# Homework #3

**CSCI E-3 Introduction to Web Development Using Javascript**

Harvard Extension School

## What you're learning:

This assignment is designed to give you a chance to exercise your knowledge of objects and functions, creating and reading from a JSON data structure and an array, string handling, simple Web page interactions (reading from a form and writing to the page), and introduce the window.localStorage object.

## What you need to do:

**Requirement #1:** For this assignment, you will write an HTML form that allows your user to input data of some type. It should contain at least three data fields that represent anything that interests you. For example, an individual record could be:

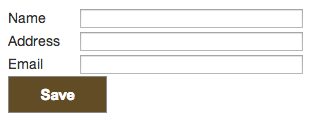
* A contact list that collects the name, phone number and email of your friends
* A movie list of your favorite films, including title, starring actors/actresses, duration, and year released
* A list of important places, including the name, a description, and geographic coordinates
* A list of the players on your favorite sports team, along with their current career stats

You get the idea - anything that interests you.

Your form will collect only one record at a time. So if you’re collecting a contact list, your form will only have the input fields for a single person’s information, along with a button to submit the information.

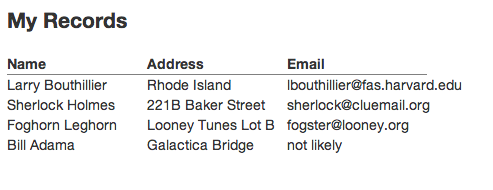
You’ve been given a working sample form in the html file that was distributed with the assignment. You may use it as is, change it in any way you like, or discard it and use something uniquely your own.

The sample provided looks something like this:



**Requirement #2:** When the submit button is clicked, two things will happen:

1. **Display the new entry on the page:** You will add the information to a list visible on the page. It does not have to be beautiful-looking, but it should be easy for users to read and understand what they are looking at. Code is provided to do this as well. Again, you may use it as is, change it in any way you like, or discard it and use something uniquely your own.  
     
   Your list doesn’t need to be fancy. Here’s a sample:



1. **Store the new entry:** You will add the new object to an array that tracks the entire list. That list should be contained in a JSON object, and should be pushed to window.localStorage so it’s persistent across page reloads. In an actual server-based web application, this array of JSON data might be sent up to a server for processing, but we’ll just save it locally for simplicity.

**Requirement #3:** There’s one more thing: When the page is first loaded, you should check for any data in localStorage, and do two things: 1) recreate your array that keeps the list of entries, and b) write all the entries to the screen so that a user always sees the entire list when they visit the page.

It is not necessary to create the ability to delete or modify existing records. We will be doing such things later in the course, but for now, it’s out of scope for this assignment.

## Some hints:

There’s a lot going on here, but if you break it down into smaller chunks, it won’t seem quite so bad. Here’s some hints for how to think about your solution. This is *one possible* solution breakdown – you could construct your code and objects in varying ways to get this done. ***You may not use any Javascript libraries*** *for this, however. This specifically is designed to exercise your knowledge of the Javascript language itself.*

After (and during!) each step here, you should be testing your code and seeing that you’re getting closer, step by step. Plan on lots of incremental iterations that bring you closer. Don’t try to solve everything at once, and don’t be surprised if you get a portion of the code working, only to have to change that code later as you continue to other requirements of the project.

1. **Create your data structure:** Your main data structure – the one that you store in localStorage, will be a JSON object that contains one property, which is an Array. You can create this first by creating a JSON object that contains one key (named as you choose), and has an empty array as its value. It might look something like this:  
   var myJsonAddresses = {"addressList" : [] };
2. **Build your HTML form:** We’ve provided a sample form. You can modify this as you like.
3. **Build a data object that will contain the data from your form:** If your form has three fields, you’ll need an object that can contain these three properties. You’ll create one of these every time you read data in from the form or for each record from localStorage. For example, for my address book example, my object might look like this:  
     
   function AddrBookEntry(n, a, e){

this.name=n;

this.addr=a;

this.email=e;

}

…or like this…

class AddrBookEntry{

constructor(n, a, e){

this.name=n;

this.addr=a;

this.email=e;

}

}

In either case, to make a new record object, I can call the constructor

new AddrBookEntry(name, address, email);

We’ll be adding more functionality to the object as we go, but this is a start.

1. **Fill in the onclick function for your Save button:** In your JS file, the onclick handler for the Save button does several things:
   1. **Get the values** from each form field
   2. **Create a new object** (using your constructor from step 3) that will contain the data from the form.
   3. **Add the information to your page** in an appealing way. A sample function has been provided that can do this.
   4. **Push the new object into your array** from step #1. You could do this with an external function, or a function that’s a property of the data object itself (a method), like, addToArray(theArray) which appends the object itself to the given Array.
   5. **Store the JSON object containing the array:** Now we need to store that information. Write another function that takes as an argument the JSON object that contains your array, and writes it to window.localStorage[whatever\_key\_you\_choose]. Remember that localStorage only stores strings, so you’ll have to call JSON.stringify() on your JSON object before you assign it to your localStorage.
   6. There’s a reason we put suggestion d) before suggestion e). When you store a string to local storage with your key of choice, it will overwrite any value already present. So, when you save your data, you’ll want to be saving all the cumulative data you’ve collected so far.

The only thing left is to write code that will display any existing data in localStorage as soon as the page is loaded. So,

1. **Read from storage:** Add code that runs each time the web page loads that will:
   1. Check to see if your localStorage contains data that already exists for this page
   2. If so, recover the JSON object by calling JSON.parse() on the data from localStorage. The goal is to recreate the JSON object and array just as it was before you refreshed the page.
   3. Iterate over that array, writing each object to the page by calling writeRowToPage()

Be sure you clear out your localStorage once in a while when you’ve gotten this working, to be sure it all still works when the localStorage starts out empty. You can do this from the Developer Tools console by running:

window.localStorage.clear()

OR

window.localStorage.removeItem(“[whatever\_key\_you\_used]”)

For testing purposes, if you’ve previously saved your JSON string (copied to the clipboard from Developer Tools or the console), you can restore it from the console as well. Here’s mine for my example:

window.localStorage["myJSON"]=JSON.stringify({"addrList":[{"name":"Larry Bouthillier","addr":"Rhode Island","email":"lbouthillier@fas.harvard.edu"},{"name":"Sherlock Holmes","addr":"221B Baker Street","email":"sherlock@cluemail.org"},{"name":"Foghorn Leghorn","addr":"Looney Tunes Lot B","email":"fogster@looney.org"},{"name":"Bill Adama","addr":"Galactica Bridge","email":"not likely"}]})

You’ll submit your assignment in the usual way, with one small change. Please name the folder you zip up like lastname\_firstname.homework3. You should also create a ZIP file named like: lastname\_firstname.homework3.zip. You should use the same folder structure we’ve been using all along, with a conspicuous HTML file in the top level that runs your program.