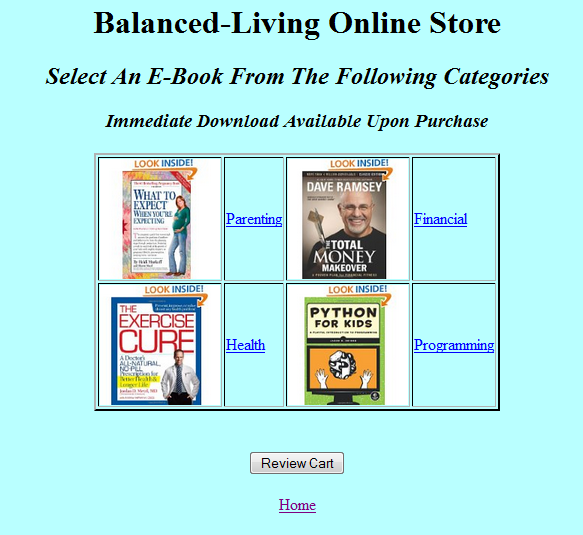
**Programming Assignment 3**

**JavaScript Shopping Cart**

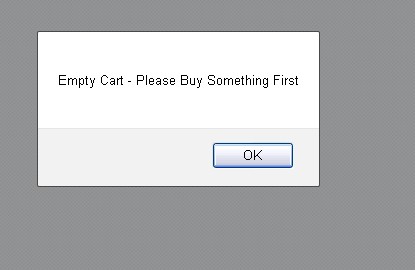
You will need to modify some of the HTML pages from Programming Assignment 1 and Programming Assignment 2 in order to create Programming Assignment 3 (Most of the modifications require that Javascript is invoked from the given page). **If you were unable to complete ProgramAssignment1 or ProgramAssignment2, please come see me.**

First, modify *store\_index.html* (ProgrammingAssignment1) by adding the *Review Cart* button and *Home* hyperlink as shown below:



Pressing the *Home* link will return the user to the homepage (*index.html*). Pressing the “Review Cart” button will allow the user to review the items he wants to purchase in the form of a shopping cart. The shopping cart should be implemented using only JavaScript client-side technology and cookies.

Initially pressing the “Review Cart” button without purchasing anything results in the following alert box displayed:

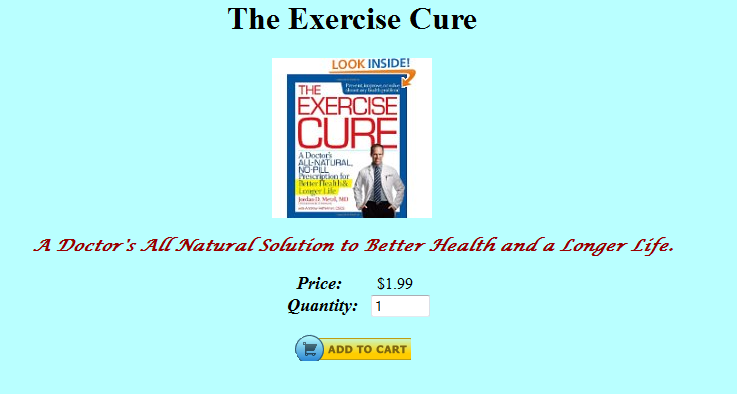


**Note**: The example above is displayed using the latest version of Mozilla. Using an earlier version of Mozilla or Internet Explorer could lead to a similar but slightly different looking alert box.

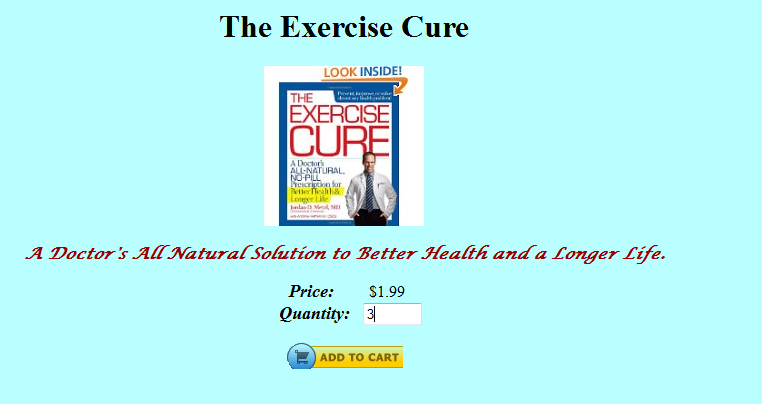
Pressing “OK” will result in the “*store\_index.html*” being redisplayed.

The 2 major categories – *parenting and health -*  should have links that correspond to the product pages - *parenting.html,* and *health.html* as implemented in ProgrammingAssignment 1. ProgrammingAssignment 2 implemented the individual Product Pages for *4* *parenting items* and *4* *health items.*

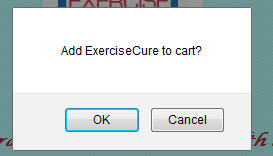
An example of a health Product Page – *The Exercise Cure* - is shown below – Note that there has been a *Continue Shopping* link added to the bottom of this page. Pressing this link will return the user to *store\_index.html*.

\\psf\Home\Desktop\Screen Shot 2014-02-13 at 11.31.19 AM.png

**Observe that the user has entered 3 in the Quantity field before pressing the “Add To Cart” button.**

\\psf\Home\Desktop\Screen Shot 2014-02-13 at 11.31.19 AM.png

Pressing the “Add To Cart” Button will display the following alert box:



The *store\_index.html* should automatically be re-displayed if the user presses “OK” thus allowing the user to purchase other items. Pressing “Cancel” does nothing and the same page (in this case *exercise.html*) is displayed as before.

The user can continue purchasing as many products as desired - up to 8 - for our implementation. For this example, suppose the user purchases two additional items besides the *Exercise Cure*. Their name (and quantity) are shown below:

**Item Name Quantity**

Wheat Belly 6

Hands Free Mama 4

Pressing the “Review Cart” button should now display the following screen illustrating the details for the purchases the user wants to make.



Pressing the “Continue Shopping” link returns the user to the store\_index.html page and allows the user to purchase more items.

The leftmost column consists of the “Catalog #” – an arbitrary number assigned to each product (e.g., for these 3 items - exercise, wheat belly, hands free mama) associated with each product. For this example, 1 is assigned to the exercise, 0 is assigned to the wheat belly, and 7 is assigned to the hands free mama. You can assign the numbers from 0 to 7 for the 8 products anyway you like. We’ll be using this catalog # as the primary key when we build the database for these products in a later assignment.

The column to the right of the “Catalog #” is the “Item” column. This column should consist of the following information about each purchased item:

The item’s name (this should be the name displayed above the image of each product on its respective product page).

The image of the product - displayed as an 80x80 thumbnail image

The “Remove” button - allowing the user to remove the product from the Shopping Cart. Details for the remove button are given below.(generate the cart.html to automatically remove the item)

The column to the right of the “Item” column is the “Price” column which contains the **individual** price for a given item.

The column to the right of the “Price” column is the “Quantity” column. This is the value entered by the user on the product page and corresponds to the number of a certain product that the user wants to purchase.

The rightmost column is the Total purchase for a given product - This is computed by the price\*quantity.

These 5 columns represent the data displayed for one product (e.g., one row) in the shopping cart.

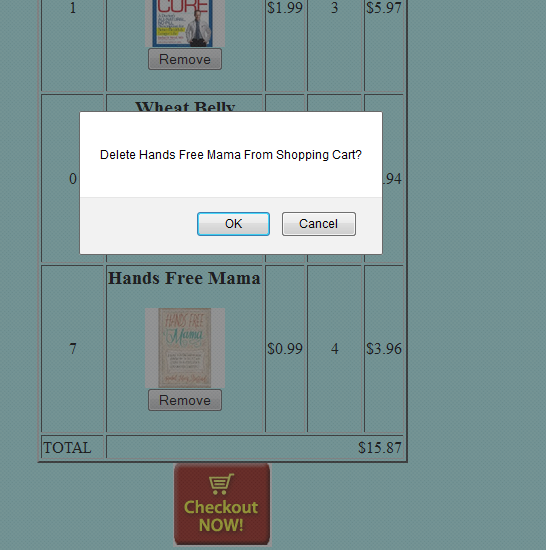
The 5 columns are repeated for each product the user has selected to purchase. In this case, 3 products have been purchased resulting in 3 product rows.

The last row of the shopping cart illustrates the total amount purchased. This is the summation of the rightmost column of the shopping cart and shows the total money owed by the user for all his purchases. **You must display all numeric data to 2 decimal places.** Hint: you can use the toFixed(2) method to accomplish this.

**The data displayed in the shopping cart should be symmetrical, neat and readable.**

Pressing the “Remove” button for a given product (e.g., the *Hands Free Mama* for example) will result in the following actions:

1. The user will be prompted with a confirm box as shown below for removing the *Hands Free Mama*:



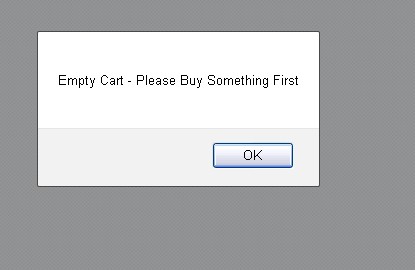
Pressing “OK” will remove the *Hands Free Mama* from the shopping cart. The shopping cart page **must immediately** reload and display that the *Hands Free Mama* e-book is removed from the shopping cart as shown below:



Note how the last row displaying the TOTAL amount purchased is automatically updated. Pressing “Cancel” on the confirm box above redisplays the Shopping Cart page as shown previously with no changes made to it.

**Hint**: Selecting the “Review Cart” button could redirect the user to another page named “*Cart.html*” (although you can just invoke a JavaScript function from this button if you like). The “onload” event handler of the <body> element for *Cart.html* could reference a Javascript function that builds the Shopping Cart display. The logic for removing an item from the shopping cart can be as simple as removing the corresponding cookie and then reloading *Cart.html*. There are many other ways to display the Shopping Cart and you are free to implement this part anyway you like.

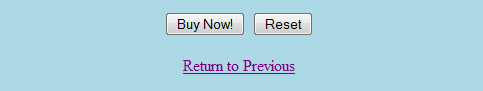
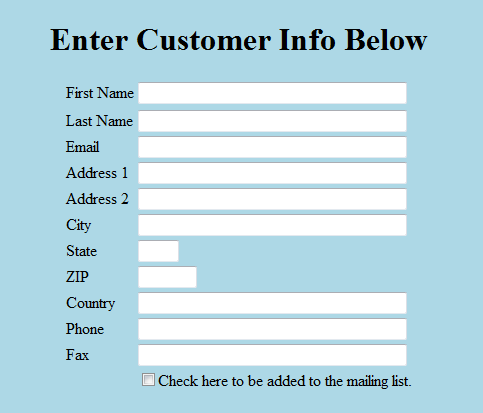
If all items are removed, simply display the following alert box:





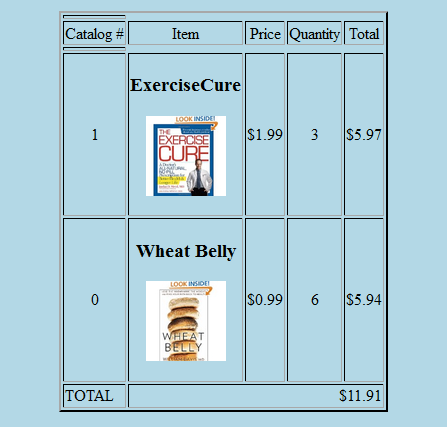
Pressing the *Checkout NOW!* button will display a **modified** checkout.html page from ProgrammingAssignment1 as shown below:





.

\



Your shopping cart displays everything as before with the exception of the remove button. Only the shopping cart logic will be checked for this checkout.html. None of the previous requirements as described in ProgrammingAssignment1 (e.g., validation) will be checked for this assignment with the **exception of the Credit Card Number described below:**

**Credit Card Number Validation**

Once the user completely enters the credit card number - detected by the *onchange* event - your program needs to perform the following validation:

Implement the following algorithm for validating the Credit Card Number Field:

CC field must be 16 digits in length

Even digits are multiplied by two, odd digits are left alone. **Store the results as a string**.

Accumulate all the individual digits of the resulting string into one sum.

If the resulting sum is an even multiple of ten (but not zero), the Credit Card Number is good.

Otherwise, it is an invalid CC and display an appropriate error message using an **alert box.**

*Test your Credit Card Number field using the following test Credit Card Number:*

**4111111111111111**

# Other Considerations

The main objective of this project is to implement a JavaScript shopping cart and Credit Card validation system. You should only use client-side technology (XHTML, CSS, JavaScript) when implementing this project. I will **not** deduct points if your implemented pages do not look exactly like the ones in this document - **but make sure that your pages are neat, readable and meet all functional requirements.**  Feel free to utilize comments within your HTML – it will help you when examining this project in the future.

**Testing Your Project**

Please test your ProgrammingAssignment3 project on a machine **other** than the one you developed on. This will assure that you have no hard-coded paths and your project has no special requirements. Please use *Mozilla or IE* when testing your project (this is what I’ll be using too). You will need to debug with IE if using VS. **Remember, if it doesn’t run on the computers in lab 329/328/338, it most likely won’t run on my lap top when grading your work either!**

**Make sure that your submit all required files for ProgrammingAssignment3. Any missing files will result in deducted points for this assignment.**

**What you Submit**

Please zip up the folder (e.g., ProgrammingAssignment3) where you implement the *complete* project. Submit the compressed file via Blackboard before the due date. Please submit all required files. **It is important to zip up your *complete* folder so I can run your work on our system the same way that you run it.**