Building a Website Blocker Extension in Minutes

This tutorial explains how you can build a website blocker extension that will run on multiple browsers.

Web browsers have become integral to our lives, and are used on a wide range of devices, including desktops, laptops, tablets and smartphones. In 2019, an estimated 4.3 billion people used Web browsers. The most used browser today is Google Chrome, with a 64 per cent global market share across all devices, followed by Apple Safari with 17 per cent.

A few popular Web browsers

- **Google Chrome**: This popular cross-platform Web browser has been developed by Google. It was first released in 2008 for the Windows platform and was later ported to Linux, MacOS, iOS and Android. You can download Google Chrome from *https://www.google.com/chrome/*.
- **Opera**: This is a freeware browser for Windows, Android, iOS, MacOS and Linux, developed by Opera Software. It is a Chromium based browser using the Blink layout engine. What differentiates it from other browsers is a unique user interface, and a few other features. You can download Opera from *https://www.opera.com/hi/download*.
- **Brave**: This is a free and open source Web browser developed by Brave Software Inc., based on the Chromium framework. This browser has ad-blocking and website tracker-blocking features inbuilt. It also allows users to send cryptocurrency contributions in the form of Basic Attention Tokens to websites and content creators. You can download Brave from *https://brave.com/download/*.
- **Microsoft Edge**: This is a Web browser developed by Microsoft. It was rebuilt as a Chromium based browser in 2019. It is also available for Windows, Android, iOS and MacOS. You can download Microsoft Edge from *https://www.microsoft.com/en-us/edge*.

What is a browser extension?

A browser extension is a small software module that customises or adds extra functionality to the browser. This extension enhances or adds some extra features to the browser. The extensions are mostly written using Web technologies like HTML, JavaScript and CSS. They are hosted on vendor specific stores. For example, Chrome extensions are hosted on the Chrome Web Store, Mozilla Firefox hosts its extensions on its extension store, and so on. Users can go to the store of the browser they are using to download and install any of the wide variety of browser extensions available there.

Creating an extension

We are now going to build a website blocker extension, which will run on multiple Web browsers (Google Chrome, Opera, Brave and Microsoft Edge). This extension will allow users to block websites in a browser. Users can add the website they want to block to the list. When they try to visit that site, a message that states that the website is blocked will be displayed, and access to it will be denied. If the users then want to visit the site, they will have to unblock it by deleting the website entry from the list of blocked sites in the extension.

We are going the use the following Web technologies to build this extension:

- HTML to create a table to view blocked sites, and buttons to add and delete them
- CSS to style HTML
- JavaScript (Chrome browser API) to write our logic

Setting up the development environment

Let us now set up the file structure for our browser extension. Create a new folder and give it a name, say 'web-blocker'. Now, inside that folder, create the following empty files:

- *manifest.json*
- *options.html*
- *script.js*
- *block.js*
- *style.css*

You should end up with the directory structure shown in Figure 1.

- *manifest.json* is a metadata file in JSON format. It contains basic information about the extension, like its name, a description of it, the version number, which script it should run on starting up, etc.
- *options.html* displays the UI of the extension, like the list of websites to be blocked, buttons to add and delete, websites from the list, etc.
- *script.js* controls the interactions and the logic of the *options.html* file.

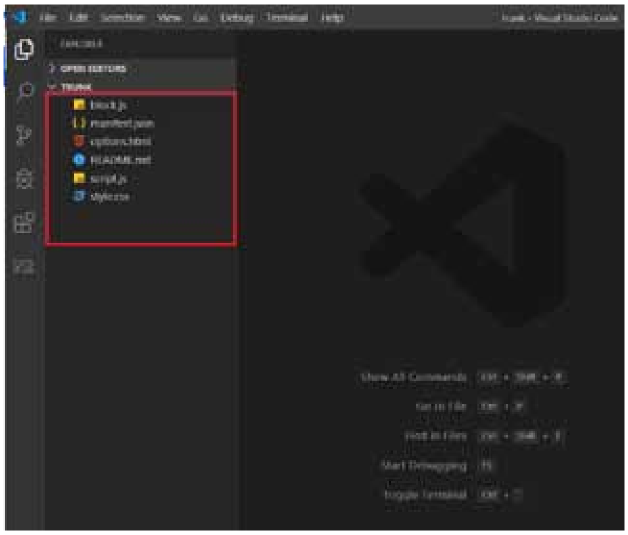


Figure 1: Extension project file structure

- *block.js:* In this file we are going to intercept the outgoing request from the browser. If the request contains one of the blocked websites from the list, we will reject the request, thus blocking that website.
- *style.css* adds styles to the *options.html* page to make it look good.

Creating the manifest file

Open the *manifest.json* file and copy the following JSON code:

```
{
    "name": "Website Blocker",
    "version": "1.0.0",
    "description": "Blocks websites based on domain names",
    "background": { "scripts": ["block.js"] },
    "options_page": "options.html",
    "permissions": [
        "webRequest", "webRequestBlocking",
        "http://*/*", "https://*/*",
        "storage"
    ],
    "manifest_version": 2
}
```

As you can see, this code is in JSON format. It consists of general information about the browser extension, such as its version, name, description, permissions required for the extension, and so on.

The parameter description is as follows:

- *manifest_version* specifies the version of the manifest; the value must be 2
- *name* specifies the name of the extension
- *version* specifies the version of the extension
- *description* specifies a short description of the extension
- *background* specifies that the script *block.js* should load as soon as the user starts the browser
- *options_page* specifies the UI page to be displayed for extension options
- *permission* specifies which browser APIs are to be used. We are using *webRequest*, *webRequestBlocking*, *https://*/**, *http://*/** and *storage*

Note: It is assumed that you have some basic knowledge of Web technologies like HTML, CSS and JavaScript. If you don't, W3Schools (*http://www.w3schools.com/)* is a good place to start. The site has some great tutorials for Web technologies that are easy to follow.

To add HTML, open the *options.html* file located in the extension directory, and add the following HTML code:

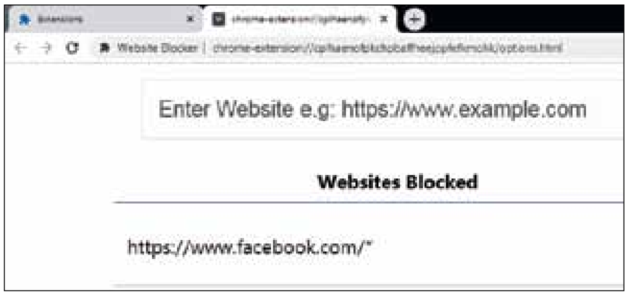
```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title></title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width,
initial-scale=1">
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <div class="header">
        <input class="add-website" type="url"
placeholder="Enter Website e.g: https://www.example.com">
        <button id="add">+</button>
    </div>
    <div class="main">
        <table>
            <tr>
                <th>Websites Blocked</th>
            </tr>
            <tr id="no-website-msg">
                <td>No websites blocked yet.</td>
            </tr>
        </table>
    </div>
    <script src="script.js" defer></script>
</body>
</html>
```

In the above HTML code, we have added the following elements:

- <*input*> - The user will enter the website to be blocked in this input field
- <*button*> - To add and delete websites added to the list
- <*table*> - To display the list of blocked/added websites

To add CSS styles, use the following code:

```
body {
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-
serif;
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: column;
}

.add-website {
    width: 300px;
    padding: 10px;
    margin: 10px;
    border-radius: 4px;
    outline: none;
    border: 1px solid #eee;
}

#add {
    border: none;
    border-radius: 50%;
    background-color: #0081cb;
    color: #ffffff;
    cursor: pointer;
    width: 26px;
    height: 26px;
    cursor: pointer;
}

.delete {
    border: none;
    border-radius: 50%;
    background-color: #e6463b;
    color: #ffffff;
    cursor: pointer;
    width: 26px;
    height: 26px;
    cursor: pointer;
    float: right;
    margin-right: 6px;
    margin-top: 10px;
}

table {
    height: auto;
    width: 388px;
}

table tr th {
    border-bottom: 1px solid #0081cb;
    padding: 5px;
}

table td {
    padding: 8px;
    line-height: 32px;
    width: 100%;
}

table td {
    border-bottom: 1px solid #e2e2e2;
}
```



Enter Website e.g: https://www.example.com

Websites Blocked

https://www.facebook.com/*

Figure 2: Website blocker user interface

This CSS will style the UI as shown in Figure 2.

To add JavaScript, open the *script.js* file and copy the following JavaScript code:

```
let urls = []; // Global variable to store URLs

// Load init when the content is loaded.
window.addEventListener('DOMContentLoaded', init);
window.addEventListener('DOMContentLoaded', loadList);

//Init Function - Initialize, load, addEventListeners
function init() {
    const addButton = document.getElementById('add');
    const deleteButton = document.getElementsByClassName('de
lete');
    const noWebsiteMsg = document.getElementById('no-website-
msg');
    chrome.storage.local.getBytesInUse(['websites'], function
(bytes) {
        if (bytes) {
            chrome.storage.local.get("websites", function
(res) {
                if (res != {}) {
                    urls = JSON.parse(res.websites);
                    if (urls == undefined || urls.length ==
0) {
                        noWebsiteMsg.style.display = "";
                    } else {
                        noWebsiteMsg.style.display = "none";
                    }
                    if (deleteButton) {
                        for (let i = 0; i < deleteButton.
length; i++) {
                            deleteButton[i].
addEventListener('click', function (e) {
                                deleteWebsite(e);
                            });
                        }
                    }
                }
            });
        } else {
            urls = [];
        }
    });
    addButton.addEventListener('click', function () {
        const url = document.getElementsByClassName('add-
website')[0].value;
        if (url.length > 0) {
            noWebsiteMsg.style.display = "none";
            urls.push(url + "/*");
            chrome.storage.local.set({
                "websites": JSON.stringify(urls)
            }, function (value) {
                console.log(value);
            });
            save();
            loadList();
            document.getElementsByClassName('add-website')
[0].value = "";
        }
    });
}

function loadList() {
    const deleteButton = document.getElementsByClassName('de
lete');
    const tblNode = document.getElementsByTagName('body')[0];
    let tblRow = document.createElement('tr');
    let tblData = document.createElement('td');
    let tblButton = document.createElement('button');
    let storedURLs = [];
    chrome.storage.local.getBytesInUse(['websites'], function
(bytes) {
        if (bytes) {
            chrome.storage.local.get("websites", function
(res) {
                if (res != {}) {
                    storedURLs = JSON.parse(res.websites);

                    if (storedURLs != undefined || storedURLs
!= []) {
                        for (let i = 0; i < storedURLs.
length; i++) {
                            tblButton.innerText = "-";
                            tblButton.setAttribute('class',
'delete');
                    tblRow.setAttribute('id', "row" + i);
                    tblButton.setAttribute('id', i);
                            tblData.innerText =
storedURLs[i];
                            tblRow.appendChild(tblData);
                            tblRow.appendChild(tblButton);
                            document.
getElementsByTagName('table')[0].appendChild(tblRow);
                            if (deleteButton) {

                                deleteButton[i].
addEventListener('click', function (e) {
                                    deleteWebsite(e);
                                });

                            }
                        }
                    }

                }
            });

        } else {
            storedURLs = [];
        }
    });

}

function deleteWebsite(e) {
    urls.splice(e.target.id, 1);
    chrome.storage.local.set({
        "websites": JSON.stringify(urls)
    }, function () {});
    if (document.getElementById("row" + e.target.id))
        document.getElementById("row" + e.target.id).
remove();
    save();
    init();
}

function save() {
    chrome.extension.sendRequest({
        urls: "save"
    }, function (response) {

    });
}
```

Here, we have declared four functions: *init()*, *loadList()*, *deleteWebsite()* and *save()*.

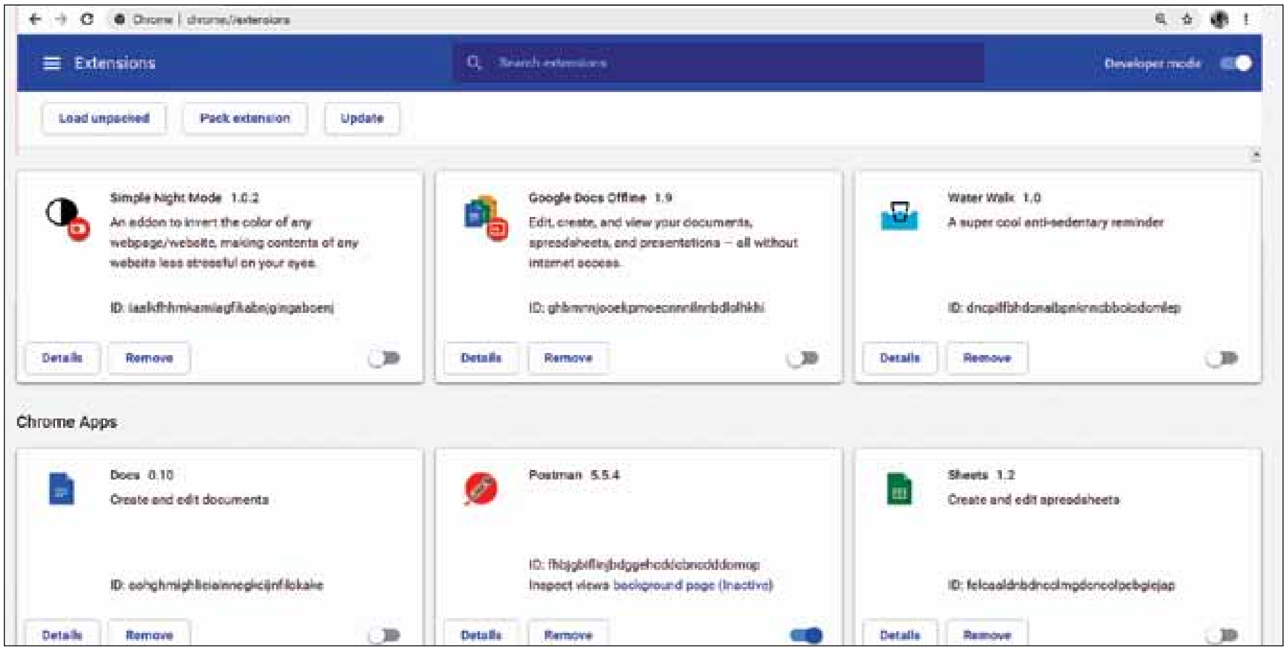
- *init()* initialises all the HTML elements, and here we can also add event listeners for buttons
- *loadList()* loads the list of websites we have added/deleted to and from the list
- *deleteWebsite()* deletes the website from the list
- *save()* saves the websites we have added to the list, into storage

We have used Chrome's storage API to store the website list.

Now open the *block.js* file and copy the following JavaScript code:

```
function blockRequest() {
    return {
        cancel: true
    };
}
chrome.extension.onRequest.addListener(
    function (request, sender, sendResponse) {
        if (request.urls == 'save') {
            let getStoredURLs = [];
            chrome.storage.local.getBytesInUse('websites',
function (bytes) {
                if (bytes) {
                    chrome.storage.local.get("websites",
function (res) {
                        if (res != {}) {
                            getStoredURLs = JSON.parse(res.
websites);
                            if (getStoredURLs.length > 0) {
                                const filter = {
                                    urls: getStoredURLs,
                                };
                                chrome.webRequest.
onBeforeRequest.addListener(
                                    blockRequest,
                                    filter, ["blocking"]
                                );
                            } else if (getStoredURLs.length
== 0) {

                                if (chrome.webRequest.
onBeforeRequest.hasListener(blockRequest)) {
                                    chrome.webRequest.
onBeforeRequest.removeListener(blockRequest);
                                }
                            }
                        }
                    });
                } else {
                    getStoredURLs = false;
                }
            });
        }
    }
);
```

Here, we intercept the Web request made using *chrome.*



Simple Night Mode 1.0.2
An addon to invert the color of any webpage/website, making contents of any website less stressful on your eyes.
ID: iaslkfhhmkamiagfikabnjgingaboenj

Google Docs Offline 1.9
Edit, create, and view your documents, spreadsheets, and presentations — all without internet access.
ID: ghbmnnjooekpmoecnnnilnnbdlolhkhi

Water Walk 1.0
A super cool anti-sedentary reminder
ID: dncpilfbhdonalbpnknncbbokdomlep

Docs 0.10
Create and edit documents
ID: aohghmighlieiainnegkcijnfilokake

Postman 5.5.4
ID: fhbjgbiflinjbdggehcddcbncdddomop

Sheets 1.2
Create and edit spreadsheets
ID: felcaaldnbdncclmgdcncolpebgiejap

Figure 3: Extension loader interface to load/unload extensions

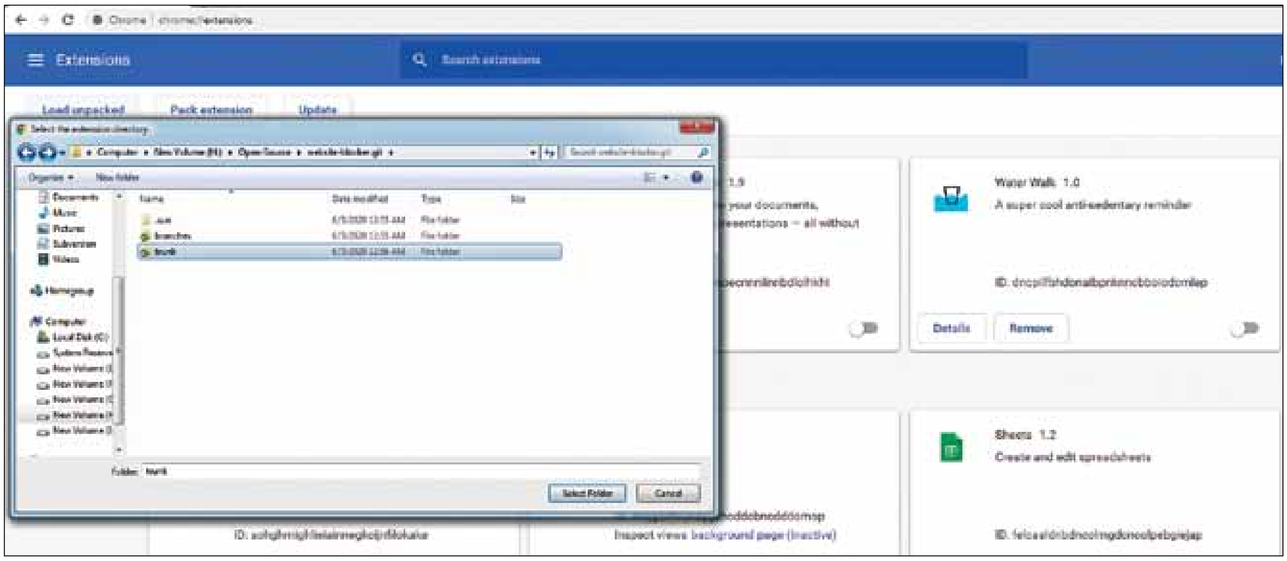


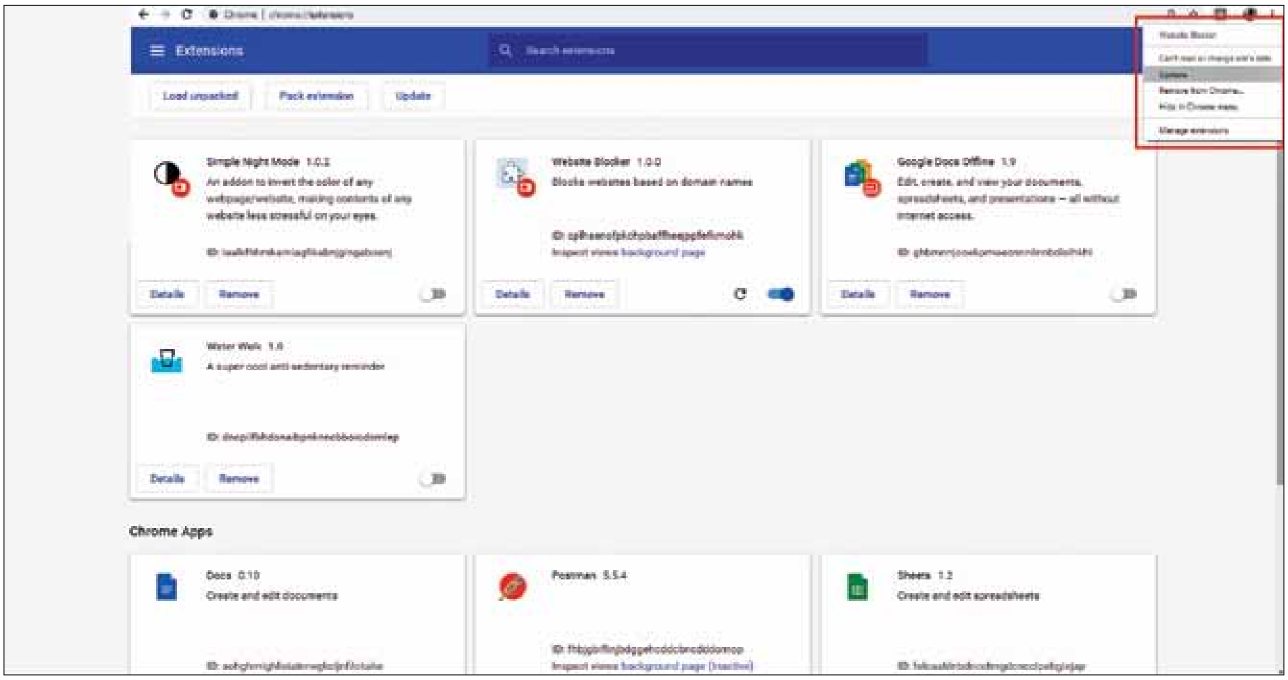
Figure 4: Folder selection dialogue box to select our extension

webRequest.onBeforeRequest API and check if it matches the list of websites that we have stored. If it does, we block the request; else, we allow it.

Loading the extension

First, let us load the extension in the Chrome browser.

1. Open up the Chrome menu by clicking the icon that's to the right of the address bar and select *Extensions* under the *Tools* menu or just enter *chrome://extensions* in the address bar before hitting *Enter*, as shown in Figure 3.
2. Check the *Developer mode* checkbox in the top right-hand corner, if it hasn't been checked.
3. Now, click on *Load unpacked*, which will open up the *file-selection* dialogue box, as shown in Figure 4.
4. Navigate to the directory where your extension files are and select it. Another easy way is to simply drag and drop the extension folder onto *chrome://extensions* in your browser to load it. The extension will install/ load right away. If you make some errors while creating the extension, it will not load and will display an error message at the top of the page. You have to rectify the error and then try again.



Simple Night Mode 1.0.2
An addon to invert the color of any webpage/website, making contents of any website less stressful on your eyes.

Website Blocker 1.0.0
Blocks websites based on domain names

Google Docs Offline 1.9
Edit, create, and view your documents, spreadsheets, and presentations — all without internet access.

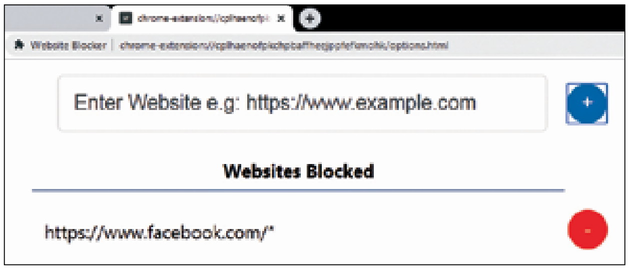
Water Walk 1.6
A super cool anti-sedentary reminder

Docs 0.10
Create and edit documents

Postman 5.5.4

Sheets 1.2
Create and edit spreadsheets

Figure 5: Our extension in the top-right corner of the browser

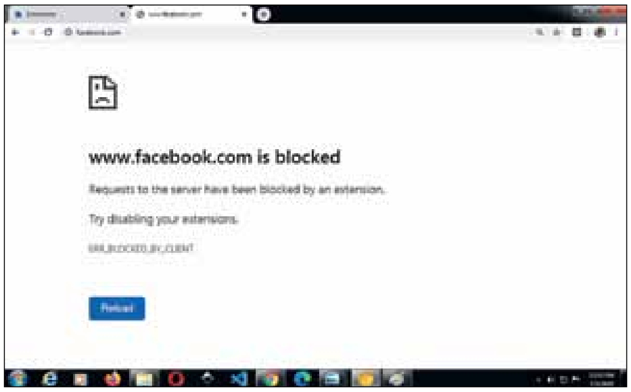


Enter Website e.g: https://www.example.com

Websites Blocked

https://www.facebook.com/*

Figure 6: Website blocker list shown by the extension



www.facebook.com is blocked

Requests to the server have been blocked by an extension.

Try disabling your extensions.

ERR_BLOCKED_BY_CLIENT

Figure 7: The extension blocking the request to a website

Running the extension

After installing/loading the extension, you can test it by right-clicking the extension icon shown at the top-right corner near the address bar, as shown in Figure 5. This will show the context menu. Then select *Options*, which will open the options page. Here you can add the website you want to block. As an example, I added *https://www.facebook.com* to the list, as shown in Figure 6; so as soon as I try to visit *https://www.facebook.com*, the request gets blocked and a message is displayed, as shown in Figure 7. Now, the website is blocked, unless I delete it from the list.

Testing the extension in the Opera, Brave and Microsoft Edge browsers

In Opera, navigate to *opera://extensions*. In Brave, navigate to *brave://extensions* and in Edge, navigate to *edge://extensions*. Then, enable *Developer Mode* and follow the same steps that you did for Google Chrome.

Note: You can also refer to and download this project's source code from my GitHub repository. Visit *https://github.com/aniketkudale/website-blocker.*

END

References

[1] *https://www.google.com/chrome/*
[2] *https://developer.chrome.com/extensions/api_index*

By: Aniket Eknath Kudale

The author is a senior member of the technical staff at TIBCO Software Inc., Pune, with more than five years' experience. His interests include Web technologies, computer vision and security.