Shell development concepts

Overview

This tutorial covers the basics of creating and using a shell component and an inner application.

- Concepts
- Creating a shell component
- Using a shell component in a test application
- Creating and using a shell bundle in an inner application
- Environment-specific tutorials

Prerequesites

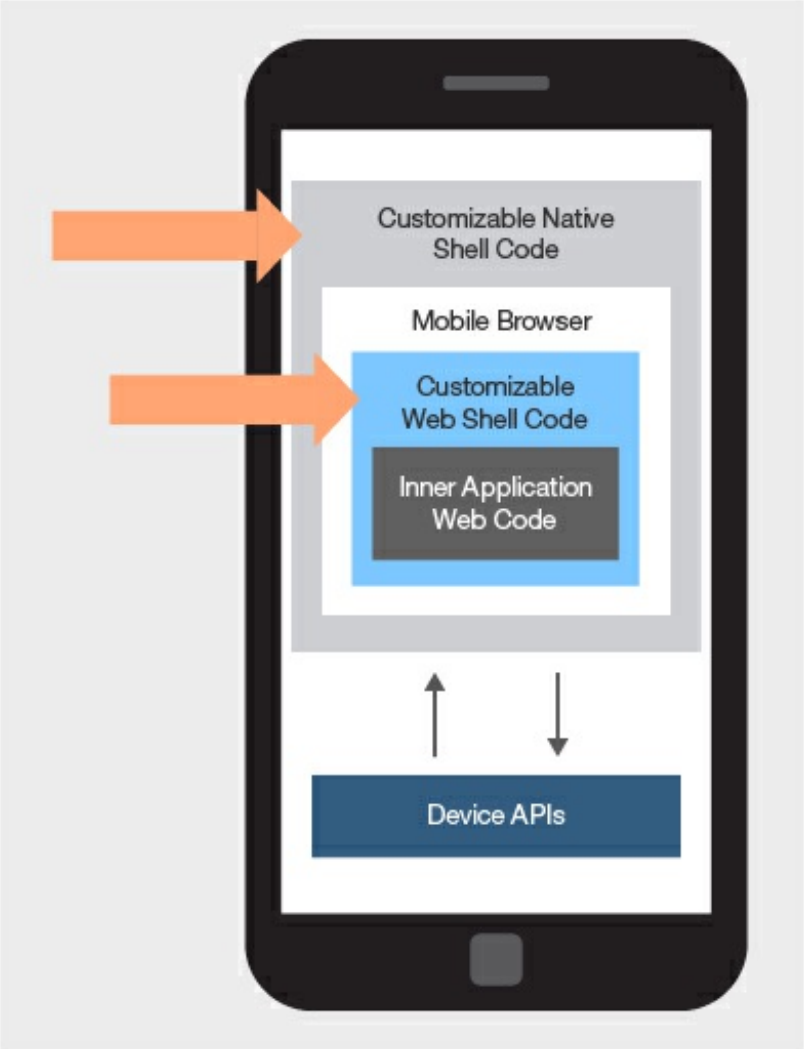
A prerequisite for successful completion of these tutorials about shell components is a good knowledge of the concepts that underlie the development of hybrid applications with IBM MobileFirst Platform Foundation.
Make sure that you fully understand MobileFirst development principles, specifically for development on iOS and Android environments.

Concepts

The main idea behind the shell component methodology is to create two levels of development inside the organization:

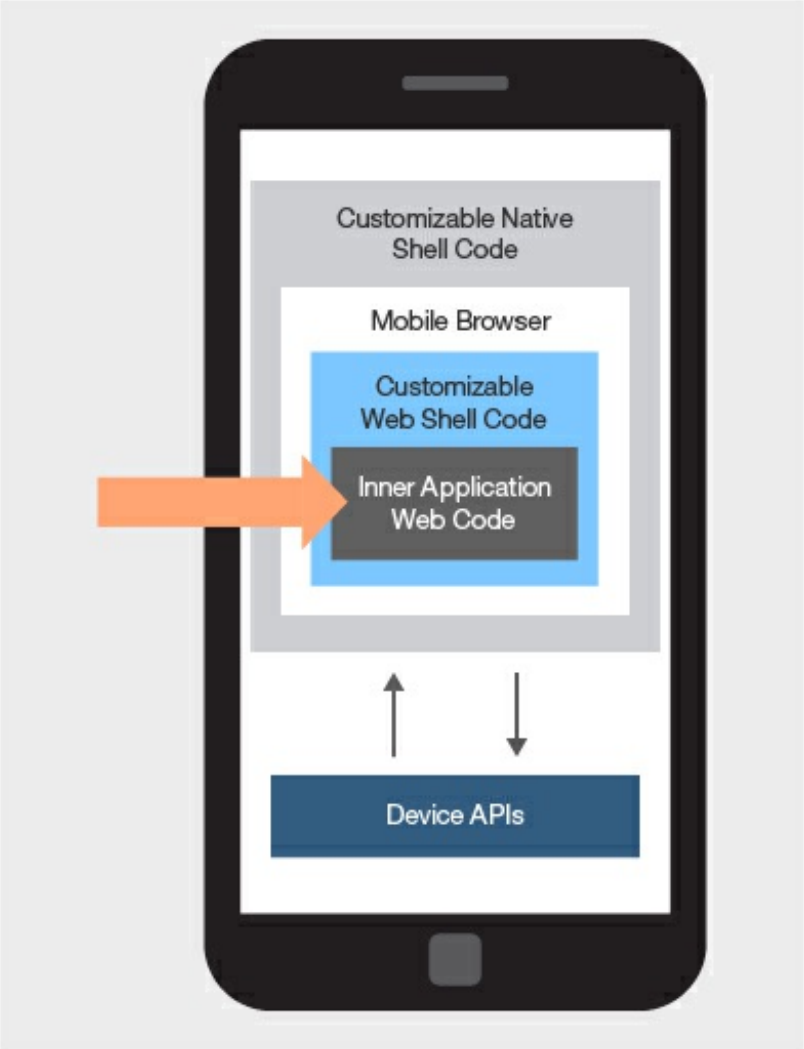
- Developers who are skilled in native development implement native and web code-bases which can be used as a starting point for one or more applications. For example:
  - Native functionality to be invoked from JavaScript (Cordova plug-ins)
  - Authentication framework
  - Security configuration
  - Web resources that are shared between applications, such as logotypes and themes
- Developers who have fewer skills in native development but more web expertise receive a ready-to-use shell component and use it as a wrapper to create the organization applications. For example:
  - Business logic
  - UI development
  - Data integration

Shell component



A shell component is a component to be used by inner applications as a code base wrapper. It usually consists of native classes and shell-specific web resources that are going to be used in inner applications. The shell component is implemented by shell developers and sent to developers of inner applications.

Inner application



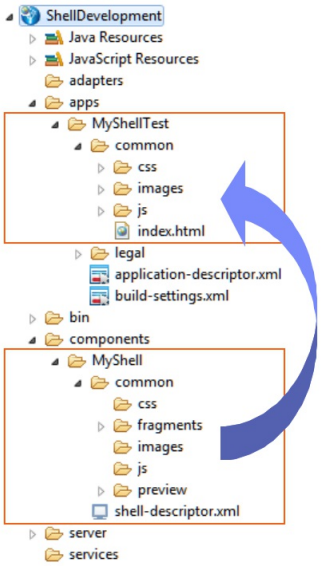
An inner application is a set of web resources (HTML / JavaScript / CSS) that are run inside the shell component.

Test application

The shell component is not executable by itself. After it is created, an inner application is automatically added to the project by MobileFirst Studio. This application is used by the shell developer to test the shell component functionality.

Creating a shell component

A shell component is a building block that is used to create inner applications.
Add a shell component to your project and name it `MyShell`.
Note: The `MyShellTest` application was automatically created for you; this application is a test application as described in section Test application. You can use it to test and debug the shell component.



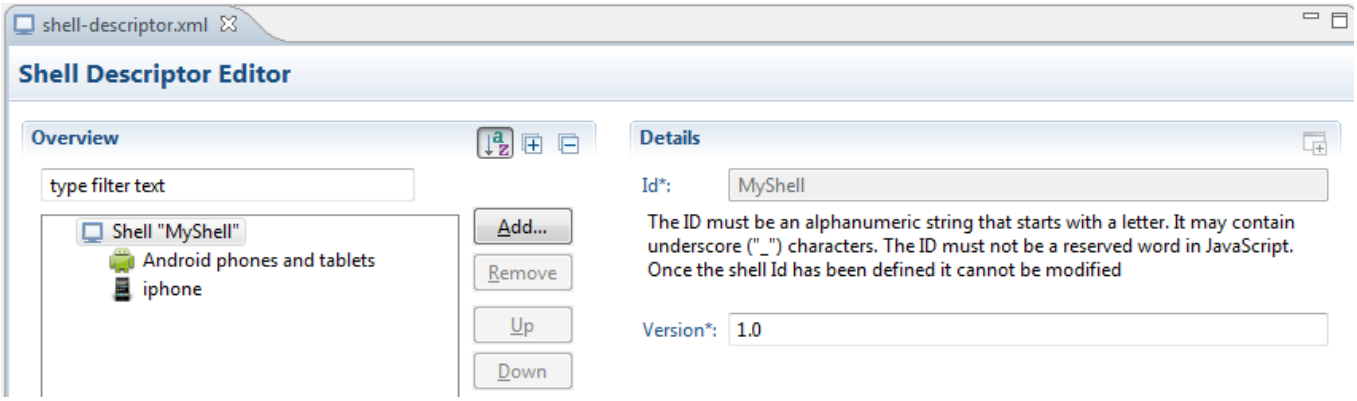
The common folder

The `common` folder of the shell component contains the following folders:

- **`css`**, **`images`**, **`js`** – These folders contain web resources that are added automatically to inner applications at build time.
- **`fragments`** – This folder contains HTML fragments that are added to predefined locations in the main HTML file of the inner application.
- **`preview`** – This folder can be used to implement stubs for simulating native functionality in the MobileFirst Operations Console preview instead of receiving exceptions.

The shell-descriptor.xml file

The `shell-descriptor.xml` file contains shell component metadata and application-specific properties. Application-specific properties that are set in the shell descriptor are used in all inner applications.
You can edit the `shell-descriptor.xml` file in Design mode or Source mode.




```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<shell xmlns="http://www.worklight.com/shell-descriptor" id="MyShell" platformVersion="7.0.0.00" versi
on="1.0">
    <iphone version="1.0"/>
    <android version="1.0"/>
    <features/>
</shell>
```

Using a shell component in a test application

Follow these instructions to develop a functioning shell component.

1. Create a `myshell.js` file in the `MyShell\common\js` folder.

2. Add the following function to it:

```
function sayHelloFromShell(){
    alert("Hello from Shell");
}
```

3. Modify the `body-top.wlfragment` file and add the following lines to it:

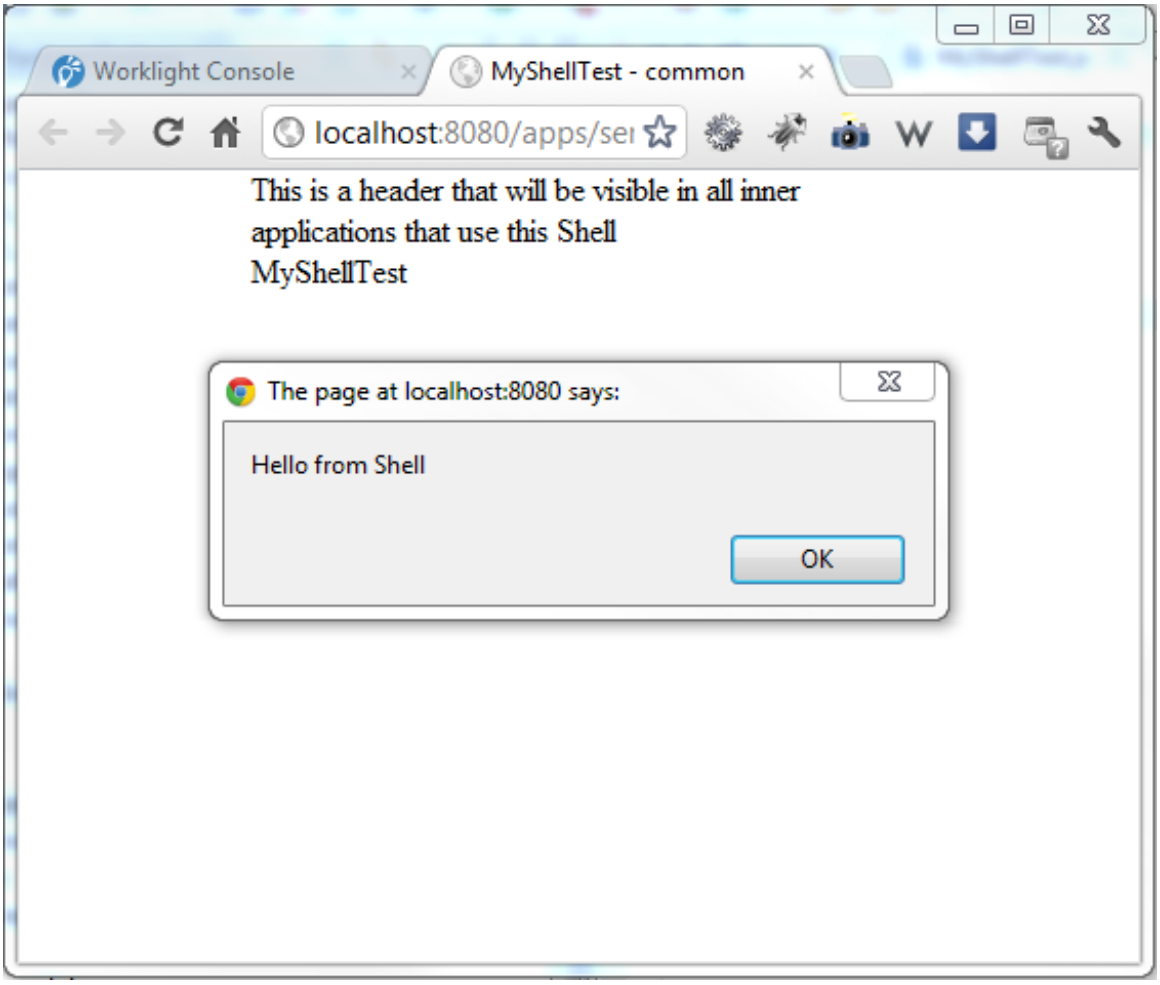
```
<h1>This is a header that will be visible in all inner applications that use this Shell</h1>
<script src="js/myshell.js"></script>
```

4. Modify the `main.js` file in the `apps/MyShellTest/common/js` folder.
   Invoke the function that you previously added in the shell component.

```
function wlCommonInit(){
    sayHelloFromShell();
}
```

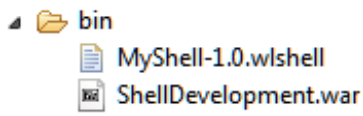
   Note: The `sayHelloFromShell()` function is not a part of the inner application, but is from the shell component.

5. Build and deploy the `MyShellTest` application.
   When your application is built and deployed, you find it in the MobileFirst Operations Console as a regular hybrid application.
6. Preview your `MyShellTest` application. Note that it contains web resources from both the shell component and the inner application.



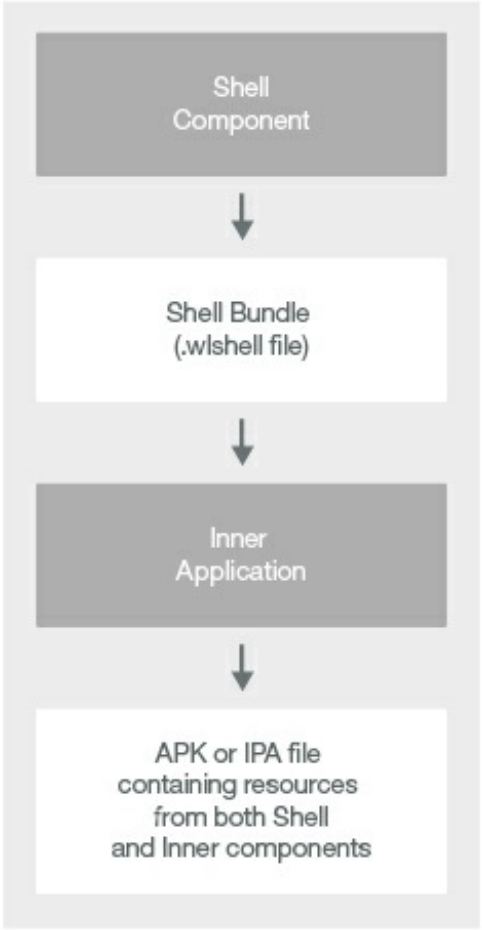
Creating and using a shell bundle in an inner application

When the shell developer builds a shell component, a `.wlshell` file is created in the project `bin\` folder. This file is called a shell bundle and can be sent for inner application developers to use.
A shell developer who works with a test application is not required to explicitly create a shell bundle. The test application references the shell component source code directly from the location that is specified in its `application-descriptor.xml` file.
However, when the shell developer wants to send the shell component to the inner application developer, it becomes necessary to create a shell bundle.



To create a shell bundle, right-click a shell component folder and select **Run As > Build Shell Component**.
The `.wlshell` file is created in the `bin\` folder of your project, as described previously.
The shell developer can send this file to inner application developers.

The inner application developer must copy the shell bundle file to a MobileFirst project.
When inner application developers create a new inner application, they must specify the location of a shell bundle file.
If a new shell bundle file is received from shell component developers, inner application developers must replace the existing shell bundle file and rebuild their applications.


Shell
Component
Shell Bundle
(.wlshell file)
Inner
Application
APK or IPA file
containing resources
from both Shell
and Inner components