(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 114546436 A
(43)申请公布日 2022.05.27

(21)申请号 202111563732.8

(22)申请日 2021.12.20

(71)申请人 北京达佳互联信息技术有限公司
地址 100085 北京市海淀区上地西路6号1幢1层101D1-7

(72)发明人 冯淼森

(74)专利代理机构 北京润泽恒知识产权代理有限公司 11319
专利代理师 李娜

(51)Int.Cl.
G06F 8/65(2018.01)
G06F 8/71(2018.01)

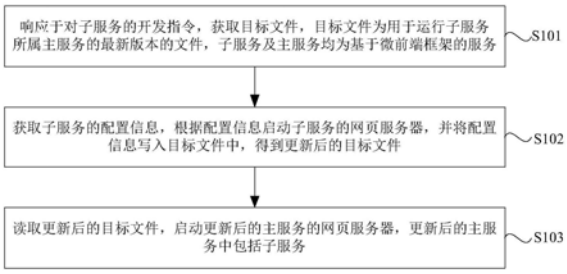
权利要求书2页 说明书10页 附图3页

(54)发明名称

一种微前端服务更新方法、装置、电子设备及存储介质

(57)摘要

本公开关于一种微前端服务更新方法、装置、电子设备及存储介质，包括：响应于对子服务的开发指令，获取目标文件，目标文件为子服务所属主服务的最新版本的目标文件，子服务及主服务均为基于微前端框架的服务；获取子服务的配置信息，根据配置信息启动子服务的网页服务器，并将配置信息写入目标文件中，得到更新后的目标文件；读取更新后的目标文件，启动更新后的主服务的网页服务器，更新后的主服务中包括子项。这样，在接收到开发指令之后，可以自动对主服务的目标文件进行配置，开发者只需要专注于微前端应用中子服务的开发配置以及业务开发，微前端服务可以实现自动更新，从而提高了微前端本地集成开发的效率。



CN 114546436 A

1.一种微前端服务更新方法,其特征在于,应用于电子设备,所述方法包括:

响应于对子服务的开发指令,获取目标文件,所述目标文件为用于运行所述子服务所属主服务的最新版本的文件,所述子服务及所述主服务均为基于微前端框架的服务;

获取所述子服务的配置信息,根据所述配置信息启动所述子服务的网页服务器,并将所述配置信息写入所述目标文件中,得到更新后的目标文件;

读取所述更新后的目标文件,启动更新后的主服务的网页服务器,所述更新后的主服务中包括所述子服务。

2.根据权利要求1所述的微前端服务更新方法,其特征在于,所述响应于对子服务的开发指令,获取目标文件,包括:

响应于对子服务的开发指令,从预设数据库获取所述主服务的目标文件的最新版本信息;

判断所述最新版本信息与本地存储的所述主服务的目标文件的版本信息是否一致;

若不一致,则从所述预设数据库获取所述主服务的目标文件,并对本地存储的所述主服务的目标文件进行更新。

3.根据权利要求2所述的微前端服务更新方法,其特征在于,若所述最新版本信息与本地存储的所述主服务的目标文件的版本信息一致,所述方法还包括:

获取本地存储的所述主服务的目标文件。

4.根据权利要求1所述的微前端服务更新方法,其特征在于,在所述获取目标文件之后,所述方法还包括:

从所述目标文件中获取依赖软件包的安装文件,根据所述安装文件下载并安装所述依赖软件包,所述依赖软件包为所述主服务所需的外部文件;

所述读取所述更新后的目标文件,启动更新后的主服务的网页服务器,包括:

读取所述更新后的目标文件,调用所述依赖软件包,启动更新后的主服务的网页服务器。

5.根据权利要求1所述的微前端服务更新方法,其特征在于,所述将所述配置信息写入所述目标文件中,得到更新后的目标文件,包括:

根据所述配置信息,生成所述子服务的模拟文件;

查询所述目标文件中是否包括所述子服务对应的模拟文件;

若包括,则将所述目标文件中的模拟文件替换为所生成的模拟文件,得到所述更新后的目标文件;

若不包括,则将所述模拟文件添加至所述目标文件中,得到所述更新后的目标文件。

6.根据权利要求5所述的微前端服务更新方法,其特征在于,所述读取所述更新后的目标文件,启动更新后的主服务的网页服务器,包括:

读取所述更新后的目标文件所包括的至少一个子服务的模拟文件,得到所述至少一个子服务的配置信息;

启动更新后的主服务的网页服务器,并根据所述至少一个子服务的配置信息,在所述更新后的主服务中对所述至少一个子服务进行注册。

7.根据权利要求1所述的微前端服务更新方法,其特征在于,所述获取所述子服务的配置信息,包括:

响应于所述开发指令，利用脚手架工具创建所述子服务的模板文件；

响应于对所述模板文件的编辑指令，获取所述子服务的配置信息。

8.一种微前端服务更新装置，其特征在于，应用于电子设备，所述装置包括：

获取单元，被配置为执行响应于对子服务的开发指令，获取目标文件，所述目标文件为用于运行所述子服务所属主服务的最新版本的文件，所述子服务及所述主服务均为基于微前端框架的服务；

更新单元，被配置为执行获取所述子服务的配置信息，根据所述配置信息启动所述子服务的网页服务器，并将所述配置信息写入所述目标文件中，得到更新后的目标文件；

启动单元，被配置为执行读取所述更新后的目标文件，启动更新后的主服务的网页服务器，所述更新后的主服务中包括所述子服务。

9.一种电子设备，其特征在于，包括：

处理器；

用于存储所述处理器可执行指令的存储器；

其中，所述处理器被配置为执行所述指令，以实现如权利要求1至7中任一项所述的微前端服务更新方法。

10.一种计算机可读存储介质，其特征在于，当所述计算机可读存储介质中的指令由微前端服务更新电子设备的处理器执行时，使得微前端服务更新电子设备能够执行如权利要求1至7中任一项所述的微前端服务更新方法。

一种微前端服务更新方法、装置、电子设备及存储介质

技术领域

[0001] 本公开涉及应用测试领域，尤其涉及一种微前端服务更新方法、装置、电子设备及存 储介质。

背景技术

[0002] 微前端应用是由多个前端应用聚合在一起组成的，其中，各个前端应用之间彼此独立， 互不干扰。对于用户而言，一个微前端应用就是一个完整的应用，但是微前端应用所包括 的多个前端应用在开发和测试过程中是互相独立的，然后再通过某种方式组合，得到微前 端应用。

[0003] 也就是说，在开发和测试过程中，微前端应用作为一个主服务，其中包括多个前端应 用分别对应的子服务，目前前端开发业内主流的微前端框架，能够实现子服务之间的技术 栈无关，支持子服务的独立开发、独立部署，以及子服务的运行时隔离。

[0004] 但在开发过程中，需要开发者在本地分别下载并启动主服务服务器和子服务服务器， 如果远程主服务代码仓库有更新，开发者并不能实时感知到，还需要手动更新，而且，还 需要开发者手动修改主服务和子服务的开发配置项，才能启动微前端开发模式。因此，整 体流程繁琐，微前端接入开发成本高，微前端服务的更新不够方便，开发体验和效率非常 的低下。

发明内容

[0005] 本公开提供一种微前端服务更新方法、装置、电子设备及存储介质，以至少解决相关 技术中整体流程繁琐，微前端接入开发成本高，微前端服务的更新不够方便，开发体验和 效率非常的低下的问题。本公开的技术方案如下：

[0006] 根据本公开实施例的第一方面，提供一种微前端服务更新方法，应用于电子设备，所 述方法包括：

[0007] 响应于对子服务的开发指令，获取目标文件，所述目标文件为用于运行所述子服务所 属主服务的最新版本的文件，所述子服务及所述主服务均为基于微前端框架的服务；

[0008] 获取所述子服务的配置信息，根据所述配置信息启动所述子服务的网页服务器，并将 所述配置信息写入所述目标文件中，得到更新后的目标文件；

[0009] 读取所述更新后的目标文件，启动更新后的主服务的网页服务器，所述更新后的主服 务中包括所述子服务。

[0010] 可选的，所述响应于对子服务的开发指令，获取目标文件，包括：

[0011] 响应于对子服务的开发指令，从预设数据库获取所述主服务的目标文件的最新版本信 息；

[0012] 判断所述最新版本信息与本地存储的所述主服务的目标文件的版本信息是否一致；

[0013] 若不一致，则从所述预设数据库获取所述主服务的目标文件，并对本地存储的所

述主 服务的目标文件进行更新。

[0014] 可选的，若所述最新版本信息与本地存储的所述主服务的目标文件的版本信息一致，所述方法还包括：

[0015] 获取本地存储的所述主服务的目标文件。

[0016] 可选的，在所述获取目标文件之后，所述方法还包括：

[0017] 从所述目标文件中获取依赖软件包的安装文件，根据所述安装文件下载并安装所述依 赖软件包，所述依赖软件包为所述主服务所需的外部目标文件；

[0018] 所述读取所述更新后的目标文件，启动更新后的主服务的网页服务器，包括：

[0019] 读取所述更新后的目标文件，调用所述依赖软件包，启动更新后的主服务的网页服务 器。

[0020] 可选的，所述将所述配置信息写入所述目标文件中，得到更新后的目标文件，包括：

[0021] 根据所述配置信息，生成所述子服务的模拟文件；

[0022] 查询所述目标文件中是否包括所述子服务对应的模拟文件；

[0023] 若包括，则将所述目标文件中的模拟文件替换为所生成的模拟文件，得到所述更新后 的目标文件；

[0024] 若不包括，则将所述模拟文件添加至所述目标文件中，得到所述更新后的目标文件。

[0025] 可选的，所述读取所述更新后的目标文件，启动更新后的主服务的网页服务器，包括：

[0026] 读取所述更新后的目标文件所包括的至少一个子服务的模拟文件，得到所述至少一个 子服务的配置信息；

[0027] 启动更新后的主服务的网页服务器，并根据所述至少一个子服务的配置信息，在所述 更新后的主服务中对所述至少一个子服务进行注册。

[0028] 可选的，所述获取所述子服务的配置信息，包括：

[0029] 响应于所述开发指令，利用脚手架工具创建所述子服务的模板文件；

[0030] 响应于对所述模板文件的编辑指令，获取所述子服务的配置信息。

[0031] 根据本公开实施例的第二方面，提供一种微前端服务更新装置，应用于电子设备，所 述装置包括：

[0032] 获取单元，被配置为执行响应于对子服务的开发指令，获取目标文件，所述目标文件 为用于运行所述子服务所属主服务的最新版本的文件，所述子服务及所述主服务均为基于 微前端框架的服务；

[0033] 更新单元，被配置为执行获取所述子服务的配置信息，根据所述配置信息启动所述子 服务的网页服务器，并将所述配置信息写入所述目标文件中，得到更新后的目标文件；

[0034] 启动单元，被配置为执行读取所述更新后的目标文件，启动更新后的主服务的网页服 务器，所述更新后的主服务中包括所述子服务。

[0035] 可选的，所述获取单元，被配置为执行：

[0036] 响应于对子服务的开发指令，从预设数据库获取所述主服务的目标文件的最新版

本信 息；

[0037] 判断所述最新版本信息与本地存储的所述主服务的目标文件的版本信息是否一致；

[0038] 若不一致，则从所述预设数据库获取所述主服务的目标文件，并对本地存储的所述主 服务的目标文件进行更新。

[0039] 可选的，所述获取单元，被配置为执行：

[0040] 若所述最新版本信息与本地存储的所述主服务的目标文件的版本信息一致，则获取本 地存储的所述主服务的目标文件。

[0041] 可选的，所述获取单元，被配置为执行：

[0042] 从所述目标文件中获取依赖软件包的安装文件，根据所述安装文件下载并安装所述依 赖软件包，所述依赖软件包为所述主服务所需的外部文件；

[0043] 所述启动单元，被配置为执行：

[0044] 读取所述更新后的目标文件，调用所述依赖软件包，启动更新后的主服务的网页服务 器。

[0045] 可选的，所述更新单元，被配置为执行：

[0046] 根据所述配置信息，生成所述子服务的模拟文件；

[0047] 查询所述目标文件中是否包括所述子服务对应的模拟文件；

[0048] 若包括，则将所述目标文件中的模拟文件替换为所生成的模拟文件，得到所述更新后 的目标文件；

[0049] 若不包括，则将所述模拟文件添加至所述目标文件中，得到所述更新后的目标文件。

[0050] 可选的，所述更新单元，被配置为执行：

[0051] 读取所述更新后的目标文件所包括的至少一个子服务的模拟文件，得到所述至少一个 子服务的配置信息；

[0052] 启动更新后的主服务的网页服务器，并根据所述至少一个子服务的配置信息，在所述 更新后的主服务中对所述至少一个子服务进行注册。

[0053] 可选的，所述更新单元，被配置为执行：

[0054] 响应于所述开发指令，利用脚手架工具创建所述子服务的模板文件；

[0055] 响应于对所述模板文件的编辑指令，获取所述子服务的配置信息。

[0056] 根据本公开实施例的第三方面，提供一种微前端服务更新电子设备，包括：

[0057] 处理器；

[0058] 用于存储所述处理器可执行指令的存储器；

[0059] 其中，所述处理器被配置为执行所述指令，以实现上述第一项所述的微前端服务更新 方法。

[0060] 根据本公开实施例的第四方面，提供一种计算机可读存储介质，当所述计算机可读存 储介质中的指令由微前端服务更新电子设备的处理器执行时，使得微前端服务更新电子设 备能够执行上述任一项所述的微前端服务更新方法。

[0061] 根据本公开实施例的第五方面，提供一种计算机程序产品，包括计算机程序/指令，所述计算机程序/指令被处理器执行时实现上述第一项所述的微前端服务更新方法。

[0062] 本公开的实施例提供的技术方案至少带来以下有益效果：

[0063] 电子设备响应于对子服务的开发指令，获取目标文件，目标文件为用于运行子服务所 属主服务的最新版本的文件；获取子服务的配置信息，根据配置信息启动子服务的网页服 务器，并将配置信息写入目标文件中，得到更新后的目标文件；读取更新后的目标文件， 启动更新后的主服务的网页服务器，更新后的主服务中包括子项。

[0064] 也就是说，在接收到开发指令之后，可以自动对主服务的目标文件进行配置，这样， 开发者只需要专注于微前端应用中子服务的开发配置以及业务开发，无需关心子服务对应 的主服务的代码如何拉取、更新、配置，微前端服务可以实现自动更新，从而提高了微前 端本地集成开发的效率。

[0065] 应当理解的是，以上的一般描述和后文的细节描述仅是示例性和解释性的，并不能限 制本公开。

附图说明

[0066] 此处的附图被并入说明书中并构成本说明书的一部分，示出了符合本公开的实施例， 并与说明书一起用于解释本公开的原理，并不构成对本公开的不当限定。

[0067] 图1是根据一示例性实施例示出的一种微前端服务更新方法的流程图。

[0068] 图2是根据一示例性实施例示出的一种图1中S101的流程图。

[0069] 图3是根据一示例性实施例示出的一种图1中S102的流程图。

[0070] 图4是根据一示例性实施例示出的一种微前端服务更新装置的框图。

[0071] 图5是根据一示例性实施例示出的一种用于微前端服务更新的电子设备的框图。

[0072] 图6是根据一示例性实施例示出的一种用于微前端服务更新的装置的框图。

具体实施方式

[0073] 为了使本领域普通人员更好地理解本公开的技术方案，下面将结合附图，对本公开实 施例中的技术方案进行清楚、完整地描述。

[0074] 需要说明的是，本公开的说明书和权利要求书及上述附图中的术语“第一”、“第二” 等是用于区别类似的对象，而不必用于描述特定的顺序或先后次序。应该理解这样使用的 数据在适当情况下可以互换，以便这里描述的本公开的实施例能够以除了在这里图示或描 述的那些以外的顺序实施。以下示例性实施例中所描述的实施方式并不代表与本公开相一 致的所有实施方式。相反，它们仅是与如所附权利要求书中所详述的、本公开的一些方面 相一致的装置和方法的例子。

[0075] 图1是根据一示例性实施例示出的一种微前端服务更新方法的流程图，如图1所示， 该微前端服务更新方法应用于电子设备，具体包括以下步骤。

[0076] 在步骤S101中，响应于对子服务的开发指令，获取目标文件，目标文件为用于运行子服务所属主服务的最新版本的文件，子服务及主服务均为基于微前端框架的服务。

[0077] 在本公开中，电子设备可以运行微前端应用，微前端应用是由多个前端应用聚合在一 起组成的，其中，各个前端应用之间彼此独立，互不干扰，在开发和测试过程中，微前端 应用作为一个主服务，其中包括多个前端应用分别对应的子服务，开发者可以分别对单独 的前端应用对应的子服务进行开发和测试。

[0078] 本步骤中，目标文件为用于运行子服务所属主服务的最新版本的文件，可以是代码文 件，也可以是编译后的指令文件，具体不做限定。一种实现方式中，可以直接从预设数据 库获取目标文件，另一种实现方式中，在本地存储主服务的目标文件的备份，在这种情况 下，可以比较本地存储主服务的目标文件与预设数据库中的目标文件的版本。

[0079] 具体来说，如图2所示，为一种实现方式中，S101的流程示意图，其中包括如下步骤：

[0080] S1011：响应于对子服务的开发指令，从预设数据库获取主服务的目标文件的最新版 本信息。

[0081] 其中，预设数据库可以为git数据库，在git数据库中存储主服务的目标文件。主服务的目标文件通常为package.json文件，目标文件的版本信息通常为package.json中的version字段，可以是数字组成的版本号，也可以是版本更新的日期，等等，具体不作限 定。

[0082] S1012：判断最新版本信息与本地存储的主服务的目标文件的版本信息是否一致，若 不一致，则执行S1013；若一致，则执行S1014。

[0083] S1013：从预设数据库获取主服务的目标文件，并对本地存储的主服务的目标文件进 行更新。这样，通过对目标文件的最新版本信息的一致性判断，可以确保获取到的目标文 件是最新版本的，同时也可以及时、主动地对本地存储的目标文件进行更新。

[0084] 一种实现方式中，对本地存储的主服务的目标文件进行更新，可以是根据本地存储版 本与预设数据库中版本之间的差异，对本地存储版本进行修改，或者，也可以是在最新版 本信息与本地存储的主服务的目标文件的版本信息不一致的情况下，删除本地存储的主服 务的目标文件，然后，重新从预设数据库下载并进行备份。

[0085] S1014：获取本地存储的主服务的目标文件，作为目标文件。这样，可以减少从预设数据库拉取数据的次数和数据量，减少系统资源的占用。

[0086] 这样，通过在本地存储主服务的目标文件的备份，可以减少从预设数据库获取目标文 件的次数，从而减少数据传输负载，提高效率，而且，可以实现自动拉取微前端主服务最 新代码，减少开发者所需的操作流程，提高开发效率。

[0087] 在本公开中，子服务和主服务之间采用的是微前端开发框架，如qiankun(乾坤)框架或single-spa(single page web application，单页应用程序)框架，等等，那么，需要预先在本地安装微前端开发脚手架工具，然后，才能接收到开发指令，其中，开发指令中可 以携带微前端开发脚手架工具中特定的命令函数以及传递参数，比如，开发者可以通过依 次输入如下内容，发送开发指令：“ksc serve—port 3000—force”。

[0088] 一种实现方式中，在获取目标文件之后，可以从目标文件中获取依赖软件包的安装文 件，根据安装文件下载并安装依赖软件包。其中，安装文件可以为脚本文件，也可以是普 通源码文件，具体不做限定，比如，安装文件可以为npm install，依赖软件包为主服务所 需的外部文件，通常在package.json文件dependencies和devDependencies字段，外部文 件可以为代码文件或编译后的指令文件，等等。

[0089] 这样，由于目标文件是用于运行主服务的最新版本的文件，因此，依赖软件包也是主 服务所需的最新的外部文件，在安装依赖软件包之后，可以读取更新后的目标文件，通过 调用依赖软件包，启动更新后的微前端主服务的网页服务器，主服务的网页服务器也就是 用于提供主服务的网页服务器。换句话说，可以实现对主服务的自动更新，提高微前端

本 地集成开发的效率。

[0090] 在步骤S102中,获取子服务的配置信息,根据配置信息启动子服务的网页服务器,并将配置信息写入目标文件中,得到更新后的目标文件。

[0091] 在本步骤中,根据开发人员的配置,确定子服务的配置信息,其中,子服务的配置信 息可以从配置文件.sub-app.js中读取,其中包括子服务的域名(host)、端口号(port)、服务名称(project name)、服务首页url(uniform resource locator,统一资源定位系统)地址等一项或多项信息。

[0092] 将配置信息写入目标文件中,具体可以为写入目标文件所包括的模拟文件中,模拟文 件即为mock文件,子服务的配置文件可以位于目标文件中该子服务的根目录下。在测试 过程中,对于某些不容易构造或者不容易获取的对象,创建一个对应的虚拟的mock文件, 对该mock文件进行测试,从而实现对这些不容易构造或者不容易获取的对象的测试。

[0093] 具体而言,获取子服务的配置信息,可以包括:首先,响应于开发指令,利用脚手架工具创建子服务的模板文件;然后,响应于对模板文件的编辑指令,获取子服务的配置信息。

[0094] 脚手架工具是一种可以自动创建服务模板文件的工具,模板文件中可以限定服务的基 础结构、提供服务的规范和约定等等信息,也就是说,基于脚手架工具创建的模板文件进 行子服务的配置,可以减少同一主服务中的重复开发工作,进一步实现微前端本地集成开 发调试的自动化。

[0095] 如图3所示,为一种实现方式中,S102的流程示意图,其中包括如下步骤:

[0096] S1021:根据配置信息,生成子服务的模拟文件。

[0097] 其中,模拟文件中可以包括对应的子服务的服务标识(project key)、服务名称、服 务首页url地址等信息,服务标识用于查询目标文件中是否包括子服务对应的模拟文件, 可以根据子服务的键值projectKey进行查询。

[0098] S1022:查询目标文件中是否包括子服务对应的模拟文件;若包括,则执行S1023;若不包括,则执行S1024。

[0099] S1023:将目标文件中的模拟文件替换为所生成的模拟文件,得到更新后的目标文件。

[0100] S1024:将模拟文件添加至目标文件中,得到更新后的目标文件。

[0101] 也就是说,若目标文件中已经存在同名子服务模拟文件,则直接覆盖同名子服务模拟 文件,若目标文件中不存在同名子服务模拟文件,则将模拟文件添加至目标文件中,这样, 可以根据子服务的最新的模拟文件启动运行子服务的网页服务器,子服务的网页服务器也 就是用于提供子服务的网页服务器,避免出现子服务的网页服务器启动后提供旧版本的子 服务的情况。

[0102] 在步骤S103中,读取更新后的目标文件,启动更新后的主服务的网页服务器,更新后的主服务中包括子服务。

[0103] 一种实现方式中,读取更新后的目标文件,启动更新后的主服务的网页服务器,具体 可以包括:

[0104] 首先,读取更新后的目标文件所包括的至少一个子服务的模拟文件,得到至少一个子 服务的配置信息;然后,启动更新后的主服务的网页服务器,并根据至少一个子服务

的配 置信息,在更新后的主服务中对至少一个子服务进行注册。

[0105] 其中,子服务的配置信息可以包括子服务的服务标识、服务名称以及服务首页url地 址等任意一项或多项。这样,通过读取目标文件中的模拟文件,电子设备无需完整的构造 或获取主服务的数据,就可以启动更新后的主服务的网页服务器,同时,新的子服务可以 自动注册到主服务中,也就实现了在主服务中自动对子服务进行更新,减少了开发人员的 手动操作。

[0106] 对目标文件进行更新后,可以读取更新后的目标文件,启动更新后的主服务的网页服 务器。由于更新后的目标文件中包括子服务的配置信息,因此,读取更新后的目标文件之 后,可以解析模拟文件中子服务的配置信息,进而将注册子服务列表到微前端框架当中, 从而实现自动启动微前端子服务的网页服务器,进而,可以对子服务进行进一步的开发和 测试。

[0107] 由以上可见,本公开的实施例提供的技术方案,在接收到开发指令之后,可以自动对 主服务的目标文件进行配置,这样,开发者只需要专注于微前端应用中子服务的开发配置 以及业务开发,无需关心子服务对应的主服务的代码如何拉取、更新、配置,微前端服务可以实现自动更新,从而提高了微前端本地集成开发的效率。

[0108] 图4是根据一示例性实施例示出的一种微前端服务更新装置框图,该装置应用于电子 设备,所述装置包括:

[0109] 获取单元201,被配置为执行响应于对子服务的开发指令,获取目标文件,所述目标 文件为用于运行所述子服务所属主服务的最新版本的文件,所述子服务及所述主服务均为 基于微前端框架的服务;

[0110] 更新单元202,被配置为执行获取所述子服务的配置信息,根据所述配置信息启动所 述子服务的网页服务器,并将所述配置信息写入所述目标文件中,得到更新后的目标文件;

[0111] 启动单元203,被配置为执行读取所述更新后的目标文件,启动更新后的主服务的网 页服务器,所述更新后的主服务中包括所述子服务。

[0112] 一种实现方式中,所述获取单元201,被配置为执行:

[0113] 响应于对子服务的开发指令,从预设数据库获取所述主服务的目标文件的最新版本信 息;

[0114] 判断所述最新版本信息与本地存储的所述主服务的目标文件的版本信息是否一致;

[0115] 若不一致,则从所述预设数据库获取所述主服务的目标文件,作为目标文件,并对本 地存储的所述主服务的目标文件进行更新。

[0116] 一种实现方式中,所述获取单元201,被配置为执行:

[0117] 若所述最新版本信息与本地存储的所述主服务的目标文件的版本信息一致,则获取本 地存储的所述主服务的目标文件。

[0118] 一种实现方式中,所述获取单元201,被配置为执行:

[0119] 从所述目标文件中获取依赖软件包的安装文件,根据所述安装文件下载并安装所述依 赖软件包,所述依赖软件包为所述主服务所需的外部文件;

[0120] 所述启动单元203,被配置为执行:

[0121] 读取所述更新后的目标文件，调用所述依赖软件包，启动更新后的主服务的网页服务 器。

[0122] 一种实现方式中，所述更新单元202，被配置为执行：

[0123] 根据所述配置信息，生成所述子服务的模拟文件；

[0124] 查询所述目标文件中是否包括所述子服务对应的模拟文件；

[0125] 若包括，则将所述目标文件中的模拟文件替换为所生成的模拟文件，得到更新后的目 标文件；

[0126] 若不包括，则将所述模拟文件添加至所述目标文件中，得到更新后的目标文件。

[0127] 一种实现方式中，所述更新单元202，被配置为执行：

[0128] 读取所述更新后的目标文件所包括的至少一个子服务的模拟文件，得到所述至少一个 子服务的配置信息；

[0129] 启动更新后的主服务的网页服务器，并根据所述至少一个子服务的配置信息，在所述 更新后的主服务中对所述至少一个子服务进行注册。

[0130] 一种实现方式中，所述更新单元202，被配置为执行：

[0131] 响应于所述开发指令，利用脚手架工具创建所述子服务的模板文件；

[0132] 响应于对所述模板文件的编辑指令，获取所述子服务的配置信息。

[0133] 由以上可见，本公开的实施例提供的技术方案，在接收到开发指令之后，可以自动对 主服务的目标文件进行配置，这样，开发者只需要专注于微前端应用中子服务的开发配置 以及业务开发，无需关心子服务对应的主服务的代码如何拉取、更新、配置，微前端服务可以实现自动更新，从而提高了微前端本地集成开发的效率。

[0134] 关于上述实施例中的装置，其中各个模块执行操作的具体方式已经在有关该方法的实 施例中进行了详细描述，此处将不做详细阐述说明。

[0135] 图5是根据一示例性实施例示出的一种用于微前端服务更新的电子设备的框图。

[0136] 在示例性实施例中，还提供了一种包括指令的计算机可读存储介质，例如包括指令的 存储器，上述指令可由电子设备的处理器执行以完成上述方法。可选地，计算机可读存储 介质可以是ROM、随机存取存储器(RAM)、CD-ROM、磁带、软盘和光数据存储设备 等。

[0137] 在示例性实施例中，还提供一种计算机程序产品，当其在计算机上运行时，使得计算 机实现上述微前端服务更新的方法。

[0138] 由以上可见，本公开的实施例提供的技术方案，在接收到开发指令之后，可以自动对 主服务的目标文件进行配置，这样，开发者只需要专注于微前端应用中子服务的开发配置 以及业务开发，无需关心子服务对应的主服务的代码如何拉取、更新、配置，从而提高了微前端本地集成开发的效率。

[0139] 图6是根据一示例性实施例示出的一种用于微前端服务更新的装置800的框图。

[0140] 例如，装置800可以是移动电话，计算机，数字广播电子设备，消息收发设备，游戏控制台，平板设备，医疗设备，健身设备，个人数字助理等。

[0141] 参照图6，装置800可以包括以下一个或多个组件：处理组件802，存储器804，电 力组件806，多媒体组件808，音频组件810，输入/输出(I/O)的接口812，传感器组件 814，以及通信组件816。

[0142] 处理组件802通常控制装置800的整体操作，诸如与显示，电话呼叫，数据通信，相

机操作和记录操作相关联的操作。处理组件802可以包括一个或多个处理器820来执行指令，以完成上述的方法的全部或部分步骤。此外，处理组件802可以包括一个或多个模块，便于处理组件802和其他组件之间的交互。例如，处理组件802可以包括多媒体模块，以 方便多媒体组件808和处理组件802之间的交互。

[0143] 存储器804被配置为存储各种类型的数据以支持在设备800的操作。这些数据的示例 包括用于在装置800上操作的任何应用程序或方法的指令，联系人数据，电话簿数据，消息，图片，视频等。存储器804可以由任何类型的易失性或非易失性存储设备或者它们的 组合实现，如静态随机存取存储器(SRAM)，电可擦除可编程只读存储器(EEPROM)， 可擦除可编程只读存储器(EPROM)，可编程只读存储器(PROM)，只读存储器(ROM)， 磁存储器，快闪存储器，磁盘或光盘。

[0144] 电源组件807为装置800的各种组件提供电力。电源组件807可以包括电源管理系统， 一个或多个电源，及其他与为装置800生成、管理和分配电力相关联的组件。

[0145] 多媒体组件808包括在所述装置800和用户之间的提供一个输出接口的屏幕。在一些 实施例中，屏幕可以包括液晶显示器(LCD)和触摸面板(TP)。如果屏幕包括触摸面 板，屏幕可以被实现为触摸屏，以接收来自用户的输入信号。触摸面板包括一个或多个触 摸传感器以感测触摸、滑动和触摸面板上的手势。所述触摸传感器可以不仅感测触摸或滑 动动作的边界，而且还检测与所述触摸或滑动操作相关的持续时间和压力。在一些实施例 中，多媒体组件808包括一个前置摄像头和/或后置摄像头。当设备800处于操作模式， 如拍摄模式或视频模式时，前置摄像头和/或后置摄像头可以接收外部的多媒体数据。每 个前置摄像头和后置摄像头可以是一个固定的光学透镜系统或具有焦距和光学变焦能力。

[0146] 音频组件810被配置为输出和/或输入音频信号。例如，音频组件810包括一个麦克风(MIC)，当装置800处于操作模式，如呼叫模式、记录模式和语音识别模式时，麦克 风被配置为接收外部音频信号。所接收的音频信号可以被进一步存储在存储器804或经由 通信组件816发送。在一些实施例中，音频组件810还包括一个扬声器，用于输出音频信 号。

[0147] I/O接口812为处理组件802和外围接口模块之间提供接口，上述外围接口模块可以 是键盘，点击轮，按钮等。这些按钮可包括但不限于：主页按钮、音量按钮、启动按钮和锁定按钮。

[0148] 传感器组件814包括一个或多个传感器，用于为装置800提供各个方面的状态评估。 例如，传感器组件814可以检测到设备800的打开/关闭状态，组件的相对定位，例如所述组件为装置800的显示器和小键盘，传感器组件814还可以检测装置800或装置800 一个组件的位置改变，用户与装置800接触的存在或不存在，装置800方位或加速/减速 和装置800的温度变化。传感器组件814可以包括接近传感器，被配置用来在没有任何的 物理接触时检测附近物体的存在。传感器组件814还可以包括光传感器，如CMOS或CCD 图像传感器，用于在成像应用中使用。在一些实施例中，该传感器组件814还可以包括加 速度传感器，陀螺仪传感器，磁传感器，压力传感器或温度传感器。

[0149] 通信组件816被配置为便于装置800和其他设备之间有线或无线方式的通信。装置800可以接入基于通信标准的无线网络，如WiFi，运营商网络(如2G、3G、4G或5G)， 或它们的组合。在一个示例性实施例中，通信组件816经由广播信道接收来自外部广播管 理系统的广播信号或广播相关信息。在一个示例性实施例中，所述通信组件816还包括近 场通信

(NFC)模块,以促进短程通信。例如,在NFC模块可基于射频识别(RFID)技 术,红外数据协会(IrDA)技术,超宽带(UWB)技术,蓝牙(BT)技术和其他技术来 实现。

[0150] 在示例性实施例中,装置800可以被一个或多个应用专用集成电路(ASIC)、数字信号处理器(DSP)、数字信号处理设备(DSPD)、可编程逻辑器件(PLD)、现场可 编程门阵列(FPGA)、控制器、微控制器、微处理器或其他电子元件实现,用于执行第 一方面和第二方面所述的方法。

[0151] 在示例性实施例中,还提供了一种包括指令的非临时性计算机可读存储介质,例如包 括指令的存储器804,上述指令可由装置800的处理器820执行以完成上述方法。可选地, 例如,存储介质可以是非临时性计算机可读存储介质,例如,所述非临时性非临时性计算 机可读存储介质计算机可读存储介质可以是ROM、随机存取存储器(RAM)、CD-ROM、 磁带、软盘和光数据存储设备等。

[0152] 在示例性实施例中,还提供了一种包含指令的计算机程序产品,当其在计算机上运行 时,使得计算机执行上述实施例中第一所述的微前端服务更新方法。

[0153] 由以上可见,本公开的实施例提供的技术方案,在接收到开发指令之后,可以自动对 主服务的目标文件进行配置,这样,开发者只需要专注于微前端应用中子服务的开发配置 以及业务开发,无需关心子服务对应的主服务的代码如何拉取、更新、配置,微前端服务可以实现自动更新,从而提高了微前端本地集成开发的效率。

[0154] 本领域技术人员在考虑说明书及实践这里公开的发明后,将容易想到本公开的其它实 施方案。本申请旨在涵盖本公开的任何变型、用途或者适应性变化,这些变型、用途或者 适应性变化遵循本公开的一般性原理并包括本公开未公开的本技术领域中的公知常识或 惯用技术手段。说明书和实施例仅被视为示例性的,本公开的真正范围和精神由下面的权 利要求指出。

[0155] 应当理解的是,本公开并不局限于上面已经描述并在附图中示出的精确结构,并且可 以在不脱离其范围进行各种修改和改变。本公开的范围仅由所附的权利要求来限制。

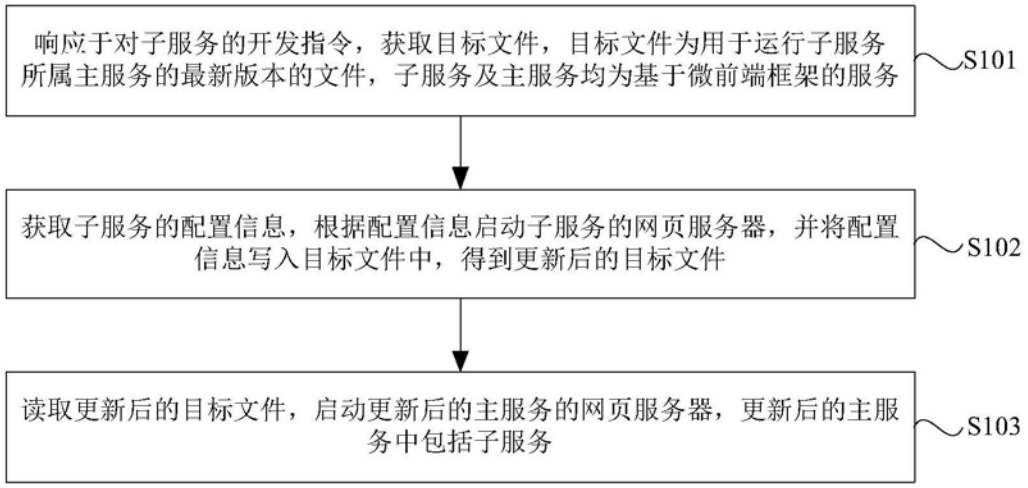

响应于对子服务的开发指令，获取目标文件，目标文件为用于运行子服务所属主服务的最新版本的文件，子服务及主服务均为基于微前端框架的服务
S101
获取子服务的配置信息，根据配置信息启动子服务的网页服务器，并将配置信息写入目标文件中，得到更新后的目标文件
S102
读取更新后的目标文件，启动更新后的主服务的网页服务器，更新后的主服务中包括子服务
S103

图1

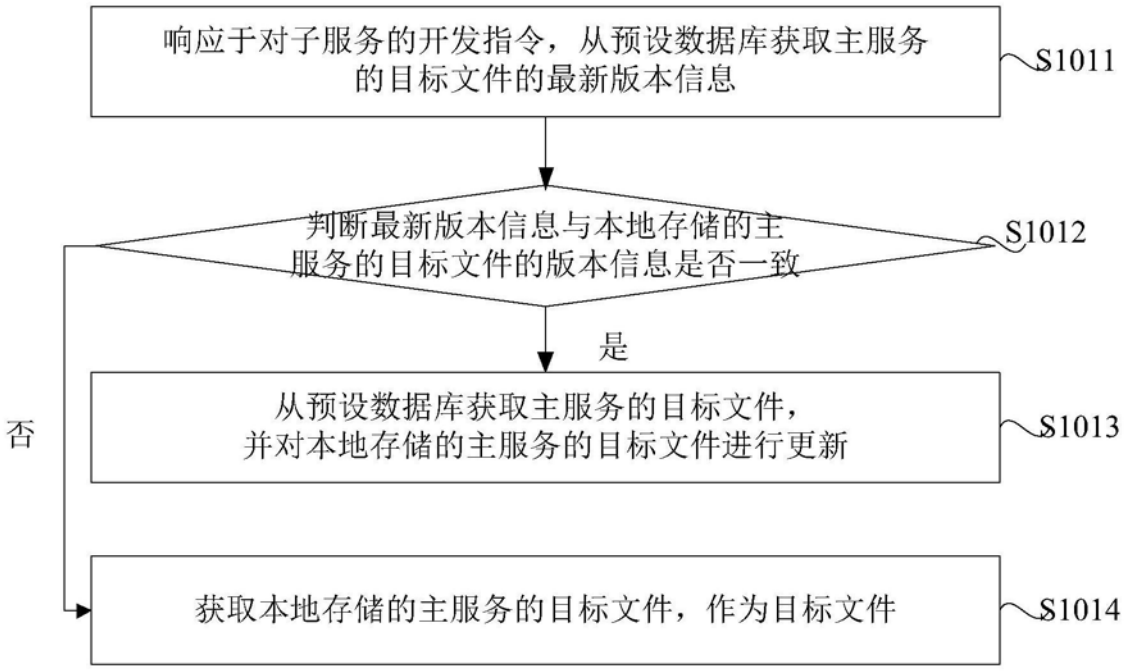

响应于对子服务的开发指令，从预设数据库获取主服务的目标文件的最新版本信息
S1011
判断最新版本信息与本地存储的主服务的目标文件的版本信息是否一致
S1012
是
否
从预设数据库获取主服务的目标文件，并对本地存储的主服务的目标文件进行更新
S1013
获取本地存储的主服务的目标文件，作为目标文件
S1014

图2

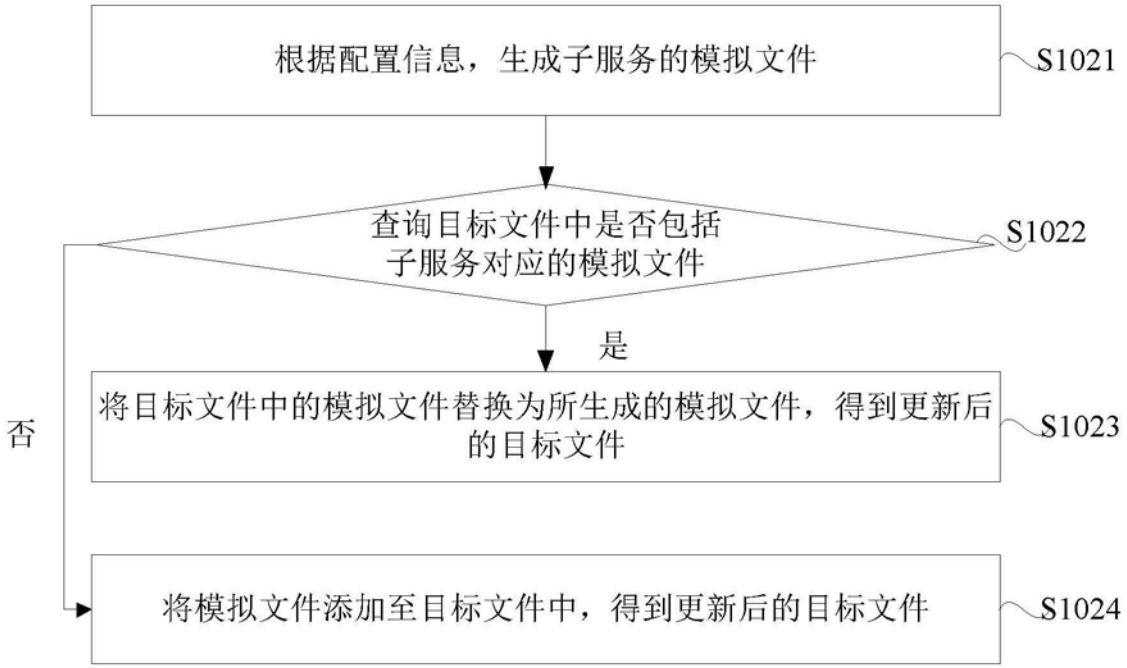

根据配置信息，生成子服务的模拟文件
S1021
查询目标文件中是否包括
子服务对应的模拟文件
S1022
是
否
将目标文件中的模拟文件替换为所生成的模拟文件，得到更新后的目标文件
S1023
将模拟文件添加至目标文件中，得到更新后的目标文件
S1024

图3

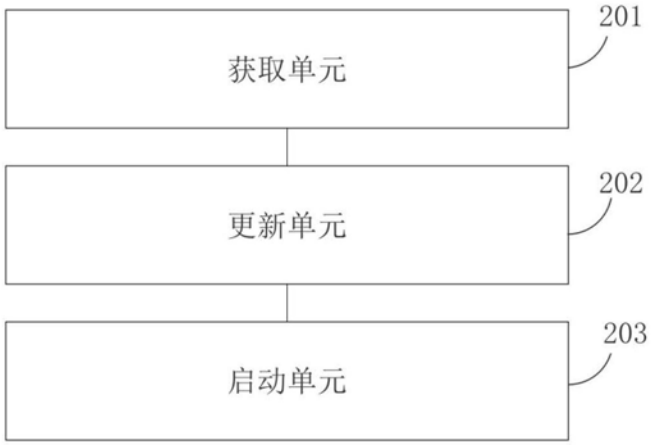

获取单元
201
更新单元
202
启动单元
203

图4

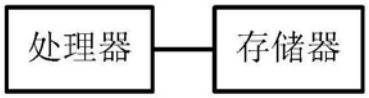

处理器
存储器

图5

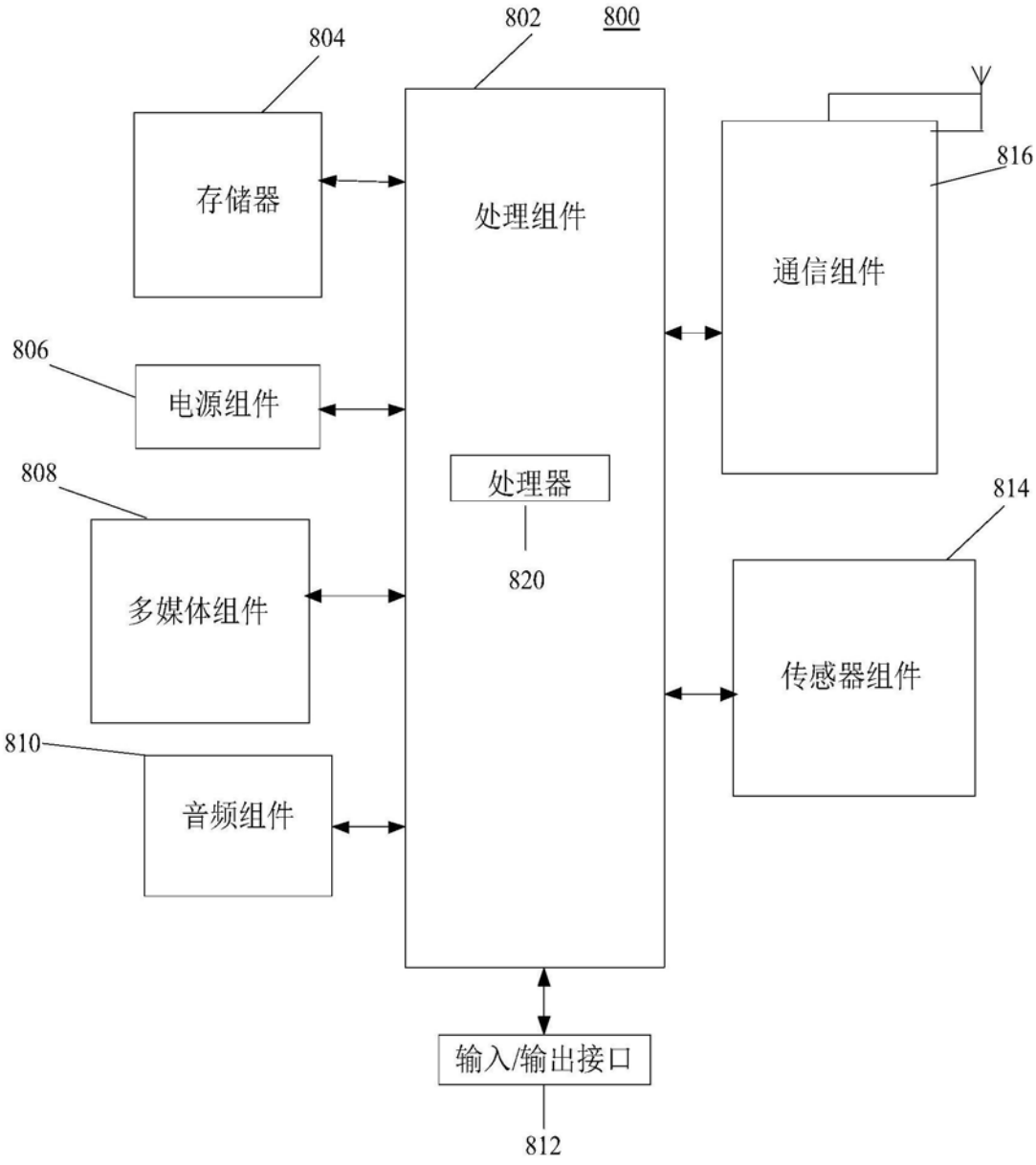

800
802
804
存储器
处理组件
处理器
820
806
电源组件
808
多媒体组件
810
音频组件
816
通信组件
814
传感器组件
输入/输出接口
812

图6