Exercise Unit01 – Laying out a Page

Convert a basic HTML 4 page to an HTML5 page. The

CSS documents have already been created for you, such that, when you're finished, your HTML5 page should render exactly like the HTML4 page.

1. Open Unit01-LayingOutAPage/html4-layout.html
2. Save the file as html5-layout.html.
3. Turn the page from an HTML 4 page into an HTML5 page. Make sure to change the stylesheet reference to point to style-html5.css.
4. Challenge: …

Exercise Unit 2 – Create the App

In this exercise, you will convert a basic HTML 4 page to an HTML5 page. The

CSS documents have already been created for you, such that, when you're finished, your HTML5 page should render exactly like the HTML 4 page.

1. Add an exercise?

Exercise Unit 3

a. Sections

In this exercise, you will modify an HTML page we worked on earlier in the course to replace meaningless div elements with meaningful section and article elements.

1. Open html5-sections/Exercises/html5-layout.html.
2. Replace meaningless div elements with meaningful section and article

elements. Note that there is room for interpretation here, so there is no one correct solution.

1. To keep the page looking as it did before, you will also need to modify html5-sections/Exercises/style-html5.css.

b. Determining the Outline

In this exercise, you will try to determine the outline of an HTML page.

1. Review the code below.

2. Create a list either on paper or in a text editor or word processor that shows the HTML outline as specified by the HTML5 specification.

Exercise Unit 4 – Audio and Video

In this exercise, you will create an HTML5 file from scratch that plays video files.

1. Create a new HTML5 file called video-multiple-sources.html in the html5-audio-and-video/Exercises directory.
2. Write the code to include the justin.mp4 (mime type is video/mp4) and

justin.ogv (mime type is video/ogg) files as source options. Both files are located in the html5-audio-and-video/Media directory.

1. Play around with video attributes such as: controls, autoplay, and loop.

Exercise Unit 5 – Forms

In this exercise, you will create an HTML5 quiz that validates form entries and reports the percentage of both the valid (but not necessarily correct) answers and the percentage of correct answers.

1. Open html5-forms/Exercises/quiz.html in your editor.
2. Make the following changes to the form:
3. Add placeholders to all questions.
4. Make all questions required.
5. Question 1 should only accept valid colors.
6. Question 2 should only accept integers greater than or equal to 20.
7. Question 3 should only accept the pattern shown in the footnote below18 (don't look if you want to figure out the pattern yourself).
8. Question 4 should only accept valid dates.
9. Question 5 should only accept valid URLs and should provide a list of common search engines to choose from, but should not limit the answer to those shown in the list.
10. At the bottom of the form:
11. Add a bar showing the percentage of valid (but not necessarily correct) answers answered. Give it an id of "quiz-progress".
12. Add a bar showing the percentage of correct answers answered. Give it an id of "quiz-success".

4. Finish the updateMeasures() function so that it correctly updates the two bars added above on every form change. Hint: one way to do this is to loop through the input fields stored in the questions variables.

**Challenge:**

1. Add code so that the result of the formula the user enters in question 3 is displayed next to the input field like this:
2. Fix the two bars at the bottom of the form so that they present as follows in Opera and other browsers that do not support the progress and meter elements: You'll need to change both your HTML and JavaScript to make this work. Page 102

Exercise Unit 6 – Storage

In this exercise, you will create a quiz application that allows the user to save and resume later. It also protects the user from losing data if he/she accidentally refreshes.

1. Open html5-storage/Solutions/saving-quiz-challenge.html in your browser and play with the application:
   1. Answer one or more questions and then refresh the browser. You will get a dialog giving you the chance to use refill the form with values stored in sessionStorage:
   2. Press OK and your values get added back.
   3. Refresh again and press Cancel on the dialog. Your values do not get added back.
   4. Refresh again. You don't get the JavaScript dialog because the sessionStorage key/value pairs were removed when you pressed Cancel in the previous step.
   5. Answer one or more questions again and click the Save My Answers for Later button.
   6. Close and reopen the browser. You will get a dialog giving you the chance to use refill the form with values stored in localStorage:
   7. If you click Cancel, the localStorage key/value pairs will be removed and you'll have to start the quiz over.
   8. If you click OK, the form will be refilled with your previous answers and they will be saved into sessionStorage.
   9. Also notice that the footer contains the time and date the quiz answers were last saved
2. Now open html5-storage/Exercises/saving-quiz.html in your editor. Page 120 of 180 © Copyright 2011 Webucator. All rights reserved. HTML5 Web Storage
3. In the the addLoadEvents() function:
   1. Loop through the inputs and add event listeners that capture change events to save the associated key/value pair in sessionStorage. Don't forget to use the prefix.
   2. Add an event listener to the Save button to capture a click event and call saveAnswers.
   3. Call the refill() function at the end.
4. Write the code in the saveAnswers() function to save all the answers in localStorage.
5. The refill() function:
   1. calls hasAnswers(), which returns "session", "local", or false,

depending on if and where it finds saved answers. If there are no saved answers, it returns without doing anything.

* 1. declares some variables:
     1. confirmed - we'll change it to true if the user wishes to refill the form.
     2. msg - the message to ask the user if he/she wants to refill the form.
     3. questions - the question inputs
  2. loops through the inputs. On the first iteration, it prompts the user with the message. If the user clicks Cancel, the key/value pairs are deleted (via the deleteAnswers() function) and the function returns/ends. Otherwise, we iterate through the questions. This is where you come in.
  3. Add code to populate the question inputs from the appropriate storage location (based on the value of fillFrom).

Challenge:

1. Notice that an external script called dateFormat.js is included. That extends the Date object prototype with a format() method, which you use as follows:

var now = new Date();

var dateMask = "yyyy-mm-dd H:MM:ss";

var formattedNow = now.format(dateMask);

1. Use this to write out the date last saved to the output element below the form.
2. You will need to store the date in sessionStorage and localStorage as appropriate. Don't forget the prefix.
3. Note that the dateMask variable is already set in the code.

Exercise Unit 7 – Canvas

1. Sailboat

In this exercise, you will use HTML5 canvas to draw a simple sailboat like the one shown below:

1. Open html5-canvas/Exercises/sailboat.html in your editor.
2. Add the JavaScript code necessary to draw the sailboat pictured above.

Challenge:

Have the left sale blink between different colors. You'll need to use some JavaScript skills to make this happen.

If JavaScript isn't your thing, try adding a little person on your sailboat.

1. Snowman

In this exercise, you will use circles and squares to create a snowman like the one pictured below:

1. Open html5-canvas/Exercises/snowman.html in your editor.
2. Add the JavaScript code necessary to draw the snowman pictured above. You will need to add:
   1. A layer of snow on the ground.
   2. Three balls for the body and head.
   3. Eyes, mouth and nose.
   4. A hat.
   5. Arms.
   6. Buttons.
   7. A sun.

Challenge:

In this exercise, you will start with two images found in the html5-canvas/Exercis es/Images folder:

1. south-america.gif - a map of South America.
2. flags.png - a picture containing small graphics of country flags.

You will create the following drawing:

Notice the text under the graphics.

Version: 1.1.1. Printed: 2011-08-10. Page 161 of 180 HTML5 Canvas

1. Open html5-canvas/Exercises/south-america in your editor.
2. Add the JavaScript code necessary to:
   1. Create the image objects and set their source values.
   2. Draw the backdrop (the map).
   3. Place the flags using the sprite method shown earlier. Each flag is 18 pixels wide and 13 pixels high. The source and destination positions are shown in the table below.
   4. Add the country names.

|  |
| --- |
| Country Source X Source Y Destination X Destination Y |
| Chile 283 88 100 250 |
| Argentina 255 4 130 300 |
| Brazil 171 60 200 170 |
| Paraguay 59 452 170 250 |
| Uruguay 59 564 185 310 |
| Bolivia 59 200 135 210  Peru 31 424 75 170 |

Exercise Unit 8 – APIs

In this exercise, you will convert a basic HTML 4 page to an HTML5 page. The

CSS documents have already been created for you, such that, when you're finished, your HTML5 page should render exactly like the HTML 4 page.

1. Open html5-laying-out-a-page/Exercises/html4-layout.html.
2. Save the file as html5-layout.html.
3. Turn the page from an HTML 4 page into an HTML5 page. Make sure to change the stylesheet reference to point to style-html5.css.

Exercise Unit 9 – Animation

In this exercise, you will convert a basic HTML 4 page to an HTML5 page. The

CSS documents have already been created for you, such that, when you're finished, your HTML5 page should render exactly like the HTML 4 page.

1. Open html5-laying-out-a-page/Exercises/html4-layout.html.
2. Save the file as html5-layout.html.
3. Turn the page from an HTML 4 page into an HTML5 page. Make sure to change the stylesheet reference to point to style-html5.css.

Exercise Unit 10 – Unit Testing

ADD..