Chapter 4

How to test and debug a JavaScript application

Section 4-10

Testing and Debugging

Testing vs. debugging

The goal of testing

- To **find** all errors before the application is put into production.

The goal of debugging

- To **fix** all errors before the application is put into production.

Three common test phases

- Check the user interface to make sure that it works correctly.
- Test the application with valid input data to make sure the results are correct.
- Test the application with invalid data or unexpected user actions. Try everything you can think of to make the application fail.

Three Types of Errors that can Occur

- Syntax errors
  - Invalid JavaScript Statements
- Runtime errors
  - Happens when a program runs and hits an exception
  - Example: Divide by zero
- Logic errors
  - There are no Syntax or Runtime errors but you don’t get the results you want

Common syntax errors

- Misspelling keywords like:
  getElementByID instead of getElementById.
- Omitting required parentheses, quotation marks, or braces.
- Not using the same opening and closing quotation mark.
- Omitting the semicolon at the end of a statement.
- Misspelling or incorrectly capitalizing an identifier like:
  defining a variable named salesTax
  and referring to it as salestax.

Problems with HTML references

- Referring to an attribute value or other HTML component incorrectly, like referring to an id as **salesTax** when the id is **sales_tax**.

Problems with data and comparisons

- Not testing to make sure that a user entry is the right data type before processing it.

- Not using the **parseInt** or **parseFloat** method to convert a user entry into a numeric value before processing it.

- Using one equal sign (=)instead of two (==)when testing for equality.

Problems with undeclared variables

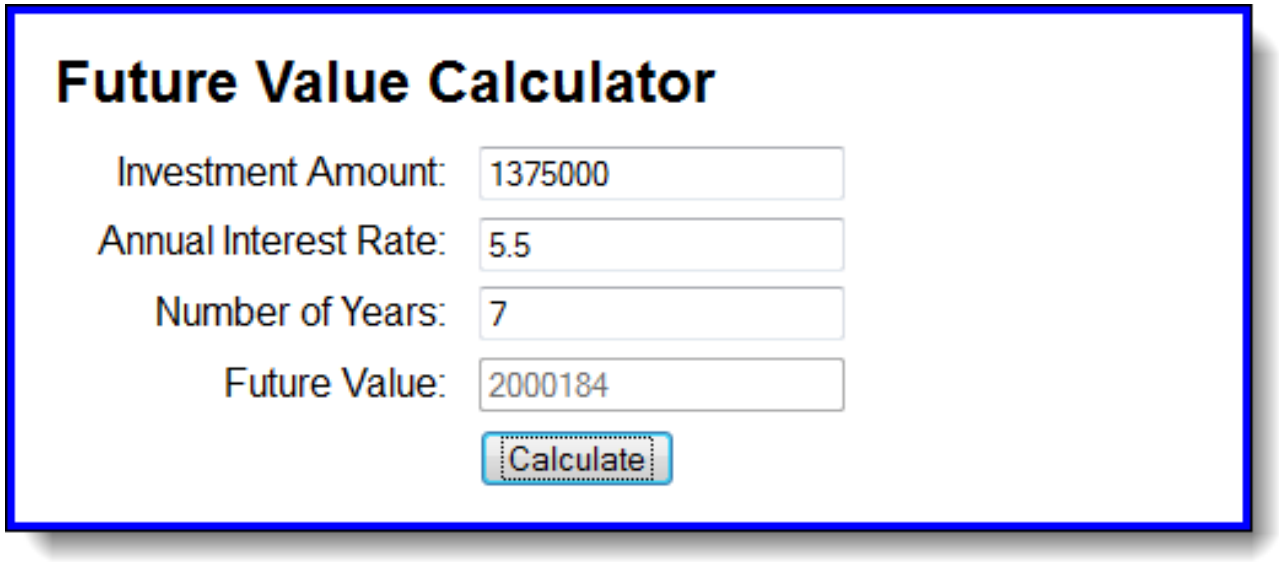
- If you assign a value to a variable that hasn't been declared, the JavaScript engine treats it as a global variable.
- This can happen when you misspell a variable name, as in this example:

```
var calculateTax = function (subtotal, taxRate) {
    var salesTax = subtotal * taxRate;
    // salesTax is local

    salestax = parseFloat(salesTax.toFixed(2));
    // salestax is global

    return salesTax;
    // salesTax isn't rounded but salestax is
}
```

Top-Down Testing Example



Top-down coding and testing

Phase 1: No data validation

```
var $ = function (id) {
    return document.getElementById(id);
}
var calculateClick = function () {
    var investment = parseFloat( $("investment").value );
    var annualRate = parseFloat( $("rate").value );
    var years = parseInt( $("years").value );
    for ( i = 1; i <= years; i++ ) {
        investment += investment * annualRate / 100;
    }
    $("future_value").value = investment.toFixed();
}
window.onload = function () {
    $("calculate").onclick = calculateClick;
}
```

Top-down coding and testing (continued)

Phase 2: Add data validation for just the first entry

```
if (isNaN(investment) || investment <= 0) {
    alert("Investment must numeric and greater than 0.");
}
else {
    // the future value calculation from phase 1
}
```

Phase 3: Add data validation for the other entries

Phase 4: Add the finishing touches

```
// like moving the focus to the first text box
```

Exercise 4-1

- Do the exercises for this section

  (shown to the right for the link to this presentation)

Section 4-20

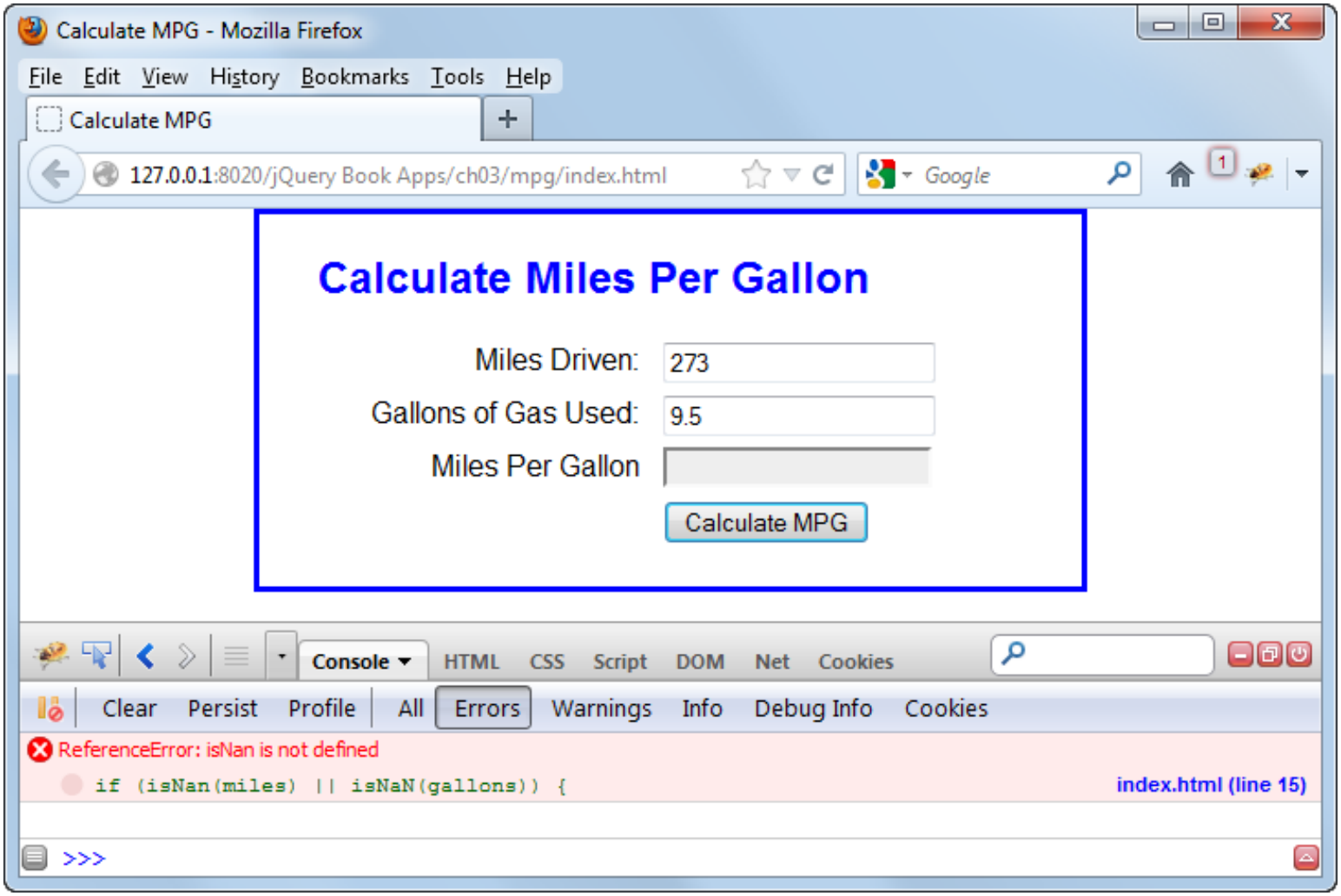
Firebug

Getting Firebug

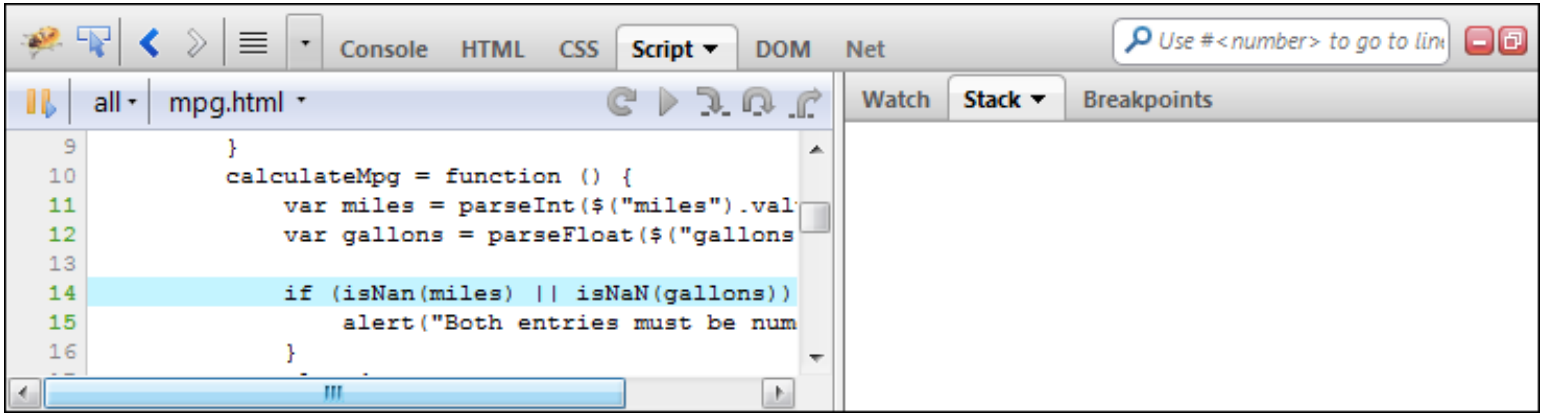
How to open and close Firebug

- Go to Firefox:Add-Ons
- Enter “Firebug” in the search box
- Click Install and follow directions

Firebug with an open Console panel



The Script panel after the link in the Console panel has been clicked



How to use Firebug

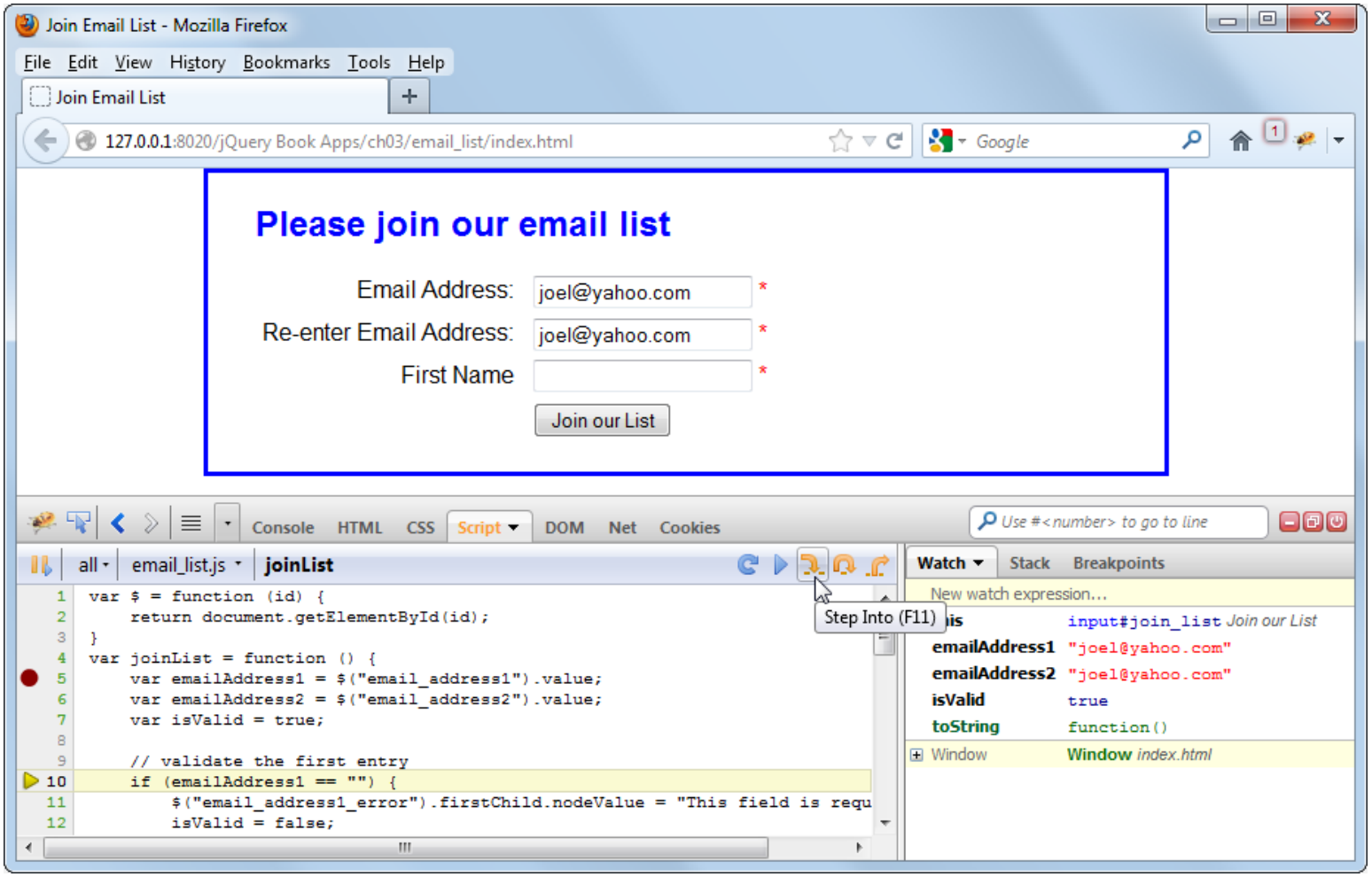
How to open and close Firebug

- Click the Firebug icon or press F12.

How to find the statement that caused an error

- Open the Console panel by clicking on the Console tab.
- Click on the line of code in the Console panel or click on the line marker to the right of the line of code.
- That will open the Script panel with the statement that that caused the error highlighted.

A breakpoint in the Firebug Script tab



How to use breakpoints and step through code

How to set or remove a breakpoint in the Script panel

- To set a breakpoint, click in the bar to the left of a statement.
- To remove a breakpoint, click on the red breakpoint marker.

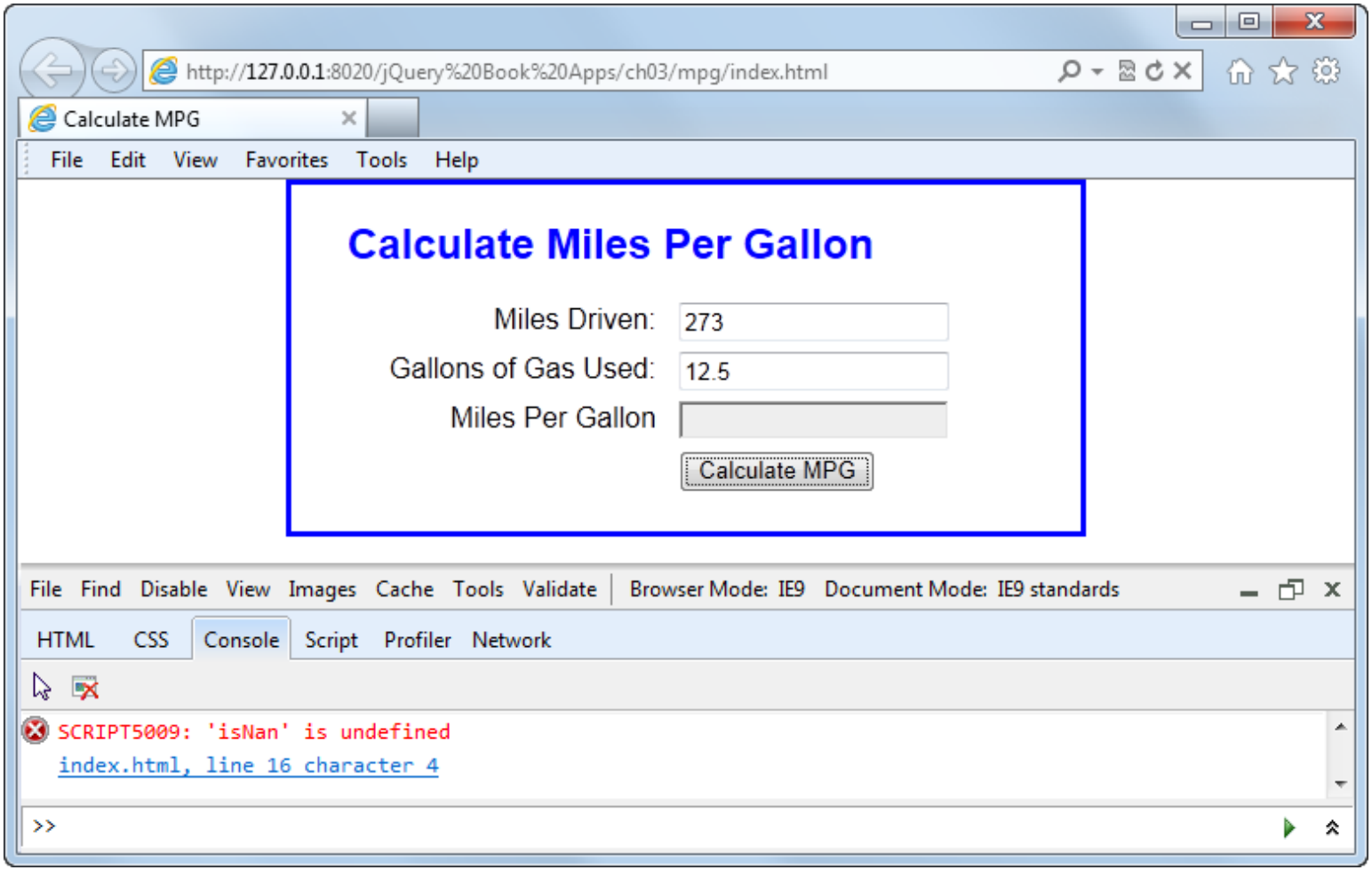
How to step through the code in the Script panel

- Click Step Into or press F11 to step through one line at a time.
- Click Step Over to run any called functions without stepping through them.
- Click Step Out to execute the rest of a function without stepping through it.
- Click Continue to resume normal execution.

How to view the current data values at each step

- Hover the mouse cursor over a variable name in the Script panel.
- View the current variables in the Watch pane to the right of the Script panel.
- Click “New watch expression...” in the Watch pane and type the variable name or expression that you want to watch.

IE with a Console panel that shows an error



Exercise 4-2

- Do the exercises for this section

  (shown to the right for the link to this presentation)

Section 4-30

Other Ways to Find Errors

How to find errors with IE

How to display the Developer Tools with the current IE

- Use the Tools→Developer Tools command, or press F12.

How to display an error message with older versions of IE

- Double-click on the error icon in the lower left corner of the browser to view the IE message box.
- If necessary, click the Show Details button to view the entire message.

JavaScript with five alert statements that trace the execution of the code

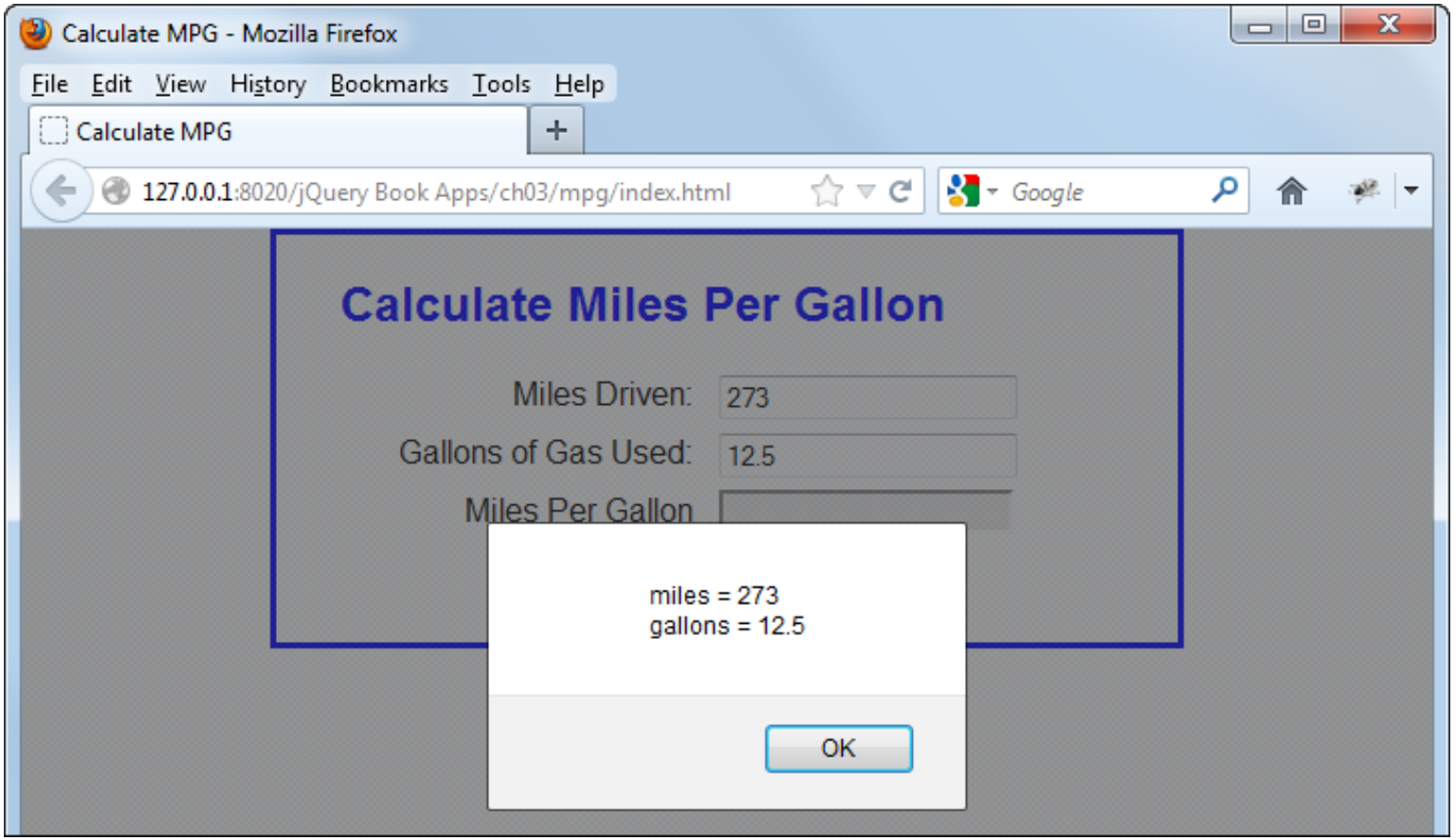
```
var $ = function (id) {
    alert("$ function has started");
    return document.getElementById(id);
}
calculateMpg = function () {
    alert("calculateMpg function has started");
    var miles = parseInt($("miles").value);
    var gallons = parseFloat($("gallons").value);
    alert("miles = " + miles +
          "\ngallons = " +  gallons);

    if (isNaN(miles) || isNaN(gallons)) {
        alert("Both entries must be numeric");
    }
    else {
        alert("The data is valid and the calc is next");
        var mpg = miles / gallons;
        $("mpg").value = mpg.toFixed(1);
    }
}
```

JavaScript with five alert statements (continued)

```
window.onload = function () {
    alert("onload function has started");
    $("calculate").onclick = calculateMpg;
    $("miles").focus();
}
```

The trace boxes are displayed as the JavaScript is executed



How to view the source code for a web page

- If it's available, use a menu command like View→Source or View→Page Source.
- You can also right-click on the page and select a command like Source, View Source, or View Page Source.

Exercise 4-3

- Do the exercises for this section

  (shown to the right for the link to this presentation)

End of Chapter 4