# JavaJam Coffee House Case Study – Textbook - Chapter 10

# In this chapter’s case study, you will use the existing JavaJam (Chapter 9) website as a starting point. You will add a new page to the JavaJam website – the Jobs page. Refer back to the site map for the JavaJam website in chapter 2’s tutorial. The Jobs page will use the same page layout as the other JavaJam web pages. You’ll apply your new skills from this chapter and code a form in the content area of the Jobs page.

# You have four tasks in this case study:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Copy yourname\_chapter8 JavaJam Coffee House website folder and rename it yourname\_chapter10.Edit the javajam.css external style sheet to configure style rules for the new Jobs page.Create the Jobs page: jobs.html. Your new page will be similar to Figure 10.37 when you have completed this step.Configure HTML5 form control features on the Jobs page. | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Form  |  |  |  |  | | --- | --- | --- | --- | | label |  | text box | | |  |  |  |  | | label |  | text box | | |  |  |  |  | | label |  | date | | |  |  |  | | | label |  | scrolling text box | | | | |  |  | | --- | --- | |  | submit button | |  FIGURE 10.38 The sketch of the form. |

# 

# 

# FIGURE 10.37 The new JavaJam Jobs page.

# Task 1: Copy yourname\_chapter8 JavaJam Coffee House website folder and rename it, yourname\_chapter10. Task 2: Configure the javajam.css. Review Figure 10.37 and the wireframe in Figure 10.38. Notice how the text labels for the form controls are on the left side of the content area but contain right-aligned text. Notice the empty vertical space between each form control. Open javajam.css in a text editor. Place your cursor on a new blank line above the media queries.

# Configure a label element selector. Set left float, block display, left alignment for text, assign a width of 120 pixels, and an appropriate amount of right padding.

# Configure the input element and textarea element selectors. Set block display and 20 pixels to bottom margin.

# Configure the submit button with a 130px left margin.

# Configure style rules to optimize the display on smart-phone-sized mobile devices by rending the label text above each form control and eliminating the submit button’s left margin. Add the following style rules to the media query that targets a maximum width of 37.5em.

# label {float: none; text-align: left;}

# input [type=”submit”] {margin-left: 0;}

# Save the javajam.css file. Validate your css.

# Task 3: Create the Jobs Page. Use the menu page as the starting point for the Jobs page. Launch a text editor and open menu.html. Save the file as jobs.html. Modify you jobs.html file to look similar to the Jobs pages (shown in Figure 10.37) as follows:

# Change the page title to an appropriate phrase.

# Replace the text contained with the <h2> tags with: Jobs at JavaJam.

# The Jobs page will contain a paragraph and a form in the main content area. Delete the table in the main element. Add a paragraph that contains the following text: Want to work at JavaJam? Fill out the form below to start your application.

# Prepare to code the HTML for the form area. Begin with a form element that uses the post method and the action attribute to invoke server-side processing. Unless directed otherwise by your instructor, configure the action attribute to send the form data to <https://webdevbasics.net/scripts/javajam.php> .

# Configure the form control for the Name information. Create a label element that contains the text “Name:”. Create a text box, configure with “myName” as the value of the id and name attributes. Use the for attribute to associate the label element with the form control.

# Configure the form control for the E-mail information. Create a label element that contains the text “E-Mail:”. Create an email type text box, configure with “myEmail” as the value of the id and name attributes. Use the for attribute to associate the label element with the form control.

# Configure the form control for the Start Date information. Create a label element that contains the text “Start Date:”. Create a date control, configure with “myStart” as the value of the id and name attributes. Use the for attribute to associate the label element with the form control.

# Configure the form control for the Experience area on the form. Create a label element that contains the text “Experience:”. Create a textarea, configure with “myExperience” as the value of the id and name attributes. Style with CSS if you’d like, by targeting the textarea element by its ID and defining your own height and width. Use the for attribute to associate the label element with the form control.

# Configure the submit button. Code an input element (use type=”submit” and value=”Apply Now”)

# Code an ending </form> tag on a lank line after the submit button.

# Save your file and test your web page in a browser. It should look similar to the page shown in Figure 10.37. If you are connected to the Internet, submit the form. This will send your form information to the server-side script configured in the form tag. A confirmation page similar to Figure 10.39 will be displayed that lists the form control names and the values you entered.

# Task 4: Configure the From with HTML5 Attributes and Values. Get more practice with the new HTML5 elements by modifying the form on the Jobs page to use HTML5 attributes and values. Modify the jobs.html file in a text editor.

# Add the following sentence to the paragraph above the form: Required fields are marked with an asterisk \*.

# Use the required attribute to require the name, e-mail, and experience form controls to be entered. Add an asterisk at the beginning of each label text.

# Configure the input element for the e-mail address (use type=”email”).

# Code the label element containing the text “Start Date” that is associated with a calendar form control to accept the date that the applicant is available to start the job (use type=”date”).

# 

# FIGURE 10.39 The form confirmation page.

# If you do not see the data posted to the <https://webdevbasics.net/scripts/javajam.php> page, do not worry about it. This .php file is not another server and may have been changed. The id(s) and name attributes in our input tags have to match the name and id(s) in the .php page. The author may have changed the code. Hopefully, you had success with this in the AwesomeCo tutorial and understand the concept.

# Save your jobs.html file and display your web page in a browser. Submit the form with missing information or only a partial e-mail address. Depending on the browser’s level of HTML5 support, the browser may perform form validation and display an error message.

# Task 4 in this case study provided you with additional practice using the new HTML5 attributes and values. The display and functioning of browsers will depend on the level of HTML5 support. See <http://www.standardista.com/html5/html5-web-forms> for an HTML5 browser support list.