Programming

LBSC 690: Jordan Boyd-Graber

University of Maryland

November 12, 2012

Adapted from Jimmy Lin’s Slides
Take-Away Messages

- Assignment 2 Review
- Midterm Recap
- Project
- Programming is a lot like cooking
- What kinds of programming languages are out there
- Basic programming you can use in webpages
Outline

1. Assignment 2 Review
2. Midterm
3. Project
4. Programming
Assignment 2

- Most did well
- CSS duplication
- Broken links
- Navigation inconsistent or missing
- Missing ALT tags and large images
  - Think about how it will display
  - Careful about large image sizes
- Lots of &nbsp; and <br /> for spacing
Outline

1. Assignment 2 Review
2. Midterm
3. Project
4. Programming
• Answers posted
• You should be worried if you got below 20
• Difference between metadata and markup
• RAM is volatile, but you can make a SSD out of it
• Bits and bytes
Outline

1. Assignment 2 Review
2. Midterm
3. Project
4. Programming
Final Project

- Information available
- Form teams and get project idea soon
- Very broad scope
  - Use technologies from class
  - Do something interesting
- Chat with me (office hours, after / before class, e-mail) with idea
Outline

1. Assignment 2 Review
2. Midterm
3. Project
4. Programming
You will learn about... 

- Different types of programming languages
- Basic programming constructs
- Controlling execution of instructions
Types of Software

- Application programs (e.g., PowerPoint)
  - What you normally think of as a “software”
- Operating system (e.g., Windows XP)
  - Software that manages your computing resources
- Compilers and interpreters
  - Software used to write other software
- Embedded software (e.g., TiVO)
  - Programs permanently embedded inside some physical device
- Software “does something”
- Instructions for telling the machine “what to do” are expressed in a programming language
- Special purpose: geared towards specific tasks
  - Spreadsheets (e.g., Excel)
  - Databases (e.g., SQL)
  - Complex math (e.g., Matlab)
- General purpose: able to accomplish anything
  - Examples: Java, JavaScript, C, C++, Perl, Python . . .
Why should you care?

- JavaScript: useful tool for making interactive webpages
- Knowing when you should ask for a programmer to help
- Helping diagnose problems in information technology
- Dealing with large amounts of information
- Automating simple tasks
- How to interact with Databases in friendlier way (e.g. HTML5)
Programming Ingredients

ingredients  data types
containers  variables
recipes  algorithms
Types of Programming: Low Level

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

```assembly
.model small
.stack
.data

main proc
    mov ax, seg message
    mov ds, ax

    mov ah, 09
    lea dx, message
    int 21h

    mov ax, 4c00h
    int 21h
main endp
end main
```
Types of Programming: High Level

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

```python
for ii in xrange(3):
    print ii, sum(x**ii for x in xrange(10))
```
How do you interact with programming languages?

- **Compiled languages**
  - Write a program as a plain-text file
  - Compile converts plain-text file into an executable

- **Interpreted languages**
  - Write a program as a plain-text file
  - Another program “runs” the file **or** allows you to interactively issue commands

- **JavaScript**
How do you interact with programming languages?

- Compiled languages
  - Write a program as a plain-text file
  - Compile converts plain-text file into a executable

- Interpreted languages
  - Write a program as a plain-text file
  - Another program “runs” the file or allows you to interactively issue commands

- JavaScript
  - Webpage is the source code
  - Browser is the interpreter
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., \( x \))
- In JavaScript, “var” declares a variable
  - create a boolean \( b \) and set it to true
    \[
    \text{var } b = \text{true};
    \]
  - create a number \( n \) and set it to 1
    \[
    \text{var } n = 1;
    \]
  - create a string \( s \) and set it to “hello”
    \[
    \text{var } s = \text{"hello"};
    \]
- Things that you can do:
  - $-x$  
    reverse the sign of $x$ (negation)
  - $6 + 5$  
    Add 6 and 5 (numeric)
  - "Hello” + “World”  
    Concatenate two strings
  - $2.1 * 3$  
    Multiply two values

- Storing results:
  - $x = 5$  
    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, all instructions end with a semicolon (;)
Controlling Execution

- Sequence
- Condition
- Repetition
```javascript
var a = 2;
var b = 3;
var c = a * b;
```
if (gender == "male") {
    greeting = "Hello, Sir";
} else {
    greeting = "Hello, Madam";
}
n = 1;
while (n <= 10) {
    document.writeln(n);
    n++;
}

for (n = 1; n <= 10; n++) {
    document.writeln(n);
}
Test Conditions

- $x == y$ true if $x$ and $y$ are equal
- $x! = y$ 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
Arrays

- A set of elements grouped together
  - For example, the number of days in each month

Each element is assigned an index

- A number is used to refer to that element
- For example, $x[4]$ is the fifth element (count from zero!)
- Arrays and repetitions work naturally together
Functions

- Reusable code for doing a single task
- A function takes in one or more parameters and returns one value

```javascript
function convertToCelsius(f) {
  var celsius = 5/9 * (f - 32);
  return celsius;
}

function weirdAddition(a, b) {
  var result = a + b - 0.5;
  return result;
}
```
Calling Functions

- When you “call” a function, you invoke the set of instructions it represents.

```
c = convertToCelsius(60);

function convertToCelsius(f) {
    var celsius = 5/9 * (f-32);
    return celsius;
}
```
More Functions

```javascript
var f = 60;
c = convertToFahrenheit(f);

r = weirdAddition(2, 4);

var a = 2;
var b = 3;
r = weirdAddition(a, b);
```
Algorithms

- Derived from the name of the Persian mathematician Al-Khwarizmi
- A sequence of well-defined instructions designed to accomplish a certain task
Common Gateway Interface (CGI) [Server-side]
- User inputs information into a form
- Form values passed to the server via CGI
- Program on the server generates a Web page as a response

Specialized Servers: Souped up CGI
- PHP
- Tomcat / JSP
- Google Webapp

JavaScript [Client-side]
- Human-readable source code sent to the browser
- Web browser runs the program
JavaScript is usually kept in the `<head>` section of an HTML document

```html
<head>
<script language="JavaScript" type="text/javascript">
<!--
function calculate () {
    var num = eval(document.input.number.value);
    ...
    document.output.number.value = total;
}
//-->
</script>
</head>
...
Handling Events

- When does code actually get executed?
- Events:
  - User actions trigger events
  - Embedded in all modern GUls
- Event handlers are used to respond to events
- Examples of event handlers in JavaScript
  - onMouseover: the mouse moved over an object
  - onMouseout: the mouse moved off an object
  - onClick: the user clicked on an object
  - onLoad: the page loads for the first time
How do you get information to/from the user?
Forms provide a method for accepting input and displaying output

**HTML**

```html
<form name="input" action="">
Please enter a number:
<input size="10" value="" name="number" />
</form>

<form name="output" action="">
The sum of all numbers up to the number above is
<input size="10" value="" name="number" readonly="true" />
</form>
```

**Javascript**

```javascript
var num = eval(document.input.number.value);
document.output.number.value = num * (num + 1) / 2;
```
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
  - Debug by outputting the state of the program

- Print out the value of variables using `document.write`; is the value what you expected?
You Have Learned About

- Different types of programming languages
- Basic programming constructs
- Controlling execution of instructions
Discussion Question

Modify social networking selector


- If you are younger than 15, it displays a list of five books you should read instead of using social networking sites (use an array).
- If you are younger than 24, it sends you to Facebook.
- If you are 24 or older, it sends you to LinkedIn.
- What else might you do to improve it?