

页面展示方法、装置、计算机设备和存储介质

本申请要求于 2017 年 11 月 1 日提交中国专利局，申请号为 2017110598994，申请名称为“页面展示方法、装置、计算机设备和存储介质”的中国专利申请的优先权，其全部内容通过引用结合在本申请中。

技术领域

- 5 本申请涉及计算机技术领域，特别是涉及页面展示方法、装置、计算机设备和存储介质。

背景技术

- 在 APP（Application，应用程序）中设置埋点是很多 APP 开发商收集用户行为数据的重要手段。目前，在 APP 中设置埋点具有多种方案，如代码埋点、可视化埋点和无埋点等。其中，要实现可视化埋点或无埋点，终端需要将 APP 页面截图以及对应的页面数据上传到 Web 服务器。由于无法对 APP 页面截图中的按钮等视图进行触摸操作，Web 服务器需要对 APP 页面截图中可以进行触摸操作的视图进行识别和区别标记，并通过 Web 页面对区别标记后的 APP 页面截图进行展示，使得 APP 开发商可以在 Web 页面对 APP 页面中不同视图设置埋点。传统方式中，Web 服务器通过对可以进行触摸操作的视图添加边框，将可以进行触摸操作的视图与其他视图进行区分。但直接根据终端上传的页面数据对不同视图添加边框，难以对边框的添加位置进行准确定位，常出现边框与视图发生错位等现象，使得 APP 页面的展示效果不好。

20

发明内容

根据本申请公开的各种实施例，提供一种页面展示方法、装置、计算机设备和存储介质。

一种页面展示方法，包括：获取应用程序当前页面的页面数据；所述页面数据包括页面截图以及多个视图的视图标识、视图名称和层级信息；按照预设规则将多个视图标识添加至多个数组；多个数组具有不同的级别；查询最高级别数组中的视图标识对应的视图名称，将最高级别数组中的视图标识
5 对应的视图名称标记为父节点；查询次高级别数组中的视图标识对应的视图名称，根据多个视图之间的层级信息，将次高级别数组中的视图标识对应的视图名称标记为所述父节点的子节点；将所述子节点作为下一级别数组中的视图标识对应的视图名称的父节点进行迭代，直至最低级别数组；利用父节点以及对应的子节点生成应用程序当前页面对应的多叉树；及根据所述多叉
10 树生成多个视图分别对应的层级路径，根据所述层级路径在页面截图对相应的视图添加对应的埋点边框，将设有埋点边框的页面截图发送至预设终端，使预设终端对设有埋点边框的页面截图进行展示。

一种页面展示装置，包括：数据获取模块，用于获取应用程序当前页面的页面数据；所述页面数据包括页面截图以及多个视图的视图标识、视图名
15 称和层级信息；多叉树构建模块，用于按照预设规则将多个视图标识添加至多个数组；多个数组具有不同的级别；查询最高级别数组中的视图标识对应的视图名称，将最高级别数组中的视图标识对应的视图名称标记为父节点；
查询次高级别数组中的视图标识对应的视图名称，根据多个视图之间的层级信息，将次高级别数组中的视图标识对应的视图名称标记为所述父节点的子
20 节点；将所述子节点作为下一级别数组中的视图标识对应的视图名称的父节点进行迭代，直至最低级别数组；利用父节点以及对应的子节点生成应用程序当前页面对应的多叉树；及页面展示模块，用于根据所述多叉树生成多个视图分别对应的层级路径，根据所述层级路径在页面截图对相应的视图添加
25 对应的埋点边框，将设有埋点边框的页面截图发送至预设终端，使预设终端对设有埋点边框的页面截图进行展示。

一种计算机设备，包括存储器和一个或多个处理器，存储器中存储有计算机可读指令，计算机可读指令被处理器执行时实现本申请任意一个实施例

中提供的页面展示方法的步骤。

一个或多个存储有计算机可读指令的非易失性存储介质，计算机可读指令被一个或多个处理器执行时，使得一个或多个处理器实现本申请任意一个实施例中提供的页面展示方法的步骤。

- 5 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

- 10 为了更清楚地说明本申请实施例中的技术方案，下面将对实施例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它的附图。

图 1 为一个实施例中的页面展示方法的应用环境图；

图 2 为一个实施例中页面展示方法的流程图；

- 15 图 3 为一个实施例中应用程序当前页面对应的多叉树的示意图；

图 4 为一个实施例中在页面截图中添加埋点边框的示意图；

图 5 为另一个实施例中页面展示方法的流程图；

图 6 为一个实施例中页面展示装置的结构图；

图 7 为一个实施例中计算机设备的框图。

20

具体实施方式

为了使本申请的技术方案及优点更加清楚明白，以下结合附图及实施例，对本申请进行进一步详细说明。应当理解，此处描述的具体实施例仅仅用以解释本申请，并不用于限定本申请。

- 25 可以理解，本申请所使用的术语“第一”、“第二”等可在本文中用于描述各种元件，但这些元件不受这些术语限制。这些术语仅用于将第一个元件与另一个元件区分。举例来说，在不脱离本申请的范围的情况下，可以将第

一客户端称为第二客户端，且类似地，可将第二客户端称为第一客户端。第一客户端和第二客户端两者都是客户端，但其不是同一客户端。

本申请提供一种页面展示方法，可以应用于如图 1 所示的应用环境中。其中，第一终端 102、第二终端 104 与第三终端 106 分别通过网路与服务器 108 连接。第一终端 102、第二终端 104 和第三终端 106 分别可以是智能手机、平板电脑、台式计算机、车载电脑中的至少一种，但并不局限于此。服务器 108 可以是独立的物理服务器，也可以是多个物理服务器构成的服务器集群。第一终端 102 与第三终端 106 上分别运行了需要设置埋点的应用程序。第一终端 102 对应应用程序开发商，第三终端 106 对应应用程序用户。第一终端 102 与第三终端 106 可以是同一终端，也可以是不同终端。第一终端 102 与第三终端 106 上的应用程序可以是原生应用程序（Native APP），也可以是网络应用程序（Web APP），还可以是混合应用程序。第一终端 102 和第三终端 106 在应用程序中分别安装了插件 SDK（Software Development Kit，软件开发工具包）。SDK 是基于开源的 UI（User Interface，用户界面）跨平台框架预先开发生成的脚本。UI 跨平台框架可以是 React Native 框架或 Weex 框架等。SDK 包括数据采集脚本、触摸事件拦截脚本、控件匹配脚本和触摸事件上报脚本等，用于数据采集、触摸事件拦截、控件匹配和触摸事件上报等。

当需要对应用程序设置埋点时，第一终端 102 利用 SDK 获取应用程序当前页面的页面数据，将页面数据发送至服务器 108。页面数据包括多个视图的视图名称和层级信息。服务器 108 根据多个视图的视图名称和层级信息，构建应用程序当前页面对应的多叉树，根据多叉树在页面截图添加多个视图对应的埋点边框，将设有埋点边框的页面截图发送至第二终端 104，使第二终端 104 通过 Web 页面对应用程序当前页面进行展示。应用程序开发商可以通过第二终端 104 对设有埋点边框的控件进行圈选，以设置埋点。第二终端 104 将被埋点控件对应的埋点信息返回至服务器 108。服务器 108 将被埋点控件的埋点信息发送至第三终端 106。根据多叉树，服务器可以获取每个视图相对其他视图的位置关系，能够对埋点边框的添加位置进行准确定位，对应

用程序当前页面具有较好的展示效果。

图 2 为本申请一个实施例的方法的流程示意图。应该理解的是，虽然图 2 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，其可以以其他的顺序执行。而且，图 2 中的至少一部分步骤可以包括多个子步骤或者多个阶段，这些子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，其执行顺序也不必然是依次进行，而是可以与其他步骤或者其他步骤的子步骤或者阶段的至少一部分轮流或者交替地执行。

10 在一个实施例中，如图 2 所示，提供了一种页面展示方法，以该方法应用于图 1 中的服务器为例进行说明，具体包括以下步骤：

步骤 202，获取应用程序当前页面的页面数据；页面数据包括多个视图的视图标识、视图名称和层级信息。

当需要对应用程序设置埋点时，第一终端利用 SDK 获取应用程序当前页
15 面的页面数据。页面数据包括应用程序当前页面的页面截图和当前页面内多个视图的属性信息。页面截图为静态的屏幕截图。视图包括控件和视图控制器。在应用程序当前页面，用于创建和管理控件，对用户是不可见的。用户可以对可见的控件进行触摸操作。触摸操作包括点击、长按、短按或滑动等。属性信息包括视图名称、视图标识和层级信息。容易理解，视图名称可以是
20 控件名称或控制器名称，视图标识可以是控件标识或控制器标识。视图名称可以直接反映视图类型，如 UIWindow（一种控件名称）属于窗口类型控件，UIButton 属于按钮类型控件，RootViewController 属于根视图控制器等。视图标识可以唯一识别一个视图，可以是视图编号，如 ID_n（n=1,2...）等。每个视图具有对应的层级信息，层级信息包括该视图与其他视图之间的继承关系。
25 例如，控件 UIButton 继承于控件 UIView，则控件 UIButton 为控件 UIView 的下一层级视图。

步骤 204，按照预设规则将多个视图标识添加至多个数组；多个数组具

有不同的级别。

服务器建立最高级别数组，在页面数据中查询视图名称与预设根视图的视图名称相同的视图，提取与预设根视图具有相同视图名称的视图的视图标识，将提取到的视图标识添加至最高级别数组，如最高级别数组可以是 A1[ID 6]。服务器将提取到的视图标记为父视图，根据该父视图的层级信息，对页面数据进行遍历，检测是否存在该父视图对应的子视图。若页面数据中存在对应的子视图，服务器建立次高级别数组，在页面数据中提取该父视图对应的子视图的视图标识，将提取到的视图标识添加至次高级别数组，如次高级别数组可以是 A2[ID 2, ID 7, ID 8]。服务器将提取到的子视图作为父视图，在页面数据中查询是否存在对应的子视图。若页面数据中存在对应的子视图，服务器建立下一级别数组，在页面数据中提取对应的子视图的视图标识，将提取到的视图标识添加至下一级别数组，如下一级别数组可以是 A3[ID 1, ID 4, ID 5]。服务器将重新提取到的子视图作为父视图，在页面数据中查询是否存在对应的子视图。若页面数据中存在对应的子视图，服务器按照上述方式建立再下一级别数组。如此循环，直至页面数据中不存在对应的子视图，得到最低级别数组，如最低级别数组可以是 A4[ID 3, ID 9, ID 10, ID 11, ID 12]。

步骤 206，查询最高级别数组中的视图标识对应的视图名称，将最高级别数组中的视图标识对应的视图名称标记为父节点。

步骤 208，查询次高级别数组中的视图标识对应的视图名称，根据多个视图之间的层级信息，将次高级别数组中的视图标识对应的视图名称标记为父节点的子节点。

步骤 210，将子节点作为下一级别数组中的视图标识对应的视图名称的父节点进行迭代，直至最低级别数组。

步骤 212，利用父节点以及对应的子节点生成应用程序当前页面对应的多叉树。

需要说明的是，步骤 206、步骤 208、步骤 210 和步骤 212 分别与步骤 204 并列执行。换句话说，服务器每生成一个数组，将对应的数组元素作为

节点添加至多叉树中。具体的，在生成最高级别数组后，服务器查询最高级别数组的数组元素对应的视图名称，将最高级别数组的数组元素对应的视图名称标记为多叉树的根节点。例如，如图 3 所示，多叉树的根节点为最高级别数组 A1 的数组元素 ID 6 对应的视图名称 UIWindow。

- 5 在得到多叉树的根节点后，服务器按照上述方式生成次高级别数组，查询次高级别数组的数组元素对应的视图名称，将次高级别数组的数组元素对应的视图名称标记为多叉树中根节点的子节点，利用预设连接线将子节点连接至根节点，形成多叉树的多个树枝。可以理解，根节点也是上述子节点的父节点。例如，如图 3 所示，预设连接线可以是带有箭头的线段，根节点的子节点为次高级别数组 A2 的三个数组元素 ID2、ID7 和 ID8 分别对应的视图名称 UIView、ViewController 和 UIView，对应形成多叉树的三个树枝。
- 10

- 在得到多叉树根节点的子节点后，服务器按照上述方式生成下一级别数组，将上述子节点作为该下一级别数组中数组元素的父节点，查询下一级别数组的数组元素对应的视图名称，利用预设连接线将下一级别数组中数组元素对应的视图名称连接至对应的父节点，以对多叉树的树枝进行延长。服务器根据上述每个子节点对应视图的层级信息，查询每个子节点对应视图的下一层级视图，利用预设连接线将下一级别数组中相应数组元素对应的视图名称连接至对应的子节点。例如，如图 3 所示，下一级别数组 A3 包括三个数组元素 ID1、ID4 和 ID5，分别对应的视图名称为 UIView、ChildController 和 ChildController。假设次高级别数组 A2 的数组元素 ID2 对应视图不存在下一层级视图，数组元素 ID7 对应视图的下一层级视图包括数组元素 ID1 对应的视图，则将数组元素 ID1 对应视图的视图名称 UIView 连接至数组元素 ID7 对应视图的视图名称 ViewController。
- 15
- 20

- 服务器按照上述方式依次将每个数组中数组元素对应视图的视图名称作为节点添加至多叉树中，形成应用程序当前页面最终对应的多叉树。第一终端上传至服务器的页面数据经过格式转换后可能不具有层级结构，且页面数据仅包含单个视图的层级信息，使得多个视图之间的层级关系隐蔽且分散，
- 25

难以从整体上反应应用程序当前页面多个视图间的层级结构。本实施例服务器根据不具有层级结构的页面数据，对应用程序页面中多个视图的属性信息进行整合，提炼出可以从整体上反应当前页面多个视图间层级结构的多叉树。

步骤 214，根据多叉树生成多个视图分别对应的层级路径，根据层级路径在页面截图对相应的视图添加对应的埋点边框，将设有埋点边框的页面截图发送至预设终端，使预设终端对设有埋点边框的页面截图进行展示。

服务器在多叉树构建过程中，对多叉树每个节点进行编号，以使每个节点对应的视图具有视图索引。具体的，服务器对多叉树中不同层级节点对应的视图分别进行单独编号，将编号作为对应视图的视图索引。服务器可以将视图索引作为对应视图的视图名称的下标进行记录。例如，如图 3 所示，三个按钮类型视图对应的视图索引可以是 UIButton₀、UIButton₁ 和 UIButton₂。

服务器根据视图的属性信息对应用程序当前页面可以进行触摸操作的视图（以下称“可触摸视图”）进行识别。服务器根据可触摸视图的视图标识，在多叉树中查询对应的节点，将该节点标记为目标节点。服务器利用第一预设标识将多叉树中根节点至目标节点的多个节点分别对应的视图名称依次拼接，得到目录字符串，将目录字符串作为目标节点对应视图的视图路径。服务器利用第二预设标识将多叉树中根节点至目标节点的多个节点分别对应的视图索引依次拼接，得到索引字符串，将索引字符串作为目标节点对应视图的索引路径。第一预设标识和第二预设标识分别可以是“/”，“_”，“#”，“@”或“*”等。

服务器根据视图路径和索引路径对每个可以进行触摸操作的视图进行准确定位，并根据定位对可以进行触摸操作的视图添加对应的埋点边框。埋点边框可以响应应用程序开发商的点击操作。如图 4 所示，埋点边框的形状与大小可以是与对应视图的图标的形状与大小一致。服务器将设有埋点边框的页面截图发送至预设终端，即第二终端。第二终端对设有埋点边框的页面截图进行展示。应用程序开发商可以通过在第二终端对页面截图上点击埋点边框，并输入对应的触摸事件标识，实现埋点设置。

本实施例中，根据应用程序当前页面多个视图的层级信息，可以将不同视图的视图标识添加至多个数组；根据不同级别数组包含的视图标识，可以查询对应视图的视图名称；通过将最高级别数组中的视图标识对应的视图名称标记为父节点，将次高级别数组中的视图标识对应的视图名称标记为该父节点的子节点，将该子节点作为下一级别数组中的视图标识对应的视图名称的父节点进行迭代，可以构建应用程序当前页面对应的多叉树；根据视图的层级信息构建的多叉树，可以从整体上反应当前页面多个视图的层级结构，由此根据多叉树生成的视图的层级路径，可以对每个视图进行准确定位，从而可以准确在页面截图添加埋点边框，提高应用程序当前页面展示效果。

10 在一个实施例中，按照预设规则将多个视图标识添加至多个数组的步骤，包括：在页面数据中提取视图名称与预设根视图的视图名称相同的视图对应的视图标识，将视图名称与预设根视图的视图名称相同的视图标记为父视图，将提取到的视图标识添加至最高级别数组；根据多个视图之间的层级信息对页面数据进行遍历，在页面数据中提取父视图对应的子视图的视图标识，将子视图的视图标识添加至次高级别数组；将子视图作为父视图，在页面数据中查询是否存在对应的子视图；当页面数据中存在对应的子视图时，将对应的子视图的视图标识添加至下一级别数组，返回将子视图作为父视图，在页面数据中查询是否存在对应的子视图的步骤；当页面数据中不存在对应的子视图时，得到最低级别数组。

20 预设根视图包括预设根控件和预设根控制器。父视图包括父类控件和父类控制器，子视图包括子类控件和子类控制器。父视图对应的子视图是指父视图的下一层级视图，例如，当控件 UIButton 为控件 UIView 的下一层级控件时，则控件 UIButton 可以理解为控件 UIView 的子类控件。

25 服务器建立最高级别数组，在页面数据中提取控件名称与预设根控件的控件名称相同的控件对应的控件标识，将提取到的控件标识添加至最高级别数组。预设根控件可以是 UIWindow。服务器将提取到的控件标记为父类控件，根据该父类控件的层级信息，对页面数据进行遍历，检测是否存在该父

类控件对应的子类控件。当页面数据中存在对应的子类控件时，服务器建立次高级别数组，在页面数据中提取该父类控件对应的子类控件（如 subviews）的控件标识，提取控制器名称与预设根控制器的控制器名称相同的视图控制器的控制器标识，将提取到的视图控制器标记为父类控制器，将提取到的子类控件的控件标识以及提取到的控制器标识添加至次高级别数组。预设根控制器可以是 rootViewController（一种视图控制器）。服务器将提取到的子类控件作为父类控件，在页面数据中查询是否存在对应的子类控件，以及是否存在上述父类控制器对应的子类控制器（如 childrenViewController 和 presentedViewController）。

10 当页面数据中存在对应的子类控件，或存在父类控制器对应的子类控制器时，服务器建立下一级别数组，在页面数据中提取对应的子类控件的控件标识和/或上述父类控制器对应的子类控制器的控制器标识，将提取到的控件标识和/或控制器标识添加至下一级别数组。

服务器将重新提取到的子类控件作为父类控件，将提取到的子类控制器作为父类控制器，在页面数据中查询是否存在对应的子类控件，以及对应的子类控制器。当页面数据中存在对应的子类控件，或对应的子类控制器时，服务器按照上述方式建立再下一级别数组。如此循环，直至页面数据中不存在对应的子类控件，也不存在对应的子类控制器时，得到最低级别数组。

本实施例中，第一终端上传至服务器的页面数据仅包含单个视图的层级信息，使得多个视图之间的层级关系隐蔽且分散，服务器对分散的层级信息进行整合，根据各个视图的层级信息，分别将视图添加至不同级别的数组，通过数组的级别可以反映每个视图相对其他视图的层级关系，可以提高多叉树的构建效率，进而可以提高对应用程序当前页面的展示效率。

在一个实施例中，根据多叉树生成多个视图分别对应的层级路径的步骤，包括：根据属性信息对页面截图中的可触摸的视图进行识别；根据可触摸的视图对应的视图标识，在多叉树中查询对应的节点，将节点标记为目标节点；利用第一预设标识将根节点至目标节点的多个节点分别对应的视图名称依次

拼接，得到目录字符串，将目录字符串作为目标节点对应视图的视图路径；
对多叉树中不同层级节点对应的视图分别进行单独编号，对同一层级节点对应的不同类型视图分别进行单独编号，将编号作为对应视图的视图索引；利用第二预设标识将根节点至目标节点的多个节点分别对应的视图索引依次拼接，得到索引字符串，将索引字符串作为目标节点对应视图的索引路径。

根据节点对应数组元素所属数组的级别，不同节点在多叉树中具有不同层级。为了方便描述，规定最高级别数组中数组元素对应第一层级节点，次高级别数组中数组元素对应第二层级节点，如此递推。服务器在多叉树构建过程中，对多叉树每个节点进行编号，以使每个节点对应的视图具有视图索引。具体的，服务器对多叉树中不同层级节点对应的视图分别进行单独编号，对同一层级节点对应的不同类型视图分别进行单独编号，对同一层级节点对应的相同类型视图进行连续编号，将编号作为对应视图的视图索引。服务器可以将视图索引作为对应视图的视图名称的下标进行记录。例如，如图 3 所示，第一层级节点 UIwindow 与第二层级节点 UIwiew 对应的视图索引分别可以是初始编号。初始编号可以是 0。多个第四层级节点中不同类型视图 UIButton 和 UILabel 单独编号，三个相同的按钮类型视图对应的视图索引可以是 UIButton₀、UIButton₁ 和 UIButton₂，两个相同的标签类型视图对应的视图索引可以是 UILabel₀ 和 UILabel₁。

当应用程序当前页面新增或删除视图时，服务器仅对与新增或删除的视图相同类型的视图重新进行单独编号。例如，当将按钮控件 UIButton₁ 删除时，仅 UIButton₂ 变更为 UIButton₁，而对三个标签控件没有影响。对同一层级的不同类型的视图分别进行单独编号，可以尽量减少新增或删除视图对视图索引的影响，具有一定的索引路径抗干扰作用。

服务器根据可触摸视图的视图标识，在多叉树中查询对应的节点，将该节点标记为目标节点。服务器利用第一预设标识将多叉树中根节点至目标节点的多个节点分别对应的视图名称依次拼接，得到目录字符串，将目录字符串作为目标节点对应视图的视图路径。服务器利用第二预设标识将多叉树中

根节点至目标节点的多个节点分别对应的视图索引依次拼接，得到索引字符串，将索引字符串作为目标节点对应视图的索引路径。第一预设标识和第二预设标识分别可以是“/”，“_”，“#”，“@”或“*”等。

组成视图路径的各个视图名称分别对应不同层级的视图，故视图路径可以
5 可以对不同层级的视图进行区分。例如，假设上述举例中的控件 UIButton₁ 对应的视图路径为/UIwindow/ViewController/UIview/UIButton，则表示 UIwindow 为最高层级视图，ViewController 为 UIwindow 的下一层级视图，UIview 为 ViewController 的下一层级视图，UIButton 为 UIview 的下一层级视图。

多个视图可以继承于同一视图，即多个视图分别对应的上一层级视图可
10 以是同一视图，如控件 UIButton₀，控件 UIButton₁ 和 控件 UIButton₂ 可以继承于同一控件 UIview。索引路径可以反映对一个视图在对应的上一层级视图中的显示层级关系，即可以对同一层级的不同视图进行区分。例如，假设上述举例中的控件 UIButton₁ 对应的索引路径为 0_1_2_1，则表示 UIwindow 为最高层级视图，视图控制器 ViewController 为上一层级视图 UIwindow 的第 2 个
15 子视图，控件 UIview 为上一层级视图 ViewController 的第 3 个子视图，控件 UIButton 为上一层级视图 UIview 的第 2 个子视图。

本实施例中，视图路径可以对不同层级的视图进行区分，索引路径可以对同一层级的不同视图进行区分，从而视图路径和索引路径的结合可以共同完成控件唯一性定位，从而可以在页面截图上准确添加埋点边框。

20 在一个实施例中，属性信息包括控件名称、基本属性和访问属性；根据多叉树生成多个视图分别对应的层级路径的步骤，包括：根据基本属性识别控件是否为可触摸控件；当控件不是可触摸控件时，识别控件名称是否为目标集合中的集合元素及访问属性是否具有预设的属性值；当控件名称为目标集合中的集合元素，且访问属性具有预设的属性值时，将控件标记为可触
25 摸控件；根据多叉树生成多个可触摸控件分别对应的层级路径。

当应用程序为原生 APP 时，第一终端发送至服务器的属性信息包括控件名称和基本属性。基本属性包括透明属性、隐藏属性和交互属性等。交互属

性包括 `userInteractionEnabled`（一种属性名称）属性。SDK 还获取应用程序当前页面的页面截图，将页面截图以及多个控件的属性信息发送至服务器。

服务器存储了预设集合。预设集合记录了可以进行触摸操作的多个控件的控件名称，如 `UIControl`，`UITableViewCell`、`UICollectionViewCell`，`UIView` 5 及分别对应的子类控件的控件名称。触摸操作包括点击、长按、短按或滑动等。服务器识别控件的控件名称是否属于预设集合中的集合元素。当不属于时，表示该控件不可以进行触摸操作。当属于时，服务器获取该控件的透明属性、隐藏属性和交互属性各自的属性值，识别三个属性值是否分别达到各自的预设属性值。当三个属性值均为各自对应的预设属性值时，表示该控件 10 是可见不透明的，且可以与用户交互，即可以响应用户的触摸操作，则服务器将该控件标记为可触摸控件。

Web APP 或者混合 APP 中存在跨平台页面，如 H5（HTML5，超文本标记语言）页面，跨平台页面中控件的控件名称并不属于预设集合中的集合元素。当根据基本属性将控件识别为不可以进行触摸操作的控件时，服务器识 15 别控件的控件名称是否属于目标集合中的集合元素。目标集合记录了可以进行触摸操作的多个控件的控件名称，如 `RCTView`（一种控件名称）及其子类控件的控件名称。当不属于目标集合中的集合元素时，表示该控件不可以进行触摸操作。当应用程序为 Web APP 或者混合 APP 时，第一终端利用 SDK 获取的属性信息还包括访问属性。访问属性包括 `isAccessibilityElement`（一种 20 属性名称）属性。当属于目标集合中的集合元素时，服务器获取访问属性的属性值，判断该属性值是否为预设的属性值。当控件名称为目标集合中的集合元素，且访问属性具有预设属性值时，服务器将控件标记为可触摸控件。服务器按上述方式生成可触摸控件的层级路径，根据层级路径对可触摸控件添加对应的埋点边框，将添加了埋点边框的页面截图发送至第二终端。

25 本实施例中，当应用程序为原生 APP 时，根据基本属性即可对可触摸控件进行识别；当应用程序为 Web APP 或者混合 APP 时，可以进一步根据访问属性对可触摸控件进行识别，使得该可触摸控件识别方法对于多种类型的

应用程序均可通用，可提高可触摸控件的识别精度。仅需根据多叉树生成可触摸控件的层级路径，而非全部视图的层级路径，可以减少服务器资源占用。

在一个实施例中，根据层级路径在页面截图对相应的视图添加对应的埋点边框包括：对多叉树进行遍历，依次对多叉树中的多个节点进行编号，将
5 编号作为相应节点对应的视图的重叠索引；根据层级路径在页面截图对相应视图添加对应的埋点边框；当多个可触摸的视图对应的埋点边框发生重叠时，获取相应视图对应的重叠索引，根据重叠索引对埋点边框的位置进行调整。

当两个相邻层级的视图均可以响应触摸操作或弹窗显示时，会出现多个视图发生遮挡的情况，如图 4 所示，“消息”“联系人”和“动态”三个视图将“聊
10 天好友 7”对应的视图遮挡。服务器根据层级路径对可触摸视图添加对应的埋点边框。当多个视图发生遮挡时，多个视图对应的埋点边框也会出现交叉或重叠。为了解决埋点边框交叉或重叠问题，服务器对应用程序当前页面每个视图分配对应的重叠索引。具体的，服务器对多叉树进行层序遍历，依次对多叉树中的多个节点进行编号，将该编号作为节点对应视图的重叠索引。根
15 据遍历顺序，编号逐渐增大或者逐渐减小，以使服务器根据编号可以区分不同节点。层序遍历是指对多叉树中不同层级的节点从高层级向低层级进行逐层遍历，对同一层级的多个节点进行顺序遍历。

当多个视图对应的埋点边框发生重叠时，服务器获取该视图对应的重叠索引，根据重叠索引对多个视图对应的埋点边框进行展示。假设根据遍历顺
20 序，重叠索引逐渐增大，则服务器将重叠索引大的视图对应的埋点边框置于重叠索引小的视图对应的埋点边框之上进行展示。或者，服务器将重叠索引小的视图对应的埋点边框中，与被重叠索引大的视图对应的埋点边框发生交叉或重叠的部分省略展示。例如，如图 4 所示，将“聊天好友 7”视图对应的埋点边框中，与“消息”“联系人”和“动态”三个视图存在重叠的部分省略展示。

25 本实施例中，在根据层级路径对埋点边框进行准确定位的基础上，进一步通过对应用程序当前页面每个视图分配对应的重叠索引，对埋点边框的显示层级进行准确定位，避免埋点边框交叉或重叠的问题，从而可以进一步提

高对应用程序当前页面的展示效果。

在一个实施例中，如图 5 所示，提供了一种页面展示方法，以该方法应用于第三终端为例进行说明，具体包括以下步骤：

步骤 502，获取应用程序当前页面的页面数据；页面数据包括页面截图
5 以及多个视图的视图标识、视图名称和层级信息。

步骤 504，将页面数据发送至服务器，使服务器按照预设规则将多个视图标识作为数组元素添加至多个数组，多个数组具有不同的级别，并根据多个视图之间的层级信息将最高级别数组中的视图标识对应的视图名称标记为父节点，将次高级别数组中的视图标识对应的视图名称标记为父节点的子节点，将子节点作为下一级别数组中的视图标识对应的视图名称的父节点进行
10 迭代，直至最低级别数组，利用父节点和对应的子节点生成当前页面对应的多叉树，根据多叉树生成多个视图分别对应的层级路径，并根据多叉树生成多个视图分别对应的层级路径，并根据层级路径在页面截图对相应的视图添加对应的埋点边框，将设有埋点边框的页面截图发送至预设终端，使预设终
15 端对设有埋点边框的页面截图进行展示。

步骤 506，接收服务器返回的多个被埋点视图对应的层级路径；其中，层级路径是服务器检测到预设终端多个视图对应的埋点操作时根据多叉树生成的。

步骤 508，当拦截到视图对应的触摸事件时，根据层级路径对被触摸视图和被埋点视图进行匹配，并在匹配成功时将触摸事件进行上报。
20

服务器根据第一终端上报的应用程序当前页面的页面数据，构建应用程序当前页面对应的多叉树，并根据应用程序当前页面多个视图的属性信息识别可触摸视图，根据多叉树生成多个可触摸视图分别对应的层级路径，根据层级路径在页面截图对多个可触摸视图添加对应的埋点边框。服务器将设有埋点边框的页面截图发送至第二终端。第二终端对设有埋点边框的页面截图进行展示。应用程序开发商可以通过在第二终端在页面截图上点击埋点边框，并输入对应的触摸事件标识，实现埋点设置。当第二终端获取到埋点边框对
25

应的点击操作时，获取被埋点的视图（以下称“被埋点视图”）的埋点信息，将被埋点视图的埋点信息发送至服务器。被埋点视图包括被埋点控件。服务器根据多叉树生成每个被埋点控件的层级路径。层级路径包括视图路径和索引路径。当用户通过第三终端对应用程序当前页面的某个控件进行触摸操作

5 时，第三终端利用 SDK 对被触摸的视图（以下称“被触摸视图”）对应的触摸事件进行拦截，获取被触摸视图的视图路径和索引路径，将被触摸视图的视图路径和索引路径分别与被埋点视图的视图路径和索引路径进行匹配。当被触摸视图与被埋点视图的视图路径与索引路径分别匹配成功时，标识该被触摸视图属于被埋点视图，第三终端利用 SDK 将被触摸视图对应的触摸事件上

10 报至服务器。

本实施例中，根据应用程序当前页面对应的多叉树，可以从整体上反应当前页面多个视图的层级结构，由此根据多叉树生成的视图的层级路径，可以对每个视图进行准确定位，从而可以准确在页面截图添加埋点边框；通过点击埋点边框可以对应用程序设置埋点，终端根据被埋点控件的层级路径可

15 以对被触摸控件的触摸事件进行准确上报。

在一个实施例中，获取应用程序页面的页面数据的步骤，包括：接收服务器发送的截图请求；截图请求携带了上次发送至服务器的页面截图对应的截图标识；对应用程序当前的页面进行截图，得到当前的页面截图，获取当前的页面截图对应的截图标识；将页面数据发送至服务器的步骤，包括：将

20 上次页面截图的截图标识与当前页面截图的截图标识进行对比；当上次页面截图的截图标识与当前页面截图的截图标识不一致时，将当前的页面截图及对应的截图标识发送至服务器。

第一终端向服务器发送可视化埋点请求，以主动触发设置埋点时，或当服务器检测到第二终端显示的页面截图关闭时，服务器向第一终端发送截图

25 请求，以提示第一终端此时可以进行下一次的数据传输。截图请求携带了第一终端上次发送至服务器的页面截图对应的截图标识。截图标识可以是 md5 值（Message-Digest Algorithm 5，信息-摘要算法 5）等。第一终端利用 SDK

对应用程序当前页面进行截图，获取应用程序当前页面的页面截图及对应的截图标识。SDK 对比截图请求携带的截图标识与当前页面的截图标识是否一致。当截图标识不一致时，向服务器发送当前页面的页面截图及对应的多个视图的属性信息；否则，向服务器返回重复截图提示，而不再向服务器发送当前页面的页面数据。本实施例只有在截图请求携带的截图标识与当前页面的截图标识不一致时，才进行页面数据的上传，可以减少不必要的重复传输，从而可以减少网络资源的占用。

在一个实施例中，层级路径包括视图路径和索引路径；根据层级路径对被触摸视图和被埋点视图进行匹配，并在匹配成功时将触摸事件进行上报的步骤，包括：获取被触摸视图的视图路径和索引路径；获取服务器发送的多个被埋点视图的视图路径和索引路径；将被触摸视图的视图路径与被埋点视图的视图路径进行匹配，将被触摸视图的索引路径与被埋点视图的索引路径进行匹配；当被触摸视图与被埋点视图的视图路径和索引路径分别匹配成功时，将被触摸视图对应的触摸事件上报至服务器。

第三终端按照预设规则在应用程序对应的安装文件中，查询预设类视图对应的响应函数。预设规则可以是利用系统方法，如 runtime(一种系统方法)。安装文件可以是 iPA 文件 (iPhoneApplication, 苹果程序应用文件)，APK 文件 (Android Package, 安卓程序应用文件) 等，包括多个视图对应的响应函数。预设类视图包括 UIControl (一种视图名称)，UITableView，UICollectionView，UIView 以及分别对应的子类视图。第三终端在查询到预设类视图对应的响应函数中，插入预先编写的触摸事件拦截函数。应当理解，不同预设类视图对应的响应函数可以不同，对应的触摸事件拦截函数的插入位置也可以不同。触摸事件拦截函数为 SDK 的触摸事件拦截脚本中的方法函数，从而可以利用 SDK 对触摸事件进行拦截。

第三终端利用 SDK 对服务器发送的被埋点视图的视图路径和索引路径进行缓存。当用户通过第三终端对应用程序当前页面某个控件进行触摸操作时，第三终端利用 SDK 对被触摸视图对应的触摸事件进行拦截，读取缓存的

应用程序当前页面多个视图的属性信息。视图包括控件和视图控制器。在应用程序当前页面，视图控制器用于创建和管理控件，对用户是不可见的，但是构建层级结构所必须的。属性信息包括视图名称，视图索引和层级信息。

视图索引可以是 SDK 根据应用程序当前页面多个视图之间的层级关系预先给多个视图分配的。SDK 根据该层级关系对不同层级的视图分别进行单独编号，对同一层级的不同类型的视图分别进行单独编号，对同一层级的相同类型视图进行连续编号，将编号作为相应视图对应的视图索引。

SDK 根据属性信息生成被触摸视图对应的视图路径和索引路径。具体的，SDK 根据被触摸视图的层级信息向上逐级回溯，即识别被触摸视图是否存在上一层级视图。当存在上一层级视图时，提取该上一层级视图的视图名称和视图索引。SDK 根据被触摸视图的层级信息继续向上逐级回溯，即识别被触摸视图的上一层级视图是否仍存在对应的上一层级视图，若存在，提取对应的视图名称和视图索引，直至最高层级视图。

根据被触摸视图的层级信息，SDK 利用第一预设标识将提取到的多个视图名称逐级拼接，即将最高层级视图至被触摸视图的多个视图的视图名称依次拼接，得到目录字符串，将该目录字符串作为被触摸视图对应的视图路径。SDK 利用第二预设标识将提取到的多个视图索引逐级拼接，即将最高层级视图至被触摸视图的多个视图的视图索引依次拼接，得到索引字符串，将该索引字符串作为被触摸视图对应的索引路径，从而得到被触摸视图对应的层级路径。第一预设标识和第二预设标识分别可以是“/”，“_”，“#”，“@”或“*”等。

组成视图路径的各个视图名称分别对应不同层级的视图，故视图路径可以对不同层级的视图进行区分。例如，假设上述举例中的控件 UIButton₁ 对应的视图路径为/UIwindow/ViewController/UIView/UIButton，则表示 UIwindow 为最高层级视图，ViewController 为 UIwindow 的下一层级视图，如此类推。

多个视图可以继承于同一视图，即多个视图分别对应的上一层级视图可以是同一视图，如控件 UIButton₀，控件 UIButton₁ 和 控件 UIButton₂ 可以继承于同一控件 UIView。索引路径可以反映对一个视图在对应的上一层级视图中

的显示层级关系，即可以对同一层级的不同视图进行区分。例如，假设上述举例中的控件 UIButton₁ 对应的索引路径为 0_1_2_1，则表示 UIWindow 为最高层级视图，视图控制器 ViewController 为上一层级视图 UIWindow 的第 2 个子视图，如此类推。

- 5 视图路径可以对不同层级的视图进行区分，索引路径可以对同一层级的不同视图进行区分，从而视图路径和索引路径的结合可以共同完成控件唯一性定位。可以理解，层级路径也可以将视图路径与索引路径结合展示。例如，将视图索引作为对应视图的视图名称的下标，利用第一预设标识或第二预设标识按照上述方式将多个带有下标的视图名称进行逐级拼接，得到如下层级
- 10 路径：/UIwindow₀/ViewController₁/UIView₂/UIButton₁。

- SDK 将被触摸视图的视图路径与被埋点视图的视图路径进行匹配，将被触摸视图的索引路径与被埋点视图的索引路径进行匹配。当存在被埋点视图的视图路径与被触摸视图的视图路径相匹配，且该被埋点视图的索引路径与被触摸视图的索引路径相匹配时，将被触摸视图对应的触摸事件上报至服务器。
- 15 应当理解，接收触摸事件上报信息的服务器与向第三终端下发埋点信息的服务器可以相同，也可以不同。

- 本实施例中，视图路径可以对不同层级视图对应的控件进行区分，索引路径可以对同一层级视图对应的不同控件进行区分，视图路径和索引路径的结合可以对控件准确定位，从而能够提高控件匹配精度，进而可以对控件的
- 20 触摸事件进行准确上报。

- 在一个实施例中，如图 6 所示，提供了一种页面展示装置，包括：数据获取模块 602，多叉树构建模块 604 和页面展示模块 606。其中：数据获取模块 602，用于获取应用程序当前页面的页面数据；页面数据包括页面截图以及多个视图的视图标识、视图名称和层级信息。多叉树构建模块 604，用于
- 25 按照预设规则将多个视图标识添加至多个数组；多个数组具有不同的级别；查询最高级别数组中的视图标识对应的视图名称，将最高级别数组中的视图标识对应的视图名称标记为父节点；查询次高级别数组中的视图标识对应的

视图名称，根据多个视图之间的层级信息，将次高级别数组中的视图标识对应的视图名称标记为父节点的子节点；将子节点作为下一级别数组中的视图标识对应的视图名称的父节点进行迭代，直至最低级别数组；利用父节点以及对应的子节点生成应用程序当前页面对应的多叉树。页面展示模块 606，

- 5 用于根据多叉树生成多个视图分别对应的层级路径，根据层级路径在页面截图对相应的视图添加对应的埋点边框，将设有埋点边框的页面截图发送至预设终端，使预设终端展示设有埋点边框的页面截图。

在一个实施例中，提供了一种计算机设备，如图 7 所示，该计算机设备包括通过系统总线连接的处理器、存储装置和网络接口。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装置包括一个或多个非易失性存储介质、一个或多个内存储器。该服务器的非易失性存储介质存储有操作系统和计算机可读指令。该计算机设备的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。该计算机设备的网络接口用于与外部的终端通过网络连接通信，比如，接收第一终端发送的应用程序当前页面的页面截图及多个视图的属性信息等。该计算机可读指令被处理器执行时以实现一种页面展示方法。

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，前述计算机可读指令可存储于一非易失性存储介质中，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，存储介质可为磁碟、光盘、只读存储记忆体（ReCd-Only Memory，ROM）等。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。以上实施例仅表达了本申请的几种实施方式，并不能因此而理解为对发明专利范围的限制。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。