转 前端开发工程师必须关注的几个性能指标

2017年10月29日 00:00:00 程序员的那些事_ 阅读数：202

文 | 张贤（网易前端资深工程师）

关于页面相应时间，有一条著名的**“2-5-8原则”**。当用户访问一个页面：

在2秒内得到响应时，会感觉系统响应很快；

在2-5秒之间得到响应时，会感觉系统的响应速度还可以；

在5-8秒以内得到响应时，会感觉系统的响应速度很慢，但可以接受；

而超过8秒后仍然无法得到响应时，用户会感觉系统糟透了，进而选择离开这个站点，或者发起第二次请求。

对于一个网站如果希望抓住用户，网站的速度以及稳定性是首当其冲的。

从各式各样的前端监控平台中，你都可以获得页面很多的性能指标。本文将介绍几个几个比较关键的指标，并给出相应的优化思路。

开始渲染时间

该时间点表示浏览器开始绘制页面，在此之前页面都是白屏，所以也称为白屏时间。

该时间点可用公式Time To Start Render = TTFB（Time To First Byte） + TTDD（Time To Document Download） + TTHE（Time To Head End）表示。其中TTFB表示浏览器发起请求到服务器返回第一个字节的时间，TTDD表示从服务器加载HTML文档的时间，TTHE表示文档头部解析完成所需要的时间。在高级浏览器中有对应的属性可以获取该时间点。Chrome可通过chrome.loadTimes().firstPaintTime获取，IE9+可以通过performance.timing.msFirstPaint获取，在不支持的浏览器中可以根据上面公式通过获取头部资源加载完的时刻模拟获取近似值。开始渲染时间越快，用户就能更快的看见页面。

对于该时间点的优化有：

1)优化服务器响应时间，服务器端尽早输出

2)减少html文件大小

3)减少头部资源,脚本尽量放在body中

DOM Ready

该时间点表示dom解析已经完成，资源还没有加载完成, 这个时候用户与页面的交互已经可用了。用公式TimeTo Dom Ready = TTSR(Time To Start Render) + TTDC(Time To Dom Created) + TTST(Time To Script)可以表示。TTSR上面已经介绍过了，TTDC表示DOM树创建所耗时间。TTST表示BODY中所有静态脚本加载和执行的时间。在高级浏览器中有DOMContentLoaded事件对应，MDN上有关DOMContentLoaded事件描述的文档如下，

The DOMContentLoaded event is fired when the document has been completely loaded and parsed, without waiting for stylesheets, images, and subframes to finish loading (the load event can be used to detect a fully-loaded page).

详细规范可以查看W3C的HTML5规范。从MDN文档上可以看出该事件主要是指dom文档加载解析完成，看上去很简单，但是DOMContentLoaded事件的触发与css,js息息相关，现在有专门的名词Critical Rendering Path(关键呈现路径)来描述，在文章【关键呈现路径】中详细介绍了关键呈现路径对DOMContentLoaded的影响。

在不支持DOMContentLoaded事件的浏览器中可以通过模拟获取近似值，主要的模拟方法有：

1）低版本webkit内核浏览器可以通过轮询document.readyState来实现

2）ie中可通过setTimeout不断调用documentElement的doScroll方法，直到其可用来实现

具体实现方法可以参考主流框架(jquery等)的实现。 DOM Ready时间点意味着用户与页面可以进行交互了，因此越早越好，对于该时间点的优化有：

1）减少dom结构的复杂度，节点尽可能少，嵌套不要太深

2）优化关键呈现路径

首屏时间

该时间点表示用户看到第一屏页面的时间，这个时间点很重要但是很难获取，一般都只能通过模拟获取一个近似时间。一般模拟方法有：

1）不断获取屏幕截图，当截图不再变化时，可以视为首屏时间。可参考webPagetest的Speed Index算法；

2）一般影响首屏的主要因素是图片的加载，通过页面加载完后判断图片是否在首屏内，找出加载最慢的一张即可视为首屏时间。当然还需考虑其他细节，具体可参考【7天打造前端性能监控系统】

针对该时间点的优化有：

1）页面首屏的显示尽量不要依赖于js代码，js尽量放到domReady后执行或加载

2）首屏外的图片延迟加载

3）首屏结构尽量简单，首屏外的css可延迟加载

onload

该时间点是window.onload事件触发的时间，表示原始文档和所有引用的内容已经加载完成，用户最明显的感觉就是浏览器tab上loading状态结束。

该时间点的优化方式有：

1）减少资源的请求数和文件大小

2）将非初始化脚本放到onLoad之后执行

3）无需同步的脚本异步加载

为了优化整站性能，页面onload的时候可以考虑做一些预加载，把其它页面需要用到的资源预先加载进来。