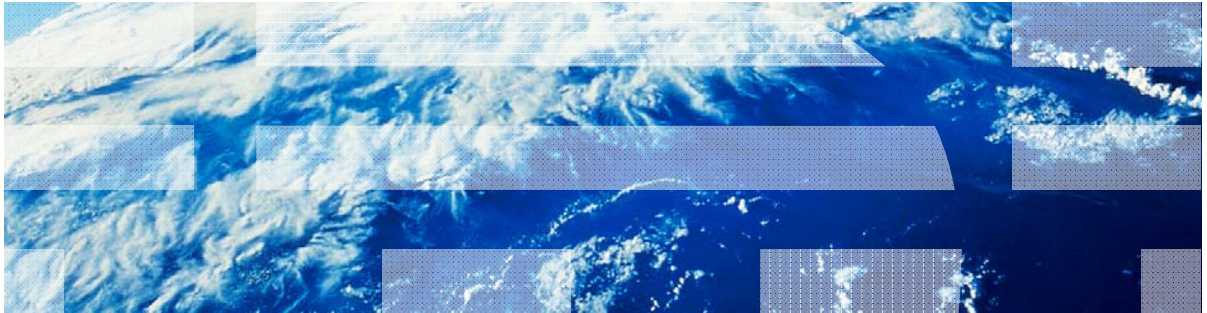


# ***IBM Worklight V6.0.0 Getting Started***

## Using Worklight API in native Android applications



## Trademarks

- IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Worklight is a trademark or registered trademark of Worklight, an IBM Company. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “[Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)” at [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).
- Other company products or service names may be trademarks or service marks of others.
- This document may not be reproduced in whole or in part without the prior written permission of IBM.

## About IBM®

- See <http://www.ibm.com/ibm/us/en/>

# Agenda

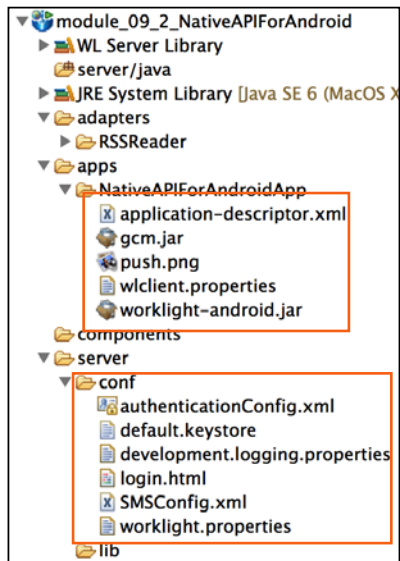
- Creating a Worklight native API
- Create and configure an Android native application
- Initializing the WLClient
- Invoking a Worklight procedure
- Receiving a procedure response

## ***Creating a Worklight native API (1 of 4)***

- IBM Worklight® provides the ability for the native Android applications to communicate with a Worklight server by using the IBM Worklight native API library.
- To serve a native Android application, the Worklight server must be aware of it.
- The Worklight native API is located under the `apps` folder of your Worklight project.
- The Worklight native API folder serves two purposes:
  - It contains a native API library and configuration file that must be copied to your native Android project.
  - It contains the **application-descriptor.xml** file, which must be deployed to a Worklight server as an entry point, similar to a Worklight application.
- In this module, you learn how to create a Worklight native API and how to use its components in your native Android application.

## Creating a Worklight native API (2 of 4)

- A Worklight native API contains several components:



The **application-descriptor.xml** file is used to define application metadata and to configure security settings to be enforced by a Worklight server.

The **wlclient.properties** file contains the connectivity settings to be used by a native Android application.

This file must be copied to your native Android project.

The **worklight-android.jar** is a Worklight API library that must be copied to your native Android project.

The **gcm.jar** and **push.png** files are used for Google push notifications services

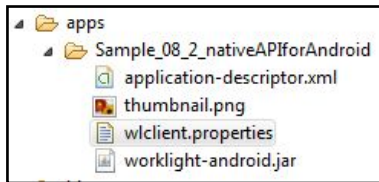
Like with any Worklight project, you can create server configuration by modifying files under the **server\conf** folder.

## ***Creating a Worklight native API (3 of 4)***

1. In Worklight Studio, create a Worklight project, and add a Worklight Native API.
2. In the New Worklight Native API dialog, enter your application name, and select **Android** for the **Environment** field.
3. Right-click the Worklight native API folder and select **Build and Deploy**.

## Creating a Worklight native API (4 of 4)

- Edit the ***wlclient.properties*** file, which holds server configuration:



```

6 wlServerProtocol = http
7 wlServerHost = 10.0.0.7
8 wlServerPort = 8080
9 wlServerContext = /
0 wlAppId = NativeAPIForAndroidApp
1 wlAppVersion = 1.0
2 wlEnvironment = Androidnative
3 #For Push Notifications, uncomment
4 #GcmSenderId =

```

- wlServerProtocol*** – The communication protocol to the Worklight Server. Can be either ***http*** or ***https***.
- wlServerHost*** – The hostname of the Worklight Server.
- wlServerPort*** – The port of the Worklight Server.
- wlServerContext*** – The context root path of the application on Worklight server.
- wlAppId*** – The application ID as defined in the ***application-descriptor.xml*** file.
- wlAppVersion*** – The application version.
- wlEnvironment*** – The target environment of the native application (Android/iOS).
- GcmSenderId*** - This property defines the GCM Sender Id to be used for push notifications. By default, this property is commented.

# Agenda

- Creating a Worklight native API
- Create and configure a native Android application
- Initializing the WLClient
- Invoking a Worklight procedure
- Receiving a procedure response



## Create and configure an Android native application

- Create a native Android application.
- Copy the file **worklight-android.jar** from the Worklight native API folder to the new native Android application under the **/libs** directory.
- Copy the file **wlclient.properties** from the Worklight native API folder to the new native Android application under the **/assets** directory.
- Add the internet permission to the **AndroidManifest.xml** file.

```
android:targetSdkVersion="15" />  
<uses-permission android:name="android.permission.INTERNET"/>  
<application
```

- Add WiFi permission to the **AndroidManifest.xml** file.

```
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>  
<application
```

- Add the Worklight UI activity to the **AndroidManifest.xml** file.

```
</activity>  
<activity android:name="com.worklight.wlclient.ui.UIActivity" />
```

## Agenda

- Creating a Worklight native API
- Create and configure an Android native application
- Initializing the WLCLient
- Invoking a Worklight procedure
- Receiving a procedure response

## Initializing the WLClient

- Start by creating an instance of the **WLClient**.
- The **WLClient** instance requires a reference to the activity in which it is running.

```
final WLClient client = WLClient.createInstance(this);
```
- To establish a connection to a Worklight Server, use the **connect** method by specifying the **MyConnectListener** class instance as a parameter.
  - You learn how to create it in following slides.

```
buttonConnect.setOnClickListener(new OnClickListener() {  
    public void onClick(View v) {  
        updateTextView("Connecting...");  
        client.connect(new MyConnectListener());  
    }  
});
```

## *MyConnectListener*

- The **WLClient** instance tries to connect to a Worklight server according to properties of the **wlclient.properties** file.
- After the connection is established, it invokes one of the methods of the **MyConnectListener** class.
- First, specify that the **MyConnectListener** class implements the **WLResponseListener** interface.

```
public class MyConnectListener implements WLResponseListener {
```

- The **WLResponseListener** interface defines two methods:
  - **The public void onSuccess (WLResponse response) { }**
  - **The public void onFailure (WLFailResponse response) { }**
- Use them to process connection success or connection failure.

# Agenda

- Creating a Worklight native API
- Create and configure an Android native application
- Initializing the WLClient
- Invoking a Worklight procedure
- Receiving a procedure response

## Invoking a Worklight procedure (1 of 3)

- After the connection is established with a Worklight Server, you can use the **WLClient** instance to invoke adapter procedures.

```
buttonInvoke.setOnClickListener(new OnClickListener() {  
    public void onClick(View v) {  
        updateTextView("Invoking procedure...");  
  
        String adapterName = "RSSReader";  
        String procedureName = "getFeedsFiltered";  
  
        WLProcedureInvocationData invocationData =  
            new WLProcedureInvocationData(adapterName, procedureName);  
  
        Object[] parameters = new Object[] {"param1", "param2", 3, false};  
        invocationData.setParameters(parameters);  
  
        WLRequestOptions options = new WLRequestOptions();  
        options.setTimeout(30000);  
  
        WLClient client = WLClient.getInstance();  
        client.invokeProcedure(invocationData, new MyInvokeListener(), options);  
    }  
});
```

- Create a **WLProcedureInvocationData** object with the adapter and procedure names.

## Invoking a Worklight procedure (2 of 3)

- After the connection is established with a Worklight Server, you can use the **WLClient** instance to invoke adapter procedures.

```
buttonInvoke.setOnClickListener(new OnClickListener() {  
    public void onClick(View v) {  
        updateTextView("Invoking procedure...");  
  
        String adapterName = "RSSReader";  
        String procedureName = "getFeedsFiltered";  
  
        WLProcedureInvocationData invocationData =  
            new WLProcedureInvocationData(adapterName, procedureName);  
  
        Object[] parameters = new Object[] {"param1", "param2", 3, false};  
        invocationData.setParameters(parameters);  
  
        WLRequestOptions options = new WLRequestOptions();  
        options.setTimeout(30000);  
  
        WLClient client = WLClient.getInstance();  
        client.invokeProcedure(invocationData, new MyInvokeListener(), options);  
    }  
});
```

- Add the required parameters as an object array and set request options (for example: timeout).

## Invoking a Worklight procedure (3 of 3)

- After the connection is established with a Worklight Server, you can use the **WLClient** instance to invoke adapter procedures.

```
buttonInvoke.setOnClickListener(new OnClickListener() {  
    public void onClick(View v) {  
        updateTextView("Invoking procedure...");  
  
        String adapterName = "RSSReader";  
        String procedureName = "getFeedsFiltered";  
  
        WLProcedureInvocationData invocationData =  
            new WLProcedureInvocationData(adapterName, procedureName);  
  
        Object[] parameters = new Object[] {"param1", "param2", 3, false};  
        invocationData.setParameters(parameters);  
  
        WLRequestOptions options = new WLRequestOptions();  
        options.setTimeout(30000);  
  
        WLClient client = WLClient.getInstance();  
        client.invokeProcedure(invocationData, new MyInvokeListener(), options);  
    }  
});
```

- Get existing the **WLClient** instance and use it to invoke adapter procedure.
- Specify the **MyInvokeListener** class instance as a parameter.
- You learn how to define it in the following slides.



## Agenda

- Creating a Worklight native API
- Create and configure an Android native application
- Initializing the WLClient
- Invoking a Worklight procedure
- Receiving a procedure response

## Receiving a procedure response (1 of 3)

- After the procedure invocation is completed, the **WLClient** instance calls one of the methods of the **MyInvokeListener** class.
- As before, you must specify that the **MyConnectListener** class implements the **WLResponseListener** interface.

```
import com.worklight.wlclient.api.WLFailResponse;  
import com.worklight.wlclient.api.WLResponse;  
import com.worklight.wlclient.api.WLResponseListener;  
  
public class MyInvokeListener implements WLResponseListener {
```

- The **onSuccess** and **onFailure** methods is invoked by the **WLClient**.

## Receiving a procedure response (2 of 3)

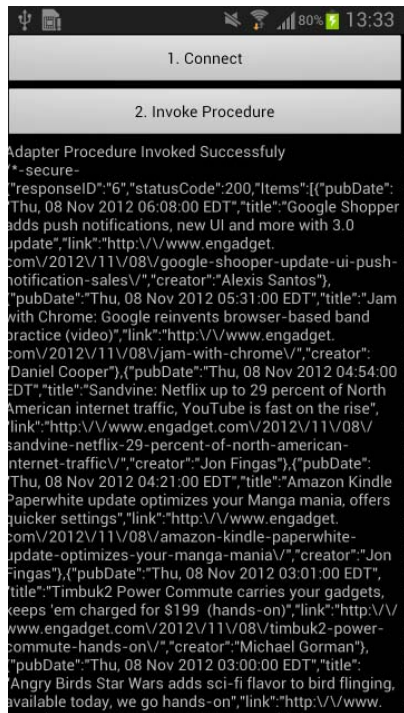
- If the procedure invocation is successful, the **onSuccess** method of **MyInvokeListener** is invoked.
  - Use it to get data that are retrieved from the adapter.

```
public class MyConnectListener implements WLResponseListener {  
    public void onSuccess(WLResponse response) {  
        String responseText = response.getResponseText();  
        Log.d("ConnectSuccess", responseText);  
        AndroidNativeApp.updateTextView("Connected Successfully\n" + responseText);  
    }  
  
    public void onFailure(WLFailResponse response) {  
        String responseText = response.getResponseText();  
        Log.d("ConnectFail", responseText);  
        AndroidNativeApp.updateTextView("Connection Failure\n" + responseText);  
    }  
}
```

- The **response** object contains the response data.
- You can use its methods and properties to retrieve the required information.

## Receiving a procedure response (3 of 3)

- The sample for this training module can be found in the Getting Started page of the IBM Worklight documentation website at <http://www.ibm.com/mobile-docs>.
- Sample contains two projects:
  - The **NativeAPIForAndroid.zip** contains a Worklight Native API to be deployed to your Worklight server.
  - The **AndroidNativeApp.zip** contains native Android application that uses a Worklight native API library to communicate with a Worklight server.
- Make sure to update the **wlclient.properties** file in the AndroidNativeApp with the relevant server settings.



# Notices

- Permission for the use of these publications is granted subject to these terms and conditions.
- This information was developed for products and services offered in the U.S.A.
- IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.
- IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:
  - IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.
- For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:
  - Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan Ltd.  
1623-14, Shimotsuruma, Yamato-shi  
Kanagawa 242-8502 Japan
- **The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.**
- This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.
- Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.
- IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.
- Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:
  - IBM Corporation  
Dept F6, Bldg 1  
294 Route 100  
Somers NY 10589-3216  
USA

- Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.
- The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.
- Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

## COPYRIGHT LICENSE:

- This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.
- Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:
  - © (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp., enter the year or years. All rights reserved.

## Privacy Policy Considerations

- IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.
- Depending upon the configurations deployed, this Software Offering may use session cookies that collect session information (generated by the application server). These cookies contain no personally identifiable information and are required for session management. Additionally, persistent cookies may be randomly generated to recognize and manage anonymous users. These cookies also contain no personally identifiable information and are required.
- If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent. For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at <http://www.ibm.com/privacy>, and IBM's Online Privacy Statement at <http://www.ibm.com/privacy/details> the sections entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at <http://www.ibm.com/software/info/product-privacy>.

# Support and comments

- For the entire IBM Worklight documentation set, training material and online forums where you can post questions, see the IBM website at:
  - <http://www.ibm.com/mobile-docs>
- **Support**
  - Software Subscription and Support (also referred to as Software Maintenance) is included with licenses purchased through Passport Advantage and Passport Advantage Express. For additional information about the International Passport Advantage Agreement and the IBM International Passport Advantage Express Agreement, visit the Passport Advantage website at:
    - <http://www.ibm.com/software/passportadvantage>
  - If you have a Software Subscription and Support in effect, IBM provides you assistance for your routine, short duration installation and usage (how-to) questions, and code-related questions. For additional details, consult your IBM Software Support Handbook at:
    - <http://www.ibm.com/support/handbook>
- **Comments**
  - We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this document. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.
  - For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.
  - When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state.
  - Thank you for your support.
  - Submit your comments in the IBM Worklight forums at:
    - <https://www.ibm.com/developerworks/mobile/worklight/connect.html>
  - If you would like a response from IBM, please provide the following information:
    - Name
    - Address
    - Company or Organization
    - Phone No.
    - Email address

***Thank You***

