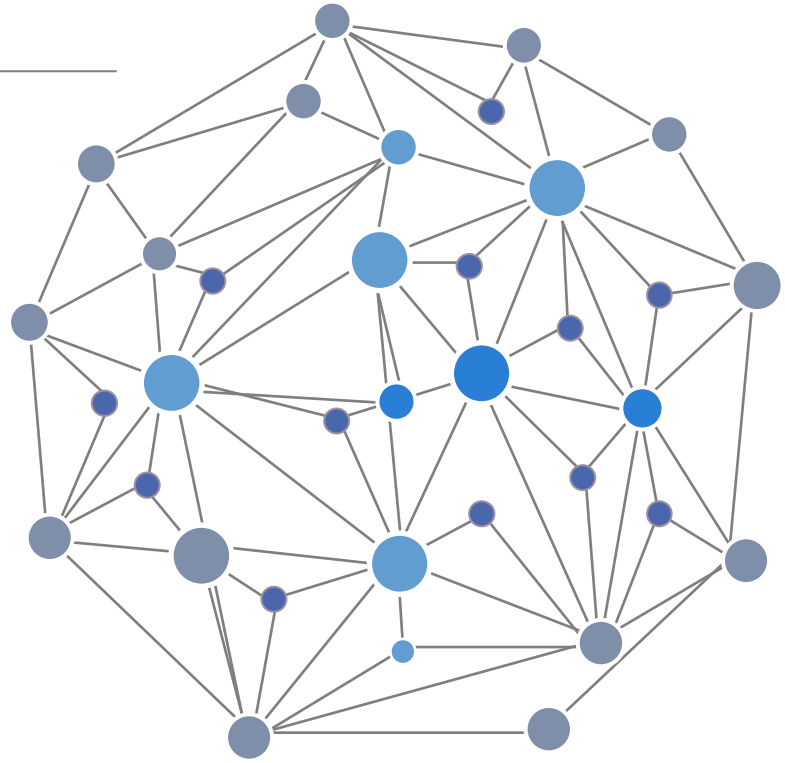
Web 程序设计

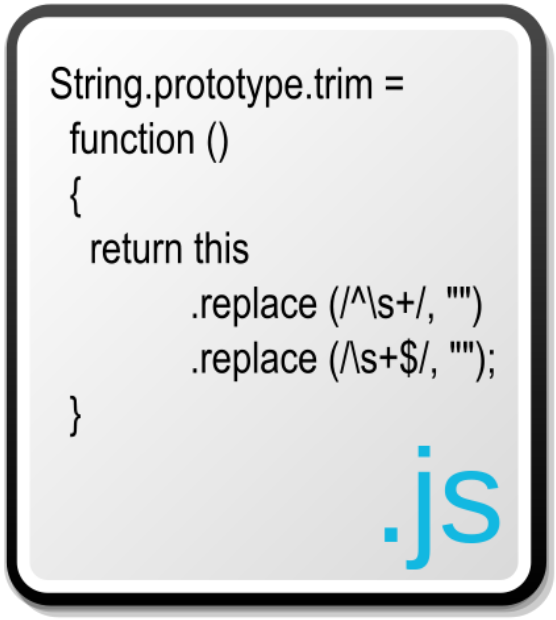


第六讲 JavaScript 技术基础介绍

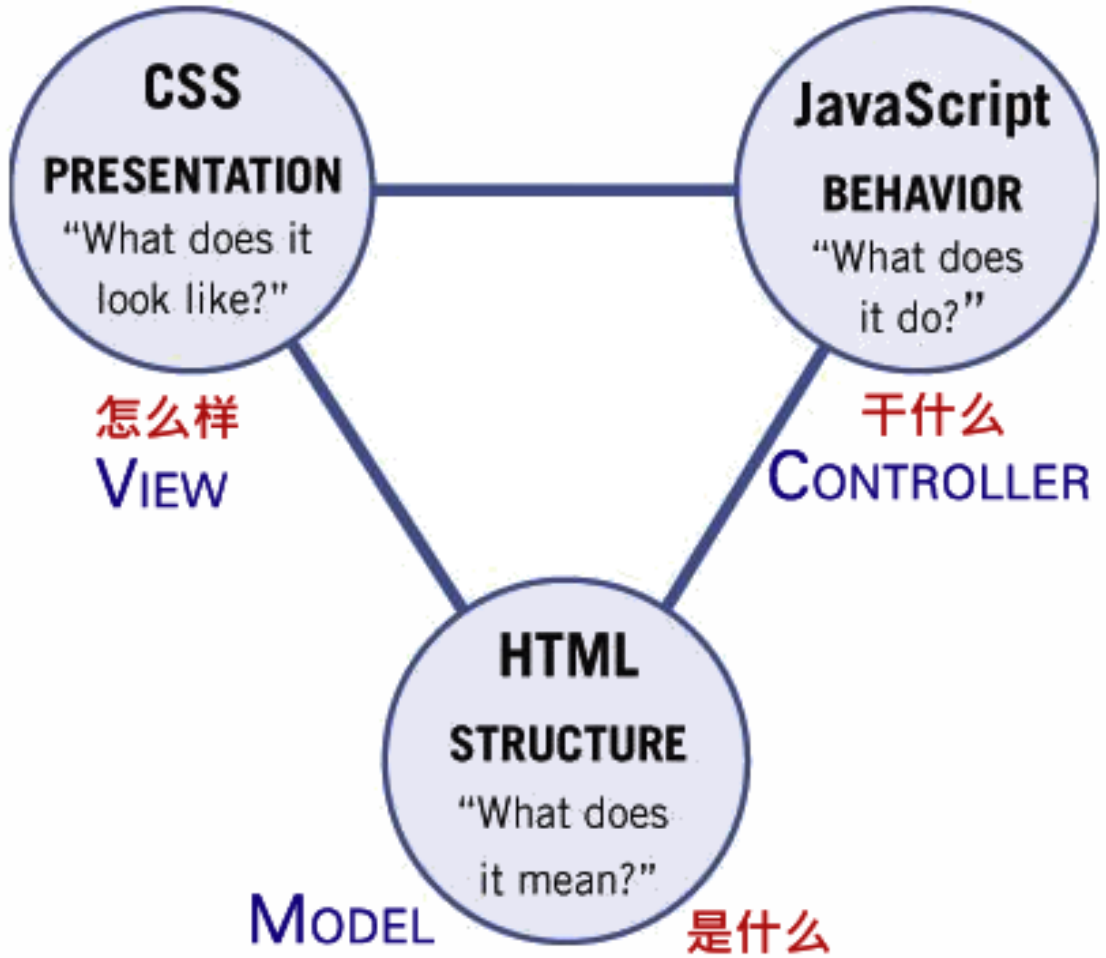
福州大学 计算机与大数据学院
软件工程系 陈昱

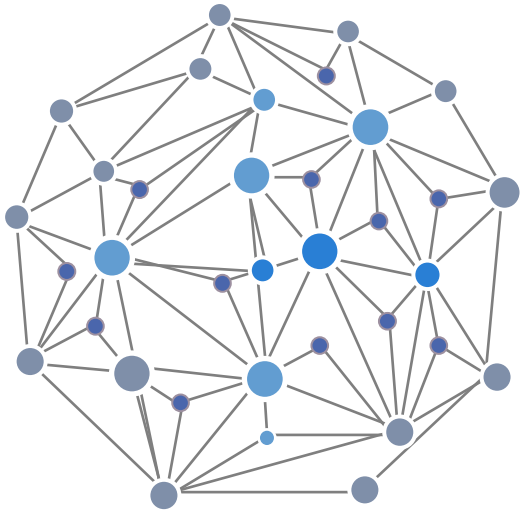
内容提要

- 什么是 JavaScript
- JavaScript 语法基础
- JavaScript 标准对象
- 浏览器对象模型 BOM
- 关于 JavaScript 的阅读资料



W3C 三层结构:客户端的"MVC"模式



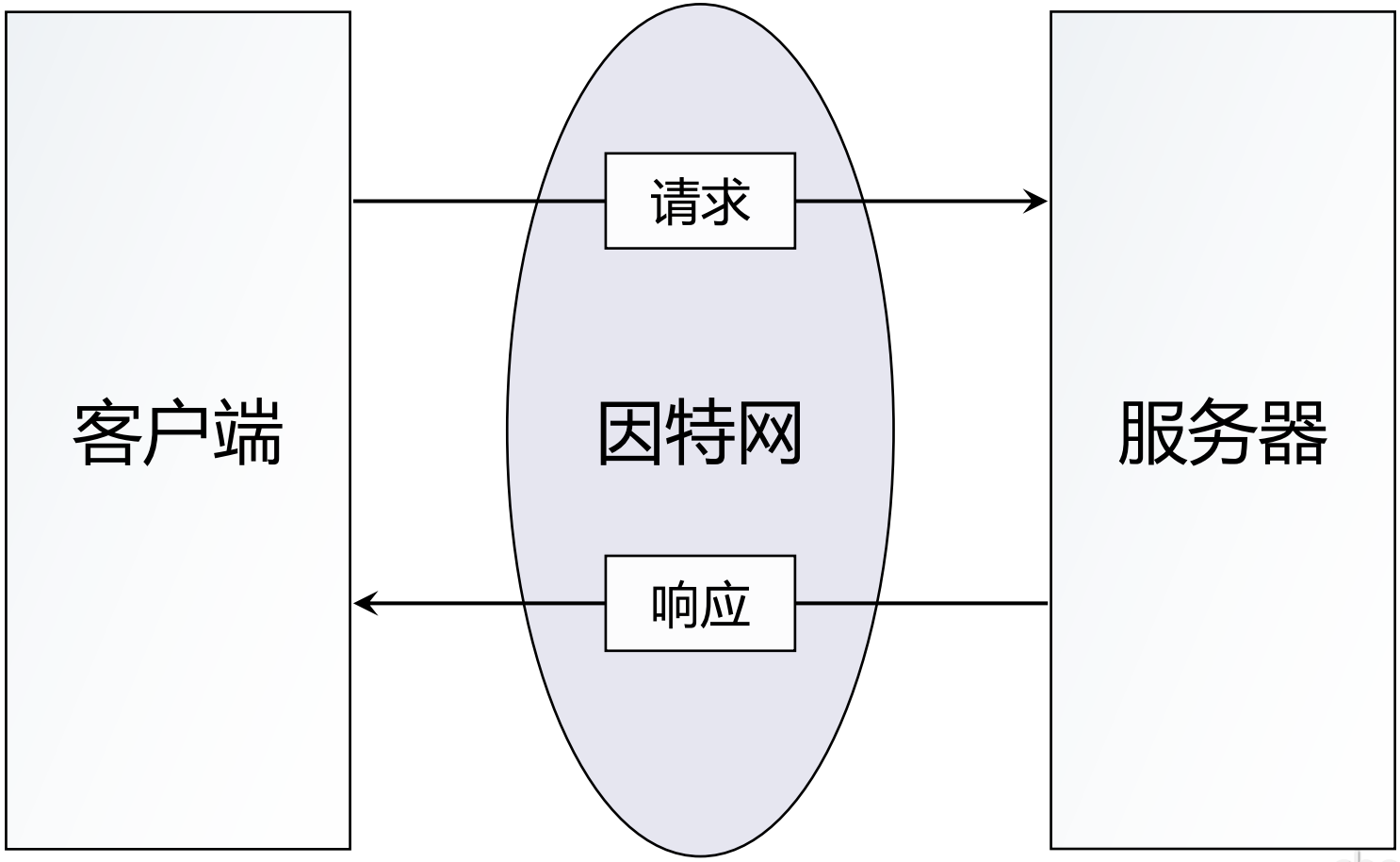


什么是 JavaScript

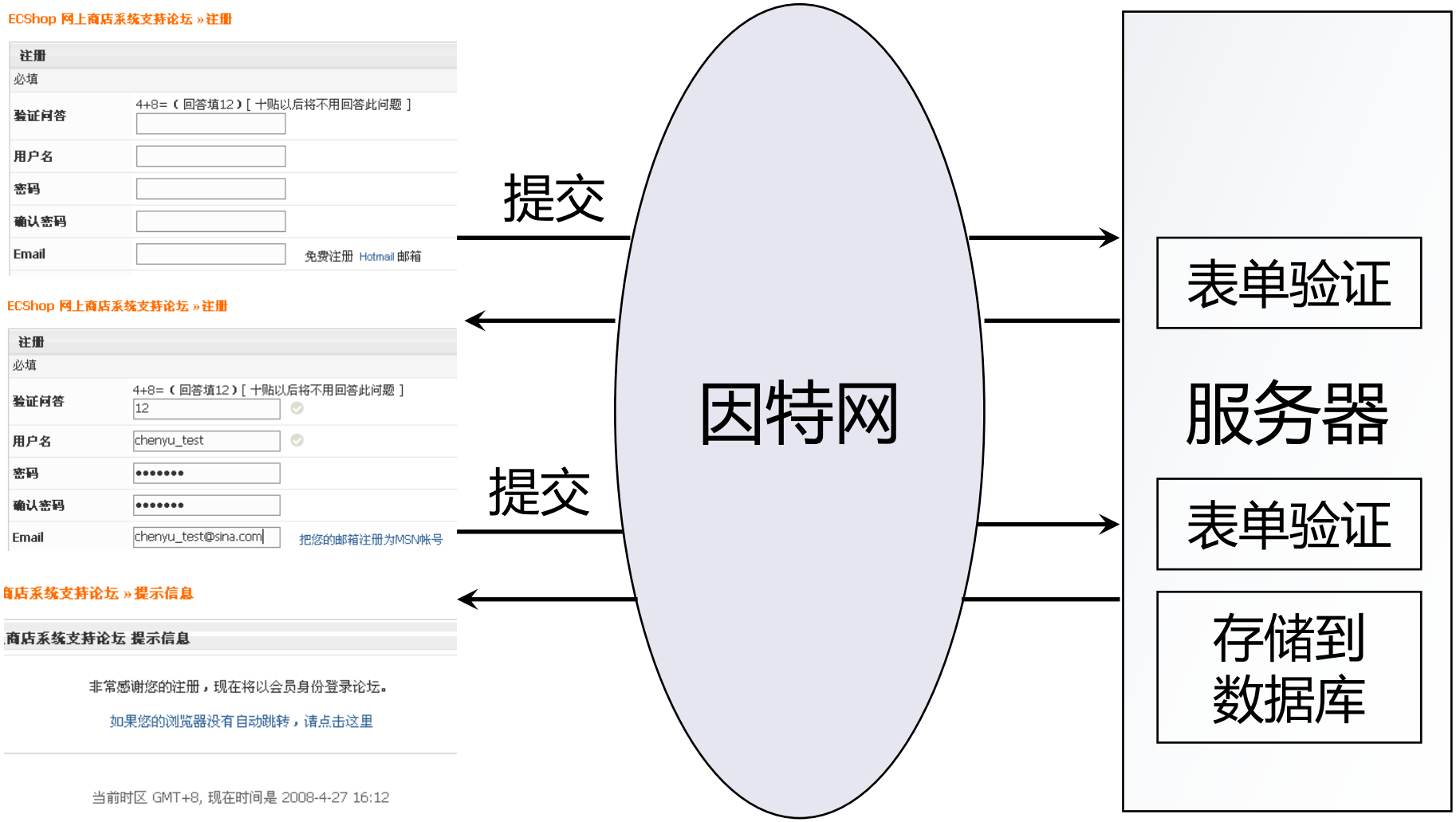
表单验证 Form Validation

客户—服务器通信模型

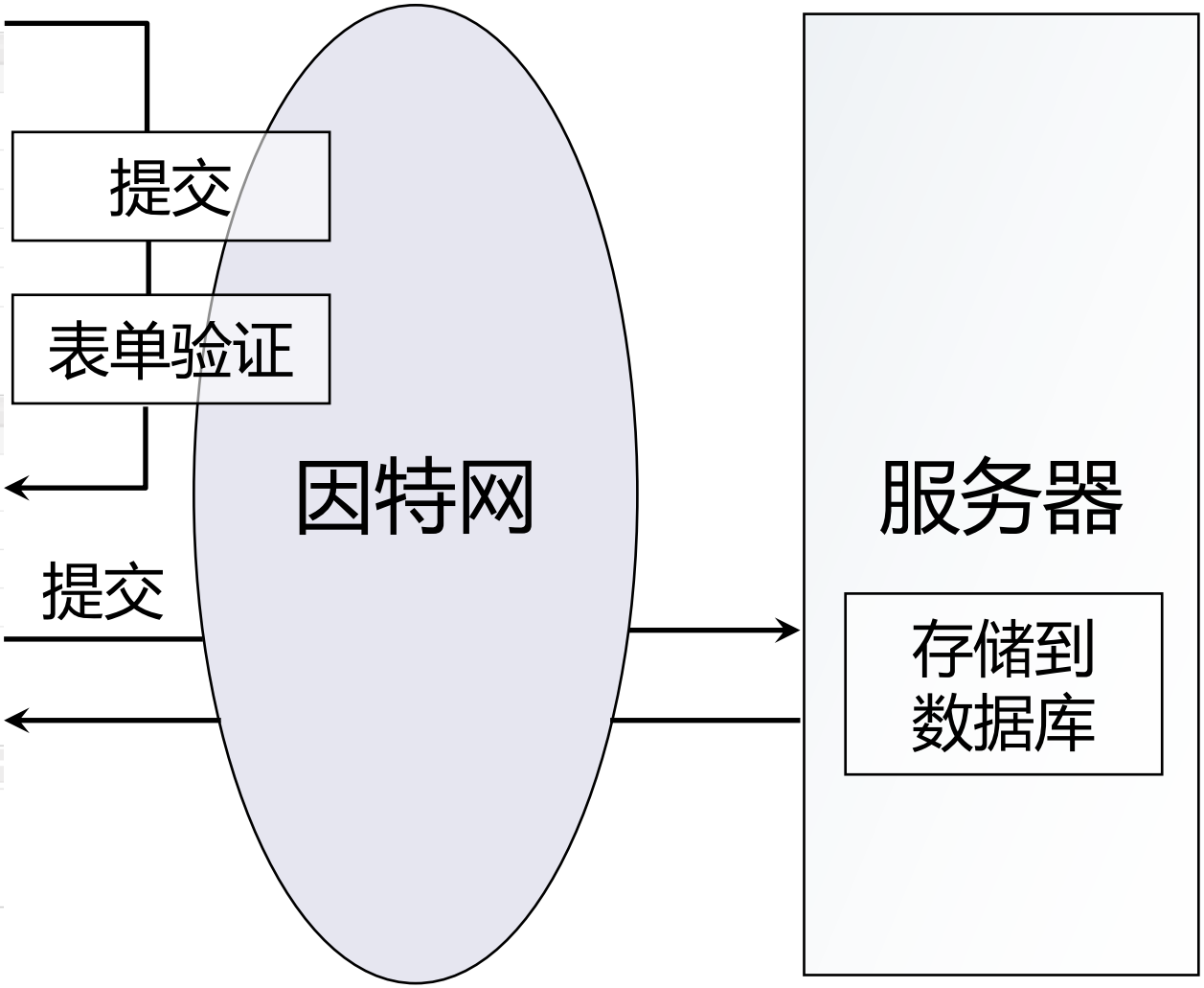
- 客户端发起请求，服务器端响应请求



JS 没出现前的表单验证过程



使用 JS 的表单验证过程



客户端编程 vs. 服务器端编程

- PHP 已经允许我们创造动态网页了，为什么我们还要使用客户端脚本呢?
- 客户端脚本 (JavaScript) 好处:
  - 可用性: 不用传送给服务器就能修改页面 (更快的UI)
  - 高效: 可以不用等待服务器，快速对页面进行小的修改
  - 事件驱动: 可以对用户的动作做出响应，例如点击鼠标或者敲击键盘
- 服务器端编程 (PHP) 好处:
  - 安全性: 通往服务器私有数据的接口; 客户端不能看到源代码
  - 兼容性: 不受制于浏览器的兼容性问题
  - 功能强大: 可以写文件, 连接到服务器, 连接到数据库, ...

JavaScript 概述

- JavaScript 是什么?
  - JavaScript 是一种脚本语言 (轻量级的编程语言)
  - JavaScript 是一种解释语言（不经过编译)
  - 运行在客户端的浏览器上，由浏览器解释执行
  - JavaScript 被设计用来向页面添加交互行为
  - 被绝大多数浏览器支持

JavaScript 能做什么

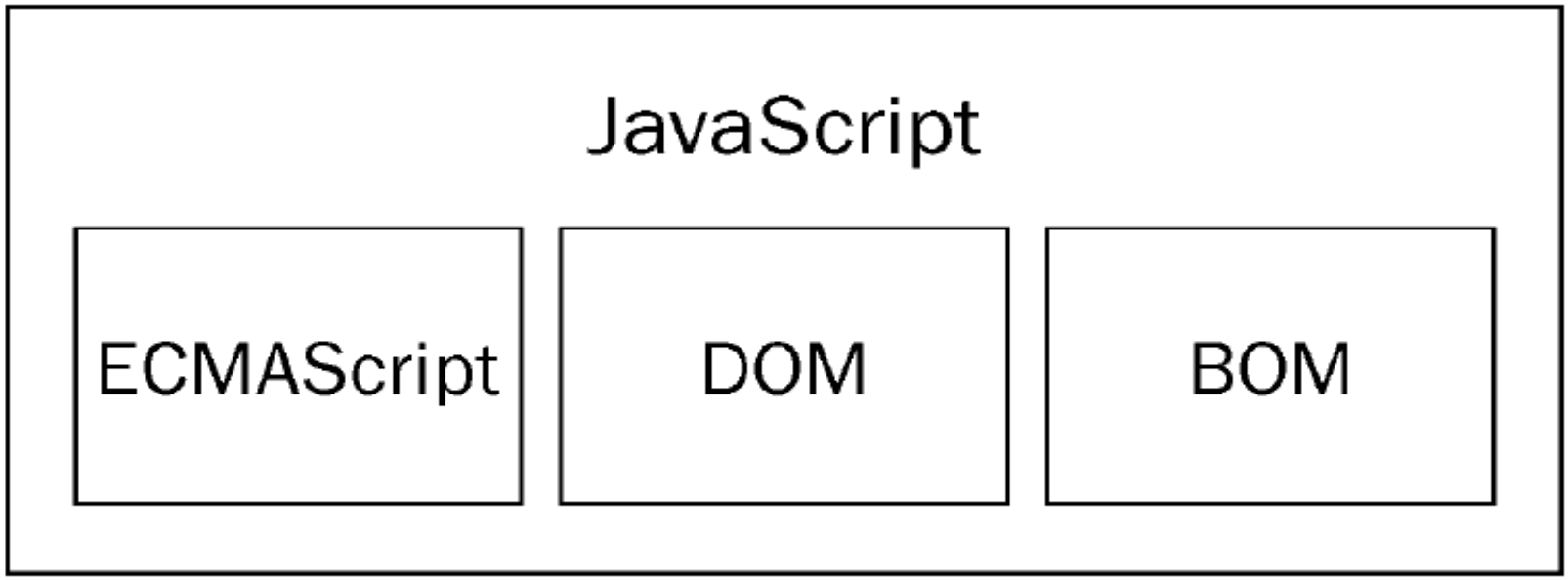
1. 让客户端拥有“计算”的能力
   - 比如表单验证（字符串比较，模式匹配）
2. 操纵浏览器行为
   - 比如弹出式广告，弹出对话框
3. 能够响应用户事件
   - 比如点击、键入、拖动
4. 修改(添加，删除，移动) (X)HTML 元素
5. 修改元素的 CSS 属性
6. 发起 HTTP 请求，获取数据

JavaScript 不能做什么

1. 不能读写计算机文件系统中的文件
   - filesystem.read('/my/password/file');
   - filesystem.write('horridvirus.exe');
2. 不能执行任何其他程序
   - execute('horridvirus.exe')
3. 不能和其他计算机建立连接，除了通过 HTTP 协议发起请求
   - var security_hazard = connection.open('malicious.com');

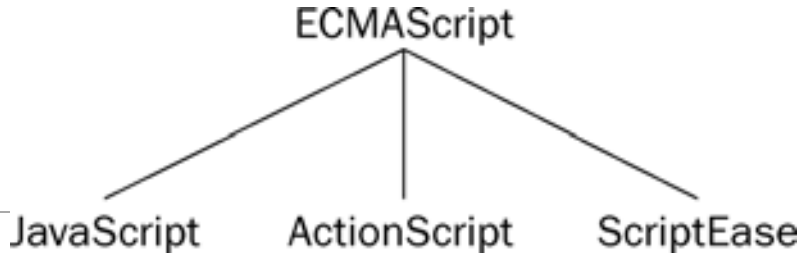
JavaScript 的组成

- 核心 (ECMAScript)　　*ECMA*
- 文档对象模型 (DOM)　　*W3C*
- 浏览器对象模型 (BOM)　*Browser Vendor*



http://www.w3school.com.cn/js/pro_js_implement.asp

JavaScript 的组成

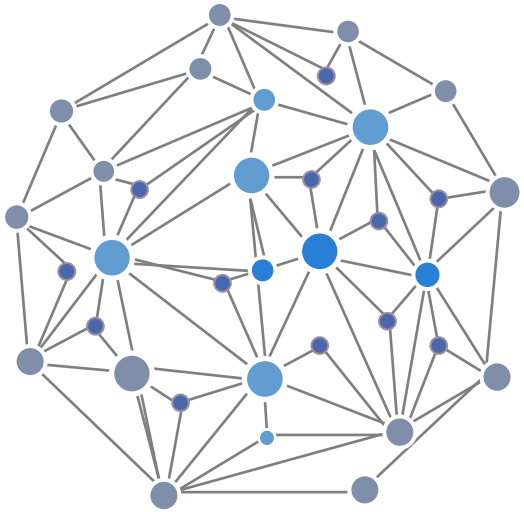


- ECMAScript 描述了语法，定义了语言
  - ECMAScript 除了用于定义 JavaScript，还用于 Flash 中的 ActionScript 等语言
  - 2015年6月17日，ECMAScript 6 (ES6)发布正式版本，即 *ECMAScript 2015*
- DOM（文档对象模型）是操作 HTML 和 XML 文档的应用程序编程接口（API）
- BOM（浏览器对象模型）可以对浏览器窗口进行访问和操作的 API

概念辨析：DHTML 是什么?

- Dynamic HTML
- 一种通过结合(X)HTML 、层叠样式表 (CSS) 、用户端脚本语言（如 JavaScript）和文档对象模型 (DOM) 来创建动态网页内容的方法

- DHTML 只是一种制作网页的概念，事实上并没有存在一种技术标准叫 DHTML



JavaScript 语法基础

JavaScript vs. PHP

- 相似点：
  - 都是解释型，不是编译型
  - 都是语法、规则、类型宽松的
  - 都是大小写敏感的
  - 都提供了强大的文本处理，内置正则表达式
- 不同点：
  - JS 是更加面向对象的，如：noun.verb (), 较少面向过程，如: verb (noun)
  - JS 关注于用户接口和与文档的交互；PHP 更适合 HTML 输出和文件/表单的处理
  - JS 代码运行在客户端浏览器上; PHP 代码运行在服务器上

JavaScript 语法基础

- 如何放置 JavaScript 代码
  - 外链式
  - 内嵌式
- 数据类型 & 表达式
- 控制语句
- 函数定义

外链 JavaScript 文件

- 将 JavaScript 写入一个外部文件之中，一般以 .js 为后缀
- script 元素可以放在 head，也可以放在 body, 差别在于执行时机不同

```
<head>
  <script type="text/javascript" src="script.js">
  </script>
</head>
<body>
</body>
```

直接内嵌 JavaScript

- 使用<script>元素，type 属性来定义脚本语言，如果是 XHTML，用 CDATA 包含程序内容，如果是 HTML 不需要 CDATA。

```
<html>
<body>
    <script type="text/javascript">
    //<![CDATA[
    alert('Hello world!');
    //]]>
    </script>
  </body>
</html>
```

直接内嵌 JavaScript

- 内嵌的 JavaScript 也分两种
  - 一种位于head部分，一种位于body部分

```
<html>
<head>
<script type="text/javascript">
 function message() {
   alert("This alert box was called with the onload event");
 }
</script>
</head>
<body onload="message()">
</body>
</html>
```

常见写法

- 可以把所有的 js 相关代码，包括外链和内嵌的 <script> 放置在 </body> 的前面

- 这样可以加快页面的加载速度
- 而且加载执行 js 的时候，页面的 HTML 代码已经准备就绪

提示

- 外部脚本不能包含 <script> 标签

- <script src="myScript.js"></script> 不能写成简写属性的形式
  - 不能写<script src="myScript.js" />

基本语法

- 每个语句后用；结束（非必须，但推荐）
- 语句块用 {...} 表示
- 注释类似 C++/Java

  // ……　　单行注释

  /* … */　　多行注释

数据类型

- Number
  - JavaScript不区分整数和浮点数，统一用Number 表示；十六进制用0x前缀
  - 123；0.456; 1.2345e3; 0xa5b4c3d2;
  - NaN; Infinity;
- 字符串
  - 以单引号 ' 或双引号 " 括起来的任意文本
  - 'abc'; "xyz"
- null；对象 ……

逻辑值与逻辑运算符

- 逻辑值： true false
- 逻辑运算符： < >= <= && || === ==！ !== !=
- 大多数逻辑操作符会自动转换类型:
  - ✓5 < “7” 为 true
  - ✓42 == 42.0 为 true
  - ✓“5.0” == 5 为 true
- === 和 !== 是严格相等检测，同时检查类型和值:
  - ➢“5.0” === 5 为 false

变量和赋值

- 变量名规则和 C++ 一样
  - 大小写英文、数字、$和_的组合，对大小写敏感
- 变量声明中没有类型（弱类型语言）

  var *name1*, *name2*, ...;
- 给变量赋值

  var foo = "Hege";
- 不使用 var 也可以动态创建变量：

  foo = some value;

表达式

- 和 C++ 非常类似

- 算术运算符: + - * / % ++ -- etc. as in C++
- 逻辑运算符: <, <=, == ...
- 字符串连接符: +
  - 可以将其他类型数据转换成字符串

基本输入输出

- 输出：
  - document.write(str); // 在页面上插入字符串
  - alert("msg"); // 弹出警告对话框
- 输入： val = window.prompt("msg");
- 调试输出： console.log(val); // F12 console

基本的控制结构

- if (*cond*)

  stmt

  else // optional else part

  stmt

- while (*cond*)

  stmt

- for (*init; test; incr*)

  stmt

基本的控制结构

- switch (choice) {
  case val: *stmt*
  break;
  …
  default: // optional
  stmt
  }
- do *stmt*
  while (cond);

```
<html><head>
  <title>Folding Puzzle</title>
</head>
<body>
 <script type="text/javascript">
    distanceToSun = 1.49e8*1000*1000; // 地球到太阳的距离
    thickness = .007; // 纸的厚度, 0.007毫米

    foldCount = 0;
    while (thickness < distanceToSun) {
        thickness *= 2;
        foldCount++;
    }
    document.write("Number of folds = " +  foldCount);
  </script>
</body></html>
```

将一张纸对折多少次能到达太阳?

定义函数

- function fname(parm1, parm2, ...)

  {

  //code

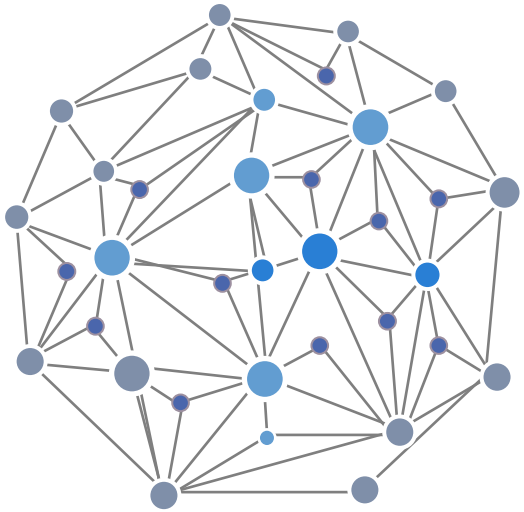
  return expr;

  }
- 注意
  - 没有返回值则无需返回语句

```
function isPrime(n)
// Assumes: n > 0
// Returns: true if n is prime, else false
{
  if (n < 2) {
    return false;
  }
  else if (n == 2) {
    return true;
  }
  else {
      for (var i = 2; i <= Math.sqrt(n); i++) {
        if (n % i == 0) {
          return false;
        }
      }
      return true;
  }
}
```

- 参数列表没有类型声明
- 返回值没有类型

- 第一次使用的变量如果前面标上 var，则表明它是局部变量



JavaScript 标准对象

JavaScript 对象

- JavaScript 是面向对象的编程语言
- 对象是一种特殊的数据结构，对象拥有属性和方法
- JavaScript 内建了许多对象

- JavaScript 的面向对象特性逐渐在加强：
  - 抽象，继承，封装和多态

JavaScript 对象

- JavaScript 对象是包含多个命名值的集合
  - var person = {firstName:"Bill", lastName:"Gates", age:62, eyeColor:"blue"};

- 对象属性： 对象中的命名值，被称为**属性。**以**“名称:值”**对书写。

- 对象方法：方法是在对象上执行的**动作。**方法以函数定义被存储在属性中。

创建对象

- 对象用 new 创建

  personObj=new Object();

- 动态创建属性

  personObj.firstname="John";
  personObj.lastname="Doe";
  personObj.age=50;
  personObj.eyecolor="blue";

定义方法

- JavaScript 中为对象定义方法的写法比较特殊

```
personObj.eat = function() {
        ..........
}
```

- 函数名可以被赋值给方法名、事件响应函数（因为事实上，函数也是个对象，Function Object）

构造函数

```
function person(firstname,lastname,age,eyecolor) {
  this.firstname=firstname;
  this.lastname=lastname;
  this.age=age;
  this.eyecolor=eyecolor;
}

myFather=new person("John","Doe",50,"blue");
myMother=new person("Sally","Rally",48,"green");
```

联合使用构造函数和原型方法

```
function Car(sColor,iDoors,iMpg) {
  this.color = sColor;
  this.doors = iDoors;
  this.mpg = iMpg;
  this.drivers = new Array("Mike","John");
}

Car.prototype.showColor = function() {
  alert(this.color);
};

var oCar1 = new Car("red",4,23);
var oCar2 = new Car("blue",3,25);

oCar1.drivers.push("Bill");

alert(oCar1.drivers);      //输出 "Mike,John,Bill"
alert(oCar2.drivers);      //输出 "Mike,John"
```

动态原型方法

```
function Car(sColor,iDoors,iMpg) {
  this.color = sColor;
  this.doors = iDoors;
  this.mpg = iMpg;
  this.drivers = new Array("Mike","John");

  if (typeof Car._initialized == "undefined") {
    Car.prototype.showColor = function() {
      alert(this.color);
    };

    Car._initialized = true;
  }
}
```

数学对象 Math

- 数学对象包含了许多函数和常量
  - Math.sqrt
  - Math.pow
  - Math.abs
  - Math.max
  - Math.min
  - Math.floor
  - Math.ceil
  - Math.round

  - Math.PI
  - Math.E

  - Math.random 函数返回 [0..1) 之间的随机数

```
a = Math.round(4.7);
b = Math.random();
c = Math.floor(Math.random()*11);
```

日期对象 Date

- 用于处理日期和时间

```
today = new Date(); // sets to current date & time
newYear = new Date(2002,0,1); //sets to Jan 1, 2002 12:00AM
```

- newYear.getYear()
- newYear.getMonth()
- newYear.getDay()
- newYear.getHours()
- newYear.getMinutes()
- newYear.getSeconds()
- newYear.getMilliseconds()

字符串 String

- 类 String 用于处理字符数据
- 字符串常量用 " " 包含
- 可以使用 C/C++ 的 \? 形式的转义字符
- length 属性可以得到字符串的长度
- 拥有许多处理字符串的方法
  - 比如 toUpperCase() 将字符串转换成大写

```
var txt="Hello World!";
document.write(txt.length);
document.write(txt.toUpperCase());
```

数组 Array

- 定义数组

  var list = new Array(size);
- 通过下标访问：list[index];
  - 下标从 0 开始
- 可以保存任意类型的值
- list.length 返回数组长度

数组初始化

- 初始化的两种方法

```
var mycars=new Array();
mycars[0]="Saab";
mycars[1]="Volvo";
mycars[2]="BMW";
```

```
var mycars=new Array("Saab","Volvo","BMW");
```

- 初始化时可以有数组元素未分配值

二维数组

- 声明一个数组作为行
- 为数组中的每个元素分配一个数组
- 不需要保持矩形的形式
- 可以通过 Array[i][j] 的形式访问

引用传递参数

- 数组和对象在作为参数传递给函数时，是作为引用传递
- 标量（数值等）作为参数是值传递

for...in 循环

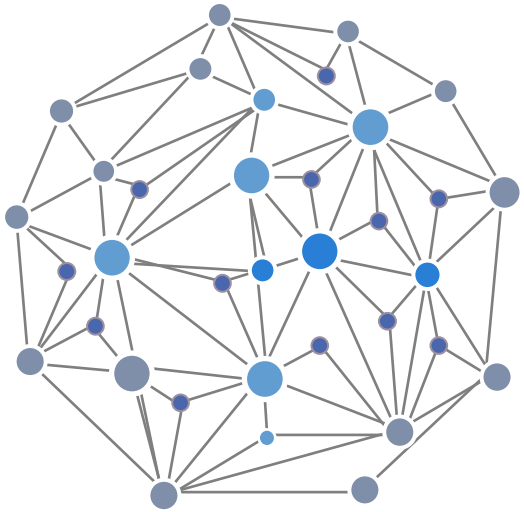
- for...in 循环主干部分里的代码针对每个元素属性执行一次

```
for (变量 in 对象) {
        在此执行代码
}
```

for...in 循环

```
<script type="text/javascript">
var x
var mycars = new Array()
mycars[0] = "Saab"
mycars[1] = "Volvo"
mycars[2] = "BMW"

for (x in mycars) {
    document.write(mycars[x] + "<br />")
}
</script>
```



BOM 浏览器对象模型

一些与浏览器密切相关的对象

由浏览器制造商定义和实现

BOM 对象

- window 对象　　控制浏览器窗口
- navigator 对象　　包含浏览器信息
- screen 对象　　用来获取屏幕大小等
- location 对象　　当前网页的地址信息
- history 对象　　浏览历史

window 对象

- 用于控制浏览器窗口

```
// 弹出窗口（广告，海报。。。）
window.open('page.html','popup',
                'width=300,height=400');
```

```
// 修改状态栏上的文字
window.status = 'Take a look at this wonderful site!';
```

navigator 对象

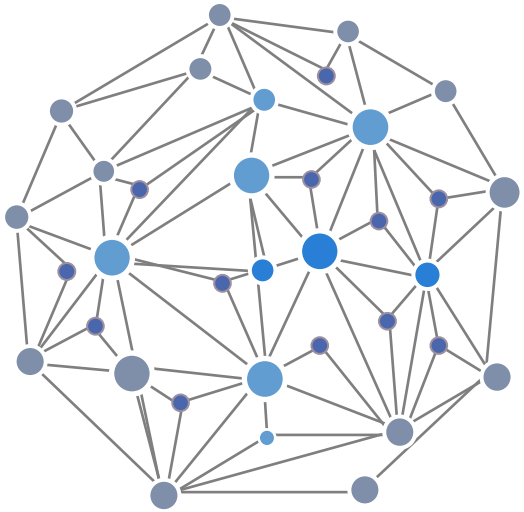
- navigator 对象包含了有关访问者浏览器的所有信息
- navigator.appName 给出浏览器的名字
- navigator.appVersion 给出浏览器的版本

```
if (navigator.appName == "Netscape") {
    document.write('<link rel=stylesheet '+
      'type="text/css" href="netscape.css">');
  }
  else {
    document.write('<link rel=stylesheet ' +
      'type="text/css" href="MSIE.css">');
  }
```

第六讲课后练习

- 学习
  - 课本 第 19 章
  - W3Schools.com JavaScript 教程中文版
    - http://www.w3school.com.cn/js/

- 编程作业 hw4



关于 JavaScript 的阅读材料

手册，书籍

中文在线教程

- w3schools 中文版
  - JavaScript 教程
    http://www.w3school.com.cn/js/
  - JavaScript 高级教程
    http://www.w3school.com.cn/js/index_pro.asp
  - HTML DOM 教程
    http://www.w3school.com.cn/htmldom/
  - DHTML 教程
    http://www.w3school.com.cn/dhtml/

中文在线教程

- w3schools 中文版
  - jQuery 教程
    https://www.w3school.com.cn/jquery/
  - AJAX 教程
    https://www.w3school.com.cn/ajax/
  - JSON 教程
    https://www.w3school.com.cn/json/

JS and DOM 参考手册

- 完整的 JavaScript 参考手册：
  - JavaScript 本地对象和内置对象
  - Browser 对象（BOM）
  - HTML DOM 对象

- http://www.w3school.com.cn/jsref/index.asp

Extended Reading

- ppk on JavaScript
  by Peter-Paul Koch
- http://www.quirksmode.org
  - 作者网站拥有丰富参考资料
- 很不错的初级教程
  - 淘宝 UED 团队翻译
  - UED: 用户体验设计

Extended Reading

- Professional JavaScript for Web Developers 4rd
- JavaScript 高级程序设计（第4版）
- 相当全面深入的 JavaScript 教程

JavaScript 框架学习

- JQuery
  - 直接看官方文档和源码
    http://www.jquery.org
  - 中文入门教程
    http://www.k99k.com/jQuery_getting_started.html

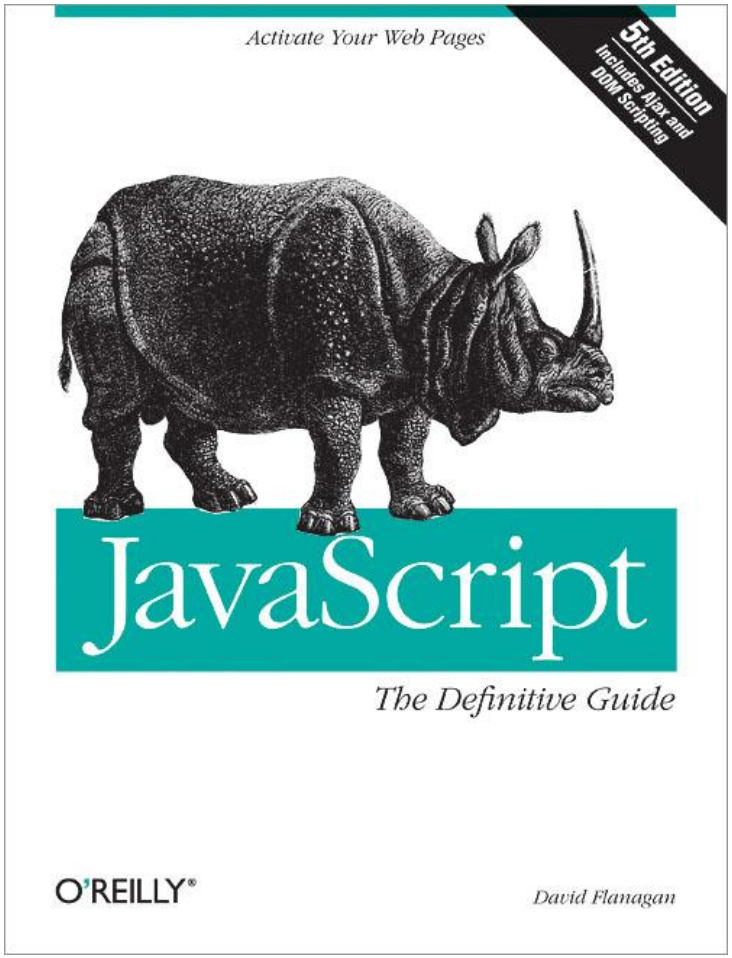
JavaScript Reference

- O'Reilly

  JavaScript: The Definitive Guide

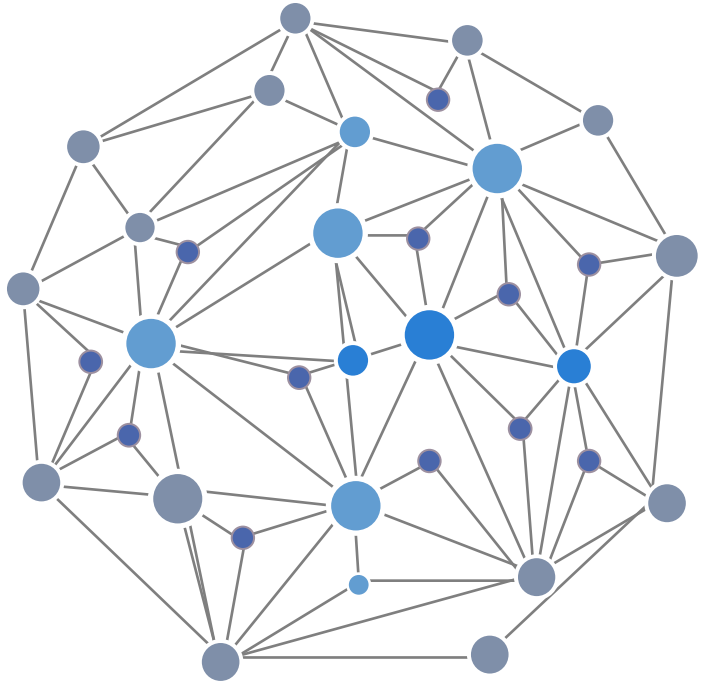
  6th Edition

- JavaScript 权威指南（第六版）



THANKS

本章结束



陈昱

福州大学福州大学 计算机与大数据学院 软件工程系