JavaScript and AJAX

2nd Edition

Tutorial 8

Designing Rollovers and Slide Shows

Objectives

- Access the images object collection
- Create an image object
- Preload images
- Create a rollover effect with a custom property
- Use anonymous functions in event handlers
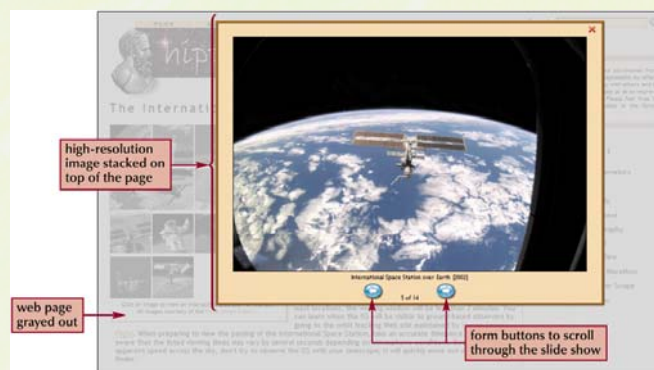- Design an interactive image gallery

Objectives (cont'd)

- Load high-resolution graphic images
- Create slide captions
- Create a page overlay
- Set an object's opacity
- Create and apply transition effects
- Apply anonymous functions to timedelayed commands

Setting Up a Slide Show

- Each thumbnail is linked to a large-resolution image file

Setting Up a Slide Show

- Marking slide images
  - Distinguish slide show images from other images on the Web site by enclosing them in links and adding attributes

Setting Up a Slide Show

- Loading the slide images
  - SetupSlideshow() function runs when the page loads

  - Creates an array of slide show images
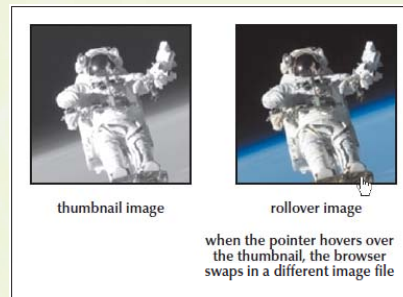
Setting Up a Slide Show

- Referencing inline images
  - To reference the collection of inline images within a document, use the `document.images` object collection
  - To reference a specific inline image, use `document.images[`*`idref`*`]` where *idref* is either the index number of the image object or the text of its id or name attribute

Creating an Image Rollover

- **Rollover:** Visual effect in which an object changes appearance in response to the pointer hovering over it
  - Commonly created through CSS using the hover pseudo-class
- To change source of inline image from original thumbnail to rollover image, change the value of the src property

  imgObject`.src = `*rollover*`;`

thumbnail image   rollover image

when the pointer hovers over the thumbnail, the browser swaps in a different image file

Creating an Image Rollover

- **Image object:** JavaScript object that stores an image and makes it available for quick retrieval
- To create an image object, run the command
  ```
  var imgObject = new Image();
  ```
- To set the dimensions of a new image object, use
  ```
  var imgObject = new Image(width, height);
  ```
- To define the source of an image object, use
  ```
  imgObject.src = url;
  ```

Creating an Image Rollover

- Define image objects as **custom properties**
  - Object properties defined by the programmer
  - Can contain text strings, numeric values, Boolean values, or other objects
  - Advantage: They become attached to the object and can be referenced throughout the program code
  - Code to define these properties will be stored in the createRollover() function

Creating an Image Rollover

- The createRollover() function

Creating an Image Rollover

- Use anonymous functions to swap images in response to onmouseout and onmouseover events
- To apply an anonymous function to an object event, enter

```
object.onevent = function() {
    commands
}
```

Creating an Image Gallery

- To create an application that requires minimal rewriting of the source Web page, write a function to create the gallery and then append it to the body of the current document

```
function createGallery() {
    var galleryBox = document.createElement("div");
    galleryBox.id = "galleryBox";

    // Insert a high-resolution slide
    var slide = document.createElement("img");
    slide.id = "gallerySlide";
    slide.src = "slide01.jpg";
    galleryBox.appendChild(slide);

    document.body.appendChild(galleryBox);
}
```

inserts an inline image displaying the first image in the slide show

creates a slide gallery box

appends the gallery to the document

Creating an Image Gallery

- Creating image objects for the high-resolution images:
  - Create a **custom property** to associate each thumbnail with a high-resolution version of the image
  - Preload image objects so that the slide show runs smoothly as it switches from one image to another
- Adding a slide caption:
  - Code retrieves caption text from image's alt attribute
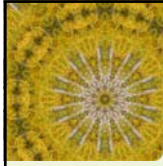
Hiding and Viewing
the Slide Gallery

- To hide the gallery when clicked, add an image button to the gallery box
- To display the slide gallery by clicking any of the thumbnail images, create the showGallery() function, which will be run with the onclick event handler

```
function showGallery() {

    // Reveal the slide show
    document.getElementById("galleryBox").style.display = "block";

    // Halt propagation of the click event
    return false;
}
```

- The changeSlide() function replaces the current slide image with a new slide and changes the caption

Navigating Through
the Slide Gallery

- Add navigation buttons to the slide gallery

```
<p id="galleryFooter">
    <input type="image" src="back.png" />
    <input type="image" src="forward.png" />
</p>
```

```
    // Insert the slide caption
    var slideCaption = document.createElement("p");
    slideCaption.id = "slideCaption";
    slideCaption.innerHTML = slides[0].alt;
    galleryBox.appendChild(slideCaption);

    // Create the gallery footer
    var galleryFooter = document.createElement("p");
    galleryFooter.id = "galleryFooter";

    // Create a button to go to the previous slide
    var slideBack = document.createElement("input");
    slideBack.type = "image";
    slideBack.src = "back.png";
    galleryFooter.appendChild(slideBack);

    // Create a button to go to the next slide
    var slideForward = document.createElement("input");
    slideForward.type = "image";
    slideForward.src = "forward.png";
    galleryFooter.appendChild(slideForward);

    galleryBox.appendChild(galleryFooter);

    document.body.appendChild(galleryBox);

}
```

gallery footer

button to go to the previous slide

button to go to the next slide

appends the footer to the slide gallery box

Navigating Through
the Slide Gallery

buttons to move forward and backward through the slide show
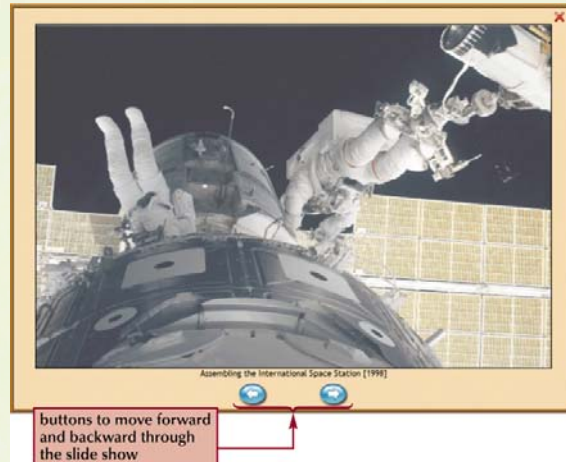
Navigating Through
the Slide Gallery

- Program actions of navigation buttons to keep track of the slide number by attaching an index number to each slide and to each image as a custom property


```
for (var i = 0; i < slides.length; i++) {

    // Create a rollover for each slide
    createRollover(slides[i]);

    // Attach a high-resolution image object to each slide
    createHighRes(slides[i], i);
}

if (slides.length > 0) {
    createGallery(slides);
}

}

function createHighRes(thumb, index) {

    thumb.big = new Image();
    thumb.big.src = thumb.src.replace(/_thumb/, "");

    // Display high-resolution image in slide gallery
    thumb.onclick = showGallery;

    // Set the index of the slide
    thumb.big.index = index;
}
```

index values are based on the position of the slide in the slides array

adds the index value of each slide as a custom property

Navigating Through the Slide Gallery

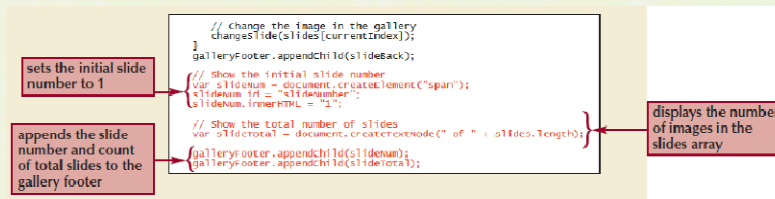- To display the slide number

```
<span id="slideNumber">number</span> of total
```

- Append the span element and the text node to the gallery footer by adding commands to the createGallery() function

```
                // Change the image in the gallery
        changeSlide(slides[currentIndex]);
    }
    galleryFooter.appendChild(slideBack);

    // Show the initial slide number
    var slideNum = document.createElement("span");
    slideNum.id = "slideNumber";
    slideNum.innerHTML = "1";

    // Show the total number of slides
    var slideTotal = document.createTextNode(" of " + slides.length);

    galleryFooter.appendChild(slideNum);
    galleryFooter.appendChild(slideTotal);
```

sets the initial slide number to 1

appends the slide number and count of total slides to the gallery footer

displays the number of images in the slides array

Creating a Page Overlay

- Design the page overlay effect

```
<div id = "pageOverlay"></div>
```

- The createOverlay() function

```
function createOverlay() {
    // Create an overlay to obscure the view of the web page
    var pageOverlay = document.createElement("div");
    pageOverlay.id = "pageOverlay";

    document.body.appendChild(pageOverlay);
}
```

creates the page overlay

appends it to the document body
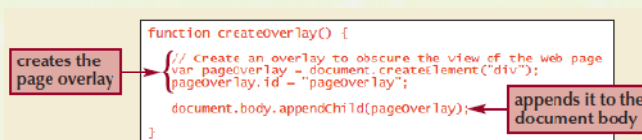
- Define the appearance of the overlay by adding new styles to the gallery.css style sheet

Working with Opacity Values

- Makes the page overlay semitransparent so that the rest of the Web page can be dimly seen through it
- **Opacity** defines the percentage to which an object is nontransparent or opaque

opacity = 0%   opacity = 25%   opacity = 50%   opacity = 75%   opacity = 100%
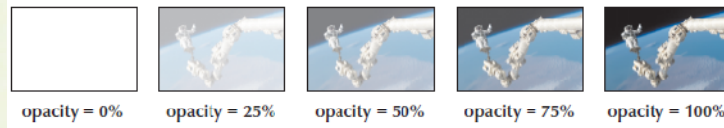
Setting Opacity with CSS

- To set opacity value for objects under Internet Explorer, use the style

    filter: alpha(opacity = *value*)

  where *value* ranges from 0 (completely transparent) to 100 (completely opaque)
- To set opacity under non-IE browsers, use the style

    opacity: *value*

  where *value* ranges from 0 (completely transparent) to 1 (completely opaque)

Setting Opacity
with JavaScript

- To set opacity value for objects under Internet Explorer, use the object property

      $object$.style.filter = "alpha(opacity = $value$)";

  where *object* is the object and *value* is the opacity level ranging from 0 to 100
- If filters have been defined for the object under IE, use

      $object$.filters["$alpha$"].opacity = $value$;

  or

      $object$.filters.alpha.opacity = $value$;
- To set the opacity under non-IE browsers, use

      $object$.style.opacity = $value$;

  where *value* ranges from 0 to 1

Creating Fade-Ins
and Fade-Outs

- **Transition:** Application of a visual effect
- **Fade-in effect**
  - Opacity changes over time from completely transparent to completely opaque
  - Use JavaScript's setTimeout() method to set different opacity values at different delay times
  - Parameters of fadeIn() function: objID, maxOpacity, fadeTime, delay
- **Fade-out effect**
  - An object changes from being opaque to completely transparent

Creating Fade-Ins and Fade-Outs

```
function fadeIn(objID, maxOpacity, fadeTime, delay) {

    // Calculate the interval between opacity changes
    var fadeInt = Math.round(fadeTime*1000)/maxOpacity;

    // Loop up the range of opacity values
    for (var i = 0; i <= maxOpacity; i++) {
        setTimeout("setOpacity('" + objID + "', " + i + ")", delay);
        delay += fadeInt;
    }

}

function fadeOut(objID, maxOpacity, fadeTime, delay) {

    // Calculate the interval between opacity changes
    var fadeInt = Math.round(fadeTime*1000)/maxOpacity;

    // Loop down the range of opacity values
    for (var i = maxOpacity; i >= 0; i--) {
        setTimeout("setOpacity('" + objID + "', " + i + ")", delay);
        delay += fadeInt;
    }

}
```

gradually fades in the object

gradually fades out the object

Applying a Fade-In

```
function changeSlide(slide) {

    // Set object references
    var galleryBox = document.getElementById("galleryBox");
    var oldSlide = document.getElementById("gallerySlide");
    var slideCaption = document.getElementById("slideCaption");
    var slideNum = document.getElementById("slideNumber");

    // Replace current slide with new slide
    setOpacity("gallerySlide", 0);
    var newSlide = oldSlide.cloneNode(true);
    newSlide.src = slide.big.src;
    newSlide.index = slide.big.index;
    galleryBox.replaceChild(newSlide, oldSlide);
    fadeIn("gallerySlide", 100, 0.5, 0);

    // Replace current caption with new caption
    slideCaption.innerHTML = slide.alt;

    // Update the slide number
    slideNum.innerHTML = newSlide.index + 1;
}
```

makes the current slide completely transparent

after replacing the old slide with the new slide, fades in the new slide

Applying a Fade-Out

```
function createGallery(slides) {

    var galleryBox = document.createElement("div");
    galleryBox.id = "galleryBox";

    // Insert a button to close the gallery
    var galleryTitle = document.createElement("p");
    galleryTitle.id = "galleryTitle";

    var closeButton = document.createElement("input");
    closeButton.type = "image";
    closeButton.src = "galleryclose.png";
    closeButton.onclick = function() {
        fadeOut("galleryBox", 100, 0.5, 0);
        fadeOut("pageOverlay", 80, 0.5, 0);
        galleryBox.style.display = "none";
        document.getElementById("pageOverlay").style.display = "none";
    }
}
```

fades out the gallery box and page overlay

Creating Fade-Ins and Fade-Outs

- Using anonymous functions with timed commands
  - To delay the implementation of a command block, place the block in an anonymous function nested within setTimeout() method:
    ```
    setTimeout(function() {
         commands;
    }, delay);
    ```
  - To repeatedly run a command block at specified intervals, use an anonymous function and the setInterval() method:
    ```
    setInterval(function() {
         commands;
    }, interval);
    ```
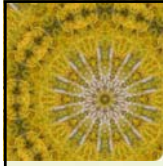
Creating Fade-Ins and Fade-Outs

- Internet Explorer supports a library of transition filters designed to dynamically change the appearance of an object over a specified time interval

filter: revealTrans(duration = 2, transition = 5)

First Sample Image