

应用程序控制方法、装置、电子设备和存储介质

本申请要求于2021年1月4日提交中国专利局、申请号202110000662.9、申请名称为“多媒体内容的编辑控制方法、装置、电子设备和存储介质”的中国专利申请优先权。

技术领域

本申请涉及计算机技术领域，尤其涉及应用程序控制方法、装置、电子设备和存储介质。

发明背景

随着通信技术的发展和智能终端的普及，子应用作为轻型应用，已成为移动互联网应用的重要形式，如用户可以通过基于原生应用的子应用，来办理各种业务或进行社交活动等。其中，原生应用可以是社交类应用，如即时通讯软件；工具类应用，如美图软件，办公软件；影音娱乐类应用，如短视频软件、游戏软件；内容发布类应用，如第三方消费点评软件；内容资讯类应用，如浏览器，新闻软件；系统类应用，例如车载应用等。由于子应用具有不需要下载安装即可使用的特点，不会占用电子设备的存储空间，因此得到用户的广泛认可。

在相关技术中，子应用要实现音视频能力，只能依赖第三方音视频编辑库才能实现。但是，在网页端引入第三方音视频编辑库，会导致前端应用包体积很大，而子应用一般都会限制包体积。

发明内容

本申请实施例提供应用程序控制方法、装置、电子设备和存储介质，用以为子应用提供多媒体内容编辑能力的同时，不会增加应用包体积。

本申请实施例提供的第一种应用程序控制方法，由电子设备执行，所述方法包括：

在电子设备中运行原生应用和嵌套于所述原生应用中的子应用；

在所述子应用检测到针对目标多媒体内容的操作请求时，向所述原生应用发送对象创建指令，所述对象创建指令包含待创建的原生对象对应的参数信息，以使所述原生应用根据所述参数信息创建所述原生对象，所述原生对象用于在所述子应用中与所述目标多媒体内容进行操作；

当确定所述原生应用已创建所述原生对象时，向所述原生应用发送对象调用指令，以使所述原生应用调用所述原生对象，并执行所述原生对象的功能代码；

接收所述原生应用返回的执行结果，并根据所述执行结果，对所述目标多媒体内容进行相应的操作。

本申请实施例提供的第二种应用程序控制方法，由电子设备执行，所述方法包括：

在电子设备中运行原生应用和嵌套于所述原生应用中的子应用；

在接收到所述子应用发送的对象创建指令后，根据所述对象创建指令中待创建的原生对象对应的参数信息，创建所述原生对象，其中，所述对象创建指令是所述子应用检测到针对目标多媒体内容的操作请求时发送的，所述原生对象用于在所述子应用中对所述目标多媒体内容进行操作；

接收到所述子应用发送的对象调用指令后，调用所述原生对象，并执行所述原生对象的功能代码，其中，所述对象调用指令是所述子应用确定所述原生应用已创建所述原生对象时发送的；

将执行结果返回给所述子应用，以使所述子应用根据所述执行结果，对所述目标多媒体内容进行相应的操作。

本申请实施例提供的第一种应用程序控制装置，包括：

运行单元，用于在电子设备中运行原生应用和嵌套于所述原生应用中的子应用；

第一传输单元，用于在所述子应用检测到针对目标多媒体内容的操作请求时，向所述原生应用发送对象创建指令，所述对象创建指令包含待创建的原生对象对应的参数信息，以使所述原生应用根据所述参数信息创建所述原生对象，所述原生对象用于在所述子应用中对所述目标多媒体内容进行操作；

第二传输单元，用于当确定所述原生应用已创建所述原生对象时，向所述原生应用发送对象调用指令，以使所述原生应用调用所述原生对象，并执行所述原生对象的功能代码；

控制单元，用于接收所述原生应用返回的执行结果，并根据所述执行结果，对所述目标多媒体内容进行相应的操作。

本申请实施例提供的第二种应用程序控制装置，包括：

运行单元，用于在电子设备中运行原生应用和嵌套于所述原生应用中的子应用；

对象创建单元，用于在接收到所述子应用发送的对象创建指令后，根据所述对象创建指令中待创建的原生对象对应的参数信息，创建所述原生对象，其中，所述对象创建指令是所述子应用检测到针对目标多媒体内容的操作请求时发送的，所述原生对象用于在所述子应用中对所述目标多媒体内容进行操作；

对象调用单元，用于接收到所述子应用发送的对象调用指令后，调用所述原生对象，并执行所述原生对象的功能代码，其中，所述对象调用指令是所述子应用确定所述原生应用已创建所述原生对象时发送的；

第一反馈单元，用于将执行结果返回给所述子应用，以使所述子应用根据所述执行结果，对所述目标多媒体内容进行相应的操作。

本申请实施例提供了一种电子设备，包括处理器和存储器，其中，所述存储器存储有程序代码，当所述程序代码被所述处理器执行时，使得所述处理器执行上述任意一种应用程序控制方法的步骤。

本申请实施例提供一种计算机程序产品或计算机程序，该计算机程序产品或计算机程序包括计算机指令，该计算机指令存储在计算机可读存储介质中。计算机设

备的处理器从计算机可读存储介质读取该计算机指令，处理器执行该计算机指令，使得该计算机设备执行上述任意一种应用程序控制方法的步骤。

本申请实施例提供一种计算机可读存储介质，其包括程序代码，当所述程序产品在电子设备上运行时，所述程序代码用于使所述电子设备执行上述任意一种应用程序控制方法的步骤。

本申请实施例提供了一种应用程序控制方法、装置、电子设备和存储介质。具体的，在通过子应用对目标多媒体内容进行操作时，向原生应用发送对象创建指令，并在确定原生应用已经创建好原生对象之后，再向原生对象发送对象调用指令，该原生对象是用于对多媒体内容进行操作的对象，因而通过调用该原生对象，并执行相应的功能代码，即可实现对目标多媒体内容的操作。由于本申请实施例结合了原生系统的能力，通过子应用与原生应用的交互，使得子应用无需依赖第三方操作功能库，比如第三方音视频编辑库，就可以拥有基础的多媒体内容编辑能力，并且不会增加子应用的包体积。

本申请的其它特征和优点将在随后的说明书中阐述，并且，部分地从说明书中变得显而易见，或者通过实施本申请而了解。本申请的目的和其他优点可通过在所写的说明书、权利要求书、以及附图中所特别指出的结构来实现和获得。

附图简要说明

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，并不构成对本申请的不当限定。在附图中：

图 1 为相关技术中的一种在小程序中引入第三方音视频编辑库的示意图；

图 2 为本申请实施例中的一种应用场景的一个可选的示意图；

图 3 为本申请实施例中的第一种应用程序控制方法的流程示意图；

图 4 为本申请实施例中的一种小程序与原生应用之间的交互流程的示意图；

图 5 为本申请实施例中的一种原生应用里的插件根据小程序发送的指令创建原生对象的示意图；

图 6 为本申请实施例中的一种未引入同层渲染时的界面层级示意图；

图 7 为本申请实施例中的一种引入同层渲染时的界面层级示意图；

图 8 为本申请实施例中的一种 WKWebView 节点树的示意图；

图 9 为本申请实施例中的原生应用存储原生对象的示意图；

图 10 为本申请实施例中的第二种应用程序控制方法的流程示意图；

图 11 为本申请实施例中的一个可选的交互实现时序流程示意图；

图 12 为本申请实施例中的第一种应用程序控制装置的组成结构示意图；

图 13 为本申请实施例中的第二种应用程序控制装置的组成结构示意图；

图 14 为应用本申请实施例的一种电子设备的一个硬件组成结构示意图。

实施方式

为使本申请实施例的目的、技术方案和优点更加清楚，下面将结合本申请实施例中的附图，对本申请的技术方案进行清楚、完整地描述，显然，所描述的实施例是本申请技术方案的一部分实施例，而不是全部的实施例。基于本申请文件中记载的实施例，本领域普通技术人员在没有作出创造性劳动前提下所获得的所有其他实施例，都属于本申请技术方案保护的范围。

下面对本申请实施例中涉及的部分概念进行介绍。

原生应用：原生应用也称为“主应用”，它和网络应用相反，原生应用被安装在终端设备上，它能够使用终端设备的硬件功能（如扬声器、加速度传感器、摄像头等）。可以从应用程序市场中下载原生应用。本申请实施例中的子应用和原生应用具有嵌套关系，子应用嵌套于原生应用中。

子应用：是指一种基于原生应用的框架开发完成的应用程序，具有自己的生命周期和基本接口。用户无需从应用程序市场中下载和安装子应用，而是在安装并运行原生应用后，通过原生应用中包括的 URL 地址获取子应用。例如，在原生应用中，用户扫描子应用的二维码，从中获取子应用的 URL 地址，从而下载得到子应用；或者，在原生应用中直接搜索子应用，获取 URL 地址并下载。例如，子应用为基于原生应用运行的小程序，可以承载原生应用下的特定功能。例如，在原生应用“微视”中，使用小程序给视频加红包；比如，在原生应用“美图秀秀”中，使用小程序做基础图片编辑；又如，在原生应用“大众点评”中，使用小程序上传视频发布评价。

DOM (Document Object Model, 文档对象模型)：是处理可扩展标志语言的标准编程接口。在网页上，组织页面（或文档）的对象被组织在一个树形结构中，用来表示文档中对象的标准模型，就称为 DOM。

对象和类：对象是具有类这种引用类型的变量。类和对象是面向对象编程技术中的最基本的概念。其中，具有同种属性的对象称为类。

原生对象 (Native Object)：也可以叫做本地对象或者内部对象，是在原生应用的运行过程中动态创建的数据结构，该数据结构包括类名、方法名和参数等元素。

字典：是一种以键-值对形式存储数据的数据结构，比如：名字-电话号码，通过名字就能找到对应的电话号码，名字就是键(key)，电话号就是值(value)。字典中的键，是值在字典中的索引。对于 JavaScript 来说，字典类 (Dictionary) 的基础是 Array (数组) 类，JS 中的 Array 既是一个数组，同时也是一个字典。在本申请实施例中，原生应用可基于全局字典存储创建的所有原生对象，以保持子应用对原生对象的调用。

插件 (Plug-in)：是一种遵循一定规范的应用程序接口 (API) 编写出来的程序。在本申请实施例中的插件，主要是在原生应用侧提供的音视频插件，这些插件有大量 API，可以让子应用灵活组合、设计出功能丰富的音视频编辑能力。

同层渲染：指通过一定的技术手段，把原生组件直接渲染到 WebView（网页视图）层级上，此时原生组件层已经不存在，原生组件已被直接挂载到 WebView 节点上。在本申请实施例中，目标多媒体内容的播放界面是展示在子应用的子应用页面中的，在渲染过程中，该播放界面与子应用页面是通过同层渲染得到的。

下面对本申请实施例的设计思想进行简要介绍：

随着通信技术的不断发展和计算机技术的普及，各种应用程序（Application，APP）走进人们的日常生活，而子应用作为一种不需要从应用程序市场中下载安装即可使用的程序，只需要扫描二维码，或搜一搜，就能立即使用。其具有无需安装、无需卸载、用完即走，不占手机内存的特点，更是得到各行各业的广泛关注。目前各种行业的 APP 纷纷开发子应用，以便于自己的推广，抢占用户流量，而在子应用中实现音视频播放，正逐渐成为人们迫切实现的一种需求。

在相关技术中，子应用要实现音视频能力，只能依赖第三方音视频编辑库才能实现。例如如图 1 所示，在小程序中带上 FFmpeg（Fast Forward Mpeg，一种开源计算机程序）或 WebRTC（Web Real-Time Communication，网页实时通信）库，以实现小程序的音视频编辑能力。其他子应用也都是类似的处理方式。但是，在网页端引入第三方音视频编辑库，会导致前端应用包体积很大，而子应用一般都会限制包体积。

有鉴于此，本申请实施例提出了一种应用程序控制方法、装置、电子设备和存储介质。具体的，在通过子应用对目标多媒体内容进行操作时，向原生应用发送对象创建指令，并在确定原生应用已经创建好原生对象之后，再向原生对象发送对象调用指令，该原生对象是用于对多媒体内容进行操作的对象，因而通过调用该原生对象，根据执行结果即可实现对目标多媒体内容的操作。由于本申请实施例结合了原生系统的能力，通过子应用与原生应用的交互，让子应用拥有非常完善的音视频编辑能力，包括基础音视频编辑能力。并且，本申请实施例中，不需要引入第三方音视频编辑库，没有包体积增量，使得子应用拥有和原生应用一样好的性能，相比现有技术，具有更小的包体积，让音视频应用更容易实现和落地。

以下结合说明书附图对本申请的优选实施例进行说明，应当理解，此处所描述的优选实施例仅用于说明和解释本申请，并不用于限定本申请，并且在不冲突的情况下，本申请中的实施例及实施例中的特征可以相互组合。

如图 2 所示，其为本申请实施例的应用场景示意图。该应用场景图中包括两个终端设备 210 和一个服务器 220。终端设备 210 与服务器 220 之间可以通过通信网络进行通信。

在一种可选的实施方式中，通信网络是有线网络或无线网络。终端设备 210 以及服务器 220 可以通过有线或无线通信方式进行直接或间接地连接，本申请在此不做限制。

在本申请实施例中，终端设备 210 为用户使用的电子设备，该电子设备可以是个人计算机、手机、平板电脑、笔记本、电子书阅读器、智能家居等具有一定计算能力，并且运行有即时通讯类软件及网站、或者社交类软件及网站的计算机设备。服务器 220 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或者分布式系统，还可以是提供云服务、云数据库、云计算、云函数、云存储、网络服务、云通信、中间件服务、域名服务、安全服务、CDN（Content Delivery Network，内容分发网络）、以及大数据和人工智能平台等基础云计算服务的云服务器。

其中，每个终端设备 210 中都可以安装有具有多媒体内容编辑能力的原生应用，例如，具有音视频编辑能力的原生应用，具有图像动画编辑能力的原生应用等，子应用则嵌套于原生应用中。

本申请实施例主要是以音视频编辑为例进行举例说明的，子应用中的基础音视频编辑能力可以用在各种有音视频编辑需求的产品上。子应用可以基于该能力实现如：音视频裁剪、变速、特效、模板、红包效果等功能。常见的可以在子应用中实现音视频编辑能力的产品包括：影音娱乐类应用，比如可以通过短视频软件小程序给发布的视频加红包，让用户浏览短视频的同时领取红包，还可以在用户录制短视频时提供特效；再比如游戏软件，可以让用户通过小程序边玩游戏边直播等。工具类，比如美图软件，可以让用户通过小程序做基础图片编辑；再比如办公软件，可以通过小程序为用户提供模板。对于内容发布类，比如第三方消费点评软件，可以让用户在小程序上传视频发布评价。内容资讯类应用，比如浏览器，可以让用户通过网页小程序进行音视频裁剪、变速等。社交类，比如即时通讯软件，可以通过小程序浏览音视频，等等。

另外，本申请实施例中列举的小程序，不仅可以嵌套在上述所列举的终端 APP 上，也可流转到互联网汽车上，为用户提供从出行到生活一站式的服务，实现上车前、行车中、下车后串联的服务场景。对应的，原生应用可以为系统类应用，例如车载应用，可以让用户通过车载小程序实现音视频裁剪、变速、特效、模板、红包效果等功能，基于上述音视频编辑能力的实现，可以围绕车场景为用户提供从出行到生活的各类智慧服务。

与相关技术相比，本申请实施例中的应用程序控制方法，结合了网页视图（WebView）的 JS 交互技术、WebView 同层渲染、子应用插件模式、内存管理机制。

下面主要以子应用为小程序为例，结合附图对本申请实施例中的编辑控制方法进行详细介绍：

参阅图 3 所示，为本申请实施例提供的一种应用程序控制方法的实施流程图，由电子设备执行，例如图 2 中的终端设备 210，该方法应用于子应用，具体实施流程如下：

S30：在电子设备中运行原生应用和嵌套于原生应用中的子应用。

在本申请实施例中，原生应用作为主应用，子应用嵌套于原生应用中。

S31：在子应用检测到针对目标多媒体内容的操作请求时，向原生应用发送对象创建指令，对象创建指令包含待创建的原生对象对应的参数信息，以使原生应用根据参数信息创建原生对象，原生对象用于在子应用中对目标多媒体内容进行操作。

在本申请实施例中，对多媒体内容进行的操作，包括各种编辑控制，可以分为多种控制类型，比如：裁剪，变速，编解码，贴纸，添加特效，加载模板等。这里的原生对象是指对多媒体内容进行具体的某一类控制的原生对象，例如，在对目标多媒体内容进行裁剪时，需要创建一类用于对多媒体内容进行裁剪控制的原生对象 A；在对目标多媒体内容进行变速时，需要创建一类用于对多媒体内容进行变速控制的原生对象 B，等等。

在通过小程序对目标多媒体内容进行编辑控制时，需要向原生应用发送对象创建指令，该指令中包含的参数信息具体是指对象类型、对象类等，以指示该原生对象用于对多媒体内容进行哪一类型的编辑控制，不同的控制对应不同的音视频编辑能力。比如对音视频进行裁剪，对应的类名为“TAVClip”，对音视频进行编解码，对应的类名为“TAVPlayer”等。

其中，原生应用在接收到子应用发送的对象创建指令后，会根据对象创建指令中待创建的原生对象对应的参数信息，来创建原生对象，为原生对象分配内存。

另外，考虑到使用网页端提供的基础 API 实现小程序音视频编辑能力时，由于网页端在音视频处理上的 API 不够完善，会造成部分能力很难实现或者性能较差。本申请实施例结合了原生应用的能力，直接暴露原生 API，让小程序拥有基础的音视频编辑能力，并不会增加包体积，还能够具有更好的渲染性能。

可选的，在原生应用侧，原生应用根据对象创建指令中待创建的原生对象对应的参数信息，创建原生对象时，需要解析该指令中的参数信息，通过解析参数信息，从预先配置的 API 集合中，选取创建原生对象所需调用的目标 API；调用目标 API，创建原生对象。其中，该 API 集合中的 API 是与子应用的音视频编辑能力相关联的 API。在本申请实施例中的 API 集合主要是指 API 列表，该列表可以包括上述所列举的 6 种 API。

例如，需要通过小程序对目标多媒体内容进行裁剪时，参数信息中的类名可以是“TAVClip”，对应的目标 API 即裁剪 API；若需要通过小程序对音视频进行编解码时，参数信息中的类名可以是“TAVPlayer”，对应的目标 API 即编解码 API，等等。

需要说明的是，本申请实施例中所列举的原生应用提供的音视频编辑能力相关的 API 只是举例说明，实际上，任何一种音视频编辑能力相关的 API 都适用于本申请实施例，在此不做具体限定。

在原生对象内存管理机制上，通过小程序和原生应用交互创建的原生对象，

存在如下问题：原生对象在被创建后，如果没有被持有，内存会被系统释放。小程序下一次发消息就会找不到这个对象。为解决上述问题，本申请实施例中提出了基于全局字典的对象存储方法。

在原生层，使用了全局字典存储创建的对象，只要是小程序创建的对象都会存储到全局字典中，保持引用。

S32：当确定原生应用已创建原生对象时，向原生应用发送对象调用指令，以使原生应用调用原生对象，并执行原生对象的功能代码；

在本申请实施例中，子应用将包含对象调用指令的JS消息发送到原生应用，原生应用根据JS消息解析出内存地址和类名，然后调用原生对象，并且执行原生对象的功能代码，进行相应的处理，进而将执行结果返回给子应用。

具体的，原生应用中的插件在收到JS消息后，会自动分发到用于调用原生对象的目标API上，实现大量API在分发时的简单扩展，提高音视频的编辑控制效率。需要说明的是，原生应用中的插件具体是指音视频编辑能力相关的插件，例如编解码、裁剪、变速、贴纸、特效、模板等，都是与音视频编辑相关的。

S33：接收原生应用返回的执行结果，并根据执行结果，在子应用中对目标多媒体内容进行相应的操作。

比如目标多媒体内容为视频文件A，若需要将视频文件A的播放速度调整成2倍速，子应用根据原生应用返回的执行结果，对视频文件A进行倍速播放。

本申请实施例中，子应用和原生应用之间通过底层操作系统中的浏览器组件进行交互。这种交互方式，有效利用了操作系统的底层功能，使得子应用和原生应用之间数据传输的效率更高，并且不会增加子应用的包体积。

以子应用为小程序为例，参阅图4所示，其为本申请实施例中的一种小程序与原生应用之间的交互流程的示意图，包括：小程序410、操作系统420和原生应用430。其中，在小程序410中，接收用户对目标多媒体内容411的操作请求，并给出响应。在操作系统420中，上述浏览器组件包括网页视图（WebView）421、浏览器引擎（Webkit）422。其中，网页视图421是一种基于浏览器引擎422来展现网页页面的控件。

比如用户点击打开某一网页，通过小程序410对该网页上的一个目标多媒体内容411进行操作时，小程序410的应用代码被加载到网页视图421上运行，此时小程序410根据接收到的操作请求，通过浏览器引擎422，向原生应用430发送对象创建指令。具体地，对象创建指令是以JS消息的形式进行发送。

原生应用430根据接收到的JS消息，解析对象创建指令中的参数信息，根据解析到的参数信息，判断调用哪个目标API，进而创建原生对象。也就是说，原生对象是和参数信息一一对应的。例如，根据参数信息中包括的类名，创建的原生对象可以是名称为编解码431、裁剪432、变速433、贴纸434、特效435和模板436中的任一个。

原生应用 430 创建完原生对象后,经由浏览器引擎 422,向小程序 410 返回原生对象的对象信息,该对象信息包括内存地址和类名。这样,小程序 410 接收到对象信息后,确定原生应用 430 已创建原生对象,然后经由浏览器引擎 422,向原生应用 430 发送对象调用指令。

原生应用 430 接收到对象调用指令后,调用之前已创建的原生对象,例如名称为编解码 431 的原生对象,并且执行该原生对象对应的功能代码。所谓的功能代码,是指对目标多媒体内容执行相应的操作所需的代码,在原生对象的数据结构中,是通过方法名来指示的。

由原生应用 430 执行原生对象对应的功能代码后,获得处理后的数据作为执行结果。例如,操作请求是对视频进行剪裁,那么,执行结果是裁剪后片段的时长。

最后,原生应用 430 经由浏览器引擎 422,将执行结果发送小程序 410。小程序 410 根据接收到的执行结果,直接对目标多媒体内容进行操作,例如,展示剪裁后的片段。以此来达到小程序 410 和原生应用 430 之间的交互。

在本申请实施例,基于图 4 所示的交互流程,为小程序提供三个基础接口:创建对象接口、释放对象接口、调用对象接口,具体配置如下:

接口一:创建对象接口。

```
func tav_createObject(className:String,methodName:String,param:Param)->
(object:TAVObject);
```

其中,func 表示函数 function 的缩写,该接口函数名称为 tav_createObject,className 表示类名,className:String 表示类名为字符串类型;methodName 表示方法名,methodName:String 表示方法名也为字符串类型;param 表示参数,Param 具体可以包括 name (参数名)和 value (值)。

例如,创建一个对象(object),对象名称为 TAVObject (音视频对象)。

接口二:释放对象接口。

```
func tav_releaseObject(object:TAVObject);
```

其中,该接口函数名称为 tav_releaseObject。

接口三:调用对象接口。

```
func tav_sendMsgToObject(object:TAVObject,methodName:String,param:
Param)。
```

其中,该接口函数名称为 tav_sendMsgToObject,object 表示要调用的对象,TAVObject 即要调用的对象的名称,methodName:String 表示方法名也为字符串类型;param 表示参数。

在本申请实施例,通过这三个基础接口,再配合不同的参数,即可让小程序调用原生层的各种音视频接口,可以满足小程序和原生层的基础通信。

如图 5 所示,其为本申请实施例中的一种原生应用里的插件根据小程序发送的指令创建原生对象的示意图。其中,图的上半部分是指小程序 510 发出对

象创建指令的形式。具体的，小程序 510 通过执行特定的函数，发出对象创建指令，如下函数 `xx.inokeNativePlugin`，用于指示调用原生应用中的插件（简称为原生插件），从而创建原生对象。具体代码如下：

```
xx.inokeNativePlugin ( {    /调用原生插件
  api_name: "tav_creatObject" , /接口名称: tav_creatObject (音视频_创建对象)
  data: {    /数据
    "className" : "TAVClip" ,    /类名: TAVClip (音视频裁剪)
    "methodName" : "resource" , "param" : { "resource" : resource }    /方法名: resource (资源) , 参数 {资源: 资源}
  },
  Success: {},    /成功
  fail: {}    /失败
})
```

上述代码表示，通过小程序 510 的基础接口 `tav_creatObject` 发出对象创建指令，该指令指示原生对象的类名为 `TAVClip`，用于对音视频进行裁剪；方法名称为 `resource`。

经由浏览器引擎 520，小程序 510 将对象创建指令发送给原生应用 530。然后，原生应用 530 根据 `className`（类名）、`methodName`（方法名）、`param`（参数）分发到对应的目标 API 531，创建具有相应类名的原生对象。例如，目标 API 531 为 `TAVClip`（音视频裁剪），创建的原生对象名称为裁剪 5321，类名为 `TAVClip`；例如，目标 API 531 为 `TAVPlayer`（音视频编解码），创建的原生对象名称为编解码 5322，类名为 `TAVPlayer`；例如，目标 API 531 为 `TAVExportSession`（音视频导出），创建的原生对象名称为导出 5323，类名为 `TAVExportSession`。

在本申请实施例中，由于音视频编辑 API 是由原生应用提供，所以生成的纹理画面都在原生应用。而小程序和原生应用是通过 JS 消息进行通信，所以无法做到纹理数据的传递。

为解决原生应用音视频编辑渲染上屏的问题，本申请实施例中的小程序提供了同层渲染的技术。也就是说，目标多媒体内容的播放界面展示在子应用的子应用页面中，该播放界面是与子应用页面是通过同层渲染得到的。

在本申请实施例中，同层渲染的核心原理就是将原生层的视图（view）放入到网页视图内的视图层级，可以把原生层的视图当成是一个 `html`（HyperText Markup Language，超文本标记语言）标签一样使用。这是因为网页视图在渲染的时候，可以将 DOM 元素按标签创建出原生的视图，同层渲染就是在这些视图中找到合适的位置，然后插入原生层的视图即可。

在相关技术中，小程序的内容大多是渲染在网页视图上的，如果把网页视图看成单独的一层，那么由操作系统自带的这些原生组件则位于另一个更高的

层级。两个层级是完全独立的。正如图 6 所示，表示未引入同层渲染时的界面层级示意图，非原生组件位于 WebView 层，而原生组件则位于另一个较高的层级。其中，非原生组件层渲染的是小程序页面，即子应用页面。原生组件层渲染的是目标多媒体内容的播放界面。

而本申请实施例中提出的同层渲染则是指通过一定的技术手段把原生组件直接渲染到 WebView 层上，此时原生组件层已经不存在，并且原生组件已被直接挂载到 WebView 节点上。小程序几乎可以像使用非原生组件一样去使用同层渲染的原生组件。启用同层渲染之后的界面层级如图 7 所示。在原生应用渲染的画面，直接渲染到原生视图上，然后嵌入到了小程序里。这样，子应用页面与目标多媒体内容的播放界面是通过同层渲染得到的。

参阅图 8 所示，其为本申请实施例中的一种 WKWebView 节点树的示意图。其中，WKWebView 用于进行网页的渲染与展示，可以进行自定义配置。小程序在使用 WKWebView 进行渲染时，WKWebView 在内部采用的是分层的方式进行渲染，它会将 Webkit 内核生成的合成层（CompositingLayer）渲染成一个 WK 合成视图（WKCompositingView），这是一个客户端原生的视图，而内核将多个 DOM 节点渲染到一个合成层上，因此，合成层与 DOM 节点之间不存在一对一的映射关系。当把一个 DOM 节点的层叠样式表（CSS，Cascading Style Sheets）属性设置为 overflow:scroll（溢出：滚动）之后，WKWebView 会为其生成一个 WK 子滚动视图（WKChildScrollView），与 DOM 节点存在映射关系，这是一个原生的用户界面滚动视图（UIScrollView）的子类，也就是说，WebView 里的滚动实际上是由真正的原生滚动组件来承载的，以使 WebView 滚动有更流畅的体验。

小程序的同层渲染也正是基于 WK 子滚动视图实现的，如图 8 所示，将原生组件挂载到该 WK 子滚动视图节点上作为其子视图，或者，在 WK 子滚动视图节点插入原生组件，实现同层渲染，这样，不仅解决了原生组件的层级问题，同时也让原生组件有了更丰富的展示和交互的能力，使得小程序在渲染播放界面时，可以直接使用同层的原生组件，降低了渲染的处理复杂度，提高了渲染的处理效率。

在一种可选的实施方式中，步骤 S32 的具体实现流程如下：

S321：在接收到原生应用返回的原生对象的对象信息时，确定原生应用已创建原生对象；

S322：根据对象信息，向原生应用发送对象调用指令。

这样，通过原生应用创建原生对象，进而由子应用进行调用，从而为子应用提供了非常完善的音视频基础能力，同时没有包体积增量，还能让子应用拥有和原生应用一样好的性能。

考虑到使用全局字典可以存储之前创建的对象，但是有一些情况下还是会遇到对象无法找到的问题。比如尽管创建的对象都会存储在全局字典中，但是，

如果对象上通过 get（获取）方法获取到的对象属性不在字典中，就会遇到对象无法找到的问题。如果用户在使用 get 方法调用的时候做一次字典存储，可以确保对象能被查找到，但是这样会加重 API 的实现复杂度。

在本申请实施例中，提出了一种解决这个问题的更好的方法，考虑到 get 方法得到的对象实际上一直会存在于系统内存中，因为它是存在引用关系的，因而只要有这个对象的内存地址和类名，就可以动态得到实例对象。

可选的，原生应用在创建原生对象之后，将为该原生对象所分配的内存的地址、原生对象的类名等信息作为对象信息，向小程序返回对象的数据结构中提供内存地址和类名，其中，内存地址是指原生应用为原生对象分配的内存的地址，类名是指原生对象的类型。小程序在接收到针对该对象信息后，可确定原生应用已创建原生对象。小程序在调用对象方法的时候，将内存地址和类名作为对象数据，添加到对象调用指令中，传回给原生应用的原生层，由原生层使用指针动态创建对象实例，实现对原生对象的调用，并且执行原生对象数据结构中指定的方法，也就是执行功能代码。

具体过程为：在小程序调用该原生对象时，将内容地址和类名添加到对象调用指令中，并将该对象调用指令发送给原生应用；在原生应用接收到包含有内容地址和类名的对象调用指令之后，根据内存地址确定为原生对象分配的内存，并根据类名确定原生对象的类型；通过指针索引到为原生对象分配的内存后，根据原生对象的类型，调用原生对象。

上述在对象调用指令中添加内容地址和类名的方式，正是考虑到原生对象一直存在系统内存中，并且具备一定的引用关系，那么，通过内容地址和类名来表征引用关系，就可以动态得到原生对象，提高了小程序与原生应用之间交互时获取对象的处理效率。

参阅图 9 所示，其为本申请实施例中的原生应用存储原生对象的示意图。小程序 910 在接收到针对目标多媒体内容的裁剪（Clip）请求时，向原生应用 920 发送对象创建指令，原生应用 920 在接收到该指令后，首先在临时内存中创建原生对象 921，名称为裁剪，类名为 TAVClip。之后将该原生对象 921 的内存移动到常驻内存（即通过全局字典进行对象存储），确保小程序 910 可以访问到对象。

之后，将原生应用 920 该原生对象的对象信息返回给小程序 910，具体包括 identifier（对象 ID）和 className（对象类名）。在小程序 910 要调用原生对象以执行 Clip 方法时，将 identifier 和 className 添加到对象调用指令中，发送给原生应用 920 的方法转发器 922。例如图 9 中，对象调用指令中包括 identifier 和 className，其中，identifier 为 0x3df3459，表示原生对象的内存地址为 0x3df3459，className 为 Clip。在原生应用 920 接收到这些信息后，基于方法转发器 922，根据 identifier 找到内存位置，根据 className 确定类型，之后即可通过指针索引到为原生对象分配的内存 921，根据原生对象的类型，调用原生对象的方法，也就

是执行原生对象的功能代码。

在上述实施方式中，只要有原生对象的内存地址和类名，就可以动态得到实例对象，以解决通过 get 方法获取到的对象属性不在字典中，导致对象无法找到的问题。

可选的，小程序可在完成对目标多媒体内容的操作之后，向原生应用发送针对原生对象的对象释放指令。原生应用在接收到针对原生对象的对象释放指令后，释放之前为该原生对象分配的内存。

具体的，小程序可以通过调用 release 消息，释放对象资源。比如，目标多媒体内容为视频文件 A，在完成对视频文件 A 的裁剪之后，即结束了对目标多媒体内容的编辑控制，此时可向原生应用发送针对原生对象 TAVClip 的 release 消息，即对象释放指令，该指令中可包含该原生对象的内存地址和类名、对象 id 等信息，在原生应用接收到 release 消息之后，即可通过对应的 API，释放之前为该原生对象分配的内存。另外，在退出小程序或者其他情况下，即可进行插件销毁，原生层也会自动清理掉相关的所有原生对象。

在上述实施方式中，可在小程序完成对目标多媒体内容的某一种编辑控制时，即可向原生应用发送针对用于该类编辑控制的原生对象的对象释放指令，由原生对象释放相应的内存，以避免内存溢出，以便空出足够的连续内存空间供后续使用。

基于相同的发明构思，本申请实施例还提供一种应用程序控制方法，应用于原生应用。参阅图 10 所示，为本申请实施例提供的第二种应用程序控制方法的实施流程图，该方法由电子设备执行，例如图 2 中的终端设备 210，应用于原生应用，具体实施流程如下：

S100：在电子设备中运行原生应用和嵌套于原生应用中的子应用。

S101：在接收到子应用发送的对象创建指令后，根据对象创建指令中待创建的原生对象对应的参数信息，创建原生对象，其中，对象创建指令是子应用检测到针对目标多媒体内容的操作请求时发送的，原生对象用于在子应用中对目标多媒体内容进行操作；

S102：接收到子应用发送的对象调用指令后，调用原生对象，并执行原生对象的功能代码，其中，对象调用指令是子应用确定原生应用已创建原生对象时发送的；

S103：将执行结果返回给子应用，以使子应用根据执行结果，对目标多媒体内容进行相应的操作。

可选的，在创建原生对象之后，在接收到子应用发送的对象调用指令之前，还包括：

将通过创建原生对象所获得的对象信息返回给子应用，以使子应用在接收到对象信息时，确定原生应用已创建原生对象，并根据对象信息发送对象调用指令。

可选的,根据对象创建指令中待创建的原生对象的参数信息,创建原生对象,具体包括:

通过解析对象创建指令中的参数信息,从预先配置的 API 集合中,选取创建原生对象所需调用的目标 API,其中,API 集合中的 API 是与子应用的音视频编辑能力相关联的 API;

调用目标 API,创建原生对象。

通过上述方式,通过原生应用调用与子应用的音视频编辑能力相关联的 API,来创建原生对象,使得子应用具备了一定的音视频编辑能力,避免了在子应用中添加第三方音视频编辑库,比如在小程序中带上 FFmpeg 或 WebRTC 库,从这个角度来看,降低了具备音视频编辑能力的子应用的包体积。

可选的,对象信息包括内存地址和类名,对象调用指令携带有内存地址和类名,内存地址是指原生应用为原生对象分配的内存的地址,类名是指原生对象的类型;

接收到子应用发送的对象调用指令后,调用原生对象,具体包括:

根据内存地址确定为原生对象分配的内存,并根据类名确定原生对象的类型;

通过指针索引到为原生对象分配的内存后,根据原生对象的类型调用原生对象。

可选的,在根据对象创建指令中待创建的原生对象对应的参数信息,创建原生对象之后,还包括:

使用全局字典存储原生对象,以保持子应用对原生对象的调用。

通过全局字典进行存储,使得原生对象可以常驻内存,从而可以确保子应用可以访问到原生对象,进而可以获得基础的音视频编辑能力。

可选的,方法还包括:

接收到子应用发送的针对原生对象的对象释放指令后,释放为原生对象分配的内存,其中,对象释放指令是子应用在完成对目标多媒体内容的操作之后发送的。

参阅图 11 所示,为一种应用程序控制方法的交互时序图。该方法的具体实施流程如下:

步骤 S1101:小程序在检测到通过小程序页面对目标多媒体内容的操作请求时,向原生应用发送对象创建指令;

步骤 S1102:原生应用根据对象创建指令,解析得到对象创建指令中的参数信息,并根据解析出的参数信息,从 API 集合中选取创建原生对象所需调用的目标 API;

步骤 S1103:原生应用调用目标 API 创建原生对象,并通过全局字典存储创建的原生对象;

步骤 S1104:原生应用将通过创建原生对象所获得的对象信息返回给子应用,

对象信息包括内存地址和类名；

步骤 S1105：小程序在调用原生对象时，将内存地址和类名添加到对象调用指令中，并将对象调用指令发送给原生应用；

步骤 S1106：原生应用根据内存地址确定为原生对象分配的内存资源，并根据类名确定原生对象的类型；

步骤 S1107：通过指针索引到为原生对象分配的内存后，根据原生对象的类型，调用原生对象，并且执行原生对象的功能代码，以对目标多媒体内容进行相应的数据处理；

步骤 S1108：原生应用将执行结果返回给小程序；

步骤 S1109：小程序根据执行结果，对目标多媒体内容进行相应的操作。

基于相同的发明构思，本申请实施例还提供一种应用程序控制装置。如图 12 所示，其为本申请实施例提出的一种应用程序控制装置 1200 的结构示意图，应用于子应用，可以包括：

运行单元 1201，用于在电子设备中运行原生应用和嵌套于所述原生应用中的子应用；

第一传输单元 1202，用于在子应用检测到针对目标多媒体内容的操作请求时，向原生应用发送对象创建指令，对象创建指令包含待创建的原生对象对应的参数信息，以使原生应用根据参数信息创建原生对象，原生对象用于在子应用中对目标多媒体内容进行操作；

第二传输单元 1203，用于当确定原生应用已创建原生对象时，向原生应用发送对象调用指令，以使原生应用调用原生对象，并执行原生对象的功能代码；

控制单元 1204，用于接收原生应用返回的执行结果，并根据执行结果，对目标多媒体内容进行相应的操作。

可选的，第二传输单元 1203 具体用于：

在接收到原生应用返回的原生对象的对象信息时，确定原生应用已创建原生对象；

根据对象信息，向原生应用发送对象调用指令。

可选的，对象信息包括内存地址和类名，第二传输单元 1203 具体用于：

在调用原生对象时，将内存地址和类名添加到对象调用指令中，并将对象调用指令发送给原生应用，以使原生应用根据内存地址确定为原生对象分配的内存，并根据类名确定原生对象的类型，通过指针索引到为原生对象分配的内存后，根据原生对象的类型调用原生对象。

可选的，目标多媒体内容的播放界面展示在子应用的子应用页面中，目标多媒体内容对应的播放界面与子应用页面是通过同层渲染得到的。

可选的，装置还包括：

第三传输单元 1205，用于在子应用完成对目标多媒体内容的操作之后，向原生应用发送针对原生对象的对象释放指令，以使原生应用释放为原生对象分

配的内存。

可选的，子应用与原生应用之间通过浏览器组件进行交互。

基于相同的发明构思，本申请实施例还提供一种应用程序控制装置。如图13所示，其为本申请实施例提出的一种应用程序控制装置1300的结构示意图，应用于原生应用，可以包括：

运行单元1301，用于在电子设备中运行原生应用和嵌套于所述原生应用中的子应用；

对象创建单元1302，用于在接收到子应用发送的对象创建指令后，根据对象创建指令中待创建的原生对象对应的参数信息，创建原生对象，其中，对象创建指令是在子应用检测到针对目标多媒体内容的操作请求时发送的，原生对象用于在子应用中对目标多媒体内容进行操作；

对象调用单元1303，用于接收到子应用发送的对象调用指令后，调用原生对象，并执行原生对象的功能代码，其中，对象调用指令是子应用确定原生应用已创建原生对象时发送的；

第一反馈单元1304，用于将执行结果返回给子应用，以使子应用根据执行结果，对目标多媒体内容进行相应的操作。

可选的，装置还包括：

第二反馈单元1305，用于在对象创建单元1302创建原生对象之后，在对象调用单元1303接收到子应用发送的对象调用指令之前，将通过创建原生对象所获得的对象信息返回给子应用，以使子应用在接收到对象信息时，确定原生应用已创建原生对象，并根据对象信息发送对象调用指令。

可选的，对象创建单元1302具体用于：

通过解析对象创建指令中的参数信息，从预先配置的API集合中，选取创建原生对象所需调用的目标API，其中，API集合中的API是与子应用的音视频编辑能力相关联的API；

调用目标API，创建原生对象。

可选的，对象信息包括内存地址和类名，对象调用指令携带有内存地址和类名，对象调用单元1303具体用于：

根据内存地址确定为原生对象分配的内存，并根据类名确定原生对象的类型；

通过指针索引到为原生对象分配的内存后，根据原生对象的类型，调用原生对象。

可选的，装置还包括：

存储单元1306，用于在对象创建单元1302根据对象创建指令中待创建的原生对象对应的参数信息，创建原生对象之后，使用全局字典存储原生对象，以保持子应用对原生对象的调用。

可选的，装置还包括：

内存释放单元 1307，用于在接收到子应用发送的针对原生对象的对象释放指令后，释放为原生对象分配的内存，其中，对象释放指令是子应用在完成对目标多媒体内容的操作之后发送的。

为了描述的方便，以上各部分按照功能划分为各模块（或单元）分别描述。当然，在实施本申请时可以把各模块（或单元）的功能在同一个或多个软件或硬件中实现。

在介绍了本申请示例性实施方式的应用程序控制方法和装置之后，接下来，介绍根据本申请的另一示例性实施方式的电子设备。

所属技术领域的技术人员能够理解，本申请的各个方面可以实现为系统、方法或程序产品。因此，本申请的各个方面可以具体实现为以下形式，即：完全的硬件实施方式、完全的软件实施方式（包括固件、微代码等），或硬件和软件方面结合的实施方式，这里可以统称为“电路”、“模块”或“系统”。

与上述方法实施例基于同一发明构思，本申请实施例中还提供了一种电子设备。该电子设备可以用于资源对象的转移控制。在一种实施例中，该电子设备可以是服务器，如图 2 所示的服务器 220，也可以是终端设备，如图 2 所示的终端设备 210。在该实施例中，电子设备的结构可以如图 14 所示，包括存储器 1401，通讯模块 1403 以及一个或多个处理器 1402。

存储器 1401，用于存储处理器 1402 执行的计算机程序。存储器 1401 可主要包括存储程序区和存储数据区，其中，存储程序区可存储操作系统，以及运行即时通讯功能所需的程序等；存储数据区可存储各种即时通讯信息和操作指令集等。

存储器 1401 可以是易失性存储器（volatile memory），例如随机存取存储器（random-access memory，RAM）；存储器 1401 也可以是非易失性存储器（non-volatile memory），例如只读存储器，快闪存储器（flash memory），硬盘（hard disk drive，HDD）或固态硬盘（solid-state drive，SSD）；或者存储器 1401 是能够用于携带或存储具有指令或数据结构形式的期望的程序代码并能够由计算机存取的任何其他介质，但不限于此。存储器 1401 可以是上述存储器的组合。

处理器 1402，可以包括一个或多个中央处理单元（central processing unit，CPU）或者为数字处理单元等等。处理器 1402，用于调用存储器 1401 中存储的计算机程序时实现上述应用程序控制方法。

通讯模块 1403 用于与终端设备和其他服务器进行通信。

本申请实施例中不限定上述存储器 1401、通讯模块 1403 和处理器 1402 之间的具体连接介质。本申请实施例在图 14 中以存储器 1401 和处理器 1402 之间通过总线 1404 连接，总线 1404 在图 14 中以粗线表示，其它部件之间的连接方式，仅是进行示意性说明，并不引以为限。总线 1404 可以分为地址总线、数据总线、控制总线等。为便于表示，图 14 中仅用一条粗线表示，但并不表示仅有一根总线或一种类型的总线。

存储器 1401 中存储有计算机存储介质，计算机存储介质中存储有计算机可执行指令，计算机可执行指令用于实现本申请实施例的应用程序控制方法。处理器 1402 用于执行上述的应用程序控制方法，如图 3 或图 10 所示。

在一些可能的实施方式中，本申请提供的应用程序控制方法的各个方面还可以实现为一种程序产品的形式，其包括程序代码，当程序产品在计算机设备上运行时，程序代码用于使计算机设备执行本说明书上述描述的根据本申请各种示例性实施方式的应用程序控制方法中的步骤，例如，计算机设备可以执行如图 3 或图 10 中所示的步骤。

程序产品可以采用一个或多个可读介质的任意组合。可读介质可以是可读信号介质或者可读存储介质。可读存储介质例如可以是但不限于电、磁、光、电磁、红外线、或半导体的系统、装置或器件，或者任意以上的组合。可读存储介质的更具体的例子（非穷举的列表）包括：具有一个或多个导线的电连接、便携式盘、硬盘、随机存取存储器（RAM）、只读存储器（ROM）、可擦式可编程只读存储器（EPROM 或闪存）、光纤、便携式紧凑盘只读存储器（CD-ROM）、光存储器件、磁存储器件、或者上述的任意合适的组合。

本申请的实施方式的程序产品可以采用便携式紧凑盘只读存储器（CD-ROM）并包括程序代码，并可以在计算装置上运行。然而，本申请的程序产品不限于此，在本文件中，可读存储介质可以是任何包含或存储程序的有形介质，该程序可以被命令执行系统、装置或者器件使用或者与其结合使用。

可读信号介质可以包括在基带中或者作为载波一部分传播的数据信号，其中承载了可读程序代码。这种传播的数据信号可以采用多种形式，包括但不限于电磁信号、光信号或上述的任意合适的组合。可读信号介质还可以是可读存储介质以外的任何可读介质，该可读介质可以发送、传播或者传输用于由命令执行系统、装置或者器件使用或者与其结合使用的程序。

可读介质上包含的程序代码可以用任何适当的介质传输，包括但不限于无线、有线、光缆、RF 等等，或者上述的任意合适的组合。

本领域普通技术人员可以理解：实现上述方法实施例的全部或部分步骤可以通过程序指令相关的硬件来完成，前述的程序可以存储于一计算机可读取存储介质中，该程序在执行时，执行包括上述方法实施例的步骤；而前述的存储介质包括：移动存储设备、只读存储器（ROM，Read-Only Memory）、随机存取存储器（RAM，Random Access Memory）、磁碟或者光盘等各种可以存储程序代码的介质。

或者，本申请实施例上述集成的单元如果以软件功能模块的形式实现并作为独立的产品销售或使用，也可以存储在一个计算机可读取存储介质中。基于这样的理解，本申请实施例的技术方案本质上或者说对现有技术做出贡献的部分可以以软件产品的形式体现出来，该计算机软件产品存储在一个存储介质中，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)执行本申请实施例各个实施例所述方法的全部或部分。而前述的存储介质包括:移动存储设备、ROM、RAM、磁碟或者光盘等各种可以存储程序代码的介质。

显然,本领域的技术人员可以对本申请进行各种改动和变型而不脱离本申请的精神和范围。这样,倘若本申请的这些更改和变型属于本申请权利要求及其等同技术的范围之内,则本申请也意图包含这些改动和变型在内。

权利要求书

1、一种应用程序控制方法，由电子设备执行，所述方法包括：

在电子设备中运行原生应用和嵌套于所述原生应用中的子应用；

在所述子应用检测到针对目标多媒体内容的操作请求时，向所述原生应用发送对象创建指令，所述对象创建指令包含待创建的原生对象对应的参数信息，以使所述原生应用根据所述参数信息创建所述原生对象，所述原生对象用于在所述子应用中对所述目标多媒体内容进行操作；

当确定所述原生应用已创建所述原生对象时，向所述原生应用发送对象调用指令，以使所述原生应用调用所述原生对象，并执行所述原生对象的功能代码；

接收所述原生应用返回的执行结果，并根据所述执行结果，在所述子应用中对所述目标多媒体内容进行相应的操作。

2、如权利要求1所述的方法，其中，所述当确定所述原生应用已创建所述原生对象时，向所述原生应用发送对象调用指令，具体包括：

在接收到所述原生应用返回的所述原生对象的对象信息时，确定所述原生应用已创建所述原生对象；

根据所述对象信息，向所述原生应用发送所述对象调用指令。

3、如权利要求2所述的方法，其中，所述对象信息包括内存地址和类名；

所述根据所述对象信息，向所述原生应用发送所述对象调用指令，具体包括：

在调用所述原生对象时，将所述内存地址和所述类名添加到所述对象调用指令中，并将所述对象调用指令发送给所述原生应用，以使所述原生应用根据所述内存地址确定为所述原生对象分配的内存，并根据所述类名确定所述原生对象的类型，通过指针索引到为所述原生对象分配的内存后，根据所述原生对象的类型，调用所述原生对象。

4、如权利要求1所述的方法，其中，所述目标多媒体内容的播放界面展示在所述子应用的子应用页面中，所述播放界面与所述子应用页面是通过同层渲染得到的。

5、如权利要求1所述的方法，还包括：

在所述子应用完成对所述目标多媒体内容的操作之后，向所述原生应用发送针对所述原生对象的对象释放指令，以使所述原生应用释放为所述原生对象分配的内存。

6、如权利要求1~5任一项所述的方法，其中，所述子应用与所述原生应用之间通过浏览器组件进行交互。

7、一种应用程序控制方法，由电子设备执行，所述方法包括：

在电子设备中运行原生应用和嵌套于所述原生应用中的子应用；

在接收到所述子应用发送的对象创建指令后，根据所述对象创建指令中待创建的原生对象对应的参数信息，创建所述原生对象，其中，所述对象创建指令是所述子应用检测到针对目标多媒体内容的操作请求时发送的，所述原生对象用于在所述

子应用中对所述目标多媒体内容进行操作；

接收到所述子应用发送的对象调用指令后，调用所述原生对象，并执行所述原生对象的功能代码，其中，所述对象调用指令是所述子应用确定所述原生应用已创建所述原生对象时发送的；

将执行结果返回给所述子应用，以使所述子应用根据所述执行结果，对所述目标多媒体内容进行相应的操作。

8、如权利要求7所述的方法，其中，在所述创建所述原生对象之后，在所述接收到所述子应用发送的对象调用指令之前，所述方法还包括：

将通过创建所述原生对象所获得的对象信息返回给所述子应用，以使所述子应用在接收到所述对象信息时，确定所述原生应用已创建所述原生对象，并根据所述对象信息发送所述对象调用指令。

9、如权利要求7所述的方法，其中，所述根据对象创建指令中待创建的原生对象对应的参数信息，创建所述原生对象，具体包括：

通过解析所述参数信息，从预先配置的应用程序编程接口API集合中，选取创建所述原生对象所需调用的目标API，其中，所述API集合中的API是与所述子应用的音视频编辑能力相关联的API；

调用所述目标API，创建所述原生对象。

10、如权利要求8所述的方法，其中，所述对象信息包括内存地址和类名，所述对象调用指令携带有所述内存地址和所述类名；

所述接收到所述子应用发送的对象调用指令后，调用所述原生对象，具体包括：

根据所述内存地址确定为所述原生对象分配的内存，并根据所述类名确定所述原生对象的类型；

通过指针索引到为所述原生对象分配的内存后，根据所述原生对象的类型，调用所述原生对象。

11、如权利要求7~10任一项所述的方法，其中，在创建所述原生对象之后，所述方法还包括：

使用全局字典存储所述原生对象，以保持所述子应用对所述原生对象的调用。

12、一种应用程序控制装置，包括：

运行单元，用于在电子设备中运行原生应用和嵌套于所述原生应用中的子应用；

第一传输单元，用于在所述子应用检测到针对目标多媒体内容的操作请求时，向所述原生应用发送对象创建指令，所述对象创建指令包含待创建的原生对象对应的参数信息，以使所述原生应用根据所述参数信息创建所述原生对象，所述原生对象用于在所述子应用中对所述目标多媒体内容进行操作；

第二传输单元，用于当确定所述原生应用已创建所述原生对象时，向所述原生应用发送对象调用指令，以使所述原生应用调用所述原生对象，并执行所述原生对象的功能代码；

控制单元，用于接收所述原生应用返回的执行结果，并根据所述执行结果，对

所述目标多媒体内容进行相应的操作。

13、如权利要求 12 所述的装置，其中，所述第二传输单元具体用于：在接收到所述原生应用返回的所述原生对象的对象信息时，确定所述原生应用已创建所述原生对象；根据所述对象信息，向所述原生应用发送所述对象调用指令。

14、如权利要求 13 所述的装置，其中，所述对象信息包括内存地址和类名，所述第二传输单元具体用于：在调用所述原生对象时，将所述内存地址和所述类名添加到所述对象调用指令中，并将所述对象调用指令发送给所述原生应用，以使所述原生应用根据所述内存地址确定为所述原生对象分配的内存，并根据所述类名确定所述原生对象的类型，通过指针索引到为所述原生对象分配的内存后，根据所述原生对象的类型调用所述原生对象。

15、如权利要求 12 所述的装置，其中，所述目标多媒体内容的播放界面展示在所述子应用的子应用页面中，所述播放界面与所述子应用页面是通过同层渲染得到的。

16、如权利要求 12~15 任一项所述的装置，还包括：

第三传输单元，用于在所述子应用完成对所述目标多媒体内容的操作之后，向所述原生应用发送针对所述原生对象的对象释放指令，以使所述原生应用释放为所述原生对象分配的内存。

17、一种应用程序控制装置，包括：

运行单元，用于在电子设备中运行原生应用和嵌套于所述原生应用中的子应用；

对象创建单元，用于在接收到所述子应用发送的对象创建指令后，根据所述对象创建指令中待创建的原生对象对应的参数信息，创建所述原生对象，其中，所述对象创建指令是所述子应用检测到针对目标多媒体内容的操作请求时发送的，所述原生对象用于在所述子应用中对所述目标多媒体内容进行操作；

对象调用单元，用于接收到所述子应用发送的对象调用指令后，调用所述原生对象，并执行所述原生对象的功能代码，其中，所述对象调用指令是所述子应用确定所述原生应用已创建所述原生对象时发送的；

第一反馈单元，用于将执行结果返回给所述子应用，以使所述子应用根据所述执行结果，对所述目标多媒体内容进行相应的操作。

18、如权利要求 17 所述的装置，还包括：

第二反馈单元，用于在所述对象创建单元创建所述原生对象之后，在所述对象调用单元接收到所述子应用发送的对象调用指令之前，将通过创建所述原生对象所获得的对象信息返回给所述子应用，以使所述子应用在接收到所述对象信息时，确定所述原生应用已创建所述原生对象，并根据所述对象信息发送所述对象调用指令。

19、如权利要求 17 所述的装置，其中，所述对象创建单元具体用于：通过解析所述参数信息，从预先配置的应用程序编程接口 API 集合中，选取创建所述原生对象所需调用的目标 API，其中，所述 API 集合中的 API 是与所述子应用的音视频编辑能力相关联的 API；调用所述目标 API，创建所述原生对象。

20、如权利要求 18 所述的装置，其中，所述对象信息包括内存地址和类名，所述对象调用指令携带有所述内存地址和所述类名，所述对象调用单元具体用于：根据所述内存地址确定为所述原生对象分配的内存，并根据所述类名确定所述原生对象的类型；通过指针索引到为所述原生对象分配的内存后，根据所述原生对象的类型，调用所述原生对象。

21、如权利要求 17~20 任一项所述的装置，还包括：

存储单元，用于在所述根据所述对象创建指令中待创建的原生对象对应的参数信息，创建所述原生对象之后，使用全局字典存储所述原生对象，以保持所述子应用对所述原生对象的调用。

22、如权利要求 17~20 任一项所述的装置，还包括：

内存释放单元，用于在接收到所述子应用发送的针对所述原生对象的对象释放指令后，释放为所述原生对象分配的内存，其中，所述对象释放指令是所述子应用在完成对所述目标多媒体内容的操作之后发送的。

23、一种电子设备，其包括处理器和存储器，其中，所述存储器存储有程序代码，当所述程序代码被所述处理器执行时，使得所述处理器执行权利要求 1~6 中任一所述方法的步骤或权利要求 7~11 中任一所述方法的步骤。

24、一种计算机可读存储介质，其包括程序代码，当所述程序代码在电子设备上运行时，所述程序代码用于使所述电子设备执行权利要求 1~6 中任一所述方法的步骤或权利要求 7~11 中任一所述方法的步骤。