Software Requirements Specification (SRS)

Project X

Team: Team 6
Authors: Aurimas Alkevicius, Stephen Ho, Hunter Stevens, Tran Phuc, Wang Xuhao
Customer: Front-end Web Developers
Instructor: Professor Daly

Table of Contents

1 Introduction

This document will establish a framework for the proposed software product. The first section of this document contains brief descriptions of the purpose, scope, and goals of the system to be created. It will also outline the high-level specifications of the system's perspective, functionality, and constraints. The specific requirements of a successful solution will follow next, as well as a series of various diagrams. A description of the prototype follows as well as sample scenarios of using the prototype. Finally, the document will conclude with a list of references and points of contact.

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) is to fully document the specifications and requirements for the cross-web browser package in an open-source text editor, Atom. The intended audience of this SRS will be the team developing this project, professor Daly, and possibly future generations of lost souls of Software Engineering I.

1.2 Scope

The purpose of this project is to create a cross-web browser package called X-Browser for an open-source text editor Atom. The benefit of such a package is the increased efficiency of the web page design process. X-Browser will display the real-time code edits in several different web browsers at the same time. In turn, it will allow web developers to immediately inspect any compatibility issues across various web browsers. Compatibility issues occur as a result of inconsistent support of web programming languages across various web browsers.

X-Browser package will be available to install using Atom packages Installer. The web designers will have an option to launch various locally supported web browsers. The code typed in Atom will be immediately displayed in launched browsers.

1.3 Definitions, acronyms, and abbreviations

Atom: open-source text editor.
Web browser: a software application, e.g., Chrome, Firefox, Safari, MS Edge, and IE used to access information on the World Wide Web.
Software application: a type of computer program that performs a specific function.
Package: a software collection that provides certain functionality as part of a larger system.

Open-source: is a source code that is made freely available for possible modification and redistribution.
Web programming language: a formal language, e.g., HTML, CSS, and JavaScript, comprising a set of instructions involved in Web development.
Plugin: is a software component that adds a specific feature to an existing computer program.
URL: A Uniform Resource Locator also known as a web address, is a reference to a web resource that specifies its location on a computer network and a mechanism for retrieving it.

1.4 Organization

The rest of the SRS document is organized as follows:

• **Section 2** is an overall description of the project. It includes all properties, functions, constraints, assumptions, dependencies, and requirements of the product.

• **Section 3** cites the specific requirements for the X-Browser package.

• **Section 4** provides models and diagrams, e.g., case, class, and sequence, of the product.

• **Section 5** describes a prototype of the project. It outlines functionality using various example scenarios. As well as the instructions to download and launch the prototype.

• **Section 6** lists all references used in the creation of the product

• **Section 7** lists points of contact

2 Overall Description

This section will address information on how to use X-Browser. This information will also include the context in which X-Browser is created and used. It will also explain the interface of X-Browser and its functions. This section will also cover the minimum requirements to use the product and the constraints of X-Browser.

2.1 Product Perspective

While 70% of the world's desktop internet browser users use Chrome, the other 30% of browser users are split between Safari, Firefox, Edge, Opera, and Microsoft Internet Explorer [2]. This makes it necessary for web developers to support and test features on each of these browsers, which can slow down development and testing.

X-Browser is an extension for the Atom text editor. It aims to give front-end web developers the ability to work more efficiently by avoiding switching between browsers when testing website features. Instead of having to test each feature on each browser, X-Browser displays and updates a website on multiple browsers simultaneously. After hosting a local web server or opening a HTML file in Atom, a developer can open the X-Browser menu in Atom and select which browsers to launch. The launched browsers will display the specified file or URL, and update automatically whenever the site is updated or on a timer. The browser windows will be able to be stacked or tiled via drag-and-drop or shortcut keys. With this method front-end developers will save time testing and visualizing source code behavior on the different browsers.

An internet connection will not be required since browsers and local web servers can be installed and launched locally. X-Browser will only support major browsers such as Chrome, Firefox, Safari, MS Edge, and IE.

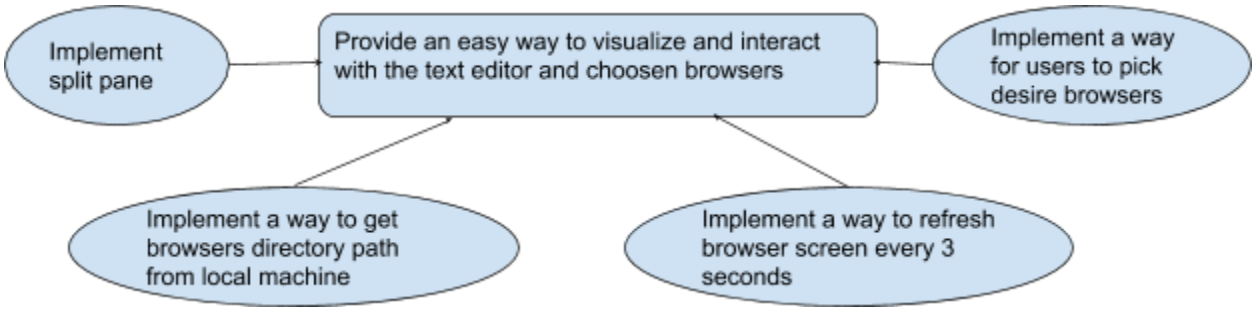
The X-Browser Atom plugin will be developed with JavaScript. Since it is an Atom plugin, it requires a system to be able to install and run Atom. Atom can be run on Windows, Linux or MacOS operating systems.

2.2 Product Functions

The major functions of X-Browser are listed below:

- Locate the browsers directory that are available on the local machine.
- Allow users to pick browsers they want to work on.
- Provide an interface that allows users to visualize their website on different browsers simultaneously.
- Refresh browsers every 3 seconds, for real-time editing and testing.

The diagram below shows the developing goal of X-Browsers. The



2.3 User Characteristics

The users for X-Browser are expected to be front-end web developers who need to test their website on multiple browsers. The user is expected to have working knowledge of the Atom interface, including installing Atom packages. For users that require processing asynchronous requests or the use of server-sided features, the user is expected to have a remote or local web server installed. Since a user may require the use of a specific web server, X-Browser does not include a specific web server. The user is expected to install and manage their own web server, if needed.

2.4 Constraints

Atom will be needed in order to use X-Browser, since Atom is only available for operating systems: macOS 10.9 or later, Windows 7 and later, and Linux.

Since X-Browser is a package on Atom, so it required some experience on using Atom to install the package, set browsers directory when X-Browsers can not detect desired browsers and set hotkeys for changing between browsers.

A properly functioning keyboard, mouse, and display screen are required.

127 MB of disk space is required to install Atom text editor.
Some prior software user knowledge is necessary to install and launch Atom and X-Browser.

An internet connection will not be required since browsers and local web servers can be installed and launched locally. X-Browser will only support major browsers such as Chrome, Firefox, Safari, MS Edge, and IE.

2.5 Assumptions and Dependencies

As Atom plugins are written in JavaScript, they can be installed and used on any system that supports the Atom editor. Atom runs on OS X 10.10 or later, Windows 7 or later, RedHat Linux, and Ubuntu Linux [3].

It is assumed that users have prior experience with Atom, therefore tutorial for interface will not be implemented. Only a brief explanation will be available on Github or Atom Install package page.

2.6 Apportioning of Requirements

This version of X-Browser is an Atom plugin, as the main features of X-Browser work well as an addition to an existing text editor. If needed, X-Browser could be remade as a standalone text editor. It would currently not be a good use of resources to create our own editor, as

X-Browser cannot be used to test websites as they would run on smartphones and tablets. A X-Browser smartphone application could be developed to enable real-time testing on smartphones or smartphone emulators.

If it would be useful for developers, a standard optional web server could be included with the plugin. This is not yet a feature, as it is assumed that most web developers will choose to use the remote/local web server of their choice.

3 Specific Requirements

1. X-Browser Package installation
    a. Package compatible with Linux, Windows, and macOS versions of Atom
    b. Self-installing X-Browser package
        i. Check for installed browsers on the local machine
            1. Add paths of found executables to the config file
            2. Populate available web browsers in the browser launch menu
        ii. Populate X-Browser Icon in Atom editor

2. X-Browser will support major web browser: Firefox, Google Chrome, Safari, MS Edge, Internet

3. Default layout of Atom and selected web browsers
    a. Divide display screen vertically in half
        i. Atom text editor occupies the left half of the display
        ii. Selected Web browsers occupy the right half of the display
            1. Divided horizontally equally between selected web browsers

4. User can navigate between web browser windows
    a. Switching between web browser windows by using hot-keys

5. Web browsers will update at real-time by refreshing the web page or tab every 3 seconds

6. User will be able to remove X-Browser package from Atom

7. X-Browser should check for updates
    a. Update automatically
    b. Update manually

8. X-Browser should be compatible with Atom updates

9. Web browser Window Management
    a. Should be able to relocate web browser window anywhere on the right side of the display
    b. Should be able to resize web browser window within the right side of the display

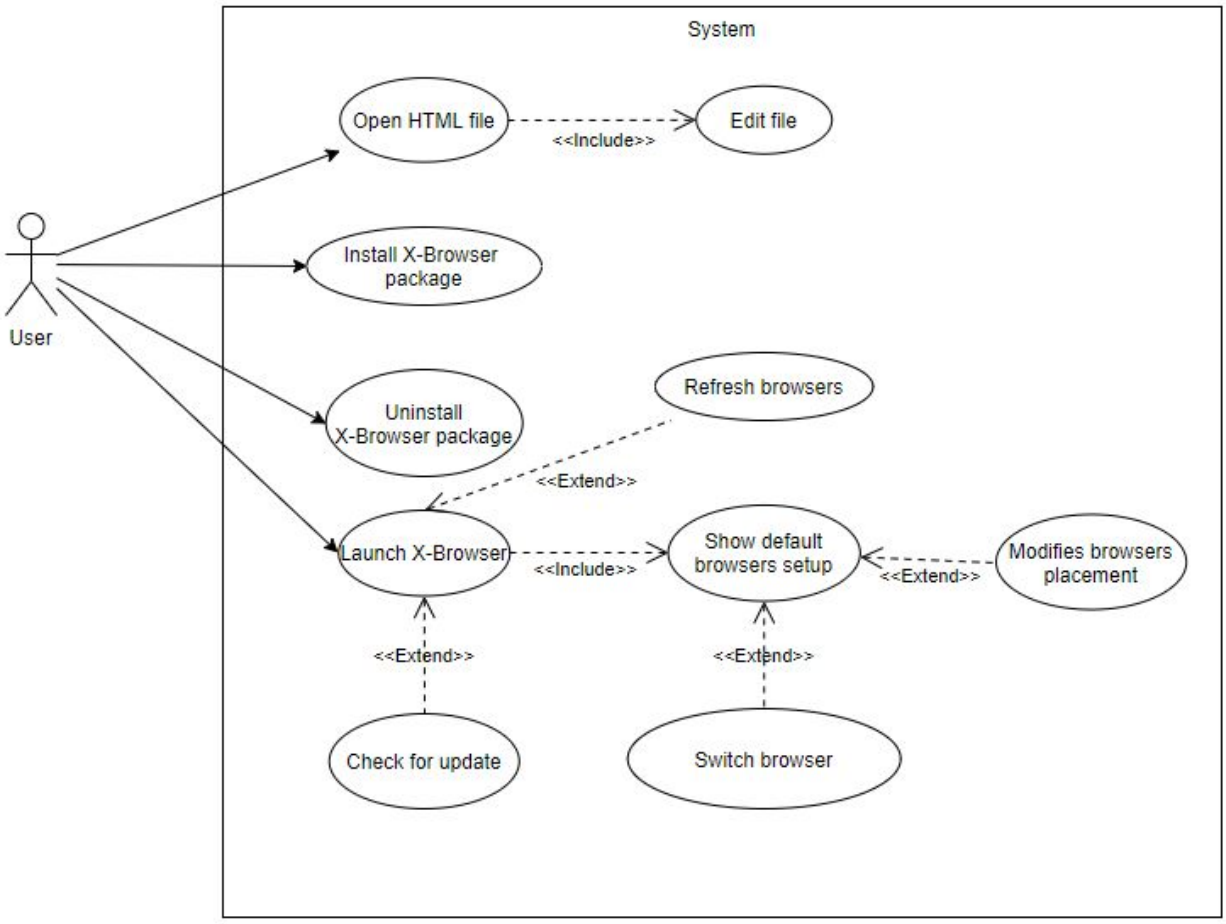
10. Atom must be launched in order to launch X-Browser

11. X-Browser must close when Atom text editor is closed

4 Modeling Requirements

Use Case Diagram:

This diagram comprises of multiple use cases and their relations towards each other



Use Case Name:	Open HTML file
Actors:	Users
Description:	User opens HTML file on Atom
Type:	Essential
Includes:	Edit file

Extends:	N/A
Cross-refs:	N/A
Uses cases:	N/A

Use Case Name:	Install X-Browser package
Actors:	Users
Description:	Installs X-Browser package from atom application
Type:	Primary
Includes:	N/A
Extends:	N/A
Cross-refs:	N/A
Uses cases:	N/A

Use Case Name:	Uninstall X-Browser package
Actors:	Users
Description:	Remove X-Browser package from Atom
Type:	Primary

Includes:	N/A
Extends:	N/A
Cross-refs:	N/A
Uses cases:	N/A

Use Case Name:	Launch X-Browser
Actors:	Users
Description:	Browser path is added to config file
Type:	Primary
Includes:	Show default browsers setup
Extends:	N/A
Cross-refs:	N/A
Uses cases:	N/A

Use Case Name:	Check for update
Actors:	N/A
Description:	Find the latest version of X-Browser to ensure X-Browser is up to date
Type:	Primary and Essential

Includes:	N/A
Extends:	Launch X-Browser
Cross-refs:	N/A
Uses cases:	N/A

Use Case Name:	Edit file
Actors:	N/A
Description:	Allows any opened html file to be edited
Type:	Primary and Essential
Includes:	N/A
Extends:	N/A
Cross-refs:	N/A
Uses cases:	N/A

Use Case Name:	Refresh browsers
Actors:	N/A
Description:	Give user a real-time testing experience by refreshing browser every 3 seconds

Type:	Essential
Includes:	N/A
Extends:	Launch X-Browser
Cross-refs:	N/A
Uses cases:	N/A

Use Case Name:	Show default browsers setup
Actors:	N/A
Description:	Browser setup is set to default when X-Browser is open
Type:	Primary and Essential
Includes:	N/A
Extends:	N/A
Cross-refs:	N/A
Uses cases:	N/A

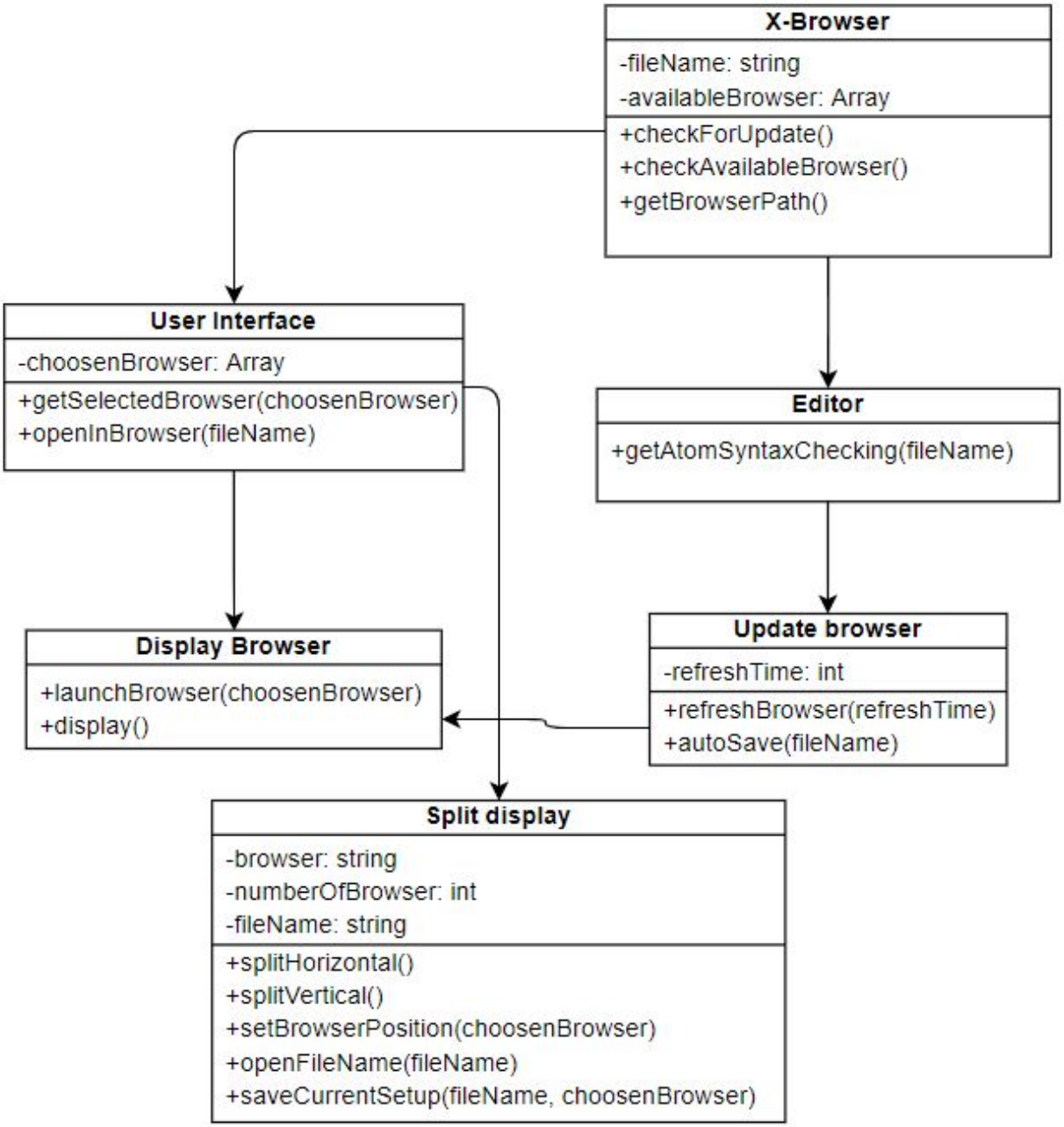
Use Case Name:	Modifies browsers placement
Actors:	N/A

Description:	Allows configuration of browser placement when multiple browsers are open
Type:	Essential
Includes:	N/A
Extends:	Show default browsers setup
Cross-refs:	N/A
Uses cases:	N/A

Use Case Name:	Switch browsers
Actors:	N/A
Description:	Opened browsers can be switched with configured hotkeys
Type:	Primary
Includes:	N/A
Extends:	Show default browsers setup
Cross-refs:	N/A
Uses cases:	N/A

Class Diagrams:

The following diagram represents the main classes that are essential to the use of the X-browser software.



Below is a brief description of each class and their functions.

Class Name	Description
X-Browser	The X-Browser class will be used in order to handle all the maintenance functions of the plugin. It will handle updating, and setting up the correct files for the use of the user.
Attributes: fileName:String	fileName is a string variable that will be used in order to save the name of the file used in Xbrowser.
availableBrowser:Array	availableBrowser will be an array that will keep track of the available browsers that the user may use in order to display their file.
Operations: checkForUpdate()	checkForUpdate() is a function that will automatically check to see if there is an update available for the X-browser plugin.
checkAvailableBrowser()	checkAvailableBrowser() is a function that will return whether or not a specific browser is available for use by the X-browser plugin.
getBrowserPath()	getBrowserpath() is a function that will return the path to which the browser is stored on the local system.
Relationships:	The X-Browser class will interact with the User interface class and the editor class. The User Interface class will access the checkAvailableBrowser function of the X-Browser class and use that to open only valid browsers.
UML Extensions:	None.

Class Name	Description
User Interface	The User Interface class will be used in order to set up all interactions done by the user. It will take input from the user and display the desired output.

Attributes: chosenBrowser:array	chosenBrowser will be an array of all the browsers that the user has selected to display their code.
Operations: getSelectedBrowser(chosenBrowser)	getSelectedBrowser(chosenBrowser) will be a function that will retrieve all information needed in order to open the chosenBrowser.
openInBrowser(fileName)	openInBrowser(fileName) is a function that will take a file name as a parameter, and open the desired file in the browsers specified by chosenBrowser.
Relationships:	The User Interface class will use functions from the X-Browser class in order to verify which browsers may be opened, as well as from displayBrowser in order to properly display which browsers are available. It will also interact with the Split Display class in order to assist with the proper display the user desires.
UML Extensions:	None

Class Name	Description
Display Browser	Display Browser will be a simple class that will be used for the sole purpose of launching the selected browsers designated by the user.
Attributes: launchBrowser(chosenBrowser)	launchBrowser() is a function that will take the chosenBrowser, and launch the browser using the function getBrowserPath().
Relationships:	The display browser class will be associated with the user interface class in order to properly display which browsers the user wishes. It will also be used by the update browser class in order to properly display the correct, updated data from the file that is desired.
UML Extensions:	None

Class Name	Description
Split display	Split Display is a class that will be used in order to handle all of the display functions of the plugin. It will handle the modifications of the orientation of the plugin in order to save the preferences of the user.
Attributes: browser:string	Browser will be a string that will be used in order to modify the display of the browsers in use.
numberOfBrowser:int	numberOfBrowser is an integer variable that will be used in order to keep track of the number of browsers the user wishes to use.
fileName: string	fileName is a string variable that will be used to keep track of the name of the file that the user is currently working on.
Operations: split_horizontal()	split_horizontal() will be a function that will split the current browsers in a horizontal manner.
split_vertical()	split_vertical() will be a function that will split the current array of browsers in a vertical manner.
setBrowserPosition(chosenBrowser)	setBrowserPosition is a function that will be used in order for the user to display the browsers in whatever orientation that they wish.
openFileName(fileName)	openFileName(fileName) is a function that will take a fileName as a parameter, and use this in order to open the desired file designated by the user.
saveCurrentSetup(fileName, chosenBrowser)	saveCurrentSetup(fileName, chosenBrowser) is a function that is used in order to store the current display setup, in order for the user to use in the future.
Relationships:	The split browser will be used in relation with the user interface class in order to properly display the proper interface that the user wishes to use.

UML Extensions:	None

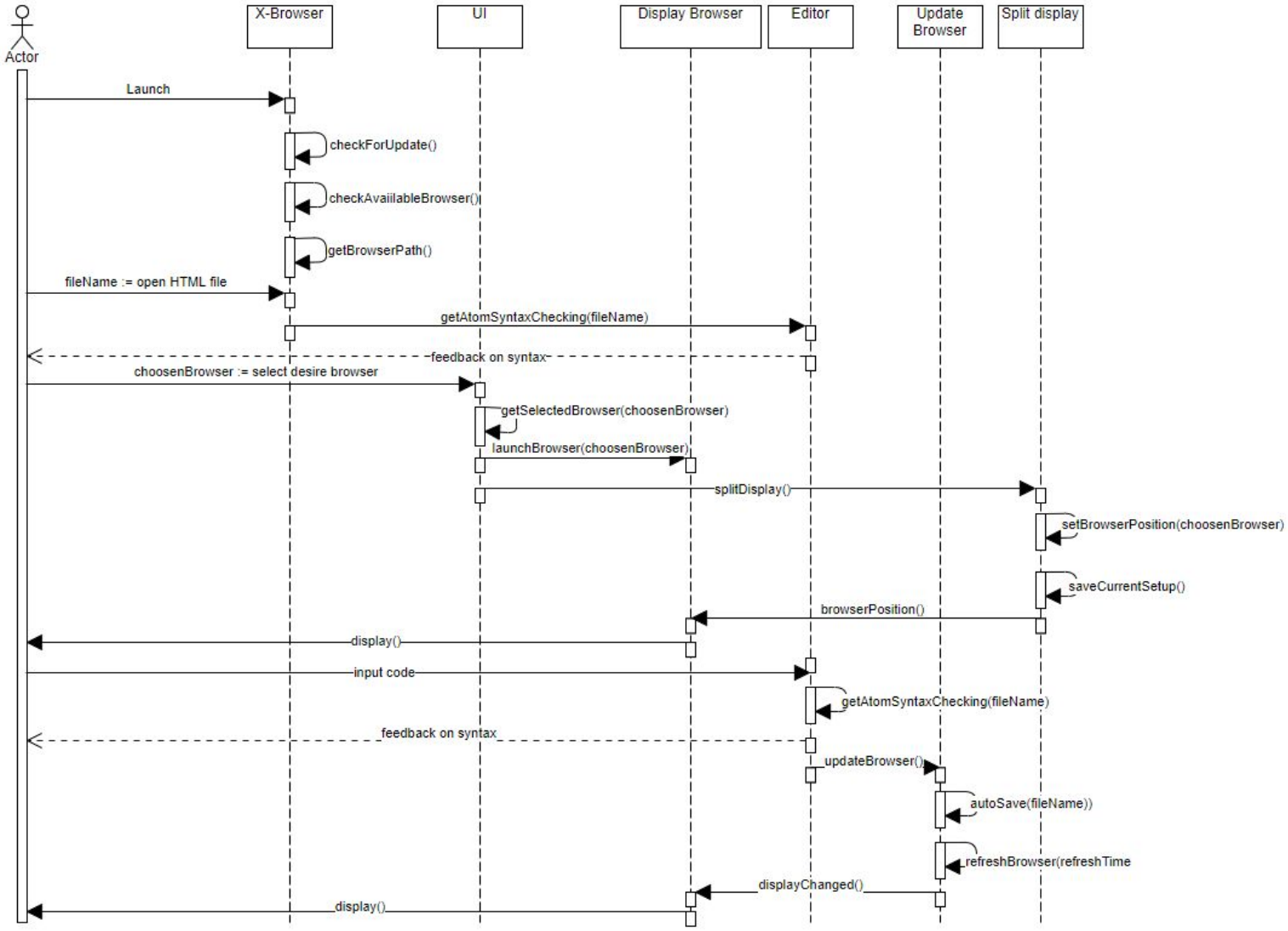
Class Name	Description
Update Browser	Update Browser is a class that will handle all the updating between the browsers and filenames.
Attributes: refreshTime: int	refreshTime will be an integer variable that will be used in order to save the current file and display in real time what is on the file.
Operations: refreshBrowser(refreshTime)	refreshBrowser(refreshTime) is a function that will update the browser based on the refreshTime in integers.
autosave(filename)	autosave(filename) is a function that will update any changes made to the selected filename.
Relationships:	The Update Browser class will be used in relation with the display browser class in order to display the most recently updated file in the selected browsers.
UML Extensions:	None.

Class Name	Description
Editor	Editor will be a simple class that will be used in order to ensure that the syntax in the file is correct.
Operations: getAtomSyntaxChecking(fileName)	getAtomSyntaxChecking(fileName) is a function that will take a fileName as a parameter, and scan the file in order to use atom to check the syntax in the file.
Relationships:	The Editor class will borrow attributes from the X-Browser class to acquire the filename into the atom editor. This class will be used for the Update Browser class in order to keep all the data up to date onto the editor.

UML Extensions:	None

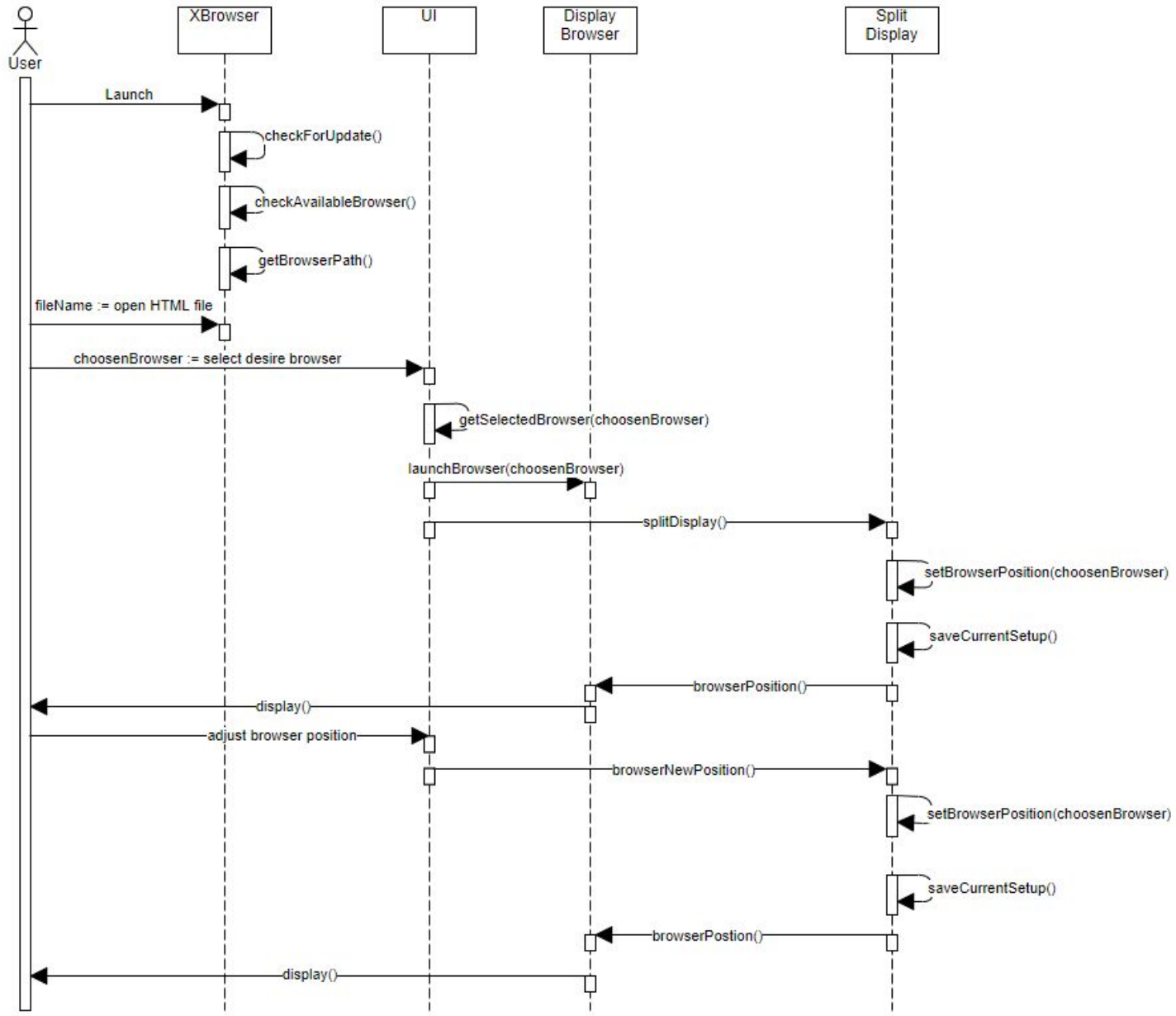
Representative Scenarios of the System:

Upon loading up the atom application the user may choose to run the X-Browser extension, which will then prompt them with the default UI, requesting they input which browsers they would like to open and edit on. They will then have the ability to open a new file, or an existing file from which the program will display. The user can then edit this file and experience real-time updating of the website on the right hand side of the program. Upon completion of their website, or their session, the user may then save all progress and exit the program accordingly.
Sequence Diagram of this scenario:

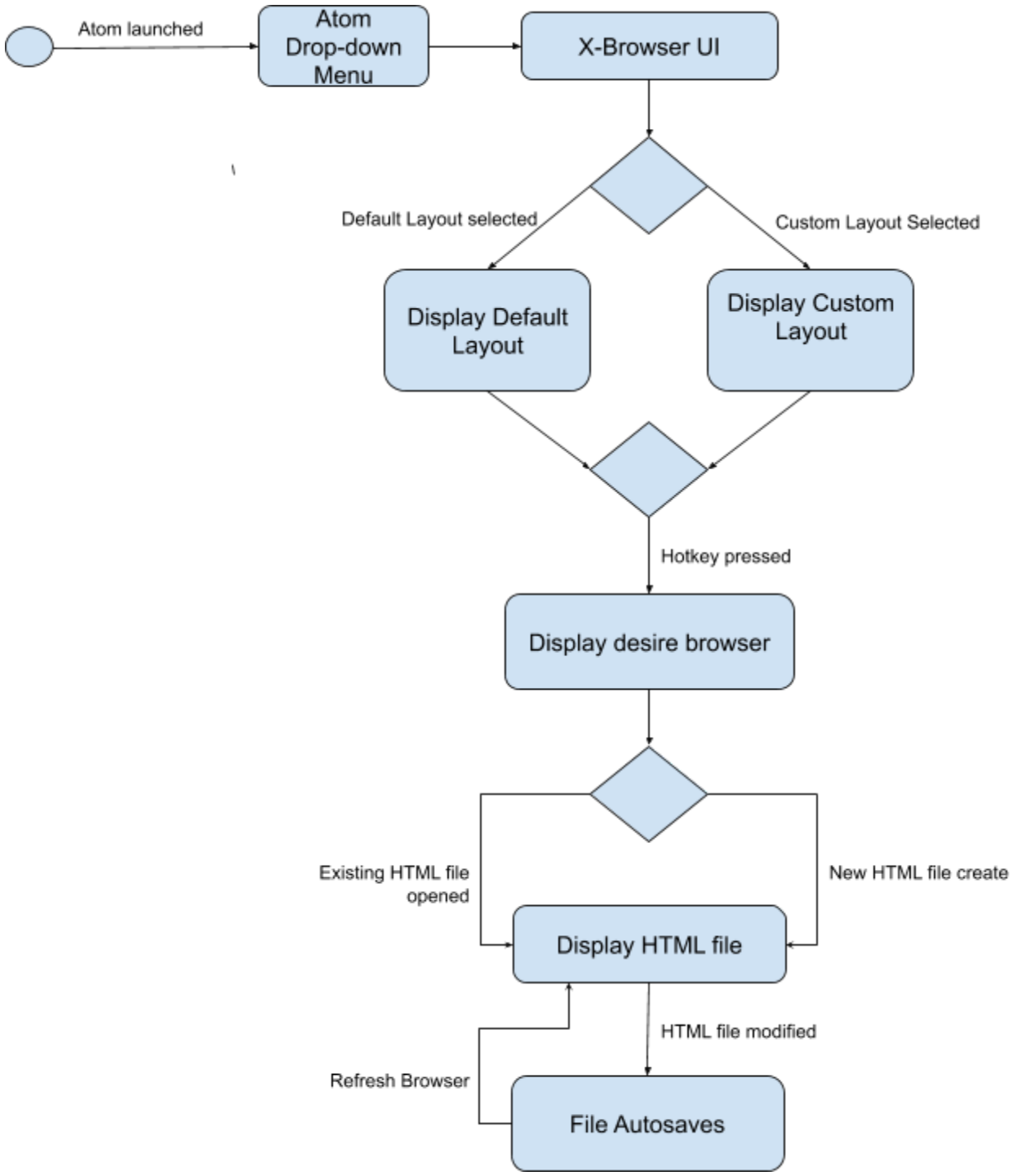


An existing user may launch the X-Browser plugin and in this case they can select to resume their previous work. If this is selected, the plugin will then open their last used file, as well as their last used browser layout. The user may then choose to add new browsers to the existing display.

This is a sequence diagram to show how this would work:



Below is a State Diagram that represents each state that our software can be in, and the transitions that make it go from state to state. The states are represented by rectangles, and the transitions represented by arrows. The diamonds represent areas where users are given a choice. Each transition requires some sort of input from the user.



5 Prototype

The prototype will demonstrate the implementation of X-Browser in the environment of Atom desktop text editor. The potential users will be able to experience the interface and features of X-Browser.

5.1 How to Run Prototype

System Requirements:

- Operating system: macOS 10.9 or later, Windows 7 and later, and Linux.

Network Requirements:

- A working internet connection is required to download Atom text editor.

Installing Atom desktop text editor:

- Follow instructions here to download and install Atom desktop text editor.

Installing X-Browser package in Atom:

1. After installing Atom, clone the git repository anywhere

`Repository: https://github.com/wxuhao/x-browser-prototype`

2. Change the working repository to the freshly made x-browser-prototype directory
3. Run `apm link` to create a symbolic link to the ~/.atom/packages directory
4. Refresh Atom

Using X-Browser:

- Once a package is installed in Atom, it will show up in the Settings View under the "Packages" tab as ‘x-browser-prototype’ (*Figure 5.2.2.* See section 5.2 below*)*.
- To launch a web browser select ‘Select browsers to launch’ and mark web browsers desired before clicking the ‘Launch’ button (*Figure 5.2.3.* See section 5.2 below*)*

5.2 Sample Scenarios

The initial screen after launching Atom text editor:

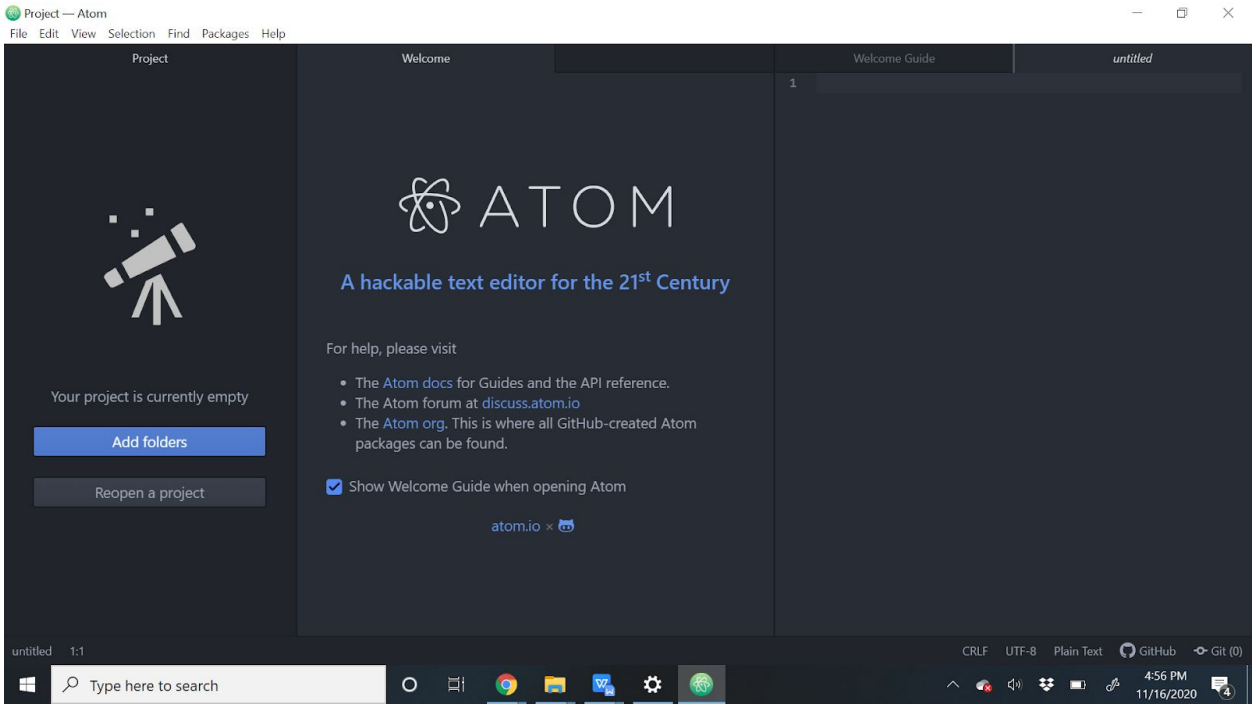


Figure 5.2.1

X-Browser in the ‘Packages’ menu:

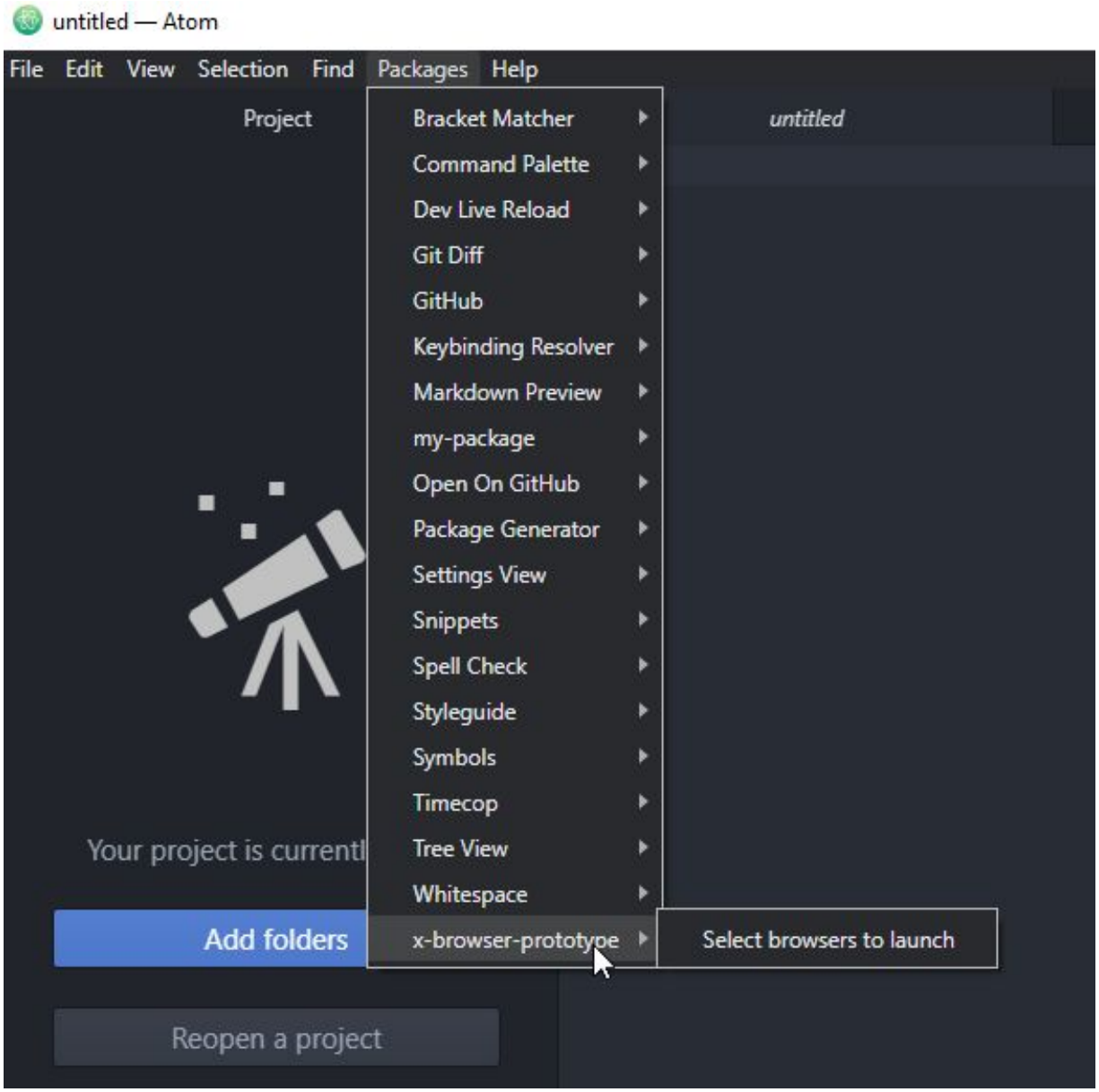


Figure 5.2.2

The ‘Select browsers’ menu for launching desired web browsers:

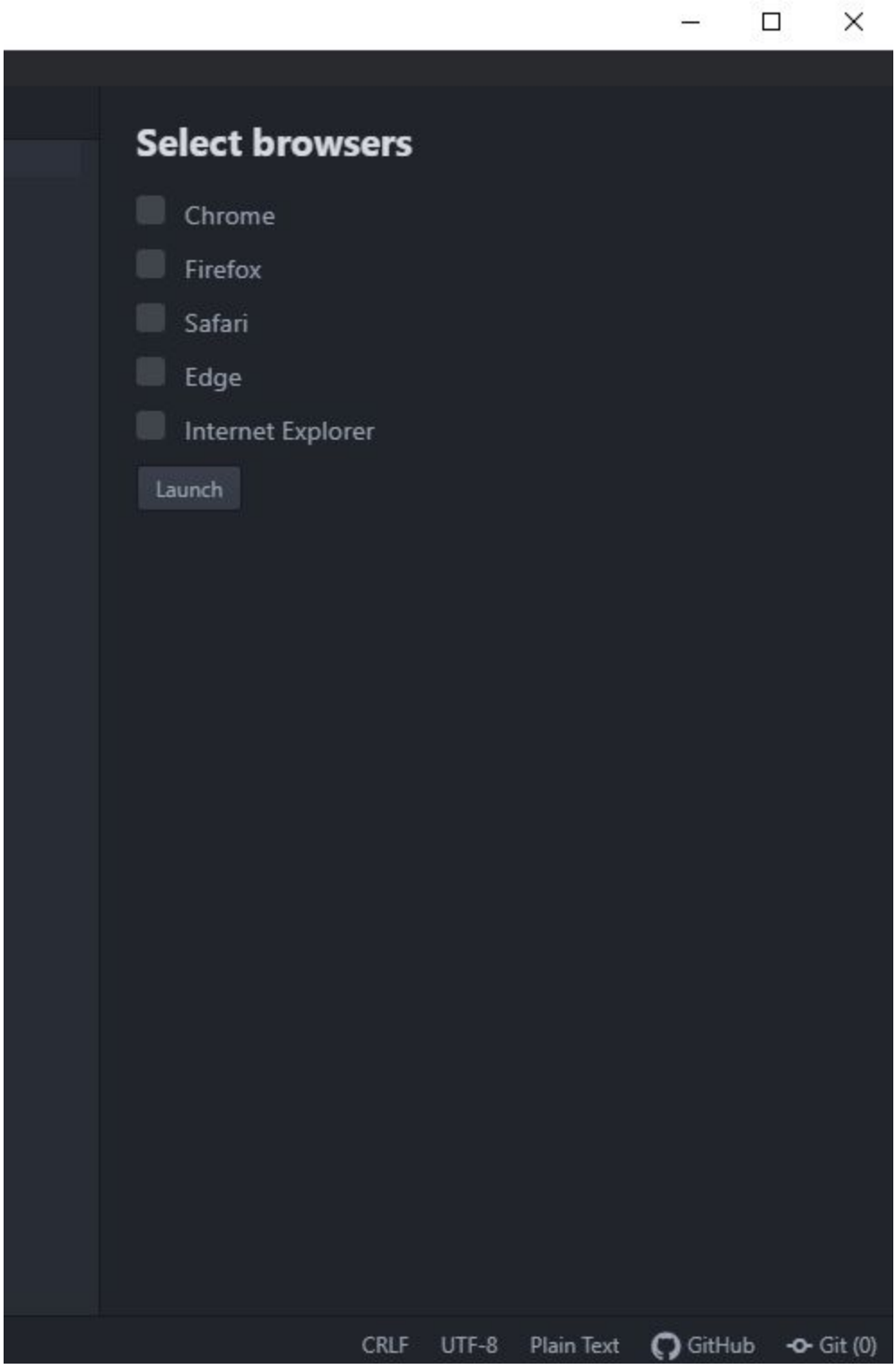


Figure 5.2.3

6 References

[1] D. Thakore and S. Biswas, “Routing with Persistent Link Modeling in Intermittently Connected Wireless Networks,” Proceedings of IEEE Military Communication, Atlantic City, October 2005.

[2]https://gs.statcounter.com/browser-market-share/desktop/worldwide#monthly-202009-202009-map

[3]https://flight-manual.atom.io/faq/sections/what-platforms-does-atom-run-on/

7 Point of Contact

For further information regarding this document and project, please contact **Prof. Daly** at University of Massachusetts Lowell (james_daly@uml.edu). All materials in this document have been sanitized for proprietary data. The students and the instructor gratefully acknowledge the participation of our industrial collaborators.