Graphical User Interface

INFO/CSE 100, Autumn 2004

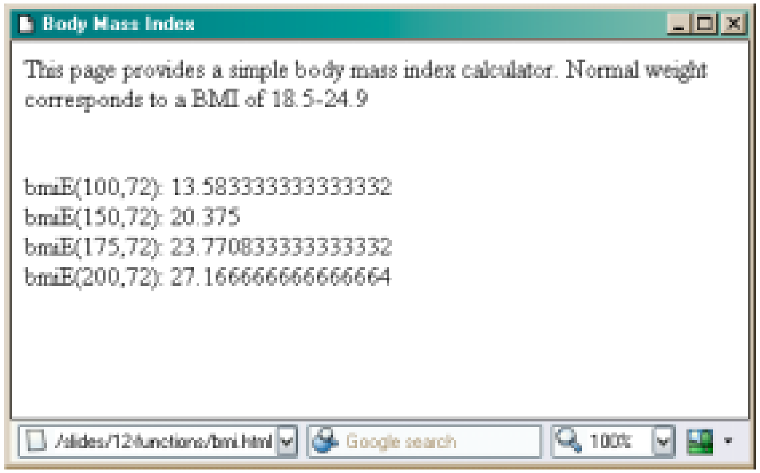
Fluency in Information Technology

http://www.cs.washington.edu/100

Readings and References

- Reading
  - » *Fluency with Information Technology*
    - Chapter 19, A JavaScript Program
- Other References
  - » W3C HTML 4.01 Specification
    - http://www.w3.org/TR/html401/
  - » W3Schools HTML 4.01 Reference
    - http://www.w3schools.com/html/html_reference.asp
  - » W3Schools JavaScript HTML DOM Objects
    - http://www.w3schools.com/js/js_obj_htmldom.asp

So far, we've just used JavaScript to calculate and create “normal” HTML



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

```
bmiE(100,72): 13.583333333333332
bmiE(150,72): 20.375
bmiE(175,72): 23.770833333333332
bmiE(200,72): 27.166666666666664
```

We defined the function in <head>

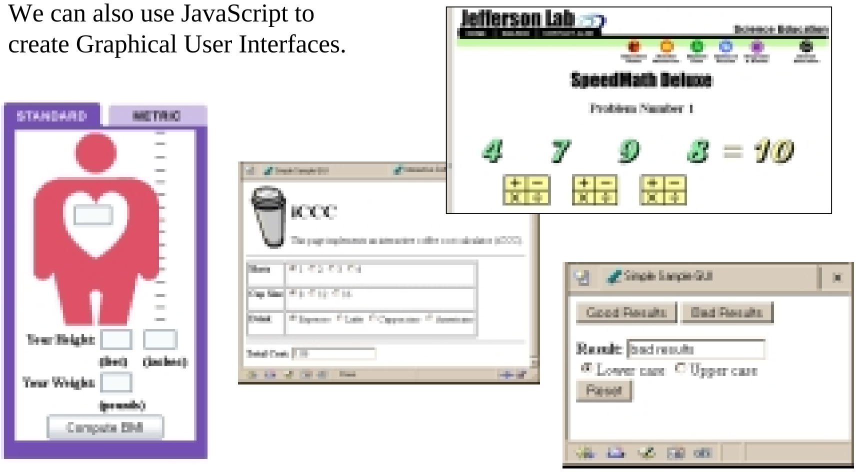
```
<head>
<title>Body Mass Index</title>
<script type="text/javascript">
// Figure Body Mass Index in English units
function bmiE( weightLBS, heightIn ) {
  var heightFt = heightIn / 12; // Change to feet
  return 4.89 * weightLBS / (heightFt * heightFt);
}
</script>
</head>
```

We used the function in <body>

```
<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>
```

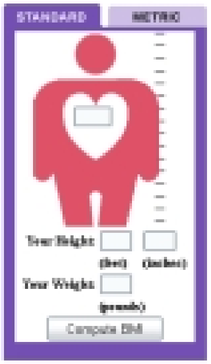
Graphical User Interfaces (GUIs)

We can also use JavaScript to create Graphical User Interfaces.



GUIs

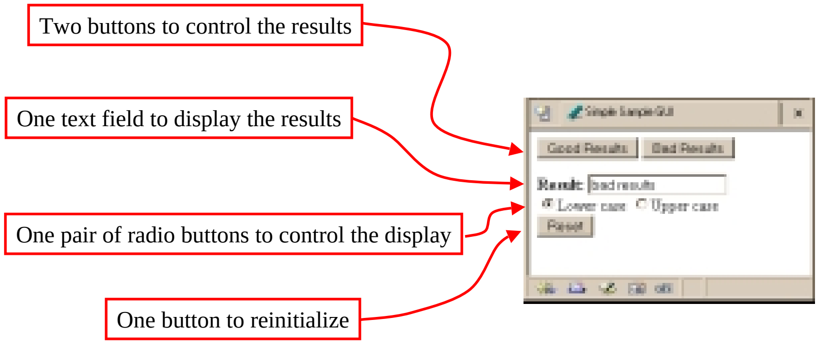
A Graphical User Interface provides an intuitive way to control a program instead of having to memorize commands



- 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

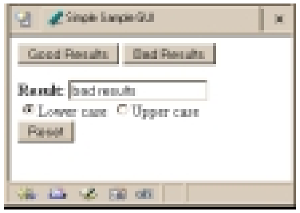
One button to reinitialize

http://www.cs.washington.edu/education/courses/100/04au/slides/13-gui/gui.html

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 id="buttonForm">
  - » <button type="button" ...
  - » <input type="text"
  - » <input type="radio"
  - » <button type="reset"

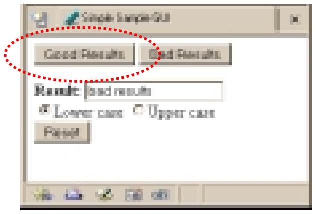
<form>

- 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 processed locally
- A form is defined with the <form> ... </form> tag
  - » 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>
```

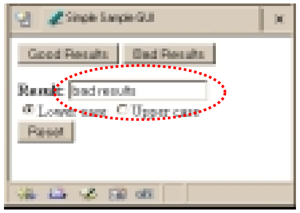
- a <button> can have one of three types
  - » type “button” is used locally
  - » type “ submit” sends data back to the server
  - » type “reset” re-initializes the form
- the value of the “onclick” attribute is some JavaScript code, in this case a call to the function `setResults(`*`string`*`)`



<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)">Lower case
<input type="radio" name="case" id="radioUC"
  onclick="setResults(document.getElementById('resultField').value)">Upper case
<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

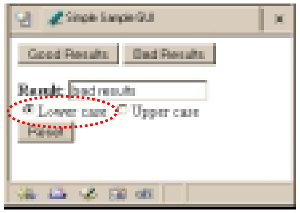


Result: bad results
● Lower case ○ Upper case

<input type="radio" ...>

```
<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)">Lower case
<input type="radio" name="case" id="radioUC"
  onclick="setResults(document.getElementById('resultField').value)">Upper case
<br><button type="reset">Reset</button>
</form>
```

- 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

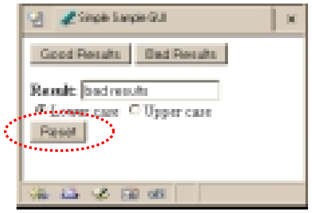


Result: bad results
● Lower case ○ Upper case

<button type="reset" ...>

```
<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)">Lower case
<input type="radio" name="case" id="radioUC"
  onclick="setResults(document.getElementById('resultField').value)">Upper case
<br><button type="reset">Reset</button>
</form>
```

- a `<button>` with `type="reset"` resets all the other controls in the same form to their original values



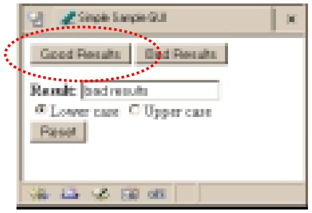
Result: bad results
● Lower case ○ Upper case

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

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



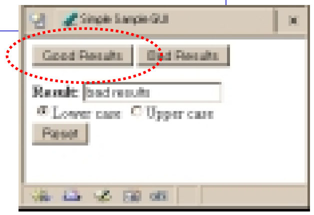
Result: bad results

● Lower case ○ Upper case

- 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

```
<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>
```



Result: bad results

● Lower case ○ Upper case

- 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)

```
<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