MobileFirst Foundation 8.0 Migration Cookbook

Overview

The purpose of this cookbook is to provide a clear and simple view of the migration steps for applications and adapters from IBM Worklight Foundation 6.2 or IBM MobileFirst Platform Foundation 6.3 - 7.1, to IBM MobileFirst Foundation 8.0.

The migration process guides you through the steps to transform classic hybrid applications into standard Cordova applications, and to update the MobileFirst SDK in native applications. Mobile Web apps are handled, too. Adapters are migrated into Maven projects, and implementation concepts such as the MobileFirst security framework, push notifications, and direct update are further clarified.

To ease some aspects of the migration process, a Migration Assistance tool is provided.
The tool helps you by identifying areas in your codebase that you will need to inspect and alter, such as APIs that are deprecated, no longer supported, or modified.

> **Note:** This cookbook does not attempt to cover all possible migration scenarios and you are advised to visit the migration user documentation topics for a comprehensive read (../).

Getting Help

The MobileFirst Foundation team is glad to help with any questions you might have.
For official support, open a PMR (https://www-947.ibm.com/support/servicerequest/newServiceRequest.action). You can also ask questions over at StackOverflow (https://stackoverflow.com/questions/tagged/ibm-mobilefirst) and in our Slack community (file:////home/travis/build/MFPSamples/DevCenter/_site/blog/2015/08/19/come-chat-with-us/).

Quick Start

Before getting started with migrating your applications and adapters, and in case you haven't already, it is recommended that you go through the Quick Start tutorials (file:////home/travis/build/MFPSamples/DevCenter/_site/tutorials/en/foundation/8.0/quick-start/) to get familiar and experience MobileFirst Foundation 8.0.

More information is provided in the tutorials (file:////home/travis/build/MFPSamples/DevCenter/_site/tutorials/en/foundation/8.0/all-tutorials/) about topics such as registering applications, building and deploying adapters, implementing and configuring security and authentication, and lots more.

Setup and Tools

Before migration can begin, you need a running MobileFirst Server instance and the Migration Assitance tool.

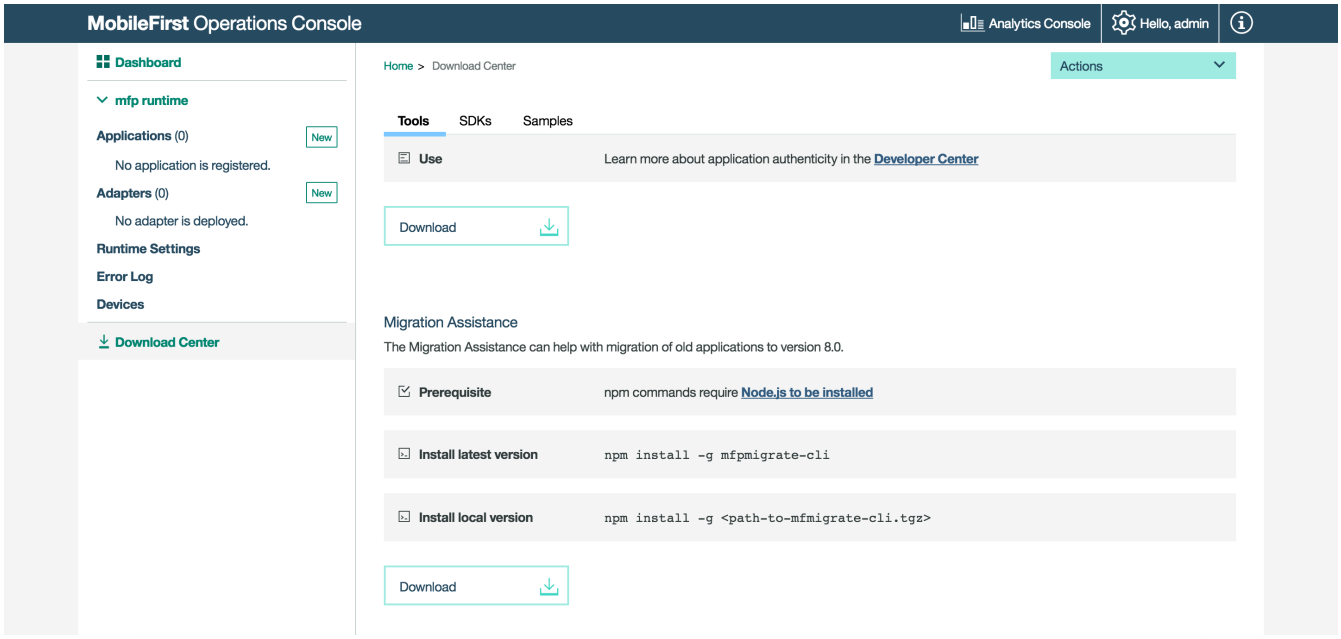
MobileFirst Server

You can set up a MobileFirst Server on IBM Bluemix (https://bluemix.net) by using the Mobile Foundation Bluemix service (file:////home/travis/build/MFPSamples/DevCenter/_site/tutorials/en/foundation/8.0/bluemix/using-mobile-foundation/).
Alternatively, you can use the MobileFirst Development Kit (file:////home/travis/build/MFPSamples/DevCenter/_site/tutorials/en/foundation/8.0/installation-configuration/development/mobilefirst/) to set up a server that runs locally.

Migration Assitance tool

After the MobileFirst Server is up and running, and because you will use the Migration Assistance tool later on, first install the tool.

1. Install NodeJS (https://nodejs.org/en/) as it is a prerequisite for the tool to work.
2. From the MobileFirst Operations Console, click on the Download Center link at the bottom of the sidebar navigation. Follow the instructions to install the Migration Assistance tool, either from NPM or by downloading a .zip file.



Migrating Applications

The steps to migrate your classic hybrid/MobileFirst Cordova, Web, and native applications include: assessing API changes by running the Migration Assistance tool, setting up the project structure, managing the application source, using package managers, and handling API changes.

Classic Hybrid Applications

In past releases, classic hybrid applications were created, developed, built, and managed by using the MobileFirst Studio plug-in for Eclipse or MobileFirst CLI. Starting with MobileFirst Foundation 8.0, support is introduced for standard Cordova applications, replacing the previous application model.

You create Cordova applications by using standard community tools, such as the Cordova CLI. The MobileFirst Cordova SDK is added to the mixture using the Cordova CLI as a set of Cordova plug-ins, which are available from npm (https://npmjs.org).

The move to standard Cordova applications opens the door for developers to use their favorite tools and their own approaches to application development. As a developer, you are now empowered with the Cordova ecosystem (http://cordova.apache.org).

Tip: You can also set up Eclipse for Cordova application development (file:////home/travis/build/MFPSamples/DevCenter/_site/tutorials/en/foundation/8.0/application-development/using-mobilefirst-cli-in-eclipse/).

> Learn more about Cordova application development (file:////home/travis/build/MFPSamples/DevCenter/_site/tutorials/en/foundation/8.0/cordova-tutorials/).

Migrating classic hybrid/MobileFirst Cordova apps to standard Cordova apps

Web applications

Much like classic hybrid applications, web applications for the Mobile Web and Desktop Browsers were also managed from the MobileFirst Studio plug-in for Eclipse or MobileFirst CLI. In MobileFirst Foundation 8.0, development of web apps is done in the traditional fashion, with the MobileFirst Web SDK available from npm (https://npmjs.org), too.

Note: In previous releases, MobileFirst Server served the web application and provided a public-facing URL. In MobileFirst Foundation 8.0, the application is registered only in the MobileFirst Server in order to provide it with security functionality, adapters capabilities, and so on, and the serving of the web application is done by standard methods, such as serving it from dedicated web servers.

> Learn more about Web application development (file:////home/travis/build/MFPSamples/DevCenter/_site/tutorials/en/foundation/8.0/web-tutorials/).

Migrating Mobile Web/Desktop Browser applications

Native applications

In previous versions of the product, native applications required either the MobileFirst Studio plug-in for Eclipse or MobileFirst CLI to first create the platform-specific artifacts (WorklightAPI folder, configuration files, and so on) followed by a manual copy and paste of those artifacts into the native projects in their respective IDEs.

Starting with MobileFirst Foundation 8.0, support is now introduced for community-favored package managers: CocoaPods for iOS (https://cocoapods.org/), Gradle for Android (https://gradle.org/), and NuGet for Windows (https://www.nuget.org/). With these tools being available for developers, adding the MobileFirst Native SDK is now streamlined for each platform.

> Learn more about Native application development (file:////home/travis/build/MFPSamples/DevCenter/_site/tutorials/en/foundation/8.0/).

Updating the SDKs in native applications

> **Supplemental reading:**
>
> - Comparison of Cordova apps developed with v8.0 versus v7.1 and earlier (../migrating-client-applications/cordova/#comparison-of-cordova-apps-developed-with-v8-0-versus-v7-1-and-before)
> - Features that are discontinued in 8.0 and features that are not included in 8.0 (../../product-overview/release-notes/deprecated-discontinued)
> - Migrating existing hybrid or cross-platform apps to Cordova apps supported by MobileFirst version 8.0 (../migrating-client-applications/cordova/#migrating-existing-hybrid-or-cross-platform-apps-to-cordova-apps-supported-by-mobilefirst-foundation-8-0)
> - Migrating encryption for Cordova iOS applications (../migrating-client-applications/cordova/#migrating-encryption-for-ios-cordova)

Your application is now compliant with the structure that is required by MobileFoundation 8.0. You can now start looking at the adapters.

Migrating Adapters

In past releases, adapters were created, developed, and built using the MobileFirst Studio plug-in for Eclipse or the MobileFirst CLI. Starting with MobileFirst Foundation 8.0, adapters are now considered as standard Apache Maven projects that are created using IBM-provided archetypes for generating the Java and JavaScript adapters.

Using Maven provides to server-side developers a simple and standard way to manage and integrate required dependencies, as well as frees them to use their favorite tools during development time. Maven projects can be created from a command-line using either Maven commands, or by using the MobileFirst CLI that calls Maven commands behind-the-scenes and provides simplified commands.

Tip: You can also setup Eclipse or IntelliJ to create and develop adapters in an IDE environment (file:////home/travis/build/MFPSamples/DevCenter/_site/tutorials/en/foundation/8.0/adapters/developing-adapters/).

Did you know? MobileFirst Foundation 8.0 introduces a DevOps-style operation mode, where once an adapter is deployed to the MobileFirst Server you can then configure various properties (for example, a username and password value for a database connection) live via the MobileFirst Operations Console, without needing to re-deploy the adapter.

> Learn more about developing Java and JavaScript adapters (file:////home/travis/build/MFPSamples/DevCenter/_site/tutorials/en/foundation/8.0/adapters).

The steps to migrate your adapters into Maven projects include: creating a matching new Maven project and copying into it the existing adapter's source code (with some modifications). This is then followed by building the Maven project to find any errors.

Prerequisites for setting up Apache Maven

Creating a new Maven project

Migrating the JavaScript source code

Migrating the Java source code

Building and testing the Maven project

Supplemental reading:

- Migrating existing adapters to work under MobileFirst Server V8.0.0 (../migrating-adapters)
- Server-side API changes in Mobile Foundation 8.0 (../../product-overview/release-notes/deprecated-discontinued/#server-side-api-changes)

With the adapters migrated to a Maven project compliant form and code, it's time look at some development concepts...

Development Topics

With applications and adapters now migrated to their new structure in MobileFirst Foundation 8.0, here is a discussion about some implementation concepts that have changed in the latest release.

Adapters

JavaScript adapters

Global variables and sessions
Avoid using global variables in JavaScript adapters to save data on the session - because there is no "session". This change in behavior first appeared in MobileFirst Platform Foundation 7.1 (http://www.ibm.com/support/knowledgecenter/SSHS8R_7.1.0/com.ibm.worklight.dev.doc/devref/c_overview_session_indep.html). However, you can use global variables to save data for use during a single request. Keep in mind, though, that this is not a recommended practice any longer.

Learn more about creating, building, deploying, and testing adaptersin the Adapters tutorials (../../adapters/).

Security framework

The security framework in MobileFirst Foundation 8.0 is different from the security framework in past releases. This is why you must re-implement some of your application back-end and client logic.

It is highly preferable that you take the time to familiarize yourself with the new security framework and its authorization flow, which are now based solely on the OAuth model. Also learn about the new authorization entities such as access tokens, security checks and scopes - which replace previously known entities such as security tests, realms, and login modules.

Learn more about the security framework and authorization concepts in the Authentication and security tutorials (../../authentication-and-security/).

Security framework differences

Notifications

In MobileFirst Foundation 8.0, a new push service is introduced, along with a new experience for setting up, configuring, and sending notifications. In addition, the concept of event-source push notifications has been withdrawn. Find detailed scenario-based migration paths in these user documentation topics:

- Migrating to push notifications from event source-based notifications (../migrating-push-notifications)
- Migration scenarios by platform (../migrating-push-notifications/#migration-scenarios)

Learn more about setting up notification support and sending authenticated and tag-based notifications in the Notifications tutorials (file:////home/travis/build/MFPSamples/DevCenter/_site/tutorials/en/foundation/8.0/notifications/).

Direct Update

The steps to deliver updates by using the Direct Update feature have changed, and some restrictions are imposed.

- Review the Migrating Direct Update (../migrating-client-applications/cordova/#migrating-direct-update) user documentation topic.

Learn more about how to use Direct Update in theUsing Direct Update in Cordova applications (file:////home/travis/build/MFPSamples/DevCenter/_site/tutorials/en/foundation/8.0/application-development/direct-update/) tutorial.

Other

In MobileFirst Foundation 8.0, several other components and features have been removed. For a full list, review the Removed Components (../migrating-client-applications/cordova/#removed-components) user documentation topic.

As a final treat, below is a video showcasing migration of a MobileFirst Platform Foundation 6.3 Hybrid application and adapter to a standard Cordova application and Maven project.

Last modified on