**《js前端开发（下）》教学大纲**

**编写人：赵晓茹**

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