Chapter

5

Adding HTML buttons to a report

This chapter contains the following topics:

- Using HTML buttons
- Examining event handlers
- Creating an HTML button
- Changing the appearance of an HTML button

Using HTML buttons

An HTML button is a report element that supports adding HTML button-type elements to your report. The HTML button element can execute client-side JavaScript code based on button events. Button events are handled through event handlers in the same way the HTML <button> tag is on a web page.

You can use an HTML button for several types of applications. For example, you can create a BIRT report which lists stock holdings that has an HTML button saying More adjacent to each row. Choosing the More button would show additional information about the transaction. Alternatively, you could create a BIRT report which lists product details like name, image, price, shipping price, with a button called Calculate on each row. Clicking the Calculate button runs custom JavaScript that calculates the shipping cost based on zip code. You can also create several buttons for a transaction that give different information based on a specified set of information.

HTML buttons can be of any size and can have different fonts, colors, or images on their face. Data from within the BIRT report can be used on the HTML button element to customize its appearance.

Figure 5-1 shows a report with three buttons in its output row. This example shows buttons for displaying the customer credit limit, a past due amount, and to dial a phone number through a phone connected to the computer running the report.

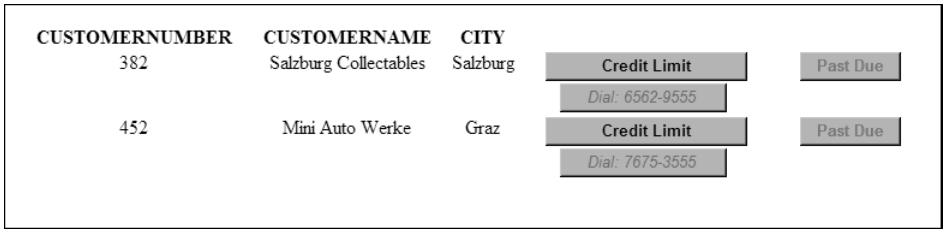

Figure 5-1 Report output with HTML buttons

Each of these buttons is designed so that it can perform actions separate from the others while still pertaining to the information displayed for the row.

Examining event handlers

Event handlers are pieces of code written JavaScript that are executed when a certain activity takes place. These activities are called events, and the event handler executes when they occur. You can place multiple event handlers on an HTML button to execute different routines based on the type of event that occurred. Through this mechanism, you can control the behavior of the HTML button element to do such things as display

messages, calculate values, or execute other pieces of code. Table 5-1 lists the events that HTML button supports when they are activated.

Table 5-1 Supported events

Event	Description
onblur	Occurs when the button loses focus, or stops being active
onclick	Occurs when the button is clicked
ondblclick	Occurs when the button is double-clicked
onfocus	Occurs when the button gets focus, or becomes active
onkeydown	Occurs when a keyboard key is pressed
onkeypress	Occurs when a keyboard key is pressed or held down
onkeyup	Occurs when a keyboard key is released
onmousedown	Occurs when a mouse button is pressed
onmousemove	Occurs when the mouse is moved
onmouseout	Occurs when the mouse is moved off the element
onmouseover	Occurs when the mouse is moved over the button
onmouseup	Occurs when a mouse button is released

When you create an HTML button element, you can use the script tab to create or change event handlers. When you add an event handler to an HTML button, a JavaScript template is provided for you to code, as shown in Figure 5-2.

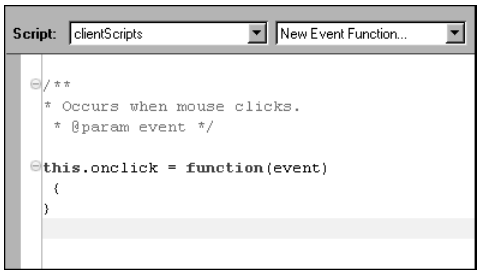

Figure 5-2 Provided template script

This line of code is a signal to the software to execute the code in the following braces when a click, or button press, event occurs.

```
this.onclick = function(event)
```

If you modify this line of code, the event handler will not operate properly. Inside the empty braces following this statement, you place the JavaScript code for the needed activity.

If several event handlers are used for a single HTML button, they are all placed within the script tab for the button and appear serially within the tab, as shown in the following code:

```
/**
* Occurs when mouse clicks.
 * @param event */

this.onclick = function(event)
 {
 /* click code here */
}

/**
* Occurs when mouse is moved off.
 * @param event */

this.onmouseout = function(event)
 {
 /* mouse off button code here */
}

/**
* Occurs when a mouse button is released.
 * @param event */

this.onmouseup = function(event)
 {
 /* mouse button up code here */
}
```

Your code for these event handlers can consist of any valid JavaScript code. From within the event handlers you can access BIRT report data or use the Actuate Web 2.0 API. For more information on event handlers and how to use them in BIRT, see *Integrating and Extending BIRT*.

Accessing report data

Use variables on the HTML button element to access data within the report for the event handlers. To create variables for use in an event handler, use the Variables tab in the HTML button property editor. Figure 5-3 shows an HTML button variable named Payment with a computed value based on the balance item from the row.

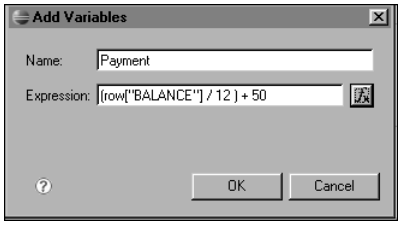

Figure 5-3 Computed value in variable

When the variables are created, you use dataObject to access them in event handlers. For example, to display the variable Payment when a button is clicked, use the following event handler code:

```
**
* Occurs when mouse clicks.
 * @param event */

this.onclick = function(event)
{
   alert("Customer payment information below = " +
        "\n" + dataObject.Payment );
}
```

The onclick event handler is activated when the button is clicked. Control then passes to the code which is located between the braces. In this example, the JavaScript alert function is called to display information in a message box with information taken from the HTML button element variable, Payment. You can also use the HTML button variables in the event handler to compute other values as necessary.

Accessing the Actuate Web 2.0 API

Actuate provides the Web 2.0 API to support integration of Actuate technology into web applications. You can leverage the Actuate Web 2.0 Report API to embed entire reports or individual interactive visualizations, such as charts or tables, into existing web pages.

Web 2.0 Integration APIs and Services access and deliver individual components of Actuate technology that support report customization through a set of shared services to other web applications. These applications can be created using different technologies, such as JavaScript, Java, .NET, PHP or hosted applications.

The event handlers can access the Actuate Web 2.0 API, as shown in the following code:

```
**
* Occurs when mouse clicks.
 * @param event */

this.onclick = function(event)
{

// Access the Web 2.0 API to get the current viewer, set the
// parameter 'zipCode', and rerun the report.
    viewer =  actuate.getCurrentViewer();
    viewer.setParameterValue( "zipCode", inputZipCode );
// Rerun the report with the new parameter value
    viewer.submit();
}
```

This example sets a report parameter to be a certain zip code and reruns the report based on this new information. For more information about using the Actuate Web 2.0 API, see *Creating Custom Web Applications using Actuate iPortal*.

Creating an HTML button

Creating a functional HTML button element requires knowledge of event handlers and JavaScript. To create an HTML button that works with data from your report you must:

- Create an HTML button element.
- Add variables to the HTML button element, and bind report data to them.
- Add event handlers to the HTML button element.

The HTML button element is only supported in HTML output format. It is not rendered in other output formats, such as PDF or Excel.

How to create an HTML button element

1. Drag an HTML button element from the palette and drop it in the desired location on the report.
    - On HTML Button Builder, in Name, type the HTML name of the element. HTML button elements each must have a unique name.

? On HTML Button Builder, in Value, type the value of the element. The value is the text that appears on the button element. Enclose static text for the HTMLbutton value in quotation marks, as shown in Figure 5-4.

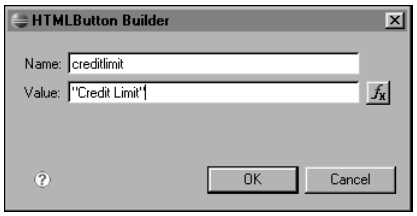

Figure 5-4 Use quoted strings for Value

Figure 5-5 shows a report with static text on the HTML button element.

Figure 5-5 Report output with static label on an HTML button

2 You can use Expression Builder to use data from the report to create the button value. This supports the ability to place computed information from the report onto the face of the button. When you finish creating the button value, choose OK. The button appears in the layout, as shown in Figure 5-6.

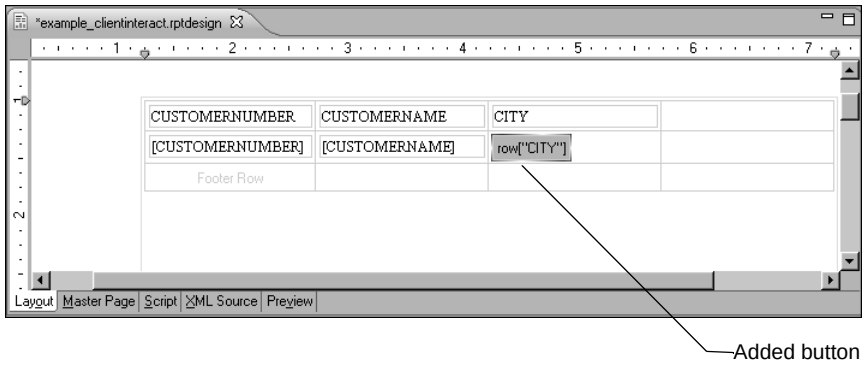

Figure 5-6 Using a computed expression for Value in an HTML button

When the report in this example runs, the button displays the name of the city on its face, as shown in Figure 5-7.

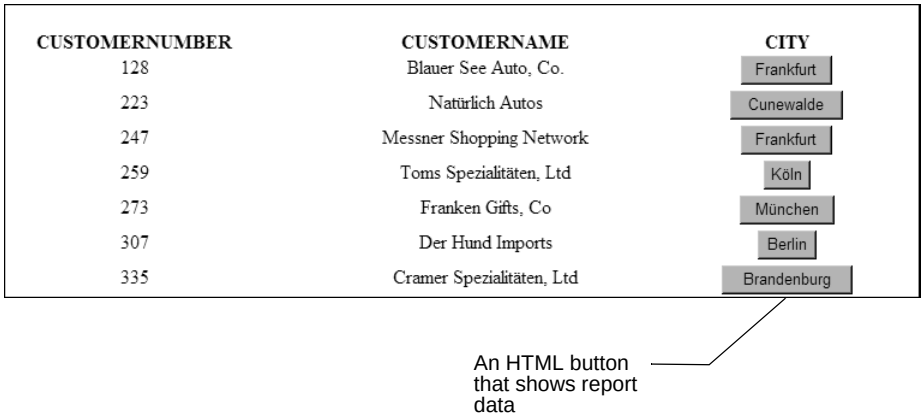

Figure 5-7 Report output with HTML buttons showing report data

How to add a variable to an HTML button

1. Choose Property Editor - HTML Button for the HTML button you wish to add a variable to.
2. Choose Variables. On Variables, choose Add.
3. On Add Variables, create the variables:
   1. In Name, type a unique name for the variable. JavaScript event handlers use name to access the variable information. It is accessed through dataObject. For example, a variable named credit is accessed in the event handler as dataObject.credit.
   2. In Expression, type the value that the variable contains. You can also use Expression Builder to create a value. Choose OK.

   When completed, Add Variables looks like the one in Figure 5-8.

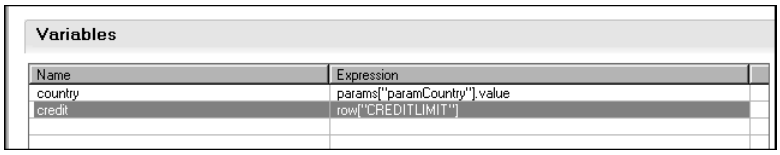

Figure 5-8 Report parameter and data row variables

How to add an event handler to an HTML button

1 On Layout, choose Script. For New Event Function, select an event to modify, as shown in Figure 5-9.

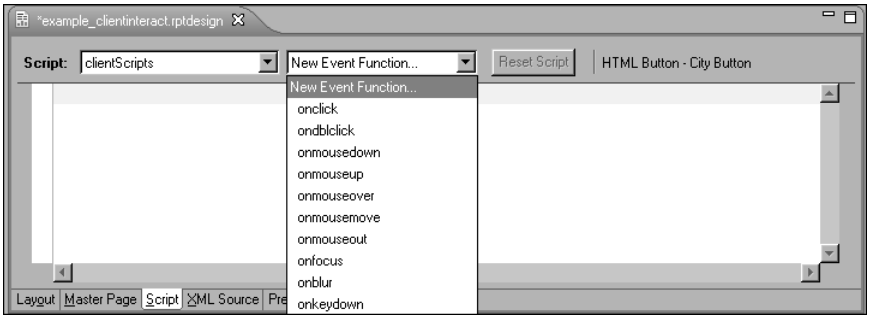

Figure 5-9 Script tab for HTML button

If the events do not appear in New Event Function, make sure the HTML button is active in the layout window, then return to the script and select an event in New Event Function.

2 Type the JavaScript code for the event handler in the event handler template, as shown in Figure 5-10.

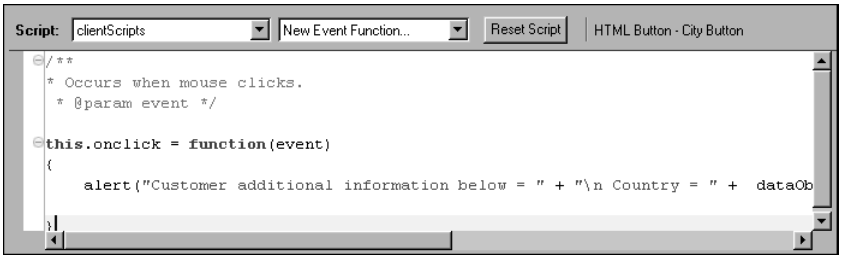

Figure 5-10 Event handler onclick with custom code

3 Run the report to test the function of the button and event handler. If you do not receive expected results, or if you receive an error, check your event handler script for possible problems.

Changing the appearance of an HTML button

You can modify an HTML button element by changing property values, such as its name, its value, or aspects of its appearance.

How to change the name or value of an HTML button

1 Double-click the HTML button.

2 In HTML Button Builder, in Name, type the new button name.

3 In Value, type the new value.

You can also accept the values of one or both as they currently exist. The HTML button Builder will contain the new values, as shown in Figure 5-11.

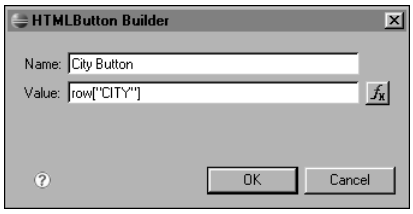

Figure 5-11 HTML Button Builder

Choose OK. The HTML button reflects the changes.

You can alter other features of the HTML button's appearance by modifying the HTML button element's properties. The tabs on the property sheet for the element support altering the appearance, visibility, and other features.

HTML Button—Properties supports the ability to change the appearance of the text and color of the button. It also supports adding an image to the face of the button. If you add an image, make sure the image file is the appropriate size for your button, because the entire image will appear in full size on the face of the button. If the button is the wrong size for your report, you must edit it with a graphic editing tool. Figure 5-12 shows the general settings.

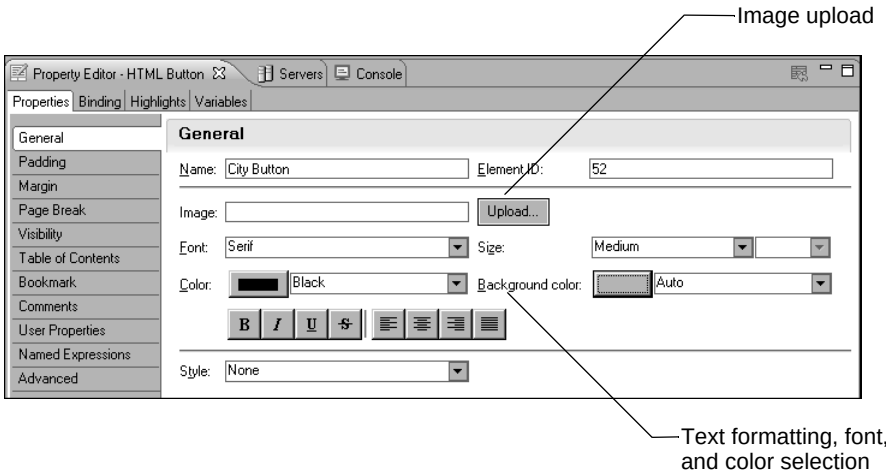

Figure 5-12 Properties—General tab

Padding, as shown in Figure 5-13, adds extra space around the text or image on the face of the HTML button element.

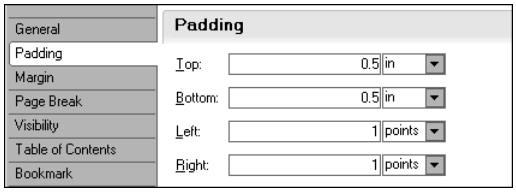

Figure 5-13 Padding tab

Padding supports the use of different units, such as inches or points. When you add padding, it affects the HTML button as shown in Figure 5-14.

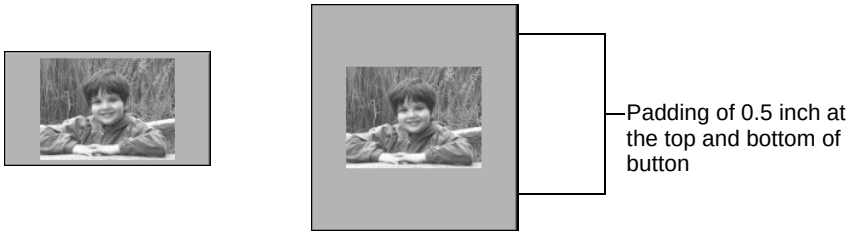

Figure 5-14 Padding added to an HTML button with image

Specifying margins is similar to specifying padding, as shown in Figure 5-15.

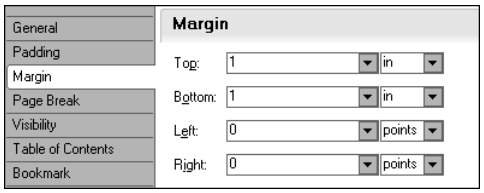

Figure 5-15 Margin tab

While padding modifies the size of the HTML button element, margins modify the space around the button and do not change the button size, as shown in Figure 5-16.

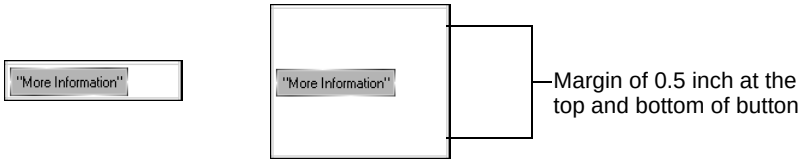

Figure 5-16 Margin space around an HTML button in a table

Visibility, Page Break, Table of Contents, and other properties operate in the same manner as they do for other report elements.