

iOS end-to-end demonstration

fork and edit tutorial (<https://github.ibm.com/MFPSamples/DevCenter/tree/master/tutorials/en/foundation/8.0/quick-start/ios/index.md>) | report issue (<https://github.ibm.com/MFPSamples/DevCenter/issues/new>)

Overview

The purpose of this demonstration is to experience an end-to-end flow where an application and an adapter are registered using the MobileFirst Operations Console, an "skeleton" Xcode project is downloaded and edited to call the adapter, and the result is printed to the log - verifying a successful connection with the MobileFirst Server.

Prerequisites:

- Xcode
- MobileFirst Developer CLI (download (file:///home/travis/build/MFPSamples/DevCenter/_site/downloads))
- *Optional.* Stand-alone MobileFirst Server (download (file:///home/travis/build/MFPSamples/DevCenter/_site/downloads))

1. Starting the MobileFirst Server

If a remote server was already set-up, skip this step.

From a **Command-line** window, navigate to the server's **scripts** folder and run the command: `./start.sh`.

2. Creating an application

In a browser window, open the MobileFirst Operations Console by loading the URL: `http://your-server-host:server-port/mfpconsole`. If running locally, use: `http://localhost:9080/mfpconsole` (`http://localhost:9080/mfpconsole`). The username/password are `admin/admin`.

1. Click on the "New" button next to **Applications** and select the desired *platform*, *identifier* and *version* values.



2. Click on the **Get Starter Code** tile and select to download the iOS Starter Code.





3. Editing application logic

1. Open the Xcode project project by double-clicking the **.xcworkspace** file.
2. Select the **[project-root]/ViewController.m/swift** file and paste the following code snippet, replacing the existing `viewDidLoad()` function:

In Objective-C:

```
- (void)viewDidLoad {
    [super viewDidLoad];

    NSURL* url = [NSURL URLWithString:@"../adapters/javaAdapter/users/world"];
    WLResourceRequest* request = [WLResourceRequest requestWithURL:url method:WLHttpMethodGet];

    [request sendWithCompletionHandler:^(WLResponse *response, NSError *error) {
        if (error != nil){
            NSLog(@"Failure: %@",error.description);
        }
        else if (response != nil){
            // Will print "Hello world" in the Xcode Console.
            NSLog(@"Success: %@",response.responseText);
        }
    }];
}
```

In Swift:

```
override func viewDidLoad() {
    super.viewDidLoad()

    let url = NSURL(string: "../adapters/javaAdapter/users/world")
    let request = WLResourceRequest(URL: url, method: WLHttpMethodGet)

    request.sendWithCompletionHandler { (WLResponse response, NSError error) -> Void in
        if (error != nil){
            NSLog("Failure: " + error.description)
        }
        else if (response != nil){
            NSLog("Success: " + response.responseText)
        }
    }
}
```

4. Creating an adapter

1. Click on the "New" button next to **Adapters** and download the **Java** adapter sample.

If Maven and MobileFirst CLI are not installed, follow the on-screen **Setting up your environment** instructions to install. Alternatively, download this prepared .adapter artifact and deploy it from the MobileFirst Operations Console using the **Actions → Deploy adapter** action.

MobileFirst Operations Console

Analytics Console

Hello, demo

Dashboard

Runtimes

mfpmfp

Settings

Applications

No application deployed

Adapters

No adapter deployed

Devices

Client Logs

Error Log

Create new

Create new

Home > mfp > Create a new Adapter

Create a new Adapter

It seems like you don't have any adapters, lets get started

Deploy Adapter

Follow these steps to set up an adapter

Hide guide

1

Setting up your environment

2

Start with a sample adapter

Console

CLI

Maven

CONSOLE

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean euismod bibendum laoreet. Proin gravida dolor sit amet lacus accumsan et viverra justo commodo. Proin sodales pulvinar tempor.

3

In your IDE of choice, edit the adapter code - REST end points and adapter descriptor

4

Build and package

5

Upload adapter

MobileFirst Operations Console

Analytics Console

Hello, demo

Dashboard

Runtimes

mfpmfp

Settings

Applications

MyBankApp

Adapters

No adapter deployed

Devices

Client Logs

Error Log

Create new

Create new

Home > mfp > Get Starter Code

Get Starter Code

Application

Adapter

Java Adapter Samples

Select the sample that you would like to download

Java

Java-security check

JavaScript Adapter Samples

Select the sample that you would like to download

JavaScript-SQL

JavaScript-SAP

JavaScript-JMS

JavaScript-HTTP

Next Step

Go to Adapter Quickstart

- From a **Command-line** window, navigate to the adapter's Maven project root folder and run the command:

```
mfpmdev adapter build
```

- When the build finishes, deploy it from the MobileFirst Operations Console using the **Actions** → **Deploy adapter** action. The adapter can be found in the **[adapter]/target** folder.

5. Testing the application

- In Xcode, select the **mfpmclient.plist** file and edit the **host** property with the IP address of the MobileFirst Server.
- Press the **Play** button.

The adapter response is then printed in the Xcode Console.

```

Date = "Tue, 19 Jan 2016 06:14:40 GMT";
"Transfer-Encoding" = Identity;
"X-Powered-By" = "Servlet/3.1";
}
Response Data:
{"access_token":"eyJhbGciOiJIUzI1NiIsInp5IjoiQVFBQ1IiIm4iOiJBTTEBEZD0dWR2NkgtekdWNN3I4cUNMZEUTM0kya2s0NXpnNnREZF9xczhmdm5Z2mRpcVRTVjRfMnQ2T0dHOENWNUNLNDFTXB3d21MDEwDLJm52aHhvwWLYGY01TYU9LSXFvZS1ySkEwdVp1dzJyS0hWjXAVNLS2V6U1ZjQ092ZlF0LW1RS2BtZno1ZkhvLWZmFVZd1hru0893QkJsMuVocUL3VVR3I21Zz3KtUdsNEVYc1BaZm0WkktSFU0b01pa51Uck5MeLJXa01tH2mDLotDV0b3NVTkEXNkZLQ08ta0D3xcG1T0t1JNjFuRgh1N2d0Mm95b0RueVUqFk1QW90hmLuSS0ZnL3HmVZmVWV1ZQ030VW1S061Z1Y0eS6Qk1odU0cc2ZkZHUxS3pc3RkLW0V0DU0JmTVRCNEdrT0hQNMWdjd0eJFRGhJHU41iwa3RS5Ijo1U1NB1iwa2Lk1jo1Y2mZDU0M2t0T1L3h00MDZjLTK5ZW0tMTZnGRINGU00Tc3In19_eyJpc3R0L01jIjB2RueWJkLm1ncC1Sn1N1Y1I6IzmlZm01MTYhLTk5YjctNDAYy050WVktEXMzRkYjRlNDk3MyIsImF1ZCI6ImNvbS5pYmububWZwiwi2XhwiJjoxNDUzMtG3NjgwMzY4LjCjZ29wZS16IiJ9_r96Ad4VUhyQ5tf_6C7w7Xdp37vNP_psgW1VNkznujY_BxE1dkcdIT3qpsnrzEBiH3qkL6B7FhnFu93vb_wqb9xKo0yNpAIX2ITrcxvQVMnoV061JpCpVij16sqeLWpkc3UIrrnPpLiBp7z0H--wR84XZPBE4ikby0dkXsg0Ld6T1QoS2JcgmLCP20sn0-w3p6vcUKwPSK30gQZyiAbbkTFNS_6VEaMER4Y8rJD8m1VcIF5Zgb3bb2E842VL0wKiuXy7YnGhdJKXqMkK1jYmJ0LJDNZdc_w-4Vyk6pjr7mXlrs6xZhXG84gy88MWG28a7TjFNpLXGd3g","token_type":"Bearer","expires_in":3599,"scope":""}
Status code:200
2016-01-19 08:14:40.410 MyApplication[93738:36590517] +[OCLogger printMessage:withMetadata:andLevelTag:andPackage:] [Line 1005] [DEBUG] [WL_AFHTTPSessionManagerWrapper_PACKAGE] ~[WLAFHTTPSessionManagerWrapper start] in WLAFHTTPSessionManagerWrapper.m:372 :: Starting the request with URL http://:9080/mfp/api/adapters/javaAdapter/users/world
2016-01-19 08:14:40.440 MyApplication[93738:36590517] +[OCLogger printMessage:withMetadata:andLevelTag:andPackage:] [Line 1005] [DEBUG] [WL_AFHTTPSessionManagerWrapper_PACKAGE] ~[WLAFHTTPSessionManagerWrapper requestFinished:responseObject:] in WLAFHTTPSessionManagerWrapper.m:388 :: Request Success
2016-01-19 08:14:40.440 MyApplication[93738:36590517] +[OCLogger printMessage:withMetadata:andLevelTag:andPackage:] [Line 1005] [DEBUG] [WL_AFHTTPSessionManagerWrapper_PACKAGE] ~[WLAFHTTPSessionManagerWrapper requestFinished:responseObject:] in WLAFHTTPSessionManagerWrapper.m:391 :: Response Status Code : 200
2016-01-19 08:14:40.440 MyApplication[93738:36590517] ~[WLAFHTTPSessionManagerWrapper requestFinished:responseObject:] [Line 393] Response Content : Hello world
2016-01-19 08:14:40.441 MyApplication[93738:36590517] Adapter invocation response: Hello world
  
```

Note: Xcode 7 enables Application Transport Security (ATS)

(https://developer.apple.com/library/ios/releasenotes/General/WhatsNewIniOS/Articles/iOS9.html#//apple_ref/doc/uid/TP40016198-SW14) by default.

To complete the tutorial, disable ATS (<http://iosdevtips.co/post/121756573323/ios-9-xcode-7-http-connect-server-error>).

1. In Xcode, right-click the **[project]/info.plist** file → **Open As** → **Source Code**
2. Paste the following:

```
<key>NSAppTransportSecurity</key>
<dict>
  <key>NSAllowsArbitraryLoads</key>
  <true/>
</dict>
```

3. Press the **Play** button.

Next steps

Learn more on using adapters in applications, and how to integrate additional services such as Push Notifications, using the MobileFirst security framework and more:

- Review the Server-side development tutorials ([../adapters/](#))
- Review the Authentication and security tutorials ([../authentication-and-security/](#))
- Review the Notifications tutorials ([../notifications/](#))
- Review All Tutorials ([../all-tutorials](#))