

To make a sticky form:

1. Open `calculator.php` (refer to Script 3.5) in your text editor or IDE.

2. Change the quantity input to read (**Script 3.6**)

```
<p>Quantity: <input type="text"
→ name="quantity" size="5"
→ maxlength="5" value="<?php if
→ (isset($_POST['quantity'])) echo
→ $_POST['quantity']; ?>" /></p>
```

The first change is to add the `value` attribute to the input. Then, print out the value of the submitted quantity variable (`$_POST['quantity']`). Since the first time the page is loaded, `$_POST['quantity']` has no value, a conditional ensures that the variable is set before attempting to print it. The end result for setting the input's value is the PHP code

```
<?php
if (isset($_POST['quantity'])) {
    echo $_POST['quantity'];
}
?>
```

This can be condensed to the more minimal form used in the script (you can omit the curly braces if you have only one statement within a conditional block, although I very rarely recommend that you do so).

3. Repeat the process for the price and tax.

```
<p>Price: <input type="text" name=
→ "price" size="5" maxlength="10"
→ value="<?php if (isset($_POST
→ ['price'])) echo $_POST['price'];
→ ?>" /></p>
```

Script 3.6 The calculator's form now recalls the previously entered values (creating a *sticky form*).

```
Script
1  <?php # Script 3.6 - calculator.php #2
2
3  $page_title = 'Widget Cost Calculator';
4  include ('includes/header.html');
5
6  // Check for form submission:
7  if (isset($_POST['submitted'])) {
8
9      // Minimal form validation:
10     if ( is_numeric($_POST['quantity']) &&
        is_numeric($_POST['price']) &&
        is_numeric($_POST['tax']) ) {
11
12         // Calculate the results:
13         $total = ($_POST['quantity'] *
        $_POST['price']);
14
15         $taxrate = ($_POST['tax'] / 100); //
        Turn 5% into .05.
16
17         $total += ($total * $taxrate); // Add
        the tax.
18
19         // Print the results:
20         echo '<h1>Total Cost</h1>
21
22         <p>The total cost of purchasing ' .
        $_POST['quantity'] . ' widget(s) at $' .
        number_format ($_POST['price'], 2) . '
        each, including a tax rate of ' .
        $_POST['tax'] . '%, is $' .
        number_format ($total, 2) . '</p>';
23
24     } else { // Invalid submitted values.
25
26         echo '<h1>Error!</h1>
27
28         <p class="error">Please enter a valid
        quantity, price, and tax.';
29
30     }
31 } // End of main isset() IF.
```

(script continues on next page)

Script 3.6 continued

```

28 // Leave the PHP section and create the
    HTML form:
29 ?>
30 <h1>Widget Cost Calculator</h1>
31 <form action="calculator.php"
    method="post">
32 <p>Quantity: <input type="text" name=
    "quantity" size="5" maxlength="5"
    value="<?php if (isset($_POST['quantity'])
    )) echo $_POST['quantity']; ?>" /></p>
33 <p>Price: <input type="text" name="price"
    size="5" maxlength="10" value="<?php if
    (isset($_POST['price'])) echo $_POST
    ['price']; ?>" /></p>
34 <p>Tax (%): <input type="text" name="tax"
    size="5" maxlength="5" value="<?php if
    (isset($_POST['tax'])) echo $_POST
    ['tax']; ?>" /></p>
35 <p><input type="submit" name="submit"
    value="Calculate!" /></p>
36 <input type="hidden" name="submitted"
    value="TRUE" />
37 </form>
38 <?php // Include the footer:
39 include ('includes/footer.html');
40 ?>

```

Total Cost

The total cost of purchasing 5 widget(s) at \$122.00 each, including a tax rate of 6%, is \$646.60.

Widget Cost Calculator

Quantity: 5

Price: 122.00

Tax (%): 6

Figure 3.11 The form now recalls the previously submitted values...

```

<p>Tax (%): <input type="text"
→ name="tax" size="5" maxlength="5"
→ value="<?php if (isset($_POST
→ ['tax'])) echo $_POST['tax']; ?>"
→ /></p>

```

4. Save the file as `calculator.php`, place it in your Web directory, and test it in your Web browser (Figures 3.11 and 3.12).

✓ Tips

- Because some PHP code in this example exists inside of the HTML form value attributes, error messages may not be obvious. If problems occur, check the HTML source of the page to see if PHP errors are printed within the value attributes.
- You should always double-quote HTML attributes, particularly the value attribute of a form input. If you don't, multi-word values like *Elliott Smith* will appear as just *Elliott* in the Web browser.
- On account of a limitation in how HTML works, you cannot preset the value of a password input type.

Error!

Please enter a valid quantity, price, and tax.

Widget Cost Calculator

Quantity:

Price: 34.50

Tax (%): 3.25

Figure 3.12 ...whether or not the form was completely filled out.