

CS1315: Introduction to Media Computation

Web Pages that Interact

Contents

- Javascript
 - **Using Javascript to make page contents vary**
 - **A taste of a new programming language**
- Using Javascript to respond to the user
- How OSCAR and Ebay work

What do other languages look like?

- We call the language “grammar” its *syntax*
- Python is a fairly traditional language in terms of syntax.
 - **Languages like Scheme, Lisp 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.**

JavaScript

- JavaScript is meant to be a *scripting* language, like Python.
 - **Scripting languages are good at solving 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 & computer.**
 - **We call that *client side* JavaScript.**

Preview: Inserting Javascript into HTML

```
<p>This is a very simple  
web page.</p>  
<p><img src =  
"mediasources/barbara.jp  
g" />  
</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

JavaScript syntax: Variables

- Variables must be declared before use.
 - **You can't just say:**
`a = 12`
 - **You can either say:**
`var a = 12;`
 - **Or:**
`var a;
a = 12`
- In other languages (like Java), you might also declare the variable's *type*
`int a = 12;`

JavaScript syntax: Blocks

- Blocks are delimited with curly braces.

```
function test()           like def  
{                         begins a  
  document.writeln("This is a test");  
}                         ends a  
                           block
```

JavaScript syntax: Individual statements

- Lots of differences:
 - **function** instead of **def**
 - **End lines with semicolons “;”**
 - (and statements can have line breaks in the middle of them.)
 - **The for statement is numeric (mostly) and has different parts to it.**
 - **You use write or writeln instead of print**
- But they're mostly detailed changes.
 - **The basic operation of JavaScript is similar to Python.**

JavaScript is all about objects

- Just about every “function” is really a method.
- For example, there is no global **print**.
- 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 belonging to the HTML *document* itself.**

Embedding JavaScript inside HTML

- JavaScript can be inside 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>`/`</body>` tags to execute the scripts.**

Like the upper area of JES

A bit like the lower area of JES

Our Simple Web Page

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

A Simple Heading

This is a very simple web page.



Adding some simple JavaScript

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD
HTML 4.01 Transition/EN"
"http://www.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.



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>
<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 insertHeading()
{
  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>
</p>
<script> insertHeading() </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>
</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 and

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 (but are also like this in C, C++ and Java)
 - && is AND
 - || is OR

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.

Contents

- Javascript
 - **Using Javascript to make page contents vary**
 - **A taste of a new programming language**
- Using Javascript to respond to the user
- How OSCAR and Ebay work



Using dialogs in JavaScript

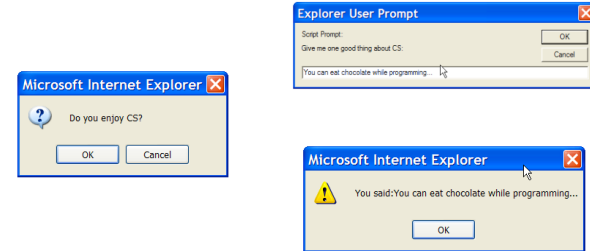
```
function check()
{
  var agree = false;
  agree = confirm("Do you enjoy CS?");
  if (agree)
    notes = prompt("Give me one good thing about CS.");
  if (!agree)
    notes = prompt("Why don't you like CS?");
  alert("You said:" + notes);
}
...
<script> check() </script>
</body>
</html>
```

agree will be true or false.

! agree is *not* agree.

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

What happens when this runs



Running when 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

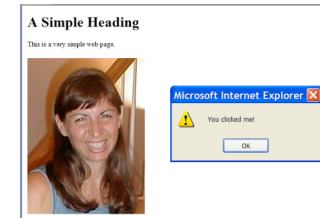
- The problem
 - **The previous program runs when the page loads.**
 - **The user sees nothing at all until they go to your page and then these dialog boxes happen.**
 - **Not what we want!**
- The solution
 - **The key to responding to users are *events*.**
 - **Events are actions taken by the user that can be *caught* by your program in JavaScript.**
 - **Events are happen when the user types a key, moves the mouse, clicks the mouse, etc.**

Events in JavaScript

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

Catching an Event

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



■ onClick: "When mouse is clicked, do this...."

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>
```

Contents

- Javascript
 - Using Javascript to make page contents vary
 - A taste of a new programming language
- Using Javascript to respond to the user
- How OSCAR and Ebay work



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.**
 - **`type = "radio"` is for radio buttons.**

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>
```



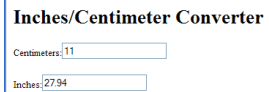
Inches/Centimeter Converter

```
<body>
<h1>Inches/Centimeter Converter</h1>
<h3>Enter a value and tab</h3>

<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>
```



Forms and CGI Scripts

- That example processed form input completely within JavaScript.
 - **OK for simple computations**
 - **But we often want to do more complicated things like query or update a database (e.g. An OSCAR course registrations)**
- Forms can also point to particular URLs
 - **Form URLs are typically CGI Scripts**
 - **CGI Scripts are programs (written in languages like Python, Perl, PHP) that will process the form, which will be passed in as a parameter.**

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.)**
- Python can be used on the server side for web pages. PSP or python server pages.