中山大學
SUN YAT-SEN UNIVERSITY

Lecture 05. Basic JavaScript for Client Side Programming

Modern Web Programming
(http://my.ss.sysu.edu.cn/wiki/display/WEB/ **supported by Deep Focus**)

School of Data and Computer Science, Sun Yat-sen University

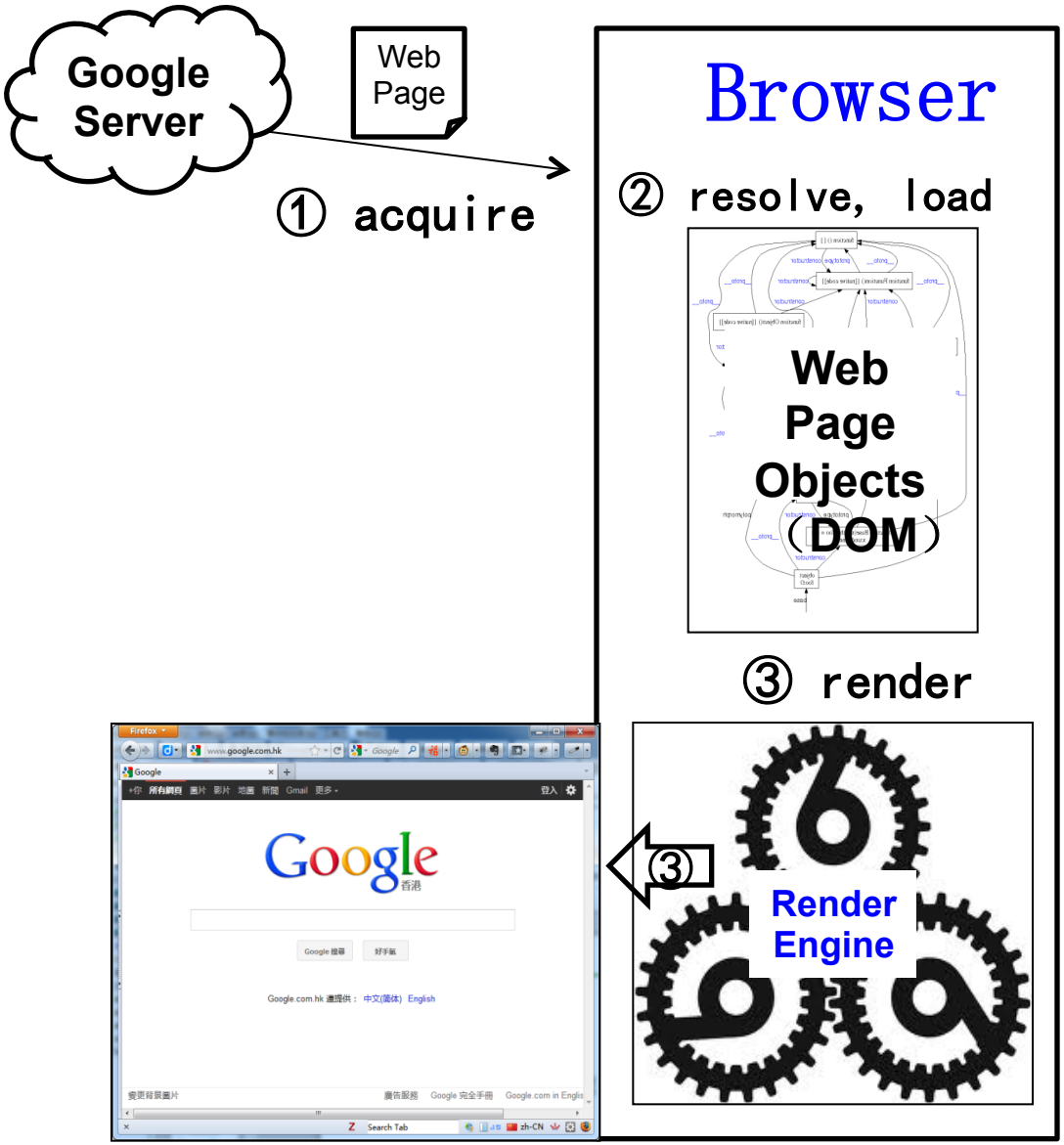
Outline

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

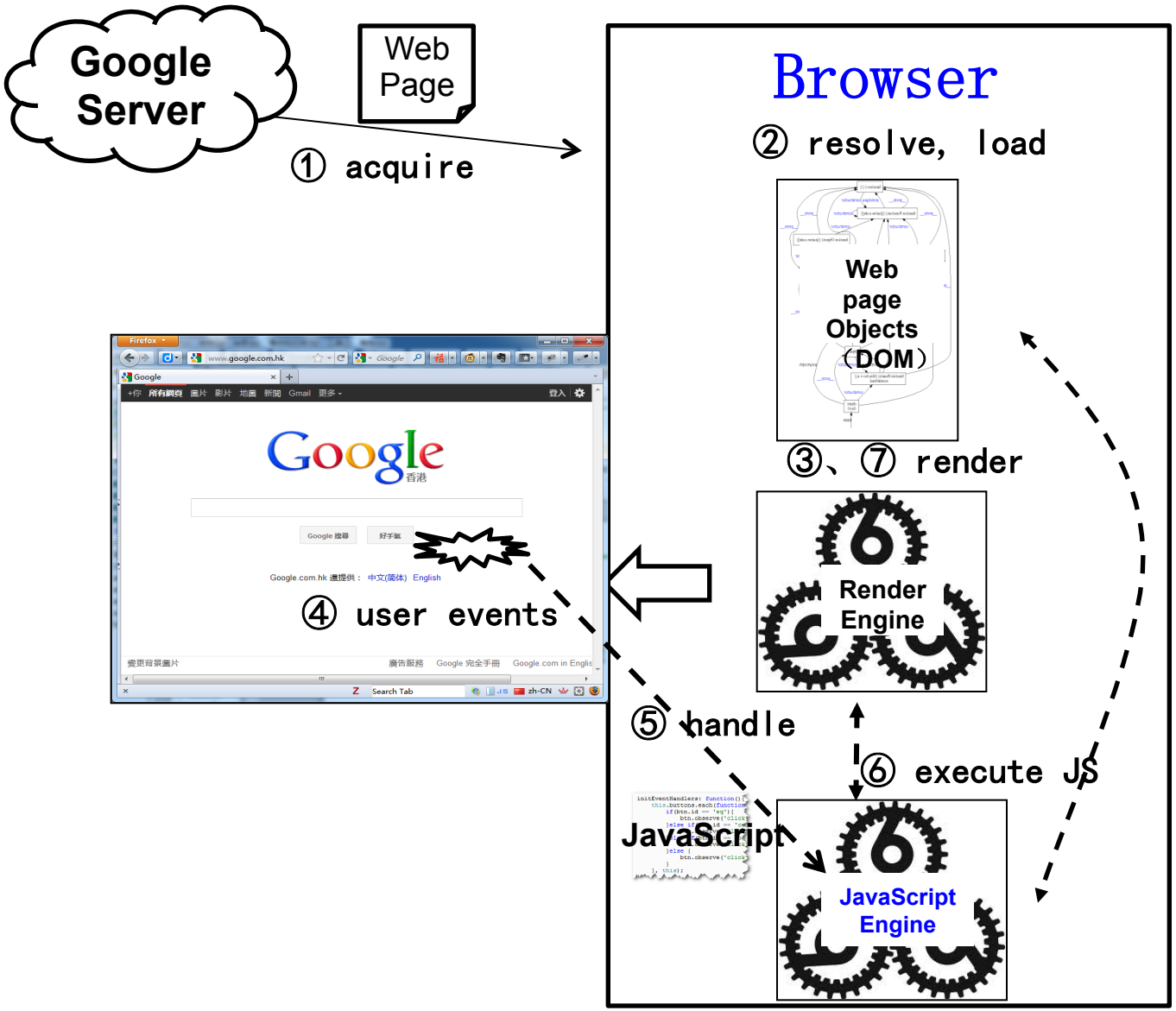
Front-end Dynamics

- Static Web pages → Dynamic Web pages
  - But, all dynamics happen on Web servers
- Not all dynamics should be on server
  - Validate data format
  - Eye-candies
  - ……
- Client (browser) dynamics is indispensable for Web apps.
  - **Dynamic HTML**
  - **A calculator example**

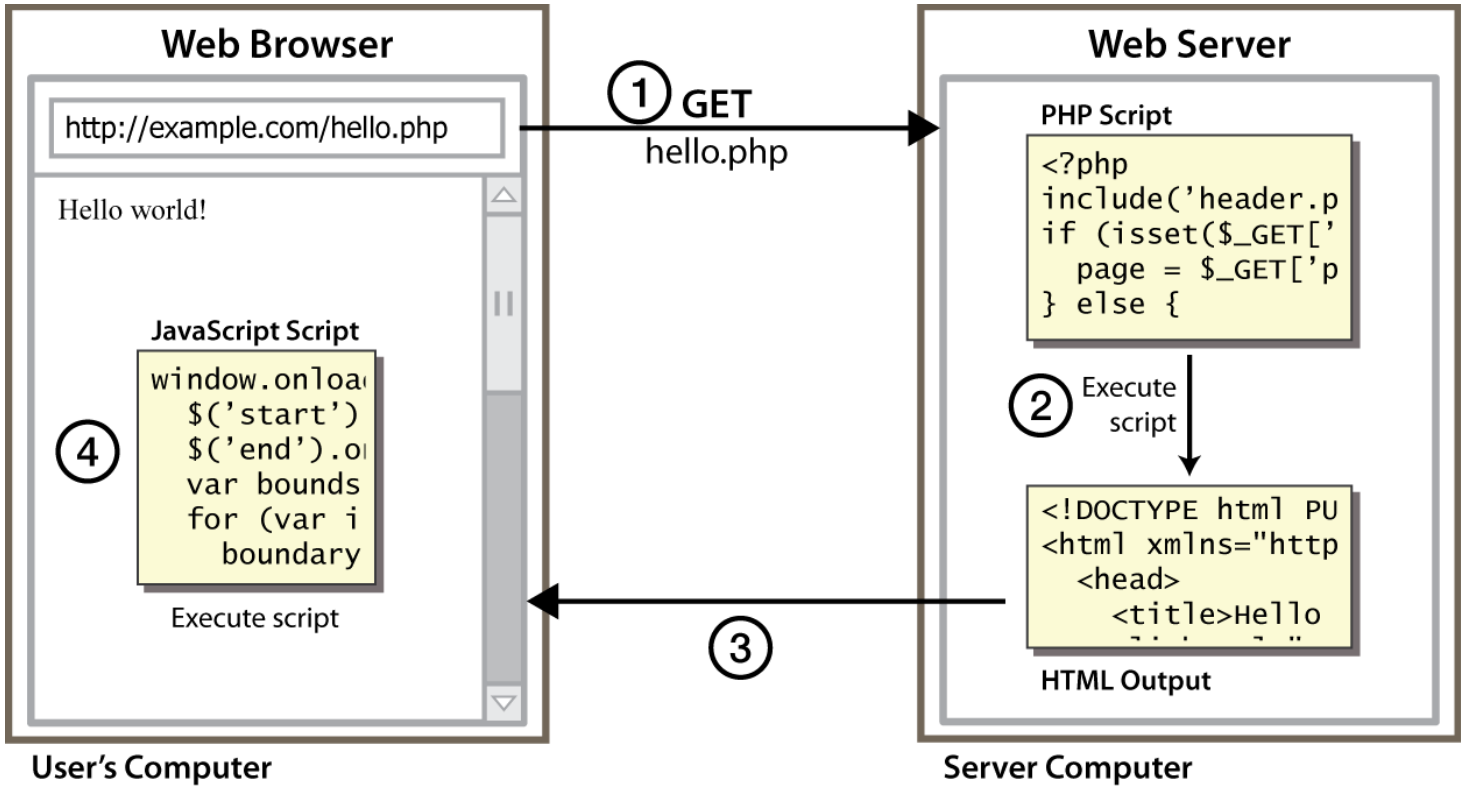
Working mechanism of browsers



Working mechanism of browsers



Client-side scripting



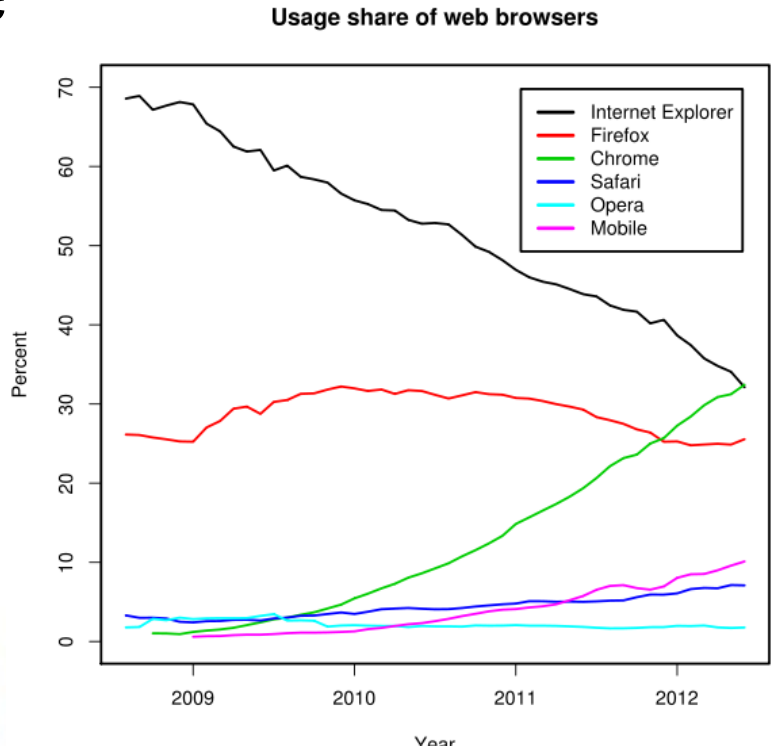
- client-side script: code **runs in browser** after page is sent back from server
  - often this code manipulates the page or responds to user actions

Event-driven programming

Browser wars

- Mosaic war
  - the winner: Netscape
- The first browser war (1995.12.7 ~ 19
  - Netscape Navigator 90%, IE 0%
  - The battle at IE 4.0 party, San Franc
  - IE 96%
  - Netscape was purchased by AOL
  - Consequence: → Web Standardiza
- The second browser war (1998 ~

From the IE team ...



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.

1995

/ FAST SLIM CORRECT

BRENDAN EICH DESIGNS MOCHA JAVASCRIPT ...IN 10 DAYS

Joins NETSCAPE in April 1995.

Essential of JavaScript

- JavaScript is a script language
- JavaScript programs are evaluated and executed by JavaScript interpreters / engines
  - Rhino, SpiderMonkey, V8, Squirrelfish, TraceMonkey
- 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 Engine Competition

- Google Chrome → Webkit's Squirrelfish, Firefox 's TraceMonkey
- An incredible JavaScript Engine Google V8 , which is as fast as binary code!
- 2009 Node.js, JavaScript for server side
- Beat apache, Nginx, IIS up!
  - Especially in high concurrence
- A new technical revolution emerges in Web front-end
- CoffeScript, Sass, Less, Haml, ……
- The legend goes on
  - Safari'sNitro, Mozilla's JägerMonkey, ……

JavaScript is the language for the Web era.

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

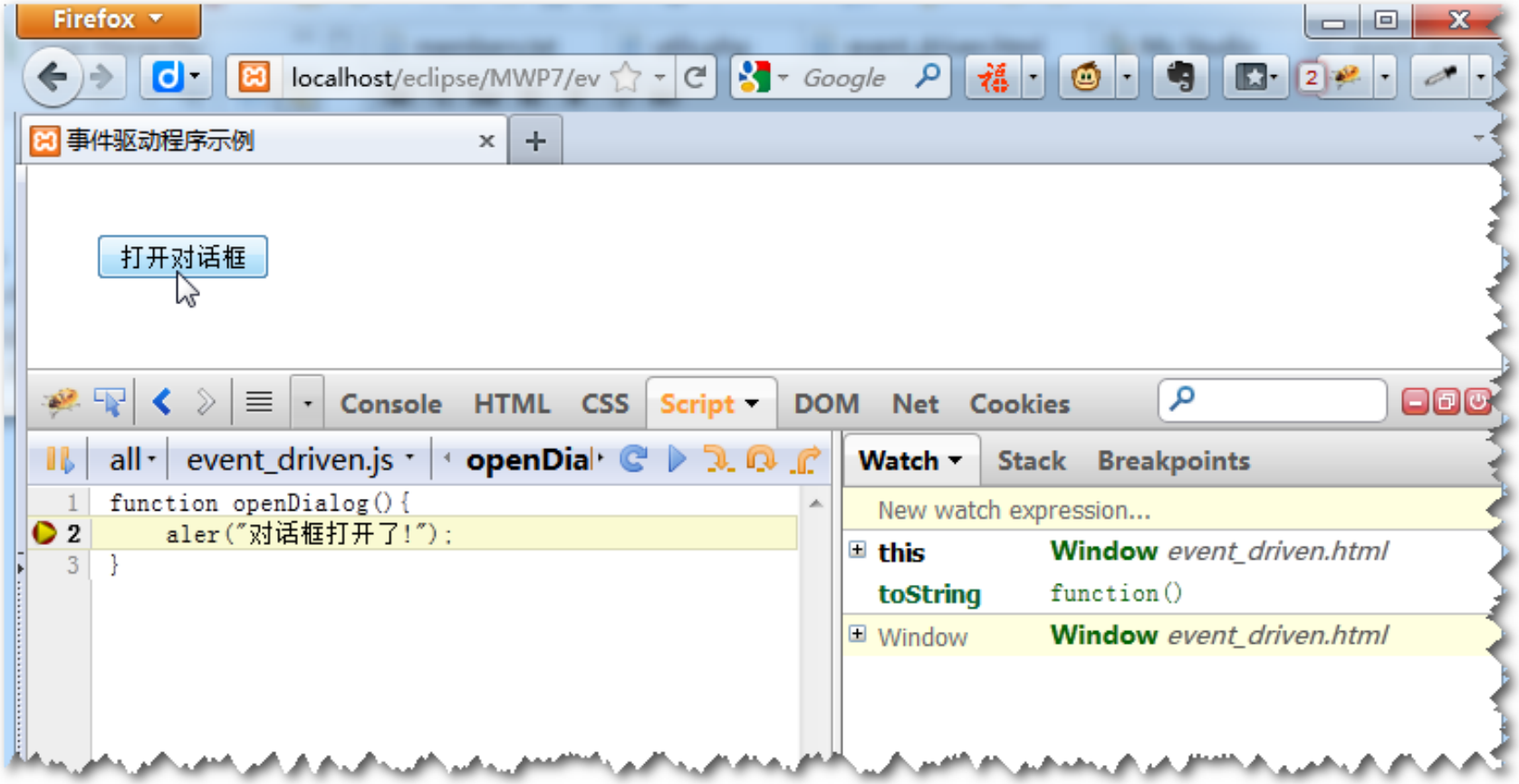
Linking to a JavaScript file: script

```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)

Run and debug JavaScript



Outline

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

Comments (same as Java)

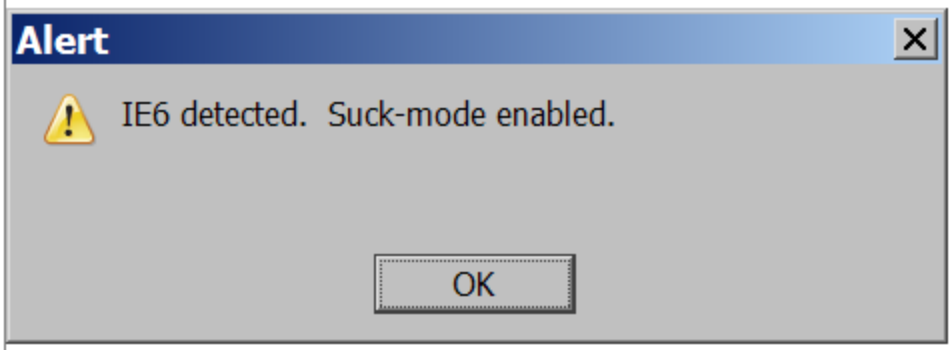
```JS
// single-line comment

/* 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

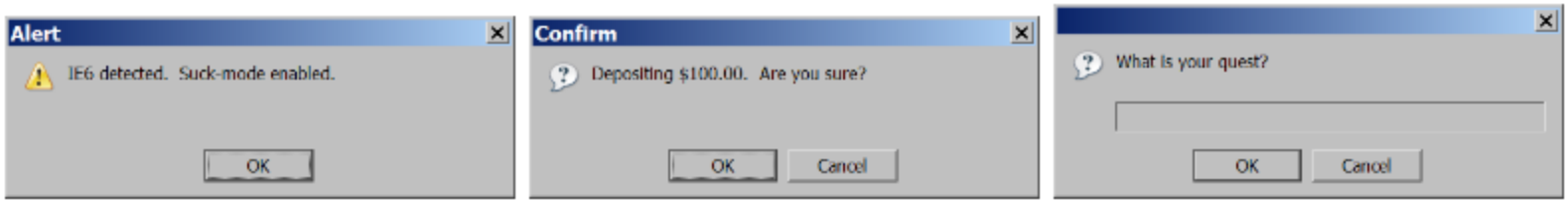


output

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

Popup boxes

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



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

Built-in types

表 7-1 JavaScript 内置数据类型

类型	描述	示例
number	数字，整数或者浮点数	42、-17、0、3.14、2.4e-6、NaN
boolean	布尔值	true、false
string	文本、字符串	“hello world”、’你好’
array	数组，一组可通过自然数下标索引访问的数据	[12, 17, ‘你好’, 0]
object	对象，包含属性和行为的实体	{name: “张三”, age: 24}

function	函数，一组可以执行的语句	function openDialog(){
		alert(“对话框打开了”);
		}

null	空	null
undefined	未定义	undefined

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

JavaScript 没有真正的整形数：JavaScript 使用 Number 类型同时表示整数和浮点数。实际上，JavaScript 中并没有真正意义上的整型数，所有的数字都表示为 64 位浮点数。只是在输出时，当浮点数没有小数部分时，省略了小数点和零。

number type

JavaScript 的 Number 类型还包括了几个特殊的常量用来表示一些特殊的数值，参见表 7-2。

表 7-2 Number 类型常数

常数	描述
number.MAX_VALUE	系统可表示的最大正有理数
number.MIN_VALUE	系统可表示的最小小数
number.NaN（或者 NaN）	不是数字
number.NEGTIVE_INFINITY	表示数值已经超出系统能表示的最大负数
number.POSITIVE_INFINITY	表示数值已经超出系统能表示的最大正数

number type

表 7-3 JavaScript 算术运算符

运算符	解释
+	加法
-	减法
*	乘法
/	除法
%	取余

number type

表 7-4 JavaScript 数学常数

常量	值
Math.PI	3.14159……，圆周率 π
Math.E	2.71828……，自然对数底 e
Math.LN2	0.69314……，2 的自然对数

表 7-5 JavaScript 数学函数

函数	解释
Math.abs(x)	绝对值
Math.ceil(x)、Math.floor(x)	天花板（向上取整）、地板（向下取整）
Math.cos(x) 、 Math.sin(x) 、Math.tan(x)	相应的三角函数
Math.log(x)	自然对数（也可给出底，计算任意底的对数）、10 为 3 的对数
Math.min(x, y, …)、Math.max(x, y, …)	最小值、最大值
Math.pow(base, exponent)	幂函数
Math.random()	随机数
Math.round(x)	四舍五入
Math.sqrt(x)	平方根

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"

表 7-8 JavaScript 字符串运算示例

表达式	值
1 + “2”	“12”
1 + “3 个和尚”	“13 个和尚”
(1+3) + “5”	“45”

More about string

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

```
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
```
JS

- accessing the letters of a string:

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

More about string

表 7-9 JavaScript 解析数字示例

表达式	值
parseInt("2")	2
parseInt("2.8")	2
parseInt("2.8 个和尚")	2
parseInt(" 2.8 个和尚")	2
parseInt("")	NaN
parseInt("有 2.8 个和尚")	NaN
parseFloat("2.8")	2.8
parseFloat("2.8 个和尚")	2.8

More about string

表 7-10 JavaScript 字符串常用方法

方法	用途
charAt(index)	在 index 处的字符，等于 str[index]
charCodeAt(index)	给出 index 处字符的编码（数字）
String.fromCharCode(code)	静态方法，将 code（数字）转换为对应的字符
indexOf(searchStr) indexOf(searchStr, fromIndex)	在 str 中从 formIndex（缺省为 0）开始查找 searchStr，找到则返回其第一次出现的位置；未找到返回-1
split(delimeter) split(delimeter, howMany)	将 str 以 delimeter 为分隔符，截断为多个字符串，将这些字符串前 howMany 个组成一个数组返回；无 howMany 参数，则返回所有
substring(start) substring(start, stop)	返回 str 从 start 到 stop 的部分，stop 缺省为 str 结束
toLowerCase()	将 str 转换为全小写字符
toUpperCase()	将 str 转换为全大写字符

Immutable vs. mutable

JavaScript 字符串不可改变!: 字符串在 JavaScript 和许多高级语言一样，例如: Java、C#等等，是不可改变的（immutable）。也就是说，一经产生，字符串本身的值就再也不会发生改变。变量赋值为字符串后，除非重新赋值，其值不变，参考源代码 7-11。PHP、C、C++等语言则不同，字符串本身的值是可以改变的，参考源代码 7-12。

- 源代码 7-11 JavaScript 字符串 immutable 示例

```
var str = "Hello";
str[0] = "W";
alert(str);    // "Hello"
```

- 源代码 7-12 PHP 字符串 mutable 示例

```
<?php
$str = "Hello";
$str[0] = "W";
echo $str;     // "Wello"
?>
```

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
  - “0” and empty array are "truthy“, which are "falsey" in PHP
- 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

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**

10 < "42"	true
10 < "42 人"	false
10 > "42 人"	false
10 == "42 人"	false
42 == "42"	true
42 == "42.0"	true
42 === 42.0	true
42 === "42"	false

  - **NaN == NaN, NaN === NaN are all false!**

Operators precedence

表 7-13 JavaScript 运算优先级（降序）

类别	操作符
括号	()
成员、索引操作符	. []
方法（函数）调用、对象创建	() new
逻辑非、负号、自增、自减、类型	! - ++ -- typeof delete void
乘、除、取模	* / %
加、减	+ -
关系比较、实例、包含	< > <= >= instanceof in
相等、不等比较	== != === !==
逻辑与	&&
逻辑非	||
赋值	= += -+ *= /= %=

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)

```
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"
```
JS

while loops (same as Java)

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

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

- break and continue keywords also behave as in Java

Variables scope

- Global and Local, the same as PHP, but without global statement when using global variables within a function

源代码 7-18 变量作用域示例

```
var a = 2;          // 全局变量

function scopeExample(){
    alert(a);       // 弹出2
    a = 2 * 2
    b = 3;          // 隐式变量声明，b为全局变量!
    var c = 5;      // 局部变量
    // 这里a为4，b为3，c为5
}
scopeExample();
//  这里a为4，b为3，c为undefined
```

Variables scope

- Function scope, not block scope

源代码 7-19 函数作用域示例

```
function scopeExample2(){
    for(var i = 0; i < 10; i++){
        var a = i * i;
    }
    // 这里i和a依然能够被访问
    alert(i);
    alert(a);
}
scopeExample2();
// i 和 a 不能被访问，undefined
```

Scope

循环变量不要忘记 var：循环变量如源代码 7-19 中的变量 i，如果忘记使用关键字 var 声明，会成为全局变量！这是个常见错误。↵

慎用全局变量：绝大多数编程语言都认为要慎用全局变量，因为它很容易被某些代码“不注意地”改动。使用全局变量还违反低耦合的原则，使用全局变量的模块间会经由它产生相互依赖。网页前端 JavaScript 代码中更要尽量避免使用全局变量，因为一个网页可能同时使用若干个脚本，包括来源不同的脚本，而所有这些脚本中的全局变量，都同时可以被其它脚本访问，这非常容易造成名称冲突和变量值被错误修改的问题。↵

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

- auto-increasing size
- different types of elements
- two ways to initialize an array
- length property (grows as needed when elements are added)

Arrays

源代码 7-20 数组示例

```
var a = [];                 //空数组
a[0] = 23;                  //23存储在数组下标为0处

var a2 = ["some", "strings", "in", "an", "array"];
a2.push("Ooh~");            //在数组末尾添加"Ooh~"
alert(a2.length);           //6
a2[100] = "Yeah!"           //将数组大小变为101，并给下标100处赋值"Yeah!"

alert(a2[99]);              //undefined
alert(a2.length);           //101
```

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

方法	用途
concat(array1, ..., arrayN)	将多个数组拼接成为一个
join() join(separator)	将数组元素以 separator 为分隔符拼接成为一个字符串
pop()	弹出数组最后一个元素
push(value) push(value1, ..., valueN)	将一个或者多个值压入数组
shift()	从头部取出一个元素，并依次向前移动其余元素
unshift(value) unshift(value1, ..., valueN)	向头部添加一个元素，并依次向后移动其余元素
reverse()	改变当前数组，将其顺序翻转
sort()	改变当前数组，将其排序
slice(startIndex) slice(startIndex, endIndex)	返回当前数组的子数组，子数组从 startIndex 开始，到 endIndex 结束，endIndex 缺省为数组长度
splice(index, count, value1, ..., valueN)	将当前数组从 index 起 count 个元素删除，并更换插入给定的多个值（value1, ..., valueN）

Splitting strings: split and join

```JS
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"
```

- 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

Array example

源代码 7-21 JavaScript 数组方法使用示例

```
var userInfo = "张三|男|28|13860000660|zhangsan@mail.cn";
var userInfoArray = userInfo.split("|");
// [张三, 男, 28, 13860000660, zhangsan@mail.cn]

var nameAndGender = userInfoArray.slice(0,2).join(", ");
alert(nameAndGender); // 张三, 男

userInfoArray.splice(3, 1, "中山大学", "18509087532");
// [张三, 男, 28, 中山大学, 18509087532, zhangsan@mail.cn]

alert(userInfoArray.join("|"));
// 张三|男|28|中山大学|18509087532|zhangsan@mail.cn
```

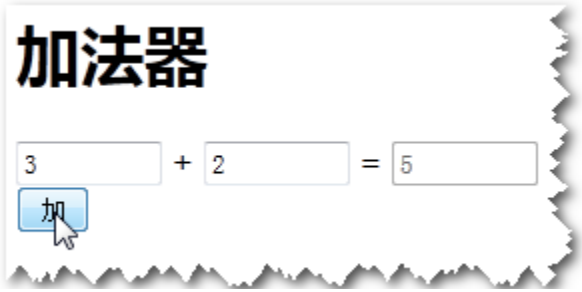
eval

源代码 7-28 eval 函数用法示例

```
eval("3 + 2 - 5 * 0");   // 5
var str = "Hello";
eval("str = str + ' World!'");
eval(alert(str)); // 弹出'Hello World!'
```

慎用 eval：eval 函数能够动态执行源代码，必须慎用。特别是当执行的源代码直接来自用户的收入，更要格外小心。如果不小心，造成代码注入的安全问题。

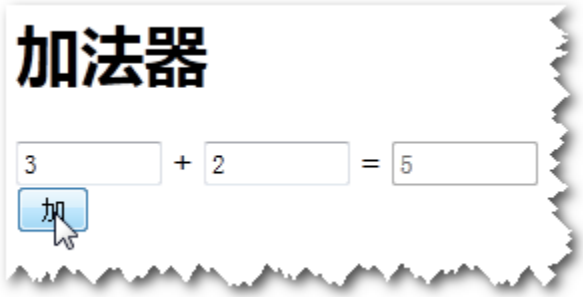
Simple Front-end App.

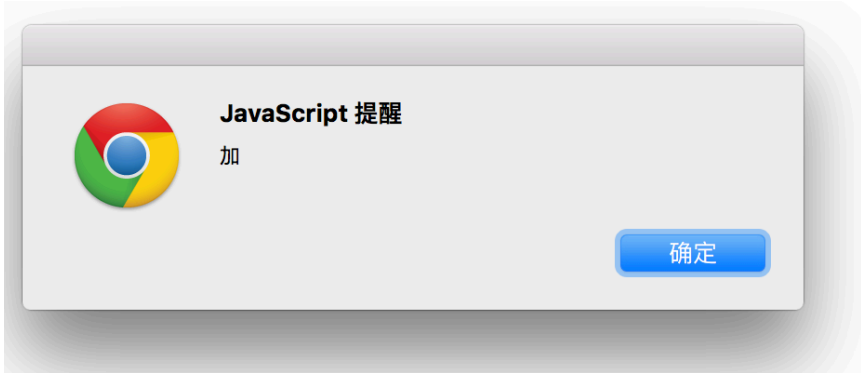



```
<h1>加法器</h1>
<div>
    <input id="left-operand"/>  +  <input id="right-operand"/>
    = <input id="result" disabled="disabled"/><br />
    <button id="add-button">加</button>
</div>
.....
```

```
window.onload = function(){
  document.getElementById("add-button").onclick = function(){
    var leftOperand = parseInt(document.getElementById("left-operand").value);
    var rightOperand = parseInt(document.getElementById("right-operand").value);
    var result = leftOperand + rightOperand;
    document.getElementById("result").value = result;
```

Simple Front-end App.






```
document.getElementById('add-button').onclick = function(event){
  alert(event.target.textContent); // 加
}
```

Calculator

Requirement Analysis

计算器通常的用例如下：

1. 响应用户对数字和算术操作符按钮的操作，记录并显示用户通过按钮输入的算术表达式。
2. 响应用户功能按钮的操作。

a) 用户按下“←”按钮，删除当前算术表达式最后一个字符，并更新显示。

b) 用户按下“CE”按钮，清除当前算术表达式。

c) 用户按下“=”按钮，计算当前表达式的结果并显示。

i. 如果，算术表达式非法，弹出警告框提醒用户，并终止计算。

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!