

67327

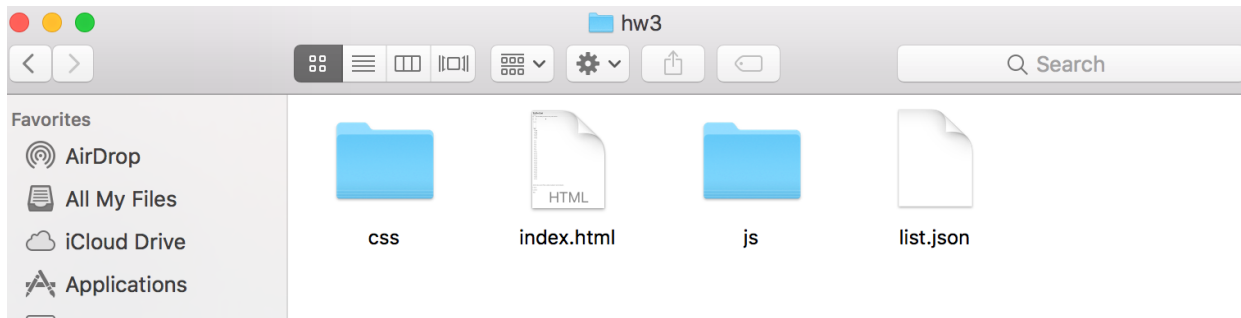
Anu Srikanth

HWs – AJAX & jQuery

Section 1:

Web Page URL: <https://hw3-vjifoexogn.now.sh/>

The organization of the website is shown in the image bellow. The JavaScript and css elements of the website are in their own folders and are linked together through the index.html page. The list. Son document has the list items that will be read in.



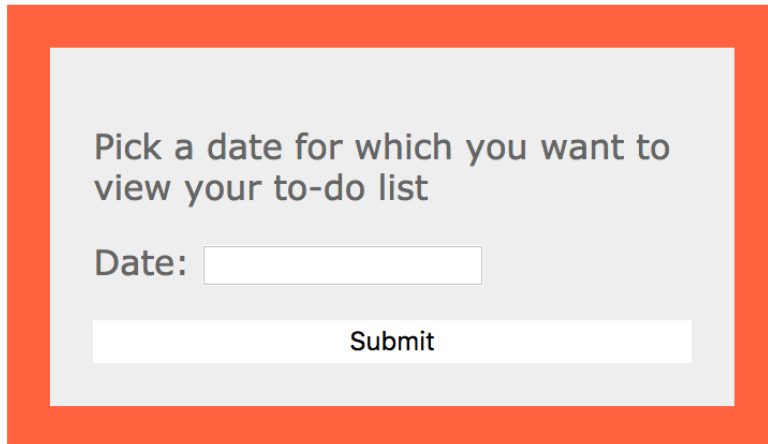
The concept of this website is to read your to do list, stored as a JSON document on your computer, and publish it to the page. To read the JSON to do list, you have to make a request from your webpage via AJAX to access the document (by giving the command the relative path to your document). After the information I fetched, you can parse it and transform it into better formatted pieces of information before publishing it to the webpage.

The following are the various functions I used in creating my website:

1. An anonymous function that fires as soon as the document is ready. This function carries out all of the cosmetic changes that need to be made to the website before any of its functionality is executed. This includes instantiating global variables, hiding templates that will later be cloned and added to the page, etc.
2. `changeDate()` is a function that changes the global date variable and changes the calendar to reflect the date that the user has selected.
3. `doXMLHttpRequest()` is the function that calls a function that processes json document. If this function is not successful, an error code is generated and displayed to the user. The request is made to the json document via an AJAX and information is added to your page.
4. `processResponse()` takes in the json response that it receives from the AJAX request and parses it into a list item format, with all the css and jess that goes into that format.
5. `makeListItem()` returns the html required for a list item that is placed after the calendar.

Section 2:

To Do List

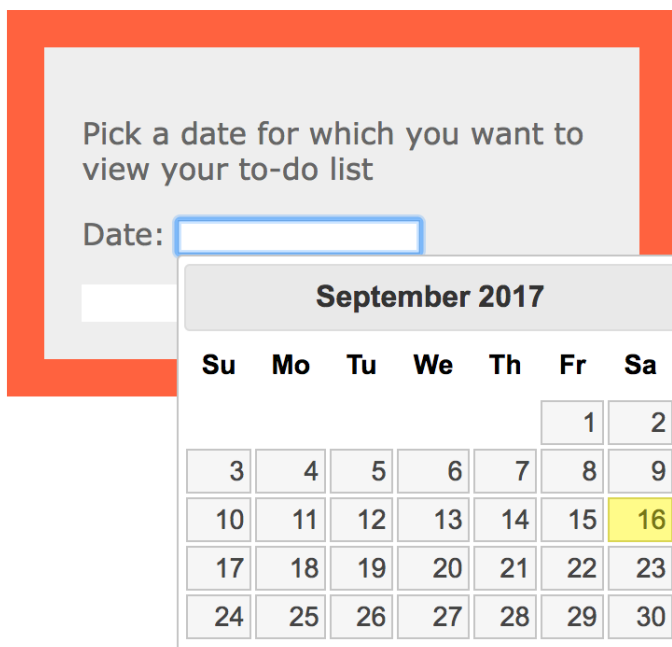


Pick a date for which you want to view your to-do list

Date:

Img1: The above form will take user inputted information and make an AJAX request. The Submit button is the button that will make a XMLHttpRequest.

To Do List



Pick a date for which you want to view your to-do list

Date:

September 2017						
Su	Mo	Tu	We	Th	Fr	Sa
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

Img2: The above image shows the user picking a date.

SEPTEMBER 2017						
Mo	Tu	We	Th	Fr	Sa	Su
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

To do: Phasellus id sapien in sapien iaculis congue.

Location: Göteborg

Time: 3:55 AM

To do: Vivamus in felis eu sapien cursus vestibulum.

Location: Stjgqing

Time: 5:12 AM

To do: Morbi sem mauris, laoreet ut, rhoncus aliquet, pulvinar sed, nisl.

Location: Nemyriv

Time: 9:42 PM

To do: Vivamus vestibulum sagittis sapien.

Location: Pho Duc

Time: 10:21 PM

To do: Integer pede justo, lacinia eget, tincidunt eget, tempus vel, pede.

Location: Stockholm

Time: 2:29 AM

To do: Morbi non lectus.

Location: Xi'an

Time: 7:35 PM

Img3: the list items present on the date that the user picked will now be displayed to the user. The list items were present in the JSON document that was requested. The information is then parsed, taken selectively when it matched what the user has requested, then displayed in a particular format to the user.

Section 3:

The various jQuery methods I used to carry out DOM manipulation are shown and explained bellow:

1. Various elements and templates on the html page are temporarily hidden. These are to be displayed when the user requests it or when an error occurs. One of these are a template that will be cloned and appended later to make a list.
2. The css attributes of an element are manipulated using jQuery method .css()

```
//jQuery Dom Manipulation
$("#calendar").hide();
$(".item_template").hide();
$(".error").hide();

//jQuery Dom Manipulation
$("#pickDate").css("border-color", "tomato");
```

3. The calendar element is displayed to the user after he requests a particular date. The necessary changes are made to the calendar element so that the user does not get information about a date he did not ask for.

```
//jQuery Dom Manipulation
$("#calendar").show();
```

4. This chained set of methods manipulate a date element in the calendar. Initially a random date is highlighted and is given the class “active” in order to attribute css attributes to it. This calendar will not be shown to the user. The user then selects a date that he wants to view, which may be different from the initially highlighted date. Therefore, whatever the active element is after the user has selected, the active class is removed from and it is given a generic class.

```
//jQuery Dom Manipulation, chained method
$(".days li .active").removeClass("active").addClass('tag');
```

5. The month and year element in the calendar needs to be changed according to what the user requested so the text method is used to manipulate that data.

```
//jQuery Dom Manipulation
$("#month").text( mon_list[mon][1]);
$("#year").text(year);
```

6. The remove() method is used to remove the dates that months do not have. For example, February does not have days 29, 30 and 31. Therefore, these elements are removed when being displayed to the user.
7. The hasClass() method is being used to check that a particular month has all of the dates that it is supposed to have. Because we don't want the page to refresh when the user makes multiple requests, when we remove elements, they do not return to the initialized state after another request is placed. Therefore, if a user wants to view a date in February and then one in August, we need the dates 29, 30, 31 to be present in August. Because we would have removed them in February, we check if the list of dates has the classes assigned to the removed elements.
8. After checking if these dates are present, whichever are not present need to be added. This is done with the help of the append() method, which appends the element to the list of dates.
9. The prepend function serves a similar purpose except the order in which the elements are added are reversed and therefore it must be done in a slightly different order.

```

if(mon_list[mon][2] == 28){
    //jQuery Dom Manipulation
    $(".opt-feb1").remove();
    $(".opt-feb2").remove();
    $(".optional").remove();
}else if(mon_list[mon][2] == 30){
    //jQuery Dom Manipulation
    if (!($(".days li").hasClass("opt-feb1"))){
        $(".days .wrap").append("<li class='opt-feb1 tag' value='29'><span>29</span></li>");
    }
    if (!($(".days li").hasClass("opt-feb2"))){
        $(".days .wrap").append("<li class='opt-feb2 tag' value='30'><span>30</span></li>");
    }if($(".days li ").hasClass("optional")){
        $(".optional").remove();
    }
}else{
    //jQuery Dom Manipulation
    if (!($(".days li").hasClass("optional"))){
        $(".days .wrap").prepend("<li class='optional tag' value='31'><span>31</span></li>");
    }
    if (!($(".days li").hasClass("opt-feb2"))){
        $(".days .wrap").prepend("<li class='opt-feb2 tag' value='30'><span>30</span></li>");
    }
    if (!($(".days li ").hasClass("opt-feb1"))){
        $(".days .wrap").prepend("<li class='opt-feb1 tag' value='29'><span>29</span></li>");
    }
}
}

```

10. The clone() method is used to duplicate some DOM elements. In this case, because the error element is a template that is hidden, we want to duplicate it and show it to the user. We want this to happen for multiple errors which is why it makes more sense to clone a template then append it to the list of errors and make it visible to the user.

```

//jQuery Dom Manipulation, chained method
$("#responseArea").append($(".error").clone().css("display", "block"));
$("#responseArea #code").text("Error code " + xhr.status);
}
}

```