Overview of client technologies

fork and edit tutorial (https://github.ibm.com/MFPSamples/DevCenter/tree/master/tutorials/en/foundation/7.1/advanced-client-side-development/overview-of-client-technologies.html) | report issue (https://github.ibm.com/MFPSamples/DevCenter/issues/new)

Overview

This tutorial provides an overview of technologies for client-side development. Each technology is then further covered in separate tutorials.

IBM MobileFirst Platform Foundation supports both native applications (Objective-C, Java, and C#) and hybrid web-based applications, and also includes and supports various technologies to help you develop mobile applications. Each technology has its advantages and disadvantages. You must know which tool to use in any situation.

Full native development

If your app requires the full power of the environment, you can develop it purely in native code. This approach is also useful when your deploy existing native apps on the MobileFirst Server.

The platform supplies a native MobileFirst client SDK to manage authentication and back-end access, and for you to benefit from more server functionality.
For more information, see the tutorials about developing native applications for MobileFirst.

Hybrid web-based development

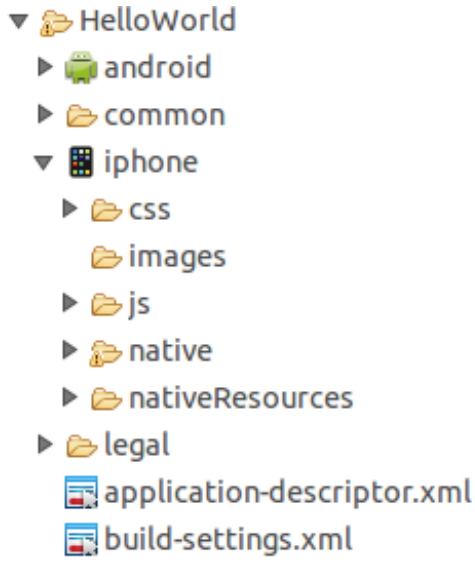
You can choose to develop your MobileFirst application as a hybrid, web-based application.
You develop MobileFirst hybrid apps by using standard web technologies: a single HTML file, JavaScript, style sheets, and images.

- Environment-specific code
- Skins
- Cordova APIs
- Cordova plug-ins
- Native UI
- Multipage techniques
- Offline access

MobileFirst apps use the MobileFirst API to access back-end data and server functionality.

Some controls are common to most hybrid environments. Examples: modal pop-ups, loading screens, and tab bars.
MobileFirst provides a JavaScript API to invoke these controls regardless of the environment and automatically renders them in a native way for each platform.
For more information, see the Common UI controls (../../advanced-client-side-development/common-ui-controls/) tutorial.

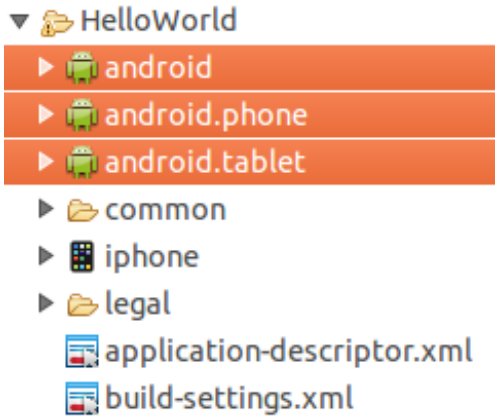
Environment-specific code



To achieve maximum adaptation to a specific environment, you can optimize web resources specifically for that environment. For example, an iPhone environment requires an iPhone-specific look and feel, which you can obtain by extending CSS or JavaScript files. MobileFirst eases the code maintenance of multiple-environment optimization.

For more information, see the Optimizing your application for various environments (../../client-side-development-basics/optimizing-your-application-for-various-environments/) tutorial.

Skins



Skins provide support for multiple form factors in a single executable file for devices of the same OS family. They are a subvariant of an environment and are packaged together in one app. The decision on which skin to use is made automatically at run time.

For more information, see the Supporting multiple form factors by using skins (../../advanced-client-side-development/supporting-multiple-form-factors-using-skins/) tutorial.

Cordova APIs

Cordova (http://cordova.apache.org/) is an open source development framework that is based on JavaScript for building multiplatform mobile apps. The MobileFirst Framework uses the Cordova library and exposes the Cordova APIs so that developers can access native device functions through those services.

For more information, see the Apache Cordova overview (../../adding-native-functionality/apache-cordova-overview/) tutorial.

Cordova plug-ins

Cordova plug-in are an abstraction that allows native components to be called by using a JavaScript API. With custom Cordova plug-ins, you can add native functions such as an encryption library or computational components and call them by using JavaScript. By default, a plug-in must be implemented for each supported environment (for example, iPhone), and various Cordova plug-ins exist on the market.

For more information, see the Adding native functionality to hybrid applications (../../adding-native-functionality/) tutorial.

Native UI

MobileFirst provides different ways to augment web applications with native pages. Developers can navigate freely between web and native pages, share data between pages, and share a single server session. For example, contact details can be accessed by using the Apache Cordova API. However, if your application can use the contact application in the device, use the Native Page API. You can also modify the native project generated by MobileFirst to write your own custom native code
For more information, see the Using native pages in hybrid applications (../../adding-native-functionality/) tutorials.

Multipage techniques

You can build applications with multiple pages in two ways:

- A single HTML file that contains all app pages. Pages are *div*s that are hidden or shown at run time.
- A separate HTML file for each application page.

The first option is better for small apps but is less scalable.
For more information, see the Building a multipage application (../../client-side-development-basics/building-a-multipage-application/) tutorial.

Offline access

MobileFirst apps work in concert with the MobileFirst Server. In offline mode, developers can detect app connectivity failures and determine the best course of action. Developers can define custom application behavior for offline and online status.

For more information, see the Data (../../data/) category.