中山大學
SUN YAT-SEN UNIVERSITY

Lecture 12
Basic JavaScript for Client Side Programming

SE-805 Web 2.0 Programming (supported by Google)

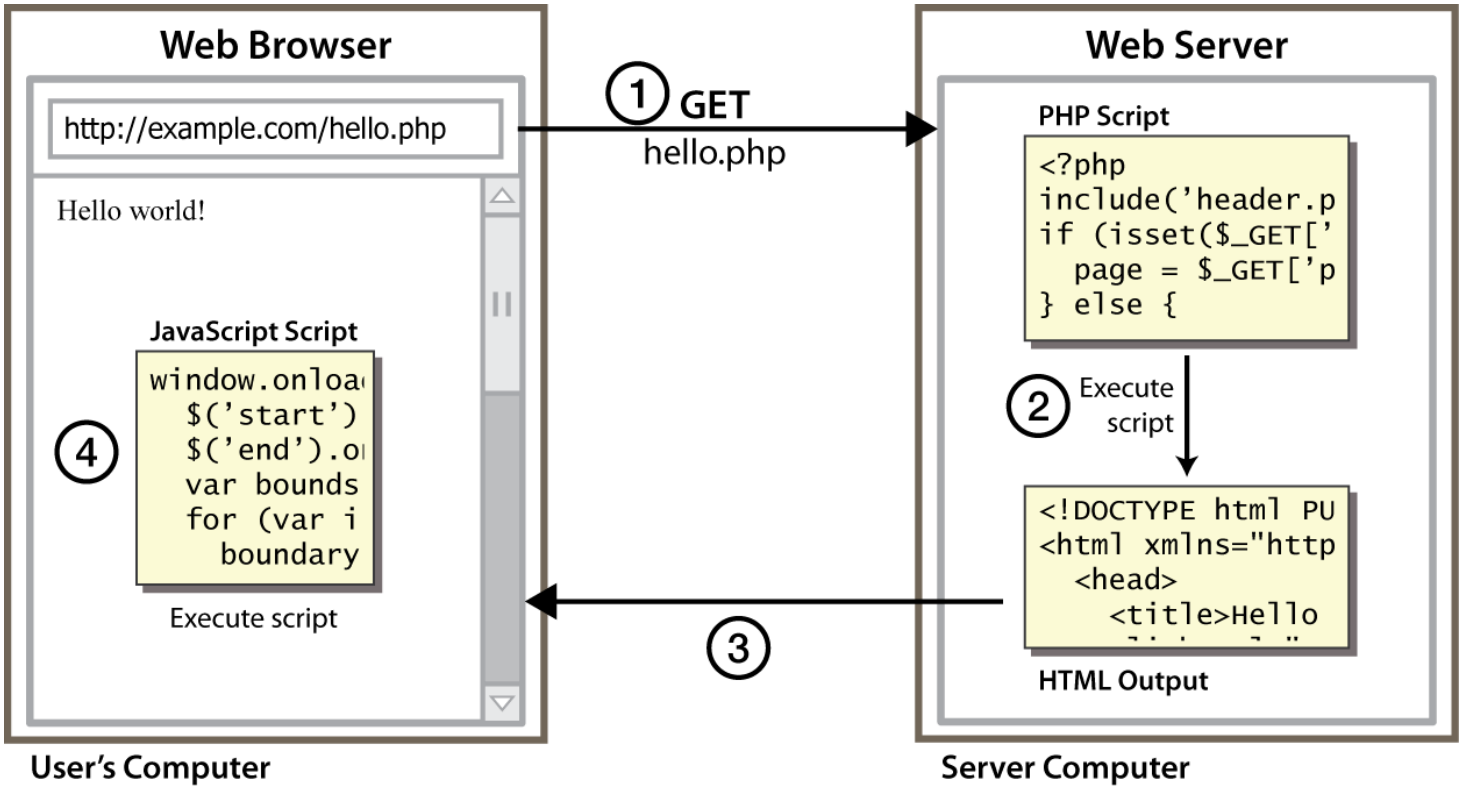
http://my.ss.sysu.edu.cn/courses/web2.0/

School of Software, Sun Yat-sen University

Outline

- **Client Side Basics**
- Introduction to JavaScript
- JavaScript Basic Syntax

Client-Side Scripting



- Client-side script: code **runs in browser** after page is sent back from server
  - Often this code manipulates the page or responds to user actions

Client-Side vs. Server-Side Programming

- PHP already allows us to create dynamic web pages. Why also use client-side scripting?
- Client-side scripting (JavaScript) benefits:
  - **Usability**: can modify a page without having to post back to the server (faster UI)
  - **Efficiency**: can make small, quick changes to page without waiting for server
  - **Event-driven**: can respond to user actions like clicks and key presses
- server-side programming (PHP) benefits:
  - **Security**: has access to server's private data; client can't see source code
  - **Compatibility**: not subject to browser compatibility issues
  - **Power**: can write files, open connections to servers, connect to databases, ...

Outline

- Client Side Basics
- **Introduction to JavaScript**
- JavaScript Basic Syntax

Essential of JavaScript

- **JavaScript** is an object-oriented scripting language used to enable programmatic access to objects within both the client application and other applications. It is primarily used in the form of client-side JavaScript, implemented as an integrated component of the web browser, allowing the development of enhanced user interfaces and dynamic websites.
- JavaScript is a dialect of the **ECMAScript** standard and is characterized as a dynamic, weakly typed, prototype-based language with first-class functions. JavaScript was influenced by many languages and was designed to look like Java, but to be easier for non-programmers to work with.

Essential of JavaScript

- JavaScript is a script language
- JavaScript programs are evaluated and executed by JavaScript interpreters / engines
  - Rhino, SpiderMonkey, V8, Squirrelfish
- The mainstream purpose and usage: Exposing objects of an application at runtime, for customizing / embedding user logics
  - OS, **browsers**, flashes, pdf apps, etc.
  - That implies two sections of learning JavaScript, **the language itself** and **objects** exposed in corresponding host applications

JavaScript vs. Java

- **Interpreted**, not compiled
  - More relaxed syntax and rules
  - Fewer and "looser" data types
  - Variables DON'T need to be declared
  - Errors often silent (few exceptions)
- Key construct is the **function** rather than the class
  - "First-class" functions are used in many situations
- Contained within a web page and integrates with its HTML/CSS content
  - **Comparability**: browsers may behave differently upon a JavaScript program
    - Different dialects/implementations of the standard (ECMAScript)
    - Different objects exposed

JavaScript vs. PHP

- Similarities:
  - Both are **interpreted**, not compiled
  - Both are relaxed about syntax, rules, and types
  - Both are case-sensitive
  - Both have built-in regular expressions for powerful text processing
- Differences:
  - JS is **more object-oriented**: noun.verb(), less procedural: verb(noun)
  - JS focuses on user interfaces and interacting with a document; PHP is geared toward HTML output and file/form processing
  - JS code runs on the client's browser; PHP code runs on the web server

Linking to a JavaScript File: script

```HTML
<script src="filename" type="text/javascript"></script>
```

```HTML
<script src="example.js" type="text/javascript"></script>
```

- script tag should be placed in HTML page's head
- Script code is stored in a separate .js file
- JS code can be placed directly in the HTML file‘s body or head (like CSS)
  - But this is **BAD** style (should separate content, presentation, and behavior)

Outline

- Client Side Basics
- Introduction to JavaScript
- **JavaScript Basic Syntax**

Comments (same as Java)

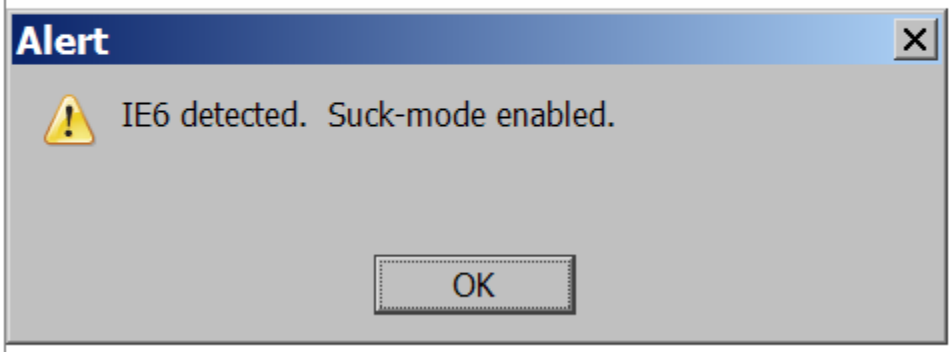
```JS
// single-line comment

/* multi-line comment */
```

- Identical to Java's comment syntax
- Recall: 4 comment syntaxes
  - HTML: <!--*comment* -->
  - CSS/JS/PHP: /* *comment* */
  - Java/JS/PHP: // *comment*
  - PHP: # *comment*

A JavaScript Statement: alert

```
alert("message");
```
JS

```
alert("IE6 detected.  Suck-mode enabled.");
```
JS



output

- A JS command that pops up a dialog box with a message

Variables and Types

```
var name = expression;
```
JS

```
var clientName = "Connie Client";
var age = 32;
var weight = 127.4;
```
JS

- Variables are declared with the var keyword (case sensitive)
- Types are not specified, but JS does have types ("loosely typed")
  - Number, Boolean, String, Array, Object, **Function**, Null, **Undefined**
  - Can find out a variable's type by calling typeof

Number Type

```
var enrollment = 99;
var medianGrade = 2.8;
var credits = 5 + 4 + (2 * 3);
```
JS

- Integers and real numbers are the **same** type (no int vs. double)
- Same operators: + -* /% ++ --= += -= *= /= %=
- Similar precedence to Java
- Many operators **auto-convert types**: "2"* 3 is 6

String Type

```JS
var s = "Connie Client";
var fName = s.substring(0, s.indexOf(" "));   // "Connie"
var len = s.length;                           // 13
var s2 = 'Melvin Merchant';
```

- Methods: charAt, charCodeAt, fromCharCode, indexOf, lastIndexOf, replace, split, substring, toLowerCase, toUpperCase
  - charAt returns a one-letter String (there is no char type)
- Length property (not a method as in Java)
- Strings can be specified with "" or ''
- Concatenation with **+** :
  - 1 +1 is 2, but"1" +1 is"11"

More about String

- Escape sequences behave as in Java: \' \" \& \n \t \\
- Converting between numbers and Strings:

```JS
var count = 10;
var s1 = "" + count;                  // "10"
var s2 = count + " bananas, ah ah ah!";   // "10 bananas, ah ah ah!"
var n1 = parseInt("42 is the answer");    // 42
var n2 = parseFloat("booyah");            // NaN
```

- Accessing the letters of a String:

```JS
var firstLetter = s[0];             // fails in IE
var firstLetter = s.charAt(0);      // does work in IE
var lastLetter = s.charAt(s.length - 1);
```

Boolean Type

```JS
var iLike190M = true;
var ieIsGood = "IE6" > 0;    // false
if ("web dev is great") {   /* true */ }
if (0) {   /* false */ }
```

- Any value can be used as a Boolean
  - "Falsey" values: 0, 0.0, NaN, "", null, and undefined
  - "Truthy" values: anything else
- Converting a value into a Boolean explicitly:
  - var boolValue = Boolean(otherValue);
  - var boolValue = !!(otherValue);

Special Values: null, NaN, undefined

```JS
var ned = null;
var benson = 9;

// at this point in the code,
//   ned is null
//   benson's 9
//   caroline is undefined
```

- NaN: not a number (only returned by the isNaN() function)
- undefined : has not been declared, does not exist
- null : exists, but was specifically assigned an null value
- Why does JavaScript have both of these?

Math Object

```JS
var rand1to10 = Math.floor(Math.random() * 10 + 1);
var three = Math.floor(Math.PI);
```

- Methods: abs, ceil, cos, floor, log, max, min, pow, random, round, sin, sqrt, tan
- Properties: E, PI

Logical Operators

- > < >= <= &&|| !== != **=== !==**
- Most logical operators automatically convert types:
  - 5 < "7" is true
  - 42 == 42.0 is true
  - "5.0" == 5 is true
- === and !== are **strict equality** tests; checks both type and value
  - "5.0" === 5 is **false**

if/else Statement

```
if (condition) {
  statements;
} else if (condition) {
  statements;
} else {
  statements;
}
```
JS

- Identical structure to Java's if/else statement
- JavaScript allows almost anything as a *condition*

for Loop (same as Java)

```JS
for (initialization; condition; update) {
  statements;
}
```

```JS
var sum = 0;
for (var i = 0; i < 100; i++) {
  sum = sum + i;
}
```

```JS
var s1 = "hello";
var s2 = "";
for (var i = 0; i < s.length; i++) {
  s2 += s1.charAt(i) + s1.charAt(i);
}
// s2 stores "hheellllloo"
```

while Loops (same as Java)

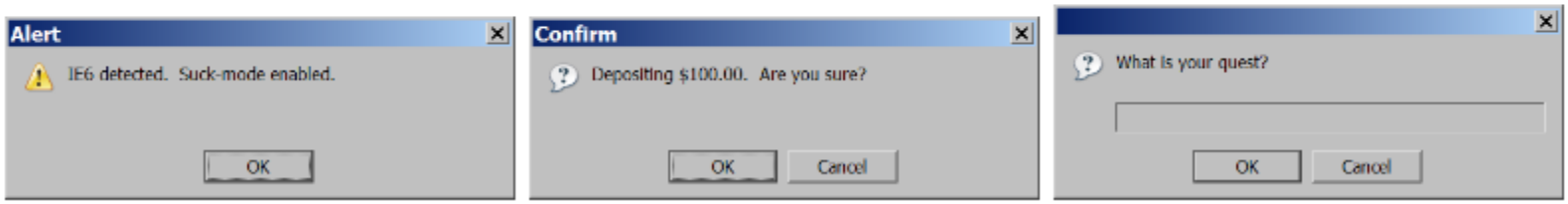
```
while (condition) {
  statements;
}
```
JS

```
do {
  statements;
} while (condition);
```
JS

- break and continue keywords also behave as in Java

Popup Boxes

```
alert("message");      // message
confirm("message");    // returns true or false
prompt("message");     // returns user input string
```
JS



Arrays

```
var name = [];                          // empty array
var name = [value, value, ..., value];  // pre-filled
name[index] = value;                    // store element
```
JS

```
var ducks = ["Huey", "Dewey", "Louie"];

var stooges = [];           // stooges.length is 0
stooges[0] = "Larry";       // stooges.length is 1
stooges[1] = "Moe";         // stooges.length is 2
stooges[4] = "Curly";       // stooges.length is 5
stooges[4] = "Shemp";       // stooges.length is 5
```
JS

- Two ways to initialize an array
- length property (grows as needed when elements are added)

Array Methods

```
var a = ["Stef", "Jason"];    // Stef, Jason
a.push("Brian");              // Stef, Jason, Brian
a.unshift("Kelly");           // Kelly, Stef, Jason, Brian
a.pop();                      // Kelly, Stef, Jason
a.shift();                    // Stef, Jason
a.sort();                     // Jason, Stef
```
JS

- Array serves as many data structures: list, queue, stack, ...
- Methods: concat, join, pop, push, reverse, shift, slice, sort, splice, toString, unshift
  - push and pop add / remove from back
  - unshift and shift add / remove from front
  - shift and pop return the element that is removed

Splitting Strings: split and join

```
var s = "the quick brown fox";
var a = s.split(" ");              // ["the", "quick", "brown", "fox"]
a.reverse();                       // ["fox", "brown", "quick", "the"]
s = a.join("!");                   // "fox!brown!quick!the"
```
JS

- split breaks apart a string into an array using a delimiter
  - can also be used with regular expressions (seen later)
- join merges an array into a single string, placing a delimiter between them

Summary

- Client Side Basics
  - Client-side vs. server-side
- Introduction to JavaScript
  - Standard, language type, purposes & uages
  - Language comparisons (Java, PHP)
- JavaScript Basic Syntax
  - Comments, alert, confirm, prompt
  - Variables and types: Number, Boolean, String (split/join)
  - null, NaN, undefined
  - Math object, logical operators
  - if/else, for, while
  - Array

Exercises

- Write JavaScript snippets in Firebug console:
  - Create a Fibonacci function, fabonacci(n), which returns the nth element of the Fibonacci sequence
  - Create a function hideVowel(str), which returns a string replacing all vowels in the given str with “*”
  - Create a functin quickSort(array), which sorts the given array using the Quick Sort algorithm

Further Readings

- Introduction of JavaScript
  http://en.wikipedia.org/wiki/JavaScript
- W3Schools JavaScript tutorial
  http://www.w3schools.com/js/default.asp
- Mozilla Developer Center JavaScript documentation
  https://developer.mozilla.org/en/javascript

Thank you!