

iOS end-to-end demonstration

Overview

The purpose of this demonstration is to experience an end-to-end flow where an application is quickly created using the MobileFirst Operations Console and connectivity is verified with the MobileFirst Server.

Prerequisites:

- Configured 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.

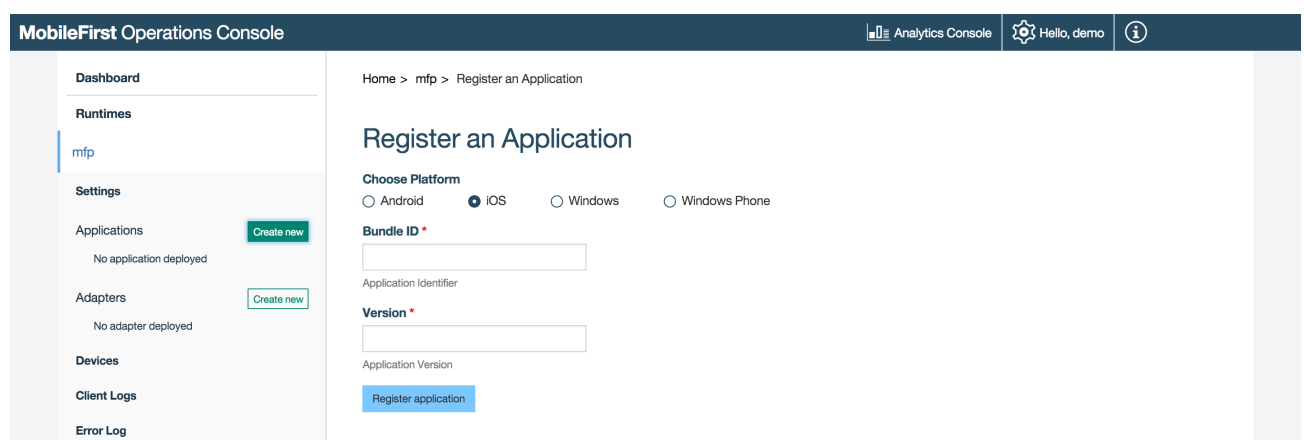
1. From a **Command-line** window, navigate to the server's **scripts** folder and run the command:

```
./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 "Create 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.
2. Select the **[project-root]/ViewController.m/swift** file and:
 - Add the following header:

In Objective-C:

```
#import <IBMMobileFirstPlatformFoundation/IBMMobileFirstPlatformFoundation.h>
```

In Swift:

```
import IBMMobileFirstPlatformFoundation
```

- Paste the following code snippet in the `viewDidLoad` function:

In Objective-C:

```

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

[request sendWithCompletionHandler:^(WLResponse *response, NSError *error) {
    if(error != nil){
        NSLog(@"%@",error.description);
    }
    else{
        NSLog(@"%@",response.responseText);
    }
};

```

In Swift:

```

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("error.description")
    }
    else {
        NSLog("response.responseText")
    }
}

```

4. Creating an adapter

1. Click on the "Create 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.

MobileFirst Operations Console

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



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, run the command:

```
mfpdev adapter deploy
```

If using a remote MobileFirst Server, run the command:

```
mfpdev adapter deploy Replace-with-remote-server-name
```

5. Testing the application

1. In Xcode, press the **Play** button.



Next steps

- Review the Client-side development tutorials (../client-side-development/)
- Review the Server-side development tutorials (../server-side-development/)
- Review the Authentication and security tutorials (../authentication-and-security/)
- Review All Tutorials (../all-tutorials/)