

Using the MobileFirst Operations Console

fork and edit tutorial (<https://github.ibm.com/MFPSamples/DevCenter/tree/master/tutorials/en/foundation/8.0/setting-up-your-development-environment/console/index.md>) | report issue (<https://github.ibm.com/MFPSamples/DevCenter/issues/new>)

Overview

The MobileFirst Platform Operations Console is a web-based UI which enables simplified work flows for both the developer and the administrator to create, monitor, secure and administer applications & adapters.

Jump to:

- Accessing the console
- Navigating the console

Accessing the console

The MobileFirst Operations Console can be accessed in the following ways:

From a locally installed MobileFirst Server

Desktop Browser

From your browser of choice, load the URL `http://localhost:9080/mfpconsole` (`http://localhost:9080/mfpconsole`). The username/password are *admin/admin*.

Command-line

From a **Command-line** window, with the MobileFirst CLI installed, run the command: `mfpdev server console`.

From a remotely installed MobileFirst Server

Desktop Browser

From your browser of choice, load the URL `http://the-server-host:server-port-number/mfpconsole`

The host server can be a customer-owner server, or running on a service such as Bluemix. The username/password are *admin/admin*.

Command-line

From a **Command-line** window, with the MobileFirst CLI installed,

1. Add a remote server definition:

Interactive Mode

Run the command: `mfpdev server add` and follow the on-screen instructions.

Direct Mode

Run the command with the following structure: `mfpdev server add [server-name] --URL [remote-server-URL] --login [admin-username] --password [admin-password] --contextroot [admin-service-name]`. For example:

```
mfpdev server add MyRemoteServer http://my-remote-host:9080/ --login TheAdmin --password ThePassword --contextroot mfpadmin
```

2. Run the command: `mfpdev server console MyRemoteServer`.

Learn more about the various CLI commands in the [Using CLI to manage MobileFirst artifacts](#) ([../using-the-mfpf-sdk/using-cli-to-manage-mobilefirst-artifacts/](#)) tutorial.

Navigating the console

Dashboard

The Dashboard provides a glance view of the deployed projects.



Runtime settings

Edit runtime properties such as Analytics server URL, global security variables, server keystore and confidential clients.



Applications

Creating applications

Provide basic application values and download Starter Code.

The screenshot shows the 'MobileFirst Operations Console' interface. The left sidebar contains a navigation menu with 'Dashboard', 'Runtimes', 'Applications', 'Adapters', 'Settings', 'Devices', and 'Error Log'. The 'Applications' section is active, showing 'No application deployed' and a 'New' button. The main content area is titled 'Register an Application' and contains the following fields:

- Application Name:** A text input field.
- Optional display name of the Application:** A text input field.
- Choose Platform:** Radio buttons for 'Android', 'iOS' (selected), and 'Windows'.
- Bundle ID:** A text input field with the note 'Application Identifier; case sensitive'.
- Version:** A text input field with the note 'Application Version'.
- Register application:** A blue button at the bottom.

The top right of the console shows 'Analytics Console', 'Hello, admin', and an information icon.

Managing applications

Manage and configure registered applications by use of Direct Update (../using-the-mfpf-sdk/direct-update/), Remote Disable, Application Authenticity (../authentication-and-security/application-authenticity/), and setting security parameters (../authentication-and-security/authorization-concepts/).

The screenshot shows the 'MobileFirst Operations Console' interface for managing a specific application. The left sidebar shows the navigation menu with 'MyApp' selected under 'Applications'. The main content area is titled 'MyApp' and shows 'iOS v 1.0 | com.sample.myapplication'. The 'Management' tab is active, showing the following information:

- Last modified:** Feb 15, 2016, 10:40 PM
- Application Access:** Status is 'Active' (selected), with options for 'Active and Notifying' and 'Access Disabled'.
- Direct Update:** A section for delivering updates to a Cordova cross-platform application by uploading a new web resources (HTML, JavaScript, and CSS) archive.
- No Web resources deployed:** A message with an 'Upload Web Resources File' button.

The top right of the console shows 'Analytics Console', 'Hello, admin', and an information icon.

Authentication and Security

Configure application security parameters, such as the default token expiration value, map scope elements to security checks, define, mandatory application scopes and configure security check options.



The screenshot shows the MobileFirst Operations Console interface. The top navigation bar includes the title 'MobileFirst Operations Console', an 'Analytics Console' link, a user profile 'Hello, admin', and an information icon. The left sidebar shows a breadcrumb trail: 'Back' > 'mfp' > 'MyApp' > 'iOS 1.0'. The main content area is titled 'MyApp iOS v 1.0 | com.sample.myapplication' and has tabs for 'Management', 'Authenticity', 'Security' (selected), 'Log Filters', and 'Configuration Files'. The 'Security' section is titled 'Security' and contains a description: 'This is where you will set up the advanced security framework configuration offered by MobileFirst Platform to protect your enterprise data and APIs.' Below this is a 'Token Configurations' section with the text 'Configure the access tokens provided by the MobileFirst Server'. A form field for 'Maximum token expiration (seconds) *' is set to '3600', with an 'Edit' button. A 'Map scope elements to security checks' section includes a 'Create New' button. At the bottom, a message states 'You have not mapped any scope elements to security checks.' with a 'Get started by clicking "Create New"' instruction and an illustration of a smartphone.

Notifications

Set-up push notifications ([../../notifications/push-notifications-overview/](#)) and related parameters, such as certificates and GCM details, define tags, as well as send notifications to devices.



The screenshot shows the MobileFirst Operations Console interface for the 'Push' settings. The top navigation bar is the same as the previous screenshot. The left sidebar shows the breadcrumb trail: 'Back' > 'mfp' > 'MyApp' > 'iOS 1.0' > 'Push'. The main content area is titled 'Push' and has tabs for 'Send Push', 'Tags', and 'Push Settings' (selected). The 'Push Settings' section is titled 'Push Notification Settings' and contains the text 'Configure your push notifications here. For detailed instructions, take a look at our [Push Notifications Guide](#).' Below this is a form for 'Apple Push Notifications Certificate'. It includes a link to 'Learn more about how to generate and use Apple Push Notification service (APNs) certificates in [Apple Push Notifications certificates guide](#)'. There are two radio buttons for 'Choose use *': 'Production' (selected) and 'Sandbox'. A 'Select PKCS 12 (.p12) File *' section has a 'No file selected' button and a 'Browse' button. A 'Password *' section has an 'Enter Password' input field and a 'Save' button.

Adapters

Creating adapters

Register an adapter and download Starter Code, as well as update an adapter on-the-fly by updating its properties without needing to re-build and re-deploy the adapter artifact.

The screenshot shows the 'Create a new Adapter' page in the MobileFirst Operations Console. The left sidebar contains a navigation menu with 'Dashboard', 'Runtimes', 'mfp', 'Applications', 'Adapters', 'Settings', 'Devices', and 'Error Log'. The main content area is titled 'Create a new Adapter' and includes a 'Deploy Adapter' button. Below the title, there is a section 'Follow these steps to create an adapter' with a 'Hide guide' link. The first step is 'Set up your development environment', which includes instructions on installing Node.js, the CLI, and Maven. A code block shows the command `npm install -g mfpdev-cli`.

MobileFirst Operations Console

Home > mfp > Create a new Adapter

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 [Hide guide](#)

- 1 Set up your development environment

Development of adapters can be done in any environment that supports Maven. In order to build, deploy and update adapters, you can use Maven or the MobileFirst Platform Foundation command line tool.

Installing the command line interface (CLI)

If you haven't, download and install Node.js. Install the CLI using npm utility of Node.js. The MFP CLI will automatically be downloaded by the npm utility.

```
npm install -g mfpdev-cli
```

Installing Maven

Follow the instructions on the [Apache Maven website](#) to download and install Maven.

Adapter properties

After an adapter is deployed, it can be configured in the console.

The screenshot shows the 'javaAdapter' configuration page in the MobileFirst Operations Console. The left sidebar contains a navigation menu with 'Dashboard', 'Runtimes', 'mfp', 'Settings', 'Applications', 'Adapters', 'Devices', 'Client Logs', and 'Error Log'. The main content area is titled 'javaAdapter' and includes a 'Delete' button. Below the title, there is a section 'Resources' with a table of API endpoints.

MobileFirst Operations Console

Home > mfp > javaAdapter

javaAdapter

Configurations **Resources** Configuration Files

Resources

URL	Methods	Security
/users	GET	
/users/hello/UserQuery	GET	
/users/newUsers	PUT	
/users/{first}/{middle}/{last}	POST	
/users/{username}	GET	

Devices

Administrators can search for devices that access the MobileFirst Server and can manage access rights. Devices can be searched for using either user ID or using a friendly name. The user ID is the identifier that was used to log-in.

A friendly name is a name that is associated with the device to distinguish it from other devices that share the user ID.

For more information, see the topic about device access management in the MobileFirst Operations Console in the user documentation.



MobileFirst Operations Console

Home > mfp > Devices

Search device by user identifier or display name (3 chars at least).

Device ID	Display Name	User ID	Model & OS Version	Device Status	Actions
7D518329-2B1A-4E2E-B04F-7BF9EA5B972B			iPhone iOS 9.2	Active	

Installed Applications

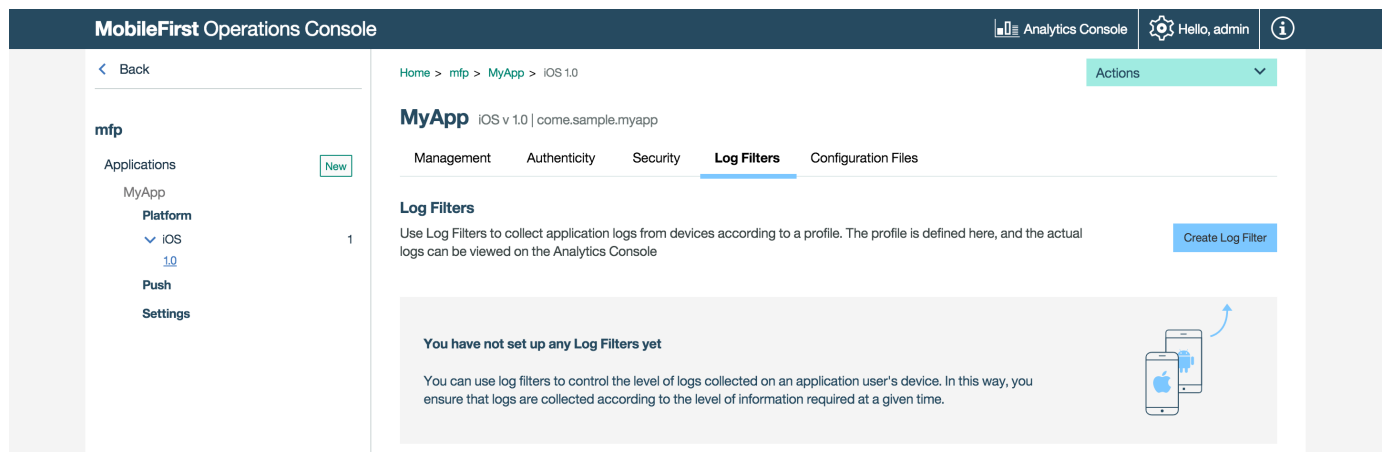
Application Name	Application Status
com.sample.PinCodeSwift	Enabled

Client logs

Administrators can use log profiles to adjust client logger configurations, such as log level and log package filters, for any combination of operating system, operating system version, application, application version, and device model.

When an administrator creates a configuration profile, the log configuration is concatenated with responses API calls such as `WLResourceRequest`, and is applied automatically.

For more information, see the topic about client-side log capture configuration from MobileFirst Operations Console in the user documentation.



MobileFirst Operations Console

Home > mfp > MyApp > iOS 1.0

MyApp iOS v 1.0 | com.sample.myapp

Management Authenticity Security **Log Filters** Configuration Files

Log Filters

Use Log Filters to collect application logs from devices according to a profile. The profile is defined here, and the actual logs can be viewed on the Analytics Console

Create Log Filter

You have not set up any Log Filters yet

You can use log filters to control the level of logs collected on an application user's device. In this way, you ensure that logs are collected according to the level of information required at a given time.

Error log

The Error log shows a list of the failed management operations that were initiated from the MobileFirst Operations Console, or from the command line, on the current runtime environment. Use the log to see the effect of the failure on the servers.

For more information, see the topic about error log of operations on runtime environments in the user documentation.



License tracking

Accessible from the top Settings buttons.

License terms vary depending on which edition (Enterprise or Consumer) of MobileFirst Platform Foundation is being used. License tracking is enabled by default and tracks metrics relevant to the licensing policy, such as active client devices and installed applications. This information helps determine whether the current usage of MobileFirst Platform is within the license entitlement levels and can prevent potential license violations.

By tracking the usage of client devices and determining whether the devices are active, administrators can decommission devices that should no longer be accessing the service. This situation might arise if an employee has left the company, for example.

For more information, see the topic about license tracking in the user documentation.



Downloads

For situations where Internet connectivity is not available, you can download a snapshot of the various development artifacts of MobileFirst Platform Foundation from the Downloads page.

Dashboard

Runtimes

mfp

Applications

No application deployed

New

Adapters

No adapter deployed

New

Settings

Devices

Error Log

Home > Downloads

Actions

Downloads

Applications

Adapters

Tools

MobileFirst Platform Development CLI

Download this thing and then run `npm install -g mfpdev-cli.tgz`

Download

Adapter Tooling

Everything you need to develop adapters using Maven.

Download

MobileFirst Extension for OAuth Security

You can protect your resources that are running on WebSphere® Application Server or WebSphere Application Server Liberty servers with OAuth-based IBM MobileFirst™ Platform Foundation security.

Download

MobileFirst SDKs for Mobile Application Development

Select the SDK for the mobile application platform for which you are developing