Exercise 1. Bluemix and the Cloud Foundry command-line interface (CLI)

What this exercise is about

In this exercise, you sign on to Bluemix and create an application. You run the application from a web browser. Next, you install the Cloud Foundry command-line tool, download the starter package to the workstation, and extract the package. You then connect to Bluemix from the command line and redeploy the application to Bluemix.

What you should be able to do

After completing this exercise, you should be able to:

- Sign on to Bluemix from a browser session
- Create a Bluemix application from an existing template
- Add a service to the application from the service catalog
- Test the application with the resource endpoint after the application is started
- Follow the “getting started” option on Bluemix to use the CLI
- Install the Cloud Foundry CLI
- Download the package
- Sign on to Bluemix from the CLI

Introduction

IBM Bluemix is an open-standard, cloud platform for building, running, and managing applications. With Bluemix, you focus on rapidly building compelling user experiences with flexible compute options, choice of DevOps tools, and a powerful set of IBM and third-party APIs and services. Learn how to quickly set up and deploy your first Bluemix application that is developed on your own workstation.

Requirements

To complete this exercise, you must install the Cloud Foundry command-line interface on a compatible Microsoft Windows workstation. For a list of compatible operating systems, see:
`https://github.com/cloudfoundry/cli`

Exercise instructions

Preface

This exercise depends on the availability of the IBM Bluemix cloud and connectivity from your own workstation to the cloud environment.

Part 1: Set up your IBM Bluemix account

You must sign up and create your IBM Bluemix and IBM Bluemix DevOps Services accounts before you start this exercise. The IBM Bluemix account provides a runtime environment for your IBM SDK for Node.js application.

Important

Complete the steps in the tutorial to sign up for an IBM Bluemix and IBM Bluemix DevOps Services account with your IBM ID. Review the “Getting Started with Bluemix and Bluemix DevOps Services with Node.js” tutorial.

`https://hub.jazz.net/tutorials/jazzeditor/`

Part 2: Explore the dashboard

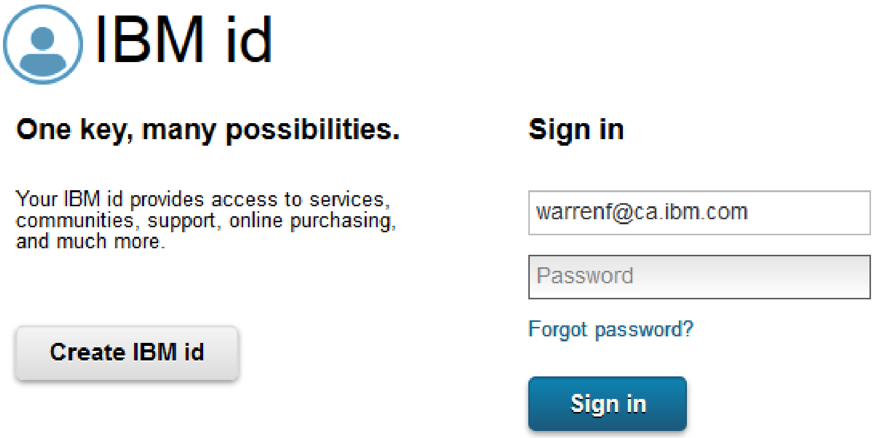
The IBM Bluemix dashboard is the web application interface to your online account. Through the dashboard, you can manage memory and resource settings for your services and applications. In this section, log in and explore the dashboard.

__ 1. Open the IBM Bluemix website (`http://bluemix.net`) in a web browser.

__ 2. Log on to IBM Bluemix.

  __ a. In the IBM Bluemix home page, click **Log in**.

  __ b. Enter your IBM ID and password.



IBM id

One key, many possibilities.

Your IBM id provides access to services, communities, support, online purchasing, and much more.

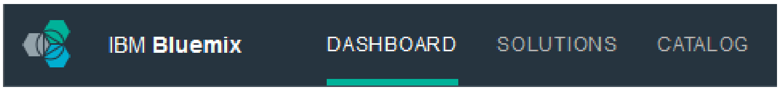
Sign in

warrenf@ca.ibm.com

  __ c. Select **Sign in**.

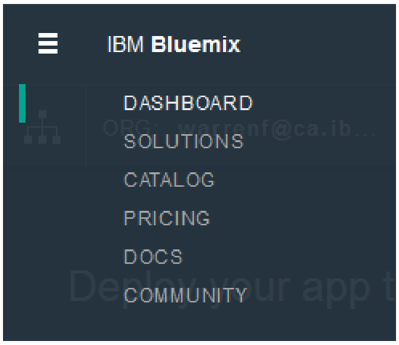
__ 3. Examine the dashboard.

__ a. From the IBM Bluemix menu, click **Dashboard**.



Note

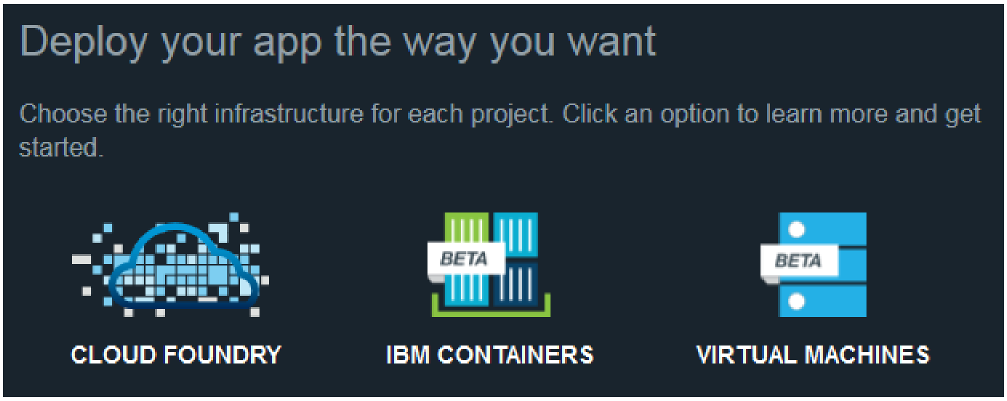
The IBM Bluemix website uses a responsive design interface: the web page layout changes according to the web browser size. For smaller displays, the website hides the menu items in a menu.



If you do not see the **Dashboard** link, increase the width of your web browser window.

__ 4. Examine the computing infrastructure for your application.

__ a. In the IBM Bluemix dashboard, examine the infrastructure types that are available to you.



Deploy your app the way you want

Choose the right infrastructure for each project. Click an option to learn more and get started.

IBM Bluemix supports three types of infrastructure:

- **Cloud Foundry** technology provides a computing infrastructure for your server runtime environments, services, and applications. IBM Bluemix takes care of the management and maintenance of the infrastructure that runs your applications.
- **IBM Containers** provide more fine-grained control over the computing infrastructure than run your application and services. A *container* is an object that holds everything

that your applications require to run. You can use and extend public images from the IBM Bluemix catalog. You can also use images from the public Docker hub.

IBM containers run Docker containers in a hosted cloud environment. Docker adds an engine to deploy an application to the virtual environment. Docker also provides an environment to run your application.

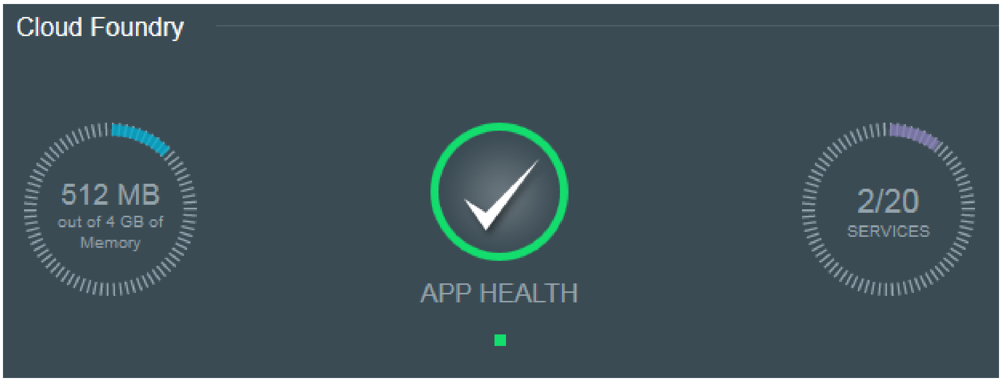
- A **virtual machine** is a software implementation of a computer that runs applications like a real workstation. You can configure the operating system, server runtime environment, and application.

  You can deploy and manage your virtual machines with the Bluemix user interface or OpenStack API from the cloud.

Note

In this exercise, you focus on building an application on a Cloud Foundry infrastructure. For more information about the IBM Container and virtual machine infrastructures, see the corresponding lectures in the course.

__ b. View your IBM Bluemix Cloud Foundry environment health.



The Cloud Foundry section of the dashboard displays the memory available for applications and services. The dashboard also displays the number of services that are running in the environment. In the middle of the dashboard, the App Health indicator shows you whether one or more applications stopped running.

The amount of memory and the number of services available to you depends on your IBM Bluemix subscription.

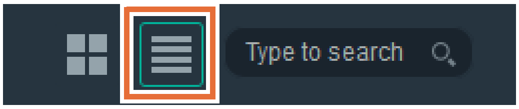
Note

Depending on the number of applications and services in your account, your environment health dashboard might not match the screen capture in this step.

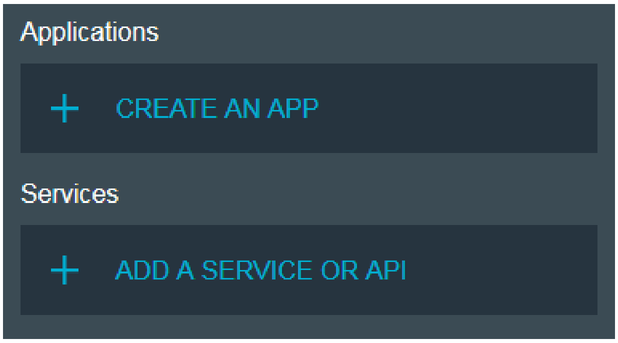
__ 5. Examine the list of applications and services in your IBM Bluemix account.

__ a. Locate the search bar for applications and services at the upper-right corner of the dashboard.

__ b. Change the dashboard view to a list view.



__ c. Examine the list of applications and services.



Applications are programs that you develop in the IBM Bluemix Cloud Foundry environment. You build your application to run on one of the runtime environments from Cloud Foundry.

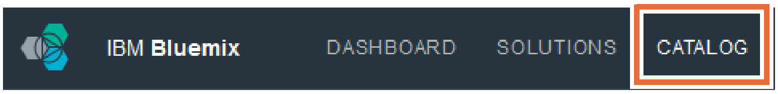
Services are pre-built components for your applications. For example, analytics packages, databases, and mobile frameworks are some of the services from IBM Bluemix.

Part 3: Create an application

The IBM Bluemix catalog lists components and services that help you build your application. In this part, create an IBM Bluemix application with the IBM SDK for Node.js run time.

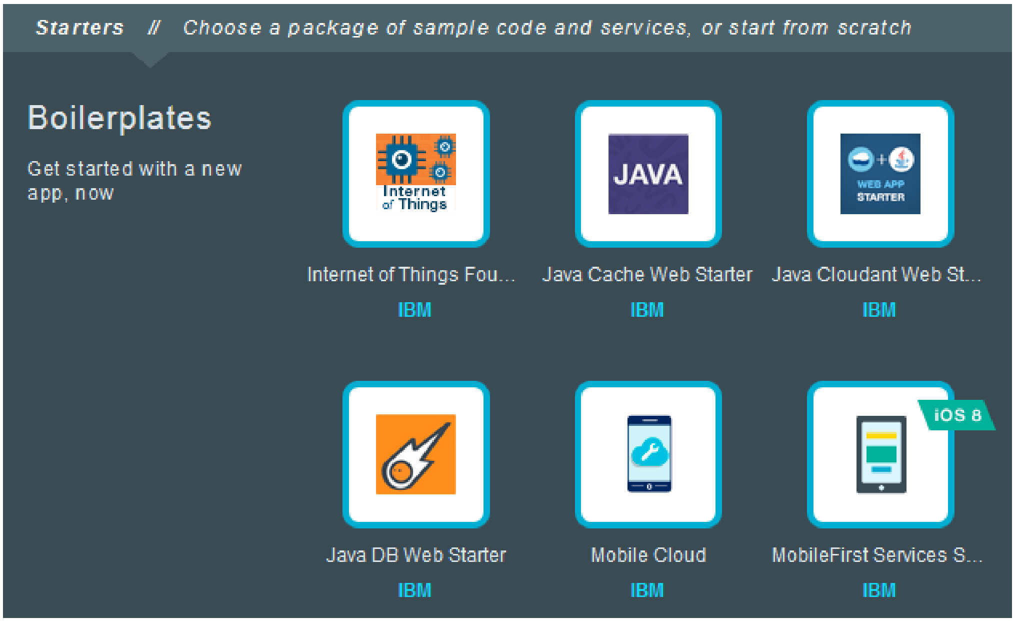
__ 1. Open the IBM Bluemix catalog.

__ a. In the IBM Bluemix website, click **Catalog** from the menu bar.



__ 2. Examine the boilerplate, runtime, and services in the catalog.

__ a. Examine the list of boilerplate catalog items.



Starters // *Choose a package of sample code and services, or start from scratch*

Boilerplates

Get started with a new app, now

Internet of Things Fou... IBM

Java Cache Web Starter IBM

Java Cloudant Web St... IBM

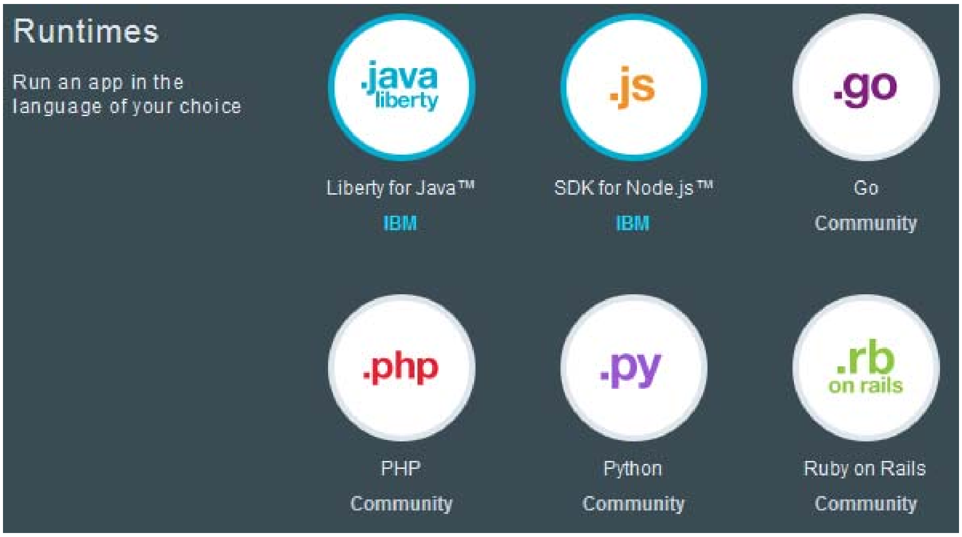
Java DB Web Starter IBM

Mobile Cloud IBM

MobileFirst Services S... IBM

Boilerplates are packages of sample applications with the Bluemix services. When you select a boilerplate, IBM Bluemix configures the services on which the application relies. IBM Bluemix also provides you with the source code and documentation for the sample application.

__ b. Examine the list of runtime environments in the catalog.



Runtimes

Run an app in the language of your choice

Liberty for Java™ IBM

SDK for Node.js™ IBM

Go Community

PHP Community

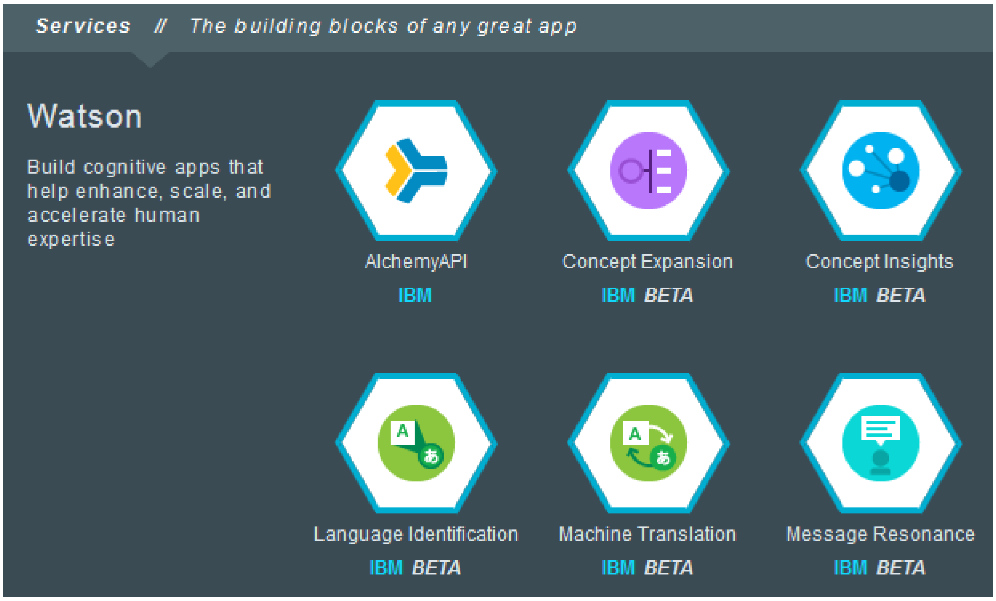
Python Community

Ruby on Rails Community

To create your own application, select the server runtime environment as a starting point. You are responsible for setting up the IBM Bluemix services for your application.

The runtime environments that IBM supports are listed as “IBM” runtimes. The “community” runtimes rely on open source projects that external groups manage.

__ c. Examine the list of IBM Bluemix services in the catalog.

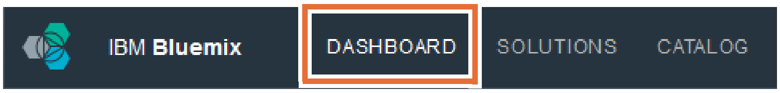


Services are extensions to the cloud environment that IBM Bluemix hosts. The service provides functions that are ready-for-use by the app's running code. The predefined services that are provided by Bluemix include database, messaging, push notifications for mobile apps, and elastic caching for web apps.

Bluemix simplifies the use of services by provisioning new instances of the service, and binding those service instances to your application. IBM Bluemix manages the services for you.

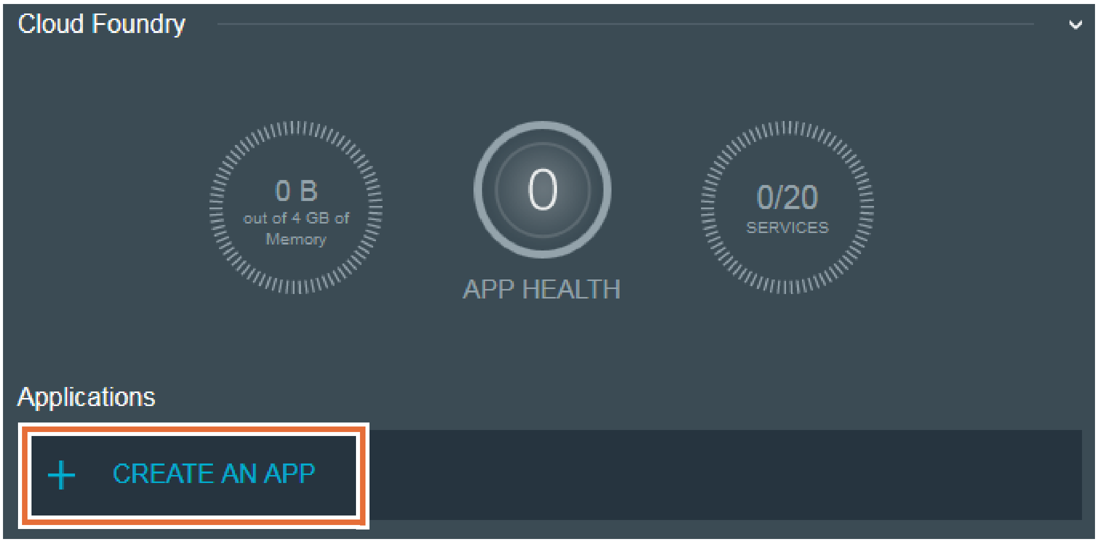
__ 3. Create an application with an instance of the IBM SDK for Node.js runtime environment.

__ a. Switch to the IBM Bluemix dashboard view.

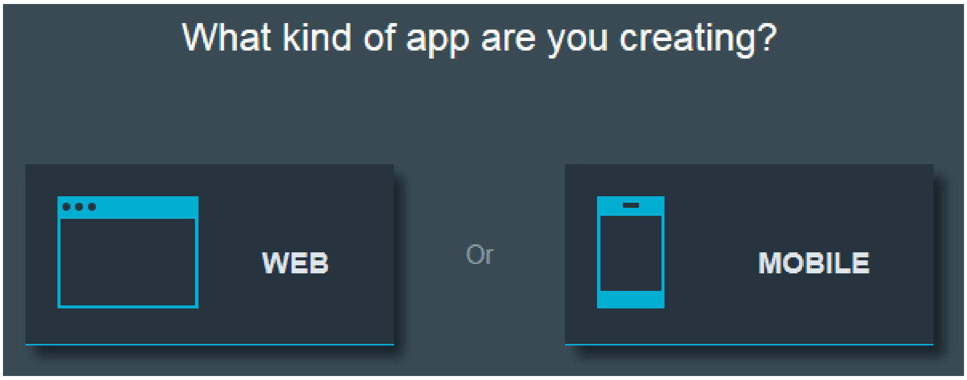


__ b. Scroll down to the **Applications** section.

__ c. Click **Create an app**.

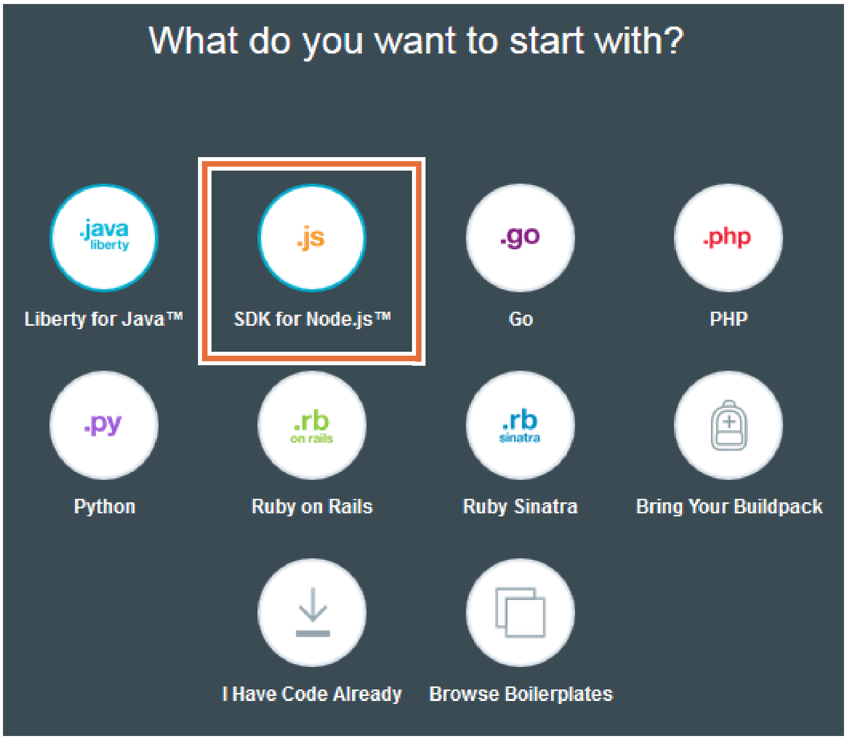


__ d. The application wizard prompts whether you want to build a web application or a mobile application. Click **Web**.



__ e. On the next page, you choose a starting point for your project in the application wizard. You can create an empty application space with a server runtime environment. You can also select an application buildpack that you created or a sample application.

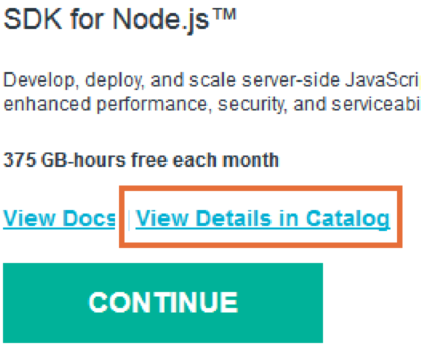
Click **SDK for Node.js**.



What do you want to start with?

__ 4. Examine the plan details for the SDK for Node.js runtime environment.

__ a. The SDK for Node.js page describes the qualities the cost for hosting the runtime environment. For more details on the pricing, click **View Details in Catalog**.

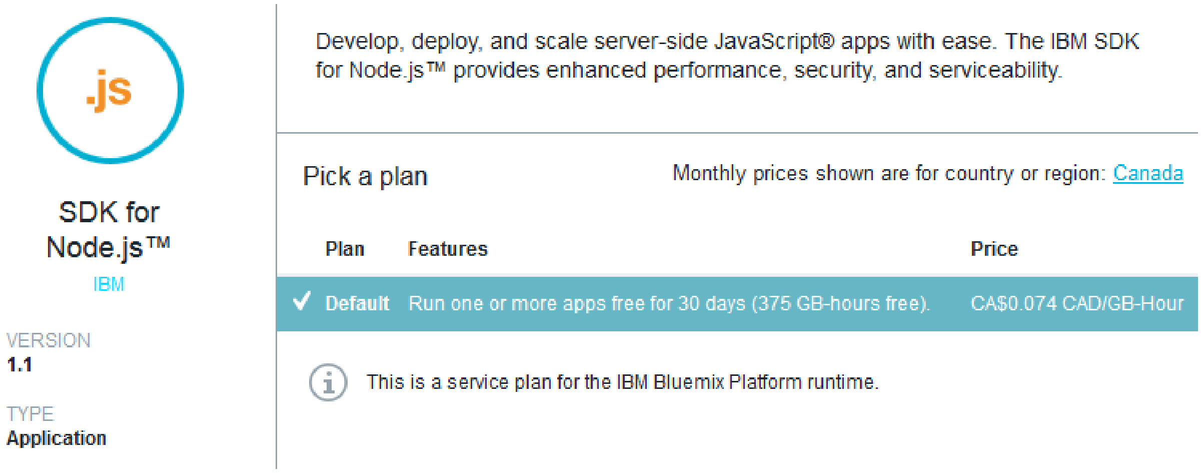


SDK for Node.js™

Develop, deploy, and scale server-side JavaScript® apps with ease. The IBM SDK for Node.js™ provides enhanced performance, security, and serviceability.

375 GB-hours free each month

__ b. Examine the plan pricing details.



.js

SDK for Node.js™

IBM

VERSION
1.1

TYPE
Application

Develop, deploy, and scale server-side JavaScript® apps with ease. The IBM SDK for Node.js™ provides enhanced performance, security, and serviceability.

Pick a plan

Monthly prices shown are for country or region: Canada

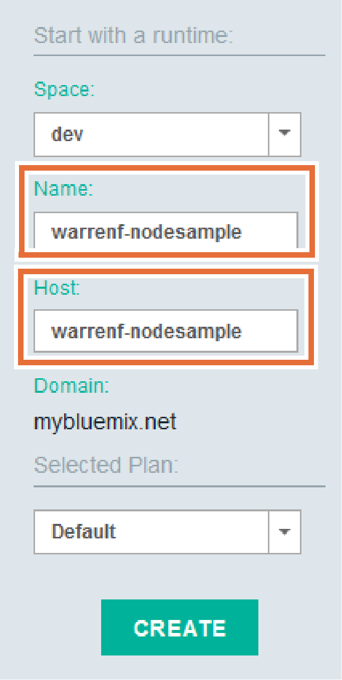
Plan	Features	Price
✓ Default	Run one or more apps free for 30 days (375 GB-hours free).	CA$0.074 CAD/GB-Hour

This is a service plan for the IBM Bluemix Platform runtime.

The **plan** explains the charges for using the cloud environment. In this example, IBM Bluemix provides a no-charge service under a threshold. After the unit-hour threshold, your IBM Bluemix account is charged at a **price** that depends on how much memory your application uses per hour.

In this example, the pricing in the **Canada** region is $0.074 Canadian dollars for each 1 GB used per hour. The service plan pricing depends on the runtime environment, your region, and your account type.

__ c. Examine the environment details for the SDK for Node.js runtime environment.



Start with a runtime:

Space: dev

Name: warrenf-nodesample

Host: warrenf-nodesample

Domain: mybluemix.net

Selected Plan: Default

The **space** represents the deployment environment within your IBM Bluemix account. In this example, *dev* represents a development sandbox for your IBM Bluemix applications.

The **name** and **host** settings represent a unique identifier for your application. Make sure that the host name is unique across all applications in the mybluemix.net domain.

__ d. Set the **name** to <*your-user-id*>`-nodesample`, where <*your-user-id*> is your own unique IBM ID.

Information

You can set the **name** and **host** value to any unique name across the `mybluemix.net` domain. Make sure that you use the same value for the rest of this exercise.

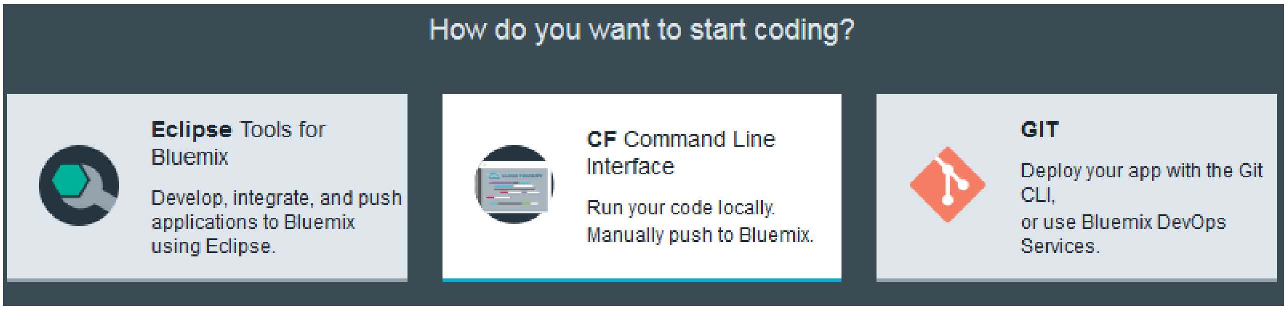
__ e. Set the **host** name to the same value as **name**.

__ f. Select the **Default** service plan.

__ g. Click **Create**.

__ 5. Examine the options for developing the Bluemix application.

The application wizard describes three options for developing your Bluemix application:


How do you want to start coding?
Eclipse Tools for Bluemix
Develop, integrate, and push applications to Bluemix using Eclipse.
CF Command Line Interface
Run your code locally. Manually push to Bluemix.
GIT
Deploy your app with the Git CLI, or use Bluemix DevOps Services.

You can develop the application on your own local workstation and deploy the code to your Bluemix application space. Securely log in to your IBM Bluemix account with the Cloud Foundry command-line interface, or **CF**. Deploy the application from your workstation to your IBM Bluemix account with this tool.

If you want to develop your application on your local workstation through an integrated development environment (IDE), download the **Eclipse Tools for Bluemix**. You can deploy your local application within Eclipse instead of command-line tools.

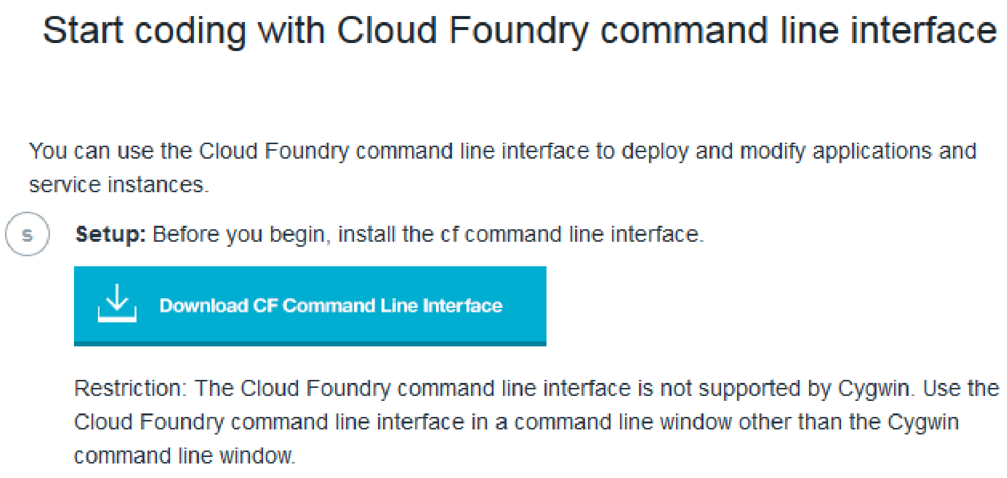
The IBM Bluemix account also supports **Git** repositories: a popular source code management system. Either link your organization’s Git repository server, or use IBM Bluemix DevOps services to host the repository on your behalf.

Note

In this exercise, you build an application with the Cloud Foundry command-line interface on your computer workstation. In the following exercises, you explore the two other options: creating an IBM Bluemix application with the Eclipse tools for Bluemix, and managing your application with a Git repository on IBM Bluemix DevOps services.

__ 6. Install the Cloud Foundry command-line tools on your own workstation.

__ a. Examine the instructions for coding a Bluemix application with the **CF** command-line interface.



Start coding with Cloud Foundry command line interface

You can use the Cloud Foundry command line interface to deploy and modify applications and service instances.

Setup: Before you begin, install the cf command line interface.

Restriction: The Cloud Foundry command line interface is not supported by Cygwin. Use the Cloud Foundry command line interface in a command line window other than the Cygwin command line window.

Information

You can also view the instructions for installing the CF command-line interface from the IBM Bluemix documentation:
`https://www.ng.bluemix.net/docs/#starters/install_cli.html`

__ b. Download the command-line interface.

__ c. Install the command-line interface on your workstation.

__ d. Create a directory to hold the source code for your application.

__ e. Open a command prompt (Microsoft Windows) or terminal (Mac OS, Linux).

__ f. Enter **`cf --version`** to confirm that the command-line interface runs correctly.

```
cf --version
```

Windows

Example:

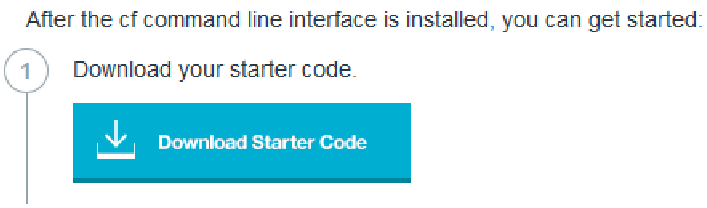
```
C:\IBM\Bluemix> cf --version
cf version 6.10.0-b78bf10-2015-02-11T22:26:25+00:00
```

Note

For more information about Mac OS and Linux versions of the Cloud Foundry command-line interface, see the IBM Bluemix documentation.

__ 7. Download and extract the sample application.

__ a. In the *Start coding with the command-line interface documentation* web page, click **Download Starter Code**.



After the cf command line interface is installed, you can get started:

1 Download your starter code.

__ b. Confirm that you downloaded a file that is named <*your-user-id*>**`-nodesample.zip`**, where <*your-user-id*> is the name of your IBM ID.

__ c. Extract the contents of the file in a source code directory for your application.

__ 8. Connect to IBM Bluemix with the **CF** command-line interface.

__ a. In a command prompt (Microsoft Windows) or terminal (Mac OS, Linux), change to the directory with the sample code.

__ b. Connect to the IBM Bluemix server with the command-line interface.

```
cf api https://api.ng.bluemix.net
```

Windows

Example:

```
C:\IBM\Bluemix> cf api https://api.ng.bluemix.net
Setting api endpoint to https://api.ng.bluemix.net...
OK
API endpoint:   https://api.ng.bluemix.net (API version: 2.19.0)
Not logged in. Use 'cf login' to log in.
```

__ 9. Log in to your IBM Bluemix account.

__ a. Log in to your IBM Bluemix account with the command-line interface.

`cf login -u` <*your-user-id*> **`-o`** <*your-organization*> **`-s dev`**

Important

When you run the login command, you must provide your IBM ID in <*your-user-id*> and the Bluemix organization to which your account belongs in <*your-organization*>.

Windows

Example:

```
C:\IBM\Bluemix> cf login -u warrenf@ca.ibm.com -o warrenf@ca.ibm.com
   -s dev
API endpoint: https://api.ng.bluemix.net

Password>
Authenticating...
OK

Targeted org warrenf@ca.ibm.com
Targeted space dev

API endpoint:   https://api.ng.bluemix.net (API version: 2.19.0)
User:           warrenf@ca.ibm.com
Org:            warrenf@ca.ibm.com
Space:          dev
```

__ 10. Upload and deploy the sample application into your Bluemix application.

__ a. Push the contents of the current directory to the IBM Bluemix application **`<your-user-id>-nodesample`**.

`cf push` <*your-user-id*>**`-nodesample`**

Windows

Example:

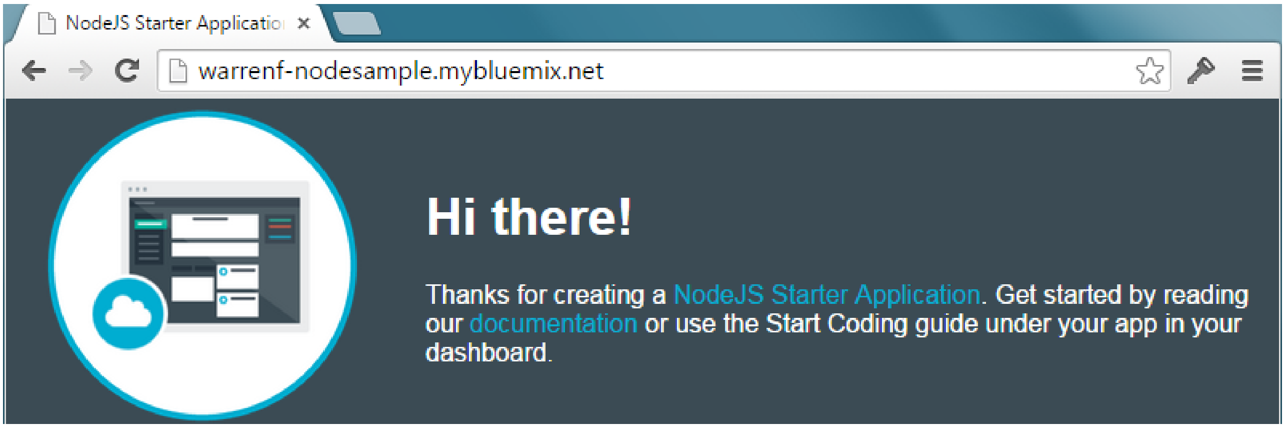
```
C:\IBM\Bluemix>cf push warrenf-nodesample
Using manifest file C:\IBM\Bluemix\manifest.yml

Updating app warrenf-nodesample in org warrenf@ca.ibm.com / space dev as
warren@ca.ibm.com...
OK

Using route warrenf-nodesample.mybluemix.net
Uploading warrenf-nodesample...
Uploading app files from: C:\IBM\Bluemix
Uploading 19.5K, 12 files
Done uploading
OK

Stopping app warrenf-nodesample in org warrenf@ca.ibm.com / space dev as
warren@ca.ibm.com...
OK

Starting app warrenf-nodesample in org warrenf@ca.ibm.com / space dev as
warren@ca.ibm.com...
-----> Downloaded app package (20K)
-----> Downloaded app buildpack cache (4.4M)
    -----> Node.js Buildpack Version: v1.14-20150309-1555
-----> Requested node range:  0.10.*
-----> Resolved node version: 0.10.36
-----> Installing IBM SDK for Node.js from cache
-----> Checking and configuring service extensions
-----> Restoring node_modules directory from cache
-----> Pruning cached dependencies not specified in package.json
-----> Installing dependencies
-----> Caching node_modules directory for future builds
-----> Cleaning up node-gyp and npm artifacts
-----> No Procfile found; Adding npm start to new Procfile
-----> Building runtime environment
-----> Checking and configuring service extensions
-----> Installing App Management
-----> Node.js Buildpack is done creating the droplet

-----> Uploading droplet (14M)

0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
1 of 1 instances running
```

```
App started

OK

App warrenf-nodesample was started using this command
`./vendor/initial_startup.rb`

Showing health and status for app warrenf-nodesample in org
warrenf@ca.ibm.com / space dev as warrenf@ca.ibm.com...
OK

requested state: started
instances: 1/1
usage: 256M x 1 instances
urls: warrenf-nodesample.mybluemix.net
last uploaded: Fri Feb 30 22:01:44 +0000 2015

     state     since                    cpu    memory         disk
details
#0   running   2015-02-30 06:02:42 PM   0.0%   47.5M of 256M   43.9M of 1G
```

__ 11. Confirm that your sample application is running.

__ a. Open **http://**<*your-user-id*>**-nodesample.bluemix.net/** in a web browser.

__ b. Confirm that the sample application appears.



Hi there!

Thanks for creating a NodeJS Starter Application. Get started by reading our documentation or use the Start Coding guide under your app in your dashboard.

__ c. Close the web browser.

End of exercise

Exercise review and wrap-up

The first part of the exercise explored the IBM Bluemix dashboard and catalog through your account. In the second part of the exercise, you created a Bluemix application with the IBM SDK for Node.js runtime environment. After installing the Cloud Foundry command-line interface, you deployed the sample application from your local workstation to your Bluemix account.

Exercise 2. Developing Bluemix applications with Eclipse

What this exercise is about

In this exercise, you work with the Eclipse development environment with the IBM Eclipse Tools for Bluemix plug-in. You learn what is needed to install and configure Eclipse for developing Bluemix applications.

What you should be able to do

After completing this exercise, you should be able to:

- Download the Eclipse and required plug-ins for developing Bluemix applications on Eclipse
- Configure Eclipse to work with Bluemix
- Push applications from Eclipse to Bluemix
- View Bluemix log files

Introduction

The Eclipse integrated development environment is an open source desktop program for designing, developing, testing, and managing applications. Eclipse supports several popular web application programming languages and platforms, including Enterprise Java and the IBM SDK for Node.js.

In this exercise, import the IBM Bluemix sample application that is written in the IBM SDK for Node.js into an Eclipse workspace. Examine the sample application source code. Install the Eclipse Tools for Bluemix and deploy the sample application to your IBM Bluemix account.

Requirements

You must complete the previous exercise in this course. Specifically, you must create an IBM Bluemix application with the IBM SDK for Node.js server run time. Also, you require a copy of the sample application that is written for the IBM SDK for Node.js.

Exercise instructions

Preface

This exercise depends on the availability of the IBM Bluemix cloud and connectivity from your own workstation to the cloud environment.

Part 1: Before you begin

Before you begin this exercise, you must complete the steps in the previous exercise. You created an IBM SDK for Node.js application in your IBM Bluemix account, and you downloaded the sample application to your workstation.

Part 2: Install Eclipse for Java EE Developer Bluemix Tools

Eclipse is an open source integrated development environment (IDE). With this application, you can design, develop, test, and deploy server applications. Eclipse supports the IBM SDK for Node.js and WebSphere Application Server Liberty runtime environments on IBM Bluemix.

In this section, download and install Eclipse for Java EE Developer on your workstation. Add the Eclipse tools for IBM Bluemix to the IDE.

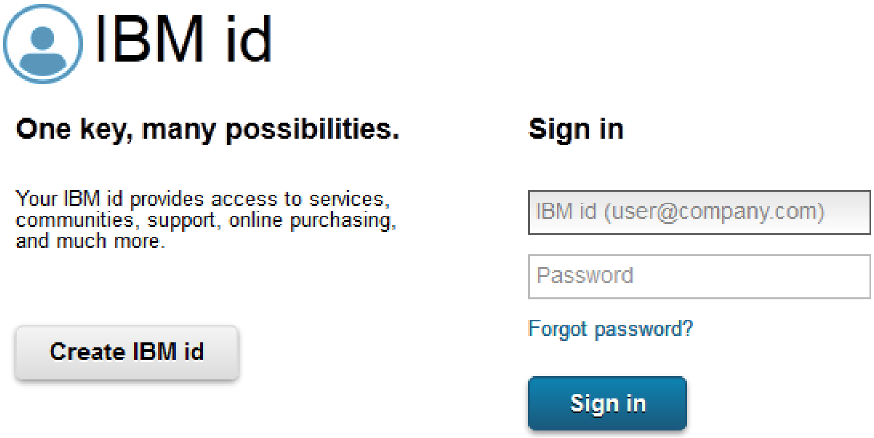
__ 1. Open the IBM Bluemix website in your web browser.

__ a. Open `http://www.bluemix.net` in a web browser window.

__ 2. Log in to your IBM Bluemix account.

__ a. Click **Log in**.

__ b. Enter your IBM ID and password.



IBM id

One key, many possibilities.

Your IBM id provides access to services, communities, support, online purchasing, and much more.

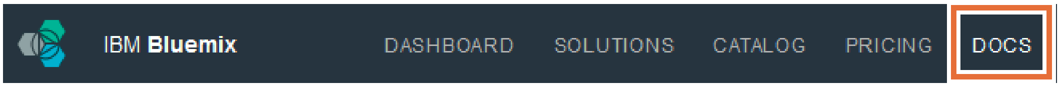
Sign in

__ c. Click **Sign in**.

__ d. Confirm that your IBM Bluemix account page is displayed.

__ 3. Review the instructions for installing the Eclipse Tools for Bluemix.

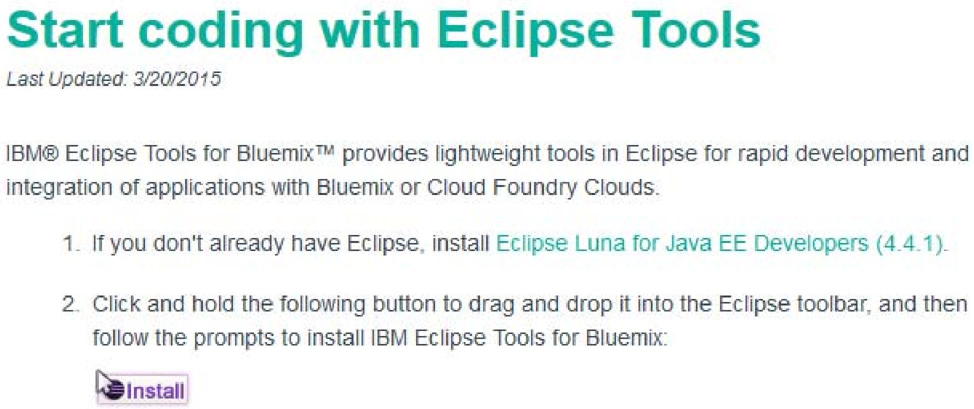
__ a. From the navigation bar, click **Docs**.



__ b. In the search bar, enter for the term **`Eclipse`**

__ c. Press Enter.

__ d. Open the article that is named "**Start coding with Eclipse Tools**".



Start coding with Eclipse Tools

Last Updated: 3/20/2015

IBM® Eclipse Tools for Bluemix™ provides lightweight tools in Eclipse for rapid development and integration of applications with Bluemix or Cloud Foundry Clouds.

1. If you don't already have Eclipse, install Eclipse Luna for Java EE Developers (4.4.1).
2. Click and hold the following button to drag and drop it into the Eclipse toolbar, and then follow the prompts to install IBM Eclipse Tools for Bluemix:

Information

The IBM Bluemix documentation is the most up-to-date resource for working with your account. If the steps outlined in the article are different from the lab exercise instructions, follow the steps in the article.

__ 4. Download and install the Eclipse Tools for IBM Bluemix.

__ a. From the "Start coding with Eclipse Tools" article, click the link to the **Eclipse for Java EE Developers** application package.

__ b. Download Eclipse for Java EE Developers.

__ c. Extract Eclipse for Java EE Developers into your local workstation.

Important

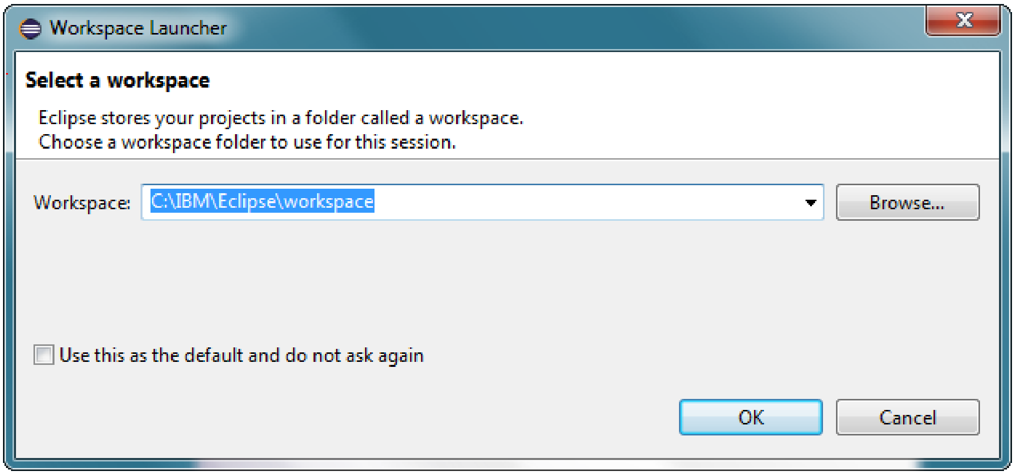
You must install a Java runtime environment (JRE) in your workstation before you start Eclipse. For more information about where to download and install the Java JRE, see: `https://www.java.com/en/download/`

__ 5. Start Eclipse for Java EE Developers.

  __ a. Go to the directory with your copy of Eclipse for Java EE Developers.

  __ b. Run **Eclipse**.

  __ c. In the workspace launcher dialog box, enter a directory to store your application source files.



Workspace Launcher

Select a workspace

Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: C:\IBM\Eclipse\workspace

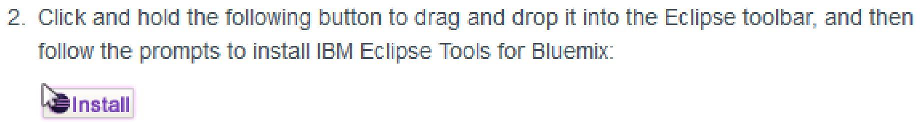
Important

Keep your application source code in a separate directory from your Bluemix application files. In this example, the Eclipse for Java EE Developers application is installed in `C:\IBM\Eclipse`. The workspace for the application source code is stored in `C:\IBM\Eclipse\workspace`.

  __ d. Click **OK**.

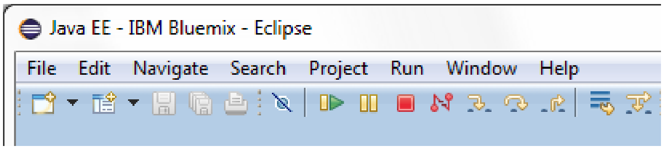
__ 6. Install the Eclipse Tools for IBM Bluemix.

  __ a. In the “Start coding with Eclipse Tools” webpage, click and hold the **Install** button.



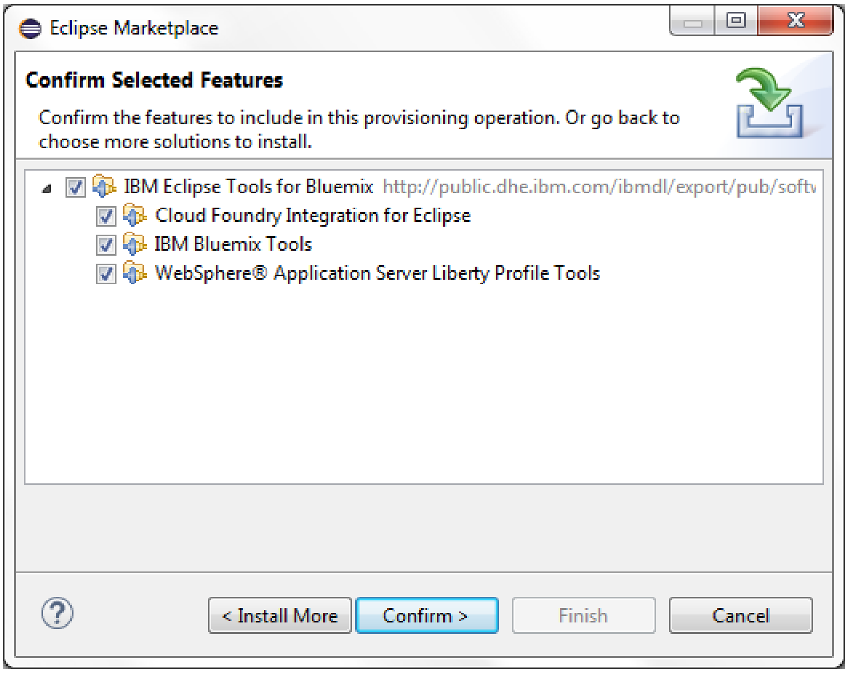
2. Click and hold the following button to drag and drop it into the Eclipse toolbar, and then follow the prompts to install IBM Eclipse Tools for Bluemix:

  __ b. Drag the installation button onto the Eclipse IDE toolbar.



Java EE - IBM Bluemix - Eclipse
File Edit Navigate Search Project Run Window Help

__ c. In the Eclipse Marketplace dialog box, select **IBM Eclipse Tools for IBM Bluemix**.



Eclipse Marketplace

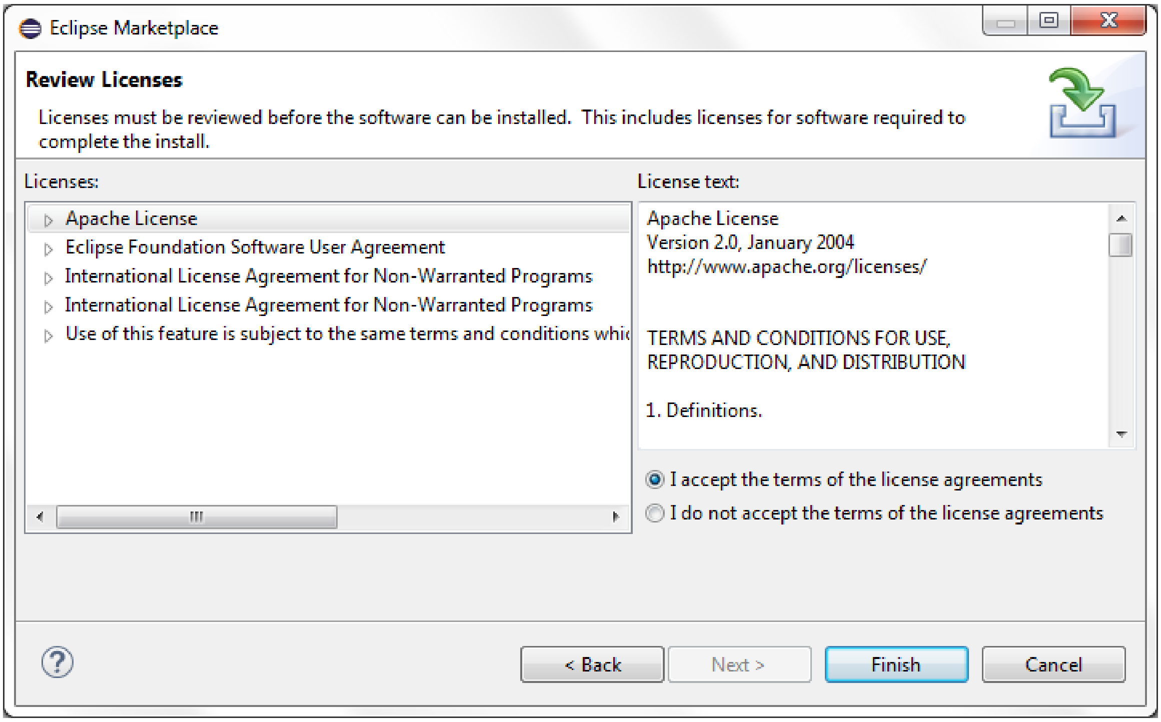
Confirm Selected Features

Confirm the features to include in this provisioning operation. Or go back to choose more solutions to install.

- ☑ IBM Eclipse Tools for Bluemix http://public.dhe.ibm.com/ibmdl/export/pub/softw
  - ☑ Cloud Foundry Integration for Eclipse
  - ☑ IBM Bluemix Tools
  - ☑ WebSphere® Application Server Liberty Profile Tools

__ d. Click **Confirm**.

__ e. Review the software licenses for the IBM Eclipse Tools for Bluemix. You must accept the license terms to install the software.



Eclipse Marketplace

Review Licenses

Licenses must be reviewed before the software can be installed. This includes licenses for software required to complete the install.

Licenses:

- Apache License
- Eclipse Foundation Software User Agreement
- International License Agreement for Non-Warranted Programs
- International License Agreement for Non-Warranted Programs
- Use of this feature is subject to the same terms and conditions whi

License text:

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE,
REPRODUCTION, AND DISTRIBUTION

1. Definitions.

● I accept the terms of the license agreements

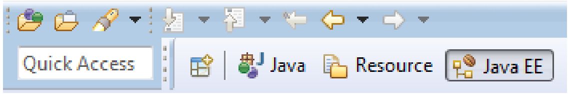
○ I do not accept the terms of the license agreements

__ f. Click **Finish**.

__ g. Confirm that the wizard completed successfully.

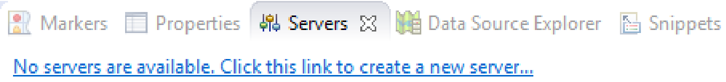
__ 7. Create a server profile for your IBM Bluemix account in the Eclipse **Servers** view.

__ a. From the main menu bar, switch to the **Java EE** perspective.



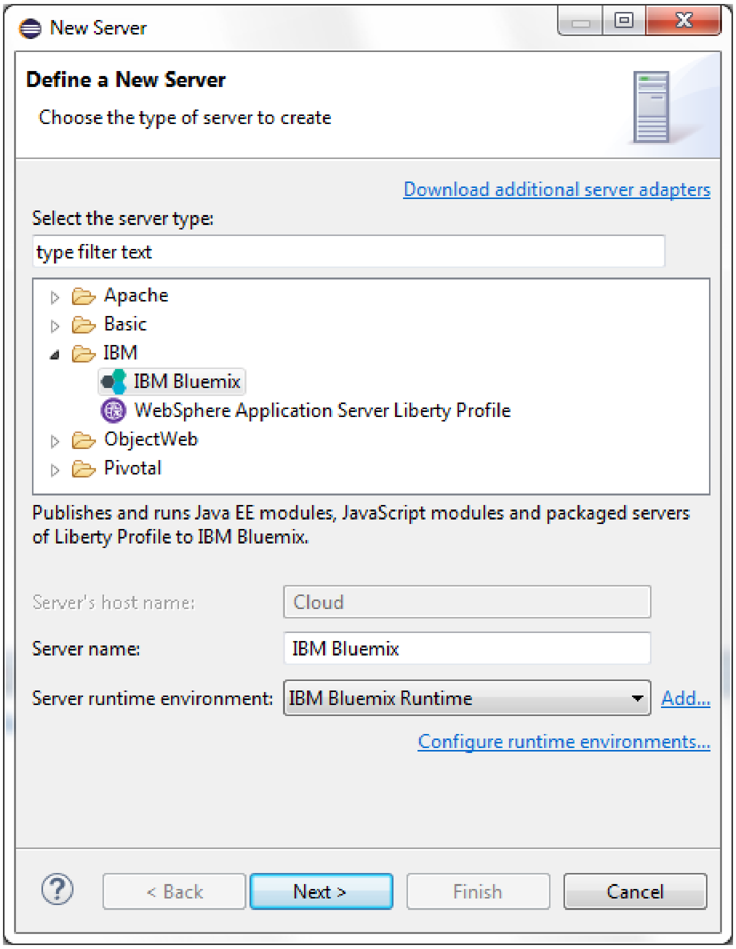
__ b. Examine the **Servers** view. There are currently no entries in the view.

__ c. Click **No servers are available. Click this link to create a new server**.


Markers
Properties
Servers
Data Source Explorer
Snippets
No servers are available. Click this link to create a new server...

__ d. In the "Define a new server" wizard, expand the **IBM** folder.

__ e. Select **IBM Bluemix** as the server type.


New Server
Define a New Server
Choose the type of server to create
Download additional server adapters
Select the server type:
type filter text
Apache
Basic
IBM
IBM Bluemix
WebSphere Application Server Liberty Profile
ObjectWeb
Pivotal

Publishes and runs Java EE modules, JavaScript modules and packaged servers of Liberty Profile to IBM Bluemix.

Server's host name: Cloud

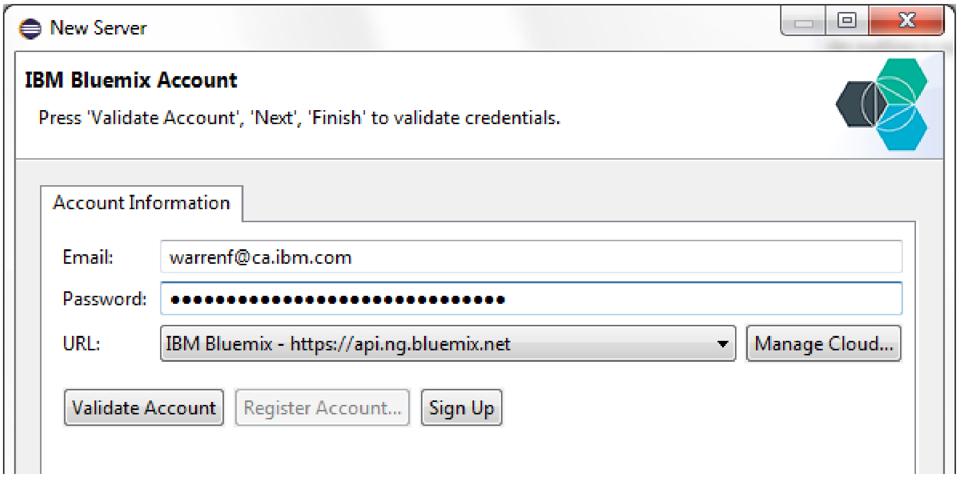
Server name: IBM Bluemix

Server runtime environment: IBM Bluemix Runtime

Add...
Configure runtime environments...
< Back
Next >
Finish
Cancel

__ f. Click **Next**.

__ g. On the “IBM Bluemix Account” page, type your IBM ID and password for the Bluemix account.



New Server

IBM Bluemix Account

Press 'Validate Account', 'Next', 'Finish' to validate credentials.

Account Information

Email: warrenf@ca.ibm.com

Password: ••••••••••••••••••••••••••••••

URL: IBM Bluemix - https://api.ng.bluemix.net

__ h. Click **Validate Account**. IBM Bluemix verifies that the IBM ID and password are correct.

Note

If the account validation fails, make sure that your IBM ID and password entries are correct for your account. You can verify your Bluemix account credentials through your web browser: `http://bluemix.net`

__ i. Click **Finish**.

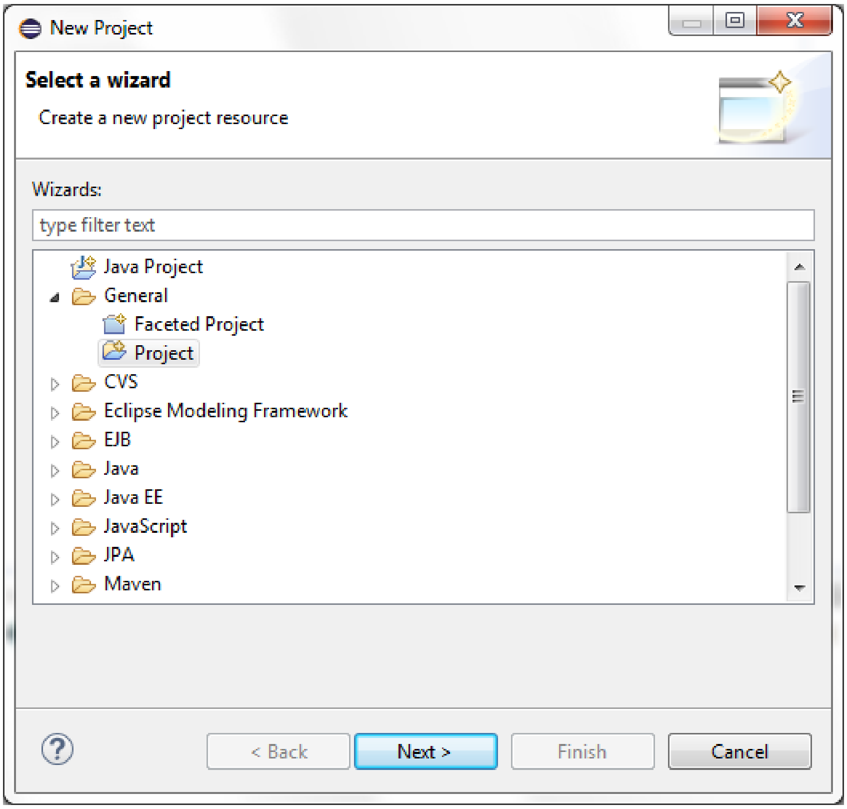
Part 3: Import the IBM Bluemix sample application to your Eclipse workspace

Copy the contents of the sample IBM SDK for Node.js application that you retrieved in the last exercise. Examine the source code for the sample application in the Eclipse integrated development environment.

__ 1. Create a project to hold the contents of your IBM Bluemix application.

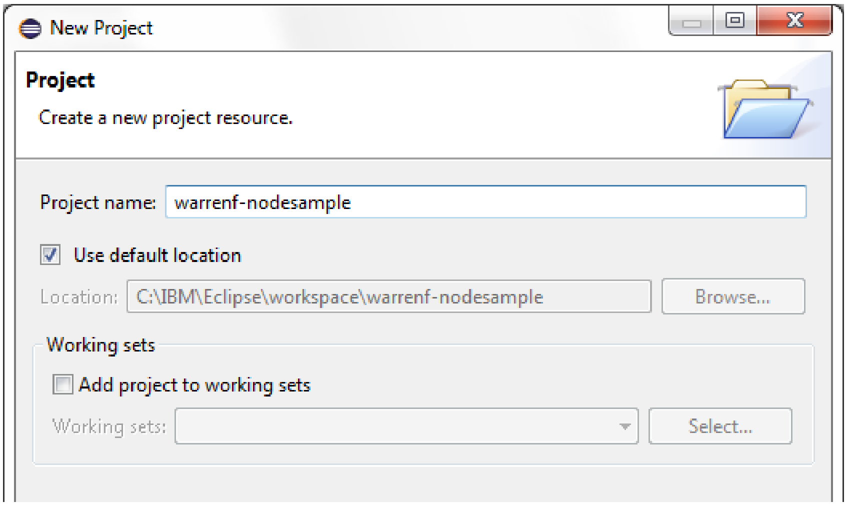
__ a. From the main menu bar, click **File > New > Project**.

__ b. On the “Select a wizard” page, select **General > Project**.



__ c. Click **Next**.

__ d. Enter **<*your-ibm-id*>-nodesample** as the name of the project. Replace *<your-ibm-id>* with the name of your IBM ID account.



Project name: warrenf-nodesample

☑ Use default location

Location: C:\IBM\Eclipse\workspace\warrenf-nodesample

☐ Add project to working sets

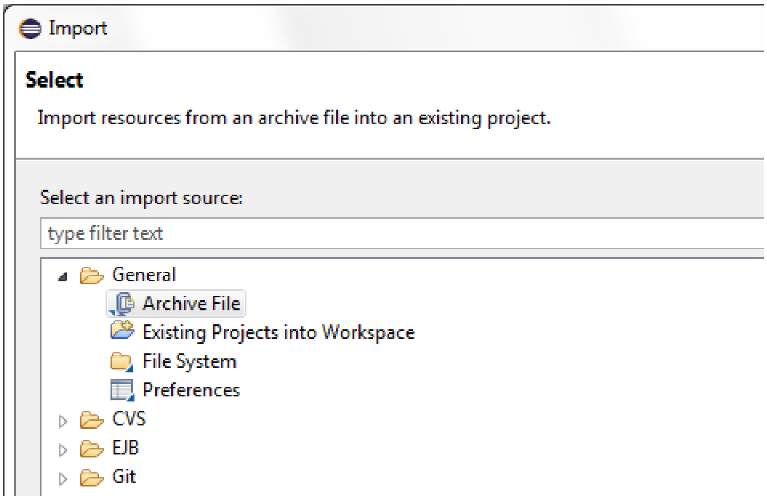
__ e. Click **Finish**.

__ 2. Import the IBM SDK for Node.js sample application into your workspace.

__ a. From the **Project Explorer** view, select the project for your Bluemix application.

__ b. From the main menu bar, click **File > Import**.

__ c. In the Import wizard, select **General > Archive File** as the source file.



Import

Select

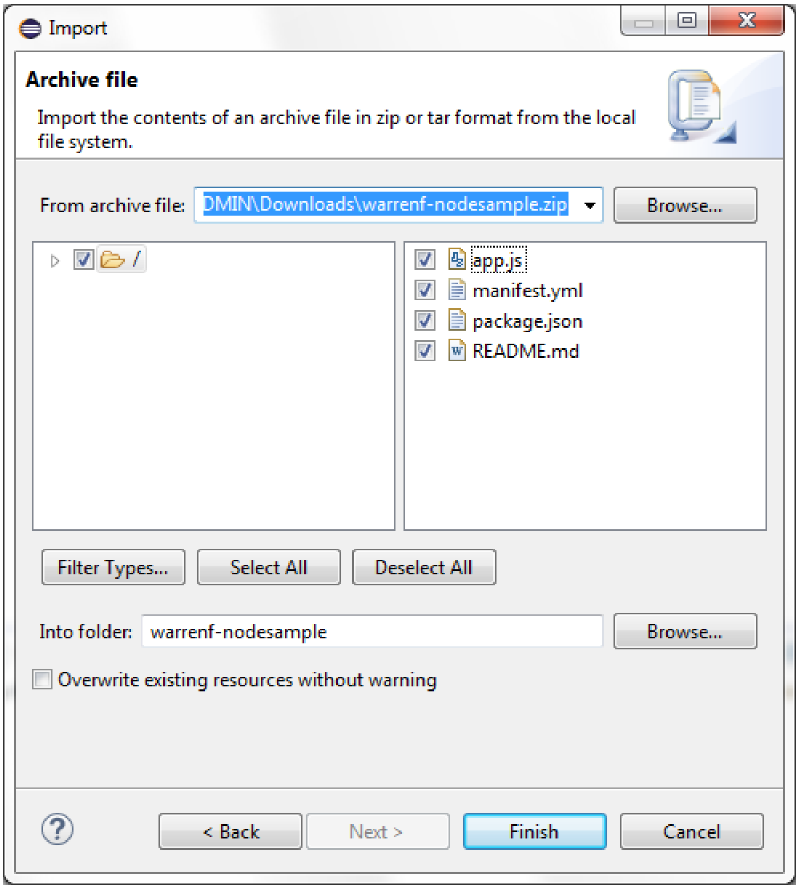
Import resources from an archive file into an existing project.

Select an import source:

type filter text

__ d. Click **Next**.

__ e. In the **From archive file** field, select the sample application compressed file. You downloaded the sample application in Exercise 1.



Import

Archive file

Import the contents of an archive file in zip or tar format from the local file system.

From archive file: DMIN\Downloads\warrenf-nodesample.zip

☑ /

☑ app.js
☑ manifest.yml
☑ package.json
☑ README.md

Into folder: warrenf-nodesample

□ Overwrite existing resources without warning

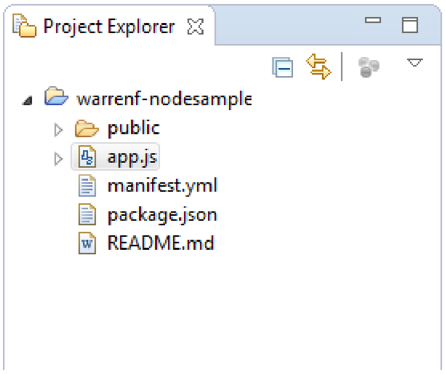
__ f. Click **Finish**.

Part 4: Examine the sample application

In this part, examine the sample application in your Eclipse workspace. The sample application runs on the IBM SDK for Node.js server runtime environment. It uses the Express web application framework as a web application solution.

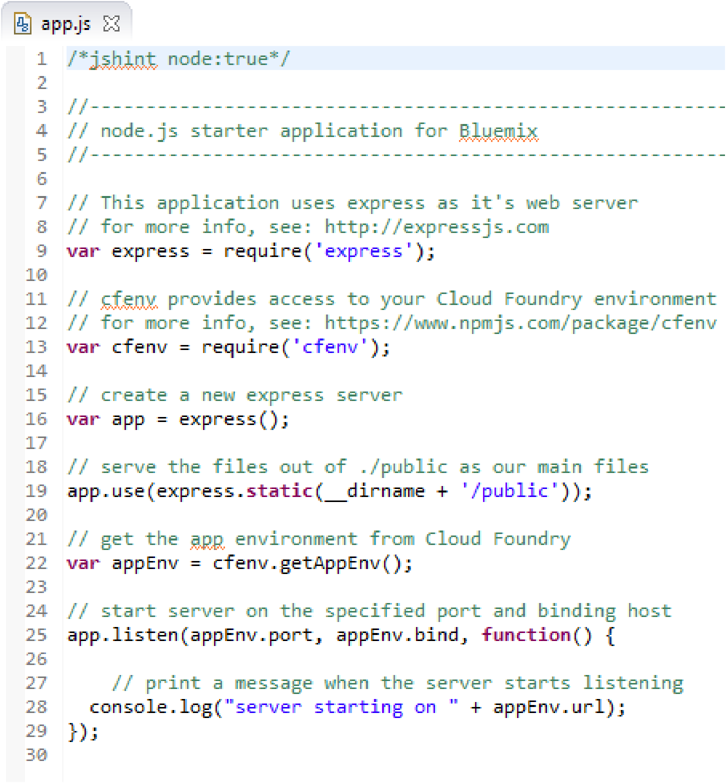
__ 1. Examine the Bluemix sample application for the IBM SDK for Node.js.

__ a. In the Project Explorer view, expand the ***<your-ibm-id>*****-nodesample** project.



__ b. Open **app.js**.

__ c. Examine the **app.js** source code in the editor.



```
/*jshint node:true*/

//------------------------------------------------------------
// node.js starter application for Bluemix
//------------------------------------------------------------

// This application uses express as it's web server
// for more info, see: http://expressjs.com
var express = require('express');

// cfenv provides access to your Cloud Foundry environment
// for more info, see: https://www.npmjs.com/package/cfenv
var cfenv = require('cfenv');

// create a new express server
var app = express();

// serve the files out of ./public as our main files
app.use(express.static(__dirname + '/public'));

// get the app environment from Cloud Foundry
var appEnv = cfenv.getAppEnv();

// start server on the specified port and binding host
app.listen(appEnv.port, appEnv.bind, function() {

    // print a message when the server starts listening
  console.log("server starting on " + appEnv.url);
});

```

```
/*jshint node:true*/

//-----------------------------------------------------------------------
// node.js starter application for Bluemix
//-----------------------------------------------------------------------

// This application uses express as its web server
// for more info, see: http://expressjs.com
var express = require('express');

// cfenv provides access to your Cloud Foundry environment
// for more info, see: https://www.npmjs.com/package/cfenv
var cfenv = require('cfenv');

// create a new express server
var app = express();

// serve the files out of ./public as our main files
app.use(express.static(__dirname + '/public'));

// get the app environment from Cloud Foundry
var appEnv = cfenv.getAppEnv();

// start server on the specified port and binding host
```

```
app.listen(appEnv.port, appEnv.bind, function() {

// print a message when the server starts listening
  console.log("server starting on " + appEnv.url);
});
```

The **app.js** is the main entry point for the sample IBM SDK for Node.js application. The sample application uses two modules: **express.js** and **cfenv**. The Express web application framework sets up an app JavaScript object to represent the web application. By default, the Node.js language provides lower-level HTTP objects to handle the interaction between the client and the server.

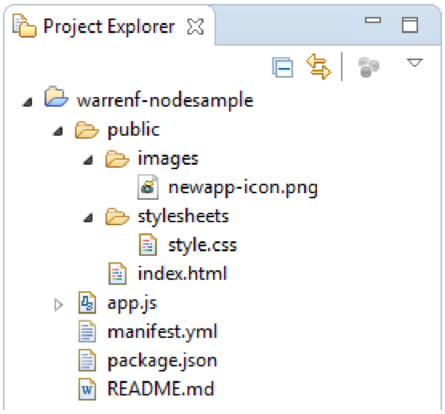
The Cloud Foundry environment module provides programmatic access to the environment variables from your IBM Bluemix account.

The HTML main welcome page is in the **`/public`** directory.

In the last line of the screen capture, the **`app.use(...)`** function configures the routes of the HTTP express object. The function reads the **`index.html`** file and returns the web page in the HTTP response message.

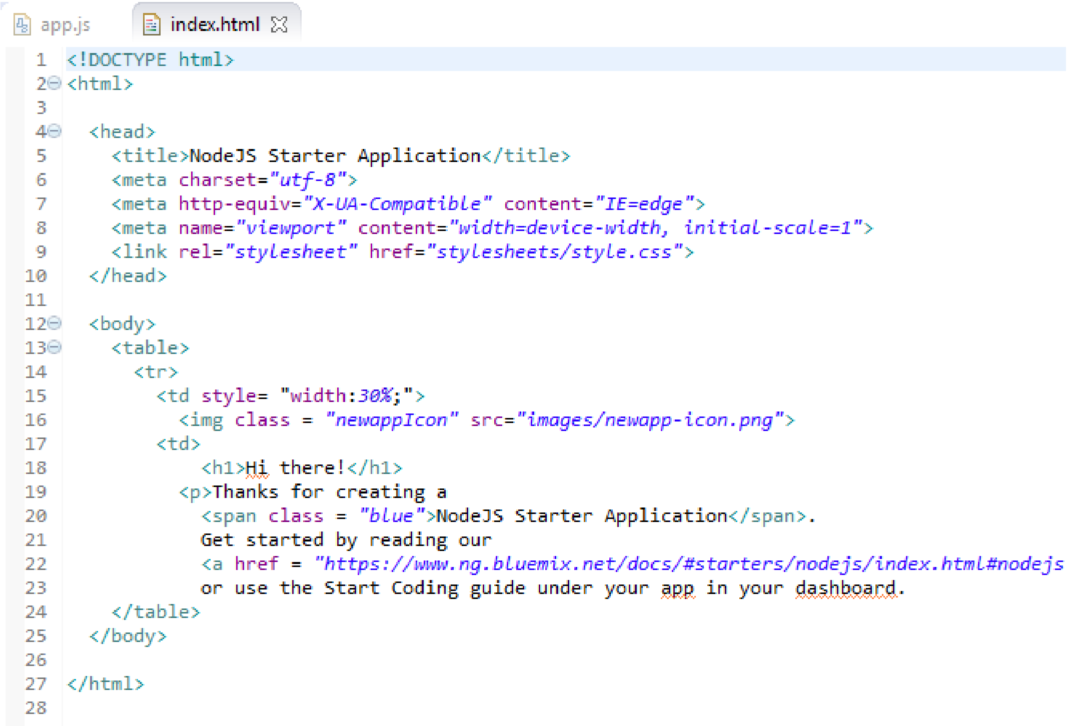
__ 2. Examine the HTML web pages for the sample application.

__ a. Expand the **public** folder in the Project Explorer view.



```
Project Explorer
◢ warrenf-nodesample
    ◢ public
        ◢ images
            newapp-icon.png
        ◢ stylesheets
            style.css
        index.html
    ▷ app.js
    manifest.yml
    package.json
    README.md
```

__ b. Open **index.html**.



```
<!DOCTYPE html>
<html>

  <head>
    <title>NodeJS Starter Application</title>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="stylesheets/style.css">
  </head>

  <body>
    <table>
      <tr>
        <td style= "width:30%;">
          <img class = "newappIcon" src="images/newapp-icon.png">
        <td>
            <h1>Hi there!</h1>
          <p>Thanks for creating a
            <span class = "blue">NodeJS Starter Application</span>.
            Get started by reading our
            <a href = "https://www.ng.bluemix.net/docs/#starters/nodejs/index.html#nodejs
            or use the Start Coding guide under your app in your dashboard.
    </table>
  </body>

</html>

```

```
<!DOCTYPE html>
<html>

  <head>
    <title>NodeJS Starter Application</title>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="stylesheets/style.css">
  </head>

  <body>
    <table>
      <tr>
        <td style= "width:30%;">
          <img class = "newappIcon" src="images/newapp-icon.png">
        <td>
            <h1>Hi there!</h1>
          <p>Thanks for creating a
            <span class = "blue">NodeJS Starter Application</span>.
            Get started by reading our
            <a href = "https://www.ng.bluemix.net/docs/#starters/nodejs/
            index.html#nodejs">documentation</a>
            or use the Start Coding guide under your app in your dashboard.
    </table>
  </body>
```

```
</html>
```

The **index.html** web page writes a message to the body of the webpage. It also provides a link to the IBM Bluemix documentation on the IBM SDK for Node.js run time.

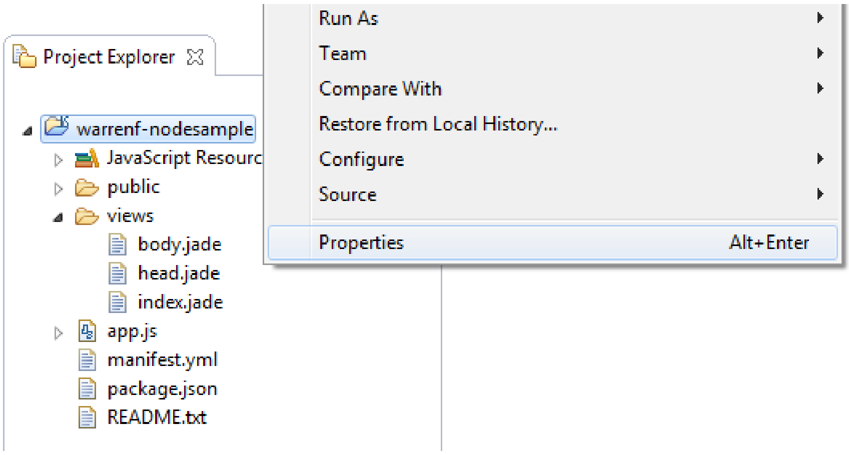
Part 5: Publish and test the sample application

Before you edit the sample application, deploy and test the application to your IBM Bluemix account development (dev) environment. The Eclipse tools for Bluemix added your IBM Bluemix account as a remote server in your Eclipse workspace.

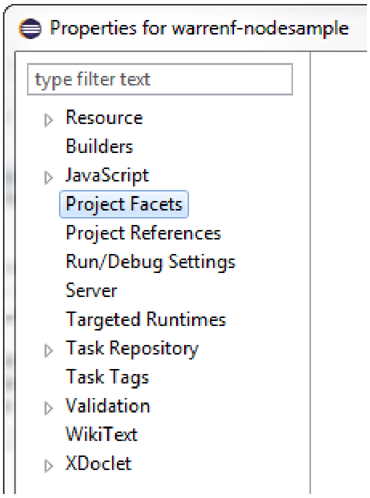
__ 1. Define the ***<your-ibm-id>*-nodesample** application type as an IBM SDK for Node.js application.

__ a. In the Project Explorer view, right-click ***<your-ibm-id>*-nodesample**.

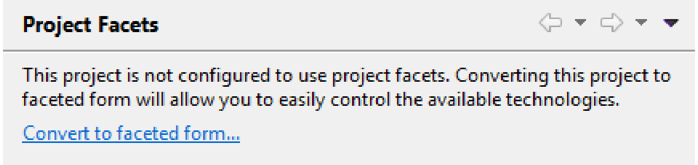
__ b. Click **Properties**.



__ c. In project properties, select the **Project Facets** category.



__ d. Click **Convert to faceted form**.

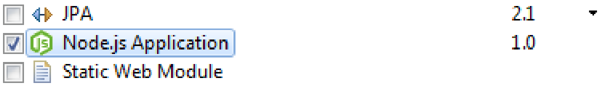


Project Facets

This project is not configured to use project facets. Converting this project to faceted form will allow you to easily control the available technologies.

Convert to faceted form...

__ e. Select the **Node.js Application** facet.



☐ JPA 2.1

☑ Node.js Application 1.0

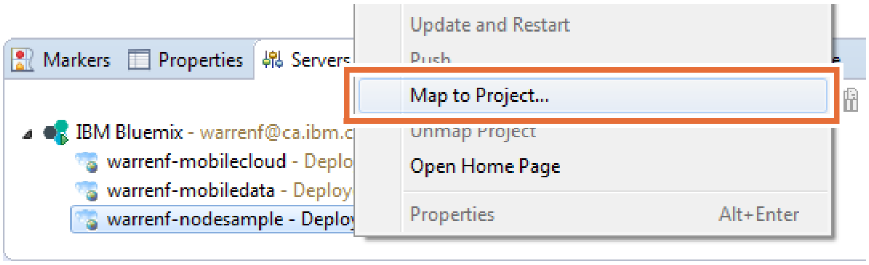
☐ Static Web Module

__ f. Click **OK**.

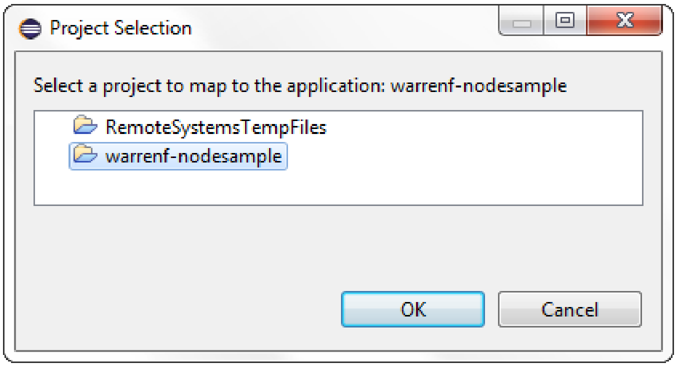
Note

To avoid creating two applications with the same name, remove the existing ***<your-ibm-id>*-nodesample** application that you created in the previous exercise.

You can also overwrite the contents of the existing ***<your-ibm-id>*-nodesample** application. Map the existing project with an IBM Bluemix application.


Markers
Properties
Servers
IBM Bluemix - warrenf@ca.ibm.c
warrenf-mobilecloud - Deplo
warrenf-mobiledata - Deploy
warrenf-nodesample - Deplo
Update and Restart
Push
Map to Project...
Unmap Project
Open Home Page
Properties Alt+Enter

Select the IBM Bluemix application and synchronize the contents with the project in the local workspace.



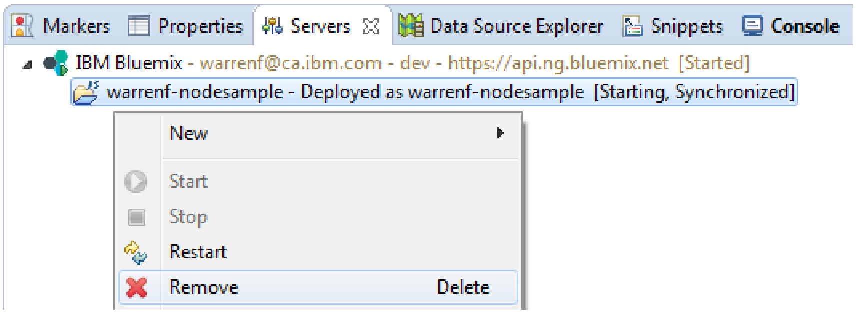
Project Selection

Select a project to map to the application: warrenf-nodesample

__ 2. Remove the existing sample application from your IBM Bluemix account.

__ a. In the **servers** view, right-click the IBM Bluemix server.

__ b. Click **Remove**.

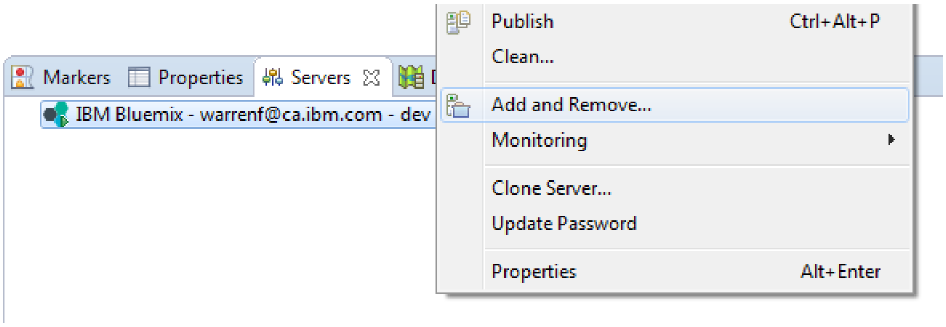

Markers
Properties
Servers
Data Source Explorer
Snippets
Console
IBM Bluemix - warrenf@ca.ibm.com - dev - https://api.ng.bluemix.net [Started]
warrenf-nodesample - Deployed as warrenf-nodesample [Starting, Synchronized]
New
Start
Stop
Restart
Remove Delete

__ c. Click **OK** to confirm the remove action.

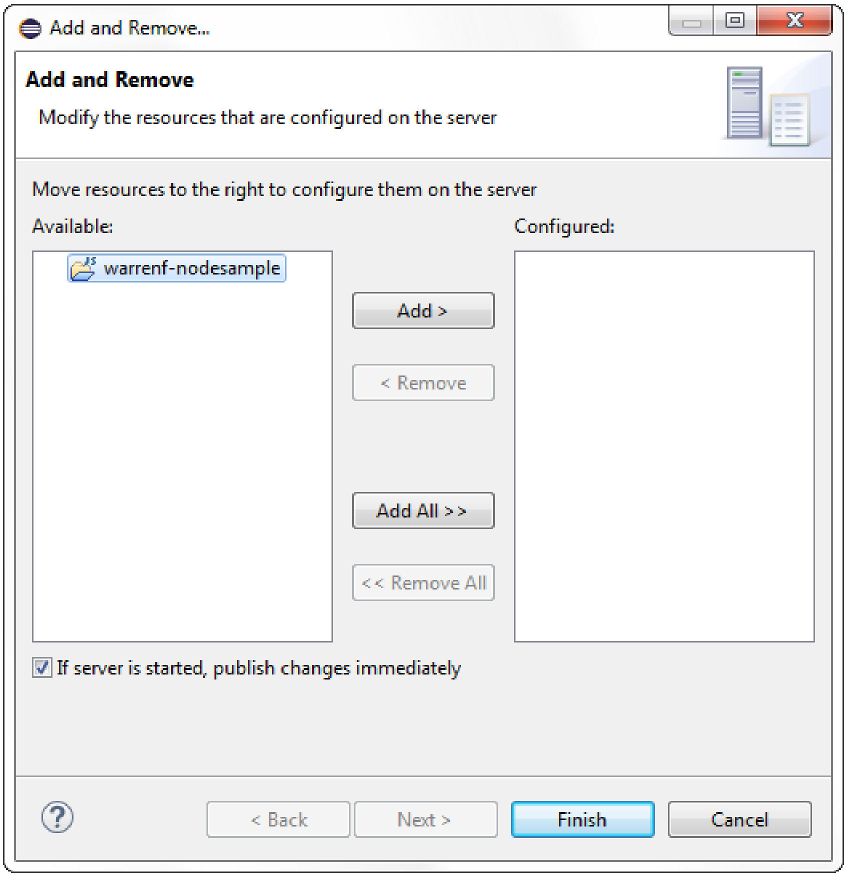
__ 3. Add the ***<your-ibm-id>*-nodesample** application to your IBM Bluemix account.

__ a. Right-click the IBM Bluemix server in the **servers** view again.

__ b. Click **Add and remove**.



__ c. Click **Add All**.



Add and Remove...

Add and Remove

Modify the resources that are configured on the server

Move resources to the right to configure them on the server

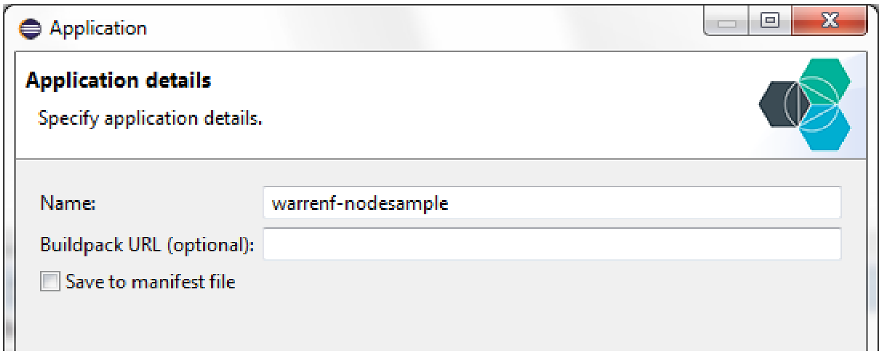
Available: warrenf-nodesample

Configured:

☑ If server is started, publish changes immediately

__ d. Click **Finish**.

__ e. In the Application Details wizard, leave the project name as the IBM Bluemix application name.



Application

Application details

Specify application details.

Name: warrenf-nodesample

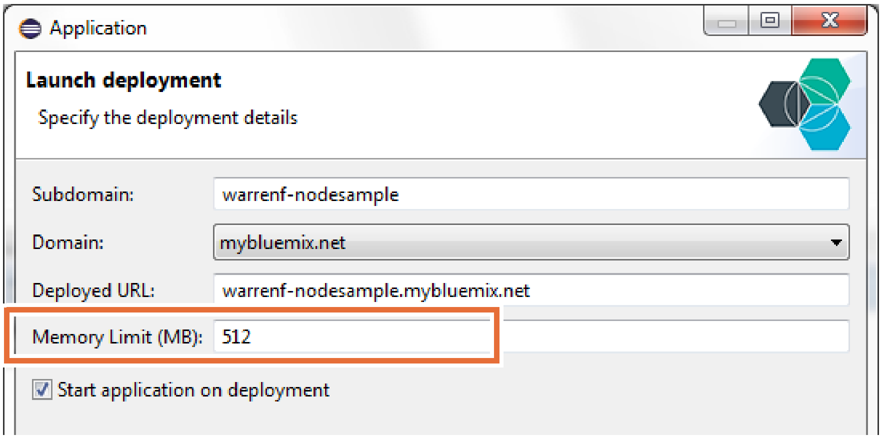
Buildpack URL (optional):

□ Save to manifest file

__ f. Click **Next**.

__ g. On the “Launch deployment” page, make sure that the **Memory limit (MB)** is set to the default value of **512** MB.

__ h. Review the Launch deployment details, including the web address for the web application.



Application

Launch deployment

Specify the deployment details

Subdomain: warrenf-nodesample

Domain: mybluemix.net

Deployed URL: warrenf-nodesample.mybluemix.net

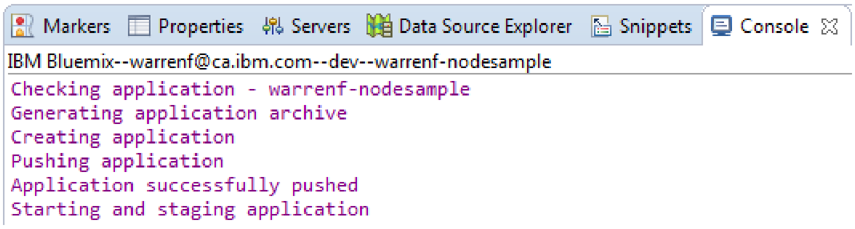
Memory Limit (MB): 512

☑ Start application on deployment

__ i. Click **Finish**.

__ 4. Verify that the Eclipse tools successfully deployed the sample application to your IBM Bluemix account.

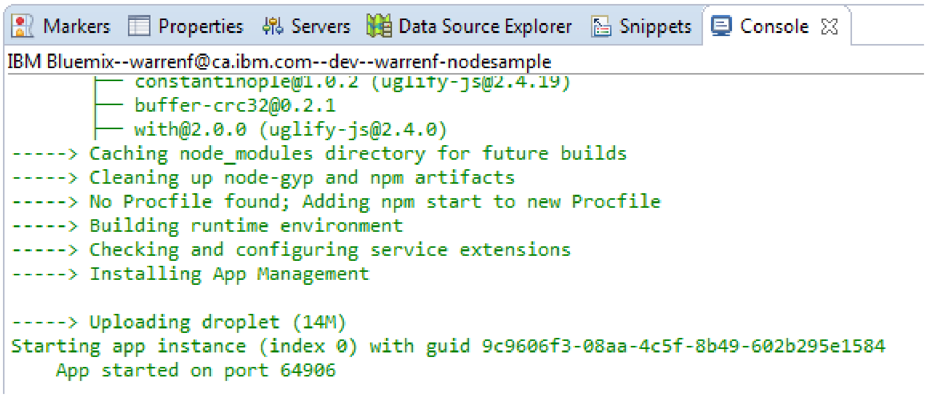
__ a. Examine the **Console** view. The Eclipse tools for Bluemix reports the deployment status in this view.



IBM Bluemix--warrenf@ca.ibm.com--dev--warrenf-nodesample

```
Checking application - warrenf-nodesample
Generating application archive
Creating application
Pushing application
Application successfully pushed
Starting and staging application
```

__ b. Confirm that the tool successfully deployed the sample application to IBM Bluemix.

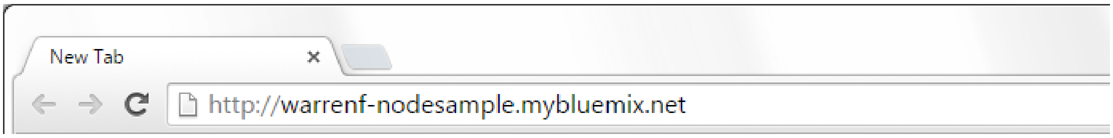


IBM Bluemix--warrenf@ca.ibm.com--dev--warrenf-nodesample

```
           ├── constantinople@1.0.2 (uglify-js@2.4.19)
           ├── buffer-crc32@0.2.1
           ├── with@2.0.0 (uglify-js@2.4.0)
-----> Caching node_modules directory for future builds
-----> Cleaning up node-gyp and npm artifacts
-----> No Procfile found; Adding npm start to new Procfile
-----> Building runtime environment
-----> Checking and configuring service extensions
-----> Installing App Management

-----> Uploading droplet (14M)
Starting app instance (index 0) with guid 9c9606f3-08aa-4c5f-8b49-602b295e1584
    App started on port 64906
```

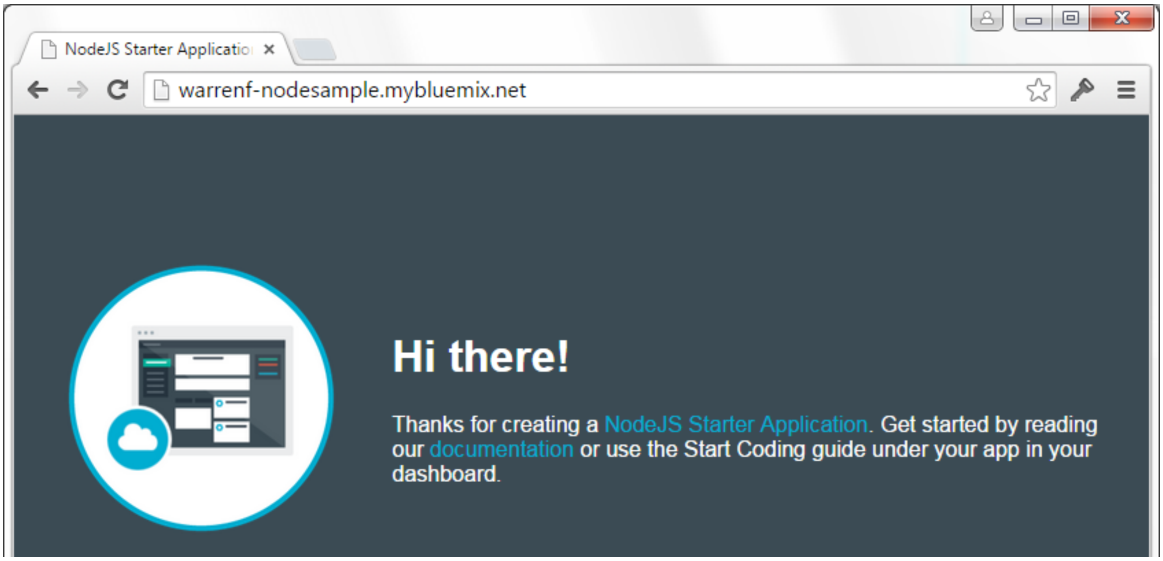
__ 5. Open the web address for the sample application.

__ a. In a web browser, go to **`http://<your-ibm-id>-nodesample.mybluemix.net`**. Replace *<your-ibm-id>* with your own IBM ID account name.



http://warrenf-nodesample.mybluemix.net

__ b. Confirm that the sample index page is displayed in your browser.



warrenf-nodesample.mybluemix.net

Hi there!

Thanks for creating a NodeJS Starter Application. Get started by reading our documentation or use the Start Coding guide under your app in your dashboard.

__ c. Close the web browser.

__ 6. Close Eclipse.

End of exercise

Exercise review and wrap-up

In the first part of the exercise, you installed the Eclipse integrated development environment (IDE) on your own computer. After installing the Eclipse tools for Bluemix, you set up a connection between your local workspace and your remote IBM Bluemix account.

In the second part of the exercise, you imported the sample application for IBM SDK for Node.js into your Eclipse workspace. You examined the source code for the application, and you deployed the application to your previously created IBM Bluemix application.

Exercise 3. Developing Bluemix applications with IBM Bluemix DevOps services

What this exercise is about

In this exercise, you work with IBM Bluemix DevOps Services to explore, develop, build, and deploy Bluemix applications.

What you should be able to do

After completing this exercise, you should be able to:

- Sign in to DevOps
- Explore public projects in DevOps
- Sign in to Bluemix and DevOps
- Create a Git repository in DevOps Services to manage your source code
- View and edit code in DevOps
- Build and deploy code from DevOps to Bluemix
- Test the application in Bluemix

Introduction

The IBM Bluemix DevOps service provides a set of online development tools. When you create a project for your application, DevOps services sets up a source code repository compatible with the Git API. In the Build and Deploy mode, you can commit and restore files in your project. You can also use your own Git client.

DevOps service also provides a rich integrated development environment (IDE) for developing, testing, and deploying IBM SDK for Node.js applications right in your web browser. The Edit Code mode provides you with syntax highlighting and API reference features that you expect in a desktop application.

Requirements

You must complete the previous exercise in this course. Specifically, you must create an IBM Bluemix application with the IBM SDK for Node.js server run time. Also, you require a copy of the sample application that is written for the IBM SDK for Node.js.

Exercise instructions

Preface

This exercise depends on the availability of the IBM Bluemix cloud and connectivity from your own workstation to the cloud environment.

Part 1: Before you begin

Before you begin this exercise, you must complete the steps in the previous exercise. You created an IBM SDK for Node.js application in your IBM Bluemix account, and you downloaded the sample application to your workstation.

Part 2: Examine the IBM Bluemix application

In this part, explore the application overview page in your IBM Bluemix account. The overview page lists the status of your application and the resources that it uses. Set up a source code repository to manage the artifacts that make up your Bluemix application.

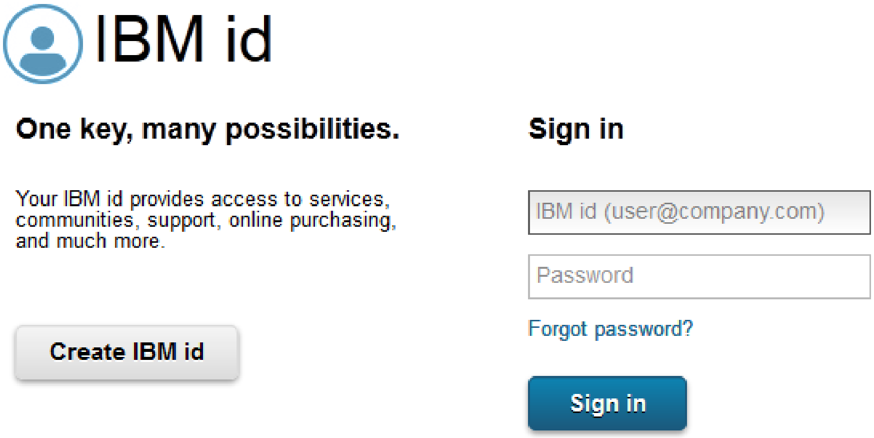
__ 1. Open the IBM Bluemix website in your web browser.

  __ a. Open `http://www.bluemix.net` in a web browser window.

__ 2. Log in to your IBM Bluemix account.

  __ a. Click **Log in**.

  __ b. Enter your IBM ID and password.



IBM id

One key, many possibilities.

Your IBM id provides access to services, communities, support, online purchasing, and much more.

Sign in

IBM id (user@company.com)

Password

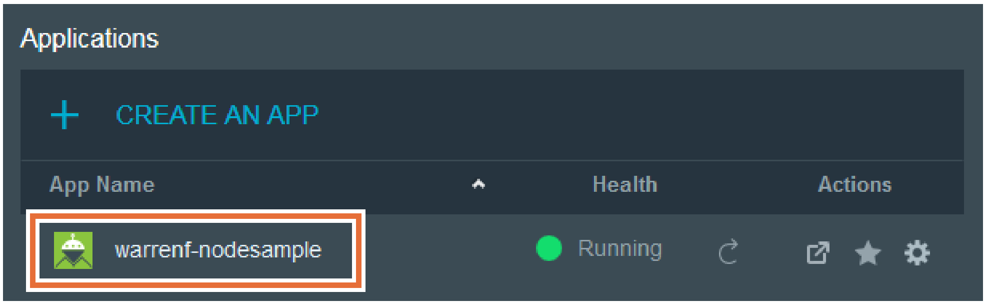
  __ c. Click **Sign in**.

  __ d. Confirm that your IBM Bluemix account page appears.

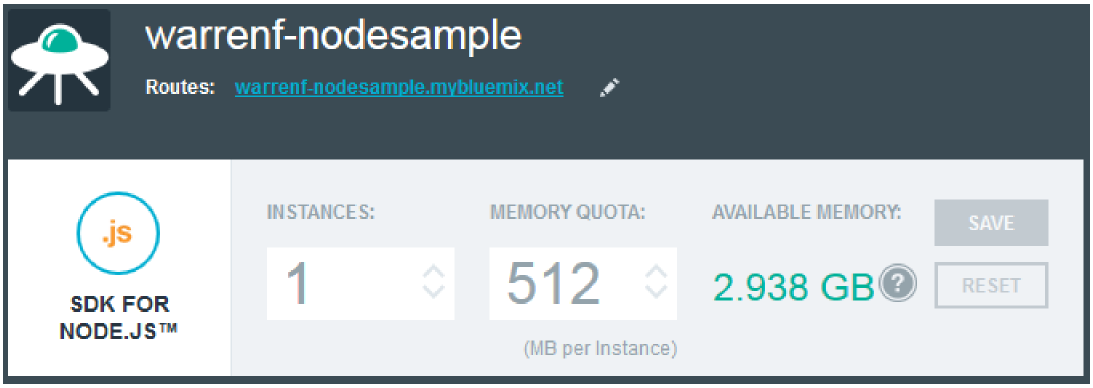
__ 3. Open the application page for your sample application.

  __ a. In the IBM Bluemix dashboard, scroll down to the **Applications** section.

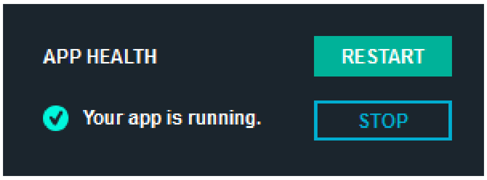
__ b. Click **<*your-ibm-id*>-nodesample**, the application that you created in the previous exercise.



__ c. Set the memory quota to **512 MB** memory.

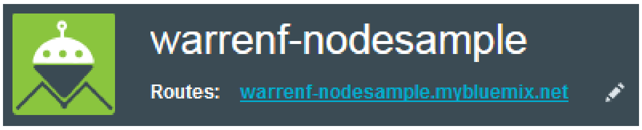


__ d. Confirm that your application is in the running state.

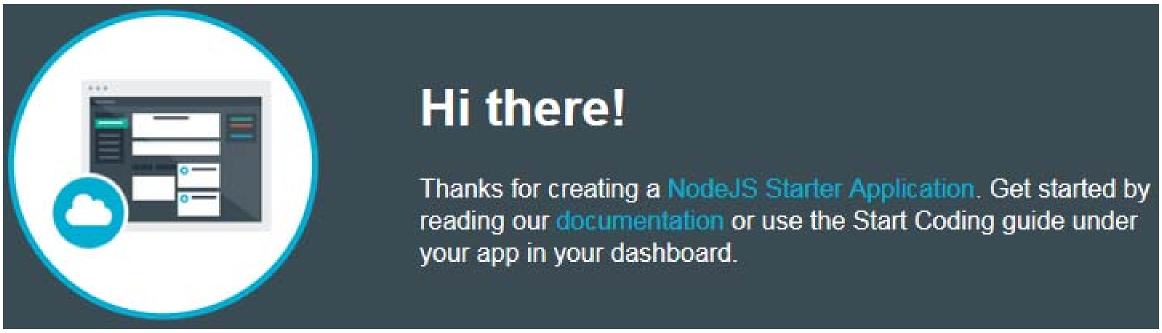


__ 4. Test the sample application.

__ a. Click the routes link for your application.



__ b. Check to see that the browser opens the NodeJS starter application web page.



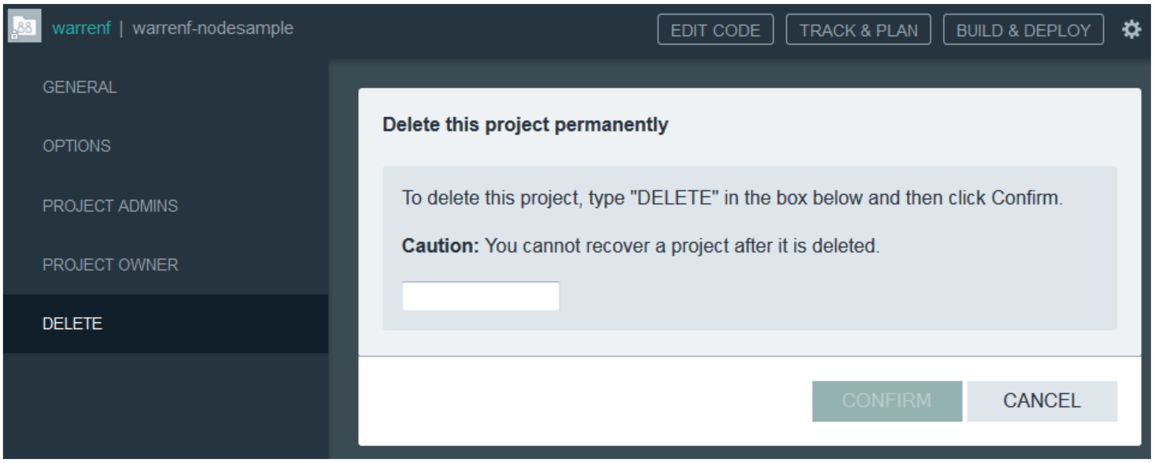
Hi there!

Thanks for creating a NodeJS Starter Application. Get started by reading our documentation or use the Start Coding guide under your app in your dashboard.

__ c. Close the NodeJS starter application web page.

Important

If you already created a Git repository on IBM Bluemix DevOps Services for the sample application, you must delete the Git repository before you continue with this exercise.



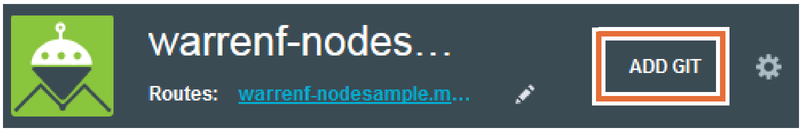
Delete this project permanently

To delete this project, type "DELETE" in the box below and then click Confirm.

Caution: You cannot recover a project after it is deleted.

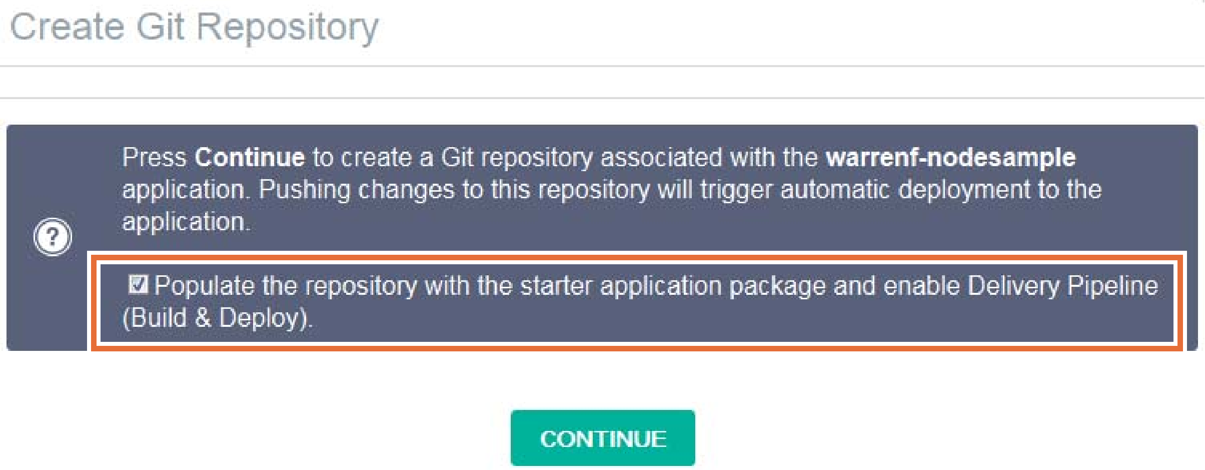
__ 5. Create a Git Repository for the starter application source code.

__ a. On the **<*your-ibm-id*>-nodesample** application page, click **Add Git**.



__ b. Select **Populate the repository with the starter application package and enable Delivery Pipeline (Build & Deploy)**.

__ c. In the Create Git Repository wizard, click **Continue**.



Create Git Repository

Press **Continue** to create a Git repository associated with the **warrenf-nodesample** application. Pushing changes to this repository will trigger automatic deployment to the application.

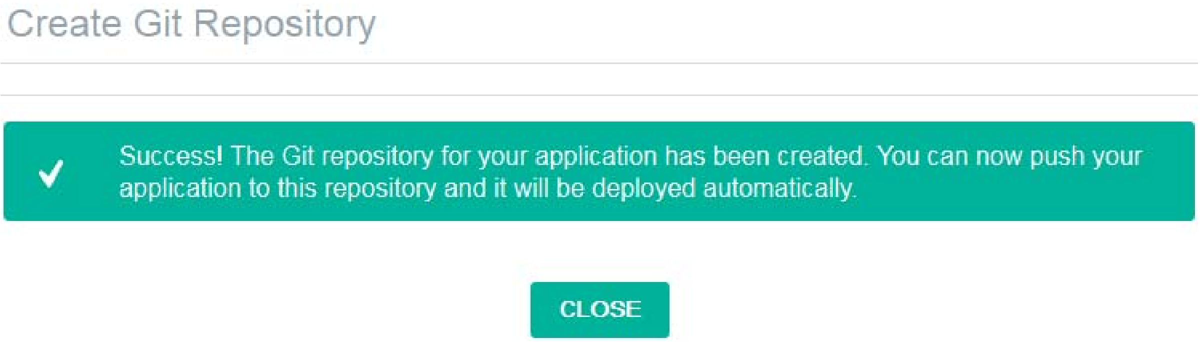
☑ Populate the repository with the starter application package and enable Delivery Pipeline (Build & Deploy).

Note

In the first lab exercise, you created the ***<your-ibm-id>*-nodesample** Bluemix application. You also downloaded a copy of the NodeJS starter application source code to your workstation. Upload the source code into the Git repository for the application.

__ d. Wait until the wizard creates the Git repository for your application.

__ e. Confirm that the wizard completed successfully.



Create Git Repository

✓ Success! The Git repository for your application has been created. You can now push your application to this repository and it will be deployed automatically.

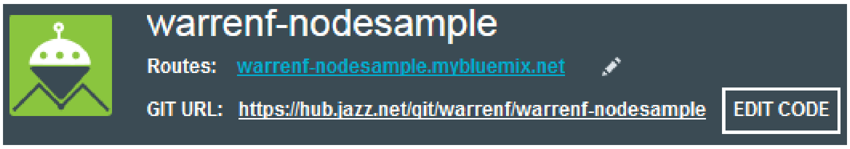
__ f. Click **Close**.

Part 3: Review the Git repository for your application

Before you import the source code for the application, examine the features of the IBM Bluemix DevOps project that manages the artifacts for your application.

__ 1. Open the Git repository for your application.

__ a. On the Bluemix application page, examine the Git URL link.



warrenf-nodesample

Routes: warrenf-nodesample.mybluemix.net

GIT URL: https://hub.jazz.net/git/warrenf/warrenf-nodesample

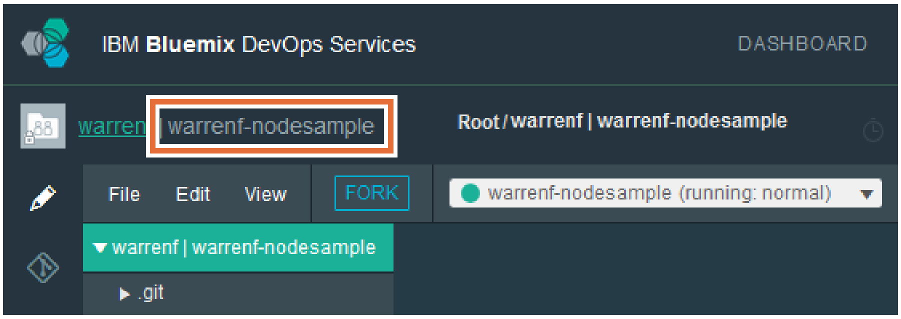
Information

The IBM Bluemix DevOps service hosts a source code repository that supports the Git API. You can use any Git client to work with the artifacts that are stored in the repository.

__ b. Click **Edit Code** to open the IBM Bluemix DevOps service web application.

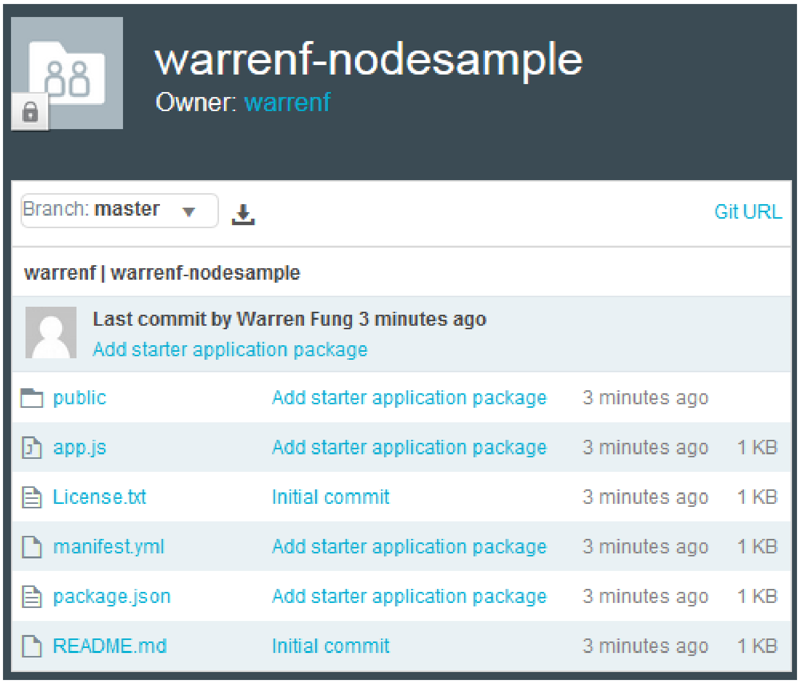
__ 2. Open the project overview page for your application.

__ a. Click ***<your-ibm-id>*-nodesample** to open the project overview page.


IBM Bluemix DevOps Services
DASHBOARD
warren
warrenf-nodesample
Root / warrenf | warrenf-nodesample
File
Edit
View
FORK
warrenf-nodesample (running: normal)
warrenf | warrenf-nodesample
.git

__ 3. Examine the IBM Bluemix DevOps service web application.

__ a. Review the IBM Bluemix DevOps services project overview page.



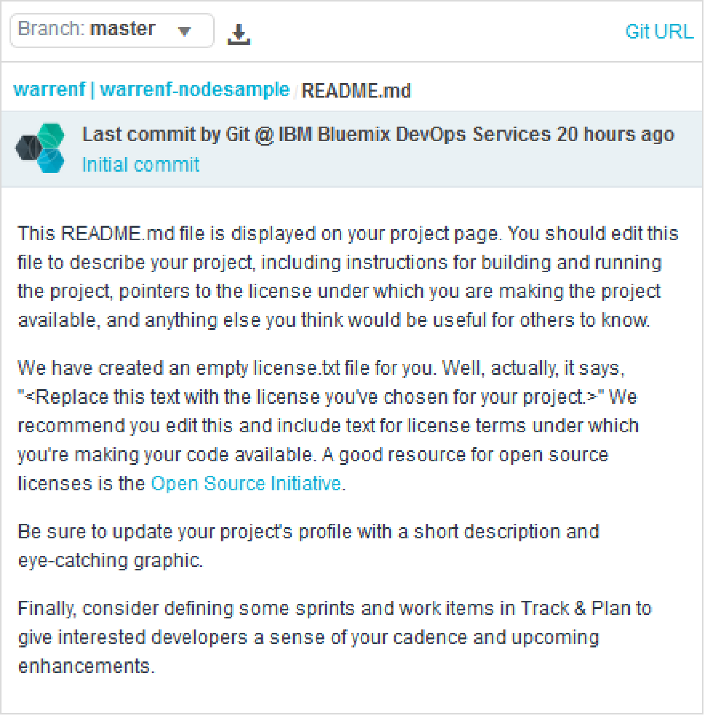
warrenf | warrenf-nodesample

Last commit by Warren Fung 3 minutes ago
Add starter application package

public	Add starter application package	3 minutes ago	
app.js	Add starter application package	3 minutes ago	1 KB
License.txt	Initial commit	3 minutes ago	1 KB
manifest.yml	Add starter application package	3 minutes ago	1 KB
package.json	Add starter application package	3 minutes ago	1 KB
README.md	Initial commit	3 minutes ago	1 KB

The overview page lists your IBM account as the owner of the Git repository. In this exercise, the repository is empty except for a license file and a readme file.

__ b. Open the **readme.md** document.



Branch: master

Git URL

warrenf | warrenf-nodesample / README.md

Last commit by Git @ IBM Bluemix DevOps Services 20 hours ago
Initial commit

This README.md file is displayed on your project page. You should edit this file to describe your project, including instructions for building and running the project, pointers to the license under which you are making the project available, and anything else you think would be useful for others to know.

We have created an empty license.txt file for you. Well, actually, it says, "<Replace this text with the license you've chosen for your project.>" We recommend you edit this and include text for license terms under which you're making your code available. A good resource for open source licenses is the Open Source Initiative.

Be sure to update your project's profile with a short description and eye-catching graphic.

Finally, consider defining some sprints and work items in Track & Plan to give interested developers a sense of your cadence and upcoming enhancements.

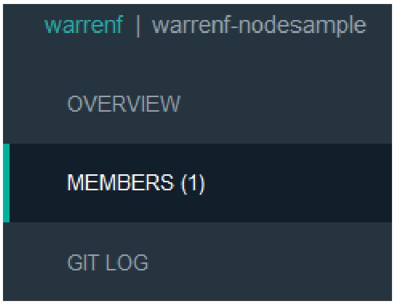
The readme file provides a quick summary to the repository that manages your application artifacts and source code. It is a suggested practice to provide an up-to-date description of your project in this document, especially for projects that are publicly shared for all users.

Information

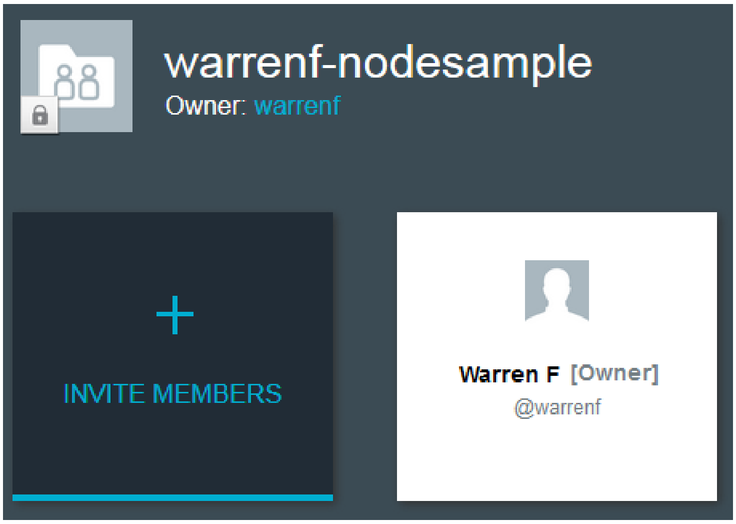
The readme file is written with the Markdown syntax, a lightweight markup language for annotating plain text documents to be displayed as HTML documents. For more information about Markdown, see `http://daringfireball.net/projects/markdown/`

__ 4. View the project members for your application.

__ a. Click **Members** from the navigation bar.



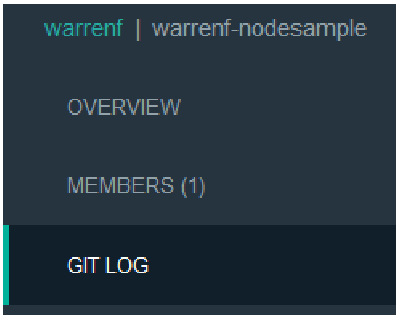
__ b. Examine the list of members for the ***<your-ibm-id>*-nodesample** project.



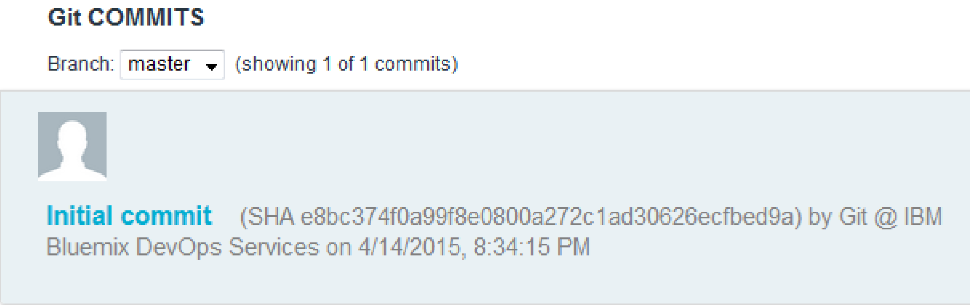
Note the lock icon beside the file folder for the project. By default, your DevOps project is private. The project has one registered member: your IBM ID. You are also the owner of the project.

__ 5. Examine the Git repository log.

__ a. Click **Git log** from the navigation bar.



__ b. Examine the **commit log** for the ***<your-ibm-id>*-nodesample** Git repository commit log.



Git COMMITS

Branch: master (showing 1 of 1 commits)

Initial commit (SHA e8bc374f0a99f8e0800a272c1ad30626ecfbed9a) by Git @ IBM Bluemix DevOps Services on 4/14/2015, 8:34:15 PM

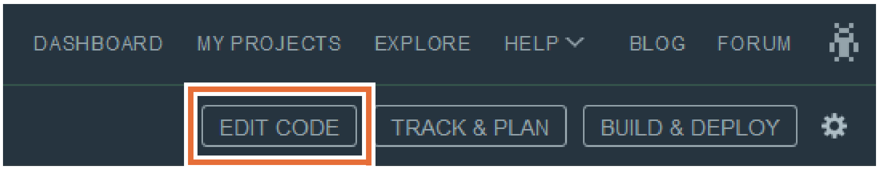
The Create Git Repository wizard that you started from the IBM Bluemix account sets up a Git repository for your application. The wizard adds the sample license and readme file into the repository. The initial commit action has a unique identifier, the user name, and the date stamp.

Part 4: Import the sample application

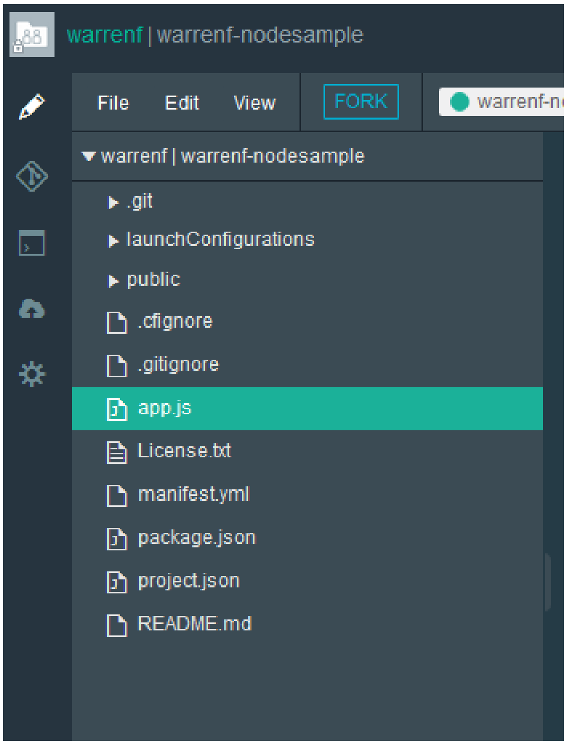
In the second exercise, you set up a sample IBM Bluemix application for IBM SDK for Node.js. Examine the contents of the application in the Git repository.

__ 1. Examine the sample application in your Bluemix DevOps services project.

__ a. Click **Edit Code** under the IBM Bluemix DevOps service site navigation bar.



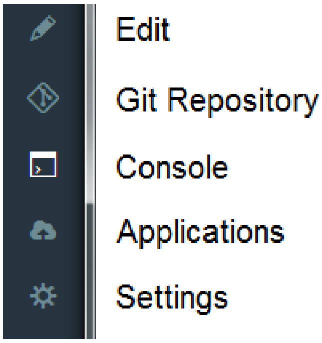
__ b. Confirm that a set of files and directories appear in the project.



```
▾ warrenf | warrenf-nodesample
    ▸ .git
    ▸ launchConfigurations
    ▸ public
      .cfignore
      .gitignore
      app.js
      License.txt
      manifest.yml
      package.json
      project.json
      README.md
```

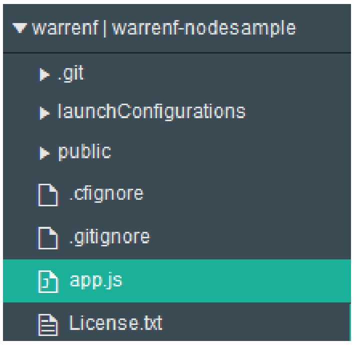
__ 2. Examine the IBM Bluemix DevOps Services Edit Code mode.

__ a. Examine the five options for the Edit Code mode.



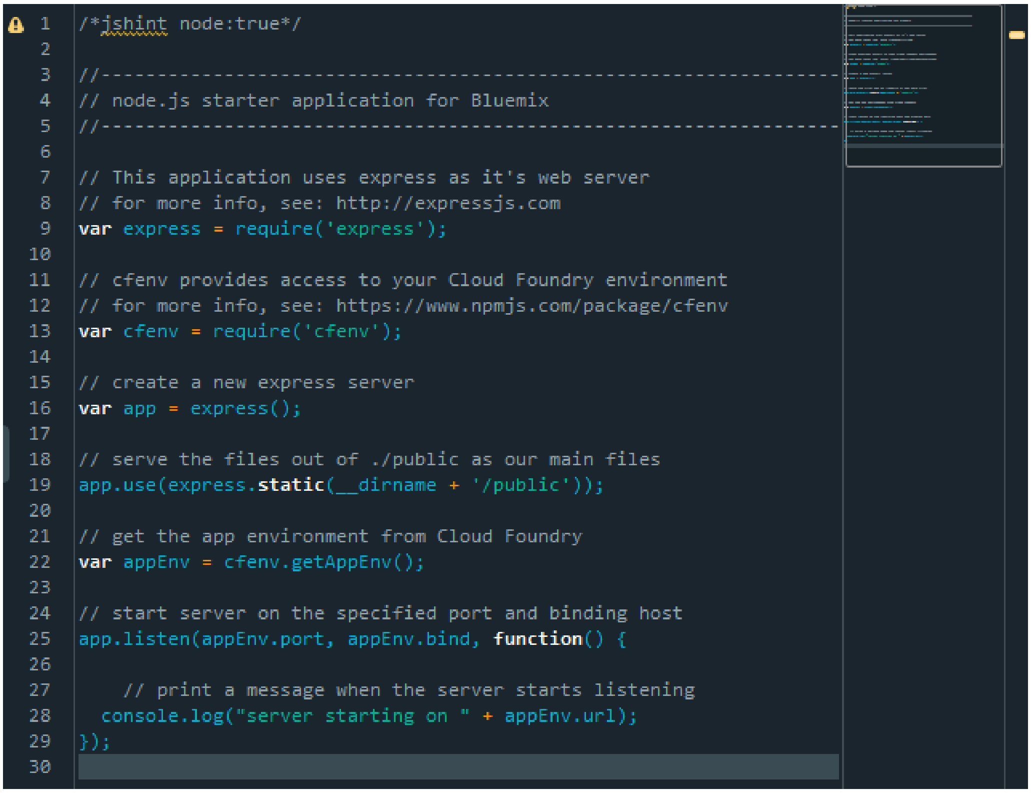
- The **Edit** option provides a graphical editor to manage and update project files.
- With the **Git repository** option, you can select and check-in files from the project into the Git repository.
- The **Console** option displays the directories of your project.
- The **Applications** option lists the deployment options for the application to your IBM Bluemix account.
- The **Settings** option shows the customization options for the Bluemix DevOps Services integrated development environment.

__ b. Click **app.js** from the project directory.



warrenf | warrenf-nodesample
- .git
- launchConfigurations
- public
- .cfignore
- .gitignore
- app.js
- License.txt

Examine the contents of the **app.js** source code in the editor.

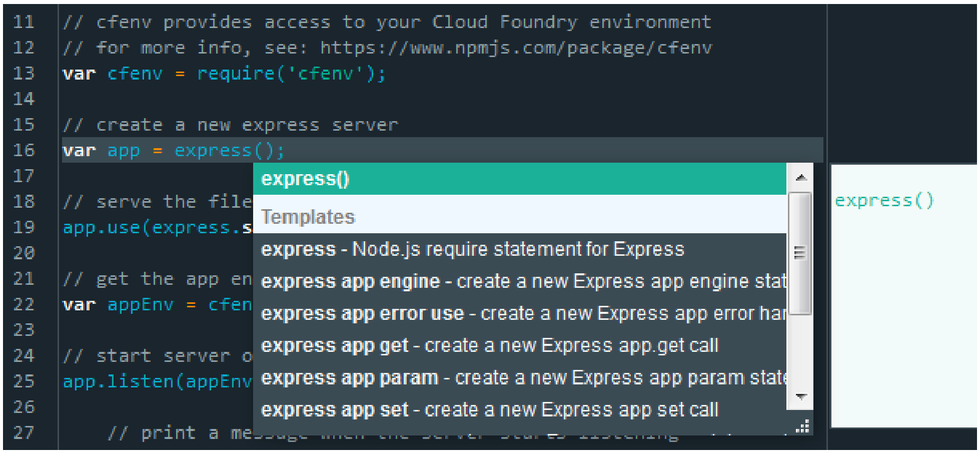


```
/*jshint node:true*/

//------------------------------------------------------------------------------
// node.js starter application for Bluemix
//------------------------------------------------------------------------------

// This application uses express as it's web server
// for more info, see: http://expressjs.com
var express = require('express');

// cfenv provides access to your Cloud Foundry environment
// for more info, see: https://www.npmjs.com/package/cfenv
var cfenv = require('cfenv');

// create a new express server
var app = express();

// serve the files out of ./public as our main files
app.use(express.static(__dirname + '/public'));

// get the app environment from Cloud Foundry
var appEnv = cfenv.getAppEnv();

// start server on the specified port and binding host
app.listen(appEnv.port, appEnv.bind, function() {

    // print a message when the server starts listening
  console.log("server starting on " + appEnv.url);
});
```

The editor in the web application provides the same features as the desktop Eclipse application: syntax highlighting, static code analysis of the JavaScript code, and a preview of the document.

__ c. Place your cursor beside the word **express** in `var app = express()`.

__ d. Press Ctrl + Space bar on your keyboard.



```
// cfenv provides access to your Cloud Foundry environment
// for more info, see: https://www.npmjs.com/package/cfenv
var cfenv = require('cfenv');

// create a new express server
var app = express();

// serve the file
app.use(express.s

// get the app en
var appEnv = cfen

// start server o
app.listen(appEnv

    // print a me
```

express()

Templates

express - Node.js require statement for Express

express app engine - create a new Express app engine stat

express app error use - create a new Express app error ha

express app get - create a new Express app.get call

express app param - create a new Express app param stat

express app set - create a new Express app set call

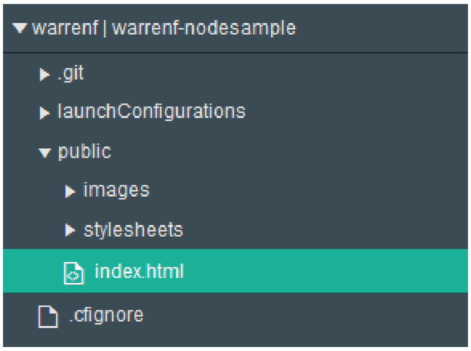
express()

The code completion feature in the JavaScript editor supports node JavaScript modules and the standard JavaScript functions.

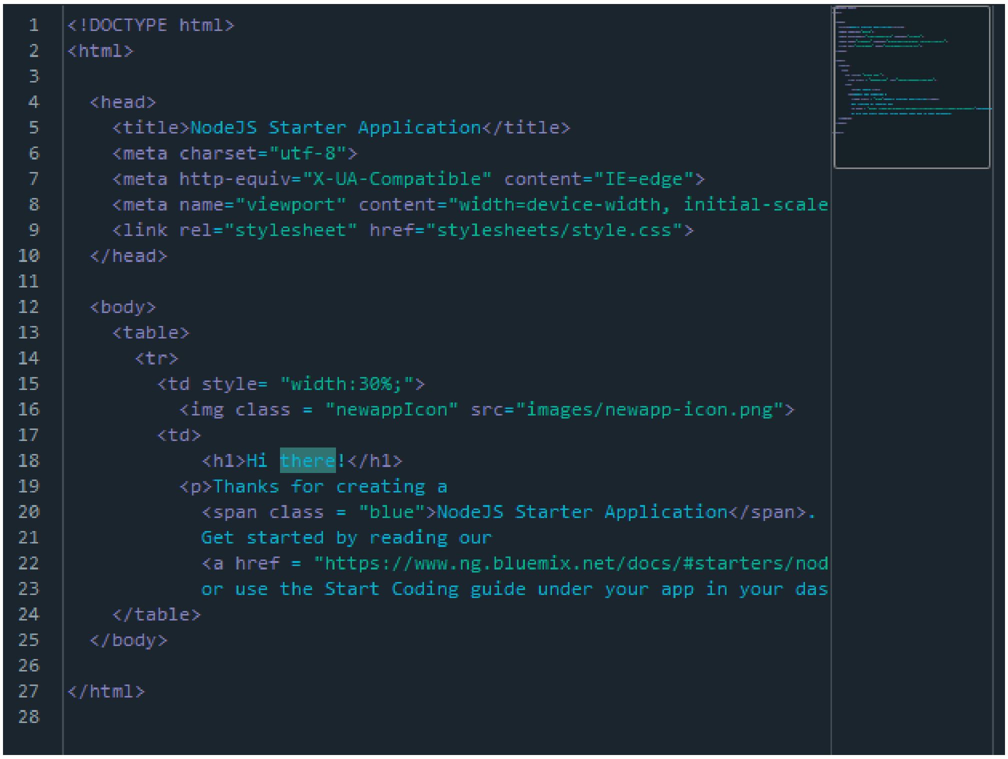
__ 3. Examine the HTML templates for the sample application welcome page.

__ a. In the project directory, expand the **public** folder.

__ b. Click **index.html**.



__ c. Examine **index.html** in the editor.

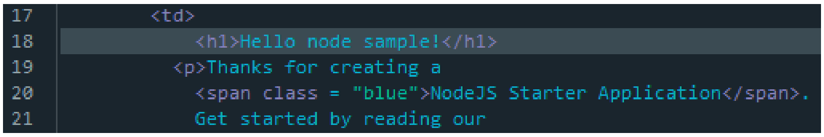


```
1  <!DOCTYPE html>
2  <html>
3
4    <head>
5      <title>NodeJS Starter Application</title>
6      <meta charset="utf-8">
7      <meta http-equiv="X-UA-Compatible" content="IE=edge">
8      <meta name="viewport" content="width=device-width, initial-scale
9      <link rel="stylesheet" href="stylesheets/style.css">
10   </head>
11
12   <body>
13     <table>
14       <tr>
15         <td style= "width:30%;">
16           <img class = "newappIcon" src="images/newapp-icon.png">
17         <td>
18             <h1>Hi there!</h1>
19           <p>Thanks for creating a
20             <span class = "blue">NodeJS Starter Application</span>.
21             Get started by reading our
22             <a href = "https://www.ng.bluemix.net/docs/#starters/nod
23             or use the Start Coding guide under your app in your das
24     </table>
25   </body>
26
27 </html>
28
```

The built-in editor also highlights and validates HTML web page markup.

__ d. Select the sentence in **line 18**.

__ e. Change the phrase to “**`Hello node sample!`**”.



```
17         <td>
18             <h1>Hello node sample!</h1>
19           <p>Thanks for creating a
20             <span class = "blue">NodeJS Starter Application</span>.
21             Get started by reading our
```

__ f. Press Ctrl + S to save your changes.

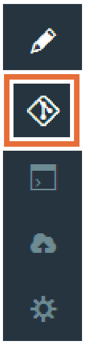
To view the effects of the source code changes, deploy and test your application on your IBM Bluemix account. In the next section, you save your changes to the Git repository before pushing the changes to your IBM Bluemix application.

Part 5: Commit your changes to the Git repository

Commit the source code, configuration files, and other artifacts to the Git repository in your IBM Bluemix DevOps project. Deploy the changes to your application from the Git repository to the application in your IBM Bluemix account.

__ 1. Switch to the **Git Repository** option in the Edit Code mode.

__ a. In the navigation bar in the Edit Code mode, click **Git Repository**.

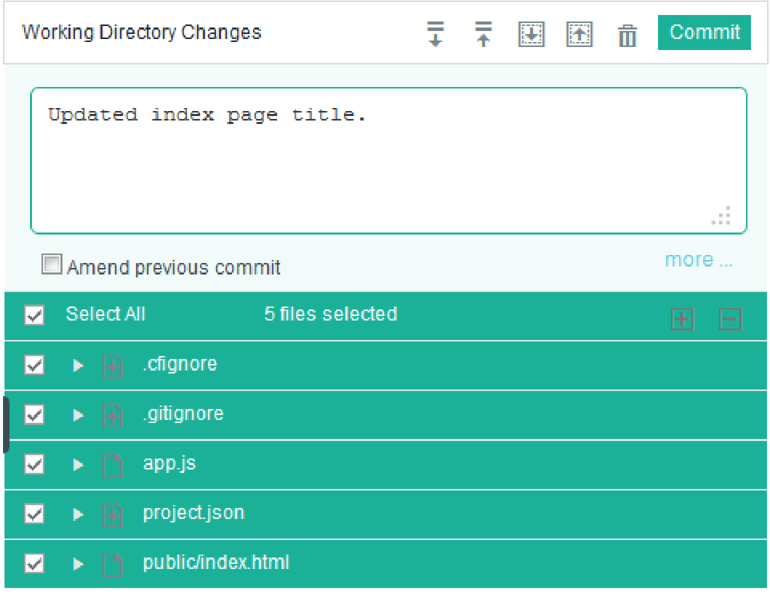


__ 2. Add and commit the changes in the project to the Git repository.

__ a. Examine the **Working directory changes** pane on the right side. The working directory changes window lists the files that you added or modified to the workspace.

__ b. Enter **`Updated index page title.`** in the comment window.

__ c. Select the **Select All** check box to highlight all the files.



```
Updated index page title.
```

□ Amend previous commit

☑ Select All 5 files selected

☑ .cfignore

☑ .gitignore

☑ app.js

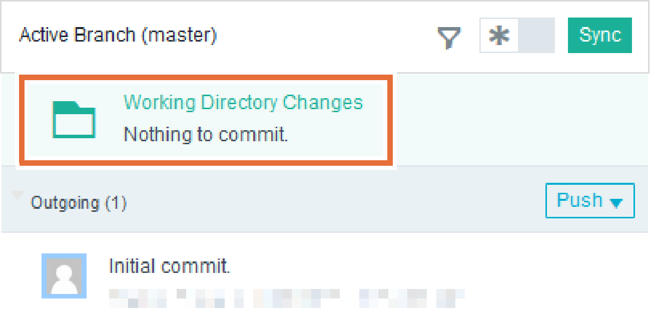
☑ project.json

☑ public/index.html

__ d. Click **Commit** to save the files into the Git repository.

__ e. Examine the **Working Directory Changes** view on the left side.

__ f. Confirm that there are no more updated files to commit from the working directory.



You can revert to the last saved version of your source code in the Git repository.

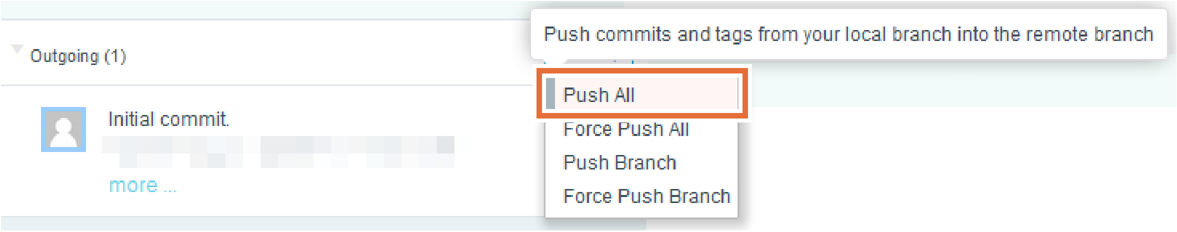
Part 6: Deploy the application from the Git repository to IBM Bluemix

There are two ways to deploy an application from your IBM Bluemix DevOps services project to your IBM Bluemix account: deploy from the Git repository, and deploy directly from your project.

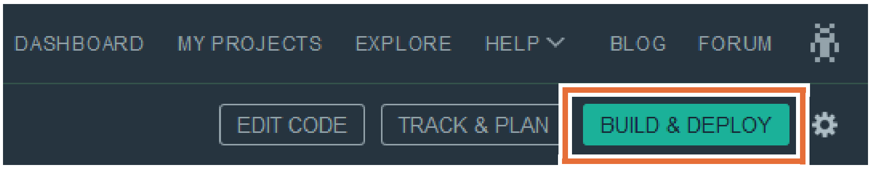
If more than one developer is working on a DevOps services project, consider checking in your changes and pushing the most recent revision from the Git repository.

__ 1. Deploy the updated application from the Git repository to your IBM Bluemix account.

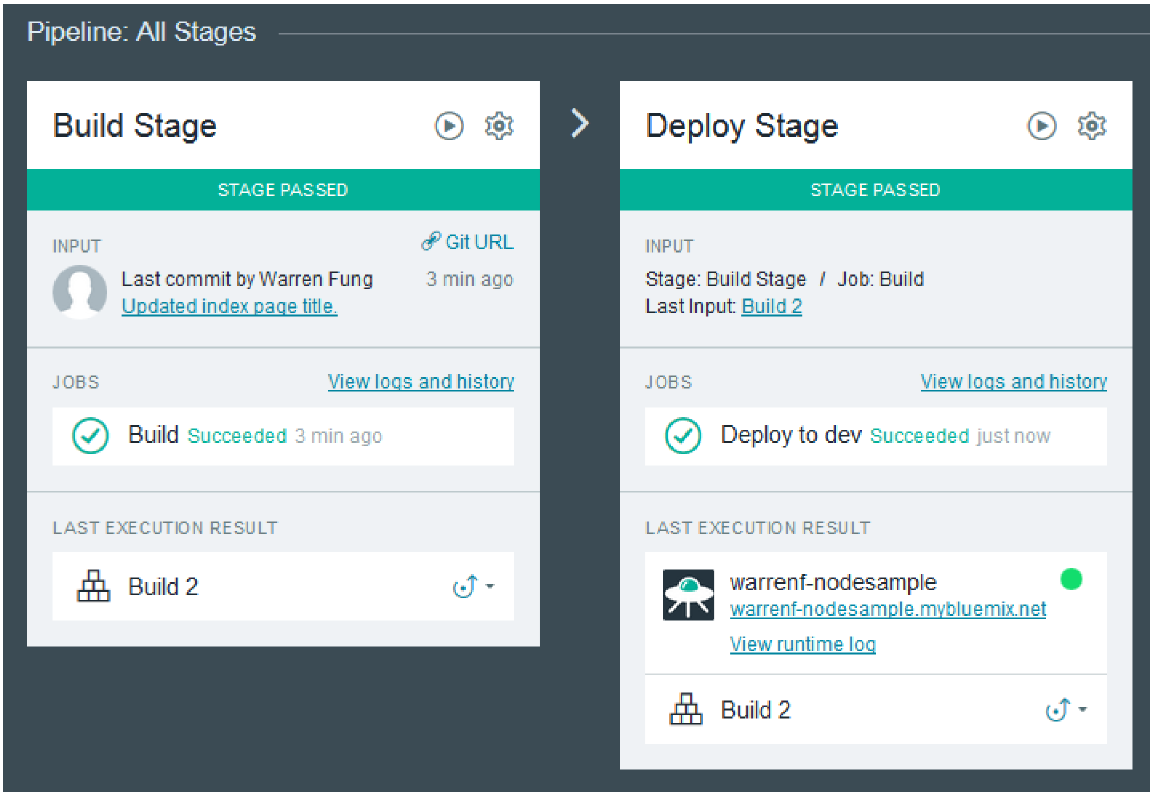
__ a. In the **Outgoing** section beneath **Working Directory Changes**, select **Push All**.



__ b. Switch to the **Build & Deploy** mode.



__ c. Examine the delivery pipeline. Wait until the build and deploy tasks complete successfully.



Pipeline: All Stages

Build Stage

STAGE PASSED

INPUT

Git URL

Last commit by Warren Fung 3 min ago
Updated index page title.

JOBS View logs and history

Build Succeeded 3 min ago

LAST EXECUTION RESULT

Build 2

Deploy Stage

STAGE PASSED

INPUT

Stage: Build Stage / Job: Build
Last Input: Build 2

JOBS View logs and history

Deploy to dev Succeeded just now

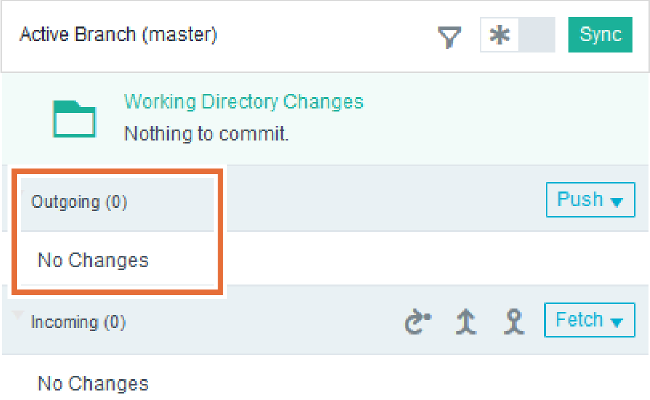
LAST EXECUTION RESULT

warrenf-nodesample
warrenf-nodesample.mybluemix.net
View runtime log

Build 2

__ d. Switch to the “Git repository” page.

__ e. Confirm that there are no more changes to push from the IBM Bluemix DevOps services project.



Active Branch (master)

Sync

Working Directory Changes
Nothing to commit.

Outgoing (0)

Push

No Changes

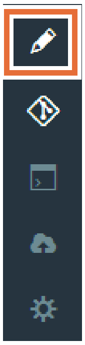
Incoming (0)

Fetch

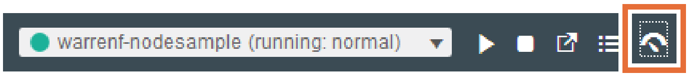
No Changes

__ 2. Examine the application dashboard for the ***<your-ibm-id>*-nodesample** application.

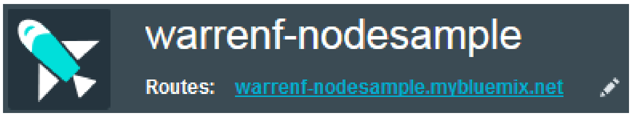
__ a. From your IBM Bluemix DevOps services web page, switch to the **edit** mode.



__ b. Click **Application Dashboard** from the server toolbar.

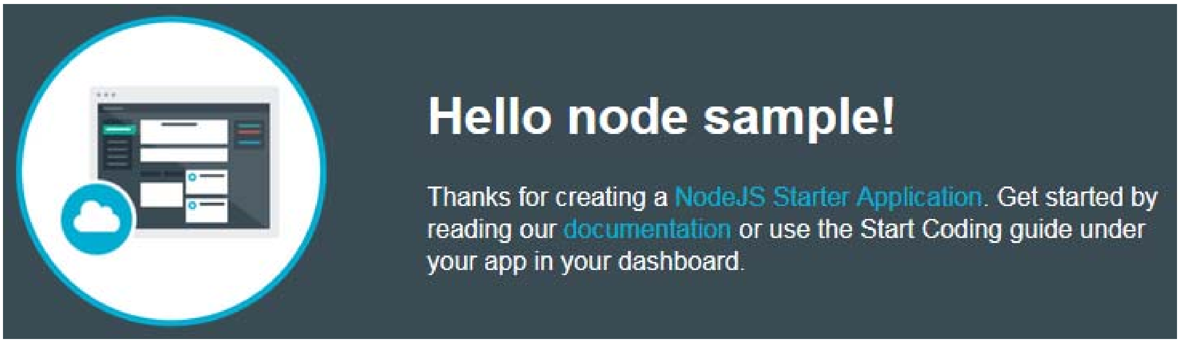


__ c. In the IBM Bluemix application dashboard, open the routes link: **`http://`*<your-ibm-id>*`-nodesample.mybluemix.net`**


warrenf-nodesample

Routes: warrenf-nodesample.mybluemix.net

__ d. Confirm that the sample application web page states “**Hello node sample!**”


Hello node sample!

Thanks for creating a NodeJS Starter Application. Get started by reading our documentation or use the Start Coding guide under your app in your dashboard.

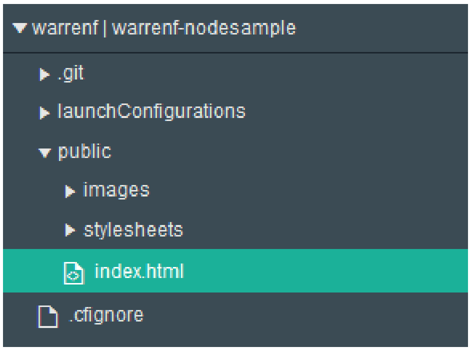
Part 7: Deploy the application directly from your project workspace to IBM Bluemix

IBM Bluemix DevOps Services also supports publishing changes from your development project to a server environment on IBM Bluemix. With this technique, you can quickly test changes to your code on an actual Bluemix account.

__ 1. Edit the message in the sample application web page.

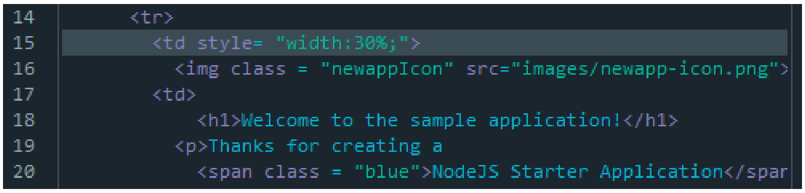
__ a. In the IBM Bluemix DevOps service website, switch to the **Edit Code** mode.

__ b. Open **index.html** in the **public** folder.



- warrenf | warrenf-nodesample
  - .git
  - launchConfigurations
  - public
    - images
    - stylesheets
    - index.html
  - .cfignore

__ c. In line 18, change the heading to “`Welcome to the sample application!`”

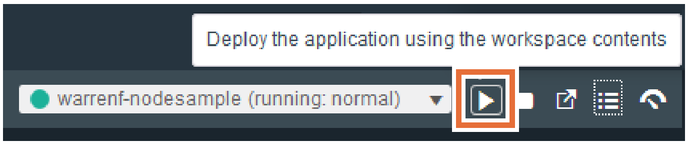


```
14    <tr>
15     <td style= "width:30%;">
16      <img class = "newappIcon" src="images/newapp-icon.png">
17     <td>
18       <h1>Welcome to the sample application!</h1>
19      <p>Thanks for creating a
20       <span class = "blue">NodeJS Starter Application</spar
```

__ d. Press Ctrl + S to save your changes.

__ 2. Deploy the application directly from your project workspace.

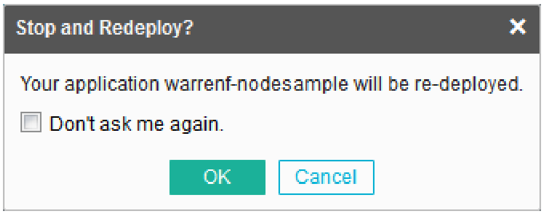
__ a. In the server toolbar, click **Deploy**.



Deploy the application using the workspace contents

● warrenf-nodesample (running: normal) ▼

__ b. DevOps Services prompts whether you want to stop and redeploy the Bluemix application. Click **OK** to confirm the operation.



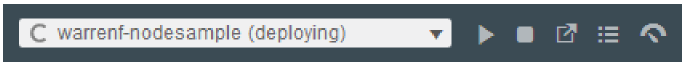
Stop and Redeploy? ✕

Your application warrenf-nodesample will be re-deployed.

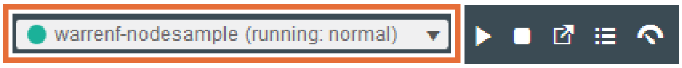
□ Don't ask me again.

OK Cancel

__ c. Wait until the environment finishes restarting and deploying the application.



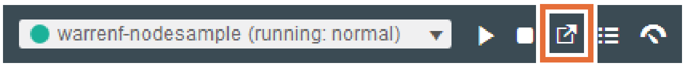
warrenf-nodesample (deploying) ▼



● warrenf-nodesample (running: normal) ▼

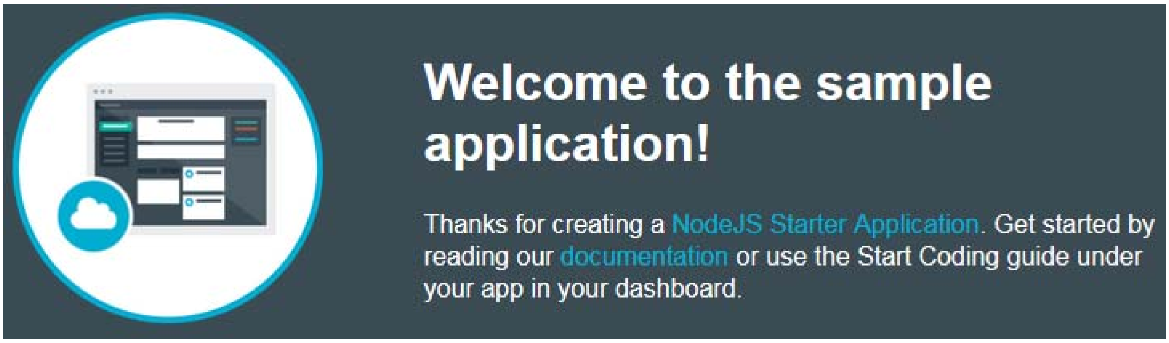
__ 3. Confirm that the changes appear in the application.

__ a. Click the **Application URL** toolbar button to view the changes to your application.



● warrenf-nodesample (running: normal) ▼

__ b. Verify that the updated heading appears in the sample application web page.



Welcome to the sample application!

Thanks for creating a NodeJS Starter Application. Get started by reading our documentation or use the Start Coding guide under your app in your dashboard.

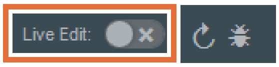
Part 8: *Automatically push changes from your DevOps service project to IBM Bluemix*

With the stop and redeploy option, you must manually trigger an application restart through the server toolbar. Although this option is more convenient than having to commit your changes to the Git repository, it can be disruptive to your software development workflow for minor changes.

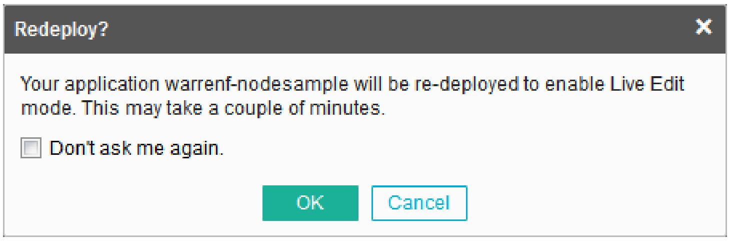
The *Bluemix Live Sync* feature automatically pushes any saved changes in the project workspace to a linked IBM Bluemix application. With IBM SDK for Node.js applications, you can update static files and view the updates on IBM Bluemix without restarting your application.

__ 1. Enable the *live edit* feature.

__ a. Beside the servers toolbar, click **LiveEdit**.



__ b. Click **OK** to redeploy the application and enable live edit mode.

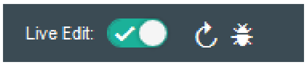


Redeploy?

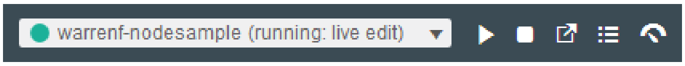
Your application warrenf-nodesample will be re-deployed to enable Live Edit mode. This may take a couple of minutes.

□ Don't ask me again.

__ c. Confirm that the live edit mode is enabled.



__ d. Verify that the IBM Bluemix application is running in a live edit mode.

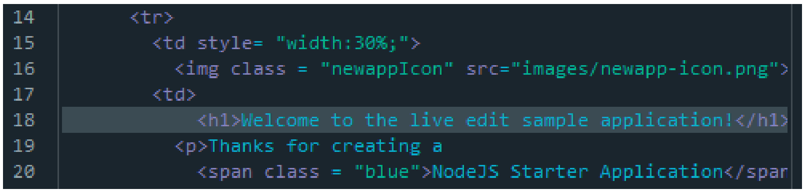


warrenf-nodesample (running: live edit)

__ 2. Change the sample application web page heading.

__ a. Open the **index.html** web page.

__ b. Edit the heading to read “**`Welcome to the live edit sample application!`**”



```
<tr>
  <td style= "width:30%;">
    <img class = "newappIcon" src="images/newapp-icon.png">
  <td>
      <h1>Welcome to the live edit sample application!</h1>
    <p>Thanks for creating a
      <span class = "blue">NodeJS Starter Application</spar
```

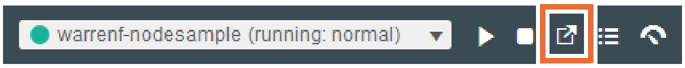
Note

With the live edit feature, you do not need to save your static files. IBM Bluemix DevOps Services pushes the changes to your IBM Bluemix account automatically.

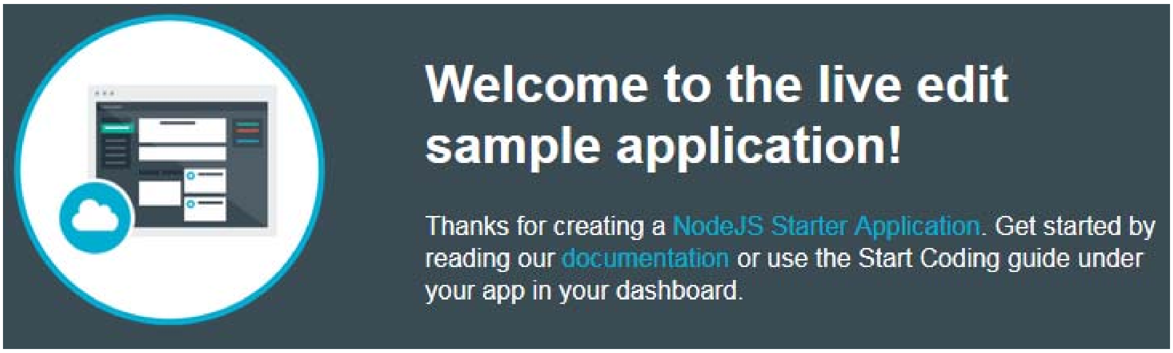
Examples of static files include HTML web pages and CSS stylesheets.

__ 3. Confirm that the changes appear in your IBM Bluemix application.

__ a. Click the **Application URL** toolbar button to view the changes to your application.



__ a. Verify that the updated heading appears in the sample application web page.



Welcome to the live edit sample application!

Thanks for creating a NodeJS Starter Application. Get started by reading our documentation or use the Start Coding guide under your app in your dashboard.

__ 4. Log out of IBM Bluemix and IBM Bluemix DevOps services.

__ 5. Close your web browser.

End of exercise

Exercise review and wrap-up

In the first part of the exercise, you set up an IBM Bluemix DevOps service project to manage your Bluemix application. You imported the sample application that is written for the IBM SDK for Node.js server run time.

In the second part of the exercise, you saved your changes into the Git repository. Through the Build and Deploy mode, you pushed your committed source code changes to your IBM Bluemix application.

In the last part of the exercise, you deployed your changes directly from the project workspace. You also used the Bluemix Live Sync feature to push changes to static files without restarting the application.

Exercise 4. Bluemix with Cloudant

What this exercise is about

In this exercise, you work with a Bluemix node application that includes a Cloudant DB service. You see how to work with a Bluemix Cloudant database from a JavaScript application and from the Cloudant console.

What you should be able to do

After completing this exercise, you should be able to:

- Sign on to Bluemix
- Add a Node.js Cloudant web starter application from the Bluemix boilerplates
- Download the application package to the workstation
- Review the application source code
- Push the application to Bluemix
- Connect to Cloudant
- Create a database
- Add data to the database
- Access the Cloudant console and documentation
- List the databases and get the data
- Test the application to see that the data is added

Introduction

In an earlier exercise, you created an SDK for the Node.js Bluemix application.

You also installed the Cloud Foundry CLI on your local workstation. Later, you also installed the Eclipse development environment and the Bluemix plug-in for Eclipse.

In this exercise, you create another Node.js application with the Cloudant NoSQL DB service from a Bluemix boilerplate template.

Requirements

You must have an IBM Bluemix account and must complete exercises 1 and 2 before doing this exercise.

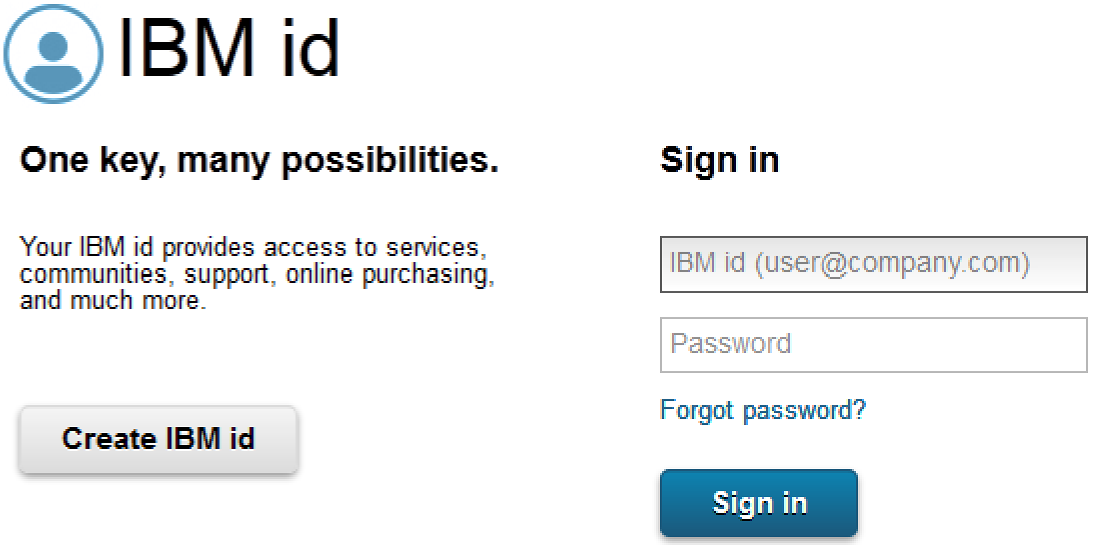
Exercise instructions

Preface

This exercise depends on the availability of the IBM Bluemix cloud and connectivity from your own workstation to the cloud environment.

Part 1: Sign on to the Bluemix cloud

From your web browser, type the address `bluemix.net`. Then, sign on to IBM Bluemix with your IBM ID and password.



IBM id

One key, many possibilities.

Your IBM id provides access to services, communities, support, online purchasing, and much more.

When you are signed on, you see the application that you created earlier in the Bluemix catalog.

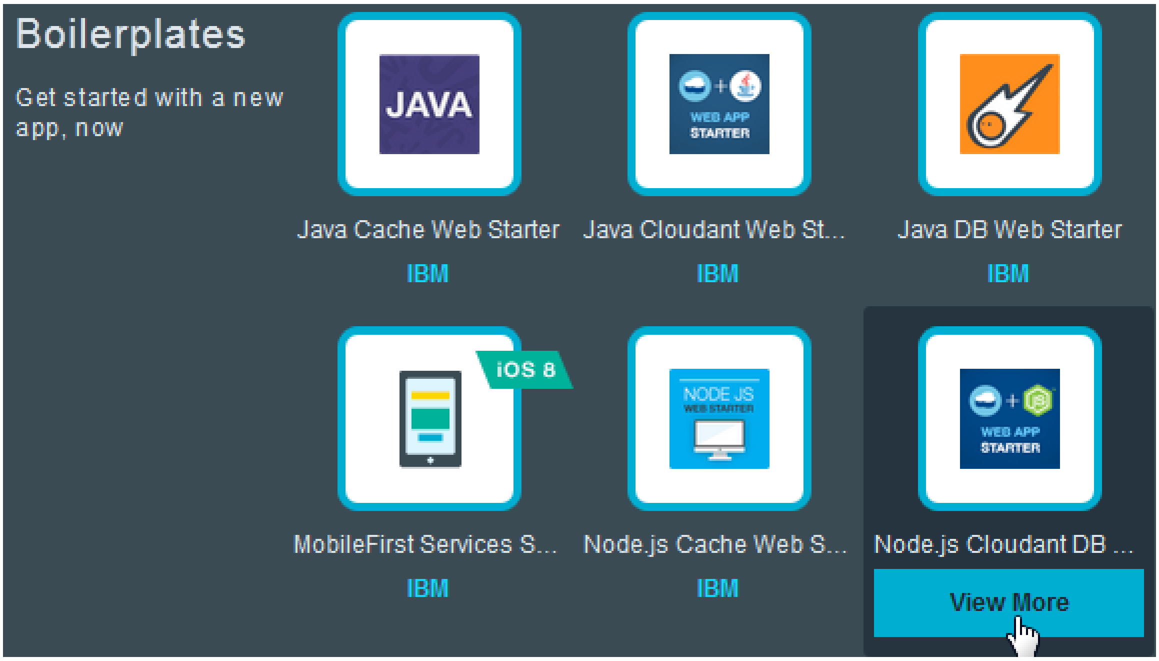
Part 2: Add a Node.js application with a Cloudant DB service

In this part, you add the Node.js application with the Cloudant DB service and Monitoring and Analytics service from the Bluemix services boilerplate.

__ 1. Add the Node.js Cloudant DB Web Starter to your Bluemix environment.

  __ a. From the Bluemix main page, click the **Catalog** tab.

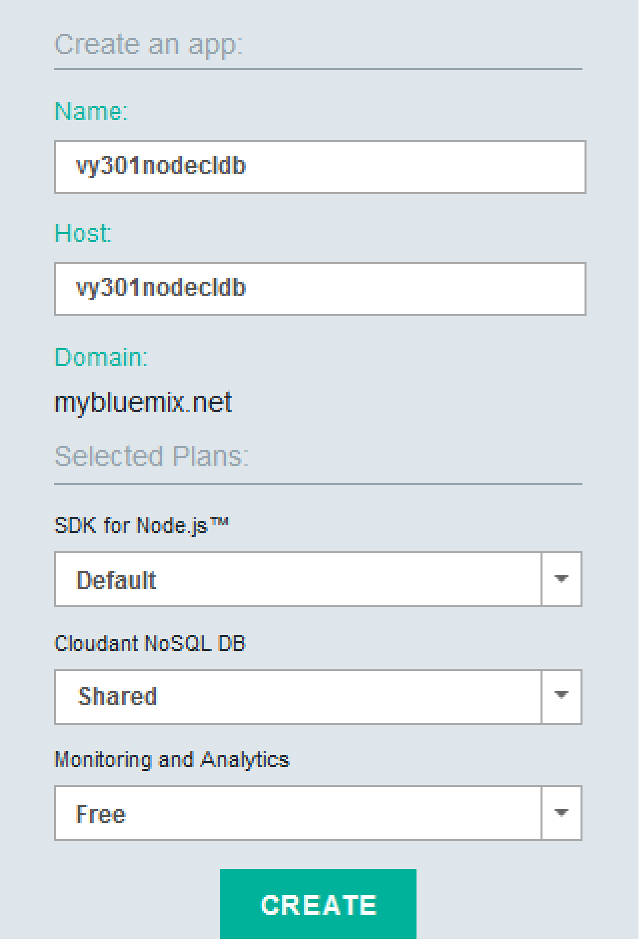
__ b. From the Boilerplates section of the Catalog, click **Node.js Cloudant DB Web Starter**.



__ c. In the Create an App area for the Web Starter application, type this information and click **Create**:

- Name: *`Your application name`*
- Host: *`same as above`*
- Domain: `mybluemix.net`
- SDK for Node.js: `Default`
- Cloudant NoSQL DB: `Shared`

- Monitoring and Analytics: `Free`



Create an app:

Name: vy301nodecldb

Host: vy301nodecldb

Domain: mybluemix.net

Selected Plans:

SDK for Node.js™: Default

Cloudant NoSQL DB: Shared

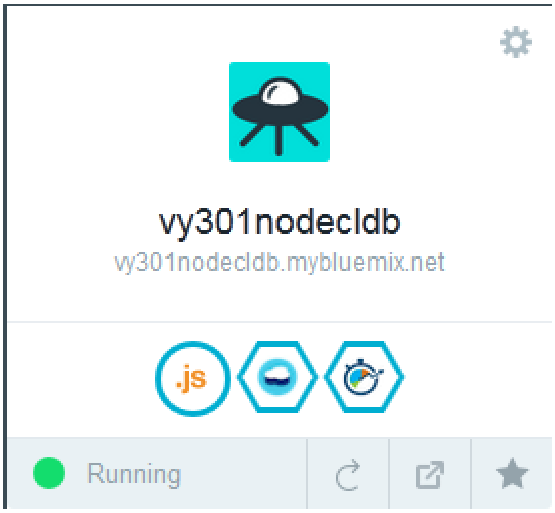
Monitoring and Analytics: Free

Important

The name of your node application differs from the name that is shown in the figures of this exercise. Each application must have a unique name in IBM Bluemix. Use the naming convention that you decided on in the earlier exercises.

The Web Starter application is successfully installed.

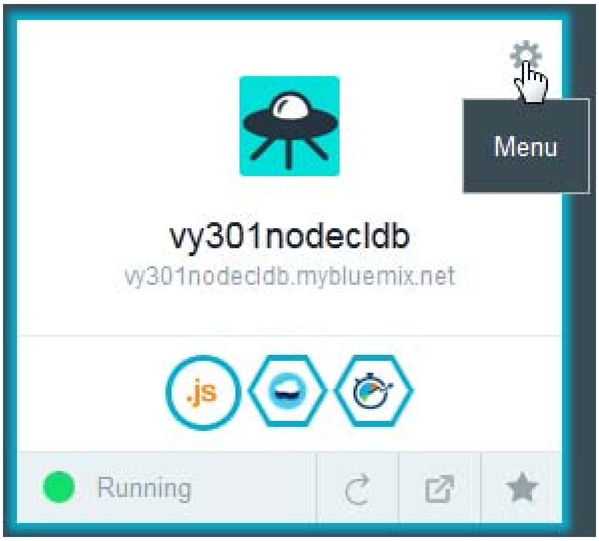
__ d. Click the option to return to the Blueprint Dashboard. After a short while, the application is started and you see the started Node.js application with the Cloudant DB service and the Monitoring and Analytics service.



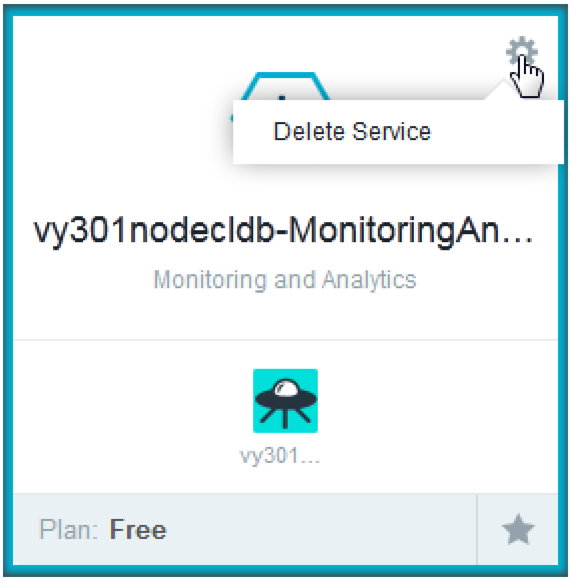
__ e. You can click the routes link for the application to test that the application runs correctly.

__ 2. Because this exercise uses only the Node.js application and Cloudant DB service, you can delete the Monitoring and Analytics service.

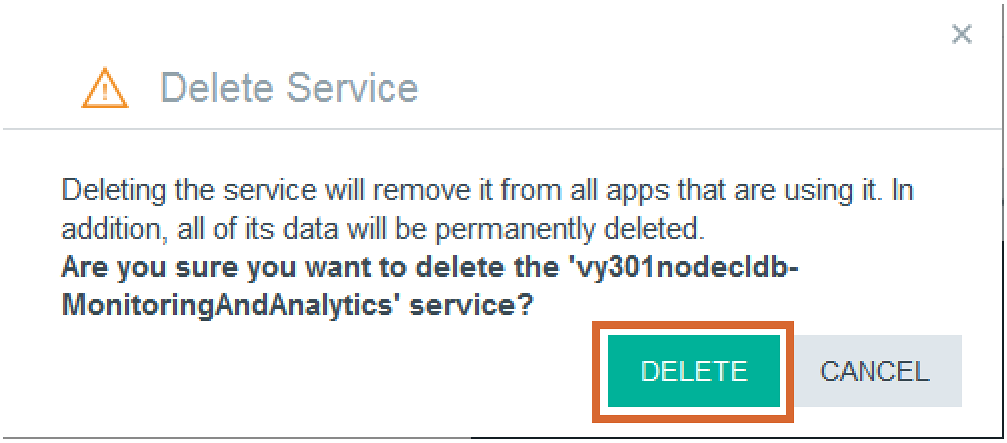
__ a. From the menu at the upper right of the application, click **Stop app**.



__ b. Now delete the Monitoring and Analytics service.



__ c. Confirm that you want to delete the service.



Delete Service

Deleting the service will remove it from all apps that are using it. In addition, all of its data will be permanently deleted.
Are you sure you want to delete the 'vy301nodecldb-MonitoringAndAnalytics' service?

Deleting the services returns unused resources to your Bluemix account.

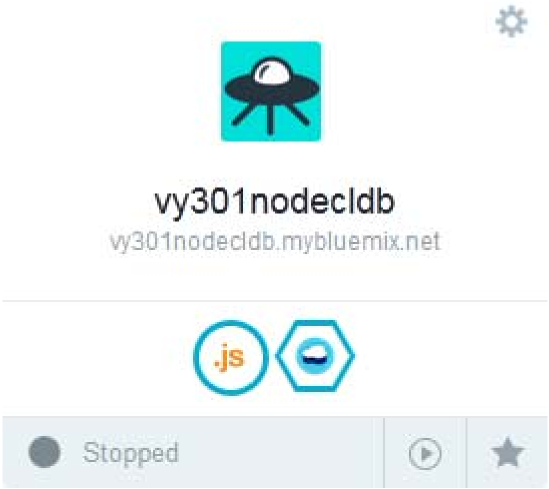
__ d. Leave the application in the stopped state on Bluemix.

Part 3: Review the service environment variables

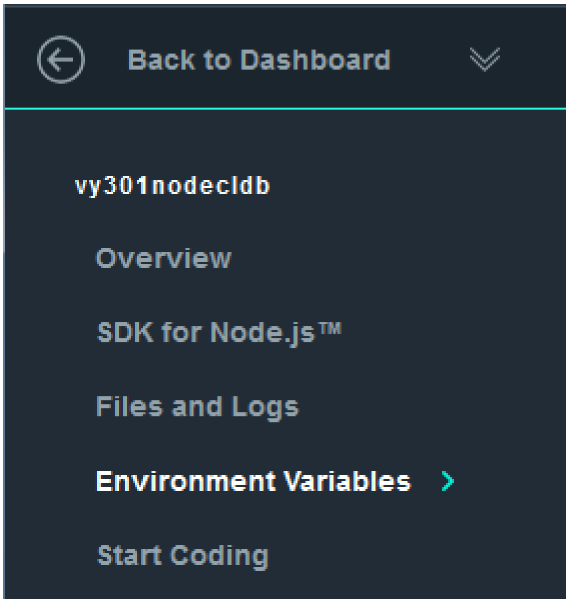
A number of environment variables are created for you when you added and bound the service to your application.

__ 1. Review and export the VCAP_SERVICES environmental variables.

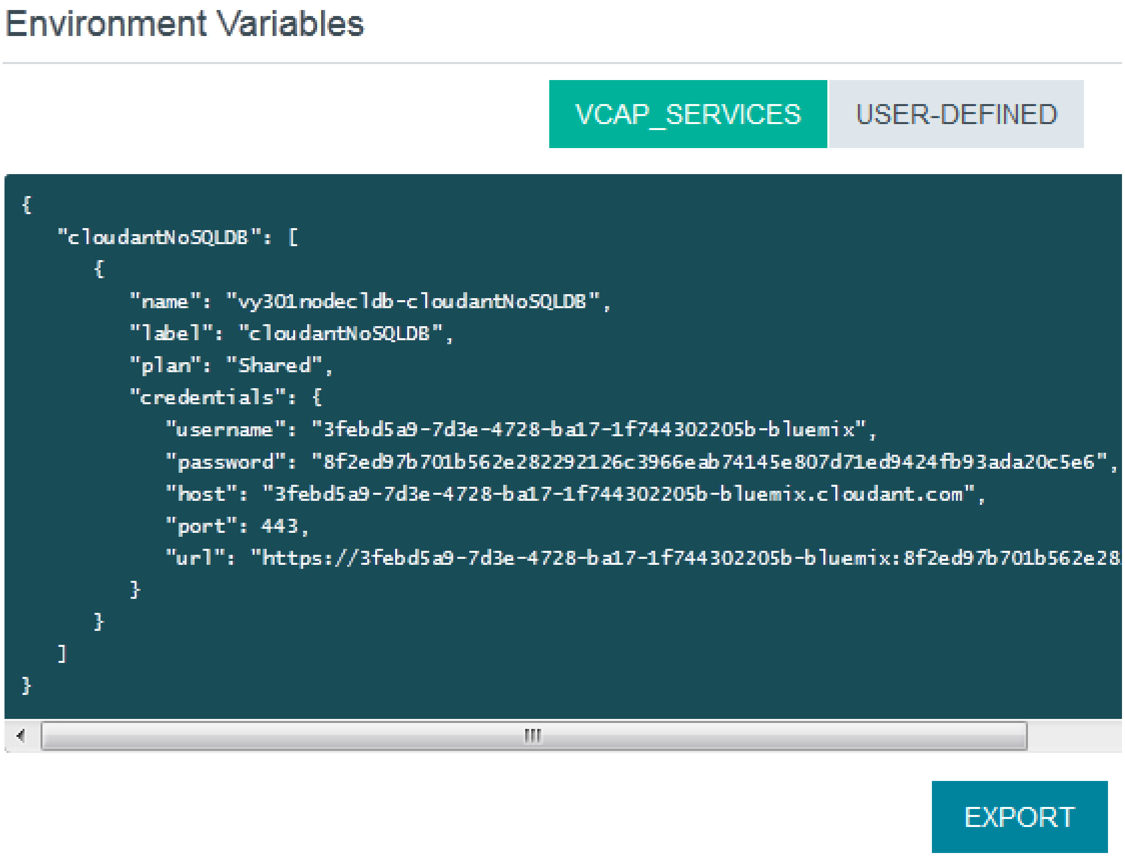
__ a. From the dashboard, click the application icon.



__ b. From the navigation pane, click **Environment Variables**.



__ c. The VCAP_SERVICES environment variables are displayed in the window.



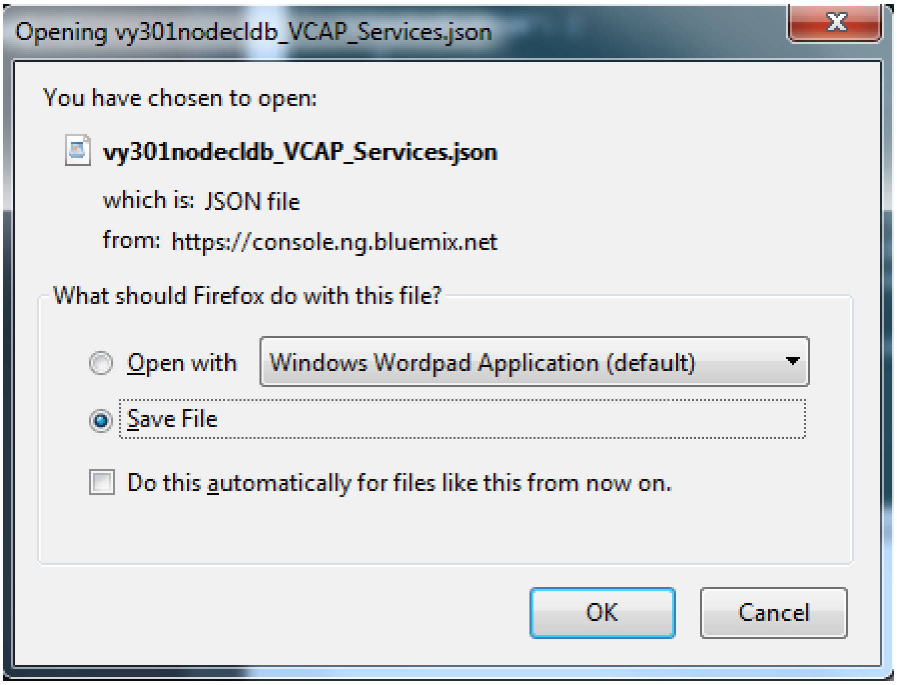
```
{
   "cloudantNoSQLDB": [
      {
         "name": "vy301nodecldb-cloudantNoSQLDB",
         "label": "cloudantNoSQLDB",
         "plan": "Shared",
         "credentials": {
            "username": "3febd5a9-7d3e-4728-ba17-1f744302205b-bluemix",
            "password": "8f2ed97b701b562e282292126c3966eab74145e807d71ed9424fb93ada20c5e6",
            "host": "3febd5a9-7d3e-4728-ba17-1f744302205b-bluemix.cloudant.com",
            "port": 443,
            "url": "https://3febd5a9-7d3e-4728-ba17-1f744302205b-bluemix:8f2ed97b701b562e28
         }
      }
   ]
}
```

These environment variables contain important data that you need for accessing the Cloudant database.

__ 2. Export the VCAP_SERVICES variables.

__ a. Click **Export.**

__ b. In the dialog box that opens, you can either open the JSON file with the editor, or save the variables to a file for later use.



Opening vy301nodecldb_VCAP_Services.json

You have chosen to open:

vy301nodecldb_VCAP_Services.json

which is: JSON file

from: https://console.ng.bluemix.net

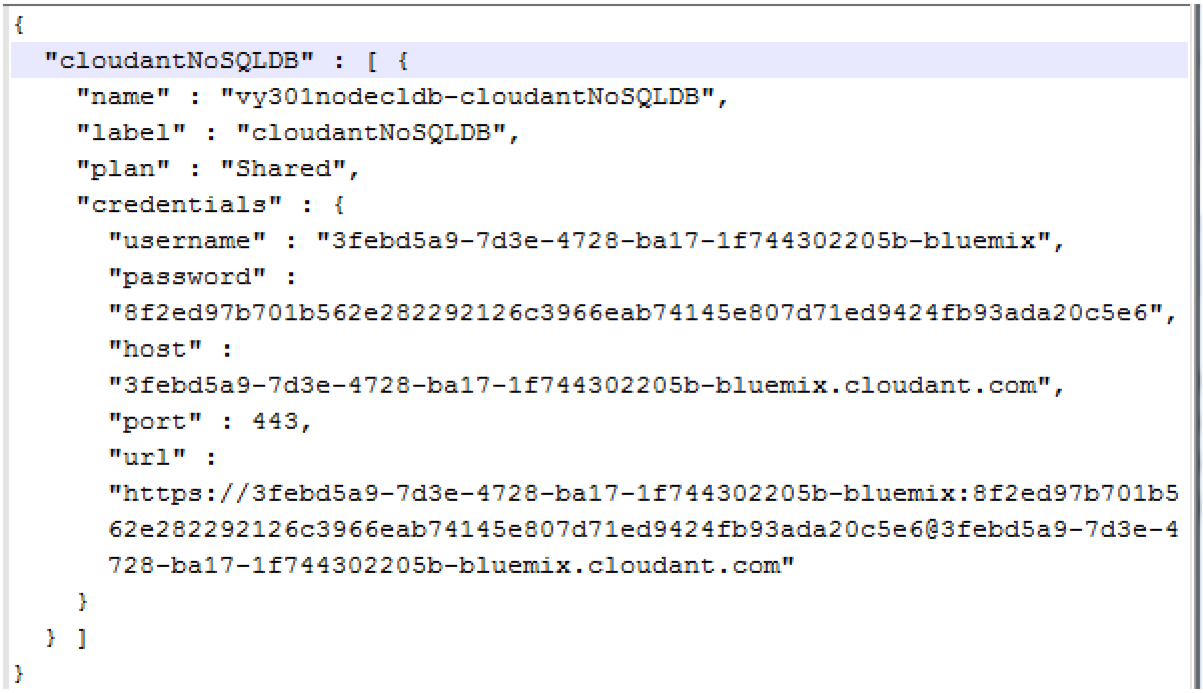
What should Firefox do with this file?

○ Open with Windows Wordpad Application (default)

● Save File

□ Do this automatically for files like this from now on.

__ c. The VCAP_SERVICES environment variables are shown in an editor.



```
{
  "cloudantNoSQLDB" : [ {
    "name" : "vy301nodecldb-cloudantNoSQLDB",
    "label" : "cloudantNoSQLDB",
    "plan" : "Shared",
    "credentials" : {
      "username" : "3febd5a9-7d3e-4728-ba17-1f744302205b-bluemix",
      "password" :
      "8f2ed97b701b562e282292126c3966eab74145e807d71ed9424fb93ada20c5e6",
      "host" :
      "3febd5a9-7d3e-4728-ba17-1f744302205b-bluemix.cloudant.com",
      "port" : 443,
      "url" :
      "https://3febd5a9-7d3e-4728-ba17-1f744302205b-bluemix:8f2ed97b701b5
      62e282292126c3966eab74145e807d71ed9424fb93ada20c5e6@3febd5a9-7d3e-4
      728-ba17-1f744302205b-bluemix.cloudant.com"
    }
  } ]
}
```

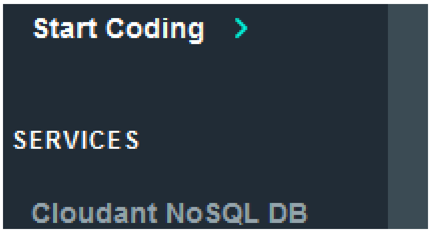
These variables are used when you access the Cloudant database.

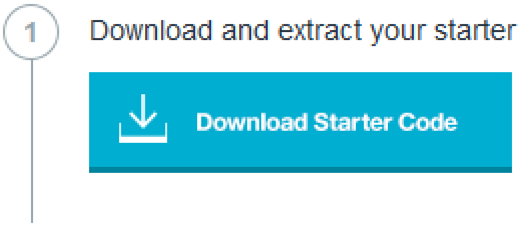
Part 4: Review the starter application on the workstation

In this part, you download and review the application on the workstation.

__ 1. Download the starter code for the node application from the Bluemix environment.

__ a. After selecting the Node.js starter application in Bluemix, click **Start Coding**. Then, click **Download Starter Code**.





__ b. Save the file on your workstation.

__ 2. Start Eclipse if it is not already started.

__ 3. Create a project in Eclipse.

__ c. From the menu, click **File > New> Project**.

__ d. In the New Project dialog box, click **General > Project**. Then, click **Next**.

__ e. In the **Project name** field, type the name that is identical to the name that you gave the application on Bluemix.

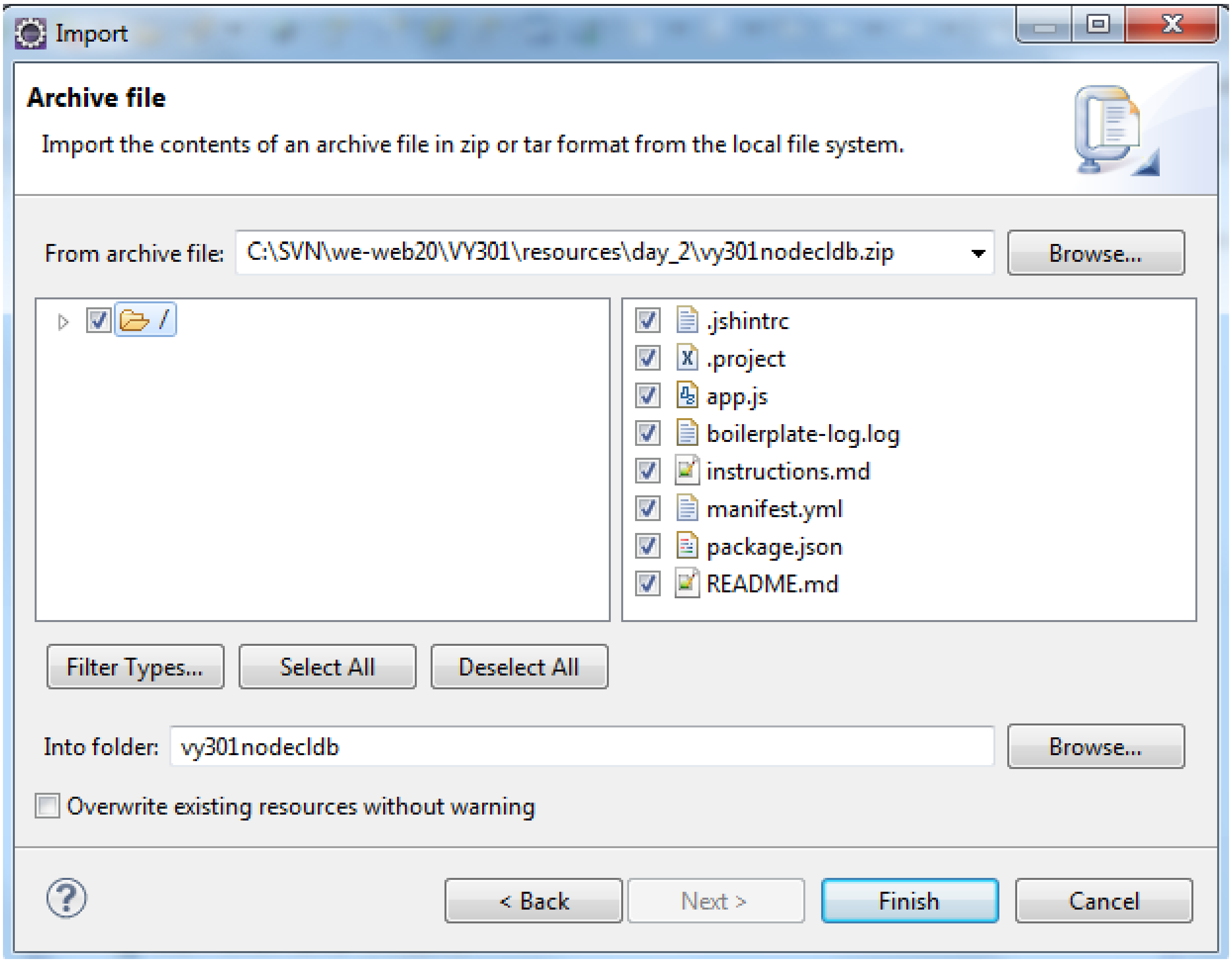
__ f. Click **Finish**.

__ 4. Import the starter code into Eclipse.

__ a. Right-click the project that you created in the Project Explorer and click **Import**.

__ b. Expand **General**, select **Archive File**, and click **Next**.

__ c. Browse to the directory where the starter package is downloaded to. Ensure that all the files are selected and click **Finish**.



Import

Archive file

Import the contents of an archive file in zip or tar format from the local file system.

From archive file: C:\SVN\we-web20\VY301\resources\day_2\vy301nodecldb.zip

☑ /

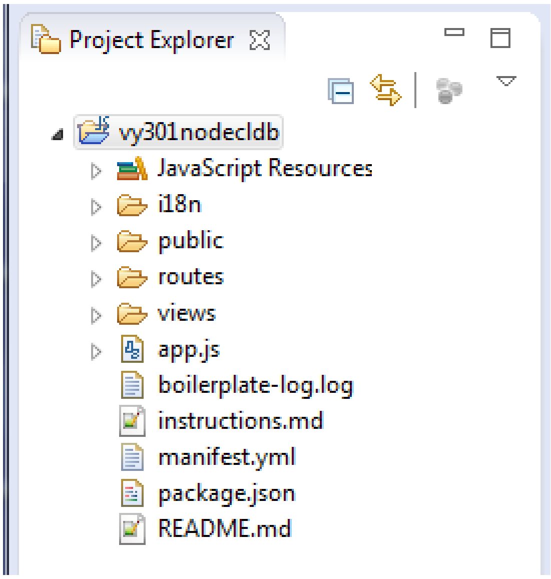
- ☑ .jshintrc
- ☑ .project
- ☑ app.js
- ☑ boilerplate-log.log
- ☑ instructions.md
- ☑ manifest.yml
- ☑ package.json
- ☑ README.md

Into folder: vy301nodecldb

□ Overwrite existing resources without warning

__ d. Select **Yes To All** to overwrite ‘package.json’ in the folder for the project.

__ e. You see the files in the Project Explorer.



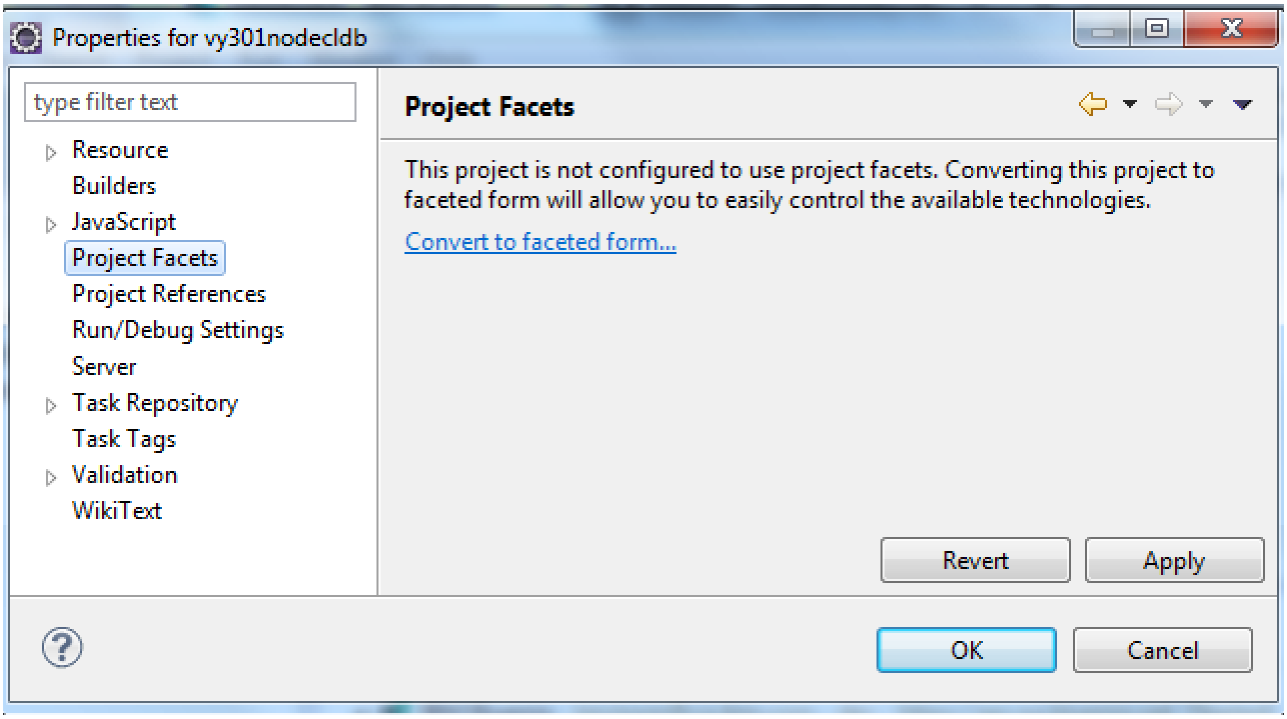
Project Explorer

```
vy301nodecldb
  JavaScript Resources
  i18n
  public
  routes
  views
  app.js
  boilerplate-log.log
  instructions.md
  manifest.yml
  package.json
  README.md
```

__ 5. Change the project in Eclipse to make it recognizable to Bluemix as a Node.js project.

__ a. In the Project Explorer, right-click the project and click **Properties**.

__ b. In the Properties dialog box, select **Project Facets**. Then, click the option **Convert to faceted form**.



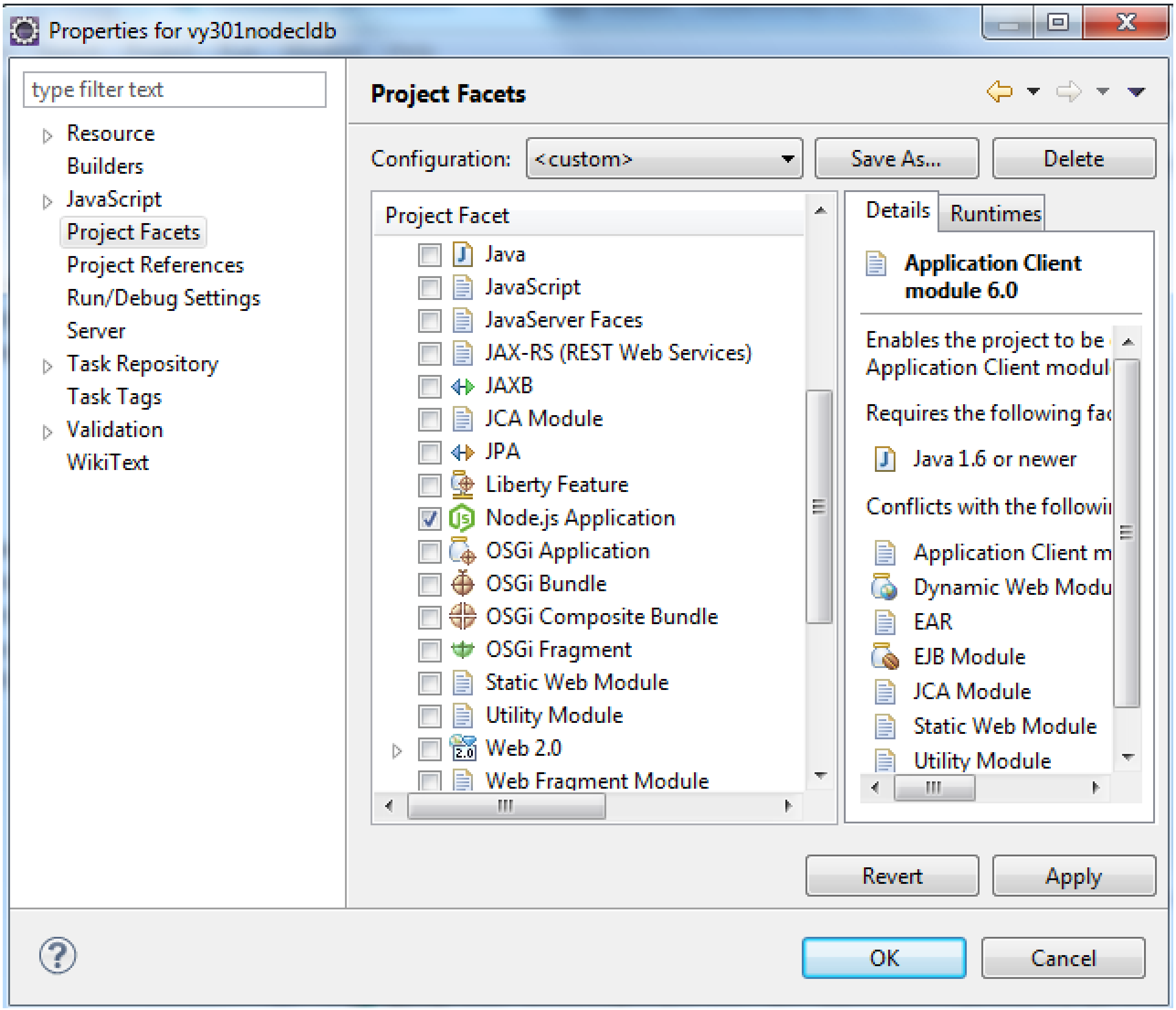
Properties for vy301nodecldb

Project Facets

This project is not configured to use project facets. Converting this project to faceted form will allow you to easily control the available technologies.

Convert to faceted form...

__ c. In the Project Facets area, select **Node.js Application**.



Project Facet

- □ Java
- □ JavaScript
- □ JavaServer Faces
- □ JAX-RS (REST Web Services)
- □ JAXB
- □ JCA Module
- □ JPA
- □ Liberty Feature
- ☑ Node.js Application
- □ OSGi Application
- □ OSGi Bundle
- □ OSGi Composite Bundle
- □ OSGi Fragment
- □ Static Web Module
- □ Utility Module
- □ Web 2.0
- □ Web Fragment Module

Application Client module 6.0

Enables the project to be
Application Client modul

Requires the following fac

Java 1.6 or newer

Conflicts with the followi

Application Client m
Dynamic Web Modu
EAR
EJB Module
JCA Module
Static Web Module
Utility Module

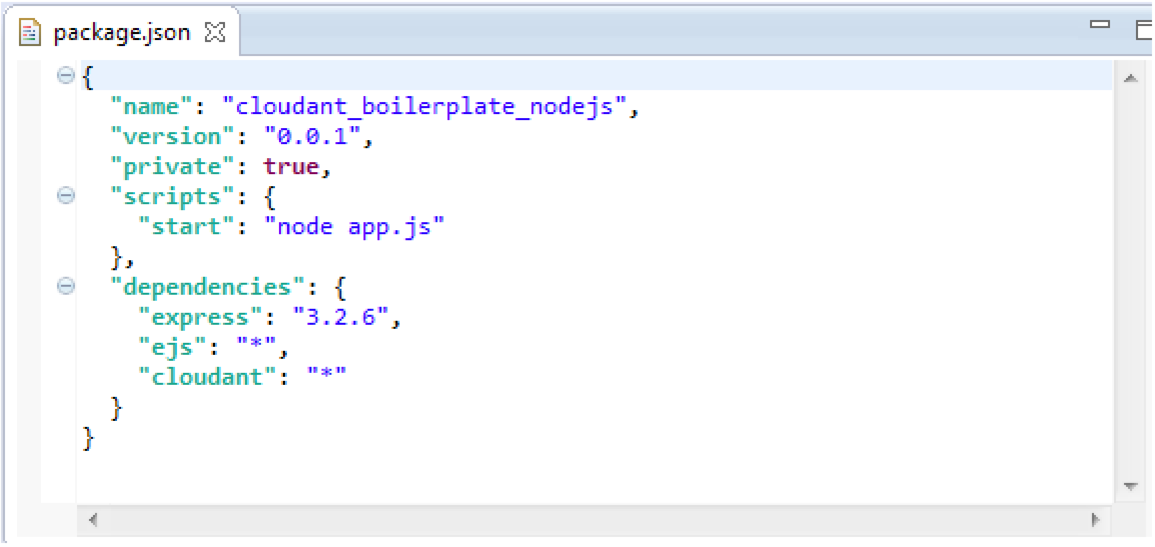
__ d. Click **Apply**. Then, click **OK**.
The project is now set as a Node.js application and can be published to the IBM Bluemix server.

__ 6. Review the source code. The package that was created with the Node.js and Cloudant Web App starter boilerplate in Bluemix includes the source code to add a Cloudant database and data to the database.

__ a. Expand the project in the Project Explorer.

__ b. Double-click **package.json** to open it in the Eclipse editor.

__ c. You see that the application runs a script with the command node app.js.



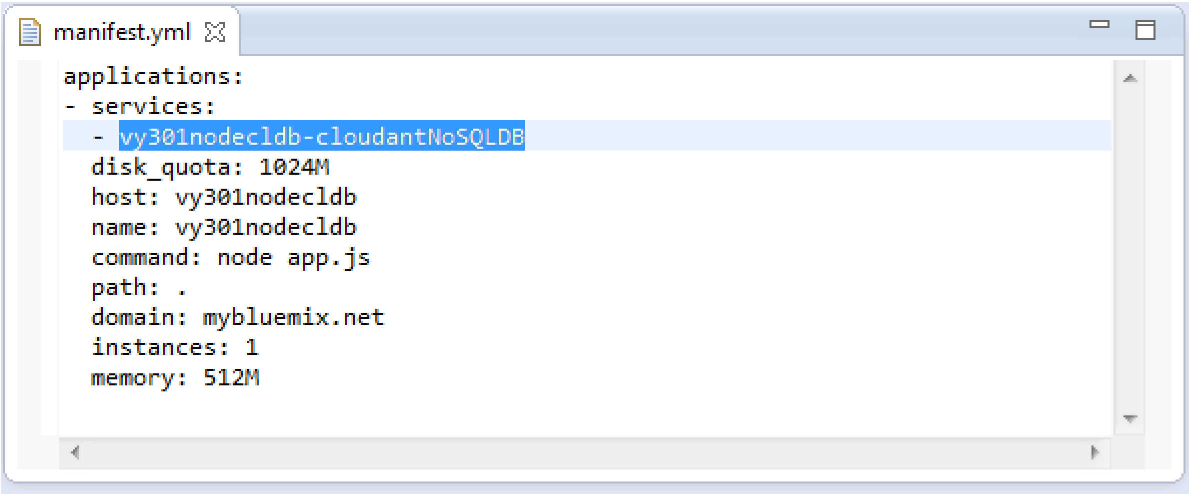
```
{
  "name": "cloudant_boilerplate_nodejs",
  "version": "0.0.1",
  "private": true,
  "scripts": {
    "start": "node app.js"
  },
  "dependencies": {
    "express": "3.2.6",
    "ejs": "*",
    "cloudant": "*"
  }
}
```

The application also has dependencies that include the Express framework and Cloudant.

__ d. Close package.json.

__ e. In the Project Explorer, double-click **manifest.yml**.

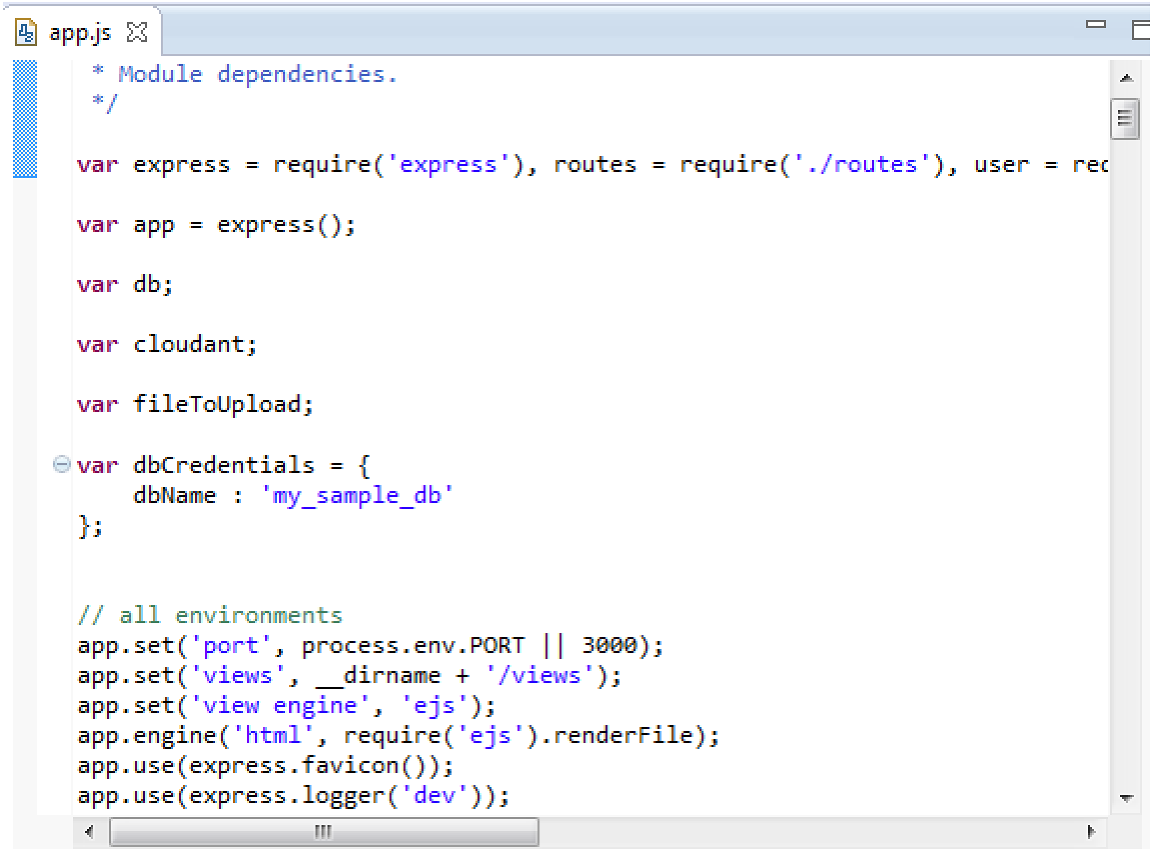
__ f. In the editor, you see the name of the application and that the application has a Cloudant service.



```
applications:
- services:
  - vy301nodecldb-cloudantNoSQLDB
  disk_quota: 1024M
  host: vy301nodecldb
  name: vy301nodecldb
  command: node app.js
  path: .
  domain: mybluemix.net
  instances: 1
  memory: 512M
```

__ g. Close **manifest.yml**.

__ h. In the Project Explorer, double-click **app.js**. The file contains the JavaScript code for accessing the Cloudant DB service.



```
 * Module dependencies.
 */

var express = require('express'), routes = require('./routes'), user = rec

var app = express();

var db;

var cloudant;

var fileToUpload;

var dbCredentials = {
    dbName : 'my_sample_db'
};


// all environments
app.set('port', process.env.PORT || 3000);
app.set('views', __dirname + '/views');
app.set('view engine', 'ejs');
app.engine('html', require('ejs').renderFile);
app.use(express.favicon());
app.use(express.logger('dev'));
```

Notice the database name in the source code. This database is added to Cloudant the first time that the application runs.

__ i. Scroll down in the code and you see the initDBConnection() function. In addition to the database name, the host, port, user, password, and URL properties are all populated from the process.env.VCAP_SERVICES variable. This code demonstrates how the Node.js application reads the environment variables that you reviewed earlier in this exercise.

```
if(process.env.VCAP_SERVICES) {
    var vcapServices = JSON.parse(process.env.VCAP_SERVICES);
    if(vcapServices.cloudantNoSQLDB) {
        dbCredentials.host = vcapServices.cloudantNoSQLDB[0].credentials.host;
        dbCredentials.port = vcapServices.cloudantNoSQLDB[0].credentials.port;
        dbCredentials.user = vcapServices.cloudantNoSQLDB[0].credentials.username;
        dbCredentials.password =
vcapServices.cloudantNoSQLDB[0].credentials.password;
        dbCredentials.url = vcapServices.cloudantNoSQLDB[0].credentials.url;
    }
    console.log('VCAP Services: '+JSON.stringify(process.env.VCAP_SERVICES));
}
```

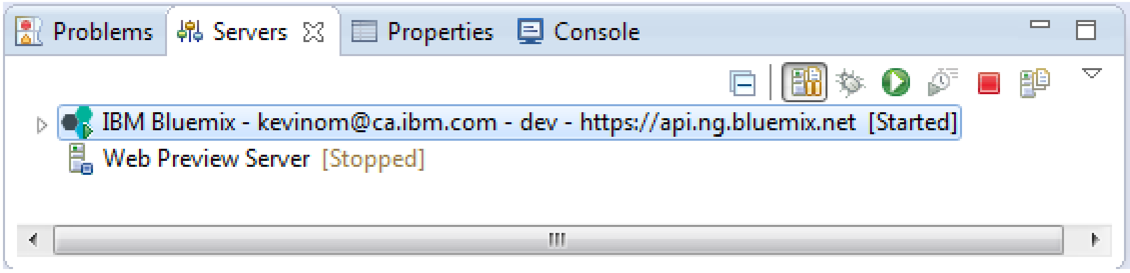
Part 5: Run the application on Bluemix

In this part, you push the application from the workstation and run it on Bluemix.

The IBM Eclipse Tools for Bluemix is installed in an earlier exercise. The IBM Bluemix server is displayed to the list of servers on the Servers tab in the Eclipse workbench.

__ 1. Start the IBM Bluemix server.

__ a. If the IBM Bluemix server is not already started, start it from the **Servers** tab in Eclipse.



IBM Bluemix - kevinom@ca.ibm.com - dev - https://api.ng.bluemix.net [Started]
Web Preview Server [Stopped]

__ b. When started, you can expand the IBM Bluemix option to see the applications that are running on Bluemix. Because the application with the same name is stopped on Bluemix itself, you do not see it in the list of started applications on the IBM Bluemix server.

Important

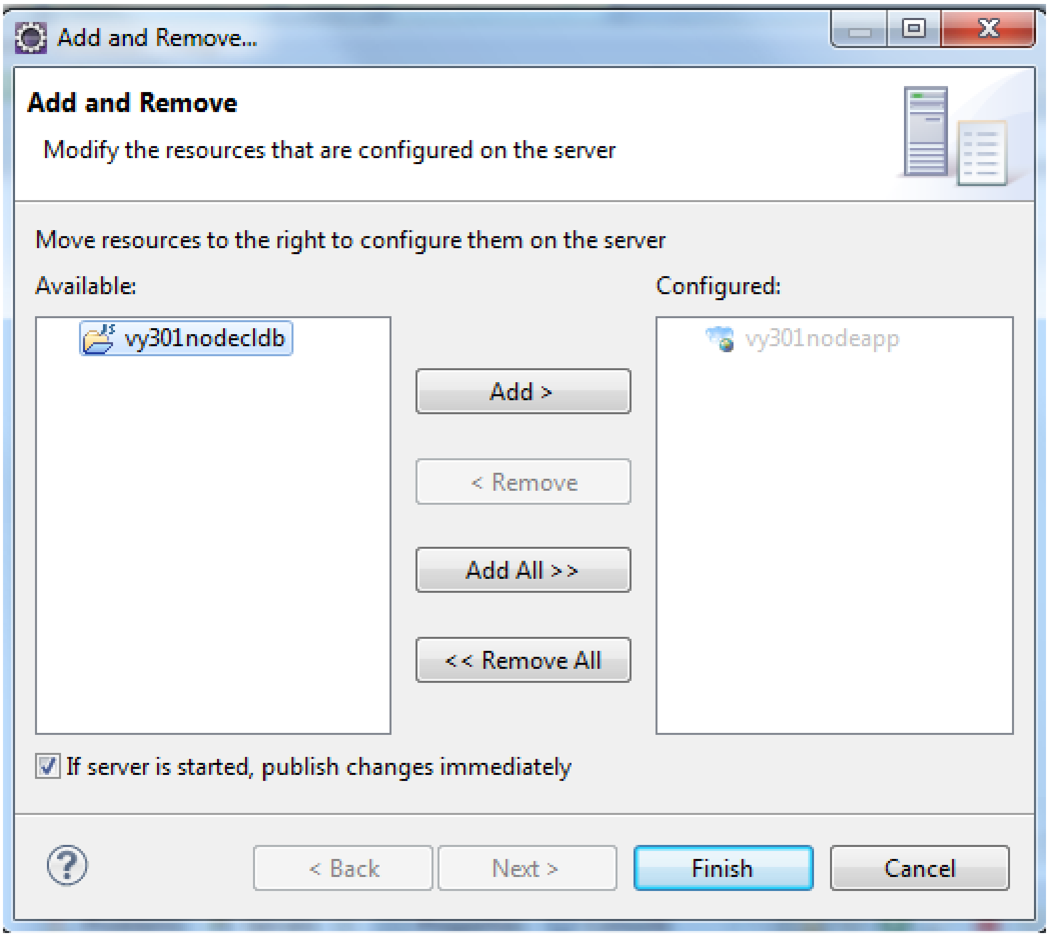
If you see the application with the same name as the project in the list of applications that are running on the Bluemix server, you should delete the project in the servers view, or in the Bluemix cloud. Then, you can re-create it from Eclipse as described in the next steps.

You might need to delete the project in the Bluemix cloud first, if any of the dialog boxes in the next step are not automatically displayed.

__ 2. Deploy the application to Bluemix from the workstation.

__ a. In the Servers view, right-click the IBM Bluemix server. Then, click **Add and Remove**.

__ b. In the Add and Remove dialog box, select the project in the left pane. Then, click **Add**.



Add and Remove...

Add and Remove

Modify the resources that are configured on the server

Move resources to the right to configure them on the server

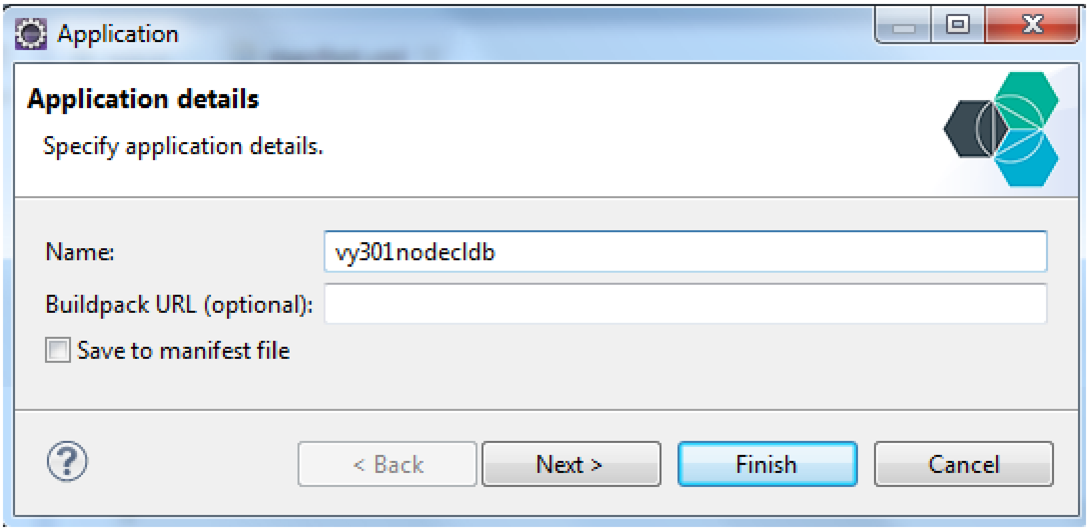
Available: vy301nodecldb

Configured: vy301nodeapp

☑ If server is started, publish changes immediately

__ c. The project is displayed in the right pane. Click **Finish**.

__ d. An Application details dialog box is displayed. Accept the default values, then click **Next**.



Application

Application details

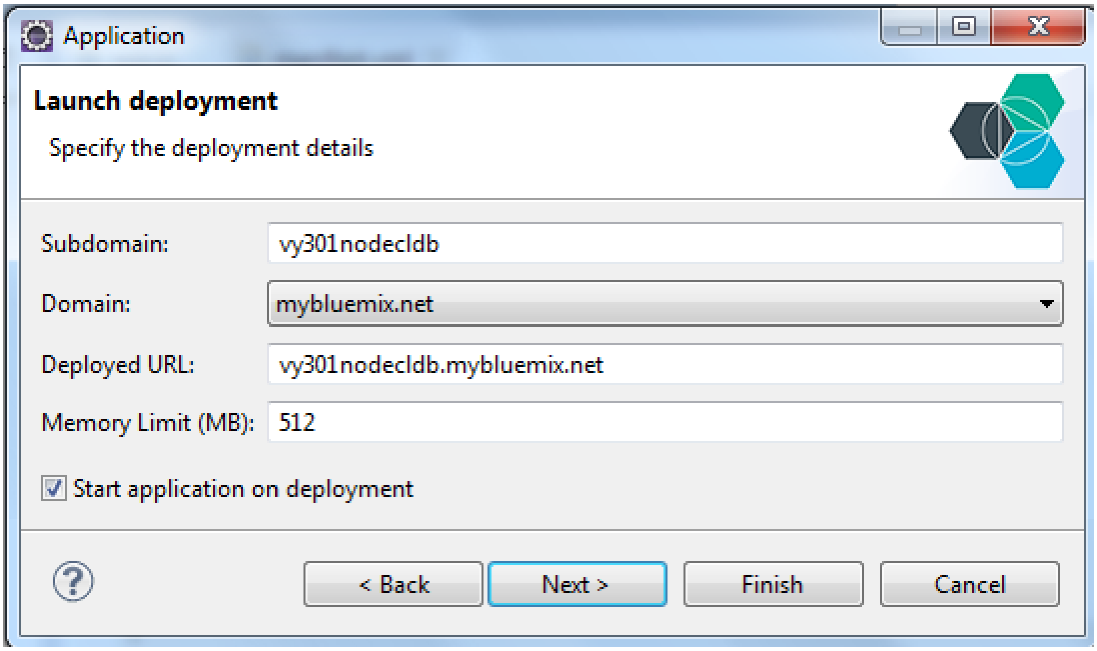
Specify application details.

Name: vy301nodecldb

Buildpack URL (optional):

☐ Save to manifest file

__ e. In the Launch deployment dialog box, accept the default values, then click **Next**.



Application

Launch deployment

Specify the deployment details

Subdomain: vy301nodecldb

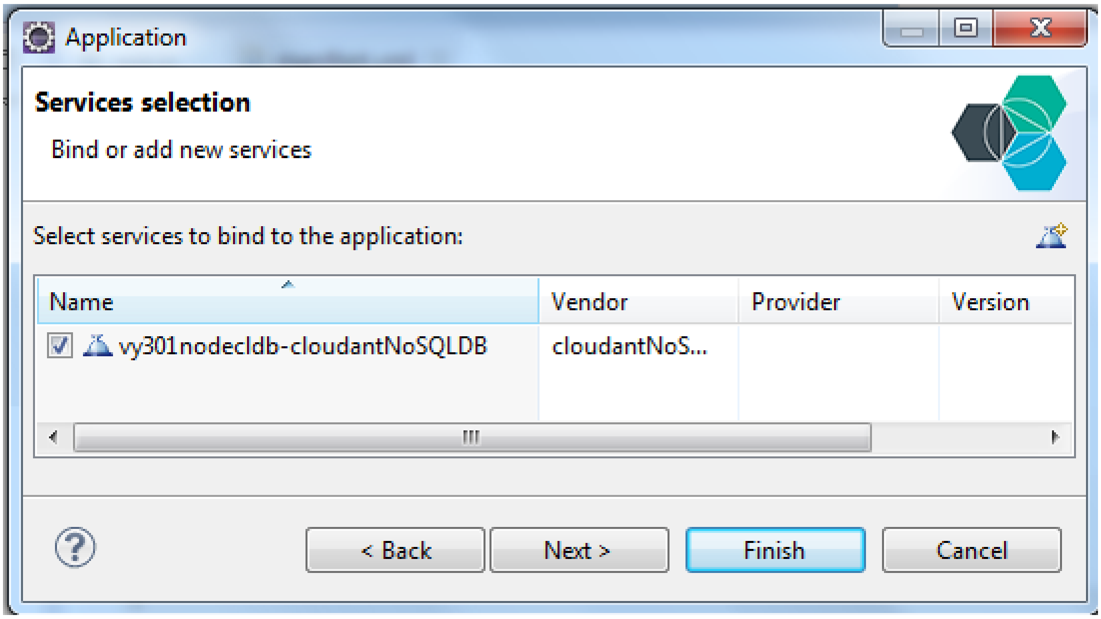
Domain: mybluemix.net

Deployed URL: vy301nodecldb.mybluemix.net

Memory Limit (MB): 512

☑ Start application on deployment

__ f. In the Services selection, select the service to bind to the application and click **Finish**.



Application

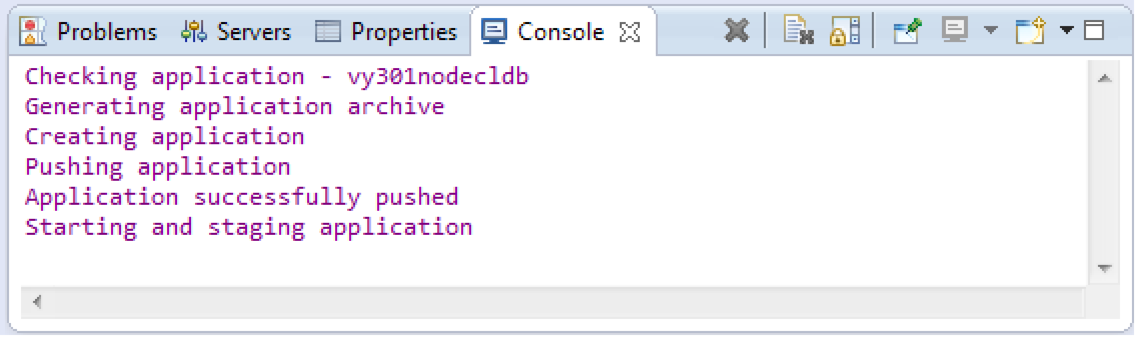
Services selection

Bind or add new services

Select services to bind to the application:

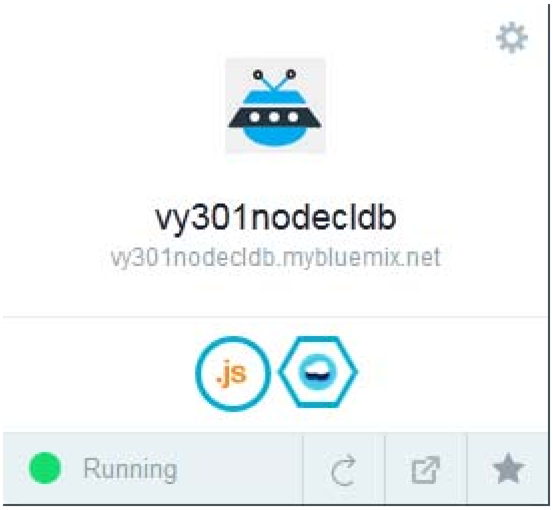
Name	Vendor	Provider	Version
☑ vy301nodecldb-cloudantNoSQLDB	cloudantNoS...		

__ g. You see some activity in the Eclipse console.

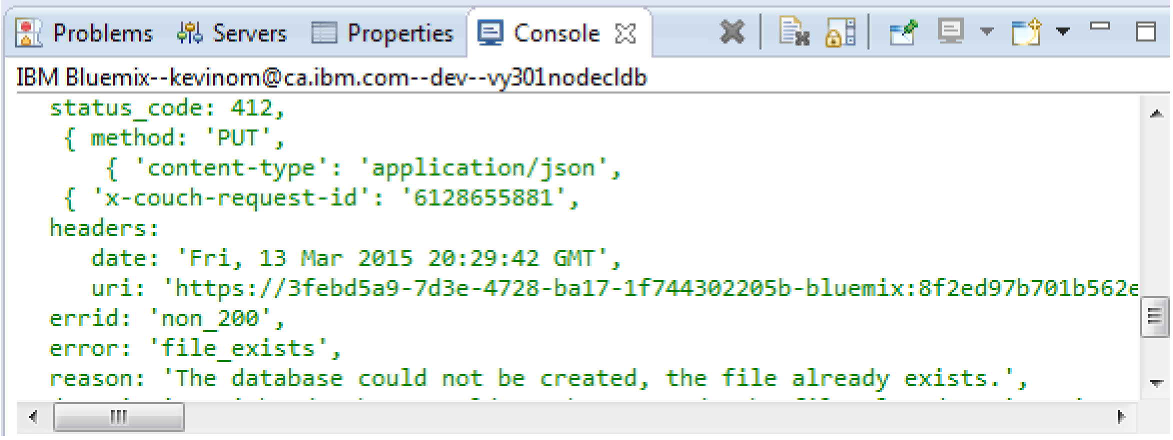


```
Checking application - vy301nodecldb
Generating application archive
Creating application
Pushing application
Application successfully pushed
Starting and staging application
```

__ h. After a short while, you see that the application is running in the Bluemix dashboard.



__ i. In the Eclipse console you see a message that the database might not be created as it exists.

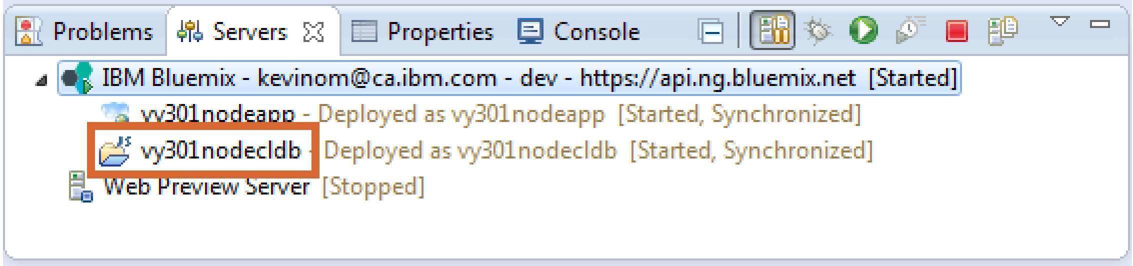


IBM Bluemix--kevinom@ca.ibm.com--dev--vy301nodecldb

```
  status_code: 412,
   { method: 'PUT',
      { 'content-type': 'application/json',
   { 'x-couch-request-id': '6128655881',
  headers:
     date: 'Fri, 13 Mar 2015 20:29:42 GMT',
     uri: 'https://3febd5a9-7d3e-4728-ba17-1f744302205b-bluemix:8f2ed97b701b562e
  errid: 'non_200',
  error: 'file_exists',
  reason: 'The database could not be created, the file already exists.',
```

The error is displayed because the application tried to add the `my_sample_db` that exists.

__ j. You also see that the application is deployed in the Servers view of Eclipse. You can now stop and start the application from the Servers view.



IBM Bluemix - kevinom@ca.ibm.com - dev - https://api.ng.bluemix.net [Started]
vy301nodeapp - Deployed as vy301nodeapp [Started, Synchronized]
vy301nodecldb - Deployed as vy301nodecldb [Started, Synchronized]
Web Preview Server [Stopped]

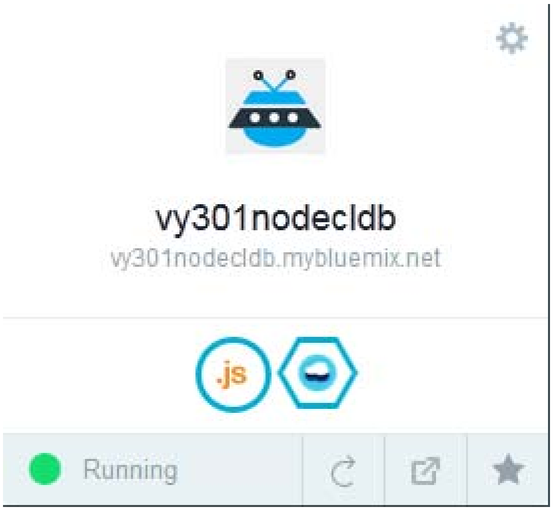
Part 6: Access the Cloudant documentation

In this part, you review the Cloudant documentation.

In particular, you look at the application programming interfaces (APIs) for accessing Cloudant databases and data. Although you are not changing the JavaScript application, reviewing the documentation gives you a better understanding of the code in the sample application.

A number of environment variables are created for you when you added and bound the service to your application.

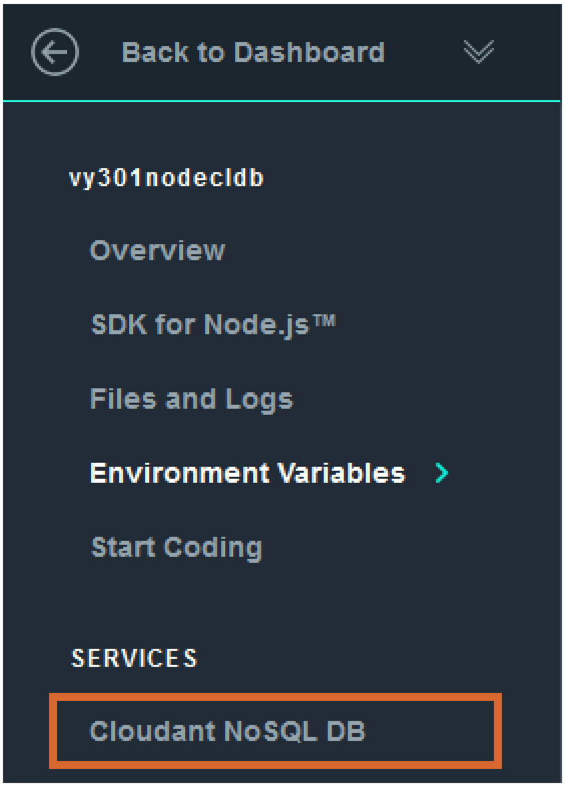
__ 1. From the Bluemix dashboard, click the application to display the details.



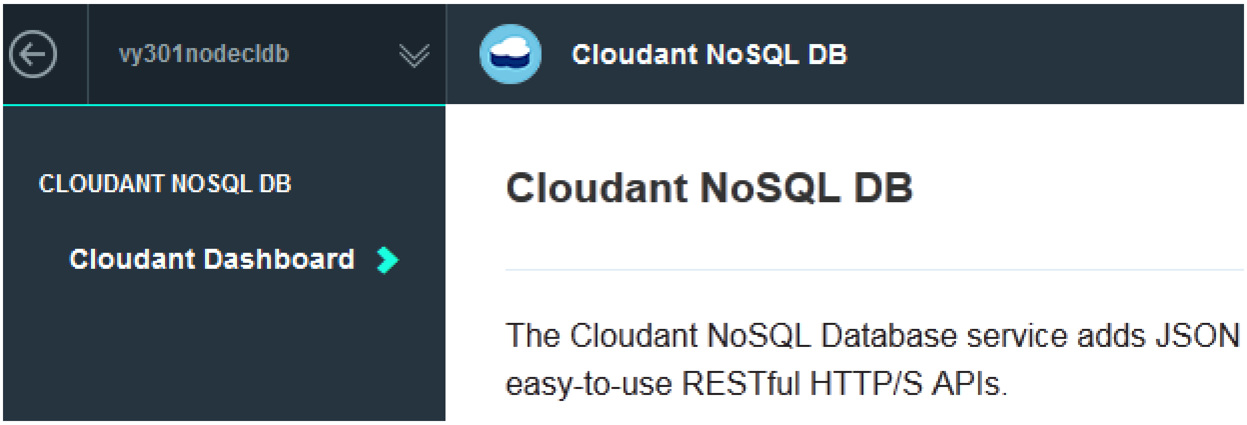
__ 2. If you click **Environment Variables**, you see that these values are the same as you saw earlier.

__ 3. Review Cloudant documentation in Bluemix.

__ a. From the navigation pane, click **Cloudant NoSQL DB**.



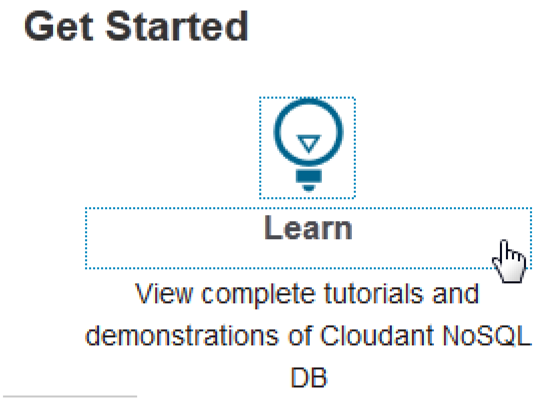
The Cloudant Dashboard is displayed.



Cloudant NoSQL DB

The Cloudant NoSQL Database service adds JSON easy-to-use RESTful HTTP/S APIs.

__ b. Go to the bottom of the first page. Then, click **Learn** under the Getting Started title.



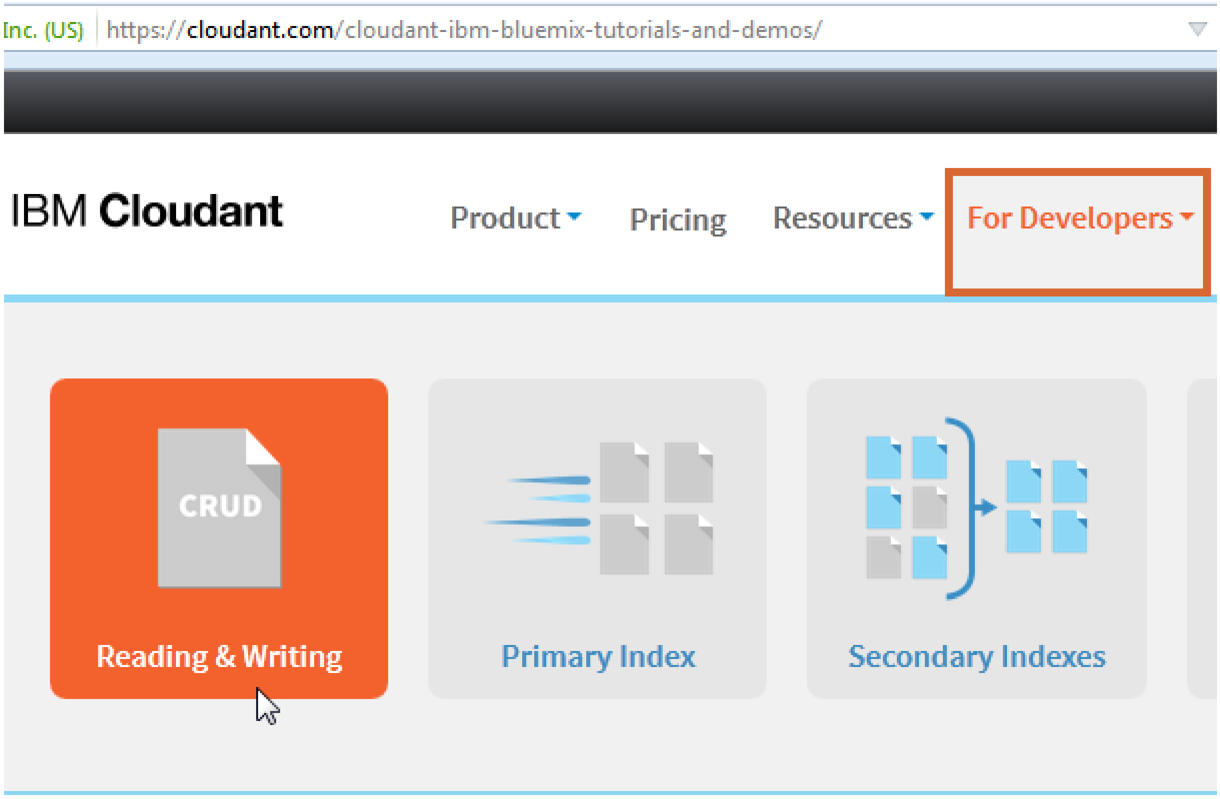
Get Started

Learn

View complete tutorials and demonstrations of Cloudant NoSQL DB

Alternatively, from another browser session, you can go to the address `https://cloudant.com/cloudant-ibm-bluemix-tutorials-and-demos/`

__ c. From the Cloudant tutorials page, click the **For Developers** tab, then click **Reading and Writing**.



Inc. (US) https://cloudant.com/cloudant-ibm-bluemix-tutorials-and-demos/

You can also get to this page by typing `https://cloudant.com/for-developers/crud/` in a browser tab.

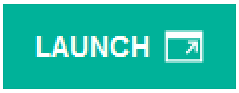
__ d. You can review the documentation for reading and writing to a Cloudant database without signing in.
When your review of the documentation is completed, click the browser tab for the Bluemix Cloudant Dashboard.

Part 7: Explore the features of the Cloudant console

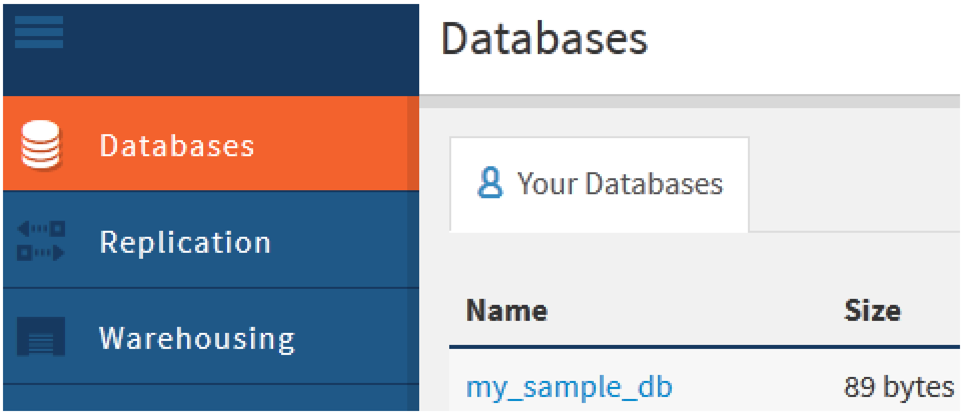
In this part, you explore the Cloudant database from the Cloudant console.

__ 1. Open the Cloudant console.

__ a. From the Cloudant dashboard, click **Launch**.



__ b. The Cloudant console opens in a new browser tab. Notice that the database named `my_sample_db` exists.

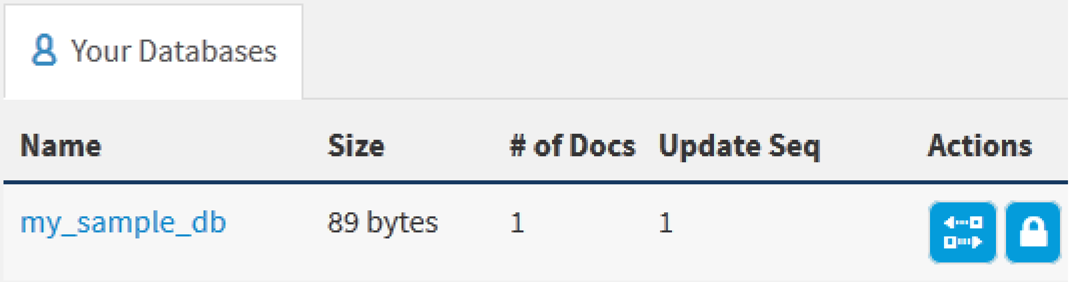


Name	Size
my_sample_db	89 bytes

You are now logged in to the Cloudant console with the default user that was generated when the Node.js and Cloudant was created.

__ 2. View the data in the database.

__ a. Click the **my_sample_db** link. Notice that there is one document in the database.

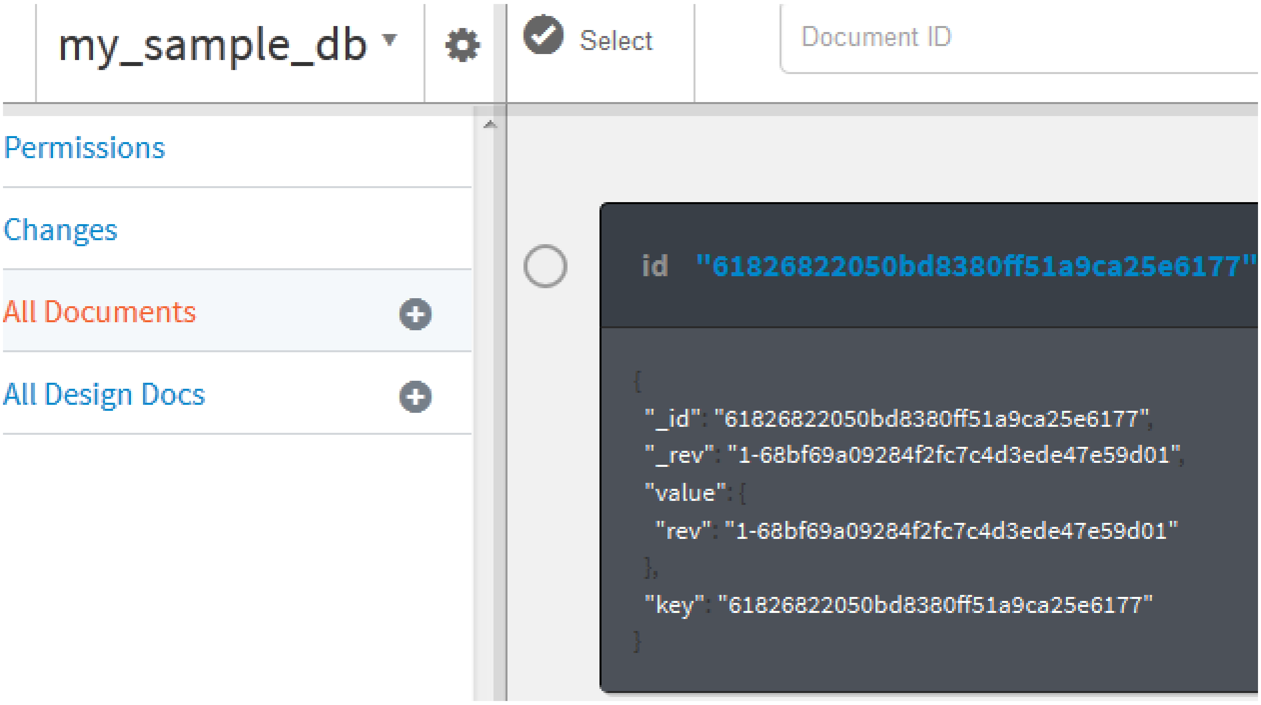


Name	Size	# of Docs	Update Seq	Actions
my_sample_db	89 bytes	1	1	

Note

If there are no documents in the database, logout from IBM Cloudant and relaunch the Cloudant console from the Cloudant dashboard.

The page displays all the documents in the database.

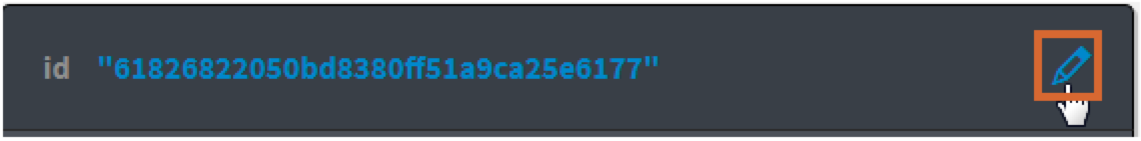


id "61826822050bd8380ff51a9ca25e6177"

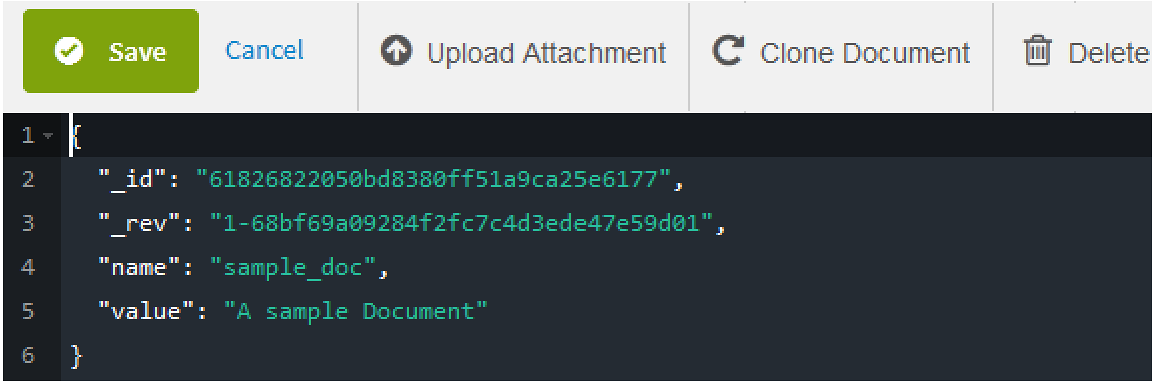
```
{
 "_id": "61826822050bd8380ff51a9ca25e6177",
 "_rev": "1-68bf69a09284f2fc7c4d3ede47e59d01",
 "value": {
  "rev": "1-68bf69a09284f2fc7c4d3ede47e59d01"
 },
 "key": "61826822050bd8380ff51a9ca25e6177"
}
```

The fields “_id” and “_rev” are generated fields. At this stage, all that you are seeing are the generated key and revision values.

__ b. Click the **Edit** icon for the document to open it with the editor.


id "61826822050bd8380ff51a9ca25e6177"

You now see the rest of the JSON name-value pairs for the document.



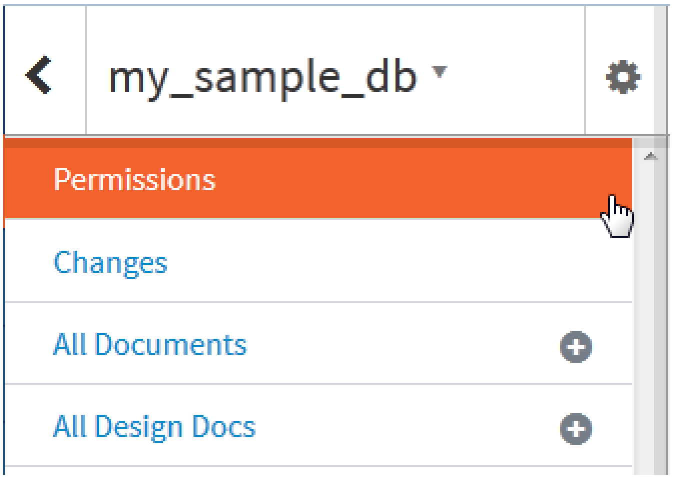
```
{
  "_id": "61826822050bd8380ff51a9ca25e6177",
  "_rev": "1-68bf69a09284f2fc7c4d3ede47e59d01",
  "name": "sample_doc",
  "value": "A sample Document"
}
```

The document editor provides a number of options for modifying the JSON document.

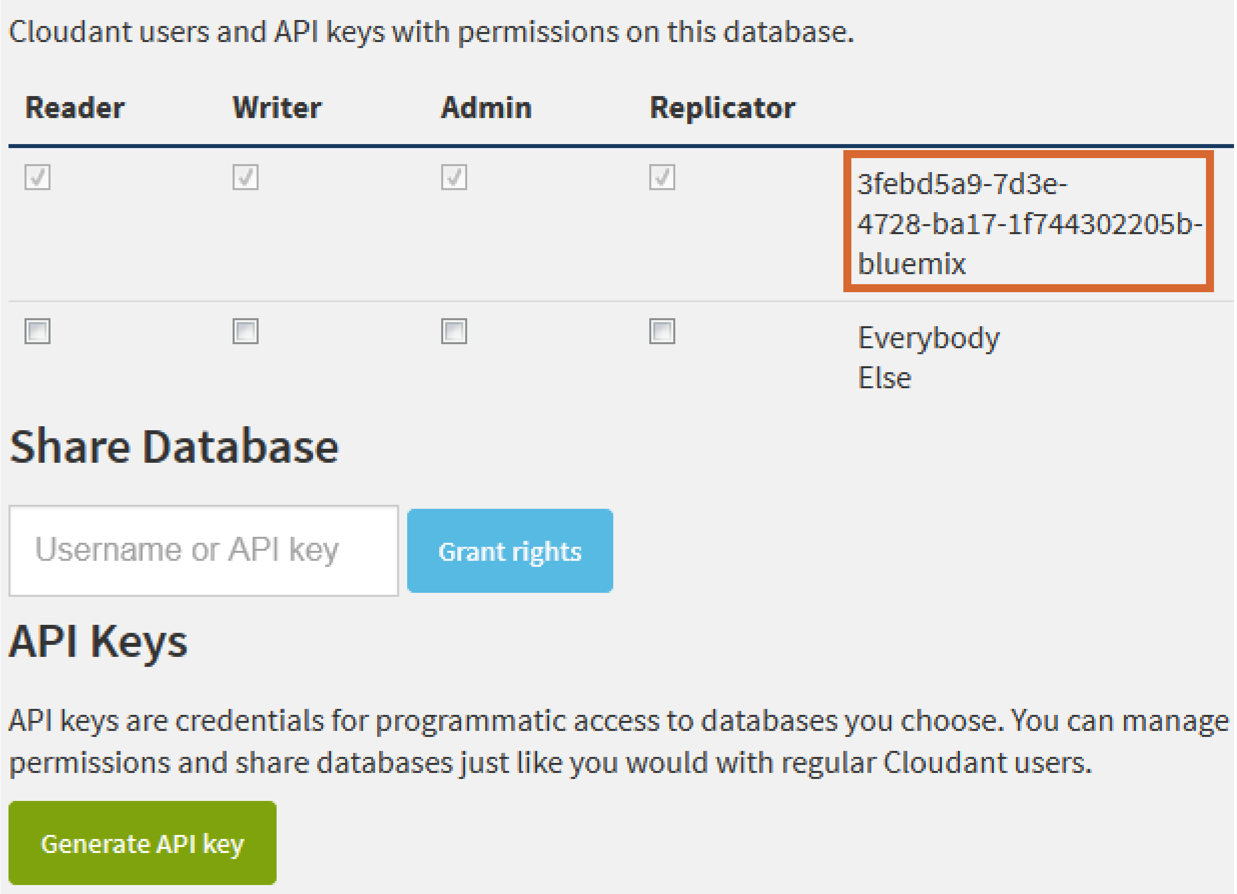
__ c. Click **Cancel** to exit without making any changes.

__ 3. Review the database permissions.

__ a. On the **Databases** tab, with the **my_sample_db** selected, click **Permissions**.



You see the Cloudant users and permissions for the database.



Cloudant users and API keys with permissions on this database.

Reader	Writer	Admin	Replicator	
☑	☑	☑	☑	3febd5a9-7d3e-4728-ba17-1f744302205b-bluemix
□	□	□	□	Everybody Else

Share Database

Username or API key

Grant rights

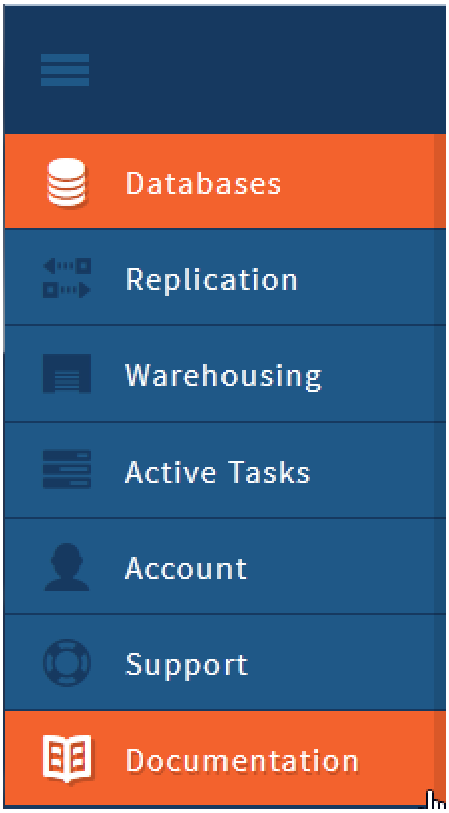
API Keys

API keys are credentials for programmatic access to databases you choose. You can manage permissions and share databases just like you would with regular Cloudant users.

Generate API key

Notice that the generated Cloudant user has full permission to access the database. This user is the same user that is signed in to the Cloudant dashboard and is the value for the username in the VCAP_SERVICES environment variable.

__ b. You are free to explore the other features of the Cloudant console, including the **Documentation** tab.



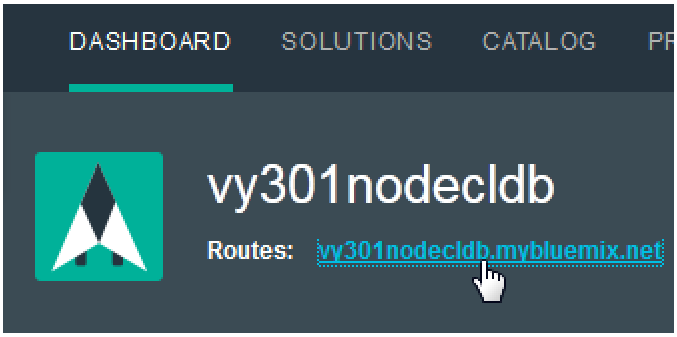
__ c. Click **Logout** when you are finished reviewing the Cloudant console.

Part 8: Run the Bluemix Cloudant application

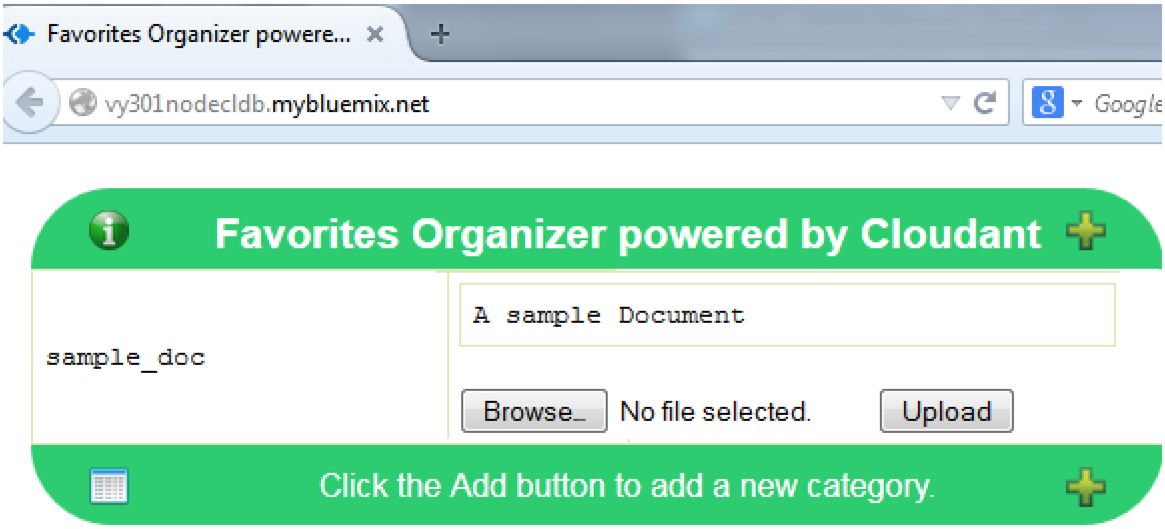
In this part, you run the Bluemix application.

__ 1. Run the application in a browser.

__ a. From the IBM Bluemix dashboard, click the routes link to run the application.



The application is displayed in a new browser tab.



Favorites Organizer powered by Cloudant

sample_doc

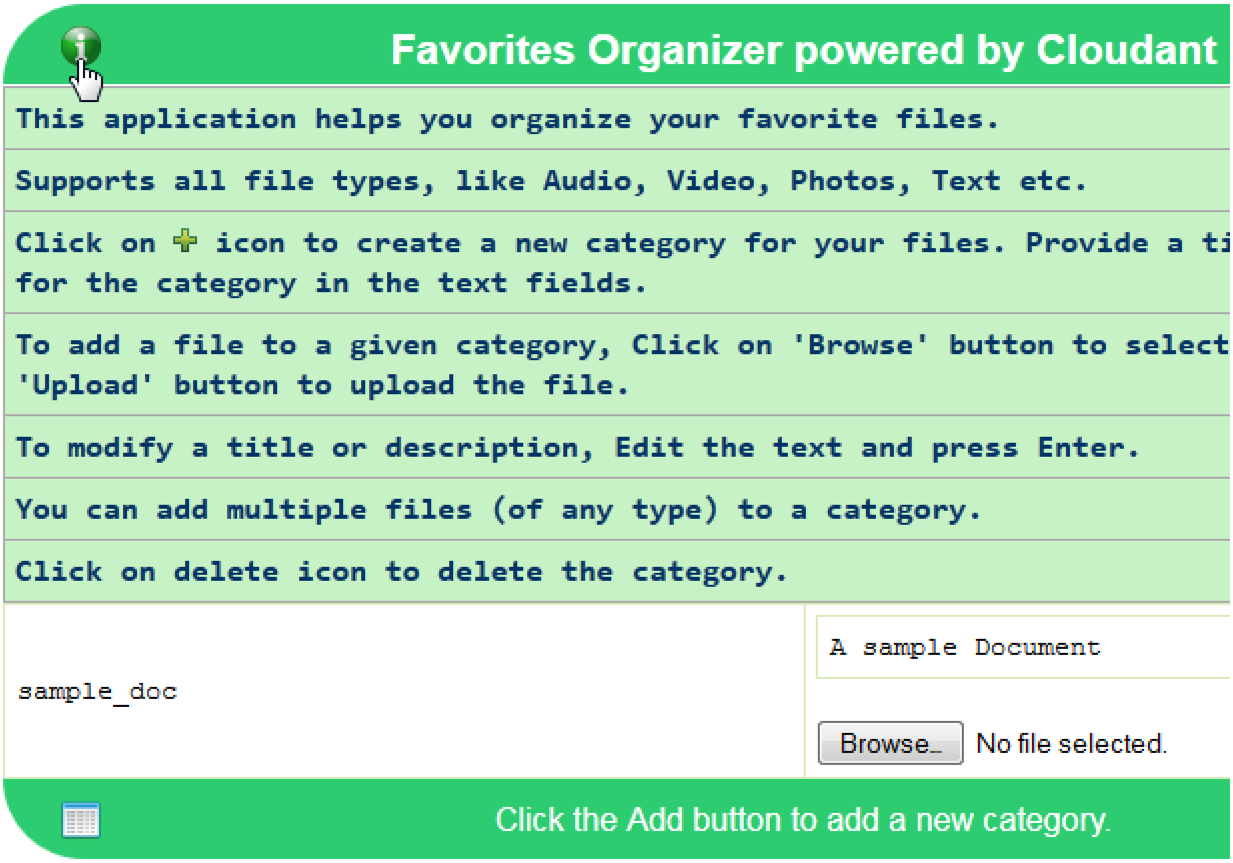
A sample Document

No file selected.

Click the Add button to add a new category.

Notice the sample_doc and the text “A sample Document”. These two text strings are the name-value pair that is stored in the Cloudant database that you saw earlier.

__ b. Click the **information** icon that is found at the upper left of the application.



Favorites Organizer powered by Cloudant

This application helps you organize your favorite files.

Supports all file types, like Audio, Video, Photos, Text etc.

Click on ✚ icon to create a new category for your files. Provide a ti
for the category in the text fields.

To add a file to a given category, Click on 'Browse' button to select
'Upload' button to upload the file.

To modify a title or description, Edit the text and press Enter.

You can add multiple files (of any type) to a category.

Click on delete icon to delete the category.

sample_doc

A sample Document

No file selected.

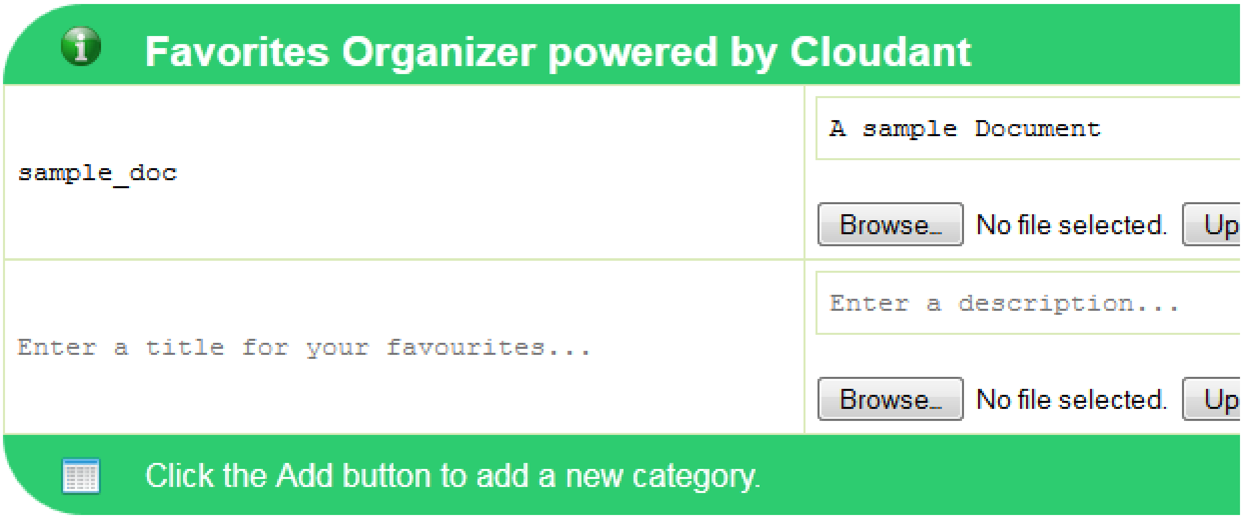
Click the Add button to add a new category.

The text describes how to add, edit, and delete categories to the application and the underlying Cloudant database.

__ 2. Add some data to the Cloudant database.

__ a. Click the '+' icon in the browser window to add a new category.

__ b. In the row that is added, type a title and description for the resource that you are adding.

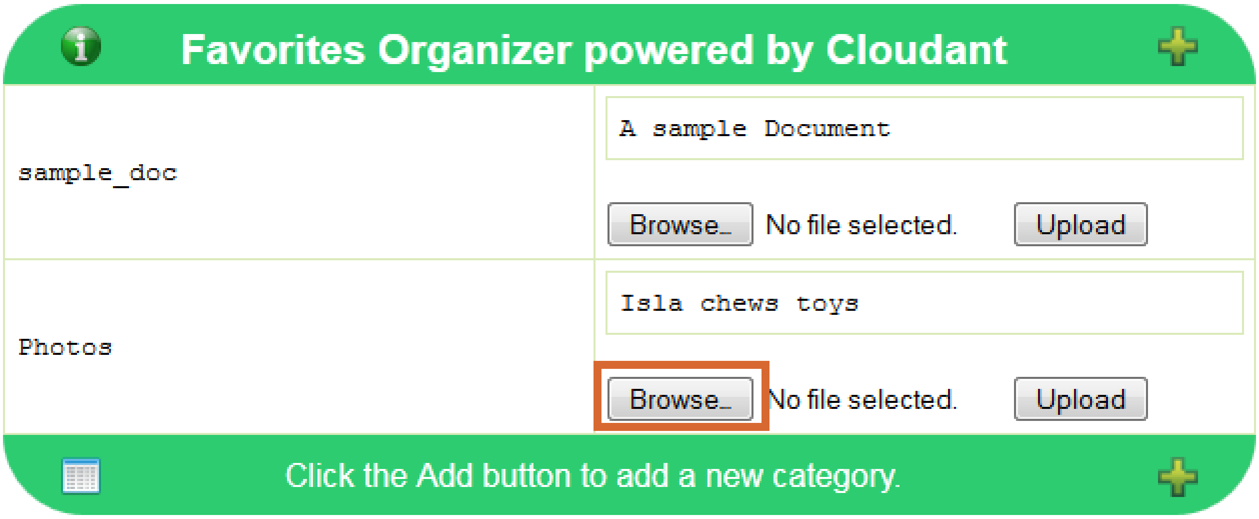


Favorites Organizer powered by Cloudant

sample_doc	A sample Document Browse… No file selected. Up
Enter a title for your favourites...	Enter a description... Browse… No file selected. Up

Click the Add button to add a new category.

__ c. Click **Browse**.

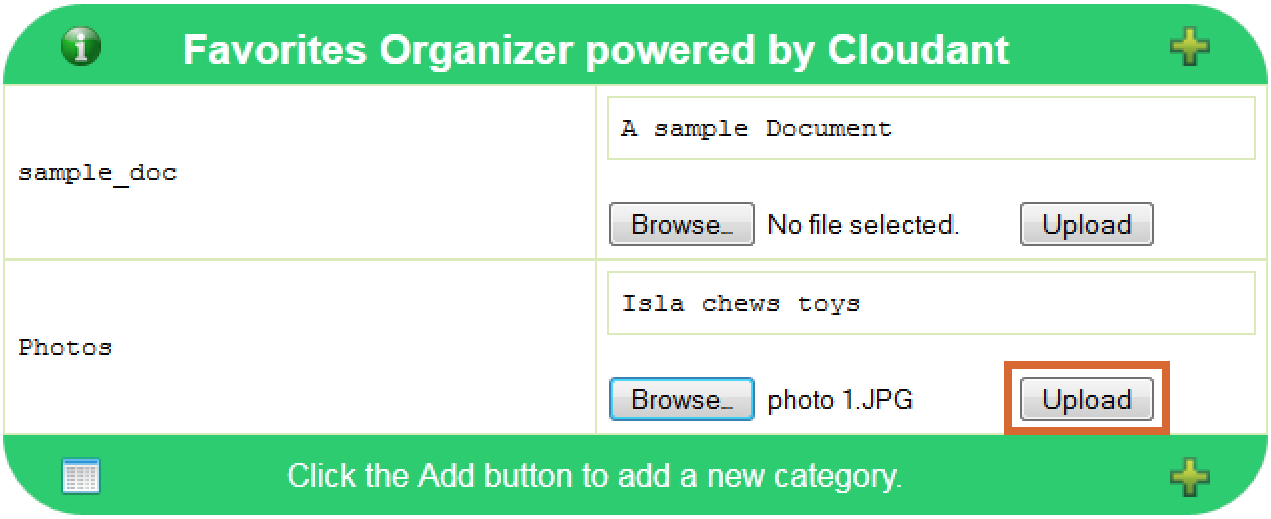


Favorites Organizer powered by Cloudant

sample_doc	A sample Document Browse… No file selected. Upload
Photos	Isla chews toys Browse… No file selected. Upload

Click the Add button to add a new category.

__ d. Go to the file that you want to upload from your workstation.



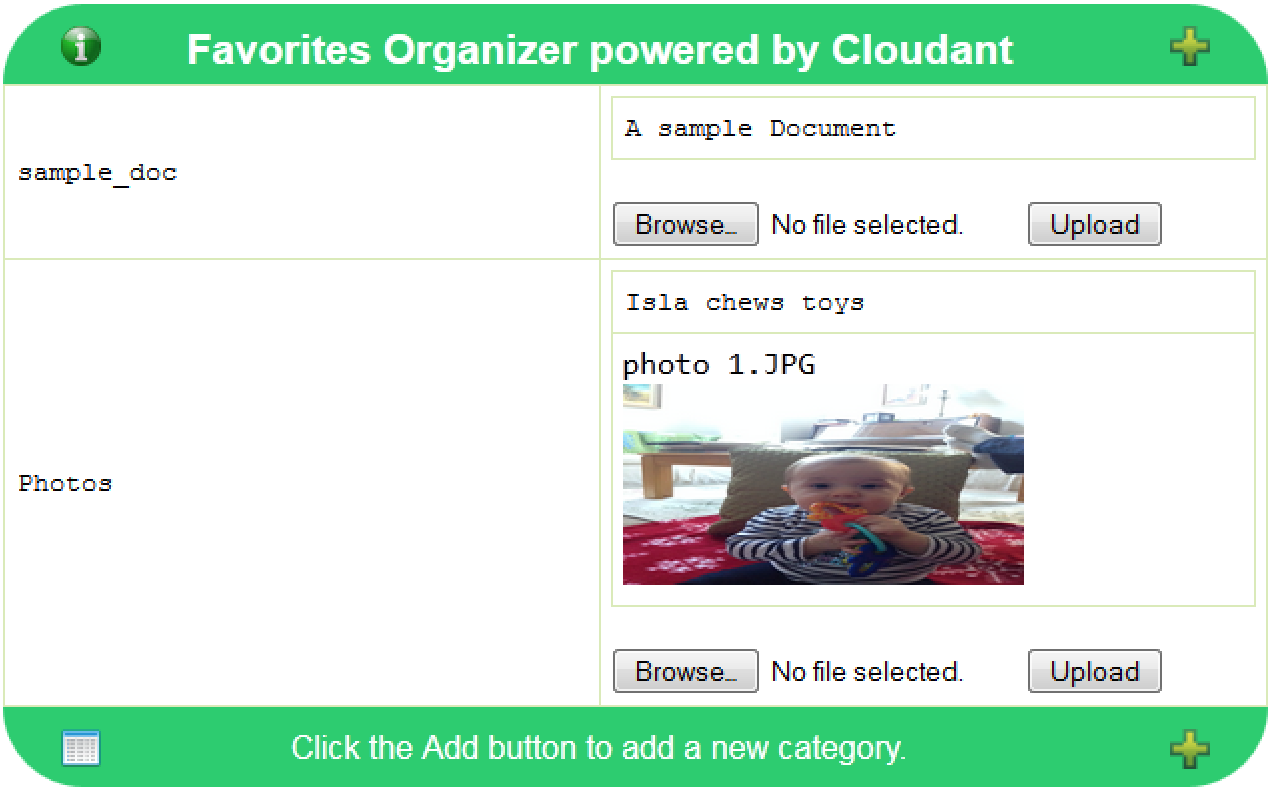
Favorites Organizer powered by Cloudant

sample_doc	A sample Document Browse... No file selected. Upload
Photos	Isla chews toys Browse... photo 1.JPG Upload

Click the Add button to add a new category.

__ e. Click **Upload** to select a file to upload.

The file is uploaded and stored on the Cloudant database.



Favorites Organizer powered by Cloudant

sample_doc	A sample Document Browse... No file selected. Upload
Photos	Isla chews toys photo 1.JPG Browse... No file selected. Upload

Click the Add button to add a new category.

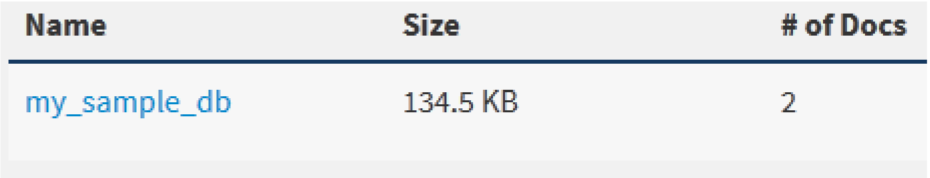
Part 9: Verify the data from the Cloudant console

In this part, you verify that the data is added from the Cloudant dashboard.

__ 1. Explore the contents of the Cloudant database.

__ a. From IBM Bluemix, open the Cloudant console as you did earlier.

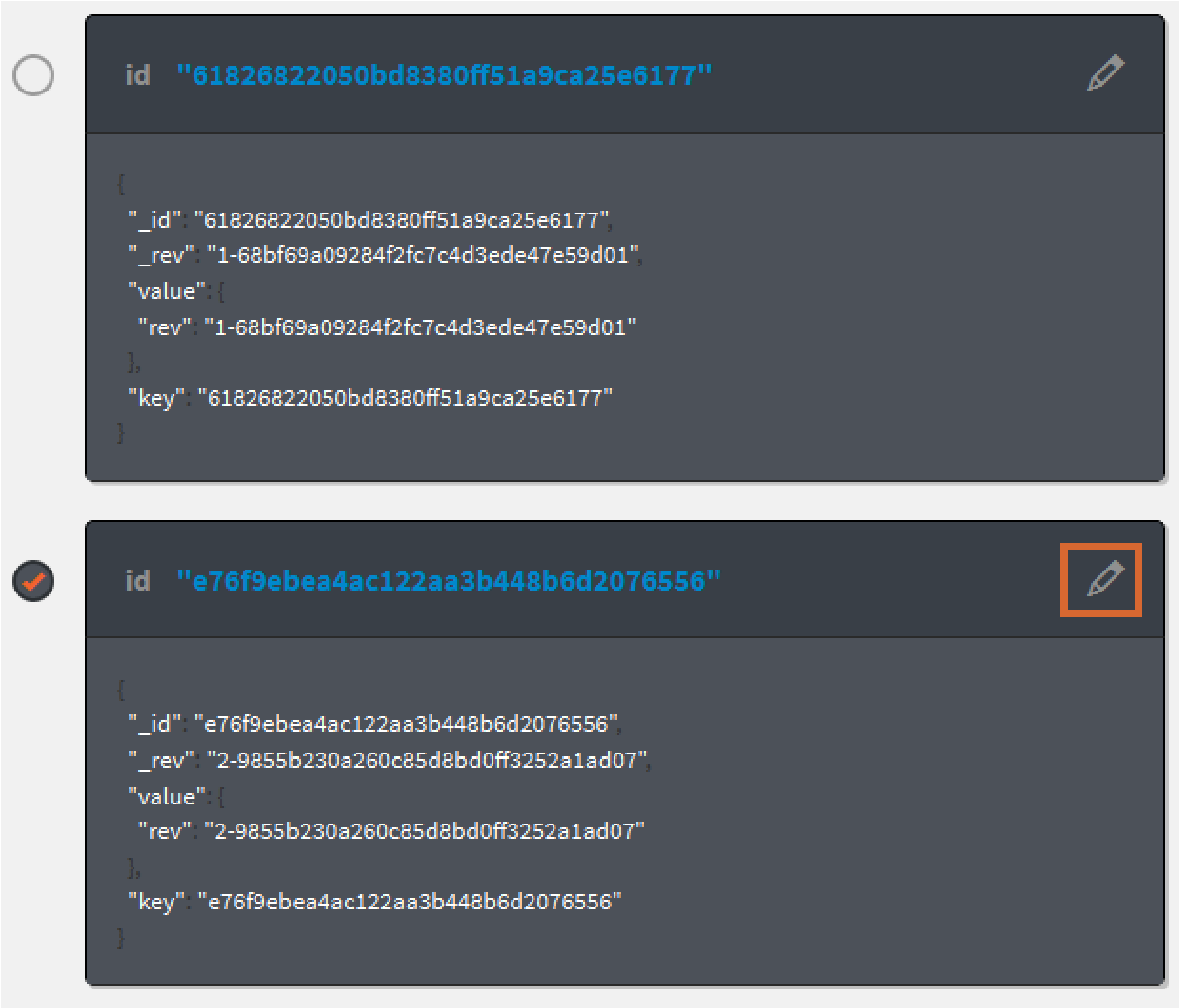
__ b. On the **Databases** tab of the Cloudant console, notice that the number of documents that are stored in the Cloudant database is now **2**.



Name	Size	# of Docs
my_sample_db	134.5 KB	2

__ c. Click the **my_sample_db** link to open the database.

__ d. Select the document that was added last. The latest document might be displayed first in some cases. Then, click the **Edit** icon.



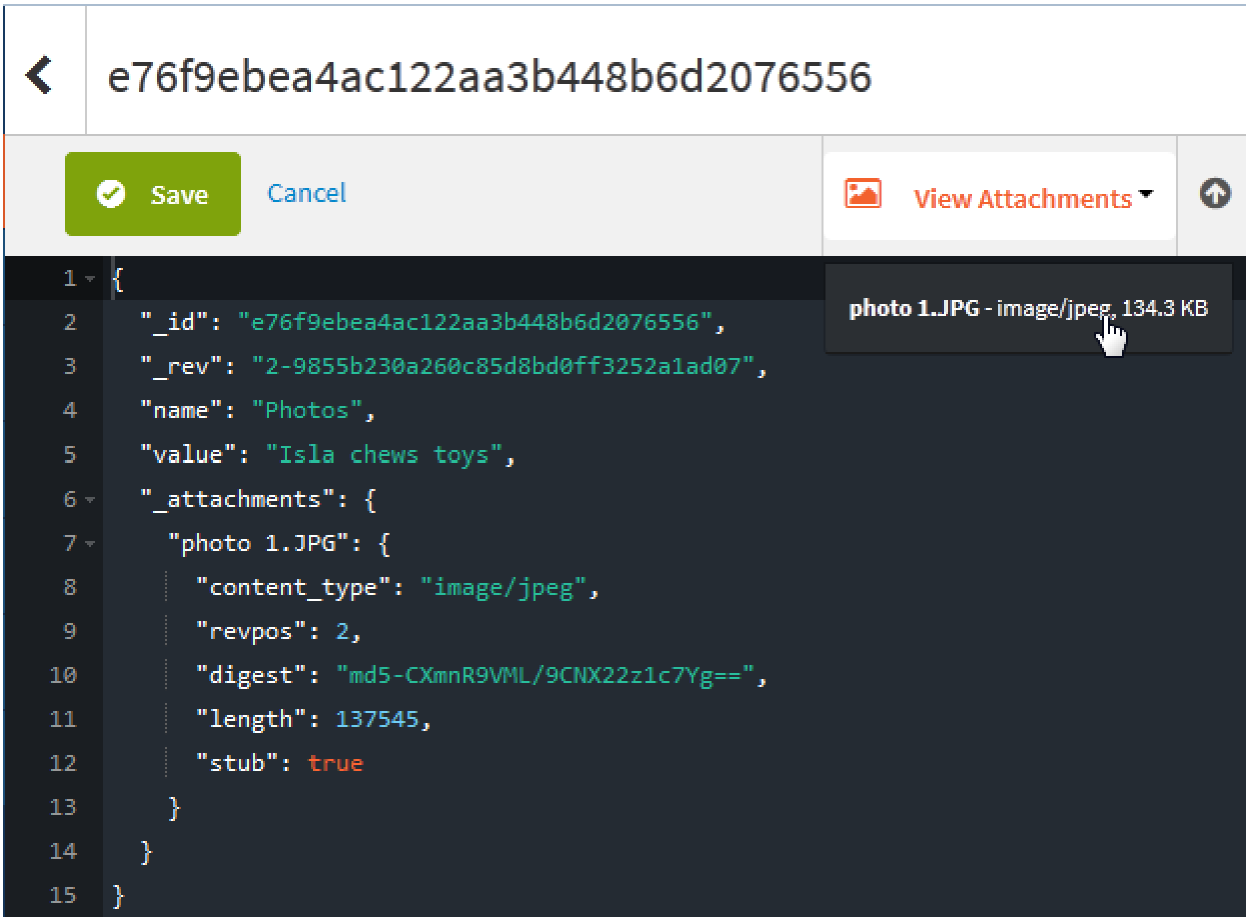
○ id "61826822050bd8380ff51a9ca25e6177"

```
{
 "_id": "61826822050bd8380ff51a9ca25e6177",
 "_rev": "1-68bf69a09284f2fc7c4d3ede47e59d01",
 "value": {
  "rev": "1-68bf69a09284f2fc7c4d3ede47e59d01"
 },
 "key": "61826822050bd8380ff51a9ca25e6177"
}
```

✓ id "e76f9ebea4ac122aa3b448b6d2076556"

```
{
 "_id": "e76f9ebea4ac122aa3b448b6d2076556",
 "_rev": "2-9855b230a260c85d8bd0ff3252a1ad07",
 "value": {
  "rev": "2-9855b230a260c85d8bd0ff3252a1ad07"
 },
 "key": "e76f9ebea4ac122aa3b448b6d2076556"
}
```

The details of the JSON document that is stored on the Cloudant database are displayed.



e76f9ebea4ac122aa3b448b6d2076556
Save
Cancel
View Attachments
photo 1.JPG - image/jpeg, 134.3 KB

```
{
  "_id": "e76f9ebea4ac122aa3b448b6d2076556",
  "_rev": "2-9855b230a260c85d8bd0ff3252a1ad07",
  "name": "Photos",
  "value": "Isla chews toys",
  "_attachments": {
    "photo 1.JPG": {
      "content_type": "image/jpeg",
      "revpos": 2,
      "digest": "md5-CXmnR9VML/9CNX22z1c7Yg==",
      "length": 137545,
      "stub": true
    }
  }
}
```

__ e. Click the **View Attachments** tab and the file name to display the contents of the attachment.

End of exercise

Exercise review and wrap-up

In this exercise, you worked with the Node.js Cloudant web starter application from the Bluemix boilerplates to explore how a Node.js JavaScript application can access a Cloudant database.

Exercise 5. Building a mobile data Bluemix application

What this exercise is about

In this exercise, you work with a Bluemix mobile data project.

You create the Bluemix Mobile Data cloud component from a Blueprint boilerplate that includes a number of services.

You clone a sample client application from DevOps that is implemented as an Android application.

You download the Bluemix Android SDK Java archive files and import the files into Eclipse so that the mobile application can integrate with the Mobile Data application that runs on Bluemix.

Finally, you test the client web application in an Android emulator that connects to the Mobile Data application that runs on the Bluemix cloud.

What you should be able to do

After completing this exercise, you should be able to:

- Sign in to Bluemix
- Create a mobile data project in Bluemix
- Set up the Android development environment in Eclipse
- Clone the mobile application from IBM Bluemix DevOps
- Configure the mobile BlueList application in Eclipse
- Review the Android Java code
- Install the Android virtual device
- Run the BlueList Android application
- Review the data on the Bluemix cloud

Introduction

This exercise is based on the IBM DeveloperWorks article titled “Building an Android app using the IBM Mobile Data for Bluemix cloud service”. The article is found at the address:

`http://www.ibm.com/developerworks/library/mo-android-mobiledata-app/index.html`

Requirements

You must have an IBM Bluemix ID and must complete the previous exercises before doing this exercise.

In these earlier exercises, you installed the Cloud Foundry CLI on your local workstation. You also installed the Eclipse development environment and the Bluemix plug-in for Eclipse.

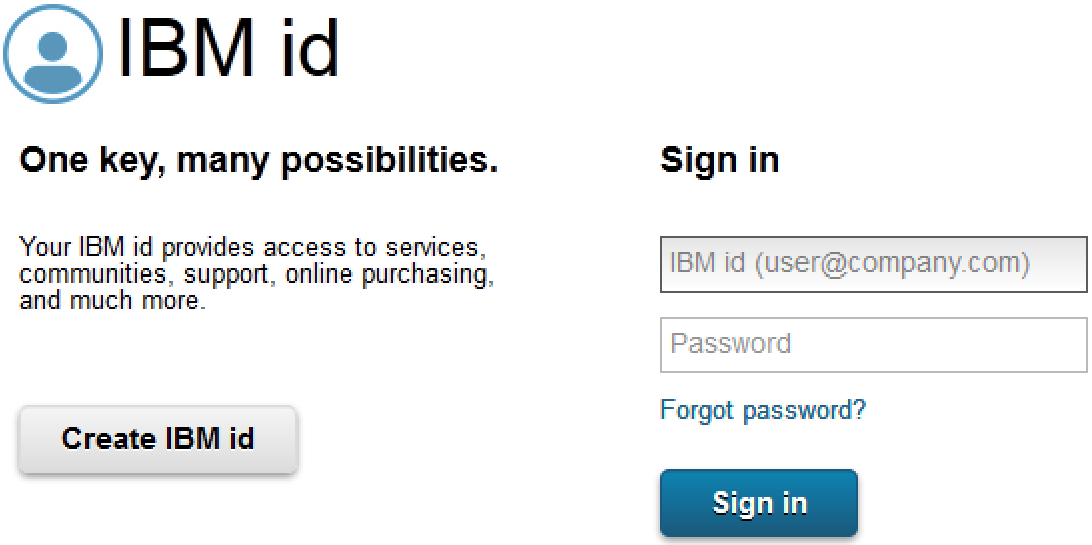
Exercise instructions

Preface

This exercise depends on the availability of the IBM Bluemix cloud and connectivity from your own workstation to the cloud environment.

Part 1: Sign on to the Bluemix cloud

From your web browser, type the address `bluemix.net`. Then, sign on to IBM Bluemix with your IBM ID and password.



IBM id

One key, many possibilities.

Your IBM id provides access to services, communities, support, online purchasing, and much more.

When you are signed on, you see the applications that you created earlier in the Bluemix catalog.

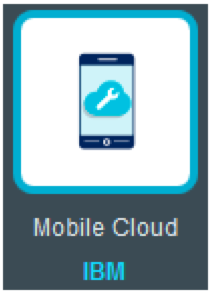
Part 2: Create a Mobile Data application on Bluemix

In this part, you create the Mobile Data Cloud application from the Bluemix services boilerplate.

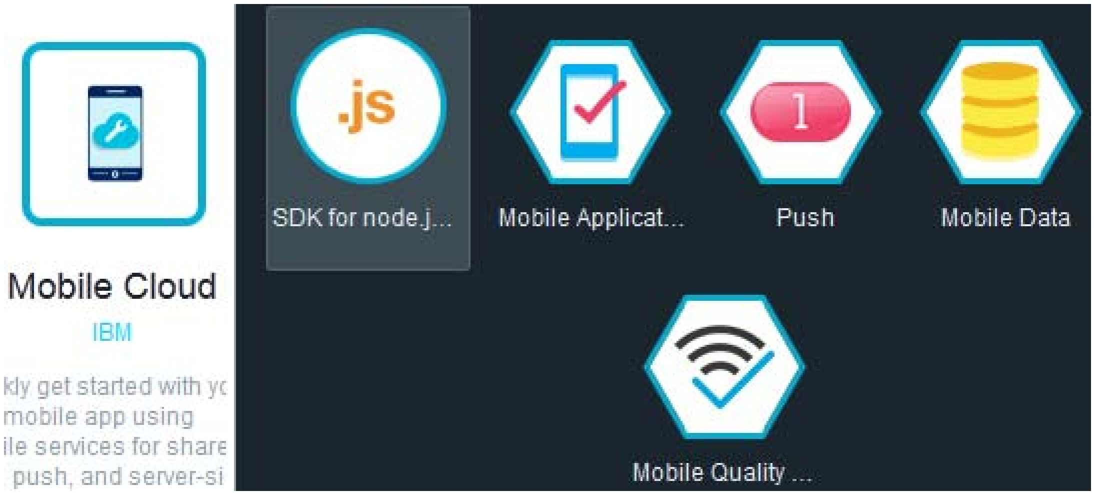
__ 1. Add Mobile Cloud to your Bluemix environment.

__ a. From the Bluemix main page, click the **Catalog** tab.

__ b. From the Boilerplates section of the Catalog, click **Mobile Cloud**.



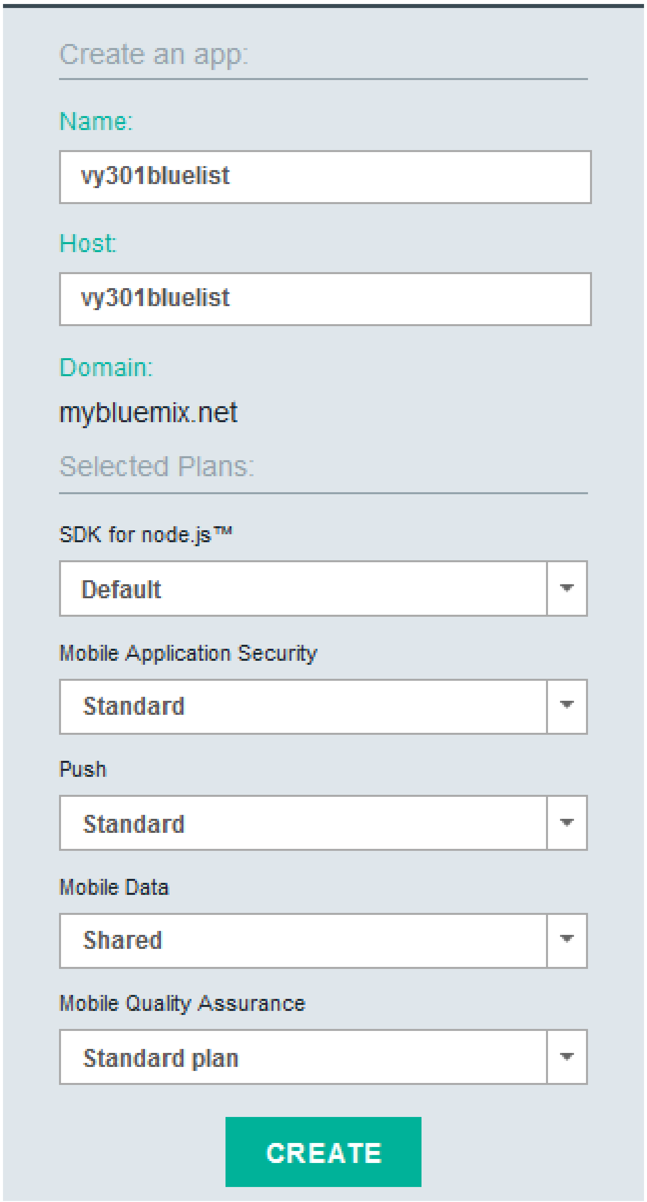
The Mobile Cloud boilerplate includes a Node.js run time and a number of services.



__ c. In the Create an App area for the mobile cloud application, type or select the following, with the unique naming convention that you adopted in the earlier exercises and click **Create**:

- Name: *`Your application name`*
- Host: *`same as above`*
- Domain: `mybluemix.net`
- SDK for Node.js: `Default`
- Mobile Application Security: `Default`
- Push: `Standard`
- Mobile Data: `Shared`

- Mobile Quality Assurance: `Standard plan`



Create an app:

Name: vy301bluelist

Host: vy301bluelist

Domain: mybluemix.net

Selected Plans:

SDK for node.js™: Default

Mobile Application Security: Standard

Push: Standard

Mobile Data: Shared

Mobile Quality Assurance: Standard plan

The Mobile Data application and its four services are created.

The application runs on the IBM Bluemix cloud. Later, you create an Android mobile application on your workstation by cloning the code from IBM DevOps services.

Part 3: Download the Bluemix Android SDK to the workstation

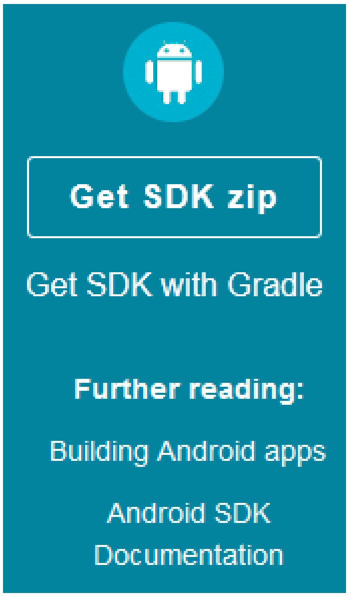
In this part, you download the Android SDK from the Bluemix Mobile Data application.

__ 1. Open the instructions for downloading the Mobile Cloud Services Android SDK for IBM Bluemix.

__ a. From a browser tab, type `https://hub.jazz.net/project/bluemixmobilesdk/ibmbluemix-android/overview`

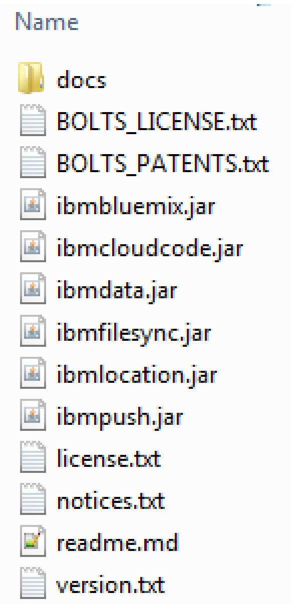
__ b. Follow the instructions in the **readme.md** file for downloading the Mobile Cloud Services Android SDK for IBM Bluemix.

__ c. The compressed file for the SDK can be downloaded from `https://mbaas-catalog.ng.bluemix.net/sdk/ibm-bluemix-sdk-android.zip` or search the IBM Bluemix documentation at `https://www.ng.bluemix.net/docs/#starters/mobile/index.html` for the Mobile Cloud SDK for Android.



__ d. Save the file to a folder on the workstation.

__ e. Extract the compressed file to a folder on the workstation.



Name

docs
BOLTS_LICENSE.txt
BOLTS_PATENTS.txt
ibmbluemix.jar
ibmcloudcode.jar
ibmdata.jar
ibmfilesync.jar
ibmlocation.jar
ibmpush.jar
license.txt
notices.txt
readme.md
version.txt

You see a number of Java Archive files and text files.

__ f. Later, you import these Java Archive files into your project in Eclipse.

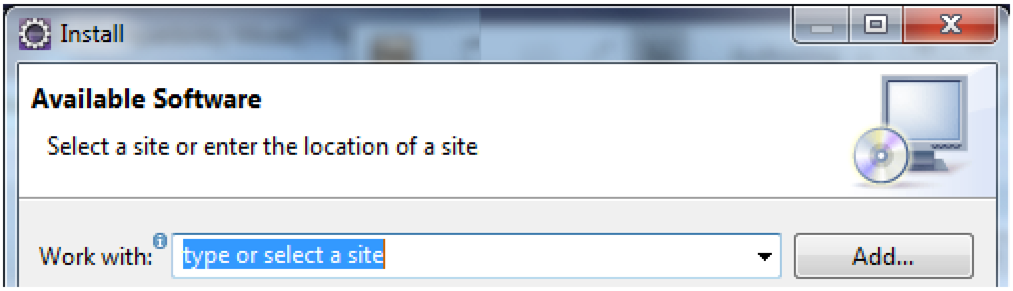
Part 4: Set up the Android development environment

This exercise uses the Eclipse Android Developer Tools (ADT) as the developer environment. You can use the Android Developer Studio instead of the Eclipse ADT. However, instructions are only provided for getting the Android mobile client to work on the Eclipse ADT.

__ 1. Install the Eclipse ADT plug-in.

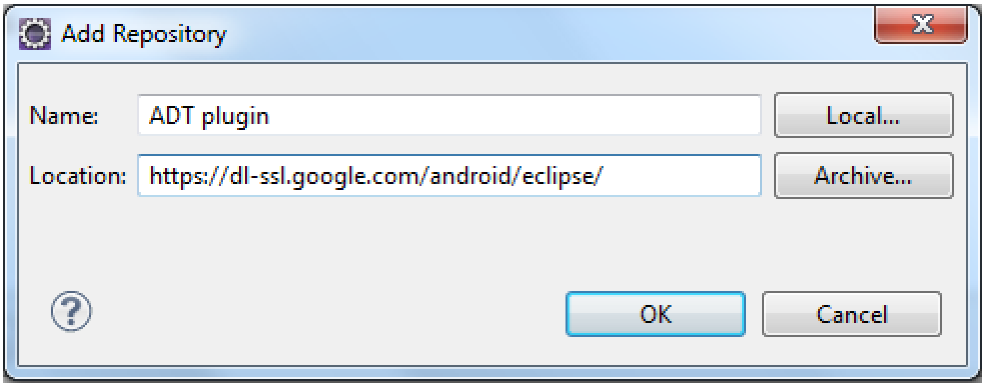
__ a. From the Eclipse menu, click **Help > Install New Software**.

__ b. In the Install dialog box, click **Add**.



__ c. In the Add Repository dialog box, type or select the following input values and click **OK**:

- Name: `ADT plug-in`
- Location: `https://dl-ssl.google.com/android/eclipse/`

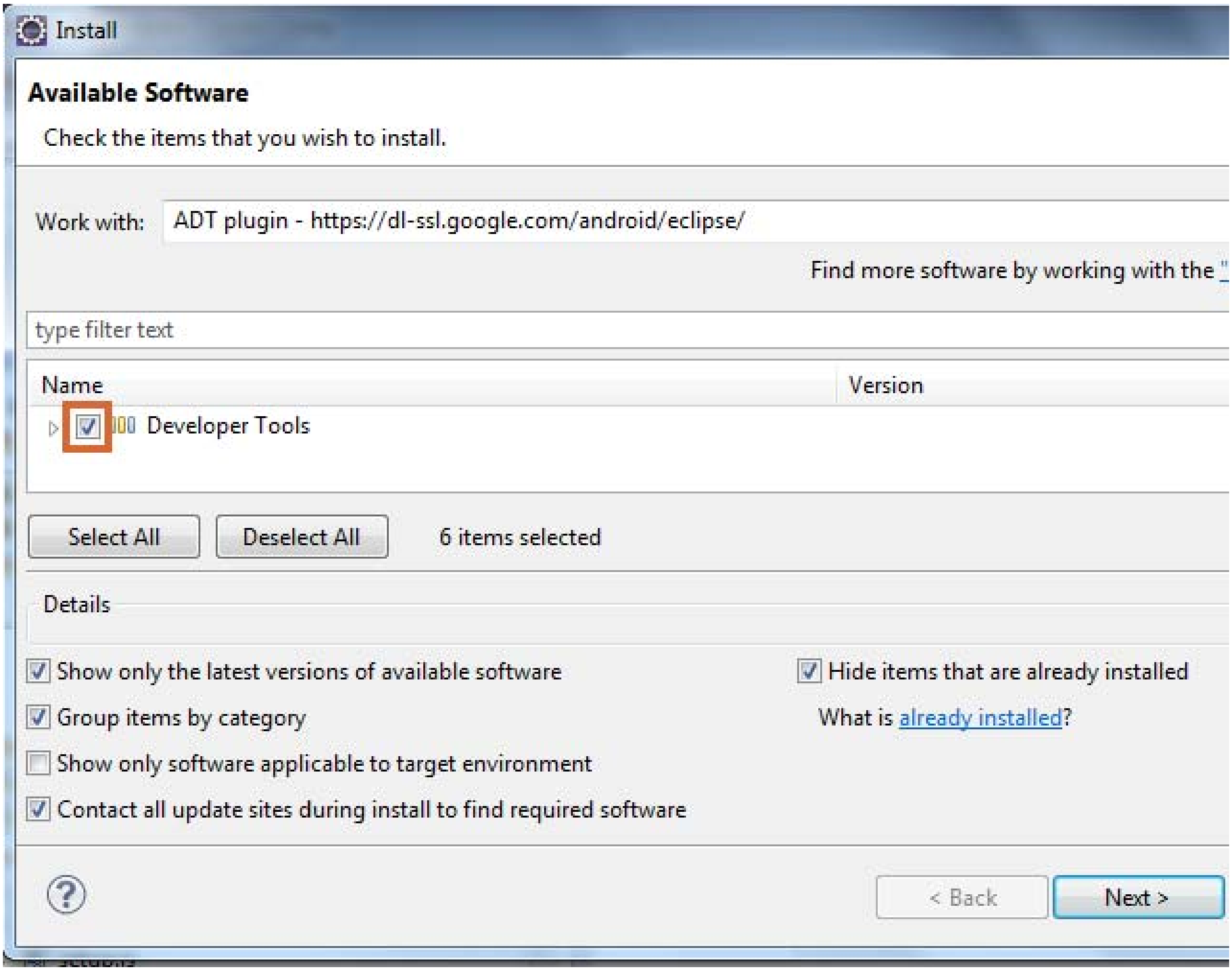


Add Repository

Name: ADT plugin

Location: https://dl-ssl.google.com/android/eclipse/

__ d. Select **Developer Tools** on the page.



Available Software

Check the items that you wish to install.

Work with: ADT plugin - https://dl-ssl.google.com/android/eclipse/

Find more software by working with the "

type filter text

Name	Version
☑ Developer Tools	

6 items selected

Details

☑ Show only the latest versions of available software

☑ Group items by category

☐ Show only software applicable to target environment

☑ Contact all update sites during install to find required software

☑ Hide items that are already installed

What is already installed?

__ e. Click **Next**.

__ f. Accept the defaults on the next page, then click **Finish**.

__ g. Click **OK** in the Security Warning dialog box that warns that you are installing software that contains unsigned contents.

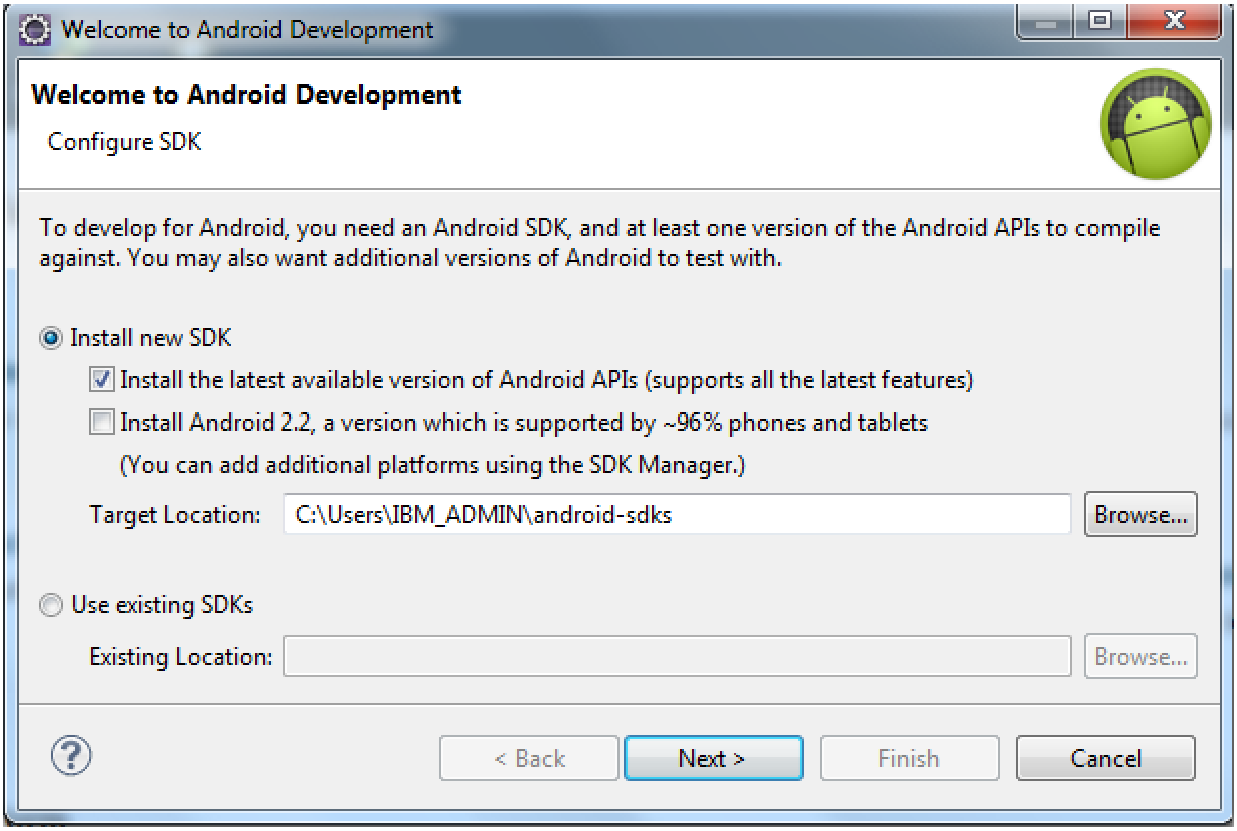
The Android Developer Tools plug-in is installed.

__ h. Restart Eclipse when the restart Eclipse prompt is displayed.

__ 2. Install the Android SDK.

__ a. After Eclipse is restarted, you are prompted to configure the Android SDK. Select **Install new SDK**.

__ b. Select **Install the latest available version of Android APIs**.



Welcome to Android Development

Welcome to Android Development

Configure SDK

To develop for Android, you need an Android SDK, and at least one version of the Android APIs to compile against. You may also want additional versions of Android to test with.

● Install new SDK

☑ Install the latest available version of Android APIs (supports all the latest features)

□ Install Android 2.2, a version which is supported by ~96% phones and tablets

(You can add additional platforms using the SDK Manager.)

Target Location: C:\Users\IBM_ADMIN\android-sdks

○ Use existing SDKs

Existing Location:

__ c. Click **Next**.

__ d. Accept the license.

__ e. Click **Install**.

The Android SDK is installed to the target location.

__ 3. Install the Android APIs that the BlueList application requires.

__ a. In Windows Explorer, go to the target location where the Android SDK is installed. In the case of the example that is seen earlier, the location is at `C:\Users\IBM_ADMIN\android-sdks`.

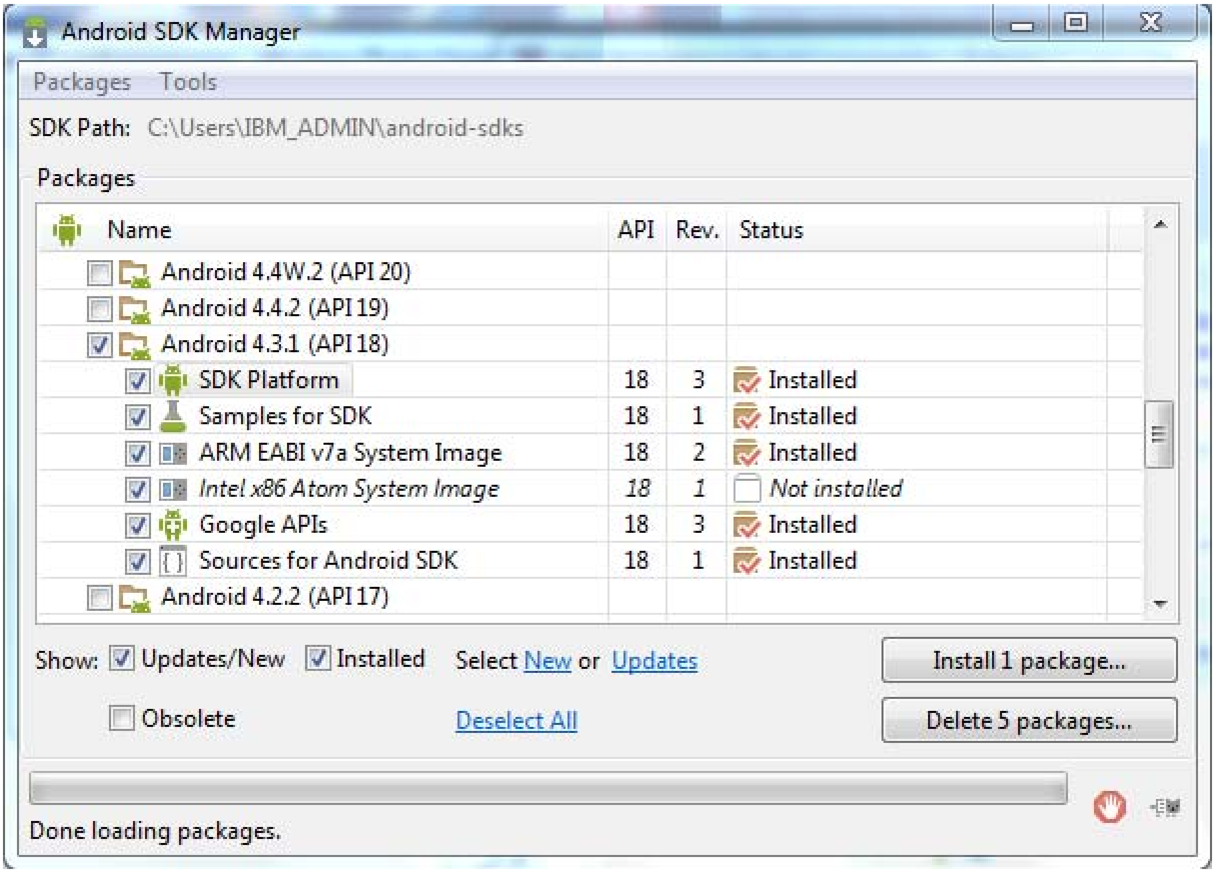
__ b. From the tools subdirectory, double-click **SDK Manager.exe** to run the SDK Manager.

Note

If you are running Eclipse Luna, you can run the command from the menu. Click **Window > Android SDK Manager**.

__ c. From the Android SDK Manager, select all the packages for **Android 4.3.1 (API 18)**.

__ d. Install the packages and any updates that the Android SDK Manager selects.



SDK Path: C:\Users\IBM_ADMIN\android-sdks

Packages

	Name	API	Rev.	Status
☐	Android 4.4W.2 (API 20)			
☐	Android 4.4.2 (API 19)			
☑	Android 4.3.1 (API 18)			
☑	SDK Platform	18	3	Installed
☑	Samples for SDK	18	1	Installed
☑	ARM EABI v7a System Image	18	2	Installed
☑	*Intel x86 Atom System Image*	*18*	*1*	*Not installed*
☑	Google APIs	18	3	Installed
☑	Sources for Android SDK	18	1	Installed
☐	Android 4.2.2 (API 17)			

Show: ☑ Updates/New ☑ Installed Select New or Updates

☐ Obsolete Deselect All

Done loading packages.

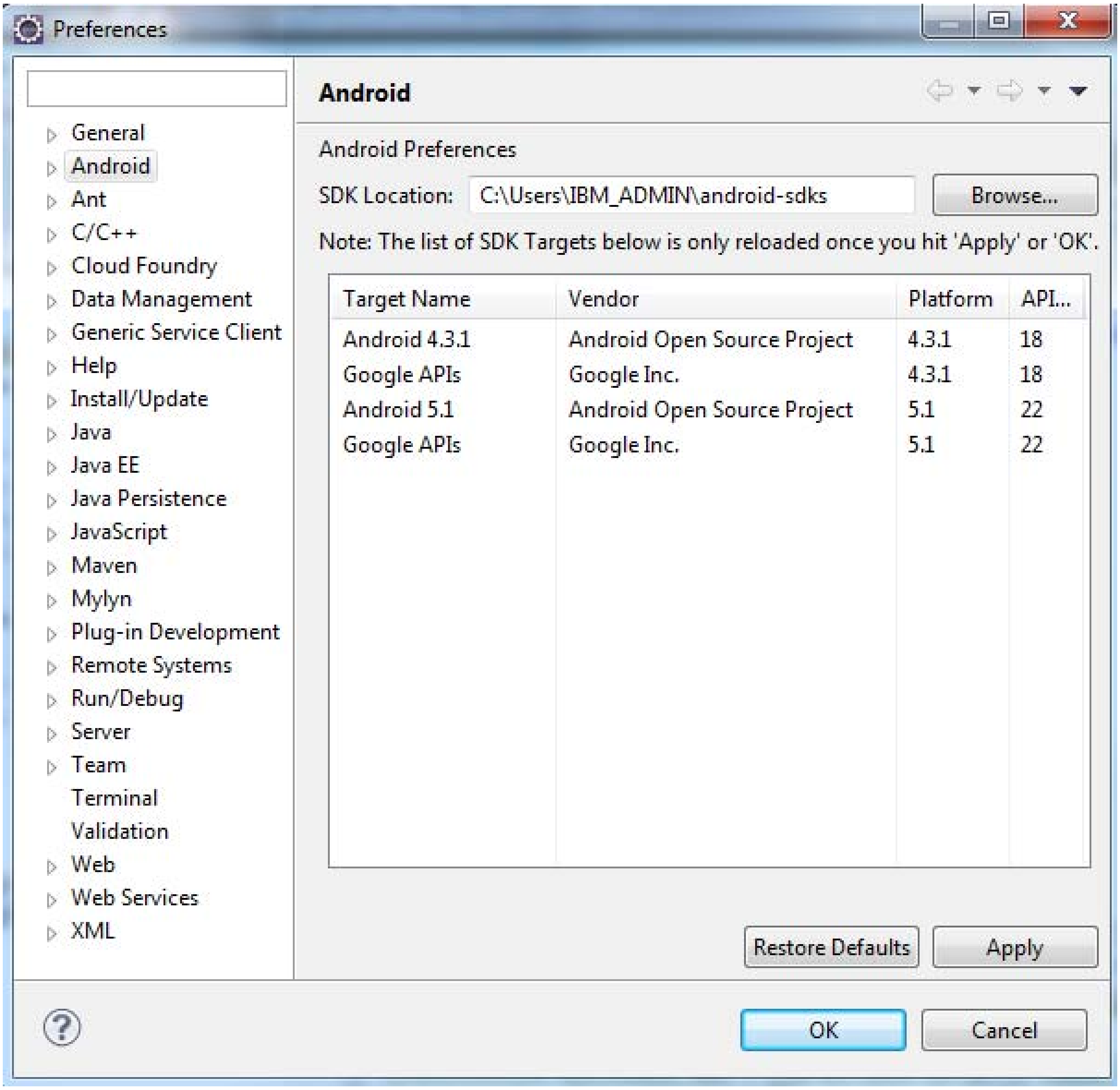
The packages are installed.

__ e. Close the Android SDK Manager.

__ 4. Configure the Android preferences in Eclipse.

__ a. From the Eclipse menu, click **Window > Preferences**.

__ b. Select **Android**. The list of SDK target APIs is displayed.



SDK Location: C:\Users\IBM_ADMIN\android-sdks

Note: The list of SDK Targets below is only reloaded once you hit 'Apply' or 'OK'.

Target Name	Vendor	Platform	API...
Android 4.3.1	Android Open Source Project	4.3.1	18
Google APIs	Google Inc.	4.3.1	18
Android 5.1	Android Open Source Project	5.1	22
Google APIs	Google Inc.	5.1	22

__ c. Click **OK**.

The Eclipse environment is now set up to run Android applications.

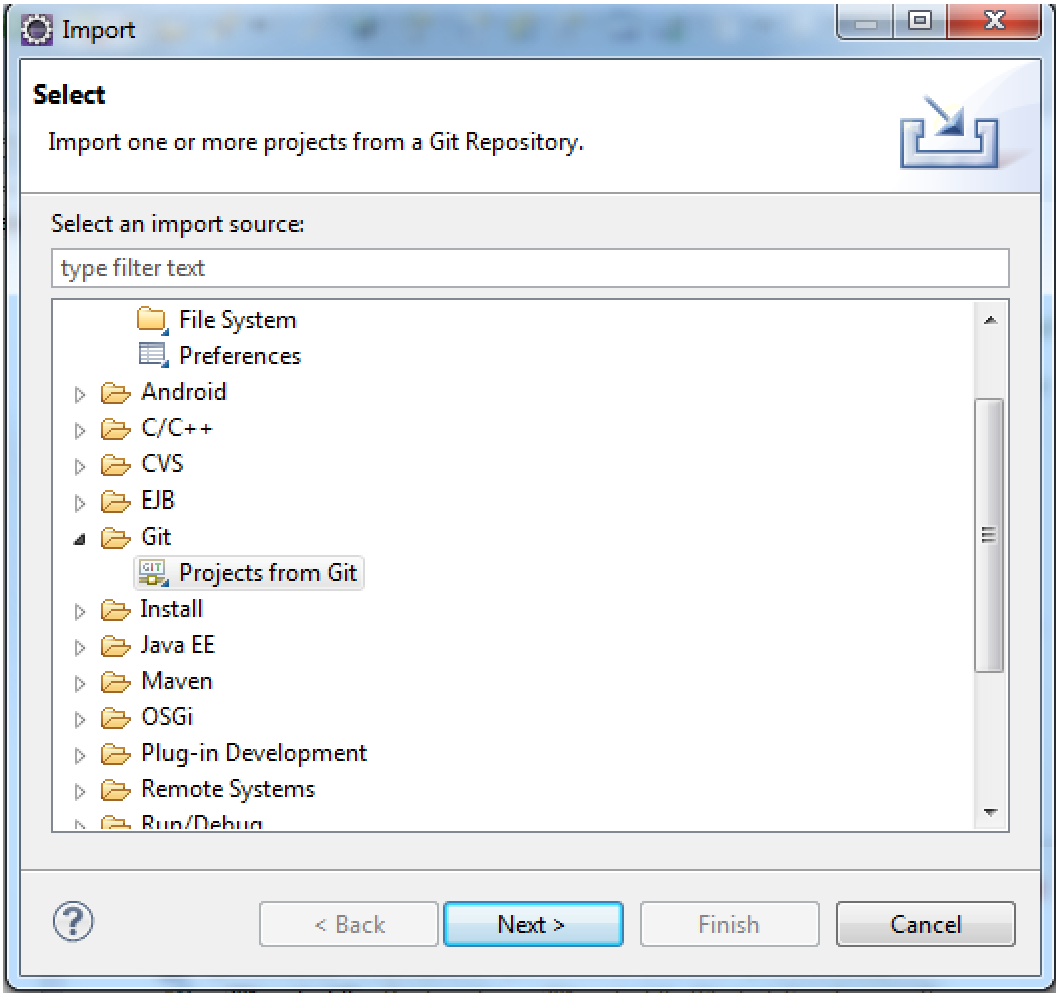
Part 5: Download and configure the Android BlueList mobile application

In this part of the exercise, you clone the sample application from IBM DevOps services directly to the Eclipse workspace. Then, you configure the mobile application to access the IBM Bluemix Mobile Data application that you defined earlier in the exercise.

__ 1. Perform a Git clone in Eclipse.

__ a. From the Eclipse main menu, click **File > Import**.

__ b. On the Select page of the Import wizard, select **Git > Projects from Git**.



Import

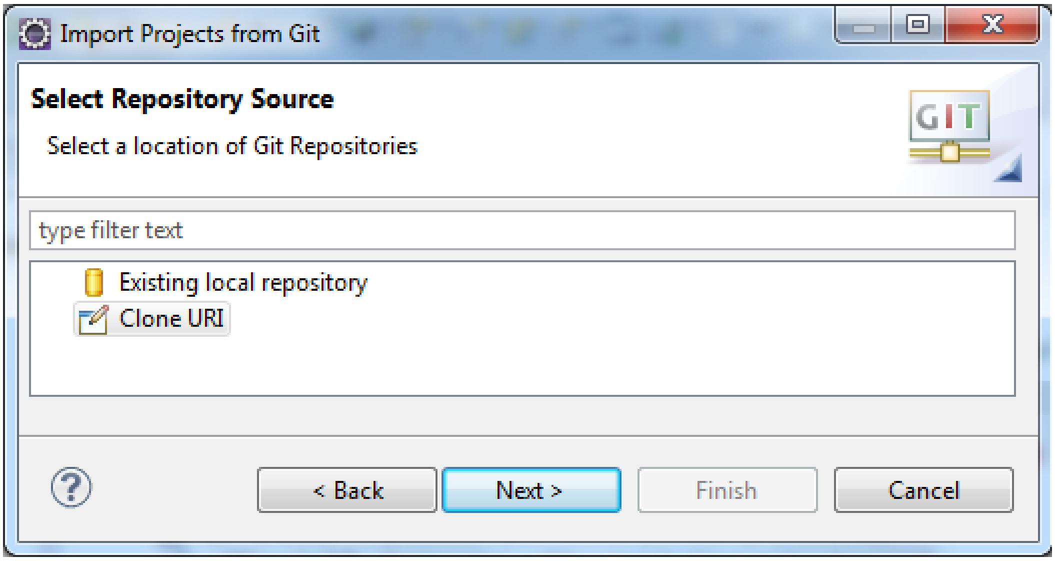
Select

Import one or more projects from a Git Repository.

Select an import source:

__ c. Click **Next**.

__ d. On the Select Repository Source page, select **Clone URI**.



Import Projects from Git

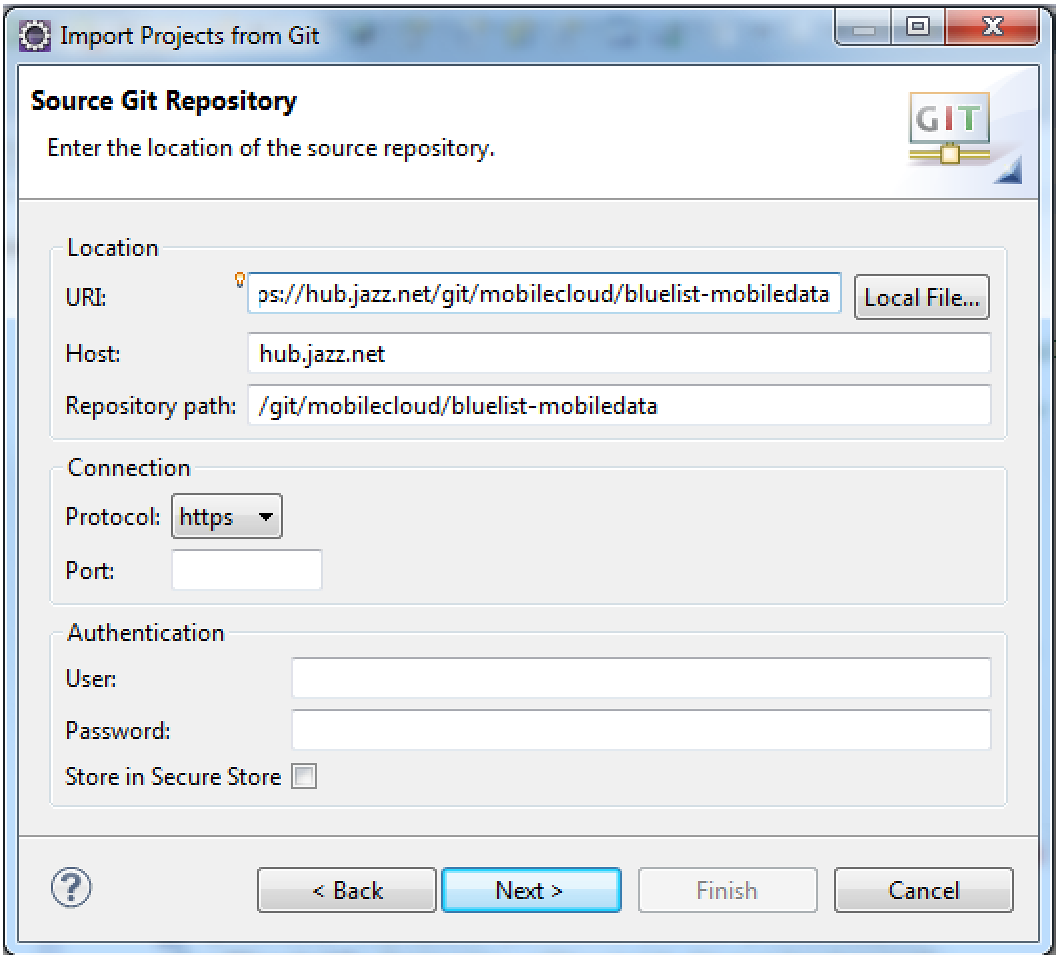
Select Repository Source

Select a location of Git Repositories

__ e. Click **Next**.

__ f. On the Source Git Repository page, type the following values and click **Next**:

- URI: `https://hub.jazz.net/git/mobilecloud/bluelist-mobiledata`
- Host: `hub.jazz.net`
- Repository path: `/git/mobilecloud/bluelist-mobiledata`



Import Projects from Git

Source Git Repository

Enter the location of the source repository.

Location

URI: ps://hub.jazz.net/git/mobilecloud/bluelist-mobiledata

Host: hub.jazz.net

Repository path: /git/mobilecloud/bluelist-mobiledata

Connection

Protocol: https

Port:

Authentication

User:

Password:

Store in Secure Store ☐

__ g. On the Branch Selection page, leave the default values (master selected).
Click **Next**.

__ h. On the Local Destination page, leave the default values.
Click **Next**.

__ i. On the “Select a wizard to use for importing projects” page, leave the default values (import existing projects).
Click **Next**.

__ j. On the “Import projects from a Git repository” page, ensure that the project is selected.
Click **Finish**.

You now see the project in the Project Explorer.

Note

There are some errors in the Java code of the project. These errors are fixed next.

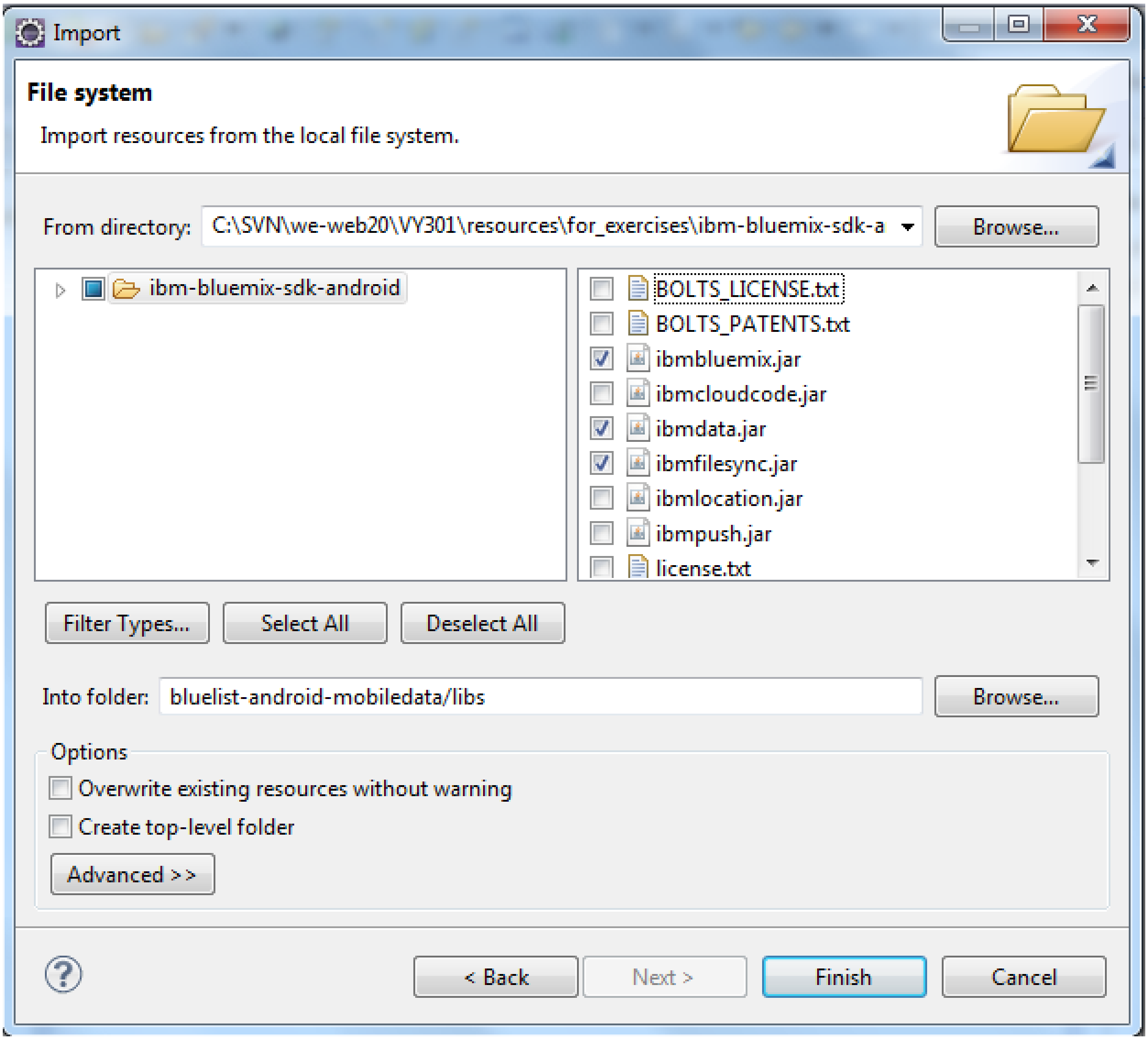
__ 2. Add a libs folder to the project.

- __ a. Right-click the bluelist-android-mobiledata project in the Project Explorer. Then, click **New > Folder**.
- __ b. Type `libs` for the folder name.
- __ c. Click **Finish**. The folder is added to the project hierarchy.

__ 3. Import the Java archive files from the Mobile Cloud Services Android SDK for IBM Bluemix that you downloaded earlier into the libs folder.

- __ a. In the Project Explorer, right-click the `libs` folder. Then, click **Import**.
- __ b. In the Import dialog box, select **File System** under the General option.
- __ c. Click **Next**.
- __ d. On the “File system” page, click **Browse** alongside the From directory.
- __ e. Go to the folder into which you extracted the Mobile Cloud SDK for Android.

__ f. Select **ibmbluemix.jar**, **ibmdata.jar**, and **ibmfilesync.jar**.



File system

Import resources from the local file system.

From directory: C:\SVN\we-web20\VY301\resources\for_exercises\ibm-bluemix-sdk-a

■ ibm-bluemix-sdk-android

- □ BOLTS_LICENSE.txt
- □ BOLTS_PATENTS.txt
- ☑ ibmbluemix.jar
- □ ibmcloudcode.jar
- ☑ ibmdata.jar
- ☑ ibmfilesync.jar
- □ ibmlocation.jar
- □ ibmpush.jar
- □ license.txt

Into folder: bluelist-android-mobiledata/libs

Options

□ Overwrite existing resources without warning

□ Create top-level folder

__ g. Click **Finish**.

The files are imported into the libs folder.

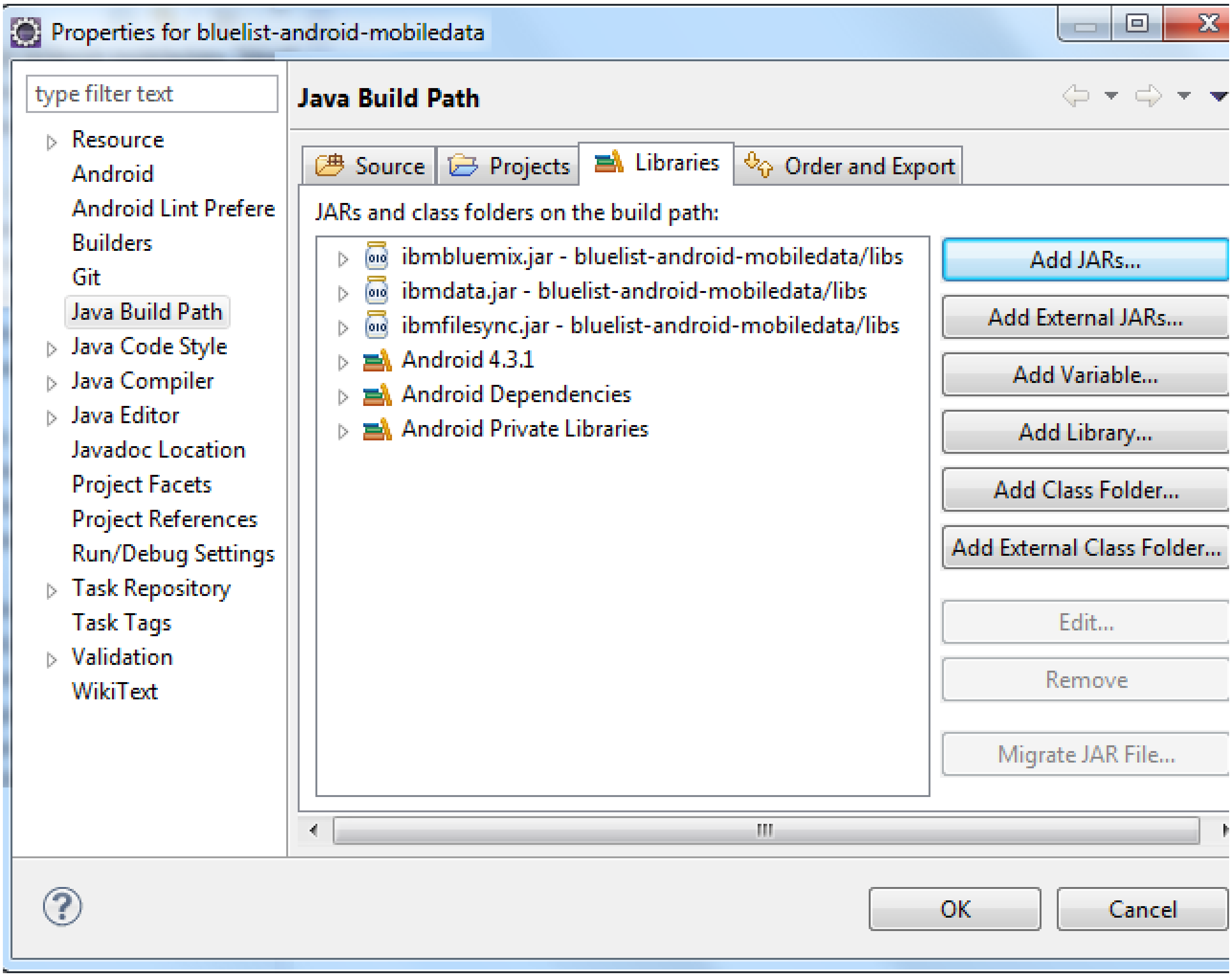
__ 4. Configure the Java build path for the project.

__ a. In the Project Explorer, right-click the **bluelist-android-mobiledata** project, then click **Build Path > Configure Build Path**.

__ b. Click the **Libraries** tab in the Properties dialog box.

__ c. Click **Add JARs**.

__ d. Add each of the JARs from the libs folder.
When complete, the Java Build Path dialog box includes the three JARs on the build path.



JARs and class folders on the build path:

- ibmbluemix.jar - bluelist-android-mobiledata/libs
- ibmdata.jar - bluelist-android-mobiledata/libs
- ibmfilesync.jar - bluelist-android-mobiledata/libs
- Android 4.3.1
- Android Dependencies
- Android Private Libraries

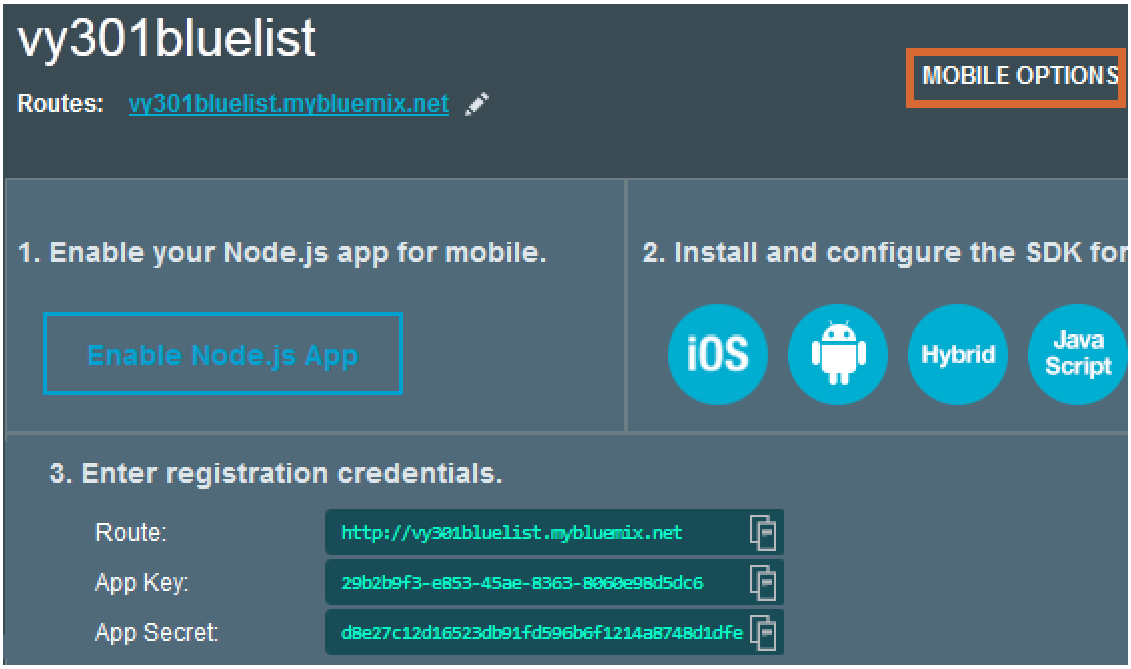
__ e. Click **OK**.

There are no errors in the bluelist-android-mobiledata project.

__ 5. Update the `bluelist.properties` file with the applicationID, applicationSecret, and applicationRoute values.

__ a. From the Dashboard in the IBM Bluemix browser window, click the application that you created at the start of the exercise.

__ b. Click **Mobile Options**.



vy301bluelist
Routes: vy301bluelist.mybluemix.net
MOBILE OPTIONS
1. Enable your Node.js app for mobile.
Enable Node.js App
2. Install and configure the SDK for
iOS
Hybrid
Java Script
3. Enter registration credentials.

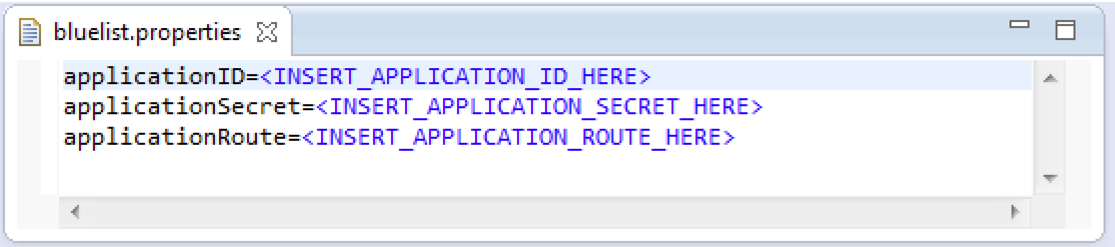
Route: http://vy301bluelist.mybluemix.net

App Key: 29b2b9f3-e853-45ae-8363-8060e98d5dc6

App Secret: d8e27c12d16523db91fd596b6f1214a8748d1dfe

You see the registration credentials for the Bluemix application. There is an option to copy each of these values to the clipboard.

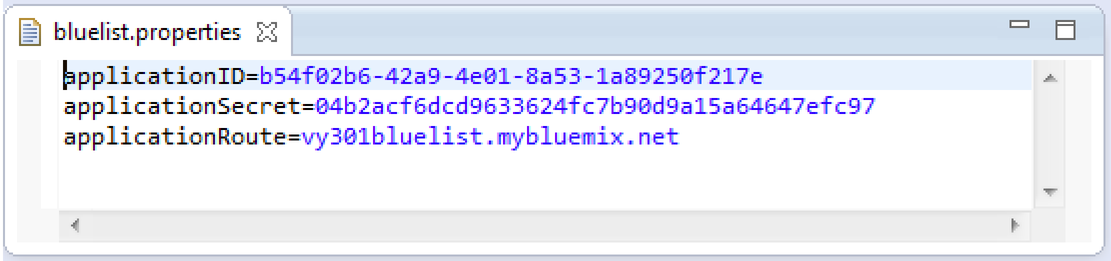
__ c. In the bluelist-android-mobiledata project in Eclipse, expand the assets folder. Then, double-click **bluelist.properties** to open it with the editor.



```
applicationID=<INSERT_APPLICATION_ID_HERE>
applicationSecret=<INSERT_APPLICATION_SECRET_HERE>
applicationRoute=<INSERT_APPLICATION_ROUTE_HERE>
```

__ d. Paste the values for each of the properties into the editor. Do NOT include string quotation marks around the values.

Example values are shown in the figure.



```
applicationID=b54f02b6-42a9-4e01-8a53-1a89250f217e
applicationSecret=04b2acf6dcd9633624fc7b90d9a15a64647efc97
applicationRoute=vy301bluelist.mybluemix.net
```

__ e. Save the changes.

__ f. The application is now ready to run.
But first, you should review the Java code for the Android application.

Part 6: Review the Android Java code

Now you review the Java code for the mobile application.

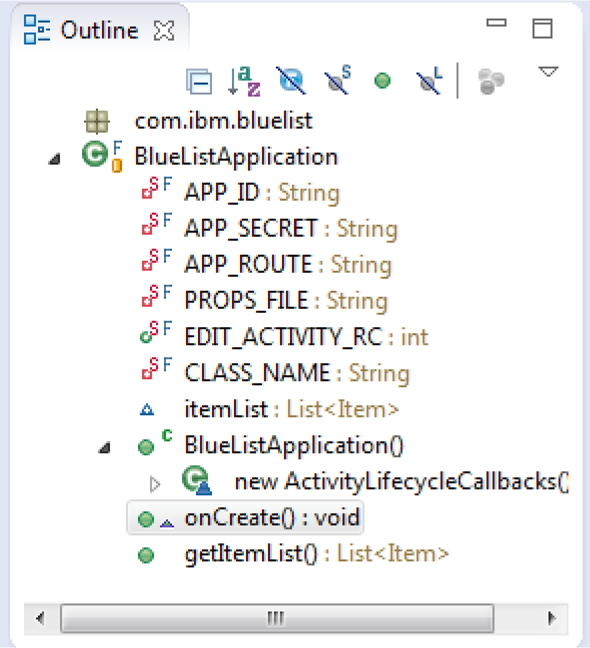
Android mobile applications are written in Java. In this part of the exercise, you see the Java code that communicates between the mobile application that runs on an Android emulator on the workstation and the IBM Bluemix application in the cloud.

__ 1. Review the Android mobile application source code in Eclipse.

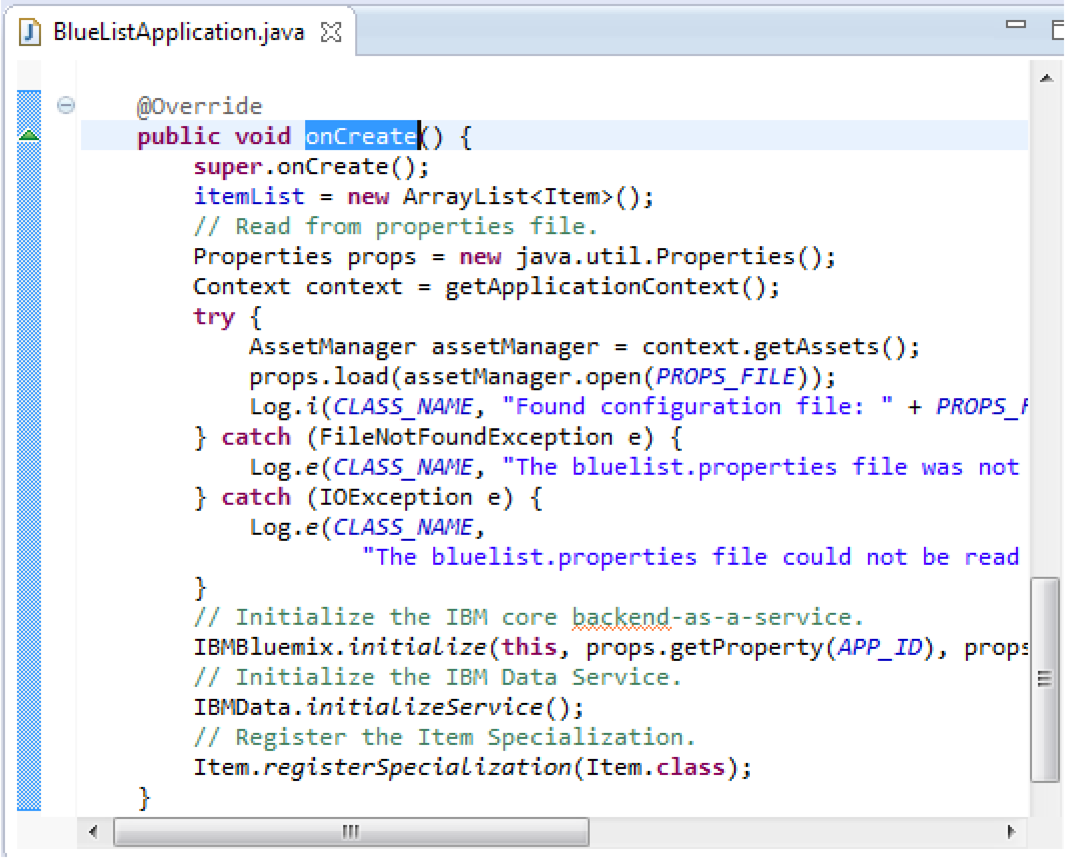
__ a. In the bluelist-android-mobiledata project, expand the **src** folder.
Then, double-click **BluelistApplication.java** to open it with the editor.

You see the source code in the editor.

__ b. In the Outline view, click the **onCreate()** method.



The editor positions the source code at the onCreate() method.



```
    @Override
    public void onCreate() {
        super.onCreate();
        itemList = new ArrayList<Item>();
        // Read from properties file.
        Properties props = new java.util.Properties();
        Context context = getApplicationContext();
        try {
            AssetManager assetManager = context.getAssets();
            props.load(assetManager.open(PROPS_FILE));
            Log.i(CLASS_NAME, "Found configuration file: " + PROPS_F
        } catch (FileNotFoundException e) {
            Log.e(CLASS_NAME, "The bluelist.properties file was not
        } catch (IOException e) {
            Log.e(CLASS_NAME,
                    "The bluelist.properties file could not be read
        }
        // Initialize the IBM core backend-as-a-service.
        IBMBluemix.initialize(this, props.getProperty(APP_ID), props
        // Initialize the IBM Data Service.
        IBMData.initializeService();
        // Register the Item Specialization.
        Item.registerSpecialization(Item.class);
    }
```

Notice that the method reads the `bluelist.properties` file. Then, the code initializes the IBM Bluemix environment and the IBMData service.

__ c. You can optionally review the source code in the other Java files. When complete, close the files in the editor.

Part 7: Install the Android virtual device

This part of the exercise assumes that you installed the Eclipse ADT plug-in and the Android SDK and that there are no Android virtual devices defined.

If you already have Android devices that are defined, skip this section and go to the next part in the exercise.

__ 1. Add the Android virtual device in Eclipse.

__ a. Go to the directory where the Android SDK is installed on your workstation.

__ b. Double-click **AVD Manager.exe**.

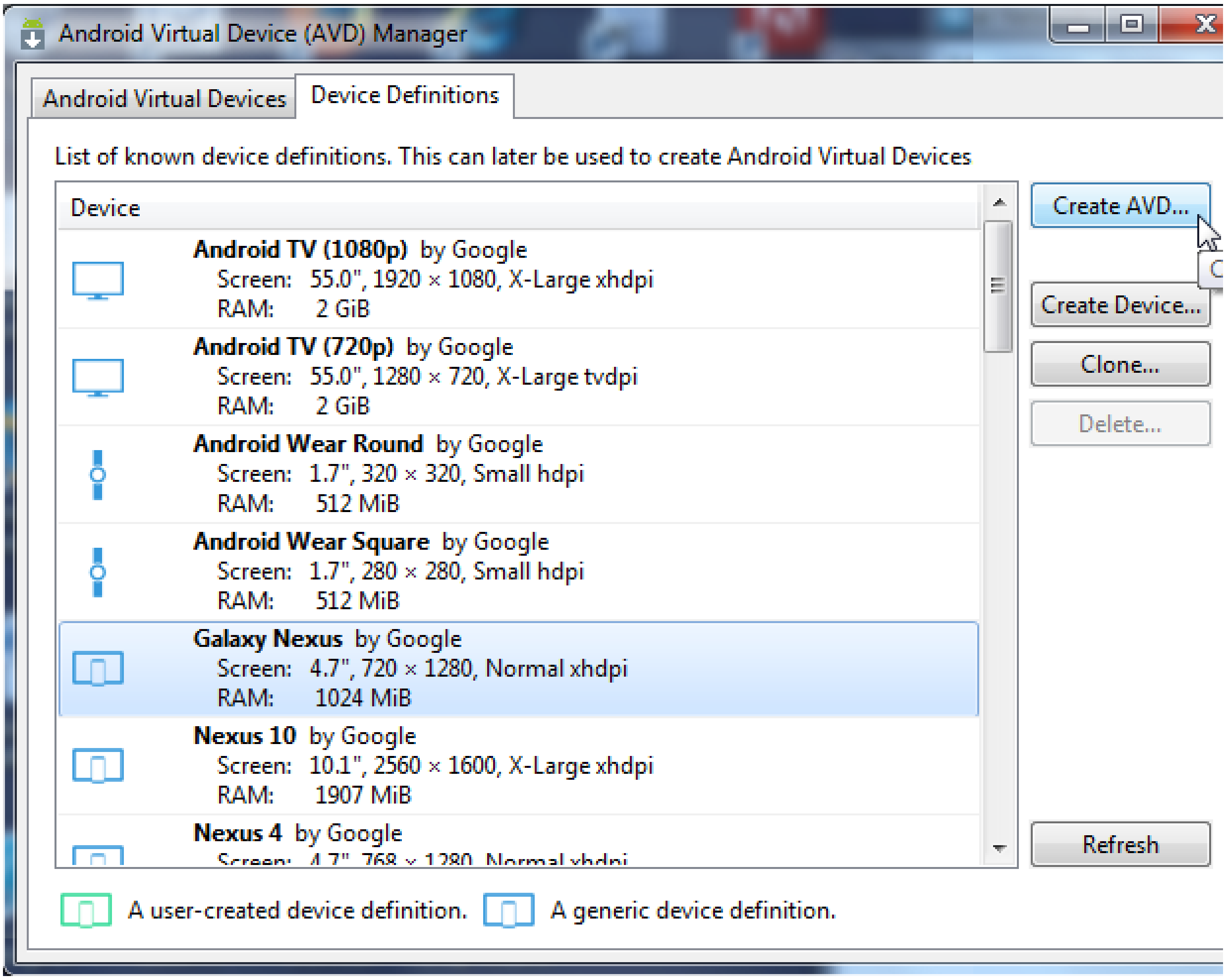
Note

If you do not find this file, run **android.bat** from the `/tools` directory where the Android SDK is installed on your workstation. Then, click **Tools > AVD Manager**.

It is a good practice to update the Android SDK to the latest version by using the `android.bat` utility.

__ c. Click the **Device Definitions** tab in the Android Virtual Device Manager dialog box.

__ d. Select a device such as Galaxy Nexus. Then, click **Create AVD**.



List of known device definitions. This can later be used to create Android Virtual Devices

Device

Android TV (1080p) by Google
Screen: 55.0", 1920 × 1080, X-Large xhdpi
RAM: 2 GiB

Android TV (720p) by Google
Screen: 55.0", 1280 × 720, X-Large tvdpi
RAM: 2 GiB

Android Wear Round by Google
Screen: 1.7", 320 × 320, Small hdpi
RAM: 512 MiB

Android Wear Square by Google
Screen: 1.7", 280 × 280, Small hdpi
RAM: 512 MiB

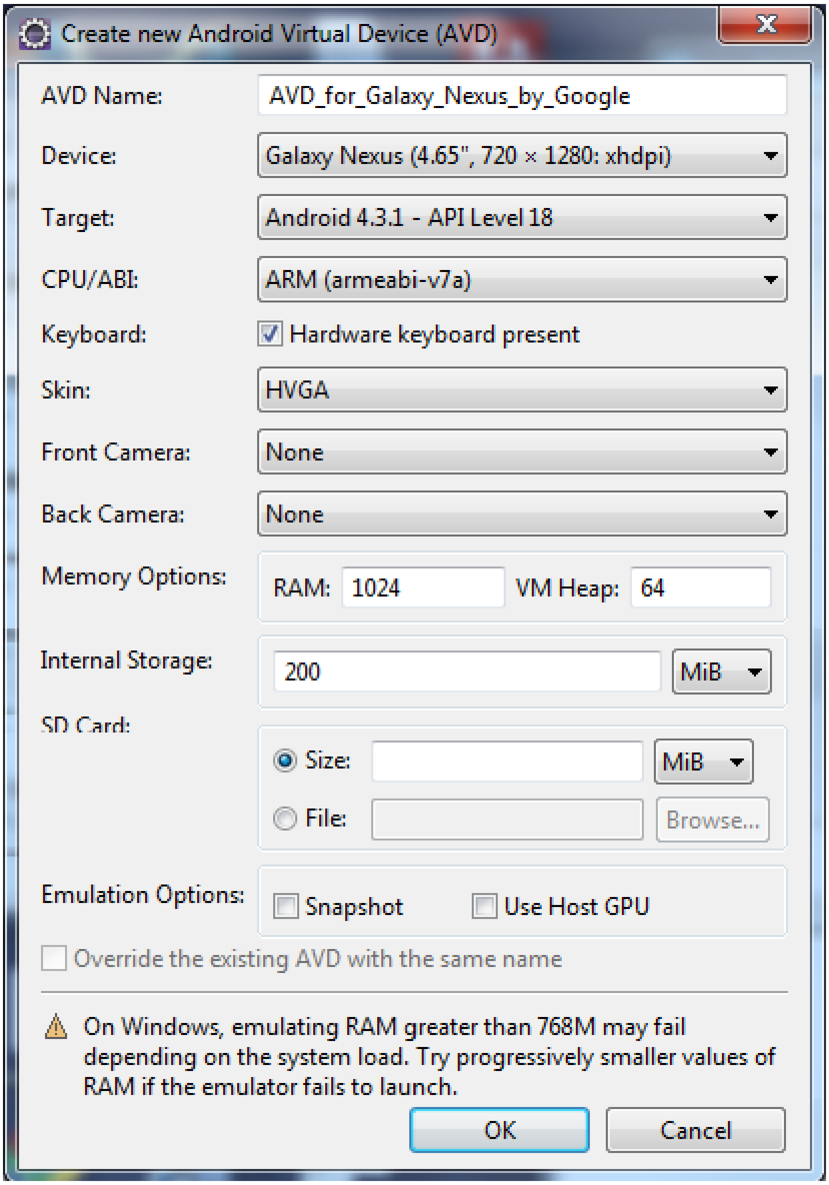
Galaxy Nexus by Google
Screen: 4.7", 720 × 1280, Normal xhdpi
RAM: 1024 MiB

Nexus 10 by Google
Screen: 10.1", 2560 × 1600, X-Large xhdpi
RAM: 1907 MiB

Nexus 4 by Google
Screen: 4.7", 768 × 1280, Normal xhdpi

A user-created device definition. A generic device definition.

__ e. In the “Create new Android Virtual Device (AVD)” dialog box, ensure that the target matches the API level of the application.



Create new Android Virtual Device (AVD)

AVD Name: AVD_for_Galaxy_Nexus_by_Google

Device: Galaxy Nexus (4.65", 720 × 1280: xhdpi)

Target: Android 4.3.1 - API Level 18

CPU/ABI: ARM (armeabi-v7a)

Keyboard: ☑ Hardware keyboard present

Skin: HVGA

Front Camera: None

Back Camera: None

Memory Options: RAM: 1024 VM Heap: 64

Internal Storage: 200 MiB

SD Card:

● Size: MiB

○ File: Browse...

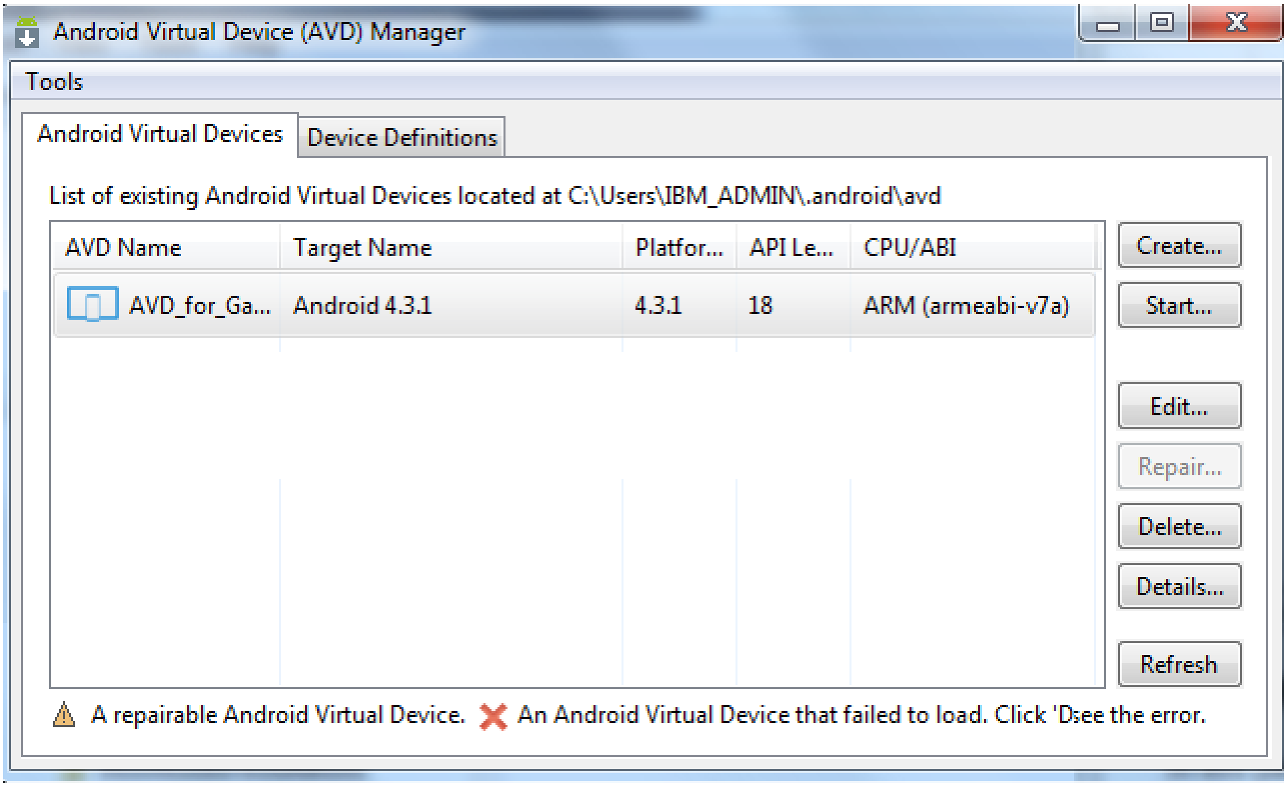
Emulation Options: □ Snapshot □ Use Host GPU

□ Override the existing AVD with the same name

⚠ On Windows, emulating RAM greater than 768M may fail depending on the system load. Try progressively smaller values of RAM if the emulator fails to launch.

__ f. Click **OK**.

The device is added to the list of Android virtual devices.



List of existing Android Virtual Devices located at C:\Users\IBM_ADMIN\.android\avd

AVD Name	Target Name	Platfor...	API Le...	CPU/ABI
AVD_for_Ga...	Android 4.3.1	4.3.1	18	ARM (armeabi-v7a)

⚠ A repairable Android Virtual Device. ✖ An Android Virtual Device that failed to load. Click 'Dsee the error.

Part 8: Run the BlueList Android application

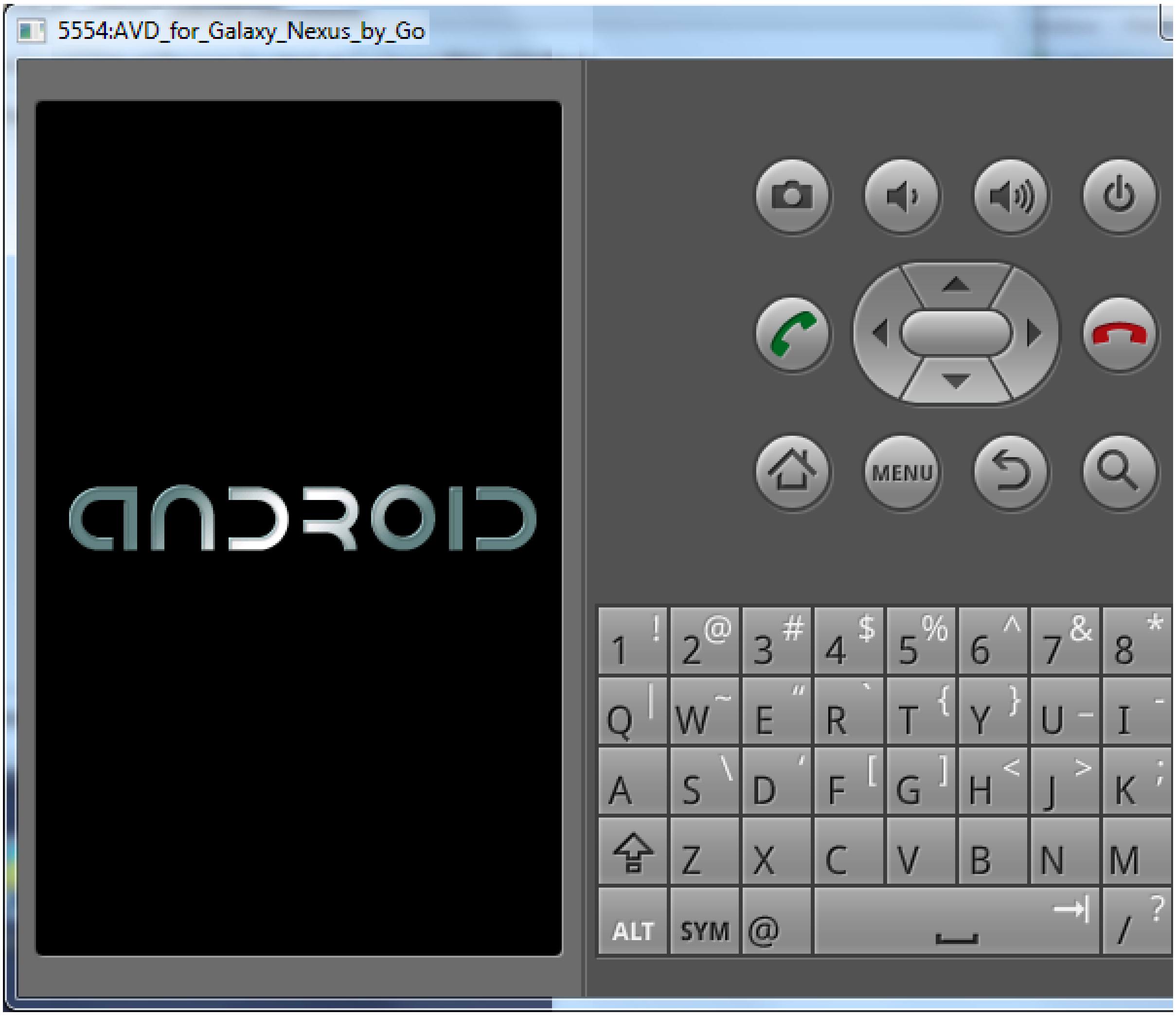
In this part, you run the mobile BlueList application in an Android emulator from Eclipse.

The Android application the Mobile Data application that runs on IBM Bluemix and persists the data on a cloud database.

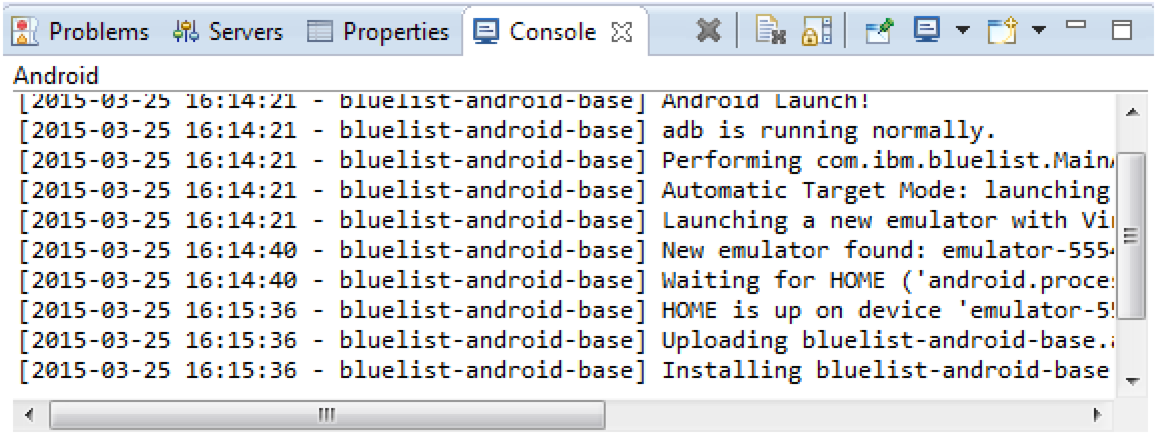
__ 1. Run the Android application in Eclipse.

__ a. From the Project Explorer, right-click the application, then click **Run As > Android Application**.

After a few moments, you see the Android emulator.



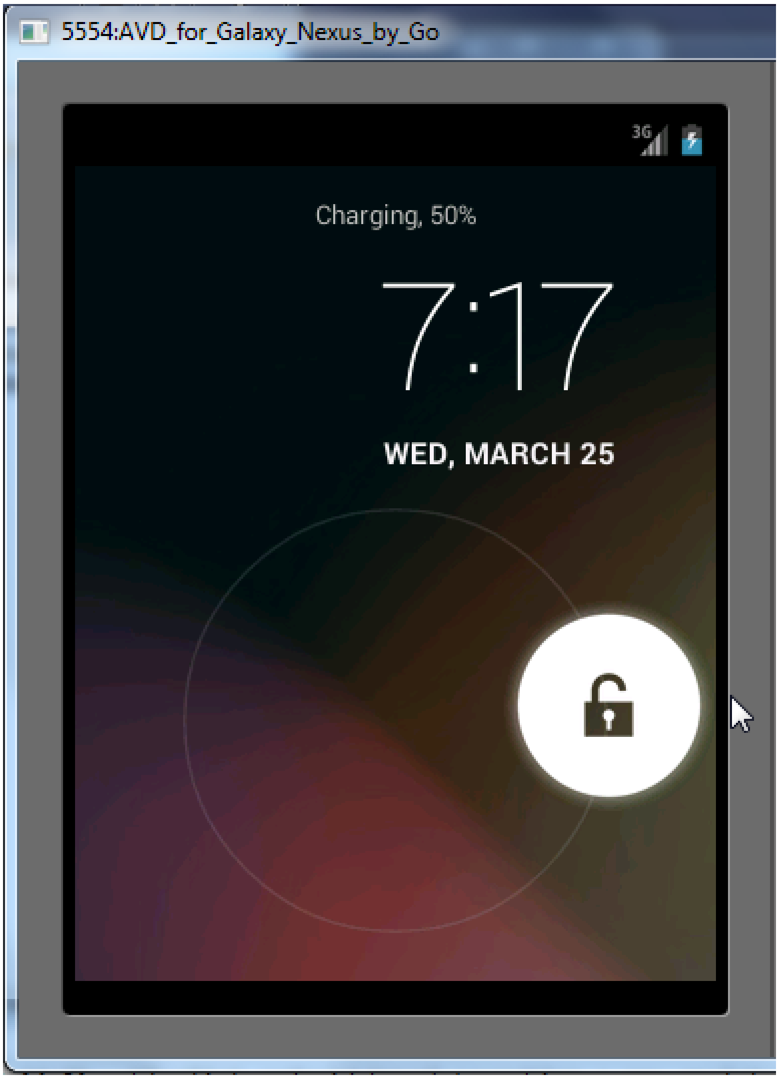
In the Eclipse console, you see that the Android application is being installed.



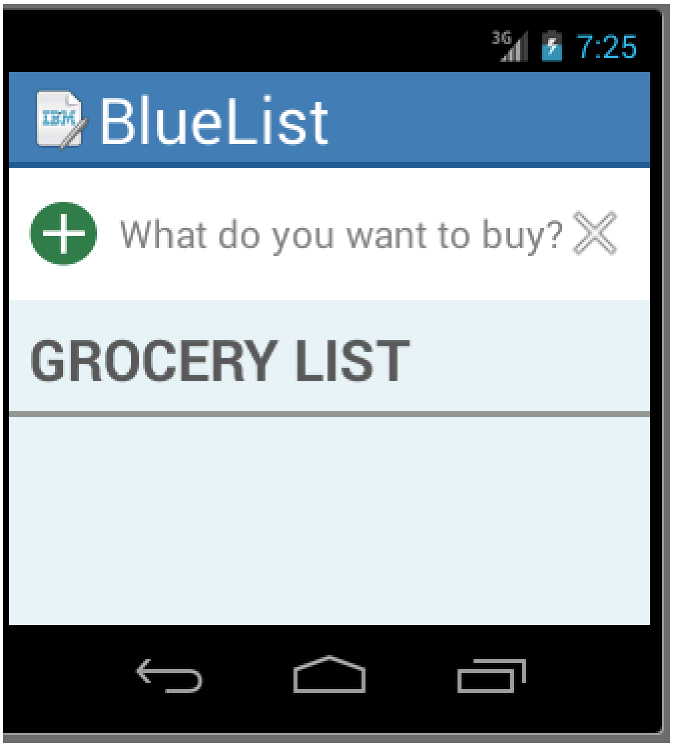
Android

```
[2015-03-25 16:14:21 - bluelist-android-base] Android Launch!
[2015-03-25 16:14:21 - bluelist-android-base] adb is running normally.
[2015-03-25 16:14:21 - bluelist-android-base] Performing com.ibm.bluelist.Main
[2015-03-25 16:14:21 - bluelist-android-base] Automatic Target Mode: launching
[2015-03-25 16:14:21 - bluelist-android-base] Launching a new emulator with Vi
[2015-03-25 16:14:40 - bluelist-android-base] New emulator found: emulator-555
[2015-03-25 16:14:40 - bluelist-android-base] Waiting for HOME ('android.proce
[2015-03-25 16:15:36 - bluelist-android-base] HOME is up on device 'emulator-5
[2015-03-25 16:15:36 - bluelist-android-base] Uploading bluelist-android-base.
[2015-03-25 16:15:36 - bluelist-android-base] Installing bluelist-android-base
```

__ b. After some time, the lock screen is displayed. Unlock the Android emulator by swiping the lock icon to the right.

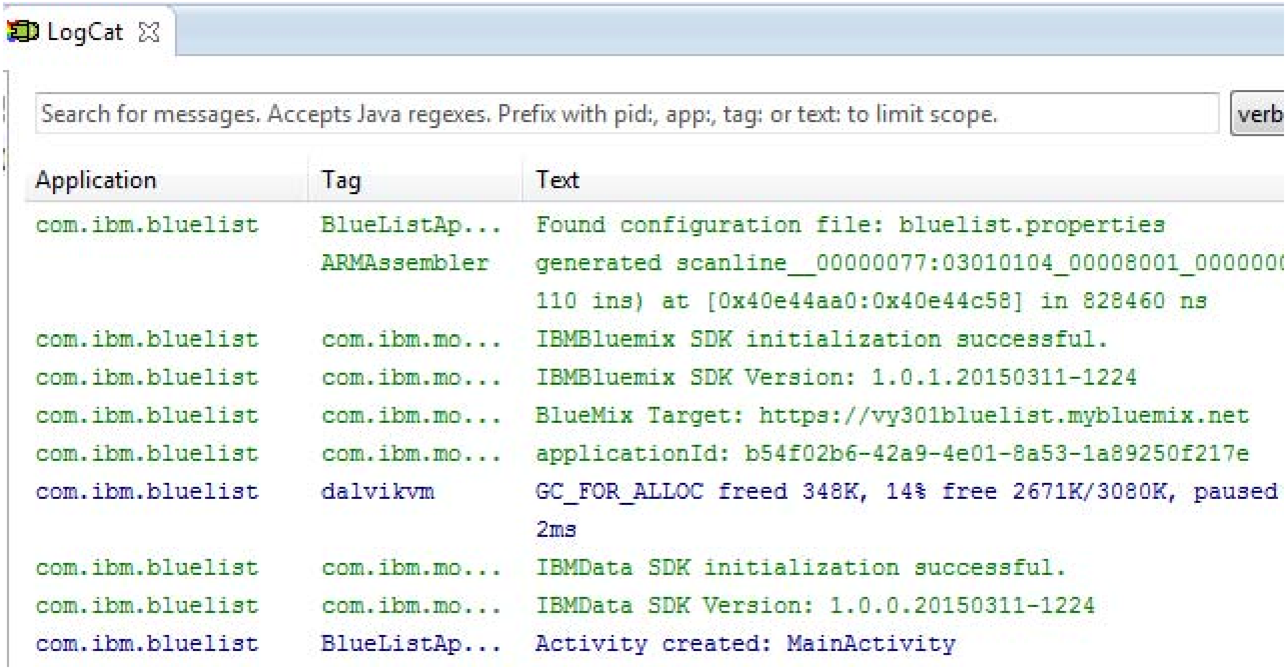


Now you see the BlueList application in the emulator.



The BlueList mobile application is successfully started and is now communicating with the Mobile Data application on IBM Bluemix.

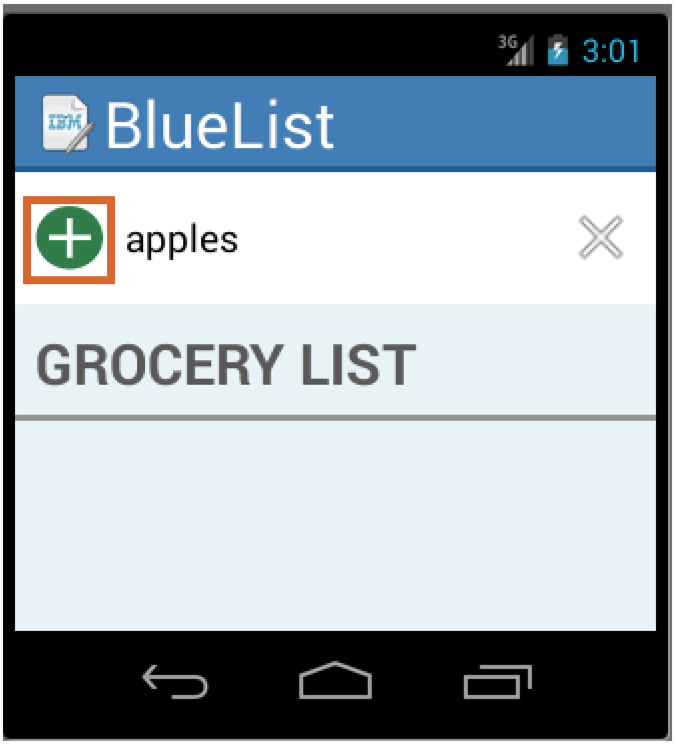
The Android log view in Eclipse displays some messages.


LogCat
Search for messages. Accepts Java regexes. Prefix with pid:, app:, tag: or text: to limit scope.
verb

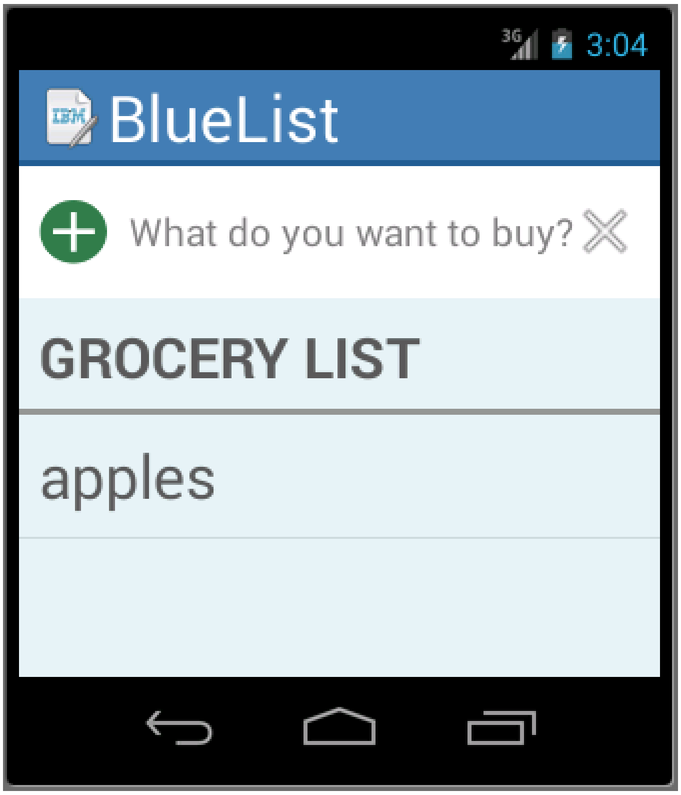
```
Application        Tag              Text
com.ibm.bluelist   BlueListAp...    Found configuration file: bluelist.properties
                   ARMAssembler     generated scanline__00000077:03010104_00008001_000000
                                    110 ins) at [0x40e44aa0:0x40e44c58] in 828460 ns
com.ibm.bluelist   com.ibm.mo...    IBMBluemix SDK initialization successful.
com.ibm.bluelist   com.ibm.mo...    IBMBluemix SDK Version: 1.0.1.20150311-1224
com.ibm.bluelist   com.ibm.mo...    BlueMix Target: https://vy301bluelist.mybluemix.net
com.ibm.bluelist   com.ibm.mo...    applicationId: b54f02b6-42a9-4e01-8a53-1a89250f217e
com.ibm.bluelist   dalvikvm         GC_FOR_ALLOC freed 348K, 14% free 2671K/3080K, paused
                                    2ms
com.ibm.bluelist   com.ibm.mo...    IBMData SDK initialization successful.
com.ibm.bluelist   com.ibm.mo...    IBMData SDK Version: 1.0.0.20150311-1224
com.ibm.bluelist   BlueListAp...    Activity created: MainActivity
```

__ 2. Type some data into the BlueList mobile application.

__ a. Type items into the grocery list, and click the plus icon to add the typed value to the list.



The item is displayed in the list.



The data items are persisted even when you rerun the mobile emulator. The data is being stored on a cloud database.

Part 9: Review the data on the Bluemix cloud

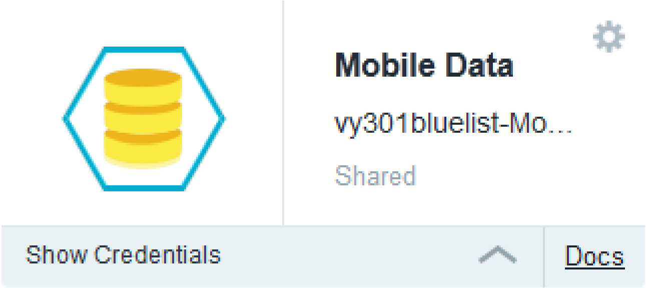
Here, you review the data in the cloud.

__ 1. Sign on to Bluemix, if you are not already signed on.

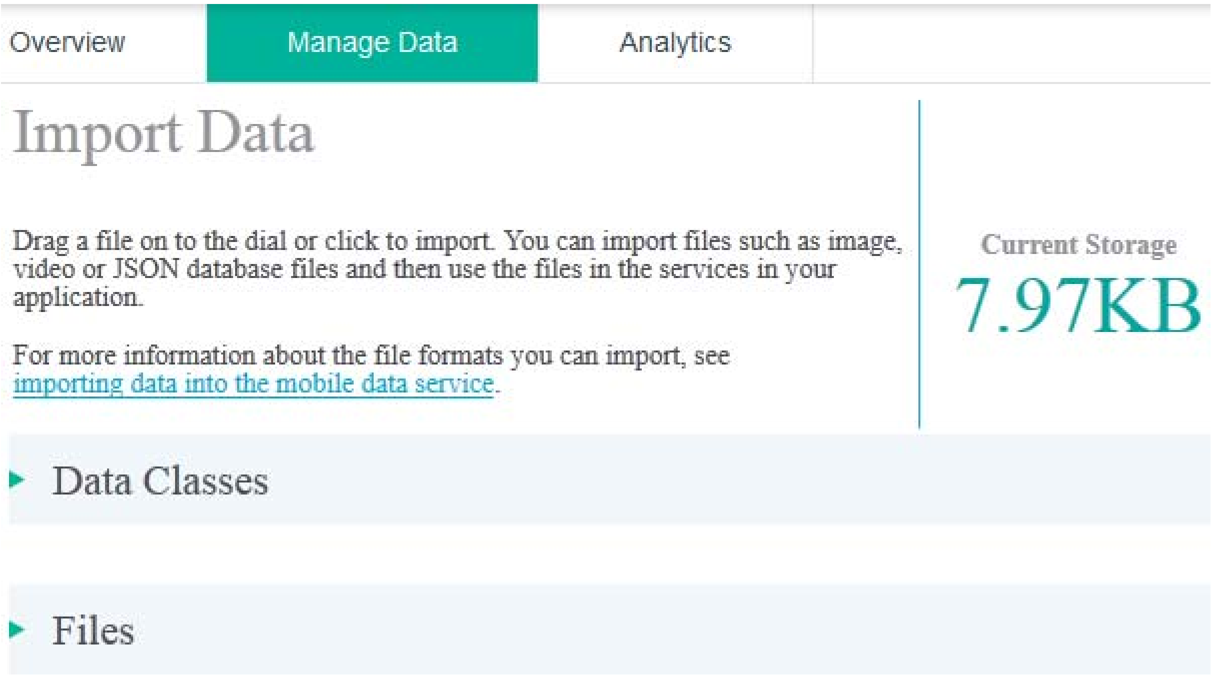
__ 2. Select the application in the Dashboard view.

__ 3. Review the mobile data.

__ a. Click the **Mobile Data** service.



__ b. Click the **Manage Data** tab.



Import Data

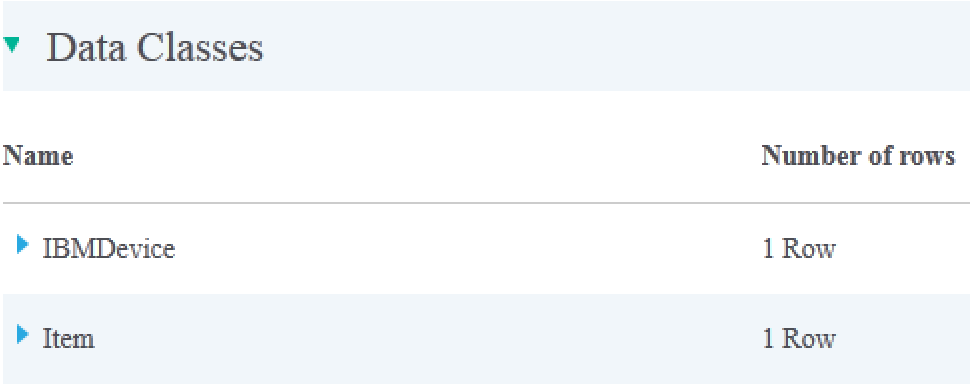
Drag a file on to the dial or click to import. You can import files such as image, video or JSON database files and then use the files in the services in your application.

For more information about the file formats you can import, see importing data into the mobile data service.

Current Storage
7.97KB

Notice that the current storage has a nonzero value, indicating that some data is stored.

__ c. Expand **Data Classes**.

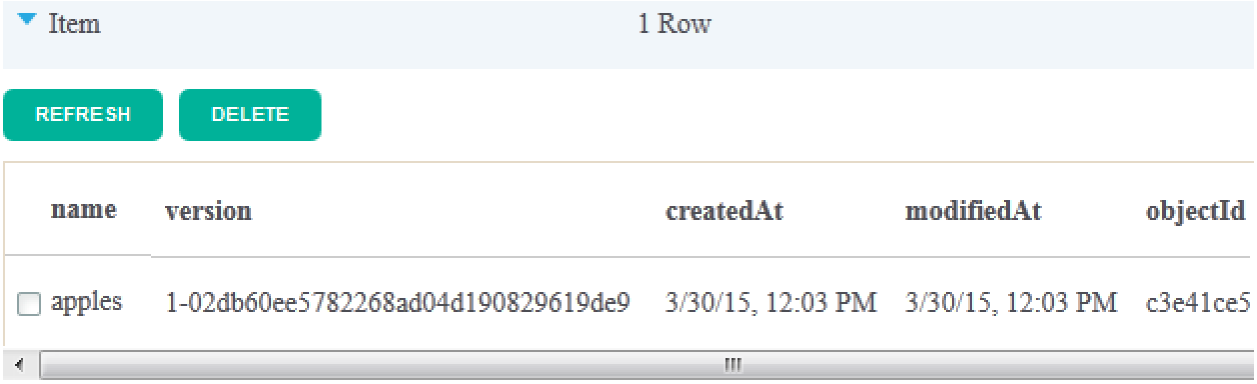


▼ Data Classes

Name	Number of rows
▸ IBMDevice	1 Row
▸ Item	1 Row

Notice that the Item row includes the number of items that were added to the BlueList mobile application.

__ d. Expand **Item**.



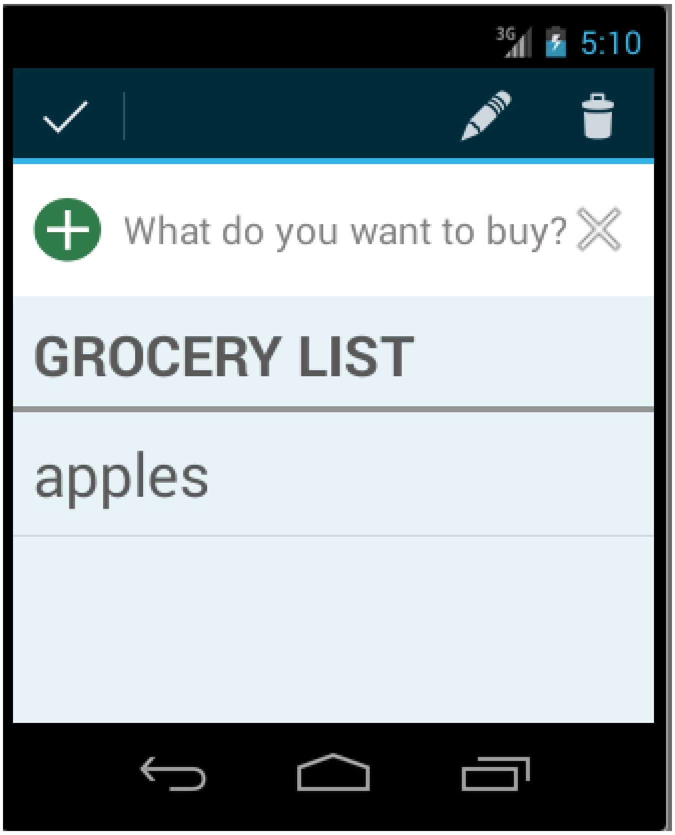
▼ Item 1 Row

name	version	createdAt	modifiedAt	objectId
□ apples	1-02db60ee5782268ad04d190829619de9	3/30/15, 12:03 PM	3/30/15, 12:03 PM	c3e41ce5

You see the item that you added in the BlueList Android application.

__ 4. Feel free to restart the Android mobile application and add, modify, or delete items in the list.

__ 5. To modify or delete an item in the Android emulator, swipe the list item to the left. An edit (pen) or delete (trash can) icon is displayed at the top of the emulator.



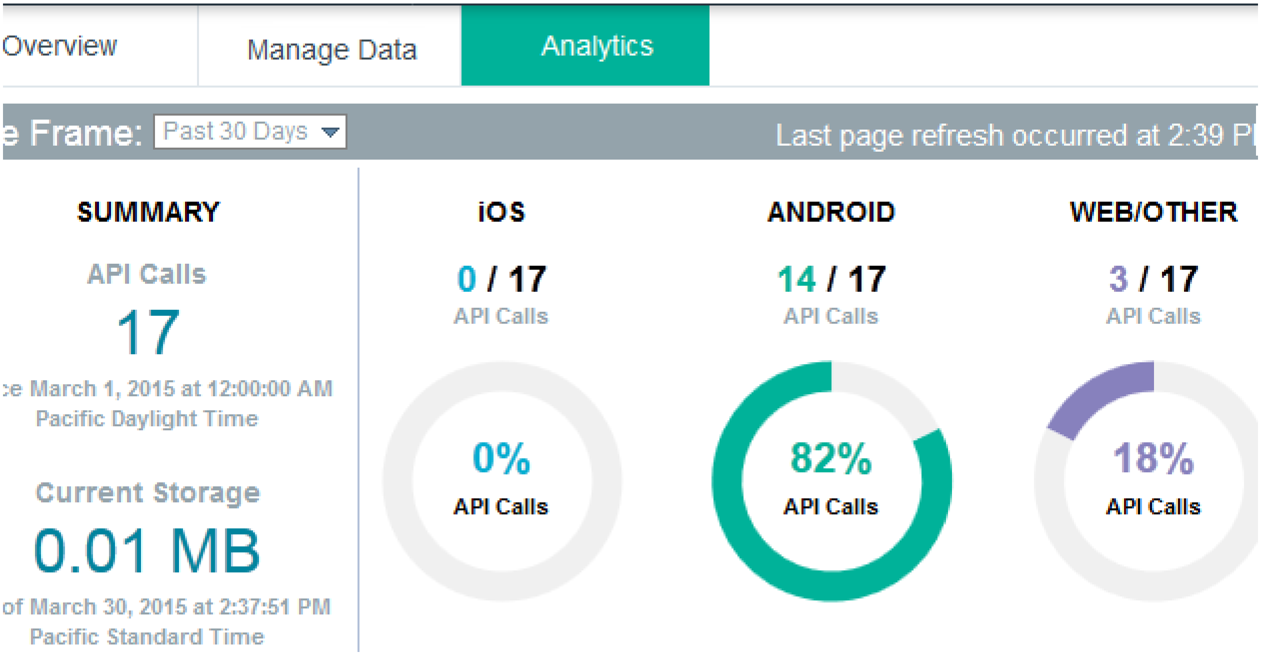
What do you want to buy?

GROCERY LIST

apples

__ 6. If you delete the item it is removed from the list. Refresh the Bluemix browser and notice that the Item row is no longer displayed in the Data Classes area.

__ 7. In Bluemix, click the **Analytics** tab to see the analysis of the API calls by device.



e Frame: Past 30 Days

Last page refresh occurred at 2:39 P

SUMMARY

API Calls

17

:e March 1, 2015 at 12:00:00 AM
Pacific Daylight Time

Current Storage

0.01 MB

of March 30, 2015 at 2:37:51 PM
Pacific Standard Time

iOS	ANDROID	WEB/OTHER
0 / 17	14 / 17	3 / 17
API Calls	API Calls	API Calls
0%	82%	18%
API Calls	API Calls	API Calls

End of exercise

Exercise review and wrap-up

In the exercise, you created the Bluemix Mobile Data cloud component from a Blueprint boilerplate.

You downloaded the Android SDK to the workstation. You made a Git clone of a sample client Android application from DevOps and reviewed the code in Eclipse.

You installed the Android development environment. Next, you tested the application with an Android emulator that ran in Eclipse.

You reviewed the changes to the data that was persisted on a cloud database that runs on Bluemix.

Exercise 6. Extending the Bluemix mobile data application to access it from a mobile web application

What this exercise is about

In this exercise, you add a mobile web application that displays the Bluemix data list in a web browser.

You download a sample application from Bluemix.

You change the sample application on your workstation so that it can access the Bluemix data service.

You run and test the mobile web application on the workstation.

Finally, you push the application to Bluemix and run the mobile web application on Bluemix.

What you should be able to do

After completing this exercise, you should be able to:

- Install the necessary command-line software on the workstation
- Download the sample mobile web application
- Configure the mobile web application to work with the Bluemix mobile data application
- Create a local instance of the mobile web application
- Run the local instance of the mobile web application
- Push the mobile web application to Bluemix
- Test the mobile web application in Bluemix

Introduction

This exercise complements the IBM Mobile Data for Bluemix cloud service application that was created in an earlier exercise. The exercise adds a web front end that displays the data from the earlier application in a web browser.

Requirements

You must have an IBM Bluemix account and must complete the previous exercises before doing this exercise.

Exercise instructions

Preface

This exercise depends on the availability of the IBM Bluemix cloud and connectivity from your own workstation to the cloud environment.

Part 1: Install the Node.js software

In the first part of the exercise, you get the mobile-web application to run on the workstation. To get the application to work on the local environment, you need to install the Node.js runtime environment and some node-dependent modules. You can install either the IBM Node.js SDK, or you can download and install the Node.js runtime environment from `http://nodejs.org`

In this part, instructions are given to install the IBM Node.js SDK.

__ 1. Verify that the Java Runtime Environment (JRE) works properly on your computer.

  __ a. Open a command prompt in Microsoft Windows.

  __ b. Run **`java -fullversion`** in the prompt.

  __ c. Verify that the Java Runtime Environment (JRE) works properly.

```
java full version JRE 1.7.0 ...
```

  __ d. If the command does not work, download and install the Java 7 Runtime Environment on your computer.

N

There are two versions of the Java Runtime Environment: 32-bit and 64-bit. Choose the version that matches your operating environment.

__ 2. Download a copy of the IBM SDK for Node.js.

  __ a. Open the web page: `www.ibm.com/developerworks/web/nodesdk/`

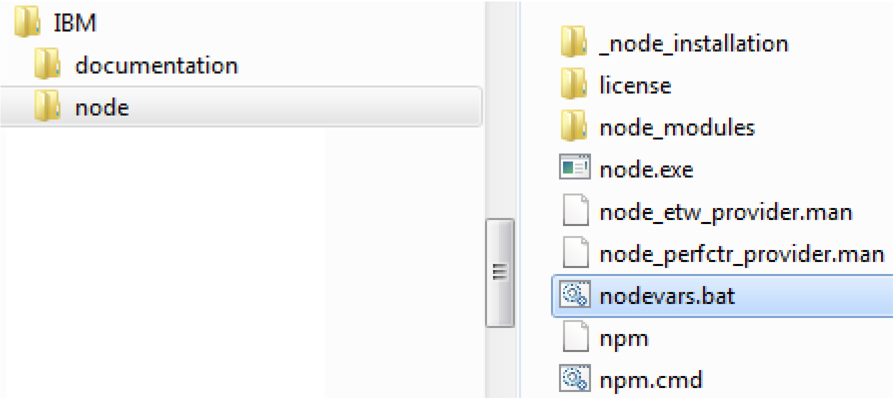
  __ b. Download the version of the IBNSDK for Node.js for your operating system.

__ 3. Install IBM SDK for Node.js on your computer.

  __ a. Run the installer application.

  __ b. Choose the installation directory for the installation.

__ c. Confirm that the IBM SDK for Node.js software is installed, and record the installation directory.



IBM
documentation
node

_node_installation
license
node_modules
node.exe
node_etw_provider.man
node_perfctr_provider.man
nodevars.bat
npm
npm.cmd

__ 4. Set up the environment variables for Node.js.

__ a. Open a command prompt in Microsoft Windows.

__ b. Run the **nodevars** script to set up the environment.

__ c. Confirm that the script configured the environment variables for Node.js.

```
C:\IBM\node]# nodevars.bat
Your environment has been set up for using Node.js 0.10.26 (x64) and npm.
```

N

Node comes with a version of npm installed. However, npm gets updated more frequently than Node does, so make sure that you have the latest version.

Part 2: Install Git software

In this part of the exercise, you install Git software.

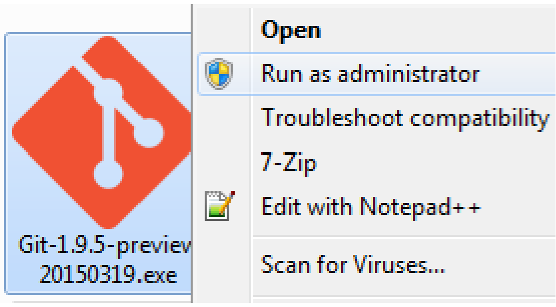
__ 1. Check whether Git is already installed on your computer.

__ a. On a command line, type `git version`.
If a version of Git is displayed, skip this part, and go to the next part of the exercise.

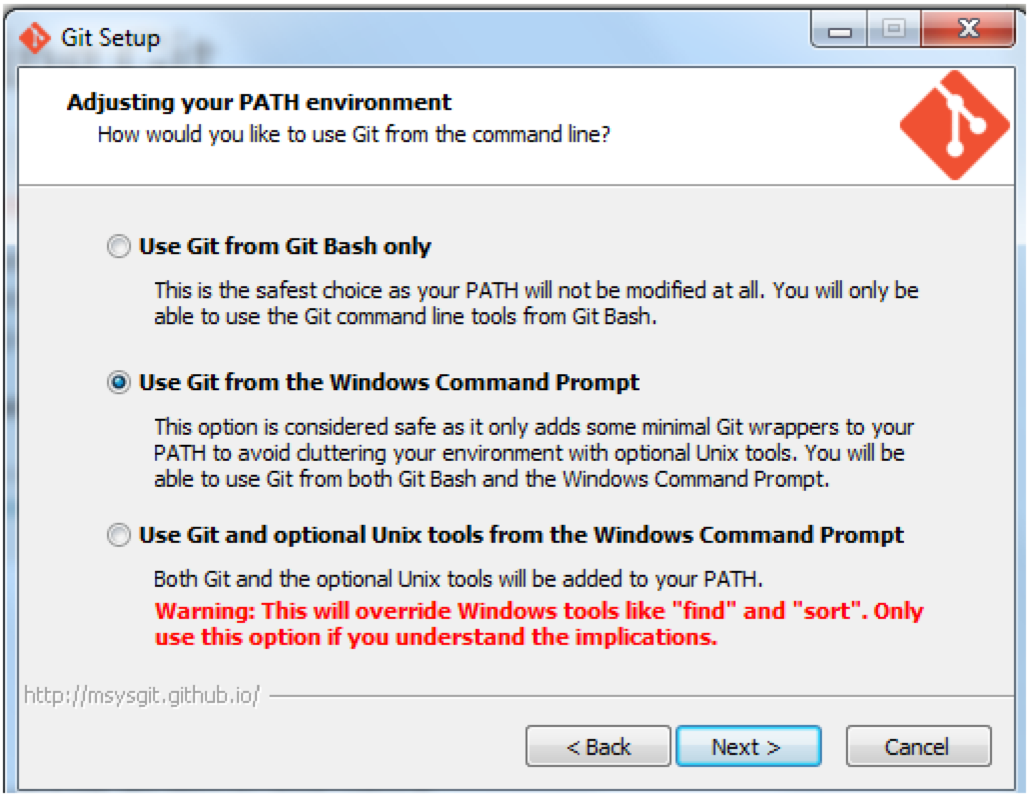
__ 2. If Git is not installed, go to the Git website `git-scm.com/downloads` and download the Git version for your operating system.

__ 3. Run the installation program executable file.

__ a. After downloading the file, start the installation executable program.



__ b. In the Git Setup wizard on the “Adjusting your PATH environment” page, select **Use Git from the Windows Command Prompt**.



Adjusting your PATH environment

How would you like to use Git from the command line?

○ **Use Git from Git Bash only**

This is the safest choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

● **Use Git from the Windows Command Prompt**

This option is considered safe as it only adds some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools. You will be able to use Git from both Git Bash and the Windows Command Prompt.

○ **Use Git and optional Unix tools from the Windows Command Prompt**

Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.

__ c. Run through the remaining installation pages. You can leave the rest of the options as their default values.

__ 4. When Git is installed, verify that it works on a command line.

__ a. Type `git version`

You see the version that is installed displayed in the command prompt.

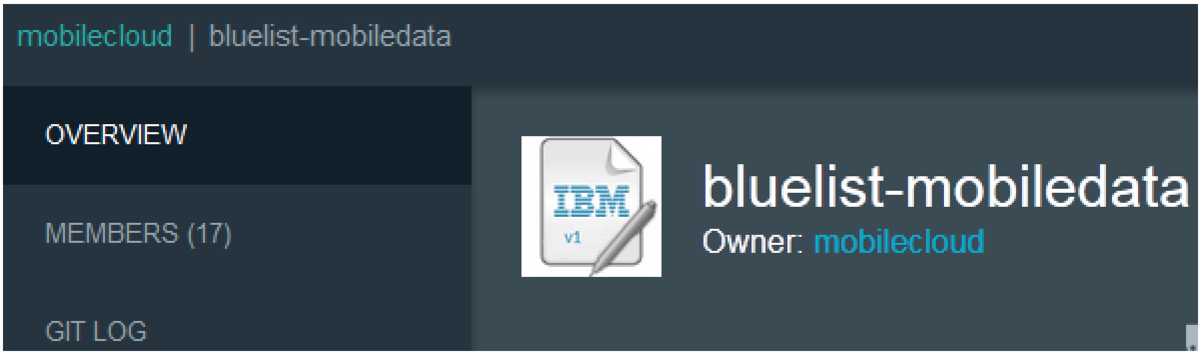
Part 3: Download and configure the sample mobile web project

In this part of the exercise, you download and extract the sample mobile web application.

__ 1. Sign in to your Bluemix DevOps services account.

In the browser, type:
`https://hub.jazz.net/project/mobilecloud/bluelist-mobiledata/overview`

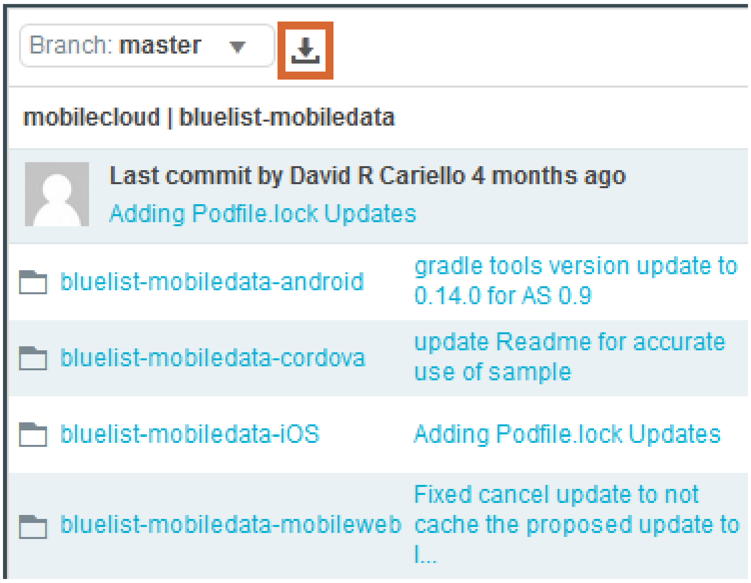
You see the bluelist-mobile data sample project.



mobilecloud | bluelist-mobiledata

__ 2. Download the sample.

__ a. From the DevOps services project, click the icon to download the entire project.

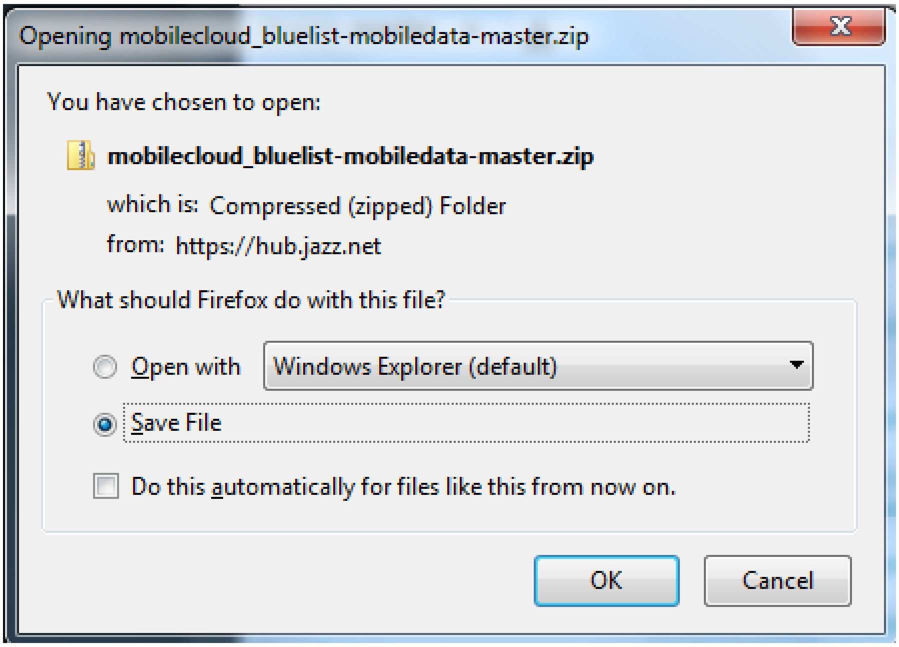


mobilecloud | bluelist-mobiledata

Last commit by David R Cariello 4 months ago
Adding Podfile.lock Updates

bluelist-mobiledata-android	gradle tools version update to 0.14.0 for AS 0.9
bluelist-mobiledata-cordova	update Readme for accurate use of sample
bluelist-mobiledata-iOS	Adding Podfile.lock Updates
bluelist-mobiledata-mobileweb	Fixed cancel update to not cache the proposed update to l...

__ b. Save the file on the local workstation.



Opening mobilecloud_bluelist-mobiledata-master.zip

You have chosen to open:

mobilecloud_bluelist-mobiledata-master.zip

which is: Compressed (zipped) Folder

from: https://hub.jazz.net

What should Firefox do with this file?

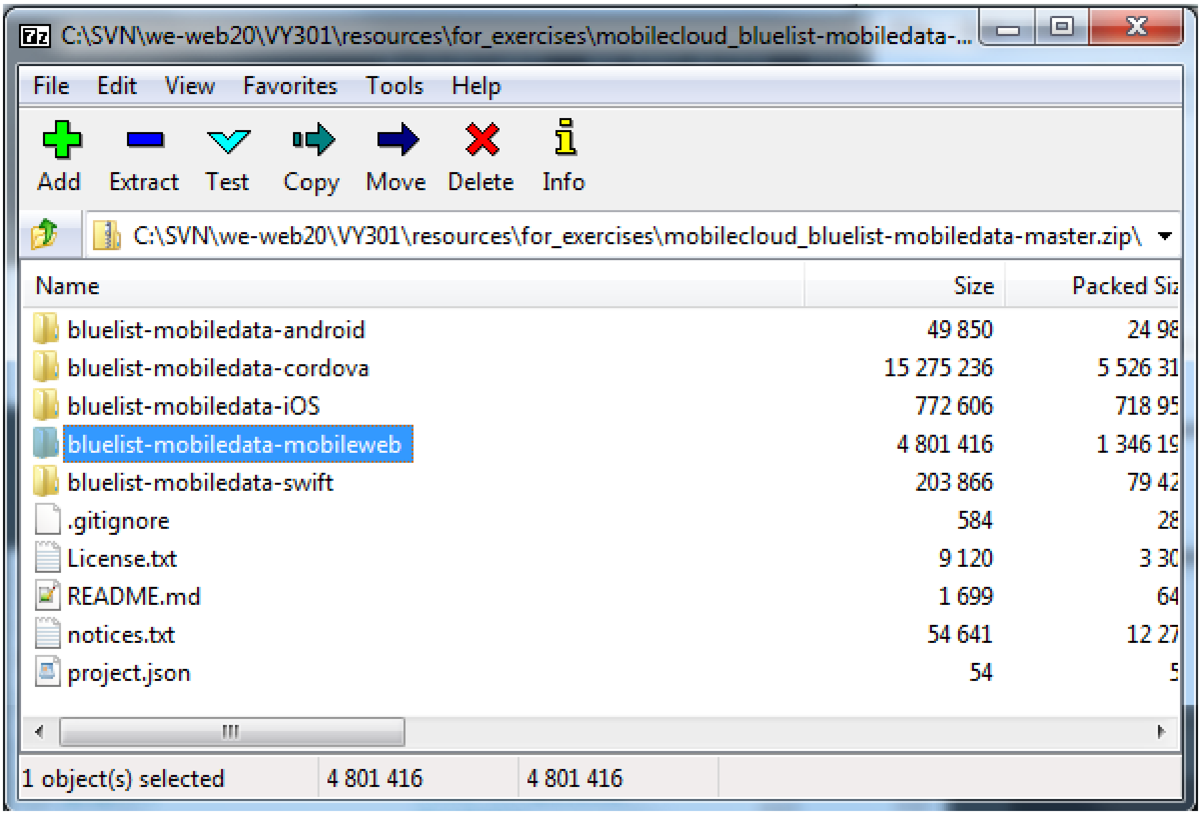
○ Open with Windows Explorer (default)

● Save File

□ Do this automatically for files like this from now on.

__ 3. Create a folder and extract the mobile web files.

__ a. Extract bluelist-mobiledata-mobileweb to the folder you created.

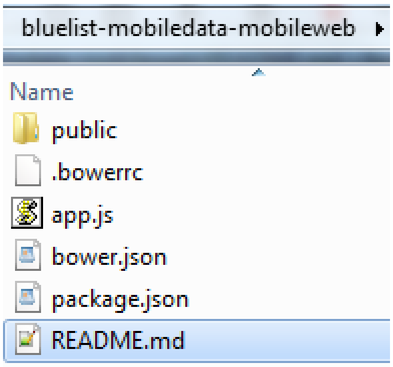

C:\SVN\we-web20\VY301\resources\for_exercises\mobilecloud_bluelist-mobiledata-...
File
Edit
View
Favorites
Tools
Help
Add
Extract
Test
Copy
Move
Delete
Info

C:\SVN\we-web20\VY301\resources\for_exercises\mobilecloud_bluelist-mobiledata-master.zip\

Name	Size	Packed Siz
bluelist-mobiledata-android	49 850	24 98
bluelist-mobiledata-cordova	15 275 236	5 526 31
bluelist-mobiledata-iOS	772 606	718 95
bluelist-mobiledata-mobileweb	4 801 416	1 346 19
bluelist-mobiledata-swift	203 866	79 42
.gitignore	584	28
License.txt	9 120	3 30
README.md	1 699	64
notices.txt	54 641	12 27
project.json	54	5

1 object(s) selected | 4 801 416 | 4 801 416

The files are extracted to the bluelist-mobiledata-mobileweb subfolder of your directory.



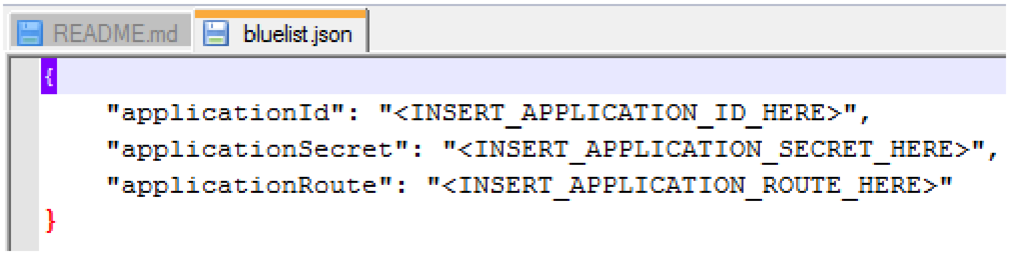
Name
- public
- .bowerrc
- app.js
- bower.json
- package.json
- README.md

__ 4. In the bluelist-mobiledata-mobileweb subfolder, open **README.md** in a text editor.

The README.md text file describes the steps that are needed to get the application to work on the local workstation.

__ 5. Configure the mobile web application to access the mobile data application on Bluemix.

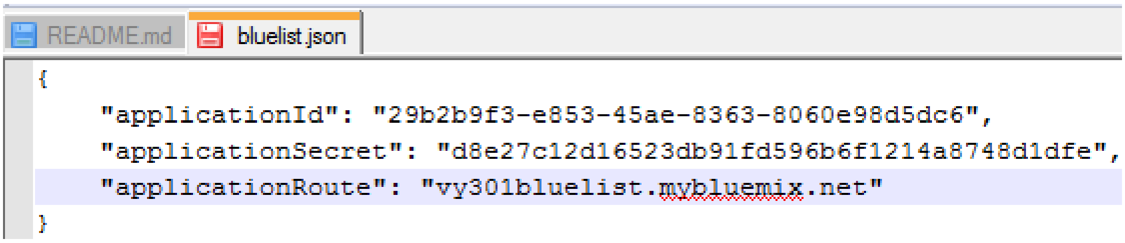
__ a. Modify the **bluelist.json** file in the **public** directory to include the application ID, application secret, and application route for the Bluemix mobile data application that you created in the earlier exercise.



```
{
    "applicationId": "<INSERT_APPLICATION_ID_HERE>",
    "applicationSecret": "<INSERT_APPLICATION_SECRET_HERE>",
    "applicationRoute": "<INSERT_APPLICATION_ROUTE_HERE>"
}
```

__ b. Save the changes.

Here is the example with completed values.



```
{
    "applicationId": "29b2b9f3-e853-45ae-8363-8060e98d5dc6",
    "applicationSecret": "d8e27c12d16523db91fd596b6f1214a8748d1dfe",
    "applicationRoute": "vy301bluelist.mybluemix.net"
}
```

Part 4: Install other application-dependent software

In this part of the exercise, you install some additional software that is needed to run the sample mobile web application on your workstation.

__ 1. Install the prerequisite software on your workstation.

__ a. Open a command prompt in Microsoft Windows.

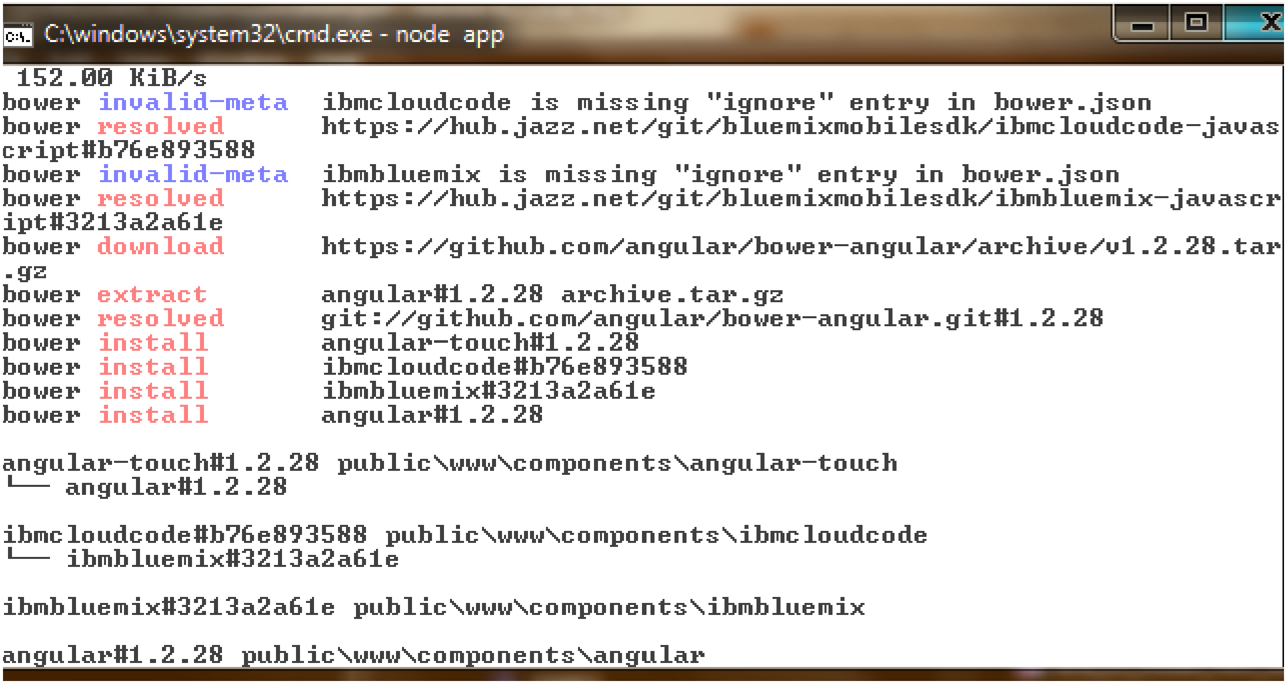
__ b. Change directory to the bluelist-mobiledata-mobileweb subfolder.

__ c. Ensure that the environment variables for Node.js are set up.

__ d. In the command prompt, type `npm install --production`
Because npm was installed by default with Node.js earlier, you are obtaining the latest version.

__ e. From the same directory, install the bower dependency manager.
Type `npm install -g bower`

The bower and angular software dependencies are installed.



```
 152.00 KiB/s
bower invalid-meta  ibmcloudcode is missing "ignore" entry in bower.json
bower resolved      https://hub.jazz.net/git/bluemixmobilesdk/ibmcloudcode-javas
cript#b76e893588
bower invalid-meta  ibmbluemix is missing "ignore" entry in bower.json
bower resolved      https://hub.jazz.net/git/bluemixmobilesdk/ibmbluemix-javascr
ipt#3213a2a61e
bower download      https://github.com/angular/bower-angular/archive/v1.2.28.tar
.gz
bower extract       angular#1.2.28 archive.tar.gz
bower resolved      git://github.com/angular/bower-angular.git#1.2.28
bower install       angular-touch#1.2.28
bower install       ibmcloudcode#b76e893588
bower install       ibmbluemix#3213a2a61e
bower install       angular#1.2.28

angular-touch#1.2.28 public\www\components\angular-touch
└── angular#1.2.28

ibmcloudcode#b76e893588 public\www\components\ibmcloudcode
└── ibmbluemix#3213a2a61e

ibmbluemix#3213a2a61e public\www\components\ibmbluemix

angular#1.2.28 public\www\components\angular
```

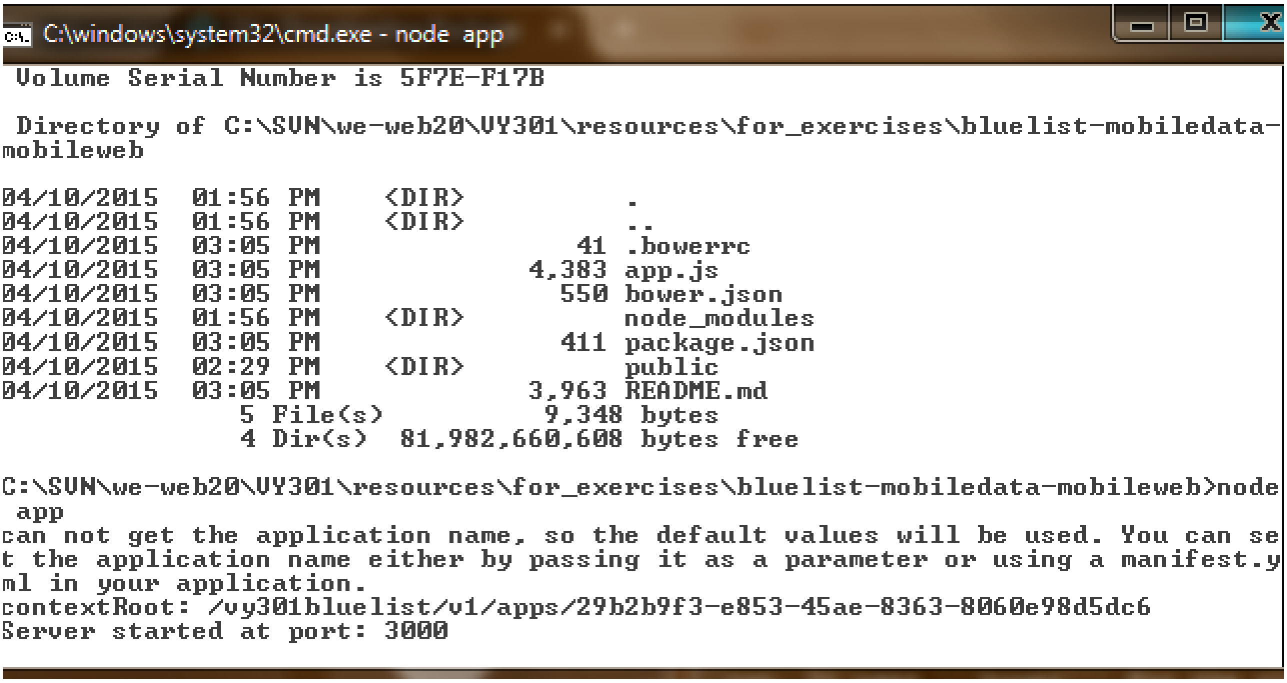
Part 5: Run the mobile web application on the workstation

In this part, you run the Node.js application on the workstation.

__ 1. Run the Node application on the workstation.

__ a. From the command prompt, in the bluelist-mobiledata-mobileweb directory, type `node app`

The command prompt displays a message that the application is started.

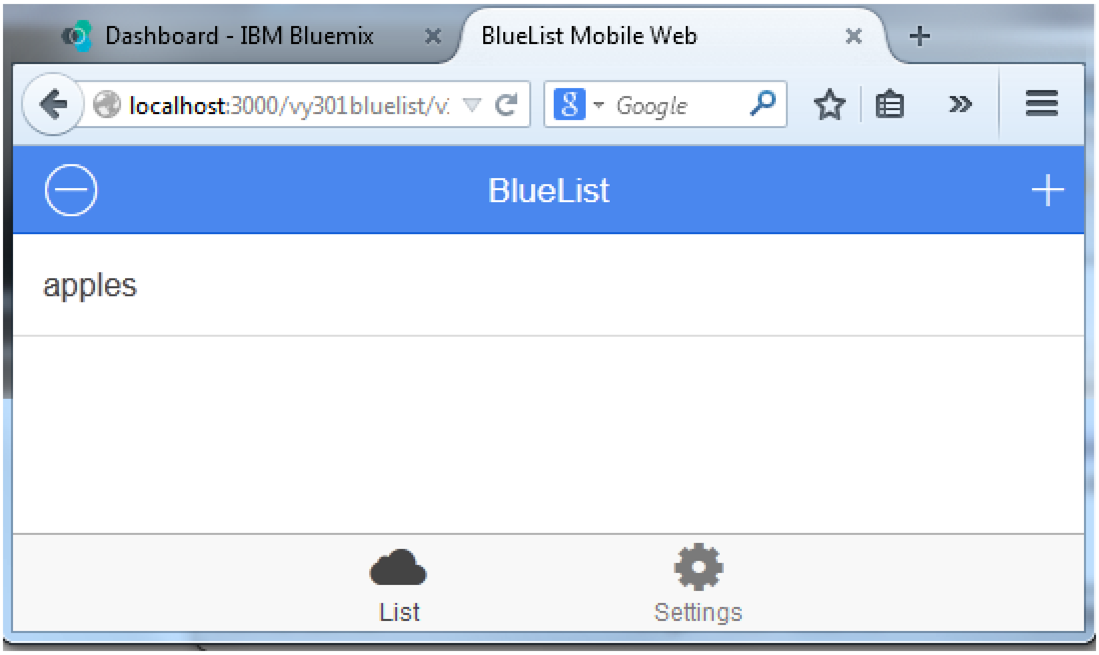


```
 Volume Serial Number is 5F7E-F17B

 Directory of C:\SVN\we-web20\VY301\resources\for_exercises\bluelist-mobiledata-
mobileweb

04/10/2015  01:56 PM    <DIR>          .
04/10/2015  01:56 PM    <DIR>          ..
04/10/2015  03:05 PM                41 .bowerrc
04/10/2015  03:05 PM             4,383 app.js
04/10/2015  03:05 PM               550 bower.json
04/10/2015  01:56 PM    <DIR>          node_modules
04/10/2015  03:05 PM               411 package.json
04/10/2015  02:29 PM    <DIR>          public
04/10/2015  03:05 PM             3,963 README.md
               5 File(s)          9,348 bytes
               4 Dir(s)  81,982,660,608 bytes free

C:\SVN\we-web20\VY301\resources\for_exercises\bluelist-mobiledata-mobileweb>node
 app
can not get the application name, so the default values will be used. You can se
t the application name either by passing it as a parameter or using a manifest.y
ml in your application.
contextRoot: /vy301bluelist/v1/apps/29b2b9f3-e853-45ae-8363-8060e98d5dc6
Server started at port: 3000
```

__ 2. View the mobile web client in the browser.

__ a. Open a browser session, then type `http://localhost:3000`

The mobile web client is displayed in the browser.



apples

The data in the list corresponds to the data that you typed in the earlier exercise.

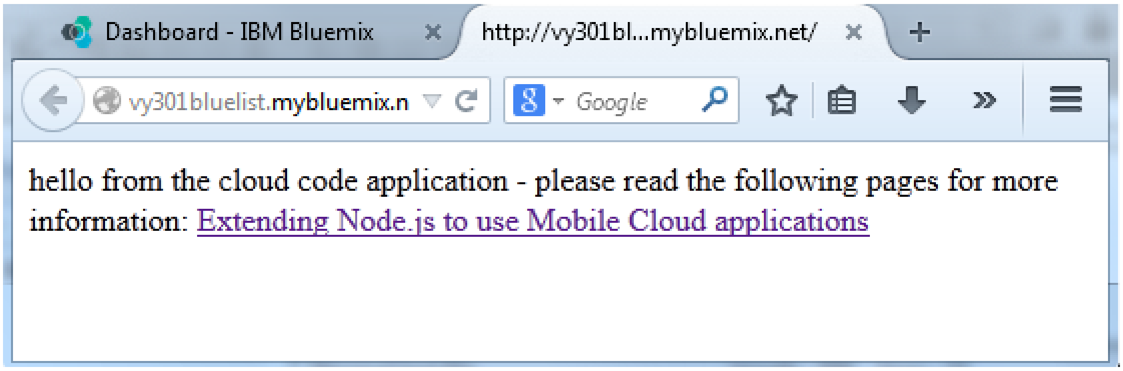
Part 6: Deploy the mobile web application to Bluemix

In this part, you deploy the application to Bluemix so that the mobile web client can be run from Bluemix. The Cloud Foundry command-line option is already installed on your workstation in an earlier exercise.

__ 1. View the web page for the Bluemix application.

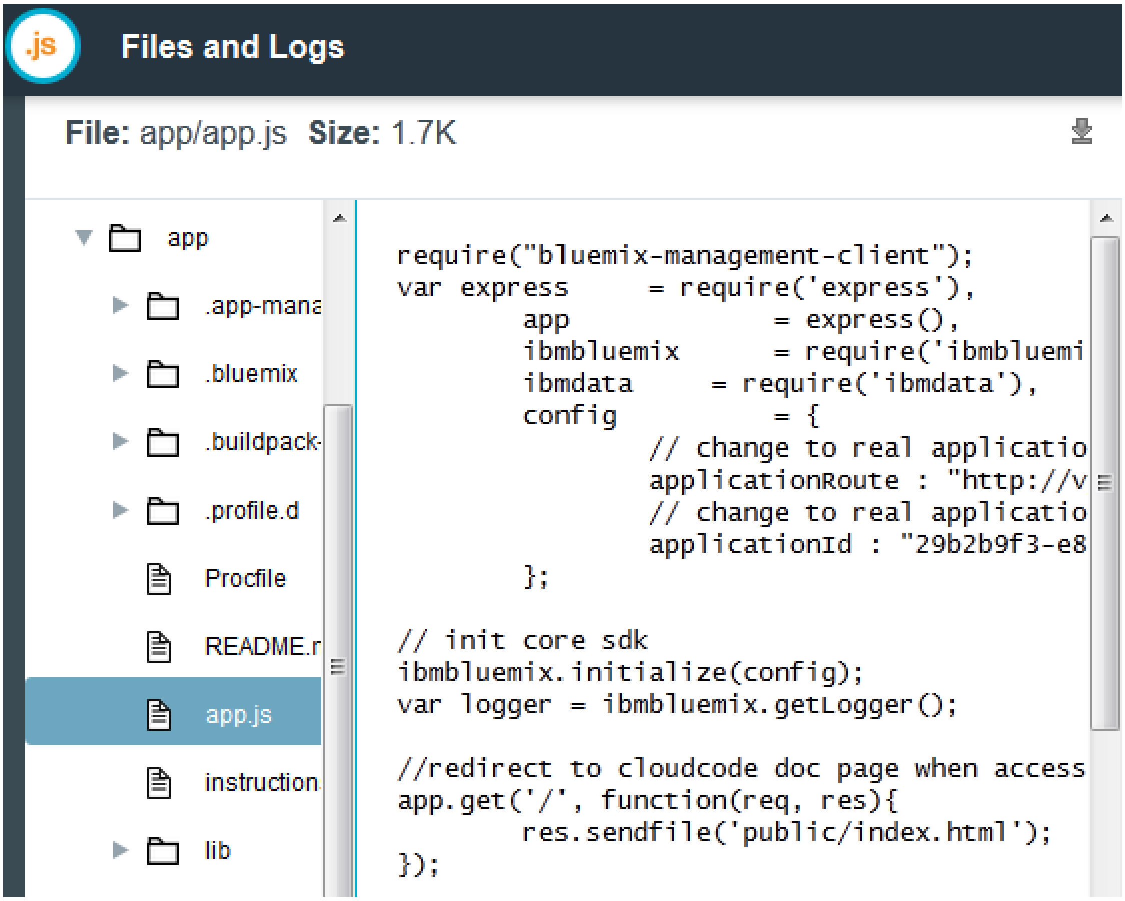
__ a. Open another browser tab with the URL that you used to replace the value of <INSERT_APPLICATION_ROUTE_HERE> earlier in the exercise.

The browser displays the web page for the Bluemix Mobile Data application.



hello from the cloud code application - please read the following pages for more information: Extending Node.js to use Mobile Cloud applications

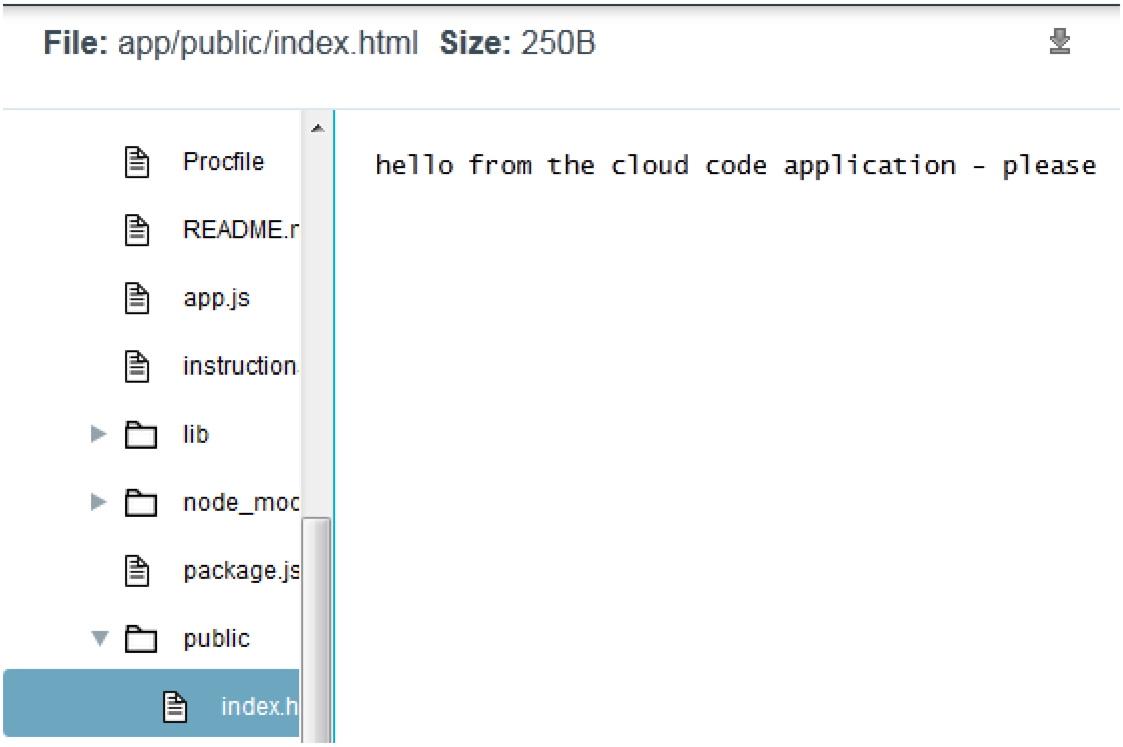
__ b. The app.js on Bluemix specifies what the default Mobile Data application displays in the web browser. You can view the app.js file under Files and Logs on Bluemix.



File: app/app.js **Size:** 1.7K

```
require("bluemix-management-client");
var express     = require('express'),
        app             = express(),
        ibmbluemix      = require('ibmbluemi
        ibmdata     = require('ibmdata'),
        config          = {
                // change to real applicatio
                applicationRoute : "http://v
                // change to real applicatio
                applicationId : "29b2b9f3-e8
        };

// init core sdk
ibmbluemix.initialize(config);
var logger = ibmbluemix.getLogger();

//redirect to cloudcode doc page when access
app.get('/', function(req, res){
        res.sendfile('public/index.html');
});
```

The existing `public/index.html` file displays the hello from the cloud text that you saw displayed earlier in the browser.



File: app/public/index.html Size: 250B

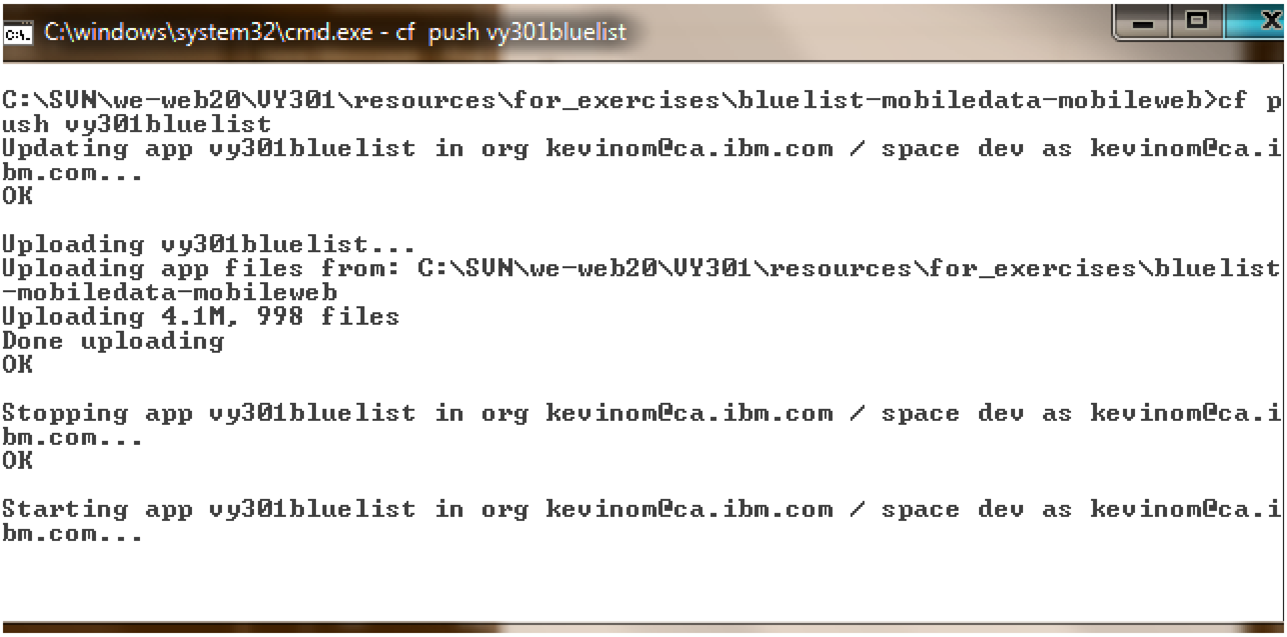
```
hello from the cloud code application - please
```

__ c. Next, you push a new app.js file from the workstation that displays the mobile web page as the default page for the bluelist Mobile Data application.

__ 2. Push the mobile web application to Bluemix.

__ a. From the command prompt, in the bluelist-mobiledata-mobileweb directory, type `cf push ${yourAppName}`

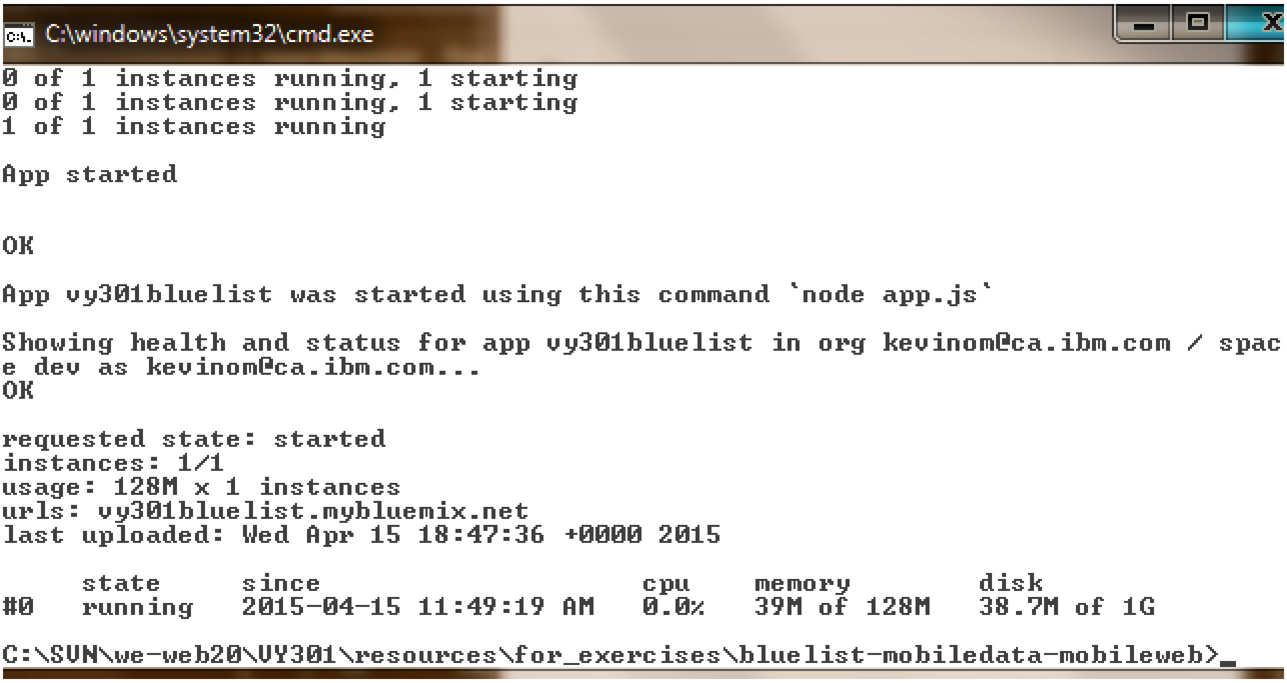
__ b. In this case, replace `${yourAppName}` with the unique name that you gave for the Mobile Data application when you created the application from the boilerplate on Bluemix.



```
C:\SVN\we-web20\VY301\resources\for_exercises\bluelist-mobiledata-mobileweb>cf p
ush vy301bluelist
Updating app vy301bluelist in org kevinom@ca.ibm.com / space dev as kevinom@ca.i
bm.com...
OK

Uploading vy301bluelist...
Uploading app files from: C:\SVN\we-web20\VY301\resources\for_exercises\bluelist
-mobiledata-mobileweb
Uploading 4.1M, 998 files
Done uploading
OK

Stopping app vy301bluelist in org kevinom@ca.ibm.com / space dev as kevinom@ca.i
bm.com...
OK

Starting app vy301bluelist in org kevinom@ca.ibm.com / space dev as kevinom@ca.i
bm.com...
```

The application is uploaded to Bluemix. The existing Bluemix application is stopped and restarted.

The instance of the Bluemix Mobile Data application is started.

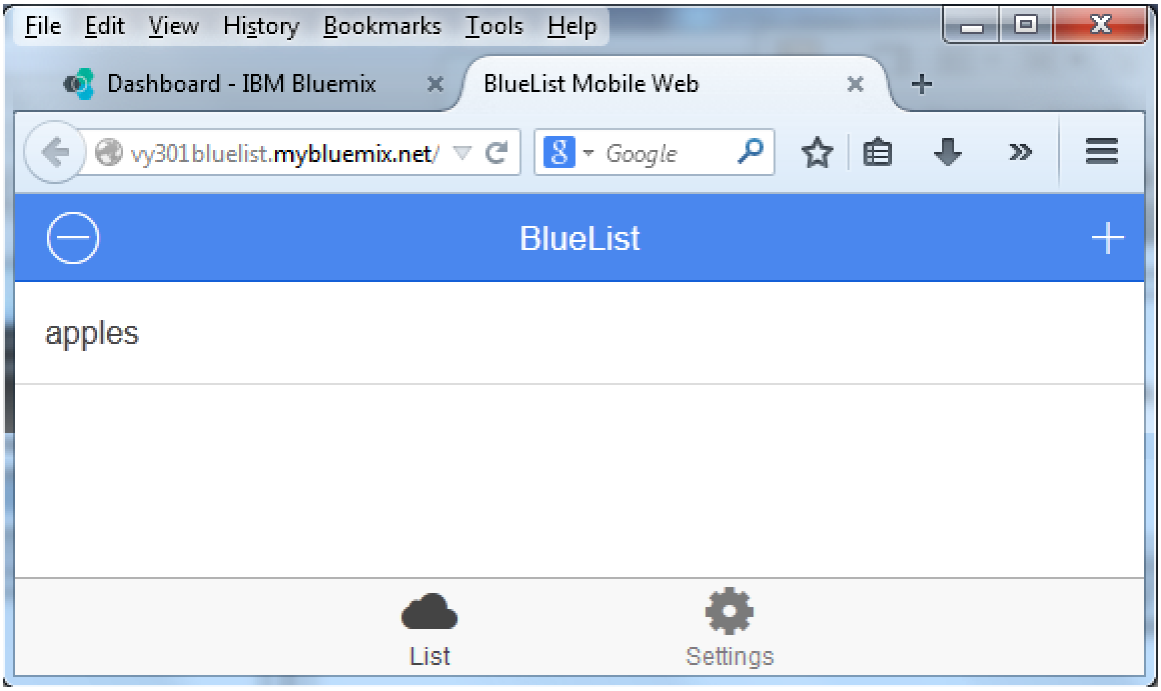


```
0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
1 of 1 instances running

App started


OK

App vy301bluelist was started using this command `node app.js`

Showing health and status for app vy301bluelist in org kevinom@ca.ibm.com / spac
e dev as kevinom@ca.ibm.com...
OK

requested state: started
instances: 1/1
usage: 128M x 1 instances
urls: vy301bluelist.mybluemix.net
last uploaded: Wed Apr 15 18:47:36 +0000 2015

     state     since                    cpu    memory         disk
#0   running   2015-04-15 11:49:19 AM   0.0%   39M of 128M    38.7M of 1G

C:\SVN\we-web20\VY301\resources\for_exercises\bluelist-mobiledata-mobileweb>_
```

__ 3. Review the default web page in the browser.

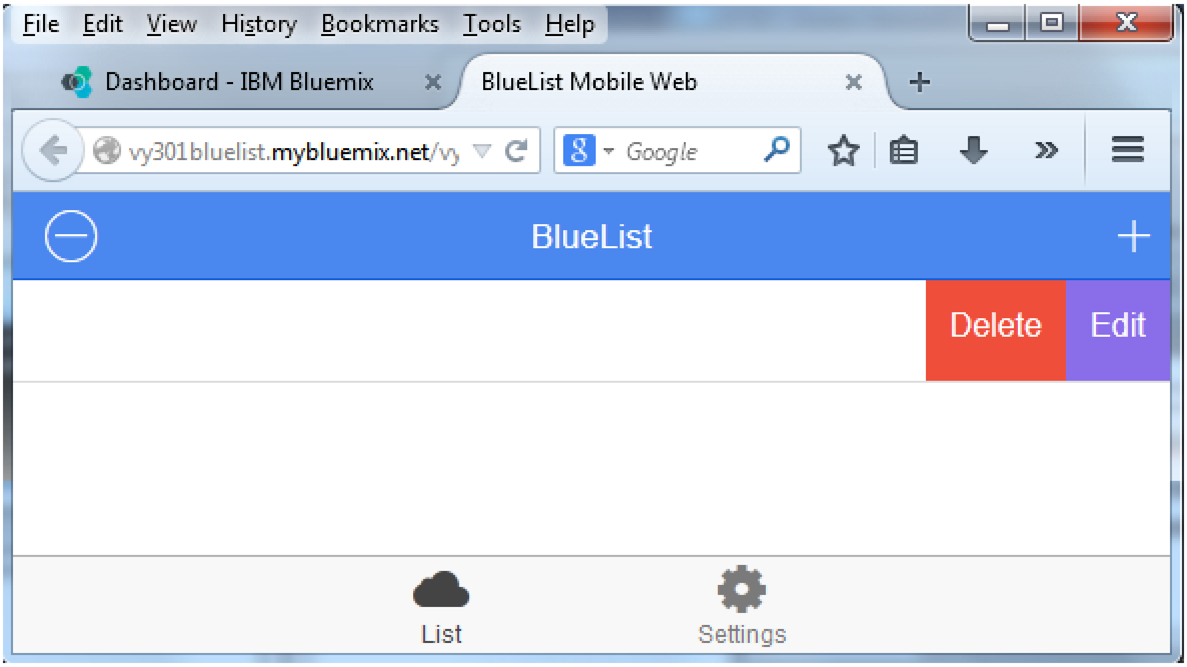
__ a. In your web browser, type the URL for the route address for the Bluemix Mobile Data application.

You now see the web page for the bluelist-mobile-web application.

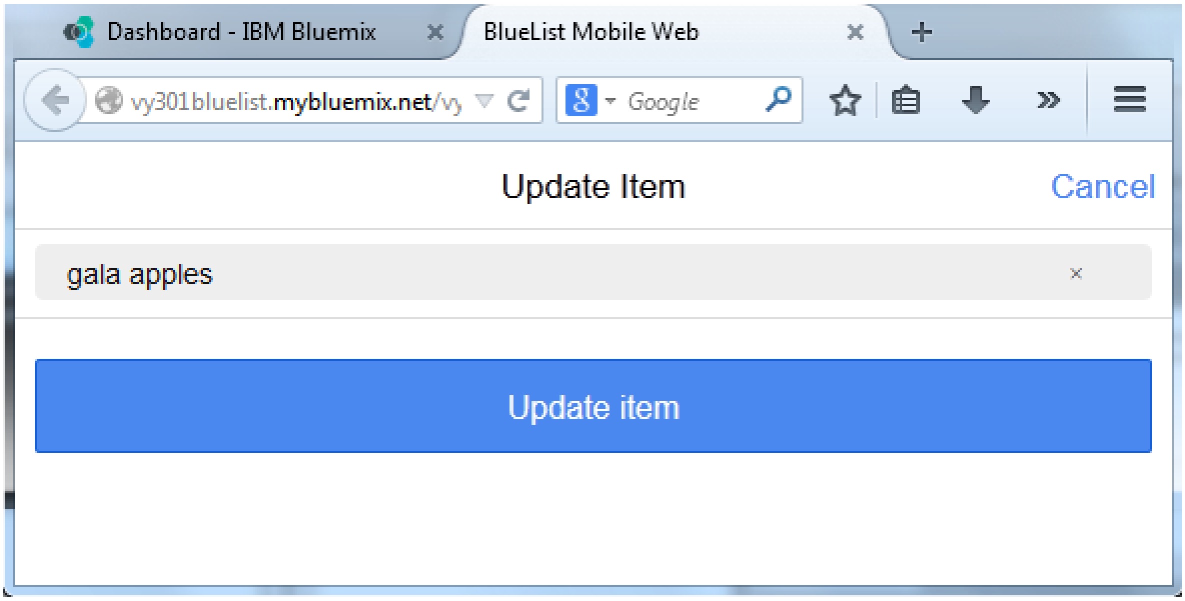


The landing page for the Bluemix Mobile Data application is changed to the bluelist web page.

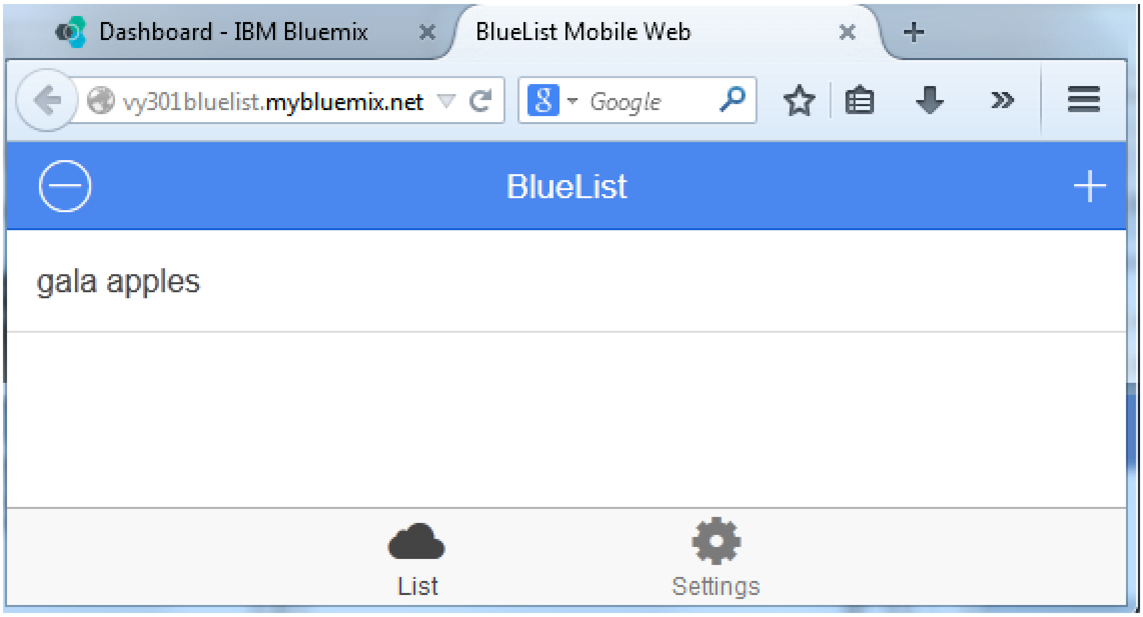
__ b. You can modify the items on the page by dragging the item to the left. The page displays the options to delete or edit the item.



__ c. If you click **Edit**, you can change the value in the list by typing a new value and clicking the **Update Item** button.



__ d. Swipe the row with **Delete** and **Edit** to the right to display the updated list.



End of exercise

Exercise review and wrap-up

In the exercise, you download the bluelist-mobiledata-mobile application from Bluemix and you configure it to run locally on the workstation. The locally installed mobile web application communicates with the IBM Bluemix cloud to run Bluemix and data services.

Then, you push the application to Bluemix. The mobile web application replaces the landing page for the Bluemix Mobile Data application with the bluelist web page. Now the web client and the application both run on the IBM Bluemix cloud.

Finally, you test that the application works correctly on the IBM Bluemix cloud.