iOS end-to-end demonstration

Overview

The purpose of this demonstration is to experience an end-to-end flow:

1. A scaffold application - an application that is pre-bundled with the MobileFirst client SDK, is registered and downloaded from the MobileFirst Operations Console.
2. An new or provided adapter is deployed to the MobileFirst Operations Console.
3. The application logic is changed to make a resource request.

End result:

- Successfully pinging the MobileFirst Server.
- Successfully retrieving data using a MobileFirst Adapter.

Prerequisites:

- Xcode
- *Optional*. 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

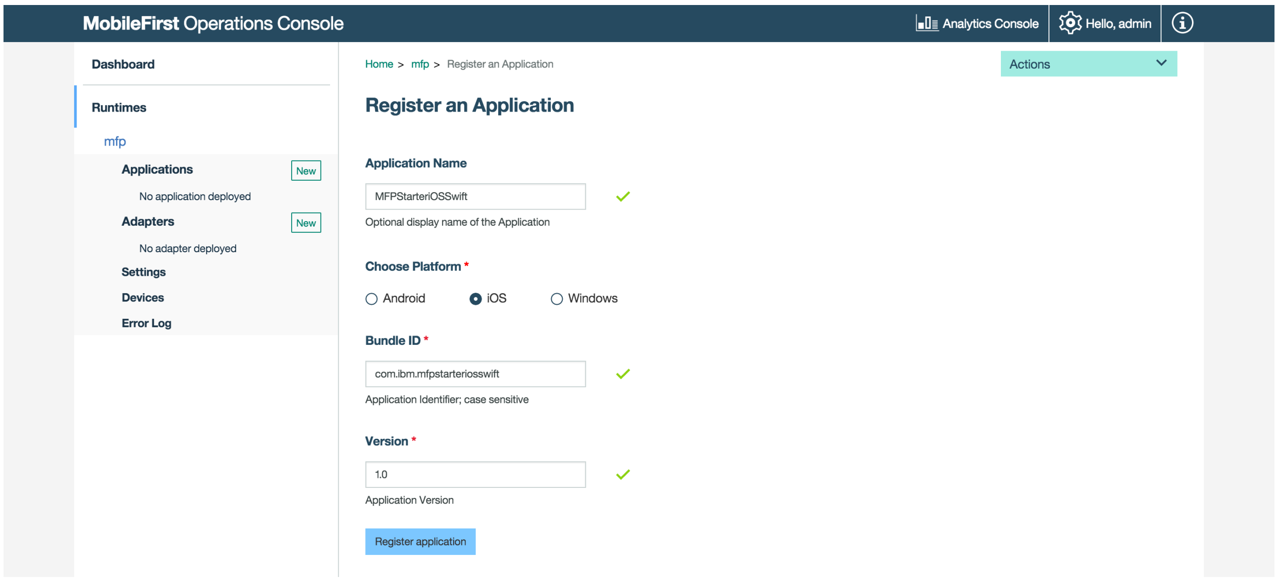
> If a remote server was already set-up, skip this step.

From a **Command-line** window, navigate to the server's folder and run the command: `./run.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 the **New** button next to **Applications**
   - Select the **iOS** platform
   - Enter **com.ibm.mfpstarteriosobjectivec** or **com.ibm.mfpstarteriosswift** as the **application identifier** (depending on the application scaffold you will download in the next step)
   - Enter **1.0** as the **version** value
   - Click on **Register application**



Register an Application

Application Name: MFPStarteriOSSwift ✓
Optional display name of the Application

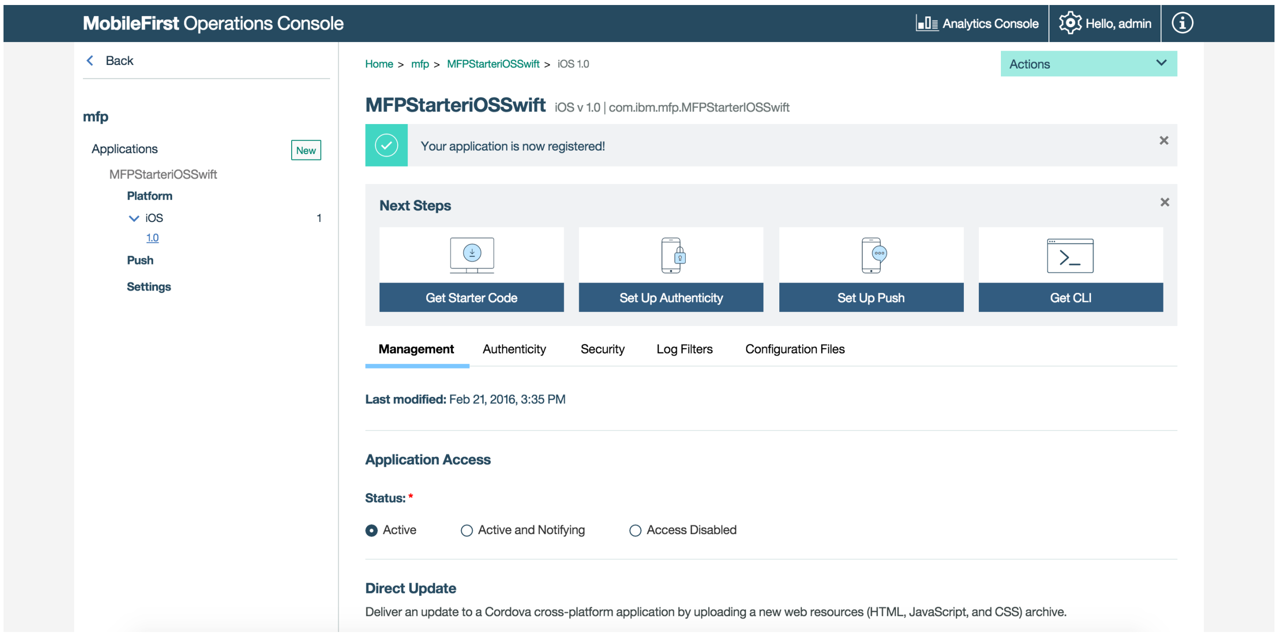
Choose Platform *: Android [✓ iOS] Windows

Bundle ID *: com.ibm.mfpstarteriosswift ✓
Application Identifier; case sensitive

Version *: 1.0 ✓
Application Version

Register application

2. Click on the **Get Starter Code** tile and select to download the iOS Objective-C or iOS Swift application scaffold.



MFPStarteriOSSwift iOS v 1.0 | com.ibm.mfp.MFPStarterIOSSwift

Your application is now registered!

Next Steps

Last modified: Feb 21, 2016, 3:35 PM

Application Access

Status: * [✓ Active] Active and Notifying Access Disabled

Direct Update
Deliver an update to a Cordova cross-platform application by uploading a new web resources (HTML, JavaScript, and CSS) archive.

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 `getAccessToken()` function:

   In Objective-C:

```
- (void)testServerConnection {
    _connectionStatusText.text = @"Connecting to Server...";
    [[WLAuthorizationManager sharedInstance] obtainAccessTokenForScope: @"" withCompletionH
andler:^(AccessToken *accessToken, NSError *error) {
        if (error != nil){
            NSLog(@"Failure: %@",error.description);
            _connectionStatusText.text = @"Client Failed to connect to Server";
        }
        else if (accessToken != nil){
            NSLog(@"Success: %@",accessToken.value);
            _connectionStatusText.text = @"Client has connected to Server";
            NSURL* url = [NSURL URLWithString:@"/adapters/javaAdapter/users/world"];
            WLResourceRequest* request = [WLResourceRequest requestWithURL:url method:WLHtt
pMethodGet];
            [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:

```
@IBAction func getAccessToken(sender: AnyObject) {
    connectionStatusWindow.text = "Connecting to Server...";
    print("Testing Server Connection")
    WLAuthorizationManager.sharedInstance().obtainAccessTokenForScope(nil) { (token, error) -
> Void in
        if (error != nil) {
            self.connectionStatusWindow.text = "Client Failed to connect to Server"
            print("Did not Recieved an Access Token from Server: " + error.description)
        } else {
            self.connectionStatusWindow.text = "Client has connected to Server"
            print("Recieved the Following Access Token value: " + token.value)
            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

Download this prepared .adapter artifact (../javaAdapter.adapter) and deploy it from the MobileFirst Operations Console using the **Actions → Deploy adapter** action.

Alternatively, click the **New** button next to **Adapters**.

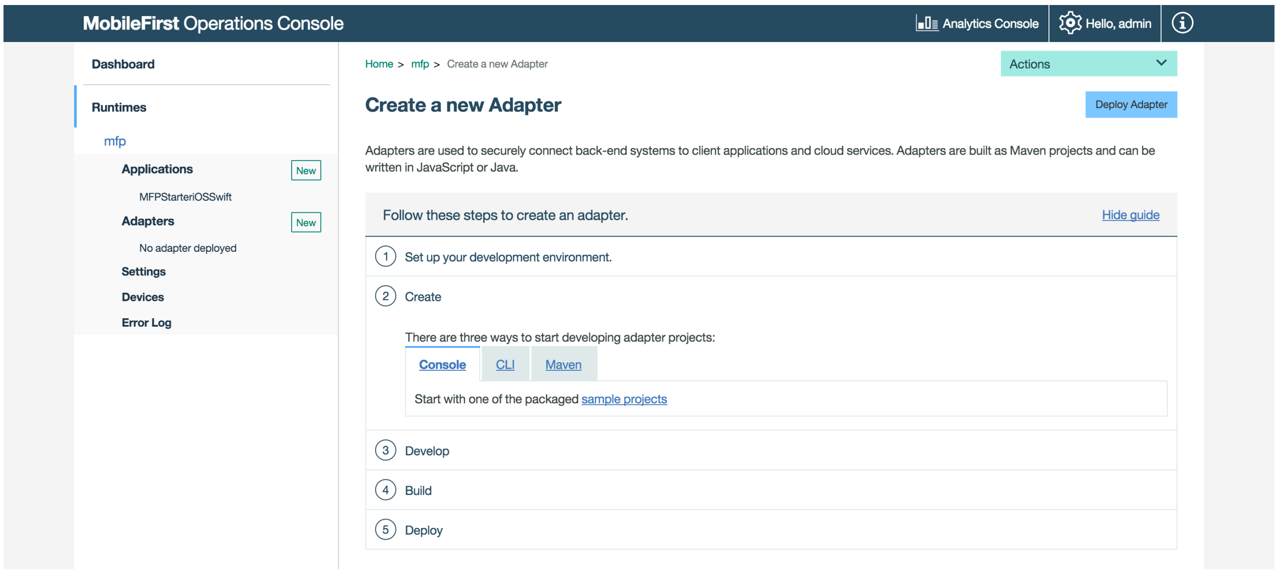
1. Select the **Actions → Download sample** option. Download the "Hello World" **Java** adapter sample.

   > If Maven and MobileFirst Developer CLI are not installed, follow the on-screen **Set up your development environment** instructions.

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

   ```
   mfpdev adapter build
   ```

3. 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.



Create a new Adapter

Adapters are used to securely connect back-end systems to client applications and cloud services. Adapters are built as Maven projects and can be written in JavaScript or Java.

Follow these steps to create an adapter.

1 Set up your development environment.

2 Create

There are three ways to start developing adapter projects:

Start with one of the packaged sample projects

3 Develop

4 Build

5 Deploy

5. Testing the application

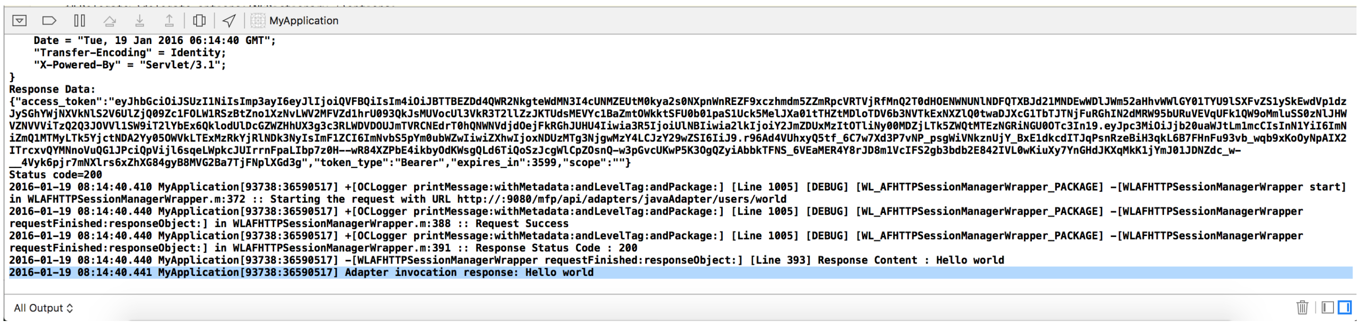
1. In Xcode, select the **mfpclient.plist** file and edit the **host** property with the IP address of the MobileFirst Server.

   Alternatively, if you have installed the MobileFirst Develper CLI then navigate to the project root folder and run the command `mfpdev app register`. If a remote server is used instead of a local server, first use the command `mfpdev server add` to add it.
2. Press the **Play** button.

Results

- Clicking the **Ping MobileFirst Server** button will display **Connected to MobileFirst Server**.
- If the application was able to connect to the MobileFirst Server, a resource request call using the deployed Java adapter will take place.

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":"eyJhbGciOiJSUzI1NiIsImp3ayI6eyJlIjoiQVFBQiIsIm4iOiJBTTBEZDd4QWR2NkgteWdMN3I4cUNMZEUtM0kya2s0NXpnWnREZF9xczhmdm5ZZmRpcVRTVjRfMnQ2T0dHOENWNUNlNDFQTXBJd21MNDEwWDlJWm52aHhvWWlGY01TYU9lSXFvZS1ySkEwdVp1dz
JySGhYWjNXVkNlS2V6UlZjQ09Zc1FOLW1RSzBtZno1XzNvLWV2MFVZd1hrU093QkJsMUVocUl3VkR3T2llZzJKTUdsMEVYc1BaZmtOWkktSFU0b01paS1Uck5MelJXa01tTHZtMDloTDV6b3NVTkExNXZlQ0twaDJXcG1TbTJTNjFuRGhIN2dMRW95bURuVEVqUFk1QW9oMmluSS0zNlJHW
VZNVVViTzQ2Q3JOVVl1SW9iT2lYbEx6QklodUlDcGZWZHhUX3g3c3RLWDVDOUJmTVRCNEdrT0hQNWNVdjdOejFkRGhJUHU4Iiwia3R5IjoiUlNBIiwia2lkIjoiY2JmZDUxMzItOTliNy00MDZjLTk5ZWQtMTEzNGRiNGU0OTc3In19.eyJpc3MiOiJjb20uaWJtLm1mcCIsInN1YiI6ImN
iZmQ1MTMyLTk5YjctNDA2Yy05OWVkLTExMzRkYjRlNDk3NyIsImF1ZCI6ImNvbS5pYm0ubWZwIiwiZXhwIjoxNDUzMTg3NjgwMzY4LCJzY29wZSI6IiJ9.r96Ad4VUhxyQ5tf_6C7w7Xd3P7vNP_psgWiVNkznUjY_BxE1dkcdITJqPsnRzeBiH3qkL6B7FHnFu93vb_wqb9xKoOyNpAIX2
ITrcxvQYMNnoVuQG1JPciQpVijl6sqeLWpkcJUIrrnFpaLIbp7z0H--wR84XZPbE4ikbyOdKWsgQLd6TiQoSzJcgWlCpZOsnQ-w3pGvcUKwP5K3OgQZyiAbbkTFNS_6VEaMER4Y8rJD8m1VcIFS2gb3bdb2E842IVL0wKiuXy7YnGHdJKXqMkK1jYmJ01JDNZdc_w-
__4Vyk6pjr7mNXlrs6xZhXG84gyB8MVG2Ba7TjFNplXGd3g","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
```

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 Using the MobileFirst Platform Foundation (../../using-the-mfpf-sdk/) tutorials
- Review the Adapters development (../../adapters/) tutorials
- Review the Authentication and security tutorials (../../authentication-and-security/)
- Review the Notifications tutorials (../../notifications/)
- Review All Tutorials (../../all-tutorials)