(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 105095280 B
(45)授权公告日 2020.02.14

(21)申请号 201410201879.6

(22)申请日 2014.05.13

(65)同一申请的已公布的文献号
申请公布号 CN 105095280 A

(43)申请公布日 2015.11.25

(73)专利权人 腾讯科技(深圳)有限公司
地址 518044 广东省深圳市福田区振兴路赛格科技园2栋东403室

(72)发明人 李骏

(74)专利代理机构 北京派特恩知识产权代理有限公司 11270
代理人 张颖玲 张振伟

(51)Int.Cl.
G06F 16/957(2019.01)

(56)对比文件
CN 101997927 A,2011.03.30,说明书第5-30段.
CN 100580672 C,2010.01.13,全文.
CN 1971559 A,2007.05.30,全文.
DE 10224102 A1,2003.12.24,全文.
CN 102436373 A,2012.05.02,全文.

审查员 朱琦

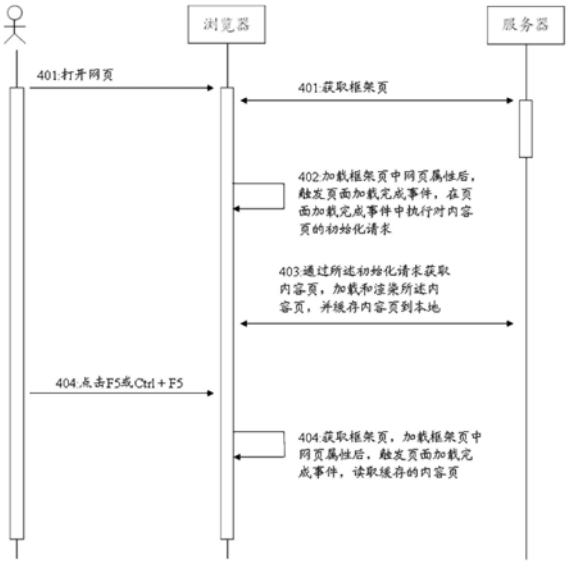
权利要求书3页 说明书8页 附图6页

(54)发明名称

一种浏览器缓存方法和装置

(57)摘要

本发明公开了一种浏览器缓存方法和装置，浏览器在首次打开网页时，获取第一页面，加载第一页面中网页属性后，触发页面加载完成事件，在页面加载完成事件中执行对第二页面的初始化请求；通过所述初始化请求获取第二页面，并缓存第二页面到本地；再次打开所述网页时，获取第一页面，加载第一页面中网页属性后，触发页面加载完成事件，读取缓存的第二页面。



CN 105095280 B

1.一种浏览器缓存方法,其特征在于,该方法包括:

预先将网页拆分成第一页面和第二页面分别进行配置,所述第一页面包括网页属性、页面加载完成事件;

浏览器在打开网页时,在本地未查找到网页的缓存内容时,确定首次打开网页,向服务器发送第一页面请求获取第一页面,加载第一页面中的网页属性后,触发页面加载完成事件,在页面加载完成事件中执行对第二页面的初始化请求;所述第一页面为网页的框架页,所述第二页面为网页的内容页;

通过所述初始化请求获取第二页面,加载和渲染第二页面,并全部缓存第二页面到本地;

再次打开所述网页时,获取第一页面,加载第一页面中网页属性后,触发页面加载完成事件,基于服务器返回的网页的版本号读取缓存的第二页面。

2.根据权利要求1所述的浏览器缓存方法,其特征在于,所述初始化请求包括页面初始化对象请求和静态资源请求;

所述第二页面包括网页的超文本标记语言(HTML)对象、和/或层叠样式表(CSS)对象、和/或JS(javascript)对象、和/或图片。

3.根据权利要求1所述的浏览器缓存方法,其特征在于,所述在页面加载完成事件中执行对第二页面的初始化请求包括:在页面加载完成事件中加入设置延迟时间(setTimeout)事件,在触发页面加载完成事件后,执行设置延迟时间事件;在延迟时间内,在本地查询是否有缓存的第二页面,没有缓存的第二页面时,在延迟时间到后向服务器发送对第二页面的初始化请求。

4.根据权利要求3所述的浏览器缓存方法,其特征在于,所述获取第一页面包括:所述浏览器接收服务器返回的第一页面和网页的版本号;

所述缓存第二页面到本地包括:所述浏览器在获取到第二页面后,将网页的版本号与所述第二页面一起缓存到本地;

所述基于服务器返回的网页的版本号读取缓存的第二页面包括:浏览器在执行设置延迟时间事件过程中,在本地查询是否有缓存的第二页面,有缓存的第二页面时,比较服务器返回的网页的版本号与缓存中的版本号是否一致,在一致时,读取缓存的第二页面。

5.根据权利要求4所述的浏览器缓存方法,其特征在于,该方法还包括:在服务器返回的网页的版本号与缓存中的版本号不一致时,不读取缓存的第二页面,在延迟时间到后向服务器发送对第二页面的初始化请求,之后缓存新的第二页面覆盖本地原有的第二页面。

6.一种浏览器缓存方法,其特征在于,该方法包括:

服务器在收到浏览器的第一页面请求时,向浏览器发送第一页面,在收到浏览器的对第二页面的初始化请求后,向浏览器发送第二页面,以使浏览器加载和渲染第二页面,并全部缓存第二页面;

其中,服务器是在浏览器打开网页时,在本地未查找到网页的缓存内容时,确定首次打开网页时发送的第一页面,所述第一页面包括网页属性、页面加载完成事件;所述第一页面为网页的框架页,所述第二页面为网页的内容页,框架页和内容页分开进行配置。

7.根据权利要求6所述的浏览器缓存方法,其特征在于,所述初始化请求包括页面初始化对象请求和静态资源请求;

所述第二页面包括网页的HTML对象、和/或CSS对象、和/或JS对象、和/或图片。

8.一种终端，其特征在于，所述终端包括：浏览器和存储器；其中，

浏览器，用于预先将网页拆分成第一页面和第二页面分别进行配置，所述第一页面包括网页属性、页面加载完成事件；在打开网页时，在本地未查找到网页的缓存内容时，确定首次打开网页，向服务器发送第一页面请求获取第一页面，加载第一页面中的网页属性后，触发页面加载完成事件，在页面加载完成事件中执行对第二页面的初始化请求；所述第一页面为网页的框架页，所述第二页面为网页的内容页；通过所述初始化请求获取第二页面，加载和渲染第二页面，并全部缓存第二页面到存储器；再次打开所述网页时，获取第一页面，加载第一页面中网页属性后，触发页面加载完成事件，基于服务器返回的网页的版本号读取存储器中缓存的第二页面；

存储器，用于缓存浏览器获取的第二页面。

9.根据权利要求8所述的终端，其特征在于，所述浏览器，还用于在触发页面加载完成事件后，执行设置延迟时间事件；在延迟时间内，在存储器中查询是否有缓存的第二页面，没有缓存的第二页面时，在延迟时间到后向服务器发送对第二页面的初始化请求。

10.根据权利要求9所述的终端，其特征在于，所述浏览器，具体用于接收服务器返回的第一页面和网页的版本号，在获取到第二页面后，将所述版本号与对应的第二页面一起缓存到存储器，再次打开所述网页时，在执行设置延迟时间事件过程中，在存储器查询是否有缓存的第二页面，有缓存的第二页面时，比较新接收的网页的版本号和存储器缓存的版本号是否一致，在一致时，读取缓存的第二页面。

11.根据权利要求10所述的终端，其特征在于，所述浏览器，还用于在新接收的网页的版本号和存储器缓存的版本号不一致时，不读取缓存的第二页面，在延迟时间到后向服务器发送对第二页面的初始化请求，之后缓存新的第二页面覆盖存储器中原有的第二页面。

12.一种服务器，其特征在于，所述服务器包括：第一页面发送模块、第二页面发送模块；其中，

第一页面发送模块，用于在收到浏览器的第一页面请求时，向浏览器发送第一页面；

第二页面发送模块，用于在收到浏览器的对第二页面的初始化请求后，向浏览器发送第二页面，以使浏览器加载和渲染第二页面，并全部缓存第二页面；

其中，第一页面发送模块是在浏览器打开网页时，在本地未查找到网页的缓存内容时，确定首次打开网页时发送的第一页面，所述第一页面包括网页属性、页面加载完成事件；所述第一页面为网页的框架页，所述第二页面为网页的内容页，框架页和内容页分开进行配置。

13.一种浏览器缓存系统，其特征在于，所述系统包括上述权利要求8至11任一项所述的终端和权利要求12所述的服务器。

14.一种终端，其特征在于，包括：

存储器，用于存储可执行指令；

处理器，用于执行所述可执行指令时，实现如权利要求1至5任一项所述的浏览器缓存方法。

15.一种服务器，其特征在于，包括：

存储器，用于存储可执行指令；

处理器，用于执行所述可执行指令时，实现如权利要求6或7所述的浏览器缓存方法。

16.一种存储介质，其特征在于，存储有可执行指令，所述可执行指令被执行时，用于实现如权利要求1至5任一项所述的浏览器缓存方法，或者如权利要求6或7所述的浏览器缓存方法。

一种浏览器缓存方法和装置

技术领域

[0001] 本发明涉及互联网缓存技术，尤其涉及一种浏览器缓存方法和装置。

背景技术

[0002] 在高并发、大流量的系统中，常常需要对前端页面进行缓存，目前针对网页(web)侧前端缓存大致有闪存(flash)、本地存储(localstorage)等解决方案，在实现过程中，目前解决方案存在实现繁琐、有存储上限和浏览器支持的问题。

发明内容

[0003] 为解决现有存在的技术问题，本发明实施例主要提供一种浏览器缓存方法和装置。

[0004] 本发明实施例的技术方案是这样实现的：

[0005] 本发明实施例提供的一种浏览器缓存方法，该方法包括：

[0006] 浏览器在首次打开网页时，获取第一页面，加载第一页面中的网页属性后，触发页面加载完成事件，在页面加载完成事件中执行对第二页面的初始化请求；

[0007] 通过所述初始化请求获取第二页面，并缓存第二页面到本地；

[0008] 再次打开所述网页时，获取第一页面，加载第一页面中网页属性后，触发页面加载完成事件，读取缓存的第二页面。

[0009] 本发明实施例还提供一种浏览器缓存方法，该方法包括：

[0010] 服务器在收到浏览器的第一页面请求时，向浏览器发送第一页面，在收到浏览器的对第二页面的初始化请求后，向浏览器发送第二页面。

[0011] 本发明实施例还提供一种终端，所述终端包括：浏览器和存储器；其中，

[0012] 浏览器，用于在首次打开网页时，获取第一页面，加载第一页面中的网页属性后，触发页面加载完成事件，在页面加载完成事件中执行对第二页面的初始化请求；通过所述初始化请求获取第二页面，并缓存第二页面到存储器；再次打开所述网页时，获取第一页面，加载第一页面中网页属性后，触发页面加载完成事件，读取存储器中缓存的第二页面；

[0013] 存储器，用于缓存浏览器获取的第二页面。

[0014] 本发明实施例还提供一种服务器，所述服务器包括：第一页面发送模块、第二页面发送模块；其中，

[0015] 第一页面发送模块，用于在收到浏览器的第一页面请求时，向浏览器发送第一页面；

[0016] 第一页面发送模块，用于在收到浏览器的对第二页面的初始化请求后，向浏览器发送第二页面。

[0017] 本发明实施例还提供一种浏览器缓存系统，所述系统包括上述的终端和服务器。

[0018] 本发明提供了一种浏览器缓存方法和装置，浏览器在首次打开网页时，获取第一页面，加载第一页面中的网页属性后，触发页面加载完成事件，在页面加载完成事件中执行

对第二页面的初始化请求;通过所述初始化请求获取第二页面,并缓存第二页面到本地;再次打开所述网页时,获取第一页面,加载第一页面中网页属性后,触发页面加载完成事件,读取缓存的第二页面;如此,使用页面分开的缓存方式对原业务代码无影响,能够支持大部分主流浏览器,实现简单,并能有效减少刷新过程的请求和网络流量,在高并发、大流量场景有绝对优势,另外,由于第二页面可以全部进行缓存,这样就可以全页面缓存HTML对象、CSS对象、JS对象、图片,没有缓存上限。

附图说明

[0019] 图1为现有技术中F5刷新页面的开销检测示意图;

[0020] 图2为现有技术中Chrome浏览器或safari浏览器的点击刷新(F5或Ctrl+F5)过程与onload事件执行过程关系示意图;

[0021] 图3为现有技术中IE浏览器的点击刷新(F5或Ctrl+F5)过程与onload事件执行过程关系示意图;

[0022] 图4为本发明实施例提供的终端侧实现浏览器缓存方法的流程示意图;

[0023] 图5为本发明实施例加载框架页中网页属性后的网页显示示意图;

[0024] 图6为本发明实施例加载和渲染内容页后的网页显示示意图;

[0025] 图7为本发明实施例提供的终端的结构示意图;

[0026] 图8为本发明实施例提供的服务器的结构示意图;

[0027] 图9为本发明实施例提供的浏览器缓存系统的结构示意图。

具体实施方式

[0028] 目前大部分浏览器一般都会将打开的网页缓存到本地的缓存目录中,当下一次打开网页时,在本地的缓存目录中读取缓存的所述网页。一般情况下,浏览器确定读取缓存跟3个字段有关:

[0029] Last-Modified:服务器上文件最后修改时间;

[0030] Expires:本地缓存目录中,文件过期时间;

[0031] Cache-control:指定请求和响应遵循的缓存机制,一般由服务器指定过期

[0032] 时间间隔,浏览器根据时间间隔生成具体失效时间。

[0033] 浏览器在请求服务器资源时,如果缓存目录中的文件的Expires时间和Cache-control值有效,那么浏览器不会发出HTTP请求,而是直接读取本地缓存目录中的本地文件。

[0034] 浏览器请求服务器资源可以分为以下3种情况:

[0035] 1)直接通过点击链接或在地址栏输入url访问;

[0036] 在这种情况下,浏览器通过Expires和Cache-control判断是否读取缓存内容。

[0037] 2)点击F5刷新页面;

[0038] 在这种情况下,浏览器忽略Expires和Cache-control,在向服务器发送的HTTP资源请求的头文件中加入if-modified-since参数,值为上一次请求Last-Modified时间,服务器判断if-moified-since参数和服务器上网页最后修改时间是否匹配,如果匹配,则返回304消息,浏览器收到304消息后直接读取缓存的网页,如果不匹配,则返回网页内容,浏

览器接收网页内容。

[0039] 3)点击Ctrl+F5刷新页面；

[0040] 浏览器忽略Expires、Cachec-control、Last-modified时间，在向服务器发送的HTTP资源请求的头文件中加上Pragma:no-cache或Pragma:no-cache，Cache-Control:max-age＝0，直接请求服务器返回网页内容。

[0041] 对比上述3种情况下，浏览器读取缓存的开销如下：

[0042] 1)直接访问链接：不发出HTTP资源请求，直接读取缓存，开销最小；

[0043] 2)F5刷新页面：如图1所示，发出约500Bytes字节流量，接收150～200Bytes的字节流量，在有cookie的情况下，HTTP资源请求会带上cookie，HTTP资源请求会比图1所示的没有cookie的情况下更大，但F5刷新页面时，总体来说开销较小，不过，创建HTTP资源请求的过程会影响页面渲染速度。

[0044] 3)Ctrl+F5刷新页面：网页的所有资源全部请求服务器，开销最大。

[0045] 以商品详情网页中某商品的测试数据为例：

[0046] 1)直接访问：浏览器接收114.5kb；

[0047] 2)F5刷新：浏览器接收171.4kb；

[0048] 3)Ctrl+F5刷新：浏览器接收1550.9kb。

[0049] 这组数据是在开启gzip加速的情况下的对比，由此可见，如果页面能做到无论F5刷新还是Ctrl+F5刷新都直接访问缓存文件，将大大减少后台服务和内容分发网络(CDN，Content Delivery Network)流量压力，在一些高并发场合(如淘宝双11，CDN流量达400G/s)将能大大提高网站性能。

[0050] 经测试，不同浏览器的点击刷新(F5或Ctrl+F5)过程和onload事件执行过程有以下两种情况：

[0051] Chrome浏览器或safari浏览器中，如图2所示，流程如下：

[0052] 步骤201：用户点击F5或Ctrl+F5；

[0053] 步骤202：浏览器收到用户点击F5或Ctrl+F5消息，打开刷新状态；

[0054] 步骤203：浏览器向服务器发送HTTP资源请求；

[0055] 步骤204：服务器返回304消息或网页资源；

[0056] 步骤205：浏览器加载网页，并渲染页面；

[0057] 本步骤中，如果服务器返回的是304消息，则浏览器在本地缓存目录中读取网页资源进行加载，并渲染页面；如果服务器返回的是网页资源，则浏览器直接根据网页资源加载网页，并渲染页面。

[0058] 步骤206：浏览器关闭刷新状态；

[0059] 步骤207：浏览器触发onload事件。

[0060] IE系列浏览器中，如图3所示，流程如下：

[0061] 步骤301：用户点击F5或Ctrl+F5；

[0062] 步骤302：浏览器收到用户点击F5或Ctrl+F5消息，打开刷新状态；

[0063] 步骤303：浏览器向服务器发送HTTP资源请求；

[0064] 步骤304：服务器返回304消息或网页资源；

[0065] 步骤305：浏览器加载网页，并渲染页面；

[0066] 本步骤中，如果服务器返回的是304消息，则浏览器在本地缓存目录中读取网页资源进行加载，并渲染页面；如果服务器返回的是网页资源，则浏览器直接根据网页资源加载网页，并渲染页面。

[0067] 步骤306：浏览器触发onload事件；

[0068] 步骤307：浏览器关闭刷新状态。

[0069] 可以看出，IE系列浏览器中，浏览器触发onload事件是在浏览器关闭刷新状态之前，Chrome浏览器或safari浏览器中，浏览器触发onload事件是在浏览器关闭刷新状态之后，两种情况中，在用户刷新过程中，浏览器均需要向服务器发送HTTP资源请求，增加了流量开销，并不利于高并发场合下的网站刷新性能。

[0070] 本发明实例中，浏览器在首次打开网页时，获取第一页面，加载第一页面中的网页属性后，触发页面加载完成事件，在页面加载完成事件中执行对第二页面的初始化请求；通过所述初始化请求获取第二页面，并缓存第二页面到本地；在再次打开所述网页并触发页面加载完成事件后，直接读取缓存的第二页面。这里，所述第一页面可以称为框架页，包括网页属性、页面加载完成事件，其中，所述网页属性包括：层叠样式表(CSS，Cascading Style Sheet)标签、JS(javascript)标签等，其中，CSS标签一般是CSS外链请求，JS标签一般是JS外链请求；所述页面加载完成事件一般为onload事件；所述初始化请求包括页面初始化对象请求和静态资源请求；所述第二页面可以称为内容页，包括网页的HTML对象、和/或CSS对象、和/或JS对象、和/或图片等网页内容，其中，所述HTML对象可以是HTML文档，所述CSS对象可以是CSS文档，所述JS对象可以是JS文档。

[0071] 另外，所述第一页面与所述第二页面需要部署到同一域名下，所述第二页面的获取可以通过ajax get方式拉取。

[0072] 下面通过附图及具体实施例对本发明做进一步的详细说明。

[0073] 本发明实施例实现一种浏览器缓存方法，在终端侧，如图4所示，该方法主要包括以下几个步骤：

[0074] 步骤401：浏览器在首次打开网页时，获取框架页；

[0075] 这里，所述框架页包括：网页属性、页面加载完成事件；浏览器在打开网页时，在本地查找是否有所述网页的缓存内容，如链接地址、图标等，在没有所述网页的缓存内容时，确定所述网页为首次打开，之后向服务器发送框架页请求获取框架页，服务器在返回框架页的同时，还可以返回网页的版本号，浏览器可以根据版本号确定是否可以读取缓存，如：在返回的版本号与缓存中的版本号一致时，可以读取缓存内容，在返回的版本号与缓存中的版本号不一致时，不读取缓存内容。

[0076] 本实施例需要预先在配置网页时，将网页拆分成框架页和内容页分别进行配置，框架页的大小要尽可能的小，即框架页仅包括不需要缓存的内容和页面加载完成事件，这样，在重新打开该网页时，请求的框架页的开销将很小；而内容页都是可以缓存的内容，如：HTML对象、CSS对象、JS对象、图片等内容。

[0077] 步骤402：浏览器加载框架页中网页属性后，触发页面加载完成事件，在页面加载完成事件中执行对内容页的初始化请求；

[0078] 其中，所述加载框架页中网页属性可以包括：读取CSS标签和/或JS标签，在CSS标签为CSS外链请求时，根据所述CSS外链请求获取CSS外链内容，在JS标签为JS外链请求时，

根据JS外链请求获取JS外链内容，如图5所示，加载网页属性后的网页会显示出CSS外链内容和/或JS外链内容。

[0079] 本步骤中，所述在页面加载完成事件中执行对内容页的初始化请求具体是：预先在页面加载完成事件中加入设置延迟时间(setTimeout)事件，设置的延迟时间可以根据浏览器读取缓存的时间确定，一般可以设置成1ms或2ms，在触发页面加载完成事件后，先执行设置延迟时间事件，在延迟时间内，先在本地查询是否有缓存的内容页，如果没有，则在延迟时间到后向服务器发送对内容页的初始化请求；如果有，则直接读取缓存的内容页，在延迟时间到后也不发送对内容页的初始化请求。这里，由于是首次打开网页，本地一定是没有缓存的内容页的，所以，在延迟时间到后向服务器发送对内容页的初始化请求。

[0080] 其中，所述对内容页的初始化请求包括内容页初始化对象请求和静态资源请求。

[0081] 步骤403：浏览器通过所述初始化请求获取内容页，加载和渲染所述内容页，并缓存内容页到本地；

[0082] 其中，所述加载和渲染所述内容页包括：将内容页包括的网页的HTML对象、和/或CSS对象、和/或JS对象、和/或图片等网页内容按照网页的版式排布，并根据所述网页内容的参数进行渲染，所述参数包括：颜色、字体大小等；其中，所述HTML对象可以是HTML文档，所述CSS对象可以是CSS文档，所述JS对象可以是JS文档；如图6所示，加载和渲染内容页后浏览器所打开的网页将显示内容页的所有内容，包括HTML对象、和/或CSS对象、和/或JS对象、和/或图片等。

[0083] 其中，浏览器还可以将网页的版本号一起与对应的内容页缓存到本地，这里的本地一般是指本地缓存目录。

[0084] 步骤404：在通过点击F5或Ctrl+F5刷新所述网页时，浏览器获取框架页，加载框架页中网页属性后，触发页面加载完成事件，读取缓存的内容页；

[0085] 这里，由于步骤403中已经将内容页缓存到本地，浏览器在触发页面加载完成事件之后，在执行设置延迟时间事件过程中，由于触发页面加载完成事件标志着刷新完成，浏览器会读取本地的缓存内容，能够读取到缓存的内容页。

[0086] 在上述实施例中，使用框架页和内容页分开的缓存方式，对网页中原业务代码无影响，能够支持大部分主流浏览器，实现简单，并且在点击F5或Ctrl+F5刷新所述网页时，不再一定需要向服务器发送HTTP资源请求和接收服务器返回的响应消息，能有效减少刷新过程的请求和网络流量，在高并发、大流量场景有绝对优势，另外，由于内容页可以全部进行缓存，这样就可以全页面缓存HTML对象、CSS对象、JS对象、图片，并没有缓存上限。

[0087] 在另一个是实例中，浏览器可以根据服务器返回的网页的版本号与缓存中的版本号不一致确定更新缓存的内容页。具体的，浏览器可以在执行设置延迟时间事件过程中，比较服务器返回的网页的版本号与缓存中的版本号是否一致，在不一致时，不读取缓存的内容页，而是在延迟时间到后向服务器发送对内容页的初始化请求，通过对内容页的初始化请求获取内容页，并加载和渲染所述内容页，之后缓存新的内容页覆盖本地原有的内容页。

[0088] 那么，如果通过对内容页的初始化请求获取的内容页出现错误，缓存的内容页也将是个错误的页面，那么岂不会在每次重新打开网页时都会读取一个错误的内容页?为了解决这个问题，本发明实施例的浏览器还可以在获取到内容页时，在内容页的cookie中设置初始值(inited)为假(false)，而在完全加载并缓存内容页后置inited为真(ture)，这

样，浏览器在读取本地缓存的内容页时，可以先读取inited，如果为ture，则能够读取缓存的内容页，如果为false，则不能读取缓存的内容页，需要在延迟时间到后向服务器发送对内容页的初始化请求，通过对内容页的初始化请求获取内容页。

[0089] 本发明又一实施例实现一种浏览器缓存方法，在服务器侧，该方法主要包括：服务器在收到浏览器的第一页面请求时，向浏览器发送第一页面，在收到浏览器的对第二页面的初始化请求后，向浏览器发送第二页面。这里，所述第一页面可以称为框架页，包括网页属性、页面加载完成事件，其中，所述网页属性包括：CSS标签、JS标签等，其中，CSS标签一般是CSS外链请求，JS标签一般是JS外链请求；所述初始化请求包括页面初始化对象请求和静态资源请求；所述第二页面可以称为内容页，包括网页的HTML对象、和/或CSS对象、和/或JS对象、和/或图片等网页内容，其中，所述HTML对象可以是HTML文档，所述CSS对象可以是CSS文档，所述JS对象可以是JS文档。

[0090] 为了实现上述方法，本发明还提供一种终端，如图7所示，该终端包括：浏览器71和存储器72；其中，浏览器71用于在首次打开网页时，获取第一页面，加载第一页面中网页属性后，触发页面加载完成事件，在页面加载完成事件中执行对第二页面的初始化请求；通过所述初始化请求获取第二页面，并缓存第二页面到存储器72；再次打开所述网页时，获取第一页面，加载第一页面中网页属性后，触发页面加载完成事件，读取存储器72中缓存的第二页面；存储器72用于缓存浏览器71获取的第二页面。

[0091] 这里，所述第一页面可以称为框架页，包括网页属性、页面加载完成事件，其中，所述网页属性包括：CSS标签、JS标签等，所述页面加载完成事件一般为onload事件；所述初始化请求包括页面初始化对象请求和静态资源请求；所述第二页面可以称为内容页，包括网页的HTML对象、CSS对象、JS对象、图片等网页内容。

[0092] 所述浏览器71还用于在触发页面加载完成事件后，执行设置延迟时间事件，在延迟时间内，在存储器72中查询是否有缓存的第二页面，在没有缓存的第二页面时，在延迟时间到后向服务器发送对第二页面的初始化请求。

[0093] 在一个实施例中，所述浏览器71具体用于接收服务器返回的第一页面和网页的版本号，在获取到第二页面后，将所述版本号与对应的第二页面一起缓存到存储器72，再次打开所述网页时，在存储器查询是否有缓存的第二页面，有缓存的第二页面时，比较新接收的网页的版本号和存储器72缓存的版本号是否一致，在一致时，读取缓存的第二页面，在不一致时，不读取缓存的第二页面，在延迟时间到后向服务器发送对第二页面的初始化请求，之后缓存新的第二页面覆盖存储器中原有的第二页面。

[0094] 另外，浏览器71还需要在获取到第二页面时，在第二页面的cookie中设置inited为假false，而在完全加载并缓存第二页面后设置inited为ture，这样，浏览器71在读取存储器72缓存的第二页面时，可以先读取inited，如果为ture，则能够读取缓存的第二页面，如果为false，则不能读取缓存的第二页面，需要在延迟时间到后向服务器发送对第二页面的初始化请求，通过所述初始化请求获取第二页面。

[0095] 本发明还提供一种服务器，如图8所示，该服务器包括：第一页面发送模块81、第二页面发送模块82；其中，

[0096] 第一页面发送模块81，可以由服务器的存储器和发送接口实现，在在收到浏览器的第一页面请求时，向浏览器发送第一页面；

[0097] 第一页面发送模块82，也可以由服务器的存储器和发送接口实现，在收到浏览器的对第二页面的初始化请求后，向浏览器发送第二页面。

[0098] 基于上述终端，本发明还实现一种浏览器缓存系统，如图9所示，该系统包括：终端91、服务器92；其中，终端91用于在首次打开网页时，通过服务器92获取第一页面，加载第一页面中网页属性后，触发页面加载完成事件，在页面加载完成事件中执行对第二页面的初始化请求；通过所述初始化请求从服务器92获取第二页面，并缓存第二页面到本地；再次打开所述网页时，获取第一页面，加载第一页面中网页属性后，触发页面加载完成事件，读取缓存的第二页面；服务器92在终端91首次打开网页时，向终端91发送第一页面，在收到终端91的对第二页面的初始化请求后，向终端92发送第二页面。

[0099] 其中，所述终端92如图5所示，包括：浏览器71和存储器72；其中，浏览器71用于在首次打开网页时，获取第一页面，加载第一页面中网页属性后，触发页面加载完成事件，在页面加载完成事件中执行对第二页面的初始化请求；通过所述初始化请求获取第二页面，并缓存第二页面到存储器72；再次打开所述网页时，获取第一页面，加载第一页面中网页属性后，触发页面加载完成事件，读取存储器72中缓存的第二页面；存储器72用于缓存浏览器71获取的第二页面。

[0100] 这里，所述第一页面可以称为框架页，包括网页属性、页面加载完成事件，其中，所述网页属性包括：CSS标签、JS标签等，所述页面加载完成事件一般为onload事件；所述第二页面可以称为内容页，包括网页的HTML对象、CSS对象、JS对象、图片等网页内容。

[0101] 所述浏览器71还用于在触发页面加载完成事件后，执行设置延迟时间事件，在延迟时间内，在存储器72中查询是否有缓存的第二页面，在没有缓存的第二页面时，在延迟时间到后向服务器发送对第二页面的初始化请求。

[0102] 在一个实施例中，浏览器71具体用于接收服务器返回的第一页面和网页的版本号，在获取到第二页面后，将所述版本号与对应的第二页面一起缓存到存储器72，再次打开所述网页时，在存储器查询是否有缓存的第二页面，有缓存的第二页面时，比较新接收的网页的版本号和存储器72缓存的版本号是否一致，在一致时，读取缓存的第二页面，在不一致时，不读取缓存的第二页面，在延迟时间到后向服务器发送对第二页面的初始化请求，之后缓存新的第二页面覆盖存储器中原有的第二页面。

[0103] 另外，浏览器71还需要在获取到第二页面时，在第二页面的cookie中设置inited为假false，而在完全加载并缓存第二页面后设置inited为ture，这样，浏览器71在读取存储器72缓存的第二页面时，可以先读取inited，如果为ture，则能够读取缓存的第二页面，如果为false，则不能读取缓存的第二页面，需要在延迟时间到后向服务器发送对第二页面的初始化请求，通过所述初始化请求获取第二页面。

[0104] 本发明图4所示实施例所述浏览器缓存方法如果以软件功能模块的形式实现并作为独立的产品销售或使用时，也可以存储在一个计算机可读取存储介质中。基于这样的理解，本领域内的技术人员应明白，本申请的实施例可提供为方法、系统、或计算机程序产品。因此，本申请可采用完全硬件实施例、完全软件实施例、或结合软件和硬件方面的实施例的形式。而且，本申请可采用在一个或多个其中包含有计算机可用程序代码的计算机可用存储介质上实施的计算机程序产品的形式，所述存储介质包括但不限于U盘、移动硬盘、只读存储器(ROM，Read-Only Memory)、磁盘存储器、CD-ROM、光学存储器等。

[0105] 本申请是根据本申请实施例的方法、设备(系统)、和计算机程序产品的流程图和/或方框图来描述的。应理解可由计算机程序指令实现流程图和/或方框图中的每一流程和/或方框、以及流程图和/或方框图中的流程和/或方框的结合。可提供这些计算机程序指令到通用计算机、专用计算机、嵌入式处理机或其他可编程数据处理设备的处理器以产生一个机器，使得通过计算机或其他可编程数据处理设备的处理器执行的指令产生用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的装置。

[0106] 这些计算机程序指令也可存储在能引导计算机或其他可编程数据处理设备以特定方式工作的计算机可读存储器中，使得存储在该计算机可读存储器中的指令产生包括指令装置的制造品，该指令装置实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能。

[0107] 相应的，本发明实施例还提供一种计算机存储介质，其中存储有计算机程序，该计算机程序用于执行本发明图4所示实施例所述浏览器缓存方法。

[0108] 综上所述，使用页面分开的缓存方式，对网页中原业务代码无影响，能够支持大部分主流浏览器，实现简单，并且在点击F5或Ctrl+F5刷新所述网页时，不再一定需要向服务器发送HTTP资源请求和接收服务器返回的响应消息，能有效减少刷新过程的请求和网络流量，在高并发、大流量场景，如抢购网站，有绝对优势，另外，由于第二页面可以全部进行缓存，这样就可以全页面缓存HTML对象、CSS对象、JS对象、图片，并没有缓存上限。

[0109] 以上所述，仅为本发明的较佳实施例而已，并非用于限定本发明的保护范围，凡在本发明的精神和原则之内所作的任何修改、等同替换和改进等，均应包含在本发明的保护范围之内。

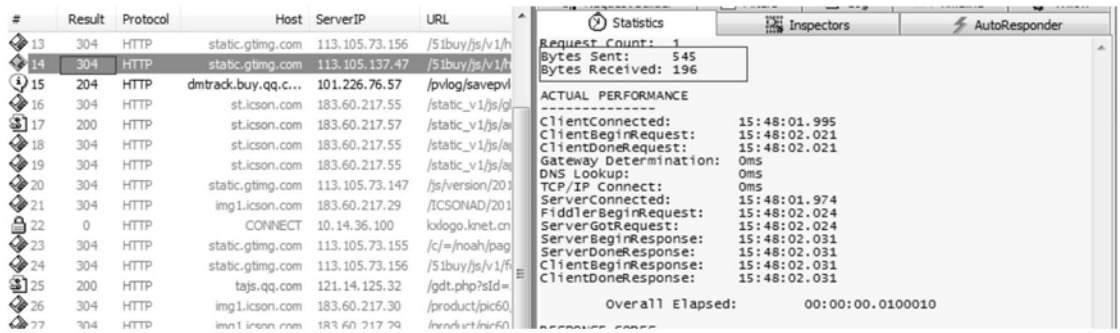

Result Protocol Host ServerIP URL
304 HTTP static.gtimg.com 113.105.73.156
304 HTTP static.gtimg.com 113.105.137.47
204 HTTP dmtrack.buy.qq.c... 101.226.76.57 /pvlog/savepvl
304 HTTP st.icson.com 183.60.217.55
200 HTTP st.icson.com 183.60.217.57
304 HTTP st.icson.com 183.60.217.55
304 HTTP st.icson.com 183.60.217.55
304 HTTP static.gtimg.com 113.105.73.147
304 HTTP img1.icson.com 183.60.217.29
0 HTTP CONNECT 10.14.36.100
304 HTTP static.gtimg.com 113.105.73.155
304 HTTP static.gtimg.com 113.105.73.156
200 HTTP tajs.qq.com 121.14.125.32
304 HTTP img1.icson.com 183.60.217.30
Statistics
Inspectors
AutoResponder
Request Count: 1
Bytes Sent: 545
Bytes Received: 196
ACTUAL PERFORMANCE
ClientConnected: 15:48:01.995
ClientBeginRequest: 15:48:02.021
ClientDoneRequest: 15:48:02.021
Gateway Determination: 0ms
DNS Lookup: 0ms
TCP/IP Connect: 0ms
ServerConnected: 15:48:01.974
FiddlerBeginRequest: 15:48:02.024
ServerGotRequest: 15:48:02.024
ServerBeginResponse: 15:48:02.031
ServerDoneResponse: 15:48:02.031
ClientBeginResponse: 15:48:02.031
ClientDoneResponse: 15:48:02.031
Overall Elapsed: 00:00:00.0100010

图1

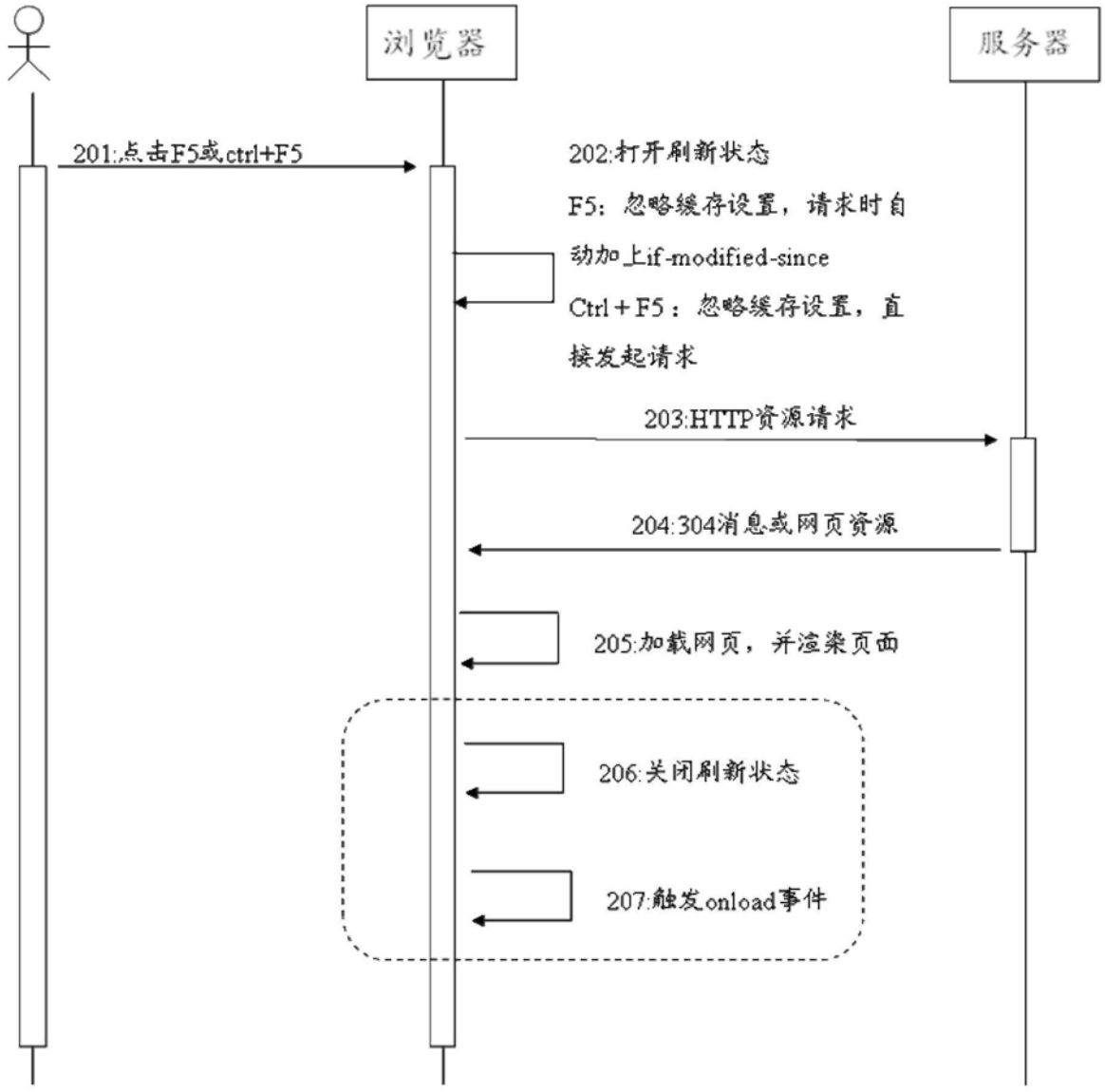

浏览器
服务器
201:点击F5或ctrl+F5
202:打开刷新状态
F5：忽略缓存设置，请求时自动加上if-modified-since
Ctrl＋F5：忽略缓存设置，直接发起请求
203:HTTP资源请求
204:304消息或网页资源
205:加载网页，并渲染页面
206:关闭刷新状态
207:触发onload事件

图2

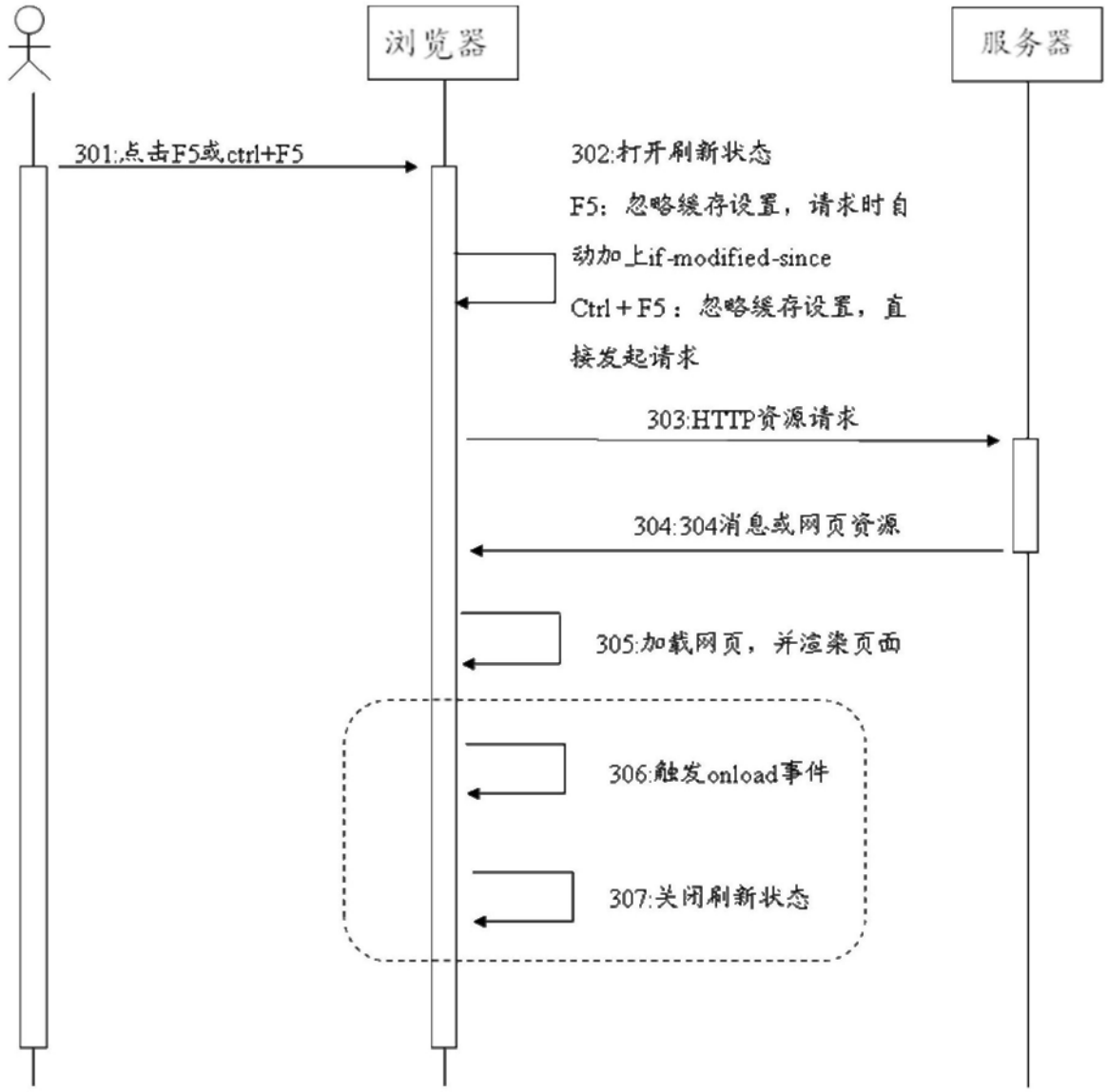

浏览器
服务器
301:点击F5或ctrl+F5
302:打开刷新状态
F5：忽略缓存设置，请求时自动加上if-modified-since
Ctrl＋F5：忽略缓存设置，直接发起请求
303:HTTP资源请求
304:304消息或网页资源
305:加载网页，并渲染页面
306:触发onload事件
307:关闭刷新状态

图3

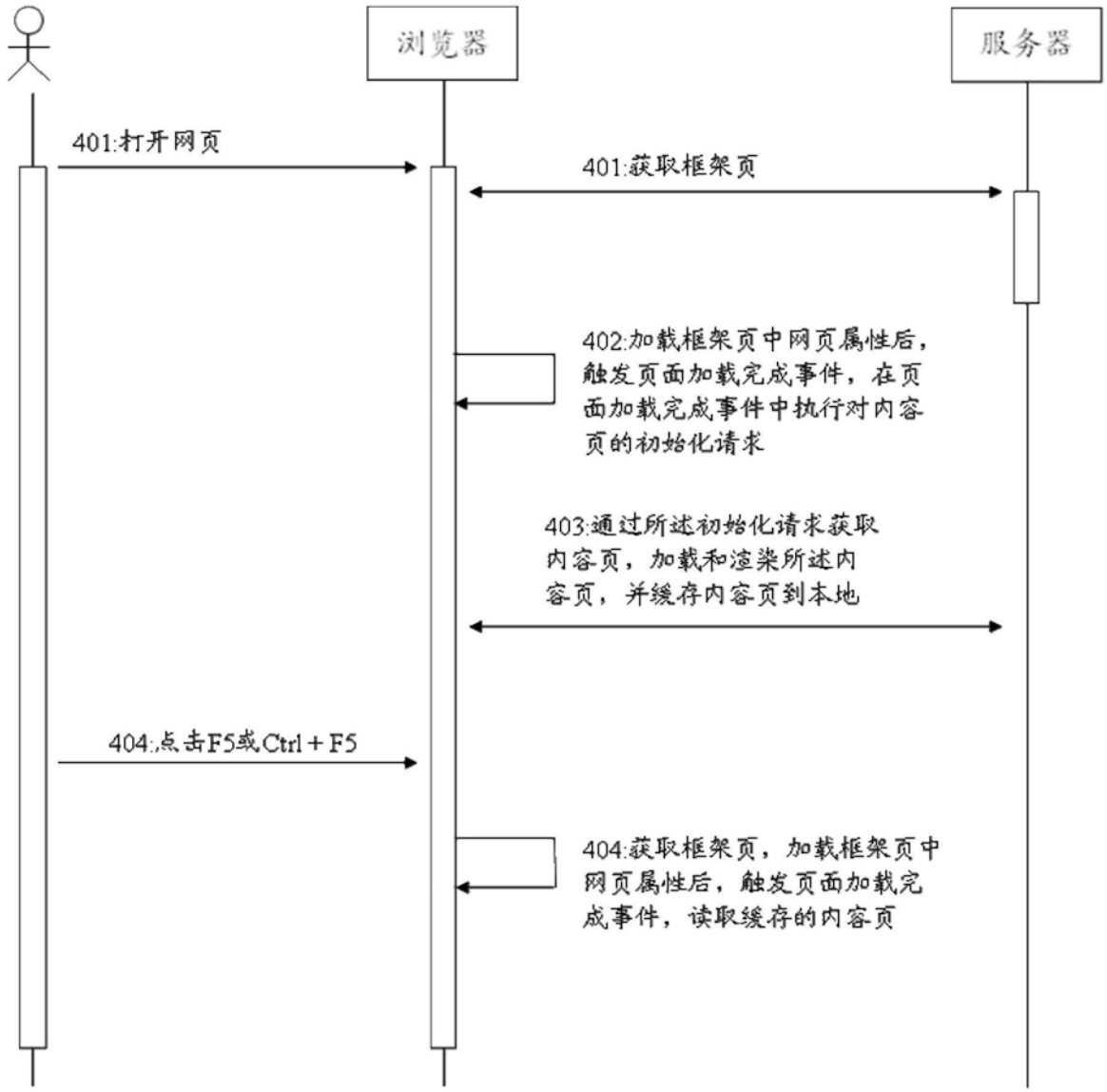

浏览器
服务器
401:打开网页
401:获取框架页
402:加载框架页中网页属性后，触发页面加载完成事件，在页面加载完成事件中执行对内容页的初始化请求
403:通过所述初始化请求获取内容页，加载和渲染所述内容页，并缓存内容页到本地
404:,点击F5或Ctrl＋F5
404:获取框架页，加载框架页中网页属性后，触发页面加载完成事件，读取缓存的内容页

图4

图5

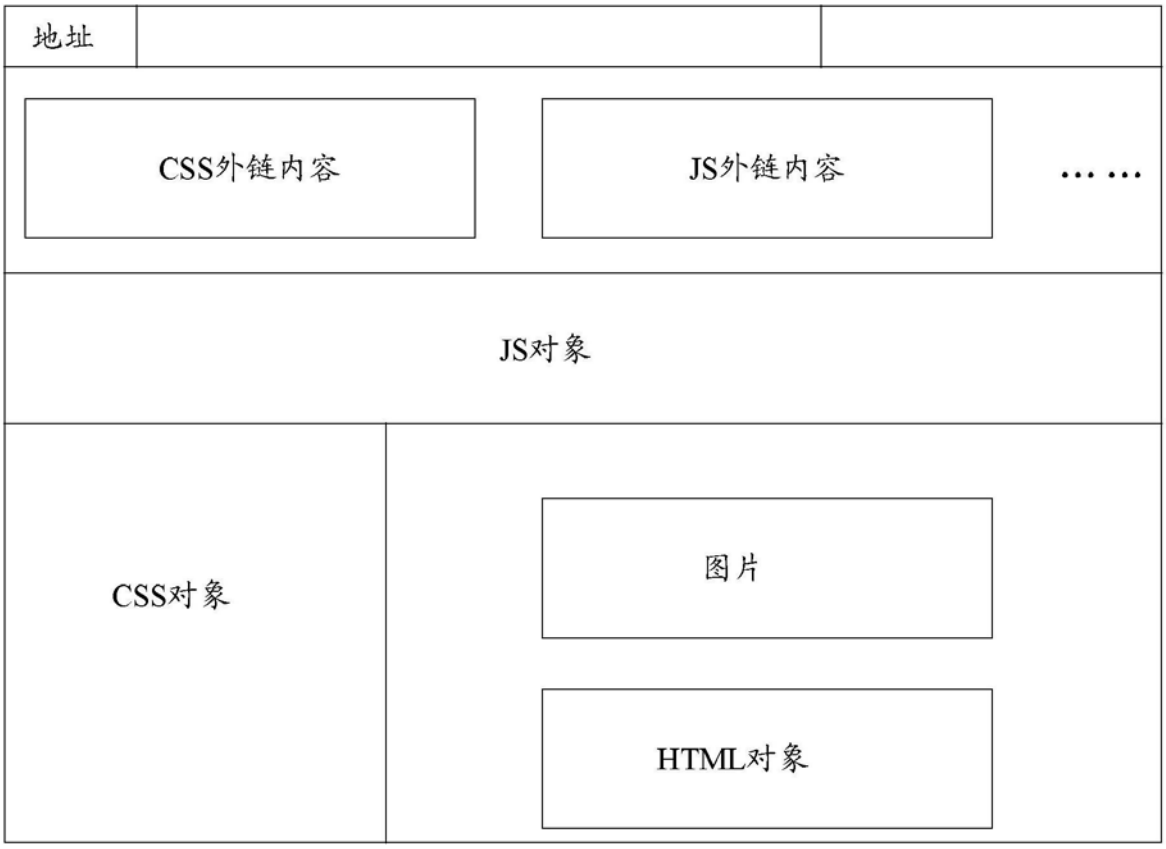


图6

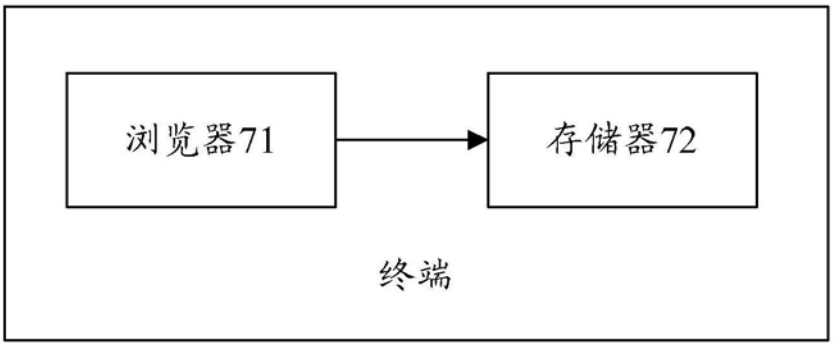


图7

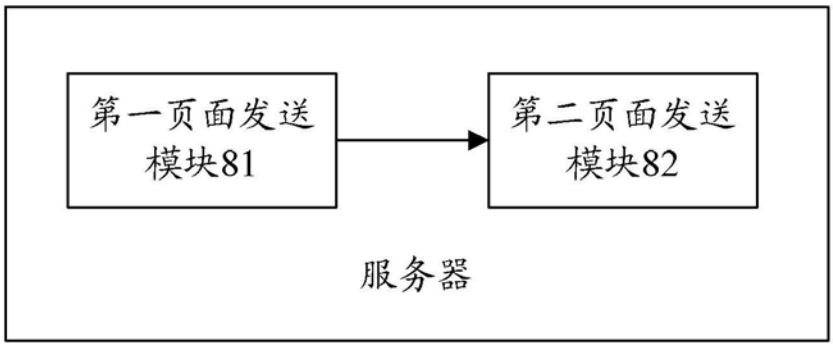


图8

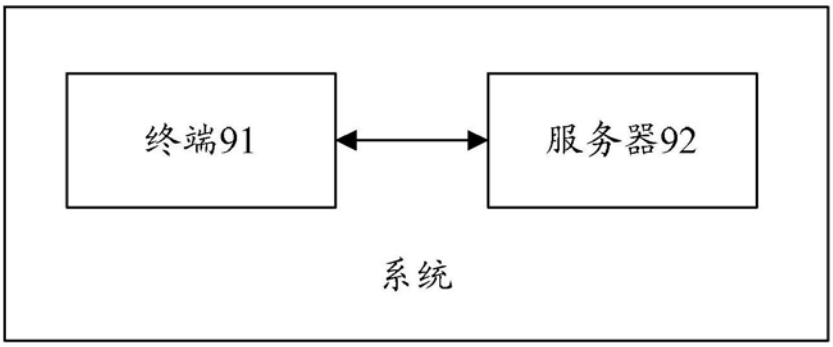


图9