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") Strin
g tag)
        throws ClientProtocolException, IOException, IllegalStateException, SAX
Exception {
    if(tag!=null && !tag.isEmpty()){
        execute(new HttpGet("/mobilefirstplatform/tag/"+ tag +"/feed"), respons
e);
    }
    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, nat
ive 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-xama
rin-android-app\/\">Integrating MQA into Xamarin.Android app<\/a> appeared first
on <a rel=\"nofollow\" href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\
\">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 andro
id 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 Assurance&#82
21; 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 professionals and d
evelopment teams with insightful and streamlined quality feedback and metrics f
rom both pre-production and production, enabling them to prioritize and take act
ion to support a dynamic mobile app strategy.<\/span><\/em><\/p><\/blockquote>\
n<p>The Features of MQA are<\/p>\n<div style=\"width: 1058px\" class=\"wp-captio
n aligncenter\"><a href=\"http:\/\/vidyasagarmsc.com\/wp-content\/uploads\/2015
\/09\/MQA1.png\"><img class=\"size-full wp-image-65\" src=\"http:\/\/vidyasagarm
sc.com\/wp-content\/uploads\/2015\/09\/MQA1.png\" alt=\"Features of Mobile Qual
ity Assurance.\" width=\"1048\" height=\"350\" \/><\/a><p class=\"wp-caption-te
xt\">Features of Mobile Quality Assurance.<\/p><\/div>\n<p><em><strong>Note<\/s
trong><\/em>: To understand more about MQA, visitÂ <a href=\"http:\/\/www-03.ib
m.com\/software\/products\/en\/ibm-mobilefirst-platform-quality-assurance\">IBM
Mobile Quality Assurance<\/a><\/p>\n<p>So, by now we should be good with the fir
st part of our blog title that is MQA. So, the next question is<\/p>\n<h4>What i
s Xamarin.Android?<\/h4>\n<p>Xamarin is a platform to create nativeÂ iOS, Andro
id, Mac and Windows apps in C#.Â Xamarin.Android allows us to create native Andr
oid applications using the same UI controls we would in Java, except with the fl
exibility and elegance of a modern language (C#).<\/p>\n<p>As we are good with t
he definitions, let&#8217;s address the problem.<\/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 MQA SDKs are iOS native SDK, Android n
ative SDK and Javascript Â SDK.<\/p>\n<p>So, we have to find a workaround to add
ress this use-case. The initial step is to download the Android MQA SDK and see
what&#8217;s provided. you can download it from <a href=\"http:\/\/www-01.ibm.co
m\/support\/knowledgecenter\/#!\/SSJML5_6.0.0\/com.ibm.mqa.uau.saas.doc\/topics
\/c_AndroidSDKsForDownload.html\">here<\/a>. Once successfully downloaded and u
nzipped, we should see a jar file namely <strong><em>MQA-Android-library-&lt;ve
rsion number&gt;.jar<\/em>Â  <\/strong>under lib folder<strong>.<\/strong><\/p>
\n<div style=\"width: 634px\" class=\"wp-caption aligncenter\"><a href=\"http:\
/\/vidyasagarmsc.com\/wp-content\/uploads\/2015\/09\/MQA2.png\"><img class=\"si
ze-full wp-image-70\" src=\"http:\/\/vidyasagarmsc.com\/wp-content\/uploads\/201
5\/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# b
ased, What can we do with this jar file?<\/p>\n<p>We haveÂ <strong>Xamarin bindi
ngs<\/strong> to our rescue, which helps using in consuming .JARs from C#.<\/p>
\n<p><strong><em>Note<\/em>:<\/strong> Steps to consume MQA Android JAR in a Xa
marin.Android app is mentionedÂ <a href=\"https:\/\/developer.xamarin.com\/guide
s\/android\/advanced_topics\/java_integration_overview\/binding_a_java_library_
(.jar)\/\">here<\/a><\/p>\n<div style=\"width: 257px\" class=\"wp-caption align
```

center\"><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&iv_load_policy=1&wmode=transparent' frameborder='0' allowfullscreen='true'><\/iframe><\/span><\/p>\n<p>Starting aÂ <span class=\"ph\"><span id=\"d6087e24\" class=\"ph\">Mobile Quality Assurance<\/s

an class=\"ph\"><span id=\"d6087e24\" class=\"ph\">Mobile Quality Assurance<\/s
pan><\/span>Â session with the Android SDK entails three steps. First, build a
configuration to define howÂ <span class=\"ph\"><span id=\"d6087e24-d6083e11a13
10\" class=\"ph\">Mobile Quality Assurance<\/span><\/span>Â works with your app
. Second, start the session itself. Third, add tracking to your activities. Ope
n MainActivity.cs<\/strong> file (Android Project) and paste the code pr
ovided below<\/p>\n<pre class=\"brush: csharp; title: ; notranslate\">using Sys
tem;\n\nusing Android.App;\nusing Android.Content;\nusing Android.Runtime;\nusi
ng Android.Views;\nusing Android.Widget;\nusing Android.OS;\n\/\/MQA references
\nusing Com.Ibm.Mqa.Config;\nusing Com.Ibm.Mqa;\n\n\nnamespace Count.Android\n{
\n\t[Activity (Label = "Count.Android", MainLauncher = true, Icon = &
amp;quot;@drawable\/icon")]\n\tpublic class MainActivity : Activity\n\t{\n\t\t
int count = 1;\n\t\t\/\/Use your own generated APP KEY\n\t\tconst string APP_KE
Y="1g59b7d884f9fdf5426162e5cb1f87a700648bce4fg0g1g379e0d3a";\n\t\tpro
tected override void OnCreate (Bundle bundle)\n\t\t{\n\t\t\tbase.OnCreate (bund
le);\n\t\t\t\/\/MQA Android session configuration \n\t\t\tConfiguration configu
ration = new Configuration.Builder(this)\n\t\t\t\t.WithAPIKey(APP_KEY) \/\/Prov
ides the quality assurance application APP_KEY\n\t\t\t\t.WithMode(MQA.Mode.Qa)
\/\/Selects the quality assurance application mode\n\t\t\t\t.WithReportOnShakeE
nabled(true) \/\/Enables shake report trigger\n\t\t\t\t.WithDefaultUser("d
efault_user@email.com") \/\/Sets a default user and user selection\n\t\t\t
\t.Build();\n\n\t\t\t\/\/Starting MQA Android Session\n\t\t\tMQA.StartNewSessio
n (this, configuration);\n\t\t\t\/\/ Set our view from the "main" lay
out resource\n\t\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\t\tB
utton button = FindViewById<Button> (Resource.Id.myButton);\n\t\t\t\n\t\t
\tbutton.Click += delegate {\n\t\t\t\tbutton.Text = string.Format ("{0} cli
cks!", count++);\n\t\t\t};\n\t\t}\n\t}\n}\n\n\n\n<\/pre>\n<p>Now, MQA is in
tegrated into Xamarin.Android app and we are good to go.<\/p>\n<p>What we have i
mplemented above is just a drop in the Ocean of MQA, to know more about MQA and
its features – VisitÂ <a href=\"http:\/\/www-01.ibm.com\/support\/knowled
gecenter\/?lang=en#!\/SSJML5_6.0.0\/com.ibm.mqa.uau.saas.doc\/mqa600saas_welcom
e.html\" target=\"_blank\">MQA Knowledge Centre<\/a><\/p>\n<p>Happy Coding !!!<
\/p>\n<p>The post <a rel=\"nofollow\" href=\"https:\/\/developer.ibm.com\/mobil
efirstplatform\/2015\/09\/01\/integrating-mqa-into-xamarin-android-app\/\">Inte
grating MQA into Xamarin.Android app<\/a> appeared first on <a rel=\"nofollow\"
href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\">IBM MobileFirst Platf
orm<\/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\/",
               "comments": [
                  "https:\/\/developer.ibm.com\/mobilefirstplatform\/2015\/08\/
19\/try-on-bluemix-and-buy-mfp\/#comments"

```
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-m
fp\/\">Try on Bluemix and migrate to on-prem MobileFirst Platform<\/a> appeared
first on <a rel=\"nofollow\" href=\"https:\/\/developer.ibm.com\/mobilefirstplat
form\">IBM MobileFirst Platform<\/a>.<\/p>",
            "encoded": "<p>Contributed By : Chethan Kumar SN (chethankumar.s
n@in.ibm.com) and Vittal Pai (vittalpai@in.ibm.com)<\/p>\n<p>With the release o
f 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 MobileService
s on Bluemix to run on MobileFirst Platform, follow the steps below.<\/p>\n<ul>
\n<li><a href=\"#migrateexisting\">Existing Bluemix Server Application<\/a><\/l
i>\n<li><a href=\"#migrateblu\">Existing Bluemix Client Application<\/a><\/li>\
n<li><a href=\"#configureclient\">Migration of Client Application<\/a><\/li>\n<
li><a href=\"#migratemfp\">Migration of JAX-RS Application to JAVA Adapter<\/a>
<\/li>\n<li><a href=\"#configoauth\">Configuring Custom-OAuth<\/a><\/li>\n<li><
a href=\"#configurepush\">Configuring Push Capability<\/a><\/li>\n<li><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 followi
ng 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 ab
le to store Name, Store, Price and image of the product. The App&amp;#8217;s pro
tected by Custom Authenticator via AMA security service provided by bluemix.<\/l
i>\n<li>On the server side, the App contains a JAX-RS class to store and manipu
late the data. It also contains the server side AMA security implementation.<\/l
i>\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 Blue
mix<\/li>\n<li>Advance Mobile Access service : which gives mobile application s
ecurity 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 Ru
ntime <\/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 i
tems). The service include the protected resource and the custom identity provi
der code. The liberty server is configured with TAI.\n<\/li>\n<li>Trust Associa
tion 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 T
AI : <a href=\"http:\/\/www-01.ibm.com\/support\/knowledgecenter\/was_beta_libe
rty\/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 aut
henticates a user by sending challenges to the client. However, custom identity
providers do not communicate directly with clients. They send challenges and rec
eive 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; titl
e: ; notranslate\">\/startAutorization<\/pre>\n<p> and\n<pre class=\"brush: pla
in; title: ; notranslate\">\/handleChallengeAnswer<\/pre>\n<pre class=\"brush:
java; title: ; notranslate\"> @POST\n\t@Consumes (&amp;quot;application\/json&amp;quot;
)\n\t@Path(&amp;quot;\/{tenantId}\/customAuthRealm_3\/startAuthorization&amp;quot;)\n\t
```

@Produces(MediaType.APPLICATION_JSON)\n\tpublic JSONObject startAuthorization(S
tring payload,\n\t\t\t@PathParam("tenantId") String deviceId,\n\t\t\t
@PathParam("realmName") String realmName) throws Exception {\n\t\tJSO
NObject returnJson = (JSONObject) JSON.parse(CHALLENGE_JSON);\n\t\treturn retur
nJson;\n\t}\n\t\n\t@POST\n\t@Consumes ("application\/json")\n\t@Path(
"\/{tenantId}\/customAuthRealm_3\/handleChallengeAnswer")\n\t@Produce
s(MediaType.APPLICATION_JSON)\n\tpublic JSONObject handleChllengeAnswer(String
payload,\n\t\t\t@PathParam("tenantId") String deviceId,\n\t\t\t@PathP
aram("realmName") String realmName) throws Exception {\n\t\t\n\t\tJSO
NObject userStoreJson = (JSONObject) JSON.parse(USER_STORE_JSON);\n\t\tJSONObje
ct failedResponseJson = (JSONObject) JSON.parse(FAILURE_JSON);\n\t\t\n\t\tif(pa
yload == null || payload.isEmpty()) {\n\t\t\treturn failedResponseJson;\n\t\t}\n
\t\tJSONObject payloadJson = (JSONObject) JSON.parse(payload);\n\t\tJSONObject
challengeAnswer = (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&quo
t;);\n\t\tString password = (String) challengeAnswer.get("password");
\n\t\t\n\t\tif (userName == null || userName.isEmpty() || password == null || pa
ssword.isEmpty()) {\n\t\t\treturn failedResponseJson;\n\t\t}\n\t\t\n\t\tif (use
rStoreJson.containsKey(userName)) {\t\n\t\t\tJSONObject userInfoJson = (JSONObj
ect) userStoreJson.get(userName);\n\t\t\tString userPassword = (String) userInf
oJson.get("password");\n\t\t\tString userDisplayName = (String) userI
nfoJson.get("displayName");\n\t\t\t\n\t\t\tif (password.equals(userPa
ssword)) {\n\t\t\t\tJSONObject returnJson = new JSONObject();\n\t\t\t\tJSONObje
ct userIdentityJson = new JSONObject();\n\t\t\t\tuserIdentityJson.put("use
rName", userName);\n\t\t\t\tuserIdentityJson.put("displayName",
userDisplayName);\n\t\t\t\t\n\t\t\t\treturnJson.put("status", "s
uccess");\n\t\t\t\treturnJson.put("userIdentity", userIdentityJs
on);\n\t\t\t\treturn returnJson;\n\t\t\t}\t\t\t\n\t\t}\n\t\t\n\t\treturn failed
ResponseJson;\n\t}\n<\/pre>\n<p>The Localstore adapter contains few http API&#8
217;s to perform some basic operations like Add, Update, Create and Delete in c
lient application.<\/p>\n<pre class=\"brush: java; title: ; notranslate\"> @GET
\n\t@Path("\/getAllItems")\n\tpublic String getAllItems() throws IOEx
ception{\n\t\tinit();\n\t\tJsonArray jsonArray = new JsonArray();\n\t\tfor(Obje
ct 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@Path("\/addItem")\n\tpublic void addItem(String itemJs
on) \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(String.valueOf(newK
ey), itemJson);\n\t\t\tURL url = this.getClass().getClassLoader().getResource(&
amp;quot;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\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 addAllItems(String itemsJs
on) \n\t\t\tthrows URISyntaxException, IOException{\n\t\ttry{\n\t\t\tinit();\n\
t\t\tclearAllData();\n\t\t\tJsonArray jsonArr = parser.parse(itemsJson).getAsJs
onArray();\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\t}\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 fo
Stream = new FileOutputStream(file);\n\t\t\tprops.store(foStream, "sav
ing new item");\n\t\t\tfoStream.close();\n\t\t\treturn "true&
amp;amp;quot;;\n\t\t}catch(IOException ioe){\n\t\t\tioe.printStackTrace();\n\t\t}\n
\t\treturn "false";\n\t}\n\n\t@DELETE\n\t@Path("\/clearAll"
)\n\tpublic String clearAllData() \n\t\t\tthrows MissingConfigurationOptionExce
ption, URISyntaxException, IOException{\n\t\t\tinit();\n\t\t\tprops.clear();\n\

t\t\tSystem.out.println("Size : "+props.size());\n\t\t\tURL url = thi
s.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, "clearing all da
ta");\n\t\t\tfoStream.close();\n\t\t\treturn "cleared";\n\t}\n<\
/pre>\n<\/li>\n Add TAI Extension in the following path of server directory
server\/usr\/extensions<br \/>\nTAI Extension Link : Download the extension.zip
from <a href=\"https:\/\/hub.jazz.net\/project\/chethan\/parkstore-bluemix-serv
er\/overview\" target=\"_blank\">here<\/a>\n<\/li>\n Add TAI Security constr
aint in web.xml file for both the projects.\n<pre class=\"brush: xml; title: ;
notranslate\"><security-constraint>\n   \t<web-resource-collection>
\n   \t   <web-resource-name>LocalstoreApplication<\/web-resource-n
ame>\n   \t   <url-pattern>\/apps\/*<\/url-pattern>\n   \t&lt
;\/web-resource-collection>\n   \t<auth-constraint>\n   &lt
;role-name>TAIUserRole<\/role-name>\n   \t<\/auth-constraint>
\n<\/security-constraint>\n<security-role id="SecurityRole_TAIUse
rRole" >\n   <role-name>TAIUserRole<\/role-name>\n<\
/security-role><\/pre>\n<\/li>\n Add OAuthTai feature in server.xml\n<pr
e class=\"brush: plain; title: ; notranslate\"><feature>usr:OAuthTai-1.0&
lt;\/feature><\/pre>\n<\/li>\n Protect the Url’s using TAI by addi
ng following code in server.xml\n<pre class=\"brush: xml; title: ; notranslate\
"> <usr_OAuthTAI id="myOAuthTAI" realmName="imfRealm"&g
t;\n\t\t<securityConstraint httpMethods="GET, POST" securedURLs=&q
uot;\/LocalstoreAdapter\/*"\/>\n\t\t<securityConstraint httpMethods=
"GET, POST" securedURLs="\/custom-oauth-java\/*"\/>\n\t&
lt;\/usr_OAuthTAI> \n\n   <webApplication id="custom-oauth-java&quo
t; location="custom-oauth-java.war" name="custom-oauth-java&quot
;>\n   <application-bnd>\n\t\t<security-role name="TAIUserRo
le">\n\t\t\t<special-subject type="ALL_AUTHENTICATED_USERS&quot
;\/>\n\t\t<\/security-role>\n\t<\/application-bnd> \n\t<\/web
Application>   \n\t <webApplication id="LocalstoreAdapter" loc
ation="LocalstoreAdapter.war" name="LocalstoreAdapter">\
n   <application-bnd>\n\t\t<security-role name="TAIUserRole&
quot;>\n\t\t\t<special-subject type="ALL_AUTHENTICATED_USERS"\/
>\n\t\t<\/security-role>\n\t<\/application-bnd> \n\t<\/webApp
lication><\/pre>\n<\/li>\n Specify the IMF Auth Url inside Server.env fi
le in liberty.\n<pre class=\"brush: xml; title: ; notranslate\">imfServiceUrl=h
ttps:\/\/imf-authserver.ng.bluemix.net\/imf-authserver<\/pre>\n<\/li>\n Cre
ate a server package which contains above two applications using following comm
and.\n<pre class=\"brush: plain; title: ; notranslate\">.\/server package ${ser
ver_name} --include=usr<\/pre>\n<\/li>\n Push the newly created server pack
age 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 a
pplication to Advance Mobile Access Service.\n<p><a href=\"https:\/\/developer.
ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-
Shot-2015-07-17-at-3.28.04-pm.png\"><img src=\"https:\/\/developer.ibm.com\/mobi
lefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-1
7-at-3.28.04-pm-1024x346.png\" alt=\"Advance Mobile Access\" width=\"980\" heigh
t=\"331\" class=\"alignnone size-large wp-image-14882\" \/><\/a>\n<\/li>\n
Register your client application in AMA dashboard. For more info refer document
ation : <a href=\"https:\/\/www.ng.bluemix.net\/docs\/services\/mobileaccess\/i
ndex.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\"><img src=\"https:\/\/developer.ibm.com\/mob
ilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-
17-at-3.42.32-pm.png\" alt=\"AMA Client Registration\" width=\"935\" height=\"4

52\" class=\"alignnone size-full wp-image-14883\" \/><\/a>\n<\/li>\n AMA pro
vides 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\/mobilefirstp
latform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-17-at-4.0
3.21-pm.png\"><img src=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-c
ontent\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-17-at-4.03.21-pm.png\
" alt=\"Custom Auth AMA\" width=\"955\" height=\"375\" class=\"alignnone size-f
ull wp-image-14890\" \/><\/a>\n<\/li>\n Add the following code inside didFin
ishLaunchingWithOptions function in AppDelegate of client application which wil
l register the realm and initialize connection with Bluemix Application.\n<pre
class=\"brush: plain; title: ; notranslate\"> IMFClient.sharedInstance().regist
erAuthenticationDelegate(customAuthDelegate, forRealm: "customAuthRealm_3&
quot;)\nIMFClient.sharedInstance().initializeWithBackendRoute("https:\/\/p
arkstore.mybluemix.net", backendGUID: "5e3ad88d-dd48-469d-b46f-2c4ad6
6b5345")<\/pre>\n<\/li>\n The following is the sample code to invoke th
e 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!) -&g
t; 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:\/\/de
veloper.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\
/Screen-Shot-2015-07-17-at-4.07.01-pm.png\"><img src=\"https:\/\/developer.ibm.c
om\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-
2015-07-17-at-4.07.01-pm-1024x367.png\" alt=\"Push AMA\" width=\"980\" height=\"
351\" class=\"alignnone size-large wp-image-14891\" \/><\/a>\n<\/li>\n Conf
igure Apple Push Notification service (APNs) which requires Apple Developer Acc
ount 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\/mobilefirstpl
atform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-12-at-6.47
.14-pm.png\"><img src=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-con
tent\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-12-at-6.47.14-pm-1024x37
7.png\" alt=\"Push Service\" width=\"980\" height=\"361\" class=\"alignnone siz
e-large wp-image-14816\" \/><\/a>\n<\/li>\n<li>Add the following code inside di
dFinishLaunchingWithOptions function in AppDelegate of client application which
will register notifications in client app.\n<pre class=\"brush: plain; title: ;
notranslate\"> let notificationTypes: UIUserNotificationType = UIUserNotificat
ionType.Badge | UIUserNotificationType.Alert | UIUserNotificationType.Sound\n
let notificationSettings: UIUserNotificationSettings = UIUserNotificationSettin
gs(forTypes: notificationTypes, categories: nil)\n   \n   application
.registerUserNotificationSettings(notificationSettings)\n   application.re
gisterForRemoteNotifications()<\/pre>\n<\/li>\n<li>Add the following code insid
e 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.sharedIns
tance().registerDeviceToken(deviceToken, completionHandler: { (response, error)
-> Void in\n   if error != nil {\n   println("Err
or during device registration \\(error.description)")\n   }\n
else {\n   println("Response during device registration json:
\\(response.responseJson.description)")\n   var tags = [&quot
;parkstore"]\n   IMFPushClient.sharedInstance().subscribeToTa
gs(tags, completionHandler: { (response:IMFResponse!, err:NSError!) -> Void

gs(tags, completionHandler: { (response:IMFResponse!, err:NSError!) -> Void in\n if err != nil {\n println("There was an error while subscribing to tag")\n }else{\n println("Successfully subscribe to tag parkstore")\n }\n })\n }<\/pre>\n<\/li>\n<li>Add the following function inside Appdelegate which triggers when push notification arrived in client app.\n<pre class=\"brush: plain; title: ; notranslate\">func application(application: UIApplication, didReceiveRemoteNotification userInfo: [NSObject : AnyObject]) {\n println("Got remote Notification. Data : \\(userInfo.description)")\n let info = userInfo as NSDictionary\n let data = info.objectForKey("aps")?.objectForKey("alert") as! NSDictionary\n let userData = data.objectForKey("body") as! String\n let alertView = UIAlertView(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 Application<\/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 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<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: { (response, error) -> Void in\n if error != nil {\n println("Error during device registration \\(error.description)")\n }\n else {\n println("Response during device registration json: \\(response.responseJson.description)")\n var tags = ["parkstore"]\n IMFPushClient.sharedInstance().subscribeToTags(tags, completionHandler: { (response:IMFResponse!, err:NSError!) -> Void in\n if err != nil {\n println("There was an error while subscribing to tag")\n }else{\n println("Successfully subscribe to tag parkstore")\n }\n })\n }<\/pre>\n<\/li>\n<li>Add the following function inside Appdelegate which triggers when push notification arrived in client app.\n<pre class=\"brush: plain; title: ; notranslate\">func application(application: UIApplication, didReceiveRemoteNotification userInfo: [NSObject : AnyObject]) {\n p

rintln("Got remote Notification. Data : \\(userInfo.description)")\n
let info = userInfo as NSDictionary\n   let data = info.objectForKey(&quot
;aps")?.objectForKey("alert") as! NSDictionary\n   let user
Data = data.objectForKey("body") as! String\n   let alertView =
UIAlertView(title: "WishList!", message: "\\(userData)", de
legate: nil, cancelButtonTitle: "OK")\n   alertView.show()\n
}\n}<\/pre>\n<\/li>\n<li>The following are the screenshots of client applicatio
n.<br \/>\n<a href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-conte
nt\/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=\"alignnon
e size-medium wp-image-14917\" \/><\/a><a href=\"https:\/\/developer.ibm.com\/m
obilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/IMG_00211.jpg\"><
img src=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads
\/sites\/32\/2015\/07\/IMG_00211-169x300.jpg\" alt=\"IMG_0021\" width=\"169\" h
eight=\"300\" class=\"alignnone size-medium wp-image-14918\" \/><\/a><a href=\"
https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/3
2\/2015\/07\/IMG_0025.jpg\"><img src=\"https:\/\/developer.ibm.com\/mobilefirst
platform\/wp-content\/uploads\/sites\/32\/2015\/07\/IMG_0025-169x300.jpg\" alt=
\"IMG_0025\" width=\"169\" height=\"300\" class=\"alignnone size-medium wp-imag
e-14920\" \/><\/a><a href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/w
p-content\/uploads\/sites\/32\/2015\/07\/IMG_0024.jpg\"><img src=\"https:\/\/de
veloper.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\
/IMG_0024-169x300.jpg\" alt=\"IMG_0024\" width=\"169\" height=\"300\" class=\"a
lignnone size-medium wp-image-14919\" \/><\/a><a href=\"https:\/\/developer.ibm
.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\/07\/IMG_0026.j
pg\"><img src=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/u
ploads\/sites\/32\/2015\/07\/IMG_0026-169x300.jpg\" alt=\"IMG_0026\" width=\"16
9\" height=\"300\" class=\"alignnone size-medium wp-image-14921\" \/><\/a>\n<\/
li>\n<\/ul>\n<h2>Migration to On-Prem<\/h2>\n<h3 id=\"configureclient\">Migratio
n of Client Application<\/h3>\n<p>Migration of Client Application includes foll
owing 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 n
ot been installed on a specific computer:<\/p>\n<ul>\n<li>Follow the “Get
ting Started” guide for CocoaPods installation: http:\/\/guides.cocoapods.
org\/using\/getting-started.html<\/li>\n<li>Open “Terminal” at the
installation location and run the “pod init” command<\/li>\n<\/ul>\
n<p>The following steps assume that the client application is working with Coco
Pods. If not, follow this “Using CocoaPods” documentation : <a href
=\"http:\/\/guides.cocoapods.org\/using\/using-cocoapods.html\" target=\"_blank
\">click here<\/a><\/p>\n<p>In both cases, the instructions below explain how t
o edit the “Podfile” file.<\/p>\n<ol>\n<li>Open the “Podfile&
amp;#8221; file located in the root of your XCode project in a favourite text edito
r.<\/li>\n<li>Comment out or remove the existing content.<\/li>\n<li>Add the fol
lowing lines:\n<pre class=\"brush: plain; title: ; notranslate\">source 'https:
\/\/github.rtp.raleigh.ibm.com\/imflocalsdks\/imf-client-sdk-specs.git'\npod 'I
MFCompatibility'<\/pre>\n<\/li>\n<li>Open “Terminal” at the location
of “Podfile”.<\/li>\n<li>Verify that the XCode project is closed.<\
/li>\n<li>Run the “pod install” command.<\/li>\n<\/ol>\n<p>Open the
[MyProject].xcworkspace file in XCode. This file is located side by side with [M
yProject].xcodeproj.<br \/>\nAn usual CocoaPods-based project is managed as a w
orkspace containing the application (the executable) and the library (all proje
ct dependencies brought by the CocoaPods manager).<\/p>\n<p>In Xcode’s Bu
ild Settings, search for “Other Linker Flags” and insert ${inherite
d} (if -ObjC is defined in this field, you can just delete it, since it is confi
gured in the CocoaPod project).<\/p>\n<h3>Client App Migration<\/h3>\n<ol>\n
Search for bluemix dependency imports like\n<pre class=\"brush: plain; title: ;

notranslate\">#import <IMFCore\/IMFCore.h>\n#import <IMFPush\/IMFPush.
h><\/pre>\n<p>Replace the above imports with <\/p>\n<pre class=\"brush: plai
n; title: ; notranslate\">#import <IMFCompatibility\/IMFCompatibility.h><
\/pre>\n<\/li>\n<li>Look for a call to the “initializeWithBackendRoute&#8
221; method and replace the route URL with your on-premise server URL. For exam
ple:\n<pre class=\"brush: plain; title: ; notranslate\">IMFClient.sharedInstanc
e().initializeWithBackendRoute("https:\/\/parkstore.mybluemix.net", b
ackendGUID: "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: ; notranslate\">IMFClient.sharedInstance().initializeWithBackendRoute(
"http:\/\/localhost:9080\/ParkStoreMFP", backendGUID: "5e3ad88d-
dd48-469d-b46f-2c4ad66b5345"<\/pre>\n<p>Note, that backendGUID parameter i
s ignored and can be empty. Look for all instantiations of IMFResourceRequest cl
ass and update it<\/li>\n<li>Look for all instantiations of IMFResourceRequest c
lass and update the request URL with absolute or relative path to the resource.
For example:\n<pre class=\"brush: plain; title: ; notranslate\">var request: IM
FResourceRequest = IMFResourceRequest(path: "https:\/\/parkstore.mybluemix
.net\/LocalstoreAdapter\/apps\/5e3ad88d-dd48-469d-b46f-2c4ad66b5345\/localstore
\/getAllItems", method: "GET")<\/pre>\n<p>should be replaced wit
h<\/p>\n<pre class=\"brush: plain; title: ; notranslate\">var request: IMFResou
rceRequest = IMFResourceRequest(path: "http:\/\/localhost:9080\/ParkStoreM
FP\/adapters\/LocalstoreAdapter\/localstore\/getAllItems", method: "G
ET")<\/pre>\n<\/li>\n<li>Add the following code inside didRegisterForRemot
eNotificationsWithDeviceToken function in Appdelegate of Client application.\n<
pre class=\"brush: plain; title: ; notranslate\"> WLPush.sharedInstance().token
FromClient = deviceToken.description<\/pre>\n<\/li>\n<li>All on-premise applica
tions require the “worklight.plist” file to be present in the applic
ation resources. In the <code>IBMMobileFirstPlatformFoundationNativeSDK<\/code>
pod we supply a file named sample.worklight.plist<\/strong>.\n<ul>\n
Locate the “sample.worklight.plist” file in the â€˜IBMMobileFirstPl
atformFoundationNativeSDKâ€™ pod.<\/li>\n<li>Copy this file to the parent (appl
ication) project and rename it to “worklight.plist”.<\/li>\n<li>Edi
t the “worklight.plist” file by setting the “application id&#8
221; key to the name of your application deployed to the on-premise MFPF server<
\/li>\n<\/ul>\n<\/li>\n<\/ol>\n<h3 id=\"migratemfp\">Migration of JAX-RS Applic
ation to JAVA Adapter<\/h3>\n<ol>\n<li>To migrate JAX-RS application 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.ibm.com\/mobilefirstpl
atform\/wp-content\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-12-at-6.50
.04-pm.png\"><img src=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-con
tent\/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:\/\/d
eveloper.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\/sites\/32\/2015\/07\/Screen-Shot
-2015-07-12-at-6.51.13-pm.png\" alt=\"Screen Shot 2015-07-12 at 6.51.13 pm\" wi
dth=\"598\" height=\"590\" class=\"alignnone size-full wp-image-14818\" \/><\/a
><\/p>\n<p><a href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-conte
nt\/uploads\/sites\/32\/2015\/07\/Screen-Shot-2015-07-12-at-6.52.28-pm.png\"><im
g src=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/
sites\/32\/2015\/07\/Screen-Shot-2015-07-12-at-6.52.28-pm.png\" alt=\"Screen Sh
ot 2015-07-12 at 6.52.28 pm\" width=\"717\" height=\"424\" class=\"alignnone si
ze-full wp-image-14819\" \/><\/a><\/li>\n<li>Add two adapters for Custom Authent
ication and Localstore and migrate the JAX-RS code as shown in the following exa
mple.<\/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 follow
ing changes in your adapter code.<\/p>\n<ul>\n remove <code>\/{tenantId}\/<
\/code><\/li>\n remove the <code>@PathParam -> PathParam(\"tenantId\") S
tring deviceId<\/code> and <code>@PathParam(\"realmName\") String realmName<\/c
ode><\/li>\n Add scope to the all http api resource <code>@OAuthSecurity (sc
ope=\"customAuthRealm_3\")<\/code><\/li>\n<\/ul>\n<p>The code looks like the fo
llowing<\/p>\n<pre class=\"brush: plain; title: ; notranslate\">\n\t@GET\n\t@OA
uthSecurity (scope="customAuthRealm_3")\n\t@Path("\/getAllItems&
quot;)\n\tpublic String getAllItems() throws MissingConfigurationOptionExceptio
n{\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@P
UT\n\t@OAuthSecurity (scope="customAuthRealm_3")\n\t@Path("\/add
Item")\n\tpublic void addItem(String itemJson) \n\t\t\tthrows MissingConfig
urationOptionException, 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().getResou
rce("data.properties"); \n\t\t\tFile file = new File(url.toURI().getP
ath());\n\t\t\tFileOutputStream foStream = new FileOutputStream(file);\n\t\t\tp
rops.store(foStream, "saving new item");\n\t\t\tfoStream.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@OAuthSecurity (scope="customAuthRealm_3")\n\t@Path("\/a
ddAllItems")\n\tpublic String addAllItems(String itemsJson) \n\t\t\tthrows
MissingConfigurationOptionException, URISyntaxException, IOException{\n\t\ttry{
\n\t\t\tinit();\n\t\t\tclearAllData();\n\t\t\tJsonArray jsonArr = parser.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\t}\n\t
\t\tURL url = this.getClass().getClassLoader().getResource("data.prope
rties"); \n\t\t\tFile file = new File(url.toURI().getPath());\n\t\t\tF
ileOutputStream foStream = new FileOutputStream(file);\n\t\t\tprops.store(foStr
eam, "saving new item");\n\t\t\tfoStream.close();\n\t\t\tretu
rn "true";\n\t\t}catch(IOException ioe){\n\t\t\tioe.printStac
kTrace();\n\t\t}\n\t\treturn "false";\n\t}\n\n\t@DELETE\n\t@O
AuthSecurity(enabled=false)\n\t@Path("\/clearAll")\n\tpublic String c
learAllData() \n\t\t\tthrows MissingConfigurationOptionException, URISyntaxExce
ption, IOException{\n\t\t\tinit();\n\t\t\tprops.clear();\n\t\t\tSystem.out.prin
tln("Size : "+props.size());\n\t\t\tURL url = this.getClass().getClas
sLoader().getResource("data.properties"); \n\t\t\tFile file = new Fil
e(url.toURI().getPath());\n\t\t\tFileOutputStream foStream = new FileOutputStre
am(file);\n\t\t\tprops.store(foStream, "clearing all data");\n\t\t\tf
oStream.close();\n\t\t\treturn "cleared";\n\t}\n<\/pre>\n<h3 id=\"con
figoauth\">Configuring Custom-OAuth<\/h3>\n<ul>\n<li>Add realm with same name y
ou had on BlueMix and login module to the authenticationConfig.xml.\n<pre class
=\"brush: xml; title: ; notranslate\"><realm name="customAuthRealm_3&qu
ot; loginModule="customAuthLoginModule_3">\n<className>com.w
orklight.core.auth.ext.CustomIdentityAuthenticator<\/className>\t\n<pa
rameter name="providerUrl" value="http:\/\/localhost:9080\/ParkS
toreMFP\/adapters\/Customauth"\/>\n<\/realm>\n\n<loginModule n
ame="customAuthLoginModule_3" expirationInSeconds="3600"&gt
;\n<className>com.worklight.core.auth.ext.CustomIdentityLoginModule<\/
className>\n<\/loginModule><\/pre>\n<\/li>\n<li>Add Custom-oauth Realm
in userIdentityRealms in Application Descriptor file of iOS Native API\n<pre cla
ss=\"brush: xml; title: ; notranslate\"><userIdentityRealms>customAuthRea
lm_3<\/userIdentityRealms><\/pre>\n<\/li>\n<\/ul>\n<h3 id=\"configurepush
\">Configuring Push Capability<\/h3>\n<ul>\n Add apns p12 certificate which
is generated from Apple Developer Account under iOS Native API Folder\n<p><a hr
ef=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sit

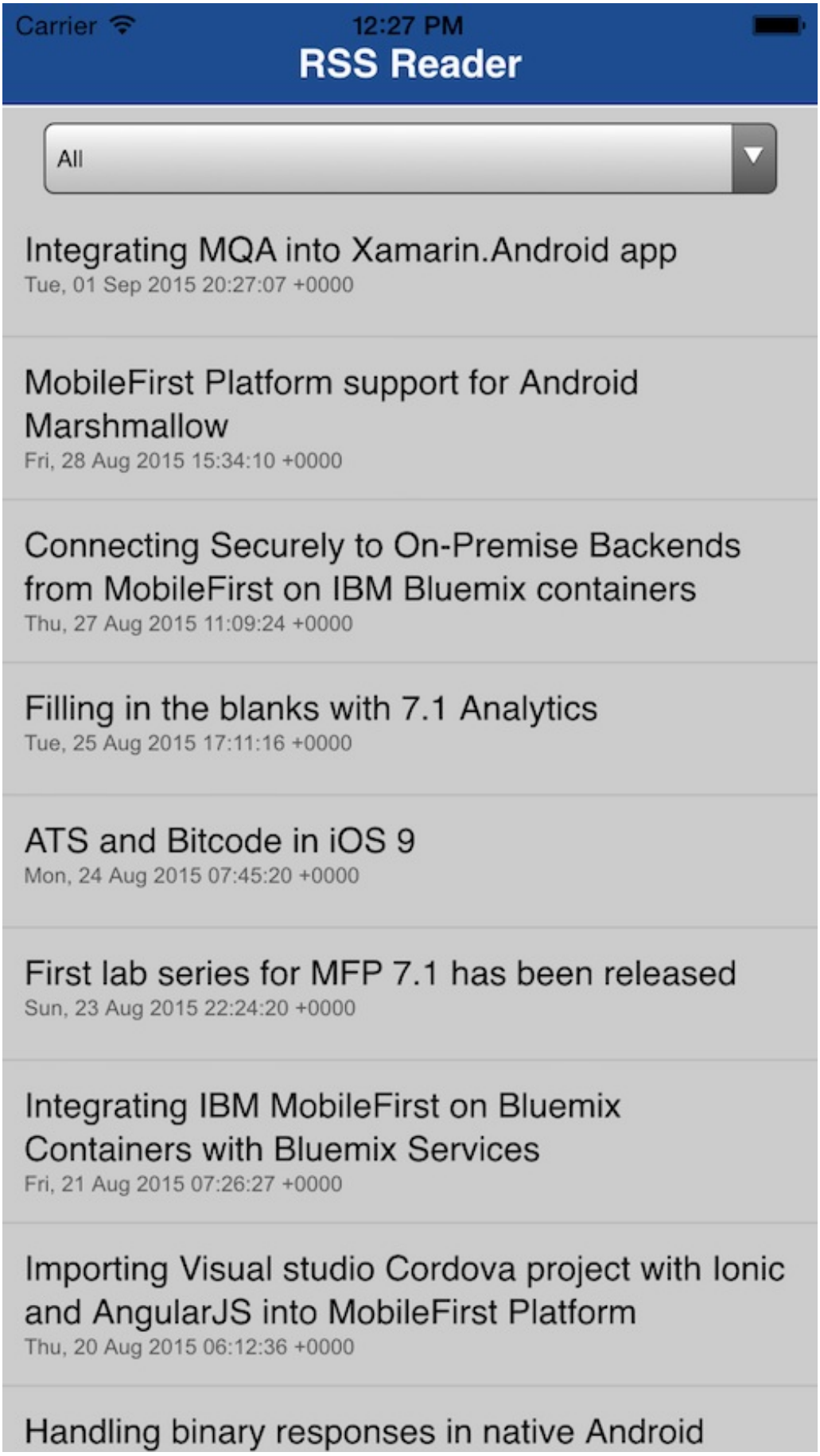
```
el=\"https:\/\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sit
es\/32\/2015\/07\/Screen-Shot-2015-07-12-at-6.58.03-pm.png\"><img src=\"https:\/
\/developer.ibm.com\/mobilefirstplatform\/wp-content\/uploads\/sites\/32\/2015\
/07\/Screen-Shot-2015-07-12-at-6.58.03-pm.png\" alt=\"Screen 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 configuration in Application Descriptor
file of iOS Native API and include the password of added apns certificate.\n<pre
class=\"brush: xml; title: ; notranslate\">&lt;pushSender password=&quot;passwo
rd&quot;\/&gt;\n&lt;tags&gt;\n  &lt;tag&gt;\n    &lt;name&gt;parkstore&lt;\/nam
e&gt;\n  &lt;\/tag&gt;\n&lt;\/tags&gt;<\/pre>\n<\/li>\n<li> Create HTTP Push Ad
apter with following function code which will send the user push notification to
the devices which is subscribed to tag &#8220;parkstore&#8221;.\n<pre class=\"br
ush: xml; title: ; notranslate\">function sendTagNotification(notificationText)
{\n    var notificationOptions = {};\n    notificationOptions.message = {};\n
notificationOptions.target = {};\n\n    notificationOptions.message.alert = not
ificationText;\n    notificationOptions.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.&quot;\n    };\n}<\/pre>\n<\/li>\n<\/ul>\n<p>By performing
above steps one can easily run iOS app built for Bluemix on MobileFirst Platform
and following are the links to samples.<\/p>\n<h3 id=\"sample\">Sample and Sourc
e Code<\/h3>\n<p>Bluemix Server : <a href=\"https:\/\/hub.jazz.net\/git\/chetha
n\/parkstore-bluemix-server\">Parkstore bluemix server<\/a><br \/>\nBluemix Cli
ent : <a href=\"https:\/\/hub.jazz.net\/git\/chethan\/parkstore-bluemix\">Parks
tore bluemix<\/a><br \/>\nMFP Server     : <a href=\"https:\/\/hub.jazz.net\/gi
t\/chethan\/parkstore-mfp-server\">Parkstore mfp server<\/a><br \/>\nMFP Client
: <a href=\"https:\/\/hub.jazz.net\/git\/chethan\/parkstore-mfp\">Parkstore mfp
<\/a><\/p>\n<p>The post <a rel=\"nofollow\" href=\"https:\/\/developer.ibm.com\
/mobilefirstplatform\/2015\/08\/19\/try-on-bluemix-and-buy-mfp\/\">Try on Bluem
ix and migrate to on-prem MobileFirst Platform<\/a> appeared first on <a rel=\"n
ofollow\" href=\"https:\/\/developer.ibm.com\/mobilefirstplatform\">IBM MobileFi
rst 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 Plat
form"
            }
         ],
         "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"
```

```
        "updatePeriod": "hourly"
      },
      "version": "2.0"
   }
}
```

Sample application

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