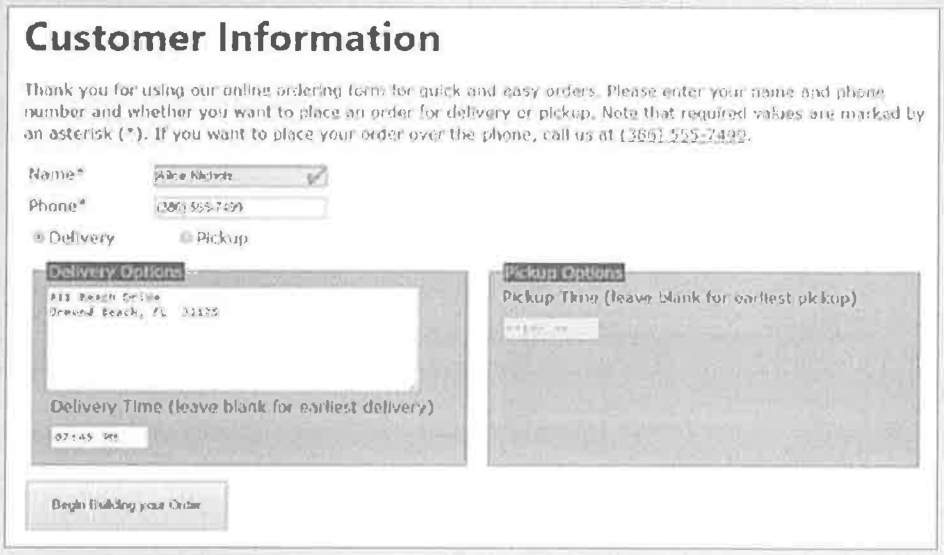
PRACTICE

Review Assignments

Data Files needed for the Review Assignments: rb_build_txt.html, rb_customer_txt.html, rb_validate_txt.css, 3 CSS files, 1 JavaScript file, 10 PNG files, 1 TXT file

Alice wants you to start work on an online order form for customers to place orders through the Red Ball Pizza website. The form will span several pages in which customers will specify whether the order is for pickup or delivery and will indicate the toppings they want on their pizza(s). Figure 7-59 shows a preview of the form customers will use to indicate their delivery option (including an address or pickup and at what time they want their order).

Figure 7-59 **Red Ball Pizza form for Customer Data**



Customer Information

Thank you for using our online ordering form for quick and easy orders. Please enter your name and phone number and whether you want to place an order for delivery or pickup. Note that required values are marked by an asterisk (*). If you want to place your order over the phone, call us at (386) 555-7499.

Name*	Alice Nichols
Phone*	(386) 555-7499

● Delivery ○ Pickup

Delivery Options
Pickup Options
Pickup Time (leave blank for earliest pickup)
Delivery Time (leave blank for earliest delivery)
Begin Building your Order

Alice has already written some of the HTML code for the web pages and designed many of the style sheets. Your job will be to write the code for the form elements and validation styles.

Complete the following:

1. Use your HTML editor to open the **rb_customer_txt.html**, **rb_build_txt.html**, and **rb_validate_txt.css** files from the html07 ▸ review folder. Enter ***your name*** and ***the date*** in the comment section of each file, and save them as **rb_customer.html**, **rb_build.html** and **rb_validate.css** respectively.
2. Return to the **rb_customer.html** file in your editor. Within the document head, insert links to the rb_forms2.css and rb_validate.css files.
3. Still within the document head, use the `script` element to link the file to the rb_formsubmit2.js file.
4. Scroll down to the `section` element and, directly after the initial paragraph, insert a `form` element that employs the action at the fictional address *`http://www.example.com/redball/customer`* using the post method.
5. Within the `form` element, insert a `div` element that encloses a label with the text **Name*** associated with the nameBox control. Also, within the `div` element, add an input text box with the ID **nameBox**, field name **custName**, and placeholder text **First and Last Name**. Make custName a required field.
6. Create a second `div` element in the web form that encloses a label with the text **Phone*** associated with the phoneBox control. Within the `div` element, add an input box with the ID **phoneBox**, field name **custPhone**, and placeholder text **(nnn) nnn-nnnn**. Make custPhone

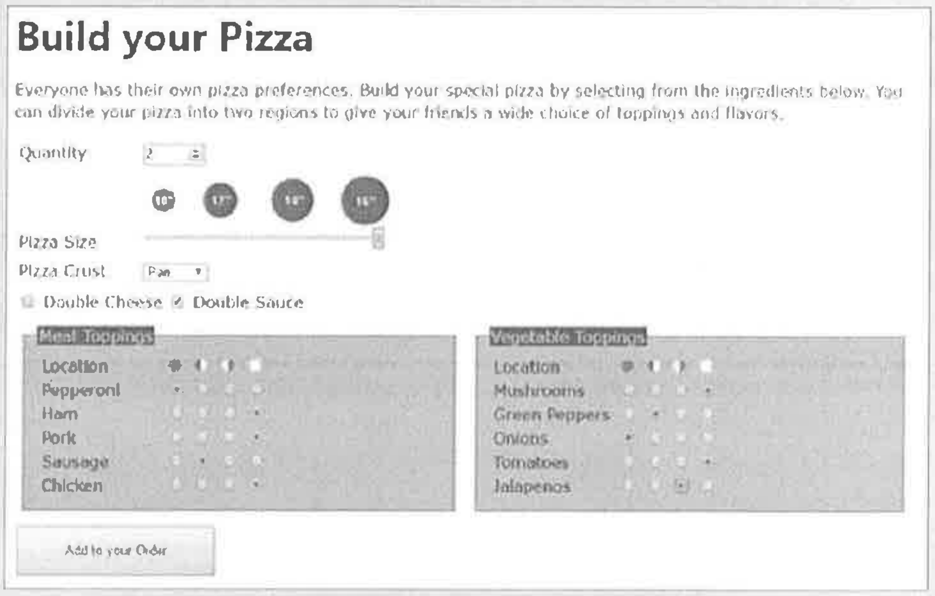
a required field and have any text entry follow the regular expression pattern **^\d{10}$|^(\(\d{3}\)\s*)?\d{3}[\s-]?\d{4}$**. (Note: You can copy the regular expression code from the rb_regex2.txt file.)

7. Add another `div` element to the web form containing the following code:
   a. Insert an `input` element to create an option button for the **orderType** field with the ID **delivery**. Make the option button checked by default. After the option button, insert a label associated with the delivery control containing the text **Delivery**.
   b. Add an `input` element to create a second option button for the **orderType** field with the ID **pickup**, followed by a label associated with the pickup control containing the text **Pickup**.
8. Next within the form, create a field set with the ID **deliveryInfo**. Within this field set, add the following:
   a. A legend containing the text **Delivery Options**.
   b. A text area box with the ID **addressBox** and field name of **delAddress** containing the placeholder text **Enter delivery address**.
   c. A label containing the text **Delivery Time (leave blank for earliest delivery)** associated with the delBox control.
   d. Add an `input` element with the ID **delBox** and field name **delTime** for storing delivery time values. Use a data type of "time" for the control.
9. Next within the web form, create a field set with the ID **pickupInfo** containing the following information for pickup orders:
   a. A legend containing the text **Pickup Options**.
   b. A label containing the text **Pickup Time (leave blank for earliest pickup)** associated with pickupBox control.
   c. Add an `input` element with the ID **pickupBox** and field name **pickupTime** for storing time values. Add the `disabled` attribute to the tag to disable this control when the form is initially opened. Use a data type of "time" for the control.
10. Finally, within the form, add a `div` element containing a submit button displaying the text **Begin Building your Order**.
11. Save your changes to the file and then go to the **rb_validate.css** file in your editor to add validation styles for the web form.
12. Within the Validation Styles section, add the following style rules:
   a. A rule that displays `input`, `select`, and `textarea` elements that have the focus with a background color of rgb(255, 255, 180).
   b. A rule that displays the nameBox and phoneBox controls that have the focus and contain valid data with a background color of rgb(220, 255, 220) and the background image file rb_okay.png at the right with no tiling contained within the background.
   c. A rule that displays the nameBox and phoneBox controls that have the focus and invalid data with a background color of rgb(255, 230, 230) and the background image file rb_warning.png at the right with no tiling contained within the background.
13. Save your changes to the style sheet and then open the **rb_customer.html** file in your browser. Verify the following:
   a. The content and the layout of the form resemble the form shown in Figure 7-59.
   b. If you submit the form by clicking the Begin Building your Own button with no customer name or phone number, the browser warns you of the missing values.
   c. As you enter text into the custName field, the input box background changes to show that the field value is valid.
   d. When you enter a phone number into the custPhone field, the input box provides inline validation to indicate whether a valid phone number has been entered.
   e. When you click the submit button for a successfully completed form, the browser displays the alert message that the form data passes the initial validation test.

   (Note: The script file used with this web page is written to enable only either the delivery option or the pickup option but not both.)

Next, you will create a form that customers will use to build their customized pizzas. A preview of the form is shown in Figure 7-60.

Figure 7-60 **Red Ball Pizza form to Build a Pizza**



Build your Pizza

Everyone has their own pizza preferences. Build your special pizza by selecting from the ingredients below. You can divide your pizza into two regions to give your friends a wide choice of toppings and flavors.

Quantity 2

Pizza Size

Pizza Crust Pan

Double Cheese ✓ Double Sauce

Meat Toppings

Location

Pepperoni

Ham

Pork

Sausage

Chicken

Vegetable Toppings

Location

Mushrooms

Green Peppers

Onions

Tomatoes

Jalapenos

14. Return to the **rb_build.html** file in your editor. Insert a link to the rb_forms2.css file and add a `script` element to link the file to the rb_formsubmit2.js file.
15. Scroll down to the `section` element, insert a `form` element below the paragraph element that employs the action at the fictional address *`http://www.example.com/redball/build`* using the `post` method.
16. Within the `form` element, add a `div` element containing a label with the text **Quantity** associated with the quantityBox control. Add a spinner control with the ID **quantityBox** and the field name **pizzaQuantity**. Have the value of the field range from 1 to 10 with a default value of 1.
17. Add a `div` element that displays images of the pizza sizes, containing the following:
    a. The inline image rb_sizes.png.
    b. The label **Pizza Size** associated with the sizeBox control.
    c. A range slider with the ID **sizeBox** and the field name **pizzaSize** ranging from 10 to 16 in steps of 2 with a default value of 14.
18. Add a `div` element that provides the selection of pizza crusts containing the following:
    a. The label **Pizza Crust** associated with the crustBox control.
    b. A selection list for the **pizzaCrust** field with the ID **crustBox** and containing the following option values and text: **Thin**, **Thick**, **Stuffed**, and **Pan**.
19. Add a `div` element containing a check box with the ID **cheeseBox** for the **doubleCheese** field followed by the label **Double Cheese** associated with the cheeseBox control. Then, add a second check box with the ID **sauceBox** for the **doubleSauce** field followed by the label **Double Sauce** also associated with that check box.

20. Customers can choose what to place on their pizzas. Create a field set containing the legend **Meat Toppings**. Add the following content to the field set.
    a. A `div` element containing the label **Location** but not associated with any form control. Next to the label, place the inline images full.png, left.png, right.png, and none.png with the alternate text "full", "left", "right", and "none" used to graphically indicate where the meat ingredients should be placed on the pizza (on the full pie, the left side, the right side, or nowhere).
    b. A `div` element containing the label **Pepperoni** and followed by four option buttons belonging to the **pepperoni** field and with the values "full", "left", "right", and "none". Make "none" checked by default.
    c. Repeat Step b to insert `div` elements with the values used in Step b but associated with the ham, pork, sausage, and chicken fields.
21. Using Figure 7-60 as your guide, repeat Step 20 to create a field set with the legend **Vegetable Toppings**, followed by `div` elements with the values used in Step 20 but associated with the mushrooms, peppers, onions, tomatoes, and jalapenos fields.
22. At the bottom of the form, add a `div` element containing a submit button with the text **Add to your Order**.
23. Save your changes to the file and then open **rb_build.html** in your browser. Verify that the content and layout of the form resemble that shown in Figure 7-60. Verify that all of the form controls work as expected, that is, you can only select one location for each ingredient option at a time.

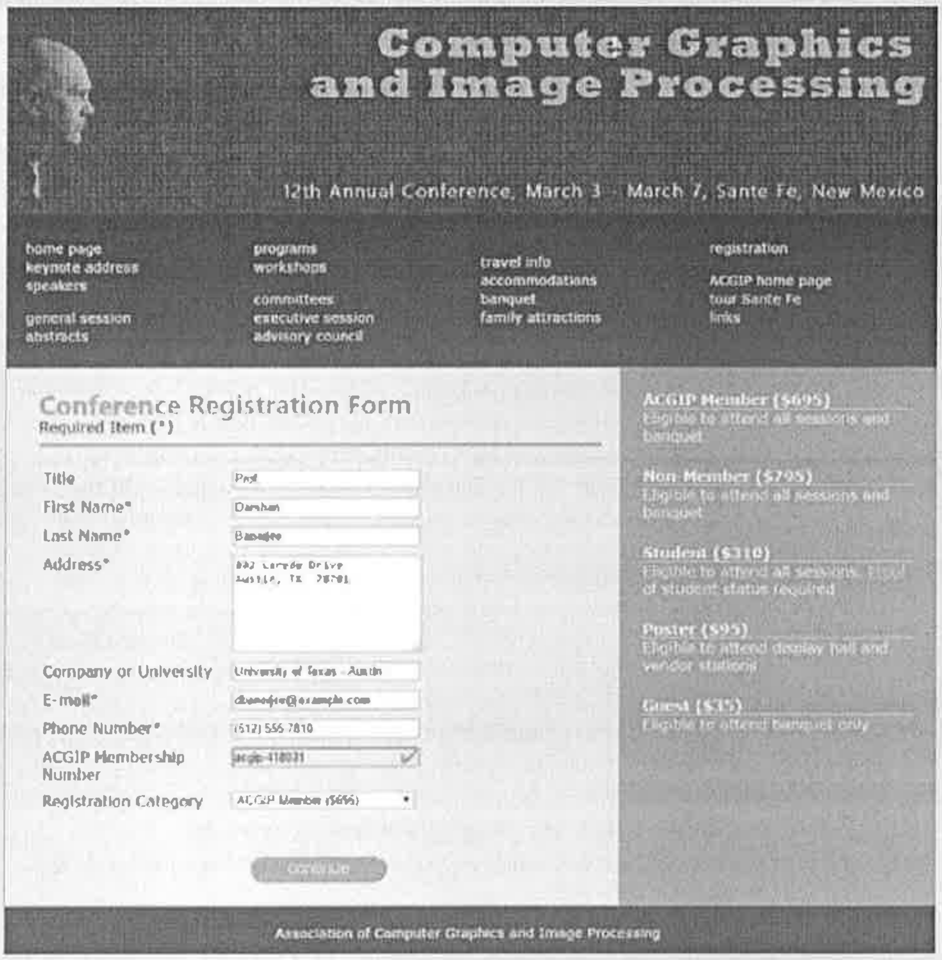
APPLY

Case Problem 1

Data Files needed for this Case Problem: cg_register_txt.html, cg_validate_txt.css, 3 CSS files, 1 JavaScript file, 4 PNG files, 1 TXT file

ACGIP Conference Professor Darshan Banerjee is the project coordinator for the annual conference of the Association of Computer Graphics and Image Processing (*ACGIP*), which takes place this year in Sante Fe, New Mexico. Darshan has asked you to work on the conference's website, starting with the registration form for conference attendees. The initial form will collect contact information for people attending the conference. Figure 7-61 shows a preview of the form you will create for Darshan.

Figure 7-61 Registration form for the ACGIP Conference


Computer Graphics and Image Processing

12th Annual Conference, March 3 - March 7, Sante Fe, New Mexico

home page
keynote address
speakers

general session
abstracts

programs
workshops

committees
executive session
advisory council

travel info
accommodations
banquet
family attractions

registration

ACGIP home page
tour Sante Fe
links

Conference Registration Form
Required Item (*)

Title	Prof
First Name*	Darshan
Last Name*	Banerjee
Address*	882 Laredo Drive Austin, TX 78701
Company or University	University of Texas - Austin
E-mail*	dbanerjee@example.com
Phone Number*	(512) 555 7810
ACGIP Membership Number	acgip-118031
Registration Category	ACGIP Member ($695)

Continue

ACGIP Member ($695)
Eligible to attend all sessions and banquet

Non-Member ($795)
Eligible to attend all sessions and banquet

Student ($310)
Eligible to attend all sessions. Proof of student status required

Poster ($95)
Eligible to attend display hall and vendor stations

Guest ($35)
Eligible to attend banquet only

Association of Computer Graphics and Image Processing

Professor Banjerjee has already written the HTML code for the page and the styles for the form elements. He wants you to write the HTML code for the web form and the CSS validation styles. Complete the following:

1. Using your editor, open the **cg_register_txt.html** and **cg_validate_txt.css** files from the html07 ▸ case1 folder. Enter ***your name*** and ***the date*** in the comment section of each file, and save them as **cg_register.html** and **cg_validate.css** respectively.
2. Return to the **cg_register.html** file in your editor. Add a link to the cg_forms.css and cg_validate.css style sheet files to the document head.
3. Add a `script` element to the document head that loads the cg_script.js file.
4. Scroll down to the `section` element and insert a web `form` element that employs the action at *http://www.example.com/cg/register* via the `post` method.
5. Add the labels and input boxes shown previously in Figure 7-61 and described in Figure 7-62. Place the input boxes directly after the labels and associate each label with its input box control. You do not need to enclose the `label` and `input` elements with `div` elements.

Figure 7-62 **Fields and controls from the registration form**

Label	Data Field	Control ID	Type	Required	Placeholder
Title	title	titleBox	text	no	
First Name*	firstName	fnBox	text	yes	
Last Name*	lastName	lnBox	text	yes	
Address*	address	addBox	text	yes	
Company or University	group	groupBox	text	no	
E-mail*	email	mailBox	email	yes	
Phone Number*	phoneNumber	phoneBox	tel	yes	(nnn) nnn-nnnn
ACGIP Membership Number	acgipID	idBox	text	no	acgip-nnnnnn

6. Create a data list named **titleList** containing the suggestions: Mr., Mrs., Ms., Prof., Dr., Assist. Prof., and Assoc. Prof. Apply the titleList data list to the titleBox control.
7. Apply the regular expression pattern **^\d{10}$|^(\(\d{3}\)\s*)?\d{3}[\s-]?\d{4}$** to the phoneNumber field. Apply the regular expression pattern **^acgip\-\d{6}$** to the acgipID field. (Note: You can copy the regular expression code for the phoneNumber field from the cg_regex.txt file.)
8. Add the **Registration Category** label associated with the regList control. Add a selection list with the ID **regList** that stores values in the **registerType** field. Populate the selection list with the option text: "ACGIP Member ($695)", "Non-Member ($795)", "Student ($310)", "Poster ($95)", and "Guest ($35)". Make the corresponding option values equal to "member", "nonmember", "student", "poster", and "guest".
9. Within the form, add a paragraph containing a submit button with the text **continue**.
10. Save your changes to the file and return to the **cg_validate.css** file in your editor to create styles for validating data entry.
11. Within the Validation Style section, add the following style rules:
    a. Display all `input`, `select`, and `textarea` elements that have the focus with a background color of rgb(245, 245, 140).
    b. When the fnBox, lnBox, addBox, mailBox, phoneBox, and idBox controls have the focus and are valid, change the background color to rgb(220, 255, 220) and display the cg_valid.png image with no tiling in the right side of the background contained within the box.
    c. When the fnBox, lnBox, addBox, mailBox, phoneBox, and idBox controls have the focus and are not valid, change the background color to rgb(255, 232, 232) and display the image cg_invalid.png with no tiling in the right side of the background contained within the box.
12. Save your changes to the style sheet and then open **cg_register.html** in your browser. Verify that the content and layout of the form resemble that shown in Figure 7-61. Verify that you must enter all required field values in the proper format for the form to be submitted successfully. Confirm that the browser performs inline validation on the firstName, lastName, address, email, phoneNumber, and acgipID fields.