

子应用页面处理方法、装置、计算机设备和存储介质

相关申请

本申请要求 2022 年 11 月 25 日申请的，申请号为 2022114898700，名称为“子应用页面处理方法、装置、计算机设备和存储介质”的中国专利申请的优先权，在此将其全文引入作为参考。

技术领域

本申请涉及页面处理技术领域，特别是涉及一种子应用页面处理方法、装置、计算机设备、存储介质和计算机程序产品。

背景技术

随着计算机技术的飞速发展，页面处理技术也随之不断地发展，从原本只涉及到的 web（World Wide Web，全球广域网）端页面（即网页）处理，逐渐发展到其他非 web 端页面处理。比如，小程序的页面处理，小程序是一种不需要下载安装即可在母应用中运行使用的轻量级的应用程序。

传统方法中，小程序的页面处理方法为，在需要打开小程序的页面时，获取页面相应的页面代码，对页面代码进行解析，获得页面解析结果，基于页面解析结果调用 DOM（Document Object Model，文档对象模型）接口来执行绘制指令进行页面绘制，生成小程序的页面。

然而，传统方法中，调用 DOM 接口来执行绘制指令进行页面绘制的方式，由于 DOM 接口的组件过多，存在页面渲染速度慢的问题。

发明内容

根据本申请提供的各种实施例，提供一种子应用页面处理方法、装置、计算机设备、计算机可读存储介质和计算机程序产品。

一方面，本申请提供了一种子应用页面处理方法，由计算机设备执行，包括：

当监测到请求获取子应用页面的事件，获取所述子应用页面的页面代码，所述子应用页面是指在子应用中显示的页面；

对所述页面代码进行解析，获得页面解析结果，遍历所述页面解析结果中代码集合，创建子应用页面相应的页面节点树，所述页面节点树是基于所述子应用页面的页面节点而生成的树状结构；

创建用于转换所述页面节点树的第一实例对象，建立所述第一实例对象和所述子应用页面的绑定关系；

在建立所述绑定关系的情况下，使用所述第一实例对象，将所述页面节点树转换为接口调用的形式，获得所述页面节点树相应的接口调用信息；及

基于所述接口调用信息进行页面绘制，生成所述子应用页面。

另一方面，本申请还提供了一种子应用页面处理装置，包括：

代码获取模块，用于当监测到请求获取子应用页面的事件，获取所述子应用页面的页面代码，所述子应用页面是指在子应用中显示的页面；

节点树生成模块，用于对所述页面代码进行解析，获得页面解析结果，遍历所述页面解析结果中代码集合，创建所述子应用页面相应的页面节点树，所述页面节点树是基于所述子应用页面的页面节点而生成的树状结构；

页面初始化模块，用于创建用于转换页面节点树的第一实例对象，建立所述第一实例对象和所述子应用页面的绑定关系；

节点树转换模块，用于在建立所述绑定关系的情况下，使用所述第一实例对象，将所

述页面节点树转换为接口调用的形式，获得所述页面节点树相应的接口调用信息；

页面绘制模块，用于基于所述接口调用信息进行页面绘制，生成所述子应用页面。

另一方面，本申请还提供了一种计算机设备，所述计算机设备包括存储器和处理器，所述存储器存储有计算机可读指令，所述处理器执行所述计算机可读指令时执行本申请各方法实施例的步骤。

另一方面，本申请还提供了一种计算机可读存储介质，所述计算机可读存储介质，其上存储有计算机可读指令，所述计算机可读指令被处理器执行时执行本申请各方法实施例的步骤。

另一方面，本申请还提供了一种计算机程序产品。所述计算机程序产品，包括计算机可读指令，该计算机可读指令被处理器执行时执行本申请各方法实施例的步骤。

本申请的一个或多个实施例的细节在下面的附图和描述中提出。本申请的其它特征和优点将从说明书、附图以及权利要求书变得明显。

附图说明

为了更清楚地说明本申请实施例或传统技术中的技术方案，下面将对实施例或传统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申请的实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据公开的附图获得其他的附图。

图1为一个实施例中子应用页面处理方法的应用环境图；

图2为一个实施例中子应用页面处理方法的流程示意图；

图3为一个实施例中将页面节点树转换为接口调用的形式的示意图；

图4为一个实施例中创建用于转换页面节点树的第一实例对象；

图5为另一个实施例中子应用页面处理方法的流程示意图；

图6为一个实施例中子应用页面从开发者代码编写到绘制显示的示意图；

图7为另一个实施例中子应用页面从开发者代码编写到绘制显示的示意图；

图8为一个实施例中实例的调用关系的示意图；

图9为又一个实施例中子应用页面处理方法的流程示意图；

图10为一个实施例中获取组件节点的第一调用信息的流程示意图；

图11为一个实施例中子应用页面处理装置的结构框图；

图12为一个实施例中计算机设备的内部结构图。

具体实施方式

下面将结合本申请实施例中的附图，对本申请实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例仅仅是本申请一部分实施例，而不是全部的实施例。基于本申请中的实施例，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，都属于本申请保护的范围。

本申请实施例提供的子应用页面处理方法，可以应用于如图1所示的应用环境中。其中，使用终端的用户102可以对终端104进行操作，终端104通过网络与服务器106进行通信。数据存储系统可以存储服务器106需要处理的数据。数据存储系统可以集成在服务器106上，也可以放在云上或其他服务器上。

用户102可以操作终端104获取子应用页面，当监测到请求获取子应用页面的事件，终端104从服务器106获取子应用页面的页面代码，子应用页面是指在子应用中显示的页面，对页面代码进行解析，获得页面解析结果，遍历页面解析结果中代码集合，创建子应用页面相应的页面节点树，页面节点树是基于子应用页面的页面节点而生成的树状结构，创建用于转换页面节点树的第一实例对象，建立第一实例对象和子应用页面的绑定关系，在建立绑定关系的情况下，使用第一实例对象，将页面节点树转换为接口调用的形式，获

得页面节点树相应的接口调用信息，基于接口调用信息进行页面绘制，生成子应用页面。

其中，终端 104 可以但不限于是各种台式计算机、笔记本电脑、智能手机、平板电脑、物联网设备和便携式可穿戴设备，物联网设备可为智能音箱、智能电视、智能空调、智能车载设备等。便携式可穿戴设备可为智能手表、智能手环、头戴设备等。服务器 106 可以用独立的服务器或者是多个服务器组成的服务器集群来实现。

在一个实施例中，如图 2 所示，提供了一种子应用页面处理方法，该方法可以由终端或服务器单独执行，也可以由终端和服务器协同执行。在本申请实施例中，以该方法应用于终端为例进行说明，包括以下步骤：

步骤 202，当监测到请求获取子应用页面的事件，获取子应用页面的页面代码，子应用页面是指在子应用中显示的页面。

其中，子应用是能够在母应用提供的环境中实现的应用程序。母应用是承载子应用的应用程序，为子应用的实现提供环境。母应用是原生应用程序。原生应用程序是可直接运行于操作系统的应用程序。子应用不用下载即可基于母应用运行，属于轻量级应用。子应用可以包括即时通信子应用、移动支付子应用和游戏子应用等多种场景下的子应用，只要其满足基于母应用运行的无需下载安装的轻量级应用这一特性，即可以称为子应用。

其中，请求获取子应用页面的事件用于获取子应用页面。比如，请求获取子应用页面的事件具体可以是指在进入子应用时请求获取子应用的初始页面的事件。又比如，请求获取子应用页面的事件具体可以是指在子应用显示有初始页面的情况下，请求获取其他页面的事件，即从初始页面切换显示其他页面的事件。

其中，页面代码是指符合子应用的页面代码结构、且表征子应用页面的代码。子应用的页面代码结构，是子应用所支持的页面代码的结构。即，满足子应用的页面代码结构，才能被子应用支持，从而才能在子应用上运行页面。比如，页面代码具体可以是指基于子应用所支持的标签语言生成的代码。举例说明，子应用所支持的标签语言可以为 WXML (WeiXin Markup Language)，WXML 是小程序框架设计的一套标签语言，结合基础组件、事件系统可以构建出页面的结构。

具体的，当监测到请求获取子应用页面的事件，终端会获取子应用页面的页面代码。在具体的应用中，在打开子应用的情况下，可以监测到请求获取子应用页面的事件，若终端内存储有子应用相应的子应用代码，可以直接从子应用代码中获取子应用页面的页面代码，若终端内未存储有子应用相应的子应用代码，则需要先从存储有子应用相应的子应用代码的服务器拉取子应用代码，再从子应用代码中获取子应用页面的页面代码。

在一个具体的应用中，若子应用页面相应的子应用为初次被打开，则终端内可能未存储有子应用相应的子应用代码，需要基于子应用的应用标识从存储有子应用相应的子应用代码的服务器拉取子应用代码。若子应用页面相应的子应用不是初次被打开，则终端内应该存储有子应用相应的子应用代码。

在一个具体的应用中，子应用代码包括可以在子应用上显示的所有子应用页面的页面代码，且每个子应用页面的页面代码都具有对应的页面标识，终端根据需要打开的子应用页面的页面标识在子应用代码中进行查询，即可获得子应用页面的页面代码。

步骤 204，对页面代码进行解析，获得页面解析结果，遍历页面解析结果中代码集合，创建子应用页面相应的页面节点树，页面节点树是基于子应用页面的页面节点而生成的树状结构。

其中，页面解析结果是指对页面代码进行解析后所得到的结果，页面代码通常是指基于子应用所支持的标签语言生成的代码，而终端内子应用的渲染层的运行环境通常无法直接理解页面代码，因此需要对页面代码进行解析，获得页面解析结果。比如，页面解析结果具体可以是指对页面代码进行解析后所得到的子应用的渲染层的运行环境可以直接理解的代码。举例说明，页面解析结果具体可以是指对页面代码进行解析后得到的 JavaScript 代码。JavaScript (简称“JS”) 是一种具有函数优先的轻量级，解释型或即时编译型的编

程语言。虽然它是作为开发 Web 页面的脚本语言而出名，但是它也被用到了很多非浏览器环境中，JavaScript 基于原型编程、多范式的动态脚本语言，并且支持面向对象、命令式、声明式、函数式编程范式。

其中，页面节点树是基于子应用页面的页面节点而生成的树状结构，是指将子应用页面的所有页面节点合成之后的树，即页面节点树是由子应用页面的页面节点拼接而成的，它包含子应用页面的所有页面节点的名称、属性值和事件回调函数等信息。页面节点是指在子应用页面中出现的节点，子应用页面的页面内容都可以通过页面节点来表示。可以理解的是，页面节点是构成子应用页面的最基本组成部分，子应用页面中的每一个部分都可以称为是一个节点。

具体的，终端会通过预配置解析工具对页面代码进行解析，获得页面解析结果，再通过遍历页面解析结果中代码集合，创建子应用页面相应的页面节点树。其中，预配置解析工具可按照实际应用场景进行配置，本实施例中在此处不对此进行限定。举例说明，预配置解析工具具体可以是预配置的标签语言编译器，标签语言编译器可以对页面代码进行编译，输出编译结果，即页面解析结果。在一个具体的应用中，标签语言可以为 WXML 标签语言，则预配置的标签编译器可以为二进制的 WXML 编译器，该 WXML 编译器接受 WXML 代码文件，通过对 WXML 代码文件进行解析后，会输出 JavaScript 代码。

步骤 206，创建用于转换页面节点树的第一实例对象，建立第一实例对象和子应用页面的绑定关系。

其中，在面向对象程序设计中，“类”在实例化之后叫做一个“实例”，“类”是静态的，不占进程内存，而“实例”拥有动态内存，实例代表一些程序的集合，本实施例中，第一实例对象是指自定义的用于转换页面节点树的实例对象，它包含一些用于转换页面节点树的程序的集合。建立绑定关系是指将第一实例对象和子应用页面相绑定，以使得可以使用第一实例对象来生成子应用页面。

具体的，由于第一实例对象也是通过实例化获得的，因此，终端会先创建用于转换页面节点树的第一接口类，再基于第一接口类，创建用于转换页面节点树的第一实例对象，在该第一接口类中包括转换页面节点树的逻辑方法，在实例化获得第一实例对象，且建立第一实例对象和子应用页面的绑定关系的情况下，可以使用转换页面节点树的逻辑方法来将页面节点树转换为接口调用的形式。

其中，建立第一实例对象和子应用页面的绑定关系是指在对子应用页面进行初始化的时候将第一实例对象和子应用页面相绑定。在具体的应用中，在对子应用页面进行初始化的时候会创建子应用页面相应的根组件实例，因此，只要在创建根组件实例的时候将根组件实例与第一实例对象相绑定，即可建立第一实例对象和子应用页面的绑定关系。

步骤 208，在建立绑定关系的情况下，使用第一实例对象，将页面节点树转换为接口调用的形式，获得页面节点树相应的接口调用信息。

其中，接口调用信息是指可以实现页面节点树的调用信息的集合。举例说明，如图 3 所示，若页面节点树中部分结构如图 3 中（a）所示，包括一个 view（视图）节点和一个内容为 text（文本）的文本节点，且文本节点为 view 节点的子节点，则该部分结构相应的接口调用信息可以如图 3 中（b）所示，在接口调用信息中包括创建一个 view 节点的调用信息、创建一个内容为 text 的文本节点的调用信息以及将 text 节点设置为 view 的子节点的调用信息。

具体的，在建立绑定关系的情况下，就可以使用第一实例对象中用于转换页面节点树的逻辑方法来将页面节点树转换为接口调用的形式，获得页面节点树相应的接口调用信息。在具体的应用中，第一实例对象提供有用于转换页面节点树的第一接口配置信息，第一接口配置信息包括第一实例对象所提供的可供调用的接口以及各接口的说明，使用第一实例对象，将页面节点树转换为接口调用的形式，获得页面节点树相应的接口调用信息，即是基于第一接口配置信息，从第一实例对象所提供的可供调用的接口中确定出实现页面节点

树的接口，将实现页面节点树的接口，作为页面节点树相应的接口调用信息。

步骤 210，基于接口调用信息进行页面绘制，生成子应用页面。

具体的，终端可以基于接口调用信息来执行相应的绘制指令，以进行页面绘制，生成子应用页面。在具体的应用中，由于接口调用信息是通过将页面节点树转换为接口调用的形式获得的，通过基于接口调用信息进行接口调用，即可执行相应的绘制指令进行页面绘制，在页面绘制完成后，即可生成子应用页面。

上述子应用页面处理方法，在监测到请求获取子应用页面的事件时，通过对获取到的子应用页面的页面代码进行解析，获得页面解析结果，遍历页面解析结果中代码集合，创建表征子应用页面的页面结构的页面节点树，能够以页面节点树的形式实现对子应用页面的表示，通过创建用于转换页面节点树的第一实例对象，建立第一实例对象和子应用页面的绑定关系，能够使得所创建的第一实例对象能够在子应用页面的生成过程中被使用，从而可以直接使用第一实例对象，将页面节点树转换为接口调用的形式，获得页面节点树相应的接口调用信息，基于接口调用信息进行页面绘制，生成子应用页面。整个过程，通过直接将页面节点树转换为接口调用的形式，而不使用 DOM 接口作为中间桥梁进行页面渲染，能够提高页面渲染速度。

在一个实施例中，创建用于转换页面节点树的第一实例对象，建立第一实例对象和子应用页面的绑定关系包括：

基于预配置的接口协议的协议模式，获取用于转换页面节点树的第一接口类，第一接口类包括转换页面节点树的逻辑方法；

基于第一接口类，创建用于转换页面节点树的第一实例对象；及

将第一实例对象作为创建参数，创建与子应用页面相应的根组件实例，以建立第一实例对象和子应用页面的绑定关系。

其中，接口协议是一种链接子应用组件组织框架与页面渲染的中间桥梁，是将页面节点树转换为接口调用的形式的基础，接口协议提供有至少一种协议模式供使用，接口协议定义有转换页面节点树的逻辑方法。协议模式是指接口协议所提供的用于实现转换页面节点树的模式，可按照实际应用场景进行配置，每种协议模式都存在相应的接口，且提供不同的协议内容，即每种协议模式都定义有不同的转换页面节点树的逻辑方法。比如，协议模式具体可以为 **Composed Mode**（组合模式），是首选模式，相对简单易用，表示工作在组件树和页面节点树上。又比如，协议模式具体可以为 **Shadow Mode**（组件模式），是指针对 **Shadow Tree**（组件树）的模式，性能较好，但协议本身比较复杂，表示仅工作在组件树上。需要说明的是，在 **Shadow Mode** 下，在将页面节点树转换为接口调用的形式时，通常需要先先将组件树转换为接口调用的形式，再实现组件树到页面节点树的转换。再比如，协议模式具体可以为 **DOM-like**（类似于 DOM 接口的模式），适用于 DOM 的协议，通常应用于适配 DOM 接口。

其中，第一接口类是指与协议模式相应的接口，包括有转换页面节点树的逻辑方法。根组件实例是指页面根组件的实例，每个子应用页面都会有一个与之对应的组件，称为子应用页面的页面根组件，在初始化子应用页面的时候，需要创建出页面根组件的实例，即根组件实例。

具体的，终端会基于预配置的接口协议的协议模式，获取与协议模式相应的、用于转换页面节点树的第一接口类，第一接口类包括转换页面节点树的逻辑方法。在创建第一接口类后，终端会实现第一接口类，再基于所实现的第一接口类创建用于转换页面节点树的第一实例对象，最后再将第一实例对象作为创建参数，创建与子应用页面相应的根组件实例，以建立第一实例对象和子应用页面的绑定关系。

在具体的应用中，由于不同的协议模式与不同的第一接口类相对应，基于协议模式即可获取与协议模式相应的、用于转换页面节点树的第一接口类。需要说明的是，第一接口类所包括的用于转换页面节点树的逻辑方法是预先配置的。在一个具体的应用中，第一接

口类具体可以是 TypeScript 接口，TypeScript 是一种开源的编程语言，通过在 JavaScript 的基础上添加静态类型定义构建而成，是 JavaScript 类型的超集，支持 ECMAScript 6 标准（JavaScript 语言的标准），可以编译成纯 JavaScript。举例说明，协议模式和 TypeScript 接口的对应关系可以如表 1 所示。

表 1

协议模式	主 TypeScript 接口
Composed Mode	glassEasel.composedBackend.Context
Shadow Mode	glassEasel.backend.Context
DOM-like	glassEasel.domlikeBackend.Context

在具体的应用中，终端会将第一实例对象作为创建参数传入用于创建根组件实例的函数，以实现根组件实例的创建。在一个具体的应用中，若用于创建根组件实例的函数的形式为 `glassEasel.Component.createWithContext()`，且第一实例对象可以用 `myCustomBackend` 来表示，则创建根组件实例可以通过代码段“`const rootComponent=glassEasel.Component.createWithContext(X, myCustomBackend)`”实现，其中的 `const rootComponent` 是指根组件实例，`X` 为在创建根组件实例时需要传入的其他参数。举例说明，`X` 具体可以包括 `body` 标签，`body` 标签是在网页中使用的一种标签，表示网页的主体部分，也就是说浏览网页的对象可以看到的内容，可以包含文本、图片、音频、视频等各种内容。

本实施例中，通过基于预配置的接口协议的协议模式，获取用于转换页面节点树的第一接口类，能够实现对转换页面节点树的逻辑方法的获取，从而可以通过基于第一接口类，创建用于转换页面节点树的第一实例对象，将第一实例对象作为创建参数，创建于子应用页面相应的根组件实例，能够建立第一实例对象和子应用页面的绑定关系，从而使得转换页面节点树的逻辑方法可以被使用。

在一个实施例中，基于第一接口类，创建用于转换页面节点树的第一实例对象包括：通过基于类实现接口的第一关键字，创建与第一接口类相应的第一实现类，第一实现类用于实现第一接口类中的转换页面节点树的逻辑方法；及

通过基于类创建实例对象的第二关键字，对第一实现类进行实例化，创建与第一实现类相应的第一实例对象。

其中，基于类实现接口是指基于接口类创建实现类，可以理解的是，在接口类中定义有逻辑方法，通过创建接口类相应的实现类，可以利用实现类来实现相应的接口类中所定义的逻辑方法，即实现相应的接口类。第一关键字是指基于类实现一个接口所用的关键字。比如，第一关键字具体可以为“implements（实现）”。基于类创建实例对象是指对实现类进行实例化，得到实现类相应的实例对象，可以理解的是，实现类是静态的，不占进程内存，因此，需要进一步对实现类进行实例化，以使得实现类所实现的接口类中所定义的逻辑方法可以被使用。第二关键字是指基于类创建相应的实例对象时所使用的关键字。比如，第二关键字具体可以为“new（新建）”。第一实现类是指用于实现第一接口类的类。第一实例对象是指对第一实现类进行实例化后得到的实例对象。

具体的，虽然第一接口类包括转换页面节点树的逻辑方法，但是第一接口类没有方法体，只能通过一个具体的类去实现第一接口类中的转换页面节点树的逻辑方法，本实施例中，即通过创建第一接口类相应的第一实现类，来实现第一接口类。可以理解的是，在第一接口类包括转换页面节点树的逻辑方法的情况下，第一实现类用于实现转换页面节点树的逻辑方法。在创建第一实现类后，由于第一实现类是静态的，不占进程内存，因此，

终端需要进一步对第一实现类进行实例化，以使得转换页面节点树的逻辑方法可以被使用，此时，终端可以通过创建实例对象的第二关键字，来创建与第一实现类相应的第一实例对象。

在具体的应用中，假设第一关键字为“implements”且第二关键字为“new”，则创建第一实例对象的过程可以表示为：第一实现类 implements 第一接口类；第一实例对象=new 第一实现类（）。在一个具体的应用中，假设第一接口类表示为 glassEasel.composedBackend.Context，第一实现类表示为 class MyCustomBackend，第一实例对象表示为 const myCustomBackend，则创建第一实例对象的过程可以表示为：

```
class MyCustomBackend implements glassEasel.composedBackend.Context {  
    // 转换页面节点树的逻辑方法  
};  
const myCustomBackend = new MyCustomBackend();
```

在一个具体的应用中，基于第一接口类，创建用于转换页面节点树的第一实例对象的过程可以如图 4 所示，终端会通过类实现接口的第一关键字，创建与第一接口类相应的第一实现类，在创建第一实现类后，通过创建实例对象的第二关键字，创建与第一实现类相应的第一实例对象。

本实施例中，通过利用基于类实现接口的第一关键字，创建与第一接口类相应的第一实现类，能够实现第一接口类，通过利用创建实例对象的第二关键字，创建与第一实现类相应的第一实例对象，能够实现对第一实现类的实例化，以使得转换页面节点树的逻辑方法可以被使用。

在一个实施例中，使用第一实例对象，将页面节点树转换为接口调用的形式，获得页面节点树相应的接口调用信息包括：

基于第一实例对象所提供的第一接口配置信息，从页面节点树的根节点开始，依次遍历页面节点树中的页面节点，分别确定各页面节点各自相应的第一调用信息，并获得页面节点树的页面配置信息相应的第二调用信息；及

基于第一调用信息和第二调用信息，获得页面节点树相应的接口调用信息。

其中，第一调用信息是指用于实现页面节点的调用信息，页面节点是指在子应用页面中出现的节点。第二调用信息是指用于实现页面配置信息的调用信息。页面配置信息是指用于配置生成子应用页面的相关信息。比如，页面配置信息具体可以是指子应用页面的显示区域宽度、显示区域高度、显示区域像素比等显示区域相关的信息。又比如，页面配置信息具体可以是指子应用页面的主题。举例说明，子应用页面的主题具体可以为“light”（明亮的）。又举例说明，子应用页面的主题具体可以为“dark”（昏暗的）。再比如，页面配置信息具体可以是指子应用页面的样式表。再比如，页面配置信息具体可以是指子应用页面中的焦点，焦点即光标，是子应用页面中闪动的线条。比如，焦点具体可以是指子应用页面中闪动的小竖线。

具体的，在第一接口配置信息中包括第一实例对象所提供的多个可供调用的接口，多个可供调用的接口中包括与各种页面节点相对应的接口，也包括与页面配置信息相对应的接口，因此，在将页面节点树转换为接口调用的形式时，终端可以基于第一实例对象所提供的第一接口配置信息，从页面节点树的根节点开始，依次遍历页面节点树中的页面节点，从多个可供调用的接口中分别确定各页面节点各自相应的第一调用信息，并从多个可供调用的接口中，获得页面节点树的页面配置信息相应的第二调用信息，通过综合第一调用信息和第二调用信息即可获得页面节点树相应的接口调用信息。

在具体的应用中，除了与各种页面节点相对应的接口以及与页面配置信息相对应的接口之外，在第一接口配置信息中还包括获得协议模式的接口、销毁第一实例对象的接口、等待下一次渲染完成时的回调接口、设置全局事件回调函数的接口以及根据坐标获得节点的接口等。其中，针对于获得协议模式的接口，通过该接口可以确定运行于何种模式下。针

对于销毁第一实例对象的接口，通过该接口可以销毁一个第一实例对象。针对于等待下一次渲染成时的回调接口，若回调中设置新的属性能够触发 CSS(Cascading Style Sheets, 层叠样式表)transition (过渡) 动画。针对于设置全局事件回调函数的接口，其相应的回调函数只有一个，可以表示冒泡状态、无特殊操作、禁用默认操作等。针对于根据坐标获得节点的接口，该接口可以用于根据坐标，获得坐标位置上的节点。

需要说明的是，与销毁第一实例对象的接口相应的，也存在创建第一实例对象的接口和获得第一实例对象的接口，通过调用创建第一实例对象的接口可以实现对第一实例对象的创建，通过获得第一实例对象的接口可以获得创建好的第一实例对象。在一个具体的应用中，终端可以通过调用创建第一实例对象的接口创建第一实例对象，还可以在创建根组件实例时，通过调用获得第一实例对象的接口获得创建好的第一实例对象，建立第一实例对象和子应用页面的绑定关系。

在具体的应用中，针对于子应用页面的显示区域宽度、显示区域高度、显示区域像素比等数值，在第一接口配置信息中包括分别与获取显示区域宽度、显示区域高度、显示区域像素比等数值相对应的接口，数值为该相对应的接口的返回值。针对于子应用页面的主题，在第一接口配置信息中提供有获取表征主题的字符串的接口，该获取表征主题的字符串的接口的返回值为表征主题的字符串。

在具体的应用中，针对于子应用页面的样式表，在第一接口配置信息中包括分别与注册样式表、插入样式表项、禁用一段已插入的样式表相对应的接口。在一个具体的应用中，针对于与注册样式表相对应的接口，该接口可以以样式表路径和与样式表路径对应的 CSS 样式表为参数。其中，CSS 样式表需要是可以解读的格式，如果在 CSS 样式表中包含接口调用等引用逻辑，其路径对应的内容有可能也被另一次注册样式表调用注册，该注册时间可能会早于或晚于本次注册。针对于插入样式表项的接口，该接口可以以样式表路径为参数，即其内容来自指定的样式表路径。此外，在插入样式表项的接口中还包括一个可选的样式范围标识符，该样式范围标识符用于返回新建的样式表索引，如果样式范围标识符不为正整数，则视为空，为空时视为全局生效的样式表。

在具体的应用中，针对于子应用页面中的焦点，在第一接口配置信息中包括分别与设置焦点所在页面节点、获得焦点所在页面节点相对应的接口。需要说明的是，如果节点不可聚焦，即在子应用页面中不存在焦点，则需要移除焦点。

本实施例中，通过基于第一实例对象所提供的第一接口配置信息，从页面节点树的根节点开始，依次遍历页面节点树中的页面节点，能够分别确定各页面节点各自相应的第一调用信息，并获得页面节点树的页面配置信息相应的第二调用信息，进而可以综合第一调用信息和第二调用信息，获得页面节点树相应的接口调用信息。

在一个实施例中，基于第一实例对象所提供的第一接口配置信息，从页面节点树的根节点开始，依次遍历页面节点树中的页面节点，分别确定各页面节点各自相应的第一调用信息包括：

针对于页面节点树中的每一个页面节点，从第一实例对象所提供的第一接口配置信息中，确定与所针对的页面节点的页面节点类型相匹配的第一接口信息，基于第一接口信息确定所针对的页面节点相应的第一调用信息。

其中，页面节点类型是指在子应用页面中出现的页面节点的种类，包括普通节点、文本节点、片段节点、组件节点、虚节点等，其中，文本节点是指仅承载文本，无子节点的节点。片段节点是指临时承载节点树片段的节点，它用于表示节点数组，方便批量插入和移除节点。组件节点和虚节点是指仅在 Shadow Mode 下存在的节点，组件节点拥有自己的组件树，可以为虚节点或者非虚节点。普通节点是指在 Composed Mode 下除了文本节点和片段节点之外的节点。第一接口信息是指在第一接口配置信息中与页面节点相应的接口信息。

具体的，在第一接口配置信息中包括第一实例对象所提供的多个可供调用的接口，多

个可供调用的接口中包括与各种页面节点相匹配的接口，针对于页面节点树中的每一个页面节点，终端会从第一实例对象所提供的的第一接口配置信息中包括的多个可供调用的接口中，确定与所针对的页面节点的页面节点类型相匹配的第一接口信息，基于第一接口信息确定所针对的页面节点相应的第一调用信息。在具体的应用中，基于第一接口信息，终端可以将所针对的页面节点的节点配置信息转换为接口调用的形式，进而获得所针对的页面节点相应的第一调用信息。

在具体的应用中，针对于根节点，在第一接口配置信息中包括与获得根节点相应的接口。需要说明的是，在 Shadow Mode 下，根节点必须是组件节点，在 Composed Mode 下，根节点必须是普通节点。针对于普通节点、文本节点、片段节点等页面节点，在第一接口配置信息中分别包括与创建各种页面节点相应的接口。

本实施例中，可以基于页面节点类型，从第一接口配置信息中确定出所针对的页面节点的第一接口信息，进而可以基于第一接口信息实现对所针对的页面节点的第一调用信息的确定。

在一个实施例中，基于第一接口信息确定所针对的页面节点相应的第一调用信息包括：

基于第一接口信息获取第二实例对象，第二实例对象提供对页面节点树进行转换的第二接口配置信息；

使用第二实例对象所提供的第二接口配置信息，获得所针对的页面节点相应的第二接口信息；及

根据第一接口信息和第二接口信息，获得所针对的页面节点相应的第一调用信息。

其中，第二实例对象是指自定义的用于转换页面节点的实例对象，它包含一些用于转换页面节点的程序的集合。第二接口配置信息包括第二实例对象所提供的可供调用的接口以及各接口的说明。第二接口信息是指在第二接口配置信息中与页面节点相应的接口信息。

具体的，在需要将所针对的页面节点转换为接口调用的形式时，终端会基于第一接口信息获取第二实例对象，第二实例对象提供有对页面节点树进行转换的第二接口配置信息。在获取到第二实例对象后，就可以使用第二实例对象所提供的第二接口配置信息，获得所针对的页面节点相应的第二接口信息，进而可以通过综合第一接口信息和第二接口信息，获得所针对的页面节点相应的第一调用信息。其中，第一接口信息是以已创建的第二实例对象为返回值的，因此，终端基于第一接口信息可以直接获取到第二实例对象。

在具体的应用中，第二接口配置信息包括第二实例对象所提供的可供调用的接口以及各接口的说明，使用第二实例对象所提供的第二接口配置信息，获得所针对的页面节点相应的第二接口信息，即是基于第二接口配置信息，从第二实例对象所提供的可供调用的接口中确定出实现所针对的页面节点的接口，以将所针对的页面节点转换为接口调用的形式。

本实施例中，在基于第一接口信息获取第二实例对象的基础上，可以通过使用第二实例对象所提供的第二接口配置信息，获得所针对的页面节点相应的第二接口信息，从而可以综合第一接口信息和第二接口信息，获得所针对的页面节点相应的第一调用信息。

在一个实施例中，子应用页面处理方法还包括：

通过第一实例对象，创建用于转换页面节点的第二接口类，第二接口类包括转换页面节点的逻辑方法；

通过基于类实现接口的第一关键字，创建与第二接口类相应的第二实现类，第二实现类用于实现第二接口类中的转换页面节点的逻辑方法；及

通过基于类创建实例对象的第二关键字，对第二实现类进行实例化，创建与第二实现类相应的第二实例对象。

具体的，在确定所针对的页面节点的第一接口信息之前，终端需要先通过第一实例对象创建第二实例对象，在通过第一实例对象创建第二实例对象时，可以先创建用于转换页面节点的第二接口类，再通过基于类实现接口的第一关键字，来创建与第二接口类相应的第二实现类，来实现第二接口类中的转换页面节点的逻辑方法，最后通过基于类创建实例

对象的第二关键字，对第二实现类进行实例化，创建与第二实现类相应的第二实例对象。

可以理解的是，虽然第二接口类包括转换页面节点的逻辑方法，但是第二接口类没有方法体，只能通过一个具体的类去实现第二接口类中的转换页面节点的逻辑方法，本实施例中，即通过创建与第二接口类相应的第二实现类，来实现第二接口类，即第二实现类用于实现转换页面节点的逻辑方法。在创建第二实现类后，由于第二实现类是静态的，不占进程内存，因此，终端需要进一步对第二实现类进行实例化，以使得转换页面节点的逻辑方法可以被使用，此时，终端可以通过基于类创建实例对象的第二关键字，来创建与第二实现类相应的第二实例对象。

本实施例中，通过利用基于类实现接口的第二关键字，创建与第二接口类相应的第二实现类，能够实现第二接口类，通过利用基于类创建实例对象的第二关键字，创建与第二实现类相应的第二实例对象，能够实现对第二实现类的实例化，以使得转换页面节点的逻辑方法可以被使用。在一个实施例中，使用第二实例对象所提供的第二接口配置信息，获得所针对的页面节点相应的第二接口信息包括：

基于所针对的页面节点的节点配置信息，从第二实例对象所提供的第二接口配置信息中，获得所针对的页面节点相应的第二接口信息。

其中，节点配置信息是指用于配置生成页面节点的相关信息。比如，节点配置信息具体可以是节点标识、范围标识符、节点风格、节点样式、文本内容、节点的外接矩形区域、节点的内部滚动位置等节点属性信息。又比如，节点配置信息具体可以是指节点与页面节点树中其他节点的关联关系。再比如，节点配置信息具体可以是指节点环境信息，节点环境信息用于表征所针对的节点是否为组件节点。

具体的，在第二接口配置信息中包括第二实例对象所提供的多个可供调用的接口，多个可供调用的接口中包括与节点配置信息相对应的接口，因此，终端可以基于第二实例对象所提供的第二接口配置信息，从多个可供调用的接口中，获得所针对的页面节点相应的第二接口信息。

在具体的应用中，除了与节点配置信息相对应的接口之外，在第二接口配置信息中还包括释放节点的接口和通知节点相关信息已经创建完毕的接口。其中，释放节点的接口可以用于释放节点，通知节点相关信息已经创建完毕的接口，用于通知节点相关信息已经创建完毕，并为节点设置一个关联值。需要说明的是，对除文本节点以外的节点，都会调用一次该通知节点相关信息已经创建完毕的接口，说明节点相关信息已经创建完毕，可以继续创建下一个节点。

在具体的应用中，针对于节点标识、范围标识符、节点风格、节点样式、文本内容、节点的外接矩形区域、节点的内部滚动位置等节点属性信息，在第二接口配置信息中包括分别与设置每种节点属性相对应的接口。此外，在第二接口配置信息中还提供有另外的设置其他节点属性和移除其他节点属性的接口，方便实现对节点属性的设置。

在一个具体的应用中，在与设置每种节点属性相对应的接口中包括将节点设置为 slot（插槽）节点的接口以及设置节点的目标 slot 的接口，slot 节点用于承载组件引用时提供的子节点。进一步的，通过设置节点的目标 slot 的接口还可以将节点设置为 slot-inherit（插槽-继承）节点，对于 slot-inherit 节点，它的子节点在页面节点树上并不视为子节点，而视为它之后的兄弟节点。这可以使得这些子节点拥有不同的目标 slot。需要说明的是，节点仅在初始化阶段、还没有子节点时才会被设为 slot-inherit 节点，一旦节点被设为 slot-inherit 节点，就不会再被设为非 slot-inherit 节点。

在一个具体的应用中，针对于设置节点的范围标识符的接口，对于同一个节点，该接口最多被调用一次，即节点的范围标识符最多被设置一次。在使用该接口时，如果传入的参数不为正整数，则视为空，在匹配样式规则时，如果使用除 class 以外的节点选择器（如标签名选择器、ID 选择器）来匹配这个节点，则样式表的范围标识符必须为空或等于这个节点的范围标识符。需要说明的是，在 Shadow Mode 下，该接口仅对普通节点和非虚组件

节点调用，在 **Composed Mode** 下，该接口仅对普通节点调用。

在一个具体的应用中，针对于设置节点风格的接口，在 **Shadow Mode** 下，该接口仅对普通节点和非虚组件节点调用，在 **Composed Mode** 下，该接口仅对普通节点调用。针对于添加节点样式类别的接口，如果样式类别标识不为非负整数，则视为空，在匹配样式规则时，如果使用这个样式类别来匹配这个节点，则样式表的范围标识符必须为空或等于样式类别的样式类别标识，需要说明的是，在 **Shadow Mode** 下，该接口仅对普通节点和非虚组件节点调用，在 **Composed Mode** 下，该接口仅对普通节点调用。

在一个具体的应用中，针对于移除指定样式类别的接口，需要名称和样式类别标识都可以匹配上，如果样式类别标识不为非负整数，则视为空。需要说明的是，在 **Shadow Mode** 下，该接口仅对普通节点和非虚组件节点调用，在 **Composed Mode** 下，该接口仅对普通节点调用。

在一个具体的应用中，针对于获得节点的外接矩形区域的接口，如果节点不是规整的矩形区域（如是内联文本或被旋转过的矩形），需要计算它的最小外接矩形区域，如果节点没有布局信息，返回全 0 值。针对于获得节点的内部滚动位置的接口，如果节点不可滚动，返回位置 0 值和自身尺寸值。

在具体的应用中，针对于节点与页面节点树中其他节点的关联关系，在第二接口配置信息中包括分别与设置每种关联关系相对应的接口。比如，这里的关联关系具体可以为追加一个子节点、移除一个子节点、插入一个子节点、删除节点、追加多个节点、删除且插入节点。

在一个具体的应用中，针对于移除一个子节点的接口，该被移除的子节点可能在后续被再次使用，如果在移除子节点时提供了节点索引号，则节点索引号必须等于子节点在子节点列表中的序号，如果节点索引号不为非负整数，则视为未赋值。

在一个具体的应用中，针对于插入一个子节点的接口，在第二接口配置信息中提供有多个接口供选择，每种接口针对于参数的不同有不同的行为，具体来说，针对于以前一子节点标识和节点索引号为参数的接口，若没有提供前一子节点标识和节点索引号，相对于追加子节点，若提供了前一子节点标识或节点索引号，相当于在前一子节点标识相应的子节点或节点索引号相应的子节点之前插入一个子节点，如果同时提供了前一子节点标识和节点索引号，则节点索引号必须等于前一子节点标识相应的子节点在子节点列表中的序号，如果节点索引号不为非负整数，则视为未赋值，被插入的子节点可以保证是没有父节点的。

针对于以后一子节点标识和节点索引号为参数的接口，如果没有提供后一子节点标识和节点索引号，相当于追加子节点，如果提供了后一子节点标识或节点索引号，则替换后一子节点标识相应的子节点或节点索引号相应的子节点，若同时提供了后一子节点标识和节点索引号，则节点索引号必须等于后一子节点标识相应的节点在子节点列表中的序号，如果节点索引号不为非负整数，则视为未赋值，被插入的子节点可以保证是没有父节点的。

在一个具体的应用中，针对于删除节点的接口，若提供有节点标识和节点数量，则删除从节点标识开始的节点数量个节点。针对于追加多个节点的接口，可以追加作为参数的片段节点中包含的所有节点，该片段节点在被追加后应该被清空，但可能被再次使用。针对于删除且插入节点的接口，若提供有节点标识和节点数量，则删除从节点标识开始的节点数量个节点，并追加作为参数的片段节点中包含的所有节点。

本实施例中，基于所针对的页面节点的节点配置信息，可以从第二实例对象所提供的第二接口配置信息中，获得所针对的页面节点相应的第二接口信息，实现了对所针对的页面节点的接口调用转换。

在一个实施例中，根据第一接口信息和第二接口信息，获得所针对的页面节点相应的第一调用信息包括：

当所针对的页面节点为组件节点，基于第二接口信息获得预创建的第三实例对象，第三实例对象是指用于转换组件节点的实例对象；

使用第三实例对象，获得组件节点相应的第三接口信息；及

根据组件节点相应的第一接口信息、第二接口信息以及第三接口信息，获得组件节点相应的第一调用信息。

其中，组件节点是指组件类型的节点，通过组件的思想，可以将子应用页面拆分成一个个可以复用的模块，每一个模块就是一个组件，一个子应用页面由若干组件组合而成，一个复杂组件也可以由若干简单组件组合而成。本实施例中，组件节点主要是在 Shadow Mode 出现的，拥有自己的组件节点树。第三实例对象是指预创建的用于转换组件节点的实例，它包含一些用于转换组件节点的程序的集合。第三接口信息是指在第三实例对象所提供的接口配置信息中与组件节点相应的接口信息。

具体的，当所针对的页面节点为组件节点，终端会基于第二接口信息中获得节点环境信息的接口信息获得预创建的第三实例对象，使用组件树环境实例，获得组件节点相应的第三接口信息，综合组件节点相应的第一接口信息、第二接口信息以及第三接口信息，获得组件节点相应的第一调用信息。其中，获得节点环境信息的接口信息以已创建的第三实例对象为返回值，因此，若页面节点为组件节点，终端就可以基于获得节点环境信息的接口信息获得已创建的第三实例对象。

在具体的应用中，在使用第三实例对象之前，需要预创建第三实例对象，在创建第三实例对象时，可以先创建包括转换组件节点的逻辑方法的组件接口类，再通过基于类实现接口的第一关键字创建组件实现类后，通过利用基于类创建实例对象的第二关键字，就可以实现对组件实现类的实例化，创建第三实例对象。

本实施例中，在所针对的页面节点为组件节点的情况下，能够基于第二接口信息获得预创建的第三实例对象，从而可以使用第三实例对象，获得组件节点相应的第三接口信息，综合组件节点相应的第一接口信息、第二接口信息以及第三接口信息，获得组件节点相应的第一调用信息。

在一个实施例中，使用第三实例对象，获得组件节点相应的第三接口信息包括：

基于第三实例对象所提供的第三接口配置信息，从组件节点的根节点开始，依次遍历组件节点中子节点，分别确定各子节点各自相应的子节点调用信息；及

基于各子节点各自相应的子节点调用信息，获得组件节点相应的第三接口信息。

其中，第三接口配置信息包括第三实例对象所提供的可供调用的接口以及各接口的说明。第三接口信息是指在第三接口配置信息中与组件节点相应的接口信息。子节点调用信息是指用于实现子节点的调用信息。

具体的，在第三接口配置信息中包括第三实例对象所提供的多个可供调用的接口，多个可供调用的接口中包括与组件节点中各子节点相对应的接口，因此，在将组件节点转换为接口调用的形式时，终端可以基于第三实例对象所提供的第三接口配置信息，从组件节点树的根节点开始，依次遍历组件节点中的子节点，从多个可供调用的接口中分别确定与各子节点各自相应的子节点调用信息，综合各子节点各自相应的子节点调用信息，获得组件节点相应的第三接口信息。

本实施例中，通过基于第三实例对象所提供的第三接口配置信息，从组件节点的根节点开始，依次遍历组件节点中子节点，分能够别确定各子节点各自相应的子节点调用信息，进而可以综合各子节点各自相应的子节点调用信息，获得组件节点相应的第三接口信息。

在一个实施例中，基于第三实例对象所提供的第三接口配置信息，从组件节点的根节点开始，依次遍历组件节点中子节点，分别确定各子节点各自相应的子节点调用信息包括：

针对于组件节点中每一个子节点，从第三实例对象所提供的第三接口配置信息中，确定与所针对的子节点子节点类型相匹配的子节点接口信息；及

基于子节点接口信息获取第二实例对象，使用第二实例对象，获得所针对的子节点相应的子节点调用信息。

其中，子节点类型是指在组件节点中出现的子节点的种类，包括普通节点、文本节点、

组件节点、外部组件节点、虚节点等，外部组件节点是一个已经预创建好的节点树，可以与其他部分直接拼接在一起。

具体的，在第三接口配置信息中包括第三实例对象所提供的多个可供调用的接口，多个可供调用的接口中包括与各种子节点相匹配的接口，针对于组件节点中每一个子节点，终端会从第三实例对象所提供的第三接口配置信息中包括的多个可供调用的接口中，确定与所针对的子节点子节点类型相匹配的子节点接口信息，基于子节点接口信息获取第二实例对象，使用第二实例对象，获得所针对的子节点相应的子节点调用信息。其中，子节点接口信息以已创建的第二实例对象为返回值，因此，终端可以基于子节点接口信息获得已创建的第二实例对象。

在具体的应用中，第二实例对象提供有用于转换子节点的第二接口配置信息，第二接口配置信息包括第二实例对象所提供的可供调用的接口以及各接口的说明，使用第二实例对象，获得所针对的子节点相应的子节点调用信息，即是基于第二接口配置信息，从第二实例对象所提供的可供调用的接口中确定出实现子节点的接口，以将子节点转换为接口调用的形式。需要说明的是，子节点也为页面节点，因此，本实施例中使用节点实例获得所针对的子节点相应的子节点调用信息的方式与使用第二实例对象获得所针对的页面节点相应的第二接口信息的方式相似，本实施例中在此处不再赘述。

本实施例中，可以基于子节点类型，从第三接口配置信息中确定出所针对的子节点子节点接口信息，进而可以基于子节点接口信息实现对第二实例对象的获取，以使用第二实例对象，实现对所针对的子节点相应的子节点调用信息的获取。

在一个实施例中，使用第二实例对象，获得所针对的子节点相应的子节点调用信息包括：

基于所针对的子节点子节点配置信息，从第二实例对象所提供的第二接口配置信息中，获取所针对的子节点相应的子节点调用信息。

其中，子节点配置信息是指用于配置生成子节点的相关信息。比如，子节点配置信息具体可以是节点标识、范围标识符、节点风格、节点样式、文本内容、节点的外接矩形区域、节点的内部滚动位置等子节点属性信息。又比如，子节点配置信息具体可以是指节点与组件节点中其他子节点的关联关系。再比如，节点配置信息具体可以是指节点环境信息，节点环境信息用于表征所针对的子节点是否为组件节点。

具体的，在第二接口配置信息中包括第二实例对象所提供的多个可供调用的接口，多个可供调用的接口中包括与子节点配置信息相对应的接口，因此，终端可以基于第二实例对象所提供的第二接口配置信息，从多个可供调用的接口中，获取所针对的子节点相应的子节点调用信息。需要说明的是，子节点也为页面节点，因此，本实施例中基于子节点配置信息获得所针对的子节点相应的子节点调用信息的方式与基于节点配置信息获得所针对的页面节点相应的第二接口信息的方式相似，本实施例中在此处不再赘述。

本实施例中，能够基于所针对的子节点子节点配置信息，实现对所针对的子节点相应的子节点调用信息的获取。

在一个实施例中，对页面代码进行解析，获得页面解析结果包括：

确定编写页面代码所使用的标签语言，通过标签语言相应的标签语言编译器，对页面代码进行编译，得到页面解析结果。

具体的，标签语言编译器是指用于对使用标签语言所编写的代码进行编译的程序集合，通过对使用标签语言所编写的代码进行编译，可以将使用标签语言所编写的代码转换为另一种编程语言的代码。因此，在对页面代码进行解析时，终端可以先确定编写页面代码所使用的标签语言，再通过标签语言相应的标签语言编译器，来对页面代码进行编译，得到页面解析结果，即终端内子应用的渲染层的运行环境可以直接理解的代码。

本实施例中，通过先确定编写页面代码所使用的标签语言，能够利用标签语言相应的标签语言编译器，实现对页面代码的编译，得到页面解析结果。

在一个实施例中，如图 5 所示，通过一个流程示意图来说明本申请的子应用页面处理方法，该子应用页面处理方法具体包括以下步骤：

当打开子应用页面（即监测到请求获取子应用页面的事件），拉取子应用页面的页面代码（即获取子应用页面的页面代码），对页面代码进行解析，获得页面解析结果，遍历页面解析结果中代码集合，创建子应用页面相应的页面节点树，创建用于转换页面节点树的第一实例对象，建立第一实例对象和子应用页面的绑定关系，在建立绑定关系的情况下，使用第一实例对象，将页面节点树转换为接口调用的形式，获得页面节点树相应的接口调用信息，基于接口调用信息进行底层绘制显示，即基于接口调用信息进行页面绘制，生成子应用页面。

其中，第一实例对象是基于预配置的接口协议创建的，本申请中的接口协议是一种链接子应用组件组织框架与页面渲染的中间桥梁，是将页面节点树转换为接口调用的形式的基础，通过接口协议可以将页面节点树转换为底层接口调用之后传递给底层渲染进行界面展示。

为了更好的说明本申请的子应用页面处理方法，下面先对子应用页面从开发者代码编写到绘制显示的过程进行说明，如图 6 所示，该过程具体包括以下步骤：开发者编写子应用页面的页面代码，在打开子应用页面时，终端内的子应用组件组织框架会拉取到页面代码，在拿到页面代码后，会对页面代码进行解析，获得页面解析结果，遍历页面解析结果中代码集合，创建子应用页面相应的页面节点树，页面节点树是基于子应用页面的页面节点而生成的树状结构，创建用于转换页面节点树的第一实例对象，建立第一实例对象和子应用页面的绑定关系，在建立绑定关系的情况下，使用第一实例对象，将页面节点树转换为接口调用的形式，获得页面节点树相应的接口调用信息，将接口调用信息输出至终端内的 GPU（graphics processing unit，图形处理器），GPU 基于接口调用信息进行底层绘制显示，即基于接口调用信息进行页面绘制，生成子应用页面。

其中，子应用组件组织框架内置在子应用基础库中，为子应用组件提供基础的支持，子应用内所有的组件，包括内置组件和自定义组件，都是由子应用组件组织框架进行组织和管理。举例说明，子应用组件组织框架具体可以为 GlassEasel，是通用的组件化界面框架。

在一个实施例中，创建用于转换页面节点树的第一实例对象，建立第一实例对象和子应用页面的绑定关系，在建立绑定关系的情况下，使用第一实例对象，将页面节点树转换为接口调用的形式，获得页面节点树相应的接口调用信息可以理解为使用自定义后端来将页面节点树转换为接口调用的形式，该自定义后端可以被称作渲染后端，子应用组件组织框架可以将它生成的页面节点树传递给渲染后端，通过渲染后端来将接收到的页面节点树转换为接口调用的形式，获得页面节点树相应的接口调用信息，将接口调用信息输出至终端内的 GPU。

在一个实施例中，以页面代码为 wxml 组件代码，页面解析结果为 JavaScript 代码为例，从开发者代码编写到绘制显示的过程可以如图 7 所示，开发者编写 wxml 组件代码，在打开子应用页面时，终端内的子应用组件组织框架会拉取到 wxml 组件代码，在拿到 wxml 组件代码后，会对页面代码进行解析，获得 JavaScript 代码，基于 JavaScript 代码使用自定义后端，将页面节点树转换为接口调用的形式，获得页面节点树相应的接口调用信息，将接口调用信息输出至终端内的 GPU（graphics processing unit，图形处理器），GPU 基于接口调用信息进行底层绘制显示，即基于接口调用信息进行页面绘制，生成子应用页面。

在本申请的子应用页面处理方法出现之前，子应用页面采用 web DOM 作为渲染后端，但是 web DOM 需要依赖浏览器/webview（网页视图）提供，由于 web DOM 有着繁重的历史包袱，DOM 接口的组件过多，对于子应用页面的渲染性能不及原生。基于此，发明人提出了本申请中的子应用页面处理方法，通过应用本申请中的子应用页面处理方法中的渲染后端来替代原有的 web DOM 渲染后端，由于通过直接将页面节点树转换为接口调用的

形式，而不使用 DOM 接口作为中间桥梁进行页面渲染，能够提高页面渲染速度。

在一个实施例中，本申请中的渲染后端主要分别两种模式，即 Shadow Mode 和 Composed Mode。针对于 Shadow Mode，子应用组件组织框架仅工作在组件节点树上，由渲染后端完成组件节点树到页面节点树的转换。针对于 Composed Mode，子应用组件组织框架工作在组件节点树和页面节点树上。需要说明的是，两种模式需要的接口稍有不同，渲染后端必须向子应用组件组织框架提供接口，才能使得子应用组件组织框架能够正确向渲染后端输出所需要的信息。

其中，渲染后端的接口协议可以是以 TypeScript 为基础语言的协议，相应的接口形式可以为面向对象形式。举例说明，渲染后端所提供的接口可以为：Context#mode: ContextMode、Context#registerStyleSheetContent(path: string, content: any)等形式。需要说明的是，渲染后端所提供的接口主要分为两类，接口中以 Class#method(...) 形式标注的接口表示是对应 Class 的实例方法，即 Class.prototype.method(...)。标记为 async 的接口，统一为回调函数形式，即 async Class#method(...): T 实际为 Class.prototype.method(..., (T) => void)。

其中，本申请中的渲染后端主要提供有三类实例，分别为第一实例对象、第二实例对象和第三实例对象，每类实例提供不同的接口供子应用组件组织框架调用。针对于第一实例对象，每个第一实例对象可以在界面上展示一颗节点树，第二实例对象可以通过第一实例对象创建，第三实例对象在被创建后可以通过第二实例对象获取到，即可以通过第二实例对象获取组件的组件节点树环境。针对于组件节点树中的子节点，也可以使用第二实例对象来将其转换为接口调用的形式，因此，渲染后端所提供的三类实例的调用关系可以如图 8 所示，第一实例对象可以调用第二实例对象，第二实例对象可以调用第三实例对象，第三实例对象可以调用第二实例对象。

发明人认为，本申请的子应用页面处理方法，提供了必备的后端接口，支持更多特性，并且抛弃不会用到的特性、支持在组件节点树上工作也支持适配 DOM 接口，对比 web DOM 渲染后端，可以提高子应用页面的渲染速度。其中，不会用到的特性即 DOM 接口中不会被使用于子应用页面的渲染的特性。

在一个实施例中，如图 9 所示，通过一个流程示意图来说明本申请的子应用页面处理方法，该子应用页面处理方法具体包括以下步骤：

当监测到请求获取子应用页面的事件，终端获取子应用页面的页面代码，对页面代码进行解析，获得页面解析结果，遍历页面解析结果中代码集合，创建子应用页面相应的页面节点树，基于预配置的接口协议的协议模式，获取用于转换页面节点树的第一接口类，第一接口类包括转换页面节点树的逻辑方法，通过基于类实现接口的第一关键字，创建与第一接口类相应的第一实现类，第一实现类用于实现第一接口类中的转换页面节点树的逻辑方法，通过基于类创建实例对象的第二关键字，创建与第一实现类相应的第一实例对象，将第一实例对象作为创建参数，创建与子应用页面相应的根组件实例，以建立第一实例对象和子应用页面的绑定关系。

在建立绑定关系的情况下，通过第一实例对象创建第二实例对象，第二实例对象提供对页面节点树进行转换的第二接口配置信息，针对于页面节点树中的每一个页面节点，从第一实例对象所提供的第一接口配置信息中，确定与所针对的页面节点的页面节点类型相匹配的第一接口信息，基于第一接口信息获取第二实例对象，基于所针对的页面节点的节点配置信息，从第二实例对象所提供的第二接口配置信息中，获得与所针对的页面节点相应的第二接口信息，根据第一接口信息和第二接口信息，获得所针对的页面节点相应的第一调用信息，并基于第一实例对象所提供的第一接口配置信息，获得页面节点树的页面配置信息相应的第二调用信息，基于第一调用信息和第二调用信息，获得页面节点树相应的接口调用信息，基于接口调用信息进行页面绘制，生成子应用页面。

在具体的应用中，在生成子应用页面之后，若需要切换下显示其他子应用页面，终端

会获取其他子应用页面的页面代码，再基于上述实施例中的子应用页面处理方法，来生成其他子应用页面。

在一个实施例中，若预配置的接口协议的协议模式为 **Shadow Mode**，所针对的页面节点可能为组件节点，若所针对的页面节点为组件节点，其相应的第一调用信息的获取流程可以如图 10 所示，终端会基于第二接口信息获得预创建的第三实例对象，针对于组件节点中每一个子节点，从第三实例对象所提供的第三接口配置信息中，确定与所针对的子节点的子节点类型相匹配的子节点接口信息，基于子节点接口信息获取第二实例对象，基于所针对的子节点的子节点配置信息，从第二实例对象所提供的第二接口配置信息中，获取与所针对的子节点相应的子节点调用信息，基于各子节点各自相应的子节点调用信息，获得组件节点相应的第三接口信息，根据组件节点相应的第一接口信息、第二接口信息以及第三接口信息，获得组件节点相应的第一调用信息。

应该理解的是，虽然如上所述的各实施例所涉及的流程图中的各个步骤按照箭头的指示依次显示，但是这些步骤并不是必然按照箭头指示的顺序依次执行。除非本文中有明确的说明，这些步骤的执行并没有严格的顺序限制，这些步骤可以以其它的顺序执行。而且，如上所述的各实施例所涉及的流程图中的至少一部分步骤可以包括多个步骤或者多个阶段，这些步骤或者阶段并不必然是在同一时刻执行完成，而是可以在不同的时刻执行，这些步骤或者阶段的执行顺序也不必然是依次进行，而是可以与其它步骤或者其它步骤中的步骤或者阶段的至少一部分轮流或者交替地执行。

基于同样的发明构思，本申请实施例还提供了一种用于实现上述所涉及的子应用页面处理方法的子应用页面处理装置。该装置所提供的解决问题的实现方案与上述方法中所记载的实现方案相似，故下面所提供的一个或多个子应用页面处理装置实施例中的具体限定可以参见上文中对于子应用页面处理方法的限定，在此不再赘述。

在一个实施例中，如图 11 所示，提供了一种子应用页面处理装置，包括：代码获取模块 1102、节点树生成模块 1104、页面初始化模块 1106、节点树转换模块 1108 和页面绘制模块 1110，其中：

代码获取模块 1102，用于当监测到请求获取子应用页面的事件，获取子应用页面的页面代码，子应用页面是指在子应用中显示的页面；

节点树生成模块 1104，用于对页面代码进行解析，获得页面解析结果，遍历页面解析结果中代码集合，创建子应用页面相应的页面节点树，页面节点树是基于子应用页面的页面节点而生成的树状结构；

页面初始化模块 1106，用于创建用于转换页面节点树的第一实例对象，建立第一实例对象和子应用页面的绑定关系；

节点树转换模块 1108，用于在建立绑定关系的情况下，使用第一实例对象，将页面节点树转换为接口调用的形式，获得页面节点树相应的接口调用信息；

页面绘制模块 1110，用于基于接口调用信息进行页面绘制，生成子应用页面。

上述子应用页面处理装置，在监测到请求获取子应用页面的事件时，通过对获取到的子应用页面的页面代码进行解析，获得页面解析结果，遍历页面解析结果中代码集合，创建表征子应用页面的页面结构的页面节点树，能够以页面节点树的形式实现对子应用页面的表示，通过创建用于转换页面节点树的第一实例对象，建立第一实例对象和子应用页面的绑定关系，能够使得所创建的第一实例对象能够在子应用页面的生成过程中被使用，从而可以直接使用第一实例对象，将页面节点树转换为接口调用的形式，获得页面节点树相应的接口调用信息，基于接口调用信息进行页面绘制，生成子应用页面。整个过程，通过直接将页面节点树转换为接口调用的形式，而不使用 **DOM** 接口作为中间桥梁进行页面渲染，能够提高页面渲染速度。

在一个实施例中，页面初始化模块还用于基于预配置的接口协议的协议模式，获取用于转换页面节点树的第一接口类，第一接口类包括转换页面节点树的逻辑方法，基于第一

接口类，创建用于转换页面节点树的第一实例对象，将第一实例对象作为创建参数，创建与子应用页面相应的根组件实例，以建立第一实例对象和子应用页面的绑定关系。

在一个实施例中，页面初始化模块还用于通过基于类实现接口的第一关键字，创建与第一接口类相应的第一实现类，第一实现类用于实现第一接口类中的转换页面节点树的逻辑方法，通过基于类创建实例对象的第二关键字，对第一实现类进行实例化，创建与第一实现类相应的第一实例对象。

在一个实施例中，节点树转换模块还用于基于第一实例对象所提供的第一接口配置信息，从页面节点树的根节点开始，依次遍历页面节点树中的页面节点，分别确定各页面节点各自相应的第一调用信息，并获得页面节点树的页面配置信息相应的第二调用信息，基于第一调用信息和第二调用信息，获得页面节点树相应的接口调用信息。

在一个实施例中，节点树转换模块还用于针对于页面节点树中的每一个页面节点，从第一实例对象所提供的第一接口配置信息中，确定与所针对的页面节点的页面节点类型相匹配的第一接口信息，基于第一接口信息确定所针对的页面节点相应的第一调用信息。

在一个实施例中，节点树转换模块还用于基于第一接口信息获取第二实例对象，第二实例对象提供对页面节点树进行转换的第二接口配置信息，使用第二实例对象所提供的第二接口配置信息，获得所针对的页面节点相应的第二接口信息，根据第一接口信息和第二接口信息，获得所针对的页面节点相应的第一调用信息。

在一个实施例中，节点树转换模块还用于通过第一实例对象，创建用于转换页面节点的第二接口类，第二接口类包括转换页面节点的逻辑方法，通过基于类实现接口的第一关键字，创建与第二接口类相应的第二实现类，第二实现类用于实现第二接口类中的转换页面节点的逻辑方法，通过基于类创建实例对象的第二关键字，对第二实现类进行实例化，创建与第二实现类相应的第二实例对象。

在一个实施例中，节点树转换模块还用于基于所针对的页面节点的节点配置信息，从第二实例对象所提供的第二接口配置信息中，获得与所针对的页面节点相应的第二接口信息。

在一个实施例中，节点树转换模块还用于当所针对的页面节点为组件节点，基于第二接口信息获得预创建的第三实例对象，第三实例对象是指用于转换组件节点的实例对象，使用第三实例对象，获得组件节点相应的第三接口信息，根据组件节点相应的第一接口信息、第二接口信息以及第三接口信息，获得组件节点相应的第一调用信息。

在一个实施例中，节点树转换模块还用于基于第三实例对象所提供的第三接口配置信息，从组件节点的根节点开始，依次遍历组件节点中子节点，分别确定各子节点各自相应的子节点调用信息，基于各子节点各自相应的子节点调用信息，获得组件节点相应的第三接口信息。

在一个实施例中，节点树转换模块还用于针对于组件节点中每一个子节点，从第三实例对象所提供的第三接口配置信息中，确定与所针对的子节点的子节点类型相匹配的子节点接口信息，基于子节点接口信息获取第二实例对象，使用第二实例对象，获得所针对的子节点相应的子节点调用信息。

在一个实施例中，节点树转换模块还用于基于所针对的子节点的子节点配置信息，从第二实例对象所提供的第二接口配置信息中，获取与所针对的子节点相应的子节点调用信息。

在一个实施例中，节点树生成模块还用于确定编写页面代码所使用的标签语言，通过标签语言相应的标签语言编译器，对页面代码进行编译，得到页面解析结果。

上述子应用页面处理装置中的各个模块可全部或部分通过软件、硬件及其组合来实现。上述各模块可以硬件形式内嵌于或独立于计算机设备中的处理器中，也可以以软件形式存储于计算机设备中的存储器中，以便于处理器调用执行以上各个模块对应的操作。

在一个实施例中，提供了一种计算机设备，该计算机设备可以是终端，其内部结构图

可以如图 12 所示。该计算机设备包括处理器、存储器、输入/输出接口、通信接口、显示单元和输入装置。其中，处理器、存储器和输入/输出接口通过系统总线连接，通信接口、显示单元和输入装置通过输入/输出接口连接到系统总线。其中，该计算机设备的处理器用于提供计算和控制能力。该计算机设备的存储器包括非易失性存储介质、内存储器。该非易失性存储介质存储有操作系统和计算机可读指令。该内存储器为非易失性存储介质中的操作系统和计算机可读指令的运行提供环境。该计算机设备的输入/输出接口用于处理器与外部设备之间交换信息。该计算机设备的通信接口用于与外部的终端进行有线或无线方式的通信，无线方式可通过 WIFI、移动蜂窝网络、NFC（近场通信）或其他技术实现。该计算机可读指令被处理器执行时以实现一种子应用页面处理方法。该计算机设备的显示单元用于形成视觉可见的画面，可以是显示屏、投影装置或虚拟现实成像装置，显示屏可以是液晶显示屏或电子墨水显示屏，该计算机设备的输入装置可以是显示屏上覆盖的触摸层，也可以是计算机设备外壳上设置的按键、轨迹球或触控板，还可以是外接的键盘、触控板或鼠标等。

本领域技术人员可以理解，图 12 中示出的结构，仅仅是与本申请方案相关的部分结构的框图，并不构成对本申请方案所应用于其上的计算机设备的限定，具体的计算机设备可以包括比图中所示更多或更少的部件，或者组合某些部件，或者具有不同的部件布置。

在一个实施例中，还提供了一种计算机设备，包括存储器和处理器，存储器中存储有计算机可读指令，该处理器执行计算机可读指令时实现上述各方法实施例中的步骤。

在一个实施例中，提供了一种计算机可读存储介质，其上存储有计算机可读指令，该计算机可读指令被处理器执行时实现上述各方法实施例中的步骤。

在一个实施例中，提供了一种计算机程序产品，包括计算机可读指令，该计算机可读指令被处理器执行时实现上述各方法实施例中的步骤。

本领域普通技术人员可以理解实现上述实施例方法中的全部或部分流程，是可以通过计算机可读指令来指令相关的硬件来完成，所述的计算机可读指令可存储于一非易失性计算机可读存储介质中，该计算机可读指令在执行时，可包括如上述各方法的实施例的流程。其中，本申请所提供的各实施例中所使用的对存储器、数据库或其它介质的任何引用，均可包括非易失性和易失性存储器中的至少一种。非易失性存储器可包括只读存储器（Read-Only Memory, ROM）、磁带、软盘、闪存、光存储器、高密度嵌入式非易失性存储器、阻变存储器（ReRAM）、磁变存储器（Magnetoresistive Random Access Memory, MRAM）、铁电存储器（Ferroelectric Random Access Memory, FRAM）、相变存储器（Phase Change Memory, PCM）、石墨烯存储器等。易失性存储器可包括随机存取存储器（Random Access Memory, RAM）或外部高速缓冲存储器等。作为说明而非局限，RAM 可以是多种形式，比如静态随机存取存储器（Static Random Access Memory, SRAM）或动态随机存取存储器（Dynamic Random Access Memory, DRAM）等。本申请所提供的各实施例中所涉及的数据库可包括关系型数据库和非关系型数据库中至少一种。非关系型数据库可包括基于区块链的分布式数据库等，不限于此。本申请所提供的各实施例中所涉及的处理器可为通用处理器、中央处理器、图形处理器、数字信号处理器、可编程逻辑器、基于量子计算的数据处理逻辑器等，不限于此。

以上实施例的各技术特征可以进行任意的组合，为使描述简洁，未对上述实施例中的各个技术特征所有可能的组合都进行描述，然而，只要这些技术特征的组合不存在矛盾，都应当认为是本说明书记载的范围。

以上所述实施例仅表达了本申请的几种实施方式，其描述较为具体和详细，但并不能因此而理解为对发明专利范围的限制。应当指出的是，对于本领域的普通技术人员来说，在不脱离本申请构思的前提下，还可以做出若干变形和改进，这些都属于本申请的保护范围。因此，本申请专利的保护范围应以所附权利要求为准。

权利要求

1、一种子应用页面处理方法，由计算机设备执行，包括：

当监测到请求获取子应用页面的事件，获取所述子应用页面的页面代码，所述子应用页面是指在子应用中显示的页面；

对所述页面代码进行解析，获得页面解析结果，遍历所述页面解析结果中代码集合，创建所述子应用页面相应的页面节点树，所述页面节点树是基于所述子应用页面的页面节点而生成的树状结构；

创建用于转换所述页面节点树的第一实例对象，建立所述第一实例对象和所述子应用页面的绑定关系；

在建立所述绑定关系的情况下，使用所述第一实例对象，将所述页面节点树转换为接口调用的形式，获得所述页面节点树相应的接口调用信息；及

基于所述接口调用信息进行页面绘制，生成所述子应用页面。

2、根据权利要求1所述的方法，所述创建用于转换所述页面节点树的第一实例对象，建立所述第一实例对象和所述子应用页面的绑定关系包括：

基于预配置的接口协议的协议模式，获取用于转换所述页面节点树的第一接口类，所述第一接口类包括转换所述页面节点树的逻辑方法；

基于所述第一接口类，创建用于转换所述页面节点树的第一实例对象；及

将所述第一实例对象作为创建参数，创建与所述子应用页面相应的根组件实例，以建立所述第一实例对象和所述子应用页面的绑定关系。

3、根据权利要求2所述的方法，所述基于所述第一接口类，创建用于转换所述页面节点树的第一实例对象包括：

通过基于类实现接口的第一关键字，创建与所述第一接口类相应的第一实现类，所述第一实现类用于实现所述第一接口类中的转换页面节点树的逻辑方法；及

通过基于类创建实例对象的第二关键字，对所述第一实现类进行实例化，创建与所述第一实现类相应的第一实例对象。

4、根据权利要求1-3任意一项所述的方法，所述使用所述第一实例对象，将所述页面节点树转换为接口调用的形式，获得所述页面节点树相应的接口调用信息包括：

基于所述第一实例对象所提供的第一接口配置信息，从所述页面节点树的根节点开始，依次遍历所述页面节点树中的页面节点，分别确定各所述页面节点各自相应的第一调用信息，并获得所述页面节点树的页面配置信息相应的第二调用信息；及

基于所述第一调用信息和所述第二调用信息，获得所述页面节点树相应的接口调用信息。

5、根据权利要求4所述的方法，所述基于所述第一实例对象所提供的第一接口配置信息，从所述页面节点树的根节点开始，依次遍历所述页面节点树中的页面节点，分别确定各所述页面节点各自相应的第一调用信息包括：

针对于所述页面节点树中的每一个页面节点，从所述第一实例对象所提供的第一接口配置信息中，确定与所针对的页面节点的页面节点类型相匹配的第一接口信息，基于所述第一接口信息确定所述所针对的页面节点相应的第一调用信息。

6、根据权利要求5所述的方法，所述基于所述第一接口信息确定所述所针对的页面节点相应的第一调用信息包括：

基于所述第一接口信息获取第二实例对象，所述第二实例对象提供对所述页面节点树进行转换的第二接口配置信息；

使用所述第二实例对象所提供的第二接口配置信息，获得所述所针对的页面节点相应的第二接口信息；及

根据所述第一接口信息和所述第二接口信息，获得所述所针对的页面节点相应的第一调用信息。

7、根据权利要求6所述的方法，所述方法还包括：

通过所述第一实例对象，创建用于转换页面节点的第二接口类，所述第二接口类包括转换页面节点的逻辑方法；

通过基于类实现接口的第一关键字，创建与所述第二接口类相应的第二实现类，所述第二实现类用于实现所述第二接口类中的转换页面节点的逻辑方法；及

通过基于类创建实例对象的第二关键字，对所述第二实现类进行实例化，创建与所述第二实现类相应的第二实例对象。

8、根据权利要求6所述的方法，所述使用所述第二实例对象所提供的第二接口配置信息，获得与所述所针对的页面节点相应的第二接口信息包括：

基于所述所针对的页面节点的节点配置信息，从所述第二实例对象所提供的第二接口配置信息中，获得与所述所针对的页面节点相应的第二接口信息。

9、根据权利要求6所述的方法，所述根据所述第一接口信息和所述第二接口信息，获得所述所针对的页面节点相应的第一调用信息包括：

当所述所针对的页面节点为组件节点，基于所述第二接口信息获得预创建的第三实例对象，所述第三实例对象是指用于转换所述组件节点的实例对象；

使用所述第三实例对象，获得所述组件节点相应的第三接口信息；及

根据所述组件节点相应的第一接口信息、第二接口信息以及第三接口信息，获得所述组件节点相应的第一调用信息。

10、根据权利要求9所述的方法，所述使用所述第三实例对象，获得所述组件节点相应的第三接口信息包括：

基于所述第三实例对象所提供的第三接口配置信息，从所述组件节点的根节点开始，依次遍历所述组件节点中子节点，分别确定各所述子节点各自相应的子节点调用信息；及

基于各所述子节点各自相应的子节点调用信息，获得所述组件节点相应的第三接口信息。

11、根据权利要求9所述的方法，所述基于所述第三实例对象所提供的第三接口配置信息，从所述组件节点的根节点开始，依次遍历所述组件节点中子节点，分别确定各所述子节点各自相应的子节点调用信息包括：

针对于所述组件节点中每一个子节点，从所述第三实例对象所提供的第三接口配置信息中，确定与所针对的子节点的子节点类型相匹配的子节点接口信息；及

基于所述子节点接口信息获取第二实例对象，使用所述第二实例对象，获得所述所针对的子节点相应的子节点调用信息。

12、根据权利要求11所述的方法，所述使用所述第二实例对象，获得所述所针对的子节点相应的子节点调用信息包括：

基于所述所针对的子节点的子节点配置信息，从所述第二实例对象所提供的第二接口配置信息中，获取与所述所针对的子节点相应的子节点调用信息。

13、根据权利要求1至12任意一项所述的方法，所述对所述页面代码进行解析，获得页面解析结果包括：

确定编写所述页面代码所使用的标签语言，通过所述标签语言相应的标签语言编译器，对所述页面代码进行编译，得到页面解析结果。

14、一种子应用页面处理装置，包括：

代码获取模块，用于当监测到请求获取子应用页面的事件，获取所述子应用页面的页面代码，所述子应用页面是指在子应用中显示的页面；

节点树生成模块，用于对所述页面代码进行解析，获得页面解析结果，遍历所述页面解析结果中代码集合，创建所述子应用页面相应的页面节点树，所述页面节点树是基于所

述子应用页面的页面节点而生成的树状结构；

页面初始化模块，用于创建用于转换所述页面节点树的第一实例对象，建立所述第一实例对象和所述子应用页面的绑定关系；

节点树转换模块，用于在建立所述绑定关系的情况下，使用所述第一实例对象，将所述页面节点树转换为接口调用的形式，获得所述页面节点树相应的接口调用信息；

页面绘制模块，用于基于所述接口调用信息进行页面绘制，生成所述子应用页面。

15、一种计算机设备，包括存储器和处理器，所述存储器存储有计算机可读指令，其特征在于，所述处理器执行所述计算机可读指令时实现权利要求 1 至 13 中任一项所述的方法的步骤。

16、一种计算机可读存储介质，其上存储有计算机可读指令，其特征在于，所述计算机可读指令被处理器执行时实现权利要求 1 至 13 中任一项所述的方法的步骤。

17、一种计算机程序产品，包括计算机可读指令，其特征在于，该计算机可读指令被处理器执行时实现权利要求 1 至 13 中任一项所述的方法的步骤。