Basic Web Concepts

Computer Security CSE 481

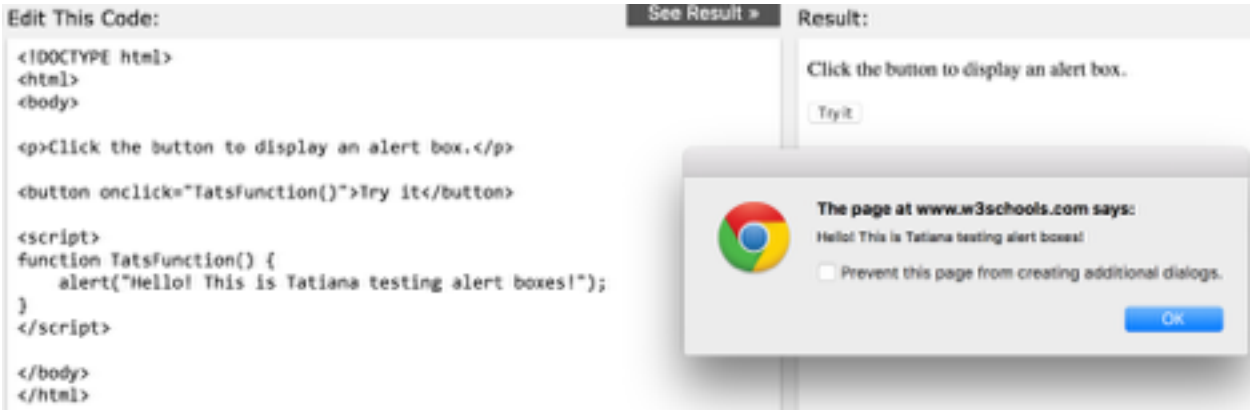
Tatiana Ensslin - October 18, 2015

Introduction

The goal of this lab is to become familiar with basic javascript so that our future javascript worm lab will be much easier to complete and understand.

1. BOM (Browser Object Model)

Alert Information: Alerts are an important part of javascript, for it allows the user to be notified or any changes, and also allows them to accept or decline them. Below is a screenshot of javascript code alert information and its alert that is created when the button “try me” is clicked.



Alerts are important and for this reason it is important for one to get familiar with the way to show the user a message in the browser. To create an alert in javascript one uses the alert() method within their function, and create a button which onclick runs the function which contains alert. In the example above, that function is TatsFunction.

2. DOM (Document Object Model)

This section of the lab goes over DOM methods, documents, elements, and events. The overall goal is to understand the DOM structure and operate on DOM objects successfully. For this section, we will change HTML elements (using getElementById and InnerHTML). Although these two tasks seem similar, they are a bit different. Seen below, innerHTML changes the content of the paragraph <p> element to new text. This is because the getElementById.innerHTML = new content when added into the script.

```
<!DOCTYPE html>
<html>
<body>

<p id="demo">This is tatiana's new text changed by the innerHTML property.
</p>

<script>
// Add code here
</script>

</body>
</html>
```

This is tatiana's new text changed by the innerHTML property.

Note above that the script has not yet been encoded. This is when getElementById is used. This is the easiest way to find an HTML element in the DOM. Above, the <p> id is "demo". When we use this method, it overrides the <p> element because it uses the id rather than just overriding the <p> elements itself which InnerHTML allows. InnerHTML builds an HTML string based on the DOM the element contains (seen below).

Edit This Code: See Result »

```
<!DOCTYPE html>
<html>
<body>

<p id="demo">This is tatiana's new text that you will not see because it is
changed by the innerHTML property.</p>

<script>
document.getElementById("demo").innerHTML = "This is the new text over ridded
by getElementById!";
</script>

</body>
</html>
```

Result:

This is the new text over ridded by getElementById!

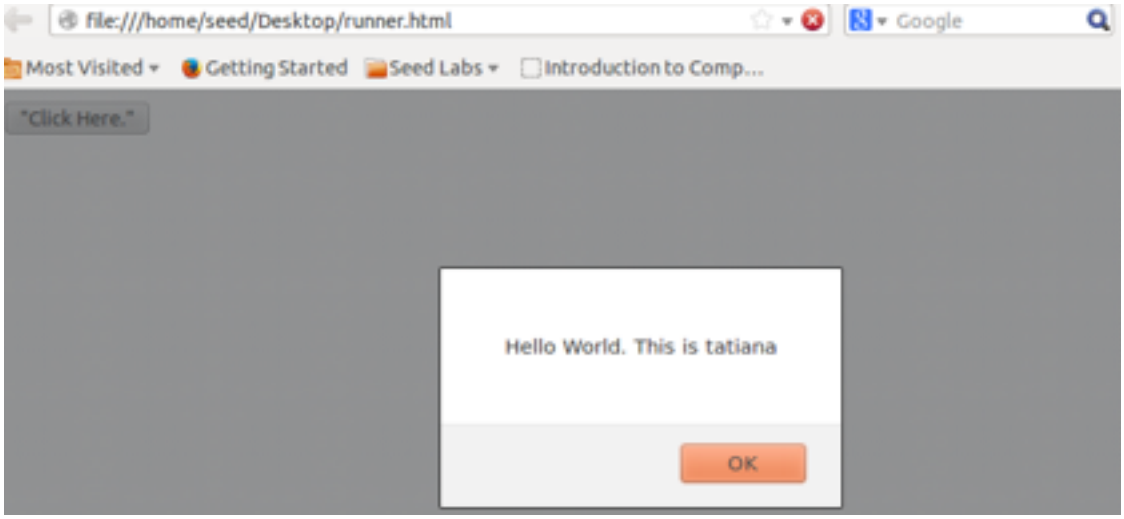
However, for getElementById.value, it gives you the current set value of a form element, rather than building a string like innerHTML does. See with the example below.

```
<!DOCTYPE html>
<html>
<body>

<input type="text" id="myText" value="Hello">

<script>
document.getElementById("myText").value = "Current set value of myText ";
</script>

</body>
</html>
```

Current set value of myT

3. HTML5 Tags

For this section we will show that we understand basic HTML5 tags by creating a javascript file in /var/www/test.js. My file contains a button and an alert seen below:



This is my HTML file runner.html. To its right is the test6.js file, which contains the alert function.

```
<!DOCTYPE html>
<html>
<head>
<button type="button" onclick"=popup()">"Click Here."</button>
<script src="/var/www/test6.js">
</script>
</head>
</html>
```

```
function popup{
alert("Hello World. This is tatiana");
}
```

4. JS Objects

In this section we learn how to operate on an object from the Javascript. This includes changing links and using scripts to change image. Seen below is changing links.
For this we use a tag. The allows the coder to specify a hyperlink to another page, and then allows the user to title it with whatever they please. Next, we

```
<!DOCTYPE html>
<html>
<body>

<img id="image" src="smiley.gif" width="304" height="228">

<script>
// Add code here
</script>

</body>
</html>
```

change an image from a smily face to a cliff's edge. Below we see an image tag with the width and height specified for the picture. The pictures title is "smiley.gif".
As you can see from the picture, the script has not yet been added. One the script is added using a document.getElementById("image").src element, the current picture will be overwritten with the new picture. Seen in the example below.

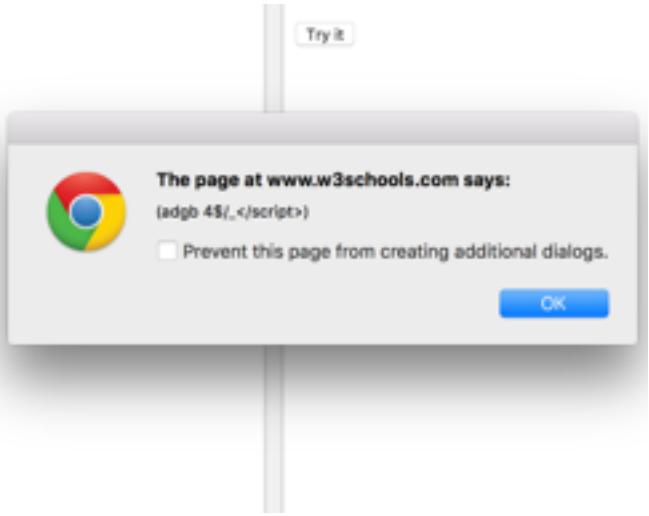
```
<!DOCTYPE html>
<html>
<body>

<img id="image" src="smiley.gif" width="304" height="228">

<script>
document.getElementById("image").src = "pic_mountain.jpg";
</script>

</body>
</html>
```

In this case, the getElementById overwrites the id "image", replacing the mountain picture over the smiley.

5. JS Strings

In this section we will get familiar with the JS String operation, and alert the string (adgb 4$/_</script>) on the screen.

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>
<button onclick="myFunction()">Try it</button>

<script>
var text1 = "(adgb 4$/_<";
var text2 = "/script>)";
text3 = text1.concat("",text2);

function myFunction() {
    alert(text3);
}
</script>


</body>
</html>
```



This is tricky because the string has an end tag located within it, which in a normal string ends the script tag before the alert can go off. In order to get this alert to pop up, we trick the code by splitting the string into two, and then concatenating it with the

concat() function. We then print out the concatenated version, seen above. This allows us to give the alert successfully.