3/23/2012

CS303 Object Oriented and Functional Programming in JavaScript

W1D2_3 Objects

Assignment for W1D2 Object Properties and References:

Part I. Complete the following tasks from The JavaScript Language book. Write your own code for each of the coding tasks in VSCode before looking at the answers. It is very helpful to have the ESLint checks running when you write your code. You can turn it off in a file if it is trivial code that you do not need to debug with the following line at the top of that trivial file:

/* eslint-disable */

However, when writing any nontrivial code it finds a lot of bugs and although it takes a little extra time for the initial coding, it will save you a lot of time in the end. Equally important, it will develop good coding practices and make your code look much more professional and make you a better faster professional, faster!

Chapter Objects: the basics.

Section Objects:

- Hello, object
- Check for emptiness (include the tests in VSCode)
- Sum object properties
- Multiply numeric properties by 2

Part II. Write a JavaScript program that will accept title, author, and libraryID values from an HTML page and create new book objects for each entry.

- Before creating the webpage first write and test the following JavaScript functions:
    - addBook, which will take title, author, and libraryID as inputs. It will create a new book object and add it to the library, which will be represented as a global array named libraryBooks. addBook should return the newly created book.
    - findTitles, which will find all the book titles in libraryBooks and return them in an alphabetically ordered array.
    - findAuthors, which will find all the authors in libraryBooks and return them in an alphabetically ordered array.
    - findIDs, which will find all the libraryIDs in libraryBooks and return them in an alphabetically ordered array.

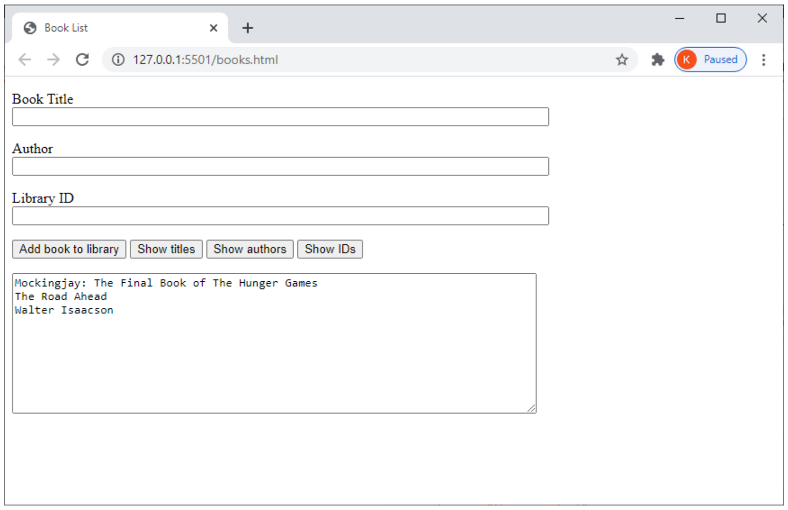
  - **Test these four functions using the Mocha test file: libraryBooksTest.js**

- Include buttons with the following values and “titles” (the part after the colon, which should appear on mouseover)
  - Titles: list all titles, sorted alphabetically
  - Authors: list all authors, sorted
  - libraryID: sorted

- There should be a separate textarea where the titles, authors, or libraryIDs will be displayed depending on which button is selected. Use the eslint conventions including JS Doc for any functions.
- Hint: you might find the myArray.join(“\n”) method useful.

```
let library = [
   {
       title:  'The Road Ahead',
       author: 'Bill Gates',
       libraryID: 1254
   },
   {
       title: 'Walter Isaacson',
       author: 'Steve Jobs',
       libraryID: 4264
   },
   {
       title: 'Mockingjay: The Final Book of The Hunger Games',
       author: 'Suzanne Collins',
       libraryID: 3245
   }];
```

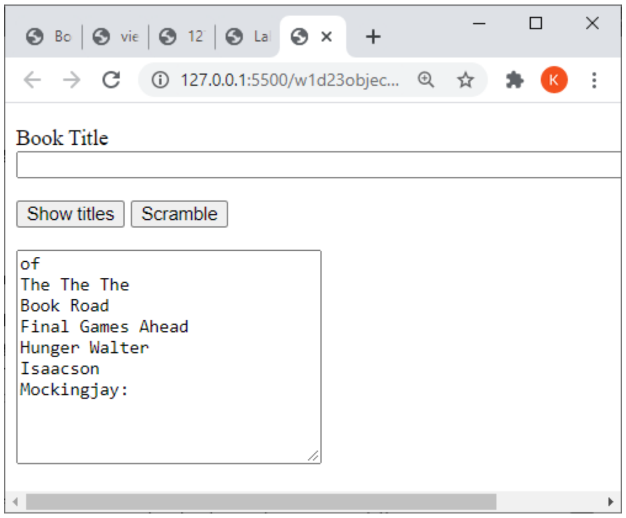
Below is a simple example of what your HTML page might look like:



```
Mockingjay: The Final Book of The Hunger Games
The Road Ahead
Walter Isaacson
```

EC: There is a “Scramble” button in the skeleton books.html file. Scramble should

- find all the titles,
- sort the words in them by length,
- display them in the text area with the 1 letter words on the first line, 2 letter words on the second line, 3 letter words on the third line, etc.



```
of
The The The
Book Road
Final Games Ahead
Hunger Walter
Isaacson
Mockingjay:
```

Assignment for W1D3 Object Methods and Constructor Functions:

Complete the following tasks from The JavaScript Language book. Try to answer them before looking at the solutions. For the ones asking you to write code do that in VSCode and use the esLint coding conventions we gave you. These are all from the chapter Objects: the basics.

Section Object methods, "this"

- Using "this" in object literal
- Create a calculator

  Hint: If you use the Plunker sandbox and test code, the following will run before each test. The sinon.stub function will intercept the first two calls in your code to the window.prompt function and instead of calling 'prompt' it will return a "2" on the first call and a "3" on the second call. This allows your test code to run without requiring human input. It is like writing a test script and telling the human tester they should enter a 2 and then a 3 for the prompts before every test.

  ```
  beforeEach(function() {

    sinon.stub(window, "prompt");

    prompt.onCall(0).returns("2");
    prompt.onCall(1).returns("3");

    calculator.read();
  });
  ```

  On the other hand, if you want to write and test the code in VSCode and NodeJS you can **use the Mocha test file, calculatorTests.js,** which hard codes the input values instead of using the sinon stub framework.
- Chaining

Section Constructor, operator "new"

- Two functions – one object
- Create new Calculator (**solve with the calculatorTests.js** Mocha test file in VSCode)

Write a constructor function Accumulator(initialValue, increment). The object it creates should:

- Store the current accumulated value in a property currentValue. The constructor should set this to be initialValue.
- The accumulate method should increase the currentValue by the increment.
- The report method should return the currentValue.
- **Run this with the accumulatorTests.js Mocha test file**.