# **准备大纲**----------------------

面试题库：<http://web.jobbole.com/83808/> 下面有好多 ！！！

1.算法 ：排序 查找 数据结构

2.基础：H5,CSS,JS,JQuery, HTTP响应码,三次握手，四次分手

3.项目经验：

1.多源下载：html css js jquery ajax php mysql

难点：进度条，前台显示。

2.http反向代理：（回顾一下）

难点：

3.零食系统：html css js jquery（有支付 就好了！）

4.在线考试系统： 难点：ajax 算分，组卷。（在线考试、自主测试），科目。

5.智能作业： 和上面差不多（班级，学生，老师，分数）

6.人员管理：增删改查

7.纯前台

4.自己博客上的东西

5.面试题

## 1.算法准备--------------------------------

排序：插入、选择、冒泡、快排。

排序：内部排序和外部排序。

内部排序（8）：插入排序：直接插入排序 希尔排序

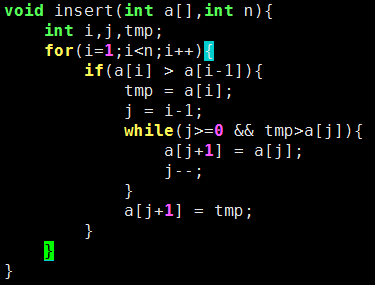
选择排序：简单选择排序 堆排序 O(nlog2n)

交换排序：冒泡排序 快速排序 O(nlog2n)

归并排序 O(nlog2n)

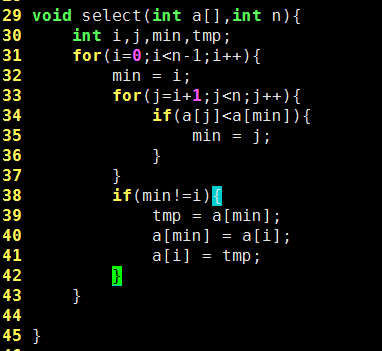
基数排序

直接插入排序（InsertSort）：稳定的，因为相等的还在原来顺序的后面。O（n^2）



简单选择排序（selectSort）：在要排序的一组数中，选出最小（或者最大）的一个数与第1个位置的数交换；然后在剩下的数当中再找最小（或者最大）的与第2个位置的数交换，依次类推，直到第n-1个元素（倒数第二个数）和第n个元素（最后一个数）比较为止。O（n^2）

//选择排序



void select\_sort(int\*a,int n)

{

register int i,j,min,t;

for(i=0;i<n-1;i++)

{

min=i;//查找最小值

for(j=i+1;j<n;j++)

if(a[min]>a[j])

min=j;//交换

if(min!=i)

{

t=a[min];

a[min]=a[i];

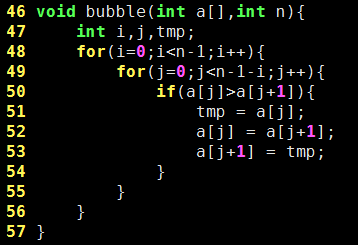
a[i]=t;

}

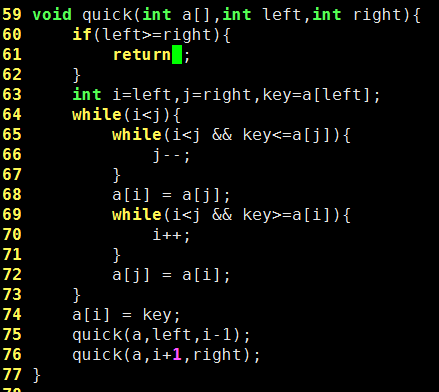
}

}

二元选择排序（good！）：改进为每趟循环确定两个元素（当前趟最大和最小记录）的位置,从而减少排序所需的循环次数。改进后对n个数据进行排序，最多只需进行[n/2]趟循环即可。 O（n^2）

冒泡排序（bubbleSort）：j 和j+1比较， 

快排（quickSort）：



void sort(int \*a, int left, int right)

{

if(left >= right)

{

return ;

}

int i = left;

int j = right;

int key = a[left];

while(i < j)

{

while(i < j && key <= a[j])

{

j--;/\*向前寻找\*/

}

a[i] = a[j];

while(i < j && key >= a[i])

{

i++;

}

a[j] = a[i];

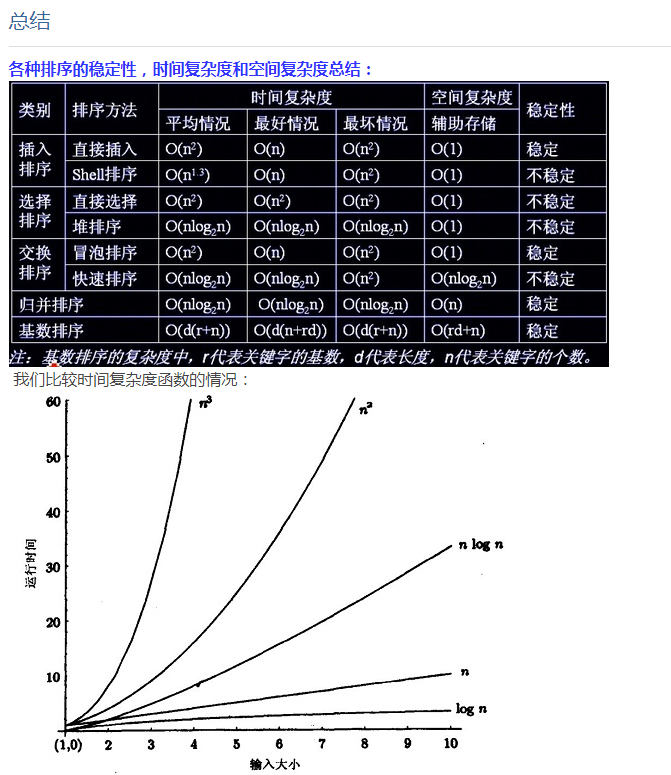
}

a[i] = key;

sort(a, left, i - 1);

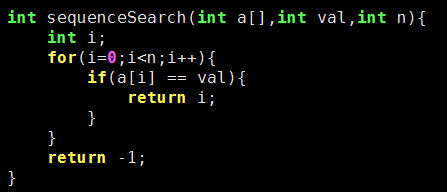
sort(a, i + 1, right);

}



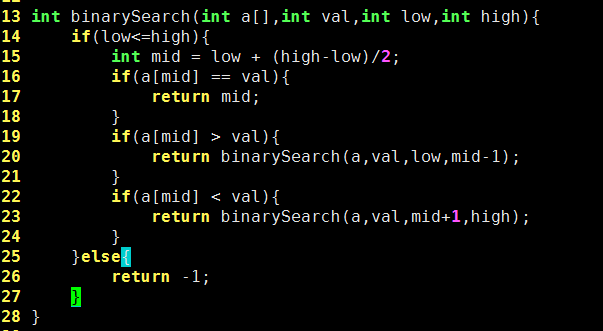
查找（7）：顺序查找，二分查找，插值查找，斐波那契查找，树表查找，分块查找，哈希查找

顺序查找：O（n）；

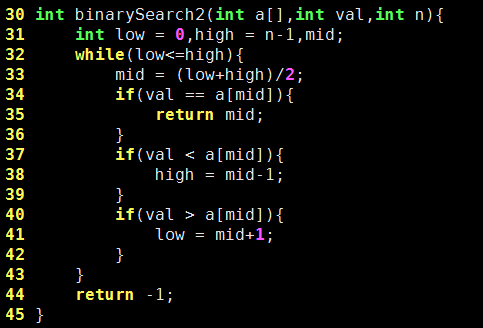


二分查找（折半查找）：数组必须是有序的。O(logn)

1.递归的



2.非递归的



## 2.面试题------------------------------------------------

作者：猪队  
链接：<https://www.nowcoder.com/discuss/59663?type=2&order=0&pos=34&page=1>  
来源：牛客网  
  
面试前笔试  
深复制  
数组去重  
String.prototype.replace的用法  
  
一面  
vue双向数据绑定  
前端性能优化  
如何测试首屏加载的时间（在文档开始处以及首屏末尾处添加一个时间戳，相减为首屏加载时间）  
css盒子模型  
es6有哪些新特性  
let和const的区别  
如何优化弹幕  
  
**腾讯**  
一面 技术面  
快排  
堆排序  
红黑树  
DNS查询过程  
负载均衡  
nginx 反向代理  
一致性哈希算法  
XSS  
CSRF  
DNS劫持  
RSA算法  
浏览器缓存  
如何让浏览器不缓存  
前端优化  
常见HTTP状态码有哪些，重点问了503遇到过吗（当我想说100那些时，面试官说不用那些，你只用说常见的就行了）  
es6有哪些新特性  
讲讲arraybuffer  
讲讲异步解决方案  
es2017的async/await实现原理  
常见的设计模式有哪些  
简单聊了下项目  
项目遇到的困难以及如何解决的  
有什么问题想问我吗  
  
  
二面 总监面（面试官比较严肃）  
讲讲项目用到哪些技术  
css有什么用  
你用过哪些css技术  
讲讲js有哪些技术  
js有哪些接口，有哪些函数  
你知道哪些排序算法  
来写个归并排序吧  
有没有什么问题想问我  
  
京东  
一面  
get和post的区别  
vue双向数据绑定（以及如何渲染的，虚拟dom）  
如何让一个不定宽高的元素水平垂直居中  
promise和setTimeout谁会优先执行  
跨域有哪些方法  
讲一讲CORS  
手写jsonp  
手写洗牌算法  
  
**微众**  
一面（面试官是架构师）  
angular和vue有什么区别  
vue,angular双向绑定  
mvvm，其中vm是什么，有什么作用  
项目（拿一个项目问，问的比较详细）  
同源策略  
跨域有哪些方法  
如何阻止别人的跨域访问  
  
二面（领导面）  
做过的项目（主要是问项目）  
以前的公司  
为什么不留以前的公司而想来微众  
为什么想做前端  
  
hr面  
自我介绍  
最近压力大吗  
有别的公司offer吗  
你认为没拿到别的公司的offer的原因是什么  
如果微众没有给到你offer怎么办  
最近令你最沮丧的一件事是什么  
令你最有成就感的一件事  
家庭情况  
学校发三方了吗

## 3.http响应码------------------------------------------

**http状态返回代码 1xx（临时响应）**  
表示临时响应并需要请求者继续执行操作的状态代码。

**http状态返回代码** 代码   说明  
100   （继续） 请求者应当继续提出请求。 服务器返回此代码表示已收到请求的第一部分，正在等待其余部分。   
101   （切换协议） 请求者已要求服务器切换协议，服务器已确认并准备切换。

**http状态返回代码 2xx （成功）**  
表示成功处理了请求的状态代码。

**http状态返回代码** 代码   说明  
200   （成功）  服务器已成功处理了请求。 通常，这表示服务器提供了请求的网页。  
201   （已创建）  请求成功并且服务器创建了新的资源。  
202   （已接受）  服务器已接受请求，但尚未处理。  
203   （非授权信息）  服务器已成功处理了请求，但返回的信息可能来自另一来源。  
204   （无内容）  服务器成功处理了请求，但没有返回任何内容。  
205   （重置内容） 服务器成功处理了请求，但没有返回任何内容。  
206   （部分内容）  服务器成功处理了部分 GET 请求。

**http状态返回代码 3xx （重定向）**  
表示要完成请求，需要进一步操作。 通常，这些状态代码用来重定向。

**http状态返回代码** 代码   说明  
300   （多种选择）  针对请求，服务器可执行多种操作。 服务器可根据请求者 (user agent) 选择一项操作，或提供操作列表供请求者选择。  
301   （永久移动）  请求的网页已永久移动到新位置。 服务器返回此响应（对 GET 或 HEAD 请求的响应）时，会自动将请求者转到新位置。  
302   （临时移动）  服务器目前从不同位置的网页响应请求，但请求者应继续使用原有位置来进行以后的请求。  
303   （查看其他位置） 请求者应当对不同的位置使用单独的 GET 请求来检索响应时，服务器返回此代码。

304   （未修改） 自从上次请求后，请求的网页未修改过。 服务器返回此响应时，不会返回网页内容。  
305   （使用代理） 请求者只能使用代理访问请求的网页。 如果服务器返回此响应，还表示请求者应使用代理。  
307   （临时重定向）  服务器目前从不同位置的网页响应请求，但请求者应继续使用原有位置来进行以后的请求。

**http状态返回代码 4xx（请求错误）**  
这些状态代码表示请求可能出错，妨碍了服务器的处理。

**//http状态返回代码 5xx（服务器错误）**  
这些状态代码表示服务器在尝试处理请求时发生内部错误。 这些错误可能是服务器本身的错误，而不是请求出错。

**http状态返回代码** 代码   说明  
500   （服务器内部错误）  服务器遇到错误，无法完成请求。  
501   （尚未实施） 服务器不具备完成请求的功能。 例如，服务器无法识别请求方法时可能会返回此代码。  
502   （错误网关） 服务器作为网关或代理，从上游服务器收到无效响应。  
503   （服务不可用） 服务器目前无法使用（由于超载或停机维护）。 通常，这只是暂时状态。  
504   （网关超时）  服务器作为网关或代理，但是没有及时从上游服务器收到请求。  
505   （HTTP 版本不受支持） 服务器不支持请求中所用的 HTTP 协议版本。

一些常见的**http状态返回代码**为：

**200 - 服务器成功返回网页  
404 - 请求的网页不存在  
503 - 服务不可用**

## 4.JS特效实现------

1.封装运动函数 startMove(obj,json,fn)实现链式运动，同时运动。

## 5.有赞电面--------

Q.let.const?const能修改对象里面的属性么?作用域都有哪些？

A：const对象的属性可以被修改。

块作用域，函数作用域

Q.js继承

A：一个对象的prototype = 父对象的示例。

Q.定位都有哪些？相对定位是相对哪里？

A相对定位，绝对定位，固定定位）。

ANSWER：absolute 生成绝对定位的元素，相对于 static 定位以外的第一个祖先元素进行定位。 relative 生成相对定位的元素，相对于其在普通流中的位置进行定位。

Q.cookie怎么实现的？

登陆的时候将登陆态 发给服务器，服务器用来与之前存的值进行比对。登陆、退出时都会向服务器发送，进行比较、删除cookie的操作。

ANSWER：Cookie实际上是一小段的文本信息。客户端请求服务器，如果服务器需要记录该用户状态，就使用response向客户端浏览器颁发一个Cookie。客户端浏览器会把Cookie保存起来。当浏览器再请求该网站时，浏览器把请求的网址连同该Cookie一同提交给服务器。服务器检查该Cookie，以此来辨认用户状态。服务器还可以根据需要修改Cookie的内容。

Cookie具有不可跨域性。

由于浏览器每次请求服务器都会携带Cookie，因此Cookie内容不宜过多，否则影响速度。Cookie的内容应该少而精。

Q.从页面输入url到展示页面发生了什么？

A:DNS解析IP，服务器返回响应，第一步会看浏览器中的缓存是否过期，没过期的话，就用浏览器的页面。否则服务器返回页面，浏览器加载DOM、渲染HTMl。从而展示页面。

ANSWER. 总体来说分为以下几个过程:

1. DNS解析
2. TCP连接
3. 发送HTTP请求
4. 服务器处理请求并返回HTTP报文
5. 浏览器解析渲染页面
6. 连接结束

Q．浏览器缓存的页面是什么？DOM结构？整个页面？<http://www.cnblogs.com/skynet/archive/2012/11/28/2792503.html>（讲的很好~~）

Cache-control：max-age|max-scale，expires

A：？？

Q：跨域？怎么解决？（<http://www.cnblogs.com/2050/p/3191744.html>）

A. 跨域，指的是浏览器不能执行其他网站的脚本。它是由浏览器的同源策略造成的，是浏览器对javascript施加的安全限制。

方法：1.jsonp, 要注意JSONP只支持GET请求，不支持POST请求。。（comet忘了）

**2、代理：**

例如www.123.com/index.html需要调用www.456.com/server.php，可以写一个接口www.123.com/server.php，由这个接口在后端去调用www.456.com/server.php并拿到返回值，然后再返回给index.html，这就是一个代理的模式。相当于绕过了浏览器端，自然就不存在跨域问题。

**3、PHP端修改header（XHR2方式）**

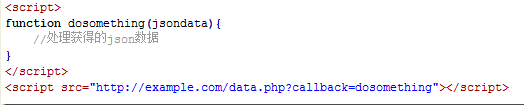
在php接口脚本中加入以下两句即可：

header('Access-Control-Allow-Origin:\*');//允许所有来源访问

header('Access-Control-Allow-Method:POST,GET');//允许访问的方式

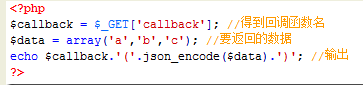
Q：jsonp怎么实现的？

A：在js中，我们直接用XMLHttpRequest请求不同域上的数据时，是不可以的。但是，在页面上引入不同域上的js脚本文件却是可以的，jsonp正是利用这个特性来实现的。

比如，有个a.html页面，它里面的代码需要利用ajax获取一个不同域上的json数据，假设这个json数据地址是<http://example.com/data.php>,那么a.html中的代码就可以这样：

我们看到获取数据的地址后面还有一个callback参数，按惯例是用这个参数名，但是你用其他的也一样。当然如果获取数据的jsonp地址页面不是你自己能控制的，就得按照提供数据的那一方的规定格式来操作了。

因为是当做一个js文件来引入的，所以<http://example.com/data.php>返回的必须是一个能执行的js文件，所以这个页面的php代码可能是这样的:

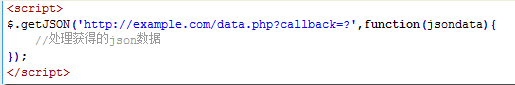


最终那个页面输出的结果是: 

所以通过<http://example.com/data.php?callback=dosomething>得到的js文件，就是我们之前定义的dosomething函数,并且它的参数就是我们需要的json数据，这样我们就跨域获得了我们需要的数据。

这样jsonp的原理就很清楚了，通过script标签引入一个js文件，这个js文件载入成功后会执行我们在url参数中指定的函数，并且会把我们需要的json数据作为参数传入。所以jsonp是需要服务器端的页面进行相应的配合的。

知道jsonp跨域的原理后我们就可以用js动态生成script标签来进行跨域操作了，而不用特意的手动的书写那些script标签。如果你的页面使用jquery，那么通过它封装的方法就能很方便的来进行jsonp操作了。



原理是一样的，只不过我们不需要手动的插入script标签以及定义回掉函数。jquery会自动生成一个全局函数来替换callback=?中的问号，之后获取到数据后又会自动销毁，实际上就是起一个临时代理函数的作用。$.getJSON方法会自动判断是否跨域，不跨域的话，就调用普通的ajax方法；跨域的话，则会以异步加载js文件的形式来调用jsonp的回调函数。

Q．请求数据，除了ajax还有什么办法？<https://www.zhihu.com/question/33561004/answer/56980806>

A．查数据库。从数据库中取值。

Q．接上，还有什么方法？

A．1. HTML赋值  
2. JS赋值  
3. script填充JSON

填充 JSON 数据到 `<script>` 标签中，前端通过 DOM 获取 JSON字符串并解析成对象。

<script type="text/template" id="data">{"username":"nimojs","userid":1}</script>

<script>

var data = JSON.parse($('#data').html());

//{username:"nimojs",userid:1}

</script>

4. AJAX获取JSON

使用 AJAX 获取JSON数据

<span id="showdata">查看资料</span>

<div style="display:none;" id="box">

<h2>用户信息</h2>

<p id="info"><img src="loading.gif" /></p>

</div>

$('#showdata').on('click',function(){

$('#box').show();

$.getJSON('/ajax/userdata/',function(oData){

// oData = {"username":"nimojs","userid":1}

$('#info').html('用户名：' + oData.username + '<br>用户ID：' + oData.userid);

})

})

5. WebSocket实时传输数据

(<http://www.runoob.com/html/html5-websocket.html> )

Q．实现异步的方法有哪些？JS引擎是怎么实现异步的？

<http://blog.csdn.net/sdfujichao/article/details/52160028>

http://www.cnblogs.com/sprying/archive/2013/05/26/3100639.html

A．1.回调(这是异步编程最基本的方法。)

2.事件监听。

三、发布/订阅

四、Promises对象

Q．布局都有哪些？flex知道么？弹性布局

Flex 布局，可以简便、完整、响应式地实现各种页面布局。目前，它已经得到了所有浏览器的支持，这意味着，现在就能很安全地使用这项功能。

<http://www.ruanyifeng.com/blog/2015/07/flex-grammar.html>

A．定位。 flex？？

Q.与缓存有关的HTTP头？

Cache-control，expires，last-modified,If-Modified-Since,ETag,If-None-Match

A．http://www.cnblogs.com/yjf512/p/3244882.html

## 6.拼多多前端研发工程师

(单选14道。不定项5道，大题3道)

Repaint, Reflow

1.触发reflow的方式？

2.HTTP的码什么意思？301，401,400，？

3.HTTP2.0的新功能？

1. 给定一个包含N个整数的数组A，找出所有不一样的组合ijk，使得A[i]+A[j]+A[k]=0, 比如var A = [-1,0,1,2,-1,4];。输出[-1,0,1],[-1,-1,2]

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Document</title>

<style>

#ipt1{

width: 0;

height: 0;

opacity: 0;

}

</style>

</head>

<body>

<input type="text" id="ipt">

<div id="ipt1">hello</div>

<script>

/\*

console.log(1 + '2' + '2'); //122

console.log(1 + +'2' + '2'); //32

console.log(1 -'1' + '2'); //02

console.log(+1 + '1' + '2'); //112

console.log('A' - 'B' + '2'); //NaN2

console.log('A' - 'B' + 2); //NaN

var a={};

b = {key:'b'};

c={

key:'c'

}

a[b] = 456;

console.log(a[b]); //456

\*/

/\*

var ipt = document.getElementById('ipt'); //mousedown.onfocus.mouseup.click顺序

ipt.onmousedown = function(){

console.log("mousedown");

};

ipt.onmouseup = function(){

console.log("onmouseup");

};

ipt.onclick = function(){

console.log("onclick");

};

ipt.onfocus = function(){

console.log("onfocus");

};

/\*

console.log(a2); //undefined

console.log(a3); //ERROR！！

var a2 = 5;

a3 = 6;

\*/

//？？？？？？？？？？？？？？？？？？？？？？

var i=1;

(function(){

var start = new Date().getTime();

var si = setInterval(function(){

var now = new Date().getTime();

if(now < (start+100)){

i++;

}else{

console.log(i);

clearInterval(si);

}

}, 10);

})();

var ipt = document.getElementById('ipt');

/\*

var s1 = new String('sonya');

var s2 = new String('sonya');

console.log(s1==s2); //false

console.log(s1===s2); //false

var a = document.createElement("p");

\*/

var A = [-1,0,1,2,-1,4];

var n = A.length;

for(var i=0;i<n;i++){

for(var j=1;j<n;j++){

for(var k=2;k<n;k++){

if(A[i]==A[j] || A[j]==A[k] || A[i]!=A[k]) {

continue;

}else if(A[i]+A[j]+A[k]==0){

console.log("["+A[i]+","+A[j]+","+A[k]+"]");

}

}

}

}

/\*for(var i=0;i<n;i++){

j=i+1;

if(a0+){

}

}\*/

/\*

function mul(){

var n = arguments.length;

var s = 1;

for(var i=0;i<n;i++){

s \*= arguments[i];

}

return s;

}

console.log(mul(2,3,4));

console.log(mul(4,3,4,2));

\*/

function mul(x) {

const result = (y) => mul(x \* y);

result.valueOf = () => x;

return result;

}

console.log(mul(2)(3)(4));

</script>

</body>

</html>

# \*[浏览器缓存机制](http://www.cnblogs.com/lovesong/p/5352973.html)

利用浏览器缓存，可以使得页面加载速度提高，也减轻服务端压力。有几个比较重要点如下：

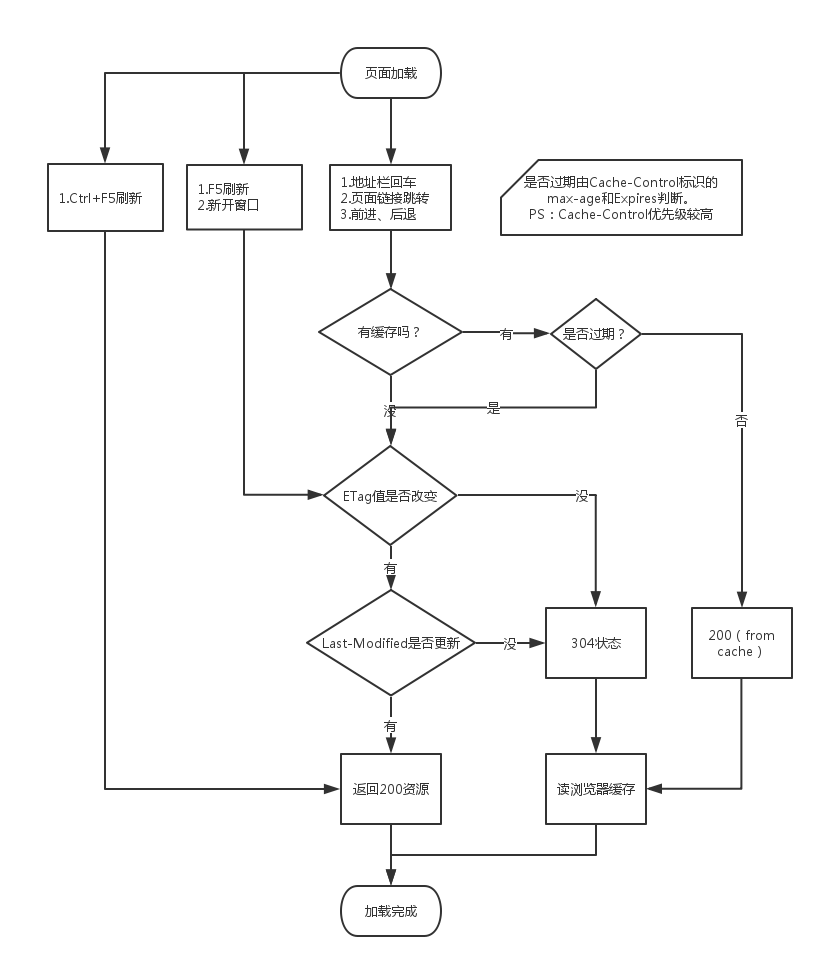
1. 浏览器是如何判断缓存是否过期？

2. 服务端如何判断缓存已失效？

3. 为什么有了Last-Modified还要Etag？

4. 200 OK( from cache )和 304 Not Modified的区别？

下面是我总结的页面利用浏览器缓存的过程（图示）：



**浏览器是如何判断缓存是否过期？**

应该是根据Response Header里面的Cache-Control和Expires这两个属性，当两个都存在时，Cache-Control优先级较高。

**Cache-Control**

该字段用于控制浏览器在什么情况下直接使用本地缓存而不向服务器发送请求。一般具有以下值：

Public：指示响应可被任何缓存区缓存。

Private：指示对于单个用户的整个或部分响应消息，不能被共享缓存处理。这允许服务器仅仅描述当用户的部分响应消息，此响应消息对于其他用户的请求无效。

no-cache：指示请求或响应消息不能缓存。

no-store：用于防止重要的信息被无意的发布。在请求消息中发送将使得请求和响应消息都不使用缓存。

max-age：指示浏览器可以接收生存期不大于指定时间（以秒为单位）的响应。

min-fresh：指示浏览器可以接收响应时间小于当前时间加上指定时间的响应。

max-stale：指示浏览器可以接收超出超时期间的响应消息。如果指定max-stale消息的值，那么浏览器可以接收超出超时期指定值之内的响应消息。

PS：其实主要关注max-age这个值就行了。

**Expires**

Expires 头部字段提供一个日期和时间，在该日期前的所有对该资源的请求都会直接使用浏览器缓存而不用向服务器请求。

例如：

Expires: Sun, 08 Nov 2009 03:37:26 GMT

注意：

1. cache-control max-age 和 max-stale将覆盖Expires header。

2. 使用Expires存在服务器端时间和浏览器时间不一致的问题。

3. 另外有人说Expires 是HTTP 1.0的东西，现在默认浏览器均默认使用HTTP 1.1，所以它的作用基本忽略（暂不明了）。

**服务端如何判断缓存已失效？**

服务端通过If-Modified-Since（Last-Modified）和If-None-Match（Etag）这两个属性的值来判断缓存是否失效的。

**Last-Modified/If-Modified-Since**

Last-Modified/If-Modified-Since要配合Cache-Control使用。

Last-Modified：响应资源的最后修改时间。

If-Modified-Since：当缓存过期时，发现资源具有Last-Modified声明，则在请求头带上If-Modified-Since（值就是Last-Modified）。服务器收到请求后发现有头If-Modified-Since则与被请求资源的最后修改时间进行比对。若最后修改时间较新，说明资源又被改动过，则响应HTTP 200整片资源内容（写在响应消息包体内）;若最后修改时间较旧，说明资源无新修改，则响应HTTP 304，告知浏览器继续使用所保存的cache。

**Etag/If-None-Match**

Etag/If-None-Match也要配合Cache-Control使用。

Etag：资源在服务器的唯一标识（生成规则由服务器决定）。Apache中，ETag的值，默认是对文件的索引节（INode），大小（Size）和最后修改时间（MTime）进行Hash后得到的。

If-None-Match：当缓存过期时，发现资源具有Etage声明，则在请求头带上If-None-Match（值就是Etag）。服务器收到请求后发现有头If-None-Match 则与被请求资源的相应校验串进行比对，决定返回200或304。

**为什么有了Last-Modified还要Etag？**

Etag的出现主要是为了解决几个Last-Modified比较难解决的问题：

1. Last-Modified标注的最后修改只能精确到秒级，如果某些文件在1秒钟以内，被修改多次的话，它将不能准确标注文件的修改时间。

2. 如果某些文件会被定期生成，当有时内容并没有任何变化，但Last-Modified却改变了，导致文件没法使用缓存。

3. 有可能存在服务器没有准确获取文件修改时间，或者与代理服务器时间不一致等情形。

**200 OK( from cache )和 304 Not Modified的区别？**

200 OK( from cache )不向服务器发送请求，直接使用本地缓存文件。304 Not Modified则向服务器询问，若服务器认为浏览器的缓存版本还可用，那么便会返回304。

**200 OK( from cache ) 出现操作：**

1.地址栏回车

2.页面链接跳转

3.前进、后退

**304 Not Modified 出现操作：**

1.F5刷新

2.新开窗口

PS：但是这样不是绝对，还是会有奇怪的情况，例如百度首页无论是怎么刷，都是200 from cache。

**总结**

在写完这篇文章时候，有些地方我还是不大清楚的，就比如为什么百度首页无论是怎么刷，都是200 from cache。200 from cache和304 Not Modified出现条件并没有什么定论。