(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 112241282 B
(45)授权公告日 2024.02.23

(21)申请号 202011167627.8

(22)申请日 2020.10.27

(65)同一申请的已公布的文献号
申请公布号 CN 112241282 A

(43)申请公布日 2021.01.19

(73)专利权人 上海万向区块链股份公司
地址 200086 上海市虹口区塘沽路463号1201室

(72)发明人 吴广 郑斌斌 程然 刘伟国 李智博 李亚峰 井辉辉 黄守毅 阙磊磊

(74)专利代理机构 上海段和段律师事务所 31334
专利代理师 李佳俊 郭国中

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

(56)对比文件
CA 2238973 A1,1997.07.24
CN 107908398 A,2018.04.13
CN 109032587 A,2018.12.18
CN 110781010 A,2020.02.11
CN 111078220 A,2020.04.28
CN 111694837 A,2020.09.22
US 2015365284 A1,2015.12.17
US 2018189735 A1,2018.07.05
US 2019324729 A1,2019.10.24
US 2020249921 A1,2020.08.06
US 2020327817 A1,2020.10.15

审查员 舒婷

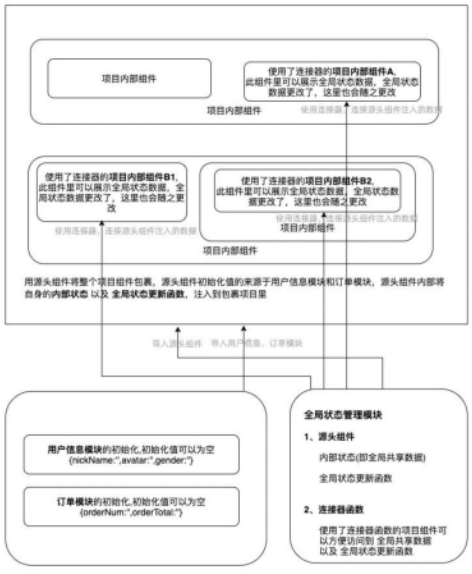
权利要求书2页 说明书9页 附图2页

(54)发明名称

基于react项目的全局状态数据管理方法及系统

(57)摘要

本发明提供了一种基于react项目的全局状态数据管理方法及系统,包括:定义全局共享数据模块以及每个全局共享数据模块对应的属性字段值;建立全局状态管理模块,通过全局状态管理模块导出源头组件和连接器;将项目最外层组件嵌套进源头组件,根据全局共享数据模块以及每个全局共享数据对应的属性字段值初始化源头组件;并通过源头组件向项目内部注入共享数据以及更新共享数据的函数;通过连接器包装需要获取全局状态数据的组件,组件在经过连接器包装处理后,获取源头组件注入的共享数据。本发明将更新状态的逻辑下放给组件内部,不分离更新状态值的逻辑,发明本身只专注数据的管理,从而也解决了各种异步状态更新的问题。



CN 112241282 B

1.一种基于react项目的全局状态数据管理系统,其特征在于,包括:

模块M1:定义全局共享数据模块以及每个全局共享数据模块对应的属性字段值;

模块M2:建立全局状态管理模块,通过全局状态管理模块导出源头组件和连接器;

模块M3:将项目最外层组件嵌套进源头组件,根据全局共享数据模块以及每个全局共享数据对应的属性字段值初始化源头组件的内部状态数据;并通过源头组件向项目内部注入共享数据以及全局状态更新函数;

模块M4:通过连接器包装需要获取全局状态数据的组件,组件在经过连接器包装处理后,获取源头组件注入的共享数据;

所述模块M3包括:

模块M3.1:在项目最外层导入多个不同全局共享数据模块及相应属性字段值和源头组件,将各全局共享数据模块数据及相应属性字段值以参数传递的方式传入源头组件,源头组件内部将接收到的各全局共享数据模块数据及相应属性字段值初始化为源头组件的内部状态数据;

模块M3.2:通过React context的原生api将源头组件内部状态数据和全局状态更新函数注入到源头组件包裹的项目组件里;

所述模块M4包括:

模块M4.1:在项目内部需要展示共享数据的组件里导入连接器,连接器包裹需要展示共享数据的组件,组件在经过连接器包装处理后,获取源头组件注入的共享数据;

模块M4.2:全局状态更新函数通过连接器的传递,组件内部获取全局状态更新函数,通过全局状态更新函数将需要更新的全局共享数据模块值传入组件内;全局状态更新函数是定义在源头组件内部,参数是需要被更新模块对象,在参数里正确填写模块名以及模块对应的属性名,即可完成数据更新;

步骤M4.3:数据从更新的组件内传递到源头组件,源头组件更新自身的内部状态,内部状态注入到包裹的项目组件里,项目组件里调用连接器函数的组件接受的参数发生改变。

2.根据权利要求1所述的基于react项目的全局状态数据管理系统,其特征在于,所述模块M1包括:根据业务模块拆分全局共享数据,并根据项目实际需求建立多个不同全局共享数据模块,每个全局共享数据模块以key-value格式存储数据;每个全局共享数据模块定义对应的属性字段值,每一个属性字段值是被全局共享的状态值。

3.根据权利要求1所述的基于react项目的全局状态数据管理系统,其特征在于,所述全局状态更新函数包括:当组件调用传入不存在的全局共享数据模块或全局共享数据模块属性字段值时,全局状态更新函数通过循环遍历找到不合法的属性,并将不合法的属性舍弃;传入的全局共享数据模块经过舍弃后,保证传入的模块对象结构没有被污染,然后将传入的全局共享数据模块和原有的状态模块合并,得到新的将要被更新的全局共享数据模块。

4.一种基于react项目的全局状态数据管理方法,其特征在于,包括:

步骤M1:定义全局共享数据模块以及每个全局共享数据模块对应的属性字段值;

步骤M2:建立全局状态管理模块,通过全局状态管理模块导出源头组件和连接器;

步骤M3:将项目最外层组件嵌套进源头组件,根据全局共享数据模块以及每个全局共享数据对应的属性字段值初始化源头组件;并通过源头组件向项目内部注入共享数据以及

更新共享数据的函数;

步骤M4:通过连接器包装需要获取全局状态数据的组件,组件在经过连接器包装处理后,获取源头组件注入的共享数据;

所述步骤M3包括:

步骤M3.1:在项目最外层导入多个不同全局共享数据模块和源头组件,将各全局共享数据模块数据以参数传递的方式传入源头组件,源头组件内部将接收到的各全局共享数据模块数据初始化为源头组件的内部状态数据;

步骤M3.2:通过React context的原生api将源头组件内部状态数据和全局状态更新函数注入到源头组件包裹的项目组件里;

所述步骤M4包括:

步骤M4.1:在项目内部需要展示共享数据的组件里导入连接器,连接器包裹需要展示共享数据的组件,组件在经过连接器包装处理后,获取源头组件注入的共享数据;

步骤M4.2:全局状态更新函数通过连接器的传递,组件内部获取全局状态更新函数,通过全局状态更新函数将需要更新的全局共享数据模块值传入组件内;全局状态更新函数是定义在源头组件内部,参数是需要被更新模块对象,在参数里正确填写模块名以及模块对应的属性名,即可完成数据更新;

步骤M4.3:数据从更新的组件内传递到源头组件,源头组件更新自身的内部状态,内部状态注入到包裹的项目组件里,项目组件里调用连接器函数的组件接受的参数发生改变。

5.根据权利要求4所述的基于react项目的全局状态数据管理方法,其特征在于,所述步骤M1包括:根据业务模块拆分全局共享数据,并根据项目实际需求建立多个不同全局共享数据模块,每个全局共享数据模块以key-value格式存储数据;每个全局共享数据模块定义对应的属性字段值,每一个属性字段值是被全局共享的状态值。

6.根据权利要求4所述的基于react项目的全局状态数据管理方法,其特征在于,所述全局状态更新函数包括:当组件调用传入不存在的全局共享数据模块或全局共享数据模块属性字段值时,全局状态更新函数通过循环遍历找到不合法的属性,并将不合法的属性舍弃;传入的全局共享数据模块经过舍弃后,保证传入的模块对象结构没有被污染,然后将传入的全局共享数据模块和原有的状态模块合并,得到新的将要被更新的全局共享数据模块。

基于react项目的全局状态数据管理方法及系统

技术领域

[0001] 本发明涉及数据处理技术领域,具体地,涉及一种基于react项目的全局状态数据管理方法及系统。

背景技术

[0002] React是一个用于构建用户界面的 JavaScript 库,使用React可方便地基于组件(可理解为独立的界面块,可嵌套)进行开发,组件的界面展示内容是通过组件内的渲染函数(render)对数据(该数据包括二部分,一是组件内部状态数据称之为state,可通过组件内部逻辑维护,二是外部传递进来的数据称之为props,组件本身不可更改props)的使用得到的。组件的这种编写方式,让开发者从界面的控制逻辑中解放出来,通过数据驱动界面,数据更新界面随之更新,从而让开发者更专注于数据的正确处理,提高界面开发效率。

[0003] 由于组件可以嵌套,项目的数据流从外层到内层步步传递,当数据传递超过3层时,数据传递和维护将变得难以维护。开发者引用的第三方解决方案,存在不易学习、难以使用的困境。

[0004] 组件的包裹和嵌套,组件使用时是以标签的形式书写的,如<componentA></componentA>表示A组件,如<componentA><componentB></componentB></componentA>,表示A组件包裹了B组件,也可以理解为B组件嵌套在A组件内部。

[0005] React createContext是React原生的api,createContext提供2个组件Provider和Consumer,Provider是容器组件,包裹需要共享数据的组件,在其包裹下的任意层级组件,都可以获取Provider组件提供的共享数据,该共享数据是通过Provider的value属性传递的进去的。被Provider包裹的子孙组件,想要获取共享数据,必须通过Consumer包裹,内部执行一个回调函数,回调函数的参数即是共享数据。

[0006] key-value格式对象,是一种通过key映射到value的数据结构,如建立一份数据对象obj01,包含“姓名”、“电话”和“是否显示”3条数据,可以写为let obj01 = {"name":"张三","phone":"15012345678","isShow":false},通过key的名字即可访问到内容,如obj01["name"]可获取姓名的内容,key-value格式对象是javascript中非常通常对象格式。

[0007] 全局状态管理开源解决方案redux,该技术提供一套严格的数据传递路径,提出了各种不易理解的概念和模块(store、action、reducer,预测化的状态管理,数据不可修改、纯函数),每次更新都要将更新逻辑剥离出去,遇到异步问题更是要引用第三方中间件才能解决,API对开发者极度不友好。无法进行多组数据同时更新的操作。

[0008] 全局状态管理开源解决方案mobx,该技术使用观察者模式来观测数据的变化(常用见概念有Observable state、Computed values、Reactions、Actions),简单、透明、可伸缩,API使用起来比较简单。但是此更新状态时依然是将更新逻辑分离出去的,无法进行多组数据同时更新的操作。

[0009] 原生api React hooks,该技术是新版React框架自身提供的API,需将项目组件写成函数式的组件,不是完整的解决方案,需开发者自己组织通用的解决方案,且使用场景仅

限函数式的组件。

[0010] 本发明解决了React项目中全局状态数据在多层嵌套组件间通信和管理困难,提供了一种全新的简单易用的状态管理方法,方便得实现任意组件间的通信,可同时更新多组状态值,可同时更新多模块的多组状态值,也解决了各种异步状态更新难的问题。

发明内容

[0011] 针对现有技术中的缺陷,本发明的目的是提供一种基于react项目的全局状态数据管理系统及方法。

[0012] 根据本发明提供的一种基于react项目的全局状态数据管理系统,包括:

[0013] 模块M1:定义全局共享数据模块以及每个全局共享数据模块对应的属性字段值;

[0014] 模块M2:建立全局状态管理模块,通过全局状态管理模块导出源头组件和连接器;

[0015] 模块M3:将项目最外层组件嵌套进源头组件,根据全局共享数据模块以及每个全局共享数据对应的属性字段值初始化源头组件的内部状态数据;并通过源头组件向项目内部注入共享数据以及全局状态更新函数;

[0016] 模块M4:通过连接器包装需要获取全局状态数据的组件,组件在经过连接器包装处理后,获取源头组件注入的共享数据。

[0017] 优选地,所述模块M1包括:根据业务模块拆分全局共享数据,并根据项目实际需求建立多个不同全局共享数据模块,每个全局共享数据模块以key-value格式存储数据;每个全局共享数据模块定义对应的属性字段值,每一个属性字段值是被全局共享的状态值。

[0018] 优选地,所述模块M3包括:

[0019] 模块M3.1:在项目最外层导入多个不同全局共享数据模块及相应属性字段值和源头组件,将各全局共享数据模块数据及相应属性字段值以参数传递的方式传入源头组件,源头组件内部将接收到的各全局共享数据模块数据及相应属性字段值初始化为源头组件的内部状态数据;

[0020] 模块M3.2:通过React context的原生api将源头组件内部状态数据和全局状态更新函数注入到源头组件包裹的项目组件里。

[0021] 优选地,所述模块M4包括:

[0022] 模块M4.1:在项目内部需要展示共享数据的组件里导入连接器,连接器包裹需要展示共享数据的组件,组件在经过连接器包装处理后,获取源头组件注入的共享数据;

[0023] 模块M4.2:全局状态更新函数通过连接器的传递,组件内部获取全局状态更新函数,通过全局状态更新函数将需要更新的全局共享数据模块值传入组件内;

[0024] 步骤M4.3:数据从更新的组件内传递到源头组件,源头组件更新自身的内部状态,内部状态注入到包裹的项目组件里,项目组件里调用连接器函数的组件接受的参数发生改变。

[0025] 优选地,所述全局状态更新函数包括:当组件调用传入不存在的全局共享数据模块或全局共享数据模块属性字段值时,全局状态更新函数通过循环遍历找到不合法的属性,并将不合法的属性舍弃;传入的全局共享数据模块经过舍弃后,保证传入的模块对象结构没有被污染,然后将传入的全局共享数据模块和原有的状态模块合并,得到新的将要被更新的全局共享数据模块。

[0026] 根据本发明提供的一种基于react项目的全局状态数据管理方法,包括:

[0027] 步骤M1:定义全局共享数据模块以及每个全局共享数据模块对应的属性字段值;

[0028] 步骤M2:建立全局状态管理模块,通过全局状态管理模块导出源头组件和连接器;

[0029] 步骤M3:将项目最外层组件嵌套进源头组件,根据全局共享数据模块以及每个全局共享数据对应的属性字段值初始化源头组件;并通过源头组件向项目内部注入共享数据以及更新共享数据的函数;

[0030] 步骤M4:通过连接器包装需要获取全局状态数据的组件,组件在经过连接器包装处理后,获取源头组件注入的共享数据。

[0031] 优选地,所述步骤M1包括:根据业务模块拆分全局共享数据,并根据项目实际需求建立多个不同全局共享数据模块,每个全局共享数据模块以key-value格式存储数据;每个全局共享数据模块定义对应的属性字段值,每一个属性字段值是被全局共享的状态值。

[0032] 优选地,所述步骤M3包括:

[0033] 步骤M3.1:在项目最外层导入多个不同全局共享数据模块和源头组件,将各全局共享数据模块数据以参数传递的方式传入源头组件,源头组件内部将接收到的各全局共享数据模块数据初始化为源头组件的内部状态数据;

[0034] 步骤M3.2:通过React context的原生api将源头组件内部状态数据和全局状态更新函数注入到源头组件包裹的项目组件里。

[0035] 优选地,所述模块M4包括:

[0036] 步骤M4.1:在项目内部需要展示共享数据的组件里导入连接器,连接器包裹需要展示共享数据的组件,组件在经过连接器包装处理后,获取源头组件注入的共享数据;

[0037] 步骤M4.2:全局状态更新函数通过连接器的传递,组件内部获取全局状态更新函数,通过全局状态更新函数将需要更新的全局共享数据模块值传入组件内;

[0038] 步骤M4.3:数据从更新的组件内传递到源头组件,源头组件更新自身的内部状态,内部状态注入到包裹的项目组件里,项目组件里调用连接器函数的组件接受的参数发生改变。

[0039] 优选地,所述全局状态更新函数包括:当组件调用传入不存在的全局共享数据模块或全局共享数据模块属性字段值时,全局状态更新函数通过循环遍历找到不合法的属性,并将不合法的属性舍弃;传入的全局共享数据模块经过舍弃后,保证传入的模块对象结构没有被污染,然后将传入的全局共享数据模块和原有的状态模块合并,得到新的将要被更新的全局共享数据模块。

[0040] 与现有技术相比,本发明具有如下的有益效果:

[0041] 1、本发明通过维护最外层组件的内部状态,并将该状态数据注入项目内部实现了全局数据共享,通过更新全局状态函数现实了react项目内的任意组件间通信,并同时更新多组状态值,同时更新多状态模块的多组状态值;

[0042] 2、本发明将更新状态的逻辑下放给组件内部,不分离更新状态值的逻辑,发明本身只专注数据的管理,从而也解决了各种异步状态更新的问题。

附图说明

[0043] 通过阅读参照以下附图对非限制性实施例所作的详细描述,本发明的其它特征、

目的和优点将会变得更明显：

[0044] 图1为项目内部组件展示共享数据示意图；

[0045] 图2为项目内部组件更新全局状态示意图。

具体实施方式

[0046] 下面结合具体实施例对本发明进行详细说明。以下实施例将有助于本领域的技术人员进一步理解本发明，但不以任何形式限制本发明。应当指出的是，对本领域的普通技术人员来说，在不脱离本发明构思的前提下，还可以做出若干变化和改进。这些都属于本发明的保护范围。

[0047] 实施例1

[0048] 根据本发明提供的一种基于react项目的全局状态数据管理系统，包括：如图1-2所示，

[0049] 模块M1：定义全局共享数据模块以及每个全局共享数据模块对应的属性字段值；

[0050] 模块M2：建立全局状态管理模块，通过全局状态管理模块导出源头组件和连接器；

[0051] 模块M3：将项目最外层组件嵌套进源头组件，根据全局共享数据模块以及每个全局共享数据对应的属性字段值初始化源头组件的内部状态数据；并通过源头组件向项目内部注入共享数据以及全局状态更新函数；

[0052] 模块M4：通过连接器包装需要获取全局状态数据的组件，组件在经过连接器包装处理后，获取源头组件注入的共享数据。

[0053] 具体地，所述模块M1包括：根据业务模块拆分全局共享数据，并根据项目实际需求建立多个不同全局共享数据模块，每个全局共享数据模块以key-value格式存储数据；每个全局共享数据模块定义对应的属性字段值，每一个属性字段值是被全局共享的状态值。

[0054] 具体地，所述模块M3包括：

[0055] 模块M3.1：在项目最外层导入多个不同全局共享数据模块及相应属性字段值和源头组件，将各全局共享数据模块数据及相应属性字段值以参数传递的方式传入源头组件，源头组件内部将接收到的各全局共享数据模块数据及相应属性字段值初始化为源头组件的内部状态数据；

[0056] 模块M3.2：通过React context的原生api将源头组件内部状态数据和全局状态更新函数注入到源头组件包裹的项目组件里。

[0057] 具体地，所述模块M4包括：

[0058] 模块M4.1：在项目内部需要展示共享数据的组件里导入连接器，连接器包裹需要展示共享数据的组件，组件在经过连接器包装处理后，获取源头组件注入的共享数据；

[0059] 模块M4.2：全局状态更新函数通过连接器的传递，组件内部获取全局状态更新函数，通过全局状态更新函数将需要更新的全局共享数据模块值传入组件内；

[0060] 步骤M4.3：数据从更新的组件内传递到源头组件，源头组件更新自身的内部状态，内部状态注入到包裹的项目组件里，项目组件里调用连接器函数的组件接受的参数发生改变。

[0061] 具体地，所述全局状态更新函数包括：当组件调用传入不存在的全局共享数据模块或全局共享数据模块属性字段值时，全局状态更新函数通过循环遍历找到不合法的属

性,并将不合法的属性舍弃;传入的全局共享数据模块经过舍弃后,保证传入的模块对象结构没有被污染,然后将传入的全局共享数据模块和原有的状态模块合并,得到新的将要被更新的全局共享数据模块。

[0062] 根据本发明提供的一种基于react项目的全局状态数据管理方法,包括:

[0063] 步骤M1:定义全局共享数据模块以及每个全局共享数据模块对应的属性字段值;

[0064] 步骤M2:建立全局状态管理模块,通过全局状态管理模块导出源头组件和连接器;

[0065] 步骤M3:将项目最外层组件嵌套进源头组件,根据全局共享数据模块以及每个全局共享数据对应的属性字段值初始化源头组件;并通过源头组件向项目内部注入共享数据以及更新共享数据的函数;

[0066] 步骤M4:通过连接器包装需要获取全局状态数据的组件,组件在经过连接器包装处理后,获取源头组件注入的共享数据。

[0067] 具体地,所述步骤M1包括:根据业务模块拆分全局共享数据,并根据项目实际需求建立多个不同全局共享数据模块,每个全局共享数据模块以key-value格式存储数据;每个全局共享数据模块定义对应的属性字段值,每一个属性字段值是被全局共享的状态值。

[0068] 具体地,所述步骤M3包括:

[0069] 步骤M3.1:在项目最外层导入多个不同全局共享数据模块和源头组件,将各全局共享数据模块数据以参数传递的方式传入源头组件,源头组件内部将接收到的各全局共享数据模块数据初始化为源头组件的内部状态数据;

[0070] 步骤M3.2:通过React context的原生api将源头组件内部状态数据和全局状态更新函数注入到源头组件包裹的项目组件里。

[0071] 具体地,所述模块M4包括:

[0072] 步骤M4.1:在项目内部需要展示共享数据的组件里导入连接器,连接器包裹需要展示共享数据的组件,组件在经过连接器包装处理后,获取源头组件注入的共享数据;

[0073] 步骤M4.2:全局状态更新函数通过连接器的传递,组件内部获取全局状态更新函数,通过全局状态更新函数将需要更新的全局共享数据模块值传入组件内;

[0074] 步骤M4.3:数据从更新的组件内传递到源头组件,源头组件更新自身的内部状态,内部状态注入到包裹的项目组件里,项目组件里调用连接器函数的组件接受的参数发生改变。

[0075] 具体地,所述全局状态更新函数包括:当组件调用传入不存在的全局共享数据模块或全局共享数据模块属性字段值时,全局状态更新函数通过循环遍历找到不合法的属性,并将不合法的属性舍弃;传入的全局共享数据模块经过舍弃后,保证传入的模块对象结构没有被污染,然后将传入的全局共享数据模块和原有的状态模块合并,得到新的将要被更新的全局共享数据模块。

[0076] 实施例2

[0077] 实施例2是实施例1的变化例

[0078] 现在有用户昵称(nickName),用户头像(avatar),用户性别(gender),订单数量(orderNum),订单价格总额(orderTotal)。这些数据需要同时展示在项目的多个组件中,并且多个组件之间嵌套关系比较复杂,用父子组件传值的方式很难解决。

[0079] 步骤1建立全局共享数据模块

[0080] 本实施例里,我们定义2个共享数据模块。模块一是用户信息模块对象(User模块),其内部字段包括3个字段,分别为用户昵称(nickName),用户头像(avatar),用户性别(gender),User模块对象内容为:{nickName:"nickName01",avatar:"http://www.avatar.com/myAvatar.png"},模块二是订单模块对象(Order模块),其内部字段包括2个字段,分别为订单数量(orderNum),订单价格总额(orderTotal),Order模块对象内容为{orderNum:10,orderTotal:100}.

[0081] 步骤2建立全局状态管理模块

[0082] 步骤3全局状态数据的初始化

[0083] 在项目最外层,导入用户信息模块对象和订单模块对象,导入源头组件,用源头组件将项目的入口组件包裹,向源头组件传入外部参数User模块对象和Order模块对象。源头组件的内部将User模块对象数据和Order模块对象数据初始化为源头组件的内部状态,并通过React context的原生api将其内部状态数据和更新状态数据的函数(函数内部this指向绑定为源头组件)注入到其包裹的项目组件里。

[0084] 步骤4项目内部组件展示共享数据

[0085] 如图1所示;

[0086] 步骤4.1单组件的展示共享数据

[0087] 在项目内部需要展示共享数据的组件里(此处定义为项目内部A组件),导入全局状态管理模块的连接器函数,将A组件作为连接器函数的参数,执行连接器函数得到一个新的A组件,因为连接器内部给A组件包裹了一层组件并将源头组件在[实施例步骤 3]里注入的数据通过单层父子组件传值的方式,新的A组件里可以通过this.props.user、this.props.order方式方便地获取用户模块和订单模块的全部数据,在渲染界面时可以任意的使用这些模块的值。

[0088] 步骤4.2多组件展示共享数据

[0089] 项目内部的其他组件,重复A组件的步骤,同样可以获取用户模块和订单模块的全部数据,也可以在渲染界面时任意的使用这些模块的值。从而实现了在项目内任意组件里可以便捷地获取全局状态的值。

[0090] 步骤5项目内部组件更新全局状态;

[0091] 如图2所示;

[0092] 步骤5.1组件内部执行全局状态更新函数

[0093] 通过[步骤 4],我们顺利的在A组件内部获取到了全局状态数据。现在A组件内部因为业务要求需要更新用户模块的昵称值。在[实施例步骤 3]里源头组件不仅向项目内部组件注入了内部状态数据并且还注入了更新状态数据的函数,通过连接器的传递,A组件内部可以通过this.props.updateGlobleState的方式获取全局状态更新函数,将需要更新的模块值传入,如更新昵称值this.props.updateGlobleState({user:{nickName:"newNickname"}}),这样函数在A组件内部执行了,但该函数的内部this指向还是和源头组件绑定的,所以函数执行的结果是更新了源头组件的内部状态User模块对象的值。

[0094] 步骤5.2全局状态更新函数执行前后的数据的传递

[0095] 数据从A组件传递到源头组件,源头组件更新自身内部状态,内部状态注入到包裹的项目组件里,项目组件里调用连接器函数的组件接受的参数发生改变,其展示界面与传

入参数相关的部分也随之改变。即A组件执行了一次全局状态更新函数,完成了将自身新的状态值传递给了其他使用了连接器的组件,并实时更新了它们的界面。

[0096] 步骤5.3 组件内部更新多模块状态数据

[0097] 在[步骤5.1]里,A组件更新的只是"昵称(nickName)"值单个状态,如果这里要更新"昵称(nickName)"、"用户头像(avatar)"、"订单数量(orderNum)"、"订单价格总额(orderTotal)"4个状态,同样我们可以在A组件里执行this.props.updateGlobleState({user:{nickName:"newNickname",avatar:"newAvatar"},order:{orderNum:"newOrderNum",orderTotal:"newOrderTotal"}}),由[步骤 2.3]可知,我们只要在参数里正确填写模块名以及模块对应的属性名,即可完成数据更新。更新数据只和数据格式有关和对应属性是否存在有关,这样我们就可以方便地更新各模块对应的属性值,不管是单模块单属性值,还是多状态模块多组状态值,这样设计可以方便的解决了异步更全局状态的问题。

[0098] 步骤5.4组件内部更新异步任务与逻辑下放

[0099] 假设现在A组件需要个更新全局状态是"用户头像(avatar)",但是获取"用户头像"的值是异步过程,我们需要在A组件里经过一定的逻辑处理得到"用户头像"的值后,才能调用全局状态更新函数进行状态更新。此处"用户头像"新状态值的获取的逻辑是放在A组件里进行的,这种将逻辑下放A组件,不破坏A组件自身的逻辑的完整性,可以清晰的知道"用户头像"状态值在A组件是如何被获取如何被更新的,保证了开发的连贯性。全局状态更新函数只关心数据,与异步任务与组件内部逻辑无关。

[0100] 实施例3

[0101] 实施例3是实施例1和/或实施例2的变化例

[0102] React项目是基于组件来进行开发的,独立组件便于抽象与组装,但组件的层层嵌套给组件间通信带来了极大不便,开发者引用的第三方解决方案,存在不易学习、难以使用以及组件自身逻辑与状态更新逻辑生硬剥离的困境。本申请提供了一条全新的全局状态数据管理方案,可方便得实现React项目内的任意组件间通信,可同时更新多组状态值,可同时更新多状态模块的多组状态值。将更新状态的逻辑下放给组件内部,不分离更新状态值的逻辑,方案本身只专注数据的管理,从而也解决了各种异步状态更新的问题。

[0103] 步骤1全局共享数据模块的定义

[0104] 全局共享数据是根据业务模块来拆分的,根据项目实际需求建立若干份不同模块对象,如用户信息模块对象,购物车模块对象,订单模块对象等等,每个模块对象下以key-value对象格式存储数据,如用户信息模块对象,有昵称 、头像2个属性值,可以写成{nickName:"xx",avatar:"xxxx"}等,这里定义的每一个属性对应的值都是要被全局共享的状态值。在项目运行之前,全局状态模块以及模块对应的属性字段值需要被明确的定义。

[0105] 步骤2全局状态管理模块的定义

[0106] 建立一个模块,该模块导出二个对象。

[0107] 一个导出对象是源头组件,该组件用来包裹整个项目,然后向项目内部注入共享数据以及更新共享数据的函数(外部组件向被包裹的组件注入共享数据,内部组件通过特定的包装能实时获取到注入进来的数据,此功能是React createContext实现的原生api),该组件本系统将之定义为源头组件。

[0108] 另一个导出对象是函数,该函数的作用是包装需要获取全局状态数据的组件,组件在经过该函数包装处理后得到一个新的组件(此处处理方案称之为高阶组件),新的组件可以便捷获取源头组件注入的共享数据,该导出的函数本系统将之定义为连接器。

[0109] 步骤2.1源头组件内部状态数据的初始化

[0110] 项目初始化时将[步骤 1]定义的不同数据模块导入项目的最外层(导入模块是javascript语言的原生语法),以传参数传递的方式将各模块数据传入源头组件,源头组件接受到外部参数,将各模块对象以模块名称为属性合并成一个更大的对象数据,将合并后的对象数据初始化为内部状态(组件的内部状态是react的原生定义,组件通过维护内部状态的变化从而实现对界面的控制)。

[0111] 步骤2.2全局状态更新函数的定义(updateGlobleState)

[0112] 组件内部可以通过setState方法来更新其内部状态(setState方法是react内部实现的api),但被源头组件包裹的内部组件却无法更新源头组件内部状态,我们需定义一个函数用来更新内部状态,并通过React createContext提供的Provider,Consumer机制,将此全局状态更新函数连同源头组件的内部状态一起注入内部组件,这样内部组件通过连接器包装后,就可以获得源头组件的状态和全局状态更新函数。该函数是定义在源头组件内部,参数是需要被更新模块对象,其的结构是类似于源头组件内部状态的对象结构,可以是单个或多个模块对象,每个模块对象里的属性值可以是单个或多个, 无返回值。传入参数的模块个数及模块下的属性个数决定了源头组件更新内部状态的对应内容,如果传入单模块单属性,那么源头组件更新单模块单属性,如果传入多模块多属性,那么源头组件将更新多模块多属性。灵活的参数结构决定了全局状态更新的多样性。

[0113] 步骤2.3全局状态更新函数的内部实现

[0114] 步骤2.3.1参数过滤

[0115] 当其他组件调用该方法传入不存的模块名或者模块对象里不存在的属性时,程序将通过循环遍历找到不合法的属性,并将之舍弃,因为不存在的模块名或模块属性会污染最初的源头组件的内部状态数据结构,程序只保留内部状态存在的模块名及模块对象下存在的属性。此处的参数过滤有效的阻止了使用者乱传参数,或参数传入不正确的情况。

[0116] 步骤2.3.2参数合并

[0117] 传入的模块对象经过参数过滤后,保证传入的模块对象结构没有被污染,然后将传入的模块对象和原有的状态模块对象合并(传入的模块对象覆盖原有的状态模块对象),得到新的将要被更新模块对象,此时使用setState方法将新模块对象值传入(react内部api,可以部分或全部更新组件内部状态),即可更新源头组件的内部状态。传入的模块对象和原有的状态模块对象合并的目的是,保证每次模块对象被更新时是完整的全新的一个对象,而传入的模块对象可以只传递需要更新的部分属性字段,其他属性字段合并时函数内部会自动补全。这样设计目的是,其他组件在实际场景中可以方便地更新单个状态值或多组状态值。

[0118] 本领域技术人员知道,除了以纯计算机可读程序代码方式实现本发明提供的系统、装置及其各个模块以外,完全可以通过将方法步骤进行逻辑编程来使得本发明提供的系统、装置及其各个模块以逻辑门、开关、专用集成电路、可编程逻辑控制器以及嵌入式微控制器等的形式来实现相同程序。所以,本发明提供的系统、装置及其各个模块可以被认为

是一种硬件部件,而对其内包括的用于实现各种程序的模块也可以视为硬件部件内的结构;也可以将用于实现各种功能的模块视为既可以是实现方法的软件程序又可以是硬件部件内的结构。

[0119] 以上对本发明的具体实施例进行了描述。需要理解的是,本发明并不局限于上述特定实施方式,本领域技术人员可以在权利要求的范围内做出各种变化或修改,这并不影响本发明的实质内容。在不冲突的情况下,本申请的实施例和实施例中的特征可以任意相互组合。

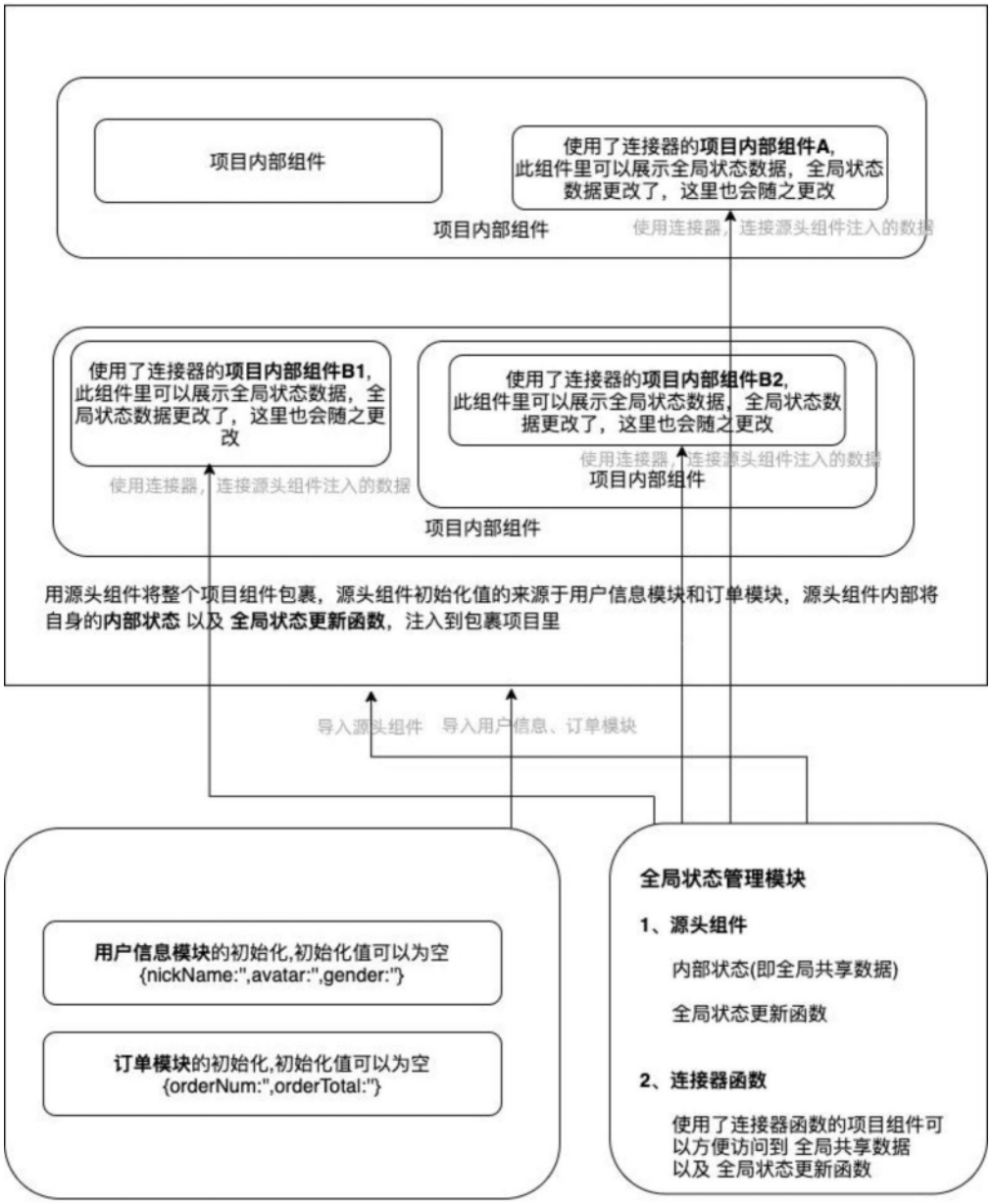

项目内部组件
使用了连接器的项目内部组件A，此组件里可以展示全局状态数据，全局状态数据更改了，这里也会随之更改
项目内部组件
使用连接器，连接源头组件注入的数据
使用了连接器的项目内部组件B1，此组件里可以展示全局状态数据，全局状态数据更改了，这里也会随之更改
使用连接器，连接源头组件注入的数据
使用了连接器的项目内部组件B2，此组件里可以展示全局状态数据，全局状态数据更改了，这里也会随之更改
使用连接器，连接源头组件注入的数据
项目内部组件
项目内部组件
用源头组件将整个项目组件包裹，源头组件初始化值的来源于用户信息模块和订单模块，源头组件内部将自身的内部状态 以及 全局状态更新函数，注入到包裹项目里
导入源头组件
导入用户信息、订单模块
用户信息模块的初始化,初始化值可以为空
{nickName:'',avatar:'',gender:''}
订单模块的初始化,初始化值可以为空
{orderNum:'',orderTotal:''}
全局状态管理模块
1、源头组件
内部状态(即全局共享数据)
全局状态更新函数
2、连接器函数
使用了连接器函数的项目组件可以方便访问到 全局共享数据 以及 全局状态更新函数

图1

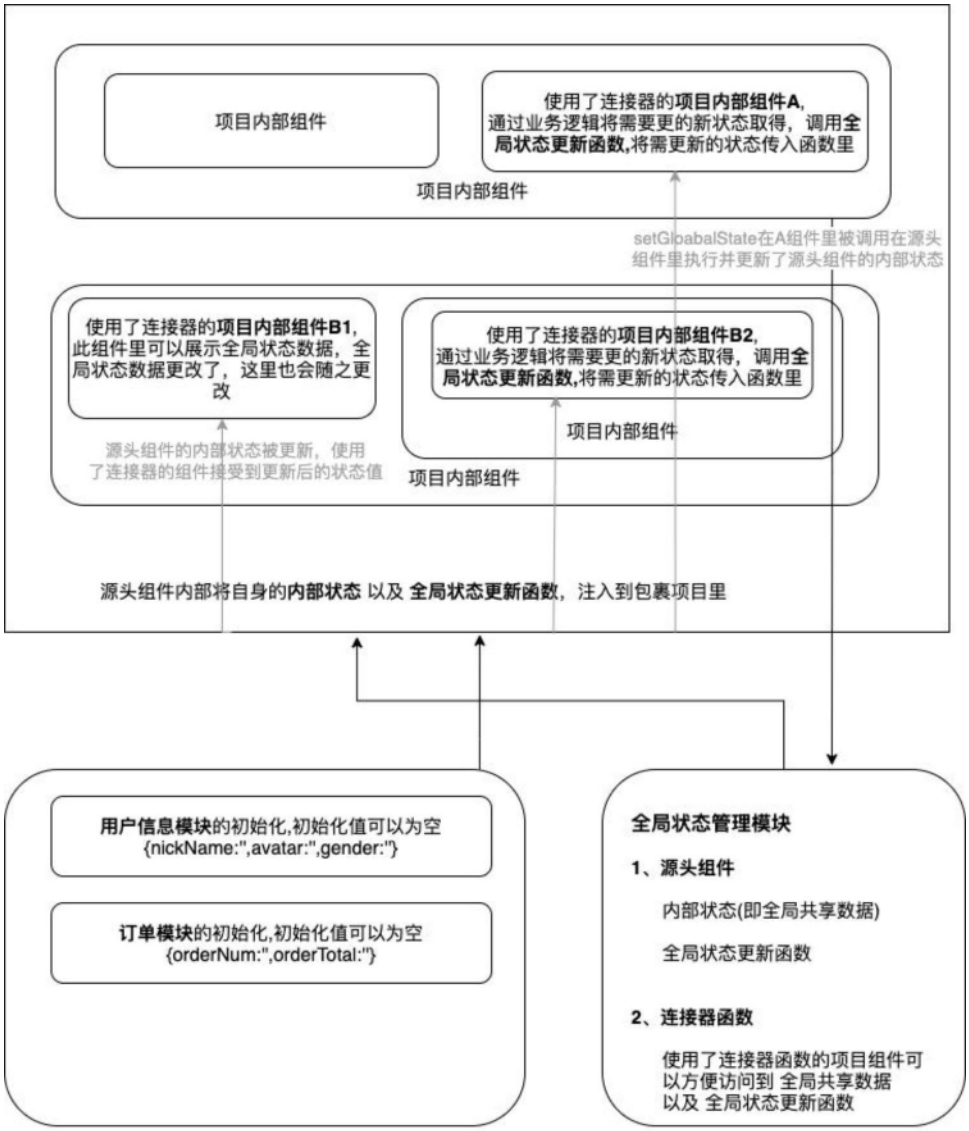

项目内部组件
使用了连接器的项目内部组件A，通过业务逻辑将需要更的新状态取得，调用全局状态更新函数,将需更新的状态传入函数里
项目内部组件
setGloabalState在A组件里被调用在源头组件里执行并更新了源头组件的内部状态
使用了连接器的项目内部组件B1，此组件里可以展示全局状态数据，全局状态数据更改了，这里也会随之更改
使用了连接器的项目内部组件B2，通过业务逻辑将需要更的新状态取得，调用全局状态更新函数,将需更新的状态传入函数里
项目内部组件
源头组件的内部状态被更新，使用了连接器的组件接受到更新后的状态值
项目内部组件
源头组件内部将自身的内部状态 以及 全局状态更新函数，注入到包裹项目里
用户信息模块的初始化,初始化值可以为空
{nickName:",avatar:",gender:"}
订单模块的初始化,初始化值可以为空
{orderNum:",orderTotal:"}
全局状态管理模块
1、源头组件
内部状态(即全局共享数据)
全局状态更新函数
2、连接器函数
使用了连接器函数的项目组件可以方便访问到 全局共享数据 以及 全局状态更新函数

图2