

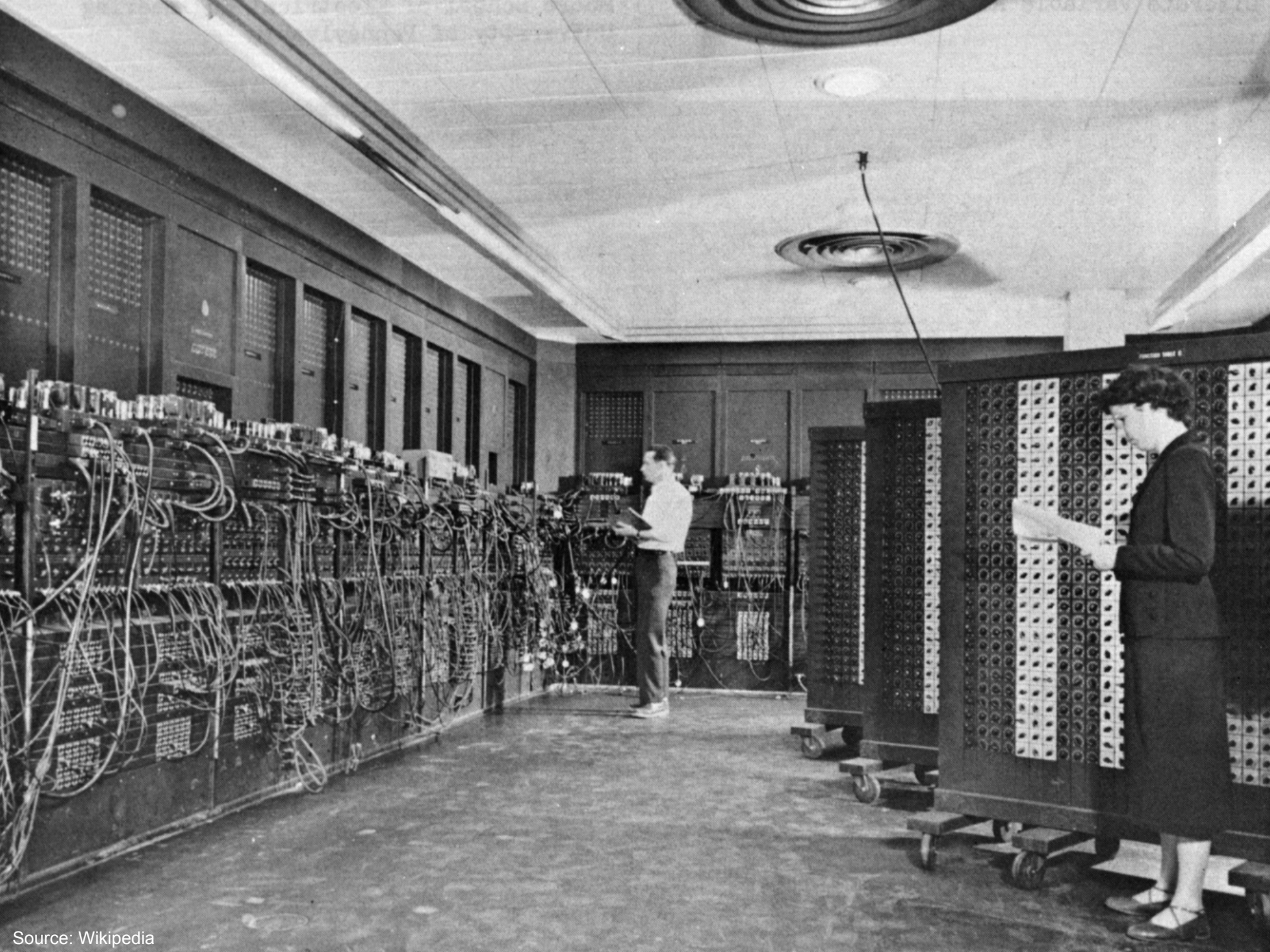
INFM 603: Information Technology and Organizational Context

# **Session 3: JavaScript - Structured Programming**



Jimmy Lin  
The iSchool  
University of Maryland

Wednesday, February 12, 2013



# Types of Programming

- Low-level languages

- Directly specifies actions of the machine
- Example: assembly language

- High-level languages

- Specifies machine instructions at a more abstract level
- Compiler/interpreter translates instructions into machine actions
- Example: JavaScript

# What's JavaScript?

- Programming language for the web
- Client-side, runs in the browser
- Provides programmatic access to the HTML page in which it's embedded (the DOM)
- Enables richly-interactive websites!

# What's a Document?

- Content
- Structure
- Appearance
- Behavior



Programming... is a lot like cooking!

# Data Types and Variables

- Data types = things that you can operate on
  - Boolean: true, false
  - Number: 5, 9, 3.1415926
  - String: “Hello World”
- Variables hold values of a particular data type
  - Represented as symbols (e.g., x)
  - How should you name variables?
- In JavaScript, var declares a variable
  - `var b = true;`          create a boolean b and set it to true
  - `var n = 1;`                create a number n and set it to 1
  - `var s = “hello”;`        create a string s and set it to “hello”

# Expressions & Statements

- Things that you can do:
  - `-x` reverse the sign of x (negation)
  - `6 + 5` add 6 and 5
  - `2.1 * 3` multiply two values
  - `"Hello" + "World"` concatenate two strings
- The simplest statements store results of expressions:
  - `x = 5` set the value of x to be 5
  - `x += y` `x = x + y`
  - `x *= 5` `x = x * 5`
  - `x++` increase value of x by 1
- In JavaScript, statements end with a semicolon (;)



# Cooking Analogy

- Data types are like?
- Variables are like?
- Statements are like?

# Sequence of Instructions



```
var a = 2;  
var b = 3;  
var c = a * b;
```

# Where does the JavaScript go?

```
<!DOCTYPE html>  
<html>  
<head>  
<meta charset=utf-8 />  
<title>My Title</title>
```

```
<script>
```

```
...
```

```
</script>
```

JavaScript in the header, processed  
before the page is loaded

```
<script src="code.js">
```

```
</script>
```

JavaScript in an external file,  
processed before the page is loaded

```
</head>
```

```
<body>
```

```
<script>
```

```
...
```

```
</script>
```

JavaScript in the body, processed as  
the page is loaded

```
</body>
```

```
</html>
```

# Temperature Conversion Demo

- A few useful statements:
  - `var t = prompt("message here", "default");`
  - `document.writeln("message here");`
  - `console.log("message here");`
  - `alert ("message here");`
- Tip: what if you want to have a quote inside a quote?
- Your turn:
  - Convert the temperature now Celsius to Fahrenheit

# Programming Tips

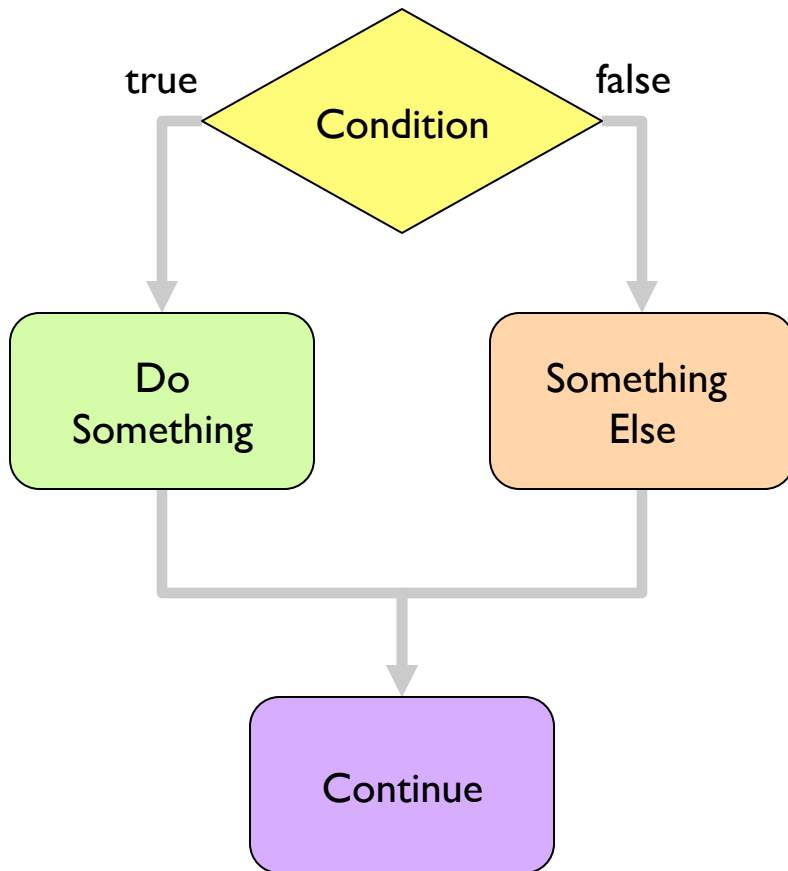
- Details are everything!
  - Careful where you place that comma, semi-colon, etc.
- Write a little bit of code at a time
  - Add a small new functionality, make sure it works, then move on
  - Don't try to write a large program all at once
  - If it doesn't work, revert back to previous version that worked
- Debug by outputting the state of the program
  - Simulate what you think the program is doing
  - Print out the value of variables using `document.writeln` or `console.log`
  - Is the value what you expected?
- Use the Chrome JavaScript console!

# Controlling Execution

- Conditional
- Loops

Programming... is a lot like cooking!

# Conditional



```
if (gender == "male") {  
    greeting = "It's a boy!";  
} else {  
    greeting = "It's a girl!";  
}
```

Note, the text in red is part of the “template” of the conditional

Note the indentation...

# Multiple if-else clauses

```
if ( expression ) {  
    ...  
} else if ( expression ) {  
    ...  
} else if ( expression ) {  
    ...  
} else {  
    ...  
}
```



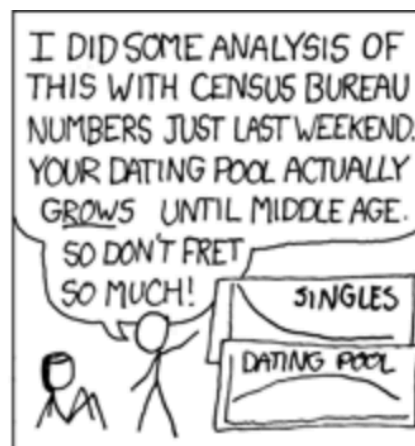
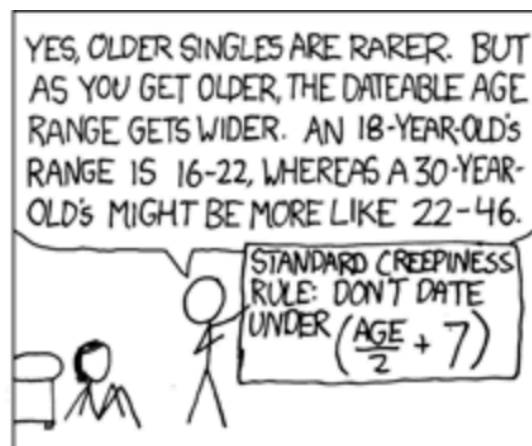
# Nested if-else clauses

```
if ( expression ) {  
    if ( expression ) {  
        ...  
    } else {  
        ...  
    }  
} else if ( expression ) {  
    ...  
} else if ( expression ) {  
    ...  
} else {  
    ...  
}
```

Note this is where indentation become important...

# Test Conditions: Boolean Expressions

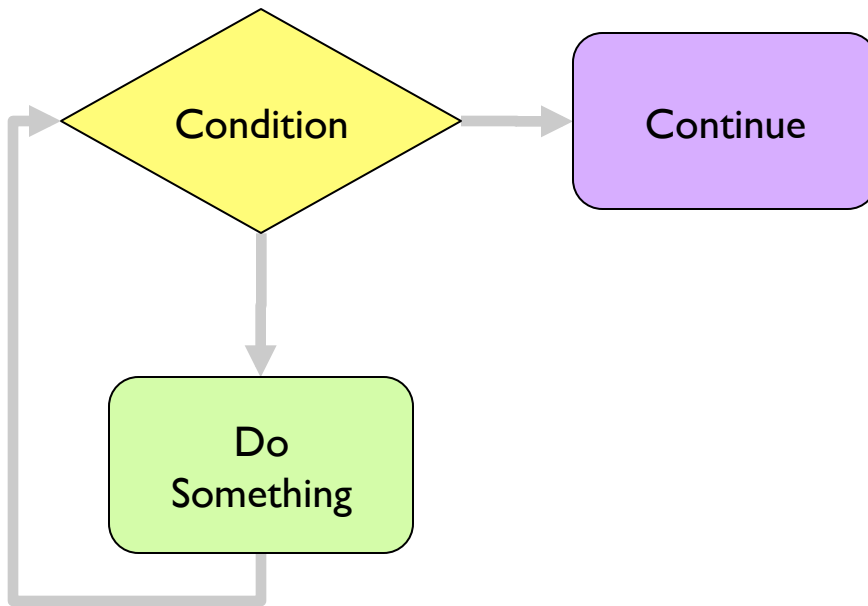
- `x == y`      true if x and y are equal (note common gotcha!)
- `x != y`      true if x and y are not equal
- `x > y`      true if x is greater than y
- `x <= y`      true if x is smaller than or equal to y
- `x && y`      true if both x and y are true
- `x || y`      true if either x or y is true
- `!x`          true if x is false



# Creepy Guy Formula: Exercises

- Add some error checking
  - Tip: `x == ""`
  - Tip: `exit()`
- Add some age appropriate pictures

# Loops



```
var n = 1;  
while (n <= 10) {  
    document.writeln(n);  
    n++;  
}
```

```
for (var n = 1; n <= 10; n++) {  
    document.writeln(n);  
}
```

Note, the text in red is part of the “template” of the loop

FYI: Computer scientists like to start at zero...

# Ice Cream Scoops: Exercises

- What happens if there's only one scoop?
- Change from for loop to while loop
- Alternate scoops of ice cream, chocolate and vanilla
  - Helpful tip: modulo (remainder) operation (%)
    - $3\%2 = 1$ ,  $4\%2 = 0$ ,  $5\%2 = 1$
- Randomize scoops of ice cream
  - To generate random number between 0 and 2:  
`Math.floor((Math.random()*3));`