

1. HTTP的状态码+浏览器处理缓存的机制

1xx: 信息性状态码 100, 101

2xx: 成功状态码 200: OK

3xx: 重定向状态码 301: 永久重定向, Location响应首部的值仍为当前URL, 因此为隐藏重定向; 302: 临时重定向, 显式重定向, Location响应首部的值为新的URL 304: Not Modified 未修改, 比如本地缓存的资源文件和服务器上比较时, 发现并没有修改, 服务器返回一个304状态码, 告诉浏览器, 你不用请求该资源, 直接使用本地的资源即可。

4xx: 客户端错误状态码 404: Not Found 请求的URL资源并不存在 5xx: 服务器端错误状态码 500: Internal Server Error 服务器内部错误 502: Bad Gateway 前面代理服务器联系不到后端的服务器时出现 504: Gateway Timeout 这个是代理能联系到后端的服务器, 但是后端的服务器在规定的时间内没有给代理服务器响应

浏览器处理缓存机制 见笔记

2. CSS盒子模型

从上图可以看到标准 w3c 盒子模型的范围包括 margin、border、padding、content, 并且 content 部分不包含其他部分。

ie 盒子模型(怪异盒模型)

从上图可以看到 ie 盒子模型的范围也包括 margin、border、padding、content, 和标准 w3c 盒子模型不同的是: ie 盒子模型的 content 部分包含了 border 和 padding。

3. 垂直居中 flex实现的版本 flex具体的使用 align-items:center

position: static | relative | absolute | fixed | center CSS3 | page CSS3 | sticky CSS3

默认值: static

适用于: 除 "display" 属性定义为 "table-column-group" | "table-column" 之外的所有元素

继承性: 无

动画性: 否

计算值: 指定值

媒体: 视觉

取值:

static: 对象遵循常规流, 此时4个定位偏移属性不会被应用。

relative: 对象遵循常规流, 并且参照自身在常规流中的位置通过 "top", "right", "bottom", "left" 这4个定位偏移属性进行偏移时不会影响常规流中的任何元素。

absolute: 对象脱离常规流, 此时偏移属性参照的是离自身最近的定位祖先元素, 如果没有定位的祖先元素, 则一直回溯到 body 元素。盒子的偏移位置不影响常规流中的任何元素, 其margin不与其他任何margin折叠。

fixed: 与 "absolute" 一致, 但偏移定位是以窗口为参考, 当出现滚动条时, 对象不会随着滚动。

center: 与 "absolute" 一致, 但偏移定位是以定位祖先元素的中心点为参考。盒子在其包含容器垂直水平居中。(CSS3)

page: 与 "absolute" 一致, 元素在分页媒体或者区域块内, 元素的包含块始终是初始包含块。否则取决于每个 "absolute" 模式。(CSS3)

sticky: 对象在常态时遵循常规流, 它就像是 "relative" 和 "fixed" 的合体, 当在屏幕中时按常规流排版, 当滚动到屏幕外时则表现如 "fixed", 该属性的表现是现实中你见到的吸附效果。(CSS3)

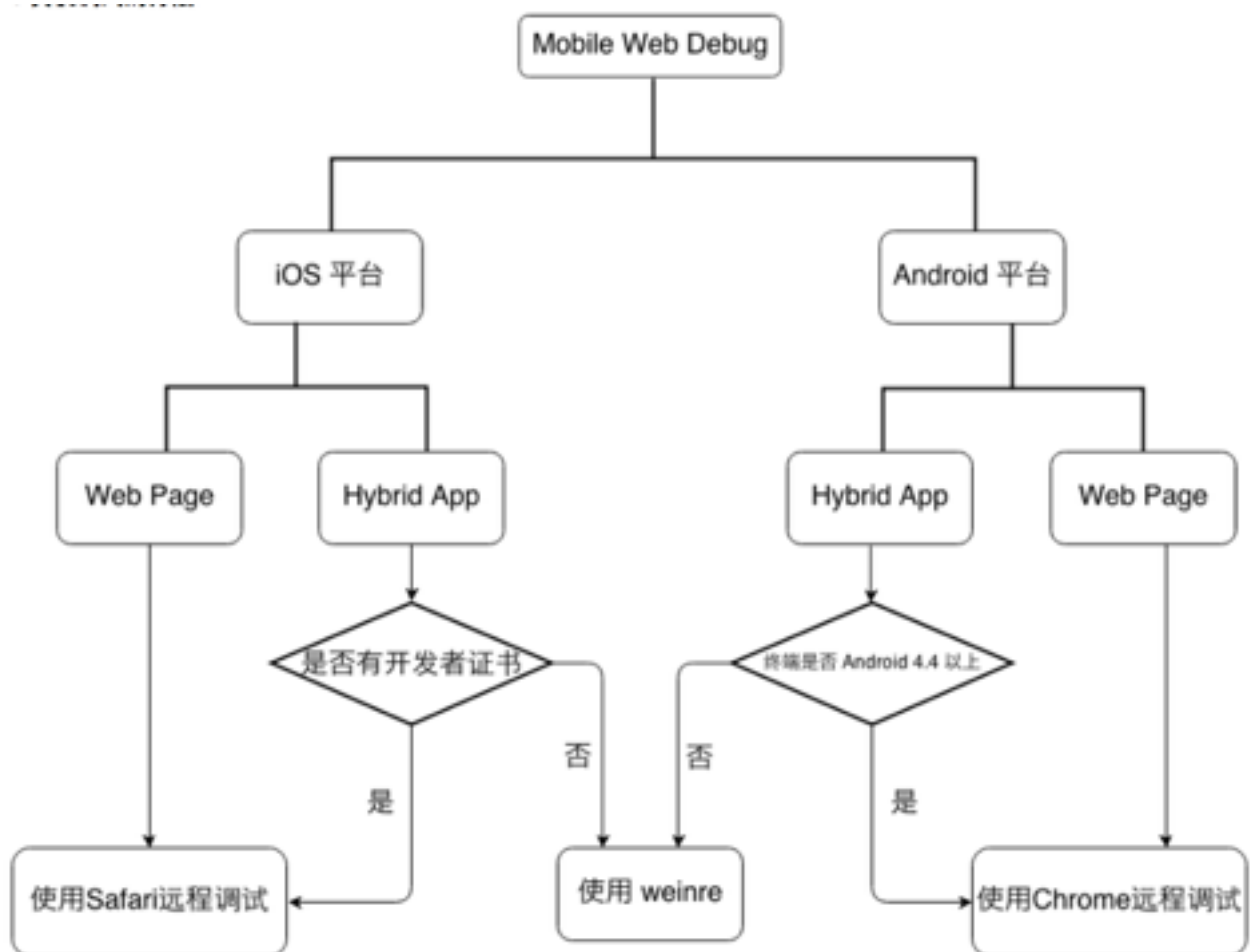
4. 定位: absolute fixed relative static

其中static:

5. 继承的几种方式: 原型继承, 构造函数继承, 两者混合的方式。缺点: 父类的构造函数执行两次, 如何避免: 使用寄生组合式继承

详见读书笔记 JS高级程序设计第6章的继承部分

6. 真机调式的方法



7. 数组ES5的一些方法: map filter every some

8. 一个动态的静态页面，一个大的字体库，实现字体文件的动态加载，有什么思路？

首先在浏览器端设置字体通过CSS的@font-face

<http://css.doyoe.com/rules/@font-face.htm>

至于要动态改变引用的字体库，思路：在本地的构建工具的基础上做一个插件，可以动态根据静态文件的内容来生成对应的字体库

在事件处理程序内部，对象 `this` 始终等于 `currentTarget` 的值，而 `target` 则只包含事件的实际目标。如果直接将事件处理程序指定给了目标元素，则 `this`、`currentTarget` 和 `target` 包含相同的值。来看下面的例子。

```
var btn = document.getElementById("myBtn");
btn.onclick = function(event){
    alert(event.currentTarget === this);    //true
    alert(event.target === this);          //true
};
```

`e.target`是当前触发click事件的真正目标

这个例子检测了 `currentTarget` 和 `target` 与 `this` 的值。由于 `click` 事件的目标是按钮，因此这三个值是相等的。如果事件处理程序存在于按钮的父节点中（例如 `document.body`），那么这些值是不相同的。再看下面的例子。

```
document.body.onclick = function(event){
    alert(event.currentTarget === document.body);    //true
    alert(this === document.body);                  //true
    alert(event.target === document.getElementById("myBtn"));    //true
};
```

9.事件冒泡，`e.target`和`e.currentTarget`的区别

```
var word='I can fly to people flyabs cannot ca';

function strSort(str) {
    var arr = str.toLowerCase().split(' ');

    arr.sort(function(a, b) {
        var len = Math.min(a.length, b.length);
        for (var i = 0; i < len; i++) {
            if (a[i] < b[i]) {
                return -1;
            } else if (a[i] > b[i]) {
                return 1;
            }
        }
        return a.length - b.length;
    });

    return arr.join(' ');
}

console.log(strSort(word));
```

10. 跨域技术：JSONP, Comet, Websocket, 前端监控Image

11. 三个题目：数组去重, InsertAfter, 对一个以空格分隔的字符串按字母表的顺序排列：

或者ES6：

12. 函数节流

13. hybrid中js怎么和native通信 jsBridge 底层的SDK

详见 标签：hybrid通信

<https://github.com/jianpx/tech-articles/blob/master/invoke-between-objective-c-and-javascript.md>

14. 实现一个动画，一个框的移动动画 CSS3, JS用setTimeout或者requestAnimationFrame, JQ的animate函数

使用jq实现：

使用CSS3：

15. 构建工具 fekit grunt webpack

<http://pinkyjie.com/2016/03/05/webpack-tips/>

16. 常用的设计模式，发布-订阅

<http://www.cnblogs.com/TomXu/archive/2012/03/02/2355128.html>

17. ES6是否了解

入门：<http://es6.ruanyifeng.com/>

