(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 113805871 B
(45)授权公告日 2023.08.15

(21)申请号 202111131451.5

(22)申请日 2021.09.26

(65)同一申请的已公布的文献号
申请公布号 CN 113805871 A

(43)申请公布日 2021.12.17

(73)专利权人 平安国际智慧城市科技股份有限公司
地址 518000 广东省深圳市前海深港合作区妈湾兴海大道3048号前海自贸大厦1-34层

(72)发明人 梁雪超

(74)专利代理机构 北京中强智尚知识产权代理有限公司 11448
专利代理师 刘敏

(51)Int.Cl.
G06F 8/34(2018.01)
G06F 8/38(2018.01)

(56)对比文件
CN 109766503 A,2019.05.17
CN 111142871 A,2020.05.12
CN 112416363 A,2021.02.26
CN 112882817 A,2021.06.01
EP 2246782 A1,2010.11.03
US 2018357055 A1,2018.12.13

审查员 林荔琳

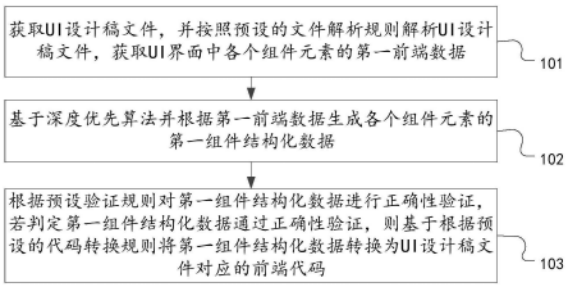
权利要求书3页 说明书9页 附图3页

(54)发明名称

前端代码的生成方法、装置及计算机设备

(57)摘要

本申请公开了一种前端代码的生成方法、装置及计算机设备，涉及计算机技术领域，可解决目前前端代码的生成方式存在重复劳动部分，导致代码生成效率低、工作量大的技术问题。包括：获取UI设计稿文件，并按照预设的文件解析规则解析所述UI设计稿文件，获取UI界面中各个组件元素的第一前端数据；基于深度优先算法并根据所述第一前端数据生成各个所述组件元素的第一组件结构化数据；根据预设验证规则对所述第一组件结构化数据进行正确性验证，若判定所述第一组件结构化数据通过所述正确性验证，则基于预设的代码转换规则将所述第一组件结构化数据转换为所述UI设计稿文件对应的前端代码。本申请适用于前端代码的自动化生成。



CN 113805871 B

1.一种前端代码的生成方法,其特征在于,包括:

获取UI设计稿文件,并按照预设的文件解析规则解析所述UI设计稿文件,获取UI界面中各个组件元素的第一前端数据;

基于深度优先算法并根据所述第一前端数据生成各个所述组件元素的第一组件结构化数据,包括:

按照深度优先搜索算法递归遍历所述第一前端数据中各个组件元素的所有路径,计算路径中每条边的权值之和,得出权值之和最大的路径;

将所述权值之和最大的路径作为主干路径,倒置所有路径的时序关系,生成多叉树结构;

按照自顶向下的顺序逐层读取所述多叉树各节点,将对象树序列化成JSON格式,得到第一组件结构化数据;

根据预设验证规则对所述第一组件结构化数据进行正确性验证,若判定所述第一组件结构化数据通过所述正确性验证,则基于预设的代码转换规则将所述第一组件结构化数据转换为所述UI设计稿文件对应的前端代码,包括:

解析所述第一组件结构化数据,以获取所述UI界面中至少一组件元素的组件信息,以及任意两个组件元素的组件关联关系,其中,所述组件信息包括:组件唯一标识、组件类型、组件元素在所述UI界面中的坐标、尺寸及外观属性;

根据所述组件唯一标识以及所述组件类型在预设代码描述库中筛选与组件元素匹配的第一代码描述;

利用逻辑配置模板生成所述组件关联关系对应的第二代码描述;

将所述第一代码描述以及所述第二代码描述通过代码编译器编译生成所述UI设计稿文件对应的前端代码。

2.根据权利要求1所述的方法,其特征在于,所述利用逻辑配置模板生成所述组件关联关系对应的第二代码描述,包括:

在第一预设映射表中提取与所述组件关联关系对应的逻辑配置模板,其中,所述第一预设映射表包括所述组件关联关系与所述逻辑配置模板的对应关系;

根据所确定出的逻辑配置模板生成对应的代码描述,并将所述组件元素在所述UI界面中的坐标、所述尺寸及外观属性添加至所述代码描述中,得到所述组件关联关系对应的第二代码描述。

3.根据权利要求1所述的方法,其特征在于,所述将所述第一代码描述以及所述第二代码描述通过代码编译器编译生成所述UI设计稿文件对应的前端代码,包括:

按照预设组合规则组合所述第一代码描述以及所述第二代码描述,得到所述UI设计稿文件对应UI界面的目标代码描述;

将所述目标代码描述载入不同的代码编译器,生成对应不同平台的前端代码。

4.根据权利要求1所述的方法,其特征在于,所述第一前端数据包括画布元素、控件信息、用户通过控件修改的信息、图层关系、布局信息、图层样式信息以及组件配置信息;

所述按照预设的文件解析规则解析所述UI设计稿文件,获取UI界面中各个组件元素的第一前端数据,包括:

读取并解析所述UI设计稿文件,提取画布元素和控件信息;

利用组件元素的组件唯一标识在所述控件信息中查取控件节点，并对所述控件节点进行还原处理，还原用户通过控件修改的信息；

根据所述画布元素中的节点布局尺寸确定图层关系和布局信息；

将所述画布元素中的节点样式描述按照CSS样式规则转化为图层样式信息；

根据所述画布元素中的节点名称确定所需要生成代码的元素组件类型，并根据第二预设映射表以及所述元素组件类型提取所述组件配置信息，所述第二预设映射表中创建有所述元素组件类型与所述组件配置信息间的映射关系。

5.根据权利要求1所述的方法，其特征在于，所述方法还包括：

将基于所述第一组件结构化数据转换的所述前端代码加载到用户配置界面；

若接收到所述用户配置页面上传的配置修改内容，则根据所述配置修改内容将所述第一前端数据调整为第二前端数据；

基于深度优先算法并根据所述第二前端数据生成各个所述组件元素的第二组件结构化数据；

根据预设的代码转换规则将所述第二组件结构化数据转换为所述UI设计稿文件对应的前端代码。

6.一种前端代码的生成装置，其特征在于，包括：

解析模块，用于获取UI设计稿文件，并按照预设的文件解析规则解析所述UI设计稿文件，获取UI界面中各个组件元素的第一前端数据；

第一生成模块，用于基于深度优先算法并根据所述第一前端数据生成各个所述组件元素的第一组件结构化数据，包括：

按照深度优先搜索算法递归遍历所述第一前端数据中各个组件元素的所有路径，计算路径中每条边的权值之和，得出权值之和最大的路径；

将所述权值之和最大的路径作为主干路径，倒置所有路径的时序关系，生成多叉树结构；

按照自顶向下的顺序逐层读取所述多叉树各节点，将对象树序列化成JSON格式，得到第一组件结构化数据；

第一转换模块，用于根据预设验证规则对所述第一组件结构化数据进行正确性验证，若判定所述第一组件结构化数据通过所述正确性验证，则基于预设的代码转换规则将所述第一组件结构化数据转换为所述UI设计稿文件对应的前端代码，包括：

解析所述第一组件结构化数据，以获取所述UI界面中至少一组件元素的组件信息，以及任意两个组件元素的组件关联关系，其中，所述组件信息包括：组件唯一标识、组件类型、组件元素在所述UI界面中的坐标、尺寸及外观属性；

根据所述组件唯一标识以及所述组件类型在预设代码描述库中筛选与组件元素匹配的第一代码描述；

利用逻辑配置模板生成所述组件关联关系对应的第二代码描述；

将所述第一代码描述以及所述第二代码描述通过代码编译器编译生成所述UI设计稿文件对应的前端代码。

7.一种存储介质，其上存储有计算机程序，其特征在于，所述程序被处理器执行时实现权利要求1至5中任一项所述的前端代码的生成方法。

8.一种计算机设备，包括存储介质、处理器及存储在存储介质上并可在处理器上运行的计算机程序，其特征在于，所述处理器执行所述程序时实现权利要求1至5中任一项所述的前端代码的生成方法。

前端代码的生成方法、装置及计算机设备

技术领域

[0001] 本申请涉及计算机技术领域，尤其涉及到一种前端代码的生成方法、装置及计算机设备。

背景技术

[0002] 随着5G技术的逐渐发展和成熟，未来大量的应用场景都需要前端进行呈现，包括智能家居，可穿戴设备等领域将带来大量的前端开发需求，如何提升开发效率和降低开发门槛变得尤为重要和紧迫。低代码平台技术正逐渐在IT和商业市场中获得关注，通过使用可视化模型和图形设计技术来构建软件，低代码平台可减少70-90％的开发时间和研发成本。

[0003] 当前，代码平台主要通过拖拽和配置前端组件的方式构建UI界面并生成代码，从效率相比直接开发大很多，但是依然存在重复劳动部分，比如组件的排布、文字录入等重复性工作，导致代码生成效率较低。

发明内容

[0004] 有鉴于此，本申请提供了一种前端代码的生成方法、装置及计算机设备，可用于解决目前前端代码的生成方式存在重复劳动部分，导致代码生成效率低的技术问题。

[0005] 根据本申请的一个方面，提供了一种前端代码的生成方法，该方法包括：

[0006] 获取UI设计稿文件，并按照预设的文件解析规则解析所述UI设计稿文件，获取UI界面中各个组件元素的第一前端数据；

[0007] 基于深度优先算法并根据所述第一前端数据生成各个所述组件元素的第一组件结构化数据；

[0008] 根据预设验证规则对所述第一组件结构化数据进行正确性验证，若判定所述第一组件结构化数据通过所述正确性验证，则基于预设的代码转换规则将所述第一组件结构化数据转换为所述UI设计稿文件对应的前端代码。

[0009] 根据本申请的另一个方面，提供了一种前端代码的生成装置，该装置包括：

[0010] 解析模块，用于获取UI设计稿文件，并按照预设的文件解析规则解析所述UI设计稿文件，获取UI界面中各个组件元素的第一前端数据；

[0011] 第一生成模块，用于基于深度优先算法并根据所述第一前端数据生成各个所述组件元素的第一组件结构化数据；

[0012] 第一转换模块，用于根据预设验证规则对所述第一组件结构化数据进行正确性验证，若判定所述第一组件结构化数据通过所述正确性验证，则基于预设的代码转换规则将所述第一组件结构化数据转换为所述UI设计稿文件对应的前端代码。

[0013] 根据本申请的又一个方面，提供了一种存储介质，其上存储有计算机程序，所述程序被处理器执行时实现上述前端代码的生成方法。

[0014] 根据本申请的再一个方面，提供了一种计算机设备，包括存储介质、处理器及存储

在存储介质上并可在处理器上运行的计算机程序,所述处理器执行所述程序时实现上述前端代码的生成方法。

[0015] 借由上述技术方案,本申请提供的一种前端代码的生成方法、装置及计算机设备,与目前前端代码的生成方式相比,本申请可首先获取UI设计稿文件,并按照预设的文件解析规则解析UI设计稿文件,获取UI界面中各个组件元素的第一前端数据;进而基于深度优先算法并根据第一前端数据生成各个组件元素的第一组件结构化数据;之后可根据预设验证规则对第一组件结构化数据进行正确性验证,在判定第一组件结构化数据通过正确性验证后,可进一步根据预设的代码转换规则将第一组件结构化数据转换为UI设计稿文件对应的前端代码。本方案通过解析UI设计稿结构信息并生成代码,相较业内流行的拖拽组件方式,拥有更高的效率且更加自动化,同时具备更低的平台学习成本;而相较于方兴未艾的AI视觉识别模式,又具备更高的识别准确度和样式还原度。

[0016] 上述说明仅是本申请技术方案的概述,为了能够更清楚了解本申请的技术手段,而可依照说明书的内容予以实施,并且为了让本申请的上述和其它目的、特征和优点能够更明显易懂,以下特举本申请的具体实施方式。

附图说明

[0017] 此处所说明的附图用来提供对本申请的进一步理解,构成本申请的一部分,本申请的示意性实施例及其说明用于解释本申请,并不构成对本地申请的不当限定。在附图中:

[0018] 图1示出了本申请实施例提供的一种前端代码的生成方法的流程示意图;

[0019] 图2示出了本申请实施例提供的另一种前端代码的生成方法的流程示意图;

[0020] 图3示出了本申请实施例提供的一种前端代码的生成装置的结构示意图;

[0021] 图4示出了本申请实施例提供的另一种前端代码的生成装置的结构示意图。

具体实施方式

[0022] 下文将参考附图并结合实施例来详细说明本申请。需要说明的是,在不冲突的情况下,本申请中的实施例及实施例中的特征可以相互结合。

[0023] 针对目前前端代码的生成方式存在重复劳动部分,导致代码生成效率低的技术问题,本申请提供了一种前端代码的生成方法,如图1所示,该方法包括:

[0024] 101、获取UI设计稿文件,并按照预设的文件解析规则解析UI设计稿文件,获取UI界面中各个组件元素的第一前端数据。

[0025] 其中,UI设计稿文件具体可为Sketch文件设计文档(视图结构),视觉稿的创建和编辑软件为Sketch;预设的文件解析规则包括利用Node.js在UI设计稿文件中解析出画布元素和控件信息,以及对控件节点进行还原处理,还原用户通过控件修改的信息,以及对图层关系、布局信信息和组件配置信息的解析处理;第一前端数据可包括画布元素、控件信息、用户通过控件修改的信息、图层关系、布局信息、图层样式信息以及组件配置信息等,画布元素具体可包括节点嵌套结构以及节点元素信息,节点元素信息可包括节点名称、子节点集合、节点布局尺寸、节点类型、节点样式描述、节点元素唯一标识、文本字段内容等;控件信息具体可为控件集合,在控件集合中包含有各个控件元素以及对应的控件实例。

[0026] 对于本实施例的执行主体可为前端代码的生成装置,可配置在客户端侧或服务端

侧,可首先获取UI设计稿文件,并按照预设的文件解析规则解析UI设计稿文件,获取UI界面中各个组件元素的第一前端数据;进而基于深度优先算法并根据第一前端数据生成各个组件元素的第一组件结构化数据;最后可根据预设的代码转换规则将第一组件结构化数据转换为UI设计稿文件对应的前端代码。

[0027] 102、基于深度优先算法并根据第一前端数据生成各个组件元素的第一组件结构化数据。

[0028] 其中,深度优先算法(Depth-First-Search,DFS)是一种用于遍历或搜索树或图的算法,沿着树的深度遍历树的节点,尽可能深的搜索树的分支。在本实施例中,可应用于深度优先算法循环遍历第一前端数据中的对象树节点,通过递归遍历每个组件节点的所有路径,筛选出主干路径,并进一步生成以主干路径中最终节点为根节点的多叉树结构,并按照自顶向下的顺序逐层读取多叉树各节点,即可将对象树序列化成JSON格式,进一步得到JSON数据,即第一组件结构化数据。其中,多叉树结构具体可为MBOM多叉树结构。

[0029] 103、根据预设验证规则对第一组件结构化数据进行正确性验证,若判定第一组件结构化数据通过正确性验证,则基于根据预设的代码转换规则将第一组件结构化数据转换为UI设计稿文件对应的前端代码。

[0030] 对于本实施例,在基于实施例步骤102生成各个组件元素的第一组件结构化数据后,可首先对所生成的第一组件结构化数据进行正确性验证,在判定第一组件结构化数据通过正确性验证时,再进一步根据预设的代码转换规则将第一组件结构化数据转换为UI设计稿文件对应的前端代码;反之,若判定第一组件结构化数据未通过正确性验证,则可进一步重复执行实施例步骤101至102。通过对第一组件结构化数据首先进行正确性验证,可保证第一组件结构化数据的正确性,进而能够基于正确的第一组件结构化数据准确生成UI设计稿文件对应的前端代码。

[0031] 在具体的应用场景中,作为一种可选方式,在根据预设验证规则对第一组件结构化数据进行正确性验证时,实施例步骤具体可以包括:预置一个组件结构化数据模板,组件结构化数据模板用于规定反映待测组件结构化数据中任一节点的位置的节点位置参数,还用于规定待测组件结构化数据与期望结果数据之间的验证关系;根据节点位置参数和验证关系生成包含测试认证点的集合文件,测试认证点包括节点位置参数、验证关系和期望结果数据;根据集合文件对待测组件结构化数据进行验证,并返回验证结果。

[0032] 其中,组件结构化数据模板可包括组件结构化数据访问模板和组件结构化数据验证模板,组件结构化数据访问模板用于规定节点位置参数,即对意欲方位的节点位置进行定位,组件结构化数据验证模板用于规定待测组件结构化数据与期望结果数据之间的验证关系,如相等、包含等。节点位置参数包括表示层级关系的层级标示和表示数组的数组标示,验证关系包括包含关系、键值大小关系和数组大小关系。相应的,在根据集合文件对待测组件结构化数据进行验证时,可包括:根据集合文件从待测组件结构化数据中获取待测节点的实际结果数据;从集合文件中获取期望结果数据;根据验证关系对实际结果数据和期望结果数据进行验证,并返回验证结果。

[0033] 进一步地,若根据验证结果判定第一组件结构化数据通过正确性验证,则可将第一组件结构化数据转换为UI设计稿文件对应的前端代码。相应的,作为一种可选方式,可获取预先定义好的代码模板,代码模板如字典库-提供代码转换的规则;进而可根据匹配的代

码模板对第一组件结构化数据进行转换处理，得到UI设计稿文件对应的前端代码。作为另一种可选方式，可根据DSL编译器依据第一组件结构化数据生成UI设计稿文件对应的前端代码，其中，DSL编译器中规定怎样描述当前页面/组件的完整结构树、引用的样式还有业务相关的数据导出规则、交互规则，DSL编译器具体以插件形式实现固定DSL输入，并在插件内部处理，以输出对应语言环境下的前端代码。

[0034] 通过本实施例中前端代码的生成方法，可首先获取UI设计稿文件，并按照预设的文件解析规则解析UI设计稿文件，获取UI界面中各个组件元素的第一前端数据；进而基于深度优先算法并根据第一前端数据生成各个组件元素的第一组件结构化数据；之后可根据预设验证规则对第一组件结构化数据进行正确性验证，在判定第一组件结构化数据通过正确性验证后，可进一步根据预设的代码转换规则将第一组件结构化数据转换为UI设计稿文件对应的前端代码。本方案通过解析UI设计稿结构信息并生成代码，相较业内流行的拖拽组件方式，拥有更高的效率且更加自动化，同时具备更低的平台学习成本；而相较于方兴未艾的AI视觉识别模式，又具备更高的识别准确度和样式还原度。

[0035] 进一步的，作为上述实施例具体实施方式的细化和扩展，为了完整说明本实施例中的具体实施过程，提供了另一种前端代码的生成方法，如图2所示，该方法包括：

[0036] 201、获取UI设计稿文件，并按照预设的文件解析规则解析UI设计稿文件，获取UI界面中各个组件元素的第一前端数据。

[0037] 对于本实施例，作为一种可选方式，第一前端数据包括画布元素、控件信息、用户通过控件修改的信息、图层关系、布局信息、图层样式信息以及组件配置信息。相应的，在按照预设的文件解析规则解析UI设计稿文件，获取UI界面中各个组件元素的第一前端数据时，实施例步骤201具体可以包括：读取并解析UI设计稿文件，提取画布元素和控件信息；利用组件元素的组件唯一标识在控件信息中查取控件节点，并对控件节点进行还原处理，还原用户通过控件修改的信息；根据画布元素中的节点布局尺寸确定图层关系和布局信息；将画布元素中的节点样式描述按照CSS样式规则转化为图层样式信息；根据画布元素中的节点名称确定所需要生成代码的元素组件类型，并根据第二预设映射表以及元组组件类型提取组件配置信息，其中，第二预设映射表中创建有元组组件类型与组件配置信息间的映射关系。

[0038] 在具体的应用场景中，具体可通过Node.js读取并解析UI设计稿文件，提取画布元素和控件信息。相应的，具体可解析出节点名称、子节点集合、节点布局尺寸、节点类型、节点样式描述、节点元素唯一标识、文本字段内容等画布元素，以及包含各个控件元素以及对应的控件实例的控件信息。相应的，节点元素可包括普通节点元素和控件元素，在利用组件元素的组件唯一标识在控件信息中查取控件节点，并对控件节点进行还原处理，还原用户通过控件修改的信息时，可通过节点元素唯一标识SymbalID在控件集合中查找控件实例并替换覆盖字段Override，进一步还原用户通过控件修改的信息。对于本实施例，控件元素为UI设计稿文件对应UI图下的组件元素，由于控件模版中包含可编辑信息，如：atrributedString，如果用户对控件元素进行了编辑，则节点中会包含相应的覆盖字段Override，需要从Override中的overrideName中截取id字段并查找到相应的节点，还原用户通过控件修改的信息。具体的，针对控件类型的元素，其对应的元素类型为控件实例，并包含SymbalID字段时，具体可通过SymbalID字段匹配控件信息，找到其控件实例类型并替

换,对控件元素进行还原处理,还原用户通过控件修改的信息,进一步将控件元素还原替换为普通节点元素。

[0039] 相应的,在根据画布元素中的节点布局尺寸确定图层关系和布局信息时,具体可根据节点布局尺寸字段中的位置信息(如x,y,width,height),计算图层关系和布局信息。对于本实施例,由于在UI设计稿文件中,所有节点都组成树形结构数据,子节点包含于父节点的layers字段的集合中。且layers中元素的顺序是根据用户绘制元素的先后顺序插入的,并不能反应时间视图中元素的左右或上下关系,故为了得到正确的视图布局,需要通过元素位置和尺寸进行计算,重新排列和生成布局图层。在具体的应用场景中,在计算图层关系时,由于子节点包含于父节点的layers字段的集合中,故可根据位置信息(如x,y,width,height)确定各个节点的尺寸大小以及位置,以及各个节点之间的位置包含关系,进一步得到父节点以及对应子节点之间的图层关系layers,其中,在图层关系layers中,子节点的图层包含于父节点。相应的,在计算布局时,可先通过比较集合中元素的x,y值判断元素的布局是横向或纵向布局(如元素对应的x值不同,y相同,则可判断元素的布局是横向布局;如元素对应的x值相同,y不同,则可判断元素的布局是纵向布局);确定之后再对集合元素进行排序,进一步得到布局信息(如横向布局只要比较x值大小即可知道元素的左右顺序,纵向布局只要比较y值大小即可知道元素的上下顺序)。

[0040] 在具体的应用场景中,在将画布元素中的节点样式描述按照CSS样式规则转化为图层样式信息时,具体可在提取出节点样式描述对应的样式字段后,将样式字段中的blur,borders,shadows,innerShadows,fills等参数分别按照web中的CSS样式规则进行转换,进而转换得到图层样式在CSS中的样式描述信息。相应的,在根据第二预设映射表以及画布元素中的节点名称提取所需要生成代码的元素组件类型,并根据元组组件类型提取组件配置信息时,具体可预先根据前端UI库,建立UI控件库,将UI控件库中各个控件名与UI库中的组件名称一一对应的设置,并建立第二预设映射表。对于本实施例,可根据第二预设映射表以及控件名从UI设计稿文件映射出所需要生成代码的元素组件,得到UI设计稿文件所包含所有组件的名称,并保存在comp字段;进而可根据元素组件类型,提取组件配置信息,如:文本、table组件的columns配置等。

[0041] 202、基于深度优先算法并根据第一前端数据生成各个组件元素的第一组件结构化数据。

[0042] 对于本实施例,在获取到UI界面的第一前端数据后,可进一步对第一前端数据进行节点冗余信息处理,滤除在后续实施过程中不会再应用到的各节点冗余字段,如节点布局尺寸、节点类型等;在冗余信息处理完成后,可利用深度优先算法,循环遍历对象树节点,最后将对象序列化成JSON格式,得到第一组件结构化数据,即JSON数据。

[0043] 对于本实施例,作为一种可选方式,实施例步骤202具体可以包括:按照深度优先搜索算法递归遍历第一前端数据中各个组件元素的所有路径,计算路径中每条边的权值之和,得出权值之和最大的路径;将权值之和最大的路径作为主干路径,倒置所有路径的时序关系,生成多叉树结构;按照自顶向下的顺序逐层读取多叉树各节点,将对象树序列化成JSON格式,得到第一组件结构化数据。其中,多叉树结构具体可为MBOM多叉树结构。

[0044] 203、根据预设验证规则对第一组件结构化数据进行正确性验证,若判定第一组件结构化数据通过正确性验证,则解析第一组件结构化数据,以获取UI界面中至少一组件元

素的组件信息,以及任意两个组件元素的组件关联关系。

[0045] 其中,组件信息包括:组件唯一标识、组件类型、组件元素在UI界面中的坐标、尺寸及外观属性。

[0046] 对于本实施例,在对第一组件结构化数据进行正确性验证时,具体可参见实施例步骤103中的相关描述,在此不再赘述。在解析第一组件结构化数据,获取UI界面中至少一组件元素的组件信息,以及任意两个组件元素的组件关联关系时,具体可利用现有解析工具在第一组件结构化数据中解析出至少一组件元素的组件信息,以提取出UI界面中所包含的所有组件元素,以及每个组件元素在UI界面中的位置信息,以及每个组件元素的必要参数;此外还可提取出任意两个组件元素之间存在的组件关联关系,组件关联关系可对应组件之间的数据交互或事件交互等,如弹窗交互、数据查询交互等。

[0047] 204、根据组件信息中的组件唯一标识以及组件类型在预设代码描述库中筛选与组件元素匹配的第一代码描述。

[0048] 在具体的应用场景中,可预先针对各个组件类型下的组件元素生成初始代码描述,并创建组件唯一标识和/或组件类型与初始代码描述的映射关系,并存储至预设代码描述库中。对于本实施例,在基于实施例步骤203从第一组件结构化数据中解析出组件唯一标识、组件类型后,可进一步基于解析出的组件唯一标识和/或组件类型在预设代码描述库中直接提取与组件元素匹配的初始代码描述,作为第一代码描述。

[0049] 205、利用逻辑配置模板生成组件关联关系对应的第二代码描述。

[0050] 对于本实施例,作为一种可选方式,在利用逻辑配置模板生成组件关联关系对应的第二代码描述时,可考虑模版逻辑配置,根据模版逻辑配置生成组件关联逻辑代码。具体可通过识别组件之间的关联关系,进一步根据组件之间的组件关联关系,利用逻辑模块生成相应的组件关联逻辑代码,即第二代码描述。其中,逻辑模块中配置有预先定义好的逻辑配置模板,逻辑配置模板提供初始代码描述的生成规则,可用于通过对逻辑配置模板的调用,确定与组件关联关系匹配的逻辑配置模板,进而利用匹配的逻辑配置模板生成组件关联关系对应的第二代码描述。相应的,实施例步骤205具体可以包括:在第一预设映射表中提取与组件关联关系对应的逻辑配置模板,其中,第一预设映射表包括组件关联关系与逻辑配置模板的对应关系;根据所确定出的逻辑配置模板生成对应的代码描述,并将组件元素在UI界面中的坐标、尺寸及外观属性添加至代码描述中,得到组件关联关系对应的第二代码描述。

[0051] 206、将第一代码描述以及第二代码描述通过代码编译器编译生成UI设计稿文件对应的前端代码。

[0052] 对于本实施例,作为一种可选方式,实施例步骤206具体可以包括:按照预设组合规则组合第一代码描述以及第二代码描述,得到UI设计稿文件对应UI界面的目标代码描述;将目标代码描述载入不同的代码编译器,生成对应不同平台的前端代码。其中预设组合规则具体可为在预设的组合代码库中筛选与组件关联关系匹配的组合代码,并按照组合代码,实现对第一代码描述以及第二代码描述的组合处理。

[0053] 在具体的应用场景中,为了满足用户的个性化配置,实施例步骤具体还可以包括:将基于第一组件结构化数据转换的前端代码加载到用户配置界面;若接收到用户配置页面上传的配置修改内容,则根据配置修改内容将第一前端数据调整为第二前端数据;基于深

度优先算法并根据第二前端数据生成各个组件元素的第二组件结构化数据;根据预设的代码转换规则将第二组件结构化数据转换为UI设计稿文件对应的前端代码。其中,在将前端初始化代码加载到用户配置界面时,用户可进行个性化配置,具体的,用户可点击生成页面的页面块;点击页面块通过现有方法html getAttribute方法可以获取生成代码过程中放在父级头部的所有子节点数据;并且在用户可操作区域渲染操作区--渲染操作区使用提前定义好的操作区代码模板渲染操作区域;用户修改区域参数-可以获得当前代码块最新的数据参数。进一步的,可根据配置修改内容将第一前端数据调整为第二前端数据;基于深度优先算法并根据第二前端数据生成各个组件元素的第二组件结构化数据;根据预设的代码转换规则将第二组件结构化数据转换为UI设计稿文件对应的前端代码。其中,依据第二前端数据生成UI设计稿文件对应前端代码的具体实现方式可参考实施例步骤202至206中的相关描述,在此不再赘述。

[0054] 借由上述前端代码的生成方法,可首先获取UI设计稿文件,并按照预设的文件解析规则解析UI设计稿文件,获取UI界面中各个组件元素的第一前端数据;进而基于深度优先算法并根据第一前端数据生成各个组件元素的第一组件结构化数据;之后可根据预设验证规则对第一组件结构化数据进行正确性验证,在判定第一组件结构化数据通过正确性验证后,可进一步根据预设的代码转换规则将第一组件结构化数据转换为UI设计稿文件对应的前端代码。本方案通过解析UI设计稿结构信息并生成代码,相较业内流行的拖拽组件方式,拥有更高的效率且更加自动化,同时具备更低的平台学习成本;而相较于方兴未艾的AI视觉识别模式,又具备更高的识别准确度和样式还原度。

[0055] 进一步的,作为图1和图2所示方法的具体实现,本申请实施例提供了一种前端代码的生成装置,如图3所示,该装置包括:解析模块31、第一生成模块32、第一转换模块33;

[0056] 解析模块31,可用于获取UI设计稿文件,并按照预设的文件解析规则解析UI设计稿文件,获取UI界面中各个组件元素的第一前端数据;

[0057] 第一生成模块32,可用于基于深度优先算法并根据第一前端数据生成各个组件元素的第一组件结构化数据;

[0058] 第一转换模块33,可用于根据预设验证规则对第一组件结构化数据进行正确性验证,若判定第一组件结构化数据通过正确性验证,则基于预设的代码转换规则将第一组件结构化数据转换为UI设计稿文件对应的前端代码。

[0059] 在具体的应用场景中,在根据预设的代码转换规则将第一组件结构化数据转换为UI设计稿文件对应的前端代码时,第一转换模块33,具体可用于解析第一组件结构化数据,以获取UI界面中至少一组件元素的组件信息,以及任意两个组件元素的组件关联关系,其中,组件信息包括:组件唯一标识、组件类型、组件元素在UI界面中的坐标、尺寸及外观属性;根据组件唯一标识以及组件类型在预设代码描述库中筛选与组件元素匹配的第一代码描述;利用逻辑配置模板生成组件关联关系对应的第二代码描述;将第一代码描述以及第二代码描述通过代码编译器编译生成UI设计稿文件对应的前端代码。

[0060] 相应的,在利用逻辑配置模板生成组件关联关系对应的第二代码描述时,第一转换模块33,具体可用于在第一预设映射表中提取与组件关联关系对应的逻辑配置模板,其中,第一预设映射表包括组件关联关系与逻辑配置模板的对应关系;根据所确定出的逻辑配置模板生成对应的代码描述,并将组件元素在UI界面中的坐标、尺寸及外观属性添加至

代码描述中，得到组件关联关系对应的第二代码描述。

[0061] 在具体的应用场景中，在将第一代码描述以及第二代码描述通过代码编译器编译生成UI设计稿文件对应的前端代码时，第一转换模块33，具体可用于按照预设组合规则组合第一代码描述以及第二代码描述，得到UI设计稿文件对应UI界面的目标代码描述；将目标代码描述载入不同的代码编译器，生成对应不同平台的前端代码。

[0062] 在具体的应用场景中，第一前端数据包括画布元素、控件信息、用户通过控件修改的信息、图层关系、布局信息、图层样式信息以及组件配置信息，相应的，解析模块31，可用于读取并解析UI设计稿文件，提取画布元素和控件信息；利用组件元素的组件唯一标识在控件信息中查取控件节点，并对控件节点进行还原处理，还原用户通过控件修改的信息；根据画布元素中的节点布局尺寸确定图层关系和布局信息；将画布元素中的节点样式描述按照CSS样式规则转化为图层样式信息；根据画布元素中的节点名称确定所需要生成代码的元素组件类型，并根据第二预设映射表以及元组组件类型提取组件配置信息，第二预设映射表中创建有元组组件类型与组件配置信息间的映射关系。

[0063] 相应的，在基于深度优先算法并根据第一前端数据生成各个组件元素的第一组件结构化数据时，第一生成模块32，具体可用于按照深度优先搜索算法递归遍历第一前端数据中各个组件元素的所有路径，计算路径中每条边的权值之和，得出权值之和最大的路径；将权值之和最大的路径作为主干路径，倒置所有路径的时序关系，生成多叉树结构；按照自顶向下的顺序逐层读取多叉树各节点，将对象树序列化成JSON格式，得到第一组件结构化数据。

[0064] 在具体的应用场景中，如图4所示，该装置还包括：加载模块34、调整模块35、第二生成模块36、第二转换模块37；

[0065] 加载模块34，可用于将基于第一组件结构化数据转换的前端代码加载到用户配置界面；

[0066] 调整模块35，可用于若接收到用户配置页面上传的配置修改内容，则根据配置修改内容将第一前端数据调整为第二前端数据；

[0067] 第二生成模块36，可用于基于深度优先算法并根据第二前端数据生成各个组件元素的第二组件结构化数据；

[0068] 第二转换模块37，可用于根据预设的代码转换规则将第二组件结构化数据转换为UI设计稿文件对应的前端代码。

[0069] 需要说明的是，本实施例提供的一种前端代码的生成装置所涉及各功能单元的其他相应描述，可以参考图1至图2的对应描述，在此不再赘述。

[0070] 基于上述如图1至图2所示方法，相应的，本实施例还提供了一种存储介质，存储介质具体可为易失性或非易失性，其上存储有计算机可读指令，该可读指令被处理器执行时实现上述如图1至图2所示的前端代码的生成方法。

[0071] 基于这样的理解，本申请的技术方案可以以软件产品的形式体现出来，该软件产品可以存储在一个存储介质(可以是CD-ROM，U盘，移动硬盘等)中，包括若干指令用以使得一台计算机设备(可以是个人计算机，服务器，或者网络设备等)执行本申请各个实施场景的方法。

[0072] 基于上述如图1至图2所示的方法和图3、图4所示的虚拟装置实施例，为了实现上

述目的,本实施例还提供了一种计算机设备,该计算机设备包括存储介质和处理器;存储介质,用于存储计算机程序;处理器,用于执行计算机程序以实现上述如图1至图2所示的前端代码的生成方法。

[0073] 可选的,该计算机设备还可以包括用户接口、网络接口、摄像头、射频(Radio Frequency,RF)电路,传感器、音频电路、WI-FI模块等等。用户接口可以包括显示屏(Display)、输入单元比如键盘(Keyboard)等,可选用户接口还可以包括USB接口、读卡器接口等。网络接口可选的可以包括标准的有线接口、无线接口(如WI-FI接口)等。

[0074] 本领域技术人员可以理解,本实施例提供的一种计算机设备结构并不构成对该实体设备的限定,可以包括更多或更少的部件,或者组合某些部件,或者不同的部件布置。

[0075] 存储介质中还可以包括操作系统、网络通信模块。操作系统是管理上述计算机设备硬件和软件资源的程序,支持信息处理程序以及其它软件和/或程序的运行。网络通信模块用于实现存储介质内部各组件之间的通信,以及与信息处理实体设备中其它硬件和软件之间通信。

[0076] 通过以上的实施方式的描述,本领域的技术人员可以清楚地了解到本申请可以借助软件加必要的通用硬件平台的方式来实现,也可以通过硬件实现。

[0077] 通过应用本申请的技术方案,与目前现有技术相比,本申请可首先获取UI设计稿文件,并按照预设的文件解析规则解析UI设计稿文件,获取UI界面中各个组件元素的第一前端数据;进而基于深度优先算法并根据第一前端数据生成各个组件元素的第一组件结构化数据;之后可根据预设验证规则对第一组件结构化数据进行正确性验证,在判定第一组件结构化数据通过正确性验证后,可进一步根据预设的代码转换规则将第一组件结构化数据转换为UI设计稿文件对应的前端代码。本方案通过解析UI设计稿结构信息并生成代码,相较业内流行的拖拽组件方式,拥有更高的效率且更加自动化,同时具备更低的平台学习成本;而相较于方兴未艾的AI视觉识别模式,又具备更高的识别准确度和样式还原度。

[0078] 本领域技术人员可以理解附图只是一个优选实施场景的示意图,附图中的模块或流程并不一定是实施本申请所必须的。本领域技术人员可以理解实施场景中的装置中的模块可以按照实施场景描述进行分布于实施场景的装置中,也可以进行相应变化位于不同于本实施场景的一个或多个装置中。上述实施场景的模块可以合并为一个模块,也可以进一步拆分成多个子模块。

[0079] 上述本申请序号仅仅为了描述,不代表实施场景的优劣。以上公开的仅为本申请的几个具体实施场景,但是,本申请并非局限于此,任何本领域的技术人员能思之的变化都应落入本申请的保护范围。

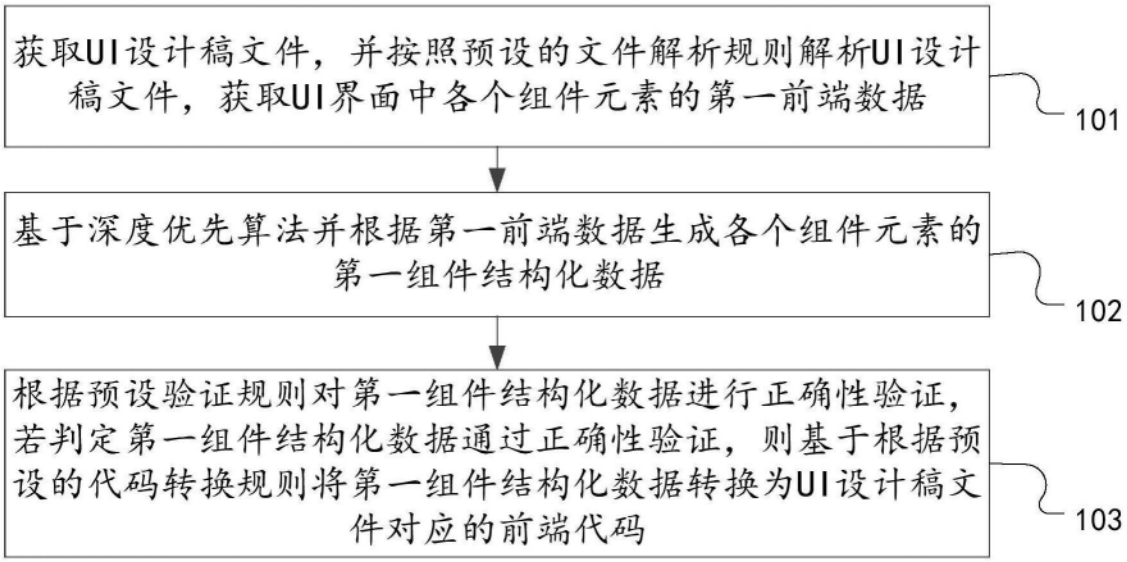

获取UI设计稿文件，并按照预设的文件解析规则解析UI设计稿文件，获取UI界面中各个组件元素的第一前端数据
101
基于深度优先算法并根据第一前端数据生成各个组件元素的第一组件结构化数据
102
根据预设验证规则对第一组件结构化数据进行正确性验证，若判定第一组件结构化数据通过正确性验证，则基于根据预设的代码转换规则将第一组件结构化数据转换为UI设计稿文件对应的前端代码
103

图1

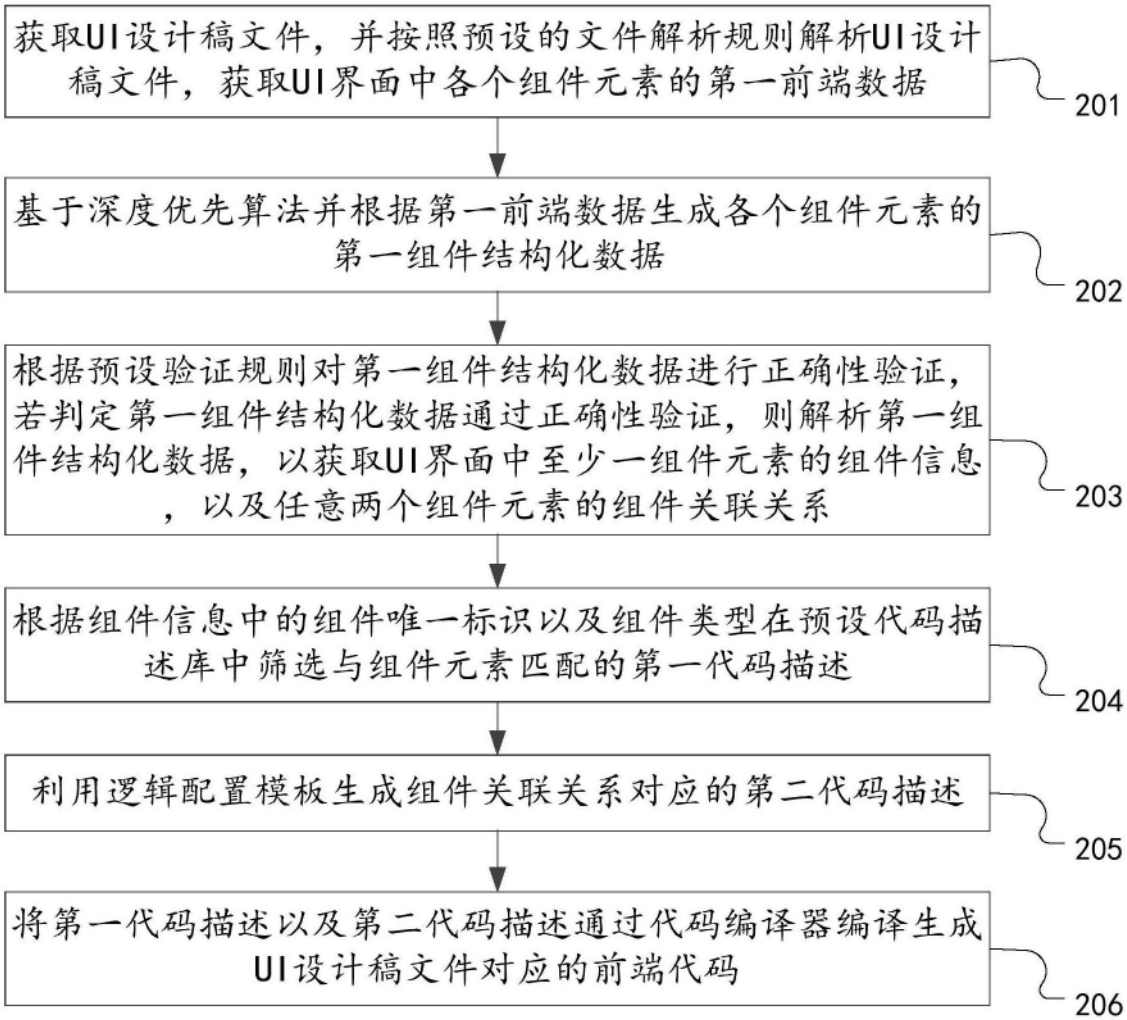

获取UI设计稿文件，并按照预设的文件解析规则解析UI设计稿文件，获取UI界面中各个组件元素的第一前端数据
201
基于深度优先算法并根据第一前端数据生成各个组件元素的第一组件结构化数据
202
根据预设验证规则对第一组件结构化数据进行正确性验证，若判定第一组件结构化数据通过正确性验证，则解析第一组件结构化数据，以获取UI界面中至少一组件元素的组件信息，以及任意两个组件元素的组件关联关系
203
根据组件信息中的组件唯一标识以及组件类型在预设代码描述库中筛选与组件元素匹配的第一代码描述
204
利用逻辑配置模板生成组件关联关系对应的第二代码描述
205
将第一代码描述以及第二代码描述通过代码编译器编译生成UI设计稿文件对应的前端代码
206

图2

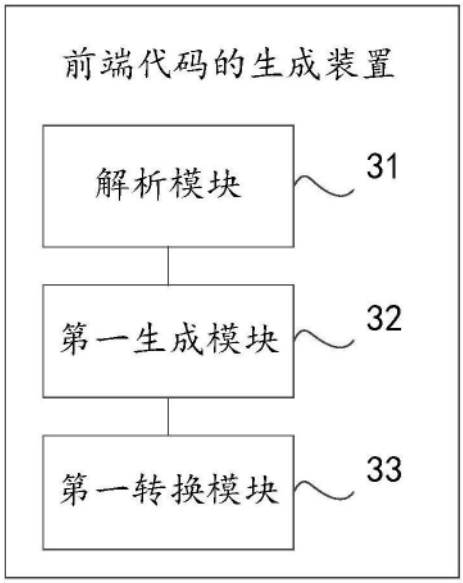


图3

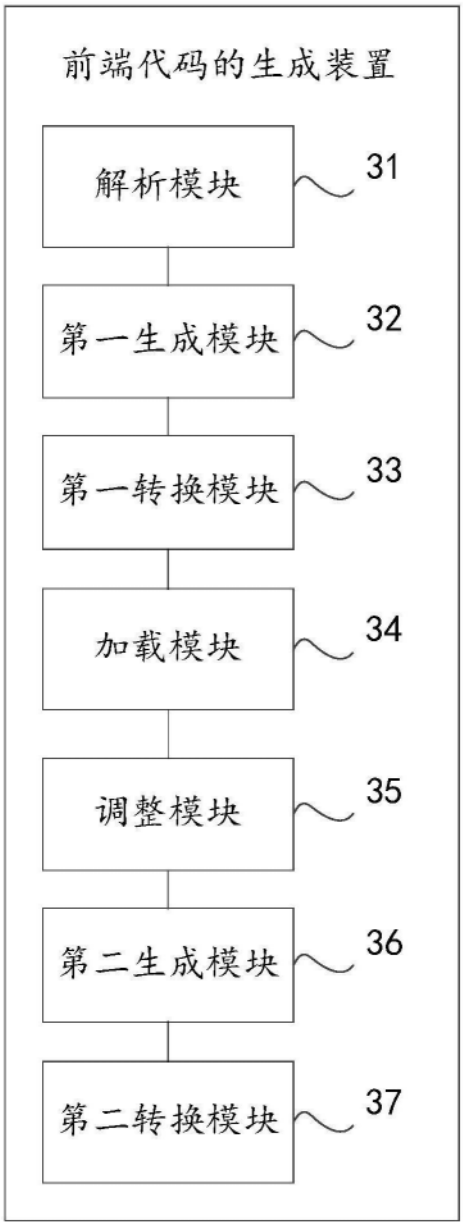

前端代码的生成装置
解析模块
31
第一生成模块
32
第一转换模块
33
加载模块
34
调整模块
35
第二生成模块
36
第二转换模块
37

图4