

Introduction to JavaScript Training

Navigator, History, and Location Objects

Lesson 1, Activity 2: The navigator Object

In the days of Netscape 4, it was common practice to branch code based on your user's browser. This is almost always a bad idea. It's better practice to check for and respond to feature support than to base your code on the type or version of the user's browser. That said, in some cases, it can be useful to be able to know what browser type and version your visitor is using. Consider, for example, a support form. Many support forms ask users what browser and operating system they're using as this can be useful information for debugging the problem. But users sometimes do not provide accurate information. And sometimes the support tech needs additional information like whether cookies are enabled. The `window.navigator` object to the rescue!

The following demo shows some of the `navigator` properties and their values. Open it in your browser to see what it reports.

The properties specified in the [HTML5 specification](#) are highlighted in yellow.

Code Sample:

[NavigatorHistoryLocation/Demos/navigator.html](#)

```

---- CODE OMITTED ----
<ol>
<li>appName: <strong><script type="text/javascript">document.write(navigator.appCodeName);</script></strong></li>
<li>appMinorVersion: <strong><script type="text/javascript">document.write(navigator.appMinorVersion);</script></strong></li>
<li class="html5">appName: <strong><script type="text/javascript">document.write(navigator.appName);</script></strong></li>
<li class="html5">appVersion: <strong><script type="text/javascript">document.write(navigator.appVersion);</script></strong></li>
<li>browserLanguage: <strong><script type="text/javascript">document.write(navigator.browserLanguage);</script></strong></li>
<li>buildID: <strong><script type="text/javascript">document.write(navigator.buildID);</script></strong></li>
<li>cookieEnabled: <strong><script type="text/javascript">document.write(navigator.cookieEnabled);</script></strong></li>
<li>cpuClass: <strong><script type="text/javascript">document.write(navigator.cpuClass);</script></strong></li>
<li>language: <strong><script type="text/javascript">document.write(navigator.language);</script></strong></li>
<li>mimeTypes: <strong><script type="text/javascript">document.write(navigator.mimeTypes);</script></strong></li>
<li>onLine: <strong><script type="text/javascript">document.write(navigator.onLine);</script></strong></li>
<li>oscpu: <strong><script type="text/javascript">document.write(navigator.oscpu);</script></strong></li>
<li class="html5">platform: <strong><script type="text/javascript">document.write(navigator.platform);</script></strong></li>
<li>plugins: <strong><script type="text/javascript">document.write(navigator.plugins);</script></strong></li>
<li>product: <strong><script type="text/javascript">document.write(navigator.product);</script></strong></li>
<li>productSub: <strong><script type="text/javascript">document.write(navigator.productSub);</script></strong></li>
<li>securityPolicy: <strong><script type="text/javascript">document.write(navigator.securityPolicy);</script></strong></li>
<li>systemLanguage: <strong><script type="text/javascript">document.write(navigator.systemLanguage);</script></strong></li>
<li class="html5">userAgent: <strong><script type="text/javascript">document.write(navigator.userAgent);</script></strong></li>
<li>userLanguage: <strong><script type="text/javascript">document.write(navigator.userLanguage);</script></strong></li>
<li>userProfile: <strong><script type="text/javascript">document.write(navigator.userProfile);</script></strong></li>
<li>vendor: <strong><script type="text/javascript">document.write(navigator.vendor);</script></strong></li>
<li>vendorSub: <strong><script type="text/javascript">document.write(navigator.vendorSub);</script></strong></li>
</ol>
</body>
</html>

```

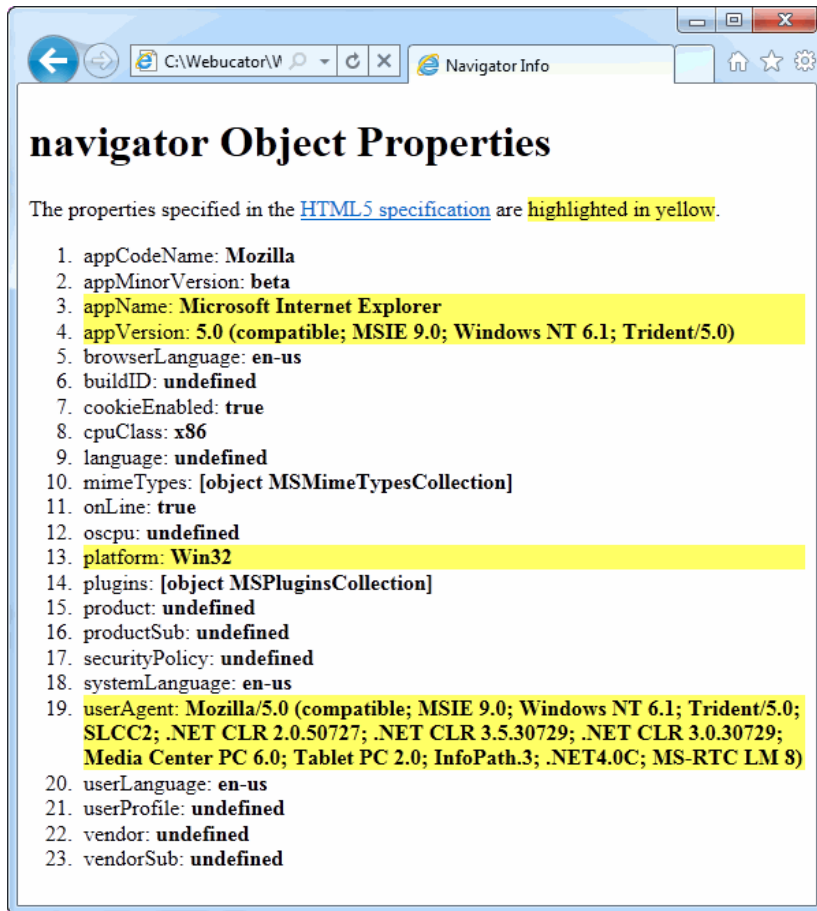
Below we show the results in Google Chrome 9 (beta) and Internet Explorer 9 (beta):

The screenshot shows a web browser window with a single tab titled 'Navigator Info'. The address bar contains 'navigator.html'. The main content area has the heading 'navigator Object Properties'. Below the heading, a paragraph states: 'The properties specified in the [HTML5 specification](#) are highlighted in yellow.' This is followed by a numbered list of 23 properties of the navigator object. Properties 3, 4, 13, 19, and 22 are highlighted in yellow in the original image.

navigator Object Properties

The properties specified in the [HTML5 specification](#) are highlighted in yellow.

1. `appName`: **Mozilla**
2. `appMinorVersion`: **undefined**
3. `appName`: **Netscape**
4. `appVersion`: **5.0 (Windows; U; Windows NT 6.1; en-US) AppleWebKit/534.13 (KHTML, like Gecko) Chrome/9.0.597.86 Safari/534.13**
5. `browserLanguage`: **undefined**
6. `buildID`: **undefined**
7. `cookieEnabled`: **true**
8. `cpuClass`: **undefined**
9. `language`: **en-US**
10. `mimeType`: **[object DOMMimeTypeArray]**
11. `onLine`: **true**
12. `oscpu`: **undefined**
13. `platform`: **Win32**
14. `plugins`: **[object DOMPluginArray]**
15. `product`: **Gecko**
16. `productSub`: **20030107**
17. `securityPolicy`: **undefined**
18. `systemLanguage`: **undefined**
19. `userAgent`: **Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US) AppleWebKit/534.13 (KHTML, like Gecko) Chrome/9.0.597.86 Safari/534.13**
20. `userLanguage`: **undefined**
21. `userProfile`: **undefined**
22. `vendor`: **Google Inc.**
23. `vendorSub`:



Checking for Disabled Features

You may have noticed the `cookieEnabled` property in the screenshots above. In some cases, a browser may support a feature, but the user might have disabled it. This is most common with cookies, which users sometimes disable for (perceived) security reasons. The following demo shows how to check whether cookies are disabled:

Code Sample:

<NavigatorHistoryLocation/Demos/cookie-check.html>

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>Cookie Check</title>
</head>
<body>
<script>
if (!navigator.cookieEnabled) {
  document.write("<h1>Warning: Cookies Required</h1>");
  document.write("<p>Please turn your cookies on and refresh the page.</p>");
  document.bgColor="red";
}
</script>
<p>Rest of page goes here...</p>
</body>
</html>
```

Note that [Google Chrome has a bug](#) that causes it to report false positives for `navigator.cookieEnabled`.

Lesson 1, Activity 3: Feature Detection

Although browsers have come a long way in the past several years, they unfortunately do not all support the W3C specifications to the same degree. This is particularly true with the introduction of HTML5. It is often necessary to branch the code based on the browser's support of a feature. A case in point is the HTML5 canvas element, which is used to create drawings natively in the browser. Take a look at the following code:

Code Sample:

[NavigatorHistoryLocation/Demos/canvas.html](#)

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>Canvas</title>
<script>
function drawPath() {
  var canvas=document.getElementById("my-canvas");
  if (canvas.getContext) {
    alert("Canvas Supported.");
    //create your awesome drawing here
  } else {
    alert("Canvas Not Supported.");
  }
}
</script>
</head>
<body>
<canvas id="my-canvas" height="200" width="200">Your browser doesn't support canvas.</canvas>
<button onclick="drawPath();" >Draw</button>
</body>
</html>
```

If the browser properly supports canvas, it will recognize the `getContext` method of the canvas object and execute the code in the `if` condition.

The major advantage of checking for the feature rather than checking for the type or version of the browser is that you don't have to keep updating your code to account for changes in browser versions and support.

Lesson 1, Activity 5: history Object

Like `navigator`, the `history` object is a property of the `window` object. For security and privacy reasons, the information you can get about the user's session history is limited to the number of entries, which you get by reading the `history.length` property. That's generally not so useful. However, there are a few methods of the `history` object which you may indeed find useful:

history Methods

Method	Description
<code>go(int)</code>	Advances or moves back (negative int) through the history.
<code>back()</code>	The same as <code>go(-1)</code>
<code>forward()</code>	The same as <code>go(1)</code>

If any of the above methods fail, no error is reported. In other words, if `history.back()` is called on a page that has no history, nothing at all happens.

Code Sample:

[NavigatorHistoryLocation/Demos/history-1.html](#)



Try this:

1. Open [NavigatorHistoryLocation/Demos/history-1.html](#) in your browser.
2. Click the **Page 2** link.
3. Click the **Page 3** link.
4. Now click the **Back** and **Forward** buttons several times to navigate back and forth through the pages.

If you took the above steps exactly, it should have worked as expected, but now try this:

1. Open [NavigatorHistoryLocation/Demos/history-1.html](#) in your browser.
2. Click the **Page 3** link.
3. Now click the **Back** button.

Notice it goes back from **Page 3** to **Page 1**, which doesn't seem intuitive from the user's point of view. The problem is that the JavaScript isn't aware of the "appropriate" order of the pages as laid out in the **Table of Contents**. The JavaScript is just using the `history` object to simulate the browser's **Back** and **Forward** buttons.

As such, this makes using the `history` object in this way relatively useless and not recommended. In the exercise later in this lesson, we'll look at a more useful way of using the `history` object.

Lesson 1, Activity 6: location Object

The `location` object is also a property of the `window` object, therefore you can call the methods directly in your JavaScript like a global function. Here are a few properties and methods of the `location` object.

location Properties

Property	Description
<code>href</code>	Returns the full location of the page. The value can be set to change the page.
<code>protocol</code>	Returns the protocol used to deliver the page (e.g., "http").
<code>host</code>	Returns the host of the page (e.g., "www.webucator.com"). It will include the port if included.
<code>hostname</code>	Returns the host of the page (e.g., "www.webucator.com"), but not the port.
<code>port</code>	Returns the port of the host if included (e.g., "80").
<code>pathname</code>	Returns the path after the domain name (e.g., "/webdesign/javascript.cfm" for http://www.webucator.com/webdesign/javascript.cfm).
<code>search</code>	Returns the querystring, including the question mark (e.g., "?q=javascript+training" for http://www.google.com/search?q=javascript+training).
<code>hash</code>	Returns the hash, including the hash mark (e.g., "#dom-location-hash" for http://www.w3.org/TR/html5/history.html#dom-location-hash).

location Methods

Method	Description
<code>assign(url)</code>	Navigates to the <code>url</code> argument. Same as <code>location.href=url;</code> .
<code>replace(url)</code>	Replaces the current page in the history stack with the <code>url</code> argument.
<code>reload()</code>	Reloads the page.

The properties and methods above are demonstrated in the following code sample:

Code Sample:

NavigatorHistoryLocation/Demos/location.html

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>Location Info</title>
</head>
<body>
<h1>location Object Properties</h1>
<ol>
<li><a href="?firstName=Nat">Add Search</a></li>
<li><a href="#props">Add Hash</a></li>
<li><a href="javascript:location.replace('#props')">Add Hash with <code>replace()</code></a></li>
<li><a href="javascript:location.assign('#props')">Add Hash with <code>assign()</code></a></li>
<li><a href="javascript:location.reload()">Reload</a></li>
<li><a href="location.html">Start Over</a></li>
</ol>

<ol id="props">
<li>href: <strong><script type="text/javascript">document.write(location.href);</script></strong></li>
<li>protocol: <strong><script type="text/javascript">document.write(location.protocol);</script></strong></li>
<li>host: <strong><script type="text/javascript">document.write(location.host);</script></strong></li>
<li>hostname: <strong><script type="text/javascript">document.write(location.hostname);</script></strong></li>
<li>port: <strong><script type="text/javascript">document.write(location.port);</script></strong></li>
<li>pathname: <strong><script type="text/javascript">document.write(location.pathname);</script></strong></li>
<li>search: <strong><script type="text/javascript">document.write(location.search);</script></strong></li>
<li>hash: <strong><script type="text/javascript">document.write(location.hash);</script></strong></li>
<li>history.length: <strong><script type="text/javascript">document.write(history.length);</script></strong></li>
</ol>
</body>
</html>
```

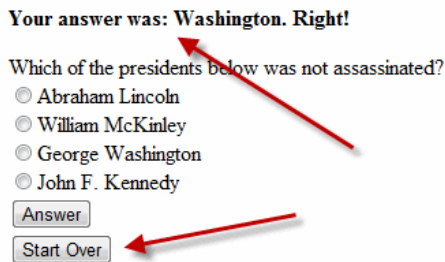
Most of the `location` properties and methods above are only used in advanced applications that require customizing how the site/application responds to the user pressing the **Back** and **Forward** buttons. The exception is the `href` property, which is commonly used to create "button links" as you'll see in the exercise that follows.

Lesson 1, Activity 8: Creating a Simple Quiz

Duration: 15 to 25 minutes.

In this exercise, you will create a simple quiz that asks one question per page and does not allow the user to navigate back to past questions.

1. Open [NavigatorHistoryLocation/Exercises/question1.html](#) for editing.
 1. Add code so that the page tries to navigate forward.
2. Open [NavigatorHistoryLocation/Exercises/question2.html](#) for editing.
 1. Again, add code so that the page tries to navigate forward.
 2. Add code to determine if the user chose the correct answer ("Washington") and report the answer.
 3. Add a button to the end of the form that, when clicked, takes the user back to [start-quiz.html](#). If the user answers correctly, the page should look like this:



3. Make the same modifications to [question3.html](#) and [end-quiz.html](#).
4. Test your solution in a browser by opening [NavigatorHistoryLocation/Exercises/start-quiz.html](#) and working through the quiz. You should not be able to go back to a previous question without starting the whole quiz over.

Solution:

[NavigatorHistoryLocation/Solutions/question1.html](#)

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>History Test: Question 1</title>
<script type="text/javascript">
  history.go(1);
</script>
</head>
<body>
<form action="question2.html">
  Who was the first president of the United States?<br>
  <input type="radio" name="answer" value="Lincoln">Abraham Lincoln<br>
  <input type="radio" name="answer" value="McKinley">William McKinley<br>
  <input type="radio" name="answer" value="Washington">George Washington<br>
  <input type="radio" name="answer" value="Kennedy">John F. Kennedy<br>
  <input type="submit" value="Answer">
</form>
</body>
</html>
```

Solution:

[NavigatorHistoryLocation/Solutions/question2.html](#)

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>History Test: Question 2</title>
<script type="text/javascript">
  history.go(1);
  var s = location.search;
  var start = s.indexOf("=")+1;
  var answer = s.substring(start);
  var response = (answer == "Washington") ? "Right!" : "Wrong!";
</script>
</head>
<body>
<p style="font-weight:bold;">
```

```

<script type="text/javascript">
  document.write("Your answer was: " + answer + ". " + response);
</script>
</p>
<form action="question3.html">
  Which of the presidents below was not assassinated?<br>
  <input type="radio" name="answer" value="Lincoln">Abraham Lincoln<br>
  <input type="radio" name="answer" value="McKinley">William McKinley<br>
  <input type="radio" name="answer" value="Washington">George Washington<br>
  <input type="radio" name="answer" value="Kennedy">John F. Kennedy<br>
  <input type="submit" value="Answer"><br>
  <input type="button" onclick="location.href='start-quiz.html';" value="Start Over">
</form>
</body>
</html>

```

Solution:

[NavigatorHistoryLocation/Solutions/question3.html](#)

```

<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>History Test: Question 3</title>
<script type="text/javascript">
  history.go(1);
  var s = location.search;
  var start = s.indexOf("=")+1;
  var answer = s.substring(start);
  var response = (answer == "Washington") ? "Right!" : "Wrong!";
</script>
</head>
<body>
<p style="font-weight:bold;">
<script type="text/javascript">
  document.write("Your answer was: " + answer + ". " + response);
</script>
</p>
<form action="end-quiz.html">
  Which president was known as "His Excellency"?<br>
  <input type="radio" name="answer" value="Lincoln">Abraham Lincoln<br>
  <input type="radio" name="answer" value="McKinley">William McKinley<br>
  <input type="radio" name="answer" value="Washington">George Washington<br>
  <input type="radio" name="answer" value="Kennedy">John F. Kennedy<br>
  <input type="submit" value="Answer"><br>
  <input type="button" onclick="location.href='start-quiz.html';" value="Start Over">
</form>
</body>
</html>

```

Solution:

[NavigatorHistoryLocation/Solutions/end-quiz.html](#)

```

<!DOCTYPE HTML>
<html>
<head>
<meta charset="UTF-8">
<title>History Test: End</title>
<script type="text/javascript">
  var s = location.search;
  var start = s.indexOf("=")+1;
  var answer = s.substring(start);
  var response = (answer == "Washington") ? "Right!" : "Wrong!";
  history.go(1);
</script>
</head>
<body>
<p style="font-weight:bold;">
<script type="text/javascript">
  document.write("Your answer was: " + answer + ". " + response);
</script>
</p>
<p>You're done.</p>
<input type="button" onclick="location.href='start-quiz.html';" value="Start Over">

```

```
</body>  
</html>
```