(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 103942137 A
(43)申请公布日 2014.07.23

(21)申请号 201310025121.7

(22)申请日 2013.01.23

(71)申请人 腾讯科技(深圳)有限公司
地址 518000 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 符阳辉

(74)专利代理机构 深圳翼盛智成知识产权事务所(普通合伙) 44300
代理人 欧阳启明 李捷

(51)Int.Cl.
G06F 11/36(2006.01)

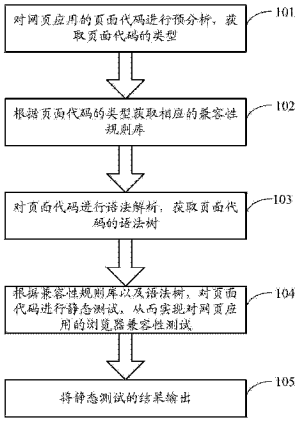
权利要求书2页 说明书6页 附图3页

(54)发明名称

浏览器兼容性测试方法及装置

(57)摘要

本发明涉及一种浏览器兼容性测试方法及装置。该浏览器兼容性测试方法包括步骤对网页应用的页面代码进行预处理,获取页面代码的类型;根据页面代码的类型获取相应的兼容性规则库;对页面代码进行语法解析,获取页面代码的语法树;以及根据兼容性规则库以及语法树,对页面代码进行静态测试,从而实现对网页应用的浏览器兼容性测试。本发明还涉及一种浏览器兼容性测试装置。本发明的浏览器兼容性测试方法及装置通过对网页应用的页面代码进行静态测试,实现浏览器的兼容性测试;整个测试过程操作简单,维护成本低。



CN 103942137 A

1. 一种浏览器兼容性测试方法，其特征在于，包括步骤：

对网页应用的页面代码进行预处理，获取所述页面代码的类型；

根据所述页面代码的类型获取相应的兼容性规则库；

对所述页面代码进行语法解析，获取所述页面代码的语法树；以及

根据所述兼容性规则库以及所述语法树，对所述页面代码进行静态测试，从而实现对所述网页应用的浏览器兼容性测试。

2. 根据权利要求 1 所述的浏览器兼容性测试方法，其特征在于，所述浏览器兼容性测试方法还包括步骤：

根据对所述页面代码进行兼容性判断的多个正则表达式和 / 或所述多个正则表达式之间的逻辑判断关系，使用文本格式的 json 文件创建所述兼容性规则库。

3. 根据权利要求 1 所述的浏览器兼容性测试方法，其特征在于，所述对网页应用的页面代码进行预分析，获取所述页面代码的类型的步骤包括：

对所述页面应用的页面代码进行词法分析，获取所述页面代码的记号序列；以及

根据所述网页应用的页面代码的记号序列，获取所述页面代码的类型。

4. 根据权利要求 3 所述的浏览器兼容性测试方法，其特征在于，所述根据所述网页应用的页面代码的记号序列，获取所述页面代码的类型的步骤包括：

获取所述记号序列的第一个记号；

如所述第一个记号为“<”，则判断所述页面代码的类型为 HTML 类型；

如所述第一个记号为“{”或“[”，则判断所述页面代码的类型为 JSON 类型；

如所述第一个记号为“@”、“*”、“#”、“.”或“：”，则判断所述页面代码的类型为 CSS 类型；

否则判断所述页面代码的类型为 JavaScript 类型。

5. 根据权利要求 1 所述的浏览器兼容性测试方法，其特征在于，所述浏览器兼容性测试方法还包括步骤：

将所述静态测试的结果输出。

6. 一种浏览器兼容性测试装置，其特征在于，包括：

类型获取模块，用于对网页应用的页面代码进行预处理，获取所述页面代码的类型；

规则库获取模块，用于根据所述页面代码的类型获取相应的兼容性规则库；

语法树获取模块，用于对所述页面代码进行语法解析，获取所述页面代码的语法树；以及

测试模块，用于根据兼容性规则库以及所述语法树，对所述页面代码进行静态测试，从而实现对所述网页应用的浏览器兼容性测试。

7. 根据权利要求 6 所述的浏览器兼容性测试装置，其特征在于，根据对所述页面代码进行兼容性判断的多个正则表达式和 / 或所述多个正则表达式之间的逻辑判断关系，使用文本格式的 json 文件创建所述兼容性规则库。

8. 根据权利要求 6 所述的浏览器兼容性测试装置，其特征在于，所述类型获取模块包括：

词法分析单元，用于对所述页面应用的页面代码进行词法分析，获取所述页面代码的记号序列；以及

类型获取单元，用于根据所述网页应用的页面代码的记号序列，获取所述页面代码的类型。

9. 根据权利要求 8 所述的浏览器兼容性测试装置，其特征在于，所述类型获取单元具体用于：

获取所述记号序列的第一个记号；

如所述第一个记号为“<”，则判断所述页面代码的类型为 HTML 类型；

如所述第一个记号为“{”或“[”，则判断所述页面代码的类型为 JSON 类型；

如所述第一个记号为“@”、“*”、“#”、“.”或“：”，则判断所述页面代码的类型为 CSS 类型；

否则判断所述页面代码的类型为 JavaScript 类型。

10. 根据权利要求 6 所述的浏览器兼容性测试装置，其特征在于，所述浏览器兼容性测试装置还包括：

输出模块，用于将所述静态测试的结果输出。

浏览器兼容性测试方法及装置

技术领域

[0001] 本发明涉及互联网测试领域，特别是涉及一种操作简单、维护成本低的浏览器兼容性测试方法及装置。

背景技术

[0002] 随着社会的发展，越来越多的用户通过各种自己喜爱的浏览器进行网页浏览。由于浏览器的种类繁多，同时很多用户一直使用非标准的旧版本的浏览器，造成网页开发工程师必须具备跨浏览器开发的能力，其开发的网页应用必须能兼容多种主流的浏览器，以确保网页应用在不同的浏览器环境中具有相同的外观和功能。

[0003] 因此开发出来的网页应用需要在不同的浏览器中重复进行测试，尽管某些自动化测试工具可以一定程度上实现测试的自动化，但仍需测试者在不同的浏览器环境中运行自动化测试工具。此外，测试者需要确定在不同浏览器上的测试结果之间的差异，这样要求测试者具备较高的专业技能，以及需要付出大量的时间和精力。

[0004] 故，有必要提供一种操作简单、维护成本低的浏览器兼容性测试方法及装置，以解决现有技术所存在的问题。

发明内容

[0005] 本发明的目的在于提供一种基于网页应用的页面代码静态分析的浏览器兼容性测试方法及装置；整个测试过程操作简单，维护成本低，以解决现有的浏览器兼容性测试方法及装置操作复杂以及维护成本高的技术问题。

[0006] 为解决上述问题，本发明提供的技术方案如下：

[0007] 本发明涉及一种浏览器兼容性测试方法，其包括步骤：

[0008] 对网页应用的页面代码进行预处理，获取所述页面代码的类型；

[0009] 根据所述页面代码的类型获取相应的兼容性规则库；

[0010] 对所述页面代码进行语法解析，获取所述页面代码的语法树；以及

[0011] 根据所述兼容性规则库以及所述语法树，对所述页面代码进行静态测试，从而实现对所述网页应用的浏览器兼容性测试。

[0012] 本发明还涉及一种浏览器兼容性测试装置，其包括：

[0013] 类型获取模块，用于对网页应用的页面代码进行预分析，获取所述页面代码的类型；

[0014] 规则库获取模块，用于根据所述页面代码的类型获取相应的兼容性规则库；

[0015] 语法树获取模块，用于对所述页面代码进行语法解析，获取所述页面代码的语法树；以及

[0016] 测试模块，用于根据兼容性规则库以及所述语法树，对所述页面代码进行静态测试，从而实现对所述网页应用的浏览器兼容性测试。

[0017] 相较于现有技术，本发明的浏览器兼容性测试方法及装置通过对网页应用的页面

代码进行静态测试，实现浏览器的兼容性测试；整个测试过程操作简单，维护成本低。解决了现有的浏览器兼容性测试方法及装置操作复杂以及维护成本高的技术问题。

附图说明

[0018] 图 1 为本发明的浏览器兼容性测试方法的优选实施例的流程图；

[0019] 图 2 为本发明的浏览器兼容性测试方法的优选实施例的步骤 101 的详细流程图；

[0020] 图 3 为本发明的浏览器兼容性测试装置的优选实施例的结构示意图；

[0021] 其中，附图标记说明如下：

[0022] 31、类型获取模块；

[0023] 32、规则库获取模块；

[0024] 33、语法树获取模块；

[0025] 34、测试模块；

[0026] 35、输出模块。

具体实施方式

[0027] 以下各实施例的说明是参考附加的图式，用以例示本发明可用以实施的特定实施例。

[0028] 请参照图 1，图 1 为本发明的浏览器兼容性测试方法的优选实施例的流程图。该浏览器兼容性测试方法包括：

[0029] 步骤 101，对网页应用的页面代码进行预处理，获取页面代码的类型；

[0030] 步骤 102，根据页面代码的类型获取相应的兼容性规则库；

[0031] 步骤 103，对页面代码进行语法解析，获取页面代码的语法树；

[0032] 步骤 104，根据兼容性规则库以及语法树，对页面代码进行静态测试，从而实现对网页应用的浏览器兼容性测试；

[0033] 步骤 105，将静态测试的结果输出；

[0034] 本优选实施例的浏览器兼容性测试方法结束于步骤 105。

[0035] 下面详细说明本优选实施例的浏览器兼容性测试方法的各步骤的具体流程。

[0036] 在步骤 101 中，对网页应用的页面代码进行预处理，获取页面代码的类型。具体的流程如图 2 所示，图 2 为本发明的浏览器兼容性测试方法的优选实施例的步骤 101 的详细流程图。

[0037] 首先，对页面应用的页面代码进行词法分析，获取页面代码的记号（Token）序列。这里的记号为一个字符串，是构成源代码的最小单元，可使用词法分析器进行词法分析。词法分析器从左到右一个字符一个字符地读取页面应用的页面代码，即对构成页面代码的字符流进行扫描，然后根据构成规则识别记号（token），最终得到页面代码的记号序列。

[0038] 随后根据页面应用的页面代码的记号序列，获取页面代码的类型，具体为：获取记号序列中的第一个记号，如第一个记号为“<”，则判断页面代码的类型为 HTML（超文本标记语言，Hypertext Markup Language）类型（HTML 的页面代码一般以“<”字符开始）；如第一个记号为“{”或“[”，则判断页面代码的类型为 JSON（JavaScript Object Notation）类型（JSON 的页面代码一般以“{”或“[”字符开始）；如第一个记号为“@”、“*”、“#”、“.”或“:”，

则判断页面代码的类型为CSS(Cascading Style Sheets,层叠样式表单)类型(CSS的页面代码一般以“@”、“*”、“#”、“.”或“：”字符开始);否则判断页面代码的类型为JavaScript类型。在本步骤中通过页面代码的预定义的语法特征判定该代码的类型,上述仅为举例,使用其他通过页面代码的语法特性判定代码的类型的方法均属于本发明的保护范围。

[0039] 随后来到步骤102。

[0040] 在步骤102中,根据页面代码的类型获取相应的兼容性规则库,由于这里的兼容性规则库是根据对页面代码进行兼容性判断的多个正则表达式和/或多个正则表达式之间的逻辑判断关系,使用文本格式的json((JavaScript Object Notation)文件进行创建,不同的页面代码类型均具有相应的兼容性规则库。在本步骤中选定相应的兼容性规则库。

[0041] 随后来到步骤103。

[0042] 在步骤103中,对页面代码进行语法解析,获取页面代码的语法树。这里的语法分析是在词法分析的基础上将记号序列组合成各类语法短语,如“程序”、“语句”以及“表达式”等。可使用自上向下算符优先的算法实现的语法分析器进行语法分析。不同类型的页面代码使用相应的语法分析器对其进行语法分析,从而可获取各种页面代码的语法树。语法树的具体获取算法为本领域的现有算法,在这里不做详细介绍。

[0043] 随后来到步骤104。

[0044] 在步骤104中,根据兼容性规则库以及语法树对相应的页面代码进行静态测试。其中相应的兼容性规则库在步骤102中获取,页面代码的语法树在步骤103中获取。

[0045] 具体的兼容性规则库中的判断规则可以为：

[0046] 如有多个正则表达式进行兼容性判断,正则表达式之间可使用“或”逻辑、“与”逻辑或“非与”逻辑等。其中“或”逻辑是指多个正则表达式中有至少一个不匹配,则认为兼容性有问题,即具有兼容性的页面代码需要匹配所有的正则表达式。“与”逻辑是指多个正则表达式全部不匹配,才认为兼容性有问题,即部分正则表达式匹配的页面代码也是具有兼容性的。“非与”逻辑是指多个正则表达式部分不匹配,则认为兼容性有问题,即全部正则表达式均匹配的页面代码或全部表达式均不匹配的页面代码是具有兼容性的。其中“非与”逻辑一般用在某段代码A可能在特定浏览器中不兼容,但是如网页代码中同时具有代码B可以克服该不兼容的问题,因此代码A和代码B同时出现,或代码A和代码B同时不出现的网页代码均是具有兼容性的。

[0047] 如使用单独的正则表达式进行兼容性判断,则在正则表达式中可使用包含判断、比较判断以及替换操作等。其中包含判断用于根据语法树判断相关代码的父子关系;比较判断用于判断相关代码中的某个属性值的大小;而替换操作用于使用上一个正则表达式的判断结果替换正则表达式中的某个变量,以便进行进一步的兼容性判断。

[0048] 当然具体的判断规则包括但不限于上面举例的这些,也可采用其他的判断规则,并可对规则进行添加、删除以及修改,因此判断规则的具体设定并不影响本发明的保护范围。在本步骤中可根据页面代码的语法树对页面代码中的具体代码进行分类,再使用兼容性规则库中相应的规则对页面代码进行静态的兼容性测试,这样可大大缩短页面代码静态兼容性测试的测试时间。

[0049] 随后来到步骤105。

[0050] 在步骤105中,测试系统将静态测试的结果输出,结果包括但不限于问题代码的

位置、原因以及对应的兼容性规则库中的相关判断规则。这样技术人员可以根据该输出结果对页面应用的非兼容部分进行修改，使之符合兼容性规则库中的相关判断规则，从而使该页面应用可兼容于各种浏览器。

[0051] 本发明还涉及一种浏览器兼容性测试装置，请参照图 3，图 3 为本发明的浏览器兼容性测试装置的优选实施例的结构示意图。该浏览器兼容性测试装置包括类型获取模块 31、规则库获取模块 32、语法树获取模块 33、测试模块 34 以及输出模块 35。类型获取模块 31 用于对网页应用的页面代码进行预处理，获取所述页面代码的类型；规则库获取模块 32 用于根据页面代码的类型获取相应的兼容性规则库；语法树获取模块 33 用于对页面代码进行语法解析，获取页面代码的语法树；测试模块 34 用于根据兼容性规则库以及语法树，对页面代码进行静态测试，从而实现对网页应用的浏览器兼容性测试；输出模块 35 用于将静态测试的结果输出。

[0052] 本优选实施例的浏览器兼容性测试装置使用时，首先类型获取模块 31 的词法分析单元对页面应用的页面代码进行词法分析，获取页面代码的记号序列；类型获取模块 31 的类型获取单元根据网页应用的页面代码的记号序列，获取页面代码的类型。具体为：获取所述记号序列的第一个记号；如第一个记号为“<”，则判断页面代码的类型为 HTML 类型；如第一个记号为“{”或“[”，则判断页面代码的类型为 JSON 类型；如第一个记号为“@”、“*”、“#”、“.”或“：”，则判断页面代码的类型为 CSS 类型；否则判断页面代码的类型为 JavaScript 类型。

[0053] 随后规则库获取模块 32 根据页面代码的类型获取相应的兼容性规则库；语法树获取模块 33 对页面代码进行语法解析，获取页面代码的语法树；然后测试模块 34 根据兼容性规则库以及语法树对相应的页面代码进行静态测试，其中兼容性规则库根据对页面代码进行兼容性判断的多个正则表达式和 / 或多个正则表达式之间的逻辑判断关系进行创建；最后输出模块 35 将静态测试的结果输出。

[0054] 本优选实施例的浏览器兼容性测试装置的具体工作原理与上述的浏览器兼容性测试方法的优选实施例中的描述相同或相似，具体可参见上述浏览器兼容性测试方法的优选实施例中的相关描述。

[0055] 下面使用一段网页应用的代码对本发明的浏览器兼容性测试方法及装置的具体使用流程进行说明。

[0056] 具体的网页代码如下：

[0057]

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=utf-8"/>
<style type="text/css">
body {
  color: blue;
}
h1 {
  font-size: 18px;
}
</style>
<script type="text/javascript">
window.onload=function(){
  var $header=document.getElementById("header");
  alert($header.currentStyle.fontWeight);
  alert($header.currentStyle.fontSize);
  alert($header.currentStyle.color);
```

[0058]

```
}
</script>
</head>
<body>
  <h1 id="header" style="color:red;">Header 1</h1>
</body>
</html>
```

[0059] 首先对该页面代码进行词法分析，获取页面代码的记号序列，其中的第一个记号为“<”，因此判断该页面代码的类型为 HTML 类型。随后根据页面代码的类型对页面代码进行语法分析，获取页面代码的语法树。由于 Firefox Chrome Safari 等浏览器不支持 currentStyle 语句，而 IE6IE7IE8 等浏览器不支持 getComputedStyle 语句；因此在兼容性规则库中对这两个语句进行“非与”逻辑判断，只有 currentStyle 语句和 getComputedStyle 语句同时出现(即同时支持各种浏览器)，或 currentStyle 语句和 getComputedStyle 语句同时不出现(即不存在上述两种语句)，该页面代码才不会出现浏览器兼容性问题。最后将静态测试结果输出。技术人员可以根据输出结果对浏览器兼容性问题进行相应的处理。

[0060] 本发明的浏览器兼容性测试方法及装置通过对网页应用的页面代码进行静态测试，实现浏览器的兼容性测试；由于不需要使用各种浏览器进行实际测试，因此整个测试过程操作简单，维护成本低。解决了现有的浏览器兼容性测试方法及装置操作复杂以及维护成本高的技术问题。

[0061] 综上所述，虽然本发明已以优选实施例揭露如上，但上述优选实施例并非用以限制本发明，本领域的普通技术人员，在不脱离本发明的精神和范围内，均可作各种更动与润饰，因此本发明的保护范围以权利要求界定的范围为准。

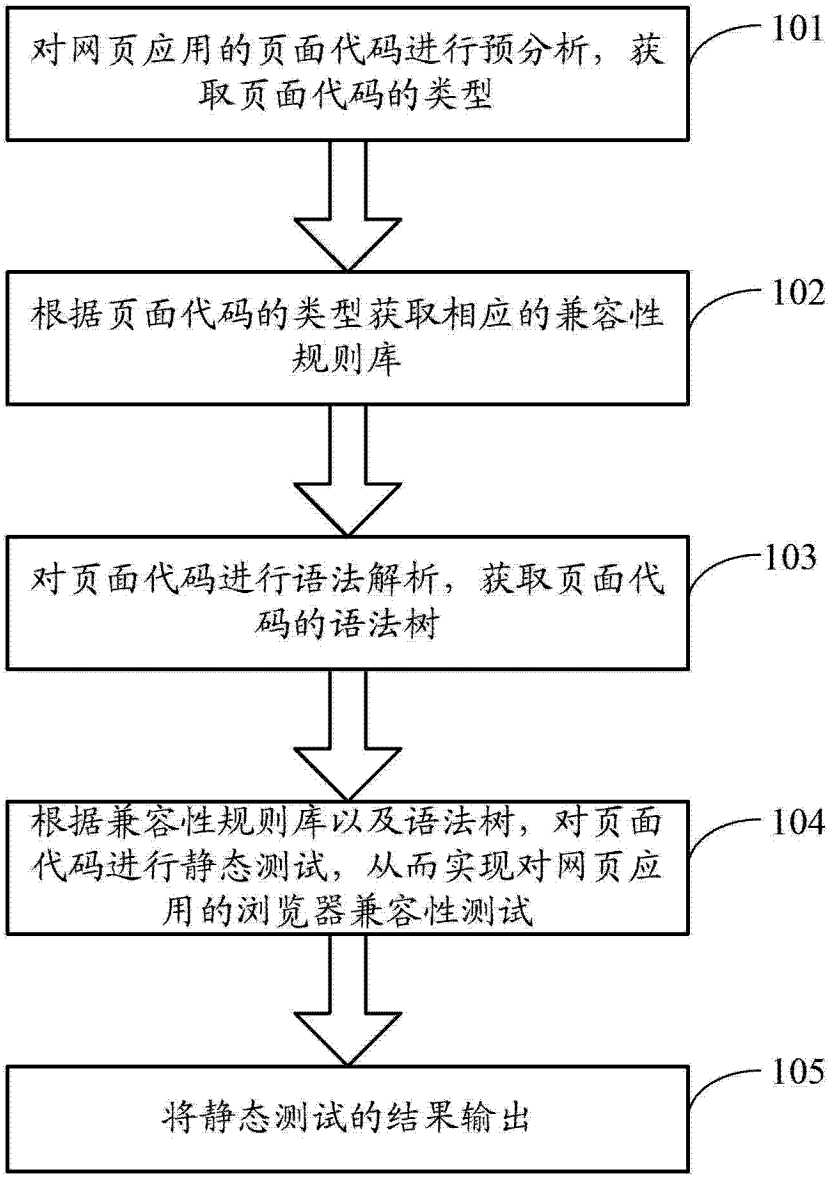

对网页应用的页面代码进行预分析，获取页面代码的类型
101
根据页面代码的类型获取相应的兼容性规则库
102
对页面代码进行语法解析，获取页面代码的语法树
103
根据兼容性规则库以及语法树，对页面代码进行静态测试，从而实现对网页应用的浏览器兼容性测试
104
将静态测试的结果输出
105

图 1

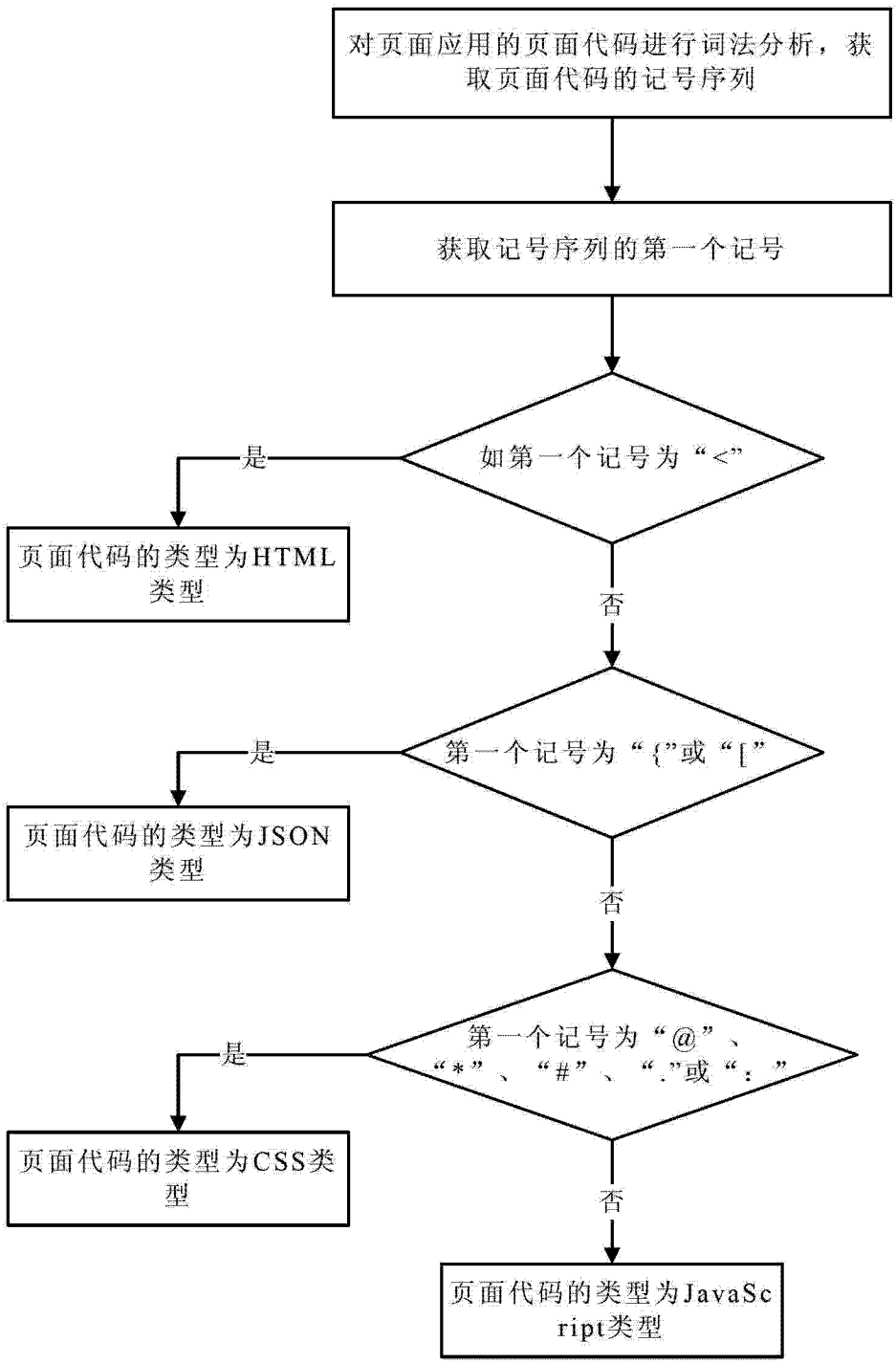

对页面应用的页面代码进行词法分析，获取页面代码的记号序列
获取记号序列的第一个记号
如第一个记号为“<”
是
页面代码的类型为HTML类型
否
第一个记号为“{”或“[”
是
页面代码的类型为JSON类型
否
第一个记号为“@”、“*”、“#”、“.”或“：”
是
页面代码的类型为CSS类型
否
页面代码的类型为JavaScript类型

图 2

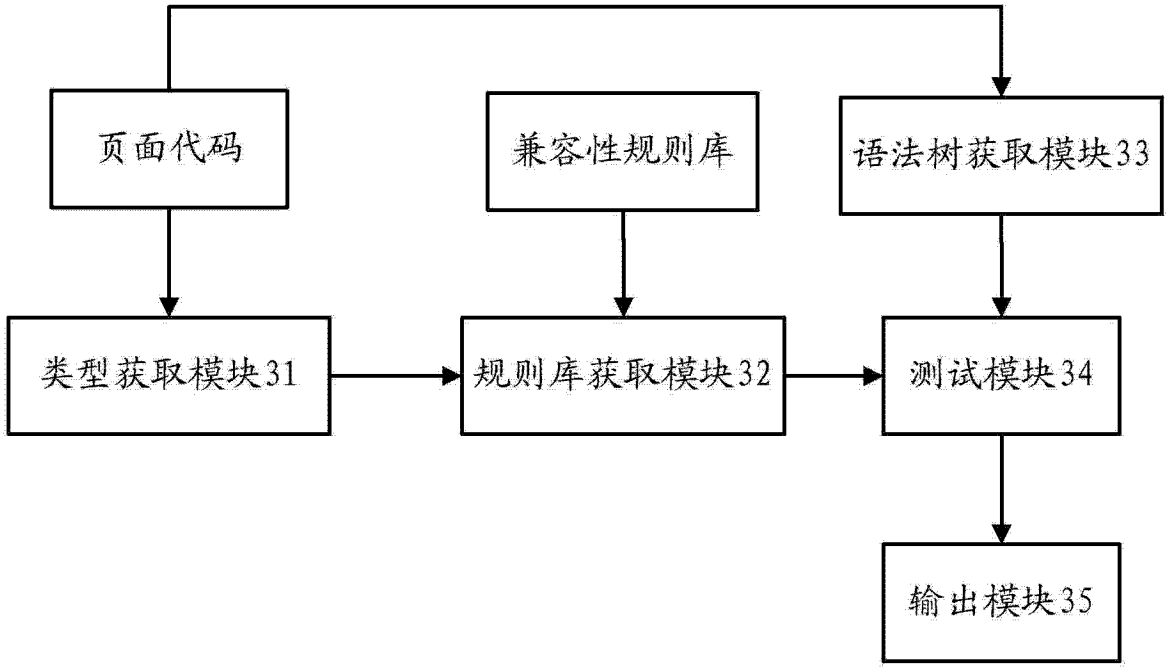


图 3