Assignment #2

JavaScript

CS193C Summer 2020, Young

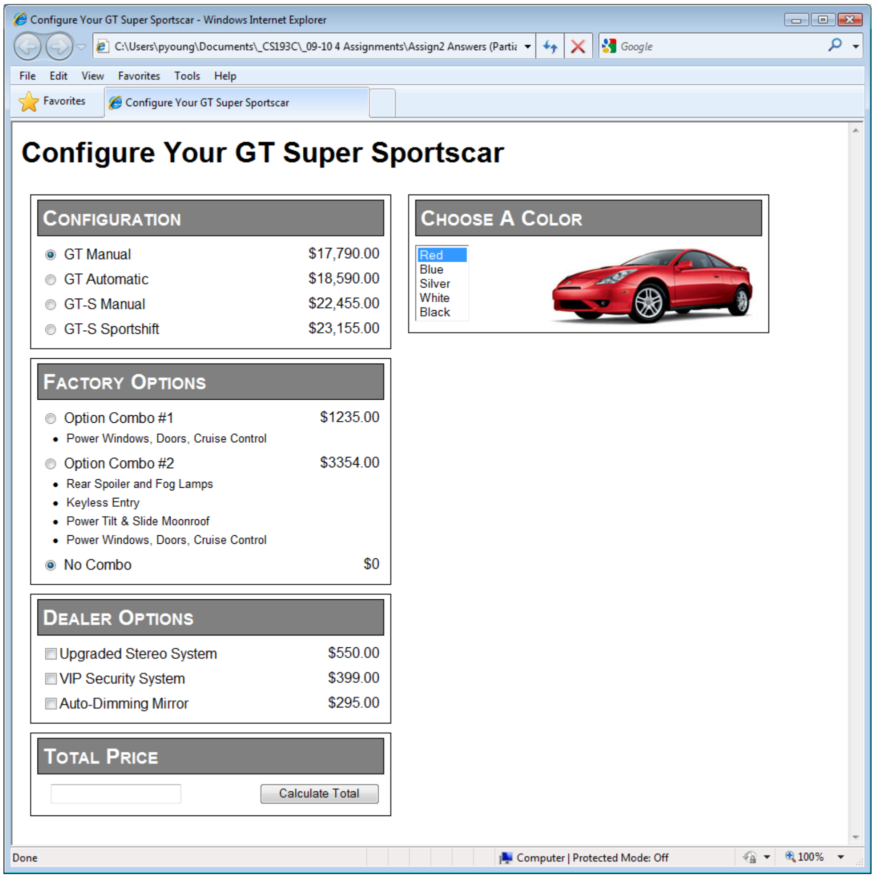
In this assignment we get a chance to work with JavaScript. This assignment is due Thursday, July 16th at 1:30pm.

<u>For this assignment, you are not allowed to use WYSIWYG editors (e.g., no Dreamweaver) and should only use a text editor to compose your assignment</u>. In addition, your files should work on both the latest version of Mozilla Firefox and Google Chrome.

You may use float, flexbox, or grid layout for the problems in this assignment.

Car Options

In this part of the assignment you will create a webpage which allows a user to configure a car to see how much it costs. Here is a screenshot showing my version of the assignment:


Configure Your GT Super Sportscar - Windows Internet Explorer
C:\Users\pyoung\Documents_CS193C_09-10 4 Assignments\Assign2 Answers (Partia
Google
File Edit View Favorites Tools Help
Favorites
Configure Your GT Super Sportscar
Done
Computer | Protected Mode: Off
100%

Configure Your GT Super Sportscar

Configuration

● GT Manual	$17,790.00
○ GT Automatic	$18,590.00
○ GT-S Manual	$22,455.00
○ GT-S Sportshift	$23,155.00

Factory Options

○ Option Combo #1	$1235.00
• Power Windows, Doors, Cruise Control	
○ Option Combo #2	$3354.00
• Rear Spoiler and Fog Lamps • Keyless Entry • Power Tilt & Slide Moonroof • Power Windows, Doors, Cruise Control	
● No Combo	$0

Dealer Options

□ Upgraded Stereo System	$550.00
□ VIP Security System	$399.00
□ Auto-Dimming Mirror	$295.00

Total Price

Calculate Total

Choose A Color

Red
Blue
Silver
White
Black

As you can see, the car comes in four different configurations:

GT Manual	$17,790.00
GT Automatic	$18,590.00
GT-S Manual	$22,455.00
GT-S Sportshift	$23,155.00

The user chooses a basic configuration and then chooses a factory options package and adds on any dealer options desired. Here are the factory options:

- Option Combo #1 $1235.00
  - Power Windows, Doors, Cruise Control

- Option Combo #2 $3354.00
  - Rear Spoiler and Fog Lamps
  - Keyless Entry
  - Power Tilt & Slide Moonroof
  - Power Windows, Doors, Cruise Control

- No Combo $0

and here are the dealer options:

- Upgraded Stereo System $550.00
- VIP Security System $399.00
- Auto-Dimming Mirror $295.00

When the user clicks on the "Calculate Total" button, you should determine the price of the car, given their configuration and option choices.

In addition, as you can see on the right of the webpage I've given the user an option to choose a color for their car. When they change the color in the list, the picture of the car should automatically change. I've included five car photos for you to use with this assignment.

I've gussied up the webpage using stylesheets. Make sure you get the layout right—heading across the top of the webpage and then two columns. Additional styling isn't strictly necessary, but is a nice opportunity for CSS practice. In addition, you may notice that I've converted the total cost back to a price ($20,375.00 instead of simply 20375). While you don't have to do this, it's pretty easy to do. The Number object's toFixed could be used to ensure that you always have two digits after the decimal point. However Number's toLocaleString method will take care of both the digits after the decimal point and put in the commas for you.

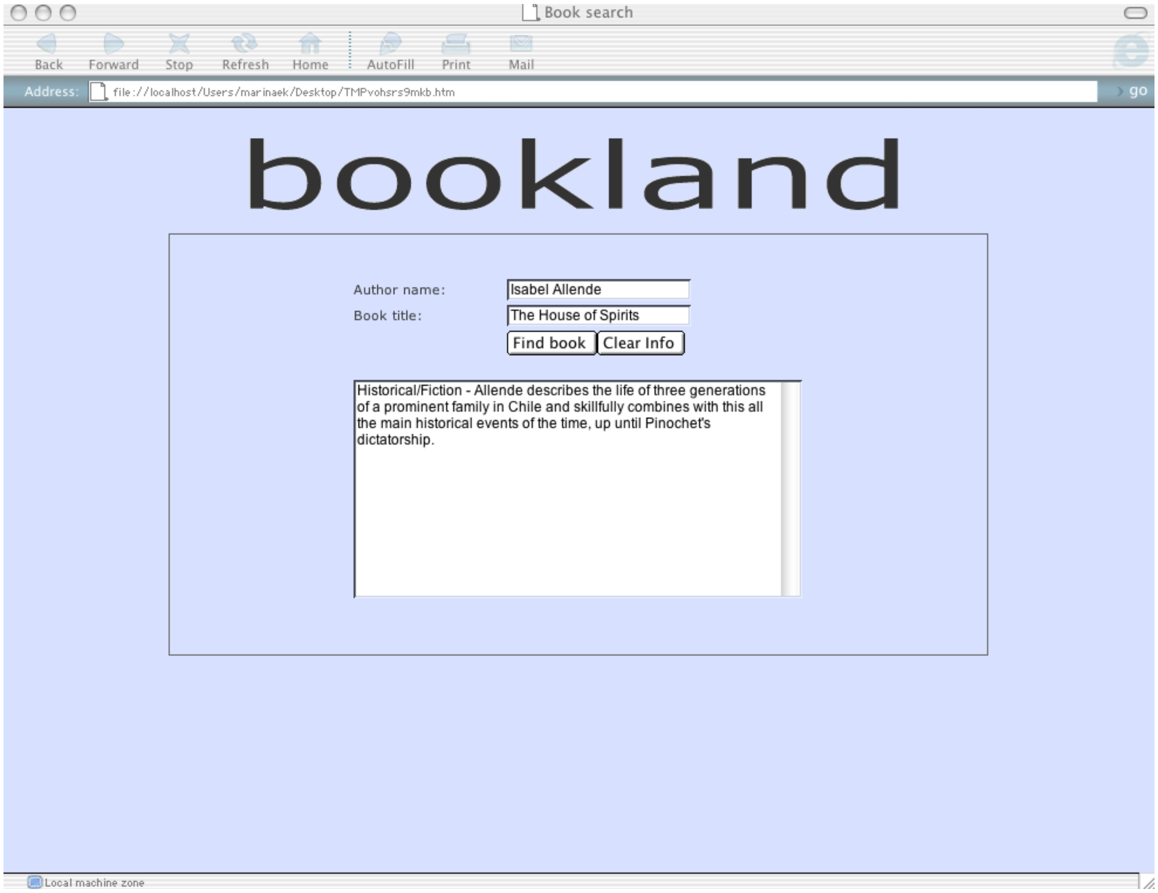
Bookland

For this part of the assignment, you will create a webpage that allows the user to search for a book with a specific title and/or by a specific author. Then, you will display information about the book on the web page.

We are providing book data that you can use in a file called books.txt. For each book, we provide the author's name, the book's title, and a short description of the book. You should copy and paste the text from the books.txt file into your JavaScript file, placing the data found there in whatever data structures you choose to use for your JavaScript program. *It is **not** intended that you should use JavaScript to read the file contents* – in fact, browser-based JavaScript can't read files from a computer, as allowing webpages to read a user's files would be a major security breach.

The way this should work is as follows: You should store the book data we provide in arrays. You can use multiple synchronized arrays (one array for the authors, a second matching array for the title, and a third for the descriptions) or one array of objects. The user of your webpage will type in a name and/or a title. You will search for this specific book (or books), using the given name and/or title, in your array(s). Once you find a match, you should display its information on the page. You may need to display information for several books if there are several matches. Please see the *Things to note* paragraph later on to find out exactly what you need to display.

Since we are only displaying text on the page, we can just use a textarea in which to display the information. Here's what the page might look like:



file://localhost/Users/marinaek/Desktop/TMPvohsrs9mkb.htm

Author name: Isabel Allende

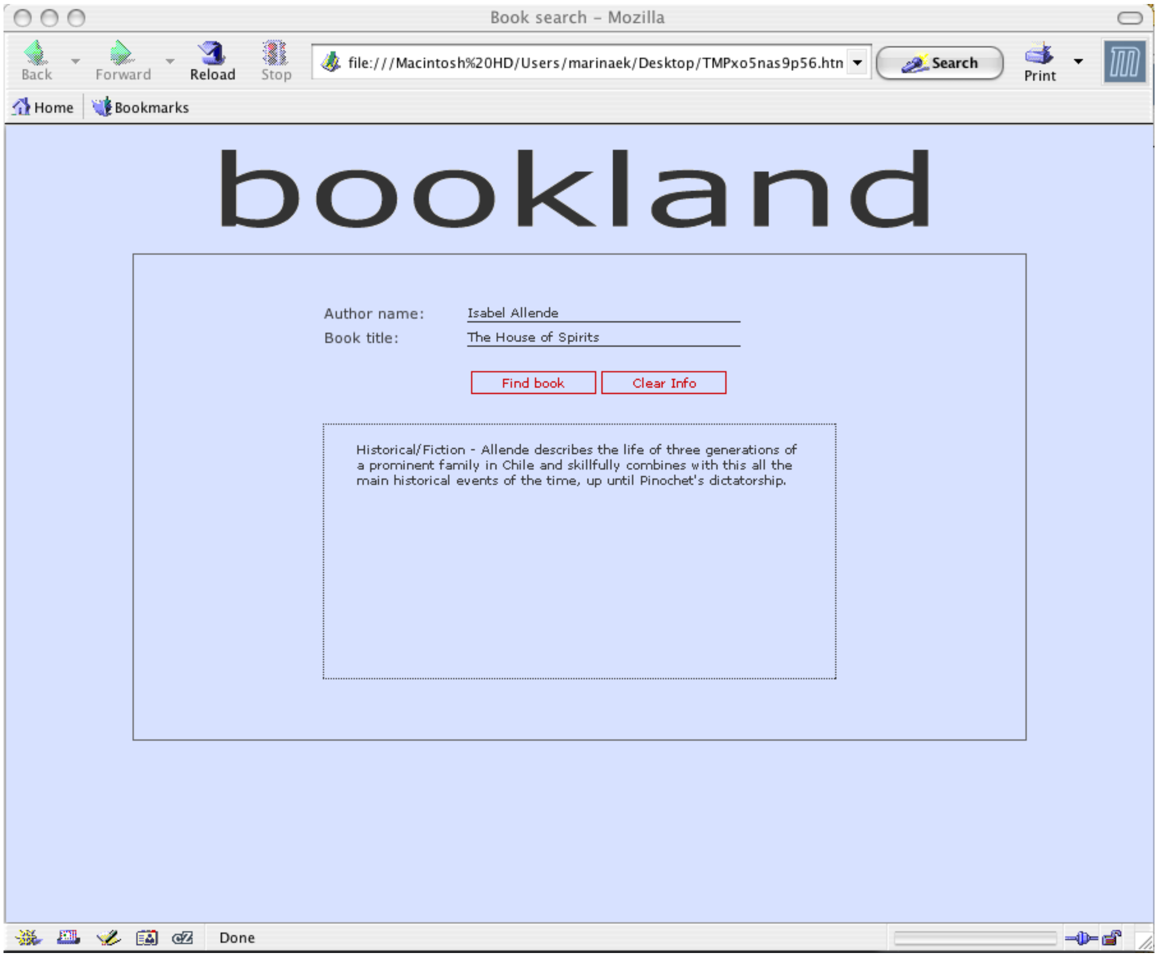
Book title: The House of Spirits

Historical/Fiction - Allende describes the life of three generations of a prominent family in Chile and skillfully combines with this all the main historical events of the time, up until Pinochet's dictatorship.

In this example, the user typed in "Isabel Allende" as the author's name and "The House of Spirits" as the book title. When the user clicked "Find book", the short description for this specific book was displayed in the textarea.

Things to note: You do not have to worry about case sensitivity. If you want to make the search case insensitive, go for it but it is not required. If a book is not found, your page should do something reasonable, i.e. either display "Book not found" in the textarea or display nothing. If the user enters both a name and a title, you should display the description of the first match, i.e. the first book found that satisfies the given criteria. If the user only enters an author name, you can either display the title and description of the first match, i.e. the first book that you found by that author, or you can display the titles and descriptions of all books by that author. Similarly, if the user only enters a title, you can display the author and description of the first match, i.e. the first book that you found with that title, or you can display the authors and descriptions of all books with that title. All information must be displayed in the textarea. Assume the user has to enter full names to get a match – so Isabel will not get a match, but Isabel Allende will.

Once you get the array search working, try to see how you can improve the page using style sheets. Several professional web pages actually use form elements to display text information on pages, as we are doing in this problem. However, they also use CSS to hide the form elements and to make their pages look better. We'd like you to experiment with different CSS to see how you can change the form elements. Here's our version of the web page with CSS:



bookland

Author name: Isabel Allende

Book title: The House of Spirits

Find book | Clear Info

Historical/Fiction - Allende describes the life of three generations of a prominent family in Chile and skillfully combines with this all the main historical events of the time, up until Pinochet's dictatorship.

We have changed the appearance of the text fields, buttons and textarea using several different style attributes. Your page does not have to look like ours. Feel free to change things any way you want to, as long as you experiment with CSS and form elements.

Arrow Functions

As I mentioned in lecture, "arrow functions" are a way of defining anonymous functions in JavaScript. They are used quite frequently, so in this last part of the assignment you'll get a chance to experiment with them.

Arrow functions have several important characteristics. They provide a nice compact way to write short functions used for things like event handling. They also have important characteristics in regards to JavaScript objects and *this*. We're going to ignore that second characteristic for now, and just focus on their use for writing clear and compact code.

For this part of the assignment, I am providing you with four different files:

arrow-functions.js – This file has a number of short snippets showing how arrow functions might be used, followed by a number of short examples of how arrow functions are written. *You do not need to modify (or turn in) this file*.

arrow-example.html – This file is another example file. It includes two buttons, one button has an event handler assigned to it using a traditional function definition, the second does essentially the same thing, but has its event handler defined using an arrow function. *You do not need to modify (or turn in) this file*.

fill-in.html – This file includes a text field that our user should be filling in. Three seconds after the webpage loads, the background of the text field turns red, and a reminder to fill in the text is placed next to the text field. This is done by having a function called warnUser called 3 seconds after the webpage loads. Eliminate the warnUser function by replacing it with an arrow function.

In actual usage, we would have to extend this program to cancel the timer if the user filled in the text field before the time ran out. But for simplicity, we're not going to do that.

graduation.html – This file lists the names of five graduate students. When the user clicks on the "Graduate" button, all five will earn their doctorate, and their names will be updated to reflect this. This involves two different functions. The grantDoctorate function appends a "Dr." to the front of their names and the graduateEveryone function is called when the user clicks on the "Graduate" button. Replace both the grandDoctorate and the graduateEveryone functions with arrow functions.

Credits

The Car photos (and the car prices) are from a Toyota Celica and were originally from www.toyota.com. I've eliminated car options and car colors to keep the assignment simple.

Book problem was created by Marina Kassianidou, former CS193C TA. The short description of each book in the books.txt file is taken from the books' back cover.