Chapter 6:
Navigation and javascript framework

Status bar in javascript:

The window.statusbar property returns a Statusbar object representing the status bar of browser. It appears at the bottom of browser. However, it is almost impossible for you to interact with the Statusbar via Javascript

```
window.statusbar
// Or simple:
Statusbar
```

The tendency of modern browsers is to make the Viewport window as wide as possible, therefore, they remove other components like Statusbar, or make Menubar smaller

For modern browsers, you see the status bar appear only when a user moves the mouse on the surface of a link.

statusbar.visible

The statusbar.visible property returns true if the status bar is displayed on browser. However, this is unreliable property. You get a true value, which does not mean that you are seeing the status bar.

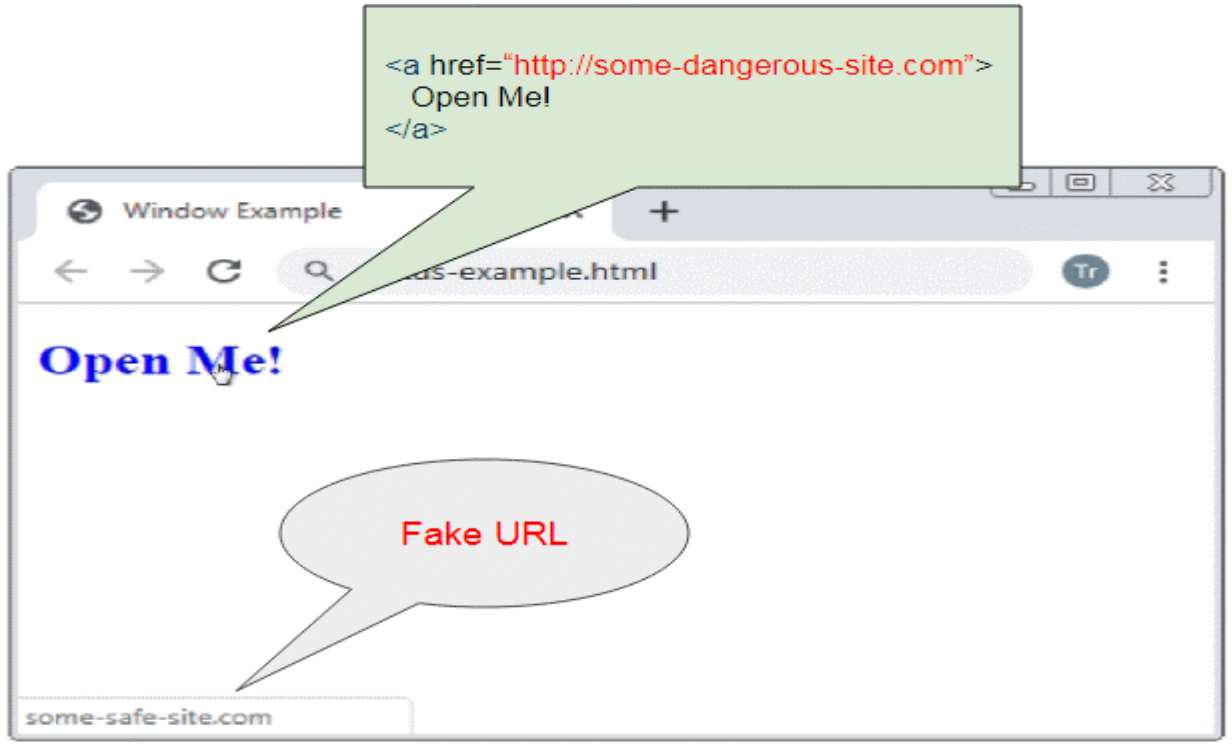
```
<html>
<head>
   <title>Statusbar</title>
   <meta charset="UTF-8">

</head>
<body>
   <h3>statusbar.visible</h3>

   <br/><br/>
   <button onclick="alert(statusbar.visible)">
      alert(statusbar.visible)
   </button>

</body>
</html>
```

The status property of the window object help you establish a content of text shown on he status bar. By default, for security reasons, most browsers disable this feature for JavaScript. However, if an user wants, they are able to enable this feature for JavaScript by entering the "Options" of the browser.



Before clicking on a link, users often move the mouse over the link's surface to preview its address displayed in the status bar, and only click this link when they feel safe. Some websites may take advantage of window.status to display a fake content.

The status property is deprecated.
It should be avoided to prevent RUN-TIME ERRORS in the future.

Status Bar Example:1

```
<html>
   <head>
     <title>JavaScript Status Bar</title></head>
     <body onLoad="window.status='Welcome!';return true">
   </body>
 </html>
```

onLoad tells the browser that as soon as your page finished loading, it will display in your current window's status bar (window.status) the message "Welcome!". The return true is necessary because with out, it won't work.

Status Bar Example: 2

```
<html>
 <head>
  <title>JavaScript Status Bar</title>
 </head>
 <body>
  <a href="http://www.htmlcenter.com"
  onMouseOver="window.status='HTMLcenter';return true"
  onMouseOut="window.status= '';return true">

    HTMLcenter

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

When the user moves his mouse over the link, it will display “HTMLcenter” in the status bar. When he moves his mouse away from the link the status bar will display nothing.

Moving the message along the status bar

```
<html>    <head>    <title>Javascript ScrollText</title>
      <script language="javascript">
         msg="This is an example of scrolling message";
         spacer="..... .....";
         pos=0;
         function ScrollMessage()
         {
            window.status=msg.substring(pos,msg.length)+spacer;
            pos++;
            if(pos>msg.length)
               pos=0;
            window.setTimeout("ScrollMessage()",100);
         }
         ScrollMessage();
      </script>
   </head>
   <body>
      <p>Scrolling Message Example</p>
      <p>Look at the status line at the bottom of the page</p>
   </body>
</html>
```

JavaScript frameworks

- JavaScript frameworks are a collection of libraries containing code written in JavaScript, making life a lot easier for software developers.
- Each JavaScript framework offers pre-built codes for different areas and different purposes in software development, saving time for the developer.

JavaScript frameworks

JavaScript frameworks provide structure for your code, so in that sense, they're pretty essential to your programming. There are many useful JavaScript frameworks that developers regularly use, and we will cover some of these in this article. A JavaScript framework provides a blueprint, so you have a guide to follow rather than having to start coding your website from scratch.

Most frameworks are open-source, meaning they are constantly being improved by the community that uses them, so they are always up to date. They are by no means set in stone either; you are free to tweak the framework you choose to suit your own website or application.

Why use a JavaScript framework?

- Developers created JavaScript frameworks to make life easier for themselves. They allow programmers to use the most up-to-date JavaScript features and tools without having to go through the arduous task of coding them from scratch by themselves.
- These frameworks are templates that provide a foundation for software applications. It collects shared resources like libraries, reference documents, images and more and packages them for developers to use. With these frameworks, programmers can add better functionality and more to a web page and website.

Popular JavaScript Framework libraries

AngularJS

AngularJS is an open-source framework that came into being in October 2010 and is the oldest available. It's a good one to choose when you're thinking about which framework would be best to go for; brilliantly, it is supported by Google! There are even apps built into cars made by General Motors that have been developed in Angular – it has emerged as a bit of a market leader in JavaScript frameworks. Google's lead Angular developer, Igor Minar, believes that Angular is the most widely used framework because it, more so than others, encourages regular updates and developments.

FRONT-END FRAMEWORKS

REACT

React.js is an efficient and flexible JavaScript library for building user interfaces created by Facebook. Technically, React is a JS library, but it is often discussed as a web framework and is compared to any other open source JavaScript framework.

React makes it easy to create interactive user interfaces because it has predictable JavaScript code that is easy to debug. Furthermore, it provides a REACT component system where blocks of JavaScript code can be written once and reused repeatedly in different parts of the application or even other applications.

ANGULAR

AngularJS is a popular enterprise-level JavaScript framework used for developing large and complex business applications. It is an open-source web framework created by Google and supported by both Google and Microsoft.

VUE

Vue.js is a progressive framework for building user interfaces. It is an up-and-coming framework that helps developers in integrating with other libraries and existing projects. It has an ecosystem of libraries that allow developers to create complex and solid single-page applications.

BACK-END FRAMEWORKS

EXPRESS

Express.js is a flexible, minimalistic, lightweight, and well-supported framework for Node.js applications. It is likely the most popular framework for server-side Node.js applications. Express provides a wide range of HTTP utilities, as well as high-performance speed. It is great for developing a simple, single-page application that can handle multiple requests at the same time.

NEXT.JS

Next.js is a minimalistic framework that allows a a JavaScript developer to create a server-side rendering and static web applications using React.js. It is one of the newest and hottest frameworks that takes pride in its ease of use. Many of the problems developers experience while building applications using React.js are solved using Next.js. It has many important features included “out of the box,” and makes development a JavaScript breeze.

Banner Ads

Displaying banners ads is a common practice for showing advertisements on web pages to the visitors. Banners ads are normally created using standard graphic tools such as Photoshop, Paintbrush Pro, and other software. Banner ads can be static or animated. Animated images are animated GIF files or flash movies. Flash movies are created using Macromedia Flash and the browsers must have installed flash plugin to view the movies. On the other hand, you can create some animated effect using JavaScript, like rotating static banner ads at a certain time interval.

Document images

- The images property returns a collection of all <img> elements in a document.
- The images property returns an HTMLCollection.
- The images property is read-only.

HTMLCollection

- An HTMLCollection is an array-like collection (list) of HTML elements.
- The elements in a collection can be accessed by index (starts at 0).
- The length Property returns the number of elements in the collection.

Syntax:

document.images

Properties:

Property	Description
length	The number of <img> elements in the collection.

Methods:

Method	Description
[*index*]	Returns the element with the specified index (starts at 0). Returns null if the index is out of range.
item(*index*)	Returns the element with the specified index (starts at 0). Returns null if the index is out of range.
namedItem(*id*)	Returns the element with the specified id. Returns null if the id does not exist.

Return Value:

Type	Description
Object	An HTMLCollection Object. All <img> elements in the document. The elements are sorted as they appear in the document.

The images Property

```
<html>
<body>

<h1>The Document Object</h1>
<h2>The images Property</h2>

<img src="klematis.jpg" alt="flower" width="150" height="113">
<img src="klematis2.jpg" alt="flower" width="152" height="128">
<img src="smiley.gif" alt="Smiley face" width="42" height="42">

<p>The number of images is:</p>
<p id="demo"></p>

<script>
let numb = document.images.length;
document.getElementById("demo").innerHTML = numb;
</script>

</body>
</html>
```

The Document Object

The images Property

The number of images is:

3

Example:

```
<html>
<body>
<h1>The Document Object</h1>
<h2>The images Property</h2>

<img src="klematis.jpg" alt="flower" width="150" height="120">
<img src="klematis2.jpg" alt="flower" width="150" height="120">

<p>Display the URL of each img element:</p>
<p id="demo"></p>

<script>
const myImages = document.images;
let text = "";
for (let i = 0; i < myImages.length; i++) {
  text +=  myImages[i].src + "<br>";
}
document.getElementById("demo").innerHTML = text;
</script>
</body>
</html>
```

The Document Object

The images Property

Display the URL of each img element:

https://www.w3schools.com/jsref/klematis.jpg
https://www.w3schools.com/jsref/klematis2.jpg

Example:

```
<html>
<body>

<h1>The Document Object</h1>
<h2>The images Property</h2>

<img src="klematis.jpg" alt="flower" width="150" height="120">
<img src="klematis2.jpg" alt="flower" width="150" height="120">

<p>The URL of the first image is:</p>
<p id="demo"></p>

<script>
let src= document.images.item(0).src;
document.getElementById("demo").innerHTML = src;
</script>

</body>
</html>
```

Add a black border to the first <img> element:

```
<html>
<body>

<h1>The Document Object</h1>
<h2>The images Property</h2>

<p>Add a black border to the first image in the document:</p>
<img src="klematis.jpg" alt="flower" width="150" height="120">
<img src="klemais2.jpg" alt="flower" width="150" height="120">

<p id="demo"></p>

<script>
const img = document.images[0];
img.style.border = "10px solid black";
</script>

</body>
</html>
```

The Document Object

The images Property

Add a black border to the first image in the document:

Window setTimeout()

- The setTimeout() method calls a function after a number of milliseconds.
- 1 second = 1000 milliseconds**.**

```
<html>
<body>
<h1>The Window Object</h1>
<h2>The setTimeout() Method</h2>

<p>Wait 5 seconds for the greeting:</p>

<h2 id="demo"></h2>

<script>
const myTimeout = setTimeout(myGreeting, 5000);

function myGreeting() {
  document.getElementById("demo").innerHTML = "Happy Birthday!"
}
</script>
</body>
</html>
```

The Window Object
The setTimeout() Method
Wait 5 seconds for the greeting:

The Window Object
The setTimeout() Method
Wait 5 seconds for the greeting:
Happy Birthday!

Creating Rotating Banner Ads

Rotating banners ads comprises several banner images that constantly rotate on a webpage at a fix time interval. You can create these banner images using standard graphics tools.

```
<html>
<head>
<script language="Javascript">
MyBanners=new Array('banner1.jpg','banner2.jpg','banner3.jpg','banner4.jpg')
banner=0
function ShowBanners()
{ if (document.images)
{ banner++
if (banner==MyBanners.length) {
banner=0}
document.ChangeBanner.src=MyBanners[banner]
setTimeout("ShowBanners()",5000)
}}
</script>
<body onload="ShowBanners()">
<center>
<img src="banner1.jpg" width="900" height="120" name="ChangeBanner"/>
</center>
</body>
</html>
```

Creating Rotating Banner Ads with URL Links

Creating rotating banner images will provide the visitor to your webpage with some basic information. However, if you want the visitor to get more information by clicking on the banner images, you need to create rotating banner ads that contain URL links.

```
<html>
<head>
<script language="Javascript">
MyBanners=new Array('1.jpg','2.jpg','3.jpg')
MyBannerLinks=new Array('http://www.a.net/','http://www.b.com/','http://c.com/')
```

```
 banner=0
function ShowLinks()
{
        document.location.href="http://www."+ MyBannerLinks[banner]
}
function ShowBanners()
{
        if (document.images)
        {
                banner++
        if (banner==MyBanners.length)
        {
                banner=0
        }
        document.ChangeBanner.src=MyBanners[banner]
        setTimeout("ShowBanners()",5000)
        }
}
</script>
<body onload="ShowBanners()">
<center>
<a href="javascript: ShowLinks()">
<img src="1.jpg" width="900" height="120" name="ChangeBanner"/></a>
</center></body></html>
```

Slide Show

The JavaScript code for the slideshow is almost similar to the JavaScript code of the rotating banners but it gives control to the user to choose the banner ads he or she wants to see by clicking on the forward and backward buttons.

To create the JavaScript slideshow, first of all, you need to create a few banner images using some graphics tools, or you can snap some photos with your digital camera or your smartphone.

```
<html ><head>
<script language="Javascript">
MySlides=new Array('1.jpg','2.jpg','3.jpg')
Slide=0
function ShowSlides(SlideNumber)
{
Slide=Slide+SlideNumber ;
if (Slide>MySlides.length-1)
{
Slide=0
}
if (Slide<0)
{
Slide=MySlides.length-1;
}
document.DisplaySlide.src=MySlides[Slide];
}
 </script>
```

```
</head>
<body>
<P align="center"><img src="1.jpg" name="DisplaySlide" width="900″ height="120″ /><p>
<center>
<table border=0>
<tr>
<td align=center>
<input type="button" value="Back" onclick="ShowSlides(-1)">
<input type="button" value="Forward" onclick="ShowSlides(1)">
</td>
</tr>
</table>
</center>
</body>
</html>
```