

optimize

Long operation

如果都是主线程操作，时间长了之后浏览器会有假死状态（弹框提示，是否关闭），但是异步微拆分操作或者使用webworker的方式，可以使主线程继续能接收到用户的相应。

Layout && repaint

每次重排，必然会导致重绘。例如元素位置，尺寸改变都会发生重排。减少重排？

1：尽量不要在布局信息改变时做查询。

2：使用fragment（文档片段），只触发了一次重排，且只访问了一次实时的DOM。

3：使用绝对定位，脱离文档流，不会影响到其他元素。

3：.....

none

声明元素正常渲染，没有包含的应用。

strict

声明所有的包含规则应用于这个元素。这样写等价于 `contain: size layout style paint。`

content

声明这个元素上除了 `size` 的所有包含规则。等价于 `contain: layout style paint。`

size

声明这个元素可以被变换尺寸，不需要去检查它依赖尺寸变化。

layout

声明没有外部元素可以影响它内部的布局，反之亦然。









style

声明那些同时会影响这个元素和其子孙元素的属性，不会超出这个元素的包含范围。

paint

声明这个元素的子孙节点不会在它边缘外显示。如果一个元素在视窗外或其他原因不可见，则同样保证它的子孙节点不会被显示。

200 or 304

 mini-login-embedderV3.js?v=579228 login-openaccount.taobao.com/assets/js	<div>▼ General</div> <div>Request URL: https://static.wuage.com/common/statistic/footer.js</div> <div>Request Method: GET</div> <div>Status Code: 304</div> <div>Remote Address: 180.149.158.231:443</div> <div>Referrer Policy: no-referrer-when-downgrade</div> <div>▼ Response Headers</div> <div>accept-ranges: bytes</div> <div>age: 902</div> <div>cache-control: max-age=900</div> <div>content-type: application/javascript</div> <div>date: Tue, 19 Sep 2017 05:37:40 GMT</div> <div>eagleid: b4959b0815058003628045420e</div> <div>etag: "59b7adf0-29b"</div> <div>expires: Tue, 19 Sep 2017 05:16:38 GMT</div> <div>last-modified: Tue, 12 Sep 2017 09:50:40 GMT</div> <div>server: Tengine</div> <div>status: 304</div> <div>timing-allow-origin: *</div>
 footer.js static.wuage.com/common/statistic	
 font_292015_8sirprviwhbh85mi.woff at.alicdn.com/t	
 149680303738502024694_216_62.png img.wuage.com/public	
 149680313169691821030_48_64.gif img.wuage.com/public	
 getLoginInfo?jsonpCallback=getUserInfo0 login.wuage.com	
 plugin.php?id=security:job /	
 149724886239146187513_74_32.png	

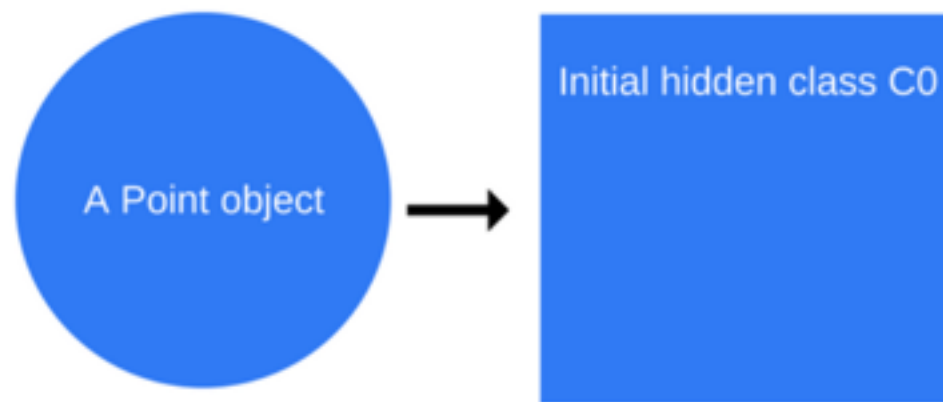
线上js和css的max-age只有15分钟,所以很容易进行freshness校验,此时就会出现304,浪费性能了,虽然比较微小。最好的办法是设置一个很长的时间,然后使用版本控制的方法。

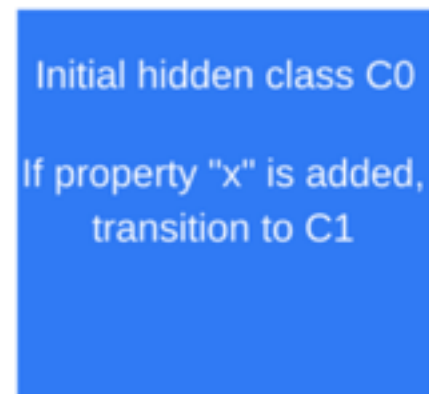
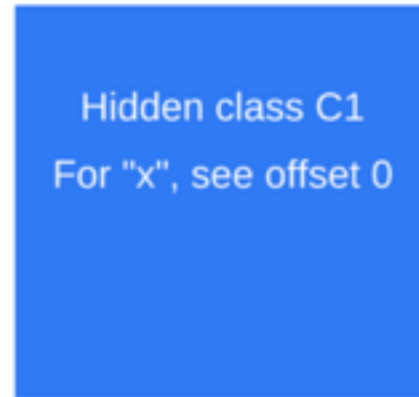
Hidden class

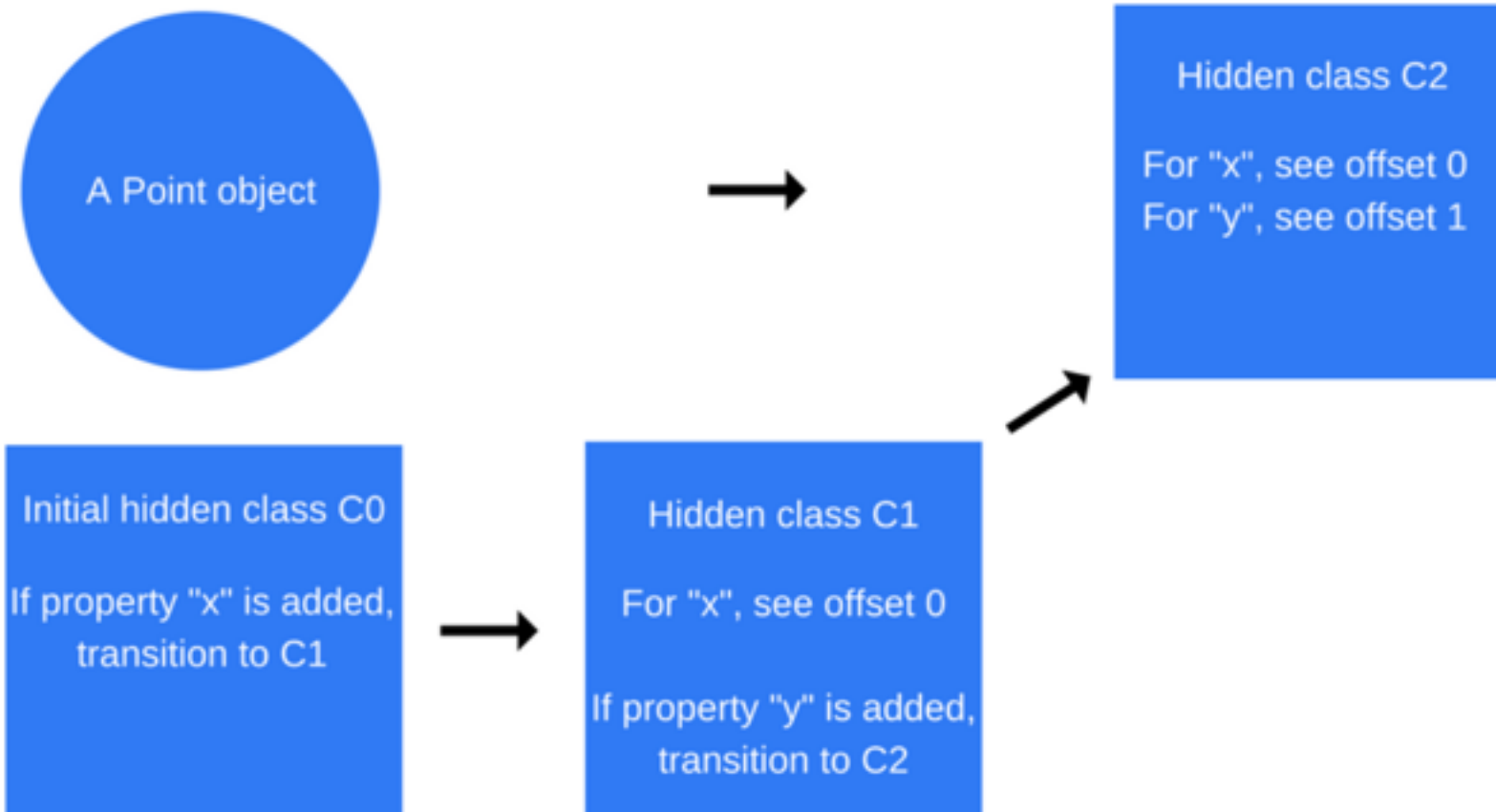
If two objects share a hidden class and the same property is added to both of them, transitions will ensure that both objects receive the same new hidden class and all the optimized code that comes with it.

<https://blog.sessionstack.com/how-javascript-works-inside-the-v8-engine-5-tips-on-how-to-write-optimized-code-ac089e62b12e>

```
function Point(x, y) {  
    this.x = x;  
    this.y = y;  
}  
var p1 = new Point(1, 2);
```







JIT


```
function add(a,b) {  
    return a+b;  
}  
for(var i=0;i<1000000;i++){  
    add(i,i);  
    add('hello','jsdt');  
}
```