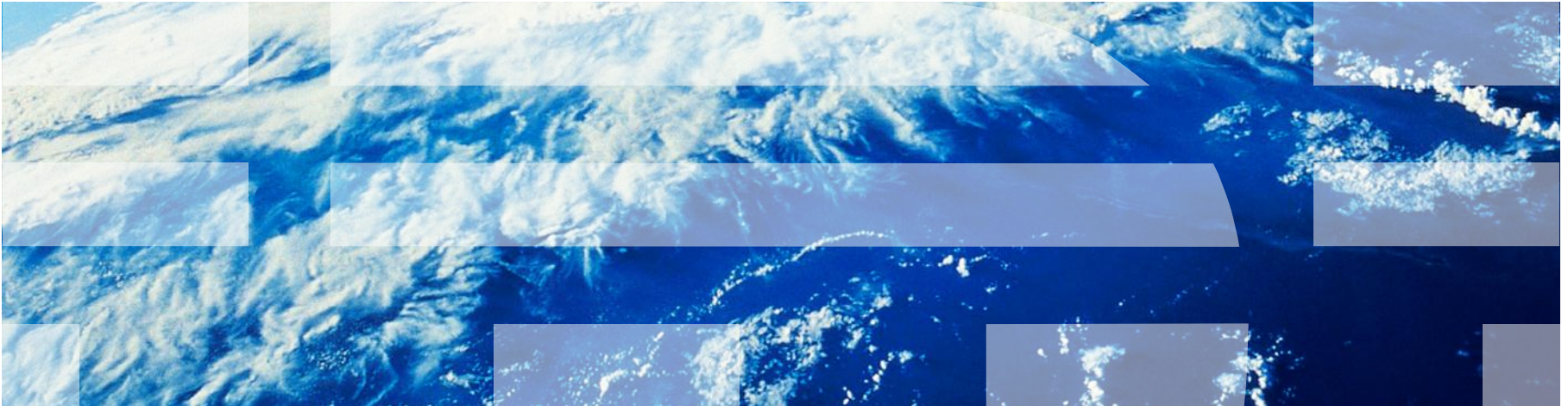# IBM Worklight Foundation V6.2.0
Getting Started

Creating your first Hybrid Worklight application

Trademarks

- IBM, the IBM logo, ibm.com, and Worklight are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

- Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

- Other company products or service names may be trademarks or service marks of others.

- This document may not be reproduced in whole or in part without the prior written permission of IBM.
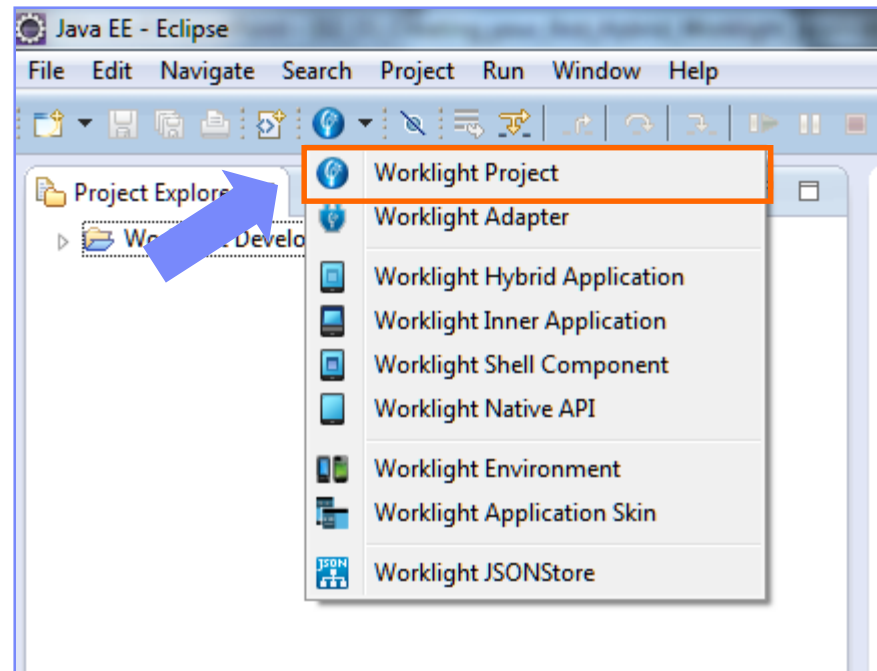
About IBM®

- See http://www.ibm.com/ibm/us/en/

Agenda

- Write your first application

- Become familiar with the application files and HTML structure

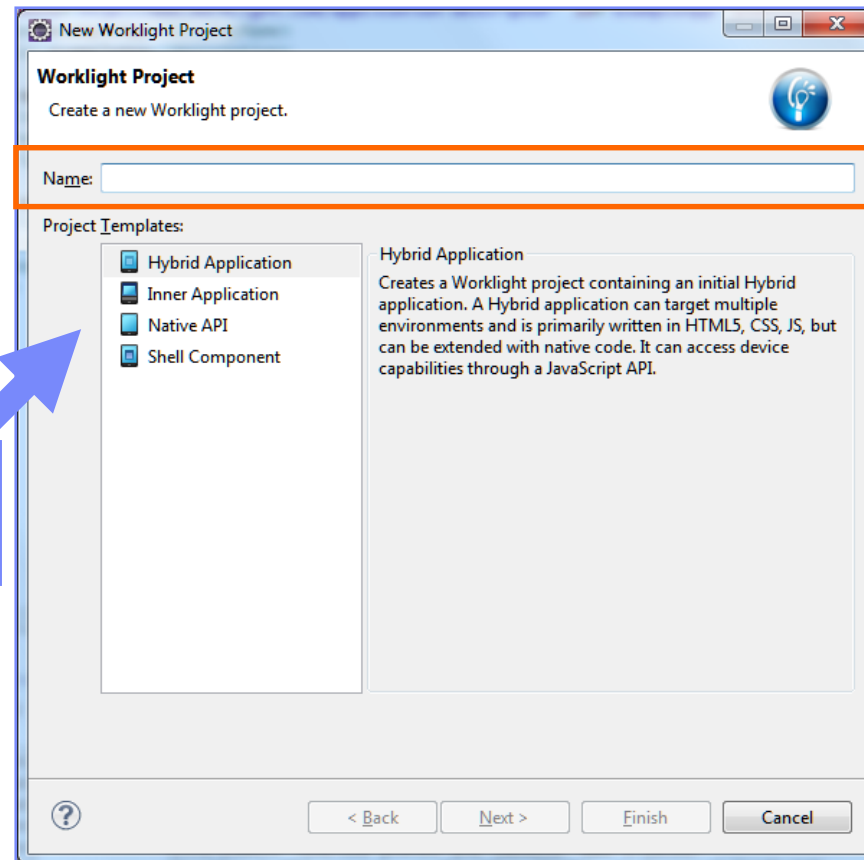- Use Worklight Studio to deploy the application

- Exercise

Hello Worklight Application

- Create a Worklight® project.
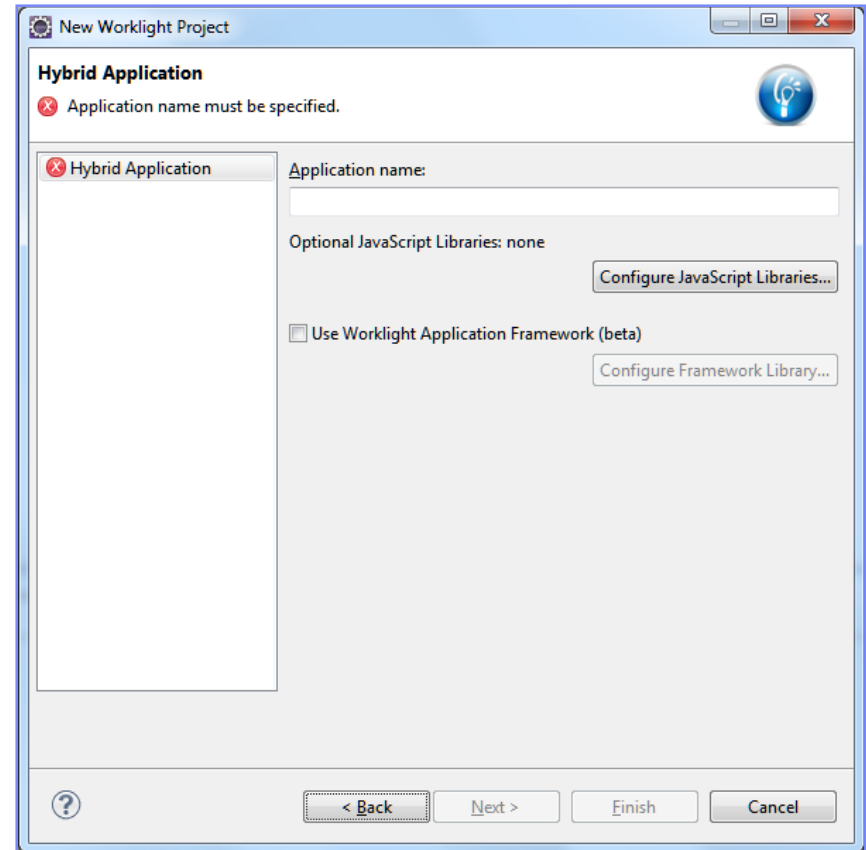
Hello Worklight Application

- Name it "HelloWorklightProject" and select a **Hybrid Application** template.

Select the required application type.
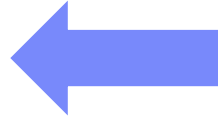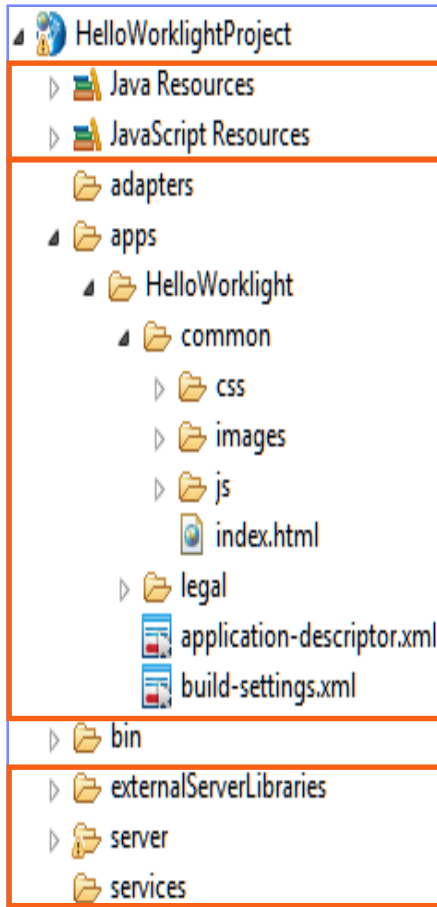
Hello Worklight Application

- Name your application "HelloWorklight".

- JavaScript™ frameworks can be added to your project in this screen.

- To learn more about adding frameworks to your application, read the ***Client-side development basics – Working with UI frameworks*** Getting Started training module.

- Click **Finish** when done.

Agenda

- Write your first application

- Become familiar with the application files and HTML structure

- Use Worklight Studio to deploy the application

- Exercise

Worklight Project Structure

HelloWorklightProject
- Java Resources
- JavaScript Resources
- adapters
- apps
  - HelloWorklight
    - common
      - css
      - images
      - js
      - index.html
    - legal
    - application-descriptor.xml
    - build-settings.xml
- bin
- externalServerLibraries
- server
- services

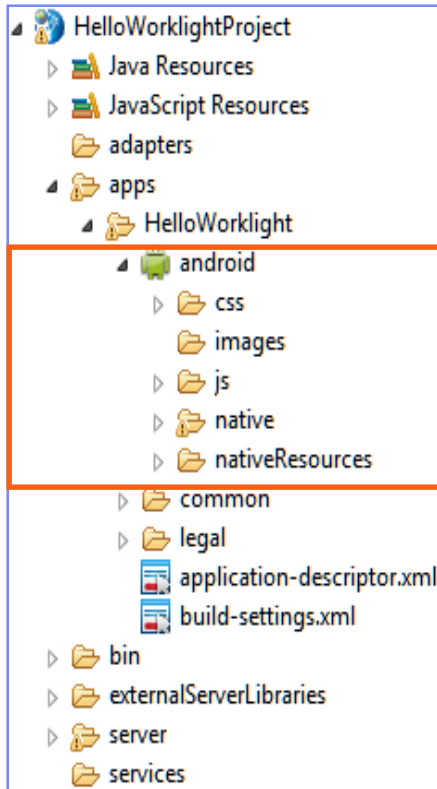**References required for application development and deployment**
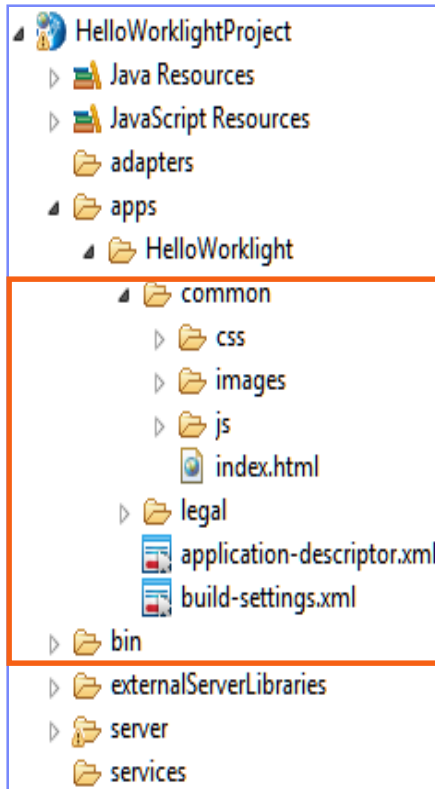
Project applications and adapters

Server customization components

HelloWorklight – Project Structure

- The default environment is called `common`.

- The `common` environment contains all the resources that are shared between environments.

- Right-click **apps** and select **New > Worklight environment** to add new environments:

  - A new environment is created.

  - The resources of the new environment have the following relationship with the common resources:

    - `images` – overrides the common images when both have the same name.

    - `css` – extends, overrides, or both, the common CSS files.

    - `js` – extends the common application instance JS object. The environment class extends the common app class.

    - `index.html` – optional HTML file that overrides the common HTML code when both have the same name.

HelloWorklight – Project Structure - continued

HelloWorklightProject
- Java Resources
- JavaScript Resources
- adapters
- apps
  - HelloWorklight
    - common
      - css
      - images
      - js
      - index.html
    - legal
    - application-descriptor.xml
    - build-settings.xml
  - bin
- externalServerLibraries
- server
- services

- Common Environment:
  - `index.html`: The main HTML file.
  - `css`: *main.css* – main application CSS file.
  - `images`: Default Worklight images for the common environment.
  - `js`:
    - `initOptions.js`: Contains initialization options for the application.
    - `main.js`: The main JavaScript file for the application.
    - `messages.js`: JSON object that holds all app messages. Can be used as the source for translation.

- The `legal` folder holds all legal related docs.

- The `application-descriptor.xml` file contains the application metadata.

- The `build-settings.xml` file contains configuration options for minification and concatenation.

- The `bin` folder contains generated `.wlapp` and `.wladapter` files that can be used to deploy to a remote Worklight Server.

HelloWorklight – Project Structure - continued
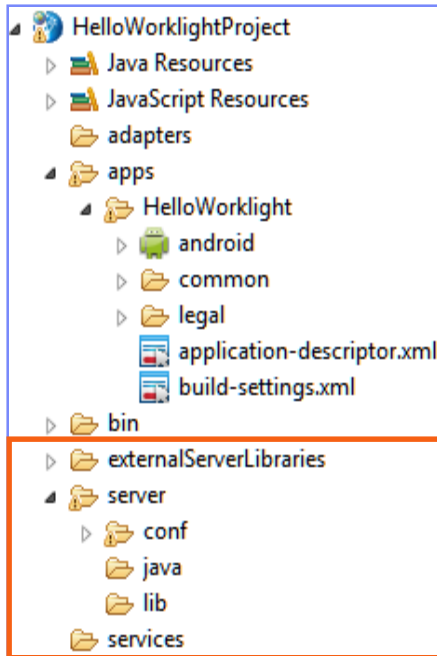
- The `externalServerLibraries` folder contains the libraries to be placed in external service servers and used for access token validation (by the service).

- The `server` folder contains files that are used for server-side customization of a project:
  - The `conf` folder contains:
    - `authenticationConfig.xml`, which defines authentication realm and security tests.
    - `default.keystore`, a default SSL certificate that is provided by Worklight.
    - `login.html,` which presents a login form for web environments and Worklight Console.
    - `SMSConfig.xml`, which defines SMS Gateways.
    - worklight.properties, which defines the properties that are used by the Worklight Server.
  - The `java` folder is used to hold Java™ classes that will be compiled and deployed to a Worklight Server after you build your application. You can put your custom Java code here.
  - The `lib` folder is used for JAR files that are deployed to the server.

- The `services` folder contains any back-end services that were discovered.

HelloWorklight – Project Structure - continued
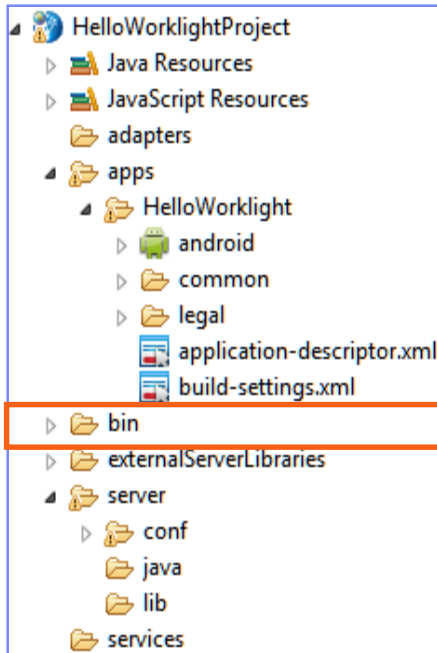
- The `bin` folder contains project artifacts that are deployed to the Worklight Server.

- The evaluation version of Worklight Studio deploys those artifacts to the embedded Worklight Development Server automatically as a part of the build process.

  - `.wlapp` files are application bundles.

  - `.wladapter` files are adapters.

  - `.jar` and `.war` files are server customization files that contain `worklight.properties`, `authenticationConfig.xml`, and custom Java code.

HelloWorklight – application-descriptor.xml

- An application descriptor is an XML file that stores the metadata for an application.

- This file can be edited with the Design or Source editors.
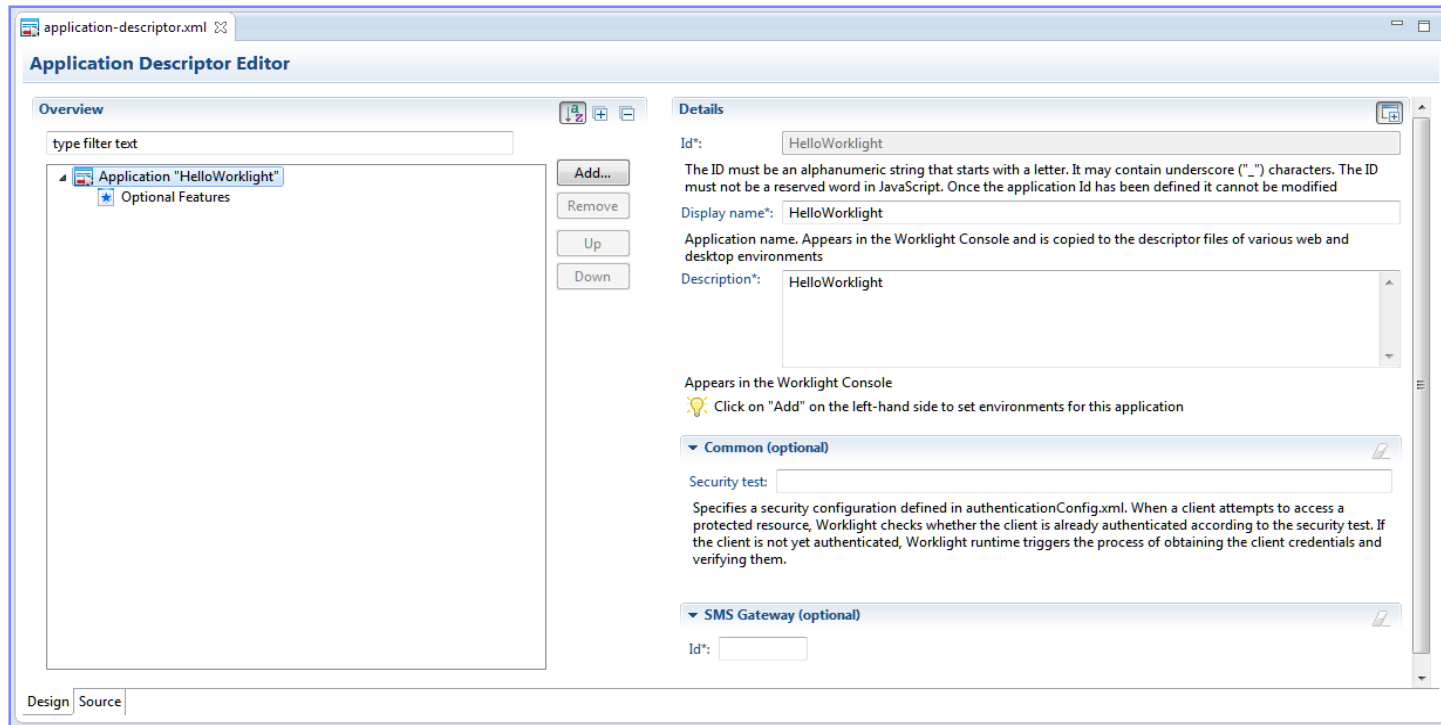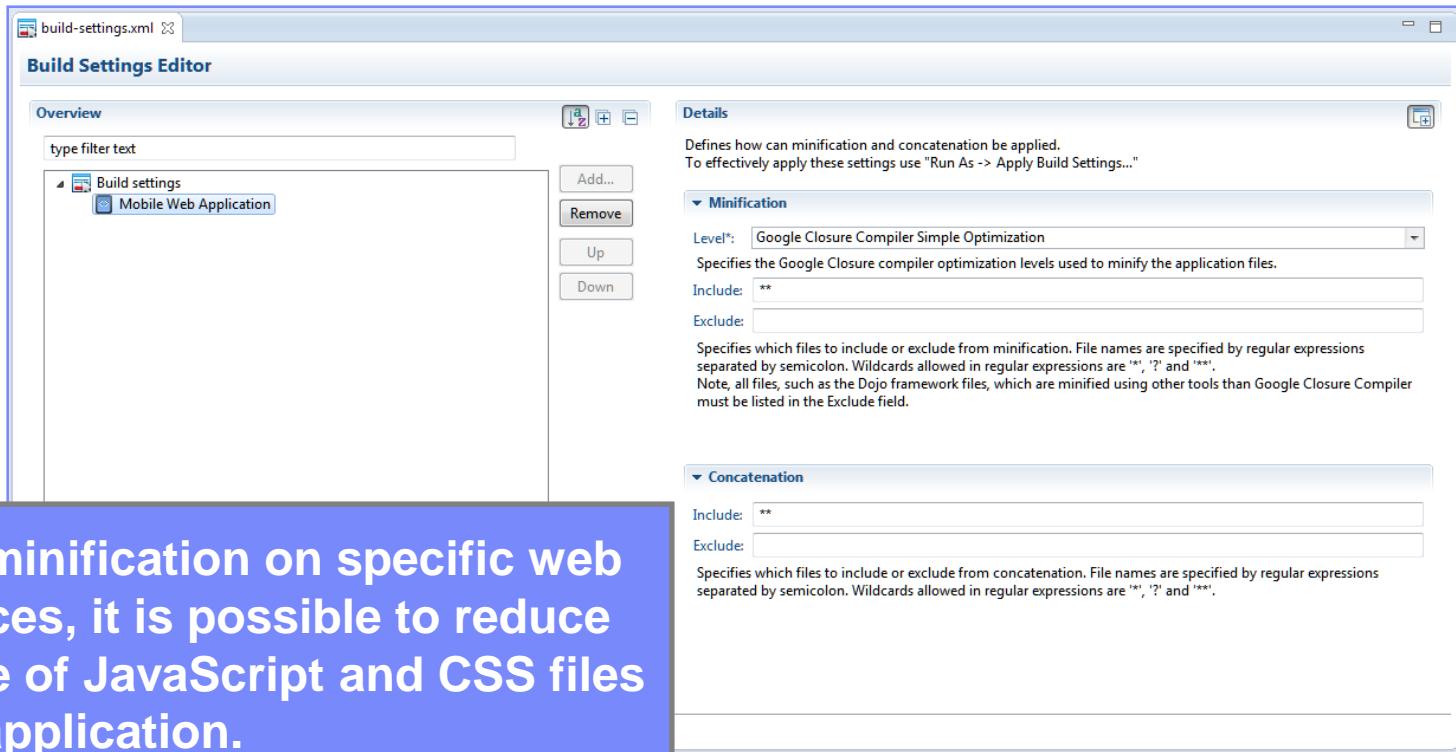
HelloWorklight - application-descriptor.xml - continued

- Based on the W3C Widget Packaging and Configuration.

- Contains application properties that are used at build time.

```xml
<!-- Attribute "id" must be identical to application folder name -->
<application id="HelloWorklight"
    xmlns="http://www.worklight.com/application-descriptor"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

    <displayName>HelloWorklight</displayName>
    <description>HelloWorklight</description>
    <author>
        <name>application's author</name>
        <email>application author's e-mail</email>
        <copyright>Copyright My Company</copyright>
        <homepage>http://mycompany.com</homepage>
    </author>
    <mainFile>index.html</mainFile>
    <thumbnailImage>common/images/thumb
    <features>
    </features>
</application>
```

You can specify the application name, description, author details, and thumbnail image to be displayed in the Worklight Console.

HelloWorklight - application-descriptor.xml - continued

- Based on the W3C Widget Packaging and Configuration.

- Contains application properties that are used at build time.

```xml
<iphone bundleId="com.HelloWorklight" version="1.0">
    <worklightSettings include="true"/>
    <security>
        <encryptWebResources enabled="false"/>
        <testWebResourcesChecksum enabled="false" ignoreFileExtensions="png, jpg, jpeg, gif, mp4, mp3"/>
    </security>
</iphone>
<android version="1.0">
    <worklightSettings include="true"/>
    <security>
        <encryptWebResources enabled="false"/>
        <testWebResourcesChecksum enabled="false" ig
        <publicSigningKey>Replace this text with the
    </security>
</android>
```

Environment-specific information is inserted automatically as new environments are added to the Worklight project.

HelloWorklight – build-settings.xml

- The Build Settings is an XML file that contains configuration options for minification and concatenation of the Desktop Browser and Mobile Web environments' web resources.

- This file can be edited with the Design or Source editors.

Using minification on specific web resources, it is possible to reduce the size of JavaScript and CSS files in the application.

HelloWorklight – build-settings.xml - continued

- The Build Settings is an XML file that contains configuration options for minification and concatenation of the Desktop Browser and Mobile Web environments' web resources.

- This file can be edited with the Design or Source editors.
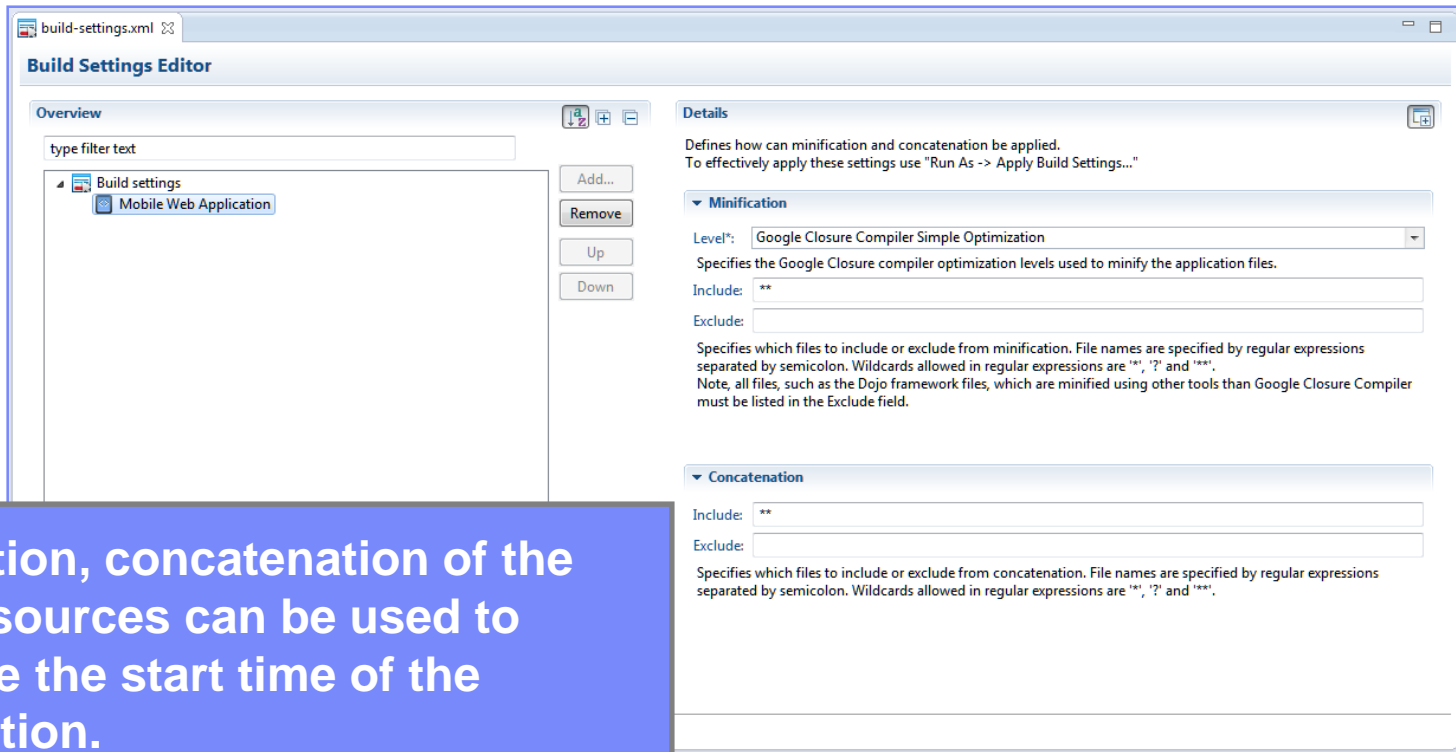
In addition, concatenation of the web resources can be used to improve the start time of the application.

HelloWorklight – Single DOM model

- Worklight hybrid applications use a single DOM model.

- The single DOM model means never navigate between various HTML files by using hyperlinks or by changing the `window.location` property.

- Instead, you must implement multi-page interfaces by loading an external HTML file content by using Ajax requests and injecting it into an existing DOM.

  – You must do so because the main application HTML file loads Worklight client-side JavaScript framework files. After the browser navigates from one HTML file to another, the JavaScript context and loaded scripts are lost.

- Most JavaScript UI frameworks available today (for example, jQuery Mobile, Dojo Mobile, Sencha Touch) provide an extensive range of APIs to achieve the required multi-page navigation.

- This module explains the principles of a single-page application.

- Principles of multi-page applications that are built with a single DOM model are explained in later modules.

Index.html

- During application run time, the main HTML document cannot be replaced by another HTML document.

```
<!DOCTYPE HTML>
<html>
    <head>
        <meta charset="UTF-8">
        <title>HelloWorklight</title>
        <meta name="viewport" content="width=devi
        <link rel="shortcut icon" href="images/fav
        <link rel="apple-touch-icon" href="images/
        <link rel="stylesheet" href="css/main.css
        <script>window.$ = window.jQuery = WLJQ;</script>
    </head>
    <body style="display: none;">
        <!--application UI goes here-->
        Hello Worklight
        <script src="js/initOptions.js"></script>
        <script src="js/main.js"></script>
        <script src="js/messages.js"></script>
    </body>
</html>
```

Default application HTML template complies with HTML5 standard markup, but any other DOCTYPE can be specified.

Index.html - continued

- During application run time, the main HTML document cannot be replaced by another HTML document.

```html
<!DOCTYPE HTML>
<html>
    <head>
        <meta charset="UTF-8">
        <title>HelloWorklight</title>
        <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0, user-scalable=0">
        <link rel="shortcut icon" href="images/favicon.png">
        <link rel="apple-touch-icon" href="images/apple-touch-icon.png">
        <link rel="stylesheet" href="css/main.css">
        <script>window.$ = window.jQuery = WLJQ;</script>
    </head>
    <body style="display: none;">
        <!--application UI goes here-->
        Hello Worklight
        <script src="js/initOptions.js"></script>
        <script src="js/main.js"></script>
        <script src="js/messages.js"></script>
    </body>
</html>
```

Worklight client-side framework uses jQuery library for internal functions. By default, the $ char is assigned to the internal jQuery in the main JavaScript file.

Remove this line if your application does not require jQuery, or if you want to use your own version of jQuery.

Index.html - continued

- During application run time, the main HTML document cannot be replaced by another HTML document.

```html
<!DOCTYPE HTML>
<html>
    <head>
        <meta charset="UTF-8">
        <title>HelloWorklight</title>
        <meta name="viewport" content="width=device-
        <link rel="shortcut icon" href="images/favic
        <link rel="apple-touch-icon" href="images/ap
        <link rel="stylesheet" href="css/main.css">
        <script>window.$ = window.jQuery = WLJQ;</sc
    </head>
    <body style="display: none;">
        <!--application UI goes here-->
        Hello Worklight
        <script src="js/initOptions.js"></script>
        <script src="js/main.js"></script>
        <script src="js/messages.js"></script>
    </body>
</html>
```

Worklight client framework initialization is bound to the `onload` **event specified in the initOptions.js file. For more information about the initialization options, see the Worklight user documentation.**

Index.html - continued

- During application run time, the main HTML document cannot be replaced by another HTML document.

```html
<!DOCTYPE HTML>
<html>
    <head>
        <meta charset="UTF-8">
        <title>HelloWorklight</title>
        <meta name="viewport" content="width=device-widt
        <link rel="shortcut icon" href="images/favicon.png">
        <link rel="apple-touch-icon" href="images/apple-touch-icon.png">
        <link rel="stylesheet" href="css/main.css">
        <script>window.$ = window.jQuery = WLJQ;</script>
    </head>
    <body style="display: none;">
        <!--application UI goes here-->
        Hello Worklight
        <script src="js/initOptions.js"></script>
        <script src="js/main.js"></script>
        <script src="js/messages.js"></script>
    </body>
</html>
```

The arrow indicates where to insert your HTML code.

initOptions.js

- The `initOptions.js` file contains Worklight framework initialization settings.

- It is also responsible for initializing the Worklight framework once the `<body>` element finishes loading.

- By default, the Worklight application starts in offline mode (the application does not attempt to connect to the Worklight Server).

  – To connect to the Worklight Server, use `WL.Client.connect()`

- Some default initialization options are documented in the file itself, with the entire set of options available in the Worklight user documentation topic for the API method "`WL.Client.init`".

Main.js

- When you create an application, a `main.js` file is created and holds its JavaScript portion.

- It contains a `wlCommonInit()` function that is invoked automatically after the Worklight framework initialization finishes.

- You can add your application initialization code there.

- This function is used in environment-specific JavaScript files to have a common initialization start point. There are more details in subsequent Getting Started modules.

```
function wlCommonInit(){
    // Common initialization code goes here
}
```

Main.js - continued

- As discussed previously, the Worklight application starts in offline mode by default.

- To begin communicating with Worklight Server, you need to follow the instructions in the default `wlCommonInit()` function.

```
function wlCommonInit(){
    /*
     * Use of WL.Client.connect() API before any connectivity to a Worklight Server is required.
     * This API should be called only once, before any other WL.Client methods that communicate with the Worklight Server.
     * Don't forget to specify and implement onSuccess and onFailure callback functions for WL.Client.connect(), e.g:
     *
     *     WL.Client.connect({
     *             onSuccess: onConnectSuccess,
     *             onFailure: onConnectFailure
     *     });
     *
     */

    // Common initialization code goes here

}
```

Agenda

- Write your first application

- Become familiar with the application files & HTML structure

- Use Worklight Studio to deploy the application

- Exercise

Building an Application
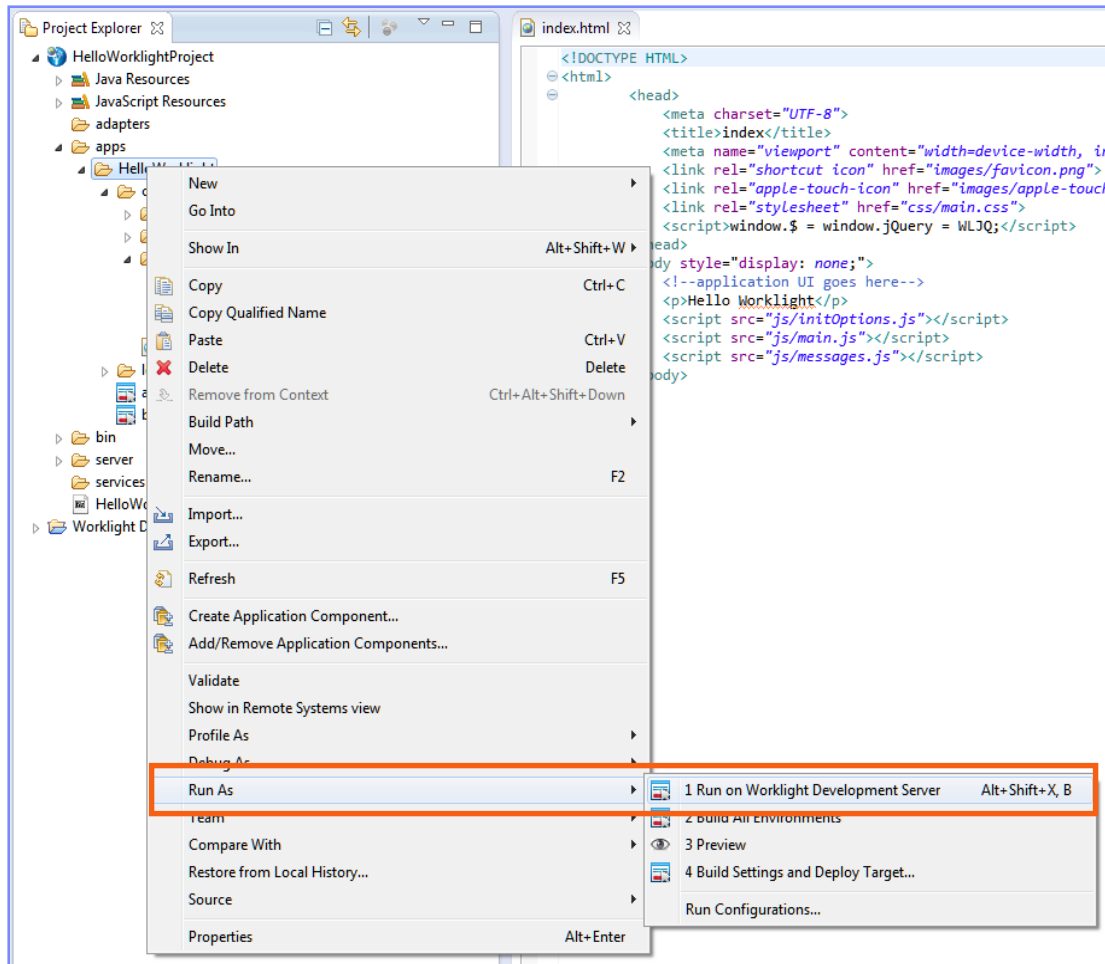
- Select an app to build, and then right-click the app name.

- Click **Run As > Run on Worklight Development Server**.

- While the application is being built and deployed, you can watch its progress in the Eclipse Console view.

- After the build completes, the application is available for preview in the **catalog** tab of the Worklight Console, as explained in the next slide.

- By default the port number is **10080**, and the context root is the project name.

Worklight Console

- To view the Worklight Console, right-click the project name and select **Open Worklight Console**.

- To view Worklight Console in an external browser window, from the top menu bar in Eclipse go to **Window > Preferences > General > Web Browser** and select the "Use external web browser" radio button. There you can also select your preferred browser.

Worklight Console - continued

- To preview the application in its current form, click "Preview as Common Resources".

Worklight Console - continued

- Since IBM Worklight Foundation V6.2.0.0, the Worklight Console URL is changed:

  – For example: **http://localhost:10080/worklightconsole**.

  – Any deployed project appears in a menu option within the console.

- The various features of Worklight Console are gradually discovered and explained as you progress through the Getting Started modules.

Build Settings and Deploy Target

- To change the target server, right-click the application name and select **Run As > Build Settings and Deploy Target**.

- Select the "Build the application to work with a different Worklight server" check box.

- Update the server and context path and click **OK**.

- Next, you must now rebuild the application.

- **Note:** Only Apache Tomcat and WebSphere® Application Server Liberty Profile are supported.
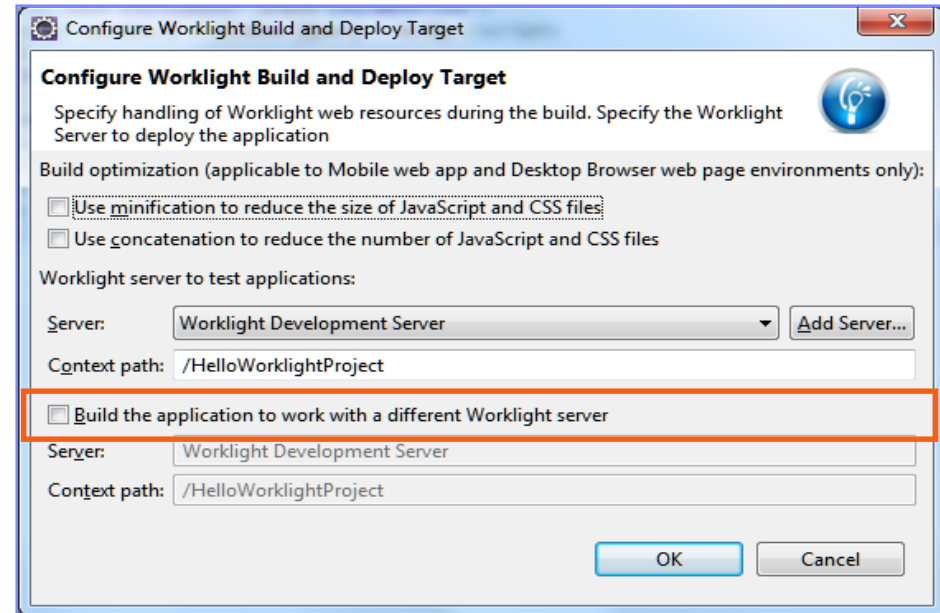
Build Settings and Deploy Target - continued

- If minification, concatenation, or both are required for the application, be sure to also enable this build setting before you build the application.

- To do so, right-click the application name and select **Run As > Build Settings and Deploy Target**.

- Choose one or all of the highlighted options.

- Click **OK**.
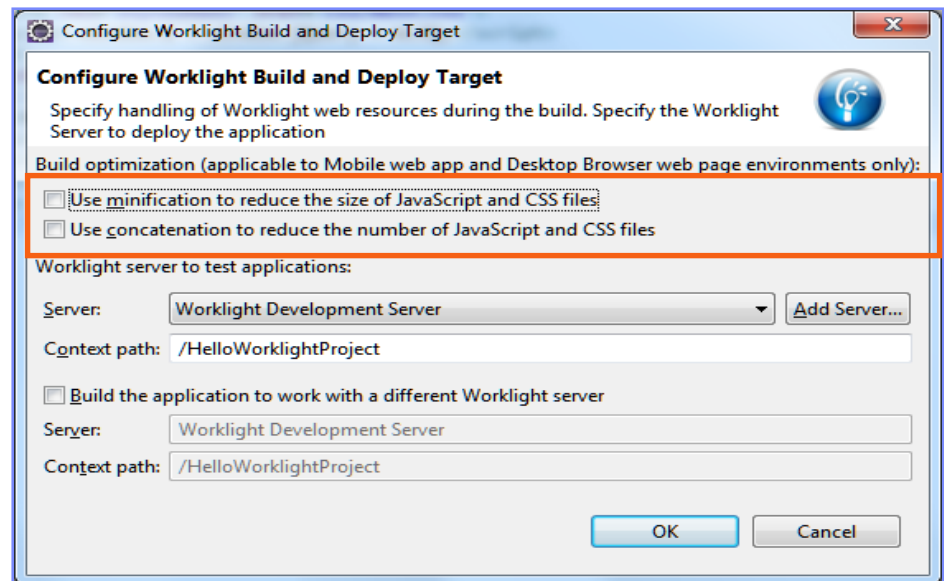
Agenda

- Write your first application

- Become familiar with the application files & HTML structure

- Use Worklight Studio to deploy the application

- Exercise

Exercise

- HelloWorklight Application

  – Use the wizard to create a **HelloWorklightProject** project.

  – Use the wizard to create an application; name it **HelloWorklight.**

  – Use the wizard to add an environment.

  – Use Worklight Studio to deploy your application.

  – Use Worklight Console to preview your application.
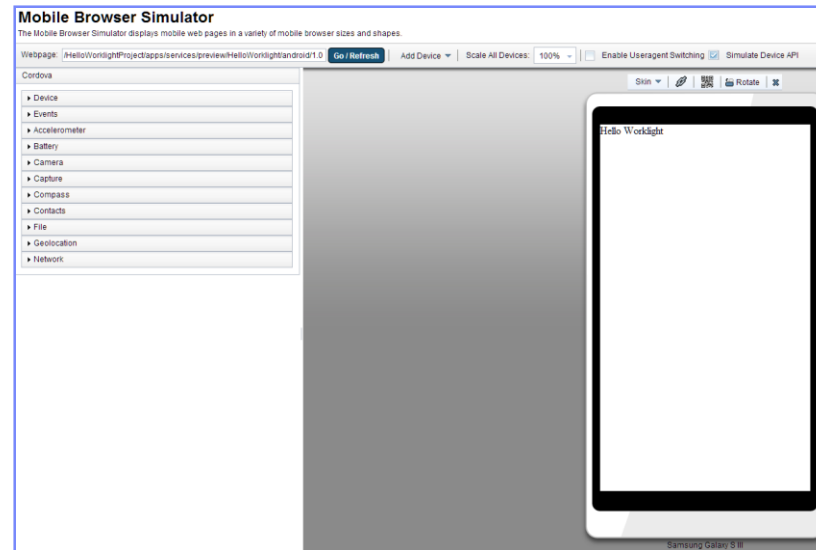
Exercise solution

- The sample for this module can be found on the Getting Started page of the Worklight documentation website at http://www.ibm.com/mobile-docs.

- The sample project contains some CSS styling. A newly created project does not contain any styling.

- To import the solution to Eclipse, right-click in a blank area in the Project Explorer view. Choose **Import > Import > General > Existing Projects into Workspace** and either select the archive (compressed file) or root directory (if decompressed).
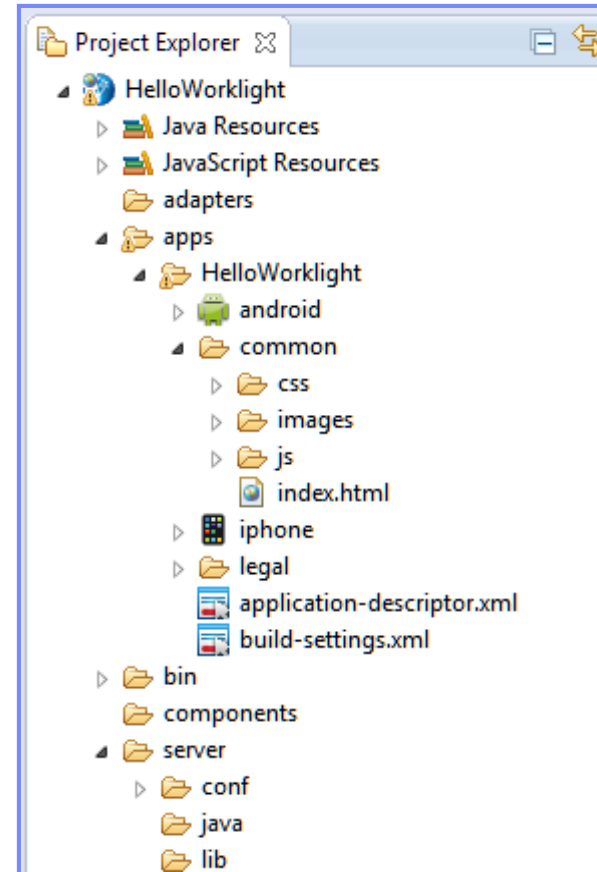
Quiz

Test your knowledge. Answers are in the following slide.

- What is the default application structure?
  - A single HTML file, and a number of CSS and JS files.
  - A number of HTML and JS files, and a single CSS file.
  - A number of HTML, JS, and CSS files.
  - A single HTML, single CSS, and a single JS file.
- Which of the following server-side components cannot be added as part of a Worklight Project?
  - Source code of custom server-side Java classes.
  - Properties that are used by the server.
  - JAR files that contain server-side Java code.
  - JMeter application to test server performance.
- The correct place to specify the URL that the application uses is:
  - authenticationConfig.xml.
  - worklight.properties.
  - application-descriptor.xml.
  - This URL is set separately per platform in a dedicated settings file.

Quiz - answers

- What is the default application structure?
  – A single HTML file, and a number of CSS and JS files.
  – A number of HTML and JS files, and a single CSS file.
  – A number of HTML, JS, and CSS files.
  – A single HTML, single CSS, and a single JS file.
- Which of the following server-side components cannot be added as part of a Worklight Project?
  – Source code of custom server-side Java classes.
  – Properties that are used by the server.
  – JAR files that contain server-side Java code.
  – JMeter application to test server performance.
- The correct place to specify the URL that the application uses is:
  – authenticationConfig.xml.
  – worklight.properties.
  – application-descriptor.xml.
  – This URL is set separately per platform in a dedicated settings file.

Notices

Support and comments

- For the entire IBM Worklight documentation set, training material and online forums where you can post questions, see the IBM website at:
  - http://www.ibm.com/mobile-docs
- **Support**
  - Software Subscription and Support (also referred to as Software Maintenance) is included with licenses purchased through Passport Advantage and Passport Advantage Express. For additional information about the International Passport Advantage Agreement and the IBM International Passport Advantage Express Agreement, visit the Passport Advantage website at:
    - http://www.ibm.com/software/passportadvantage
  - If you have a Software Subscription and Support in effect, IBM provides you assistance for your routine, short duration installation and usage (how-to) questions, and code-related questions. For additional details, consult your IBM Software Support Handbook at:
    - http://www.ibm.com/support/handbook
- **Comments**
  - We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this document. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.
  - For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.
  - When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state.
  - Thank you for your support.
  - Submit your comments in the IBM Worklight Developer Edition support community at:
    - https://www.ibm.com/developerworks/mobile/worklight/connect.html
  - If you would like a response from IBM, please provide the following information:
    - Name
    - Address
    - Company or Organization
    - Phone No.
    - Email address

Thank You