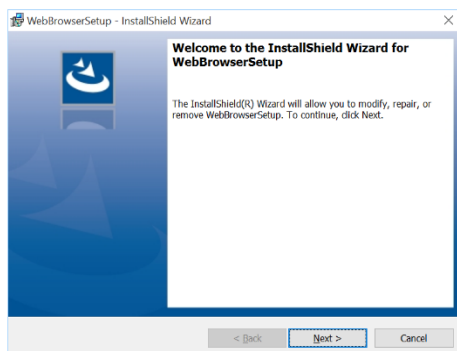
- Searching for websites and displaying them as raw html or graphical view
- Favourites
- History
- Navigation controls
- Homepage
- Multiple Tabs
- Multiple Windows

Installation

To install the application first you would have to run the setup file.

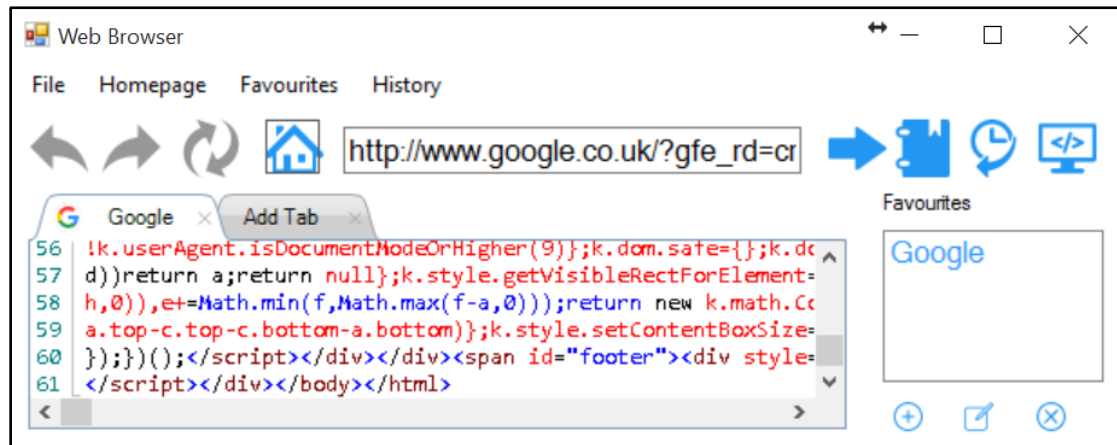


Then follow the setup steps to install the applications



Using the Application

Once you open the application it will open the following window and automatically loads the homepage if it is set.



Navigation

To navigate to a page first you need to enter the URL into the address bar



Then click on the go button or press enter on the keyboard.



The resulting page will be displayed here and in text format by default.



To display in graphical format click on switch view button






The webpage graphical representation will then be display here




History

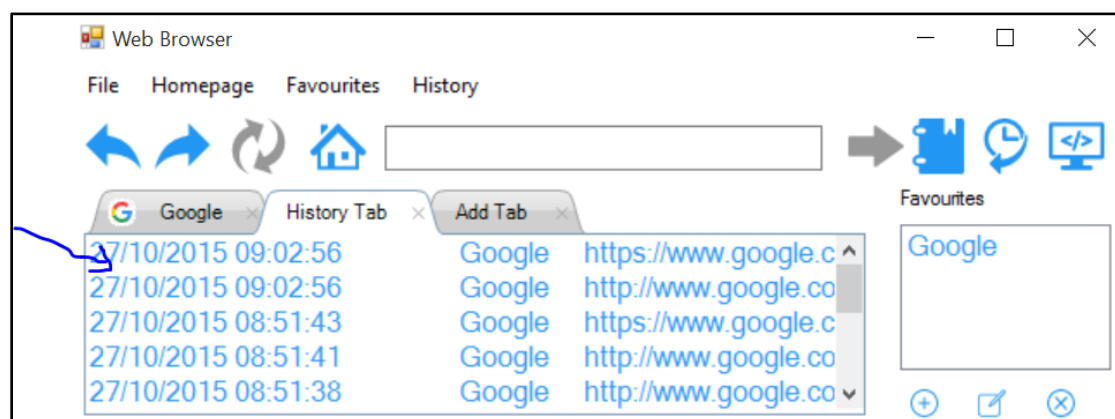
Local History

Every tab has a local history which gives you the ability to press the back button  to go back to previous page, or the forward button  to go to the next page if you have went back already in the selected tab.

To reload the currently loaded page just click on the refresh button. 

Global History

The global history for all tabs can be accessed by clicking the history button  this will the open the history tab, which will display a list of the history.

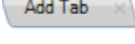








By clicking on any history item in the list you will navigate to that items webpage in a new tab.


To clear the history just click on the history context menu   File  Homepage  Favourites  History and click on the clear history option  Clear History Ctrl + Shift + H or using the keyboard shortcut Ctrl + Shift + H

Tabs


Adding Tab

To open a new tab click on the add tab button  or by clicking the File context menu   File  Homepage  Favourites  History and click on the new tab option  New Tab Ctrl + T or by using the keyboard shortcut Ctrl + T.

Navigate to Tab

To open an already open tab just click on the tab .

Closing Tab

To close an open tab just click on the close tab icon  on the tab you want to close.

Window

Maximising Window

To maximise a window just click on the maximise icon.



Minimising Window

To minimise a window just click on the minimise icon.



Closing Window

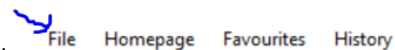
To close a window just click on the close icon.



Note by closing the main window you will close all windows.

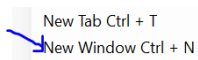
Add new Window

To add a new window just click on File context menu



and


then click on the new window option



or by using the keyboard shortcut Ctrl + N.

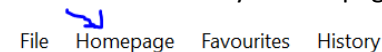
Homepage

Navigate to Homepage

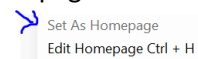
To navigate to homepage just click on the homepage button  which will load the homepage in the currently selected tab.

Set as Current

To set the currently loaded page as the homepage just click on the Homepage context menu

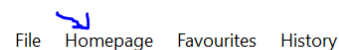


and click on Set As Homepage option.

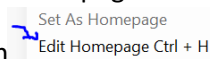


Edit Homepage

To edit homepage just click on the Homepage context menu



and click on Edit Homepage option




or by using the keyboard shortcut Ctrl + H.

Favourites

Enable/Disable Favourites Display

You can disable/enable the favourites display






By clicking the favourites button  which will show the favourite display as seen in the image before or will hide the favourite display as seen in the image below

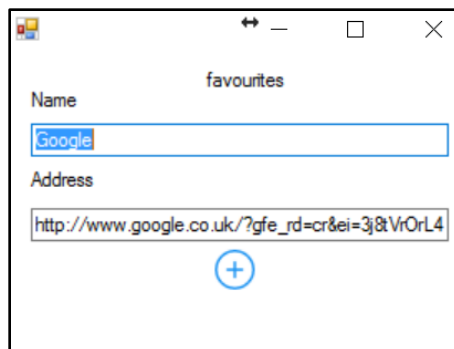


Add to Favourites

To add currently loaded page to favourites just click on the Favourites context menu

File Homepage  Favourites History and click on the Add To favourites option  Add To Favourites or if you have the favourites display enabled just click on the add button. 

This will open the add favourites window with pre-set name field and address field to that of the current page, however you can edit to your liking.

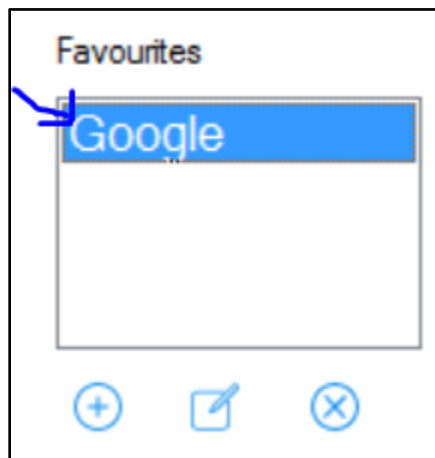



Then click on the add button to add to favourites.



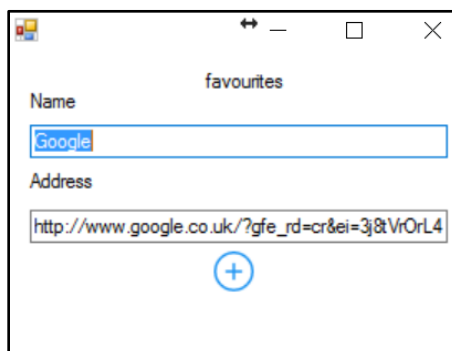
Edit Favourites

To edit a favourite in the favourites display you must first select a favourite in the list

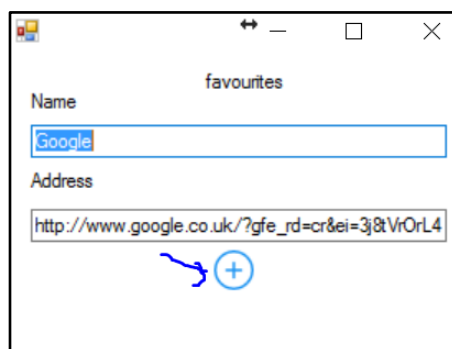


Then click on the edit button 

This will open the favourites window with pre-set name field and address field to that of the selected item where you can edit to your liking.

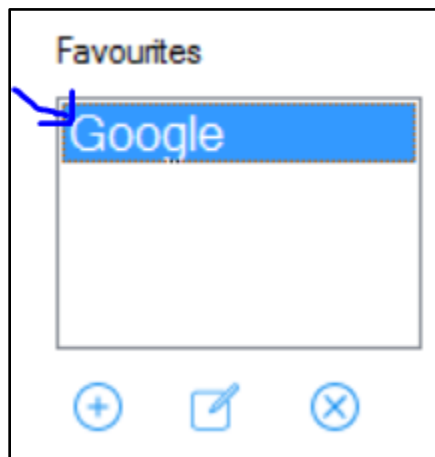


Then click on the add button to save the changes.



Delete Favourite

To delete a favourite in the favourites display you must first select a favourite in the list



Then click on the delete button. ⊗

Developer Guide

For readability, methods and classes are adequately named and internal commentary is present to aid in understanding of the code, I have also added xml style comments for all functions. The following guide will explain code that is implemented.

Functions

Program Class

This is the entry point of the program.

Main function

This function loads the main windows form.

WindowGUI Class

This class represent and holds data for the browser window

WindowGUI

This is the constructor for the class wish is used to initialise the components and load all the data from files and set the display to the homepage if available.

getfavourite

This is used to get the favourites from the file and set the favourite list with the result

getHistoy

This is used to get the history from the file.

goBtn_Click

This is used to handle the go button click which sets the selected tab display to the current address in the address bar.

setDisplay

This is used to set the display of the supplied selected tab to the supplied url result web page. This is done using a separate thread.

addTab

This is used to add a new tab to the selected window.

addTabBtn_Click

This handles the add tab button click by calling the add tab function to add a new tab to the selected window.

favouriteBtn_Click

This handles the favourite button click, which shows/hides the favourite display related controls.

backBtn_Click

This handles the back button click by loading the previous page visited in the selected tab and enable the forward button.

forwardBtn_Click

This handles the back button click by loading the next page visited in the selected tab if you have already clicked back and enables the back button.

refreshBtn_Click

This handles the refresh button click which reloads the currently visited page.

newTabToolStripMenuItem_Click

This handles the new tab tool menu strip menu item click by adding a new tab to the selected window.

newWindowToolStripMenuItem_Click

This handles the new window tool menu strip menu item click by adding a new window.

homepageBtn_Click

This handles the homepage button click by loading the homepage in the currently selected tab.

addressBarTextChanged

This handles the address bar text changes by either enabling or disabling the go button and set as homepage menu item if the address bar is not empty/empty.

tabSelect

This handles the tab select event by enabling/disabling the buttons and sets the address bar to the relevant address based on the selected tab data.

addressBar_KeyDown

This handles the address bar enter key pressed event by setting the display to the web page of the address.

switchViewBtn_Click

This handles the switch view button click by switching between text and graphical display.

setAsHomepageToolStripMenuItem_Click

This handles the set as homepage tool menu strip menu item click by setting homepage to the current address.

editHomepage

This is used to handle the add button click in the homepage addControl click by setting homepage to the address field.

editHomepageToolStripMenuItem_Click

This is used to handle the edit homepage tool menu strip menu item click by editing the currently set homepage using an modified version of the addControl control.

updatefavouriteList

This is used to update the favourite list with the favourites list stored in memory.

addfavouriteThis

This is used to add a favourite to memory and save it to the file.

Infavourites

This is used to check if the favourite already exists in the favourites list saved in memory.

addTofavouritesToolStripMenuItem_Click

This is used to handle the add To favourites tool menu strip menu item click by adding the currently loaded web page using an modified version of the addControl control.

listBox1_MouseClick

This is used to handle the favourite item click to show a tool tip for the selected item.

listBox1_MouseDoubleClick

This is used to handle the favourite item double click to show a load the page for the selected item in the selected tab.

editfavouriteBtn_Click

This handles the favourite display edit button click by editing the currently selected item using a modified version of the addControl control.

editfavourite

This is used to handle the add button click in the favourite addContorl click by setting the selected favourite to edited version of the favourite using the name field and address field values.

deletefavouriteBtn_Click

This handles the favourite display edit button click by deleting the currently selected item from favourites.

findHistoryTab

This is used to find the index of the history tab and returns the index if it exists otherwise returns null

historyBtn_Click

This handles the history button click by switching to the history tab if it exists otherwise it will open a new one.

updateHistory

This is used to update the history listbox items to view the latest history stored.

historyItemClick

This is used to handle the history item double click to show a load the page for the selected item in a new tab.

clearHistoryToolStripMenuItem_Click

This is used to handle the clear history tool menu strip menu item click by clearing the history stored and updating the file.

WindowGUI_KeyDown

This is used to handle all keyboard events in the currently selected window.

Tab Class

This is used to deal with the Tab related logic.

Tab

This is the constructor used to initialise a tab, add the related controls and initialise the local history and local history pointer.

getTitleFromHtml

This is used to get the title from raw html, which is used to set the tab title.

setDisplay

This is used to set the display of tab to the webpage of the supplied url.

addLocalHistory

This is used to add the supplied url to the local history.

OnBrowserAddressChanged

This is used to handle the ChromiumWebBrowser address changed event to add to history and update the display accordingly.

WebPage Class

This class is used to store webpage data and related logic

WebPage

This is used to construct a new webpage object.

getWebPage

This is used to get the webpage for the supplied url including html and icon.

getPage

This is used to get the html of the requested url webpage.

getResponse

This is used to get a HttpResponseMessage from the result of the HttpRequest of the supplied url;

getIcon

This is used to get the icon from the supplied url.

XMLManager Class

This is used to manage getting and setting xml data from the file.

getDataFromFile

This is used to get the xml data from the supplied file.

saveDataToFile

This is used to convert the supplied list of objects to xml and save them to the supplied file.

getList

This is used to get a list of objects given xml elements.

AddControl Class

This is used to supply a template for adding/editing items.

AddControl

This is used to construct an object of the class.

History Class

This is used to represent history as an object

History

This class has 2 constructors, the empty constructor is used by the xml deserialize method while the non empty one is used to construct an object with the values set.

Favourite Class

This is used to represent favourite as an object

Favourite

This class has 2 constructors, the empty constructor is used by the xml deserialize method while the non empty one is used to construct an object with the values set.

ExtObject Class

This is used to extend the object class with function.

XmlSerialize

This is used to serialise an object to a xml representation of the object in a string.

ExtString Class

This is used to extend the string class with function.

DeserializeXML

This is used to deserialize a string containing the xml representation of the object to an instance of that object.

Testing

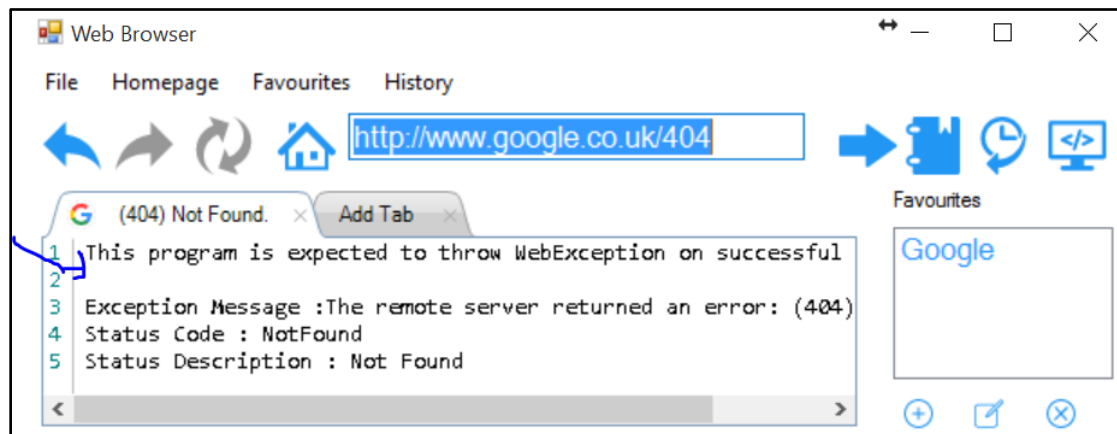
The application went through many test, which are discussed below in more detail.

Functionalities

All functionalities have been tested and are all working.

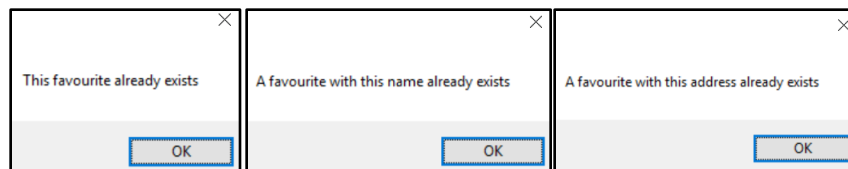
Web Exceptions

The application can handle web exceptions and returns a message in the text display with a relevant message.



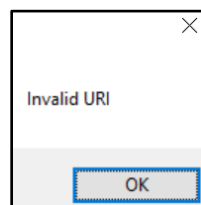
Adding an Existing Favourite

Returns a message box with relevant message based on if it's a duplicate name, address or both.



Using an Invalid URI

If you try to browse with an invalid URI, it will not cause the application to stop and it will return a message box with the error.



Conclusion

I am proud of the user interface design, the functionalities that my application supports, and learning how to add custom elements into your project. I would have wanted to remove the add button close icon from the add button if I had the time and I would have liked to add the ability to not close the application once the main window is closed when there exists another window open and incognito functionality.