(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 111813385 A
(43)申请公布日 2020.10.23

(21)申请号 202010644014.2

(22)申请日 2020.07.07

(71)申请人 赞同科技股份有限公司
地址 200043 上海市杨浦区政高路11号701单元

(72)发明人 李宇翔 蔡宇

(74)专利代理机构 北京尚铖知识产权代理事务所(普通合伙) 11723
代理人 王海荣

(51)Int.Cl.
G06F 8/20(2018.01)
G06F 9/445(2018.01)

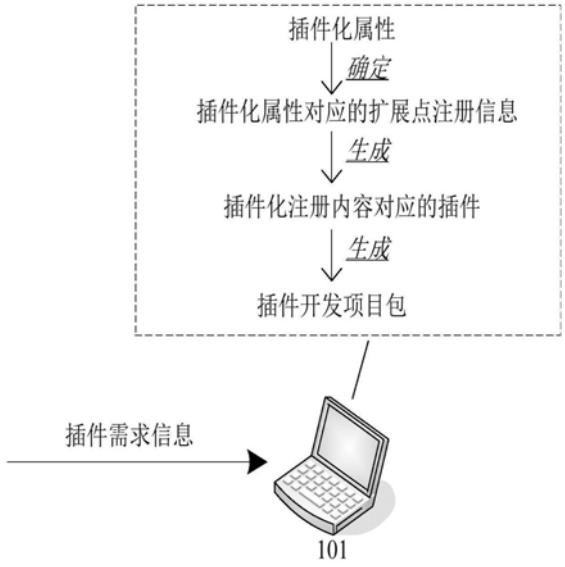
权利要求书2页 说明书13页 附图5页

(54)发明名称

一种基于Web应用的页面插件化方法、装置及设备

(57)摘要

本发明公开了一种基于Web应用的页面插件化方法，该方法在获取插件需求信息之后，可以先根据插件化属性，确定插件化属性对应的扩展点注册信息；然后，根据插件化注册内容以及扩展点注册信息，在插件化目标区域生成插件化注册内容对应的插件；接着，根据目标插件类型，在所生成的插件中确定目标插件，并根据目标插件生成插件开发项目包。可见，本申请可以实现在各个市场下均可以通过该插件开发项目包，使得web应用的整个界面上所有内容均可以实现插件化，从而实现一个更加轻量、可维护的Web应用的页面插件化架构方案，进而提高Web应用页面插件化的使用便捷性以及维护效率。



CN 111813385 A

1.一种基于Web应用的页面插件化方法,其特征在于,所述方法包括:

获取插件需求信息,其中,所述插件需求信息包括web应用页面中的待插件化目标区域、插件化属性、插件化注册内容以及目标插件类型;

根据所述插件化属性,确定所述插件化属性对应的扩展点注册信息;

根据所述插件化注册内容以及所述扩展点注册信息,在所述插件化目标区域生成所述插件化注册内容对应的插件;

根据所述目标插件类型,在所生成的插件中确定目标插件,并根据所述目标插件生成插件开发项目包;其中,所述目标插件的插件类型为所述目标插件类型。

2.根据权利要求1所述的方法,其特征在于,所述根据所述插件化属性,确定所述插件化属性对应的扩展点注册信息,包括:

根据所述插件化属性,在注册机中确定所述插件化属性对应的规范注册信息;

将所述规范注册信息作为所述插件化属性对应的扩展点注册信息。

3.根据权利要求2所述的方法,其特征在于,所述根据所述插件化属性,在注册机中确定所述插件化属性对应的规范注册信息,包括:

根据所述插件化属性,在所述注册机中确定扩展点注册关键字与所述插件化属性相同的规范注册信息。

4.根据权利要求3所述的方法,其特征在于,所述根据所述插件化属性,在所述注册机中确定扩展点注册关键字与所述插件化属性相同的规范注册信息,包括:

根据所述插件化属性,在注册机中利用注册应用程序接口,确定扩展点注册关键字与所述插件化属性相同的规范注册信息。

5.根据权利要求1所述的方法,其特征在于,所述插件化注册内容包括插件名称和插件化属性对应的插件化属性值;所述根据所述插件化注册内容以及所述扩展点注册信息,在所述插件化目标区域生成所述插件化注册内容对应的插件,包括:

根据所述插件化注册内容对所述扩展点注册信息进行调整,得到目标扩展点注册信息,其中,所述目标扩展点注册信息中的扩展点注册值为所述插件化属性值,所述目标扩展点注册信息的名称为所述插件名称;

将所述目标扩展点注册信息在所述注册机中进行注册,生成所述插件化注册内容对应的插件。

6.根据权利要求5所述的方法,其特征在于,所述插件化注册内容还包括第一注册业务范围,且扩展点注册关键字在第一注册业务范围内所对应的插件化属性值是唯一的;

相应地,所述目标扩展点注册信息还包括第一注册业务范围,且,所述目标扩展点注册信息中的第一注册业务范围、扩展点注册关键字和插件化属性值的适应形式为以下至少一种形式:三参数形式、双参数形式、key-value注册机形式。

7.根据权利要求5所述的方法,其特征在于,所述插件化注册内容还包括第二注册业务范围,且扩展点注册关键字在第二注册业务范围内所对应的插件化属性值为多个不同的插件化属性值;

相应地,所述目标扩展点注册信息还包括第二注册业务范围和合并标识,所述合并标识用于在利用获取注册应用程序接口,获取扩展点注册关键字在第二注册业务范围内所对应的插件化属性值时,将该扩展点注册关键字在第二注册业务范围内所对应的多个不同的

插件化属性值合并后输出。

8.根据权利要求5所述的方法，其特征在于，所述将所述目标扩展点注册信息在所述注册机中进行注册，生成所述插件化注册内容对应的插件，包括：

利用注册应用程序接口，将所述目标扩展点注册信息在所述注册机中进行注册，生成所述插件化注册内容对应的插件。

9.根据权利要求5所述的方法，其特征在于，所述根据目标插件类型，在所生成的插件中确定目标插件，并根据所述目标插件生成插件开发项目包，包括：

根据所述目标插件类型，在所生成的插件中确定目标插件；

在注册机中，获取所述目标插件对应的目标扩展点注册信息；

根据所述目标扩展点注册信息，确定所述目标插件对应的开发包；

将所述目标插件对应的开发包进行整合，生成插件开发项目包。

10.根据权利要求5或9所述的方法，其特征在于，所述方法还包括：

扫描所述插件开发项目包，获取目标插件加载清单；

加载所述目标插件加载清单中目标插件对应的开发包中的脚本，并在注册机注册目标插件对应的目标扩展点注册信息。

11.根据权利要求1-10中任一所述的方法，其特征在于，所述插件化注册内容对应的插件包括可用性校验函数；所述可用性校验函数用于判断该插件在所述待插件化目标区域中是否可用，若所述可用性校验函数为不可用，则该插件不在所述待插件化目标区域显示，若所述可用性校验函数为可用，则该插件在所述待插件化目标区域显示。

12.根据权利要求2-10中任一所述的方法，其特征在于，所述注册机包括注册应用程序接口和注册应用程序接口。

13.根据权利要求6或7所述的方法，其特征在于，所述第一注册业务范围包括以下至少一种注册业务范围：全局业务范围、指定业务范围；所述第二注册业务范围包括以下至少一种注册业务范围：全局业务范围、指定业务范围。

14.一种基于Web应用的页面插件化装置，其特征在于，所述装置包括：

获取模块，用于获取插件需求信息，其中，所述插件需求信息包括web应用页面中的待插件化目标区域、插件化属性、插件化注册内容以及目标插件类型；

确定模块，用于根据所述插件化属性，确定所述插件化属性对应的扩展点注册信息；

第一生成模块，用于根据所述插件化注册内容以及所述扩展点注册信息，在所述插件化目标区域生成所述插件化注册内容对应的插件；

第二生成模块，用于根据所述目标插件类型，在所生成的插件中确定目标插件，并根据所述目标插件生成插件开发项目包；其中，所述目标插件的插件类型为所述目标插件类型。

15.一种电子设备，其特征在于，包括处理器以及存储有执行指令的存储器，当所述处理器执行所述存储器存储的所述执行指令时，所述处理器执行如权利要求1-13中任一所述的方法。

一种基于Web应用的页面插件化方法、装置及设备

技术领域

[0001] 本发明涉及计算机技术领域，尤其涉及一种基于Web应用的页面插件化方法、装置及设备。

背景技术

[0002] 随着软件技术的发展和用户需求的不断提高，web应用越来越复杂，用户需求也越来越高。

[0003] 现有的前端架构体系中，为了实现一个web应用可以满足不同市场以及实现不同功能，通常采用以下方式：1、维护多份源码副本，这些源码已经完全脱离关系，几乎再无合并可能，需要大量人力维护；2、针对每个市场对web应用独立分支，每个分支需要修改的代码难以控制，随着web应用的版本更迭，各个分支的代码差异会越来越大，维护成本同样很高；3、如果不区分分支，所有市场在同一份源码下维护，靠权限进行隔离，那么会导致代码大量臃肿，尤其随着胖前端项目和单页项目的流行(单页项目：指在一个页面中完成几乎所有页面加载，浏览器的页面跳转不会重定位的项目)，web应用的前端项目的体积变得越来越大，导致加载速度的优化成为一大问题。

[0004] 故此，如何实现一个更加灵活、轻量、可维护的架构方案，来解决同一web应用可以在不同市场下的定制化问题是当前亟待解决的技术难点。

发明内容

[0005] 本发明提供一种基于Web应用的页面插件化方法及装置，以实现一个更加轻量、可维护的Web应用的页面插件化架构方案，进而提高Web应用页面插件化的使用便捷性以及维护效率。

[0006] 第一方面，本发明提供了一种基于Web应用的页面插件化方法，所述方法包括：

[0007] 获取插件需求信息，其中，所述插件需求信息包括web应用页面中的待插件化目标区域、插件化属性、插件化注册内容以及目标插件类型；

[0008] 根据所述插件化属性，确定所述插件化属性对应的扩展点注册信息；

[0009] 根据所述插件化注册内容以及所述扩展点注册信息，在所述插件化目标区域生成所述插件化注册内容对应的插件；

[0010] 根据所述目标插件类型，在所生成的插件中确定目标插件，并根据所述目标插件生成插件开发项目包；其中，所述目标插件的插件类型为所述目标插件类型。

[0011] 第二方面，本发明提供了一种基于Web应用的页面插件化装置，所述装置包括：

[0012] 获取模块，用于获取插件需求信息，其中，所述插件需求信息包括web应用页面中的待插件化目标区域、插件化属性、插件化注册内容以及目标插件类型；

[0013] 确定模块，用于根据所述插件化属性，确定所述插件化属性对应的扩展点注册信息；

[0014] 第一生成模块，用于根据所述插件化注册内容以及所述扩展点注册信息，在所述

插件化目标区域生成所述插件化注册内容对应的插件。

[0015] 第二生成模块，用于根据所述目标插件类型，在所生成的插件中确定目标插件，并根据所述目标插件生成插件开发项目包；其中，所述目标插件的插件类型为所述目标插件类型。

[0016] 第三方面，本发明提供了一种可读介质，包括执行指令，当电子设备的处理器执行所述执行指令时，所述电子设备执行如第一方面中任一所述的方法。

[0017] 第四方面，本发明提供了一种电子设备，包括处理器以及存储有执行指令的存储器，当所述处理器执行所述存储器存储的所述执行指令时，所述处理器执行如第一方面中任一所述的方法。

[0018] 由上述技术方案可以看出，在获取插件需求信息之后，由于所述插件需求信息包括web应用页面中的待插件化目标区域、插件化属性、插件化注册内容以及目标插件类型。故可以先根据所述插件化属性，确定所述插件化属性对应的扩展点注册信息；然后，根据所述插件化注册内容以及所述扩展点注册信息，在所述插件化目标区域生成所述插件化注册内容对应的插件；接着，根据所述目标插件类型，在所生成的插件中确定目标插件，并根据所述目标插件生成插件开发项目包。可见，本申请可以在插件开发完成后，在编译时根据预设的目标插件类型确定目标插件，并根据目标插件生成插件开发项目包，这样，可以实现在各个市场下均可以通过该插件开发项目包，使得web应用的整个界面上所有内容均可以实现插件化，从而实现了一个更加轻量、可维护的Web应用的页面插件化架构方案，进而提高了Web应用的页面插件化的使用便捷性以及维护效率。

[0019] 上述的非惯用的优选方式所具有的进一步效果将在下文中结合具体实施方式加以说明。

附图说明

[0020] 为了更清楚地说明本发明实施例或现有的技术方案，下面将对实施例或现有技术描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图仅仅是本发明中记载的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动性的前提下，还可以根据这些附图获得其他的附图。

[0021] 图1为本发明一实施例提供的一种示例性应用场景的框架示意图；

[0022] 图2为本发明一实施例提供的一种基于Web应用的页面插件化方法的流程示意图；

[0023] 图3为本发明一实施例提供的另一种示例性应用场景的框架示意图；

[0024] 图4为本发明一实施例提供的一种基于核心模块分别与原型插件集、开发插件集组合的示意图；

[0025] 图5为本发明一实施例提供的一种插件开发项目包生成方法的流程示意图

[0026] 图6为本发明一实施例提供的一种插件注册方法的流程示意图；

[0027] 图7为本发明一实施例提供的一种基于Web应用的页面插件化装置的结构示意图；

[0028] 图8为本发明一实施例提供的一种电子设备的结构示意图。

具体实施方式

[0029] 为使本发明的目的、技术方案和优点更加清楚，下面将结合具体实施例及相应的

附图对本发明的技术方案进行清楚、完整地描述。显然，所描述的实施例仅是本发明一部分实施例，而不是全部的实施例。基于本发明中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都属于本发明保护的范围。

[0030] 为了解决现有的前端架构体系为了实现一个web应用可以满足不同市场以及实现不同功能，会导致web应用的前端项目的体积变得越来越大，导致加载速度的优化成为一大问题。本发明提供了一种基于Web应用的页面插件化方法，所述方法在获取插件需求信息之后，由于所述插件需求信息包括web应用页面中的待插件化目标区域、插件化属性、插件化注册内容以及目标插件类型。故可以先根据所述插件化属性，确定所述插件化属性对应的扩展点注册信息；然后，根据所述插件化注册内容以及所述扩展点注册信息，在所述插件化目标区域生成所述插件化注册内容对应的插件；接着，根据所述目标插件类型，在所生成的插件中确定目标插件，并根据所述目标插件生成插件开发项目包。可见，本申请可以在插件开发完成后，在编译时根据预设的目标插件类型确定目标插件，并根据目标插件生成插件开发项目包，这样，可以实现在各个市场下均可以通过该插件开发项目包，使得web应用的整个界面上所有内容均可以实现插件化，从而实现了一个更加轻量、可维护的Web应用的页面插件化架构方案，进而提高了Web应用的页面插件化的使用便捷性以及维护效率。

[0031] 举例说明，本发明实施例可以应用到如图1所示的场景。在该场景中，包括终端101；其中，终端101可以为具有数据处理功能的终端，例如，可以为智能手机、平板电脑、台式电脑、笔记本电脑等终端设备，可以理解的是，终端101还可以为其他终端设备，例如服务器等，在本实施例中对此不进行限定。

[0032] 在本场景中，终端101可以先获取插件需求信息，其中，所述插件需求信息包括web应用页面中的待插件化目标区域、插件化属性、插件化注册内容以及目标插件类型；然后，终端101可以根据所述插件化属性，确定所述插件化属性对应的扩展点注册信息；接着，终端101可以根据所述插件化注册内容以及所述扩展点注册信息，在所述插件化目标区域生成所述插件化注册内容对应的插件；接着，终端101可以根据所述目标插件类型，在所生成的插件中确定目标插件，并根据所述目标插件生成插件开发项目包；其中，所述目标插件的插件类型为所述目标插件类型。可见，终端101可以在插件开发完成后，在编译时根据预设的目标插件类型确定目标插件，并根据目标插件生成插件开发项目包，这样，可以实现在各个市场下均可以通过该插件开发项目包，使得web应用的整个界面上所有内容均可以实现插件化，从而实现了一个更加轻量、可维护的Web应用的页面插件化架构方案，进而提高了Web应用的页面插件化的使用便捷性以及维护效率。

[0033] 可以理解的是，在上述应用场景中，虽然将本发明实施方式的部分动作描述为全部由终端101执行，但是本申请在执行主体方面不受限制，只要执行了本申请实施方式所公开的动作即可。

[0034] 需要注意的是，上述应用场景仅是为了便于理解本申请而示出，本申请的实施方式在此方面不受任何限制。相反，本申请的实施方式可以应用于适用的任何场景。

[0035] 下面结合附图，详细说明本发明的各种非限制性实施方式。

[0036] 参见图2，示出了本发明实施例中的一种基于Web应用的页面插件化方法，该方法可以用于web应用的前端项目中，例如可以是胖前端(即胖客户端)；需要说明的是，web应用的前端项目可以理解为在浏览器中运行的一类项目；胖前端可以理解为功能集中在前端

(即客户端)而非后台(即服务端,比如服务器)处理;插件化可以理解为把客户端构建成一插件化平台,其中,插件的版本的升级不依赖于核心平台(核心平台是指它的核心没有具体的业务功能,而是插件的载体),可以独立升级。在本实施例中,所述方法例如可以包括以下步骤:

[0037] S201:获取插件需求信息。

[0038] 在本实施例中,插件需求信息可以理解为用于反映web应用的页面中待设置的插件的相关信息,例如,在一种实现方式中,插件需求信息可以包括web应用页面中的待插件化目标区域、插件化属性、插件化注册内容以及目标插件类型。

[0039] 其中,web应用页面中的待插件化目标区域可以理解为web应用页面中用户可以进行操作的一区域,例如可以为页面中菜单对应的区域,也可以为页面中一按键对应的区域等。插件化属性可以理解为待设置的插件的属性信息,即待设置的插件所具有的功能,例如可以包括插件的界面展示属性(即插件在页面中所展示的形式),比如按钮形式、文字形式、图标形式、子菜单形式,或者包括插件所对应的跳转链接地址等。插件化注册内容可以理解为待设置的插件的具体设置信息,例如,可以包括插件名称、插件化属性对应的插件化属性值(插件化属性的具体内容)以及注册业务范围(scope)。目标插件类型可以理解为待设置的插件的类型,例如插件的类型可以为自定义语言支持、框架整合、工具整合、附加用户界面等。

[0040] 在一种实现方式中,用户可以通过输入设备(比如麦克风、键盘)以语音输入或者键盘输入方式,向终端输入插件需求信息,或者终端从服务器侧或者其他终端获取插件需求信息。接下来,举例说明其中一种具体的场景实现方式,假设用户需要在百度APP(即web应用)的主页面左上角的菜单增加能够跳转至微博的插件(即埋设扩展点),且该插件的名称为“toolbar.left”,该插件的插件化属性包括文本、链接地址,其中,插件化属性文本对应的具体文本内容为“我的微博”,插件化属性链接地址对应的具体连接地址为“www.weibo.com/my”;则用户可以通过键盘向终端输入插件需求信息,其中,该插件需求信息包括待插件化目标区域(即主页面左上角的菜单所对应的区域)、插件化属性(即包括文本、链接地址等属性)、插件化注册内容(即插件化属性文本对应的具体文本内容为“我的微博”、插件化属性链接地址对应的具体连接地址为“www.weibo.com/my”、该插件的名称为“toolbar.left”)以及目标插件类型(即为搜索挂接类型)。

[0041] S202:根据所述插件化属性,确定所述插件化属性对应的扩展点注册信息。

[0042] 在本实施例中,接收到插件需求信息后,可以先获取插件需求信息中的插件化属性。为生成该插件需求信息对应的待设置插件,需要先在注册机中完成该待设置插件的注册;具体地,可以先根据插件化属性,确定该插件化属性对应的扩展点注册信息。需要说明的是,扩展点注册信息可以理解为预先设置的插件化属性对应的规范模板,即插件化属性对应的代码结构模板,其中,为了便于描述,可以将该插件化属性对应的规范模板称之为规范注册信息。

[0043] 需要强调的是,规范注册信息可以包括扩展点注册关键字、插件化属性值和/或注册业务范围等字段,需要说明的是,插件化属性对应的规范注册信息中的扩展点注册关键字可以是预先设置的,即可以理解为固定的预设字段,而扩展点注册关键字对应的插件化属性值和注册业务范围可以均是根据实际需求进行调整的。举例来说,当插件化属性为文

本时，规范注册信息可以为text:string；当插件化属性为连接地址时，规范注册信息可以为href:string；若插件化属性为点击按键，插件化属性可以为run函数，即run:Function；若插件化属性为子菜单，插件化属性可以为children数组，即children:Array；若插件化属性为静态图标，插件化属性可以为icon属性，即icon:Function|String；若插件化属性为动态图标，插件化属性可以为icon函数，即icon:Function|String；若插件化属性为静态文字，插件化属性可以为label属性，即label:Function|String；若插件化属性为动态文字，插件化属性可以为label函数，即label:Function|String。

[0044] 需要说明的是，在一种实现方式中，根据所述插件化属性，确定所述插件化属性对应的扩展点注册信息的方式可以包括以下步骤：

[0045] S202a：根据所述插件化属性，在注册机中确定所述插件化属性对应的规范注册信息；

[0046] S202b：将所述规范注册信息作为所述插件化属性对应的扩展点注册信息。

[0047] 在本实施例中，各类插件化属性对应的规范注册信息可以均预先存储在注册机中，故获取到插件化属性后，可以根据插件化属性，在所述注册机中确定扩展点注册关键字与所述插件化属性相同的规范注册信息。需要说明的是，由于插件化属性对应的规范注册信息中的扩展点注册关键字是预先设置的固定预设字段，并且本实施例中的注册机只提供了两个应用程序接口api：注册应用程序接口regist和获取注册应用程序接口get；故此，可以先根据所述插件化属性，在注册机中利用注册应用程序接口，确定扩展点注册关键字与所述插件化属性相同的规范注册信息，接着，将所述规范注册信息作为所述插件化属性对应的扩展点注册信息。

[0048] S203：根据所述插件化注册内容以及所述扩展点注册信息，在所述插件化目标区域生成所述插件化注册内容对应的插件。

[0049] 获取到插件化属性对应的扩展点注册信息之后，可以根据插件化注册内容以及扩展点注册信息，在所述插件化目标区域生成该插件化注册内容对应的插件。在一种实现方式中，由于扩展点注册信息为预先设置的插件化属性对应的规范模板，故可以先根据所述插件化注册内容对所述扩展点注册信息进行调整，得到调整后的扩展点注册信息，需要说明的是，为了便于说明，在本实施例中将调整后的扩展点注册信息称之为目标扩展点注册信息；具体地，在本实现方式中，插件化注册内容可以包括插件名称和插件化属性对应的插件化属性值，故可以将扩展点注册信息中的插件化属性值调整为插件化注册内容中与所述插件化属性对应的插件化属性值，以及可以将扩展点注册信息的名称调整为插件化注册内容中的插件名称，即所述目标扩展点注册信息中的扩展点注册值为所述插件化属性值，所述目标扩展点注册信息的名称为所述插件名称。接着，可以将所述目标扩展点注册信息在所述注册机中进行注册，生成所述插件化注册内容对应的插件，即完成了插件的注册，此时可以在插件化目标区域显示该插件；作为一种示例，可以利用注册应用程序接口regist，将所述目标扩展点注册信息在所述注册机中进行注册，生成所述插件化注册内容对应的插件。

[0050] 接下来，举例说明其中一种具体的场景实现方式，假设该插件需求信息包括待插件化目标区域(即主页面左上角的菜单所对应的区域)、插件化属性(即包括文本、链接地址等属性)对应的扩展点注册信息为{text:string,href:string}、插件化注册内容(即插件

化属性文本对应的具体文本内容为“我的微博”、插件化属性链接地址对应的具体连接地址为“www.weibo.com/my”、该插件的名称为“toolbar.left”)以及目标插件类型(即为搜索挂接类型);此时,可以可以将扩展点注册信息中的插件化属性值调整为插件化注册内容中与所述插件化属性对应的插件化属性值,即将{text:string,href:string}调整为{text:“我的微博”,href:“www.weibo.com/my”},以及可以将扩展点注册信息的名称调整为插件化注册内容中的插件名称“toolbar.left”,从而得到了目标扩展点注册信息toolbar.left{text:“我的微博”,href:“www.weibo.com/my”};接着,利用注册应用程序接口regist,将所述目标扩展点注册信息toolbar.left{text:“我的微博”,href:“www.weibo.com/my”}在所述注册机中进行注册,并在主页面左上角的菜单所对应的区域中生成所述插件化注册内容对应的插件。

[0051] 紧接下来,再举例说明其中一种视图显示场景下的实现方式。如果是在视图上根据所述插件化注册内容以及所述扩展点注册信息,在所述插件化目标区域生成所述插件化注册内容对应的插件,需要完成两个关键步骤:1、获取插件化注册内容;2、应用数据在视图上,如果待加载插件是组件型的插件,则需要提供一个动态组件,以加载注册内容;示例1:获取注册内容,extTools(){return IDE.activator.get("v2sual.editor.tools");},示例2:应用在视图上时,

[0052] <i:class＝"item.icon"

[0053] @click＝"openRightToolPopover(item.key||item.id)":title＝"item.title||item.name"v-for＝"item in extTools":key＝"item.key||item.id"></i>。

[0054] 需要强调的是,由于扩展点的预埋需要考虑用户可能的位置期望,故可以使用方向后缀名组合作为扩展点注册信息的名称或插件化注册内容中的插件名称,例如:.before|.after|.prefix|.suffix|.append|.inner。

[0055] 需要说明的是,在工具类软件的开发过程中,在不同插件里,同一个目标扩展点注册信息的扩展点注册关键字(key)很可能是相同,比如原型工具的“项目名”和开发工具的“项目名”,可能都是projectName,但是它们的注册业务范围scope的值可以不同,比如prototype:projectName和development:projectName,同时,工具并不止原型和开发这两种,所以本实施例中,插件化注册内容还可以包括注册业务范围,相应地,目标扩展点注册信息中也可以还包括注册业务范围字段;其中,注册业务范围可以为全局业务范围、指定业务范围等方式,例如,全局业务范围可以利用一种默认值GLOABL:projectName进行表示,指定业务范围可以用special:projectNme进行表示,需要强调的是,若插件化注册内容和目标扩展点注册信息的扩展点注册关键字之前没有注册业务范围字段时,则默认此时注册业务范围为全局业务范围。举例来说,目标扩展点注册信息为Reg.regist(‘菜单’,[A,B,C])时,虽然“菜单”前没有注册业务范围字段,但相当于注册业务范围为全局业务范围,即Reg.regist(‘global:菜单’,[A,B,C]),即在全局scope下,注册了一个通用的菜单;然后,可以注册特殊菜单:Reg.regist(‘special:菜单’,[D,E,F]);由于代码量和普通的注册机是一致的,在获取扩展点注册关键字为“菜单”且注册业务范围为“special”的目标扩展点注册信息时,可以同时获取注册业务范围为“special”和“global”的目标扩展点注册信息,并且合并注册业务范围为“special”和“global”的目标扩展点注册信息,即Reg.get(“菜单”,[A,B,C,D,E,F]);而获取扩展点注册关键字为“菜单”且注册业务范围为“global”的目

标扩展点注册信息时，仅获取注册业务范围为“global”的目标扩展点注册信息，即Reg.get(“菜单”,[A,B,C])。基于此，本实施例提供了该注册机方案，以区分scope和key来获取注册机中的目标扩展点注册信息。

[0056] 在一种实施例中，所述插件化注册内容还可以包括第一注册业务范围，且扩展点注册关键字在第一注册业务范围内所对应的插件化属性值是唯一的。相应地，在该实施例中，所述目标扩展点注册信息还包括第一注册业务范围，且，所述目标扩展点注册信息中的第一注册业务范围、扩展点注册关键字和插件化属性值的适应形式为以下至少一种形式：三参数形式、双参数形式、key-value注册机形式；其中，三参数形式可以为Regist(scope,key,value)，双参数形式可以为Regist(‘scope:key’,value)，key-value注册机形式可以为Regist(key,value)。需要说明的是，本实施例中的第一注册业务范围包括以下至少一种注册业务范围：全局业务范围、指定业务范围。

[0057] 在一种实施例中，当用户尝试进行两次注册，比如：Reg(‘A’,[{name:1}])，Reg(‘A’,[{name:2}])，用户很可能是期望得到这个列表：[{name:1},{name:2}]，而现有技术中的注册时用第二个结果覆盖第一个，所以得到的结果是：[{name:1}]，这导致了实际注册的情况与用户期待的情况是不相同的，故此，为了解决该场景中所存在的结果覆盖问题，本实施例提供了一种快捷合并的方式，具体为：当插件化注册内容还包括第二注册业务范围，且扩展点注册关键字在第二注册业务范围内所对应的插件化属性值为多个不同的插件化属性值时；相应地，在该实施例中，所述目标扩展点注册信息还可以包括第二注册业务范围和合并标识(例如true)，其中，所述合并标识用于在利用获取注册应用程序接口，获取扩展点注册关键字在第二注册业务范围内所对应的插件化属性值时，将该扩展点注册关键字在第二注册业务范围内所对应的多个不同的插件化属性值合并后输出。举例来说，假设在全局业务范围内，扩展点注册关键字对应的目标扩展点注册信息为Reg(‘A’,[{name:1}])和Reg(‘A’,[{name:2}],true)，由于Reg(‘A’,[{name:2}],true)的最后一个参数merge(合并开关)为true，故会执行内容合并之后再进行输出，即输出[{name:1},{name:2}]。需要说明的是，本实施例中的第二注册业务范围包括以下至少一种注册业务范围：全局业务范围、指定业务范围。

[0058] 可见，本实施例的注册机模式的特点在于，实现了一种非常灵活的注册和获取方式，注册机仅包括了注册应用程序接口和注册应用程序接口，即只提供了两个api：regist和get，并且，插件注册机的设计也采用了注册业务范围与扩展点注册关键字的组合关键字的方式(即scope+key的组合关键字)，这样，本实施例便可以实现了灵活的使用方法，即通过两个接口(regist和get)便可以完成所有的业务，并且注册机的注册机制灵活(即采用了注册业务范围与扩展点注册关键字的组合关键字的方式)，可以更加适应插件化体系，进而也实现了一个更加灵活、轻量、可维护的Web应用的页面插件化架构方案，进而提高了Web应用的页面插件化的使用便捷性以及维护效率。

[0059] S204：根据所述目标插件类型，在所生成的插件中确定目标插件，并根据所述目标插件生成插件开发项目包；其中，所述目标插件的插件类型为所述目标插件类型。

[0060] 为了可以实现在各个市场(即在各种操作系统环境或者各种应用程序)下均可以通过该插件开发项目包，使得web应用的整个界面上所有内容均可以实现插件化，在本实施例中，可以将各个市场中均需要具有的核心模块在编译时便生成插件开发项目包，以便后

续在各种市场下均可以结合该市场下对应的插件集，以得到适用于该市场的插件工具。其中，各个市场中均需要具有的核心模块可以理解为各个页面需要实现插件化时均需要依赖的公共部分的模块组件；例如，在一种实现方式中，如图4所示，核心模块可以为编辑器扩展点、工具栏扩展点、总览扩展点、菜单扩展点和插件注册机(即注册机)，这样，将该核心模块进行打包生成插件开发项目包后，可以利用该插件开发项目包与各个市场下的插件集进行组合，比如与原型插件集结合可以得到原型工具，以适用A市场，又比如还可以与开发插件集结合得到开发工具，以适用B市场。

[0061] 作为一种示例，在生成插件化注册内容对应的插件之后，在编译的过程中，可以先根据所述待打包插件类型，在所生成的插件中确定目标插件，需要说明的是，该待打包插件类型可以是用户预先设定的，也可以为各个市场中均需要具有的核心插件，即各个页面需要实现插件化时均需要依赖的公共部分的插件。然后，可以在注册机中，获取所述目标插件对应的目标扩展点注册信息。接着，可以根据所述目标扩展点注册信息，确定所述目标插件对应的开发包，并将所述目标插件对应的开发包进行整合，生成插件开发项目包，举例说明，如图5所示，在一种方式中，由于WebIDE打包需要执行webide-deploy-cli，故发版者可以选择重新打包一个完整的WebIDE，并选择这个WebIDE需要包含哪些插件(即选择目标插件类型并根据目标插件类型确定插件)，并生成webide.deploy包，这个包就是用于生产环境的免安装包，即插件开发项目包，当然，发版者也可以分别打包单个插件，并将单个插件对应的包直接放入到已经发版的webide.deploy包中的extensions文件夹，从而得到插件开发项目包。

[0062] 接下来，将结合图3对应的应用场景的系统架构对S201-S204进行举例说。如图3所示，所述系统架构包括插件模块、注册机模块、扩展点模块和web应用页面模块。获取到插件需求信息之后，可以根据所述插件化属性，在注册机中确定所述插件化属性对应的扩展点注册信息，并根据所述插件化注册内容以及所述扩展点注册信息，在注册机中进行扩展点注册，以生成扩展点；接着，可以基于扩展点生成web应用页面对应的插件；根据所述目标插件类型，在插件模块中所生成的插件中确定目标插件，并根据所述目标插件生成插件开发项目包。

[0063] 由上述技术方案可以看出，在获取插件需求信息之后，由于所述插件需求信息包括web应用页面中的待插件化目标区域、插件化属性、插件化注册内容以及目标插件类型。故可以先根据所述插件化属性，确定所述插件化属性对应的扩展点注册信息；然后，根据所述插件化注册内容以及所述扩展点注册信息，在所述插件化目标区域生成所述插件化注册内容对应的插件；接着，根据所述目标插件类型，在所生成的插件中确定目标插件，并根据所述目标插件生成插件开发项目包。可见，本申请可以在插件开发完成后，在编译时根据预设的目标插件类型确定目标插件，并根据目标插件生成插件开发项目包，这样，可以实现在各个市场下均可以通过该插件开发项目包，使得web应用的整个界面上所有内容均可以实现插件化，从而实现了一个更加灵活、轻量、可维护的Web应用的页面插件化架构方案，进而提高了Web应用的页面插件化的使用便捷性以及维护效率。

[0064] 需要说明的是，在一种实现方式中，所述方法在S204之后还可以包括以下步骤：

[0065] 扫描所述插件开发项目包，获取目标插件加载清单；

[0066] 加载所述目标插件加载清单中目标插件对应的开发包中的脚本，并在注册机注册

目标插件对应的目标扩展点注册信息。

[0067] 在web应用获取到插件开发项目包之后,可以获取该插件开发项目包中包括哪些目标插件,从而获得该插件开发项目包对应的目标插件加载清单。然后,由于插件开发项目包中包括各个目标插件对应的开发包,故可以根据目标插件加载清单,分别针对该目标插件加载清单中的各个目标插件,加载各个目标插件各自分别对应的开发包中的脚本;接着可以通过运行各个目标插件各自分别对应的脚本,并在注册机注册目标插件对应的目标扩展点注册信息。

[0068] 接下来,将结合图6进行举例说明。在一种实现场景中,假设发版者可以分别打包单个插件,并将单个插件对应的包直接放入到已经发版的webide.deploy包中的extensions文件夹,从而得到插件开发项目包。相应地,WebIDE在启动时,会通过自带的nodejs服务,扫描一遍自身文件夹的extensions文件夹,根据extensions文件夹中各个插件对应的包,整理生成待加载插件的清单。WebIDE在浏览器访问时,会加载这份待加载插件清单中的各个插件清单,并动态加载各个插件的脚本,并注册对应的插件,这样WebIDE就能使用对应的功能了。

[0069] 可见,本申请可以在插件开发完成后,在编译时根据预设的目标插件类型确定目标插件,并根据目标插件生成插件开发项目包,这样,可以通过扫描所述插件开发项目包,获取目标插件加载清单,以及加载所述目标插件加载清单中目标插件对应的开发包中的脚本,并在注册机注册目标插件对应的目标扩展点注册信息,可以实现在各个市场(各个操作系统或者各种应用程序)下均可以通过该插件开发项目包,使得web应用的整个界面上所有内容均可以实现插件化,从而实现了一个更加灵活、轻量、可维护的Web应用的页面插件化架构方案,进而提高了Web应用的页面插件化的使用便捷性以及维护效率。

[0070] 需要强调的是,在一种实现场景中,即使插件为可用的,但也可能因为某些额外情况,不应当加载,比如额外的权限判断等等,故此,为了可以实现让插件判断自身的可用环境,并根据判断结果保证“不可用功能”的可用性,同时以减少了界面渲染量,在本实施例的一种实现方式中,所述插件化注册内容对应的插件可以包括可用性校验函数,例如,isEnable函数。所述可用性校验函数用于判断该插件在所述待插件化目标区域中是否可用,若所述可用性校验函数为不可用,则该插件不在所述待插件化目标区域显示,若所述可用性校验函数为可用,则该插件在所述待插件化目标区域显示。具体地,在一种实现方式中,插件需要向web应用提供其可用性校验函数,以便web应用可以根据页面的用户信息情况(比如是否登录、或者用户是否为VIP用户),向插件返回true/false,若返回的为false,说明所述可用性校验函数为不可用,则该插件不在所述待插件化目标区域显示,这样,当判断结果为“不可用功能”时,不需要针对该插件进行界面渲染,可以减少界面渲染量;若返回的为true,所述可用性校验函数为可用,则该插件在所述待插件化目标区域显示;需要强调的是,如果插件不具备该函数,默认意图为该扩展永远可用。可见,本实施例可以实现让插件判断自身的可用环境,并根据判断结果保证“不可用功能”的可用性,同时以减少了界面渲染量。

[0071] 需要强调的是,在本实施例中插件可以为视觉扩展,也可以为功能扩展,其中,视觉扩展是指在前端页面可见区域(即用户可以进行操作的区域)留下一个预埋点(扩展点),由未来的业务能力提供该预埋点的内容定制,功能扩展点是为了给不可见(即不在页面中

进行显示)的内部功能提供额外的能力,由平台预埋。接下来,将介绍本实施例中各种插件对应的扩展点的规范要求。

[0072] 一、插件为视觉扩展时的扩展点规范要求:

[0073] (1)、当扩展点在没有扩展接入时,展示必须符合用户预期,例如可以不在页面中进行显示;

[0074] (2)、某一个插件接入发生异常时,不会干涉到其他的插件;

[0075] (3)、如果视觉扩展点是一个序列,同一个扩展配置中需要加上优先序设定,优先序属性名默认为category,这样可以将优先级较高的插件的扩展配置进行设置,进而优先显示优先级较高的插件。

[0076] 二、插件为功能扩展时的扩展点规范要求:

[0077] (a)、在没有插件接入时,该扩展点所在的功能模块不能报错,并且可以正确执行功能流程;举例来说,假设一级菜单支持扩展插件,那么,埋设了扩展点之后,如果该扩展点使用者不提供任何的插件,这个部分代码不应该报错,也不应该展现成任何样式,可以是不再页面中进行显示;

[0078] (b)、某一个插件接入发生异常时,不会干涉到其他的插件;

[0079] (c)、不能在涉及全局数据的位置埋设运行时扩展插件;需要强调的是,不能在涉及全局数据的位置埋设运行时扩展插件的原因为:运行时扩展插件如果会影响到全局数据,那么插件的使用者无法估计这份程序代码数据会被哪些其他插件调用,如果另外一个运行时扩展插件在修改程序代码数据后被人为卸载了,对其他模块的影响不可评估;

[0080] (d)、插件在被注销时,能正确的析构。

[0081] 至此,本实施例结合具体的应用场景实现了基于Web应用的页面插件化方法过程。当然应该认为,上述场景仅仅为示例性场景,并不对本发明提供的方法构成限定。本发明提供的方法可延申的应用在其他相同原理的基于Web应用的页面插件化过程当中。

[0082] 如图7所示,为本发明所述基于Web应用的页面插件化装置的一个具体实施例。本实施例所述装置,即用于执行上述实施例所述方法的实体装置。其技术方案本质上与上述实施例一致,上述实施例中的相应描述同样适用于本实施例中。本实施例中所述装置包括:

[0083] 获取模块701,用于获取插件需求信息,其中,所述插件需求信息包括web应用页面中的待插件化目标区域、插件化属性、插件化注册内容以及目标插件类型;

[0084] 确定模块702,用于根据所述插件化属性,确定所述插件化属性对应的扩展点注册信息;

[0085] 第一生成模块703,用于根据所述插件化注册内容以及所述扩展点注册信息,在所述插件化目标区域生成所述插件化注册内容对应的插件。

[0086] 第二生成模块704,用于根据所述目标插件类型,在所生成的插件中确定目标插件,并根据所述目标插件生成插件开发项目包;其中,所述目标插件的插件类型为所述目标插件类型。

[0087] 可选的,所述确定模块702,具体用于:

[0088] 根据所述插件化属性,在注册机中确定所述插件化属性对应的规范注册信息;

[0089] 将所述规范注册信息作为所述插件化属性对应的扩展点注册信息。

[0090] 可选的,所述确定模块702,进一步具体用于:

[0091] 根据所述插件化属性，在所述注册机中确定扩展点注册关键字与所述插件化属性相同的规范注册信息。

[0092] 可选的，所述确定模块702，进一步具体用于：

[0093] 所述根据所述插件化属性，在所述注册机中确定扩展点注册关键字与所述插件化属性相同的规范注册信息，包括：

[0094] 根据所述插件化属性，在注册机中利用注册应用程序接口，确定扩展点注册关键字与所述插件化属性相同的规范注册信息。

[0095] 可选的，所述插件化注册内容包括插件名称和插件化属性对应的插件化属性值；所述第一生成模块703，具体用于：

[0096] 根据所述插件化注册内容对所述扩展点注册信息进行调整，得到目标扩展点注册信息，其中，所述目标扩展点注册信息中的扩展点注册值为所述插件化属性值，所述目标扩展点注册信息的名称为所述插件名称；

[0097] 将所述目标扩展点注册信息在所述注册机中进行注册，生成所述插件化注册内容对应的插件。

[0098] 可选的，所述插件化注册内容还包括第一注册业务范围，且扩展点注册关键字在第一注册业务范围内所对应的插件化属性值是唯一的；相应地，所述目标扩展点注册信息还包括第一注册业务范围，且，所述目标扩展点注册信息中的第一注册业务范围、扩展点注册关键字和插件化属性值的适应形式为以下至少一种形式：三参数形式、双参数形式、key-value注册机形式。

[0099] 可选的，所述插件化注册内容还包括第二注册业务范围，且扩展点注册关键字在第二注册业务范围内所对应的插件化属性值为多个不同的插件化属性值；相应地，所述目标扩展点注册信息还包括第二注册业务范围和合并标识，所述合并标识用于在利用获取注册应用程序接口，获取扩展点注册关键字在第二注册业务范围内所对应的插件化属性值时，将该扩展点注册关键字在第二注册业务范围内所对应的多个不同的插件化属性值合并后输出。

[0100] 可选的，所述第一生成模块703，还具体用于：

[0101] 利用注册应用程序接口，将所述目标扩展点注册信息在所述注册机中进行注册，生成所述插件化注册内容对应的插件。

[0102] 可选的，所述第二生成模块704，还具体用于：

[0103] 根据所述目标插件类型，在所生成的插件中确定目标插件；

[0104] 在注册机中，获取所述目标插件对应的目标扩展点注册信息；

[0105] 根据所述目标扩展点注册信息，确定所述目标插件对应的开发包；

[0106] 将所述目标插件对应的开发包进行整合，生成插件开发项目包。

[0107] 可选的，所述装置还包括加载模块，用于：

[0108] 扫描所述插件开发项目包，获取目标插件加载清单；

[0109] 加载所述目标插件加载清单中目标插件对应的开发包中的脚本，并在注册机注册目标插件对应的目标扩展点注册信息。

[0110] 可选的，所述插件化注册内容对应的插件包括可用性校验函数；所述可用性校验函数用于判断该插件在所述待插件化目标区域中是否可用，若所述可用性校验函数为不可

用,则该插件不在所述待插件化目标区域显示,若所述可用性校验函数为可用,则该插件在所述待插件化目标区域显示。

[0111] 可选的,所述注册机包括注册应用程序接口和注册应用程序接口。

[0112] 可选的,所述第一注册业务范围包括以下至少一种注册业务范围:全局业务范围、指定业务范围;所述第二注册业务范围包括以下至少一种注册业务范围:全局业务范围、指定业务范围。

[0113] 图8是本发明实施例提供的一种电子设备的结构示意图。在硬件层面,该电子设备包括处理器,可选地还包括内部总线、网络接口、存储器。其中,存储器可能包含内存,例如高速随机存取存储器(Random-Access Memory,RAM),也可能还包括非易失性存储器(non-volatile memory),例如至少1个磁盘存储器等。当然,该电子设备还可能包括其他业务所需要的硬件。

[0114] 处理器、网络接口和存储器可以通过内部总线相互连接,该内部总线可以是ISA(Industry Standard Architecture,工业标准体系结构)总线、PCI(Peripheral Component Interconnect,外设部件互连标准)总线或EISA(Extended Industry Standard Architecture,扩展工业标准结构)总线等。所述总线可以分为地址总线、数据总线、控制总线等。为便于表示,图8中仅用一个双向箭头表示,但并不表示仅有一根总线或一种类型的总线。

[0115] 存储器,用于存放执行指令。具体地,执行指令即可被执行的计算机程序。存储器可以包括内存和非易失性存储器,并向处理器提供执行指令和数据。

[0116] 在一种可能实现的方式中,处理器从非易失性存储器中读取对应的执行指令到内存中然后运行,也可从其它设备上获取相应的执行指令,以在逻辑层面上形成基于Web应用的页面插件化装置。处理器执行存储器所存放的执行指令,以通过执行的执行指令实现本发明任一实施例中提供的基于Web应用的页面插件化方法。

[0117] 上述如本发明图2所示实施例提供的基于Web应用的页面插件化装置执行的方法可以应用于处理器中,或者由处理器实现。处理器可能是一种集成电路芯片,具有信号的处理能力。在实现过程中,上述方法的各步骤可以通过处理器中的硬件的集成逻辑电路或者软件形式的指令完成。上述的处理器可以是通用处理器,包括中央处理器(Central Processing Unit,CPU)、网络处理器(Network Processor,NP)等;还可以是数字信号处理器(Digital Signal Processor,DSP)、专用集成电路(Application Specific Integrated Circuit,ASIC)、现场可编程门阵列(Field-Programmable Gate Array,FPGA)或者其他可编程逻辑器件、分立门或者晶体管逻辑器件、分立硬件组件。可以实现或者执行本发明实施例中的公开的各方法、步骤及逻辑框图。通用处理器可以是微处理器或者该处理器也可以是任何常规的处理器等。

[0118] 结合本发明实施例所公开的方法的步骤可以直接体现为硬件译码处理器执行完成,或者用译码处理器中的硬件及软件模块组合执行完成。软件模块可以位于随机存储器,闪存、只读存储器,可编程只读存储器或者电可擦写可编程存储器、寄存器等本领域成熟的存储介质中。该存储介质位于存储器,处理器读取存储器中的信息,结合其硬件完成上述方法的步骤。

[0119] 本发明实施例还提出了一种可读介质,该可读存储介质存储有执行指令,存储的

执行指令被电子设备的处理器执行时，能够使该电子设备执行本发明任一实施例中提供的基于Web应用的页面插件化方法，并具体用于执行上述基于Web应用的页面插件化所述的方法。

[0120] 前述各个实施例中所述的电子设备可以为计算机。

[0121] 本领域内的技术人员应明白，本发明的实施例可提供为方法或计算机程序产品。因此，本发明可采用完全硬件实施例、完全软件实施例，或软件和硬件相结合的形式。

[0122] 本发明中的各个实施例均采用递进的方式描述，各个实施例之间相同相似的部分互相参见即可，每个实施例重点说明的都是与其他实施例的不同之处。尤其，对于装置实施例而言，由于其基本相似于方法实施例，所以描述的比较简单，相关之处参见方法实施例的部分说明即可。

[0123] 还需要说明的是，术语“包括”、“包含”或者其任何其他变体意在涵盖非排他性的包含，从而使得包括一系列要素的过程、方法、商品或者设备不仅包括那些要素，而且还包括没有明确列出的其他要素，或者是还包括为这种过程、方法、商品或者设备所固有的要素。在没有更多限制的情况下，由语句“包括一个……”限定的要素，并不排除在包括所述要素的过程、方法、商品或者设备中还存在另外的相同要素。

[0124] 以上所述仅为本发明的实施例而已，并不用于限制本发明。对于本领域技术人员来说，本发明可以有各种更改和变化。凡在本发明的精神和原理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权利要求范围之内。

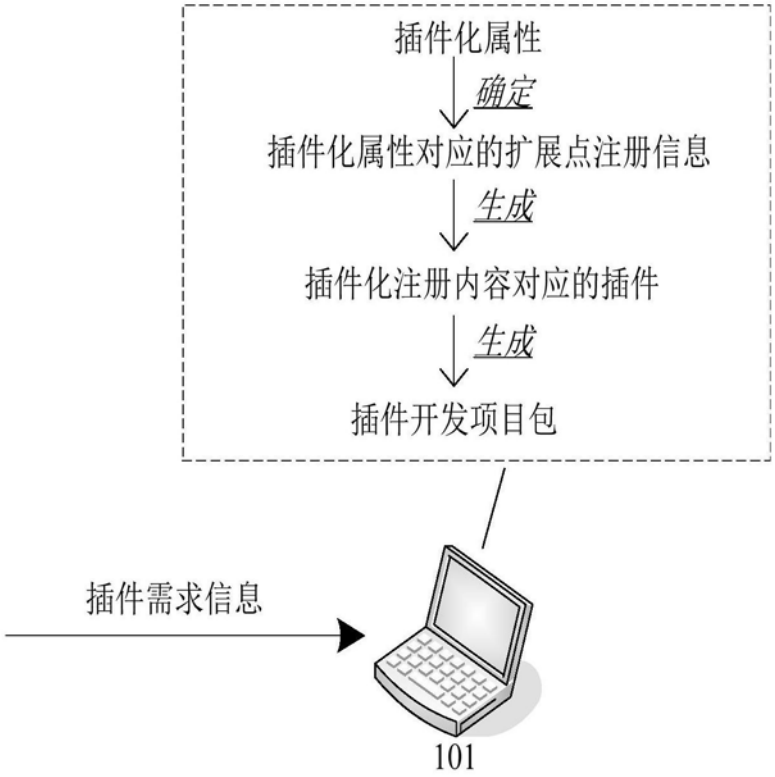

插件化属性
确定
插件化属性对应的扩展点注册信息
生成
插件化注册内容对应的插件
生成
插件开发项目包
插件需求信息
101

图1

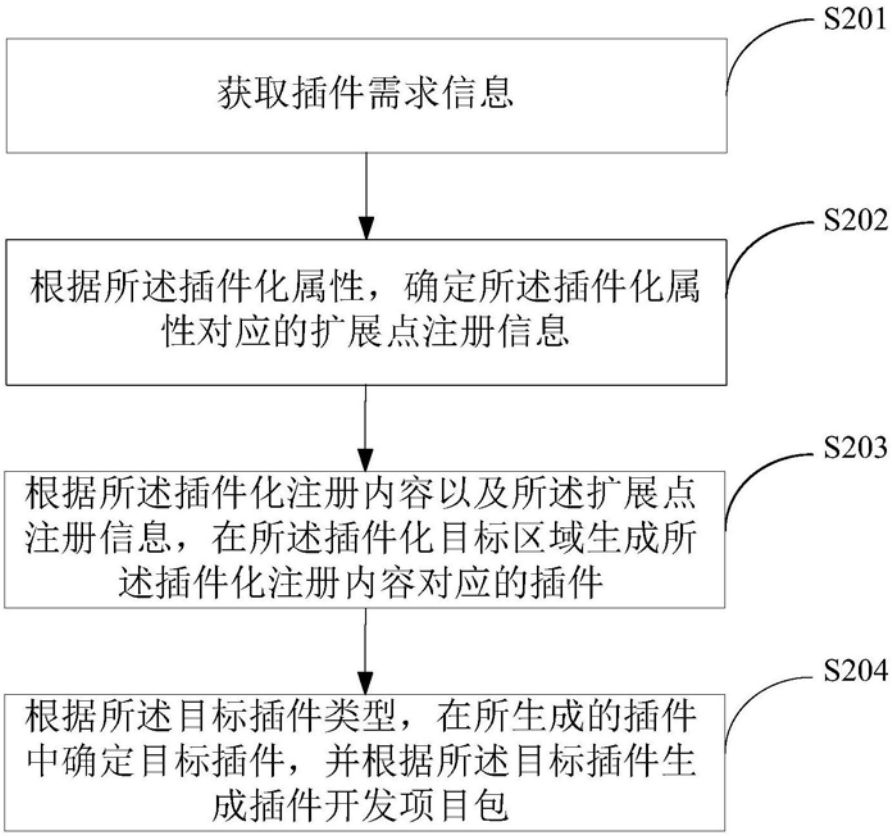

S201
获取插件需求信息
S202
根据所述插件化属性，确定所述插件化属性对应的扩展点注册信息
S203
根据所述插件化注册内容以及所述扩展点注册信息，在所述插件化目标区域生成所述插件化注册内容对应的插件
S204
根据所述目标插件类型，在所生成的插件中确定目标插件，并根据所述目标插件生成插件开发项目包

图2

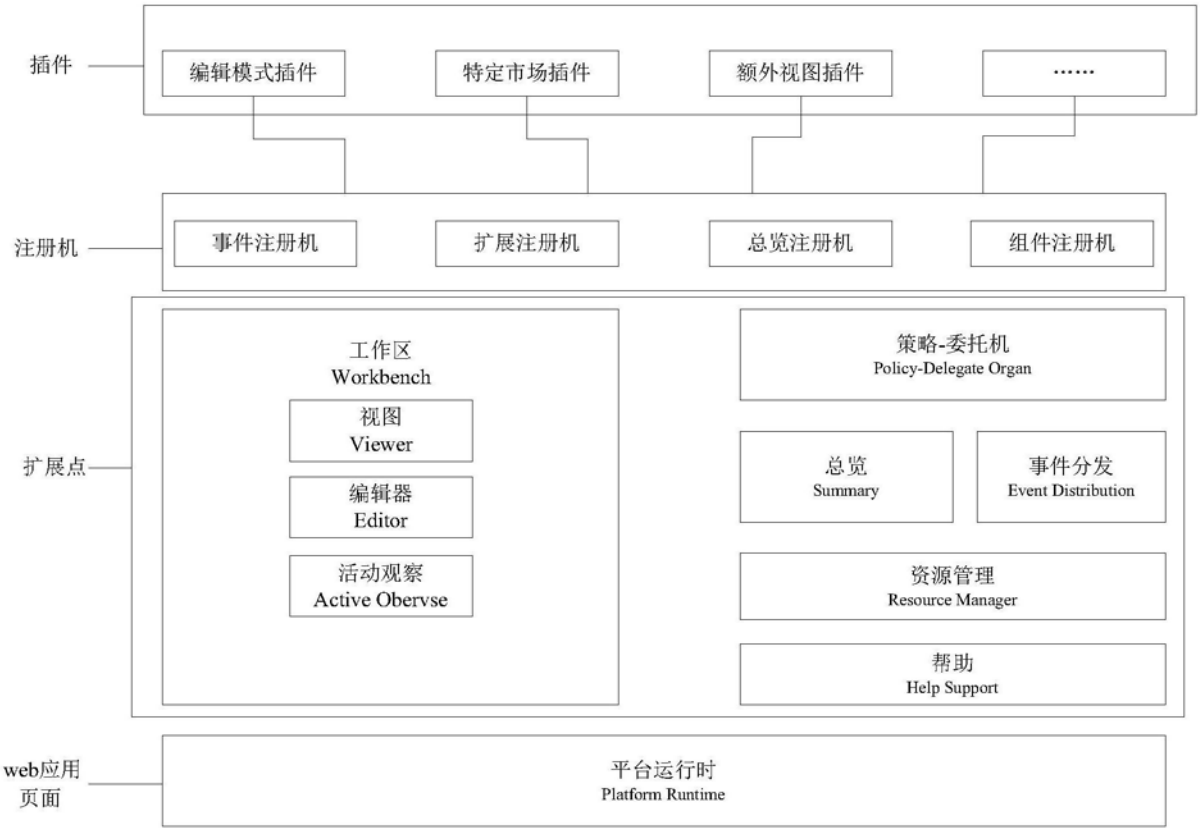

插件
编辑模式插件
特定市场插件
额外视图插件
……
注册机
事件注册机
扩展注册机
总览注册机
组件注册机
扩展点
工作区
Workbench
视图
Viewer
编辑器
Editor
活动观察
Active Obervse
策略-委托机
Policy-Delegate Organ
总览
Summary
事件分发
Event Distribution
资源管理
Resource Manager
帮助
Help Support
web应用
页面
平台运行时
Platform Runtime

图3

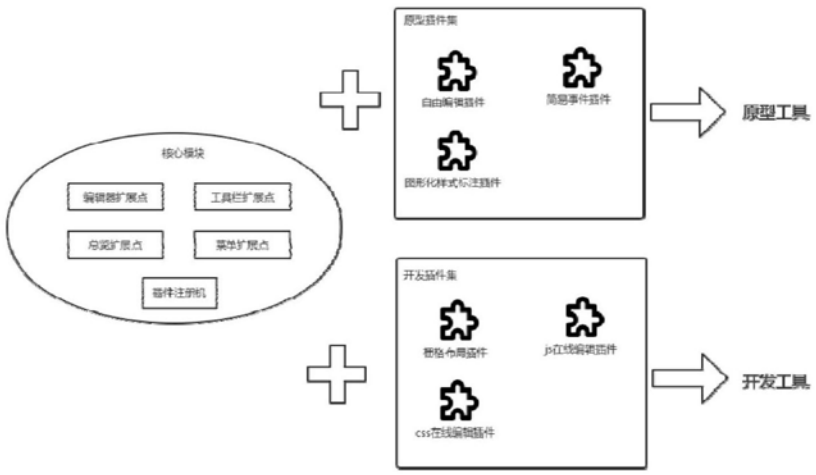

核心模块
编辑器扩展点
工具栏扩展点
总览扩展点
菜单扩展点
组件注册机
原型组件集
自由编辑器件
简易事件器件
图形化样式标注器件
原型工具
开发组件集
悬浮布局器件
js在线编辑器件
css在线编辑器件
开发工具

图4

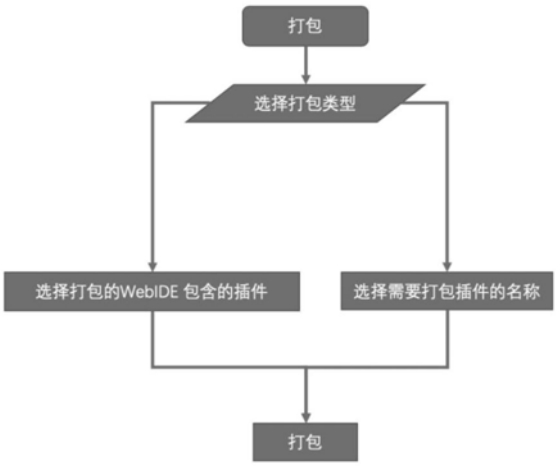

打包
选择打包类型
选择打包的WebIDE 包含的插件
选择需要打包插件的名称
打包

图5

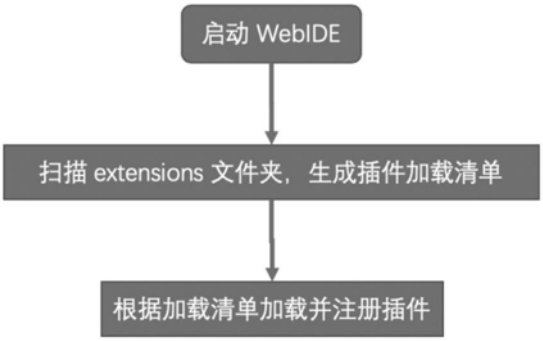

启动 WebIDE
扫描 extensions 文件夹，生成插件加载清单
根据加载清单加载并注册插件

图6

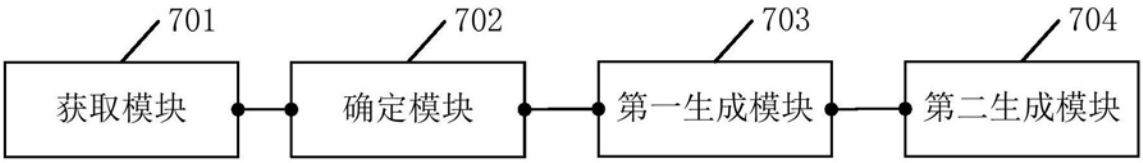

701
获取模块
702
确定模块
703
第一生成模块
704
第二生成模块

图7

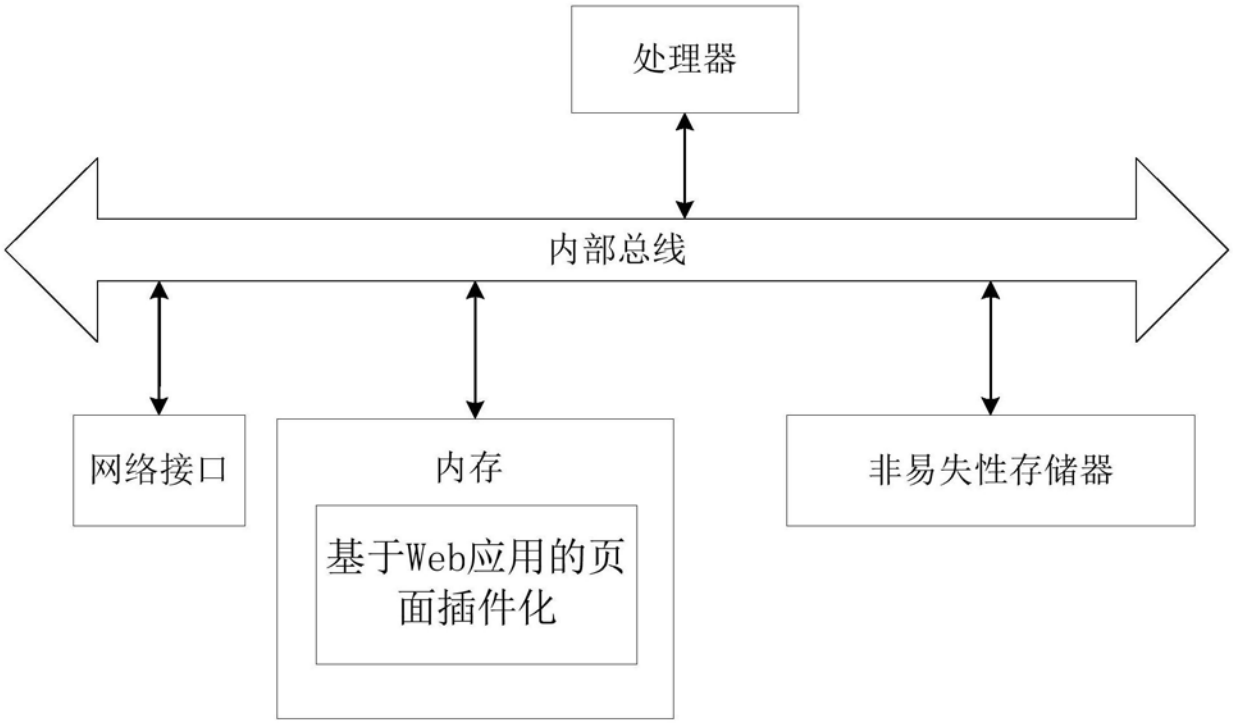


图8