CS 416
Web Programming

Javascript continued

Dr. Williams
Central Connecticut State University

Agenda

- Javascript
  - Debugging
  - Mixed types
  - Math
  - Events
  - Functions

Syntax errors

- an error in the format of an HTML or JavaScript statements is known as a **syntax error**
  - some syntax errors are ignored by the browser e.g., misspelling an HTML tag name
  - most JavaScript syntax errors will generate an error message

```
document.write("This example is illegal since the
                                 string is broken across lines");
```

yields: Error: unterminated string literal

```
document.write("The value of x is " x);
```

yields: Error: missing) after argument list

Viewing errors

- In Chrome to view an error
  - right click on the page
  - choose inspect element
  - select console in the shown window
- In IE
  - Double click the exclamation point in the bottom left
  - click show details
- In Firefox
  - Click Firefox in top left
  - Choose web developer, error console

Debugging Javascript

- For syntax errors development environments such as NetBeans can be useful
- For run errors where you can set breakpoints
- For Firefox install the FireBug plugin
- For Chrome click the settings icon, then go to tools|Developer tools then click Sources

Javascript and HTML

- Can be interspersed
  HTML
  javascript
  HTML
  javascript
  HTML
- Incorporating HTML with dynamic elements
  - …</script>HTML<script>…
  - document.write()
- Output of javascript treated as plain HTML by browser

Data types

- Each unit of information belongs to the general category **data type**
- Defines the way information can be handled by a program
- Javascript data types
  - String
  - Number – parseInt(), parseFloat()
  - Booleans – Boolean()

Data types (cont)

- Each data type associated with set of predefined operations
- Operations can be common syntax with own meaning or completely different
  - Ex.
    - "1" + "2" = "12"
    - 1 + 2 = 3
    - **"1" + 2 = "12"**

Working with numbers

- x = prompt("x:",""); //sets x to string
- // assume prompt set x to "5"
- y=x+x; // y = "55";
- z=**parseInt**(x)+**parseInt**(x); // z=10

Or

- y=parseInt(x);
- z=y+y;

Data assignment

- Unlike many languages, **variable types** can be overwritten many times through course of execution
- Tracing assignments required to understand code
- In evaluating expressions, precedence matters
  - Expressions enclosed in "()" evaluated first
  - Multiplication/division applied before plus/minus

Mixed types

```
Num1 = prompt("number:","2");
    #num1="2"

Num2 = parseInt(num1);
    #num1="2",num2=2

Num3 = num1+num2+.5;
    #num1="2",num2=2,num3="22.5"

Num3 =
   parseFloat(Num3)+parseInt(num1)
    #num1="2",num2=2,num3=24.5
```

More mixed types

5 +” plus “ + 6 + 4 * 5 + (7*5 +”5”)

5 +” plus”+((6 +4)*5+7)*5 +”5”

Math library

Function	Inputs	Description
Math.sqrt	One number	Returns square root of number
Math.max	Two numbers	Returns larger of two numbers
Math.min	Two numbers	Returns smaller of two numbers
Math.abs	One numbers	Absolute value of number
Math.floor	One numbers	Returns the floor (Round down) of the number
Math.ceil	One numbers	Returns the ceiling (Round up) of the number
Math.round	One numbers	Traditional rounding
Math.pow	Two numbers	Takes 1st number raises it to the power of the 2nd number. Math.pow(x,y) = x^y

Random numbers

- Math.random()
  - Generates psuedo-random number between [0,1)
  - Useful for random chance involved with pages
    - Ex. Display one ad 25% of the time

Examples

Function can be used like a number

```
2*Math.random()+1

Math.floor(2*Math.random()+1)
```

- Rather than [0,1) can also get random integers [0,N]:

```
Math.floor((N+1)*Math.random())
```

(useful for next HW assignment)

Your turn

- Create a page with the JavaScript to do the following:
- Prompt the user for a height and width
- Write to the page on separate lines:
  - **`Perimeter:  34`**
  - **`Area:  24.2`**
  - **`Rounded area: 24`**

Event-driven Pages

- Basis of making web interactive to user actions
  - Mouse click
  - Mouse over
  - Form entry
- Pages that respond to user actions are known as *event-driven* pages
  - JavaScript can be combined with HTML elements such as buttons, text fields, and text areas to produce event-driven pages

Event handlers

- An *event handler* is an HTML element that can be programmed to respond to a user’s actions
  - the simplest event handler is a button
  - a button can be associated with JavaScript code that will execute when the button is clicked

HTML Page Events

- onload – event occurs after everything on page has loaded
- onbeforeunload – event occurs when user leaves page (back, new url, clicking link) Limited in Chrome/Firefox
- onunload – similar to onbeforeunload but not supported by Chrome or Firefox
- onresize – occurs when window is resized
- onscroll – occurs when user scrolls up/down/right/left

Alert

- Similar to prompt but used for messages
- Displays string with only option being to click ok

```
alert(“message”)
```

Acting on events

- Page level events specified on body tag
- Event can specify javascript functions to be executed

```
<body onload="alert('page loaded');">
```

- Multiple events can be programmed

```
<body onload="visitor=prompt('name','');
  alert('Thanks for visiting'+visitor);">
```

Button events

general form of a button element:

```
<input type="button"
value="BUTTON_LABEL"
onclick="JAVASCRIPT_CODE" />
```

- the TYPE attribute of the INPUT element identifies the element to be a button
- the VALUE attribute specifies the text label that appears on the button
- the ONCLICK attribute specifies the action to take place
  - any JavaScript statement(s) can be assigned to the ONCLICK attribute
  - this can be (and frequently is) a call to a JavaScript function

Button example

```
<input type="button"
   value="Next"
   onclick="alert('last page!');" />
```

- the predefined `alert` function displays a message in a new window
  - here, the message 'last page!' is displayed at the click of the button
- a string can be denoted using either double("…") or single ('…') quotes
  - here, single quotes must be used to avoid confusion with the ONCLICK quotes

Multiple actions

- `<input type="button" value="Random number" onclick="luckyNum=Math.random();alert(luckyNum)”; />`
  – Event can have multiple actions in line
  – Basic syntax remains the same.

Text Box event handlers

- a button provides a simple mechanism for user interaction in a Web page
  - by clicking the button, the user initiates some action

- a *text box* is an event-handler that can display text (a word or phrase)
  - unlike an alert window, the text box appears as a box embedded in the page
  - text can be written to the box by JavaScript code (i.e., the box displays output)
  - Events - **`onkeyup`**`='changeSomething( this );'`

Text box element

general form of a text box element:

```
<input type="text" id="BOX_NAME"
   size="NUM_CHARS" value="INITIAL_TEXT" />
```

- the TYPE attribute of the INPUT element identifies the element to be a text box
- the ID attribute gives the element an identifier so that it can be referenced
- the SIZE attribute specifies the size of the box (number of characters that fit)
- the VALUE attribute specifies text that initially appears in the box

Using an element in Javascript

- To interact with an element in Javascript it must be referenced
  - specify the *absolute name* of the box
  - There are two general ways to reference: getting by id, and referencing within a form element
  - By id the general syntax is:
    ```
    document.getElementById('ELEMENT_NAME')
    ```
  - Within a form element the syntax is:
  - `document.FORM_NAME.ELEMENT_NAME`

Writing to text box element

- to display text in a text box, a JavaScript assignment is used to assign to its value attribute
  - as part of the assignment, must specify the *absolute name* or path of the box
  - `document.getElementById('BOX_NAME').value = VALUE_TO_BE_DISPLAYED;`

Reading from text box

- text boxes can also be used for receiving user input
  - the user can enter text directly into the box
  - that text can then be accessed by JavaScript code via the absolute name of the box

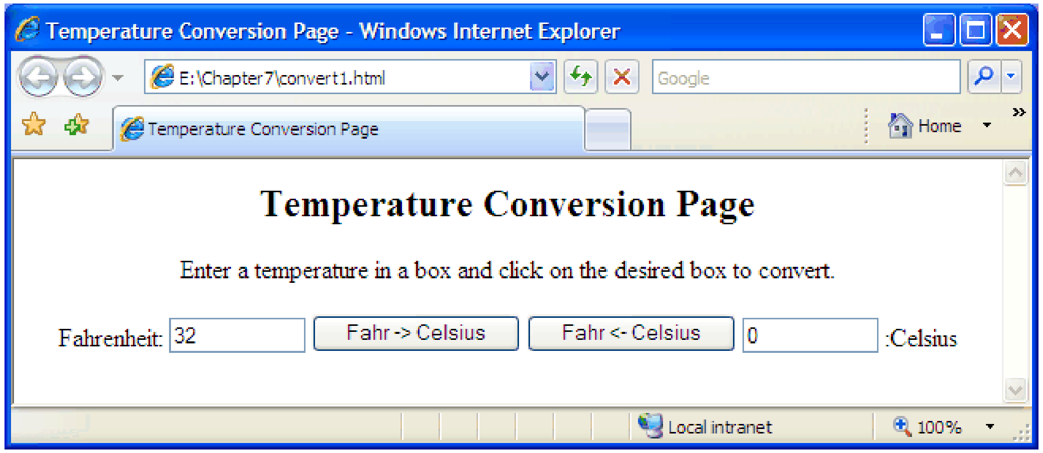
```
some_val = document.getElementById('BOX_NAME').value
```

Reading cont.

- note that the value retrieved from a text box is always a string
  - if the user enters a number, say 93, then the absolute name will access "93"
  - similar to `prompt`, you must use `parseFloat` to convert the string to its numeric value

Text box cont.

- Text boxes can be used for both entry as well as output


Temperature Conversion Page - Windows Internet Explorer

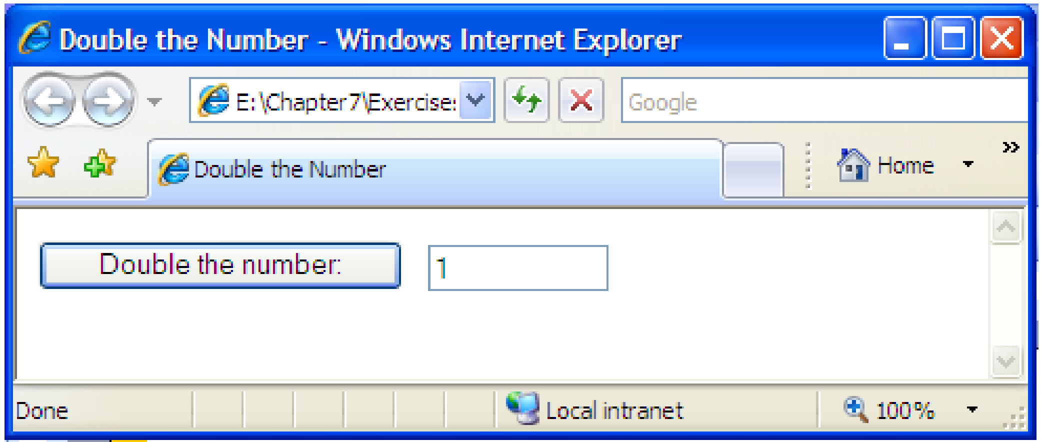
E:\Chapter7\convert1.html

Temperature Conversion Page

Enter a temperature in a box and click on the desired box to convert.

Fahrenheit: 32 [Fahr -> Celsius] [Fahr <- Celsius] 0 :Celsius

Local intranet 100%


Double the Number - Windows Internet Explorer

E:\Chapter7\Exercise

[Double the number:] 1

Done Local intranet 100%

Text areas

- a text area is similar to a text box but it can contain any number of text lines

```
<textarea name="TEXTAREA_NAME" rows=NUM_ROWS
cols=NUM_COLS wrap="virtual">
INITIAL_TEXT
</textarea>
```

  - the NAME attribute gives the element a name so that it can be referenced
  - the ROWS attribute specifies the height (number of text lines) of the area
  - the COLS attribute specifies the width (number of characters) of the area
  - the WRAP attribute ensures that the text will wrap from one line to the next

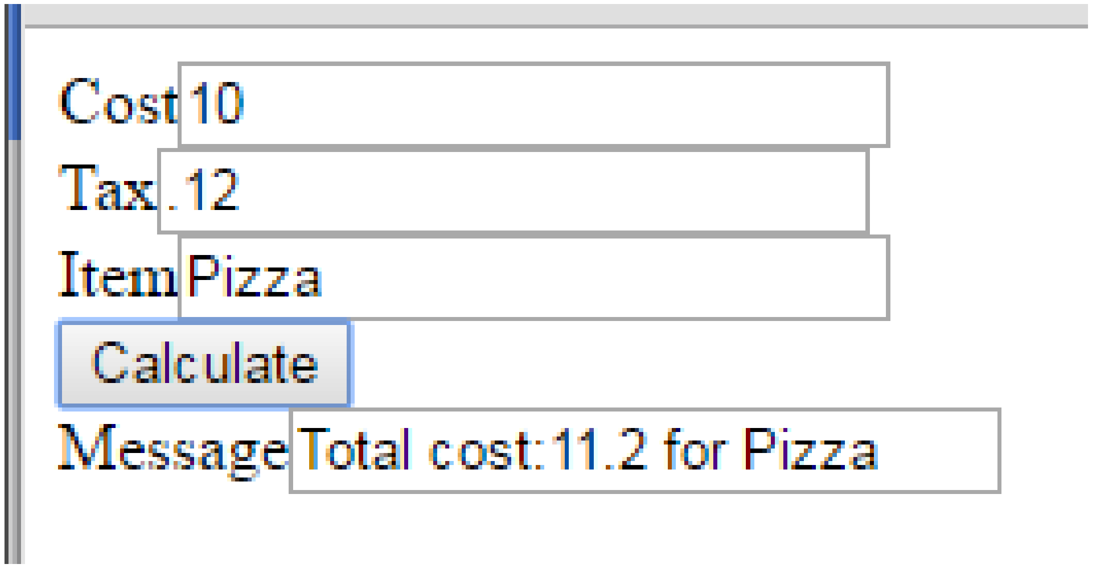
    instead of running off the edge of the text area

Text area

- Unlike a text box, opening and closing tags are used to define a text area
  - any text appearing between the tags will be the initial text in the text area
  - otherwise, the contents of a text area are accessed/assigned in the same way as a text box by using the "value" attribute

Write HTML/Javascript

Click calculate and outputs bottom message



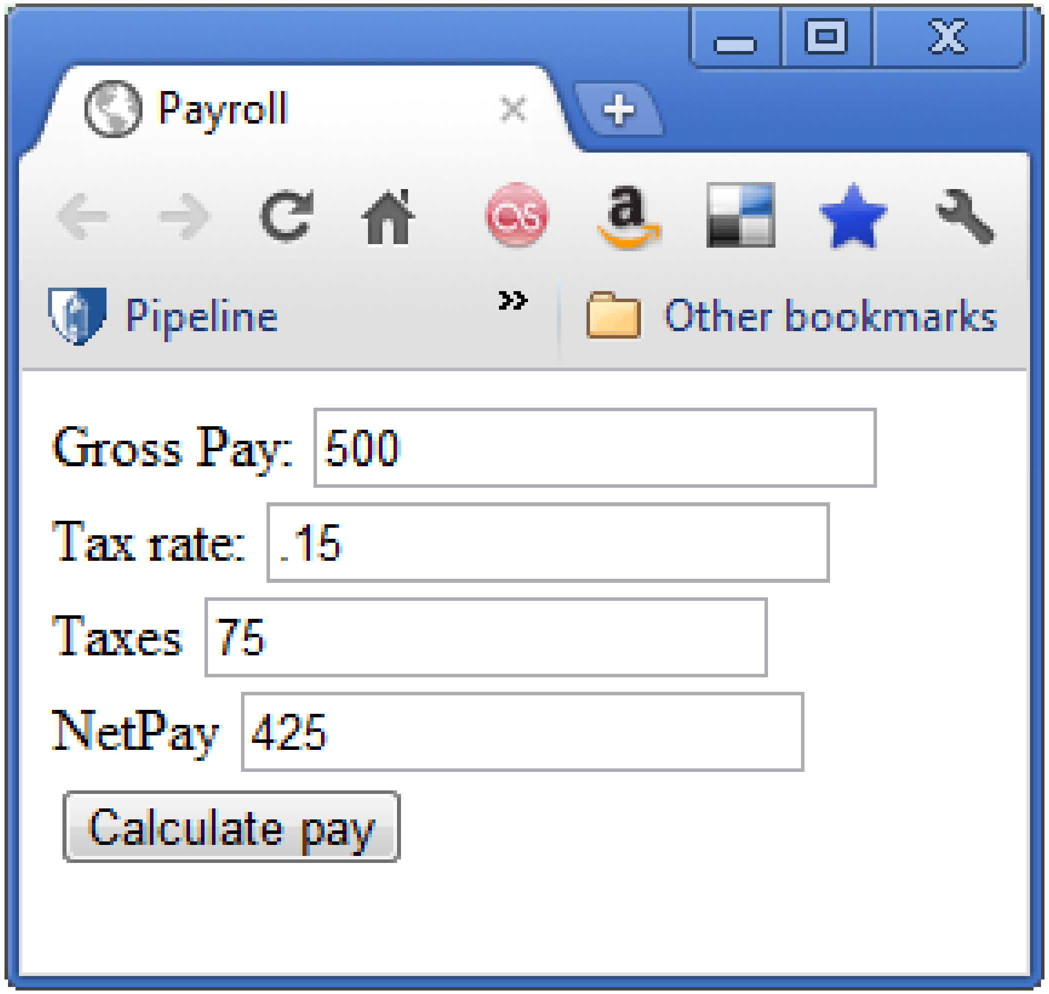
Cost 10

Tax .12

Item Pizza

Message Total cost:11.2 for Pizza

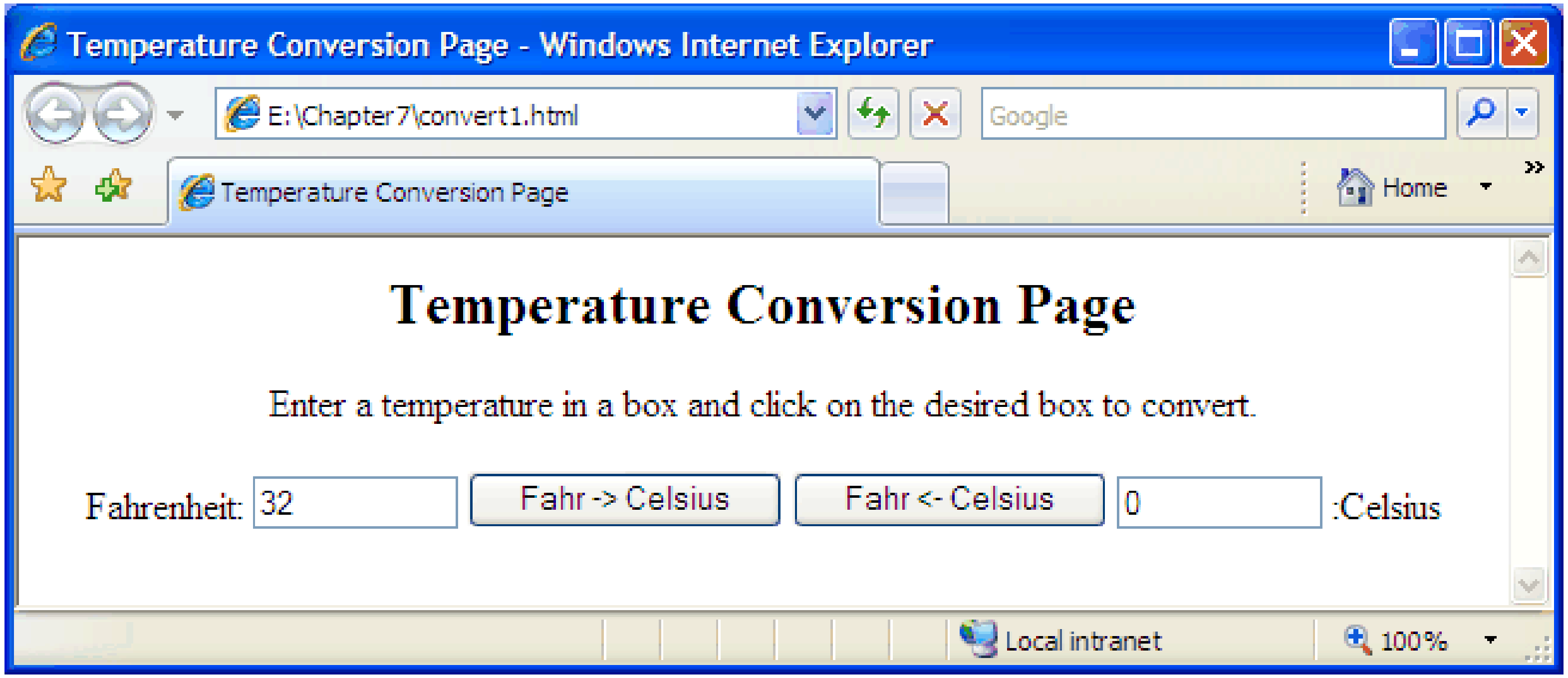
Write HTML/Javascript



Gross Pay: 500
Tax rate: .15
Taxes 75
NetPay 425

Temp conversion

- Convert temperature as user types


Temperature Conversion Page - Windows Internet Explorer

E:\Chapter7\convert1.html

Temperature Conversion Page

Enter a temperature in a box and click on the desired box to convert.

Fahrenheit: 32 [Fahr -> Celsius] [Fahr <- Celsius] 0 :Celsius

$$°C \times 9/5 + 32 = °F$$
$$(°F - 32) \times 5/9 = °C$$