Functions

INFO/CSE 100, Spring 2006
Fluency in Information Technology

http://www.cs.washington.edu/100
Readings and References

• Reading
  » Fluency with Information Technology
    • Chapter 19, Bean Counter
    • Chapter 20, Abstraction and Functions

• Other References
  » W3Schools JavaScript tutorial
    http://www.w3schools.com/js/default.asp
  » W3Schools JavaScript HTML DOM Objects
    http://www.w3schools.com/js/js_obj_htmldom.asp
  » Mozilla Browser
    http://www.mozilla.org/
Midterm #1

![Graph showing the relationship between percentage score and grade point for Midterm 1. The graph has a linear trend, indicating that higher scores correspond to higher grade points.]
Functions

A *function* is a way to bundle a set of instructions and give them a name so that you can reuse them easily.

Functions have a specific layout:

- `<name>` ← the function name is an identifier
- `<parameter list>` ← list of input variables for the function
- `<statements>` ← the statements do the work

```plaintext
function <name> ( <parameter list> ) {
  <statements>
}
```
Example Function

Write a simple function to compute the Body Mass Index when the inputs are in English units (i.e., US units)

```javascript
// Calculate Body Mass Index in English units
// weight in pounds
// height in inches
// returns body mass index

function bmiE(weightLBS, heightIN) {
    var heightFt = heightIN / 12; // convert to feet
    return 4.89 * weightLBS / (heightFt * heightFt);
}
```
Develop the function

First, make sure you understand what you want the function to do and how it will accomplish the task.

```cpp
// Calculate Body Mass Index in English units
// weight in pounds
// height in inches
// returns body mass index

function name(parameter list) {
    statements
}
```
Pick a name for the function

Function names are identifiers
  » start with a letter
  » should have a fairly obvious meaning
  » should not be one of the Javascript reserve words

// Calculate Body Mass Index in English units
// weight in pounds
// height in inches
// returns body mass index

function bmiE(parameter list) {
  statements
}

Pick the parameters

Parameter names are also identifiers

» these are the variable names that your function will use when it is performing its calculations

» should have a fairly obvious meaning

```javascript
// Calculate Body Mass Index in English units
// weight in pounds
// height in inches
// returns body mass index

function bmiE(weightLBS, heightIN) {
    statements;
}
```
Functions without Parameters!

- Function do not have to have parameters
  » But we still need to include the parentheses

```javascript
// Print out Greeting
// Typical Greeting is "Hello World"

function giveGreeting() {
    document.write("Hello World!");
}
```
Write the function body

The function body includes whichever statements are required to implement the desired capability.

```javascript
// Calculate Body Mass Index in English units
// weight in pounds
// height in inches
// returns body mass index

function bmiE(weightLBS, heightIN) {
  var heightFt = heightIn / 12; // convert to feet
  return 4.89 * weightLBS / (heightFt * heightFt);
}
```
A Simple Testing Template

```html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Body Mass Index</title>
<script type="text/javascript">
// Figure Body Mass Index in English units
function bmiE( weightLBS, heightIn ) {
  var heightFt = heightIn / 12; // Change to feet
  return 4.89 * weightLBS / (heightFt * heightFt);
}
</script>
</head>
<body>
<p>This page provides a simple body mass index calculator.
Normal weight corresponds to a BMI of 18.5-24.9</p>
<script type="text/javascript">
  document.writeln("<br>bmiE(100,72): "+bmiE(100,72));
  document.writeln("<br>bmiE(150,72): "+bmiE(150,72));
  document.writeln("<br>bmiE(175,72): "+bmiE(175,72));
  document.writeln("<br>bmiE(200,72): "+bmiE(200,72));
</script>
</body>
</html>
```
Try the function and see how it works

This page provides a simple body mass index calculator. Normal weight corresponds to a BMI of 18.5-24.9

bmiE(100, 72): 13.583333333333333
bmiE(150, 72): 20.375
bmiE(175, 72): 23.770833333333333
bmiE(200, 72): 27.166666666666666
Fancy Function Features

```html
<head>
<title>Body Mass Index</title>
<script type="text/javascript">
// Calculate Body Mass Index in English units
// weight in pounds
// height in inches
// returns body mass index
function bmiE(weightLBS, heightIN) {
    var heightFt = heightIn / 12; // convert to feet
    return 4.89 * weightLBS / (heightFt * heightFt);
}
</script>
</head>

<script> in <head> location, comments, keywords, formal parameters, curly brackets, parentheses, operators, expressions, assignment statement, return statement, semi-colon
```
Using Fancy Functions

This page provides a simple body mass index calculator. Normal weight corresponds to a BMI of 18.5-24.9

```javascript
document.writeln("<br>bmiE(100,72): " + bmiE(100,72) + ");
document.writeln("<br>bmiE(150,72): " + bmiE(150,72) + ");
document.writeln("<br>bmiE(175,72): " + bmiE(175,72) + ");
document.writeln("<br>bmiE(200,72): " + bmiE(200,72) + ");
```

<script> in <body> location, document, writeln function call, strings, string concatenation, bmiE function call, arguments (aka actual parameters)
Global or Local?!?

• Scope of a variable describes where and when it can be referenced
  
  » Local variables are only known inside of a function (curly braces)
  
  » Global variables are know by all the Javascript inside of <script> </script> pairs

```javascript
// Calculate Percentage of Study Hours/Week
// time in hours
// returns hours
var days = 7;
function calculateStudyHrs(time) {
    var totalHrs = 24 * days;
    return time/totalHrs;
}
```
Comments on Debugging

• Debugging JavaScript can be hard
  » The browsers all implement things a little differently, particularly old browsers
  • upgrade if you are using something old!
Use the W3Schools TryIt Editor

This page provides a simple body mass index calculator. Normal weight corresponds to a BMI of 18.5-24.9.

- bmiE(100,72): 13.58333333333332
- bmiE(150,72): 20.375
- bmiE(175,72): 23.77083333333332
- bmiE(200,72): 27.16666666666664

Edit the text above, and click on the button to see the result.

http://www.w3schools.com/js/tryit.asp?filename=tryjs_text
Display results using `alert(...)`

Use the `alert("This is a message")` function
Use an editor that helps you

The TextPad editor helps you by doing syntax highlighting.
Display results using writeln(...)
Use a browser that helps you

• All browsers try to be forgiving of errors, which means that they generally don't produce a lot of error messages
  » use a browser that helps you debug like Mozilla
enable Mozilla JavaScript Console
The Mozilla JavaScript console helps you by showing good error messages.
Graphical User Interfaces (GUIs)

We can also use JavaScript to create Graphical User Interfaces.
A Graphical User Interface provides an intuitive way to control a program instead of having to memorize commands.

- text fields with labels to request user entry
- text fields with labels to display results
- buttons to command action
- radio buttons and checkboxes to set conditions
A simple example

This GUI has several simple controls.

- Two buttons to control the results
- One text field to display the results
- One pair of radio buttons to control the display
- One button to reinitialize

A simple example

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Simple Sample GUI</title>
<script type="text/javascript">
javascript function code
</script>
</head>

<body>

HTML form layout and specification

</body>
</html>
Layout of the GUI

- The layout of the page is controlled with HTML in the body of the page

```
<body>
  HTML form layout and specification
</body>
</html>
```

- The layout and controls are provided using new tags
  
  » <form name="buttonForm">
  »   <button type="button" ...
  »   <input type="text" …
  »   <input type="radio" …
  »   <button type="reset" …
<form>

- HTML forms provide a way for the user to enter data into a web page
  - A form can contain several different types of entry, control, and display elements
  - The data in a form can be passed back to the web server, or it can be processed locally on the client
    - All of our forms will be processed locally

- A form is defined with the <form id="dmvForm"> tag
  - The form has various attributes like id, so we can refer to it and its elements later
  - The form contains various elements like <input> and <button>
<button type="button" ...>

<form>
<button type="button"
    onclick="setResults('good results')">Good Results</button>
<button type="button"
    onclick="setResults('bad results')">Bad Results</button>
</form>

• a <button> can have one of three types
  » type “button” is used locally
  » type “submit” sends data back to the server
  » type “reset” re-initializes the form

• the value of the “onclick” attribute is some JavaScript code, in this case a call to the function
  setResults(string)
<input type="text" ...>

<form>
<b>Result:</b>
<input type="text" value="nada"readonly id="resultField">
<br>
<input type="radio" name="case" id="radioLC" checked onclick="setResults(document.getElementById('resultField').value)">Lowercase
<input type="radio" name="case" id="radioUC" onclick="setResults(document.getElementById('resultField').value)">Uppercase
<br><button type="reset">Reset</button>
</form>

- an <input> with type="text" is used for user input and program output
- value="nada" sets the initial (and reset) value
- readonly means that the user cannot set the value, only the script can set the value
- id="resultField" gives us a way to identify this particular control in our JavaScript
an `<input>` with `type="radio"` allows the user to select one of several choices
name="case" identifies all the buttons in the same group (only one will be selected at a time)
onclick attribute gives the JavaScript to execute when the user clicks this button
id="radioLC" gives us a way to identify this particular control in our JavaScript
• a `<button>` with type="reset" resets all the other controls in the same form to their original values
Events Cause Processing

• After drawing a page, the browser sits idle waiting for something to happen … when we give input, we cause events

• Processing events is the task of a block of code called an event handler
  » The code to execute is identified in the tag using the appropriate attribute
  » There are many event types
    • onClick, onChange, onMouseOver ...
request processing of an event

```html
<form>
  <button type="button"
    onclick="setResults('good results')">Good Results</button>
  <button type="button"
    onclick="setResults('bad results')">Bad Results</button>
</form>
```

- the **onclick** attribute defines some JavaScript to call when the button is clicked
- in this case, the code is a call to the `setResults(string)` function defined in the page `<head>`
- the appropriate string value is supplied to the `setResults(string)` function and then the function executes
process a button’s onclick event

```javascript
function setResults(resultString) {
    var tempString = resultString;
    if (document.getElementById("radioLC").checked) {
        tempString = tempString.toLowerCase();
    } else if (document.getElementById("radioUC").checked) {
        tempString = tempString.toUpperCase();
    }
    document.getElementById("resultField").value = tempString;
}
</script>
```

- the `setResults(string)` function is called by several event processors
- in every case, it takes the string that it is given, decides if upper or lower case is desired, and sets the `resultField` accordingly
setResults(resultString)

```javascript
<script type="text/javascript">
function setResults(resultString) {
    var tempString = resultString;
    if (document.getElementById("radioLC").checked) {
        tempString = tempString.toLowerCase();
    } else if (document.getElementById("radioUC").checked) {
        tempString = tempString.toUpperCase();
    }
    document.getElementById("resultField").value = tempString;
}
</script>
```

parameter variable, local variable, if/else statement, field reference, call to toLowerCase() function