Announcements

- ## Due dates extended:
  - ### Project 1B—Wednesday by 10pm
    - 1-1-1 rule Thursday by 10pm
  - ### Lab 5—Friday by 10pm

Announcements

- Vocabulary for the week has been posted in GoPost
- Reading
  - Ch 18 for today and Wednesday
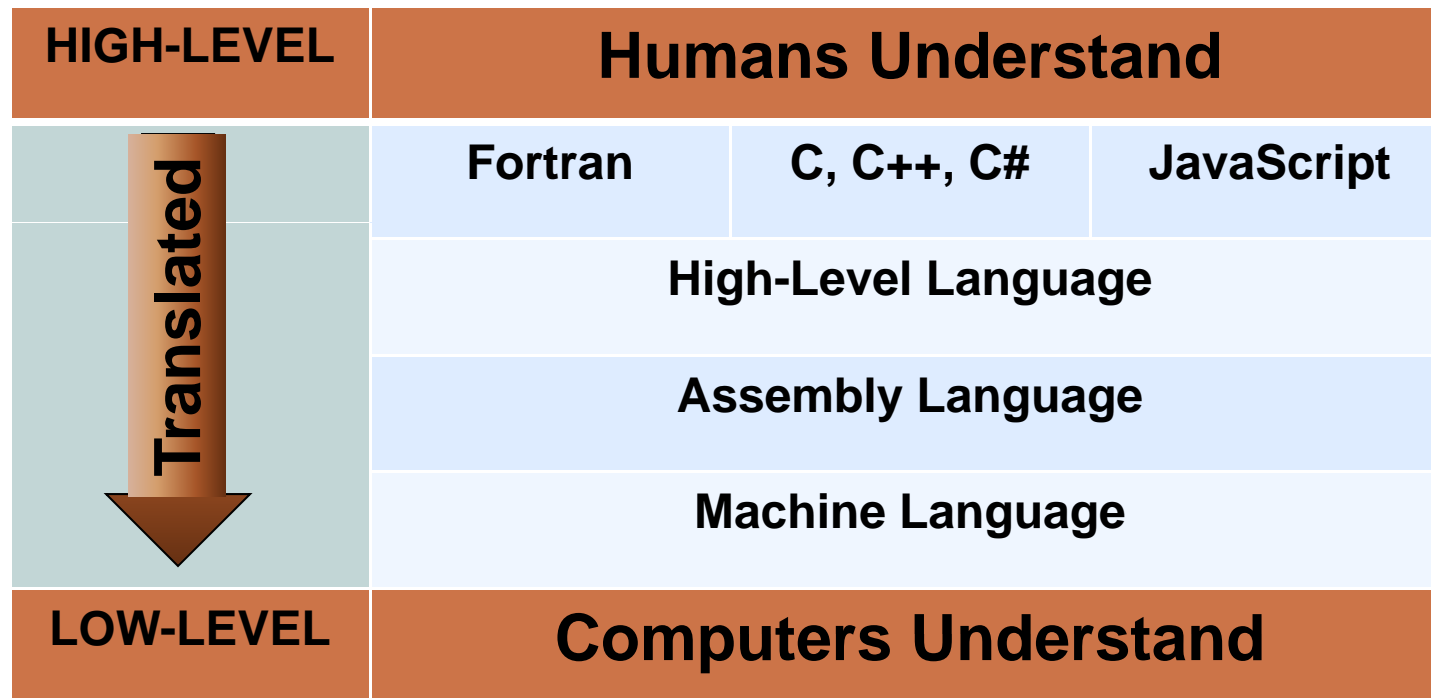 - Ch 21 for Friday

Programs Defined

- A *program* is an algorithm written for a specific programming language and specific circumstances

TYPES OF PROGRAMMING LANGUAGES

High- vs. Low-Level Languages

HIGH-LEVEL	Humans Understand		
	Fortran	C, C++, C#	JavaScript
Translated	High-Level Language		
	Assembly Language		
	Machine Language		
LOW-LEVEL	Computers Understand		

Videos

- High- and low-level programming languages

Human-Understandable Code

- Today's programs are written in "high-level" language (HLL) that we can understand (and debug)
- HLL use "real" words—if, while, when, until, push, pop, print, set, etc.
  - Words look like English
  - Have a precisely defined meaning for the computer
  - Make it easier for us to understand (and troubleshoot)
- For example:

```
if (today=='Wednesday')
        print "I have lecture today!"
else
        print "Tonight is time to study!"
```

High-Level Languages (HLL)

- **Video: High-Level Programming Languages**

Assembly language

- The lowest level language humans can understand
- Example
  - LOOP: MOV.B r0, #80 ;initialize counter

What computers understand

- Machine code
  - Assembly code is translated to binary:
    - 0011 0000 1000 0000
    - Binary is how computer stores information
      - all zeroes and ones
        - Magnetized or not
        - Off or on
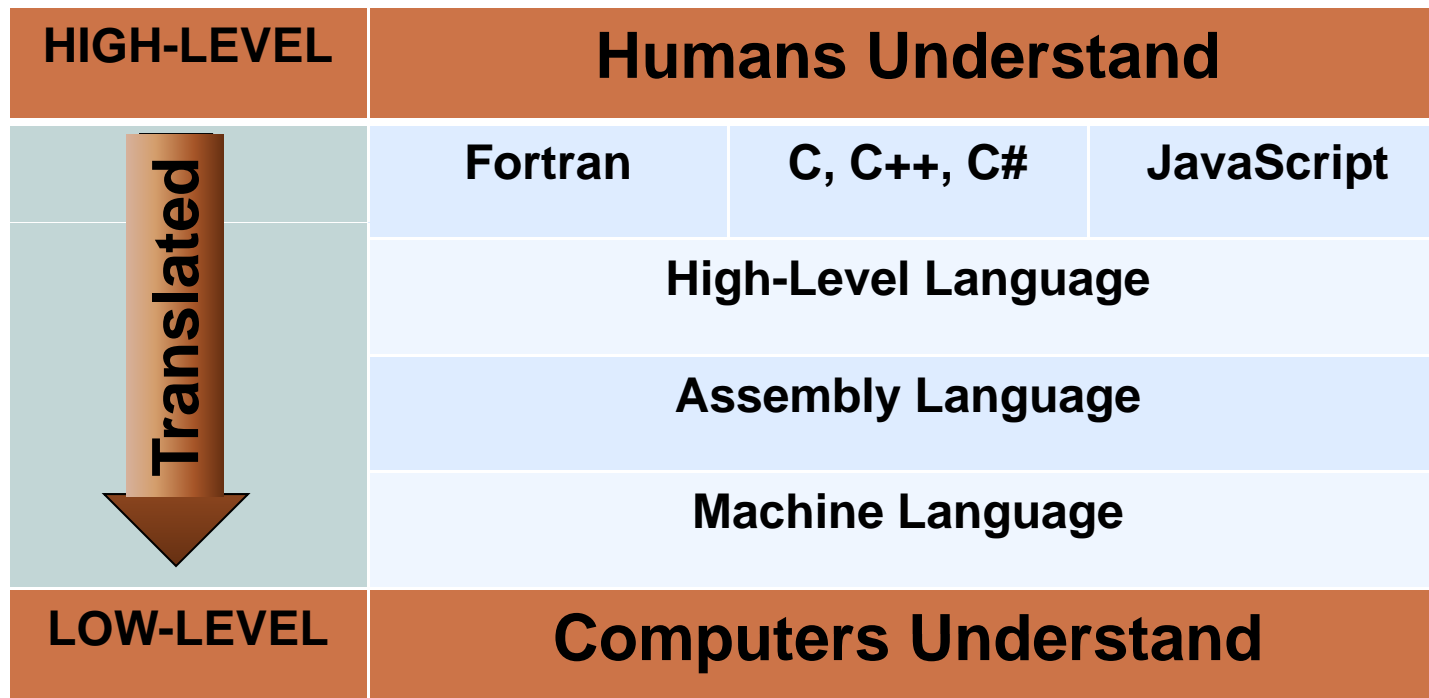   - Bumpy surface on the CD or not

Machine code

0110111000101110011101100111101100011001100111101101010111001110110011001000101110
0111011001111011000110011001111011010101111000101110011101100111101100011001100111
1011010101110000101100111011001111011000110011001111011010101110000101100111011001
1001100111101101010111000010111001110110011110110001100110011110110101011100001011
001110110011110110001100110011110110101011100001011100111011001111011000110011001111
0001100110011110110101011100001011100111011001111011000110011001

011101100011001100111011010101110000101
110011101100111101100011001100111101101
0101110000101110011101100111101100011001
1001111011000110011110110101011100001011
0011101100111101100011001100111101100110
01111011010101110000101110011101100111101
10001100110011110110101011100011001100111

Translating human to machine

HIGH-LEVEL	Humans Understand		
Translated (↓)	Fortran	C, C++, C#	JavaScript
	High-Level Language		
	Assembly Language		
	Machine Language		
LOW-LEVEL	Computers Understand		

Compiled Languages

- Compiled languages are translated to machine code (assembly language) before they are run.
- Whenever you make changes to your program, you have to recompile the program *again.*
- Because they already speak the computer's language, they run faster.
- Sometimes, they run by themselves—.exe files (NotePad2.exe)—or run with an engine (the Java virtual engine).

Compiled Languages

- Examples:
  - Java
  - C family
  - Visual Basic
  - COBOL
  - ForTRAN
  - many others

Also called *scripting* languages

INTERPRETED LANGUAGES

Interpreted Languages

- An interpreter translates from JavaScript to machine language while the Web browser renders the page
- The interpreter is part of the Web browser.
  - The JavaScript interpreter is available in all major Web browsers

How the Interpreter Works

- The interpreter translates the script to machine language while the program runs!
  - Two tasks at once—translating and running the program!
  - Scripts run slower than compiled programs

The Advantages...

- Scripted languages are interpreted *on the fly*—while they are running
  - Make changes while the program is running!
  - Provides a dynamic, responsive, and interactive experience for the user
  - Scripts respond to user input

Font Size

normal | large | largest

All about

JAVASCRIPT

JavaScript

- Java was developed by Sun Microsystems and is seen on the Web mostly in Java Applets.

JavaScript is *not* Java!

Brief History of JavaScript

- Released with Netscape Navigator in 1995.
- Ratified by the European Equipment Manufacturer's Association (ECMA)
- Result:
  - ECMAScript is the core spec for the JavaScript language
  - Netscape, MS, and the others try to conform to the spec

Divergence from standards....

Developer	Name
Netscape (now Mozilla)	JavaScript
Microsoft	JScript

- Programmers call both *JavaScript*.
- Both comply differently with the standards

Javascript & the Web

Adding interaction to a static HTML page

D.A. Clements

Objectives

- Understand how JavaScript and HTML interact
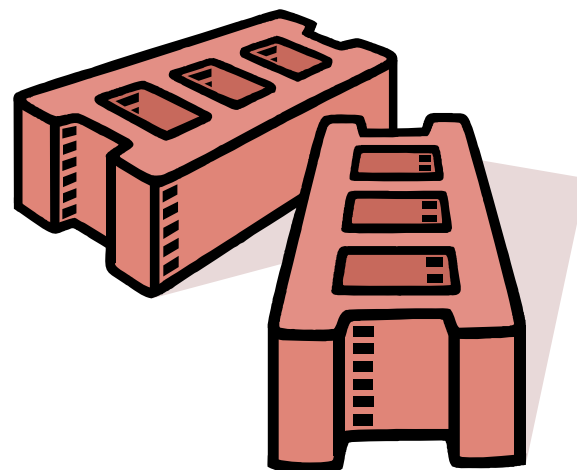- Understand where to place JavaScripts on the HTML page

Programming Concepts

- Programming: Act of formulating an algorithm or program

- Basic concepts have been developed over last 50 years to simplify common programming tasks

- Concepts will be expressed in JavaScript

The Web page

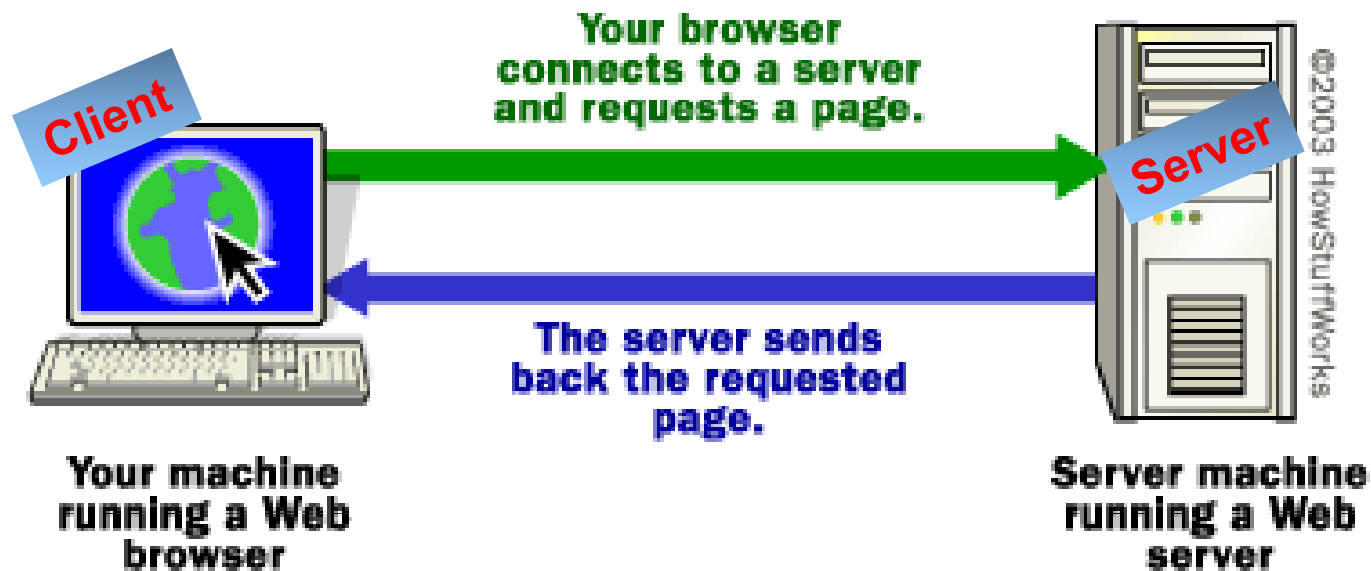- Without JavaScript the Web page is like a brick; it just sits there!

Scripts

Client-side scripts

- Your Web browser on your computer is the client

Server-side scripts

- Web server
- Database server
- File server

Your browser connects to a server and requests a page.

The server sends back the requested page.

Client

Server

©2003 HowStuffWorks

Your machine running a Web browser

Server machine running a Web server

Web browser and JavaScript

- The major Web browsers have a built-in interpreter that parses JavaScript
  - Parses: breaks into smaller pieces that can be translated into machine code

Placing JavaScripts on a Web page

Types of scripts:

- Body scripts

- Header scripts

- External scripts

Body Script

```
<html>
  <head>
    <title>Name of Page</title>
  </head>
  <body>
    <script type="text/javascript">
        //JavaScript goes here
    </script>
  </body>
</html>
```

Header Script

```
<html>
  <head>
    <title>Name of Page</title>
    <script type="text/javascript"
        //JavaScript goes here
    </script>
  </head>
  <body>
    Body content goes here
  </body>
</html>
```

Linking to External JavaScripts

- Linked in the <head>
- src gives path to file and its name

```html
<html>
   <head>
      <title>Name of Page</title>
      <script type="text/javascript"
         src="basic.js"></script>
   </head>
   <body>
      Body content goes here
   </body>
</html>
```

External JavaScripts

- ● Make changes to scripts in one place
- ● Reusable
  - ● Link to
    - ● any page,
    - ● every page, or
    - ● many sites

```javascript
function showSection(id) {
  var divs = document.getElementsByTagName("div");
  for (var i=0; i<divs.length; i++ ) {
    if (divs[i].className.indexOf("section") == -1) continue;
    if (divs[i].getAttribute("id") != id) {
      divs[i].style.display = "none";
    } else {
      divs[i].style.display = "block";
    }
  }
}

function prepareInternalnav() {
  if (!document.getElementsByTagName) return false;
  if (!document.getElementById) return false;
  if (!document.getElementById("internalnav")) return false;
  var nav = document.getElementById("internalnav");
  var links = nav.getElementsByTagName("a");
  for (var i=0; i<links.length; i++ ) {
    var sectionId = links[i].getAttribute("href").split("#")[1];
    if (!document.getElementById(sectionId)) continue;
    document.getElementById(sectionId).style.display = "none";
    links[i].destination = sectionId;
    links[i].onclick = function() {
      showSection(this.destination);
      return false;
    }
  }
}

addLoadEvent(prepareInternalnav);
```

Best Practice

- Best practice to separate Content from Action from Appearance
  - Put styles in external CSS
  - Put scripts in external JavaScript files
  - Leave only the HTML markup and content on the page
- Accomplishing that goal takes more experience….

Summary

- Understand how JavaScript and HTML interact
- Understand where to place JavaScripts on the HTML page