(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 108614722 B
(45)授权公告日 2020.09.08

(21)申请号 201810442588.4

(22)申请日 2018.05.10

(65)同一申请的已公布的文献号
申请公布号 CN 108614722 A

(43)申请公布日 2018.10.02

(73)专利权人 上海瑾盛通信科技有限公司
地址 200232 上海市徐汇区龙华中路600号2101室

(72)发明人 韩世广 陈岩

(74)专利代理机构 北京品源专利代理有限公司 11332
代理人 孟金喆

(51)Int.Cl.
G06F 9/445(2018.01)

(56)对比文件
CN 104657183 A,2015.05.27
CN 106201241 A,2016.12.07
CN 103106000 A,2013.05.15
CN 105657522 A,2016.06.08

审查员 尹文博

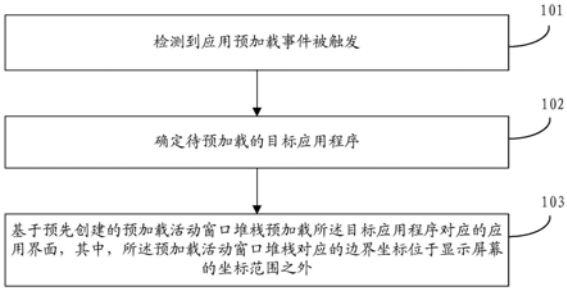
权利要求书2页 说明书11页 附图6页

(54)发明名称

应用程序预加载方法、装置、存储介质及终端

(57)摘要

本申请实施例公开了应用程序预加载方法、装置、存储介质及终端。该方法包括：检测到应用预加载事件被触发；确定待预加载的目标应用程序；基于预先创建的预加载活动窗口堆栈预加载目标应用程序对应的应用界面，其中，预加载活动窗口堆栈对应的边界坐标位于显示屏幕的坐标范围之外。本申请通过采用上述技术方案，可以基于在显示屏幕外面创建的预加载活动窗口堆栈对需要进行预加载的应用程序的应用界面进行预加载，该预加载方式能够较大程度地完成应用启动前的准备工作，提升目标应用程序的启动速度，且不会影响到前台应用程序的显示内容在显示屏幕上的显示。



CN 108614722 B

1.一种应用程序预加载方法,其特征在于,包括:

检测到应用预加载事件被触发;

确定待预加载的目标应用程序;

基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面,其中,所述预加载活动窗口堆栈对应的预加载显示区域位于显示屏幕的坐标范围之外;

向所述目标应用程序发送伪造焦点通知,使所述目标应用程序获得伪造焦点并对系统可见,并基于所述伪造焦点通知在预设时间段内保持所述目标应用程序对应的应用界面的持续绘制以及显示更新,其中,所述伪造焦点与前台应用对应的焦点相互独立;

在接收到目标应用程序的运行指令时,将所述预加载活动窗口堆栈中包含的所述目标应用程序对应的任务栈迁移至应用活动窗口堆栈的顶部,以将当前在所述预加载活动窗口堆栈中显示的应用界面迁移到显示屏幕中进行显示。

2.根据权利要求1所述的方法,其特征在于,所述基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面,包括:

判断是否存在预先创建的预加载活动窗口堆栈;

若不存在,则按照预设规则创建预加载活动窗口堆栈;

基于所创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面。

3.根据权利要求1所述的方法,其特征在于,所述基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面,包括:

创建所述目标应用程序对应的目标进程;

在预先创建的预加载活动窗口堆栈中创建所述目标应用程序对应的任务栈;

基于所述目标进程在所述任务栈中启动所述目标应用程序对应的活动窗口;

基于所启动的活动窗口绘制并显示所述目标应用程序对应的应用界面。

4.根据权利要求3所述的方法,其特征在于,所述预设时间段的长度包括所述目标应用程序中启动广告或启动动画的播放时长。

5.根据权利要求1所述的方法,其特征在于,所述预加载活动窗口堆栈对应的边界坐标为(H,0,2H,H),所述边界坐标对应的坐标系为系统坐标,所述系统坐标的原点为所述显示屏幕的左上角,H为所述显示屏幕的显示区域的长边长度。

6.一种应用程序预加载装置,其特征在于,包括:

预加载事件检测模块,用于检测应用预加载事件是否被触发;

应用确定模块,用于在检测到应用预加载事件被触发后,确定待预加载的目标应用程序;

预加载模块,用于基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面,其中,所述预加载活动窗口堆栈对应的预加载显示区域位于显示屏幕的坐标范围之外;

伪造焦点模块,用于向所述目标应用程序发送伪造焦点通知,并基于所述伪造焦点通知在预设用于向所述目标应用程序发送伪造焦点通知,使所述目标应用程序获得伪造焦点并对系统可见,并基于所述伪造焦点通知在预设时间段内保持所述目标应用程序对应的应用界面的持续绘制以及显示更新,其中,所述伪造焦点与前台应用对应的焦点相互独立;

界面迁移模块,用于在接收到目标应用程序的运行指令时,将所述预加载活动窗口堆

栈中包含的所述目标应用程序对应的任务栈迁移至应用活动窗口堆栈的顶部，以将当前在所述预加载活动窗口堆栈中显示的应用界面迁移到显示屏幕中进行显示。

7.一种计算机可读存储介质，其上存储有计算机程序，其特征在于，该程序被处理器执行时实现如权利要求1-5中任一所述的应用程序预加载方法。

8.一种终端，其特征在于，包括存储器，处理器及存储在存储器上并可在处理器运行的计算机程序，所述处理器执行所述计算机程序时实现如权利要求1-5任一所述的应用程序预加载方法。

应用程序预加载方法、装置、存储介质及终端

技术领域

[0001] 本申请实施例涉及应用程序加载技术领域，尤其涉及应用程序预加载方法、装置、存储介质及终端。

背景技术

[0002] 目前，诸如智能手机、平板电脑、笔记本电脑以及智能家电等终端已成为人们日常生活中必不可少的电子设备。随着终端设备不断地智能化，多数终端设备中都装载有操作系统，使得终端设备能够安装丰富多样的应用程序，满足用户不同的需求。

[0003] 随着终端设备的配置不断提升，多数终端设备中可以安装几十甚至上百个应用程序，而随着应用程序的功能日益丰富，应用程序运行时所需加载的资源也越来越多。当用户选择启动一个应用程序时，终端会对该应用程序启动所需的资源进行加载，待预加载完毕后，进入应用程序的初始界面，整个过程通常要花费数秒甚至十几秒的时间，导致应用程序的启动效率较低，亟需改进。

发明内容

[0004] 本申请实施例提供一种应用程序预加载方法、装置、存储介质及终端，可以优化应用程序的加载方案。

[0005] 第一方面，本申请实施例提供了一种应用程序预加载方法，包括：

[0006] 检测到应用预加载事件被触发；

[0007] 确定待预加载的目标应用程序；

[0008] 基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面，其中，所述预加载活动窗口堆栈对应的边界坐标位于显示屏幕的坐标范围之外。

[0009] 第二方面，本申请实施例提供了一种应用程序预加载装置，包括：

[0010] 预加载事件检测模块，用于检测应用预加载事件是否被触发；

[0011] 应用确定模块，用于在检测到应用预加载事件被触发后，确定待预加载的目标应用程序；

[0012] 预加载模块，用于基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面，其中，所述预加载活动窗口堆栈对应的边界坐标位于显示屏幕的坐标范围之外。

[0013] 第三方面，本申请实施例提供了一种计算机可读存储介质，其上存储有计算机程序，该程序被处理器执行时实现如本申请实施例所述的应用程序预加载方法。

[0014] 第四方面，本申请实施例提供了一种终端，包括存储器，处理器及存储在存储器上并可在处理器运行的计算机程序，所述处理器执行所述计算机程序时实现如本申请实施例所述的应用程序预加载方法。

[0015] 本申请实施例中提供的应用程序预加载方案，检测到应用预加载事件被触发后，确定待预加载的目标应用程序，基于预先创建的预加载活动窗口堆栈预加载所述目标应用

程序对应的应用界面，其中，预加载活动窗口堆栈对应的边界坐标位于显示屏幕的坐标范围之外。通过采用上述技术方案，可以基于在显示屏幕外面创建的预加载活动窗口堆栈对需要进行预加载的应用程序的应用界面进行预加载，该预加载方式能够较大程度地完成应用启动前的准备工作，提升目标应用程序的启动速度，且不会影响到前台应用程序的显示内容在显示屏幕上的显示。

附图说明

[0016] 图1为本申请实施例提供的一种应用程序预加载方法的流程示意图；

[0017] 图2为本申请实施例提供的一种预加载活动窗口堆栈与显示屏幕显示区域的相对位置关系示意图；

[0018] 图3为本申请实施例提供的又一种预加载活动窗口堆栈与显示屏幕显示区域的相对位置关系示意图；

[0019] 图4为本申请实施例提供的一种应用界面迁移示意图；

[0020] 图5为本申请实施例提供的另一种应用程序预加载方法的流程示意图；

[0021] 图6为本申请实施例提供的一种应用程序预加载装置的结构框图；

[0022] 图7为本申请实施例提供的一种终端的结构示意图；

[0023] 图8为本申请实施例提供的又一种终端的结构示意图。

具体实施方式

[0024] 下面结合附图并通过具体实施方式来进一步说明本申请的技术方案。可以理解的是，此处所描述的具体实施例仅仅用于解释本申请，而非对本申请的限定。另外还需要说明的是，为了便于描述，附图中仅示出了与本申请相关的部分而非全部结构。

[0025] 在更加详细地讨论示例性实施例之前应当提到的是，一些示例性实施例被描述成作为流程图描绘的处理或方法。虽然流程图将各步骤描述成顺序的处理，但是其中的许多步骤可以被并行地、并发地或者同时实施。此外，各步骤的顺序可以被重新安排。当其操作完成时所述处理可以被终止，但是还可以具有未包括在附图中的附加步骤。所述处理可以对应于方法、函数、规程、子例程、子程序等等。

[0026] 图1为本申请实施例提供的一种应用程序预加载方法的流程示意图，该方法可以由应用程序预加载装置执行，其中该装置可由软件和/或硬件实现，一般可集成在终端中。如图1所示，该方法包括：

[0027] 步骤101、检测到应用预加载事件被触发。

[0028] 示例性的，本申请实施例中的终端可包括手机、平板电脑、笔记本电脑以及智能家电等终端设备。终端中装载有操作系统。

[0029] 示例性的，应用预加载事件的触发条件可以根据实际情况设置，本申请实施例不作具体限定。例如，可以在检测到用户的动作满足预设条件(如拿起终端、输入屏幕解锁操作或输入终端解锁操作等)时，触发应用预加载事件；或者可以在检测到前台应用程序发生变更时，触发应用预加载事件；或者可以在预加载应用的预测流程结束后，立即(或经过预设时长后)触发应用预加载事件；或者可以定时间隔触发等等。当应用预加载事件被触发后，系统可以通过读取标志位或接收触发指令等方式检测到应用预加载事件已被触发，具

体的检测方式本申请实施例也不做限定。

[0030] 步骤102、确定待预加载的目标应用程序。

[0031] 示例性的，待预加载的应用程序可以理解为用户可能即将打开的应用程序，可以是预先设置好的固定的应用程序，也可以是采用一定的方式预测出来的应用程序等。待预加载的应用程序可以包括一个或多个。目标应用程序可以理解为当前需要进行预加载操作的待预加载的应用程序。目标应用程序也可以包括一个或多个。当待预加载的应用程序仅有一个时，目标应用程序相应的也仅有一个；当待预加载的应用程序有多个时，目标应用程序可以为一个(即将多个待预加载的应用程序逐一确定为目标应用程序，并依次进行预加载操作)，也可以为多个(即将2个以上待预加载的应用程序确定为目标应用程序，并同时进行预加载操作，也就是说多个目标应用程序的预加载过程可以是并行进行的)。

[0032] 可选的，可采用预测模型对待预加载的应用程序进行预测。该预测模型可以是机器学习模型，所采用的算法可以包括循环神经网络(Recurrent neural networks,RNN)、长短期记忆(Long Short-Term Memory,LSTM)网络、门限循环单元、简单循环单元、自动编码器、决策树、随机森林、特征均值分类、分类回归树、隐马尔科夫、K最近邻(k-NearestNeighbor,KNN)算法、逻辑回归模型、贝叶斯模型、高斯模型以及KL散度(Kullback-Leibler divergence)等等。

[0033] 可选的，可在用户使用终端的过程中，采集训练样本，采用训练样本对预设初始模型进行训练，最终得到用于预测待预加载的应用程序的预测模型。示例性的，训练样本中包含的元素可包括应用程序被打开的时间序列或次序序列；可包括应用程序被打开的时间、地点以及频次等；可包括终端的运行状态，如移动数据网络的开关状态、无线热点的连接状态、所连接的无线热点的身份信息、当前运行的应用程序、上一个前景应用程序、当前应用程序在后台停留的时长、当前应用程序最近一次被切换至后台的时间、耳机插孔的插拔状态、充电状态、电池电量信息以及屏幕显示时长等等；还可包括终端中集成的传感器采集到的数据，如运动传感器、光线传感器、温度传感器以及湿度传感器等等。

[0034] 示例性的，可根据所选用的机器学习模型选择合适的样本元素，也可根据所选的样本元素确定所选用的机器学习模型，还可结合对预测准确度以及预设速度等方面的需求进行模型及样本元素的选择等，本申请实施例不做限定。

[0035] 本申请实施例中，确定待预加载的目标应用程序可以包括根据预测模型之前的输出结果确定待预加载的目标应用程序，也可以包括利用预测模型进行预测，将预测模型的当前输出结果确定待预加载的目标应用程序。

[0036] 步骤103、基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面，其中，所述预加载活动窗口堆栈对应的边界坐标位于显示屏幕的坐标范围之外。

[0037] 本申请实施例中，活动窗口可理解为一个独立的直接面向用户提供交互和操作的界面，在不同的操作系统中可能采用不同的名词来命名该界面。为了便于理解，下面以安卓(Android)操作系统为例进行下面的说明。

[0038] 在Android系统中，活动窗口被称为Activity。Activity是一个负责与用户交互的组件，其提供一个屏幕(可以理解为屏幕界面，而非实体的显示屏幕)，以供用户交互完成某项任务。在一个android应用程序中，一个Activity通常就是一个单独的屏幕，它上面可以显示一些控件也可以监听并处理用户的事件。在对Activity进行管理中，有这样两个概念：

Task(任务栈)和Stack(活动窗口堆栈)。Task对应一个应用程序,Task用于存放Activity,一个Task中可以存放一个或多个Activity,且这些Activity遵循“先进后出,后进先出”的原则。而Stack又用于对Task进行管理,通常,一个Stack对一个屏幕所需展示的各Activity所属的Task进行管理,一个Stack可以管理一个或多个Task,当然,Stack也同样遵循堆栈的基本管理原则。这里所述的屏幕并不一定是完整独立的显示屏,以“两个屏幕”为例,这两个屏幕可能只是一个完整显示屏中两个独立显示各自显示内容的区域。当然,如果终端具备两个甚至是两个以上的独立显示屏,则“两个屏幕”也可以是两个独立的显示屏。

[0039] 在Android系统中,支持多窗口模式,可包括分屏模式、画中画模式以及自由模式(FreeForm)。在多窗口模式下,应用程序所在的Stack可以有自己的尺寸(size),可以包括以终端屏幕左上角为原点的坐标系中的上下左右坐标。例如,(a,b,c,d),一般描述的是一个矩形的边界,可采用矩形左上角的坐标和右下角的坐标进行表示,即矩形的左上角坐标为(a,b),右下角坐标为(c,d),这样的矩形区域就对应Stack的size。Stack中的应用内布局是以Stack的size为准的,也就是说Activity对应的应用界面是在size的边界范围内进行显示的。

[0040] 在多窗口模式下,可以允许多个应用程序处于可见状态,包括系统和用户均可见和仅系统可见。系统和用户均可见指的是在显示屏幕上进行显示,且用户能够看到;仅系统可见指的是操作系统可见,而用户不可见,可能被前台的应用遮挡或者如本申请所要实现的在显示屏幕外显示。

[0041] 本申请实施例中,在显示屏幕外对目标应用程序的应用界面进行预加载,可以基于操作系统的多窗口机制来实现,通过多窗口机制将应用程序对应的size设在显示屏幕外而达到对用户不可见的目的,从而不会影响到前台应用程序的显示内容在显示屏幕上的显示。

[0042] 一般的,多窗口模式下,可以存在多种类型的Stack,如Home Stack表示桌面应用显示的Stack,App Stack表示第三方应用显示的Stack,还可以有其他分屏Stack等,上述三种Stack中包含的内容可在显示屏幕上显示,本申请实施例中统称为应用活动窗口堆栈。本申请实施例中,新增了预加载活动窗口堆栈(预加载Stack),用于表示预加载应用显示的Stack,并设置预加载Stack的边界坐标位于显示屏幕的坐标范围之外,待预加载的应用程序可以显示在该Stack。对于Android系统来说,可以基于Android系统的多窗口机制,新建一个专门用于显示预加载应用的Stack。本申请实施例中,新建Stack是因为可以让新建的预加载Stack拥有自己的size以及可见性,从而实现在显示屏幕外预加载的目的。

[0043] 本申请实施例中,对预加载Stack的创建时机不做限定,可以是在终端出厂前默认设置预加载Stack处于常驻状态,即预加载Stack一直存在;也可以在终端开机时或终端解锁成功后创建;还可以在应用预加载事件被触发后(可以在确定目标应用程序之前)创建等等。可选的,所述基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面,包括:判断是否存在预先创建的预加载活动窗口堆栈;若不存在,则按照预设规则创建预加载活动窗口堆栈;基于所创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面。这样设置的好处在于,在确定待预加载的目标应用程序之后,判断预加载Stack是否存在,若存在,则无需新建,若不存在,则进行创建,可以节省系统资源。可以理解的是,当目标应用程序包含多个时,也即需要在短时间内连续预加载多个应用程序时,在第

一个目标应用程序开始加载前，预加载Stack已创建完毕，那么第二个目标应用程序开始加载前，预加载Stack是存在的，可以不必进行上述判断。

[0044] 本申请实施例中，对基于预加载Stack预加载目标应用程序对应的应用界面的具体过程不做限定，例如可以基于预加载Stack的size进行应用界面的绘制并显示等。

[0045] 本申请实施例中提供的应用程序预加载方法，检测到应用预加载事件被触发后，确定待预加载的目标应用程序，基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面，其中，预加载活动窗口堆栈对应的边界坐标位于显示屏幕的坐标范围之外。通过采用上述技术方案，可以基于在显示屏幕外面创建的预加载活动窗口堆栈对需要进行预加载的应用程序的应用界面进行预加载，该预加载方式能够较大程度地完成应用启动前的准备工作，提升目标应用程序的启动速度，且不会影响到前台应用程序的显示内容在显示屏幕上的显示。

[0046] 在一些实施例中，所述基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面，包括：创建所述目标应用程序对应的目标进程；在预先创建的预加载活动窗口堆栈中创建所述目标应用程序对应的任务栈；基于所述目标进程在所述任务栈中启动所述目标应用程序对应的活动窗口；基于所启动的活动窗口绘制并显示所述目标应用程序对应的应用界面。这样设置的好处在于，能够基于屏幕坐标范围之外的预加载活动窗口堆栈对目标应用程序的应用界面进行绘制并显示，不会干扰前台应用程序的运行及显示，保证系统稳定性，同时有效提高目标应用程序启动时的速度。在创建目标进程的同时，还可包括目标进程的初始化过程。在上述步骤的执行过程中，还可能涉及其他资源的预加载，如应用服务启动、内存分配、文件内容读取及网络数据获取等，本申请实施例对其他资源的预加载过程不做限定。

[0047] 在一些实施例中，还包括：向所述目标应用程序发送伪造焦点通知，并基于所述伪造焦点通知在预设时间段内保持所述目标应用程序对应的应用界面的持续绘制以及显示更新。这样设置的好处在于，能够在目标应用程序获得焦点并对系统可见的情况下完成应用界面的绘制及显示，提高预加载的完成度，且不对前台应用的焦点产生影响。本申请实施例中的焦点又称为输入焦点，伪造焦点与前台应用对应的焦点相互独立。一般的，对于目前的Android系统来说，焦点是唯一的，例如触摸等输入操作只对焦点生效，对于输入焦点信息，系统端和应用端是一致的，系统端一旦修改输入焦点信息，就会向应用发送输入焦点信息发生变化的信息，以此方法保证系统端和应用端输入焦点信息一致。本申请实施例中，通过分离系统端和应用端输入焦点信息的方式，实现应用端可伪造焦点的目的。具体的，本申请实施例中，对预加载应用伪造焦点通知，使预加载应用具有焦点信息，而系统端的焦点信息仍然是正确的，这样处理可以使得预加载应用能够正常绘制，达到全部加载的目的。焦点存在于系统端和应用端，可认为是服务端(server)和客户端(client)，系统端记录拥有焦点的应用，应用端保存标志位(flag)标识自己是否有焦点。伪造输入焦点的时机可以是当Android的窗口系统新增窗口，需要更新焦点时，生成伪造焦点通知并进行发送。伪造焦点的方法可以是调用窗口的client端的改变窗口焦点的方法，使得该窗口获取焦点。具体的，可基于Binder机制进行伪造焦点通知的发送，Binder机制是Android系统的进程间通信的最常用的方式，采用c/s架构，即客户/服务架构。

[0048] 本申请实施例中，预设时间段可根据实际情况设计，例如可以是开始预加载后的

固定时长范围内，也可以是开始预加载到完成预加载的时段等。在一些实施例中，所述预设时间段的长度包括所述目标应用程序中启动广告或启动动画的播放时长。一些应用程序在启动的过程中，通常会播放一些广告或动画，广告或动画的播放时长通常为3秒至十几秒不等，在播放广告或动画期间，用户可能没办法进行任何操作，只能等待播放完毕，浪费用户的宝贵时间。本申请实施例这样设置的好处在于，能够在目标应用程序启动之前，在屏幕外播放完启动广告或启动动画，当目标应用程序启动时，可直接进入应用程序的主页面或其他用户可操作的界面，从而将目标应用程序的可操作时间点进一步提前，减少等待时间。

[0049] 在一些实施例中，在基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面之后，还包括：在接收到所述目标应用程序的运行指令时，将所述预加载活动窗口堆栈中包含的所述目标应用程序对应的应用界面迁移至所述显示屏幕进行显示。这样设置的好处在于，在目标应用程序真正需要启动的时候，直接将已经绘制好的应用界面迁移到显示屏幕进行显示，能够达到应用界面的快速切换，提高启动速度。

[0050] 在一些实施例中，所述将所述预加载活动窗口堆栈中包含的所述目标应用程序对应的应用界面迁移至所述显示屏幕进行显示，包括：将所述预加载活动窗口堆栈中包含的所述目标应用程序对应的任务栈迁移至应用活动窗口堆栈的顶部；更新所述任务栈的尺寸信息、配置信息和可见性，以实现所述目标应用程序对应的应用界面在所述显示屏幕进行显示。这样设置的好处在于，能够保证界面迁移过程的稳定性，保证恢复过程不会出现卡屏、黑屏或迁移速度慢等问题。

[0051] 对于一些终端来说，尤其是手机及平板电脑等移动终端，为了方便用户的使用，显示屏幕的显示方式通常包括竖屏显示和横屏显示，许多应用程序默认采用竖屏方式进行显示，而有些应用程序默认采用横屏方式进行显示(如一些网络游戏)，在终端的使用过程中，有些应用程序还会随着用户握持终端的方向而进行横竖屏显示的切换。在本申请的一些实施例中，所述预加载活动窗口堆栈对应的边界坐标为(H,0,2H,H)，所述边界坐标对应的坐标系为系统坐标，所述系统坐标的原点为所述显示屏幕的左上角，H为所述显示屏幕的显示区域的长边长度。也就是说，H对应的边是显示屏幕的显示区域的最大边，在竖屏显示时为显示屏幕的高，在横屏显示时为显示屏幕的宽。这样设置的目的是考虑到显示屏幕横屏、预加载应用横屏显示，以及一些应用程序的正常显示。图2为本申请实施例提供的一种预加载活动窗口堆栈与显示屏幕显示区域的相对位置关系示意图。如图2所示，此时显示屏幕为竖屏方式，终端系统坐标的原点为显示屏幕201的左顶点(0,0)，显示屏幕201的宽度方向为X轴，高度方向为Y轴，预加载Stack202对应的边界坐标为(H,0,2H,H)，H为屏幕高，即左边实线矩形范围内的区域为主屏幕显示区域，右边虚线矩形范围内的区域为预加载显示区域。图3为本申请实施例提供的又一种预加载活动窗口堆栈与显示屏幕显示区域的相对位置关系示意图。如图3所示，此时显示屏幕为横屏方式，终端系统坐标的原点为显示屏幕301的左顶点(0,0)，显示屏幕301的高度方向为X轴，宽度方向为Y轴，预加载Stack202对应的边界坐标为(H,0,2H,H)，H为屏幕高，即左边实线矩形范围内的区域为主屏幕显示区域，右边虚线矩形范围内的区域为预加载显示区域。

[0052] 预加载Stack各个边界如此设置的原因在于：

[0053] 左上角的横坐标为H，是为了防止横屏时显示屏幕(也可称为主屏幕)显示到预加载应用的界面，因为主屏幕除了竖屏模式，还有横屏模式，当主屏幕横屏的时候，为了防止

主屏幕显示区域显示了预加载的应用的局部，所以将预加载Stack对应的矩形区域的左上角横坐标设为屏幕高。

[0054] 左上角的纵坐标为0，是为了预加载应用能够正确计算状态栏高度。Android应用为了更好的设计用户界面(User Interface，UI)，会自定义顶部状态栏，如果上边对应的纵坐标不等于0，那么状态栏的高度可能会错误。

[0055] 右下角的横坐标为2H(2倍屏幕高)，即预加载Stack对应的矩形的宽＝屏幕高，是为了预加载Stack的size能够包含预加载时候的横屏应用(即应用界面为横屏显示方式的应用程序)。

[0056] 右下角的纵坐标为H，即预加载Stack对应的矩形的高＝屏幕高，是为了预加载Stack的size能够包含预加载时候的竖屏应用。

[0057] 基于上述的原因，发明人将预加载Stack的size设置为(H,0,2H,H)。

[0058] 此外，图4为本申请实施例提供的一种应用界面迁移示意图，如图4所示，在接收到目标应用程序的运行指令时，将预加载活动窗口堆栈中包含的目标应用程序对应的应用界面401迁移至显示屏幕201进行显示，具体的，将预加载应用界面所属的task迁移至应用活动窗口Stack的顶部，并更新该task的尺寸信息、配置信息和可见性，从而应用界面能够在显示屏幕上正常显示。

[0059] 图5为本申请实施例提供的另一种应用程序预加载方法的流程示意图，该方法包括如下步骤：

[0060] 步骤501、检测到应用预加载事件被触发。

[0061] 步骤502、确定待预加载的目标应用程序。

[0062] 步骤503、判断是否存在预先创建的预加载活动窗口堆栈，若否，则执行步骤504；若是，则执行步骤505。

[0063] 步骤504、按照预设规则创建预加载活动窗口堆栈，执行步骤505。

[0064] 步骤505、创建目标应用程序对应的目标进程，并对目标进程进行初始化。

[0065] 步骤506、在预加载活动窗口堆栈中创建目标应用程序对应的任务栈。

[0066] 步骤507、基于所述目标进程在所述任务栈中启动所述目标应用程序对应的活动窗口，并向目标应用程序发送伪造焦点通知。

[0067] 步骤508、根据所述伪造焦点通知更新目标应用程序对应的焦点标志位，在预设时段内基于所启动的活动窗口持续绘制并更新显示目标应用程序对应的应用界面。

[0068] 步骤509、在接收到目标应用程序的运行指令时，将所述预加载活动窗口堆栈中包含的所述目标应用程序对应的任务栈迁移至应用活动窗口堆栈的顶部。

[0069] 具体的，以应用活动窗口为App Stack为例，将task移动到App Stack，可包括：从原stack(即预加载Stack)中的list(列表)中删除此task，在新stack(App Stack)中顶部添加此task。每个stack都有一个list，用以记录stack包含的所有task；对应地，每个task同时会记录本task所在的stack信息。因为stack包含相对其他stack可能不同的size、configuration(配置信息)、可见性和zorder等属性，所以仅仅将task添加到新stack的list中就可以改变task的zorder等属性。添加在新stack中的顶部是为了使该task是可见的。

[0070] 步骤510、更新任务栈的尺寸信息、配置信息和可见性，以实现目标应用程序对应

的当前应用界面在显示屏幕进行显示。

[0071] 可以理解的是，在预加载过程中，目标应用程序对应的应用界面处于持续的绘制及显示更新过程中，而用户真正启动目标应用程序的时机是不确定的，可能在预加载过程中就会接收到运行指令，那么可将当前在预加载Stack中显示的应用界面迁移到显示屏幕中进行显示。若用户在预加载完成之后，打开目标应用程序，那么可将预加载完成时的最后一个应用界面迁移到显示屏幕中进行显示。

[0072] 具体的，更新任务栈的尺寸信息、配置信息是为了改变应用的边界，具体可包括：主动更新size和configuration信息。仅仅使用上面的task迁移操作无法立即改变task的size和configuration，所以可主动调用修改task size和configuration的方法，保证该应用状态正确，可以正常地在前台工作。

[0073] 具体的，更新task可见是因为，在更新了size和configuration后，此时目标应用可能还不可见，调用更新新stack可见性的方法，由顶部至底部搜索可见的应用，设置为可见，并更新对应的窗口surface可见。其中，可见的应用判断方法可包括：在zorder上未被其他应用遮挡；未被锁屏遮挡；应用的顶部activity的状态是非正在初始化或者非退出状态。

[0074] 本申请实施例提供的应用程序预加载方法，可以基于在显示屏幕外面创建的预加载活动窗口堆栈对需要进行预加载的应用程序的应用界面进行绘制及显示，较大程度地完成应用启动前的准备工作，且不会影响到前台应用程序的显示内容在显示屏幕上的显示，当目标应用程序被真正启动时，直接将预加载好的应用界面迁移到显示屏幕进行显示，从而有效提升启动速度。

[0075] 图6为本申请实施例提供的一种应用程序预加载装置的结构框图，该装置可由软件和/或硬件实现，一般集成在终端中，可通过执行应用程序预加载方法来进行应用程序的预加载。如图6所示，该装置包括：

[0076] 预加载事件检测模块601，用于检测应用预加载事件是否被触发；

[0077] 应用确定模块602，用于在检测到应用预加载事件被触发后，确定待预加载的目标应用程序；

[0078] 预加载模块603，用于基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面，其中，所述预加载活动窗口堆栈对应的边界坐标位于显示屏幕的坐标范围之外。

[0079] 本申请实施例提供的应用程序预加载装置，检测到应用预加载事件被触发后，确定待预加载的目标应用程序，基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面，其中，预加载活动窗口堆栈对应的边界坐标位于显示屏幕的坐标范围之外。通过采用上述技术方案，可以基于在显示屏幕外面创建的预加载活动窗口堆栈对需要进行预加载的应用程序的应用界面进行预加载，该预加载方式能够较大程度地完成应用启动前的准备工作，提升目标应用程序的启动速度，且不会影响到前台应用程序的显示内容在显示屏幕上的显示。

[0080] 可选的，所述基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面，包括：

[0081] 判断是否存在预先创建的预加载活动窗口堆栈；

[0082] 若不存在，则按照预设规则创建预加载活动窗口堆栈；

[0083] 基于所创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面。

[0084] 可选的，所述基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面，包括：

[0085] 创建所述目标应用程序对应的目标进程；

[0086] 在预先创建的预加载活动窗口堆栈中创建所述目标应用程序对应的任务栈；

[0087] 基于所述目标进程在所述任务栈中启动所述目标应用程序对应的活动窗口；

[0088] 基于所启动的活动窗口绘制并显示所述目标应用程序对应的应用界面。

[0089] 可选的，该装置还包括：

[0090] 伪造焦点模块，用于向所述目标应用程序发送伪造焦点通知，并基于所述伪造焦点通知在预设时间段内保持所述目标应用程序对应的应用界面的持续绘制以及显示更新。

[0091] 可选的，所述预设时间段的长度包括所述目标应用程序中启动广告或启动动画的播放时长。

[0092] 可选的，该装置还包括：

[0093] 界面迁移模块，用于在基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面之后，在接收到所述目标应用程序的运行指令时，将所述预加载活动窗口堆栈中包含的所述目标应用程序对应的应用界面迁移至所述显示屏幕进行显示。

[0094] 可选的，所述将所述预加载活动窗口堆栈中包含的所述目标应用程序对应的应用界面迁移至所述显示屏幕进行显示，包括：

[0095] 将所述预加载活动窗口堆栈中包含的所述目标应用程序对应的任务栈迁移至应用活动窗口堆栈的顶部；

[0096] 更新所述任务栈的尺寸信息、配置信息和可见性，以实现所述目标应用程序对应的应用界面在所述显示屏幕进行显示。

[0097] 可选的，所述预加载活动窗口堆栈对应的边界坐标为(H,0,2H,H)，所述边界坐标对应的坐标系为系统坐标，所述系统坐标的原点为所述显示屏幕的左上角，H为所述显示屏幕的显示区域的长边长度。

[0098] 本申请实施例还提供一种包含计算机可执行指令的存储介质，所述计算机可执行指令在由计算机处理器执行时用于执行应用程序预加载方法，该方法包括：

[0099] 检测到应用预加载事件被触发；

[0100] 确定待预加载的目标应用程序；

[0101] 基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面，其中，所述预加载活动窗口堆栈对应的边界坐标位于显示屏幕的坐标范围之外。

[0102] 存储介质——任何的各种类型的存储器设备或存储设备。术语“存储介质”旨在包括：安装介质，例如CD-ROM、软盘或磁带装置；计算机系统存储器或随机存取存储器，诸如DRAM、DDRRAM、SRAM、EDORAM，兰巴斯(Rambus)RAM等；非易失性存储器，诸如闪存、磁介质(例如硬盘或光存储)；寄存器或其它相似类型的存储器元件等。存储介质可以还包括其它类型的存储器或其组合。另外，存储介质可以位于程序在其中被执行的第一计算机系统中，或者可以位于不同的第二计算机系统中，第二计算机系统通过网络(诸如因特网)连接到第一计算机系统。第二计算机系统可以提供程序指令给第一计算机用于执行。术语“存储介质”可以包括可以驻留在不同位置中(例如在通过网络连接的不同计算机系统中)的两个或更多

存储介质。存储介质可以存储可由一个或多个处理器执行的程序指令(例如具体实现为计算机程序)。

[0103] 当然,本申请实施例所提供的一种包含计算机可执行指令的存储介质,其计算机可执行指令不限于如上所述的应用程序预加载操作,还可以执行本申请任意实施例所提供的应用程序预加载方法中的相关操作。

[0104] 本申请实施例提供了一种终端,该终端中可集成本申请实施例提供的应用程序预加载装置。图7为本申请实施例提供的一种终端的结构示意图。终端700可以包括:存储器701,处理器702及存储在存储器701上并可在处理器702运行的计算机程序,所述处理器702执行所述计算机程序时实现如本申请实施例所述的应用程序预加载方法。

[0105] 本申请实施例提供的终端,可以基于在显示屏幕外面创建的预加载活动窗口堆栈对需要进行预加载的应用程序的应用界面进行预加载,该预加载方式能够较大程度地完成应用启动前的准备工作,提升目标应用程序的启动速度,且不会影响到前台应用程序的显示内容在显示屏幕上的显示。

[0106] 图8为本申请实施例提供的另一种终端的结构示意图,该终端可以包括:壳体(图中未示出)、存储器801、中央处理器(central processing unit,CPU)802(又称处理器,以下简称CPU)、电路板(图中未示出)和电源电路(图中未示出)。所述电路板安置在所述壳体围成的空间内部;所述CPU802和所述存储器801设置在所述电路板上;所述电源电路,用于为所述终端的各个电路或器件供电;所述存储器801,用于存储可执行程序代码;所述CPU802通过读取所述存储器801中存储的可执行程序代码来运行与所述可执行程序代码对应的计算机程序,以实现以下步骤:

[0107] 检测到应用预加载事件被触发;

[0108] 确定待预加载的目标应用程序;

[0109] 基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面,其中,所述预加载活动窗口堆栈对应的边界坐标位于显示屏幕的坐标范围之外。

[0110] 所述终端还包括:外设接口803、RF(Radio Frequency,射频)电路805、音频电路806、扬声器811、电源管理芯片808、输入/输出(I/O)子系统809、其他输入/控制设备810、触摸屏812、其他输入/控制设备810以及外部端口804,这些部件通过一个或多个通信总线或信号线807来通信。

[0111] 应该理解的是,图示终端800仅仅是终端的一个范例,并且终端800可以具有比图中所示出的更多的或者更少的部件,可以组合两个或更多的部件,或者可以具有不同的部件配置。图中所示出的各种部件可以在包括一个或多个信号处理和/或专用集成电路在内的硬件、软件、或硬件和软件的组合中实现。

[0112] 下面就本实施例提供的用于应用预加载的终端进行详细的描述,该终端以手机为例。

[0113] 存储器801,所述存储器801可以被CPU802、外设接口803等访问,所述存储器801可以包括高速随机存取存储器,还可以包括非易失性存储器,例如一个或多个磁盘存储器件、闪存器件、或其他易失性固态存储器件。

[0114] 外设接口803,所述外设接口803可以将设备的输入和输出外设连接到CPU802和存储器801。

[0115] I/O子系统809，所述I/O子系统809可以将设备上的输入输出外设，例如触摸屏812和其他输入/控制设备810，连接到外设接口803。I/O子系统809可以包括显示控制器8091和用于控制其他输入/控制设备810的一个或多个输入控制器8092。其中，一个或多个输入控制器8092从其他输入/控制设备810接收电信号或者向其他输入/控制设备810发送电信号，其他输入/控制设备810可以包括物理按钮(按压按钮、摇臂按钮等)、拨号盘、滑动开关、操纵杆、点击滚轮。值得说明的是，输入控制器8092可以与以下任一个连接：键盘、红外端口、USB接口以及诸如鼠标的指示设备。

[0116] 触摸屏812，所述触摸屏812是用户终端与用户之间的输入接口和输出接口，将可视输出显示给用户，可视输出可以包括图形、文本、图标、视频等。

[0117] I/O子系统809中的显示控制器8091从触摸屏812接收电信号或者向触摸屏812发送电信号。触摸屏812检测触摸屏上的接触，显示控制器8091将检测到的接触转换为与显示在触摸屏812上的用户界面对象的交互，即实现人机交互，显示在触摸屏812上的用户界面对象可以是运行游戏的图标、联网到相应网络的图标等。值得说明的是，设备还可以包括光鼠，光鼠是不显示可视输出的触摸敏感表面，或者是由触摸屏形成的触摸敏感表面的延伸。

[0118] RF电路805，主要用于建立手机与无线网络(即网络侧)的通信，实现手机与无线网络的数据接收和发送。例如收发短信息、电子邮件等。具体地，RF电路805接收并发送RF信号，RF信号也称为电磁信号，RF电路805将电信号转换为电磁信号或将电磁信号转换为电信号，并且通过该电磁信号与通信网络以及其他设备进行通信。RF电路805可以包括用于执行这些功能的已知电路，其包括但不限于天线系统、RF收发机、一个或多个放大器、调谐器、一个或多个振荡器、数字信号处理器、CODEC(COder-DECoder，编译码器)芯片组、用户标识模块(Subscriber Identity Module，SIM)等等。

[0119] 音频电路806，主要用于从外设接口803接收音频数据，将该音频数据转换为电信号，并且将该电信号发送给扬声器811。

[0120] 扬声器811，用于将手机通过RF电路805从无线网络接收的语音信号，还原为声音并向用户播放该声音。

[0121] 电源管理芯片808，用于为CPU802、I/O子系统及外设接口所连接的硬件进行供电及电源管理。

[0122] 上述实施例中提供的应用程序预加载装置、存储介质及终端可执行本申请任意实施例所提供的应用程序预加载方法，具备执行该方法相应的功能模块和有益效果。未在上述实施例中详尽描述的技术细节，可参见本申请任意实施例所提供的应用程序预加载方法。

[0123] 注意，上述仅为本申请的较佳实施例及所运用技术原理。本领域技术人员会理解，本申请不限于这里所述的特定实施例，对本领域技术人员来说能够进行各种明显的变化、重新调整和替代而不会脱离本申请的保护范围。因此，虽然通过以上实施例对本申请进行了较为详细的说明，但是本申请不仅仅限于以上实施例，在不脱离本申请构思的情况下，还可以包括更多其他等效实施例，而本申请的范围由所附的权利要求范围决定。

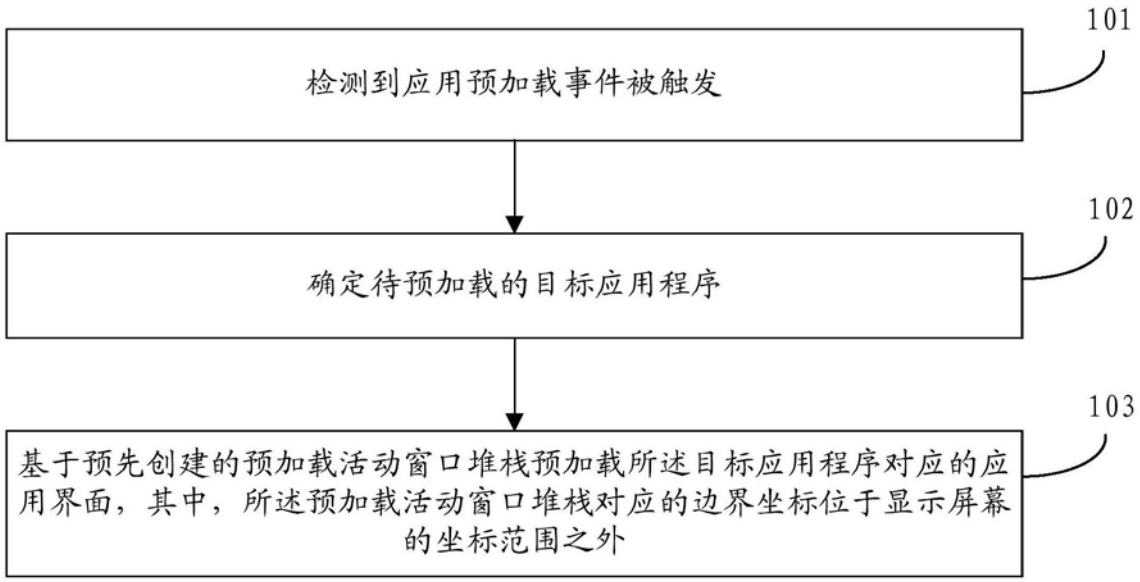

101
检测到应用预加载事件被触发
102
确定待预加载的目标应用程序
103
基于预先创建的预加载活动窗口堆栈预加载所述目标应用程序对应的应用界面，其中，所述预加载活动窗口堆栈对应的边界坐标位于显示屏幕的坐标范围之外

图1

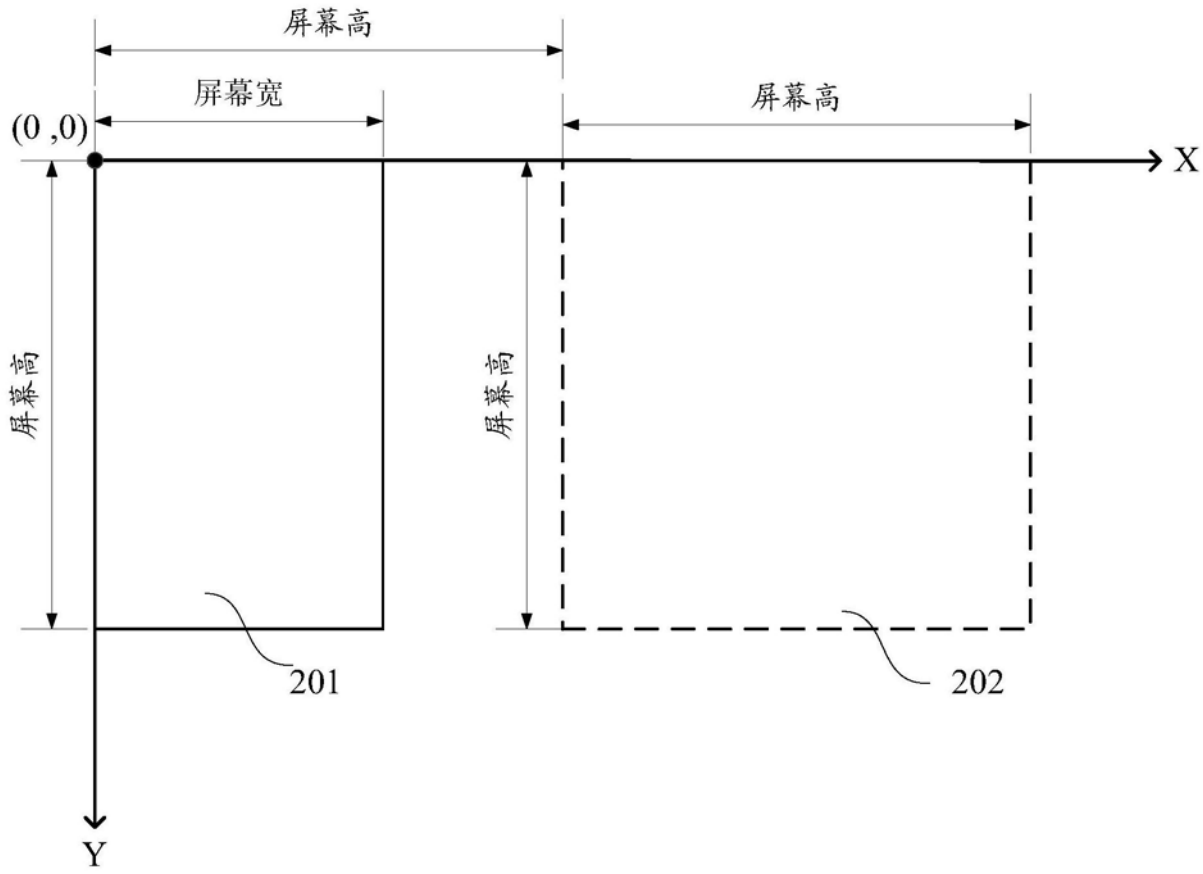

屏幕高
屏幕宽
屏幕高
(0 ,0)
X
屏幕高
屏幕高
201
202
Y

图2

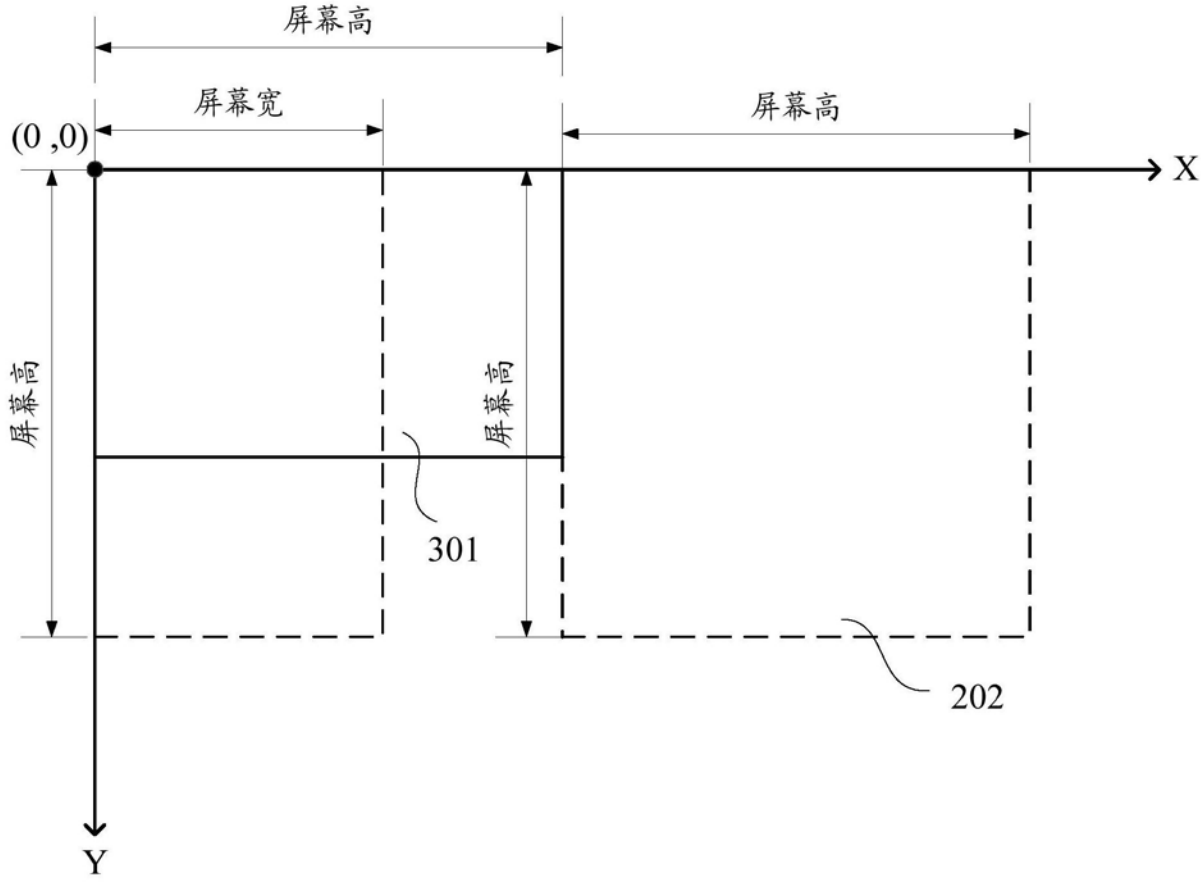


图3

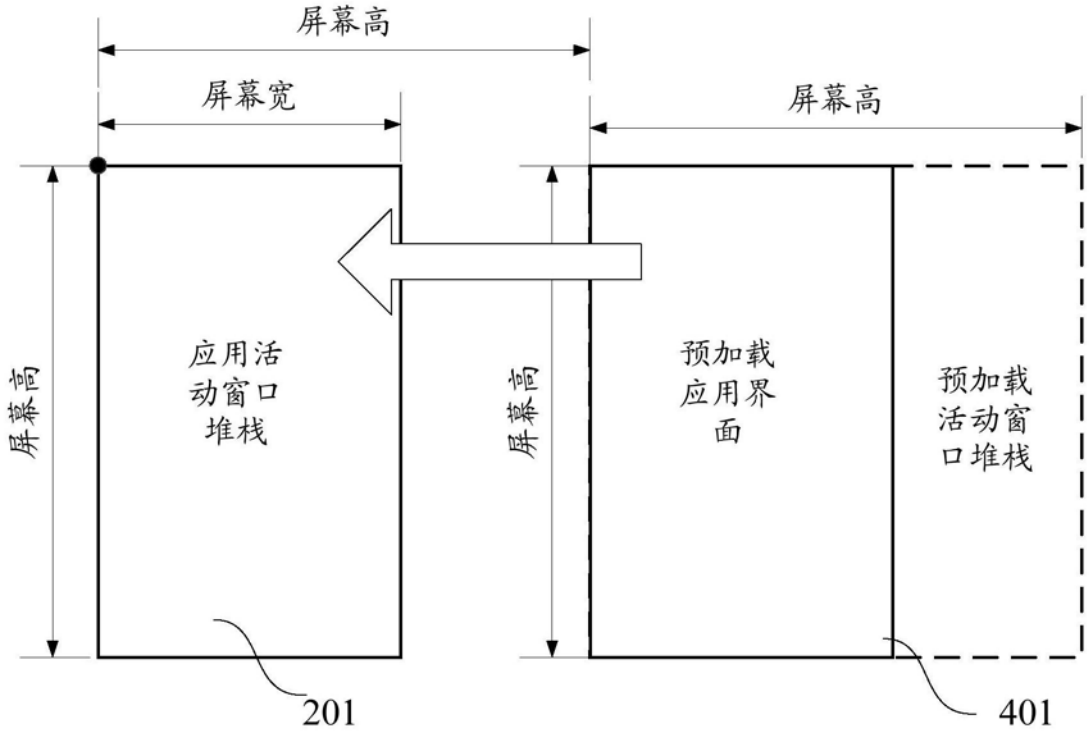


图4

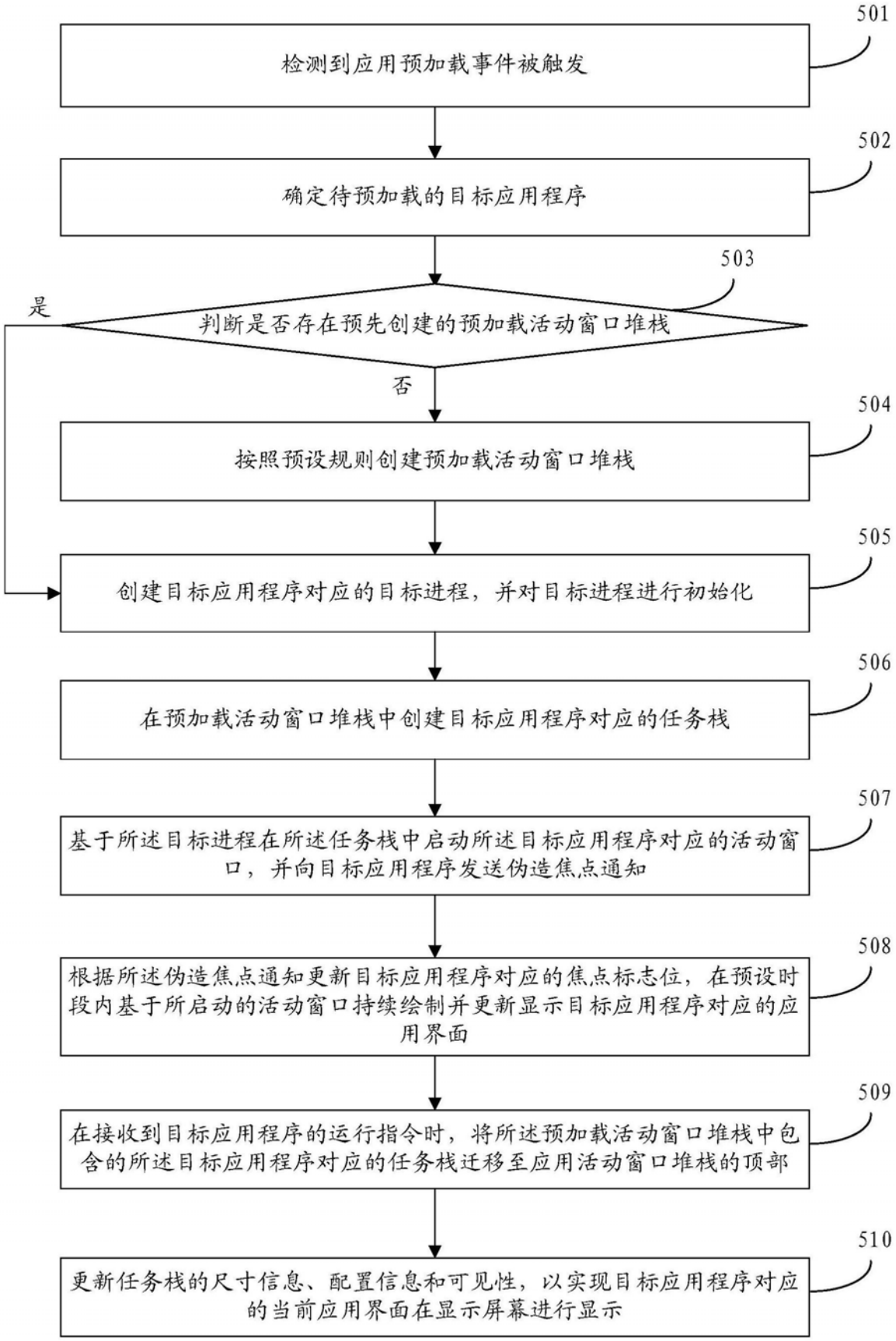

501
检测到应用预加载事件被触发
502
确定待预加载的目标应用程序
503
判断是否存在预先创建的预加载活动窗口堆栈
是
否
504
按照预设规则创建预加载活动窗口堆栈
505
创建目标应用程序对应的目标进程，并对目标进程进行初始化
506
在预加载活动窗口堆栈中创建目标应用程序对应的任务栈
507
基于所述目标进程在所述任务栈中启动所述目标应用程序对应的活动窗口，并向目标应用程序发送伪造焦点通知
508
根据所述伪造焦点通知更新目标应用程序对应的焦点标志位，在预设时段内基于所启动的活动窗口持续绘制并更新显示目标应用程序对应的应用界面
509
在接收到目标应用程序的运行指令时，将所述预加载活动窗口堆栈中包含的所述目标应用程序对应的任务栈迁移至应用活动窗口堆栈的顶部
510
更新任务栈的尺寸信息、配置信息和可见性，以实现目标应用程序对应的当前应用界面在显示屏幕进行显示

图5

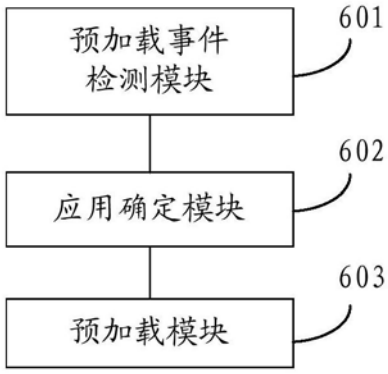


图6

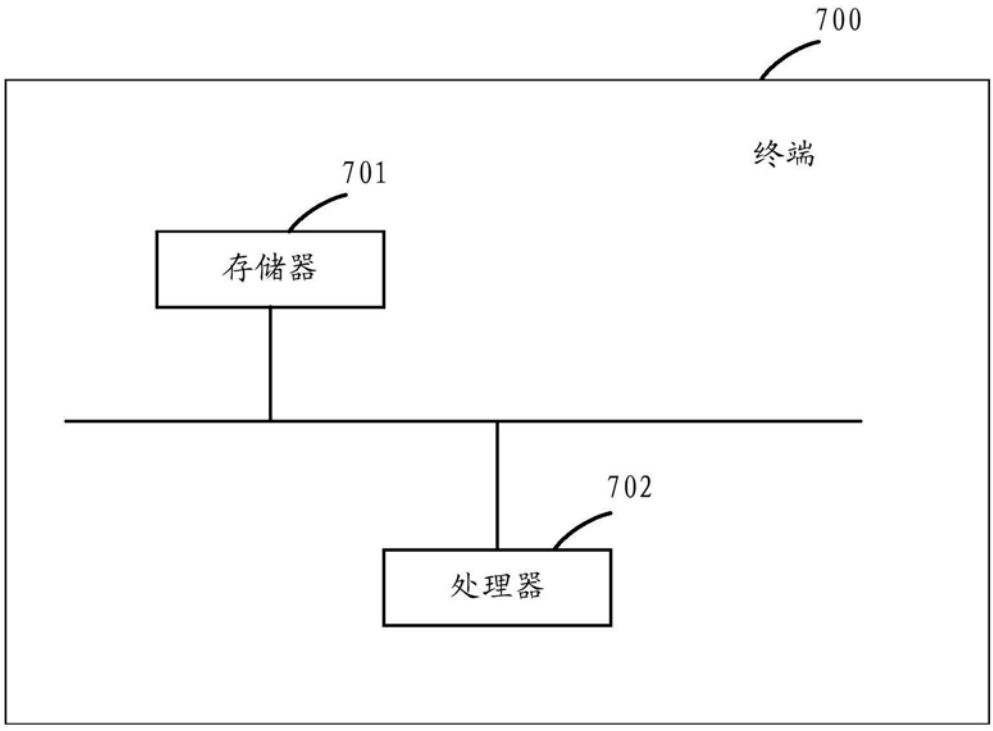


图7

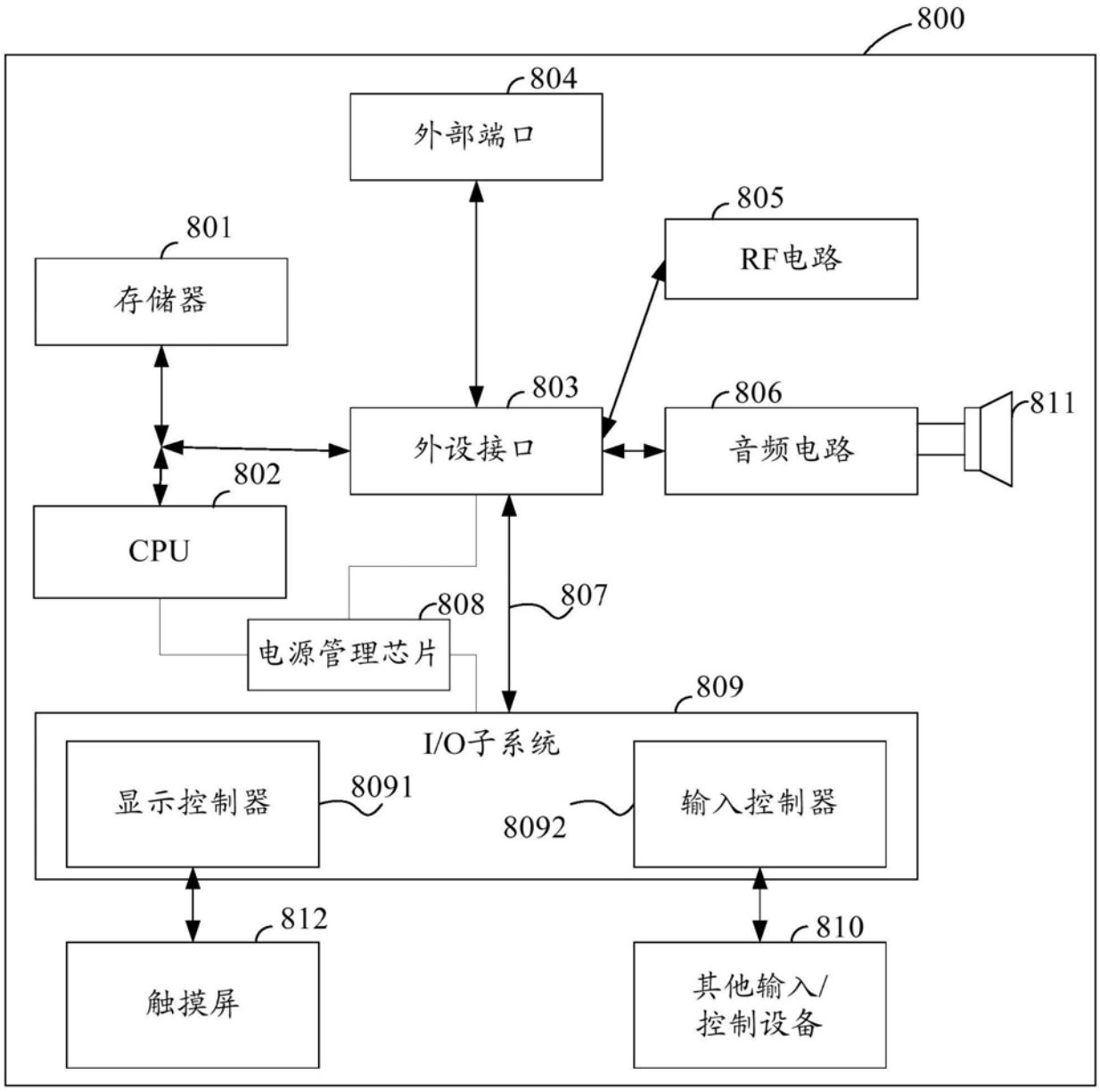

800
804
外部端口
805
RF电路
801
存储器
803
806
811
外设接口
音频电路
802
CPU
808
807
电源管理芯片
809
I/O子系统
显示控制器
8091
8092
输入控制器
812
810
触摸屏
其他输入/
控制设备

图8