Scripting the DOM

Terry Marris September 2011

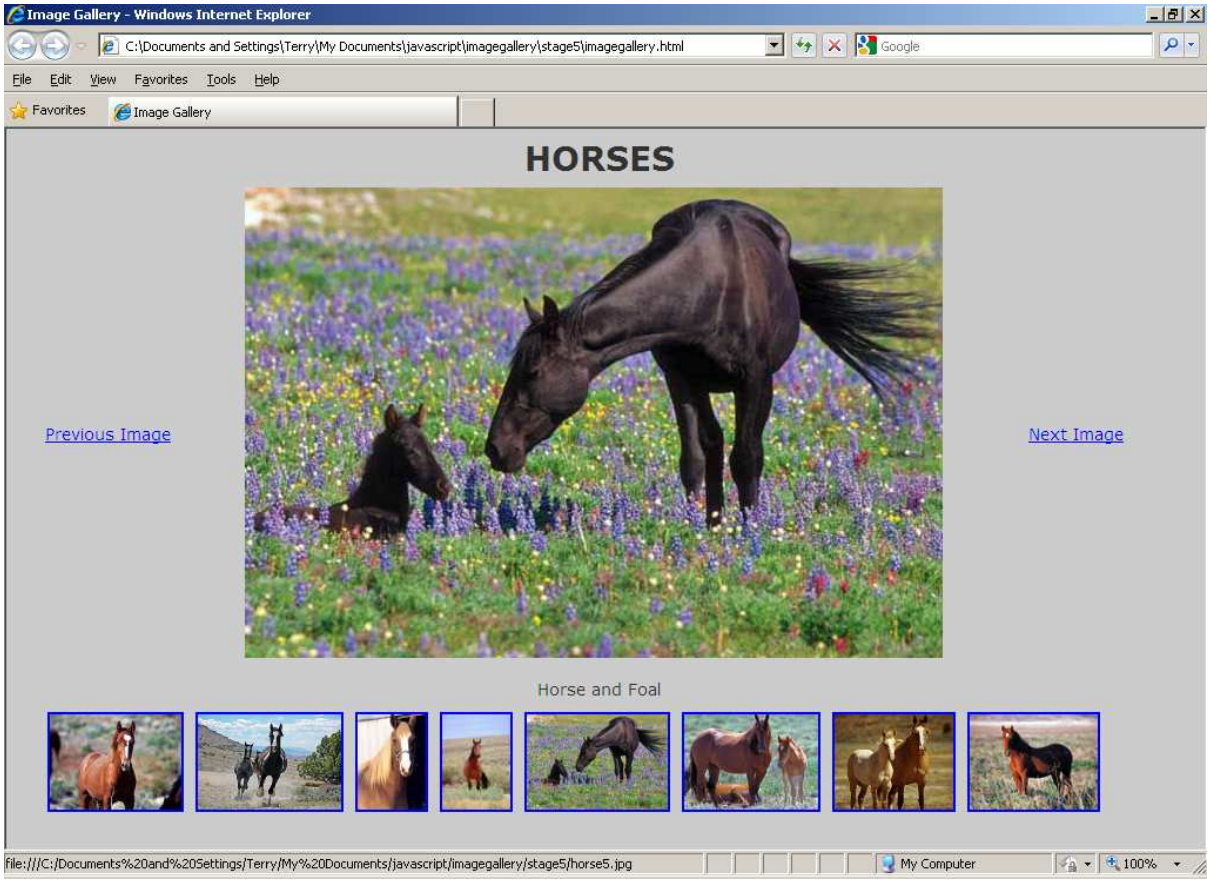
3 Image Gallery

In Chapter 2 we looked at four DOM methods:

- *getElementById()*
- *getElementsByTagName()*
- *getAttribute()*
- *setAttribute()*

Now we see how they can be used to implement a slide show. And we introduce child nodes and node types.

We have a set of thumbnails at the bottom of the screen. Clicking on a thumbnail results in the chosen image being shown enlarged along with a descriptive caption.



In addition, clicking on a link, *Previous Image* or *Next Image*, results in the previous or next image being displayed.

The images were downloaded from *http://geekphilosohper.com*, and were cropped and resized with GIMP (the GNU image manipulation program). Each image was set to 400px high, and thumbnails of each image were set to 81px high.

3.1 HTML

When a thumbnail is clicked, we want its image to appear enlarged. We name the main enlarged image *placeholder*. Initially, we place *horse1.jpg* in this location.

```
<div id="mainImage">
<img id="placeholder" src="horse1.jpg" alt="Image Gallery">
</div>
```

We need a descriptive title below the main image.

```
<div id="imageTitle">
<p id="description">Wild Horse</p>
</div>
```

And then comes our gallery of thumbnails, an unordered list with the list elements shown in a horizontal line.

```
<ul id="imagegallery">

<li>
<a href="horse1.jpg" title="Brown Horse">
<img src="thumbhorse1.jpg" alt="Brown Horse">
</a>
</li>

<li>
<a href="horse2.jpg" title="Two Horses Running">
<img src="thumbhorse2.jpg" alt="Two Horses Running">
</a>
</li>

<li>
<a href="horse3.jpg" title="Horse Portrait">
<img src="thumbhorse3.jpg" alt="Horse Portrait">
</a>
</li>

...

</ul>
```

The entire HTML is shown below.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Image Gallery</title>
<link rel="stylesheet" href="layout.css" media="screen">
</head>

<body>

<div id="title">
<h1>HORSES</h1>
</div>
```

```
<div id="leftNav">
<a href="horse8.jpg">
Previous Image
</a>
</div>

<div id="rightNav">
<a href="horse8.jpg">
Next Image</a>
</div>

<div id="mainImage">
<img id="placeholder" src="horse1.jpg" alt="Image Gallery">
</div>

<div id="imageTitle">
<p id="description">Wild Horse</p>
</div>

<div id="imageNav">
<ul id="imagegallery">

<li>
<a href="horse1.jpg" title="Brown Horse">
<img src="thumbhorse1.jpg" alt="Brown Horse">
</a>
</li>

<li>
<a href="horse2.jpg" title="Two Horses Running">
<img src="thumbhorse2.jpg" alt="Two Horses Running">
</a>
</li>

<li>
<a href="horse3.jpg" title="Horse Portrait">
<img src="thumbhorse3.jpg" alt="Horse Portrait">
</a>
</li>

<li>
<a href="horse4.jpg" title="Horse by Hill">
<img src="thumbhorse4.jpg" alt="Horse by Hill">
</a>
</li>

<li>
<a href="horse5.jpg" title="Horse and Foal">
<img src="thumbhorse5.jpg" alt="Horse and Foal">
</a>
</li>

<li>
<a href="horse6.jpg" title="Horse and Foals">
<img src="thumbhorse6.jpg" alt="Horse and Foals">
</a>
</li>
```

```
<li>
<a href="horse7.jpg" title="Two Horses Standing">
<img src="thumbhorse7.jpg" alt="Two Horses Standing">
</a>
</li>

<li>
<a href="horse8.jpg" title="Wild Horse">
<img src="thumbhorse8.jpg" alt="Wild Horse">
</a>
</li>

</ul>
</div>

<script src="showimages.js"></script>

</body>
</html>
```

3.2 CSS

The cascading stylesheet is shown below.

```
/* layout.css: style sheet for image gallery */

* {
  margin: 0;      /* resets all margins, padding and borders */
  padding: 0;
  border: 0;
  outline: none;  /* removes dotted border around links etc  */
}

body {
  font-family: verdana, helvitica, arial, sans-serif;
  font-size: 14px;
  color: #333;                /* dark grey  */
  background-color: #ccc;     /* light grey */
  width: 980px;
  margin: 2px 5px;
  padding: 2px 10px;
}

h1 {
  color: #333;
  background-color: transparent;
}

a:link {
  text-decoration: underline;       /* underlines the link text */
}

a:link, a:visited, a:hover, a:active {
  color: #0000ff;                   /* sets link colour to blue */
  background: transparent;
}
```

```
#imagegallery li {      /* lists thumbnail images horizontally */
  float: left;
  padding: 5px;
  list-style: none;
  display: inline;
}

#imagegallery li a img {      /* blue border around thumbnails */
  border: 2px solid #0000ff;
}

#title {
  margin: 2px 5px;
  padding: 2px 200px;
  text-align: center;
}

#leftNav {
  float: left;
  width: 110px;
  height: 200px;
  margin: 5px;
  padding: 200px 10px 10px 10px;
  text-align: right;
}

#rightNav {
  float: right;
  width: 110px;
  height: 200px;
  margin: 5px;
  padding: 200px 10px 10px 10px;
}

#mainImage {
  float: left;
  width: 660px;
  margin: 2px 5px;
  padding: 2px 10px;
  text-align: center;
}

#imageTitle {
  clear: both;
  margin: 2px 5px;
  padding: 2px 10px;
  text-align: center;
}

#imageNav {
  height: 101px;
  margin: 2px 5px;
  padding: 2px 10px;
}
```

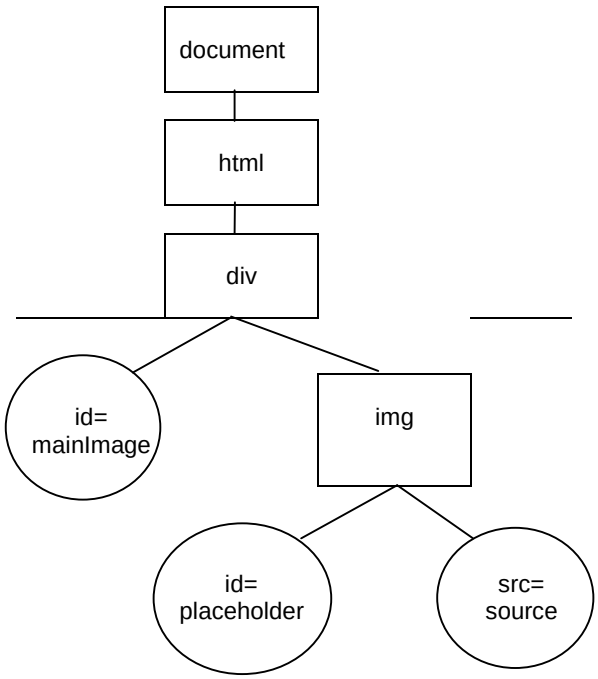
3.3 JavaScript

Now we use JavaScript to enable the visitor to select an image for enlargement. We have three methods for selecting an image. We can click on a thumbnail image. We can click on the previous or next image links shown alongside the main image. Or we can click on the left or right arrows at either end of the list of thumbnails. The selected image replaces the main image.

3.3.1 Selecting an Image

From a Thumbnail

We look at the HTML of §3.1 above and from it draw the structure shown below.

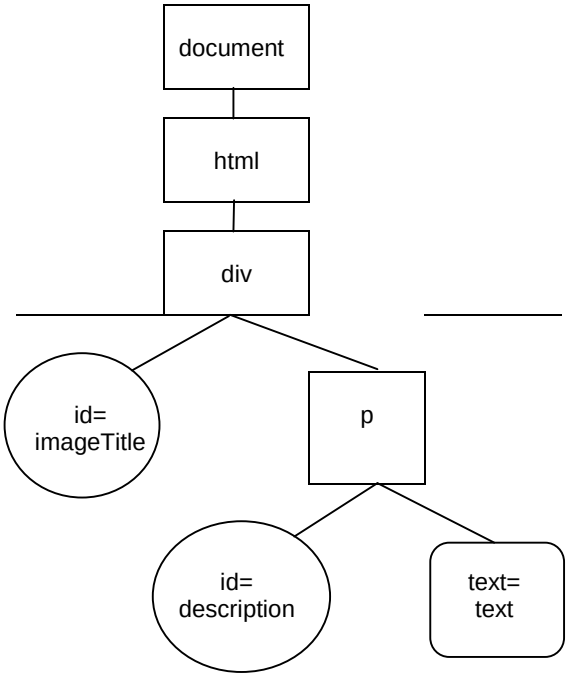


The *replaceMainImage()* function, shown below, replaces the main image with the selected image.

We use *getAttribute()* to retrieve the path to the selected image, and *setAttribute()* to change the *src* attribute of the *placeholder* image.

```
/* replaceMainImage: replaces main image with the selected image */
function replaceMainImage(selectedImage)
{
  var source = selectedImage.getAttribute("href");
  var placeholder = document.getElementById("placeholder");
  placeholder.setAttribute("src", source);
```

Now we deal with the descriptive text that accompanies each image. This text is identified by the *title* attribute of each anchor tag, <*a*>, in the HTML. We need to insert the text under the main image.



We derive the JavaScript from the structure of the HTML shown above.

```
...
var text = selectedImage.getAttribute("title");
var description = document.getElementById("description");
if (description.firstChild.nodeType == 3)  /* text node */
  description.firstChild.nodeValue = text;
...
```

Child Nodes

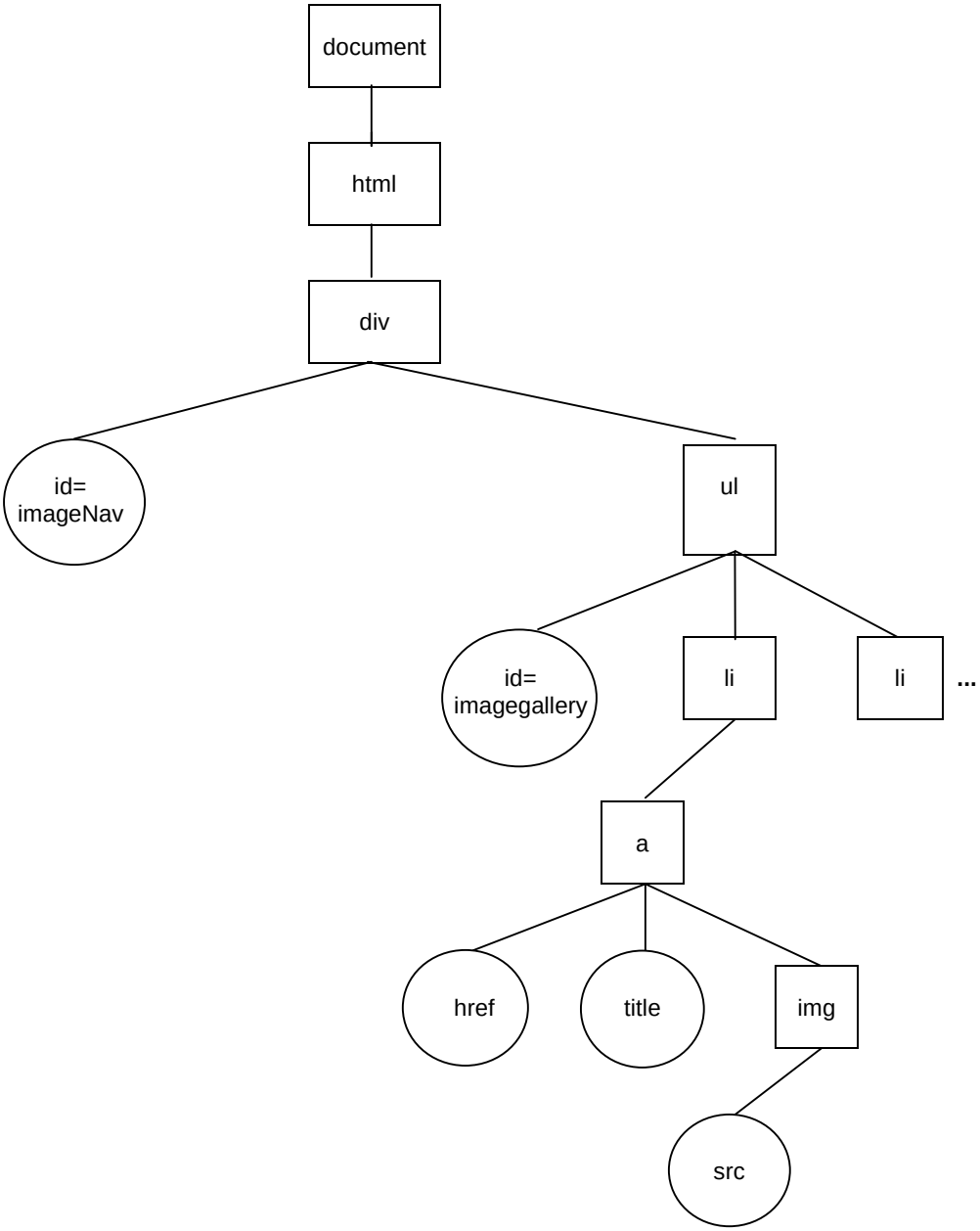
The JavaScript *childNode* property returns an array containing all the child nodes of a given element. *firstChild* and *lastChild* refer to the first and last nodes in a *childNodes* array.

The JavaScript *nodeType* property returns an integer identifying a node's type:

1 for element nodes
2 for attribute nodes
3 for text nodes

Next, we see how to replace the current main image with the image to its left or right in the picture gallery.

From Left and Right Links



We imagine the thumbnails in the gallery to be indexed from zero upwards. We set up a global variable (uggh) named *currentIndex* to hold the index of the current main image.

```
/* GLOBAL VARIABLES */
var currentIndex = 0;  /* refers to the current main image */
```

We set up an array of links to the images. We loop through this array and set *currentIndex* to the index of the current image, *source*.

```
...
var gallery = document.getElementById("imagegallery");
var links = gallery.getElementsByTagName("a");
for (var i = 0; i < links.length; i++)
  if (links[i].getAttribute("href") == source)
    currentIndex = i;
...
```

Finally, we return *false* because we intend to disable the default behaviour and take control of what happens when the link is clicked. Here is the entire function.

```
/* replaceMainImage: replaces main image with the selected image */
function replaceMainImage(selectedImage)
{
  var source = selectedImage.getAttribute("href");
  var placeholder = document.getElementById("placeholder");
  placeholder.setAttribute("src", source);

  var text = selectedImage.getAttribute("title");
  var description = document.getElementById("description");
  description.firstChild.nodeValue = text;

  var gallery = document.getElementById("imagegallery");
  var links = gallery.getElementsByTagName("a");
  for (var i = 0; i < links.length; i++)
    if (links[i].getAttribute("href") == source)
      currentIndex = i;

  return false;
}
```

Now, I did mention using navigation arrows at either end of the list of thumbnails - but I am not going to bother implementing them. You can have a go if you like.

3.3.2 Set Up

Image Gallery

A JavaScript event is an action such as a mouse click or a key press. We use the *onclick* event handler for specifying what happens when a visitor clicks on a thumbnail.

We create an array of <*a*> element nodes and name it *links*. We loop through all the links in the *imagegallery* element. We assign a function to each array element. The anonymous function passes the clicked link to the *replaceMainImage()* function. Here is the entire function.

```
/* setupGallery: attaches replaceMainImage() to the onclick event */
function setupGallery()
{
  var gallery = document.getElementById("imagegallery");
  var links = gallery.getElementsByTagName("a");
  for (var i = 0; i < links.length; i++) {
    links[i].onclick = function()
    {
      return replaceMainImage(this);
    }
  }
}
```

Previous and Next Links

The current main image is referred to by the global variable *currentIndex*. So, when the *Previous* or *Next Image* link is clicked we know what the previous or next image should be.

The function *setupNav()* sets up the navigation for the *Previous* and *Next Image* links. First, we retrieve all the links in the image gallery and store them in an array named *links*.

```
var gallery = document.getElementById("imagegallery");
var links = gallery.getElementsByTagName("a");
```

Then we retrieve the *Previous Image* link. We name it *leftLink*.

```
var left = document.getElementById("leftNav");
var leftLink = left.getElementsByTagName("a");
```

We specify the action to be taken when this link is clicked.

```
leftLink[0].onclick = function()
{
  currentIndex--;
  if (currentIndex < 0)
    currentIndex = links.length - 1;
  return replaceMainImage(links[currentIndex]);
}
```

We subtract 1 from *currentIndex*. If that makes *currentIndex* less than zero we set it to the index of the last image in the list. Finally, we replace the main image with the one referred to by *currentIndex*.

And similarly for the right link. Here is the entire function.

```
/* setupNav: sets up left and right image navigation */
function setupNav()
{
  var gallery = document.getElementById("imagegallery");
  var links = gallery.getElementsByTagName("a");

  var left = document.getElementById("leftNav");
  var leftLink = left.getElementsByTagName("a");

  var right = document.getElementById("rightNav");
  var rightLink = right.getElementsByTagName("a");

  leftLink[0].onclick = function()
  {
    currentIndex--;
    if (currentIndex < 0)
      currentIndex = links.length - 1;
    return replaceMainImage(links[currentIndex]);
  }

  rightLink[0].onclick = function()
  {
    currentIndex++;
    if (currentIndex > links.length - 1)
      currentIndex = 0;
    return replaceMainImage(links[currentIndex]);
  }
}
```

3.3.3 Loading Functions

The function *addLoadEvent()* loads our JavaScript functions when the page is being loaded by our web browser. The function is unashamedly lifted from Simon Willison *http://simonwillison.net/2004/may/26/addLoadEvent*.

```
/* addLoadEvent: loads functions on page load */
function addLoadEvent(func)
{
  var oldonload = window.onload;
  if (typeof window.onload != 'function')
    window.onload = func;
  else {
    window.onload = function()
    {
      if (oldonload)
        oldonload();
      func();
    }
  }
}
```

If *window.onload* has not already been assigned a function, the given function is assigned to *window.onload*. If *window.onload* has already been assigned a function, a new anonymous function, which first calls the original onload handler and then the new handler, is created. This allows us to load as many functions as we please.

```
addLoadEvent(setupGallery);
addLoadEvent(setupNav);
```

Now we can show the JavaScript in its entirety.

3.3.4 The JavaScript

```
/* showimages.js: manages an image gallery */

/* GLOBAL VARIABLES */
var currentIndex = 0;  /* refers to the current main image */

/* addLoadEvent: loads functions on page load */
function addLoadEvent(func)
{
  var oldonload = window.onload;
  if (typeof window.onload != 'function')
    window.onload = func;
  else {
    window.onload = function()
    {
      if (oldonload)
        oldonload();
      func();
    }
  }
}
```

```
/* replaceMainImage: replaces main image with the selected image */
function replaceMainImage(selectedImage)
{
  var source = selectedImage.getAttribute("href");
  var placeholder = document.getElementById("placeholder");
  placeholder.setAttribute("src", source);

  var text = selectedImage.getAttribute("title");
  var description = document.getElementById("description");
  description.firstChild.nodeValue = text;

  var gallery = document.getElementById("imagegallery");
  var links = gallery.getElementsByTagName("a");
  for (var i = 0; i < links.length; i++)
    if (links[i].getAttribute("href") == source)
      currentIndex = i;

  return false;
}

/* setupGallery: attaches replaceMainImage() to the onclick event */
function setupGallery()
{
  var gallery = document.getElementById("imagegallery");
  var links = gallery.getElementsByTagName("a");
  for (var i = 0; i < links.length; i++) {
    links[i].onclick = function()
    {
      return replaceMainImage(this);
    }
  }
}

/* setupNav: sets up left and right image navigation */
function setupNav()
{
  var gallery = document.getElementById("imagegallery");
  var links = gallery.getElementsByTagName("a");

  var left = document.getElementById("leftNav");
  var leftLink = left.getElementsByTagName("a");

  var right = document.getElementById("rightNav");
  var rightLink = right.getElementsByTagName("a");

  leftLink[0].onclick = function()
  {
    currentIndex--;
    if (currentIndex < 0)
      currentIndex = links.length - 1;
    return replaceMainImage(links[currentIndex]);
  }

  rightLink[0].onclick = function()
  {
    currentIndex++;
    if (currentIndex > links.length - 1)
      currentIndex = 0;
    return replaceMainImage(links[currentIndex]);
  }
}
```

```
addLoadEvent(setupGallery);
addLoadEvent(setupNav);
```

We have seen how to implement an image gallery. **Next** we look at menus.

Bibliography

Keith J *DOM Scripting: Web Design with JavaScript and the Document Object Model* Apress 2005
Haverbeke M *Eloquent JavaScript* William Pollock 2011
Briggs O et al *Cascading Style Sheets 2nd ed* FriendsOfEd 2004
www.tutorialpoint.com/javascript accessed April 2011
https://developer.mozilla.org accessed April 2011
Willison S *http://simonwillison.net/2004/may/26/addLoadEvent* accessed April 2011
www.geekphilosopher.com accessed April 2011
http://html-color-codes.info accessed April 2011