**Js**

Dom事件类

基本概念：dom事件的级别

DOM 0：element.onclick=function(){ }

DOM 2：element.addEventListener(‘click’,function(){ },false) 默认false是冒泡， true捕获

DOM 3：element. addEventListener (‘keyup’,function(){ },false)

DOM事件模型(捕获，冒泡)

DOM事件流

捕获-》目标阶段-》冒泡（上传win对象）

描述DOM事件捕获的具体流程（上-）下）

Window->document->html标签->body->….->目标元素

Event对象的常见应用

event.preventDefault()

event.stopPropagation() 阻止冒泡

event.stopImmediatePropagation() (事件响应优先级)

event.currentTarget

event.target

自定义事件

1.Event

Var eve = new Event(“custome”);

ev.addEeventListener(‘custome’,function(){ console.log(“mmm”)};);

ev.dispatchEvent(eve) 触发

2.CustomEvent

原型链

创建对象的几种方法

原型、构造函数、实例、原型链

Instanceof原理

New运算符

1. Var o1 = { name:”mm”}

Var 011 = new Object( { name:”011“})

Var M = function(){ this.name = “02”}

Var o2 = new M() 构造函数

Var P = { name:”03”}

Var 03 = Object.create（P）

HTTP协议类

HTTP协议特点

（无连接（不会保持连接，只连接一次就断掉），无状态（上次和这次的不会记住），简单快速（url固定的），灵活（数据类型的传输））

HTTP报文的组成部分

请求报文（请求行（HTTP方法+页面地址+HTTP协议+版本），请求头（key+value值来告诉服务端要哪些内容以及类型），空行（不再是请求头的部分了），请求体），响应报文（状态行，响应头，空行，响应体）

HTTP方法

Get 获取资源

Post 传输资源

Put 更新资源

Delete 删除资源

Head 获得报文首部

Post get区别

Get产生的URL地址可以被收藏，post不可以

Get在浏览器回退时无害的，post会再次提交请求✔️~

Get请求会被浏览器主动缓存，post不会，除非手动设置 ~

Get请求只能进行URL编码，post支持多种编码方式

Get请求参数会被完整的保留在浏览器历史记录里，post参数不会保留 ~

Get请求在URL中传送的参数是有长度限制的，post没有限制 ~

对参数的数据类型，get只接受ASCII字符，post没有限制

Get比post更不安全，因为参数直接暴露在URL上，所以不能传递敏感信息

Get参数通过URL传递，post放在request body中 ~

状态码

1xx 指示信息

2xx 成功

3xx 重定向-要完成请求必须进行更进一步的操作

4xx 客户端错误-请求有语法错误或请求无法实现

5xx 服务器错误-服务器未能实现合法的请求

200 OK ： 客户端请求成功

206 partial content 客户发送了一个带有range头的get请求，服务器完成

301 moved permanently:所请求的页面已经转移至新的url

302 found: 所请求的页面已经临时转移至新的url

304 not modified：客户端有缓冲的文档并发出了一个条件性的请求，服务器告诉客户，原来缓冲的文档还可以继续使用

400 bad request:客户端请求有语法错误，不能被服务器所理解

401 unauthorized：请求未经授权，

403 forbidden：禁止被请求页面的访问 ~

404 not found：请求资源不存在

500 internal server error:服务器发生不可预期的错误，原来缓冲的文档还可以继续使用

503 server unavailable 请求未完成，服务器临时过载或当机，过一段时间可能恢复正常

持久连接

Keep-Alive模式 HTTP1.1支持 1.0不支持；是客户端到服务器端的连接持续有效，出现对服务器的后续请求时，可避免建立或者重新建立连接；

HTTP协议采用“请求-应答”模式，使用时，每个请求/应答 客户端和服务器都要重新建立一个连接，完成之后立即断开连接（无连接的协议）

管线化

持久连接下完成的，把请求一次性打包传过去，响应也一次性打包发过来，仅HTTP1.1支持此技术；只有get和head请求可以进行管线化；初次连接时不应启动管线机制，因为对方服务器不一定支持HTTP/1.1版本的协议

面向对象：

类的声明

实例化

继承（原型链）

借助构造函数（部分继承，父亲的原型对象的方法不能继承）

借助原型链实现继承（子.prototype = new 父()；缺点：对个实例对象一起改变；因为俩个实例的原型链上的原型对象 是共用的）