Connecting Javascript with HTML

(Let the fun begin)

My Calculator (4 Hours Max)

Your challenge is to create a Calculator, no need to implement all the buttons, do as much as you can in 4 hours.

DO NOT spend too much time on creating this layout, just make it look nice.

Steps

1. Start with the digits buttons, when a digit is called, call a function addDigit(digit) , this function adds a digit to the current num which are kept in the globals: gNum1 and gNum2 ;
   (Tip: the function knows which gNum is the active one by checking if gNum2 is null.
2. Now add the + button, when clicked, put 0 into gNum2 (instead of the initial null) so the addDigit will now work on it.
3. When '=' is clicked – show the result
4. Now expand the functionality step-by-step, remember that something that is basic but works worth more than a lot of code that doesn't work.
5. Do not spend more than 4 hours on this.

Images Changing

Your challenge is to show 3 images on the page.

1. When an image is clicked, mark it as selected by adding a class that adds a border

2. Clean previously 'selected' image (you may use a global variable to hold the selected element)
3. Add a button 'Change Images', when clicked, change all 3 images
    a. Use an array of strings: the image urls.
    b. You may write a function getRandomImage to return one of the images urls from the array
4. Add a button 'Change Images in 5 seconds', when clicked, change all 3 images after 5 seconds
5. Add a button 'Change Images every 5 seconds', when clicked, change all 3 images every 5 seconds
    a. when clicked, change the text of the button to 'Stop!', when clicked – stop and set the button text back.
6. Change the images array to contain objects, every object should have the url for that image and a description text
7. When an image is clicked, show its description

Pop those Balloons

Your challenge is to show some balloons and make them go up slowly to the top of the page.

1. Create 2 divs that looks like balloons
    a. Create a class .balloon with:
    width, height, border-radius, and position: absolute
    b. Create classes balloon1, balloon2 with:
    background color,
    left (so they don't be on top of each other)
2. In Javascript, when page loads, select the balloons (querySelectorAll) and make them move up a bit by setting their style.bottom
3. Add your global data structure: gBalloons – this is our model!
    a. This should be an array of balloons objects
    b. Each object should have 'bottom' and 'speed' properties
4. Set an interval to update the balloon object, and then set the updated values to the balloon elements in the DOM.
5. When a balloon is clicked
    a. Hide it (by setting the style.display to none)
    b. Bonus: make the clicked balloon fade out, by setting style.opacity value in an interval.
        i. Hint: add an opacity property to your model and then decrease it by 0.1 in an interval until zero, then clearInterval it.
        ii. Challenge: how to correlate the clicked element and the model? How many different ways you can think of?

Safe Content

Your challenge is to show some secret content, (i.e. a secret photo) only to users that will be able to *login* using user and password.

1. You will have a users array with user objects (gUsers array with 3 users)

   a. User object will contain: username, password and lastLoginTime (timestamp)
2. Write a function saveUsers(users), that saves the users to localStorage using JSON.stringify()
   a. Call it to save the gUsers
   b. Then remove the Hard Coded users from the code (comment it out)
3. Write a function getUsers() that loads the users from the localStorage and uses JSON.parse()
4. When page loads prompt for user and password
5. Write a function that gets username and password and find such a user exist, the function should return the user object if found or null if not (use filter)
   a. This function uses the getUsers() function
6. If the user successfully log in, update his lastLoginDate and show him the secret content
   a. Use the saveUsers function to save the updated users array
7. Add a button: logout, when clicked ask again for user and password…

8. If the user is also an admin (add this property to your model)
   a. show him a link to admin.html
   b. save an indication: userIsAdmin in localStorage
      i. hint: note that localstorage only saves strings, booleans are not supported
9. In admin.html show the list of the users inside an HTML table
10. Use the localStorage to *protect* the admin page
    a. if not admin redirect to index page using window.location
11. **Bonus:**

a. In the admin page, also build UI for *cards* presentation
(side by side divs)

b. Allow the user to switch between the two presentation modes (cards/table)

Simple Calendar

Your challenge is to build some basic calendar functionality.

- Create your data structure: 3-4 meetings containing: title, startDate (in timestamp), endDate, array of participant names.
- Organize the meetings so that they are sorted by startDate
- Write the following functions:
  - *addMeeting(title, start, end, participants)* create a new meeting and put it in the right place in the array (may need to push the rest of the array)
  - *findNextMeeting()* – returns the next meeting on my calendar (the first meeting that starts after now)
  - getMeetingsCountFor(participantName) – returns how many meetings this participantName is invited to
- Create an HTML interface that shows the events list and activate all the functions

Car Racing

Your challenge is to create a racing car game:

1. Show two cars (simple divs will do for start) going from left to right.
2. 2 gamers will share the same keyboard
3. Gamer1 is using the keys 1,2
4. Gamer2 is using the keys ↑, ↓
5. the quicker you switch, the faster your car goes

Hints

- gCars array, with car objects that has a left property
(represents its left offset on the page)
- put a onkeyup event on the body
- an Event object has a timestamp property
- a keyboard event object has a 'code' property that has a meaningful name (print it to console to see)
- to check which player is pressing, you can add an array of 2 key codes to the car object

BONUS : Feed those Monsters

Your challenge is to stretch your brain to think about graphs…

- Create a monsters array, with monster objects: name, and power, each monster should also have an array with the names of his best monster friends.
- Write a function getMonsterByName(name) that returns a monster objects by its name.
- Write a function that find monsters that are stronger from all their friends
- Bonus: Write a function that checks if all monsters are connected to each other through any level of friendship.

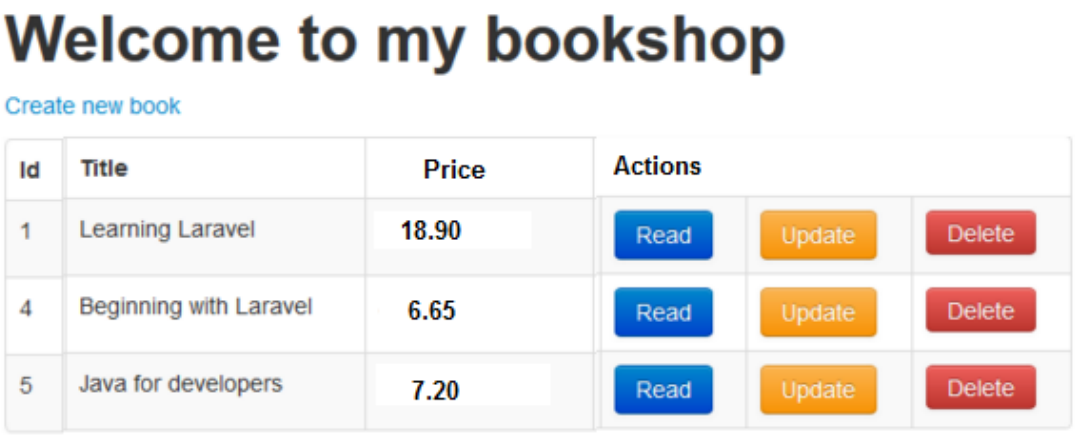
BooksAreUs

Your challenge is to build a page that shows a list of books: *id*, *name* and *price*. We will allow the user – a shop owner – to manage the books.

1. Create your Model and show the books in a table. We will use a global variable gBooks, and a function renderBooks() that will draw the table

Now, let's handle CRUD (Create, Read, Update and Delete)

2. Add an Actions column to the table (something like this :)



3. Handle delete - when the button clicked we should call the function deleteBook(bookId))
4. Support adding a new book:
    a. When clicked, call the function readAndAddNewBook() that will read (prompt) the details from the user: name and price, then will call a function addBook(name, price) that will find the next available id and push a new book into the gBooks model. Then call the renderTable() to redraw the table
5. Support updating a book:
    a. When clicked, call the function: readAndUpdateBook (bookId) that will prompt for the

book new price (name never changes) and call the function updateBook(bookId, bookPrice).
Then Call the renderTable() to redraw the table

6. Create an HTML section: **Book Details** below the table, that shows the details of a selected book including its photo (based on its id)
    a. When read is clicked, update the section to show the details of the selected book.
    b. Add a rate property for the book, set 0 as default, the rate should be a number between 0 and 10.
    c. In the Book Details, allow the user to change the rate of the book by clicking a Thumb up or Thumb down button.

Bonus

1. Make the header of the table clickable to support sorting by name or price

2. Add paging:
3. Read the data from the user using an <input> instead of prompt

Touch the Numbers

Build the game. Design it as you will.