AKQA

AKQA Web Developer exercise requirements

Instructions

Please build the page so it matches the Photoshop PSD file using the guidelines below.

- HTML
  We want to see how you have built the page, so please do not use any page template libraries such as the HTML5 Boilerplate.

- CSS
  You are welcome to use a pre-processor such as Sass or LESS but please include the compiled CSS code too.

- JavaScript
  You are allowed to use Modernizr and Babel, along with a single client side JavaScript library such as jQuery if you wish, but you must not use any other plugins, libraries or frameworks (such as React or Angular).

- Task Runners
  You are welcome to use a task runner such as Grunt, Gulp or Webpack but please ensure you provide instructions on how to start-up your project.

When submitting your completed exercise, please specify any tools you have used, including the editor you used to write your code and any developer tools.

General requirements and considerations

Generally, you should build the web page in the way that you feel it should be built, however the following should be taken into consideration:

- Accessibility:
  Ensure that your mark-up takes into account basic accessibility standards. For example, ensure form fields are accessible; that the page can be viewed and used without the need for JavaScript to be enabled and any alt attributes and similar are set. You will not be expected to write or connect up any server-side code for this exercise, merely to show your understanding of how such a page might work without JavaScript. Please also consider page weight when saving images.

- Standards compliance:
  Ensure that your mark-up is W3C compliant.

- Semantics:
  Ensure that your mark-up is semantic. Information should follow a logical flow throughout the document and that information is displayed and marked up using the most appropriate tag.

- JavaScript:
  Ensure you write neat, tidy JavaScript that you would consider suitable for a live web site.

- **Browsers:**
  The page should function and resemble the creative in the following web browsers:
  - *Google Chrome (latest)*
  - *Mozilla Firefox (latest)*
  - *Safari (latest)*
  - *Microsoft Edge (latest)*

  Please also consider how your page might display on devices of different screen sizes, including mobile.

'Your Basket' page

The page shown in the creative depicts a user's basket on a fictitious online shopping site. Three products have been added to the basket and the names of these are shown together with their cost and the quantity desired for purchase. Beneath the product list, a sub-total of the product costs are shown, VAT is then added (at 20%) and the total is shown beneath. Finally, a 'Buy Now' button allows the user to go ahead and purchase the items shown at the total price given.

The fonts shown in the creative are Droid Sans and Droid Serif, available for use in your code from http://www.google.com/fonts.

The 'Your Basket' page should implement client-side functional behaviour as follows:

- The quantity field shown beside each product should be pre-filled with the initial quantities shown in the creative. Selecting the **+** button beside the field should increment the quantity by 1. Selecting the **–** button should decrement the quantity by 1. The user should also be free to enter any positive number within the quantity field, where only whole numbers from 1 to 10 should be permitted for each product. Note: If necessary, the visual layout of the quantity field may differ from the creative if enforced so by the browser.

- Whenever a quantity is changed beside any product, the cost beside should update to reflect the price of that product multiplied by its quantity. The sub-total beneath the product list should update to reflect the sum of all the costs. The VAT cost should also update whenever the sub-total changes, as should the final total cost, which is the sum of the sub-total and the VAT.

- When selected, the delete button, indicated as a trashcan icon, should remove the item beside it entirely from the basket even if it would result in no products remaining in the basket. The totals beneath the product list should be updated accordingly whenever a product is deleted. If no products remain in the basket, the 'Buy Now' button should be disabled.

- When there and products in the basket and the 'Buy Now' button is selected, you should take the product data together with the quantities, costs and totals, and simulate how you would POST this data in JSON format to an Ajax-supported web service URL. As this is purely for simulation purposes in this exercise, you may use a blank URL for this. Display an alert message on screen once the Ajax call is complete.

If you find yourself running short on time, please code as much as you can and then describe how you plan to complete the test, through the use of comments and pseudo-code in your HTML, CSS, and JavaScript code.