(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 105975265 B
(45)授权公告日 2019.04.12

(21)申请号 201610284250.1

(22)申请日 2016.04.29

(65)同一申请的已公布的文献号
申请公布号 CN 105975265 A

(43)申请公布日 2016.09.28

(73)专利权人 掌赢信息科技(上海)有限公司
地址 200063 上海市普陀区谈家渡路28号515室

(72)发明人 王星

(74)专利代理机构 北京律和信知识产权代理事务所(普通合伙) 11446
代理人 冷文燕 武玉琴

(51)Int.Cl.
G06F 8/30(2018.01)

(56)对比文件
CN 102650952 A,2012.08.29,
CN 101232505 A,2008.07.30,
CN 104267944 A,2015.01.07,
CN 104281441 A,2015.01.14,
CN 102591627 A,2012.07.18,
US 2015120577 A1,2015.04.30,
王念桥.应用MVP模式改进软件架构.《计算机时代》.2012,(第4期),第37-38,40页.
Yang Zhang etc..An Architecture and Implement Model for Model-View-Presenter Pattern.《IEEE International Conference On Computer Science》.2010,(第8期),第532-536页.

审查员 孔丹

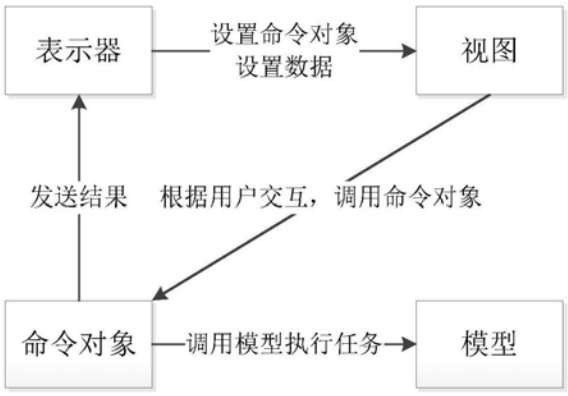
权利要求书1页 说明书5页 附图1页

(54)发明名称

一种基于改进型MVP模式的装置

(57)摘要

本发明实施例提供了一种基于改进型MVP模式的装置,属于计算机领域,包括:视图模块,用于显示用户交互界面和数据,以及根据用户交互调用对应的命令对象;表示器模块,用于维护一个或多个命令对象模块,将所述一个或多个命令对象传递给所述视图模块以在用户交互中调用,接收来自所述一个或多个命令对象模块的结果,并根据所述结果控制所述视图模块;一个或多个命令对象模块,用于执行与用户交互关联的任务或调用对应模型模块来执行与用户交互关联的任务,以及向所述表示器模块发送结果;以及一个或多个模型模块,用于提供数据相关的操作。



CN 105975265 B

1.一种基于改进型MVP模式的装置,其特征在于,所述装置包括:

视图模块,用于显示用户交互界面和数据,以及根据用户交互调用对应的命令对象,在MVP模式中引入命令模式,将原本由表示器提供的操作接口封装成一个或多个命令对象;

表示器模块,用于维护一个或多个命令对象模块,将所述一个或多个命令对象传递给所述视图模块以在用户交互中调用,接收来自所述一个或多个命令对象模块的结果,并根据所述结果控制所述视图模块;

一个或多个命令对象模块,用于执行与用户交互关联的任务或调用对应模型模块来执行与用户交互关联的任务,以及向所述表示器模块发送结果;以及

一个或多个模型模块,用于提供数据相关的操作。

2.根据权利要求1所述的装置,其特征在于,所述装置适用于安卓系统。

3.根据权利要求2所述的装置,其特征在于,所述表示器模块包括活动模块,所述命令对象模块包括动作对象模块。

4.根据权利要求1所述的装置,其特征在于,所述数据相关的操作包括数据的增加、查询、删除和修改中的至少一项。

5.根据权利要求1所述的装置,其特征在于,所述表示器模块具体用于根据所接收的来自一个或多个命令对象模块的所述结果控制所述视图的用户交互界面和/或数据的显示。

一种基于改进型MVP模式的装置

技术领域

[0001] 本发明涉及计算机领域,特别涉及一种基于改进型MVP模式的装置。

背景技术

[0002] 目前人机交互一般采用MVP(Model-View-Presenter)模式,在该模式中,模型定义了需要显示的数据或数据操作方式,视图用于显示模型中的数据以及向表示器发送数据相关的操作命令,而表示器作为中间人,协调视图和模型之间的关系,并且处理应用逻辑。但是,传统的MVP模式存在一些不足,例如,表示器与视图的关系过于紧密,表示器向视图提供一系列与视图相关的操作以用于调用,如果接口中有一个接口方法发生更改,则可能会影响到使用此接口的所有视图和表示器,不利于代码复用。

发明内容

[0003] 为了解决上述问题,本发明实施例提供了基于改进型MVP模式的装置。

[0004] 根据本发明的第一方面,提供了一种基于改进型MVP模式的装置,该装置包括:

[0005] 视图模块,用于显示用户交互界面和数据,以及根据用户交互调用对应的命令对象;

[0006] 表示器模块,用于维护一个或多个命令对象模块,将所述一个或多个命令对象传递给所述视图模块以在用户交互中调用,接收来自所述一个或多个命令对象模块的结果,并根据所述结果控制所述视图模块;

[0007] 一个或多个命令对象模块,用于执行与用户交互关联的任务或调用对应模型模块来执行与用户交互关联的任务,以及向所述表示器模块发送结果;以及

[0008] 一个或多个模型模块,用于提供数据相关的操作。

[0009] 结合本发明的第一方面,在第一种可能的实现方式中,所述装置包括安卓系统中的装置。

[0010] 结合本发明的第一方面的第一种可能的实现方式,在第二种可能的实现方式中,所述表示器模块包括活动模块,所述命令对象模块包括动作对象模块。

[0011] 结合本发明的第一方面,在第三种可能的实现方式中,所述数据相关的操作包括数据的增加、查询、删除和修改中的至少一项。

[0012] 结合本发明的第一方面,在第四种可能的实现方式中,所述表示器模块具体用于根据所述结果控制所述视图的用户交互界面和/或数据的显示。

[0013] 通过采用基于改进型MVP模式的应用架构,可以提高系统的正交性,从而提高模块化程度和代码的可复用性,以及应用的稳定性和可维护性。

附图说明

[0014] 为了更清楚地说明本发明实施例中的技术方案,下面将对实施例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的前提下，还可以根据这些附图获得其他的附图。

[0015] 图1示出了根据本发明实施例的基于改进型MVP模式的装置的示意图；

[0016] 图2示出了根据本发明实施例的安卓系统中基于改进型MVP模式的装置的示意图。

具体实施方式

[0017] 为使本发明的目的、技术方案和优点更加清楚，下面将结合本发明实施例中的附图，对本发明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。

[0018] 本发明实施例提供了一种基于改进型MVP模式的装置。通过在传统的MVP模式中引入命令(command)模式，将原本由表示器提供的操作接口封装成一个或多个命令对象，并由表示器来管理和维护这些命令对象。表示器将命令对象传递给对应的视图(view)，视图在界面上触发操作时执行对应的命令对象，命令对象执行具体的任务或调用对应的模型(model)执行具体的任务。表示器接收来自命令对象的结果，并根据该结果控制在视图中的数据的呈现。通过采用基于改进型MVP模式的应用架构，可以提高系统的正交性，从而提高模块化程度和代码的可复用性，以及应用的稳定性和可维护性。

[0019] 图1示出了根据本发明实施例的基于改进型MVP模式的装置的示意图。该装置可包括视图模块、表示器模块、一个或多个命令对象模块以及一个或多个模型模块。如图1中所示，表示器模块102用于管理一个或多个命令对象模块106，并且将一个或多个命令对象模块106传递给视图模块104，以便在视图模块104中的用户交互期间调用。此外，表示器模块102还用于接收来自一个或多个命令对象模块106的结果，并根据接收的结果控制视图模块104。具体的，表示器模块102根据接收的结果控制视图模块104中的用户交互界面和/或数据的显示。视图模块104用于显示用户交互界面和数据以及根据用户交互调用对应的命令对象模块106。命令对象模块106用于执行与用户交互关联的任务或调用对应的模型模块108来执行与用户交互关联的任务，并且将结果发送给表示器模块102。模型模块108用于提供数据相关的操作。所述数据相关的操作可包括数据的增加、查询、删除和修改中的至少一项。

[0020] 图2示出了根据本发明实施例的安卓系统中基于改进型MVP模式的装置的示意图。在安卓系统中，表示器模块可以包括活动(activity)模块，命令对象模块可以包括动作(action)对象模块，视图模块可以包括视图(view)模块，并且模型模块可以包括模型(model)模块。如图2所示，该装置可包括视图模块、活动模块、一个或多个动作对象模块以及一个或多个模型模块。

[0021] 视图模块用于显示用户交互界面和数据。具体的，视图模块可以采用各种形式呈现数据，例如列表、按钮、文本、图片及其组合等。本发明实施例对视图模块的具体形式不加以限定。视图模块还可用于显示各种类型的用户交互界面，例如登录界面和搜索界面等。具体的，视图模块可实现与界面相关的功能，例如界面显示、滑动、按钮事件、输入框的内容是否为空的检查等。视图模块可提供setXXAction(IAction)和setXXData(Data)系列方法，使得外部可以通过setXXData(Data)将具体数据传递给视图模块，并且可以通过setXXAction

(IAction)来设置界面中的用户交互触发的事件。例如,活动模块可以通过调用setXXAction(IAction)来设置视图模块中与用户交互界面中的一个或多个用户交互事件所对应的一个或多个动作。

[0022] 动作是对特定命令的封装。每一个动作对象模块可实现IAction接口。可以采用以下两种方案来定义该接口:

[0023] 方案一:

[0024] IAction接口可定义为:

[0025] public interface IAction{

[0026] public void execute();

[0027] }

[0028] 基类BaseAction可定义为:

```
public abstract class BaseAction implements IAction {
  private Context context;
  public BaseAction(Context context) {
[0029]    this.context = context;
  }
  public Context getContext() {
    return context;
  }
[0030]
}
```

[0031] 在创建具体的动作类之前,还需要定义和具体动作对应的动作接口。例如要执行按钮事件,可先定义一个ButtonClickAction接口:

[0032] public interface ButtonClickAction extends IAction{

[0033] public void setParamter(String buttonName);

[0034] }

[0035] 在ButtonClickAction接口中定义了具体的参数类型和个数。

[0036] 然后可创建一个具体的LeaderInfoButtonClickAction类,继承自BaseAction并实现ButtonClickAction接口。

[0037] 在视图模块中需要接受ButtonClickAction类型的动作对象模块,并且视图模块传递参数给动作对象模块以执行命令。

[0038] 方案一的优点是可以明确地定义传递参数的类型和个数。

[0039] 方案二:

[0040] IAction接口定义为:

[0041] public interface IAction{

[0042] public void execute(Map<String,Object>params);

[0043] }

[0044] 在IAction的execute中用Map来封装传入的参数。

[0045] 基类BaseAction：

```
public abstract class BaseAction implements IAction {

    private Context context;

    public BaseAction(Context context) {
```

[0046]

```
        this.context = context;

    }

    public Context getContext() {
        return context;
```

[0047]

```
    }

}
```

[0048] 创建具体的动作类时，只需要简单地继承BaseAction，实现execute方法。在视图模块中只接受IAction类型的动作对象模块。

[0049] 方案二的优点是提供较高程度的抽象，实现较为简单，更改接口参数对视图影响小，保证了视图和动作更高程度的可复用性。

[0050] 动作对象模块可执行与用户交互关联的任务，或者调用模型模块来执行与用户交互关联的任务。在动作对象模块执行完任务后，将结果发送给活动模块。

[0051] 活动模块用于将具体的动作对象模块传递给视图。具体的，活动可通过setXXAction(IAction)来将一个或多个动作对象模块传递给视图模块，以便视图模块在用户交互中调用。此外，活动模块还注册有BroadcastReceiver，以此来接收动作模块完成任务后发来的结果。随后，活动模块可在BroadcastReceiver中处理该结果，并反馈到视图模块上。具体的，活动模块可根据结果来控制视图模块中的用户交互界面和/或数据的显示。例如，活动模块可以通过setXXData(Data)将结果作为数据发送给视图模块。

[0052] 注册BroadcastReceiver可以采取动态注册或静态注册。采用动态注册时，可以在onResume方法中注册。采用静态注册时，可以在AndroidManifest文件中注册。注册时可指定动作，使得只有匹配动作的广播消息才会被接收。本发明实施例对BroadcastReceiver的具体注册方法不加以限定。

[0053] 模型模块用于提供数据相关的操作，例如一些底层的网络和数据库操作等，以此来完成数据的增加、查询、删除和修改等操作中的至少一项操作。

[0054] 通过在安卓系统中采用基于改进型MVP模式的装置，可以提高系统的正交性，从而提高模块化程度和代码的可复用性，以及应用的稳定性和可维护性。要说明的是，安卓系统仅是示例，本发明实施例公开的基于改进型MVP模型的装置也可适用于其他系统，例如iOS系统，WinPhone系统等，本发明实施例对此不加以限定。

[0055] 上述所有可选技术方案，可以采用任意结合形成本发明的可选实施例，在此不再

一一赘述。

[0056] 本领域普通技术人员可以理解实现上述实施例的全部或部分步骤可以通过硬件来完成，也可以通过程序来指令相关的硬件完成，所述的程序可以存储于一种计算机可读存储介质中，上述提到的存储介质可以是只读存储器，磁盘或光盘等。

[0057] 以上所述仅为本发明的较佳实施例，并不用以限制本发明，凡在本发明的精神和原则之内，所作的任何修改、等同替换、改进等，均应包含在本发明的保护范围之内。

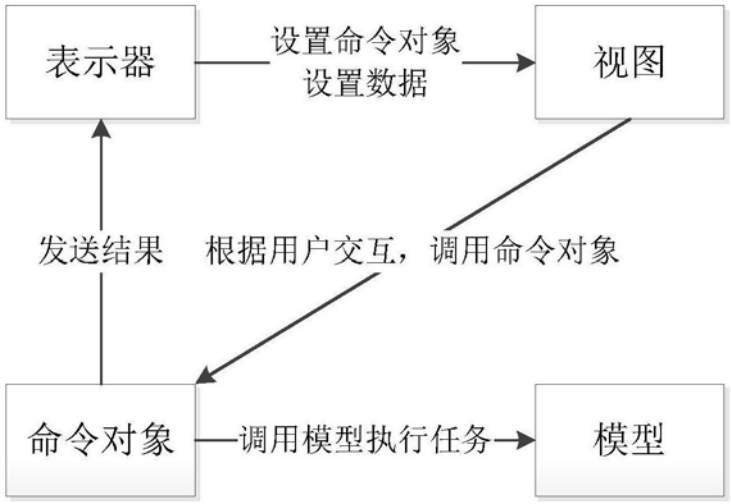

表示器
设置命令对象
设置数据
视图
发送结果
根据用户交互，调用命令对象
命令对象
调用模型执行任务
模型

图1

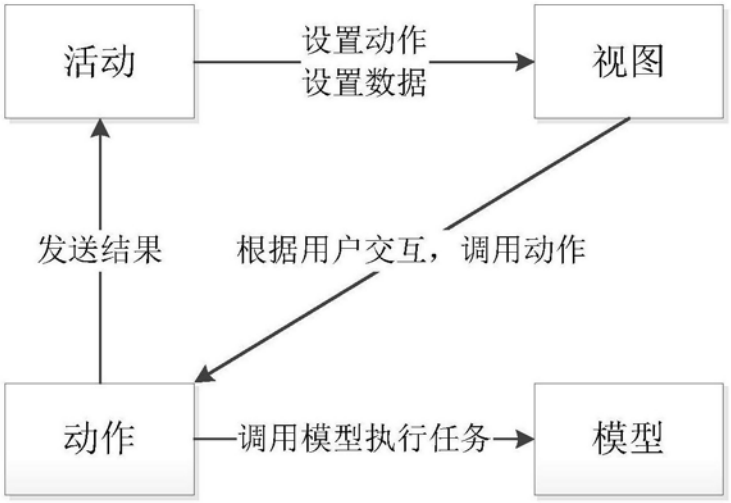

活动
设置动作
设置数据
视图
发送结果
根据用户交互，调用动作
动作
调用模型执行任务
模型

图2