(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 102799640 A
(43)申请公布日 2012.11.28

(21)申请号 201210218414.2

(22)申请日 2012.06.27

(71)申请人 用友软件股份有限公司
地址 100094 北京市海淀区北清路 68 号用友软件园

(72)发明人 张晓燕

(74)专利代理机构 北京友联知识产权代理事务所（普通合伙） 11343
代理人 尚志峰 汪海屏

(51)Int.Cl.
G06F 17/30(2006.01)

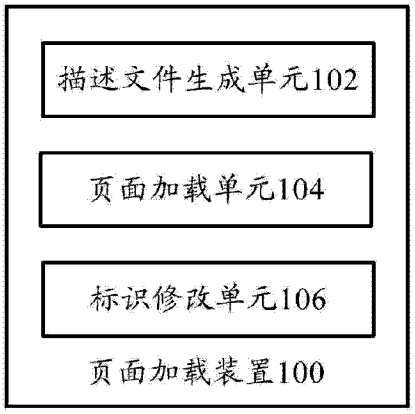
权利要求书 1 页 说明书 7 页 附图 4 页

(54)发明名称

页面加载装置和页面加载方法

(57)摘要

本发明提供了一种页面加载装置，包括：描述文件生成单元，针对页面中的应用所引用的每个外部文件生成一个描述文件，描述文件中包括关联描述文件与应用的唯一标识、关联描述文件与外部文件的文件标识；页面加载单元，在需要加载页面时，解析页面中的应用，根据应用对应的描述文件加载与应用对应的外部文件。在该技术方案中，通过针对每个外部文件生成一个描述文件，从而在加载外部文件时，可以通过唯一标识从应用关联到描述文件，在通过文件标识从描述文件关联到与应用对应的外部文件，可以快速地找到与所加载应用对应的外部文件，有效地提高了页面地加载效率。本发明还提供了一种页面加载方法，可以动态添加系统控件以及动态地修改外部文件名称，实现了页面的动态加载。



CN 102799640 A

1. 一种页面加载装置，其特征在于，包括：

描述文件生成单元，针对页面中的应用所引用的每个外部文件生成一个描述文件，所述描述文件中包括关联所述描述文件与所述应用的唯一标识、关联所述描述文件与所述外部文件的文件标识；

页面加载单元，在需要加载所述页面时，解析所述页面中的应用，根据所述应用对应的描述文件加载与所述应用对应的外部文件。

2. 根据权利要求 1 所述的页面加载装置，其特征在于，所述描述文件生成单元还用于，当所述页面需要引用额外的外部文件时，针对该额外的外部文件生成一个对应的描述文件。

3. 根据权利要求 1 所述的页面加载装置，其特征在于，还包括：

标识修改单元，当需要所述页面需要加载的外部文件的文件名改变时，对该外部文件对应的描述文件中的文件标识进行修改。

4. 根据权利要求 1 至 3 中任一项所述的页面加载装置，其特征在于：

所述描述文件还包括使用范围字段，用于标识相应的外部文件所对应的页面配置方式；以及，

在加载所述页面时，根据所述页面的配置方式选择相应的描述文件。

5. 根据权利要求 1 至 3 中任一项所述的页面加载装置，其特征在于：

所述描述文件还包括依赖类字段，用于标识该描述文件对应的外部文件与其它外部文件的依赖关系；以及，

在加载所述页面时，如果所述应用对应的外部文件依赖于其它外部文件，则同时加载所述其它外部文件。

6. 一种页面加载方法，其特征在于：

针对页面的应用所引用的每个外部文件生成一个描述文件，所述描述文件中包括关联所述描述文件与所述应用的唯一标识、关联所述描述文件与所述外部文件的文件标识；

在需要加载所述页面时，解析所述页面中的应用，根据所述应用对应的描述文件加载与所述应用对应的外部文件。

7. 根据权利要求 6 所述的页面加载方法，其特征在于，当所述页面需要引用额外的外部文件时，针对该额外的外部文件生成一个对应的描述文件。

8. 根据权利要求 6 所述的页面加载方法，其特征在于，当需要所述页面需要加载的外部文件的文件名改变时，对该外部文件对应的描述文件中的文件标识进行修改。

9. 根据权利要求 6 至 8 中任一项所述的页面加载方法，其特征在于：

所述描述文件还包括使用范围字段，用于标识相应的外部文件所对应的页面配置方式；以及，

在加载所述页面时，根据所述页面的配置方式选择相应的描述文件。

10. 根据权利要求 6 至 8 中任一项所述的页面加载方法，其特征在于：

所述描述文件还包括依赖类字段，用于标识该描述文件对应的外部文件与其它外部文件的依赖关系；以及，

在加载所述页面时，如果所述应用对应的外部文件依赖于其它外部文件，则同时加载所述其它外部文件。

页面加载装置和页面加载方法

技术领域

[0001] 本发明涉及页面加载技术领域，具体而言，涉及一种页面加载装置和一种页面加载方法。

背景技术

[0002] 随着异步加载的普及，WEB 开发已经在在更多的系统中得到了广泛的应用并且已经得到了越来越多程序员的重视。在可以很好的提高用户体验的同时，WEB 应用一个重要的方面是需要引入大量的 JS、CSS 等额外文件的信息，特别是当一个 WEB 页面根据不同的业务需求展现界面不同时，引入的文件会有大量的增加。而引入 JS、CSS 文件等也存在不可忽视的缺点就是引入量多了会严重影响 WEB 应用的速度。因为目前的浏览器在加载外部的如 JS 等文件时，在同一时间，页面只会加载一个文件。在第一个文件加载并执行完之前，第二个要引入的文件不会下载和执行。所以如何提高 JS 等的加载及执行速度是需要我们关注的重大课题。

[0003] 根据实际的应用场景，根据需要仅仅加载页面所需的文件从而提高页面的加载速度是一个重要的课题，即按需加载。在实现页面按需加载的同时，如果系统需要添加额外的一个文件，而这个文件在已经实现的每个页面中都需要引入，在这样的情况下，修改每个现有的页面会严重影响页面的加载效率；或者如果需要修改已加载的外部文件的文件名，则需要重新加载该文件，也会严重影响页面的加载效率。

[0004] 为了提高外部文件如 JS 等文件的加载及执行速度，在以往的实现中，主要采用以下几种方式：

[0005] 优先将页面的静态内容先加载完，再来处理 JS 的调用，采用这样的方案，在页面的加载过程中不会因为个别的 JS 调用加载缓慢而影响整个页面的加载，但是如果页面引用大量的 JS 代码时，此方案并不适用；

[0006] 优化 JS 代码的效率，将引用的 JS 页面做压缩处理，目前一般的网站都采用了此方式。为了提高 JS 的下载速度，压缩 JS 文件可以减少 JS 文件的大小；另外，把多个 JS 文件合并成一个也能因为减少服务器的响应次数而加快网页下载。但是，如果引用 JS 文件过多的情况下，压缩也只能部分提高速度，而将多个 JS 合并会增加 JS 文件的复杂性，并且降低了代码的可维护性；

[0007] 让影响页面速度 JS 函数延迟执行，延迟执行在一定程度上可以提高页面的渲染速度，但还是需要把多个 JS 文件引入。无法解决系统中需要引用大量 JS 文件的加载问题。

[0008] 因此，需要一种新的页面加载装置，在提高页面加载速度的同时，可以动态给系统添加额外需要引用的文件或修改引用的文件名，而不需要修改现成的每个页面，从而提高整个系统的可扩展性。

发明内容

[0009] 本发明所要解决的技术问题在于，提供一种新的页面加载装置，在提高页面加载

速度的同时，可以动态给系统添加额外需要引用的文件或修改引用的文件名，而不需要修改现成的每个页面，从而提高整个系统的可扩展性。

[0010] 有鉴于此，本发明提供了一种页面加载装置，包括：描述文件生成单元，针对页面中的应用所引用的每个外部文件生成一个描述文件，所述描述文件中包括关联所述描述文件与所述应用的唯一标识、关联所述描述文件与所述外部文件的文件标识；页面加载单元，在需要加载所述页面时，解析所述页面中的应用，根据所述应用对应的描述文件加载与所述应用对应的外部文件。在该技术方案中，通过针对每个外部文件生成一个描述文件，从而在加载外部文件时，可以通过唯一标识从应用关联到描述文件，在通过文件标识从描述文件关联到与应用对应的外部文件，可以快速地找到与所加载应用对应的外部文件，有效地提高了页面地加载效率。其中，在 WEB 页面中，所谓的应用例如可以是布局或者控件，而所谓的外部文件例如可以是 JS 文件或 CSS 文件等。

[0011] 优选地，所述描述文件生成单元还用于，当所述页面需要引用额外的外部文件时，针对该额外的外部文件生成一个对应的描述文件。在该技术方案中，在系统需要加载额外的外部文件时，不需要将每个需要加载该外部文件的页面均重新加载，只需生成一个相应的描述文件，再根据新生成的描述文件完成该外部文件的加载即可，实现了页面的动态加载，提高了页面的加载效率。

[0012] 优选地，所述的页面加载装置还包括：标识修改单元，当需要所述页面需要加载的外部文件的文件名改变时，对该外部文件对应的描述文件中的文件标识进行修改。在该技术方案中，在已加载的外部文件的文件名修改时，不需要重新加载该外部文件，只需要修改相应的描述文件即可，而不需要修改每个引入该外部文件的页面，提高了页面的加载效率。

[0013] 优选地，所述描述文件还包括使用范围字段，用于标识相应的外部文件所对应的页面配置方式；以及，在加载所述页面时，根据所述页面的配置方式选择相应的描述文件。在该技术方案中，通过在描述文件中增加使用范围字段，从而使该技术方案可以适用于不同的页面配置方式，例如 app 应用或 page 应用等，通过使用范围字段可以根据页面的配置方式选择相应的描述文件，扩展了本发明中技术方案的使用范围。

[0014] 优选地，所述描述文件还包括依赖类字段，用于标识该描述文件对应的外部文件与其它外部文件的依赖关系；以及，在加载所述页面时，如果所述应用对应的外部文件依赖于其它外部文件，则同时加载所述其它外部文件。在该技术方案中，通过依赖类字段可将相互依赖的外部文件之间进行关联，而不需要将所有的外部文件均与应用关联，使整个页面加载的逻辑性更强，效率更高。

[0015] 本发明还提供了一种页面加载方法，针对页面的应用所引用的每个外部文件生成一个描述文件，所述描述文件中包括关联所述描述文件与所述应用的唯一标识、关联所述描述文件与所述外部文件的文件标识；在需要加载所述页面时，解析所述页面中的应用，根据所述应用对应的描述文件加载与所述应用对应的外部文件。在该技术方案中，通过针对每个外部文件生成一个描述文件，从而在加载外部文件时，可以通过唯一标识从应用关联到描述文件，在通过文件标识从描述文件关联到与应用对应的外部文件，可以快速地找到与所加载应用对应的外部文件，有效地提高了页面地加载效率。其中，在 WEB 页面中，所谓的应用例如可以是布局或者控件，而所谓的外部文件例如可以是 JS 文件或 CSS 文件等。

[0016] 优选地，当所述页面需要引用额外的外部文件时，针对该额外的外部文件生成一

个对应的描述文件。在该技术方案中，在系统需要加载额外的外部文件时，不需要将每个需要加载该外部文件的页面均重新加载，只需生成一个相应的描述文件，再根据新生成的描述文件完成该外部文件的加载即可，实现了页面的动态加载，提高了页面的加载效率。

[0017] 优选地，当需要所述页面需要加载的外部文件的文件名改变时，对该外部文件对应的描述文件中的文件标识进行修改。在该技术方案中，在已加载的外部文件的文件名修改时，不需要重新加载该外部文件，只需要修改相应的描述文件即可，而不需要修改每个引入该外部文件的页面，提高了页面的加载效率。

[0018] 优选地，所述描述文件还包括使用范围字段，用于标识相应的外部文件所对应的页面配置方式；以及，在加载所述页面时，根据所述页面的配置方式选择相应的描述文件。在该技术方案中，通过在描述文件中增加使用范围字段，从而使该技术方案可以适用于不同的页面配置方式，例如 app 应用或 page 应用等，通过使用范围字段可以根据页面的配置方式选择相应的描述文件，扩展了本发明中技术方案的使用范围。

[0019] 优选地，所述描述文件还包括依赖类字段，用于标识该描述文件对应的外部文件与其它外部文件的依赖关系；以及，在加载所述页面时，如果所述应用对应的外部文件依赖于其它外部文件，则同时加载所述其它外部文件。在该技术方案中，通过依赖类字段可将相互依赖的外部文件之间进行关联，而不需要将所有的外部文件均与应用关联，使整个页面加载的逻辑性更强，效率更高。

[0020] 综上所述，通过针对每个外部文件生成一个描述文件，可以快速地找到与所加载应用对应的外部文件，有效地提高了页面地加载效率；在系统需要加载额外的外部文件时，只需生成一个相应的描述文件，再根据新生成的描述文件完成该外部文件的加载即可，以及在已加载的外部文件的文件名修改时，只需要修改相应的描述文件即可，实现了页面的动态加载，提高了页面的加载效率。

附图说明

[0021] 图 1 是根据本发明实施例的页面加载装置的框图；

[0022] 图 2 是根据本发明实施例的页面加载方法的流程图；

[0023] 图 3 是根据本发明实施例的页面加载方法的流程图；

[0024] 图 4 是根据本发明实施例的描述文件的解析过程的流程图；

[0025] 图 5 是根据本发明实施例的页面解析的过程的流程图。

具体实施方式

[0026] 为了能够更清楚地理解本发明的上述目的、特征和优点，下面结合附图和具体实施方式对本发明进行进一步的详细描述。

[0027] 在下面的描述中阐述了很多具体细节以便于充分理解本发明，但是，本发明还可以采用其他不同于在此描述的其他方式来实施，因此，本发明的保护范围并不受下面公开的具体实施例的限制。

[0028] 下面结合附图和实施例对本发明做进一步说明。需要说明的是，在不冲突的情况下，本申请的实施例及实施例中的特征可以相互组合。

[0029] 如图 1 所示，本发明提供了一种页面加载装置 100，包括：描述文件生成单元 102，

针对页面中的应用所引用的每个外部文件生成一个描述文件，描述文件中包括关联描述文件与应用的唯一标识、关联描述文件与外部文件的文件标识；页面加载单元 104，在需要加载页面时，解析页面中的应用，根据应用对应的描述文件加载与应用对应的外部文件。在该技术方案中，通过针对每个外部文件生成一个描述文件，从而在加载外部文件时，可以通过唯一标识从应用关联到描述文件，在通过文件标识从描述文件关联到与应用对应的外部文件，可以快速地找到与所加载应用对应的外部文件，有效地提高了页面地加载效率。

[0030] 其中，在 WEB 页面中，所谓的应用例如可以是布局或者控件，而所谓的外部文件例如可以是 JS 文件或 CSS 文件等。具体而言：

[0031] WEB 页面，描述一个业务逻辑的整体过程，一个页面由多个组成部分，主要包括布局和控件；

[0032] 控件，页面的基本组成部分，在应用系统中，控件是一种具有图形表示的对象，它可以显示在屏幕上并与用户实现交互，在应用中，每个控件对应一个 JS 文件和对与此控件样式的 CSS 文件描述；

[0033] 布局，页面的格局的整体配置，可以放置其它控件的容，在整个应用系统中，每个布局对应一个 JS 文件和对此布局样式的 CSS 文件描述。

[0034] 优选地，描述文件生成单元 102 还用于，当页面需要引用额外的外部文件时，针对该额外的外部文件生成一个对应的描述文件。在该技术方案中，在系统需要加载额外的外部文件时，不需要将每个需要加载该外部文件的页面均重新加载，只需生成一个相应的描述文件，再根据新生成的描述文件完成该外部文件的加载即可，实现了页面的动态加载，提高了页面的加载效率。

[0035] 优选地，的页面加载装置 100 还包括：标识修改单元 106，当需要页面需要加载的外部文件的文件名改变时，对该外部文件对应的描述文件中的文件标识进行修改。在该技术方案中，在已加载的外部文件的文件名修改时，不需要重新加载该外部文件，只需要修改相应的描述文件即可，而不需要修改每个引入该外部文件的页面，提高了页面的加载效率。

[0036] 优选地，描述文件还包括使用范围字段，用于标识相应的外部文件所对应的页面配置方式；以及，在加载页面时，根据页面的配置方式选择相应的描述文件。在该技术方案中，通过在描述文件中增加使用范围字段，从而使该技术方案可以适用于不同的页面配置方式，例如 app 应用或 page 应用等，通过使用范围字段可以根据页面的配置方式选择相应的描述文件，扩展了本发明中技术方案的使用范围。

[0037] 优选地，描述文件还包括依赖类字段，用于标识该描述文件对应的外部文件与其它外部文件的依赖关系；以及，在加载页面时，如果应用对应的外部文件依赖于其它外部文件，则同时加载其它外部文件。在该技术方案中，通过依赖类字段可将相互依赖的外部文件之间进行关联，而不需要将所有的外部文件均与应用关联，使整个页面加载的逻辑性更强，效率更高。

[0038] 本发明还提供了一种页面加载方法，包括：针对页面的应用所引用的每个外部文件生成一个描述文件，描述文件中包括关联描述文件与应用的唯一标识、关联描述文件与外部文件的文件标识；在需要加载页面时，解析页面中的应用，根据应用对应的描述文件加载与应用对应的外部文件。在该技术方案中，通过针对每个外部文件生成一个描述文件，从而在加载外部文件时，可以通过唯一标识从应用关联到描述文件，在通过文件标识从描述

文件关联到与应用对应的外部文件，可以快速地找到与所加载应用对应的外部文件，有效地提高了页面地加载效率。其中，在 WEB 页面中，所谓的应用例如可以是布局或者控件，而所谓的外部文件例如可以是 JS 文件或 CSS 文件等。

[0039] 优选地，当页面需要引用额外的外部文件时，针对该额外的外部文件生成一个对应的描述文件。在该技术方案中，在系统需要加载额外的外部文件时，不需要将每个需要加载该外部文件的页面均重新加载，只需生成一个相应的描述文件，再根据新生成的描述文件完成该外部文件的加载即可，实现了页面的动态加载，提高了页面的加载效率。

[0040] 优选地，当需要页面需要加载的外部文件的文件名改变时，对该外部文件对应的描述文件中的文件标识进行修改。在该技术方案中，在已加载的外部文件的文件名修改时，不需要重新加载该外部文件，只需要修改相应的描述文件即可，而不需要修改每个引入该外部文件的页面，提高了页面的加载效率。

[0041] 优选地，描述文件还包括使用范围字段，用于标识相应的外部文件所对应的页面配置方式；以及，在加载页面时，根据页面的配置方式选择相应的描述文件。在该技术方案中，通过在描述文件中增加使用范围字段，从而使该技术方案可以适用于不同的页面配置方式，例如 app 应用或 page 应用等，通过使用范围字段可以根据页面的配置方式选择相应的描述文件，扩展了本发明中技术方案的使用范围。

[0042] 优选地，描述文件还包括依赖类字段，用于标识该描述文件对应的外部文件与其它外部文件的依赖关系；以及，在加载页面时，如果应用对应的外部文件依赖于其它外部文件，则同时加载其它外部文件。在该技术方案中，通过依赖类字段可将相互依赖的外部文件之间进行关联，而不需要将所有的外部文件均与应用关联，使整个页面加载的逻辑性更强，效率更高。

[0043] 本发明中，在整个动态页面的渲染机制的前提是每个布局和控件对应一个 JS、CSS 文件，在此基础上针对每个布局和控件生成一个描述文件，在页面解析的过程中，根据描述文件按需加载页面所包含布局和控件。下面将对布局和页面的控件描述文件的实现以及建立过程进行详细的描述。

[0044] 布局的描述文件主要包含以下几个部分：

[0045] 唯一标识：用于标识描述 JS 文件的唯一性。

[0046] 引用的样式：用于标识 JS 文件的显示样式，即此布局对应的 CSS 文件。

[0047] 对应的 JS 文件：用于标识此文件描述的是哪个布局。

[0048] 布局解析文件：用于标识如何将页面的描述解析成 JS 布局的应用。

[0049] 对应的后台描述文件：布局描述文件对应的后台描述类。

[0050] 使用范围：布局控件应用的范围。

[0051] 控件的描述文件主要包含以下几个部分：

[0052] 唯一标识：用于标识描述 JS 文件的唯一性。

[0053] 引用的样式：用于标识控件的显示样式，即此控件对应的 CSS 文件。

[0054] 对应的 JS 文件：用于标识此文件描述的是哪个 JS 控件。

[0055] 控件解析文件：用于标识如果将页面的描述解析成 JS 控件的应用。

[0056] 对应的后台描述文件：控件描述文件对应的后台描述类。

[0057] 使用范围：布局控件应用的范围。

[0058] 依赖类：此 JS 文件依赖的其他 JS 文件。

[0059] 基于上述技术方案，以 WEB 页面的加载为例，如图 2 所示，本发明中的页面加载流程如下：

[0060] 步骤 202，针对系统的应用，将引用的外部的 JS 文件描述为一个对应的描述文件。

[0061] 步骤 204，解析系统需要引用的外部的 JS 文件的描述文件。

[0062] 步骤 206，解析一个页面应用页面，将步骤 202 中解析的文件引入到页面中，并加载这些文件。

[0063] 步骤 208，如果系统需要引入额外的引用文件时，只需要添加 JS 文件及与 JS 对应的描述文件，而不需要在页面中重新引入 JS 文件；如果需要改变某个 JS 文件文件名，不需要修改每个引入此文件的页面，只需要修改描述文件即可。

[0064] 具体而言，如图 3 所示，WEB 页面的加载流程如下：

[0065] 步骤 302，解析整个 WEB 页面的应用，解析出整个 WEB 页面所包含的布局和控件等信息。

[0066] 步骤 304，解析系统的整体布局和控件的对应的描述文件，将布局和控件对应的 JS、CSS 等文件放入容器中。此容器将在真正需要加载页面的时候调用。

[0067] 步骤 306，执行页面的顶层布局。

[0068] 步骤 308，检测该布局对应的 JS、CSS 文件是否已经包含在页面加载文件中，如果是，则进入步骤 312，如果否，则进入步骤 310。

[0069] 步骤 310，引入与该布局对应的 JS、CSS 文件。

[0070] 步骤 312，加载该布局对应的 JS、CSS 文件。

[0071] 步骤 314，判断该布局下面是否包含其他子布局，如果是，则进入步骤 308，如果否，则进入步骤 316。

[0072] 步骤 316，检测顶层布局下是否包括其它控件，如果是，则进入步骤 318，如果否，则进入步骤 324。

[0073] 步骤 318，检测控件对应的 JS、CSS 文件是否已经引入，如果是，则进入步骤 322，如果否，则进入步骤 320。

[0074] 步骤 320，引入与该控件对应的 JS、CSS 文件。

[0075] 步骤 322，加载该控件对应的 JS、CSS 文件。

[0076] 步骤 324，判断页面加载是否结束，如果否，则进入步骤 308 中。

[0077] 如图 4 所示，在以上页面的加载过程中，描述文件的解析过程具体如下：

[0078] 步骤 402，将页面加载过程中必须包含的 JS、CSS 文件导入。

[0079] 步骤 404，判断页面的应用类型，根据页面的类型选择相应的描述文件。例如系统支持 app 和 page 两种页面配置方式，不同的配置方式需要引入的 JS、CSS 等文件有所不同。

[0080] 步骤 406，解析相应的描述文件。

[0081] 步骤 408，解析描述文件的依赖属性，判断是否依赖于其它控件，如果是，则进入步骤 410，如果否，则进入步骤 412。在解析描述文件的过程中，解析出此描述文件的依赖属性，如果此属性的不为空，则表明此配置文件依赖于其他的描述文件，即此控件是否依赖于其他的控件

[0082] 步骤 410，将依赖的控件描述文件放入容器中。

[0083] 步骤 412，将此布局或控件对应的描述文件放入容器中。

[0084] 循环执行步骤 408 至 412，直至所有的配置文件均放入容器中。

[0085] 如图 5 所示，页面解析的过程具体如下：

[0086] 步骤 502，解析 WEB 应用页面的文件信息。

[0087] 步骤 504，渲染页面的布局和控件信息。

[0088] 步骤 506，从容器中取出布局的信息。

[0089] 步骤 508，将此布局对应的 JS 和 CSS 文件等加入到页面的中。

[0090] 步骤 510，解析布局下的子布局信息。

[0091] 步骤 512，从容器中将子布局对应的 JS 和 CSS 文件等加载到页面中。

[0092] 步骤 514，解析页面中包含的控件。

[0093] 步骤 516，从容器中将控件对应的 JS 和 CSS 文件等加载到页面中。

[0094] 步骤 518，判断页面是否渲染结束，如果否，则返回至步骤 510。

[0095] 基于以上页面加载方法，如果系统需要引入额外的外部文件时，只需要添加 JS 文件及与 JS 对应的描述文件，而不需要在页面中重新引入 JS 文件；如果需要改变某个 JS 文件文件名，不需要修改每个引入此文件的页面，只需要修改配置文件即可。

[0096] 具体如下：

[0097] 如果系统需要新的控件，给系统添加相应的 JS 文件和与 JS 文件对应的配置文件等信息，将需要此控件的页面中引入此控件信息即可；

[0098] 如果系统需要修改某个布局或控件的对应的文件名等信息，找出与此控件或布局对应的描述文件信息，修改此配置文件信息。

[0099] 通过以上分析可以看出：可动态配置的提高 WEB 页面渲染速度的方案在实际的应用中，只加载页面渲染时所需要的文件，在不影响现有系统的情况下，可以实现可扩展的添加额外的文件，从而实现系统的可扩展性；如果需要改变某个 JS 文件文件名，不需要修改每个引入此文件的页面，只需要修改配置文件即可，从而提高系统的稳定性。

[0100] 综上所述，通过针对每个外部文件生成一个描述文件，可以快速地找到与所加载应用对应的外部文件，有效地提高了页面地加载效率；在系统需要加载额外的外部文件时，只需生成一个相应的描述文件，再根据新生成的描述文件完成该外部文件的加载即可，以及在已加载的外部文件的文件名修改时，只需要修改相应的描述文件即可，实现了页面的动态加载，提高了页面的加载效率，提高了系统的稳定性。

[0101] 以上所述仅为本发明的优选实施例而已，并不用于限制本发明，对于本领域的技术人员来说，本发明可以有各种更改和变化。凡在本发明的精神和原则之内，所作的任何修改、等同替换、改进等，均应包含在本发明的保护范围之内。


描述文件生成单元102
页面加载单元104
标识修改单元106
页面加载装置100

图 1

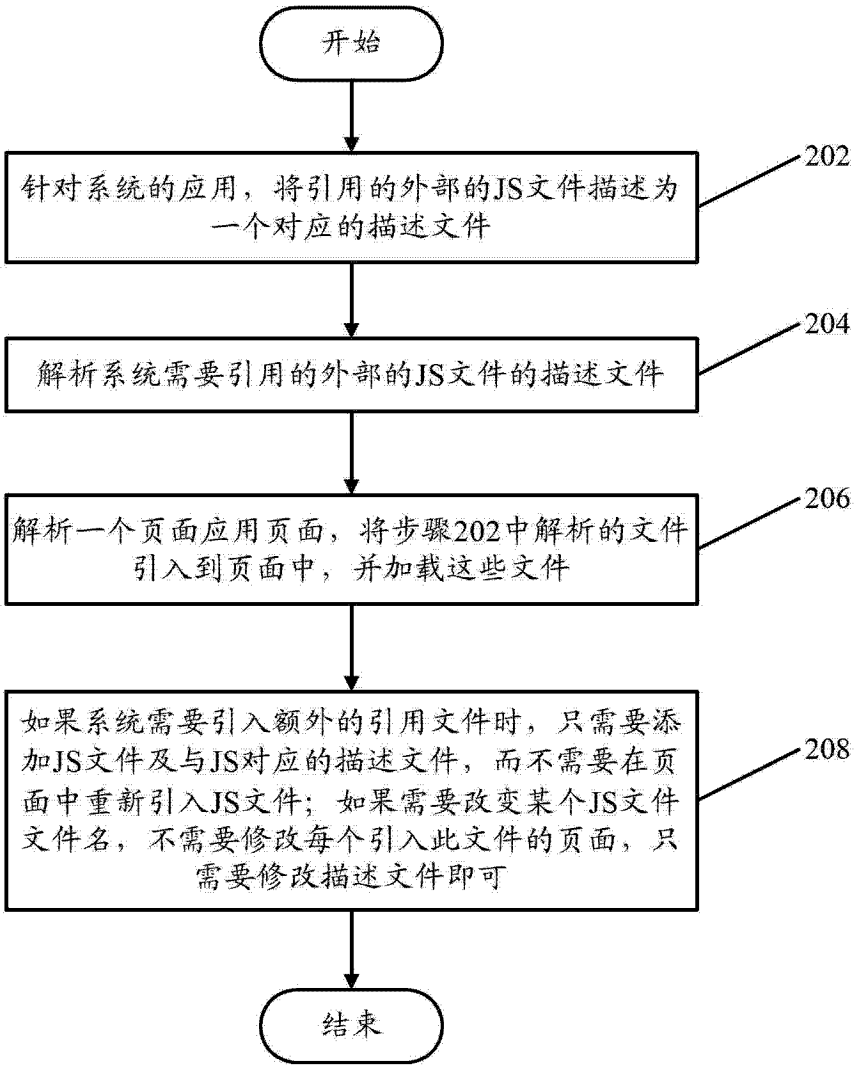

开始
针对系统的应用，将引用的外部的JS文件描述为一个对应的描述文件
202
解析系统需要引用的外部的JS文件的描述文件
204
解析一个页面应用页面，将步骤202中解析的文件引入到页面中，并加载这些文件
206
如果系统需要引入额外的引用文件时，只需要添加JS文件及与JS对应的描述文件，而不需要在页面中重新引入JS文件；如果需要改变某个JS文件文件名，不需要修改每个引入此文件的页面，只需要修改描述文件即可
208
结束

图 2

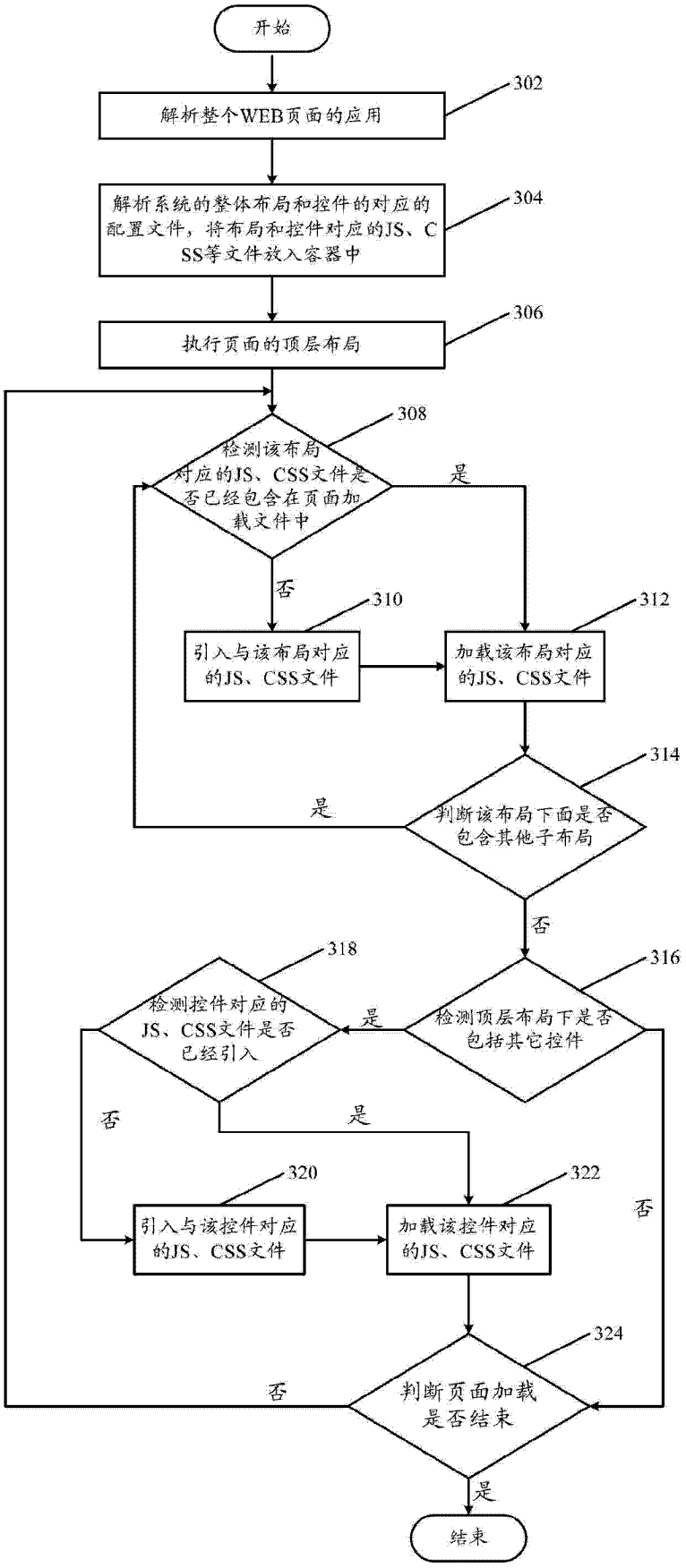

开始
302
解析整个WEB页面的应用
304
解析系统的整体布局和控件的对应的配置文件，将布局和控件对应的JS、CSS等文件放入容器中
306
执行页面的顶层布局
308
检测该布局对应的JS、CSS文件是否已经包含在页面加载文件中
是
否
310
引入与该布局对应的JS、CSS文件
312
加载该布局对应的JS、CSS文件
314
判断该布局下面是否包含其他子布局
是
否
316
检测顶层布局下是否包括其它控件
318
检测控件对应的JS、CSS文件是否已经引入
是
否
是
320
引入与该控件对应的JS、CSS文件
322
加载该控件对应的JS、CSS文件
否
324
判断页面加载是否结束
否
是
结束

图 3

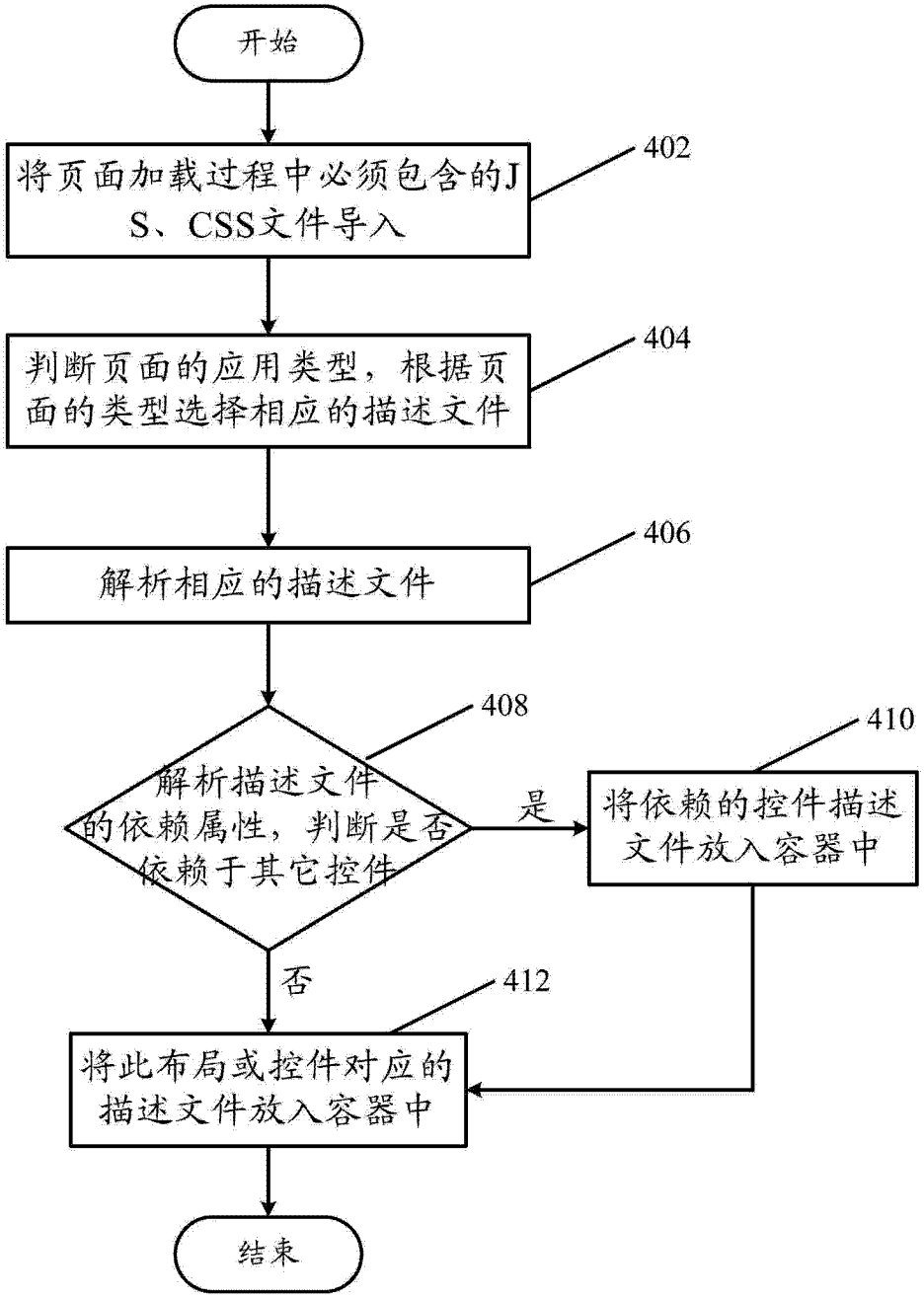

开始
将页面加载过程中必须包含的JS、CSS文件导入
402
判断页面的应用类型，根据页面的类型选择相应的描述文件
404
解析相应的描述文件
406
解析描述文件的依赖属性，判断是否依赖于其它控件
408
是
将依赖的控件描述文件放入容器中
410
否
将此布局或控件对应的描述文件放入容器中
412
结束

图 4

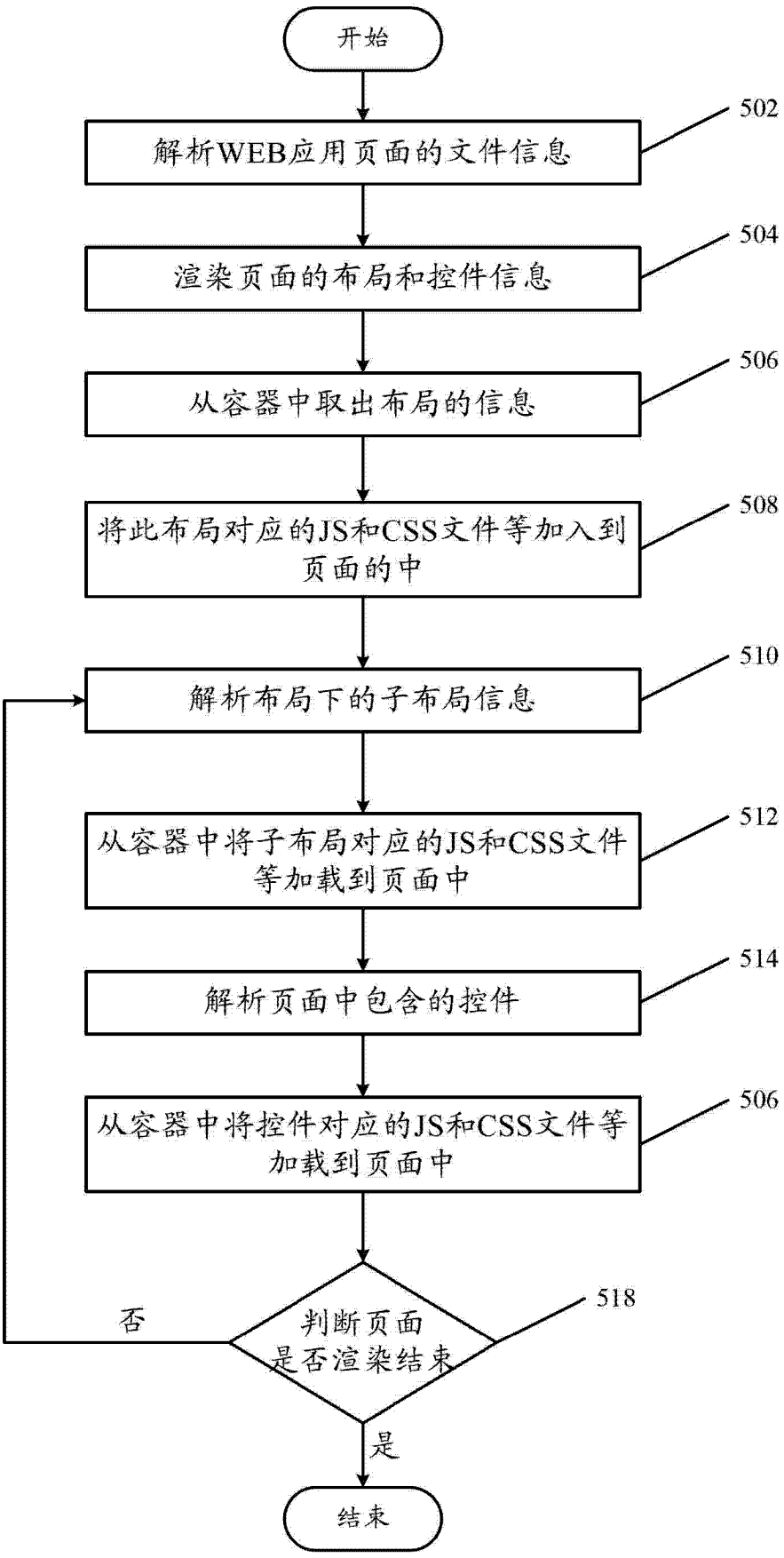

开始
解析WEB应用页面的文件信息
502
渲染页面的布局和控件信息
504
从容器中取出布局的信息
506
将此布局对应的JS和CSS文件等加入到页面的中
508
解析布局下的子布局信息
510
从容器中将子布局对应的JS和CSS文件等加载到页面中
512
解析页面中包含的控件
514
从容器中将控件对应的JS和CSS文件等加载到页面中
506
判断页面是否渲染结束
518
否
是
结束

图 5