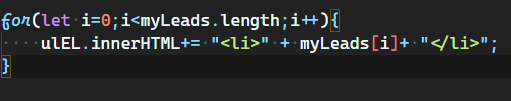
1. Create the input box and save button.
2. Add event listener to the save button and console log (button clicked).
3. Create two variables: myLeads-> should be assigned to an empty array.

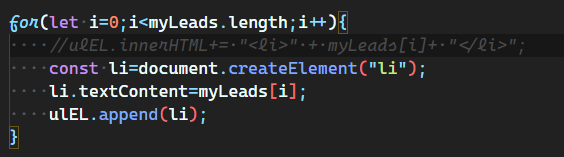
inputEL-> should be assigned to the text input field.

1. Try to fetch the text from input box and console log it when save button is clicked.
2. Create unordered list **(id=” ul-el”)** in HTML.
3. Grab the unordered list in JavaScript in variable **ulEL.**
4. Store myLeads in the unordered list and display it in browser.



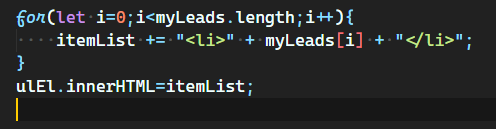
This method is for beginner another efficient way is:

* create Element.
* Set the text content
* Append to unordered list.



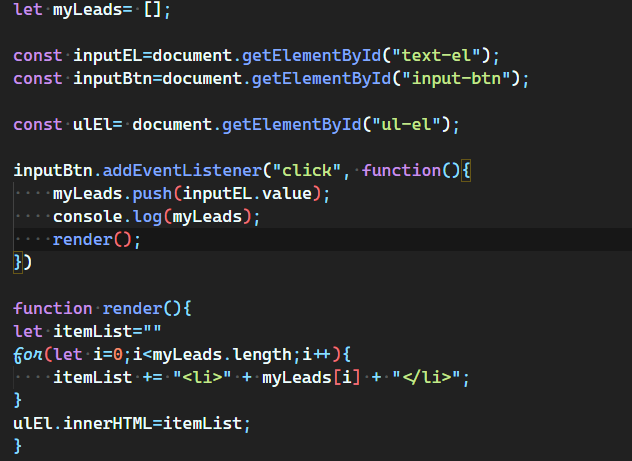
Create empty string variable name itemList=””

And insert the leads to itemList and show them on the unordered list.



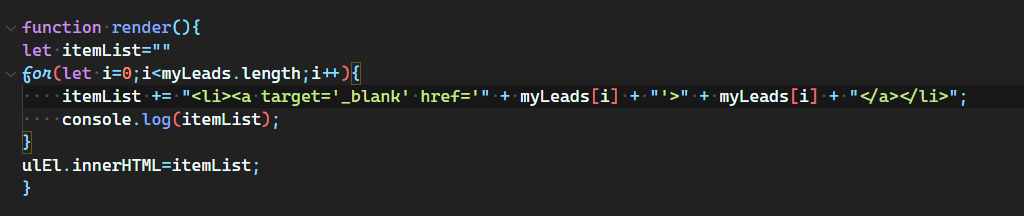
1. Now create the render function and clean the dummy data from the myLead array.

CALL the render function inside the eventListener so that when button is clicked the leads get into the unorder list and display in browser.

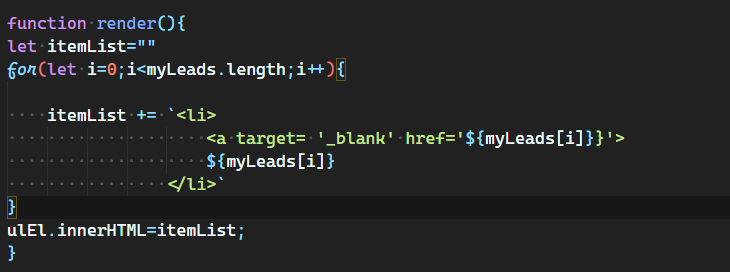


1. Clear out the input field my you append the data into unordered list.

inputEL.value=""

1.  Now make the leads clickable and set target to new tab.
2. Use **template string/literals** instead of the complex syntax.

It helps in to make code look like HTML inside JavaScript and don’t look complex.



1. Style ul li and anchor tag.
2. Deploy your project

* Make icon
* Manifest.json inside your main directory

For start deploying the project go to the chrome://extensions

1. Switch to developer mode.
2. Select the unpacked extension
3. Select the folder of your project and BOOM!!!! You added to your chrome extension.
4. Learn about Local storage.

Save the key value pair in local storage.

localStorage.setItem("myLeads", "www.examplelead.com")

Refresh the page. Get the value and console log it.

console.log(localStorage.getItem("myLeads"))

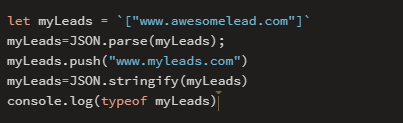
1. In local storage the value is stored in form of DOM string.

But we have taken the array myLeads in which URL are getting stored so we have converted the arrays to string.

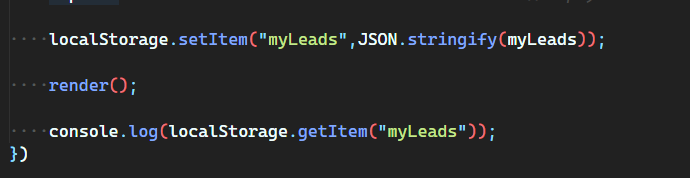
Local storage only support strings. Use **JSON.stringfy() and JSON.parse().**

**JSON.stringfy used to convert array into string**

**JSON.parse() used to convert string into array.**



1. Save myLeads array to local Storage.



1. Still, we are getting one problem that whenever we are refreshing the page the leads are getting removed.

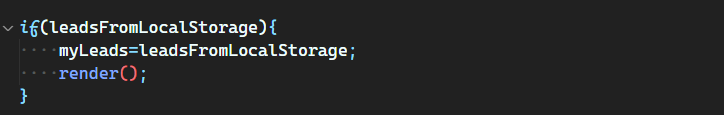
What is truthy value and falsy value

There are 6 types of falsy value:

* False
* 0
* “”
* Null
* Undefined
* NaN

Check that the leads are present in local storage?

If there are leads present in local storage then when display it else don’t show it.



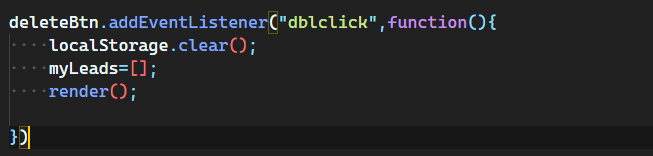
1. Create the delete button for deleting the leads.

Add some functionality

Clear local storage= localStorage.clear();

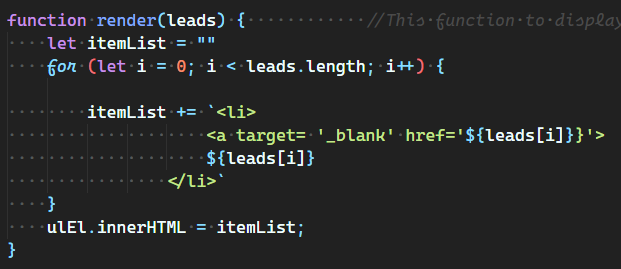
Delete all leads= this is done my setting myLeads array to empty.

Clear DOM= This is done my simply calling render function because the myLeads is now empty.



1. Now refactor the render function.

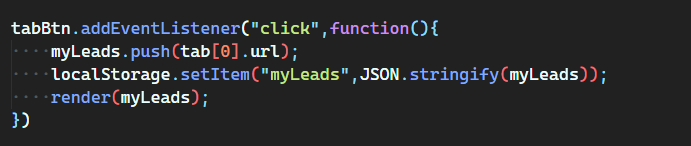
Because we are hard coded the myLeads array. Instead of that we can pass it as argument



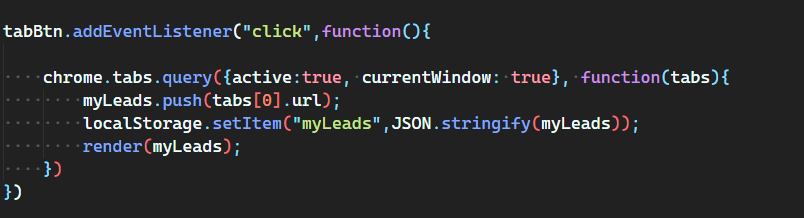
1. Now create the Save tab button so that you can save the current tab directly.

For this create the save tab button and add some functionality

addEventListener

* Create the tab array in which the URL is stored
* Push the URL inside my leads array
* Store in local storage
* Render it in the browser

1. Now we use the API to grab the tab inside the tabBtn eventListener



What we are doing with API?

We accessing the **chrome** and there is object inside chrome API **tabs** and inside tab there is method **Query** and passing some value inside query method

Active: true -> it means active tab

currentWindow: true-> it means the selected window of chrome

there is callback function which hold the tabs parameter.

1. Lastly go to the manifest.json and add permission to access the chrome.

