

METHOD AND APPARATUS FOR AWAKENING APPLICATION BY MEANS OF MOBILE BROWSER

TECHNICAL FIELD

[0001] The present application relates to the communications field, and in particular,
5 to a method and an apparatus for invoking an application (app) by means of a mobile browser.

BACKGROUND

[0002] A common application scenario for services of an ANDROID mobile device is as follows: A user executes a service process in an app or a browser of an initiator,
10 and the service process needs to invoke some functions of a third-party app, for example, third-party payment and third-party login.

[0003] After the user jumps from the app or the mobile browser of the initiator to the third-party app, and the app completes the corresponding service process, the user usually needs to return to the app or the mobile browser of the initiator to continue the
15 corresponding service process.

[0004] In actual applications, when the initiator is an app on a mobile device of the user, generally the app of the initiator can be returned to based on a jump mechanism between apps in an ANDROID system. Then, when the initiator is a mobile browser on a mobile device of the user, the mobile browser cannot be automatically returned to
20 after a third-party app completes a corresponding service.

SUMMARY

[0005] The present application provides a method for invoking an app by means of a mobile browser. The method includes: receiving first call data submitted by a mobile browser when a service call for a target app is initiated, where the first call data includes
25 a custom parameter used to invoke the target app and identification information of the mobile browser, and the identification information includes package name information

of the mobile browser; invoking the target app based on the custom parameter to execute a corresponding target service, and transferring the identification information of the mobile browser to the target app; and after the target service is executed, responding to second call data carrying the package name information of the mobile browser and submitted by the target app, and returning to the mobile browser.

[0006] Optionally, the identification information of the mobile browser further includes user agent information of the mobile browser, and when the identification information of the mobile browser is the user agent information of the mobile browser, the package name information of the mobile browser is obtained through querying a system server by the target app based on the user agent information of the mobile browser.

[0007] Optionally, the second call data is an intent constructed by the target app based on the package name information of the mobile browser and used to invoke the mobile browser, and the responding to second call data carrying the package name information of the mobile browser and submitted by the target app, and returning to the mobile browser includes: receiving the intent constructed based on the package name information of the mobile browser and submitted by the target app; querying the system server for an activity component corresponding to the package name information of the mobile browser; and invoking the activity component corresponding to the mobile browser, to return to the mobile browser.

[0008] The present application further provides a method for invoking an app by means of a mobile browser. The method includes: obtaining identification information of a mobile browser when a service call for a target app is initiated, where the identification information includes package name information of the mobile browser; and submitting, to a system, first call data for the target app, where the first call data includes a custom parameter used to invoke the target app and the identification information of the mobile browser, so that the system invokes the target app based on the custom parameter, responds, when the target app completes a corresponding target service, to an intent carrying the package name information of the mobile browser and submitted by the target app, and returns to the mobile browser.

[0009] Optionally, the obtaining identification information of a mobile browser when a service call for a target app is initiated includes: obtaining user agent information of the mobile browser; and querying a system server for the package name information of the mobile browser based on the user agent information of the mobile

browser.

[0010] Optionally, the identification information of the mobile browser further includes user agent information of the mobile browser, and when the identification information of the mobile browser is the user agent information of the mobile browser, the package name information of the mobile browser is obtained through querying a system server by the target app based on the user agent information of the mobile browser.

[0011] Optionally, the first call data includes a uniform resource locator (URL) defined based on a Schema protocol, and the submitting, to a system, first call data for the target app when a service call for a target app is initiated includes: defining the URL based on the Schema protocol when the service call for the target app is initiated; encapsulating, in the defined URL, the custom parameter used to invoke a target service page of the target app and the identification information of the mobile browser; and invoking a system interface, and submitting the URL to the system.

[0012] The present application further provides an apparatus for invoking an app by means of a mobile browser. The apparatus includes: a receiving module, configured to receive first call data submitted by a mobile browser when a service call for a target app is initiated, where the first call data includes a custom parameter used to invoke the target app and identification information of the mobile browser, and the identification information includes package name information of the mobile browser; an invoking module, configured to invoke the target app based on the custom parameter to execute a corresponding target service, and transfer the identification information of the mobile browser to the target app; and a responding module, configured to respond to second call data carrying the package name information of the mobile browser and submitted by the target app, and return to the mobile browser, after the target service is executed,

[0013] Optionally, when the identification information of the mobile browser is the user agent information of the mobile browser, the package name information of the mobile browser is obtained through querying a system server by the target app based on the user agent information of the mobile browser.

[0014] Optionally, the second call data is an intent constructed by the target app based on the package name information of the mobile browser and used to invoke the mobile browser, and the responding module is configured to receive the intent constructed based on the package name information of the mobile browser and submitted by the target app; query the system server for an activity component

corresponding to the package name information of the mobile browser; and invoke the activity component corresponding to the mobile browser, to return to the mobile browser.

5 **[0015]** The present application further provides an apparatus for invoking an app by means of a mobile browser. The apparatus includes: an acquisition module, configured to obtain identification information of a mobile browser when a service call for a target app is initiated, where the identification information includes package name information of the mobile browser; and a submitting module, configured to submit, to a system, first call data for the target app, where the first call data includes a custom parameter used to invoke the target app and the identification information of the mobile browser, so that the system invokes the target app based on the custom parameter, responds, when the target app completes a corresponding target service, to an intent carrying the package name information of the mobile browser and submitted by the target app, and returns to the mobile browser.

10 **[0016]** Optionally, the acquisition module is configured to obtain user agent information of the mobile browser; and query a system server for the package name information of the mobile browser based on the user agent information of the mobile browser.

15 **[0017]** Optionally, the identification information of the mobile browser further includes user agent information of the mobile browser, and when the identification information of the mobile browser is the user agent information of the mobile browser, the package name information of the mobile browser is obtained through querying a system server by the target app based on the user agent information of the mobile browser.

20 **[0018]** Optionally, the first call data includes a URL defined based on a Schema protocol, and the submitting module is configured to define the URL based on the Schema protocol when the service call for the target app is initiated; encapsulate, in the defined URL, the custom parameter used to invoke a target service page of the target app and the identification information of the mobile browser; and invoke a system interface, and submit the URL to the system.

25 **[0019]** In the present application, when the mobile browser of a service initiator initiates the service call for the target app, the call data carries the package name information of the mobile browser, and the system transfers the package name information to the target app. Therefore, the target app can submit the intent carrying

the package name information to the system after completing the corresponding target service, so that the system responds to the intent and returns to the mobile browser of the service initiator. After completing the service by invoking the third-party target app, the user can automatically return to the mobile browser of the initiator instead of manually selecting the mobile browser to be returned to, thereby improving user experience.

BRIEF DESCRIPTION OF DRAWINGS

[0020] FIG. 1 is a flowchart illustrating a method for invoking an app by means of a mobile browser, according to an implementation of the present application;

10 **[0021]** FIG. 2 is an example of JavaScript code for obtaining user agent information of a mobile browser by a service front-end page, according to an implementation of the present application;

[0022] FIG. 3 is a Schematic storage diagram illustrating a service list maintained on a system server, according to an implementation of the present application;

15 **[0023]** FIG. 4 is an example of JavaScript code for invoking an ALIPAY app by using a URL defined based on a Schema protocol, according to an implementation of the present application;

[0024] FIG. 5 is an example of JavaScript code for constructing an intent by a target app based on a package name of a mobile browser of a service initiator, according to an implementation of the present application;

20 **[0025]** FIG. 6 is a logical block diagram illustrating an apparatus for invoking an app by means of a mobile browser, according to an implementation of the present application;

[0026] FIG. 7 is a hardware structural diagram illustrating a mobile device carrying an apparatus for invoking an app by means of a mobile browser, according to an implementation of the present application;

[0027] FIG. 8 is a logical block diagram illustrating another apparatus for invoking an app by means of a mobile browser, according to an implementation of the present application; and

30 **[0028]** FIG. 9 is a hardware structural diagram illustrating a mobile device carrying another apparatus for invoking an app by means of a mobile browser, according to an implementation of the present application.

DESCRIPTION OF IMPLEMENTATIONS

[0029] In an ANDROID system, when a mobile browser of a service initiator initiates a service call for a third-party app, the mobile browser can usually define a URL based on a Schema protocol, and invoke a service interface of the third-party app by using the defined URL, to execute a corresponding service process.

[0030] The Schema protocol is an ANDROID system compatible protocol.

[0031] When initiating the service call for the third-party app, the mobile browser can define, based on the Schema protocol, the URL used to invoke the app, encapsulate, in the URL, a custom parameter used to invoke a service interface specified by the app, and then submit the URL to the ANDROID system by using a system application programming interface (API). After receiving the URL, the ANDROID system can parse the Schema protocol, and invoke, by using the custom parameter encapsulated in the URL, the service interface specified by the app, to execute the service call.

[0032] After user initiates the service call for the third-party app by using the mobile browser of the service initiator, and a corresponding service process is completed in the invoked specified service interface of the app, the user usually needs to return to the mobile browser of the service initiator to continue an original service process.

[0033] For example, the user initiates a service call for a third-party payment app by using a shopping page in the mobile browser. In this case, after payment is completed in an invoked payment page of the app, the user usually needs to return to a shopping complete interface in the mobile browser to continue a subsequent process.

[0034] On an existing ANDROID platform, an intent is usually constructed to implement a jump between apps.

[0035] The intent is a method defined for communication between components of ANDROID apps in the ANDROID system. The intent is responsible for describing an action when two components execute an operation, and data and additional data related to the action, and is used to provide related information for mutual calls between components.

[0036] After the user initiates the service call for the third-party app by using the mobile browser of the service initiator, and the corresponding service process is completed in the invoked specified service interface of the app, the app can submit, to the ANDROID system, an intent used to jump to the mobile browser. The ANDROID

system can parse out a parameter carried in the intent, perform intent matching, and invoke an activity component of another app corresponding to a matching intent, to jump to a specified page of the another app.

[0037] However, because a URL defined by the mobile browser based on the Schema protocol usually does not support carrying source information of an app, when the ANDROID system performs intent matching based on the parameter carried in the intent submitted by the app, it is possible that the ANDROID system cannot accurately obtain the mobile browser of the service initiator through matching due to insufficient information of the mobile browser.

[0038] In this case, the ANDROID system usually invokes all mobile browsers in the system, and pops up a browser selection window for the user, so that the user manually selects the mobile browser to be returned to.

[0039] It can be seen that the previous solution has at least the following disadvantages.

[0040] In a first aspect, the ANDROID system cannot automatically return to the mobile browser of the service initiator, and therefore, the user needs to perform manual selection. As such, the tedious operations can cause poor user experience.

[0041] In a second aspect, the ANDROID system has various types of mobile browsers (for example, a native browser default in the system, a QQ browser, a UC browser, and a 360 browser), but the user is not familiar with these browsers. Consequently, a mobile browser manually selected by the user may not be the mobile browser of the original service initiator.

[0042] In a third aspect, web apps such as a mobile browser usually use a cookie to record a user operation status (for example, a user's login status and a security control check status). If a mobile browser selected by the user is not the mobile browser of the original service initiator, the user needs to resume these statuses through additional operations when continuing the service process in the mobile browser. Consequently, operations performed by the user increase and can affect user experience.

[0043] In view of this, the present application provides a method for invoking an app by means of a mobile browser. When the mobile browser of the service initiator initiates a service call for a target app, call data carries package name information of the mobile browser, and the system transfers the package name information to the target app. Therefore, the target app can submit an intent carrying the package name information to the system after completing a corresponding target service, so that the

system responds to the intent, and returns to the mobile browser of the service initiator. After completing the service by invoking the third-party target app, the user can automatically return to the mobile browser of the initiator instead of manually selecting the mobile browser to be returned to, thereby improving user experience.

5 **[0044]** The following describes the present application by using specific implementations with reference to specific application scenarios.

10 **[0045]** FIG. 1 shows a method for invoking an app by means of a mobile browser, according to an implementation of the present application. The method is applied to a mobile device. The mobile device can carry an ANDROID system. A mobile browser and at least one target app are installed in the ANDROID system. The mobile browser interacts with the ANDROID system, to perform the following steps:

15 **[0046]** Step 101: Obtain identification information of the mobile browser when the mobile browser initiates a service call for a target app, where the identification information includes package name information of the mobile browser.

20 **[0047]** In the present implementation, the service call from the mobile browser for the target app can be manually triggered by the user in a local service interface of the mobile browser.

25 **[0048]** In a shown implementation, a user option available for the user to operate can be provided in advance in the local service interface of the mobile browser. The user option can be an interaction button or a link. For example, the user initiates a service call in a local shopping page of the mobile browser for an ALIPAY app to execute a payment process. The user option can be an interaction button "use ALIPAY for payment" provided in the shopping page, or can be a link that can trigger a jump to a payment page of the ALIPAY app.

30 **[0049]** The user can touch the user option in the service interface, to trigger the mobile browser to initiate a service call for a third-party target app. After listening to the user's touch event for the user option at background, the mobile browser can immediately initiate a service interface call for the target app by using the ANDROID system.

35 **[0050]** After receiving the call request for a local service interface, the target app can return a URL of a corresponding service front-end page to the mobile browser. The service browser can jump to the corresponding service front-end page after receiving the URL.

40 **[0051]** The service front-end page is a web page loaded on the mobile browser and

used to execute a service in the third-party target app. In actual applications, a specific function of the service front-end page can be jointly developed by a supplier of the mobile browser and a supplier of the third-party target app.

[0052] Still in the implementation that the user initiates a service call in a local shopping page of the mobile browser for an ALIPAY app to execute a payment process, the service front-end page can be a service entry interface jointly developed by the supplier of the mobile browser and the supplier of the ALIPAY app and used to invoke the ALIPAY app to perform payment. Information about details such as a payment amount and a to-be-purchased product can be displayed in the service entry interface. The user can check the current payment based on the information displayed in the interface.

[0053] In the present implementation, after the mobile browser jumps to the service front-end page, the service front-end page, as a front end of service execution, can obtain the identification information of the mobile browser serving as a service initiator.

[0054] During implementation, the identification information of the mobile browser can include a package name (package name information) or user agent information of the mobile browser. The package name is a unique identifier defined by the ANDROID system for each app in the system. The user agent information generally records information such as a name and a version of a browser. The version information of the mobile browser generally includes information such as a system environment of the mobile browser.

[0055] In a shown implementation, when the identification information of the mobile browser is the user agent information, the service front-end page can directly obtain the user agent information of the mobile browser by executing JavaScript code loaded on the service front-end page.

[0056] For example, FIG. 2 is an example of JavaScript code for obtaining user agent information of a mobile browser by a service front-end page, according to the present implementation.

[0057] In the code example shown in FIG. 2, ua=navigator.userAgent is used to return a name of a mobile browser, and version=navigator.appVersion is used to return information about a system environment of a mobile browser.

[0058] In another shown implementation, when the identification information of the mobile browser is the package name, the service front-end page can use the obtained user agent information as an index to query a system server of the ANDROID system

for the package name of the mobile browser when the service front-end page obtains the package name of the current mobile browser.

5 **[0059]** A service list storing a mapping relationship between user agent information of a mobile browser, a package name of the mobile browser, and an activity component of the mobile browser can be maintained in the system server in advance.

[0060] After obtaining the user agent information of the current mobile browser, the service front-end page can send the user agent information of the mobile browser to the system server.

10 **[0061]** When sending the user agent information of the mobile browser to the system server, the service front-end page can upload the user agent information of the mobile browser to the system server in a form of a request by using an AJAX interface between the service front-end page and the system server.

[0062] After receiving the user agent information sent by the service front-end page, the system server can use the user agent information as an index to search the service list for the corresponding package name, and then return the obtained package name to the service front-end page.

[0063] For example, FIG. 3 is a Schematic storage diagram illustrating a service list maintained on a system server, according to the present implementation.

20 **[0064]** When searching for the corresponding package name based on the user agent information, the system server can first read the user agent information to obtain version information of the mobile browser. The version information of the mobile browser generally includes information such as a version of the mobile browser and the system environment. The system server can determine a type of the mobile browser from the service list based on the obtained information about the system environment.

25 For example, the system server can determine whether a bottom-layer environment of the mobile browser is based on the ANDROID system or an IOS system. After determining the type of the mobile browser, the system server can further search for the corresponding package name based on the name of the mobile browser in the user agent information. In this case, the obtained package name is the unique identifier of the

30 mobile browser in the ANDROID system. The system server can return the package name to the service front-end page.

[0065] Step 102: Submit first call data for the target app to a system, where the first call data includes a custom parameter used to invoke the target app and the identification information of the mobile browser.

[0066] Step 103: The system receives the first call data submitted by the mobile browser when the service call for the target app is initiated.

[0067] In a shown implementation, the first call data can be a URL defined by the service front-end page based on a Schema protocol.

5 **[0068]**

When the service front-end page initiates the service call for the target app, the service front-end page can define, based on the Schema protocol, a URL used to invoke the target app, and encapsulate, in the URL, a predefined custom parameter used to invoke a target service page of the target app.

10 **[0069]** In addition, the service front-end page can further use the obtained identification information of the current mobile browser as an additional parameter that needs to be transferred to the target app, and encapsulate the identification information in the URL. Certainly, in actual applications, the Schema protocol supported by the ANDROID system can be improved, so that the Schema protocol can support carrying
15 source information of an app of the service initiator by default, such as a package name. As such, the URL defined based on the Schema protocol carries the package name of the mobile browser of the service initiator by default instead of encapsulating the package name as the additional parameter.

[0070] For example, the user initiates a service call for the ALIPAY app based on
20 an intent protocol by using the mobile browser, and the identification information of the mobile browser is the package name. The URL defined by the service front-end page based on the Schema protocol and used to invoke the ALIPAY app can be as follows:

url=intent://schemeapi/startapp?+patams+#Intent;scheme=+encodeURICo
25 mponent(zmverify)+;package=com.mobile.android;

[0071] In the URL, "intent://schemeapi/startapp" is a protocol header defined based on a scheme. "+patams+#Intent;scheme=+encodeURIComponent (zmverify)" after the parameter identifier ? is a custom parameter used to invoke an activity component corresponding to a payment page in the ALIPAY app. The ANDROID system can
30 match the above parameters with an intent corresponding to the activity component of the ALIPAY app and registered in a system configuration file, to locate the activity component that needs to be invoked. "package=com.mobile.android;" is the package name of the mobile browser that needs to be transferred to the ALIPAY app.

[0072] In the present implementation, the URL defined by the service front-end

page based on the Schema protocol can be submitted by the service front-end page to the ANDROID system in a form of a request by invoking an API system interface.

[0073] The user can manually trigger the service front-end page to submit the service call for the target app to the ANDROID system.

5 **[0074]** In a shown implementation, a user button available for the user to operate can be provided in advance on the service front-end page. The user can touch the user button to trigger the service front-end page to submit the URL to the ANDROID system in a form of a request.

[0075] Still in the implementation that the user initiates a service call in a local shopping page of the mobile browser for the ALIPAY app to execute a payment process, and the identification information of the mobile browser of the service initiator is the package name, a user button "use ALIPAY for payment" can be provided on the service front-end page. The user button can be bound with an execution method defined in JavaScript code shown in FIG. 4.

15 **[0076]** The user can "click" the user button to trigger the service front-end page to invoke the execution method bound with the user button and invoke the API system interface to submit the URL to the ANDROID system in a form of a request.

[0077] In the JavaScript code shown in FIG. 4, zmVerifyUrl is the URL defined by the service front-end page based on the Schema protocol and used to invoke the ALIPAY app; the label "a" corresponds to the user button; "openIntentLink.dispatchEvent (ZmVerify.customClickEvent ())" indicates that the label "a" is bound with a user click event, and when the user click event corresponding to the label "a" for the user button is listened to, the URL is submitted to the ANDROID system in a form of a request.

25 **[0078]** After receiving the URL submitted by the service front-end page, the ANDROID system can parse the URL submitted by the service front-end page, to obtain the custom parameter and the identification information of the mobile browser of the service initiator that are carried in the URL.

[0079] Step 104: Invoke the target app based on the custom parameter to execute a corresponding target service, and transfer the identification information of the mobile browser to the target app.

30 **[0080]** In this implementation, after the ANDROID system parses the URL submitted by the service front-end page and obtains the custom parameter and the identification information of the mobile browser of the service initiator in the URL:the

ANDROID system can match the custom parameter in the URL with the intent corresponding to the activity component of the ALIPAY app and registered in the system configuration file (for example, the system configuration file can be an AndroidManifest.xml file in the ANDROID system), to locate the activity component that needs to be invoked. When any intent registered in the system file is obtained after the matching, the ANDROID system can immediately invoke the activity component corresponding to the intent to jump to the target service page of the target app.

[0081] In addition, after invoking the activity component corresponding to the target app and corresponding to the matching intent, the ANDROID system can further transfer, to the target app, the identification information of the mobile browser of the service initiator that is encapsulated in the URL.

[0082] It is worthwhile to note that, if the identification information of the mobile browser of the service initiator that is submitted by the service front-end page to the ANDROID system is the user agent information, the target app can send the received user agent information to the system server of the ANDROID system. The system server uses the user agent information as an index, searches, for the corresponding package name, the service list storing the mapping relationship between user agent information of a mobile browser, a package name of the mobile browser, and an activity component of the mobile browser and maintained in the system server, and returns the found package name to the target app.

[0083] Step 105: After the target service is executed, respond to second call data carrying the package name information of the mobile browser and submitted by the target app, and return to the mobile browser.

[0084] In a shown implementation, the second call data can be an intent constructed by the target app based on the package name information of the mobile browser and used to invoke the mobile browser of the service initiator.

[0085] When the ANDROID system invokes the activity component corresponding to the matching intent and jumps to the target service page of the target app, the user can execute a subsequent service process on the target service page.

[0086] For example, when the target app is the ALIPAY app, the target service page can be the payment page of the ALIPAY app, and the user can enter a payment password in the payment page to complete payment.

[0087] After the user completes the service process of the target service in the target service page:

the target app can push a service execution result to the mobile browser of the service initiator, to trigger the mobile browser to jump to a service end page.

[0088] In addition, the target app can construct the intent based on the package name of the mobile browser of the service initiator that is transferred by the ANDROID system, or the package name of the mobile browser of the service initiator that is found by the system server, and submits the intent to the ANDROID system to trigger the ANDROID system to invoke the activity component of the mobile browser of the initiator, to return to the mobile browser of the service initiator.

[0089] For example, FIG. 5 is an example of JavaScript code for constructing an intent by a target app based on a package name of a mobile browser of a service initiator, according to the present implementation. In the code example shown in FIG. 5, "new ComponentName (Package Name, appName)" is a full name of the constructed intent; "android.intent.action.View" is a standard intent supported by the ANDROID system and means to open a browser program and show the URL; "intent.FLAG_ACTIVITY_NEW_TASK" is also a standard intent supported by the ANDROID system and means to invoke an activity component corresponding to the package name and put the activity component to the top of an activity stack maintained in the ANDROID system (the activity component is an interactive window displayed in a system, and putting the activity component to the top of the stack means to move, to the uppermost layer of an ANDROID system desktop, the interactive window corresponding to the activity component).

[0090] In the code example shown in FIG. 5, the package name of the browser of the service initiator is specified in ComponentName, and the ANDROID system can accurately locate, based on the package name, the activity component that needs to be invoked. Therefore, the ANDROID system executes an explicit intent call (intent calls include an explicit call and an implicit call, where the explicit call means that an activity component of a to-be-invoked party is explicit, while the implicit call means that an activity component of a to-be-invoked party is implicit and the activity component is located and invoked through performing matching of attribute parameters defined in an intent).

[0091] In the present implementation, after receiving the intent submitted by the target app after the service process of the target service is completed, the ANDROID system can parse the intent to obtain the package name of the mobile browser of the service initiator that is carried in the intent, and then upload the package name to the

system server. The system server uses the package name as an index, searches, for the activity component of the mobile browser, the service list storing the mapping relationship between user agent information of a mobile browser, a package name of the mobile browser, and an activity component of the mobile browser and maintained in the system server, and returns a found name of the activity component to the ANDROID system.

[0092] After obtaining the activity component of the mobile browser of the service initiator, the ANDROID system can respond to the intent submitted by the target app, invoke the activity component of the mobile browser, and put the activity component of the mobile browser to the top of the activity stack maintained in the ANDROID system, to return to the mobile browser of the service initiator. In this case, the interactive window of the mobile browser is displayed at the uppermost layer of the system desktop. After the user completes the service process of the third party on the target service page of the target app, the system returns to the service end page of the mobile browser of the service initiator.

[0093] When responding to the intent submitted by the target app, the ANDROID system can accurately locate the activity component of the mobile browser of the service initiator that needs to be invoked by using package name, the system can execute the explicit intent call, and automatically jump to the service end page of the mobile browser of the service initiator. In the entire process, the system does not need to pop up the browser selection window for the user to manually select the mobile browser to be returned to, thereby improving user experience.

[0094] In the above implementation, when the mobile browser of the service initiator initiates the service call for the target app, the call data carries the package name information of the mobile browser, and the system transfers the package name information to the target app. Therefore, the target app can submit the intent carrying the package name information to the system after completing the corresponding target service, so that the system responds to the intent and returns to the mobile browser of the service initiator.

[0095] Because the intent submitted by the target app carries the package name information of the mobile browser of the service initiator, the ANDROID system can accurately locate, based on the package name information, the activity component that needs to be invoked. Therefore, when responding to the intent, the ANDROID system can execute the explicit intent call and automatically jump to the mobile browser of the

service initiator instead of popping up the browser selection window for the user to manually select the mobile browser to be returned to, thereby simplifying the operations. In addition, the user does not need to perform additional operations if the user manually selects another browser different from the mobile browser of the service initiator to resume user operation statuses recorded by a browser, thereby improving user experience.

[0096] The present application further provides an apparatus implementation corresponding to the previous method implementation.

[0100] Referring to FIG. 6, the present application provides an apparatus 60 for invoking an app by means of a mobile browser. The apparatus 60 is applied to an ANDROID system. Referring to FIG. 7, a hardware architecture involved in a mobile device carrying the ANDROID system usually includes a CPU, a memory, a non-volatile memory, a network interface, an internal bus, etc. In an example of a software implementation, the apparatus 60 for invoking an app by means of a mobile browser may usually be understood as a logical apparatus of a software and hardware combination obtained after the CPU runs a computer program loaded in the memory. The apparatus 60 includes: a receiving module 601, configured to receive first call data submitted by a mobile browser when a service call for a target app is initiated, where the first call data includes a custom parameter used to invoke the target app and identification information of the mobile browser, and the identification information includes package name information of the mobile browser; an invoking module 602, configured to invoke the target app based on the custom parameter to execute a corresponding target service, and transfer the identification information of the mobile browser to the target app; and a responding module 603, configured to: respond to second call data carrying the package name information of the mobile browser and submitted by the target app and return to the mobile browser, after the target service is executed.

[0101] In the present implementation, when the identification information of the mobile browser is user agent information of the mobile browser, the package name information of the mobile browser is obtained through querying a system server by the target app based on the user agent information of the mobile browser.

[0102] In the present implementation, the second call data is an intent constructed by the target app based on the package name information of the mobile browser and used to invoke the mobile browser.

[0103] The responding module 603 is configured to receive the intent constructed based on the package name information of the mobile browser and submitted by the target app; query the system server for an activity component corresponding to the package name information of the mobile browser; and invoke the activity component corresponding to the mobile browser, to return to the mobile browser.

[0104] Referring to FIG. 8, the present application provides an apparatus 80 for invoking an app by means of a mobile browser. The apparatus 80 is applied to a mobile browser. Referring to FIG. 9, a hardware architecture involved in a mobile device carrying the mobile browser usually includes a CPU, a memory, a non-volatile memory, a network interface, an internal bus, etc. In an example of a software implementation, the apparatus 90 for invoking an app by means of a mobile browser may usually be understood as a logical apparatus of a software and hardware combination obtained after the CPU runs a computer program loaded in the memory. The apparatus 90 includes: an acquisition module 801, configured to obtain identification information of the mobile browser when a service call for a target app is initiated, where the identification information includes package name information of the mobile browser; and a submitting module 802, configured to submit, to a system, first call data for the target app, where the first call data includes a custom parameter used to invoke the target app and the identification information of the mobile browser, so that the system invokes the target app based on the custom parameter, responds, when the target app completes a corresponding target service, to an intent carrying the package name information of the mobile browser and submitted by the target app, and returns to the mobile browser.

[0105] In the present implementation, the acquisition module 801 is configured to: obtain user agent information of the mobile browser; and query a system server for the package name information of the mobile browser based on the user agent information of the mobile browser.

[0106] In the present implementation, the identification information of the mobile browser further includes user agent information of the mobile browser.

[0107] When the identification information of the mobile browser is the user agent information of the mobile browser, the package name information of the mobile browser is obtained through querying a system server by the target app based on the user agent information of the mobile browser.

[0108] In the present implementation, the first call data includes a URL defined

based on a Schema protocol.

5 **[0109]** The submitting module 802 is configured to define the URL based on the Schema protocol when the service call for the target app is initiated; encapsulate, in the defined URL, the custom parameter used to invoke a target service page of the target app and the identification information of the mobile browser; and invoke a system interface, and submit the URL to the system.

10 **[0110]** A person skilled in the art can easily figure out another implementation solution of the present application after considering the present specification and practicing the disclosed invention here. The present application is intended to cover any variations, functions, or adaptive changes of the present application. These variations, functions, or adaptive changes comply with general principles of the present application, and include common knowledge or a commonly used technical means in the technical field that is not disclosed in the present application. The present specification and the implementations are merely considered as examples. The actual scope and the spirit of
15 the present application are pointed out by the following claims.

20 **[0111]** It should be understood that the present application is not limited to the specific structures described above and shown in the accompanying drawings, and modifications and changes may be made without departing from the scope of the present application. The scope of the present application is limited only by the appended claims.

25 **[0112]** The previous descriptions are merely examples of implementations of the present application, but are not intended to limit the present application. Any modification, equivalent replacement, or improvement made without departing from the spirit and principle of the present application should fall within the protection scope of the present application.

CLAIMS

What is claimed is:

1. A method for invoking an app by means of a mobile browser, wherein the method comprises:

5 receiving first call data submitted by a mobile browser when a service call for a target app is initiated, wherein the first call data comprises a custom parameter used to invoke the target app and identification information of the mobile browser, and the identification information comprises package name information of the mobile browser;

invoking the target app based on the custom parameter to execute a corresponding target service, and transferring the identification information of the mobile browser to the target app; and

after the target service is executed, responding to second call data carrying the package name information of the mobile browser and submitted by the target app, and returning to the mobile browser.

15 2. The method according to claim 1, wherein the identification information of the mobile browser further comprises user agent information of the mobile browser, and

when the identification information of the mobile browser is the user agent information of the mobile browser, the package name information of the mobile browser is obtained through querying a system server by the target app based on the user agent information of the mobile browser.

3. The method according to claim 1, wherein the second call data is an intent constructed by the target app based on the package name information of the mobile browser and used to invoke the mobile browser, and

the responding to second call data carrying the package name information of the mobile browser and submitted by the target app, and returning to the mobile browser comprises:

receiving the intent constructed based on the package name information of the mobile browser and submitted by the target app;

30 querying the system server for an activity component corresponding to the package name information of the mobile browser; and

invoking the activity component corresponding to the mobile browser, to return to the mobile browser.

4. A method for invoking an app by means of a mobile browser, wherein the method comprises:

obtaining identification information of a mobile browser when a service call for a target app is initiated, wherein the identification information comprises package name

5 information of the mobile browser; and

submitting, to a system, first call data for the target app, wherein the first call data comprises a custom parameter used to invoke the target app and the identification information of the mobile browser, so that the system invokes the target app based on the custom parameter, responds, when the target app completes a corresponding target service, to an intent carrying the package name information of the mobile browser and submitted by the target app, and returns to the mobile browser.

5. The method according to claim 4, wherein the obtaining identification information of a mobile browser when a service call for a target app is initiated comprises:

15 obtaining user agent information of the mobile browser; and

querying a system server for the package name information of the mobile browser based on the user agent information of the mobile browser.

6. The method according to claim 4, wherein the identification information of the mobile browser further comprises user agent information of the mobile browser, and

20 when the identification information of the mobile browser is the user agent information of the mobile browser, the package name information of the mobile browser is obtained through querying a system server by the target app based on the user agent information of the mobile browser.

7. The method according to claim 4, wherein the first call data comprises a URL defined based on a Schema protocol, and

the submitting, to a system, first call data for the target app when a service call for a target app is initiated comprises:

defining the URL based on the Schema protocol when the service call for the target app is initiated;

30 encapsulating, in the defined URL, the custom parameter used to invoke a target service page of the target app and the identification information of the mobile browser; and

invoking a system interface, and submitting the URL to the system.

8. An apparatus for invoking an app by means of a mobile browser, wherein the

apparatus comprises:

a receiving module, configured to receive first call data submitted by a mobile browser when a service call for a target app is initiated, wherein the first call data comprises a custom parameter used to invoke the target app and identification information of the mobile browser, and the identification information comprises package name information of the mobile browser;

an invoking module, configured to invoke the target app based on the custom parameter to execute a corresponding target service, and transfer the identification information of the mobile browser to the target app; and

a responding module, configured to: after the target service is executed, respond to second call data carrying the package name information of the mobile browser and submitted by the target app, and return to the mobile browser.

9. The apparatus according to claim 8, wherein when the identification information of the mobile browser is user agent information of the mobile browser, the package name information of the mobile browser is obtained through querying a system server by the target app based on the user agent information of the mobile browser.

10. The apparatus according to claim 8, wherein the second call data is an intent constructed by the target app based on the package name information of the mobile browser and used to invoke the mobile browser, and

the responding module is configured to:
receive the intent constructed based on the package name information of the mobile browser and submitted by the target app;

query the system server for an activity component corresponding to the package name information of the mobile browser; and

invoke the activity component corresponding to the mobile browser, to return to the mobile browser.

11. An apparatus for invoking an app by means of a mobile browser, wherein the apparatus comprises:

an acquisition module, configured to obtain identification information of a mobile browser when a service call for a target app is initiated, wherein the identification information comprises package name information of the mobile browser; and

a submitting module, configured to submit, to a system, first call data for the target app, wherein the first call data comprises a custom parameter used to invoke the target app and the identification information of the mobile browser, so that the system invokes

the target app based on the custom parameter, responds, when the target app completes a corresponding target service, to an intent carrying the package name information of the mobile browser and submitted by the target app, and returns to the mobile browser.

5 12. The apparatus according to claim 11, wherein the acquisition module is configured to:

obtain user agent information of the mobile browser; and

query a system server for the package name information of the mobile browser based on the user agent information of the mobile browser.

10 13. The apparatus according to claim 11, wherein the identification information of the mobile browser further comprises user agent information of the mobile browser, and

when the identification information of the mobile browser is the user agent information of the mobile browser, the package name information of the mobile browser is obtained through querying a system server by the target app based on the user agent information of the mobile browser.

15 14. The apparatus according to claim 11, wherein the first call data comprises a URL defined based on a Schema protocol, and

the submitting module is configured to:

20 define the URL based on the Schema protocol when the service call for the target app is initiated;

encapsulate, in the defined URL, the custom parameter used to invoke a target service page of the target app and the identification information of the mobile browser; and

invoke a system interface, and submit the URL to the system.