Document Object Model (DOM) Interactive Laboratory

Introduction

The Document Object Model (DOM) is a standard for accessing HTML and XML elements. This laboratory will focus on HTML DOM. As you recall from the previously reviewed DOM enables one to access elements on a webpage by defining properties and methods for viewing and manipulating parts of the page.

One might think of a webpage as a tree structure, where various elements have sub-elements. Each part of the webpage can be thought of as a node, starting with the page itself (referred to as document). A good example of a tree structure, or node tree, can be viewed at the W3Schools website at the following URL: http://www.w3schools.com/htmldom/dom_nodes.asp

In this laboratory, we will learn how to examine the DOM of a simple web form and how to make changes on the fly to element properties. Once you learn JavaScript, you will then be able to programmatically change portions of a webpage by using DOM elements.

DOM Inspector

Depending on the browser used, there will be different methods to viewing the DOM. For this laboratory, we will use the DOM Inspector which is an add-on to Firefox. If you do not have Firefox installed on your computer, you can download and install a copy by going to the following URL: http://www.mozilla.org/en-US/

When you first install Firefox, you will not have the DOM Inspector installed. As you can see in Figure 1, there is a Web Developer pull-down menu item.

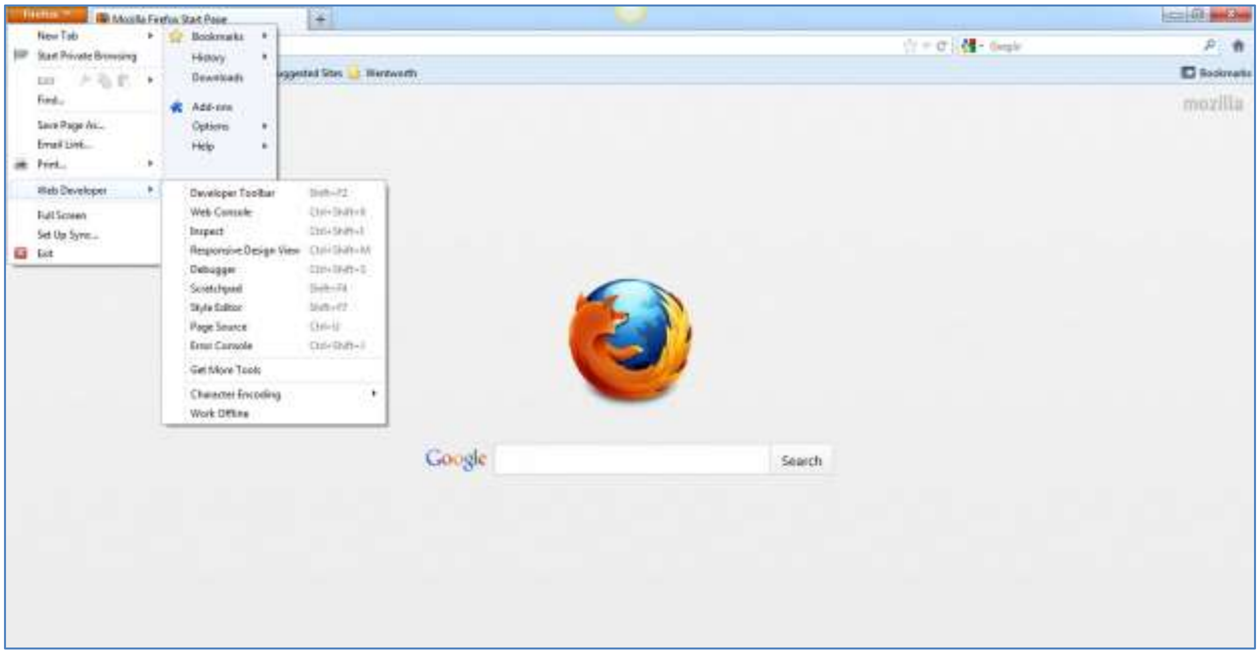


Figure 1

You can install DOM Inspector as an add-on by going to the following URL:
https://addons.mozilla.org/en-us/firefox/addon/dom-inspector-6622/

Once this has been installed, you will then see DOM Inspector as one of the choices in the pull-down menu, as shown in Figure 2.

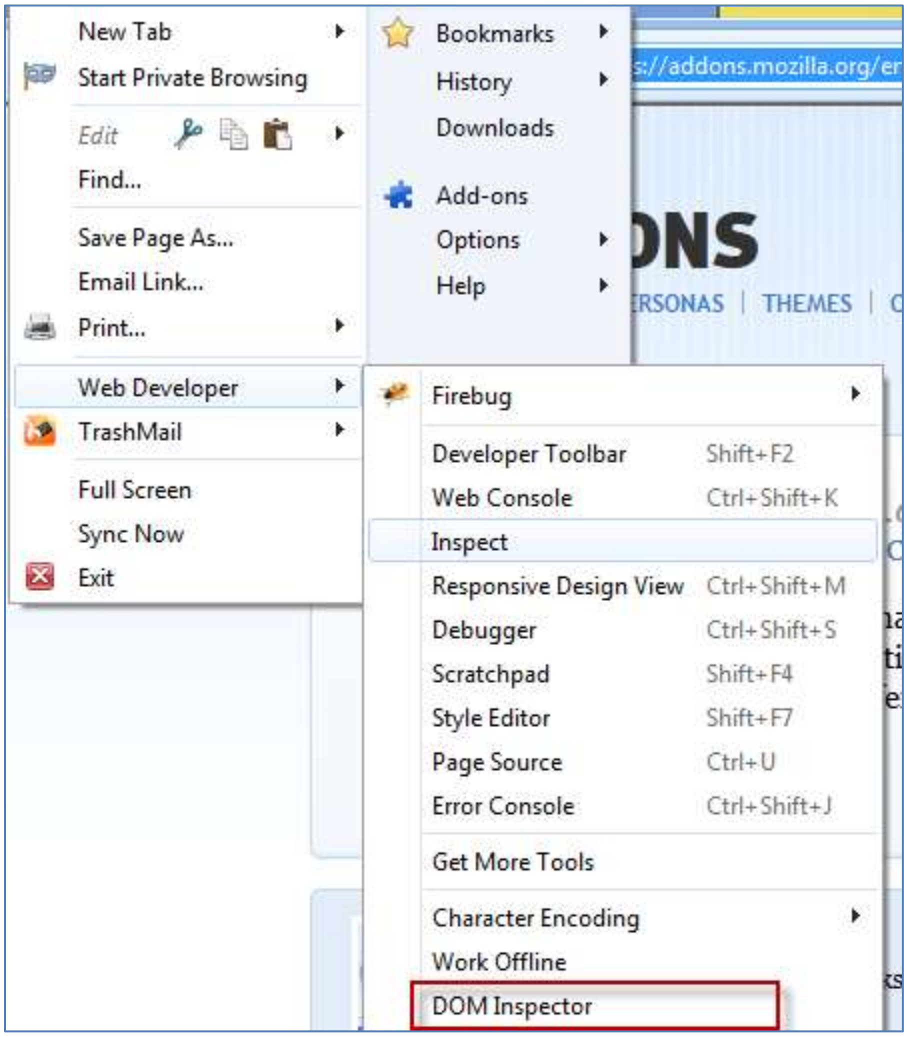
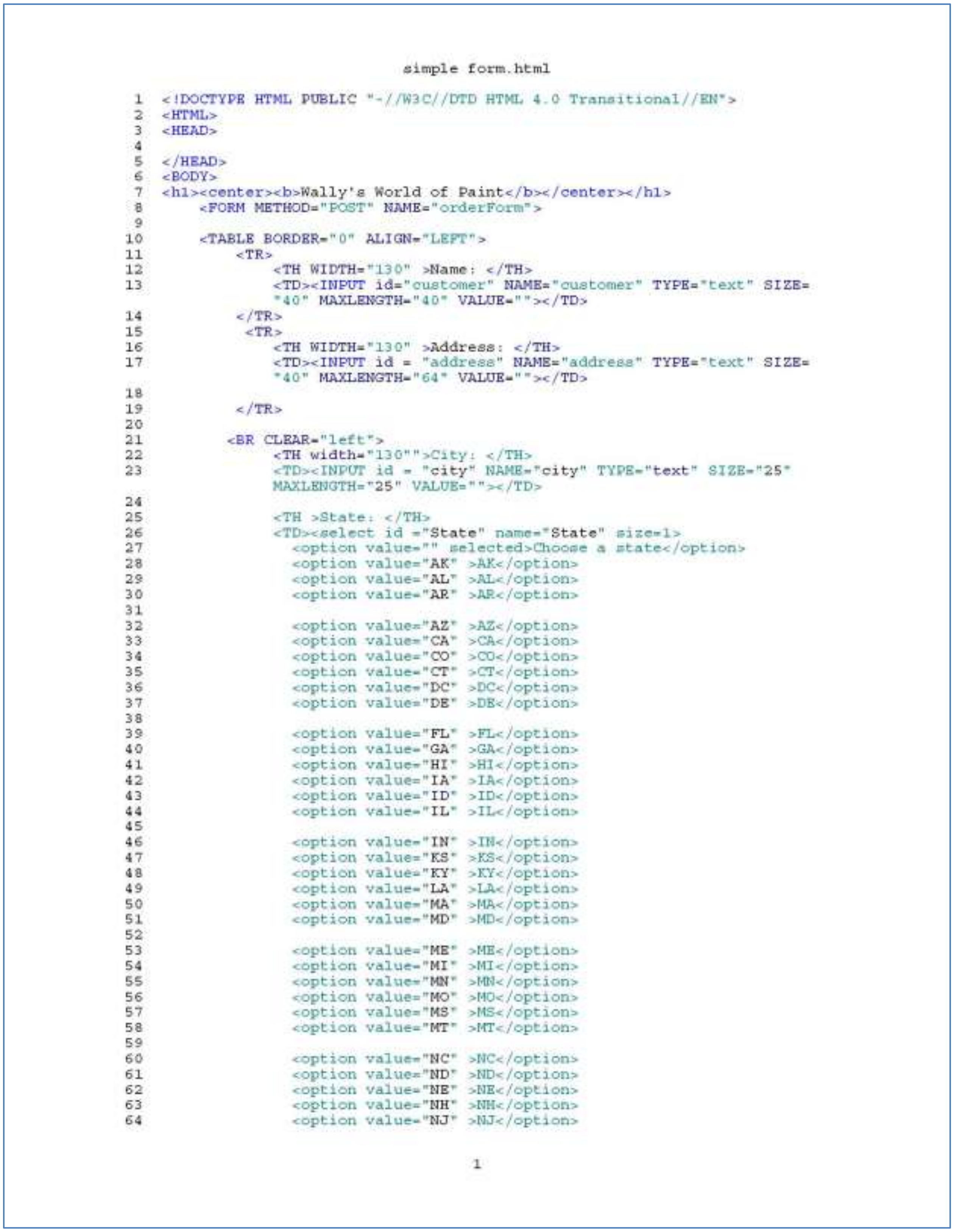


Figure 2

Using DOM Inspector

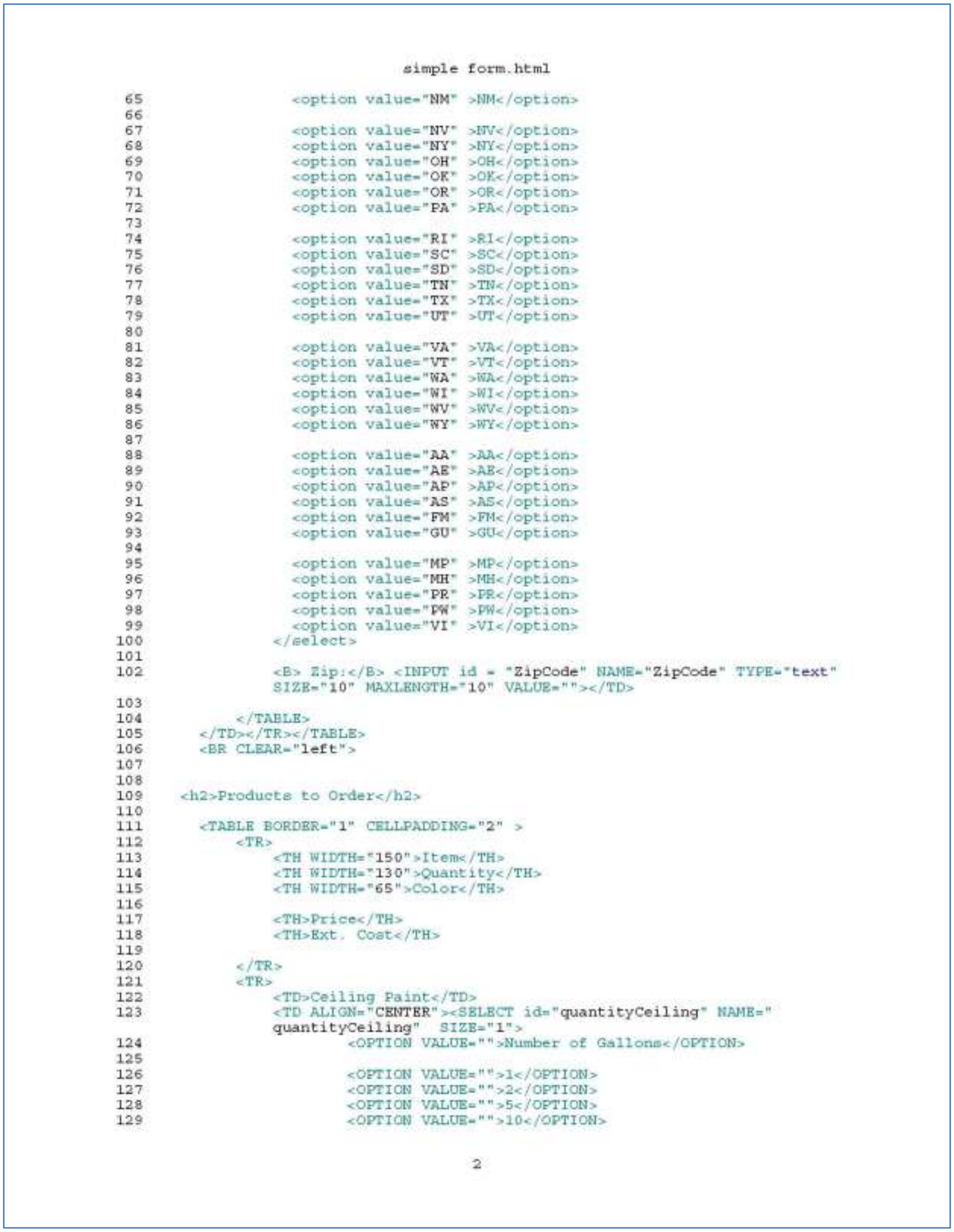
For this portion of the laboratory, we will use the following HTML file. The source code can be downloaded here. Figure 3 shows the first two pages of source code.



simple form.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>

</HEAD>
<BODY>
<h1><center><b>Wally's World of Paint</b></center></h1>
    <FORM METHOD="POST" NAME="orderForm">

    <TABLE BORDER="0" ALIGN="LEFT">
        <TR>
            <TH WIDTH="130" >Name: </TH>
            <TD><INPUT id="customer" NAME="customer" TYPE="text" SIZE=
            "40" MAXLENGTH="40" VALUE=""></TD>
        </TR>
         <TR>
            <TH WIDTH="130" >Address: </TH>
            <TD><INPUT id = "address" NAME="address" TYPE="text" SIZE=
            "40" MAXLENGTH="64" VALUE=""></TD>

        </TR>

       <BR CLEAR="left">
            <TH width="130"">City: </TH>
            <TD><INPUT id = "city" NAME="city" TYPE="text" SIZE="25"
            MAXLENGTH="25" VALUE=""></TD>

            <TH >State: </TH>
            <TD><select id ="State" name="State" size=1>
              <option value="" selected>Choose a state</option>
              <option value="AK" >AK</option>
              <option value="AL" >AL</option>
              <option value="AR" >AR</option>

              <option value="AZ" >AZ</option>
              <option value="CA" >CA</option>
              <option value="CO" >CO</option>
              <option value="CT" >CT</option>
              <option value="DC" >DC</option>
              <option value="DE" >DE</option>

              <option value="FL" >FL</option>
              <option value="GA" >GA</option>
              <option value="HI" >HI</option>
              <option value="IA" >IA</option>
              <option value="ID" >ID</option>
              <option value="IL" >IL</option>

              <option value="IN" >IN</option>
              <option value="KS" >KS</option>
              <option value="KY" >KY</option>
              <option value="LA" >LA</option>
              <option value="MA" >MA</option>
              <option value="MD" >MD</option>

              <option value="ME" >ME</option>
              <option value="MI" >MI</option>
              <option value="MN" >MN</option>
              <option value="MO" >MO</option>
              <option value="MS" >MS</option>
              <option value="MT" >MT</option>

              <option value="NC" >NC</option>
              <option value="ND" >ND</option>
              <option value="NE" >NE</option>
              <option value="NH" >NH</option>
              <option value="NJ" >NJ</option>
```

1



simple form.html

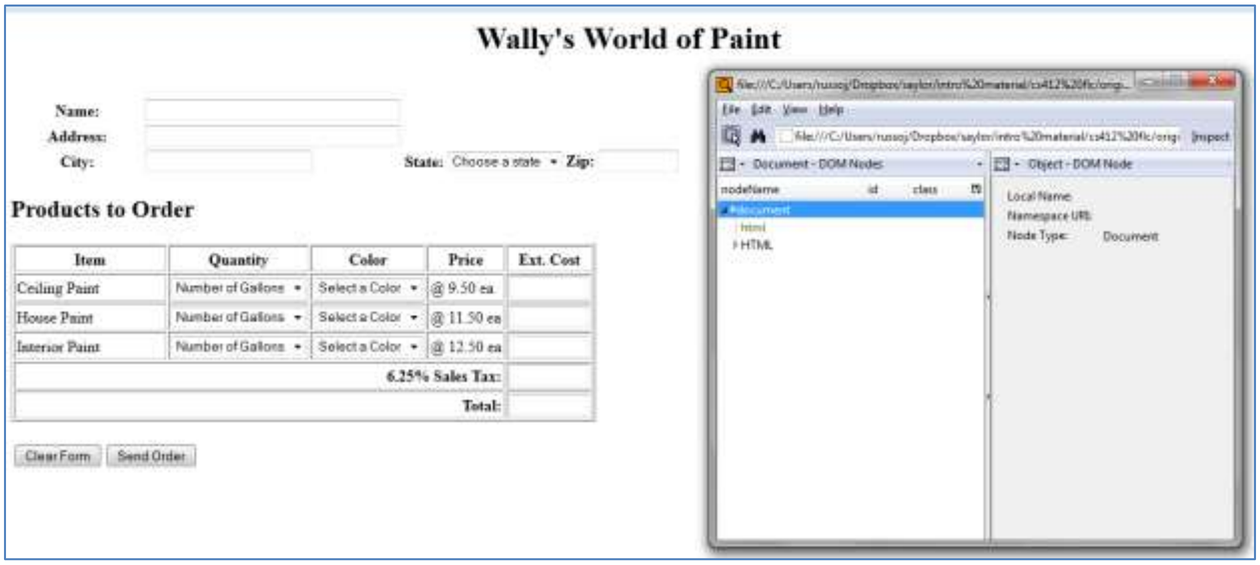
```
                <option value="NM" >NM</option>

                <option value="NV" >NV</option>
                <option value="NY" >NY</option>
                <option value="OH" >OH</option>
                <option value="OK" >OK</option>
                <option value="OR" >OR</option>
                <option value="PA" >PA</option>

                <option value="RI" >RI</option>
                <option value="SC" >SC</option>
                <option value="SD" >SD</option>
                <option value="TN" >TN</option>
                <option value="TX" >TX</option>
                <option value="UT" >UT</option>

                <option value="VA" >VA</option>
                <option value="VT" >VT</option>
                <option value="WA" >WA</option>
                <option value="WI" >WI</option>
                <option value="WV" >WV</option>
                <option value="WY" >WY</option>

                <option value="AA" >AA</option>
                <option value="AE" >AE</option>
                <option value="AP" >AP</option>
                <option value="AS" >AS</option>
                <option value="FM" >FM</option>
                <option value="GU" >GU</option>

                <option value="MP" >MP</option>
                <option value="MH" >MH</option>
                <option value="PR" >PR</option>
                <option value="PW" >PW</option>
                <option value="VI" >VI</option>
            </select>

            <B> Zip:</B> <INPUT id = "ZipCode" NAME="ZipCode" TYPE="text"
            SIZE="10" MAXLENGTH="10" VALUE=""></TD>

        </TABLE>
    </TD></TR></TABLE>
    <BR CLEAR="left">


  <h2>Products to Order</h2>

    <TABLE BORDER="1" CELLPADDING="2" >
        <TR>
            <TH WIDTH="150">Item</TH>
            <TH WIDTH="130">Quantity</TH>
            <TH WIDTH="65">Color</TH>

            <TH>Price</TH>
            <TH>Ext. Cost</TH>

        </TR>
        <TR>
            <TD>Ceiling Paint</TD>
            <TD ALIGN="CENTER"><SELECT id="quantityCeiling" NAME="
            quantityCeiling"  SIZE="1">
                    <OPTION VALUE="">Number of Gallons</OPTION>

                    <OPTION VALUE="">1</OPTION>
                    <OPTION VALUE="">2</OPTION>
                    <OPTION VALUE="">5</OPTION>
                    <OPTION VALUE="">10</OPTION>
```

2

Figure 3

Notice that we now have an ID as well as a name for elements. This is very important, since you will need the ID to access the element within the DOM Inspector. Later on, when you learn JavaScript, you will see that you can programmatically access the elements either by name or ID.

Let’s use Firefox to render the form. Also, select the DOM Inspector from the menu. Your screen should look similar to Figure 4 (I’ve rearranged the position of the DOM Inspector window in the screen capture).



Wally's World of Paint

Name:
Address:
City: State: Choose a state Zip:

Products to Order

Item	Quantity	Color	Price	Ext. Cost
Ceiling Paint	Number of Gallons	Select a Color	@ 9.50 ea	
House Paint	Number of Gallons	Select a Color	@ 11.50 ea	
Interior Paint	Number of Gallons	Select a Color	@ 12.50 ea	
			6.25% Sales Tax:	
			Total:	

Clear Form Send Order

file:///C:/Users/russoj/Dropbox/saylor/intro%20material/cs412%20fc/orig...

File Edit View Help

file:///C:/Users/russoj/Dropbox/saylor/intro%20material/cs412%20fc/orig Inspect

Document - DOM Nodes

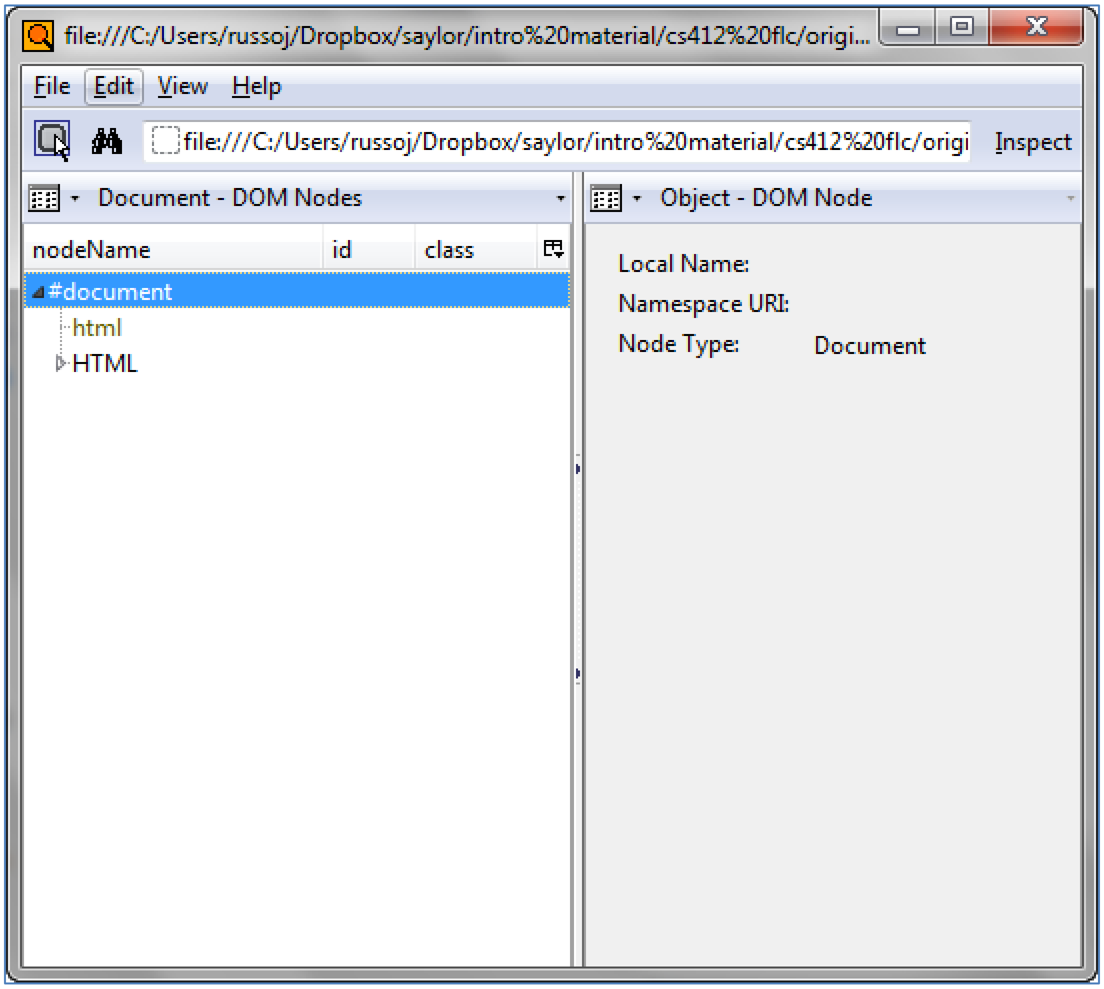
nodeName	id	class
#document		
html		
HTML		

Object - DOM Node

Local Name:
Namespace URI:
Node Type: Document

Figure 4

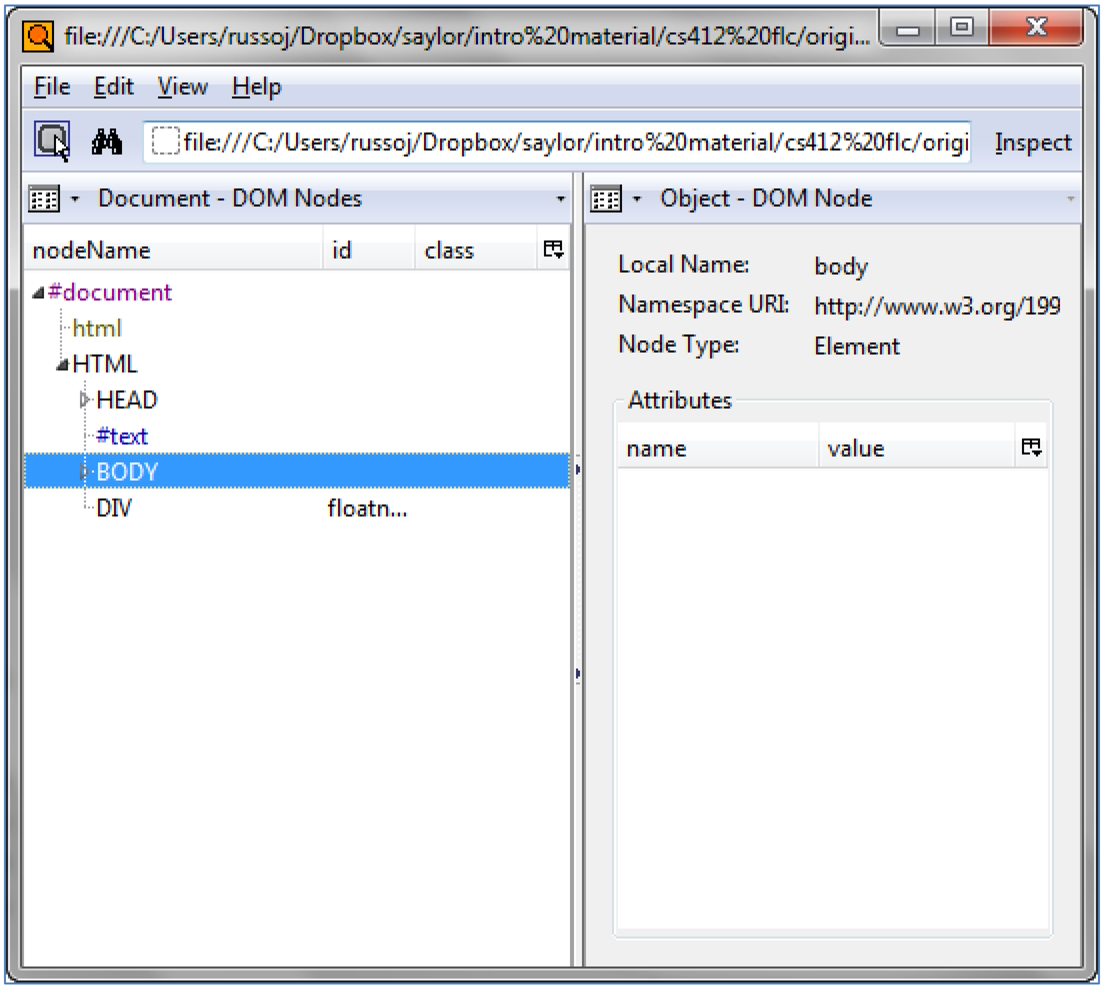
Let’s take a closer look at the DOM Inspector window, as shown in Figure 5.


file:///C:/Users/russoj/Dropbox/saylor/intro%20material/cs412%20flc/origi...
File Edit View Help
file:///C:/Users/russoj/Dropbox/saylor/intro%20material/cs412%20flc/origi
Inspect
Document - DOM Nodes
Object - DOM Node
nodeName
id
class
#document
html
HTML

Local Name:
Namespace URI:
Node Type: Document

Figure 5

Notice how the left-hand side of the window has a list of names. As we discussed above, the DOM can be thought of as a tree with nodes. The top level node is the document itself (the webpage) and then there are various levels beneath. We can expand nodes as needed. Let’s expand the HTML node by clicking on the small arrow next to the HTML node. Figure 6 shows the node expanded.


file:///C:/Users/russoj/Dropbox/saylor/intro%20material/cs412%20flc/origi...
File Edit View Help
file:///C:/Users/russoj/Dropbox/saylor/intro%20material/cs412%20flc/origi
Inspect
Document - DOM Nodes
Object - DOM Node

nodeName	id	class
#document		
html		
HTML		
HEAD		
#text		
BODY		
DIV	floatn...	

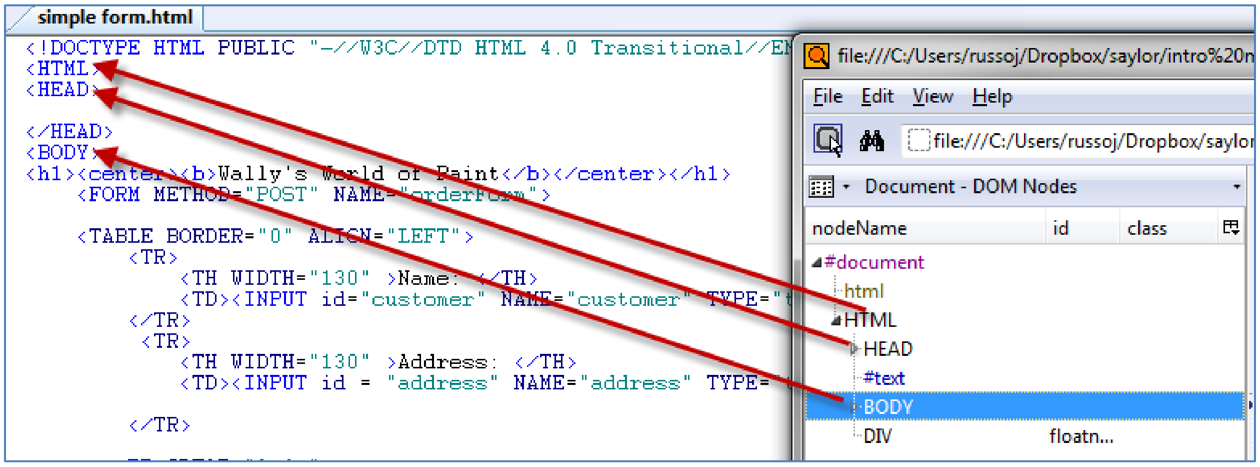
Local Name:	body
Namespace URI:	http://www.w3.org/199
Node Type:	Element

Attributes

name	value

Figure 6

Underneath the HTML node, there is the basic structure of the HTML file itself. There is a HEAD node, which corresponds to the head portion of the HTML file, as well as a BODY node which corresponds to the BODY of the HTML file. Figure 7 shows the direct correspondence between the nodes and the source code.



simple form.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN
<HTML>
<HEAD>

</HEAD>
<BODY>
<h1><center><b>Wally's World of Paint</b></center></h1>
    <FORM METHOD="POST" NAME="orderForm">

    <TABLE BORDER="0" ALIGN="LEFT">
        <TR>
            <TH WIDTH="130" >Name: </TH>
            <TD><INPUT id="customer" NAME="customer" TYPE="t
        </TR>
         <TR>
            <TH WIDTH="130" >Address: </TH>
            <TD><INPUT id = "address" NAME="address" TYPE=

        </TR>
```

file:///C:/Users/russoj/Dropbox/saylor/intro%20n

File Edit View Help

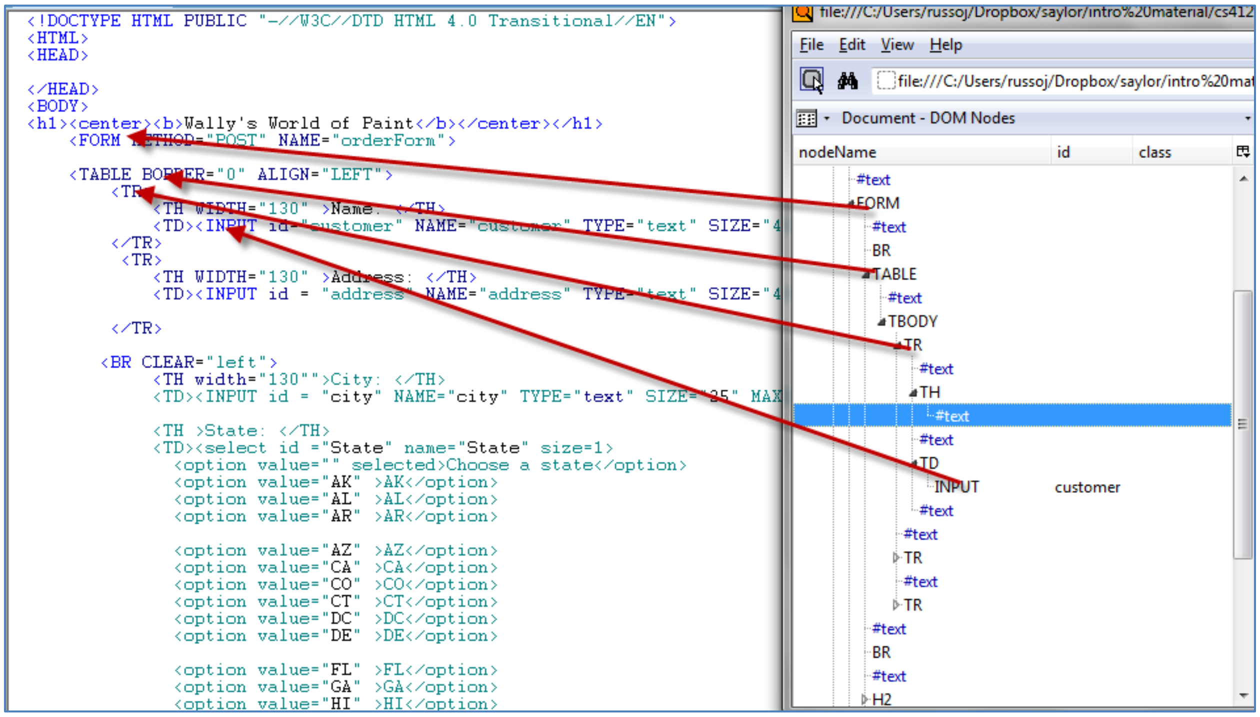
file:///C:/Users/russoj/Dropbox/saylor

Document - DOM Nodes

```
nodeName                    id          class
#document
  html
  HTML
    HEAD
    #text
    BODY
      DIV                   floatn...
```

Figure 7

We can now expand the DOM inspector further to view other parts of the webpage. For example, perhaps we want to look at the form itself. Figure 8 shows the DOM inspector with the form node expanded to show the form, table and input areas for customer name.



```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>

</HEAD>
<BODY>
<h1><center><b>Wally's World of Paint</b></center></h1>
    <FORM METHOD="POST" NAME="orderForm">

    <TABLE BORDER="0" ALIGN="LEFT">
        <TR>
            <TH WIDTH="130" >Name: </TH>
            <TD><INPUT id="customer" NAME="customer" TYPE="text" SIZE="4
        </TR>
         <TR>
            <TH WIDTH="130" >Address: </TH>
            <TD><INPUT id = "address" NAME="address" TYPE="text" SIZE="4

        </TR>

       <BR CLEAR="left">
            <TH width="130"">City: </TH>
            <TD><INPUT id = "city" NAME="city" TYPE="text" SIZE="25" MAX

            <TH >State: </TH>
            <TD><select id ="State" name="State" size=1>
              <option value="" selected>Choose a state</option>
              <option value="AK" >AK</option>
              <option value="AL" >AL</option>
              <option value="AR" >AR</option>

              <option value="AZ" >AZ</option>
              <option value="CA" >CA</option>
              <option value="CO" >CO</option>
              <option value="CT" >CT</option>
              <option value="DC" >DC</option>
              <option value="DE" >DE</option>

              <option value="FL" >FL</option>
              <option value="GA" >GA</option>
              <option value="HI" >HI</option>
```

file:///C:/Users/russoj/Dropbox/saylor/intro%20material/cs412

File Edit View Help

file:///C:/Users/russoj/Dropbox/saylor/intro%20mat

Document - DOM Nodes

```
nodeName                    id          class
    #text
    FORM
      #text
      BR
      TABLE
        #text
        TBODY
          TR
            #text
            TH
              #text
            #text
            TD
              INPUT         customer
            #text
          #text
          TR
          #text
          TR
      #text
      BR
      #text
      H2
```

Figure 8

Finding Nodes

What we have seen so far has provided a nice way to view the Document Object Model for a particular HTML document. However, it can be cumbersome to have to search around and expand nodes in order to find the part of the page we wish to inspect or possibly change. Alternately, we can find a specific node by **ID**. In order to do this, we need to press the **search** button, select **ID,** and complete the field. Figure 9 shows a search for the customer ID.

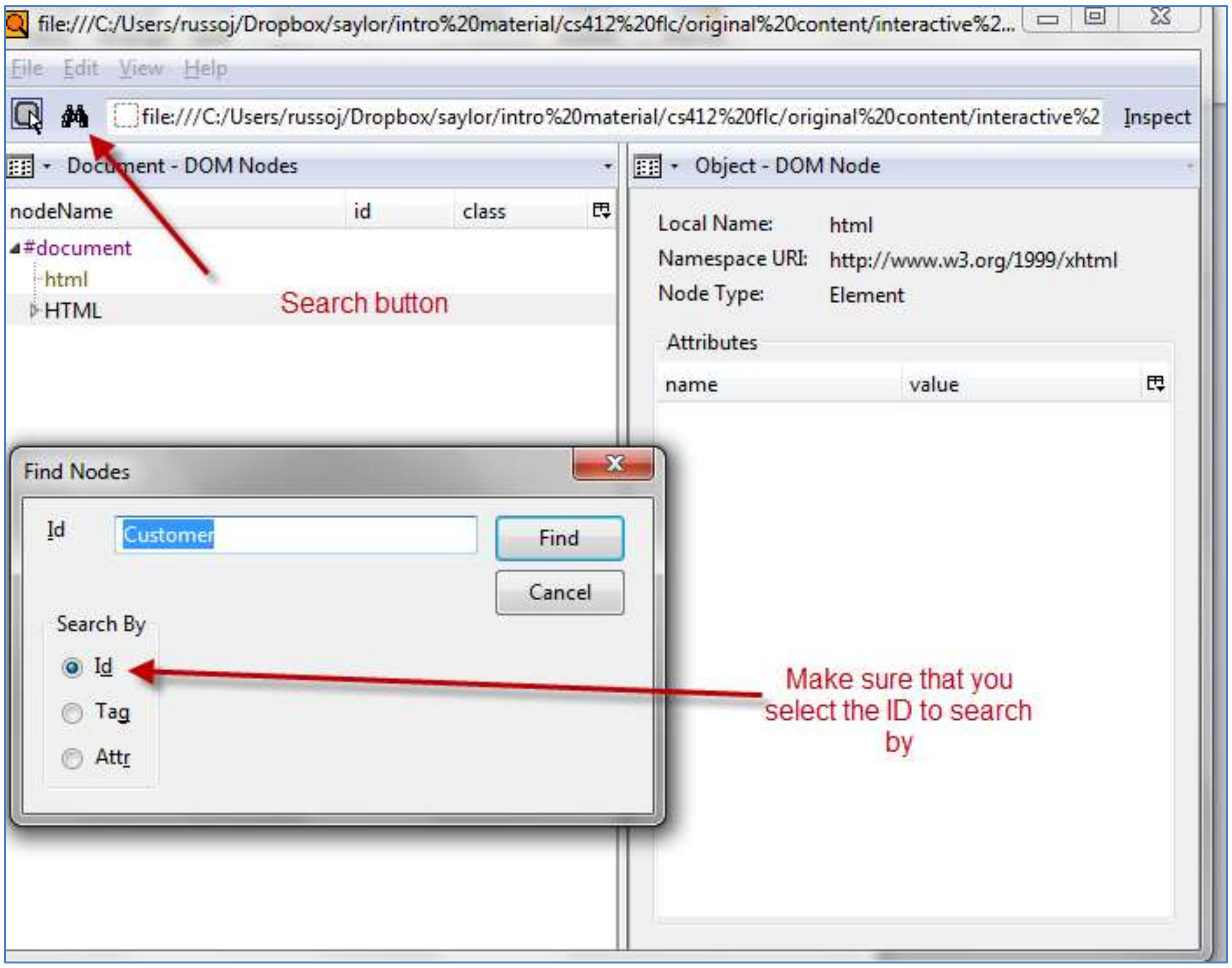
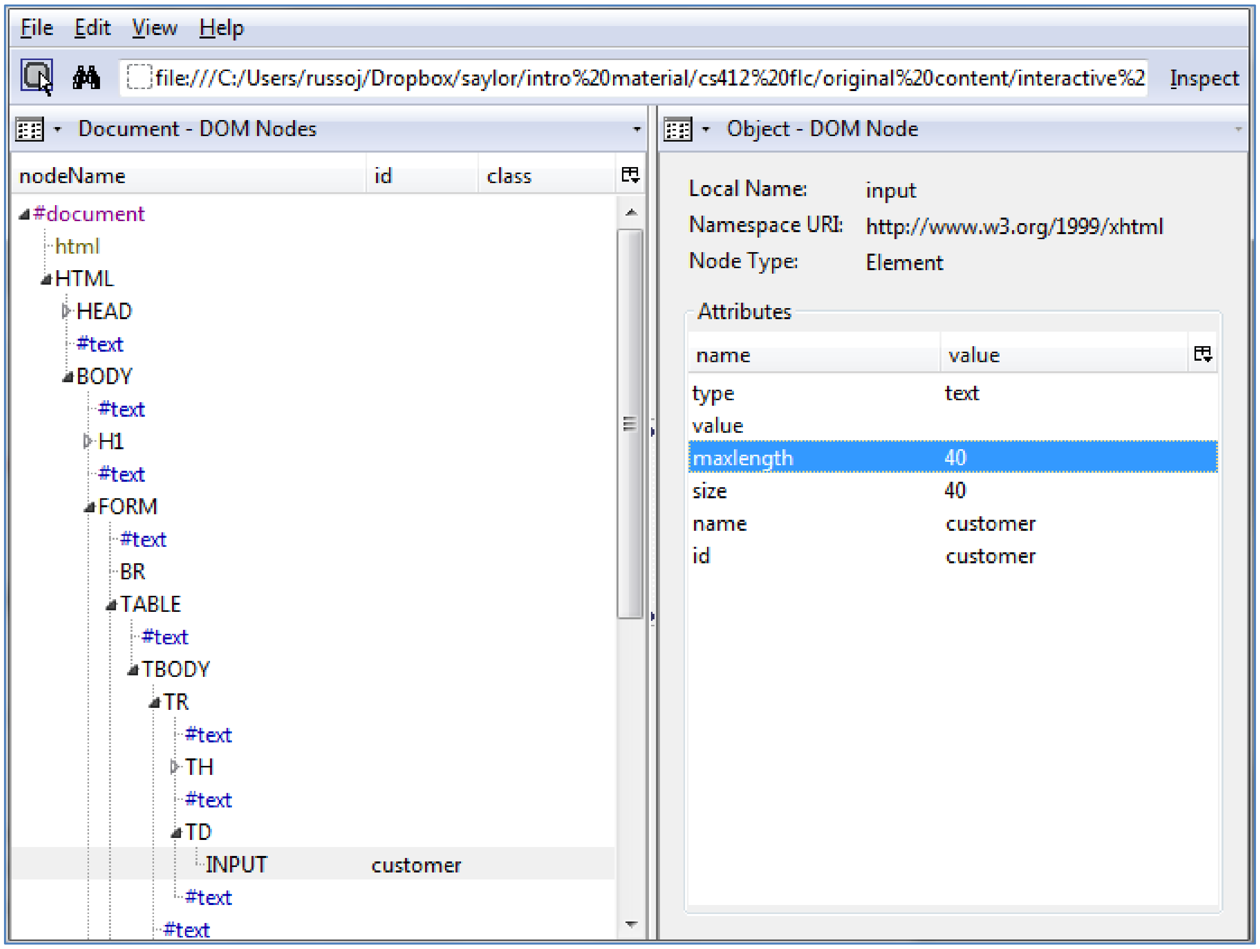

file:///C:/Users/russoj/Dropbox/saylor/intro%20material/cs412%20flc/original%20content/interactive%2...
File Edit View Help
file:///C:/Users/russoj/Dropbox/saylor/intro%20material/cs412%20flc/original%20content/interactive%2
Inspect
Document - DOM Nodes
Object - DOM Node
nodeName
id
class
#document
html
HTML
Search button
Local Name: html
Namespace URI: http://www.w3.org/1999/xhtml
Node Type: Element
Attributes
name
value
Find Nodes
Id
Customer
Find
Cancel
Search By
Id
Tag
Attr
Make sure that you select the ID to search by

Figure 9

Once you have done this, the DOM inspector will then bring you to the node with the Customer ID. As you can see from Figure 10, the left hand window shows where the node is in the DOM tree, and the right hand window shows the properties of the node. Shortly, we will look at how the properties can be modified directly in the DOM inspector.


File Edit View Help
file:///C:/Users/russoj/Dropbox/saylor/intro%20material/cs412%20flc/original%20content/interactive%2
Inspect
Document - DOM Nodes
Object - DOM Node

nodeName	id	class
#document		
html		
HTML		
HEAD		
#text		
BODY		
#text		
H1		
#text		
FORM		
#text		
BR		
TABLE		
#text		
TBODY		
TR		
#text		
TH		
#text		
TD		
INPUT	customer	
#text		
#text		

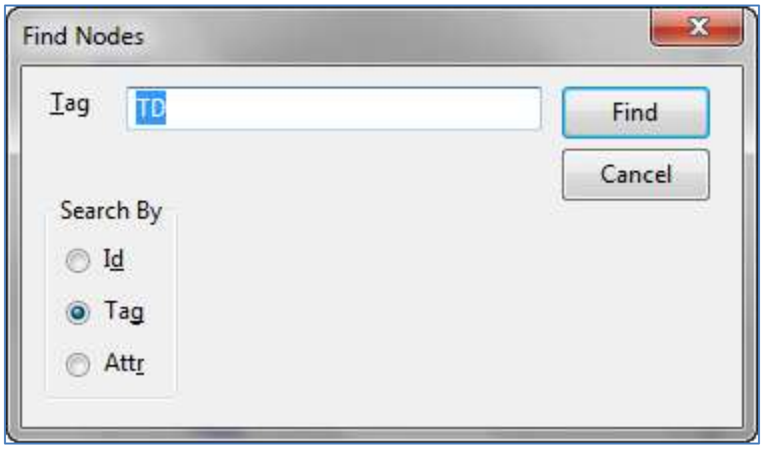
Local Name: input
Namespace URI: http://www.w3.org/1999/xhtml
Node Type: Element

Attributes

name	value
type	text
value	
maxlength	40
size	40
name	customer
id	customer

Figure 10

You might also want to find all nodes of a certain type, for example all those with a **tag** of TD. This can be done within the search window by specifying that you wish to search by **tag** (see Figure 11).



Tag: TD

Search By
- Id
- [✓ Tag]
- Attr

Figure 11

Once you click on the **Find** button, you will be brought to the first node with a TD tag. If you wish to find more nodes with this tag, simply press **control-g**.

Exercise A:
1. Find the number of nodes with a TH tag in this HTML document.

The Built-In Browser

You might be wondering how we can find nodes without knowing the ID, and how we can make changes to the DOM tree and see its impact in the HTML document. The DOM Inspector has a built-in browser which shows the webpage based on the DOM shown in the inspector. Let's first open up the browser within the DOM Inspector by clicking on the Inspect button. Figure 12 shows what your DOM Inspector should look like after doing this.

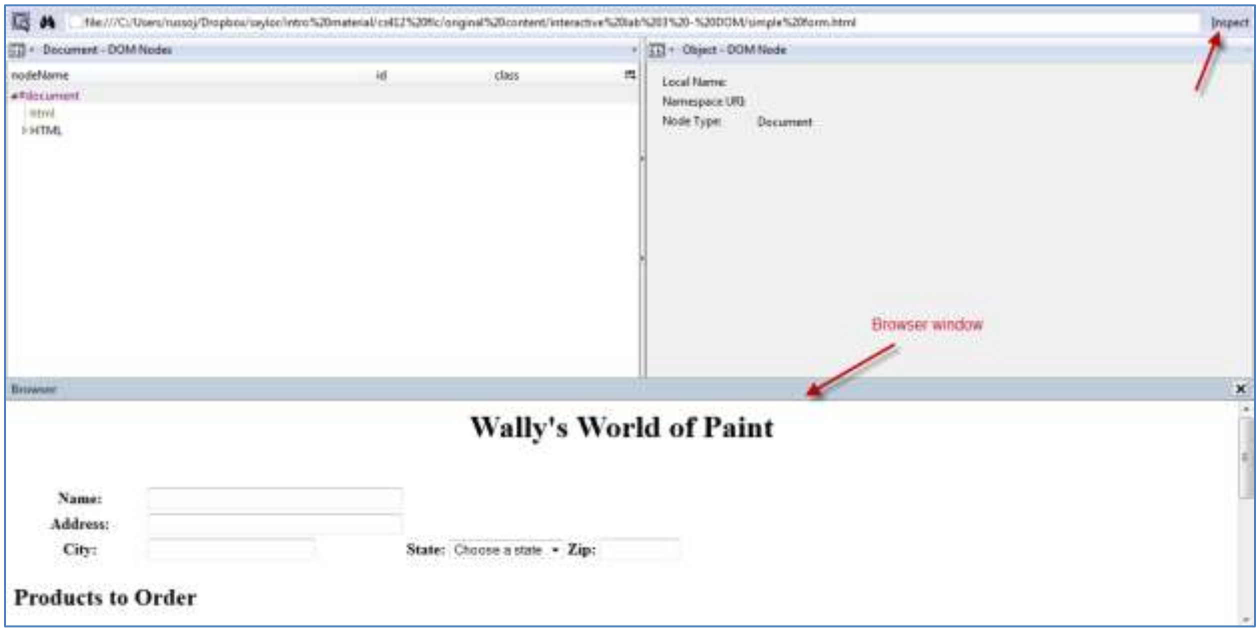
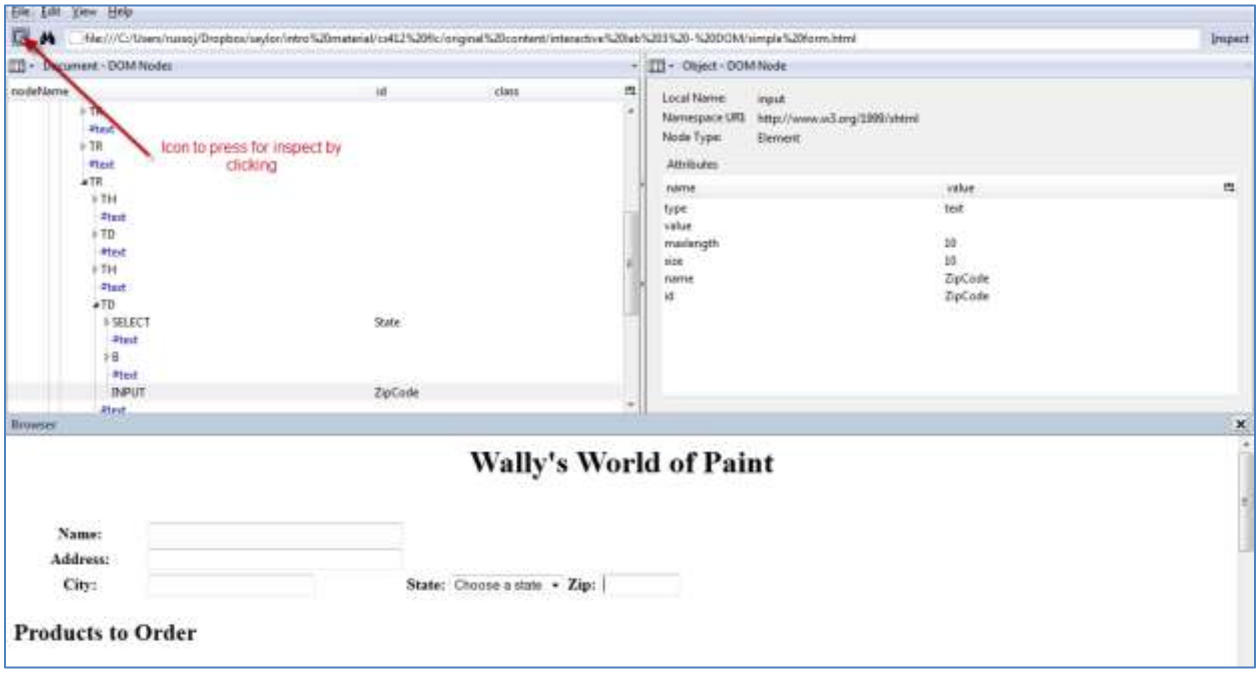


Figure 12

Now, what makes this really interesting is that you now can actually find a node by clicking on a part of the webpage displayed in the built-in browser. So, let's say that I want to find out the node for the zip code: I could simply click on the **Zip code textarea**. In order to do this, you must click on the **icon** to the left of the binoculars. If you were to do this in the DOM Inspector, your screen would look like Figure 13 (I've also placed an arrow pointing to the Icon to click on).



file:///C:/Users/russoj/Dropbox/saylor/intro%20material/cs412%20fc/original%20content/interactive%20lab%203%20-%20DOM/simple%20form.html

Document - DOM Nodes

nodeName	id	class
TR		
#text		
TR		
#text		
TR		
TH		
#text		
TD		
#text		
TH		
#text		
TD		
SELECT	State	
#text		
B		
#text		
INPUT	ZipCode	
#text		

Object - DOM Node

Local Name: input
Namespace URI: http://www.w3.org/1999/xhtml
Node Type: Element

Attributes

name	value
type	text
value	
maxlength	10
size	10
name	ZipCode
id	ZipCode

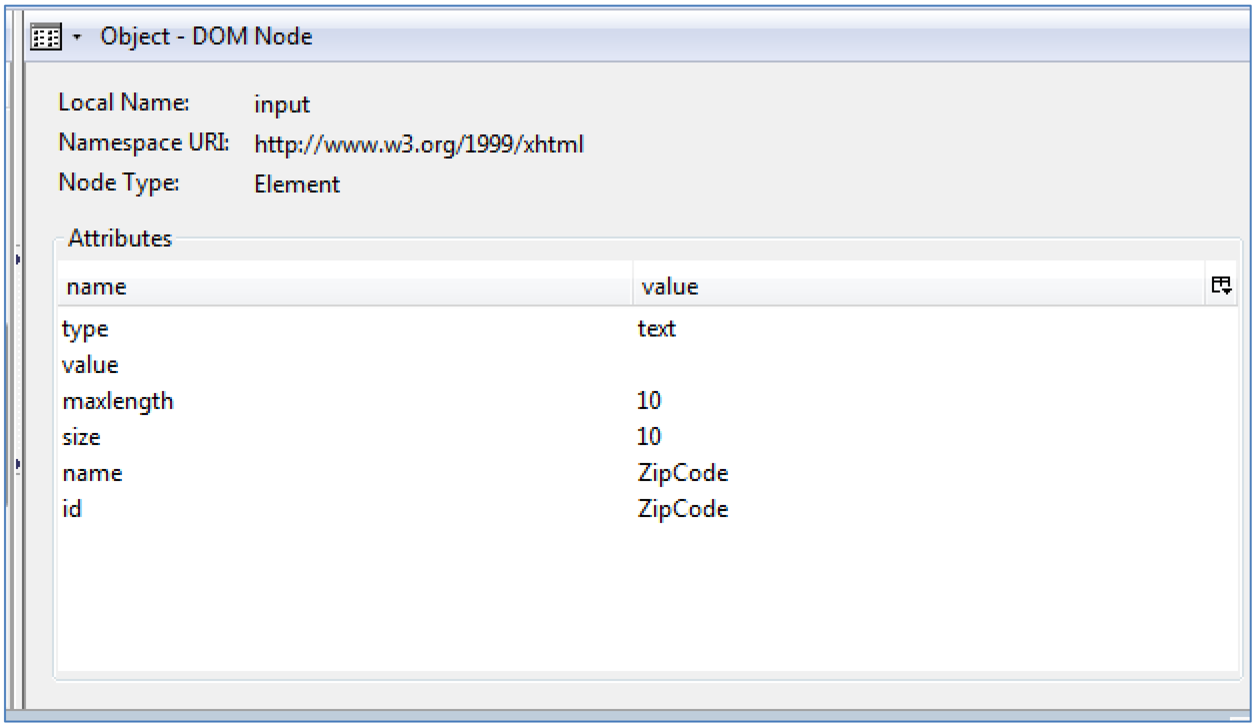
Wally's World of Paint

Name:
Address:
City: State: Choose a state Zip:

Products to Order

Figure 13

One thing you will notice is that the upper right-hand side has the information about this node. This is shown a bit clearer in Figure 14.



Object - DOM Node

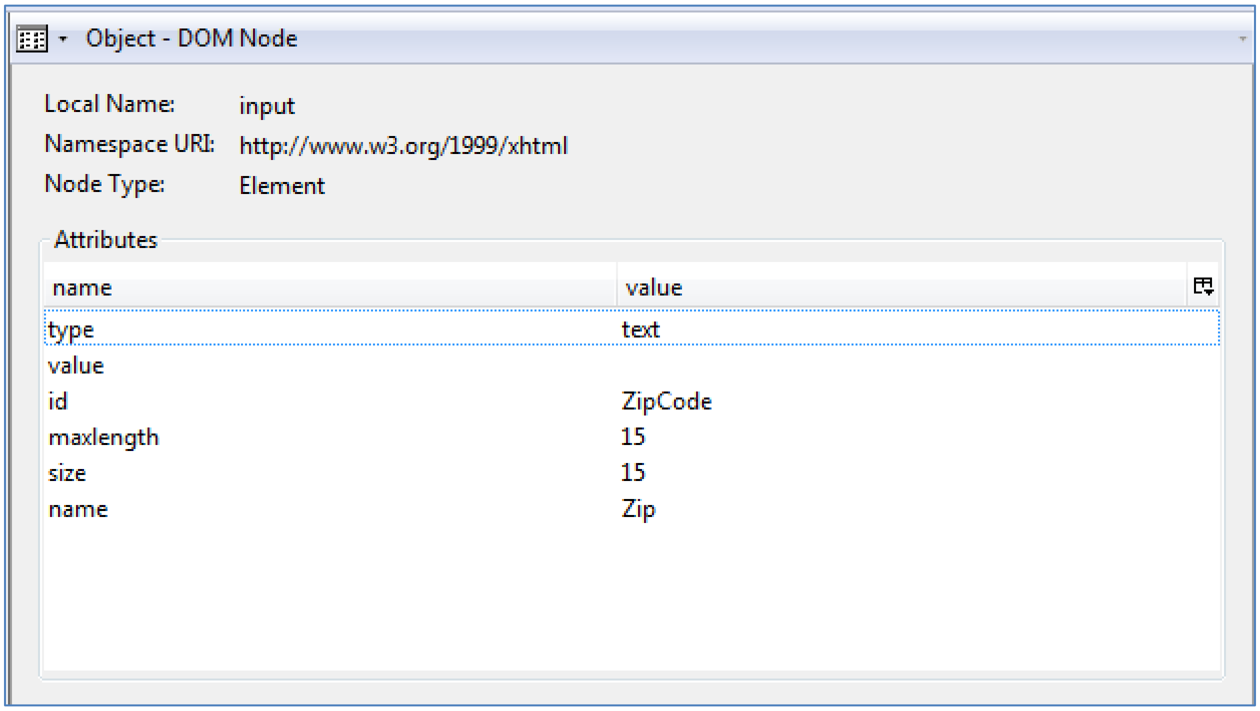
Local Name: input
Namespace URI: http://www.w3.org/1999/xhtml
Node Type: Element

Attributes

name	value
type	text
value	
maxlength	10
size	10
name	ZipCode
id	ZipCode

Figure 14

You can change the right-hand pane to reflect a change in the length of the text area or the name or ID. Let's go ahead and make this length **15** and change the name to **Zip**. Please note that in order to change each property, you must right click and then select **edit** from the menu. Figure 15 shows the changes in the right-hand pane and Figure 16 shows the changes implemented in the browser.

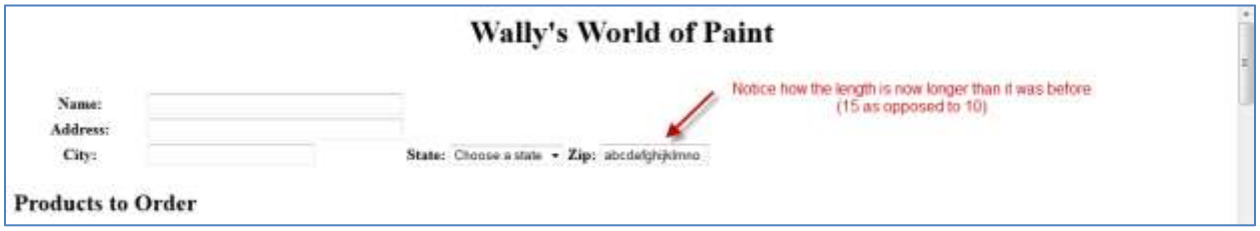


Local Name: input
Namespace URI: http://www.w3.org/1999/xhtml
Node Type: Element

Attributes

name	value
type	text
value	
id	ZipCode
maxlength	15
size	15
name	Zip

Figure 15



Wally's World of Paint

Name:
Address:
City:
State: Choose a state Zip: abcdefghijklmno

Notice how the length is now longer than it was before (15 as opposed to 10)

Products to Order

Figure 16

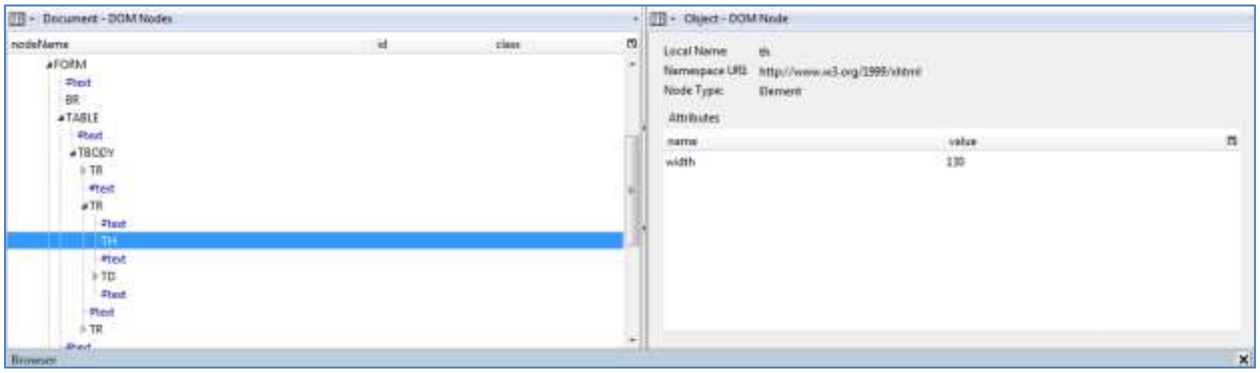
A couple of things to note here:

1. Changes made to the DOM in the DOM Inspector are *only* reflected in the DOM Inspector browser, not the actual browser where you might have the page rendered.
2. Once you close the DOM Inspector browser, the changes are gone. No changes are saved to the actual HTML file.

While not being able to save changes might not seem advantageous right now, it is quite handy to be able to do this programmatically.

Changing Text (such as headers and descriptive labels)

You might want to programmatically change the text on a page at any point in time. This can simply be done by changing the text in a specific node. Let's first click on the **Find a node by clicking on it** icon and then click on **Address**: Figure 17 shows the results of this:



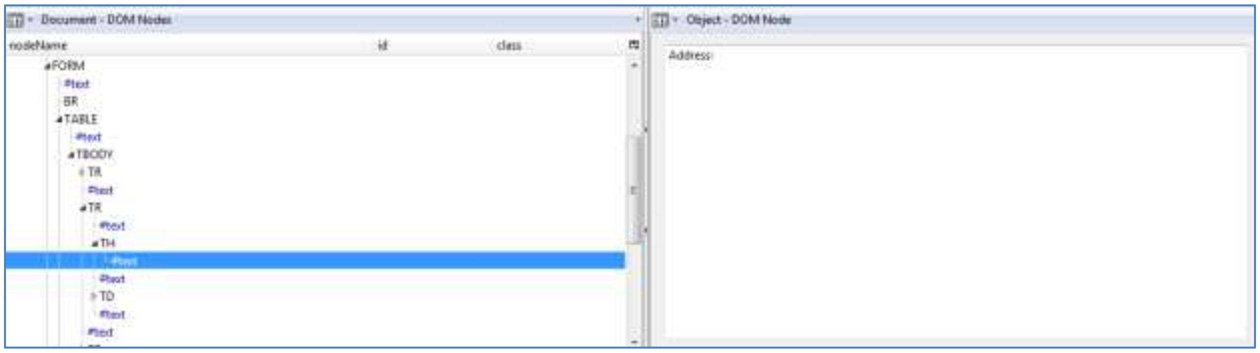
Local Name th
Namespace URI http://www.w3.org/1999/xhtml
Node Type Element

Attributes

name	value
width	130

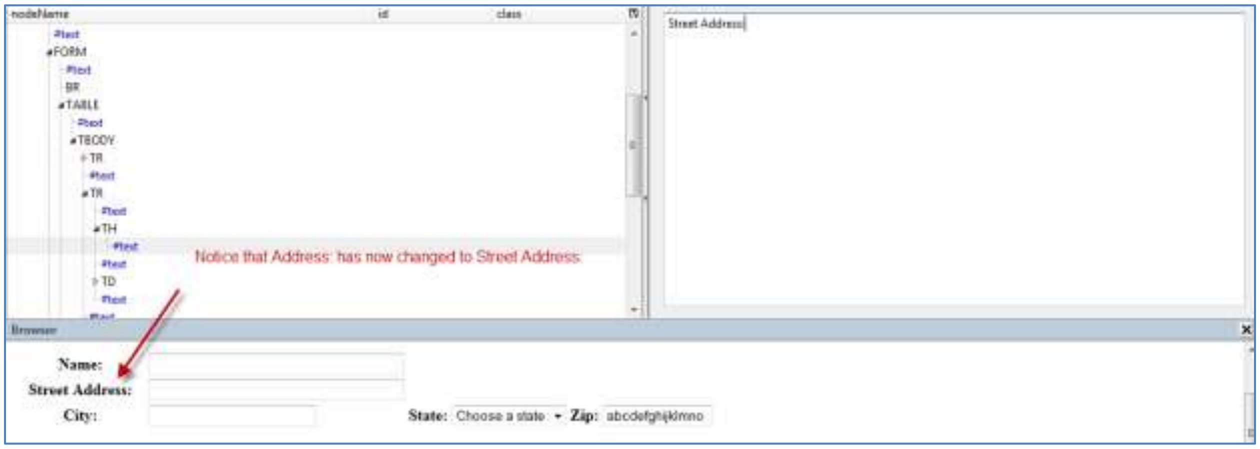
Figure 17

Notice how the blue line is on the **TH** element but the information in the right-hand side of the screen does not show **Address**: In order to access this, we need to open up the **TH** element by clicking on the **right arrow** and then clicking on **#Text**, which is a child. Figure 18 now shows the correct element, which we will change to **Street Address**: In order to change this, we can just type **Street Address** over the current text. Figure 19 shows the result after the change has been made.



Address:

Figure 18



```
nodeName                                   id          class
        #text
       ▲FORM
         #text
         BR
        ▲TABLE
          #text
         ▲TBODY
          ▷TR
           #text
          ▲TR
            #text
           ▲TH
              #text
            #text
           ▷TD
            #text
```

Street Address

Notice that Address: has now changed to Street Address:

Browser

Name:			
Street Address:			
City:		**State:** Choose a state	**Zip:** abcdefghijklmno

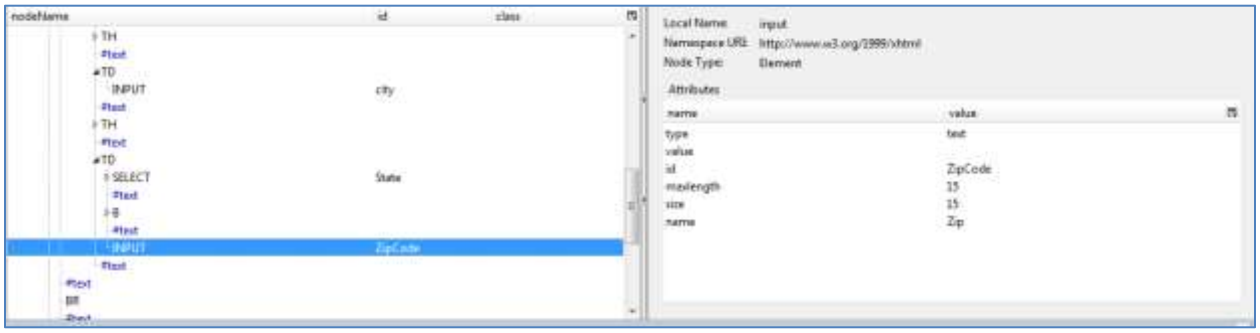
Figure 19

<u>Exercise B:</u>
1. Change the label for **Zip**: to **Zip code**:
2. Change the size and maxlength of **TextArea** city from **25** to **30**.

Adding and Deleting Elements

One of the most powerful aspects of using the DOM programmatically is that you can add and delete elements from a webpage on the fly. While you have not yet learned how to program in JavaScript, knowledge of the DOM will come in handy for some programming.

Let's first add a telephone number to our form. This can be done by finding the space where we would like to insert the telephone number after. In this case, it is the **Zip code textarea**. We can find this by clicking on the **find by clicking icon** and then clicking on the **Zip code textarea**. Figure 20 shows the selection, highlighted in blue on the left-hand side of the screen.



```
nodeName                                   id          class
           ▷TH
            #text
           ▲TD
             INPUT                         city
            #text
           ▷TH
            #text
           ▲TD
            ▷SELECT                        State
              #text
            ▷B
              #text
              INPUT                        ZipCode
            #text
        #text
        BR
```

Local Name: input
Namespace URI: http://www.w3.org/1999/xhtml
Node Type: Element

Attributes

name	value
type	text
value	
id	ZipCode
maxlength	15
size	15
name	Zip

Figure 20

At this point, we have some choices to make. We could either place the telephone number under the **Zip code** or next to it. Let's put the telephone number underneath. Since this is all contained in a table, we will need to add a new **TR** element, a **TH** element and a **TD** element as well as the appropriate text. We can add the **TR** element by going to the **TR** element right above **Zip code** (in this case, we want to go to TR that contains city, state and zip code) and right click. Figure 21 shows what this should look like. You want to select **Insert** and then **After**.

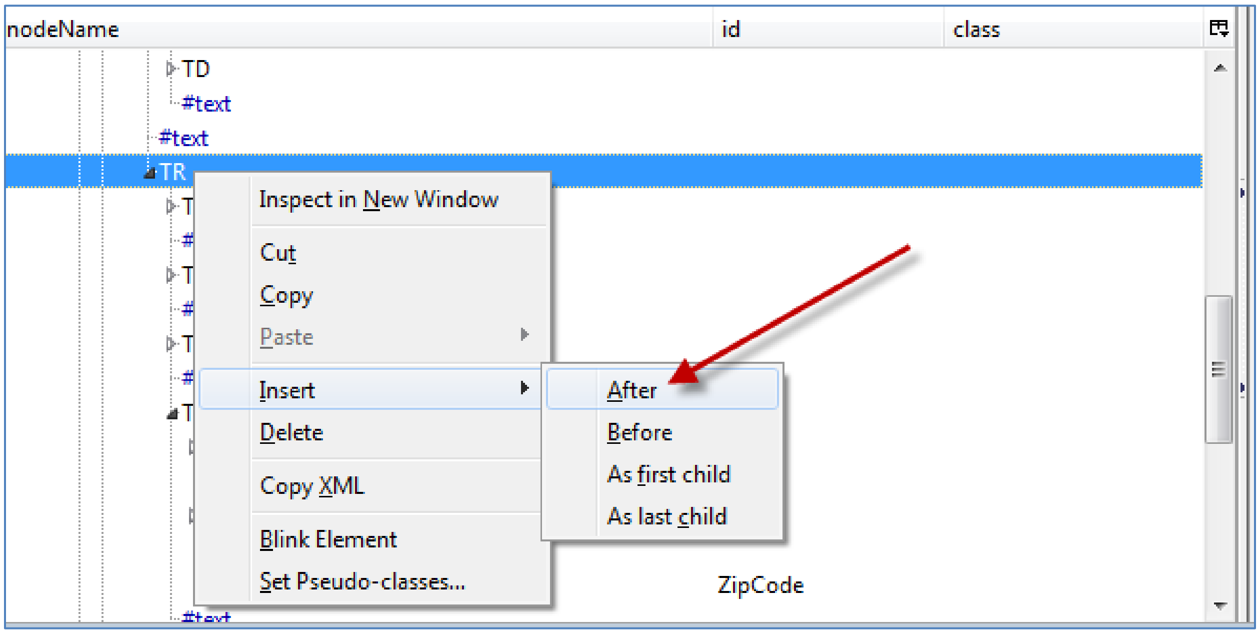
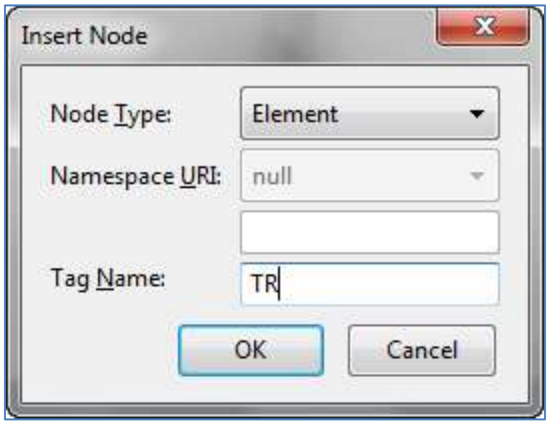


Figure 21

Once you select **After**, another window will popup, as shown in Figure 22. Select **Element** for the **Node Type** and enter **TR** for the **Tag Name**.

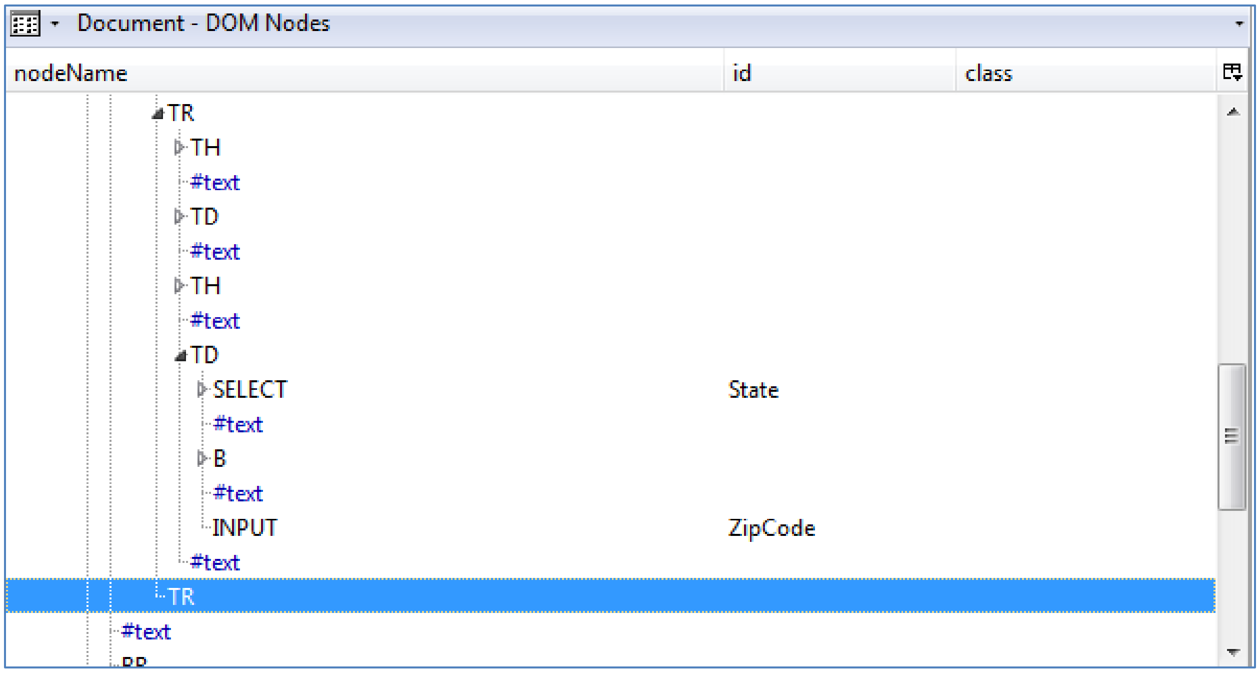


Node Type: Element
Namespace URI: null
Tag Name: TR

Figure 22

Click on **OK** and the node will be inserted.

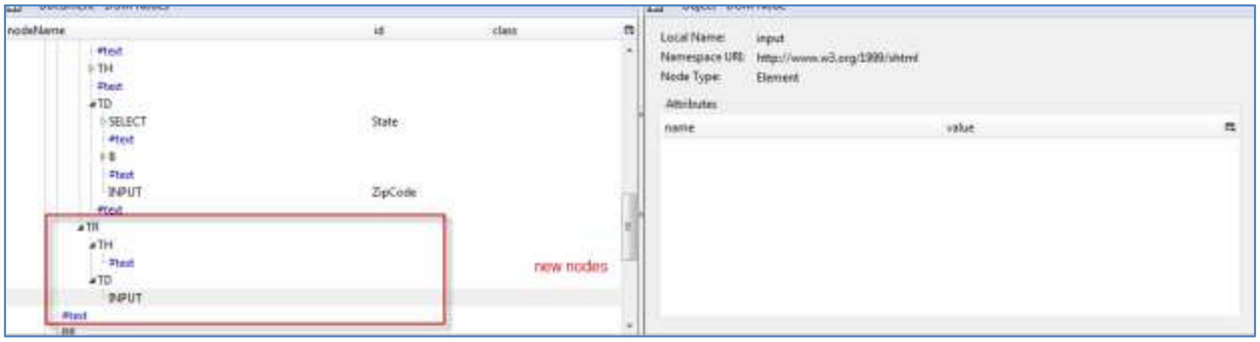
You will now see a new **TR** node inserted below **Zip code**, as shown in Figure 23.



```
Document - DOM Nodes
nodeName                                   id          class
        TR
          TH
          #text
          TD
          #text
          TH
          #text
          TD
            SELECT                         State
            #text
            B
            #text
            INPUT                          ZipCode
          #text
        TR
      #text
```

Figure 23

You next want to basically follow the same structure as the table row that contains **City**, **State** and **Zip code**. Add another child of the **TH** node, but this time select **text** for the type. Enter **telephone** for the node value. Go back to the **TR** element, add another child which will be a **TD** node type. Make sure that you specify **insert** as last child. Finally, you need to insert an input type node under the **TD** node. Insert this as a child. So far, you should see something like Figure 24.

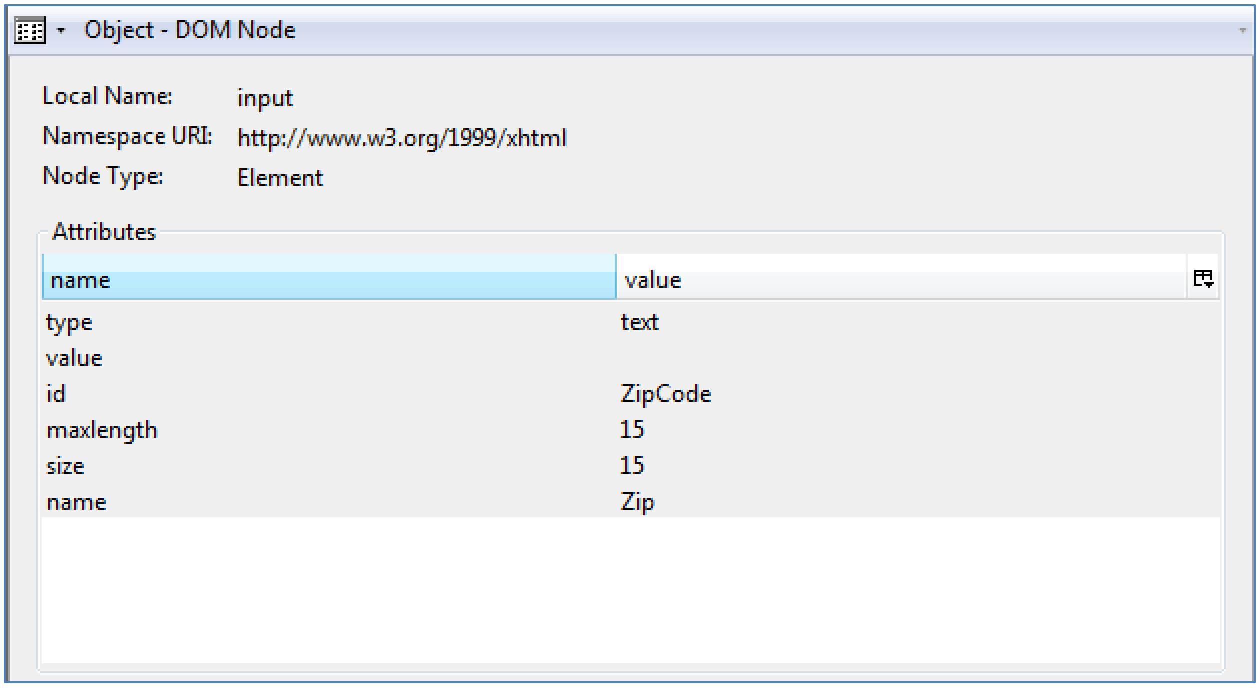


```
nodeName                                   id          class
          #text
          TH
          #text
          TD
            SELECT                         State
            #text
            B
            #text
            INPUT                          ZipCode
          #text
        TR
          TH
            #text
          TD
            INPUT
      #text
```

```
Local Name:      input
Namespace URI:   http://www.w3.org/1999/xhtml
Node Type:       Element
Attributes
name             value
```

Figure 24

Let’s go back up to the **Zip code** text area and look at the attributes in the right-hand window (see Figure 25).


Object - DOM Node

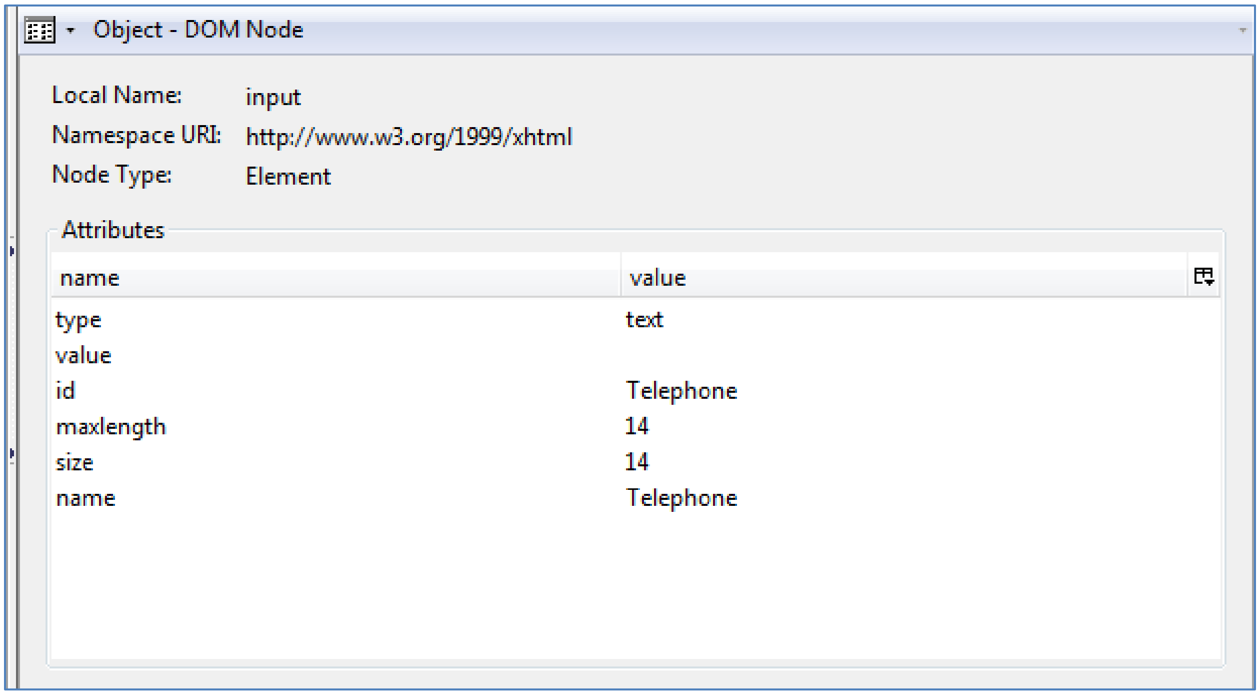
Local Name: input
Namespace URI: http://www.w3.org/1999/xhtml
Node Type: Element

Attributes

name	value
type	text
value	
id	ZipCode
maxlength	15
size	15
name	Zip

Figure 25

If you examine the input node that you just created, you will notice that there are no attributes. You could add each one individually. However, perhaps the easiest way to add these is to copy them. Hold the **shift key** down, click on the **first attribute**, click on the **last attribute**, **right click** and select **copy**. You can then paste these in the input node under the **TD** node that you just created. Figure 26 shows the values that you should enter for each attribute. Remember that you must **right-click** and select **edit** for each attribute you wish to change.



Local Name:	input
Namespace URI:	http://www.w3.org/1999/xhtml
Node Type:	Element

Attributes

name	value
type	text
value	
id	Telephone
maxlength	14
size	14
name	Telephone

Figure 26

Figure 27 shows the new rendering of the document in the DOM Inspector browser window.

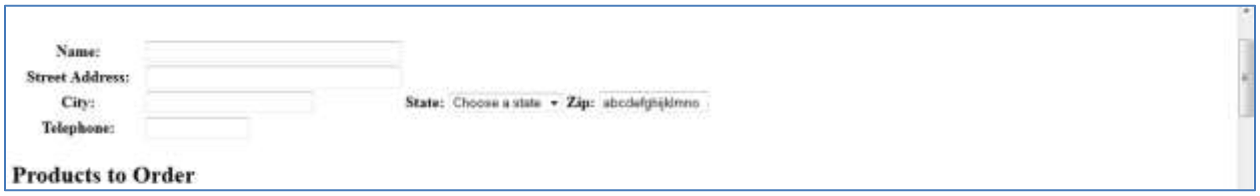


Figure 27

<u>Exercise C:</u>

1. Add a row to the table after **Sales Tax** and before **Total** for shipping charges. This should be a **textarea**.

Deleting Nodes

In a similar fashion, you can delete a node or an entire sub-tree. For example, if you wanted to remove the **city**, **state** and **zip code** table row, you could simply select the **table row** element, **right click** and select **delete** (see Figure 28).

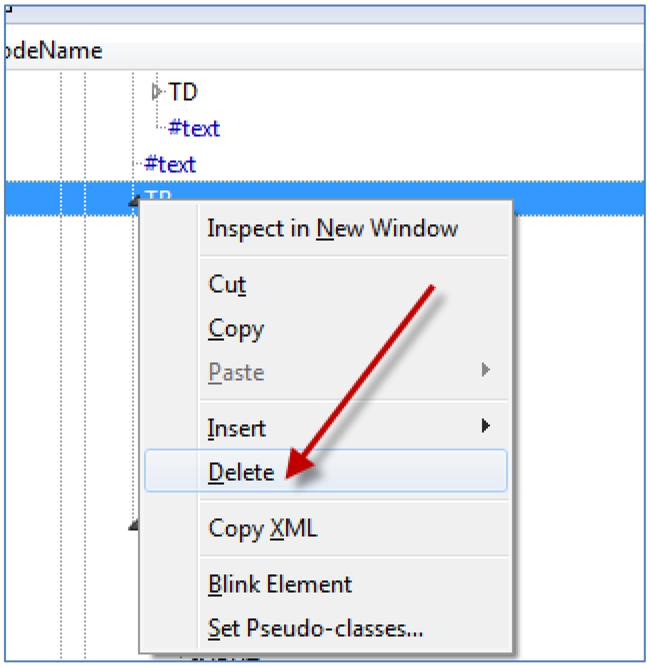


Figure 28

Cutting and Pasting Nodes

One other selection in the menu (as seen in Figure 28) is to cut and paste. It is possible to cut an entire section of the DOM tree and paste it below or above another section. You also can copy one node or a node and all sub-nodes.

> Exercise D
> As a final exercise, complete the following:
> Change the **Name** label and **textarea** to a **First Name**: label with a **firstname textarea** and a **Last Name**: label with a **lastname textarea**.

Conclusions

Much of what we have done in this lab was for the purpose of gaining some experience with the Document Object Model. In practice, you would write programs to complete most of the tasks. Following are some tasks that are common DOM manipulations programmatically:

1. Deleting some input elements due to a particular user response
2. Adding elements to a webpage due to a user response
3. Adding error messages after form input