Java HTTP Adapter

Overview

Java adapters provide free reign over connectivity to your backend. It is therefore your responsibility to ensure best practices regarding performance and other implementation details.
This tutorial covers an example of a Java adapter that connects to an RSS feed by using a Java `HttpClient`.

Prerequisite: Make sure to read the Java Adapters (../) tutorial first.

RSSAdapterApplication

`RSSAdapterApplication` extends `MFPJAXRSApplication` and is a good place to trigger any initialization required by your application.

```
@Override
protected void init() throws Exception {
    RSSAdapterResource.init();
    logger.info("Adapter initialized!");
}
```

RSSAdapterResource

`RSSAdapterResource` is where we handle the requests to your adapter.

```
@Path("/")
public class RSSAdapterResource {
}
```

`@Path("/")` means that the resources will be available at the URL `http(s)://host:port/ProjectName/adapters/AdapterName/`.

HTTP Client

RSSAdapterResource

```
private static CloseableHttpClient client;
private static HttpHost host;

public static void init() {
    client = HttpClients.createDefault();
    host = new HttpHost("developer.ibm.com");
}
```

Because every request to your resource will create a new instance of `RSSAdapterResource`, it is important to reuse objects that may impact performance. In this example we made the Http client a `static` object and initialized it in a static `init()` method, which gets called by the `init()` of

`RSSAdapterApplication` as described above.

Procedure resource

RSSAdapterResource

```
@GET
@Produces("application/json")
public void get(@Context HttpServletResponse response, @QueryParam("tag") Stri
ng tag)
        throws ClientProtocolException, IOException, IllegalStateException, SA
XException {
    if(tag!=null && !tag.isEmpty()){
        execute(new HttpGet("/mobilefirstplatform/tag/"+ tag +"/feed"), respon
se);
    }
    else{
        execute(new HttpGet("/mobilefirstplatform/feed"), response);
    }

}
```

Our adapter exposes just one resource URL which allows to retrieve the RSS feed from the backend service.

- `@GET` means that this procedure only responds to `HTTP GET` requests.
- `@Produces("application/json")` specifies the Content Type of the response to send back. We chose to send the response as a `JSON` object to make it easier on the client-side.
- `@Context HttpServletResponse response` will be used to write to the response output stream. This enables us more granularity than returning a simple string.
- `@QueryParam("tag")` String tag enables the procedure to receive a parameter. The choice of `QueryParam` means the parameter is to be passed in the query (`/RSSAdapter/?tag=MobileFirst_Platform`). Other options include `@PathParam`, `@HeaderParam`, `@CookieParam`, `@FormParam`, etc.
- `throws ClientProtocolException, ...` means we are forwarding any exception back to the client. The client code is responsible for handling potential exceptions which will be received as `HTTP 500` errors. Another solution (more likely in production code) is to handle exceptions in your server Java code and decide what to send to the client based on the exact error.
- `execute(new HttpGet("/mobilefirstplatform/feed"), response)`. The actual HTTP request to the backend service is handled by another method defined later.

Depending if you pass a `tag` parameter, `execute` will retrieve a different build a different path and retrieve a different RSS file.

execute()

RSSAdapterResource

```
public void execute(HttpUriRequest req, HttpServletResponse resultResponse)
        throws ClientProtocolException, IOException,
        IllegalStateException, SAXException {
    HttpResponse RSSResponse = client.execute(host, req);
    ServletOutputStream os = resultResponse.getOutputStream();
    if (RSSResponse.getStatusLine().getStatusCode() == HttpStatus.SC_OK){
        resultResponse.addHeader("Content-Type", "application/json");
        String json = XML.toJson(RSSResponse.getEntity().getContent());
        os.write(json.getBytes(Charset.forName("UTF-8")));

    }else{
        resultResponse.setStatus(RSSResponse.getStatusLine().getStatusCode());
        RSSResponse.getEntity().getContent().close();
        os.write(RSSResponse.getStatusLine().getReasonPhrase().getBytes());
    }
    os.flush();
    os.close();
}
```

- `HttpResponse RSSResponse = client.execute(host, req)`. We use our static HTTP client to execute the HTTP request and store the response.
- `ServletOutputStream os = resultResponse.getOutputStream()`. This is the output stream to write a response to the client.
- `resultResponse.addHeader("Content-Type", "application/json")`. As mentioned before, we chose to send the response as JSON.
- `String json = XML.toJson(RSSResponse.getEntity().getContent())`. We used `org.apache.wink.json4j.utils.XML` to convert the XML RSS to a JSON string.
- `os.write(json.getBytes(Charset.forName("UTF-8")))` the resulting JSON string is written to the output stream.

The output stream is then `flushed` and `closed`.

If `RSSResponse` is not `200 OK`, we write the status code and reason in the response instead.

Results

The adapter should return the RSS feed converted to JSON.


```
{
   "rss": {
      "channel": {
         "description": "Develop, test, manage, and secure your mobile web, na
tive and hybrid apps",
         "generator": "http:\/\/wordpress.org\/?v=4.2.4",
         "item": [
            {
               "category": [
                  "Mobile",
                  "android",
                  "Mobile Quality Assurance",
                  "mobile_development",
                  "mobilefirst",
                  "xamarin"
```

```
                ],
                "commentRss": "https:\/\/developer.ibm.com\/mobilefirstplatform
\/2015\/09\/01\/integrating-mqa-into-xamarin-android-app\/feed\/",
                "comments": [
                   "https:\/\/developer.ibm.com\/mobilefirstplatform\/2015\/09\
/01\/integrating-mqa-into-xamarin-android-app\/#comments",
                   "0"
                ],
                "creator": "Vidyasagar MSC",
                "description": "<p>The post <a rel=\"nofollow\" href=\"https:\/
\/developer.ibm.com\/mobilefirstplatform\/2015\/09\/01\/integrating-mqa-into-x
amarin-android-app\/\">Integrating MQA into Xamarin.Android app<\/a> appeared
first on <a rel=\"nofollow\" href=\"https:\/\/developer.ibm.com\/mobilefirstpl
atform\">IBM MobileFirst Platform<\/a>.<\/p>",
                "encoded": "<p>It all startedÂ when I received an email seeking
help on using MQA or to be more precise integrating MQA into Xamarin based and
roid app. Before jumping into addressing the problem, let&#8217;s define MQA.<
\/p>\n<h4>What is MQA?<\/h4>\n<p>MQA stands for &#8220;Mobile Quality Assuranc
e&#8221; and is part of the IBM MobileFirst Platform.<\/p>\n<blockquote><p><em
><span style=\"line-height: 1.5\">IBM MQA provides line of business profession
als and development teams with insightful and streamlined quality feedback and
metrics from both pre-production and production, enabling them to prioritize a
nd take action to support a dynamic mobile app strategy.<\/span><\/em><\/p><\/
blockquote>\n<p>The Features of MQA are<\/p>\n<div style=\"width: 1058px\" cla
ss=\"wp-caption aligncenter\"><a href=\"http:\/\/vidyasagarmsc.com\/wp-content
\/uploads\/2015\/09\/MQA1.png\"><img class=\"size-full wp-image-65\" src=\"htt
p:\/\/vidyasagarmsc.com\/wp-content\/uploads\/2015\/09\/MQA1.png\" alt=\"Featu
res of Mobile Quality Assurance.\" width=\"1048\" height=\"350\" \/><\/a><p cl
ass=\"wp-caption-text\">Features of Mobile Quality Assurance.<\/p><\/div>\n<p>
<em><strong>Note<\/strong><\/em>: To understand more about MQA, visitÂ <a href
=\"http:\/\/www-03.ibm.com\/software\/products\/en\/ibm-mobilefirst-platform-q
uality-assurance\">IBM Mobile Quality Assurance<\/a><\/p>\n<p>So, by now we sh
ould be good with the first part of our blog title that is MQA. So, the next qu
estion is<\/p>\n<h4>What is Xamarin.Android?<\/h4>\n<p>Xamarin is a platform t
o create nativeÂ iOS, Android, Mac and Windows apps in C#.Â Xamarin.Android al
lows us to create native Android applications using the same UI controls we wo
uld in Java, except with the flexibility and elegance of a modern language (C#
).<\/p>\n<p>As we are good with the definitions, let&#8217;s address the probl
em.<\/p>\n<p><strong>What&#8217;s the problem in integrating MQA into Xamarin
Android app?<\/strong><\/p>\n<p>At the time of this blog post, the available M
QA SDKs are iOS native SDK, Android native SDK and Javascript Â SDK.<\/p>\n<p>
So, we have to find a workaround to address this use-case. The initial step is
to download the Android MQA SDK and see what&#8217;s provided. you can downloa
d it from <a href=\"http:\/\/www-01.ibm.com\/support\/knowledgecenter\/#!\/SSJ
ML5_6.0.0\/com.ibm.mqa.uau.saas.doc\/topics\/c_AndroidSDKsForDownload.html\">h
ere<\/a>. Once successfully downloaded and unzipped, we should see a jar file
namely <strong><em>MQA-Android-library-&lt;version number&gt;.jar<\/em>Â  <\/s
trong>under lib folder<strong>.<\/strong><\/p>\n<div style=\"width: 634px\" cl
ass=\"wp-caption aligncenter\"><a href=\"http:\/\/vidyasagarmsc.com\/wp-conten
t\/uploads\/2015\/09\/MQA2.png\"><img class=\"size-full wp-image-70\" src=\"ht
tp:\/\/vidyasagarmsc.com\/wp-content\/uploads\/2015\/09\/MQA2.png\" alt=\"MQA
Android SDK \" width=\"624\" height=\"440\" \/><\/a><p class=\"wp-caption-text
\">MQA Android SDK<\/p><\/div>\n<p>As Xamarin is C# based, What can we do with
this jar file?<\/p>\n<p>We haveÂ <strong>Xamarin bindings<\/strong> to our res
cue, which helps using in consuming .JARs from C#.<\/p>\n<p><strong><em>Note<\
/em>:<\/strong> Steps to consume MQA Android JAR in a Xamarin.Android app is m
entionedÂ <a href=\"https:\/\/developer.xamarin.com\/guides\/android\/advanced
```

_topics\/java_integration_overview\/binding_a_java_library_(.jar)\/\">here<\/a><\/p>\n<div style=\"width: 257px\" class=\"wp-caption aligncenter\"><a href=\"http:\/\/vidyasagarmsc.com\/wp-content\/uploads\/2015\/09\/MQA31.png\"><\/a><p class=\"wp-caption-text\">Xamarin binding project with MQA Android .JAR file<\/p><\/div>\n<p>The files of our interest here are MQA-Android-library-2.7.4.jar<\/strong> (Version number may vary) and Metadata.xml.<\/strong><\/p>\n<ul>\n<li>MQA-Android-library-2.7.4.jar file will have all the MQA related classes and methods required for us to start an Android MQA session.<\/li>\n<li>Metadata.xml- Allows changes to be made to the final API, such as changing the namespace of the generated binding.<\/em><\/li>\n<\/ul>\n<p>Based on the errors thrown while building the project, Metadata.xml in my case looks like this<\/p>\n<pre class=\"brush: xml; title: ; notranslate\"><metadata>\n <!--\n This sample removes the class: android.support.v4.content.AsyncTaskLoader.LoadTask:\n <remove-node path="\/api\/package[@name='android.support.v4.content']\/class[@name='AsyncTaskLoader.LoadTask']" \/>\n \n This sample removes the method: android.support.v4.content.CursorLoader.loadInBackground:\n <remove-node path="\/api\/package[@name='android.support.v4.content']\/class[@name='CursorLoader']\/method[@name='loadInBackground']" \/>\n -->\n\n <remove-node path="\/api\/package[@name='ext.com.google.inject.spi']\/class[@name='InjectionPoint.Factory.1']"\/>\n <remove-node path="\/api\/package[@name='ext.com.google.inject.spi']\/class[@name='InjectionPoint.Factory.2']"\/>\n <remove-node path="\/api\/package[@name='com.applause.android.log']\/interface[@name='LoggerInterface']"\/>\n <remove-node path="\/api\/package[@name='ext.com.google.inject.internal']"\/>\n <remove-node path="\/api\/package[@name='ext.com.google.inject.matcher']"\/>\n <remove-node path="\/api\/package[@name='com.applause.android.util']\/class[@name='AbstractRequest']"\/>\n <remove-node path="\/api\/package[@name='ext.com.google.inject.spi']\/class[@name='Elements.RecordingBinder']\/method[@name='bind' and count(parameter)=1 and parameter[1][@type='ext.com.google.inject.Key']]"\/>\n\n<attr path="\/api\/package[@name='com.applause.android.messages']\/class[@name='Message']\/field[@name='message']" name="managedName">Message1<\/attr>\n<attr path="\/api\/package[@name='com.applause.android.log']" name="managedName">log<\/attr>\n<\/metadata>\n\n<\/pre>\n<p>Once all the errors are fixed and your binding project builds successfully, add a new Xamarin Android project (if you haven’t added yet). Now, add MQA binding project reference in our Xamarin android app. Note:<\/strong><\/em> Both your binding project and Xamarin.Android project should be of same target framework.Â <\/strong>You can verify this by right clicking on your project -> Options -> General.<\/p>\n<div id=\"attachment_83\" style=\"width: 270px\" class=\"wp-caption aligncenter\"><a href=\"http:\/\/vidyasagarmsc.com\/wp-content\/uploads\/2015\/09\/MQA5.png\"><\/a><p class=\"wp-caption-text\">Xamarin Android project with added reference to MQA<\/p><\/div>\n<p>Now, let’s start MQA android session in our Count.Android app. Before doing this, we should create a MQA service on IBM Bluemix. You can follow the instructions mentioned atÂ <a href=\"https:\/\/www.ng.bluemix.net\/docs\/#services\/MobileQualityAssurance\/index.html#MobileQualityAssurance\">Getting started with Mobile Quality Assurance- Bluemix<\/a>Â or watch this video.<\/p>\n<p><iframe class='youtube-player' type='text\/html' width='980' height='582' src='https:\/\/www.youtube.com\/embed\/zHRfGatcKPM?version=3&rel=1&fs=1&showsearch=0&showinfo=1&#03

```
8;iv_load_policy=1&#038;wmode=transparent' frameborder='0' allowfullscreen='tr
ue'><\/iframe><\/span><\/p>\n<p>Starting aÂ <span class=\"ph\"><span id=\"d608
7e24\" class=\"ph\">Mobile Quality Assurance<\/span><\/span>Â session with the
Android SDK entails three steps. First, build a configuration to define howÂ <
span class=\"ph\"><span id=\"d6087e24-d6083e11a1310\" class=\"ph\">Mobile Qual
ity Assurance<\/span><\/span>Â works with your app. Second, start the session
itself. Third, add tracking to your activities. Open <strong>MainActivity.cs<\
/strong> file (Android Project) and paste the code provided below<\/p>\n<pre c
lass=\"brush: csharp; title: ; notranslate\">using System;\n\nusing Android.Ap
p;\nusing Android.Content;\nusing Android.Runtime;\nusing Android.Views;\nusin
g Android.Widget;\nusing Android.OS;\n\/\/MQA references\nusing Com.Ibm.Mqa.Co
nfig;\nusing Com.Ibm.Mqa;\n\n\nnamespace Count.Android\n{\n\t[Activity (Label
= &quot;Count.Android&quot;, MainLauncher = true, Icon = &quot;@drawable\/icon
&quot;)]\n\tpublic class MainActivity : Activity\n\t{\n\t\tint count = 1;\n\t\
t\/\/Use your own generated APP KEY\n\t\tconst string APP_KEY=&quot;1g59b7d884
f9fdf5426162e5cb1f87a700648bce4fg0g1g379e0d3a&quot;;\n\t\tprotected override v
oid OnCreate (Bundle bundle)\n\t\t{\n\t\t\tbase.OnCreate (bundle);\n\t\t\t\/\/
MQA Android session configuration \n\t\t\tConfiguration configuration = new Co
nfiguration.Builder(this)\n\t\t\t\t.WithAPIKey(APP_KEY) \/\/Provides the quali
ty assurance application APP_KEY\n\t\t\t\t.WithMode(MQA.Mode.Qa) \/\/Selects t
he quality assurance application mode\n\t\t\t\t.WithReportOnShakeEnabled(true)
\/\/Enables shake report trigger\n\t\t\t\t.WithDefaultUser(&quot;default_user@
email.com&quot;) \/\/Sets a default user and user selection\n\t\t\t\t.Build();
\n\n\t\t\t\/\/Starting MQA Android Session\n\t\tMQA.StartNewSession (this, c
onfiguration);\n\t\t\t\/\/ Set our view from the &quot;main&quot; layout resou
rce\n\t\tSetContentView (Resource.Layout.Main);\n\n\t\t\t\/\/ Get our button
from the layout resource,\n\t\t\t\/\/ and attach an event to it\n\t\tButton
button = FindViewById&lt;Button&gt; (Resource.Id.myButton);\n\t\t\t\n\t\tbut
ton.Click += delegate {\n\t\t\tbutton.Text = string.Format (&quot;{0} clicks
!&quot;, count++);\n\t\t};\n\t}\n\t}\n}\n\n\n\n<\/pre>\n<p>Now, MQA is int
egrated into Xamarin.Android app and we are good to go.<\/p>\n<p>What we have
implemented above is just a drop in the Ocean of MQA, to know more about MQA a
nd its features &#8211; VisitÂ <a href=\"http:\/\/www-01.ibm.com\/support\/kno
wledgecenter\/?lang=en#!\/SSJML5_6.0.0\/com.ibm.mqa.uau.saas.doc\/mqa600saas_w
elcome.html\" target=\"_blank\">MQA Knowledge Centre<\/a><\/p>\n<p>Happy Codin
g !!!<\/p>\n<p>The post <a rel=\"nofollow\" href=\"https:\/\/developer.ibm.com
\/mobilefirstplatform\/2015\/09\/01\/integrating-mqa-into-xamarin-android-app\
/\">Integrating MQA into Xamarin.Android app<\/a> appeared first on <a rel=\"n
ofollow\" href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\">IBM Mobile
First Platform<\/a>.<\/p>",
               "guid": {
                  "content": "https:\/\/developer.ibm.com\/mobilefirstplatform
\/?p=16964",
                  "isPermaLink": "false"
               },
               "link": "https:\/\/developer.ibm.com\/mobilefirstplatform\/2015
\/09\/01\/integrating-mqa-into-xamarin-android-app\/",
               "pubDate": "Tue, 01 Sep 2015 20:27:07 +0000",
               "title": "Integrating MQA into Xamarin.Android app"
            },
            {
               "category": [
                  "Uncategorized",
                  "MobileFirst_Platform"
               ],
               "commentRss": "https:\/\/developer.ibm.com\/mobilefirstplatform
\/2015\/08\/19\/try-on-bluemix-and-buy-mfp\/feed\/",
```

\/2015\/08\/19\/try-on-bluemix-and-buy-mfp\/feed\/",
"comments": [
"https:\/\/developer.ibm.com\/mobilefirstplatform\/2015\/08\/19\/try-on-bluemix-and-buy-mfp\/#comments",
"0"
],
"creator": "ChethanKumar",
"description": "<p>The post <a rel=\"nofollow\" href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/2015\/08\/19\/try-on-bluemix-and-buy-mfp\/\">Try on Bluemix and migrate to on-prem MobileFirst Platform<\/a> appeared first on <a rel=\"nofollow\" href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\">IBM MobileFirst Platform<\/a>.<\/p>",
"encoded": "<p>Contributed By : Chethan Kumar SN (chethankumar.sn@in.ibm.com) and Vittal Pai (vittalpai@in.ibm.com)<\/p>\n<p>With the release of MobileFirst Platform v7.1, one can now migrate any existing iOS app built for MobileServices on Bluemix to MobileFirst Platform with just a handful of simple steps.<\/p>\n<p>To elucidate the process, lets look at how to migrate a simple Bluemix iOS app.<\/p>\n<p>To migrate an existing iOS app built for MobileServices on Bluemix to run on MobileFirst Platform, follow the steps below.<\/p>\n<ul>\n
<a href=\"#migrateexisting\">Existing Bluemix Server Application<\/a><\/li>\n
<a href=\"#migrateblu\">Existing Bluemix Client Application<\/a><\/li>\n
<a href=\"#configureclient\">Migration of Client Application<\/a><\/li>\n
<a href=\"#migratemfp\">Migration of JAX-RS Application to JAVA Adapter<\/a><\/li>\n
<a href=\"#configoauth\">Configuring Custom-OAuth<\/a><\/li>\n
<a href=\"#configurepush\">Configuring Push Capability<\/a><\/li>\n
<a href=\"#sample\">Sample and Source Code<\/a><\/li>\n<\/ul>\n<h2 id=\"migrateexisting\">Existing Bluemix Server Application<\/h2>\n<p>The Bluemix app has the following functionality:<\/p>\n<ul>\n<li>On the client side, the application stores a list of items and provides a way to add more items to the list. Each item can able to store Name, Store, Price and image of the product. The App’s are protected by Custom Authenticator via AMA security service provided by bluemix.<\/li>\n<li>On the server side, the App contains a JAX-RS class to store and manipulate the data. It also contains the server side AMA security implementation.<\/li>\n<\/ul>\n<p>On BlueMix we have application with the following configuration:<\/p>\n<ul>\n<li>Liberty Runtime : which used to run JAX-RS application on Bluemix<\/li>\n<li>Advance Mobile Access service : which gives mobile application security and monitoring functionality<\/li>\n<li>Push Service for iOS 8 : which provides the capability to use iOS Push features<\/li>\n<\/ul>\n<h3> Liberty Runtime <\/h3>\n<ul>\n<li>Liberty contains two projects with JAX-RS service (i.e Custom-oauth-java for Custom Authentication and LocalstoreAdapter for storing items). The service include the protected resource and the custom identity provider code. The liberty server is configured with TAI.\n<\/li>\n<li>Trust Association Interface (TAI) is a service provider API that enables the integration of third-party security services with a Liberty profile server. For more info on TAI : <a href=\"http:\/\/www-01.ibm.com\/support\/knowledgecenter\/was_beta_liberty\/com.ibm.websphere.wlp.nd.multiplatform.doc\/ae\/twlp_dev_custom_tai.html\" target=\"_blank\">click here<\/a>\n<\/li>\n<li>The custom identity provider authenticates a user by sending challenges to the client. However, custom identity providers do not communicate directly with clients. They send challenges and receive responses to the challenges by means of the Advanced Mobile Access service. When a custom identity provider successfully authenticates the user, it provides the user identity information to Advanced Mobile Access. For more information on custom authentication refer bluemix documentation : <a href=\"https:\/\/www.ng.bluemix.net\/docs\/services\/mobileaccess\/security\/id_provs\/index-gentopic2.html#custom_id_prov\" target=\"_blank\">click here<\/a>\n<p>The custom identity provider code is defined by two http API:<\/p>\n<pre class=\"brush: plain; title: ; notranslate\">\/star

tAutorization<\/pre>\n<p> and\n<pre class=\"brush: plain; title: ; notranslate
\">\/handleChallengeAnswer<\/pre>\n<pre class=\"brush: java; title: ; notransl
ate\"> @POST\n\t@Consumes ("application\/json")\n\t@Path("\/{te
nantId}\/customAuthRealm_3\/startAuthorization")\n\t@Produces(MediaType.A
PPLICATION_JSON)\n\tpublic JSONObject startAuthorization(String payload,\n\t\t
\t@PathParam("tenantId") String deviceId,\n\t\t\t\t@PathParam("re
almName") String realmName) throws Exception {\n\t\tJSONObject returnJson
= (JSONObject) JSON.parse(CHALLENGE_JSON);\n\t\treturn returnJson;\n\t}\n\t\n\
t@POST\n\t@Consumes ("application\/json")\n\t@Path("\/{tenantId
}\/customAuthRealm_3\/handleChallengeAnswer")\n\t@Produces(MediaType.APPL
ICATION_JSON)\n\tpublic JSONObject handleChllengeAnswer(String payload,\n\t\t\
t@PathParam("tenantId") String deviceId,\n\t\t\t\t@PathParam("rea
lmName") String realmName) throws Exception {\n\t\t\n\t\tJSONObject userS
toreJson = (JSONObject) JSON.parse(USER_STORE_JSON);\n\t\tJSONObject failedRes
ponseJson = (JSONObject) JSON.parse(FAILURE_JSON);\n\t\t\n\t\tif(payload == nu
ll || payload.isEmpty()) {\n\t\t\treturn failedResponseJson;\n\t\t}\n\t\tJSONO
bject payloadJson = (JSONObject) JSON.parse(payload);\n\t\tJSONObject challeng
eAnswer = (JSONObject) payloadJson.get("challengeAnswer");\n\t\t\n\t
\tif (challengeAnswer == null) {\n\t\t\treturn failedResponseJson;\n\t\t}\n\t
\t\n\t\tString userName = (String) challengeAnswer.get("userName");\
n\t\tString password = (String) challengeAnswer.get("password");\n\t
\t\n\t\tif (userName == null || userName.isEmpty() || password == null || passw
ord.isEmpty()) {\n\t\t\treturn failedResponseJson;\n\t\t}\n\t\t\n\t\tif (userS
toreJson.containsKey(userName)) {\t\n\t\t\tJSONObject userInfoJson = (JSONObje
ct) userStoreJson.get(userName);\n\t\t\tString userPassword = (String) userInf
oJson.get("password");\n\t\t\tString userDisplayName = (String) user
InfoJson.get("displayName");\n\t\t\t\n\t\t\tif (password.equals(user
Password)) {\n\t\t\t\tJSONObject returnJson = new JSONObject();\n\t\t\t\tJSONO
bject userIdentityJson = new JSONObject();\n\t\t\t\tuserIdentityJson.put(&quot
;userName", userName);\n\t\t\t\tuserIdentityJson.put("displayName&qu
ot;, userDisplayName);\n\t\t\t\t\n\t\t\t\treturnJson.put("status", &
quot;success");\n\t\t\treturnJson.put("userIdentity", userIde
ntityJson);\n\t\t\treturn returnJson;\n\t\t}\t\t\n\t\t}\n\t\t\n\treturn failedResponseJson;\n\t}\n<\/pre>\n<p>The Localstore adapter contains few ht
tp API’s to perform some basic operations like Add, Update, Create and D
elete in client application.<\/p>\n<pre class=\"brush: java; title: ; notransl
ate\"> @GET\n\t@Path("\/getAllItems")\n\tpublic String getAllItems()
throws IOException{\n\t\tinit();\n\t\tJsonArray jsonArray = new JsonArray();\n
\t\tfor(Object key : props.keySet()){\n\t\t\tjsonArray.add(parser.parse(props.
getProperty((String) key)).getAsJsonObject());\n\t\t}\n\t\treturn jsonArray.to
String();\n\t}\n\n\t@PUT\n\t@Path("\/addItem")\n\tpublic void addIte
m(String itemJson) \n\t\t\tthrows IOException, URISyntaxException{\n\t\ttry{\n
\t\t\tinit();\n\t\t\tint newKey = props.keySet().size()+1;\n\t\t\tprops.put(St
ring.valueOf(newKey), itemJson);\n\t\t\tURL url = this.getClass().getClassLoad
er().getResource("data.properties"); \n\t\t\tFile file = new File(ur
l.toURI().getPath());\n\t\t\tFileOutputStream foStream = new FileOutputStream(
file);\n\t\t\tprops.store(foStream, "saving new item");\n\t\t\tfoStr
eam.close();\n\n\t\t}catch(IOException ioe){\n\t\t\tioe.printStackTrace();\n\t
\t}\n\n\t}\n\n\t@POST\n\t@Path("\/addAllItems")\n\tpublic String add
AllItems(String itemsJson) \n\t\t\tthrows URISyntaxException, IOException{\n\
t\ttry{\n\t\t\tinit();\n\t\t\tclearAllData();\n\t\t\tJsonArray jsonArr = parse
r.parse(itemsJson).getAsJsonArray();\n\t\t\tfor(int i=0;i<jsonArr.size(
);i++){\n\t\t\t\tprops.put(String.valueOf(i+1), jsonArr.get(i).toString());\n\
t\t}\n\t\t\tURL url = this.getClass().getClassLoader().getResource(&quot
;data.properties"); \n\t\t\tFile file = new File(url.toURI().getPath(
));\n\t\t\tFileOutputStream foStream = new FileOutputStream(file);\n\t\t\tprop
s.store(foStream, "saving new item");\n\t\t\tfoStream.close(

);\n\t\t\treturn "true";\n\t\t}catch(IOException ioe){\n\t\t
\tioe.printStackTrace();\n\t\t}\n\t\treturn "false";\n\t}\n\n\t@DELE
TE\n\t@Path("\/clearAll")\n\tpublic String clearAllData() \n\t\t\tth
rows MissingConfigurationOptionException, URISyntaxException, IOException{\n\t
\t\tinit();\n\t\t\tprops.clear();\n\t\t\tSystem.out.println("Size : &quot
;+props.size());\n\t\t\tURL url = this.getClass().getClassLoader().getResource
("data.properties"); \n\t\t\tFile file = new File(url.toURI().getPat
h());\n\t\t\tFileOutputStream foStream = new FileOutputStream(file);\n\t\t\tpr
ops.store(foStream, "clearing all data");\n\t\t\tfoStream.close();\n
\t\t\treturn "cleared";\n\t}\n<\/pre>\n<\/li>\n Add TAI Extensio
n in the following path of server directory server\/usr\/extensions<br \/>\nTA
I Extension Link : Download the extension.zip from <a href=\"https:\/\/hub.jaz
z.net\/project\/chethan\/parkstore-bluemix-server\/overview\" target=\"_blank\
">here<\/a>\n<\/li>\n Add TAI Security constraint in web.xml file for both
the projects.\n<pre class=\"brush: xml; title: ; notranslate\"><security-co
nstraint>\n   \t<web-resource-collection>\n   \t   <web-resourc
e-name>LocalstoreApplication<\/web-resource-name>\n   \t   <url
-pattern>\/apps\/*<\/url-pattern>\n   \t<\/web-resource-collection
>\n   \t<auth-constraint>\n   <role-name>TAIUserRole
<\/role-name>\n   \t<\/auth-constraint>\n<\/security-constra
int>\n<security-role id="SecurityRole_TAIUserRole" >\n
<role-name>TAIUserRole<\/role-name>\n<\/security-role><\/pre
>\n<\/li>\n Add OAuthTai feature in server.xml\n<pre class=\"brush: plain;
title: ; notranslate\"><feature>usr:OAuthTai-1.0<\/feature><\/pre>
\n<\/li>\n Protect the Url’s using TAI by adding following code in s
erver.xml\n<pre class=\"brush: xml; title: ; notranslate\"> <usr_OAuthTAI
id="myOAuthTAI" realmName="imfRealm">\n\t\t<security
Constraint httpMethods="GET, POST" securedURLs="\/LocalstoreAda
pter\/*"\/>\n\t\t<securityConstraint httpMethods="GET, POST&qu
ot; securedURLs="\/custom-oauth-java\/*"\/>\n\t<\/usr_OAuthTAI
> \n\n   <webApplication id="custom-oauth-java" location=&quo
t;custom-oauth-java.war" name="custom-oauth-java">\n   &lt
;application-bnd>\n\t\t<security-role name="TAIUserRole">\n
\t\t\t<special-subject type="ALL_AUTHENTICATED_USERS"\/>\n\t\t
<\/security-role>\n\t<\/application-bnd> \n\t<\/webApplication&
gt;   \n\t <webApplication id="LocalstoreAdapter" location=&quot
;LocalstoreAdapter.war" name="LocalstoreAdapter">\n   &
lt;application-bnd>\n\t\t<security-role name="TAIUserRole">
\n\t\t\t<special-subject type="ALL_AUTHENTICATED_USERS"\/>\n\t
\t<\/security-role>\n\t<\/application-bnd> \n\t<\/webApplicatio
n><\/pre>\n<\/li>\n Specify the IMF Auth Url inside Server.env file in
liberty.\n<pre class=\"brush: xml; title: ; notranslate\">imfServiceUrl=https:
\/\/imf-authserver.ng.bluemix.net\/imf-authserver<\/pre>\n<\/li>\n Create
a server package which contains above two applications using following command
.\n<pre class=\"brush: plain; title: ; notranslate\">.\/server package ${serve
r_name} --include=usr<\/pre>\n<\/li>\n Push the newly created server packa
ge to bluemix using following command.\n<pre class=\"brush: plain; title: ; no
translate\">cf push ${app_name} -p ${path_to_server_package_zip}<\/pre>\n<\/li
>\n<\/ul>\n<h3>Advance Mobile Access service<\/h3>\n<ul>\n Bind the pushed
application to Advance Mobile Access Service.\n<p><a href=\"https:\/\/develope
r.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/Scre
en-Shot-2015-07-17-at-3.28.04-pm.png\"><img src=\"https:\/\/developer.ibm.com\
/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-20
15-07-17-at-3.28.04-pm-1024x346.png\" alt=\"Advance Mobile Access\" width=\"98
0\" height=\"331\" class=\"alignnone size-large wp-image-14882\" \/><\/a>\n<\/
li>\n Register your client application in AMA dashboard. For more info ref
er documentation : <a href=\"https:\/\/www.ng.bluemix.net\/docs\/services\/mob

ileaccess\/index.html\" target=\"_blank\">click here<\/a>\n<p><a href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-17-at-3.42.32-pm.png\"><\/a>\n<\/li>\n AMA provides Facebook, Google, or a custom identity provider to authenticate access to protected resources. Add Custom identity provider feature as it can be migrated to MFPF and specify the corresponding jax-rs custom authentication application url and realm name.<br \/>\n<a href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-17-at-4.03.21-pm.png\"><\/a>\n<\/li>\n Add the following code inside didFinishLaunchingWithOptions function in AppDelegate of client application which will register the realm and initialize connection with Bluemix Application.\n<pre class=\"brush: plain; title: ; notranslate\"> IMFClient.sharedInstance().registerAuthenticationDelegate(customAuthDelegate, forRealm: "customAuthRealm_3")\nIMFClient.sharedInstance().initializeWithBackendRoute("https:\/\/parkstore.mybluemix.net", backendGUID: "5e3ad88d-dd48-469d-b46f-2c4ad66b5345")<\/pre>\n<\/li>\n The following is the sample code to invoke the Rest url’s in client application.\n<pre class=\"brush: plain; title: ; notranslate\">var request: IMFResourceRequest = IMFResourceRequest(path: "https:\/\/parkstore.mybluemix.net\/LocalstoreAdapter\/apps\/5e3ad88d-dd48-469d-b46f-2c4ad66b5345\/localstore\/getAllItems", method: "GET")\n request.sendWithCompletionHandler { (wlResponse:IMFResponse!, err:NSError!) -> Void in<\/pre>\n<\/li>\n<\/ul>\n<h3>Push Service for iOS 8<\/h3>\n<ul>\n Bind the application with Push Service for iOS 8<br \/>\n<a href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-17-at-4.07.01-pm.png\"><\/a>\n<\/li>\n Configure Apple Push Notification service (APNs) which requires Apple Developer Account and Generate pl2 certificates. Documentation link : <a href=\"https:\/\/www.ng.bluemix.net\/docs\/services\/mobilepush\/index.html#certificates\" target=\"_blank\">click here<\/a>\n<\/li>\n Upload the generated pl2 certificate in Push service dashboard\n<p><a href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-12-at-6.47.14-pm.png\"><\/a>\n<\/li>\n<li>Add the following code inside didFinishLaunchingWithOptions function in AppDelegate of client application which will register notifications in client app.\n<pre class=\"brush: plain; title: ; notranslate\"> let notificationTypes: UIUserNotificationType = UIUserNotificationType.Badge | UIUserNotificationType.Alert | UIUserNotificationType.Sound\n let notificationSettings: UIUserNotificationSettings = UIUserNotificationSettings(forTypes: notificationTypes, categories: nil)\n \n application.registerUserNotificationSettings(notificationSettings)\n application.registerForRemoteNotifications()<\/pre>\n<\/li>\n<li>Add the following code inside didRegisterForRemoteNotificationsWithDeviceToken function in AppDelegate of client application which will register pushclient and subscribe to tag in client app.\n<pre class=\"brush: plain; title: ; notranslate\">IMFPushClient.sharedInstance().registerDeviceToken(deviceToken, completionHandler: { (respons

sharedInstance().registerDeviceToken(deviceToken, completionHandler: { (respons
e, error) -> Void in\n   if error != nil {\n   printl
n("Error during device registration \\(error.description)")\n
}\n   else {\n   println("Response during device re
gistration json: \\(response.responseJson.description)")\n
var tags = ["parkstore"]\n   IMFPushClient.sharedInstan
ce().subscribeToTags(tags, completionHandler: { (response:IMFResponse!, err:NS
Error!) -> Void in\n   if err != nil {\n
println("There was an error while subscribing to tag")\n
}else{\n   println("Successfully subscribe to tag pa
rkstore")\n   }\n   })\n   }<\/pr
e>\n<\/li>\n<li>Add the following function inside Appdelegate which triggers w
hen push notification arrived in client app.\n<pre class=\"brush: plain; title
: ; notranslate\">func application(application: UIApplication, didReceiveRemot
eNotification userInfo: [NSObject : AnyObject]) {\n   println("Got r
emote Notification. Data : \\(userInfo.description)")\n   let info =
userInfo as NSDictionary\n   let data = info.objectForKey("aps"
)?.objectForKey("alert") as! NSDictionary\n   let userData = da
ta.objectForKey("body") as! String\n   let alertView = UIAlertV
iew(title: "WishList!", message: "\\(userData)", delegate:
nil, cancelButtonTitle: "OK")\n   alertView.show()\n   }\n}
<\/pre>\n<\/li>\n<\/ul>\n<h2 id=\"migrateblu\">Existing Bluemix Client Applica
tion<\/h2>\n<p>Add the following Code snippets to the existing Bluemix Client
Application and name the application with same name which you have registered
in Advance Mobile Access Dashboard.<\/p>\n<ul>\n Add the following code in
side didFinishLaunchingWithOptions function in AppDelegate of client applicati
on which will register the realm and initialize connection with Bluemix Applic
ation.\n<pre class=\"brush: plain; title: ; notranslate\"> IMFClient.sharedIns
tance().registerAuthenticationDelegate(customAuthDelegate, forRealm: "cus
tomAuthRealm_3")\nIMFClient.sharedInstance().initializeWithBackendRoute(&
quot;https:\/\/parkstore.mybluemix.net", backendGUID: "5e3ad88d-dd48
-469d-b46f-2c4ad66b5345")<\/pre>\n<\/li>\n The following is the sampl
e code to invoke the Rest url’s in client application.\n<pre class=\"bru
sh: plain; title: ; notranslate\">var request: IMFResourceRequest = IMFResourc
eRequest(path: "https:\/\/parkstore.mybluemix.net\/LocalstoreAdapter\/app
s\/5e3ad88d-dd48-469d-b46f-2c4ad66b5345\/localstore\/getAllItems", method
: "GET")\n   request.sendWithCompletionHandler { (wlResponse:IM
FResponse!, err:NSError!) -> Void in<\/pre>\n<\/li>\n<li>Add the following
code inside didFinishLaunchingWithOptions function in AppDelegate of client ap
plication which will register notifications in client app.\n<pre class=\"brush
: plain; title: ; notranslate\"> let notificationTypes: UIUserNotificationTyp
e = UIUserNotificationType.Badge | UIUserNotificationType.Alert | UIUserNotifi
cationType.Sound\n   let notificationSettings: UIUserNotificationSettings
= UIUserNotificationSettings(forTypes: notificationTypes, categories: nil)\n
\n   application.registerUserNotificationSettings(notificationSettings)\n
application.registerForRemoteNotifications()<\/pre>\n<\/li>\n<li>Add the follo
wing code inside didRegisterForRemoteNotificationsWithDeviceToken function in
AppDelegate of client application which will register pushclient and subscribe
to tag in client app.\n<pre class=\"brush: plain; title: ; notranslate\">IMFPu
shClient.sharedInstance().registerDeviceToken(deviceToken, completionHandler:
{ (response, error) -> Void in\n   if error != nil {\n
println("Error during device registration \\(error.description)")\n
}\n   else {\n   println("Response during device re
gistration json: \\(response.responseJson.description)")\n
var tags = ["parkstore"]\n   IMFPushClient.sharedInstan
ce().subscribeToTags(tags, completionHandler: { (response:IMFResponse!, err:NS
Error!) -> Void in\n   if err != nil {\n

println("There was an error while subscribing to tag")\n
}else{\n println("Successfully subscribe to tag pa
rkstore")\n }\n })\n }<\/pr
e>\n<\/li>\n<li>Add the following function inside Appdelegate which triggers w
hen push notification arrived in client app.\n<pre class=\"brush: plain; title
: ; notranslate\">func application(application: UIApplication, didReceiveRemot
eNotification userInfo: [NSObject : AnyObject]) {\n println("Got r
emote Notification. Data : \\(userInfo.description)")\n let info =
userInfo as NSDictionary\n let data = info.objectForKey("aps"
)?.objectForKey("alert") as! NSDictionary\n let userData = da
ta.objectForKey("body") as! String\n let alertView = UIAlertV
iew(title: "WishList!", message: "\\(userData)", delegate:
nil, cancelButtonTitle: "OK")\n alertView.show()\n }\n}
<\/pre>\n<\/li>\n<li>The following are the screenshots of client application.<
br \/>\n<a href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content
\/uploads\/sites\/32\/2015\/07\/IMG_0020.jpg\"><img src=\"https:\/\/developer.
ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/IMG_00
20-169x300.jpg\" alt=\"IMG_0020\" width=\"169\" height=\"300\" class=\"alignno
ne size-medium wp-image-14917\" \/><\/a><a href=\"https:\/\/developer.ibm.com\
/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/IMG_00211.jpg\
"><img src=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/upl
oads\/sites\/32\/2015\/07\/IMG_00211-169x300.jpg\" alt=\"IMG_0021\" width=\"16
9\" height=\"300\" class=\"alignnone size-medium wp-image-14918\" \/><\/a><a h
ref=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/s
ites\/32\/2015\/07\/IMG_0025.jpg\"><img src=\"https:\/\/developer.ibm.com\/mob
ilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/IMG_0025-169x300.j
pg\" alt=\"IMG_0025\" width=\"169\" height=\"300\" class=\"alignnone size-medi
um wp-image-14920\" \/><\/a><a href=\"https:\/\/developer.ibm.com\/mobilefirst
platform\/wp-content\/uploads\/sites\/32\/2015\/07\/IMG_0024.jpg\"><img src=\"
https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/
32\/2015\/07\/IMG_0024-169x300.jpg\" alt=\"IMG_0024\" width=\"169\" height=\"3
00\" class=\"alignnone size-medium wp-image-14919\" \/><\/a><a href=\"https:\/
\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015
\/07\/IMG_0026.jpg\"><img src=\"https:\/\/developer.ibm.com\/mobilefirstplatfo
rm\/wp-content\/uploads\/sites\/32\/2015\/07\/IMG_0026-169x300.jpg\" alt=\"IMG
_0026\" width=\"169\" height=\"300\" class=\"alignnone size-medium wp-image-14
921\" \/><\/a>\n<\/li>\n<\/ul>\n<h2>Migration to On-Prem<\/h2>\n<h3 id=\"confi
gureclient\">Migration of Client Application<\/h3>\n<p>Migration of Client App
lication includes following two steps<\/p>\n<li>Configuring Cocoapods<\/li>\n<
li>Client App Migration<\/li>\n<h3 id=\"cocoapods\">Configuring Cocoapods<\/h3
>\n<p>If CocoaPods has not been installed on a specific computer:<\/p>\n<ul>\n
Follow the “Getting Started” guide for CocoaPods installation:
http:\/\/guides.cocoapods.org\/using\/getting-started.html<\/li>\n<li>Open &#8
220;Terminal” at the installation location and run the “pod init&
#8221; command<\/li>\n<\/ul>\n<p>The following steps assume that the client app
lication is working with CocoPods. If not, follow this “Using CocoaPods&
#8221; documentation : <a href=\"http:\/\/guides.cocoapods.org\/using\/using-c
ocoapods.html\" target=\"_blank\">click here<\/a><\/p>\n<p>In both cases, the
instructions below explain how to edit the “Podfile” file.<\/p>\n<
ol>\n<li>Open the “Podfile” file located in the root of your XCode
project in a favourite text editor.<\/li>\n<li>Comment out or remove the exist
ing content.<\/li>\n<li>Add the following lines:\n<pre class=\"brush: plain; t
itle: ; notranslate\">source 'https:\/\/github.rtp.raleigh.ibm.com\/imflocalsd
ks\/imf-client-sdk-specs.git'\npod 'IMFCompatibility'<\/pre>\n<\/li>\n<li>Open
“Terminal” at the location of “Podfile”.<\/li>\n<li>Ve
rify that the XCode project is closed.<\/li>\n<li>Run the “pod install&#
8221; command.<\/li>\n<\/ol>\n<p>Open the [MyProject].xcworkspace file in XCod

e. This file is located side by side with [MyProject].xcodeproj.<br \/>\nAn us
ual CocoaPods-based project is managed as a workspace containing the applicati
on (the executable) and the library (all project dependencies brought by the C
ocoaPods manager).<\/p>\n<p>In Xcode’s Build Settings, search for &#8220
;Other Linker Flags” and insert ${inherited} (if -ObjC is defined in thi
s field, you can just delete it, since it is configured in the CocoaPod projec
t).<\/p>\n<h3>Client App Migration<\/h3>\n<ol>\n<li>Search for bluemix depende
ncy imports like\n<pre class=\"brush: plain; title: ; notranslate\">#import &l
t;IMFCore\/IMFCore.h>\n#import <IMFPush\/IMFPush.h><\/pre>\n<p>Replac
e the above imports with <\/p>\n<pre class=\"brush: plain; title: ; notranslat
e\">#import <IMFCompatibility\/IMFCompatibility.h><\/pre>\n<\/li>\n<li>L
ook for a call to the “initializeWithBackendRoute” method and repl
ace the route URL with your on-premise server URL. For example:\n<pre class=\"
brush: plain; title: ; notranslate\">IMFClient.sharedInstance().initializeWith
BackendRoute("https:\/\/parkstore.mybluemix.net", backendGUID: &quot
;5e3ad88d-dd48-469d-b46f-2c4ad66b5345"<\/pre>\n<p>should be replaced with
your on-premise MFP server URL<\/p>\n<pre class=\"brush: plain; title: ; notra
nslate\">IMFClient.sharedInstance().initializeWithBackendRoute("http:\/\/
localhost:10080\/ParkStoreMFP", backendGUID: "5e3ad88d-dd48-469d-b46
f-2c4ad66b5345"<\/pre>\n<p>Note, that backendGUID parameter is ignored an
d can be empty. Look for all instantiations of IMFResourceRequest class and up
date it<\/li>\n<li>Look for all instantiations of IMFResourceRequest class and
update the request URL with absolute or relative path to the resource. For exa
mple:\n<pre class=\"brush: plain; title: ; notranslate\">var request: IMFResou
rceRequest = IMFResourceRequest(path: "https:\/\/parkstore.mybluemix.net\
/LocalstoreAdapter\/apps\/5e3ad88d-dd48-469d-b46f-2c4ad66b5345\/localstore\/ge
tAllItems", method: "GET")<\/pre>\n<p>should be replaced with<\
/p>\n<pre class=\"brush: plain; title: ; notranslate\">var request: IMFResourc
eRequest = IMFResourceRequest(path: "http:\/\/localhost:10080\/ParkStoreM
FP\/adapters\/LocalstoreAdapter\/localstore\/getAllItems", method: "
GET")<\/pre>\n<\/li>\n<li>Add the following code inside didRegisterForRem
oteNotificationsWithDeviceToken function in Appdelegate of Client application.
\n<pre class=\"brush: plain; title: ; notranslate\"> WLPush.sharedInstance().t
okenFromClient = deviceToken.description<\/pre>\n<\/li>\n<li>All on-premise ap
plications require the “worklight.plist” file to be present in the
application resources. In the <code>IBMMobileFirstPlatformFoundationNativeSDK<
\/code> pod we supply a file named sample.worklight.plist<\/strong>.\n
\n<li>Locate the “sample.worklight.plist” file in the â€˜IBMMo
bileFirstPlatformFoundationNativeSDKâ€™ pod.<\/li>\n<li>Copy this file to the
parent (application) project and rename it to “worklight.plist”.<\
/li>\n<li>Edit the “worklight.plist” file by setting the “ap
plication id” key to the name of your application deployed to the on-prem
ise MFPF server<\/li>\n<\/ul>\n<\/li>\n<\/ol>\n<h3 id=\"migratemfp\">Migration
of JAX-RS Application to JAVA Adapter<\/h3>\n<ol>\n<li>To migrate JAX-RS appli
cation to on-prem (MobileFirst Foundation) server we need to do the following
steps for server:\n<p>   Create MobileFirst Project –> Create native
API app for iOS<br \/>\n   â€‹â€‹<br \/>\n<a href=\"https:\/\/developer.i
bm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-
Shot-2015-07-12-at-6.50.04-pm.png\"><img src=\"https:\/\/developer.ibm.com\/mo
bilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-
07-12-at-6.50.04-pm.png\" alt=\"Screen Shot 2015-07-12 at 6.50.04 pm\" width=\
"595\" height=\"596\" class=\"alignnone size-full wp-image-14817\" \/><\/a><\/
p>\n<p><a href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\
/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-12-at-6.51.13-pm.png\"><img
src=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/s
ites\/32\/2015\/07\/Screen-Shot-2015-07-12-at-6.51.13-pm.png\" alt=\"Screen Sh
ot 2015-07-12 at 6.51.13 pm\" width=\"598\" height=\"590\" class=\"alignnone s

ize-full wp-image-14818\" \/><\/a><\/p>\n<p><a href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-12-at-6.52.28-pm.png\"><\/a><\/li>\n<li>Add two adapters for Custom Authentication and Localstore and migrate the JAX-RS code as shown in the following example.<\/li>\n<\/ol>\n<p>Copy the JAX-RS BlueMix code and paste it in the newly created Localstore Java adapter JAX-RS file.<\/p>\n<p>Add and remove the following changes in your adapter code.<\/p>\n<ul>\n remove <code>\/{tenantId}\/<\/code><\/li>\n remove the <code>@PathParam -> PathParam(\"tenantId\") String deviceId<\/code> and <code>@PathParam(\"realmName\") String realmName<\/code><\/li>\n Add scope to the all http api resource <code>@OAuthSecurity (scope=\"customAuthRealm_3\")<\/code><\/li>\n<\/ul>\n<p>The code looks like the following<\/p>\n<pre class=\"brush: plain; title: ; notranslate\">\n\t@GET\n\t@OAuthSecurity (scope="customAuthRealm_3")\n\t@Path("\/getAllItems")\n\tpublic String getAllItems() throws MissingConfigurationOptionException{\n\t\tinit();\n\t\tJsonArray jsonArray = new JsonArray();\n\t\tfor(Object key : props.keySet()){\n\t\t\tjsonArray.add(parser.parse(props.getProperty((String) key)).getAsJsonObject());\n\t\t}\n\t\treturn jsonArray.toString();\n\t}\n\n\t@PUT\n\t@OAuthSecurity (scope="customAuthRealm_3")\n\t@Path("\/addItem")\n\tpublic void addItem(String itemJson) \n\t\t\tthrows MissingConfigurationOptionException, URISyntaxException, IOException{\n\t\ttry{\n\t\t\tinit();\n\t\t\tint newKey = props.keySet().size()+1;\n\t\t\tprops.put(String.valueOf(newKey), itemJson);\n\t\t\tURL url = this.getClass().getClassLoader().getResource("data.properties"); \n\t\t\tFile file = new File(url.toURI().getPath());\n\t\t\tFileOutputStream foStream = new FileOutputStream(file);\n\t\t\tprops.store(foStream, "saving new item");\n\t\t\tfoStream.close();\n\n\t}catch (IOException ioe){\n\t\tioe.printStackTrace();\n\t}\n\n\t}\n\n\t@POST\n\t@OAuthSecurity (scope="customAuthRealm_3")\n\t@Path("\/addAllItems")\n\tpublic String addAllItems(String itemsJson) \n\t\tthrows MissingConfigurationOptionException, URISyntaxException, IOException{\n\t\ttry{\n\t\tinit();\n\t\tclearAllData();\n\t\tJsonArray jsonArr = parser.parse(itemsJson).getAsJsonArray();\n\t\tfor(int i=0;i<jsonArr.size();i++){\n\t\tprops.put(String.valueOf(i+1), jsonArr.get(i).toString());\n\t}\n\t\tURL url = this.getClass().getClassLoader().getResource("data.properties"); \n\t\tFile file = new File(url.toURI().getPath());\n\t\tFileOutputStream foStream = new FileOutputStream(file);\n\t\tprops.store(foStream, "saving new item");\n\t\tfoStream.close();\n\t\treturn "true";\n\t}catch(IOException ioe){\n\t\tioe.printStackTrace();\n\t}\n\treturn "false";\n}\n\n@DELETE\n\t@OAuthSecurity(enabled=false)\n\t@Path("\/clearAll")\n\tpublic String clearAllData() \n\t\tthrows MissingConfigurationOptionException, URISyntaxException, IOException{\n\t\tinit();\n\t\tprops.clear();\n\t\tSystem.out.println("Size : "+props.size());\n\t\tURL url = this.getClass().getClassLoader().getResource("data.properties"); \n\t\tFile file = new File(url.toURI().getPath());\n\t\tFileOutputStream foStream = new FileOutputStream(file);\n\t\tprops.store(foStream, "clearing all data");\n\t\tfoStream.close();\n\t\treturn "cleared";\n\t}\n<\/pre>\n<h3 id=\"configoauth\">Configuring Custom-OAuth<\/h3>\n<ul>\n<li>Add realm with same name you had on BlueMix and login module to the authenticationConfig.xml.\n<pre class=\"brush: xml; title: ; notranslate\"><realm name="customAuthRealm_3" loginModule="customAuthLoginModule_3">\n<className>com.worklight.core.auth.ext.CustomIdentityAuthenticator<\/className>\t\n<parameter name="providerUrl" value="http:\/\/localhost:10080\/ParkStoreMFP\/adapters\/Customauth"\/>\n<\/realm>\

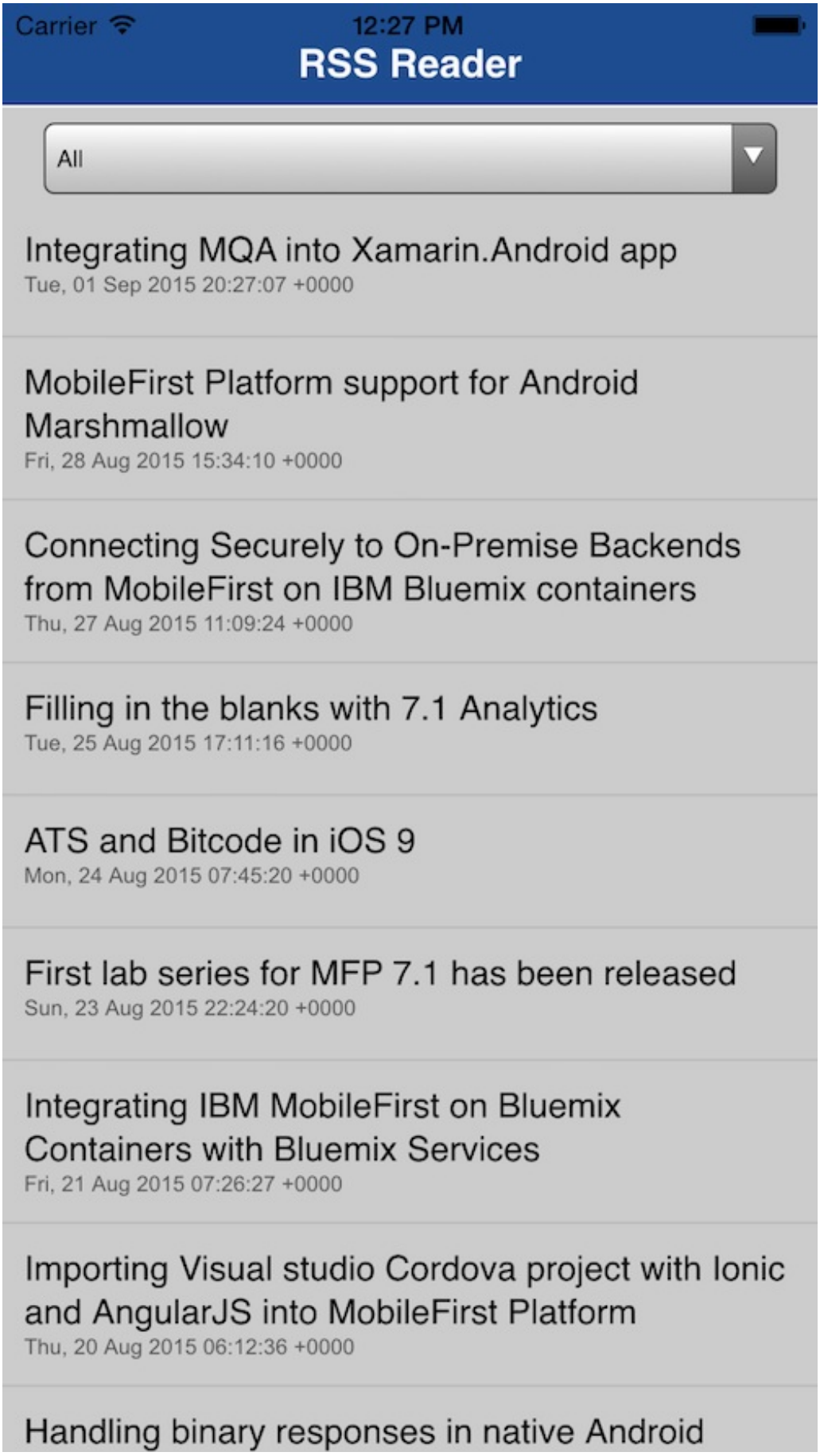
```
n\n&lt;loginModule name=&quot;customAuthLoginModule_3&quot; expirationInSecond
s=&quot;3600&quot;&gt;\n&lt;className&gt;com.worklight.core.auth.ext.CustomIde
ntityLoginModule&lt;\/className&gt;\n&lt;\/loginModule&gt;<\/pre>\n<\/li>\n<li
>Add Custom-oauth Realm in userIdentityRealms in Application Descriptor file o
f iOS Native API\n<pre class=\"brush: xml; title: ; notranslate\">&lt;userIden
tityRealms&gt;customAuthRealm_3&lt;\/userIdentityRealms&gt;<\/pre>\n<\/li>\n<\
/ul>\n<h3 id=\"configurepush\">Configuring Push Capability<\/h3>\n<ul>\n<li> A
dd apns p12 certificate which is generated from Apple Developer Account under
iOS Native API Folder\n<p><a href=\"https:\/\/developer.ibm.com\/mobilefirstpl
atform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-12-at-6.
58.03-pm.png\"><img src=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp
-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-12-at-6.58.03-pm.p
ng\" alt=\"Screen Shot 2015-07-12 at 6.58.03 pm\" width=\"286\" height=\"171\"
class=\"alignnone size-full wp-image-14820\" \/><\/a>\n<\/li>\n<li> Add Push c
onfiguration in Application Descriptor file of iOS Native API and include the
password of added apns certificate.\n<pre class=\"brush: xml; title: ; notrans
late\">&lt;pushSender password=&quot;password&quot;\/&gt;\n&lt;tags&gt;\n  &lt
;tag&gt;\n    &lt;name&gt;parkstore&lt;\/name&gt;\n  &lt;\/tag&gt;\n&lt;\/tags
&gt;<\/pre>\n<\/li>\n<li> Create HTTP Push Adapter with following function cod
e which will send the user push notification to the devices which is subscribe
d to tag &#8220;parkstore&#8221;.\n<pre class=\"brush: xml; title: ; notransla
te\">function sendTagNotification(notificationText) {\n    var notificationOpt
ions = {};\n    notificationOptions.message = {};\n    notificationOptions.tar
get = {};\n\n    notificationOptions.message.alert = notificationText;\n    no
tificationOptions.target.tagNames = [&quot;parkstore&quot;];\n\n    WL.Server.
sendMessage(&quot;ParkStoreMFP&quot;, notificationOptions);\n\n    return {\n
result : &quot;Notification sent to users subscribed to the tag parkstore.&quo
t;\n    };\n}<\/pre>\n<\/li>\n<\/ul>\n<p>By performing above steps one can eas
ily run iOS app built for Bluemix on MobileFirst Platform and following are th
e links to samples.<\/p>\n<h3 id=\"sample\">Sample and Source Code<\/h3>\n<p>B
luemix Server : <a href=\"https:\/\/hub.jazz.net\/git\/chethan\/parkstore-blue
mix-server\">Parkstore bluemix server<\/a><br \/>\nBluemix Client : <a href=\"
https:\/\/hub.jazz.net\/git\/chethan\/parkstore-bluemix\">Parkstore bluemix<\/
a><br \/>\nMFP Server     : <a href=\"https:\/\/hub.jazz.net\/git\/chethan\/pa
rkstore-mfp-server\">Parkstore mfp server<\/a><br \/>\nMFP Client     : <a hre
f=\"https:\/\/hub.jazz.net\/git\/chethan\/parkstore-mfp\">Parkstore mfp<\/a><\
/p>\n<p>The post <a rel=\"nofollow\" href=\"https:\/\/developer.ibm.com\/mobil
efirstplatform\/2015\/08\/19\/try-on-bluemix-and-buy-mfp\/\">Try on Bluemix an
d migrate to on-prem MobileFirst Platform<\/a> appeared first on <a rel=\"nofo
llow\" href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\">IBM MobileFir
st Platform<\/a>.<\/p>",
               "guid": {
                  "content": "https:\/\/developer.ibm.com\/mobilefirstplatform
\/?p=14769",
                  "isPermaLink": "false"
               },
               "link": "https:\/\/developer.ibm.com\/mobilefirstplatform\/2015
\/08\/19\/try-on-bluemix-and-buy-mfp\/",
               "pubDate": "Wed, 19 Aug 2015 10:36:51 +0000",
               "title": "Try on Bluemix and migrate to on-prem MobileFirst Pla
tform"
            }
         ],
         "language": "en-US",
         "lastBuildDate": "Tue, 08 Sep 2015 09:22:53 +0000",
         "link": [
            {
```

```
            {
               "href": "https:\/\/developer.ibm.com\/mobilefirstplatform\/feed
\/",
               "rel": "self",
               "type": "application\/rss+xml"
            },
            "https:\/\/developer.ibm.com\/mobilefirstplatform"
         ],
         "title": "IBM MobileFirst Platform",
         "updateFrequency": "1",
         "updatePeriod": "hourly"
      },
      "version": "2.0"
   }
}
```


Sample

Click to download (https://github.com/MobileFirst-Platform-Developer-Center/JavaAdapters) the MobileFirst project.

The attached sample includes an adapter called `RSSAdapter` and a hybrid application called `RSSReader` to test the adapter inside an application.


Carrier
12:27 PM
RSS Reader
All
Integrating MQA into Xamarin.Android app
Tue, 01 Sep 2015 20:27:07 +0000
MobileFirst Platform support for Android Marshmallow
Fri, 28 Aug 2015 15:34:10 +0000
Connecting Securely to On-Premise Backends from MobileFirst on IBM Bluemix containers
Thu, 27 Aug 2015 11:09:24 +0000
Filling in the blanks with 7.1 Analytics
Tue, 25 Aug 2015 17:11:16 +0000
ATS and Bitcode in iOS 9
Mon, 24 Aug 2015 07:45:20 +0000
First lab series for MFP 7.1 has been released
Sun, 23 Aug 2015 22:24:20 +0000
Integrating IBM MobileFirst on Bluemix Containers with Bluemix Services
Fri, 21 Aug 2015 07:26:27 +0000
Importing Visual studio Cordova project with Ionic and AngularJS into MobileFirst Platform
Thu, 20 Aug 2015 06:12:36 +0000
Handling binary responses in native Android