

JavaScript 2: Using JavaScript

Chapter 9

Chapter 9

1

The Document
Object Model
(DOM)

2

Modifying the
DOM

3

Events

4

Event Types

5

Forms

6

Summary

Chapter 9

1

The Document
Object Model
(DOM)

2

Modifying the
DOM

3

Events

4

Event Types

5

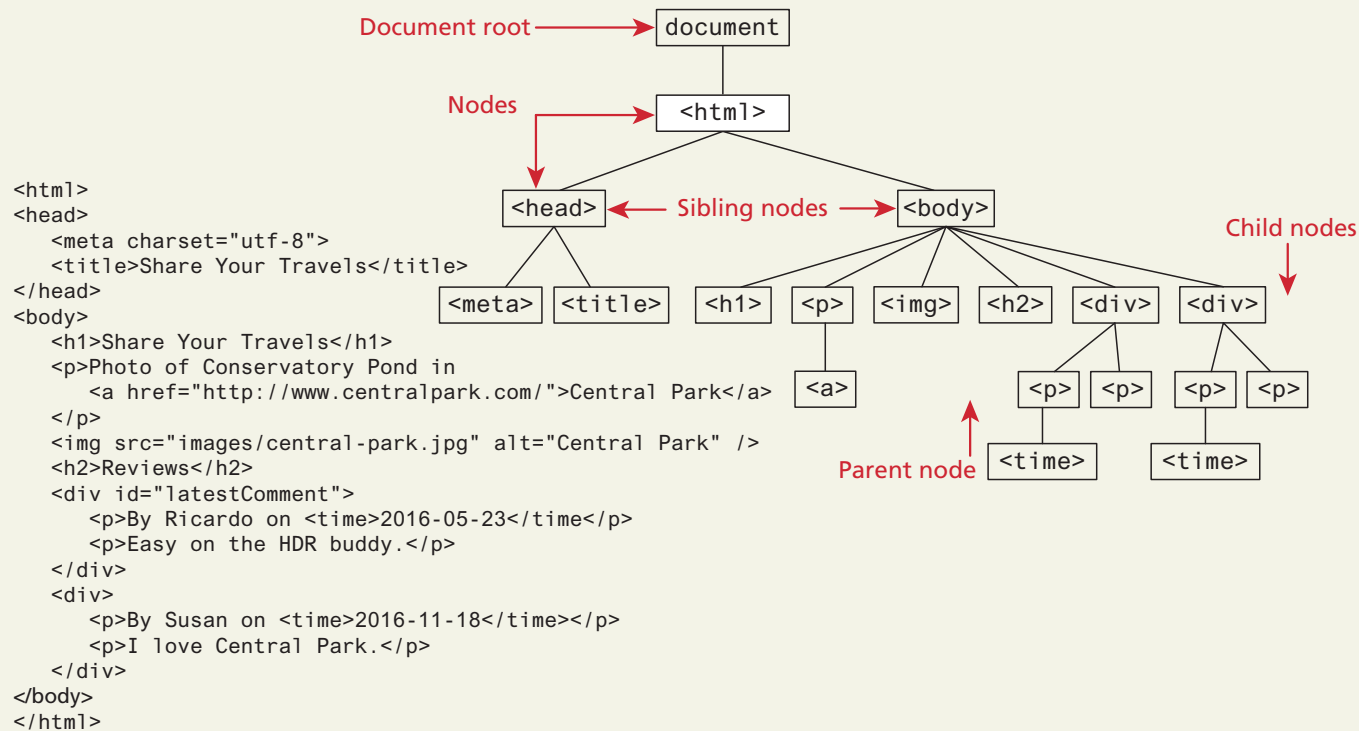
Forms

6

Summary

The Document Object Model (DOM)

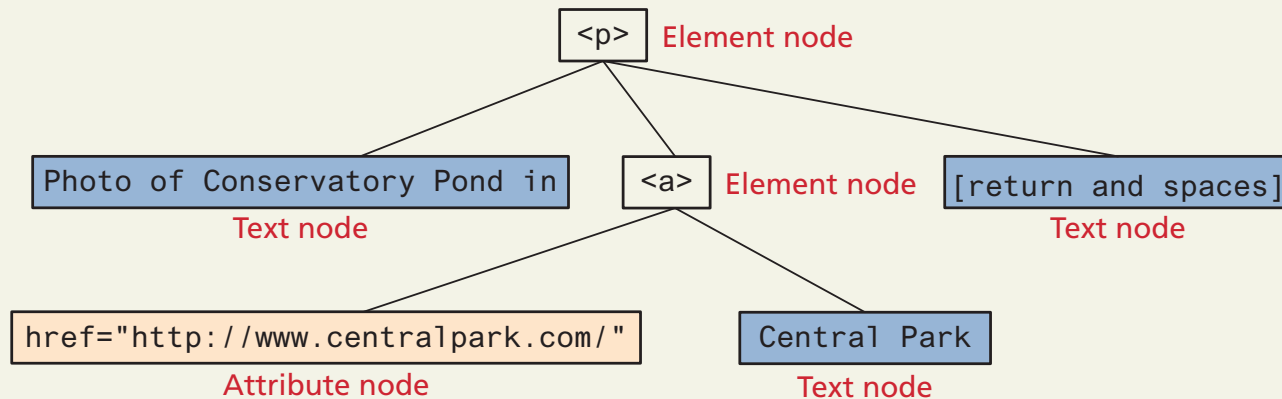
Overview



The Document Object Model (DOM)

Nodes and NodeLists

```
<p>Photo of Conservatory Pond in  
  <a href="http://www.centralpark.com/">Central Park</a>  
</p>
```



The Document Object Model (DOM)

Document Object

The DOM document object is the root JavaScript object representing the entire HTML document

// retrieve the URL of the current page

```
var a = document.URL;
```

// retrieve the page encoding, for example ISO-8859-1

```
var b = document.inputEncoding;
```

The Document Object Model (DOM)

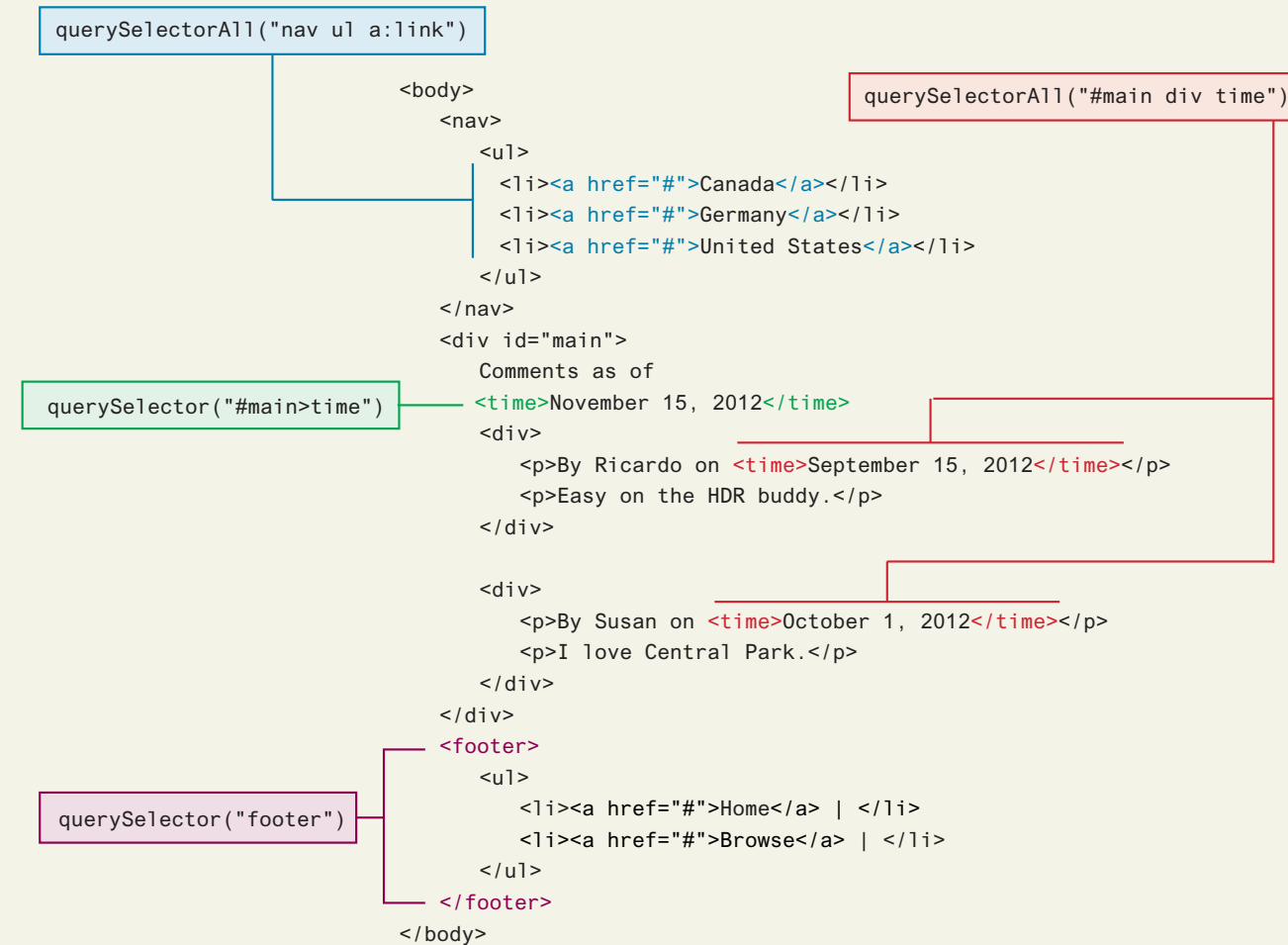
Selection Methods

Classic

- `getElementById()`
- `getElementsByTagName()`
- `getElementsByClassName()`

Newer

- `querySelector()` and
 - `querySelectorAll()`
-



The Document Object Model (DOM)

Element Node Object

Element Node object represents an HTML element in the hierarchy, contained between the opening `<>` and closing `</>` tags for this element. Every node has

- `classList`
 - `className`
 - `Id`
 - `innerHTML`
 - `Style`
 - `tagName`
-

The Document Object Model (DOM)

More common (not universal) properties

- href
 - name
 - src
 - value
-

Chapter 9

1

The Document
Object Model
(DOM)

2

Modifying the
DOM

3

Events

4

Event Types

5

Forms

6




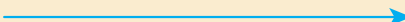
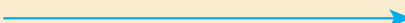
Summary

Modifying the DOM

Changing an Element's Style

```
<style>
  .box {
    margin: 2em; padding: 0;
    border: solid 1pt black;
  }
  .yellowish { background-color: #EFE63F; }
  .hide { display: none; }
</style>
<main>
  <div class="box">
    ...
  </div>
</main>
```

```
var node = document.querySelector("main div");
```

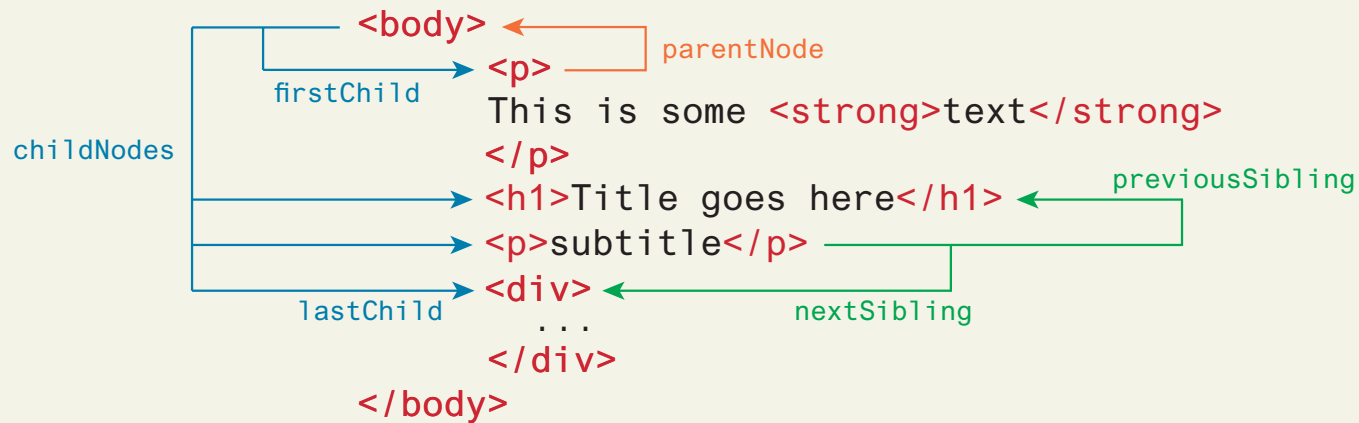
- 1 `node.className = "yellowish";`  This replaces the existing class specification with this one. Thus the `<div>` no longer has the `box` class
- 2 `node.classList.remove("yellowish");`  `node.classList.add("box");` Removes the specified class specification and adds the `box` class
- 3 `node.classList.add("yellowish");`  Adds a new class to the existing class specification
- 4 `node.classList.toggle("hide");`  If it isn't in the class specification, then add it
- 5 `node.classList.toggle("hide");`  If it is in the class specification, then remove it

Equivalent to:

- 1 `<div class="yellowish">`
- 2 `<div class="">`
`<div class="box">`
- 3 `<div class="box yellowish">`
- 4 `<div class="box yellowish hide">`
- 5 `<div class="box yellowish">`

Modifying the DOM

Meet the family



Modifying the DOM

Changing an Element's Content

```
document.getElementById("here").innerHTML =  
"foo<em>bar</em>";
```

Modifying the DOM

Creating DOM elements

- 1 Create a new text node

"this is dynamic"

```
var text = document.createTextNode("this is dynamic");
```

- 2 Create a new empty <p> element

<p></p>

```
var p = document.createElement("p");
```

- 3 Add the text node to new <p> element

```
p.appendChild(text);
```

<p> "this is dynamic" </p>

- 4 Add the <p> element to the <div>

```
var first = document.getElementById("first");  
first.appendChild(p);
```


Modifying the DOM

Creating DOM elements

- 4 Add the `<p>` element to the `<div>`

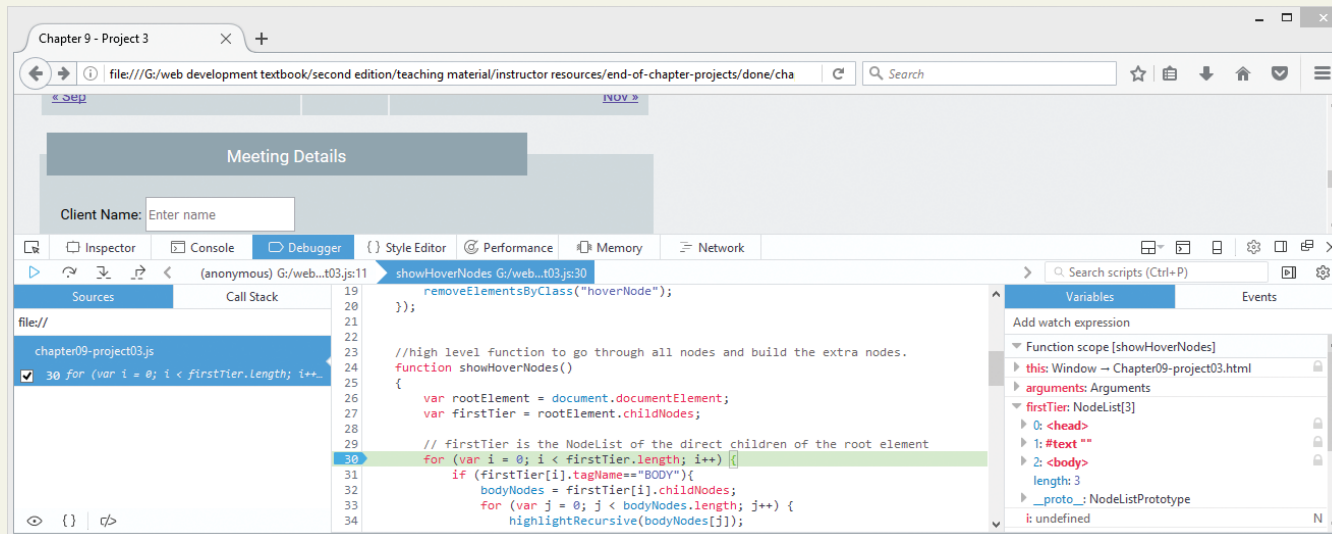
```
var first = document.getElementById("first");  
first.appendChild(p);
```

```
<div id="first">  
  <h1>DOM Example</h1>  
  <p>Existing element</p>  
  <p>this is dynamic</p>  
</div>
```

```
<div>  
  <h1> "DOM Example" </h1>  
  <p> "Existing element" </p>  
  <p> "this is dynamic" </p>  
</div>
```

Modifying the DOM

Tools - Debuggers



Modifying the DOM

Tools – Performance checkers

The screenshot shows a web browser window with a form titled "Chapter 9 - Project 3". The form contains a "Client Type" section with radio buttons for "LABEL", "School", "College", and "University", and a "Meeting Date" section with a "LABEL" and a date input field. A "Submit" button and a "FIELDSET" label are also visible.

Below the browser window, the Chrome DevTools Performance tab is open, displaying a CPU profile. The profile shows a list of functions and their execution times. The functions are listed in a table with columns for Self Time, Total Time, and Function. The functions include (idle), (program), log, highlightRecursive, appendChild, wrapObject, createElement, showHoverNodes, addEventListener, createTextNode, InjectedScript.RemoteObject, (anonymous), and _wrapObject.

Self Time	Total Time	Function
7769.9 ms	7769.9 ms	(idle)
51.0 ms	49.36 %	(program)
39.1 ms	37.82 %	▶ log
10.4 ms	10.06 %	▶ highlightRecursive
0.5 ms	0.53 %	▶ appendChild
0.4 ms	0.42 %	▶ wrapObject
0.3 ms	0.32 %	▶ createElement
0.3 ms	0.32 %	▶ showHoverNodes
0.2 ms	0.21 %	▶ addEventListener
0.2 ms	0.21 %	▶ createTextNode
0.2 ms	0.21 %	▶ InjectedScript.RemoteObject
0.1 ms	0.11 %	(anonymous)
0.1 ms	0.11 %	(anonymous)
0.1 ms	0.11 %	▶ (anonymous)
0.1 ms	0.11 %	▶ _wrapObject
0.1 ms	0.11 %	(anonymous)

The bottom of the screenshot shows the Console tab with several log messages:

```
nodeType=3 tagName=undefined
nodeType=3 tagName=undefined
nodeType=3 tagName=undefined
```

Modifying the DOM

Tools – Linters

The image shows two web-based JavaScript linters, JSLint and JSHint, processing the same code snippet. The code is as follows:

```
1 var radios = document.querySelectorAll("input[name=region]");
2 for (var i=0; i < radios.length; i++) {
3   if(radios[i].checked) {
4     // if here then this radio button was selected
5     alert(radios[i].value + " was checked")
6     this.checked = true;
7   }
8 }
```

JSLint Results: The JSLint interface (left) shows a "Warnings" section with the message "JSLint was unable to finish." and a list of specific errors:

- Unexpected 'for'. *line 1 column 0*
- Unexpected 'var'. *line 1 column 3*
- Unexpected trailing space. *line 3 column 55*
- Unexpected trailing space. *line 5 column 29*

JSHint Results: The JSHint interface (right) shows a "One warning" section with the message "Missing semicolon." at line 5, column 55.

Chapter 9

1

The Document
Object Model
(DOM)

2

Modifying the
DOM

3

Events

4

Event Types

5

Forms

6

Summary

Events

JavaScript event is an action that can be detected by JavaScript

- Many of them are initiated by user actions
- some are generated by the browser itself.

We say that an event is *triggered* and then it is *handled* by JavaScript functions

Events

Event-Handling Approaches – Inline Hook

HTML document using the inline hooks

```
...  
<script type="text/javascript" src="inline.js"></script>  
...  
<form name='mainForm' onsubmit="validate(this);">  
  <input name="name" type="text"  
    onchange="check(this);"   
    onfocus="highlight(this, true);"   
    onblur="highlight(this, false);">  
  <input name="email" type="text"  
    onchange="check(this);"   
    onfocus="highlight(this, true);"   
    onblur="highlight(this, false);">  
  <input type="submit"  
    onclick="function (e) {  
      ...  
    }">  
...
```

inline.js

```
function validate(node) {  
  ...  
}  
function check(node) {  
  ...  
}  
function highlight(node) {  
  ...  
}
```

Notice that you can define an entire event handling function within the markup. This is NOT recommended!

Events

Event-Handling Approaches – Event Property Approach

```
var myButton = document.getElementById('example');  
myButton.onclick = alert('some message');
```

Events

Event-Handling Approaches – Event Listener Approach

```
var myButton = document.getElementById('example');
```

```
myButton.addEventListener('click', alert('some  
message'));
```

```
myButton.addEventListener('mouseout', funcName);
```

Events

Event-Handling Approaches – Event Listener Approach (anon function)

```
myButton.addEventListener('click', function() {  
    var d = new Date();  
    alert("You clicked this on "+ d.toString());  
});
```

Events

Event Object

When an event is triggered, the browser will construct an event object that contains information about the event.

```
div.addEventListener('click', function(e) {  
  
    // find out where the user clicked  
  
    var x = e.clientX;  
  
    ...  
})
```

Events

Event Object

- `bubbles` Indicates whether the event bubbles up through the DOM
 - `cancelable` Indicates whether the event can be cancelled
 - `target` The object that generated (or dispatched) the event
 - `type` The type of the event (see Section 9.4)
-

Chapter 9

1

The Document
Object Model
(DOM)

2

Modifying the
DOM

3

Events

4

Event Types

5

Forms

6

Summary

Event Types

Mouse Events

- `click` The mouse was clicked on an element
 - `dblclick` The mouse was double clicked on an element
 - `mousedown` The mouse was pressed down over an element
 - `mouseup` The mouse was released over an element
 - `mouseover` The mouse was moved (not clicked) over an element
 - `mouseout` The mouse was moved off of an element
 - `mousemove` The mouse was moved while over an element
-

Event Types

Keyboard Events

- `keydown` The user is pressing a key (this happens first)
 - `keypress` The user presses a key (this happens after `keydown`)
 - `keyup` The user releases a key that was down (this happens last)
-

Event Types

Touch Events

Touch events are a new category of events that can be triggered by devices with touch screens

Limited Browser support (2017)

The different events (e.g., touchstart, touchmove, and touchend) are analogous to some of the mouse events (mousedown, mousemove, and mouseup).

Event Types

Form Events

- Blur
 - Change
 - Focus
 - Reset
 - select
 - Submit
-

Event Types

Frame Events

- abort An object was stopped from loading
 - error An object or image did not properly load
 - load When a document or object has been loaded
 - resize The document view was resized
 - scroll The document view was scrolled
 - unload The document has unloaded
-

Chapter 9

1

The Document
Object Model
(DOM)

2

Modifying the
DOM

3

Events

4

Event Types

5

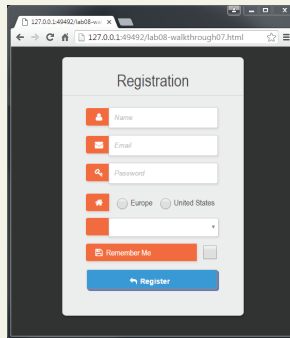
Forms

6

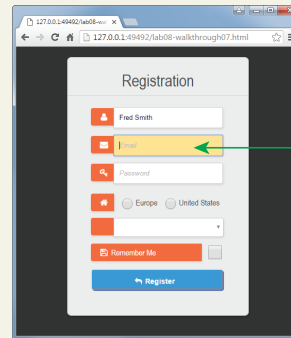
Summary

Forms

Responding to Form Movement Events



How form appears when no controls have the focus



When a control has the focus, then change its background color

```
// This function is going to get called every time the focus or blur events are
// triggered in one of our form's input elements.
function setBackground(e) {
  if (e.type == "focus") {
    e.target.style.backgroundColor = "#FFE393";
  }
  else if (e.type == "blur") {
    e.target.style.backgroundColor = "white";
  }
}

// set up the event listeners only after the DOM is loaded
window.addEventListener("load", function() {
  var cssSelector = "input[type=text],input[type=password]";
  var fields = document.querySelectorAll(cssSelector);
  for (i=0; i<fields.length; i++) {
    fields[i].addEventListener("focus", setBackground);
    fields[i].addEventListener("blur", setBackground);
  }
});
```

Here we use the `style` property instead of the `classList` property because of specificity conflicts (i.e., attribute selectors override class selectors).

Selects the fields that will change.

Assigns the `setBackground()` function to change the background color of the control depending upon whether it has the focus.

Forms

Responding to Form Changes Events

```
// depending on the state of the region radio buttons
// change the options of the select list
var label = document.getElementById("payLabel");
var select = document.getElementById("payment");
select.disabled = true;
var radios = document.querySelectorAll("input[name=region]");
// listen to each radio button
for (var i=0; i < radios.length; i++) {
    // whenever a radio button changes, modify the select
    // list as well as the label beside it
    radios[i].addEventListener("change",
        function (e) {
            select.disabled = false;
            select.innerHTML = "";
            addOption(select, "Select Payment Type" , "0");
            var choice = e.target.value;
            if (choice == "United States") {
                // display the dollar symbol
                label.classList.remove("fa-euro");
                label.classList.add("fa-dollar");
                addOption(select, "American Express" , "1");
                addOption(select, "Mastercard" , "2");
                addOption(select, "Visa" , "3");
            }
            else if (choice == "Europe") {
                // display the euro symbol
                label.classList.remove("fa-dollar");
                label.classList.add("fa-euro");
                addOption(select, "Bitcoin" , "4");
                addOption(select, "PayPal" , "5");
            }
        }
    );
}

function addOption(select, optionText, optionValue) {
    var opt = document.createElement('option');
    opt.appendChild( document.createTextNode(optionText) );
    opt.value = optionValue;
    select.appendChild(opt);
}
```

The screenshot shows a 'Registration' form with fields for Name, Email, Password, and a region selector (Europe/United States). The 'payment' dropdown is disabled, indicated by a grey background and a downward arrow icon. A green arrow points from the code's initial state to this dropdown.

1 Initially the <select> list is disabled.

The screenshot shows the same registration form, but the 'payment' dropdown is now enabled. The 'United States' radio button is selected, and the 'payment' dropdown shows 'Select Payment Type' as the selected option. A green arrow points from the code's logic for the 'United States' choice to this dropdown.

2 But when user changes a radio button, enable the select list, ...

The screenshot shows the registration form with the 'payment' dropdown populated with five options: 'Select Payment Type', 'American Express', 'Mastercard', 'Visa', and 'PayPal'. A green arrow points from the code's logic for adding options to this dropdown.

4 ... and then populate the list with appropriate option values,

Use the DOM functions from Section 9.2 to create a new <option> element, populate it with the appropriate text, and then add it to the <select> element.

Forms

Validating a Submitted Form

- Field Validation
 - Number Validation
 - Other (non JavaScript) Form validation reminder
-

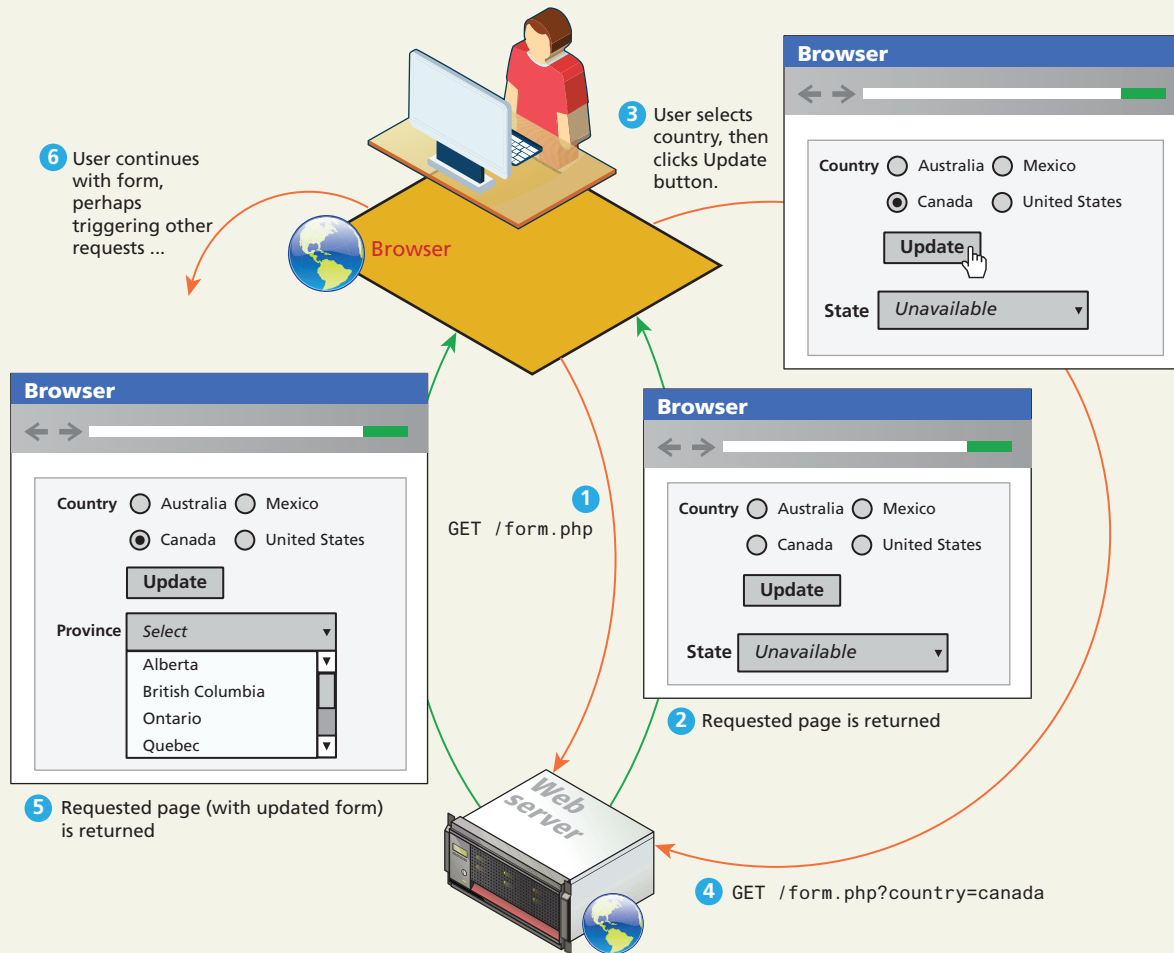
Forms

Submitting Forms

```
var formExample =  
document.getElementById("loginForm");  
  
formExample.submit();
```

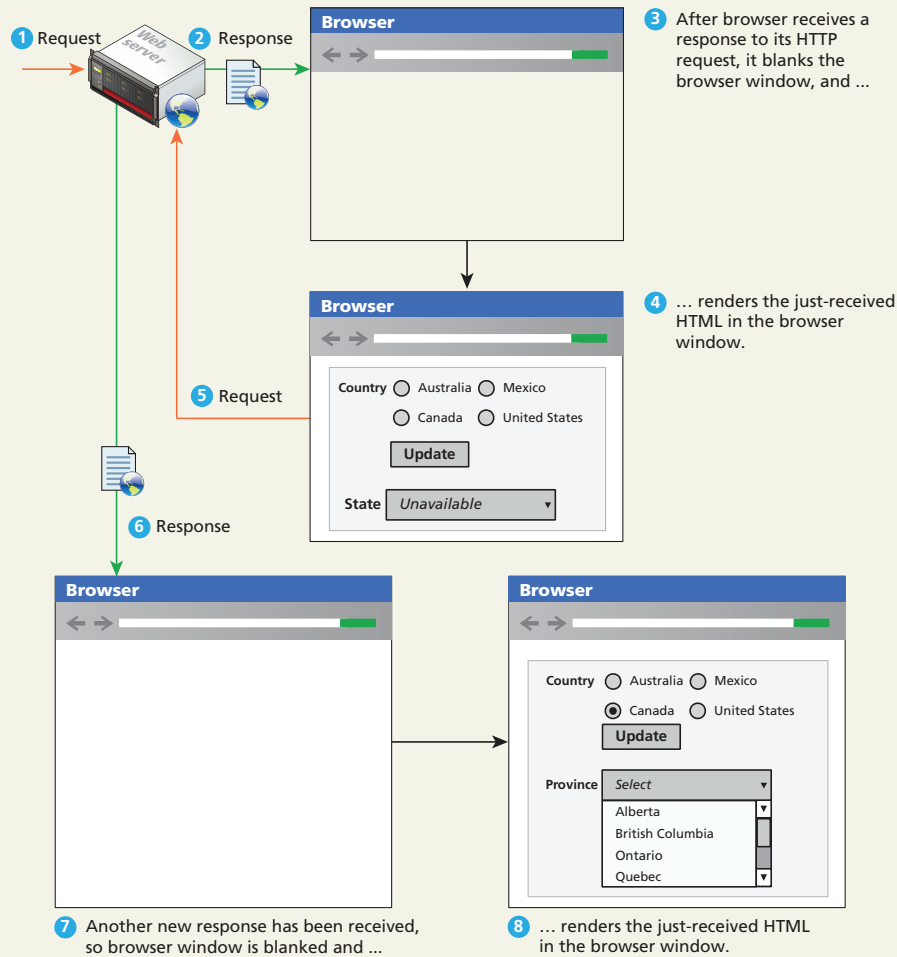
AJAX

First the Normal Request Response Loop



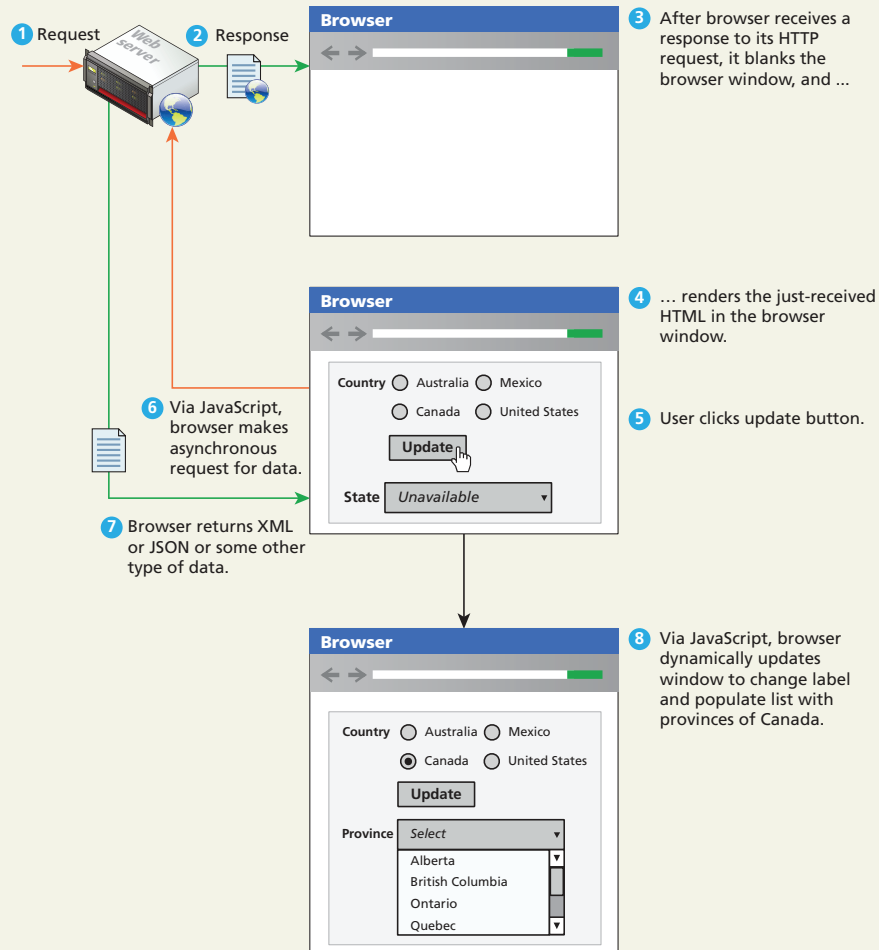
AJAX

The problem



AJAX

The solution



Chapter 9

1

The Document
Object Model
(DOM)

2

Modifying the
DOM

3

Events

4

Event Types

5

Forms

6

Summary

Summary

Key Terms

- blur
 - Document Object Model (DOM)
 - document root
 - DOM document object
 - DOM tree
 - element node
 - event
 - event bubbling
 - event delegation
 - event handler
 - event listener
 - event object
 - event propagation
 - event type
 - focus
 - form events
 - frame events
 - keyboard events
 - linter
 - mouse events
 - node
 - selection methods
 - touch events
-

Summary

Questions?