List Filter and Pagination

Description of code

Follow along with script.js and index.html

Create three variables.

1. One is the student list which will be a variable length, expect that you will use program in the future and list may change.
2. Two is the number of list items per page. This variable can be easily changed and may require it in the future.
3. Third is a div element to be used. It’s className has been set to divPagination but it should be just pagination. I cannot change it now. When I refractor I may change it. It is important to create this as a div. We will use it soon.

Next is another variable/function called showPage(). This function is the first thing that happens after the global variables of course. It’s purpose is the show the students on the page 10 at a time. This function accepts two parameters; list of students and page number. The first thing I do is make list equal to student\_list. This may be unnecessary to do. I probably should stick with just student\_list. Removed. Now I am just using student\_list.

Now, create startIndex using (page \* number of items) - number of items and endIndex using page \* number of items.

Then loop through the student list seeking the list between startIndex and EndIndex. If they are in the range then do nothing, otherwise make the display property ‘none’.

Call showPage(1,1)

Call appendPageLinks(1,1)

Create a new variable/function reset(). This will clear the divPagination and set its innerHTML to an empty string;

Function appendPageLinks(list, page)

The function should be called create pagination or just pagination to be more concise. The purpose of this function is to create the buttons needed to allow the user to see all the students in the list 10 at a time. We are not trying to do anything else but put buttons on the stage and make the work.

The first thing is list. This creates an HTML Collection. This is not the same as the Div that it lives in. We are looking for the content of list here. Then append divPagination. So to be clear divPagination is now a child element of list. But again this is an HTML Collection. It is text in array format. Need more study on this.

Now, create the ul and append it to divPagination. If you view it on the console you will have list.divPagination.ul

Now you can create the li’s one at a time and create the href links to them. I crammed the content into a span, but it probably would be more elegant to add them directly to the li element we are creating. Let me try that. Later.

New variable numPages that can change depending of the size of the list.

Enter the for loop. For every page(remember 10 students = 1 page) create one li, create the span. Check if the current page link should be active or not. This changes the link image to the same as the rollover state. Then append both to the ul.

End with add a listener to the ul. Call showPage(send the list. I question this, does it need me to send a new list from pagination?) NOPE took it out! STOP, put it back in. It’s breaking. Now the page keeps reloading with the . Call reset. This clears the pagination list so we can rebuild with the new active link. This is all it ever needs to do besides create the links.

Search for students section

This section allows the user to search the full list of students and display all matches. I started with creating a form, input, and search button. Then append the input field and button to the form. Not sure if this step is necessary. Create a parentDiv and searchDiv. The purpose of the parentDiv is because I kept getting confused and I needed to get back to the parent div. However, I realize now that I need to change the display property of the matching students only and stuff the element into a new array.

Borrowed the noResults section from brunomarchir/list-pagination-and-filtering// thanks brunomarchir. This section displays a message on the stage under the search field when there are no results.

Start function searchForStudent(text), where text is the input from user.

I started to create a new div and ul thinking I need create a new list and then display that. No that is not what needs to happen.

# What really needs to happen

Loop through the student list. Compare the names on the list, they can be found in the h3 element of each student. Then make that student visible. If they do not make the list turn them invisible or display = “none’. It’s that simple. However we need to access the h3.

If the student matches then put their information into a new list called filteredList. When done send/call showPage(filteredList, 1)

That’s the idea, this feature is incomplete and does not work, yet!