|  |  |  |
| --- | --- | --- |
| **LAB 9**  **JAVASCRIPT 2: DOM AND EVENTS** | | |
|  | **BẠN SẼ HỌC ĐƯỢC GÌ**   * Truy cập và sửa đổi phần tử DOM HTML bằng JavaScript * Tạo người nghe sự kiện để phản ứng với các sự kiện * Các công cụ và thủ thuật để giúp bạn phát triển JavaScript |  |
|  |
| **Thời gian gần đúng Các bài tập trong phòng thí nghiệm này sẽ mất khoảng 90 phút để hoàn thành.** |
| **Fundamentals of Web**  **Development, 2nd Ed**    Randy Connolly and Ricardo Hoar | | |
| Textbook by Pearson  http://www.funwebdev.com    Date Last Revised: Feb 9, 2017 | | |

**THE DOCUMENT OBJECT MODEL**

**(DOM)**

PREPARING DIRECTORIES

1. Nếu bạn đã hoàn thành, hãy tạo một thư mục trong ổ đĩa cá nhân của bạn cho tất cả các phòng thí nghiệm cho cuốn sách này.

2. Từ thư mục phòng thí nghiệm chính (được tải xuống từ trang web của sách giáo khoa bằng cách sử dụng mã được cung cấp cùng với sách hoặc ở một vị trí chung do người hướng dẫn của bạn cung cấp), sao chép thư mục có tiêu đề lab09 vào thư mục khóa học của bạn được tạo ở bước một.

Trong phòng thí nghiệm trước, bạn đã sử dụng hàm document.write () như một cách để thao tác đánh dấu bằng JavaScript. Mặc dù tốt từ góc độ học JavaScript, nhưng nhìn chung nó không sử dụng

JavaScript. Trong phần đầu tiên của phòng thí nghiệm này, bạn sẽ học cách sử dụng

Mô hình đối tượng tài liệu (DOM) với JavaScript như một cách xử lý nội dung tốt hơn.

***Exercise 9.1*** — BASIC DOM SELECTION

1. Kiểm tra lab09-walkENC01.html trong trình duyệt của bạn và sau đó trong trình chỉnh sửa bạn chọn. Lưu ý rằng nó tham chiếu một tệp JavaScript ở cuối đánh dấu. Để hiểu rõ hơn các bước còn lại trong bài tập này, bạn nên liên tục tham khảo lại phần đánh dấu*.*
2. Chỉnh sửa js / lab09-walkENC01.js bằng cách thêm các mục sau và sau đó kiểm tra lab09walkENC01.html trong trình duyệt.

**var msg = document.getElementById("msgArea"); msg.value = "here is some text";**

*Cái này chọn phần tử <textarea> và đặt giá trị của nó.*

1. Thêm mã sau đây và kiểm tra.

**var title = document.getElementById("pageTitle"); title.innerHTML = "New Title";**

1. Thay đổi mã trước thành sau:

**document.getElementById("pageTitle").innerHTML = "Even newer title";**

*Điều này thực hiện điều tương tự mà không cần sử dụng một biến tạm thời.*

1. Thêm mã sau đây đểkiểm tra.

**msg.value = document.getElementById("first").innerHTML;**

*Lưu ý rằng thuộc tínhHTML bên trong của một nút trả về tất cả HTML bên trong nó.*

1. Thay đổi mã trước đó thành mã sau và kiểm tra.

msg.value = document.getElementById("first").**innerText**;

*Lưu ý rằng thuộc tính InternalText của một nút trả về nội dung văn bản bị tước của bất kỳ thẻ nào.*

1. Thêm mã sau đây để kiểm tra.

**msg.value = "My class is " + msg.getAttribute("class");**

*Như bạn có thể thấy, phương thức getAttribution trả về giá trị của thuộc tính được chỉ định (trong trường hợp này là thuộc tính lớp).*

1. Thêm mã sau đây để kiểm tra.

**var elem = document.getElementsByTagName("textarea"); elem[0].value = "selected by tag name";**

*GetEuityByTagName trả về một mảng các nút phù hợp. Trong trường hợp này, nó là một phần tử chỉ có một phần tử.*

1. Thêm mã sau đây để kiểm tra.

**elem = document.getElementsByTagName("span"); for (i=0; i<elem.length; i++) {**

**document.getElementById("msgArea").value += "\n" + elem[i].innerText; }**

*Thao tác này sẽ chọn tất cả các phần tử <span> và thêm nội dung của chúng vào <textarea>.*

1. Thêm mã sau đây để kiểm tra:

**document.getElementById("msgArea").value = ""; var counts = document.getElementsByClassName("count-number"); for (i=0; i<counts.length; i++) {**

**document.getElementById("msgArea").value += "\n" + counts[i].innerText; }**

*Điều này sử dụng hàm getElementsByClassName () để tìm tất cả các phần tử có các lớp khớp.*

1. Thêm mã sau đây để kiểm tra:

**var ps = document.querySelector("#first p"); document.getElementById("msgArea").value = ps.innerHTML;**

*Hàm querySelector () cho phép bạn sử dụng bất kỳ bộ chọn CSS nào để chọn một phần tử. Nếu có nhiều phần tử có thể khớp với bộ chọn, nó chỉ trả về kết quả khớp đầu tiên.*

1. Thêm mã sau đây để kiểm tra.

**ps = document.querySelectorAll("#first p"); for (i=0; i<ps.length; i++) {**

**document.getElementById("msgArea").value += ps[i].innerHTML + "\n"; }**

*Hàm querySelector () trả về một mảng gồm tất cả các phần tử khớp.*

1. Thêm mã sau đây và kiểm tra.

**document.getElementById("msgArea").setAttribute("class", "hidden");**

*Điều này đặt thuộc tính lớp của phần tử <textarea> thành ẩn (đây là lớp CSS của trình trợ giúp được xác định trong Bootstrap).*

1. Cuối cùng, thêm mã sau đây và kiểm tra.

**document.getElementById("features").style.backgroundColor = "green";**

Cái này đặt màu nền của phần tử <div> của id có tên là “features”.

Trong bài tập tiếp theo, bạn sẽ được tiếp xúc với các phương thức khác nhau cho việc di chuyển trên phạm vi cấu trúc của DOM.

|  |
| --- |
| ***EXERCISE 9.2*** — DOM FAMILY RELATIONS |

1. Mở lab09-walkENC02.html trong trình duyệt và mở bảng điều khiển JavaScript.
2. Nhập nội dung sau vào bảng điều khiển: document.getElementById("first")

*Điều này sẽ hiển thị nội dung của phần tử <div> có tên “first”.*

1. Nhập nội dung sau vào bảng điều khiển: document.getElementById("first").firstChil

*Bạn có thể ngạc nhiên với kết quả. Bạn có thể mong đợi đứa trẻ đầu tiên là phần tử <h1>, nhưng thay vào đó, nó là một khối văn bản tương ứng với khoảng trắng (trả về và khoảng trắng / tab) giữa hai phần tử.*

1. Nhập nội dung sau vào bảng điều khiển: document.getElementById("first").firstChild.nextSibling

*Điều này sẽ hiển thị phần tử tiếp theo ở cùng "cấp độ".*

1. Nhập nội dung sau vào bảng điều khiển

. document.getElementById("first").parentNode

1. Enter the following into

lab09-walkthrough02.js and then test.

**var stories = document.getElementById("stories"); for (i=0;i<stories.childNodes.length; i++)**

**{ console.log("child " + i + " =" + stories.childNodes[i] + " nodeType=" + stories.childNodes[i].nodeType); }**

*Bước này thông qua các nút con của phần tử <div> có tên “first”.*

1. 7 Sửa đổi vòng lặp của bạn để thay đổi màu nền của các nút không phải là văn bản của phần tử <div> có tên là First First. Kết quả sẽ trông tương tự như hiển thị trong Figure 9.1.

*Figure*

*9*

*.*

*1*

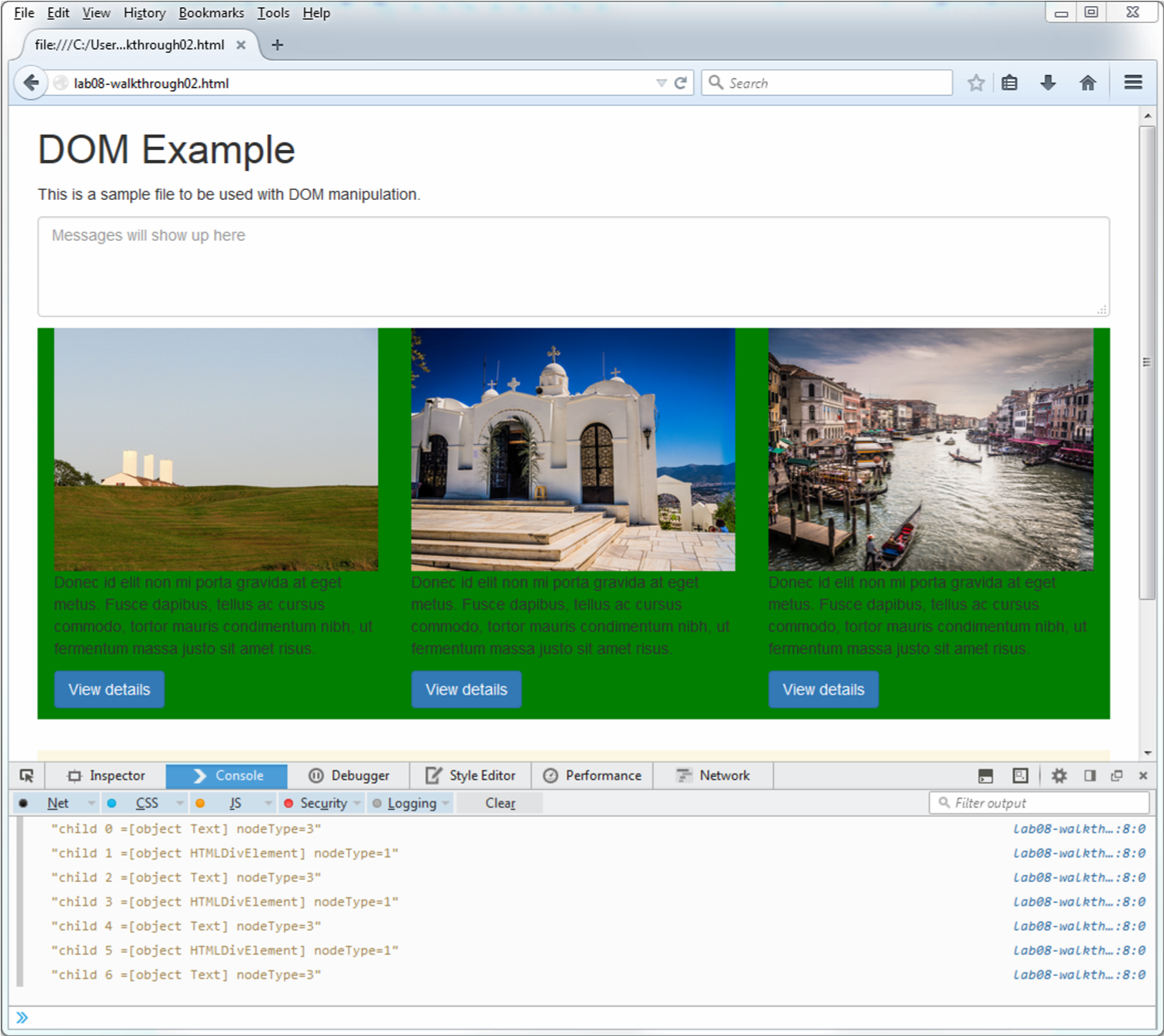
*–*

*Finished*

*Exercise*

*9.*

*2*



***EXERCISE 9.3*** — MODIFYING THE DOM

1. Mở lab09-walkENC03.js và thêm các mục sau.

*/\* mã ở đây \*/*

**var first = document.getElementById("first");**

**var text = document.createTextNode("this is programmatically created"); var p = document.createElement("p");**

**p.appendChild(text); first.appendChild(p);**

*Điều này tạo ra một phần tử <p>, thêm một số văn bản vào nó và sau đó nối phần tử <p> mới vào phần tử <div> với tên trước.*

1. Kiểm tra trên trình duyệt. Sử dụng các công cụ kiểm tra để kiểm tra HTML đã tạo.
2. Thêm dòng này và kiểm tra.

**var icons = document.getElementById("iconList"); icons.removeChild( document.getElementById("deleteThisOne") );**

*Thao tác này sẽ xóa một trong các phần tử <li>.*

**XỬ LÍ SỰ KIỆN**

Một trong những cách sử dụng JavaScript phổ biến nhất là sử dụng nó để đáp ứng các hành động khác nhau do người dùng khởi tạo. Chúng thường được gọi là các sự kiện và JavaScript đáp ứng các sự kiện này được gọi là xử lý sự kiện. Như sách giáo khoa chỉ ra, có một số cách viết trình xử lý sự kiện bằng JavaScript. Trong phòng thí nghiệm này, chúng tôi gắn bó với phương pháp tiếp cận người nghe sự kiện ưa thích.

***EXERCISE 9.4*** — SIMPLE EVENT HANDLING

1. Mởlab09-walkthrough04.html và kiểm tra trên trình duyệt.
2. Mở lab09-walkthrough04.js, và kiểm tra trên trình duyệt.

*/\* mã ở đây \*/* **function simpleHandler(event) { alert("button was clicked");**

**} var btn = document.getElementById("testButton"); btn.addEventListener("click", simpleHandler);**

*Điều này xác định chức năng xử lý sự kiện và gán nó cho sự kiện nhấp chuột của nút. Thay vào đó, khá phổ biến là sử dụng một hàm ẩn danh cho trình xử lý sự kiện, như được hiển thị trong các bước tiếp theo.*

1. Thêm mã sau đây và kiểm tra.

**var img = document.getElementById("mainImage");**

*/\* changes the style of the image when it is moused over \*/* **img.addEventListener("mouseover", function (event) { img.className = "makeItGray"; });**

*This changes the class of the image when the mouse moves over the image. We now need to remove the class styling when the mouse leaves the image.*

1. Thêm mã sau đây và kiểm tra.

*/\* loại bỏ kiểu dáng khi chuột rời khỏi hình ảnh \*/* **img.addEventListener("mouseout", function (event) { img.className = "makeItNormal";**

**});**

1. Thay đổi hàm SimpleHandler () như sau và kiểm tra.

function simpleHandler(event) {

**var content = document.getElementById("content"); if (btn.innerHTML == "Hide") { content.style.display = "none"; btn.innerHTML = "Show";**

**} else {**

**content.style.display = "block"; btn.innerHTML = "Hide";**

**}** }

*Điều này làm cho nút chuyển đổi khả năng hiển thị của đoạn nội dung.*

6 Thay vì chỉ ẩn / hiển thị đoạn văn, chúng tôi sẽ sử dụng các chuyển tiếp CSS3 cũng như chức năng trì hoãn JavaScript. Thay đổi mã của bạn để sau đây và kiểm tra.

function simpleHandler(event) {

var content = document.getElementById("content"); if (btn.innerHTML == "Hide") { btn.innerHTML = "Show";

**content.className = "makeItDisappear";**

*// thay đổi chế độ hiển thị sau độ trễ 1000 mili giây*  **setTimeout(function(){**

**content.style.display = "none";**

**},1000);**

} else {

btn.innerHTML = "Hide"; content.style.display = "block";

*// thay đổi lớp sau khi trì hoãn 500 mili giây*  **setTimeout(function(){**

**content.className = "makeItNormal";**

**},500);**

} }

***EXERCISE 9.5*** — RESPONDING TO LOAD EVENTS

1. Mở lab09-walkENC05.html và kiểm tra trên trình duyệt.
2. Mở lab09-walkENC05.html và kiểm tra trên trình duyệt.

**var divToGet = document.getElementById("div1"); alert(divToGet.innerHTML);**

1. Hiển thị bảng điều khiển JavaScript trong trình duyệt của bạn và sau đó hiển thị Mở lab09walkENC05.js.

*Nếu chúng tôi lưu và làm mới trang của mình, chúng tôi có thể mong đợi cảnh báo nói "Đây là div 1" vì đó là những gì được chứa trong div với id = div1. Tuy nhiên, không có gì được cảnh báo và chúng tôi gặp lỗi (thông báo sẽ khác nhau trong các trình duyệt khác nhau).*

1. 4 Lý do cho lỗi trên là tập lệnh được thực thi trước khi DOM được tải đầy đủ. Để thực thi tập lệnh sau khi DOM được tải, chúng ta phải sử dụng sự kiện onload. Hãy tự thử điều này bằng cách di chuyển mã bên trong trình nghe cho sự kiện onload như sau (và sau đó kiểm tra):

**window.addEventListener("load", function(){**  var divToGet = document.getElementById("div1"); alert(divToGet.innerHTML);

**});**

***EXERCISE 9.6*** — DEBUGGING EVENTS

1. Mở lab09-walkENC05.html và hiển thị bảng điều khiển JavaScript.
2. Trong Chrome, để truy cập trình gỡ lỗi, bạn phải nhấp vào tab Nguồn trong Bảng điều khiển và sau đó mở tệp JavaScript mà bạn muốn gỡ lỗi.

Bây giờ bạn sẽ thêm một điểm dừng vào tập lệnh của mình bằng cách nhấp vào bên trái của dòng chứa cuộc gọi đến alert().

1. 3 Thử làm mới trang. Đường thẳng với điểm dừng sẽ được tô sáng như trong Hình 9.2. Bây giờ bạn có thể kiểm tra trạng thái của các biến cục bộ và toàn cầu.

*Figure*

*9*

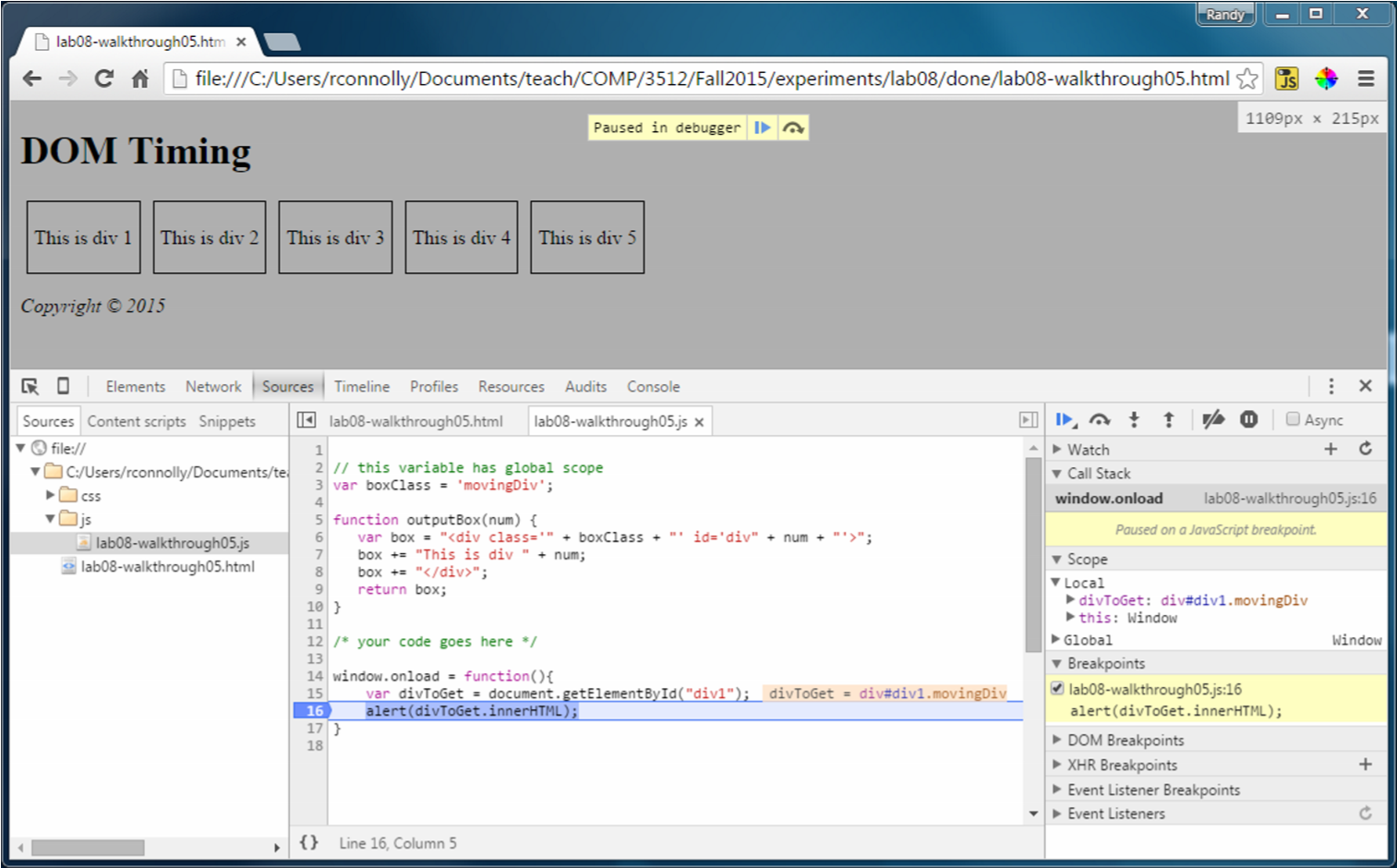
*.*

*2*

*–*

*Chrome*

*Debugging*



1. 4 Trong FireFox, để truy cập trình gỡ lỗi, bạn phải nhấp vào tab Trình gỡ lỗi trong các công cụ dành cho nhà phát triển và sau đó mở tệp JavaScript mà bạn muốn gỡ lỗi.

Bạn có thể thêm một điểm dừng vào tập lệnh của mình bằng cách nhấp vào bên trái của dòng chứa cuộc gọi đến alert().

1. 5 Thử làm mới trang. Đường thẳng với điểm dừng sẽ được tô sáng như trong Hình 9.3. Bây giờ bạn có thể kiểm tra trạng thái của các biến cục bộ và toàn cầu.

*Figure*

*9*

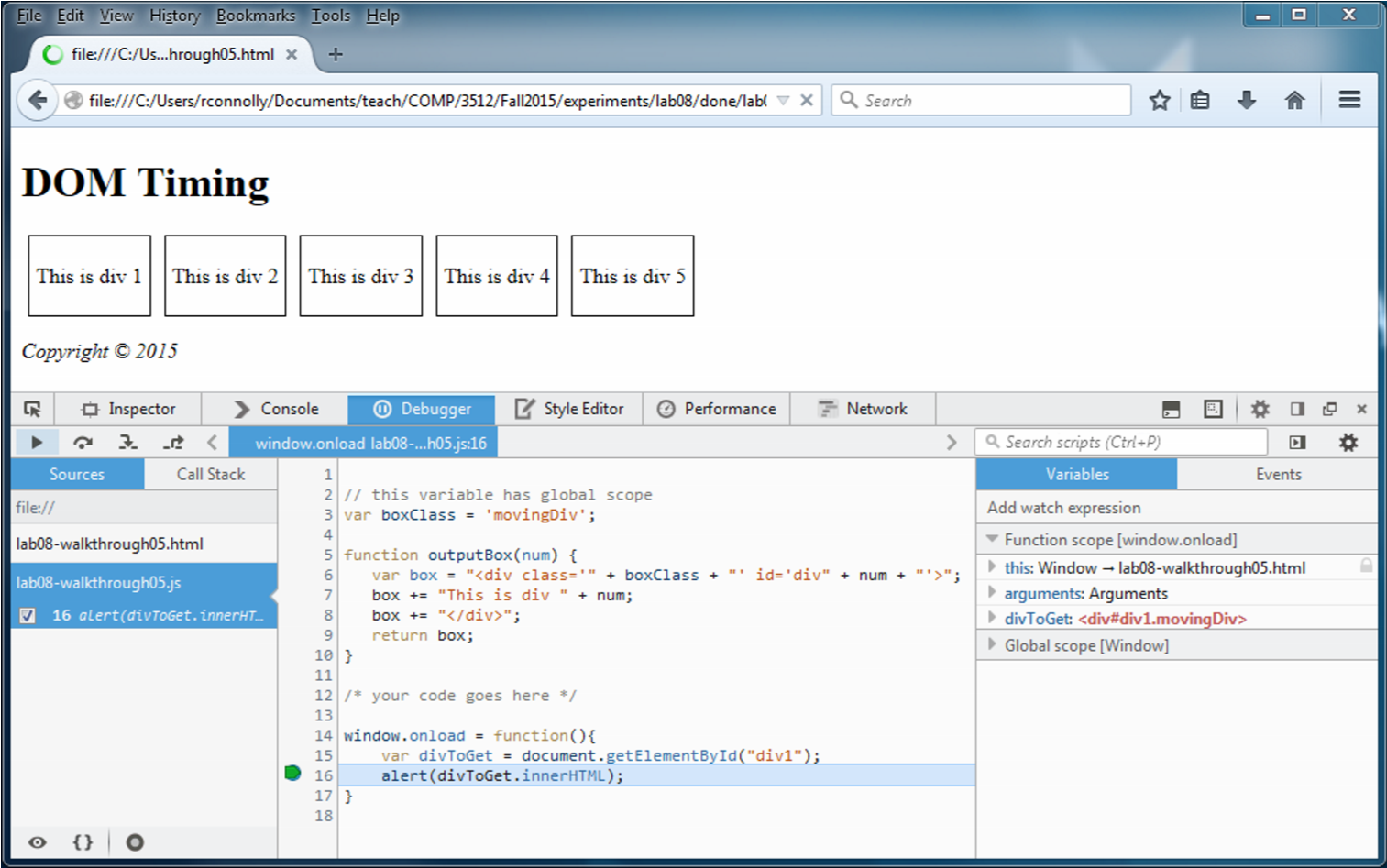
*.*

*3*

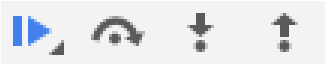
*–*

*FireFox*

*Debugging*



1. 6 Khi bạn đã đặt điểm dừng, bạn có thể tiếp tục thực thi tập lệnh hoặc tiếp tục thực hiện từng dòng một bằng các nút thực hiện:



1. 7 Chỉnh sửa lab09-walkENC05.js và chỉnh sửa mã như sau:

window.addEventListener("load", function(){ **var divs = document.querySelectorAll(".movingDiv"); for (i=0; i<divs.length; i++)**

**{**

**divs[i].addEventListener("mouseover", function (e) { alert("triggered by " + e.target.id);**

**});**

**}**

});

1. 8 Kiểm tra trong trình duyệt bằng cách di chuyển chuột qua các hộp <div> khác nhau.
2. 9 Sử dụng trình gỡ lỗi và đặt điểm theo dõi trước cuộc gọi cảnh báo (). Kiểm tra nội dung của biến e và thuộc tính e.target.

***EXERCISE 9.7*** — WORKING WITH FORMS

1. Mở lab09-walk phiên07.html và kiểm tra trong trình duyệt.
2. 2 Thêm chức năng sau vào lab09-walk phiên07.js:

**function setBackground(e) { if (e.type == "focus") {**

**e.target.style.backgroundColor = "#FFE393"; }**

**else if (e.type == "blur") {**

**e.target.style.backgroundColor = "white";**

**}**

**}**

*Hàm này sẽ được gọi mỗi khi các sự kiện tiêu điểm hoặc mờ được kích hoạt trong một trong các phần tử đầu vào dạng biểu mẫu của chúng tôi.*

1. Thêm mã sau đây:

**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);**

**}**

**});**

*Điều này gán một chức năng ẩn danh cho sự kiện tải của trình duyệt. Hàm gán setBackground() đến các sự kiện làm mờ và tiêu điểm của các yếu tố <input> có liên quan. 4 Kiểm tra trên trình duyệt. Tab giữa các yếu tố khác nhau*.

|  |
| --- |
| ***EXERCISE 9.8*** — FORM VALIDATION |

1 Kiểm tra lab09-walk phiên08.html và sau đó thêm mã sau vào lab09walkENC08.js.

*/\* chịu trách nhiệm thiết lập trình lắng nghe sự kiện trên trang \*/* **function init() {**

**document.getElementById("sampleForm").addEventListener("submit", checkForEmptyFields); }**

*/\* khởi tạo trình xử lý khi trang đã sẵn sàng \*/* **window.addEventListener("load", init);**

*/\* đảm bảo các trường mẫu không trống\*/*

**function checkForEmptyFields(e) {**

**var cssSelector = "input[type=text],input[type=password]"; var fields = document.querySelectorAll(cssSelector);**

*// vòng lặp thông qua các yếu tố đầu vào tìm kiếm các giá trị trống*

**var fieldList = [];**

**for (i=0; i<fields.length; i++) {**

**if (fields[i].value == null || fields[i].value == "") {**  *// vì một trường trống, ngăn chặn việc gửi biểu mẫu*

**e.preventDefault(); fieldList.push(fields[i]); }**

**}**

*// bây giờ thiết lập thông báo lỗi*

**var msg = "The following fields can't be empty: "; if (fieldList.length > 0) { for (i=0; i<fieldList.length; i++) { msg += fieldList[i].id + ",";**

**} alert(msg);**

**}**

**}**

1. Kiểm tra trên trình duyệt. Thử nghiệm bằng cách điền vào các lĩnh vực khác nhau.
2. Sửa đổi kiểm tra EmptyFields()chức năng bằng cách thêm vào sau đây:

function checkForEmptyFields(e) {  *// hide the error message element*

**var errorArea = document.getElementById("errors"); errorArea.className = "hidden";**

var cssSelector = "input[type=text],input[type=password]"; var fields = document.querySelectorAll(cssSelector);

*// vòng lặp thông qua các yếu tố đầu vào tìm kiếm các giá trị trống*  var fieldList = []; for (i=0; i<fields.length; i++) {

if (fields[i].value == null || fields[i].value == "") { // since a field is empty prevent the form submission

e.preventDefault(); fieldList.push(fields[i]);

}

}

*// bây giờ thiết lập thông báo lỗi* var msg = "The following fields can't be empty: "; if (fieldList.length > 0) { for (i=0; i<fieldList.length; i++) { msg += fieldList[i].id + ",";

}

**errorArea.innerHTML = "<p>" + msg + "</p>"; errorArea.className = "visible";**

}

}

*Instead of displaying the error message inside an alert box this places it within a <div> element. The result should look similar to that shown in Figure 9.4.*

*Figure*

*9*

*.*

*4*

*–*

*Finished*

*Exercise*

*9.*

*8*

