CS 1124 Media Computation

Lecture 11.2, November 5, 2008
Steve Harrison

When should you use objects?

- Define your own objects when you have:
  - **Data in groups, like both pictures and sounds.**
  - **Behavior that you want to define over that group.**
- Use existing objects:
  - **Always—they're very powerful!**
  - **Unless you're not comfortable with dot notation and the idea of methods.**
    - Then functions work just fine.

Using “map()” with slides

- Slides are now just objects, like any other kind of object in Python.
- They can be in lists, for example.
- “map()” takes a function and list, executing the function with elements of the list

PlaySlideShow with map()

```
def playslideshow():
    slide1 = slide(getMediaPath("barbara.jpg"), getMediaPath("bassoon-c4.wav"))
    slide2 = slide(getMediaPath("beach.jpg"),getMediaPath("bassoon-e4.wav"))
    slide3 = slide(getMediaPath("santa.jpg"),getMediaPath("bassoon-g4.wav"))
    slide4 = slide(getMediaPath("jungle2.jpg"),getMediaPath("bassoon-c4.wav"))
    map(showSlide,[slide1,slide2,slide3,slide4])
```

TODAY

- Objects
- Other Languages
  - Javascript

What do other languages look like?

- We call the language “look” its *syntax*
- Python is a fairly traditional language in terms of syntax.
  - **Languages like Scheme and Squeak are significantly different.**
- Some points of difference:
  - **Whether or not variables have to be declared before first use.**
  - **Details of how individual lines are written.**
  - **Details of how blocks are defined.**

TODAY

- Objects
- Other Languages
    - Javascript

JavaScript

- JavaScript is meant to be a *scripting* language, like Python.
  - Scripting languages are meant for non-professional programmers to solve simple tasks.
  - It's designed to look like Java to ease the transition in either way.
- JavaScript can be used by the web server (used on the computer accessed via the Internet), or it can be used within an HTML page.
  - If it's within the HTML page, it's actually executed by the user's browser.
  - We call that client side JavaScript.

JavaScript syntax

- Variables must be declared before use.
  - **var a = 12;**
  - **var a;**
    a = 12;
- Blocks are delimited with curly braces.

```
function test()
{
   document.writeln("This is a test");
}
```

- Statements:
  - **function instead of def**
  - **End lines with semicolons ";" (But lines can have returns in the middle of them.)**
  - **The for statement is numeric (mostly) and has different parts**

JavaScript is all about objects

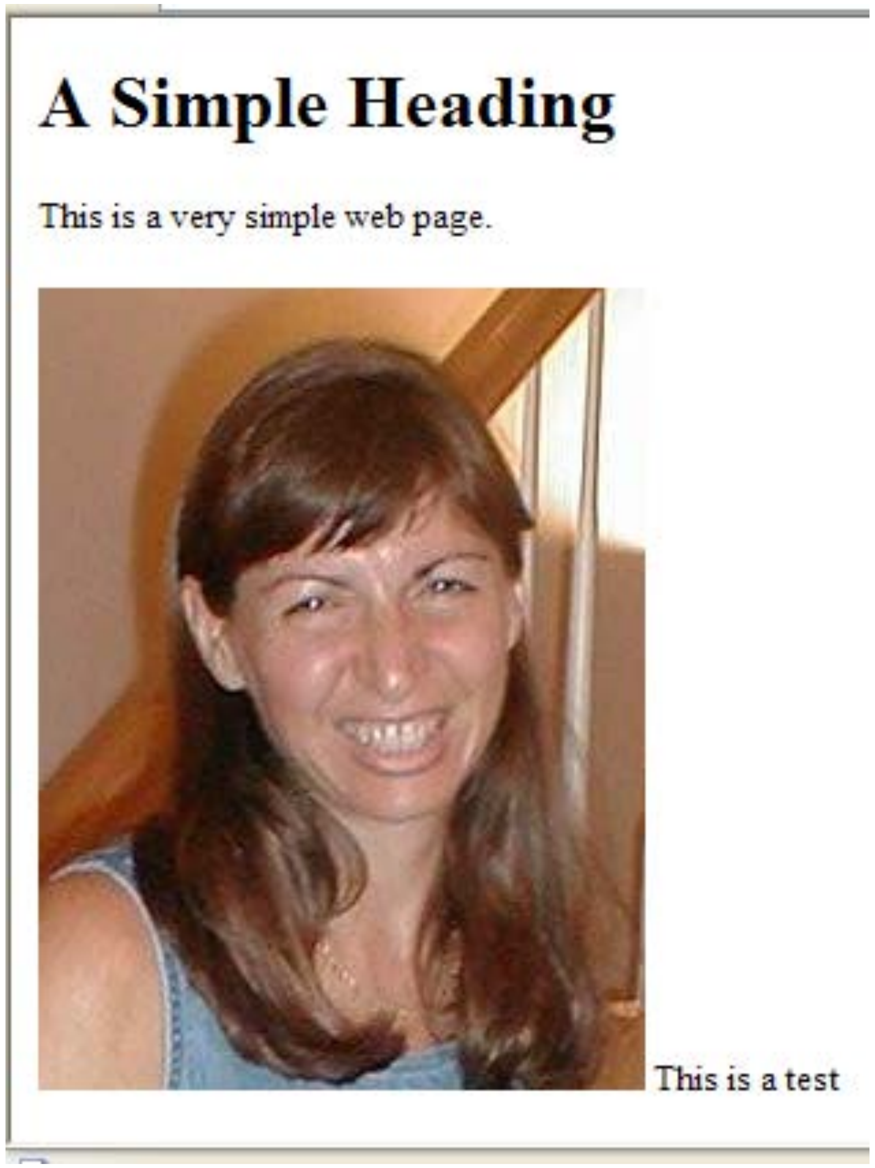
- Just about every function is actually a method.
- For example, there is no global **print**.
  - **each kind of object has its own print-like methods**
- There is a function **write** or **writeln**
  - **Writeln adds a new line (‘\n’) at the end.**
- But these aren’t global functions.
  - **To write into the document, you use document.write()**
  - **document.write() is a method on the HTML document itself.**

Embedding JavaScript inside HTML

- JavaScript sits inside of HTML pages.
  - **You wrap <script> </script> tags around the JavaScript.**
- You can have <script> tags in two kinds of places.
  - **Inside the <head></head> tags to define functions used elsewhere.**
  - **Inside the body, where the scripts are actually executed.**

Adding some simple JavaScript

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD
      HTML 4.01 Transition//EN" "http://
      wwww.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>The Simplest Possible Web Page</title>
<script>
function test()
{
   document.writeln("This is a test");
}
</script>
</head>
<body>
<h1>A Simple Heading</h1>
<p>This is a very simple web page.</p>
<p><image src="mediasources/barbara.jpg" />
<script> test() </script></p>
</body>
</html>
```



A Simple Heading

This is a very simple web page.

This is a test

Going into detail on the function

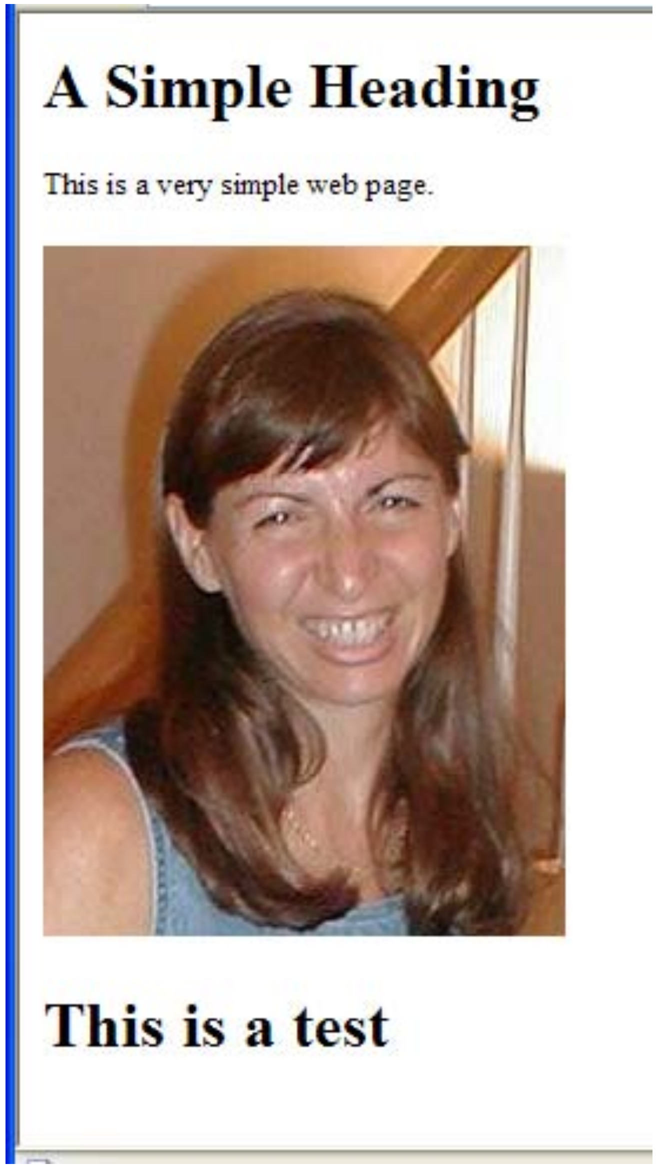
```
<script>
function test()
{
   document.writeln("This is a test");
}
</script>
</head>
<body>
<h1>A Simple Heading</h1>
<p>This is a very simple web page.</p>
<p><image src="mediasources/barbara.jpg" />
<script> test() </script></p>
```

Here’s a function named “test” with no inputs, that only writes out a string.

Here we execute the function.

Can also insert HTML

```
<script>
function insertHead()
{
   document.writeln("<h1>This is a test</h1>");
}
</script>
</head>
<body>
<h1>A Simple Heading</h1>
<p>This is a very simple web page.</p>
<p><image src="mediasources/barbara.jpg" />
</p>
<script> insertHead() </script>
</body>
</html>
```



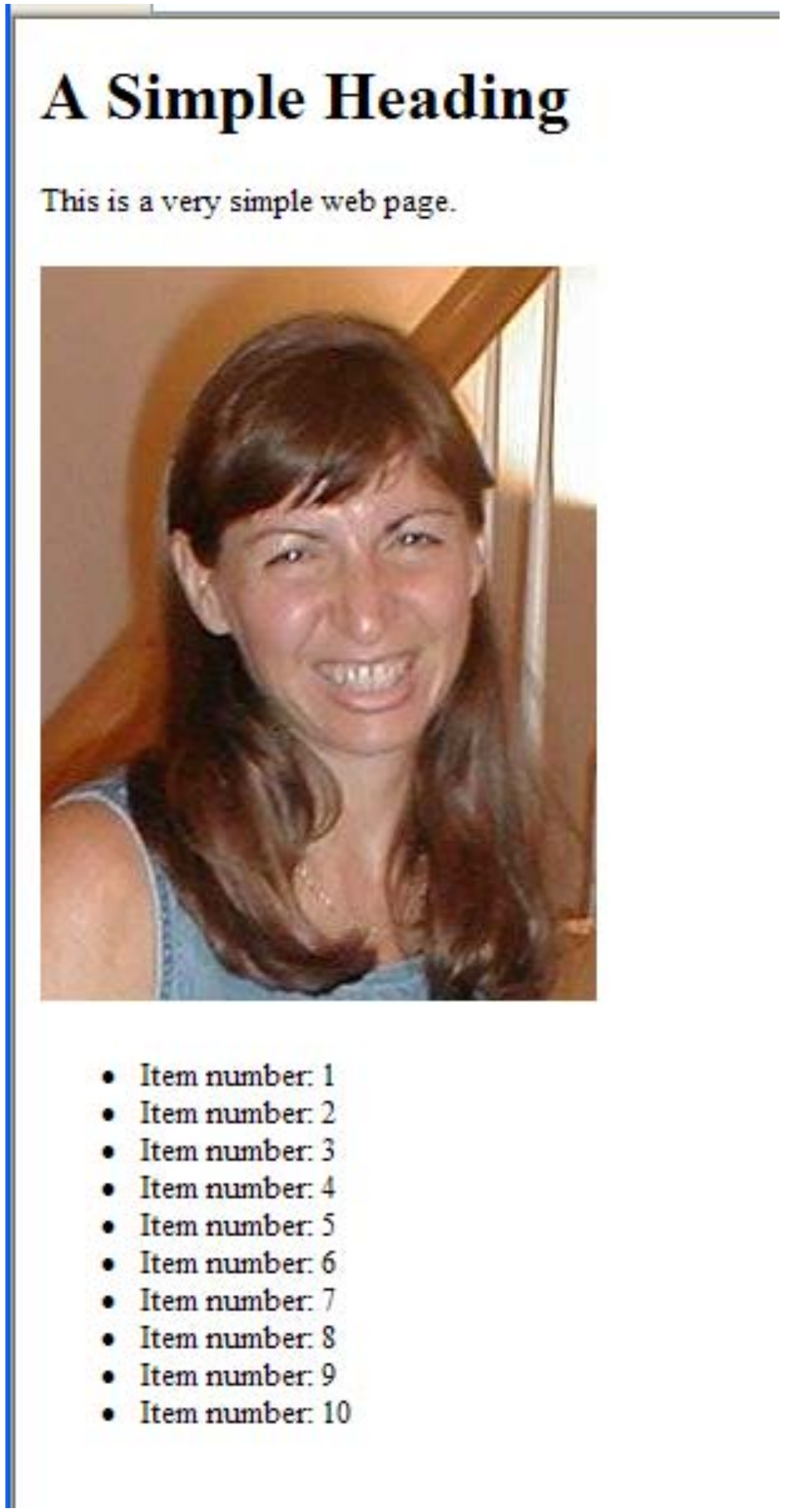
A Simple Heading

This is a very simple web page.

This is a test

Using loops

```
<html>
<head>
<title>The Simplest Possible Web Page</title>
<script>
function countToTen()
{
  document.write("<ul>");
  for (i=1; i<= 10; i++)
  {
    document.write("<li>Item number: ",i);
  }
  document.write("</ul>");
}
</script>
</head>
<body>
<h1>A Simple Heading</h1>
<p>This is a very simple web page.</p>
<p><image src="mediasources/barbara.jpg" />
</p>
<script> countToTen() </script>
</body>
</html>
```



A Simple Heading

This is a very simple web page.

- Item number: 1
- Item number: 2
- Item number: 3
- Item number: 4
- Item number: 5
- Item number: 6
- Item number: 7
- Item number: 8
- Item number: 9
- Item number: 10

Explaining that function

```
function countToTen()
{
  document.write("<ul>");
  for (i=1; i<= 10; i++)
  {
    document.write("<li>Item number: ",i);
  }
  document.write("</ul>");
}
```

We can write out <ul> and </ul>

Creating an item for each value of i

Explaining that Loop

```
for (i=1; i<= 10; i++)
   {
      document.write("<li>Item
     number: ",i);
   }
```

- A **for** loop has three parts, separated by semi-colons.
  - **What to do first.**
    - For us, set i to 1
  - **When to stop**
    - For us, i<=10
  - **What to do each time through the loop**
    - i++ means to increment the value of i by 1.
- It’s a notation that was invented by the programming language C and has been adopted by many languages

Operators in JavaScript

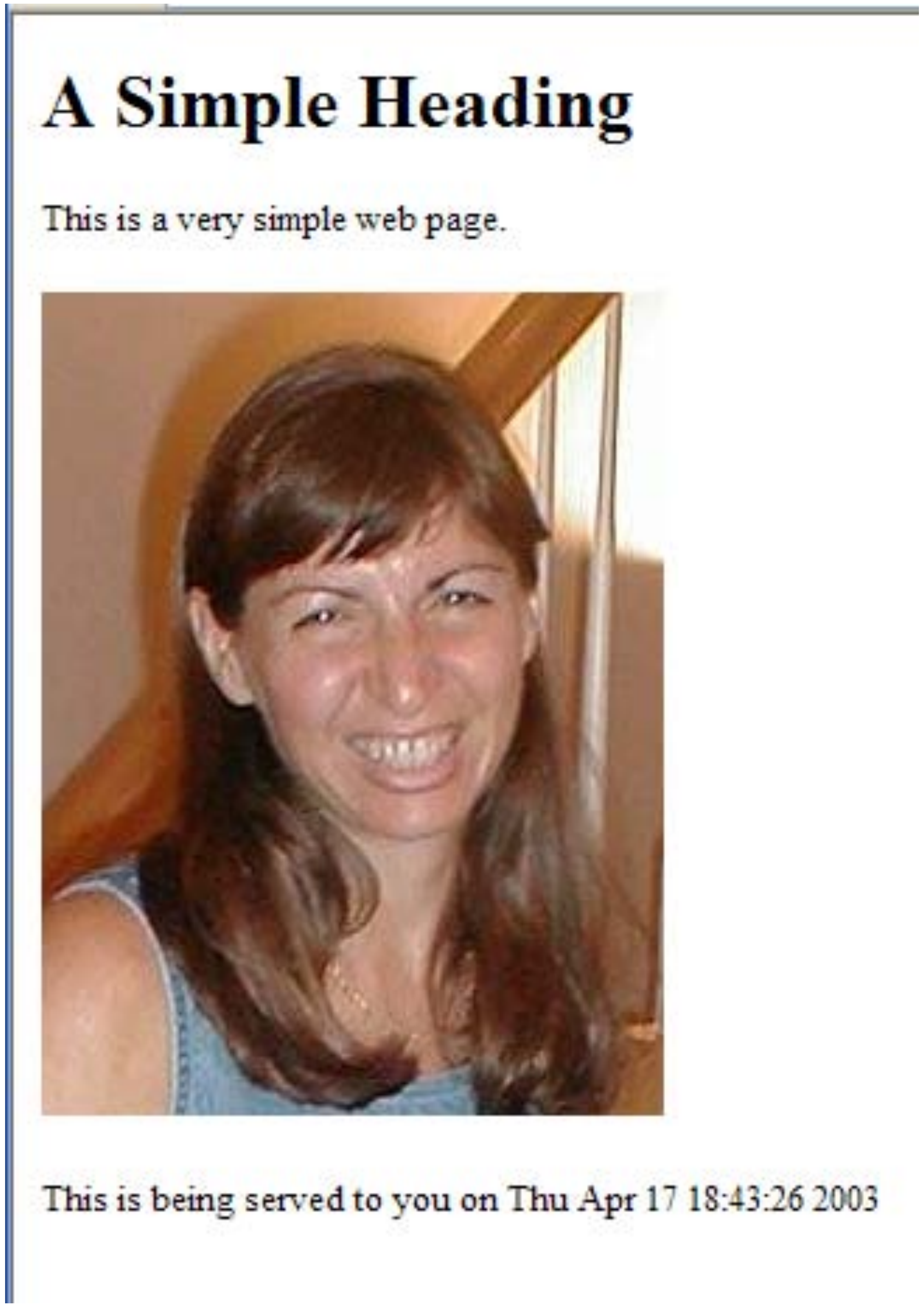
- The same kind of operators you know in Python
  - **+ - * /**
    - + even works for strings, as well as numbers
  - **< <= > >=**
  - **== and !=**
  - **! for not**
- Logical operators are a little weird
  - **&& is AND**
  - **|| is OR**

Can we display anything useful?

- Sure!
  - **Anything you can compute.**
  - **Anything that you can get from built-in functions (mostly methods).**
    - There are lots of them.
- You don’t *have* to have a function either.
  - **You can just put the script in-line**

Displaying the date and time

```
<p>This is a very simple web
    page.</p>
<p><image src="mediasources/
    barbara.jpg" />
</p>
<p>This is being served to you on
    <script>document.write(Date());
</script></p>
```



A Simple Heading

This is a very simple web page.

This is being served to you on Thu Apr 17 18:43:26 2003

Using dialogs in JavaScript

```
function check()
{
  var agree = false;
  agree = confirm('Do you enjoy CS?');
  if (agree)
    notes=prompt("Give me one good thing about CS:");
  if (! agree)
    notes=prompt("Why don't you like CS?");
  alert("You said:"+notes);
}
…
<script> check() </script>
</body>
</html>
```

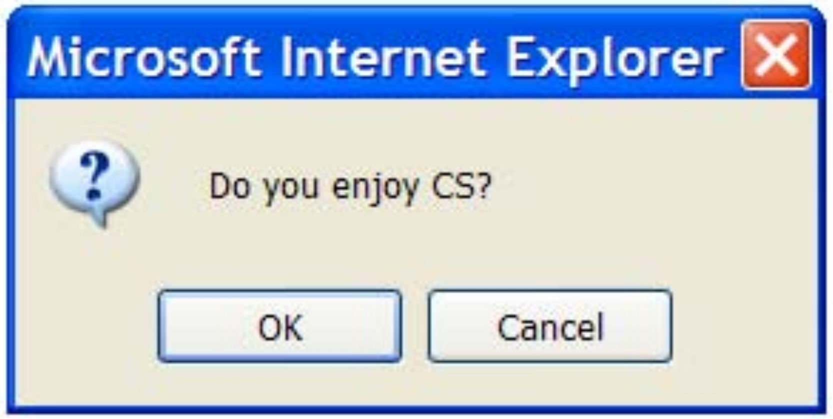
agree will be true or false.

! agree is *not* agree.

Notice: We can indent or not indent as we want here. Indentation is *not* important in JavaScript (or most other languages.)

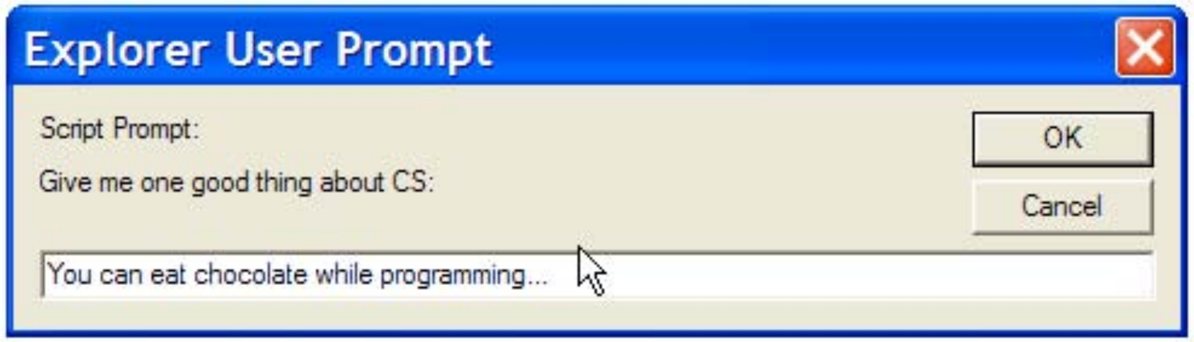
What happens when this runs

What happens when this runs



Do you enjoy CS?

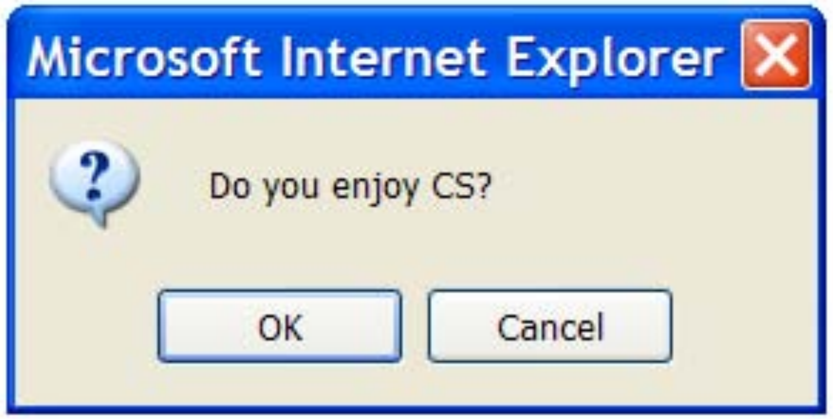
What happens when this runs



Script Prompt:

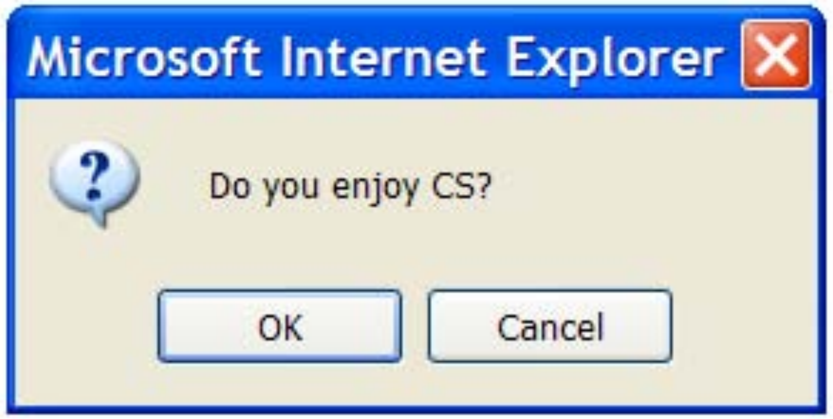
Give me one good thing about CS:

You can eat chocolate while programming...

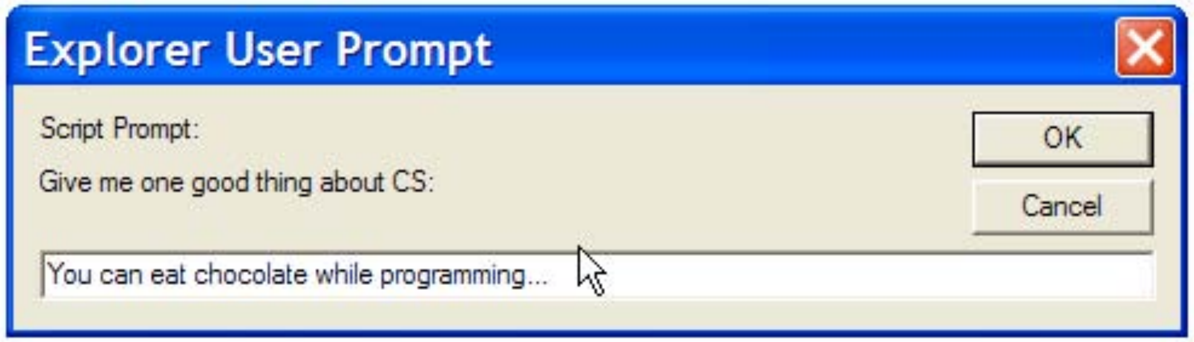


Do you enjoy CS?

What happens when this runs



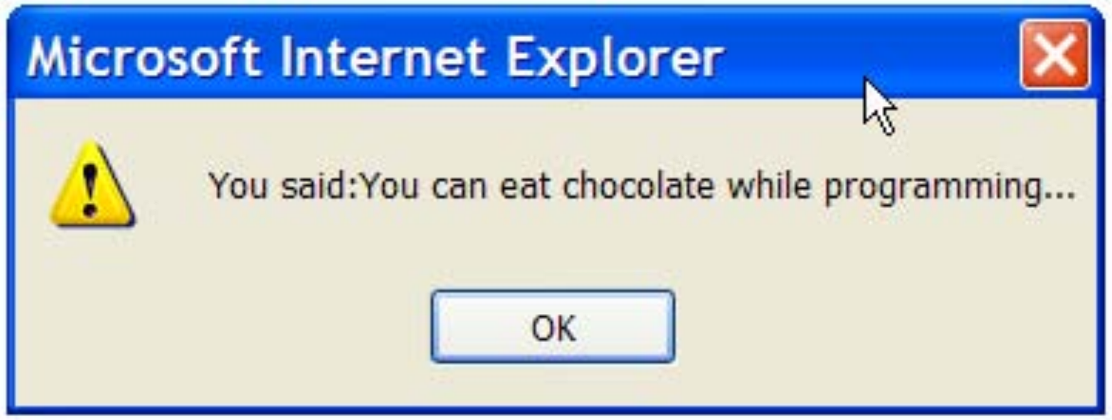
Do you enjoy CS?



Script Prompt:

Give me one good thing about CS:

You can eat chocolate while programming...



You said:You can eat chocolate while programming...

Different kinds of dialogs

- Confirm: Puts up a prompt, returns true or false.
- Alert: Beeps and displays one thing with an OK button. No return.
- Prompt: Asks the user for a line of text. Returns that text.

Running on Loading the Page

- This program runs when the page loads.
- Is that what you *really* want to happen?
  - **The user sees nothing at all until they go to your page and then these dialog boxes happen.**
- Isn’t it more natural for dialog boxes to pop up when you click on something?

Events: Key to responding to users

- The key to responding to users are *events*.
- Events are actions taken by the user that can be *caught* by your program in JavaScript.
- Events are caused by the user typing a key, moving the mouse, clicking the mouse.

Events in JavaScript

- onKeyPress
- onKeyUp
- onKeyDown
- onClick
- onDblClick
- onMouseOver
- onMouseOut
- onMouseMove
- onMouseDown
- onMouseUp
- onChange (for text fields)
- onLoad
- And many, many more
  - **Some of them depend on the browser.**

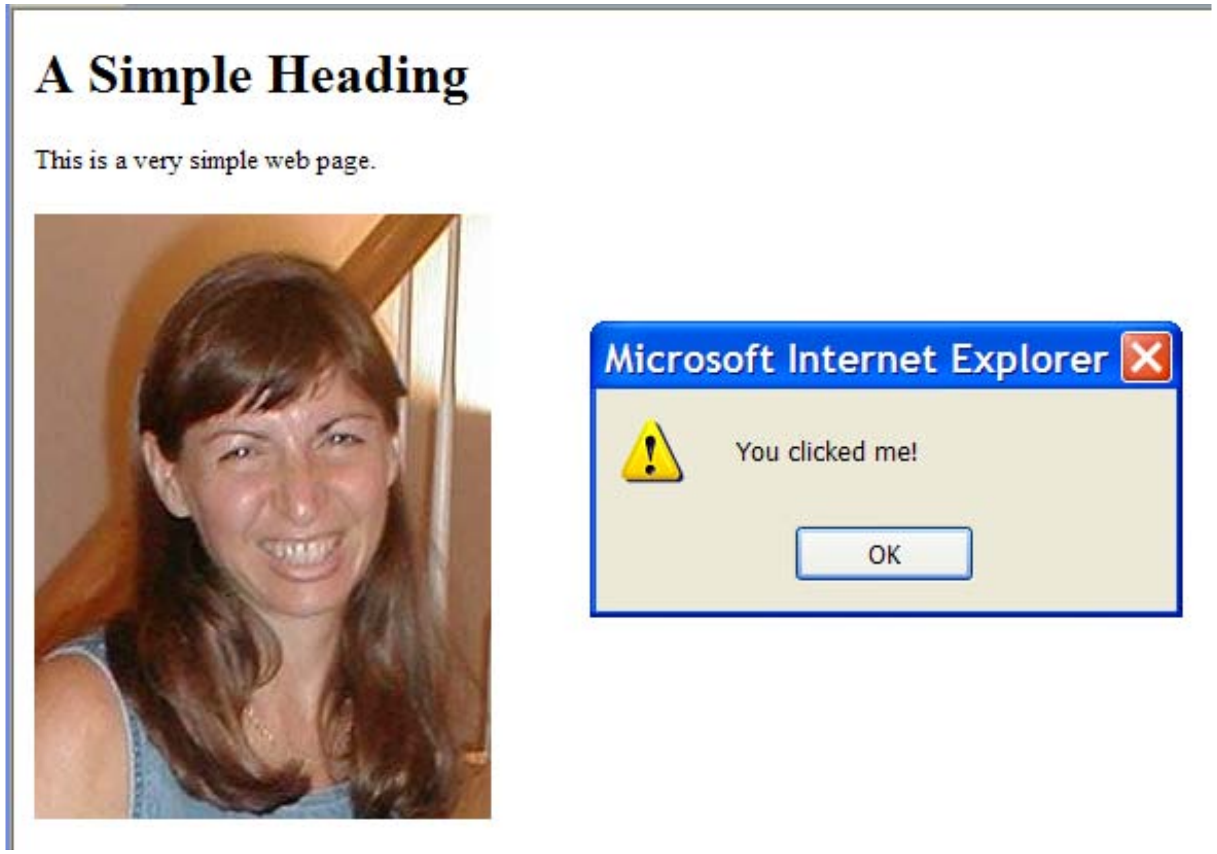
Catching an event

- On appropriate tags, assign the event to some JavaScript code in a string.
  - **Most of these can be used with anchor or image tags.**
  - **onLoad can be used with the <body> tag.**
  - **onChange can be used with text fields.**
    - We haven’t seen them yet.

Example Event Code

```
<body>
<h1>A Simple Heading</h1>
<p>This is a very simple web
    page.</p>
<p><image src="mediasources/
    barbara.jpg"
    onClick='alert("You clicked
    me!")' />
</p>


</body>
```



A Simple Heading

This is a very simple web page.

You clicked me!

Opening another Window

- To open another window, we use the function open()
- Open can take up to three inputs:
  - First input is the URL to open.
  - Second is the name of the window
  - Third is a string with a variety of inputs possible.

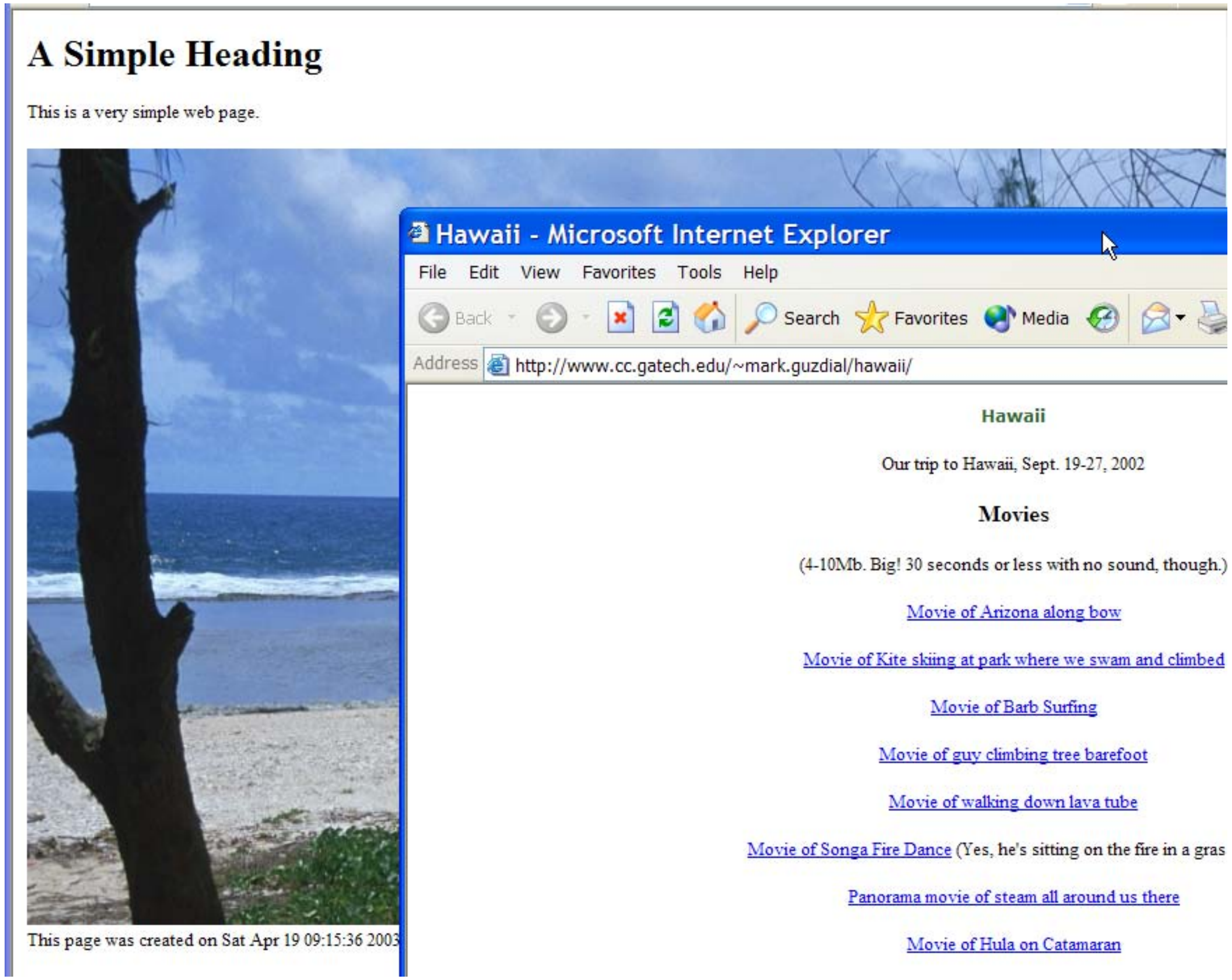
Popping up a window on a click

```
<html>
<head>
<title>The Simplest Possible Web Page</title>
<script>
function goToHawaii()
{
  var win=open('http://www.cc.gatech.edu/~mark.guzdial/hawaii/','Hawaii');
}
</script>
</head>
<body>
<h1>A Simple Heading</h1>
<p>This is a very simple web page.</p>
<p><image src="mediasources/beach.jpg" onClick="goToHawaii()" />
This page was created on <script> document.write(Date()); </script></p>
</body>

</html>
```

notice: event in HTML does not require <script></script>

Popping up a Window


Hawaii - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Media
Address

http://www.cc.gatech.edu/~mark.guzdial/hawaii/

A Simple Heading

This is a very simple web page.

This page was created on Sat Apr 19 09:15:36 2003

Hawaii

Our trip to Hawaii, Sept. 19-27, 2002

Movies

(4-10Mb. Big! 30 seconds or less with no sound, though.)

Movie of Arizona along bow

Movie of Kite skiing at park where we swam and climbed

Movie of Barb Surfing

Movie of guy climbing tree barefoot

Movie of walking down lava tube

Movie of Songa Fire Dance (Yes, he's sitting on the fire in a gras

Panorama movie of steam all around us there

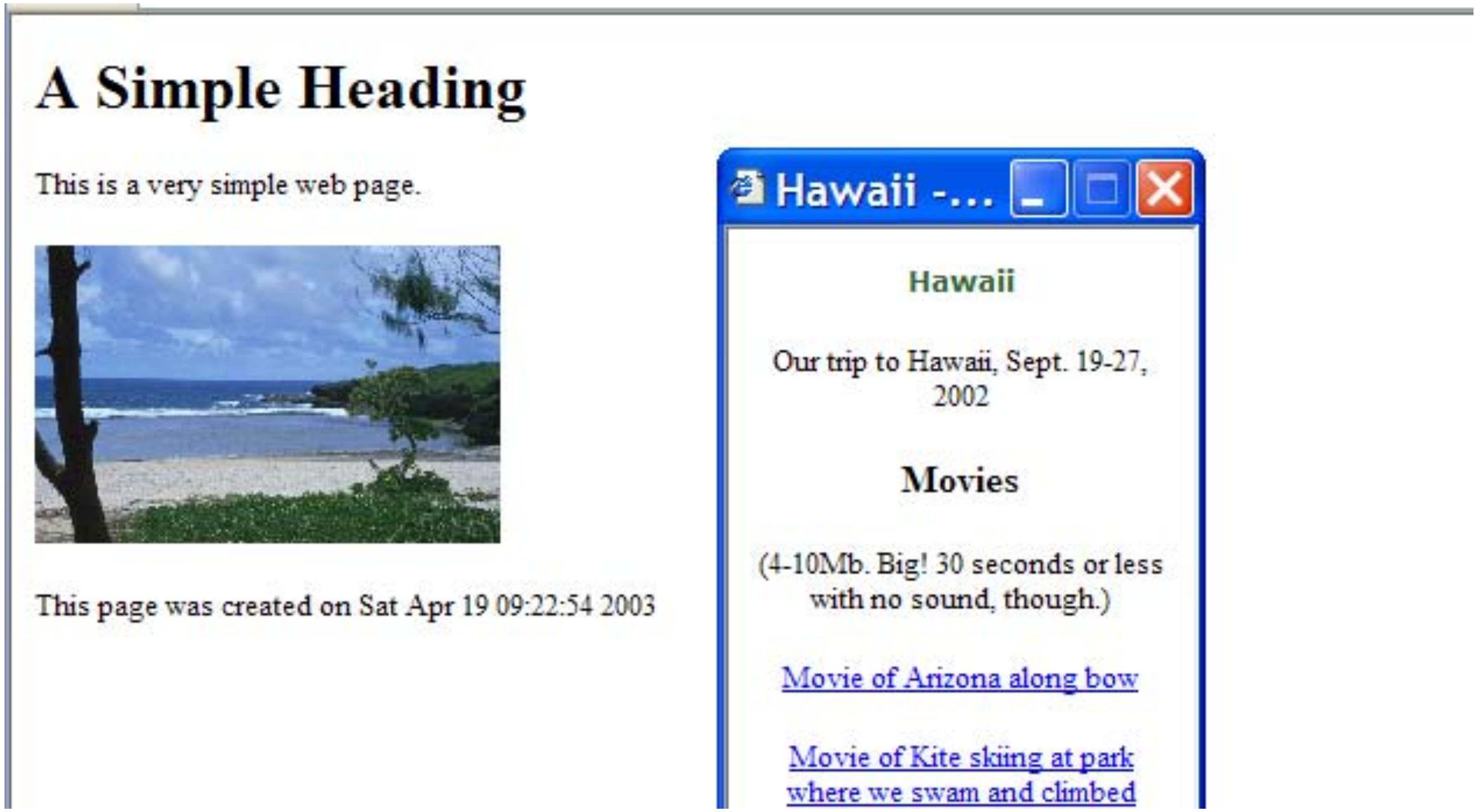
Movie of Hula on Catamaran

Changing the window’s characteristics

```
<head>
<title>The Simplest Possible Web Page</title>
<script>
function goToHawaii()
{
  var
  win=open('http://www.cc.gatech.edu/~mark.guzdial/hawaii/','Hawaii',
   "titlebar=no,width=200");
}
</script>
</head>
```

Note: Even something this simple doesn’t work on all browsers!

Changing the window's characteristics



A Simple Heading

This is a very simple web page.

This page was created on Sat Apr 19 09:22:54 2003

Hawaii

Our trip to Hawaii, Sept. 19-27, 2002

Movies

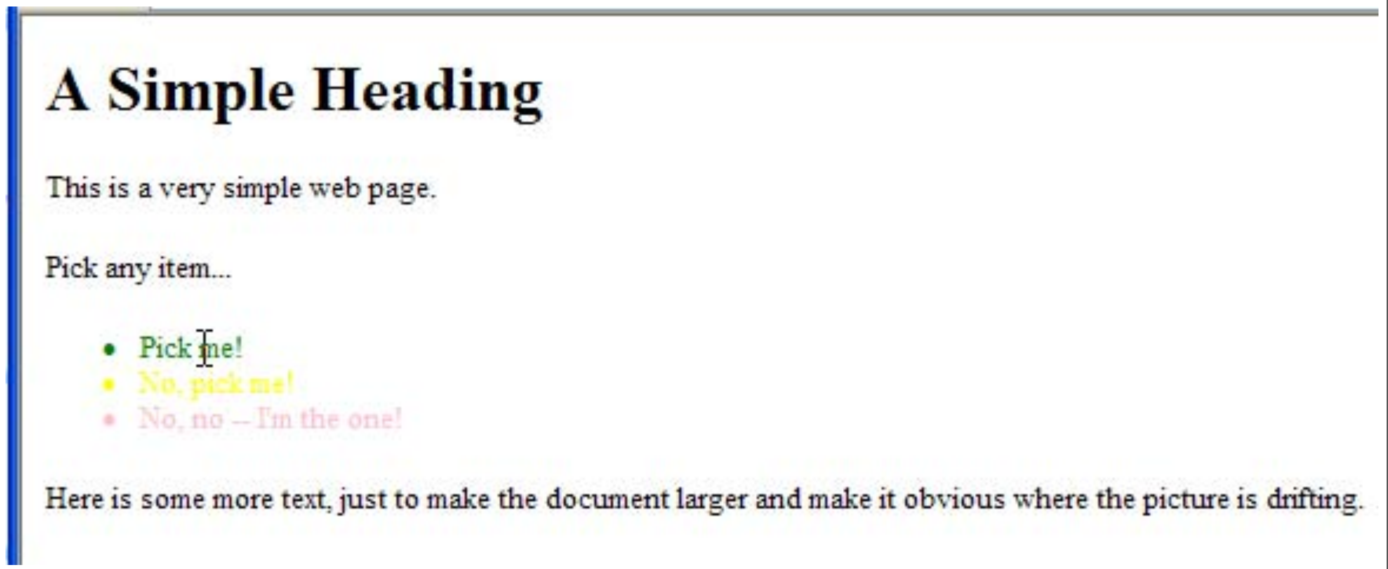
(4-10Mb. Big! 30 seconds or less with no sound, though.)

Movie of Arizona along bow

Movie of Kite skiing at park where we swam and climbed

Controlling colors with mouseOver and mouseOut

```
<body>
<h1>A Simple Heading</h1>
<p>This is a very simple web page.</p>
<p>Pick any item...</p>
<ul>
<li onmouseover="this.style.color='green'"
onmouseout="this.style.color='black'">Pick me!</li>
<li onmouseover="this.style.color='red'"
onmouseout="this.style.color='yellow'">No, pick me!</li>
<li onmouseover="this.style.color='magenta'"
onmouseout="this.style.color='pink'">No, no -- I'm the one!</li>
</ul>
```



A Simple Heading

This is a very simple web page.

Pick any item...

- Pick me!
- No, pick me!
- No, no -- I'm the one!

Here is some more text, just to make the document larger and make it obvious where the picture is drifting.

KIDS TRY THIS AT HOME!

Fields and Buttons in HTML

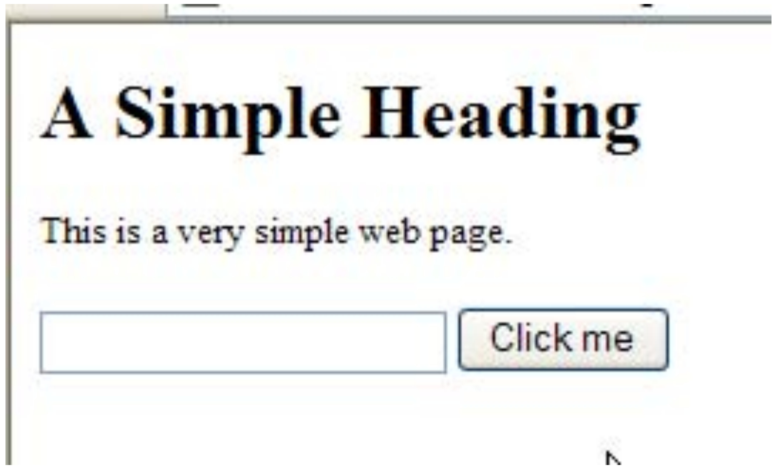
- To create fields and buttons in HTML, we need a *form.*
  - Forms are delimited with <form> and </form>
- Examples of things we can have in forms.
  - <input type=“text” name=“afield1”>
  - <input type=“button” value=“Click me”>
  - Type=“textarea” is for more than one line of text.
    - Like the CoWeb edit text.
  - Type=“radio” is for radiobuttons.

Simple Form

```
<html>
<head>
<title>Simplest Form in HTML</title>

</head>
<body>
<h1>A Simple Heading</h1>
<p>This is a very simple web page.</p>
<form>
<input type="text" name="afield">
<input type="button" value="Click me">
</form>
</body>

</html>
```



A Simple Heading

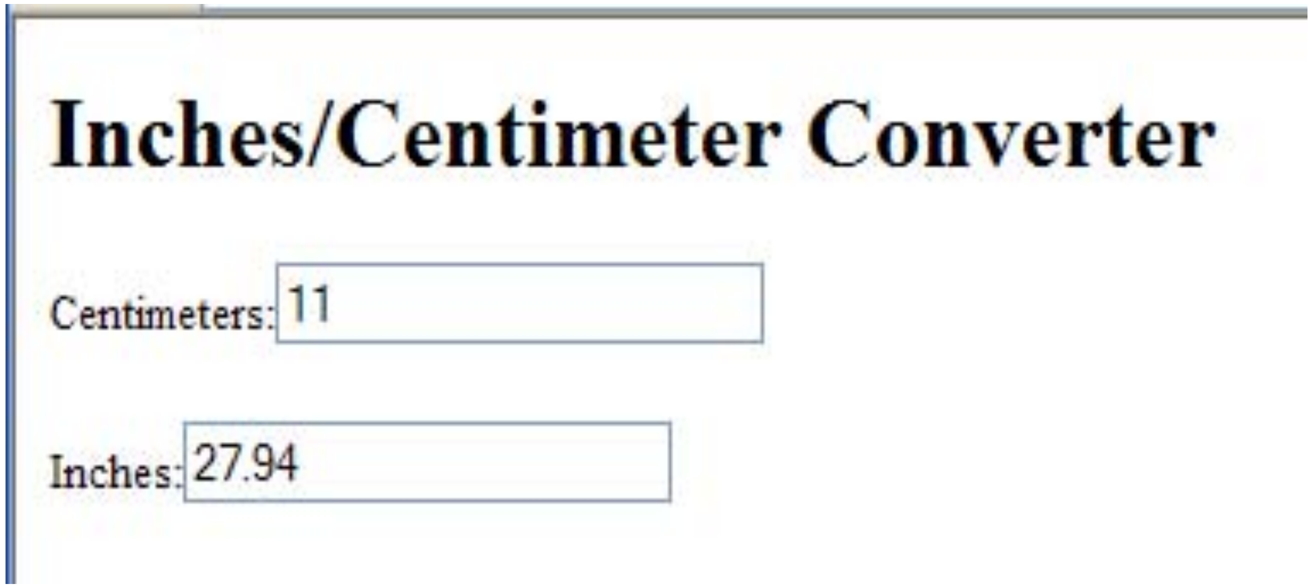
This is a very simple web page.

Forms and CGI Scripts

- Forms can also point to particular URLs
  - **Form URLs are typically CGI Scripts**
  - **CGI Scripts are programs (written in Perl or Python) that will process the form, which will be passed in as a parameter.**
- We can also do processing of form input completely from within JavaScript.

Inches/Centimeter Converter

```
<body>
<h1>Inches/Centimeter Converter</h1>
<form>
<p>Centimeters:<input type="text" name="cm"
onchange="this.form.inches.value=this.value / 2.54"></p>
<p>Inches:<input type="text" name="inches"
onchange="this.form.cm.value = this.value * 2.54"></p>
</form>
</body>
```



Inches/Centimeter Converter

Centimeters: 11

Inches: 27.94

Doing Multimedia in JavaScript

- It’s possible to do multimedia in JavaScript, but it’s not like in Python.
  - **We can’t control pixels or samples.**
- Most common way to do JavaScript is through plugins.
  - **Like Apple QuickTime, RealVideo, Netscape LiveAudio**
- Can do some simple animations from JavaScript.

Animated Motion in JavaScript

- There is a setInterval() function that can make a JavaScript function run at regular intervals.
  - **We use that to schedule motion to occur.**
  - **However, most cool motion in pages is done with Flash.**
- Divisions (<div></div>) can be controlled with styles, that can have positions.
- We then make a function to adjust the position of the division.

Animated Motion in JavaScript

```
<html>
<head>
<title>The Simplest Possible Web Page</title>
<style>
#barb { position: absolute; left:0; top: 0; }
</style>
<script>
function drift()
{
   var object = document.all.barb.style;
   object.pixelTop = object.pixelTop + 5;
   object.pixelLeft = object.pixelLeft + 5;
}
</script>
</head>
<body ONLOAD="setInterval('drift()', 100)">
<h1>A Simple Heading</h1>
<p>This is a very simple web page.</p>
<div id="barb">
<p><image src="mediasources/barbara.jpg"  />
</p>
</div>
<p>Here is some more text, just to make the document larger and make
it obvious where the picture is drifting.
</body>

</html>
```

Why JavaScript?

- To do simple processing from within HTML.
- To control plugins.

Bottom-up or Top-down?

- Is Javascript more suited to one or the other?
- Why?

JavaScript vs. Python

	Python	**Javascript**
write code	*JES*	*text editor*
interpret/ execute	*JES*	*browser (IE, Firefox, ...)*
context	*execute in command area*	*in HTML only*
variables	*just use*	*declare*
blocks	*“:” + indent*	*“{“ + “}”*
termination	*new line*	*“;”*

JavaScript vs. Python

- JavaScript’s syntax is much like other programming languages.
- JavaScript can’t do everything that Python can.
- Python is a more full-featured programming language.
- But Python can’t be embedded inside of HTML.
    - **(Well, not cross-platform. It can on Windows with a Python plugin for your browser.)**

Questions?

Coming Attractions

- Friday:
  - Mind reading web page(s) due 2:00 PM
    - can be more than one page
    - use HTML & Javascript
    - EXTRA: layout page using FrontPage or Dreamweaver; try NewMediaCenter in Torgersen
  - lab: objects in Python (using turtle graphics), DrJava