**《js前端开发（下）》教学大纲**

**编写人：赵晓茹**

1. **为什么学习本门课程：**

**学习的关键是培养学生综合的职业能力，为企业培养高素质的技能型人才。在日常教学，尤其是专业课程教学中，教师在教授学生专业知识的同时，要着力提高学生的职业素养和综合能力。情境教学以能诱发学生学习主动性，强化学生学习感受性，学习过程贯穿实践性，关注学生发展的原则，成为提高学生职业素养的有效途径。**

**1、JavaScript无处不在:**

JavaScript可运行在所有主要平台的所有主流浏览器上，也可运行在每一个主流操作系统的服务器端上。现如今我们在为网站写任何一个主要功能的时候都需要有懂能够用JavaScript写前端的开发人员。而后端代码是用什么编写的则无所谓——不管是Java，PHP，.NET，Node.js还是其他——但是客户端就一定需要一个JavaScript开发人员。事实上，现在很多开发人员依然没有认识到JavaScript有多么强大。不知道正在阅读的你是否也是这些井底之蛙的一员。如果你确实是一个了不起的JavaScript程序员，那么基本上所有有关于服务器端平台的工作都能够胜任。如果再了解这些平台基本上就可以上任了。

1. **市面需求：**

随着JavaScript的日渐成熟，我们对JavaScript程序员的需求正在持续增长，你会发现JavaScript出现得越来越多，甚至领先于其他计算机语言。最后，再看看市面上的网站，你会发现JavaScript会无处不在的活跃中。

1. **更好的薪酬：**

根据现目前的的薪资调查显示，JavaScript架构师的薪酬待遇已经处于很好的一个阶段了，而且随着JavaScript需求的增长，你可以预见它的薪酬肯定还会提高。

关于薪酬需求呈现的画面略有不同。虽然我们对JavaScript的需求是显而易见的，但是很明显，几种需求较小的语言反而薪酬更高。我怀疑这是因为许多公司仍然认为JavaScript是任何一个程序员都会用的语言。但是，当他们认识到“任何一个程序员都会JavaScript”，实际上是“任何一个程序员只会写蹩脚的JavaScript代码”的时候，也是他们经历惨痛教训的时候。然后需要真正的JavaScript程序员进入清理留下的烂摊子。

**4、JavaScript程序员更能被委以重任**

比如一个公司唯一一个用这样的方式写JavaScript以消除变量对全局范围的污染的程序员。这只是最简单的例子。很多人称自己为JavaScript程序员，却不知道这些基本知识。所以，JavaScript大有可为。

而且不要忘了，现在很多支持性能分析和内存泄漏的工具才刚处于起步阶段。如果你现在就掌握了这些技能，那么将来肯定可以更好地推销自己。

1. **学习本门课程的方法**

**本课程在教学过程中，将采用任务驱动配合案例教学，在解决困难以后，提出“任务需求—分析任务思路—引出解决问题”的方法，层层深入，从语法到案例，最后引导学生自主完成任务。**

1. **学习本门课程的方法**

新一轮基础教育课程改革,不仅表现在课程目标、结构、内容等方面的变化, 尤其集中地表现在对学生学会学习的倡导.本文以历史上教育家对培养学生掌握学习方法的重视以及这一思想的时代意义为切入点,针对当前教育——无论是理论层 面,还是实践层面——对培养学生掌握学习方法极度忽视的现实,提出和论述了将学习方法作为一门课程在实施的现实意义,并进一步论述学习方法这门课程的实施与评价的可能方式。

1. **教学目标**
2. **知识目标（学什么知识）**
3. 掌握对象、类的概念
4. 掌握prototype的用法
5. 掌握对象的实例化
6. 掌握各种继承方式
7. 学会封装组件
8. 掌握常用的设计模式
9. 学会ES5的新特性
10. 掌握简单框架的封装
11. **能力目标（产出物是什么）**
12. 让学生真正理解Javascript的处理机制和灵活运用，锻炼学生的思维能力同时掌握发生问题，处理问题的能力。
13. 让学生精通OOP编程
14. 掌握组件的封装
15. 掌握框架制作的思想和常用的设计模式
16. **参考教材**

**《JavaScript高级程序设计》，李松峰 曹力 译 人民邮电出版社**

**《JavaScript权威指南》 淘宝前段团队 译 机械工业出版社**

1. **课程内容安排**
2. **理解对象**
   1. 理解对象
      1. 创建对象
      2. 属性的查询和设置
      3. 删除属性
      4. 检测属性
      5. 枚举属性
      6. 属性getter和setter
      7. 序列化对象
      8. 对象方法
3. **创建对象**
   1. 创建对象
      1. 工厂模式
      2. 构造函数模式
      3. 原型模式
      4. 组合使用构造器和原型模式
      5. 动态原型模式
4. **继承**
   1. 原型链
   2. 借用构造函数
   3. 继承方式
      1. 组合继承
      2. 原型式继承
      3. 寄生式继承
      4. 寄生组合式继承
5. 组件封装
   1. Tab框的封装
   2. 无缝滚动图片组件封装
6. **ECMScript 5中的数组方法**
   1. 数组方法
      1. forEach方法
      2. map方法
      3. filter方法
      4. ervery ()和some()
      5. reduce和reduceRight

4.2 综合案例

1. 表单脚本
   1. 表单序列化
   2. 富文本编辑
      1. 使用contenteditable属性
      2. 操作富文本
      3. 富文本选区
      4. 表单与富文本
2. JSON
   1. 语法
      1. 简单值
      2. 对象
      3. 数组
   2. 解析与序列化
      1. JSON对象
      2. 序列化选项
      3. 解析选项
   3. 综合案例
3. **客户端检测**
   1. 能力检测
      1. 更可靠的能力检测
      2. 能力检测，不是浏览器检测事件处理之非IE浏览器
   2. 怪癖检测.
   3. 用户代理检测
4. **事件**
   1. 模拟事件
   2. 自定义事件
   3. 综合按钮
5. **Window对象**
   1. 浏览器的定位和导航
   2. 浏览历史
   3. 浏览器和屏幕信息
   4. 作为Window对象属性的文档元素
   5. 综合案例
6. **Ajax(一)**
   1. Ajax基础
      1. 什么是AJAX
   2. Ajax请求服务器数据的4个步骤
      1. 创建一个Ajax对象
      2. 连接服务器
      3. 发送请求，告诉服务器要什么
      4. 接收返回值
   3. 实例
      1. 验证用户名是否存在
7. **AJAX(二)**
   1. Ajax的原理
      1. GET方式
      2. GET缓存问题
      3. POST方式
   2. 编写Ajax库：
8. **脚本化CSS**
   1. 查询计算出的样式
   2. 脚本化CSS类
   3. 脚本化样式表
9. **ECMAScript5中的类**
   1. 让属性不可枚举
   2. 定义不可变的类
   3. 封装对象状态
   4. 防止类的扩展
10. 脚本化HTTP
    1. 使用script发送HTTP请求：JSONP
    2. 总结
11. 表格排序和移动
    1. 画出流程图
       1. 根据流程图将复用代码进行封装
    2. 表格排序
    3. 表格移动
12. **设计模式**
    1. 单体模式
    2. 享元模式
    3. 观察者模式
13. **简单框架实现（一）**
    1. 框架的基本实现
14. **简单框架实现（二）**
    1. 框架的代码优化
15. **最佳实践**
    1. 可维护性
       1. 什么是可维护的代码
       2. 松散耦合 强内聚 弱耦合
    2. 性能