IBM Single Sign On service for Bluemix – Build your first Java application using external authentication.

12/16/2015

IBM EcoDev

Contents

Introduction

When developing an application supporting multiple users, one of the first decisions you will need to face is how to authenticate each users. It might seem like a simple approach to create a database table and populate it with credentials, but this quickly creates further challenges. For one, how will your application be able to integrate with existing applications and their authentication services? This lab will introduce the IBM Single Sign On (SSO) service for Bluemix which you can use to configure your web application to authenticate users from social media sites, enterprise directories via SAML federation, and also a ldap-based cloud identity source.

This lab will walk through the steps to add and configure the IBM Single Sign On service for Bluemix to the java starter application. It will provide step-by-step instructions to create the application, add the service to the Bluemix workspace, configure a cloud identity source, bind and integrate the service into the application, and modify the application code to use external authentication.

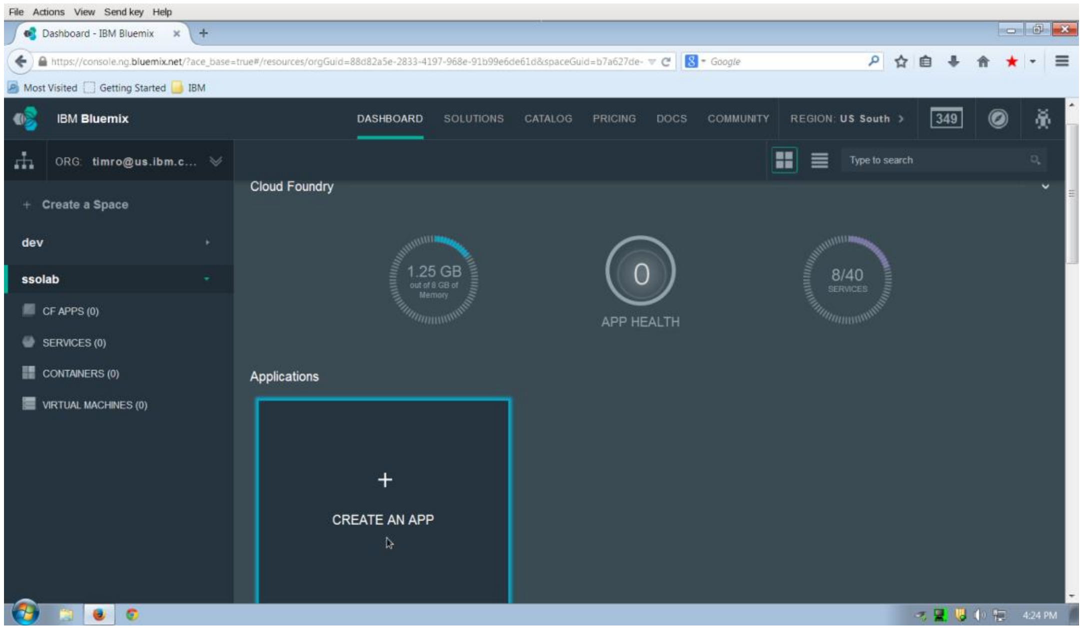
The lab requires a workstation (Windows, Mac OS X or Linux are all fine) with the following items installed:

- bluemix.net account (sign up for a 30-day free trial)
- Java JRE 7.1 or higher (www.java.com/download)
- Multiple web browsers (it's most convenient to use one browser for working with the Bluemix and Single Sign On control panel in one and a separate browser for testing your application, recent versions of Firefox, Chrome, Opera, Safari, Internet Explorer)
- Eclipse mars or later (www.eclipse.org/downloads)
- Cloud Foundry CLI tool version 6.9 or later – there is a step in the application setup where a link is provided for installation if it is not already on your workstation.

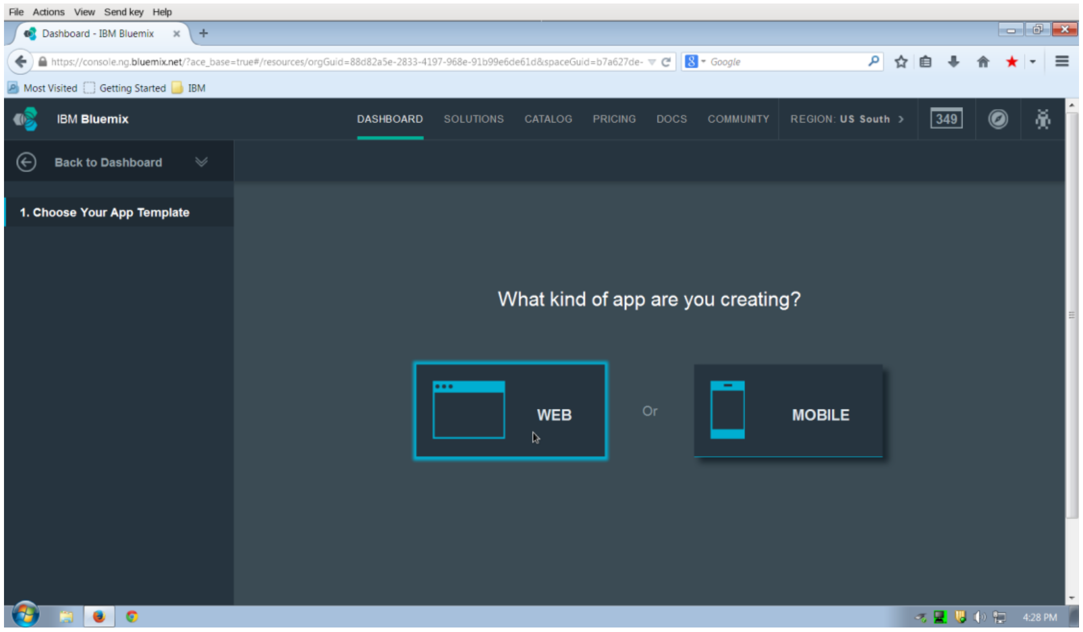
Now let’s get started.

Create Java starter application and add Single Sign On service to dashboard:

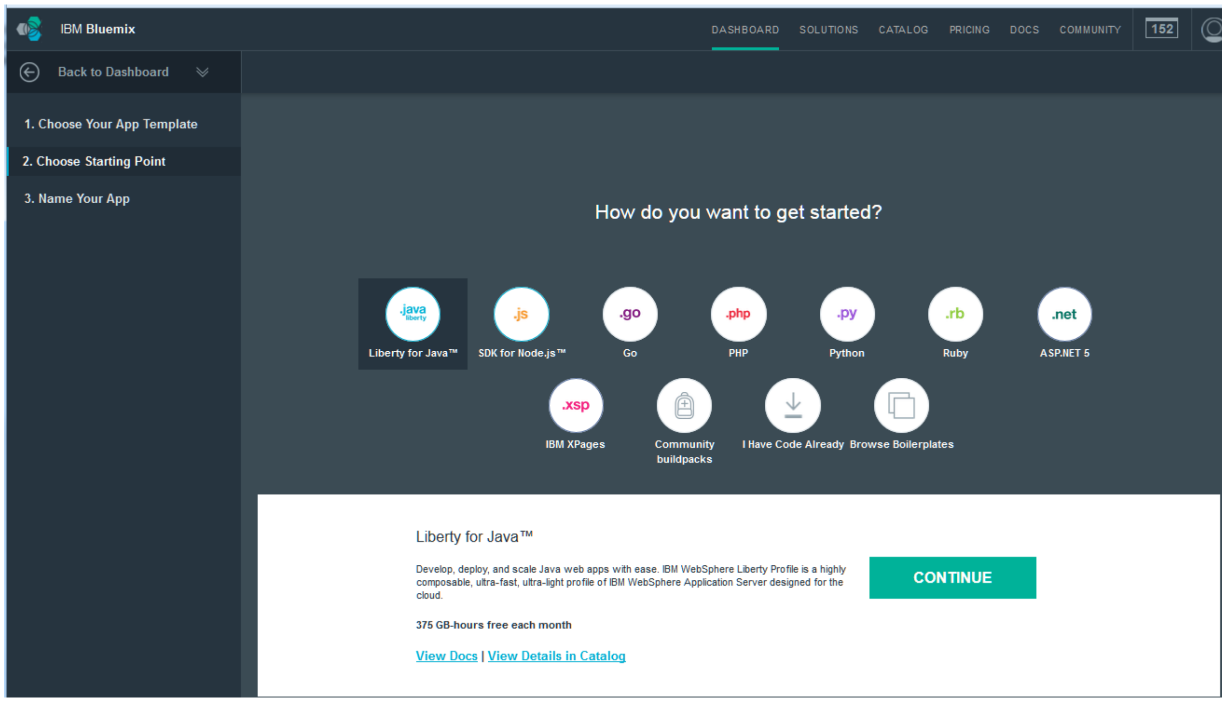
Sign in to Bluemix and go to the Dashboard. Under Cloud Foundry, click on the "CREATE AN APP" button to get started:



Then select Web app:



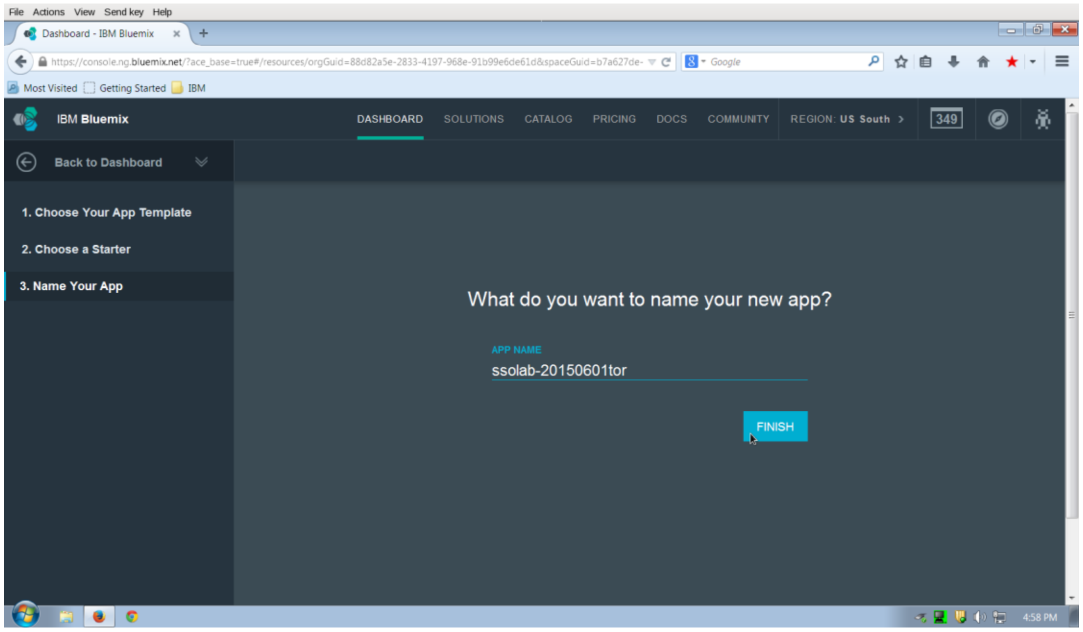
Next, you will need to select the runtime for the application. For this lab select the Liberty for Java SDK, and confirm by clicking on CONTINUE:



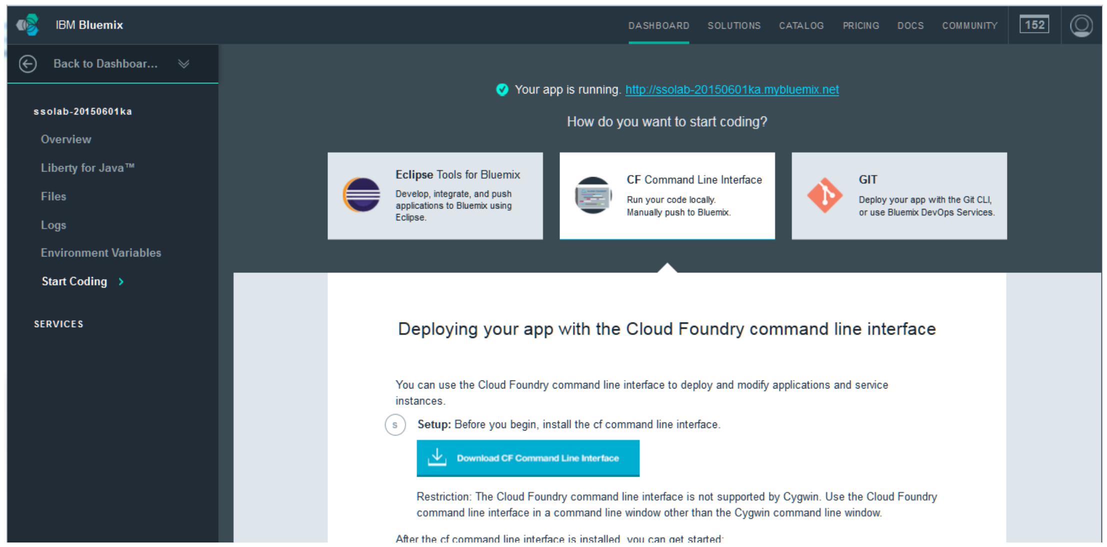
Develop, deploy, and scale Java web apps with ease. IBM WebSphere Liberty Profile is a highly composable, ultra-fast, ultra-light profile of IBM WebSphere Application Server designed for the cloud.

375 GB-hours free each month

To complete adding the new application, provide a name. Each application in Bluemix public requires a unique name. One easy way to do this for a test app is to append the date and your initials to the name, and then select FINISH:



Bluemix will bring up the application's Start Coding panel with a message at the top about the application starting the staging process. If you do not already have the Cloud Foundry (cf) CLI tool installed on your workstation, click on the link shown to go to GitHub and download the current version for your workstation (it will be later than 6.9 but that is ok). After the download is completed, install the cf tool and then proceed to the next step.



Your app is running. http://ssolab-20150601ka.mybluemix.net

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.

Deploying your app with the Cloud Foundry command line interface

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

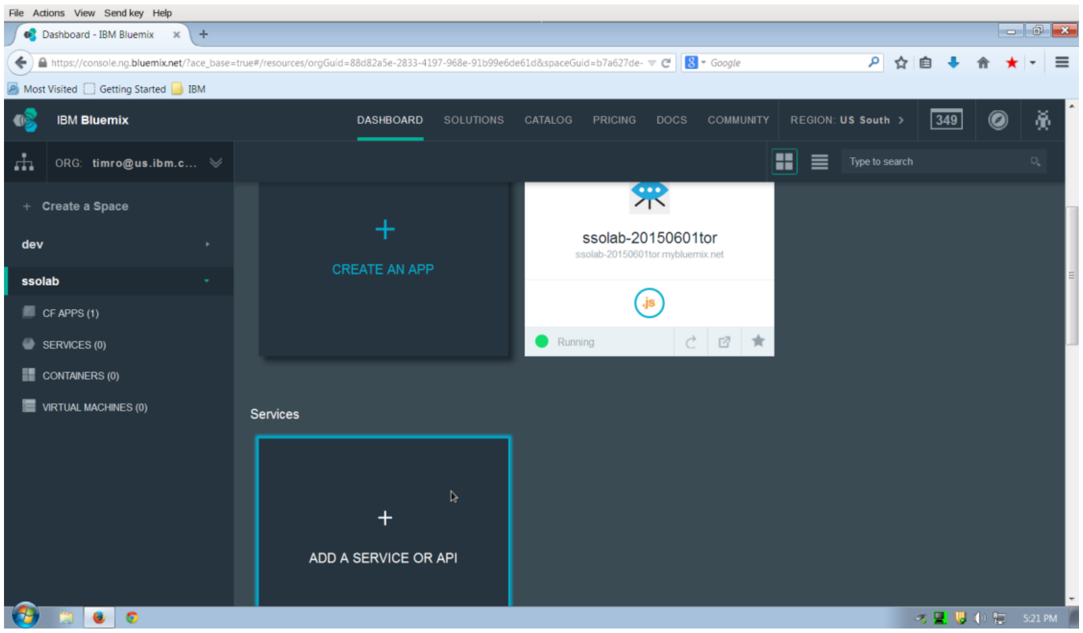
Setup: Before you begin, install the cf command line interface.

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.

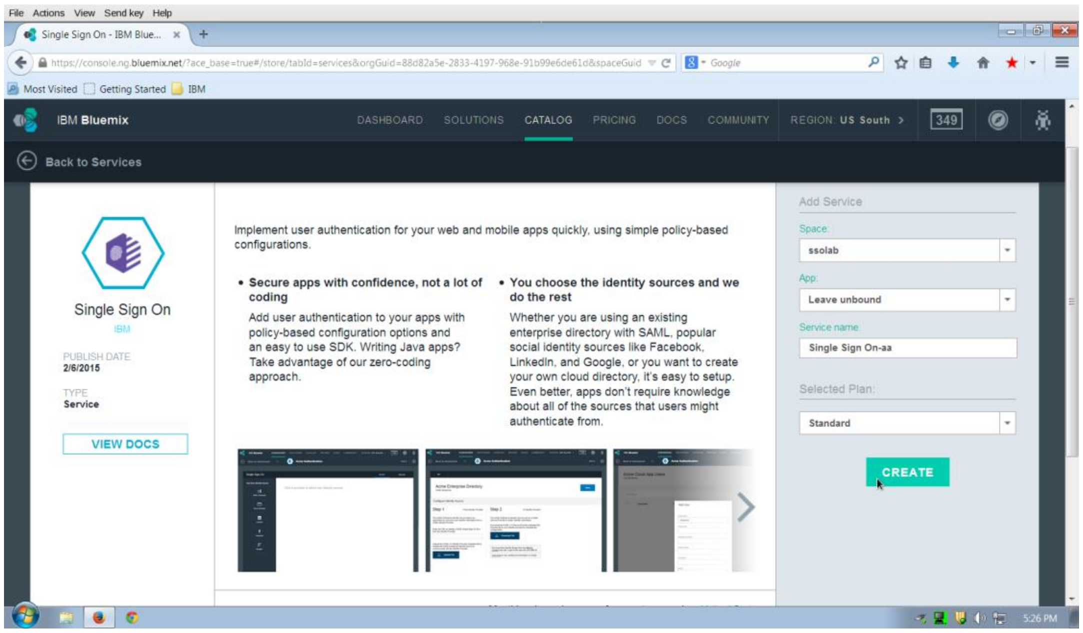
After the cf command line interface is installed, you can get started:

Next, we will add the Single Sign On service to the Dashboard. Before proceeding, you will need to select an id that will become the prefix of the URL for the service when used when applications redirect users to the service for selecting an identity provider, and also for handling callbacks from identity providers. This id can be up to 32 characters, and must start with an alphanumeric character. Make a note of it here if you like: ____

Then, go back to the Bluemix Cloud Foundry Dashboard and select Add a Service or API:



Scroll down in the services list to the Security section and then click on the Single Sign On icon, this will bring up a configuration panel for adding the service. On the right hand side, select Leave unbound under the App: section (if you entered from the application then the application is automatically listed in the App: section). You can change the Service name: that will be shown in the dashboard or leave it at


File Actions View Send key Help
Single Sign On - IBM Blue...
https://console.ng.bluemix.net/?ace_base=true#/store/tabId=services&orgGuid=88d82a5e-2833-4197-968e-91b99e6de61d&spaceGuid
Most Visited Getting Started IBM
IBM Bluemix
DASHBOARD SOLUTIONS CATALOG PRICING DOCS COMMUNITY REGION: US South
349
Back to Services
VIEW DOCS
CREATE
5:26 PM

Single Sign On
IBM
PUBLISH DATE
2/6/2015
TYPE
Service

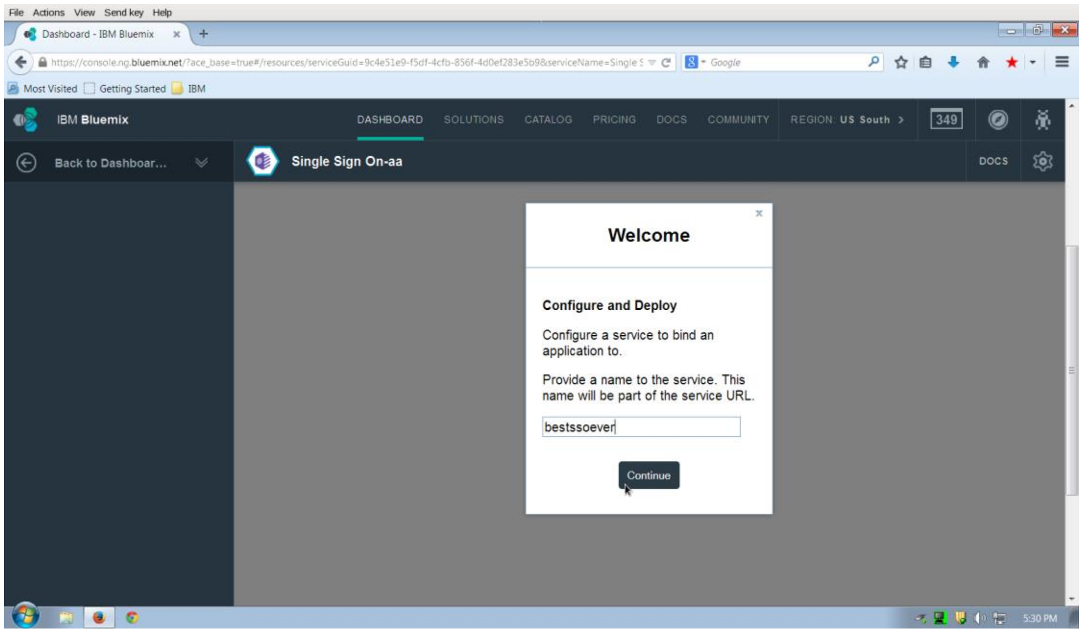
Implement user authentication for your web and mobile apps quickly, using simple policy-based configurations.

- **Secure apps with confidence, not a lot of coding**
  Add user authentication to your apps with policy-based configuration options and an easy to use SDK. Writing Java apps? Take advantage of our zero-coding approach.
- **You choose the identity sources and we do the rest**
  Whether you are using an existing enterprise directory with SAML, popular social identity sources like Facebook, LinkedIn, and Google, or you want to create your own cloud directory, it's easy to setup. Even better, apps don't require knowledge about all of the sources that users might authenticate from.

Add Service
Space: ssolab
App: Leave unbound
Service name: Single Sign On-aa
Selected Plan: Standard

the default. Then click on the CREATE button:

A panel will display to prompt for a “name” for the service. This is the id that you chose earlier, it is not the name that was shown in the previous panel as the Service name. Once provided, click on the Continue button:


Welcome

Configure and Deploy

Configure a service to bind an application to.

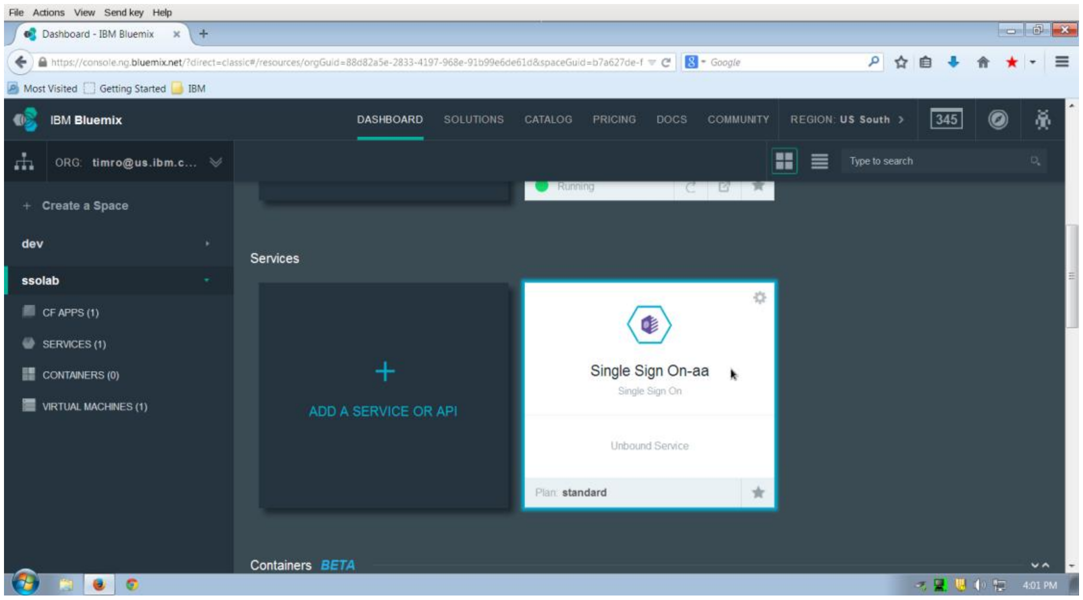
Provide a name to the service. This name will be part of the service URL.

bestssoever

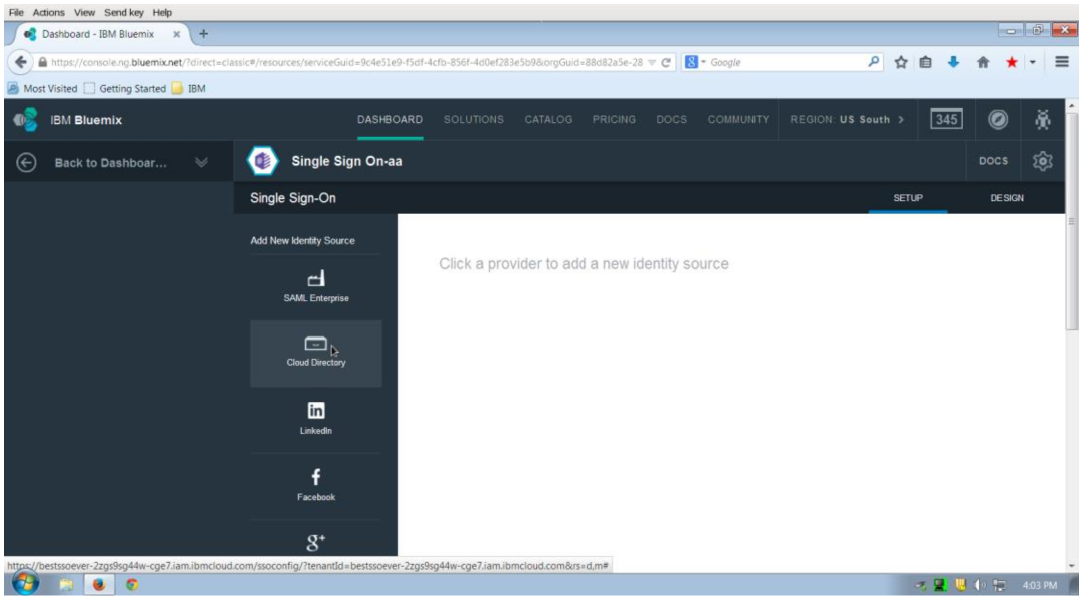
This will add the service to the dashboard and now we will be able to create a simple Cloud directory and add a testing user for external authentication in the next section of the lab.

Configure a Cloud directory identity source

Back in the IBM Bluemix Cloud Foundry dashboard, scroll down to find the SSO service that you just created. Click on the service to open up the configuration panel:

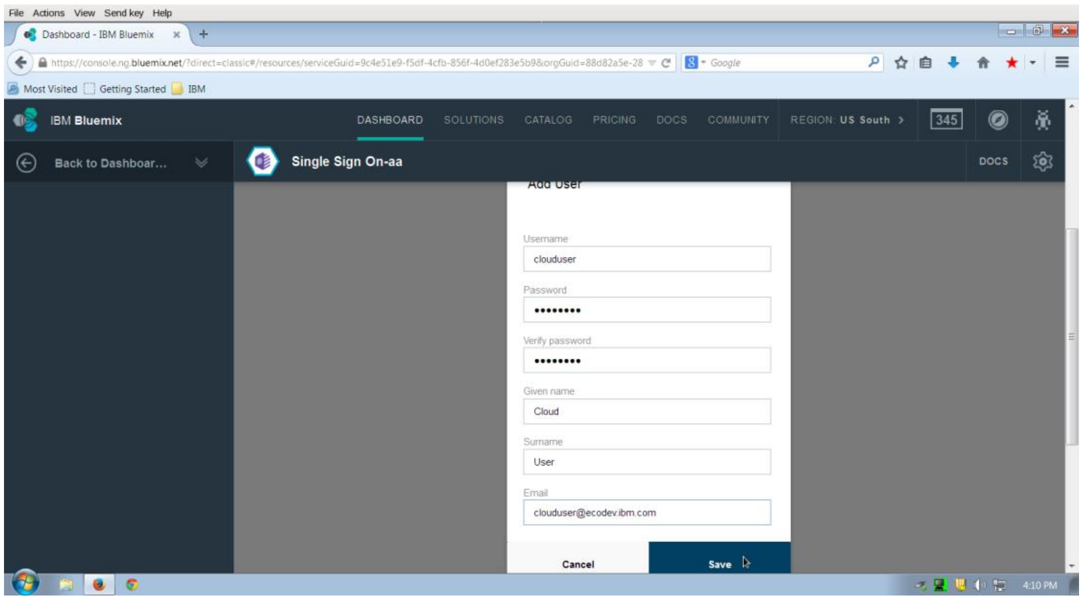


In the configuration panel, click on the Cloud Directory icon to add and configure a directory in the cloud for our lab exercises:



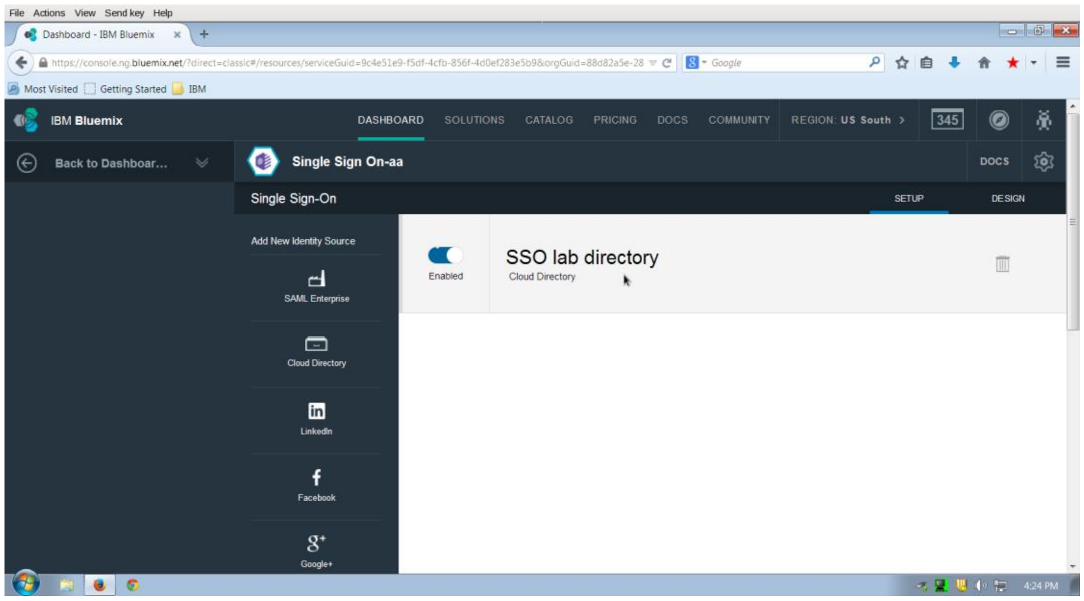
In the next screen, you may give the identity source a new name other than the default and see the current list of users. It's empty at first so click on the add icon:

And put in some information for a test user. This can be any user information that you prefer, however make a note of the username and password set since that will be used when testing the external authentication service:

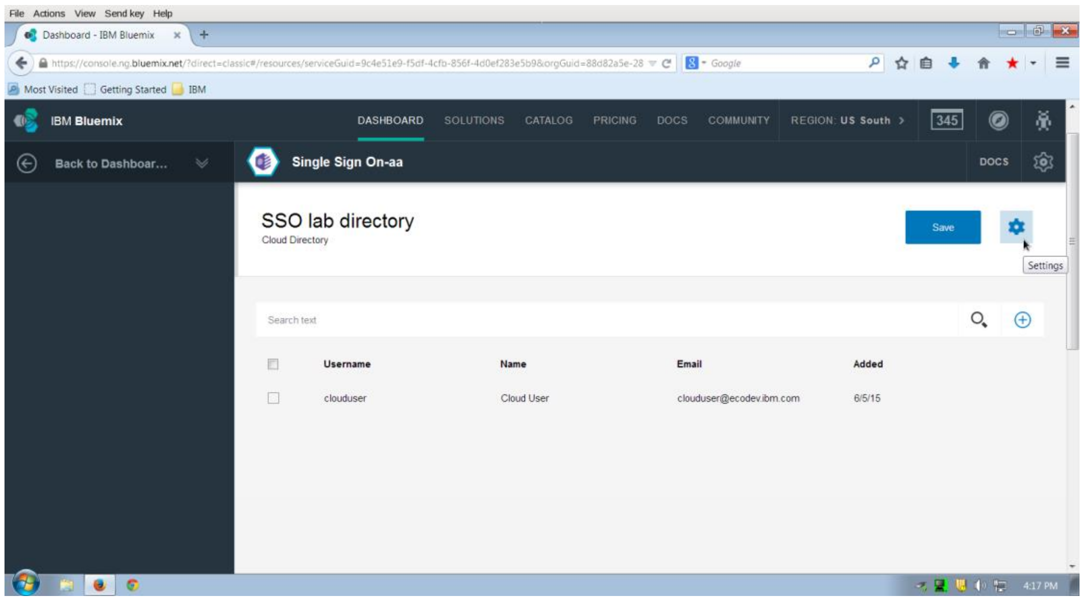


Username: clouduser
Password: ••••••••
Verify password: ••••••••
Given name: Cloud
Surname: User
Email: clouduser@ecodev.ibm.com

Save the user by clicking on the Save button, click on the Save button for the identity source configuration panel, and then re-open the service by selecting it:



After re-opening the identity source configuration panel, a settings icon button will be shown in the upper right section of the panel. To manage the password policy applied to users in this identity source, you can click on the settings icon button for the identity source:



Username	Name	Email	Added
clouduser	Cloud User	clouduser@ecodev.ibm.com	6/5/15

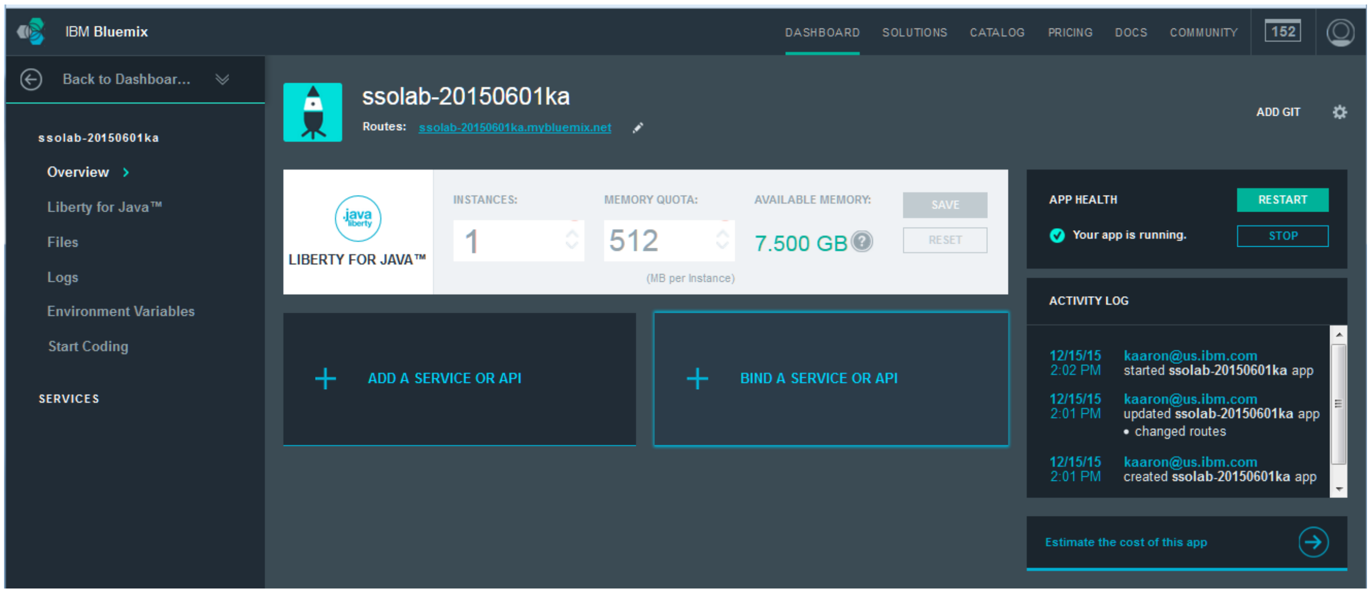
The settings control has two option panels. The Auto Consent option tells the SSO service if it should prompt the user after an authentication to allow the SSO service to retrieve personal (e.g. name and e-mail address) information from the identity source. Leave this setting to On for this lab to observe this prompt in action. The Password Policy option allows you to select various levels of password quality and lifetime. At the present time, these policies are fixed and you may select from one of the three. Choose Medium, and then click on Save.

Lastly, click on save in the identity source configuration panel to return to the main SSO configuration panel

The name shown for the identity source will be the name you selected or the default name of Cloud Directory. The toggle (shown as Enabled) next to the name allows the service administrator to enable or disable a particular identity source for use by applications. Leave the toggle at Enabled and click on the Back to Dashboard link in the upper left to return to the IBM Bluemix Cloud Foundry dashboard.

Bind the Single Sign On service to the application

To bind the Single Sign On service with the configured identity source, first click on the application icon in the dashboard to bring up the application overview page. Then click on the Bind a Service or API button:

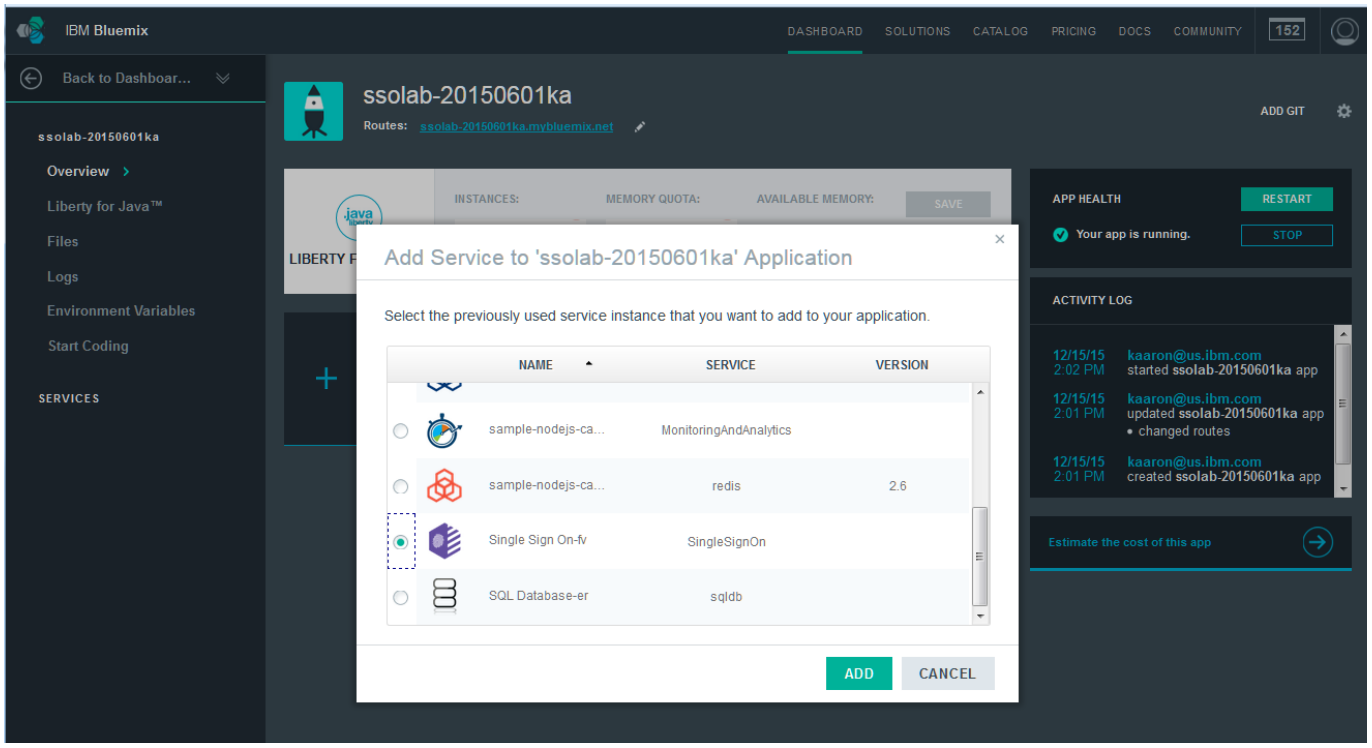


12/15/15 2:02 PM kaaron@us.ibm.com started ssolab-20150601ka app

12/15/15 2:01 PM kaaron@us.ibm.com updated ssolab-20150601ka app
• changed routes

12/15/15 2:01 PM kaaron@us.ibm.com created ssolab-20150601ka app

A panel will be displayed, select the radio button next to the Single Sign On service and then click on the ADD button:



ssolab-20150601ka

Routes: ssolab-20150601ka.mybluemix.net

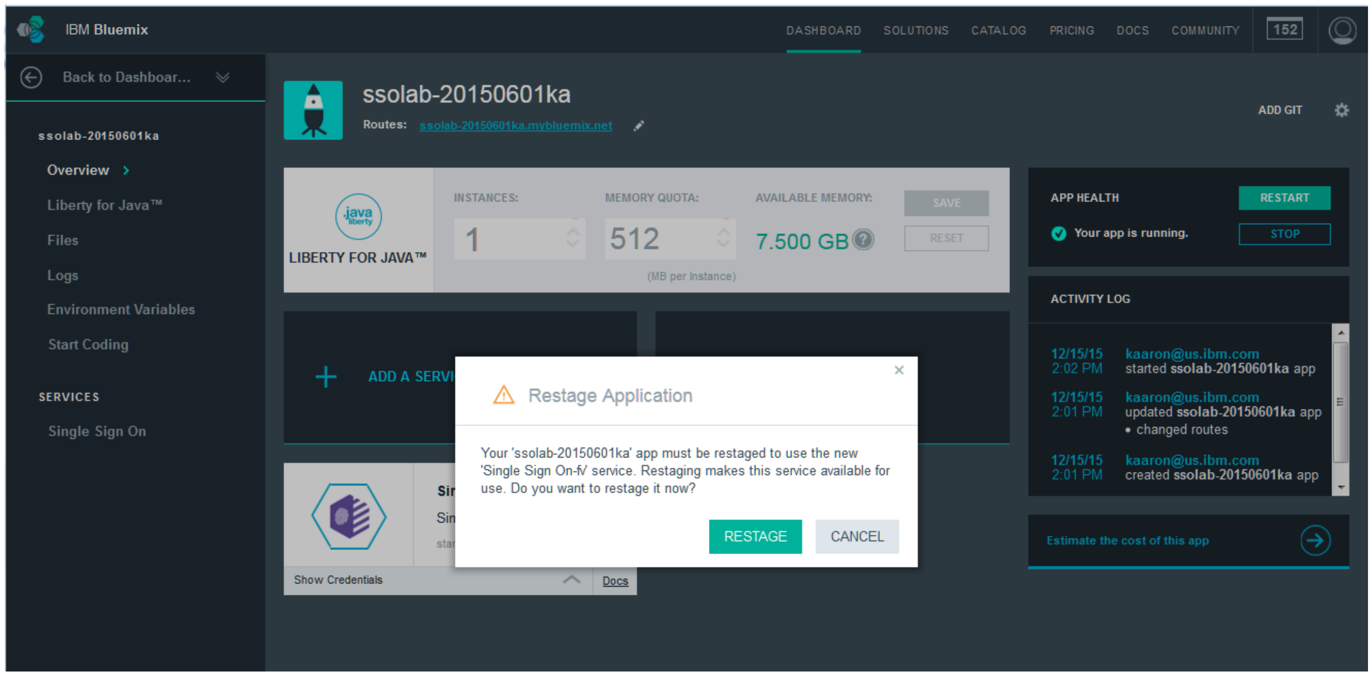
Add Service to 'ssolab-20150601ka' Application

Select the previously used service instance that you want to add to your application.

NAME	SERVICE	VERSION
sample-nodejs-ca...	MonitoringAndAnalytics	
sample-nodejs-ca...	redis	2.6
● Single Sign On-fv	SingleSignOn	
SQL Database-er	sqldb	

12/15/15 2:02 PM kaaron@us.ibm.com started ssolab-20150601ka app
12/15/15 2:01 PM kaaron@us.ibm.com updated ssolab-20150601ka app • changed routes
12/15/15 2:01 PM kaaron@us.ibm.com created ssolab-20150601ka app

You will be prompted to restage the application. A restage is used to add the VCAP_SERVICES environment variables to the application so that it will be able to access the Single Sign On service bound to the application. Click on the RESTAGE button:



ssolab-20150601ka

Routes: ssolab-20150601ka.mybluemix.net

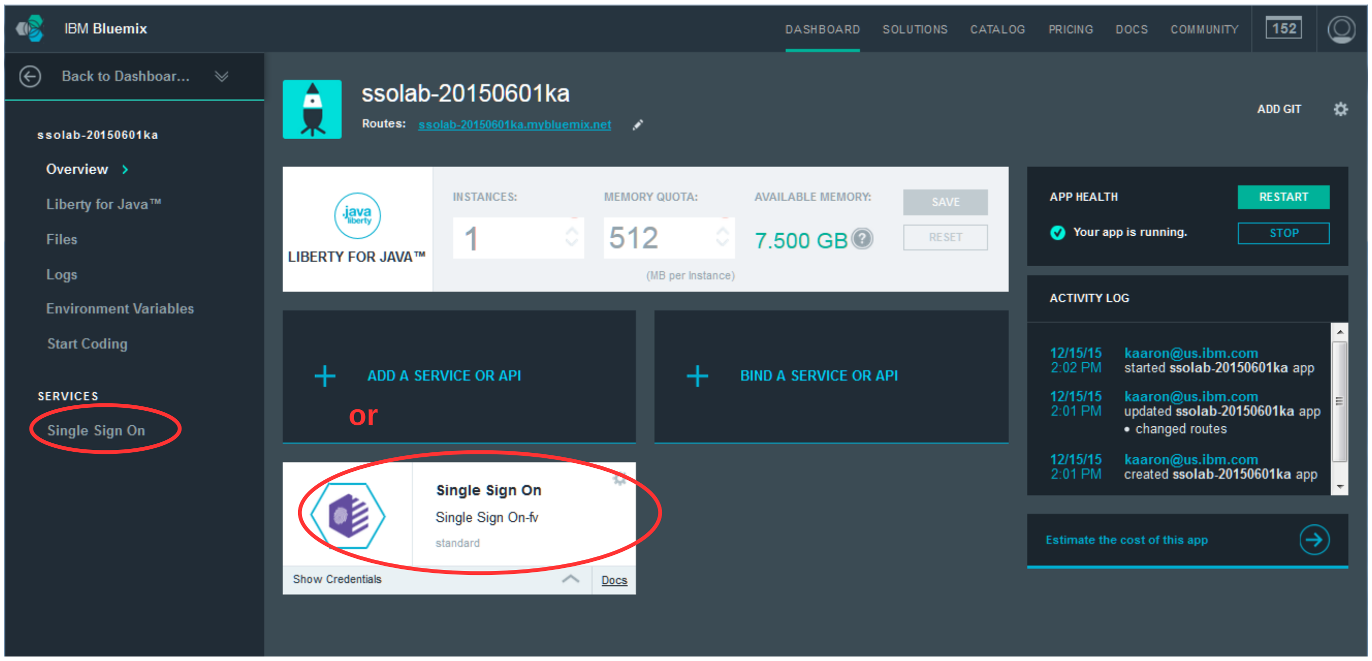
INSTANCES: 1 MEMORY QUOTA: 512 (MB per Instance) AVAILABLE MEMORY: 7.500 GB

Restage Application

Your 'ssolab-20150601ka' app must be restaged to use the new 'Single Sign On-fv' service. Restaging makes this service available for use. Do you want to restage it now?

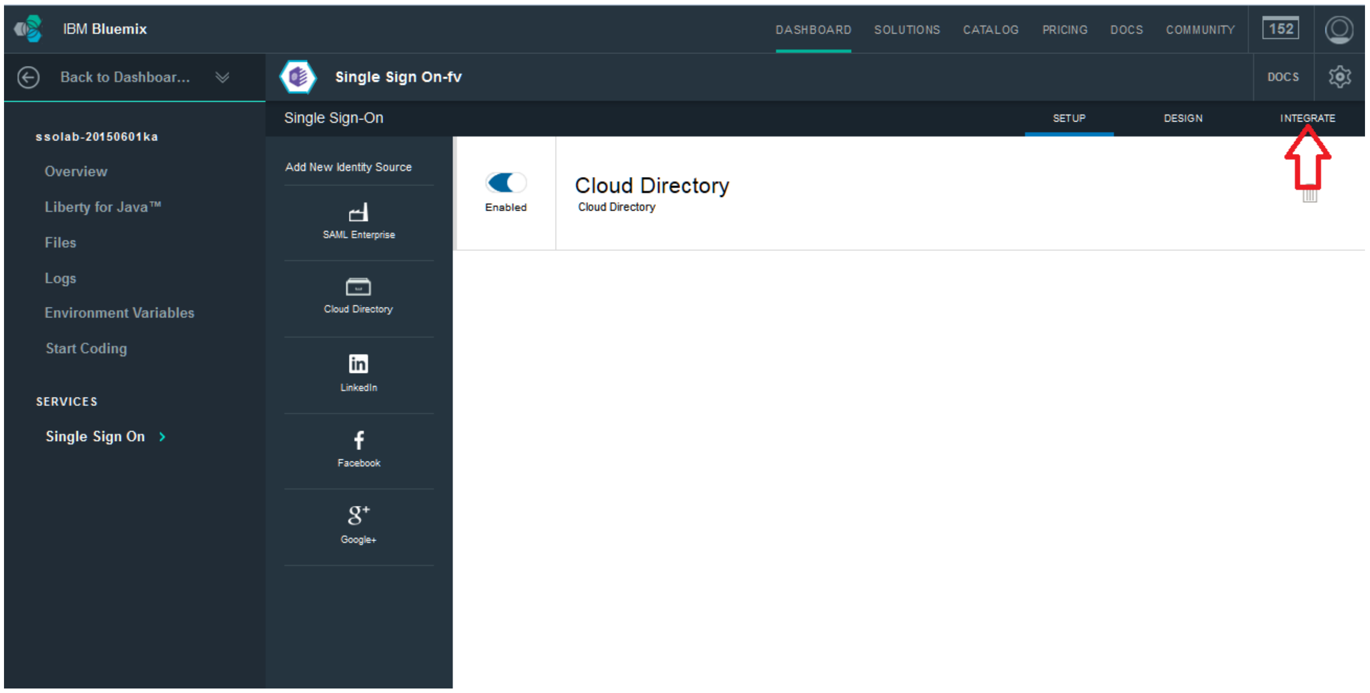
12/15/15 2:02 PM kaaron@us.ibm.com started ssolab-20150601ka app
12/15/15 2:01 PM kaaron@us.ibm.com updated ssolab-20150601ka app • changed routes
12/15/15 2:01 PM kaaron@us.ibm.com created ssolab-20150601ka app

The restage and restart of the application will complete quickly, however, you do not need to wait before proceeding. On the left side of the application's overview panel, click on the Single Sign On label or click on the icon for the service. This will open the configuration panel for the service in the context of this application:



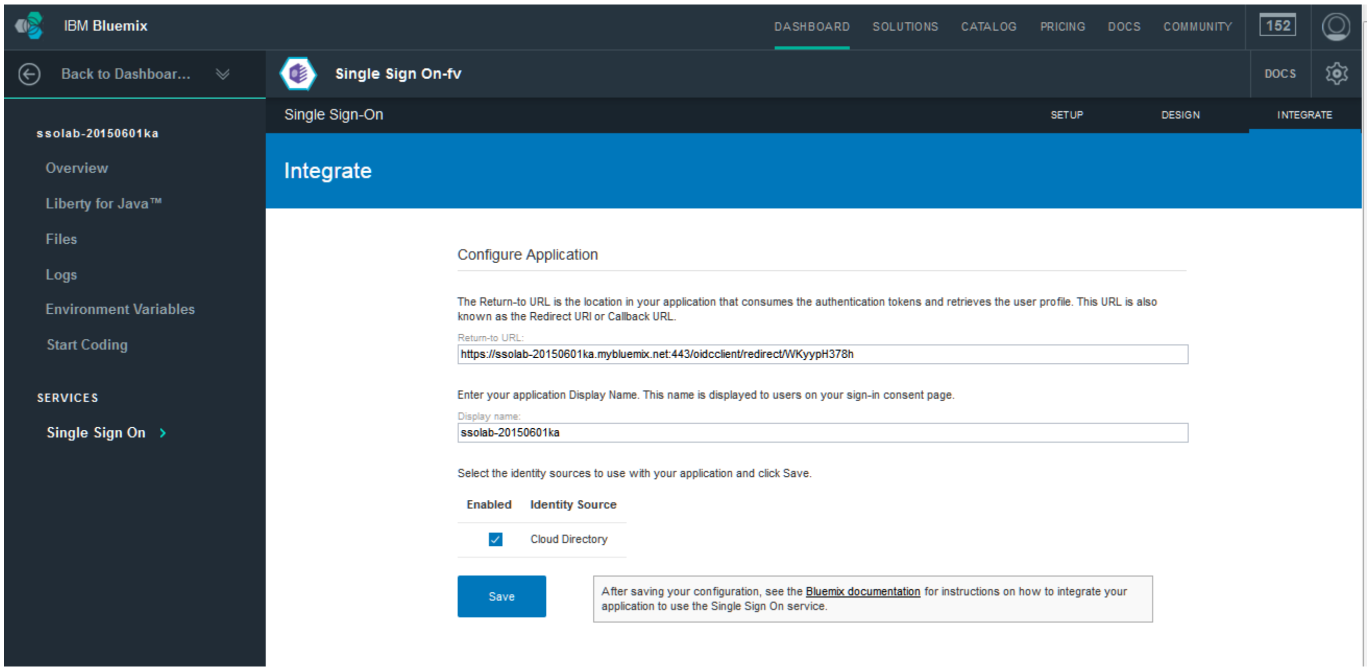
12/15/15 2:02 PM kaaron@us.ibm.com started ssolab-20150601ka app
12/15/15 2:01 PM kaaron@us.ibm.com updated ssolab-20150601ka app
• changed routes
12/15/15 2:01 PM kaaron@us.ibm.com created ssolab-20150601ka app

When accessing the configuration panel from within an application, an INTEGRATE tab will also be shown. For the Java application we are configuring, a module needs to be downloaded from the Single Sign On service. The link to this code is available from the INTEGRATE tab. Although we will return later to the INTEGRATE tab to finish configuring the application, click on the label to switch to this tab now:



Since this is binding to a Java app, the Return-to URL will already be set. The SSO Service will invoke the Return-to URL after authentication is completed.

For Liberty for Java Applications, the Single Sign On service leverages the OpenID Connect (OIDC) client feature from Liberty and the Bluemix Liberty build pack. As a result, Java applications running on Bluemix do not need to include any code to support the OpenID Connect protocol or Single Sign On. (see http://www.ng.bluemix.net/docs/services/SingleSignOn/configure_apps.html#tsk_configuringlibforjavaapp for more info.) We will need to add some security constraints in the Java app to see the Single Sign On work.



Configure Application

The Return-to URL is the location in your application that consumes the authentication tokens and retrieves the user profile. This URL is also known as the Redirect URI or Callback URL.

Return-to URL: https://ssolab-20150601ka.mybluemix.net:443/oidcclient/redirect/WKyypH378h

Enter your application Display Name. This name is displayed to users on your sign-in consent page.

Display name: ssolab-20150601ka

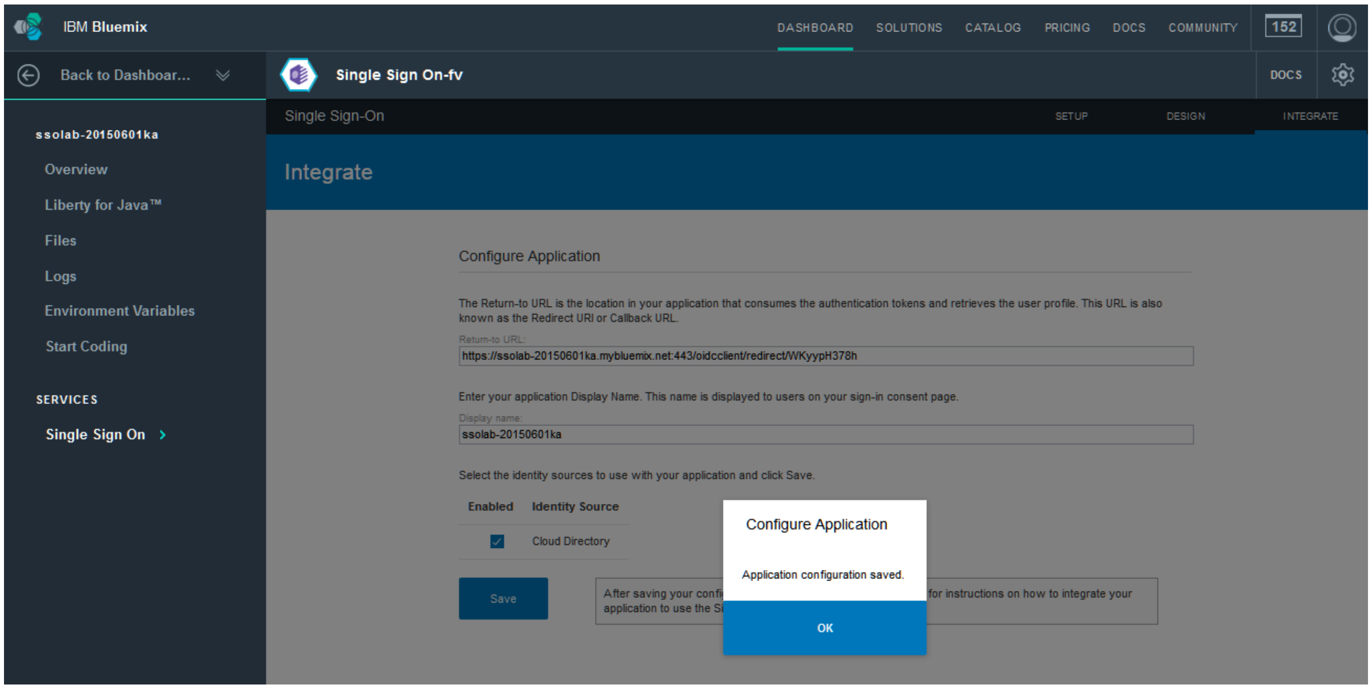
Select the identity sources to use with your application and click Save.

Enabled	Identity Source
✓	Cloud Directory

Save

After saving your configuration, see the Bluemix documentation for instructions on how to integrate your application to use the Single Sign On service.

You may change the Display name for the application from the default. This is the name used by the SSO service when prompting a user for consent to allow the retrieval of personal information from the identity source. Once the Display name updated (if desired), click on the Save button:



Configure Application

The Return-to URL is the location in your application that consumes the authentication tokens and retrieves the user profile. This URL is also known as the Redirect URI or Callback URL.

Return-to URL: https://ssolab-20150601ka.mybluemix.net:443/oidcclient/redirect/WKyypH378h

Enter your application Display Name. This name is displayed to users on your sign-in consent page.

Display name: ssolab-20150601ka

Select the identity sources to use with your application and click Save.

Enabled	Identity Source
✓	Cloud Directory

After saving your confi... for instructions on how to integrate your application to use the S...

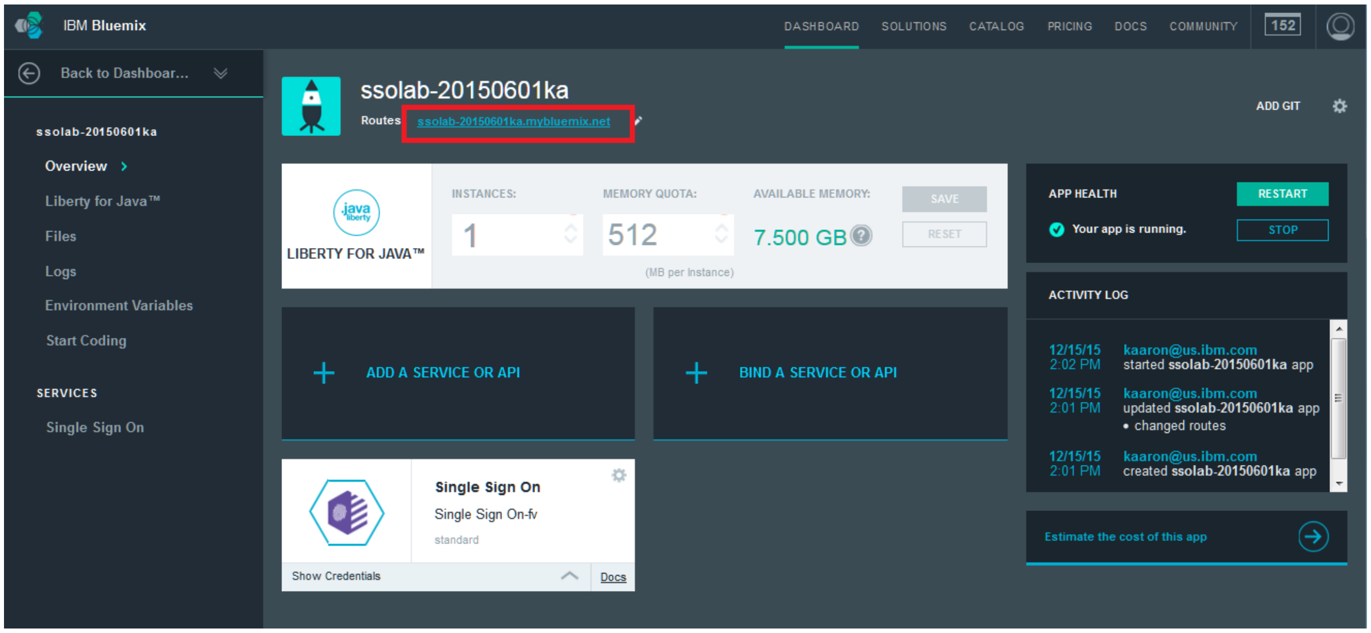
Configure Application

Application configuration saved.

After saving, click on Back to Dashboard in the upper left to return to the IBM Bluemix Cloud Foundry dashboard. At this point, the SSO service is ready to be used with the application.

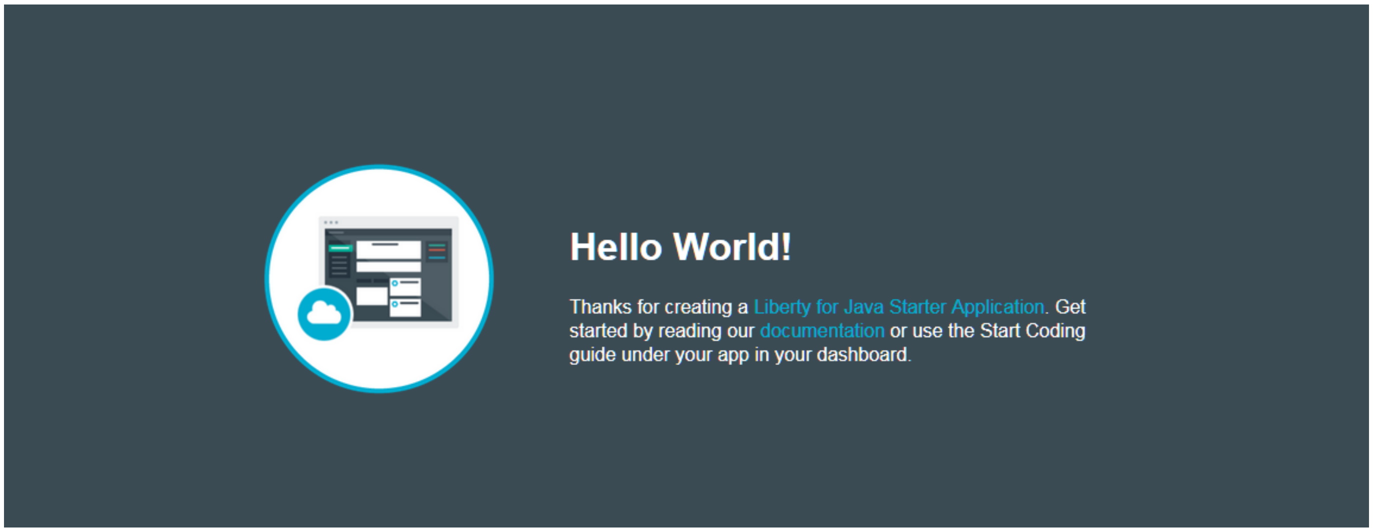
Run the app

Now that it is all configured in Bluemix, go ahead and test that the app is working. From the Bluemix Dashboard, click the SSO application you created above. Click on the app url to open the application's home page.



12/15/15 2:02 PM kaaron@us.ibm.com started ssolab-20150601ka app
12/15/15 2:01 PM kaaron@us.ibm.com updated ssolab-20150601ka app
• changed routes
12/15/15 2:01 PM kaaron@us.ibm.com created ssolab-20150601ka app

You should see the home screen for the java starter app.



Hello World!

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

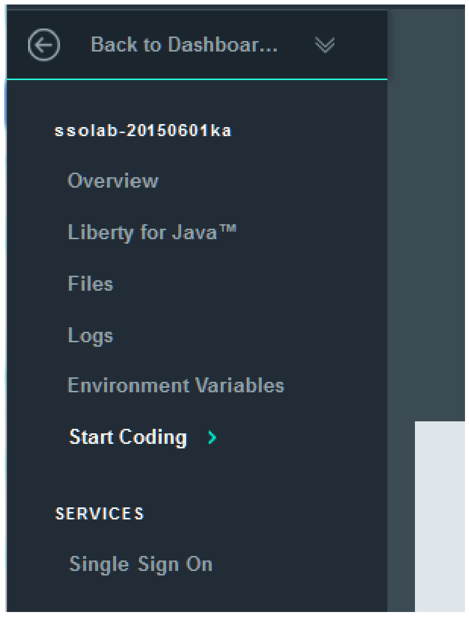
Since you have not put any security constraints in place, no pages in the app or protected.

Add security constraints to Java app

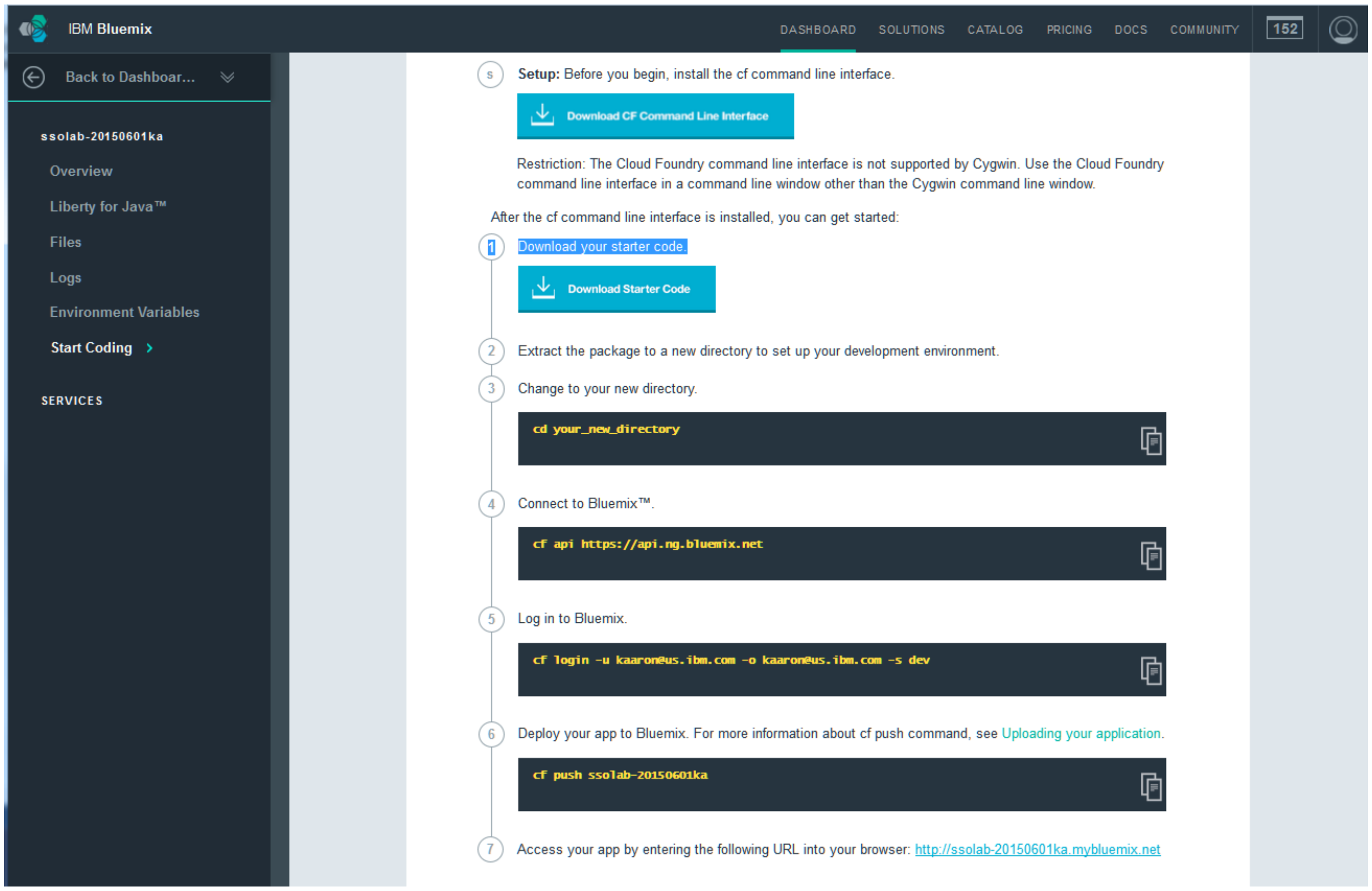
Now that Single Sign On is bound and configured to our java app, let's add security constraints so only the user you added to the Cloud Directory will be able to see the app.

Download Starter Code

Now that your app and SSO is bound together, you can download the starter code. From the Bluemix dashboard, click your application to enter your application Overview. In the left menu, click Start Coding.



Scroll down to the series of numbered steps starting with the Download Starter Code button. Make a note of these steps (they will not match the picture and instead will be customized for your Bluemix id, current Bluemix space and application name). Complete steps 1, 2 and 3 as shown to get a copy of your code.



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

Download 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.

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

1. Download your starter code.

   Download Starter Code

2. Extract the package to a new directory to set up your development environment.
3. Change to your new directory.

```
cd your_new_directory
```

4. Connect to Bluemix™.

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

5. Log in to Bluemix.

```
cf login -u kaaron@us.ibm.com -o kaaron@us.ibm.com -s dev
```

6. Deploy your app to Bluemix. For more information about cf push command, see Uploading your application.

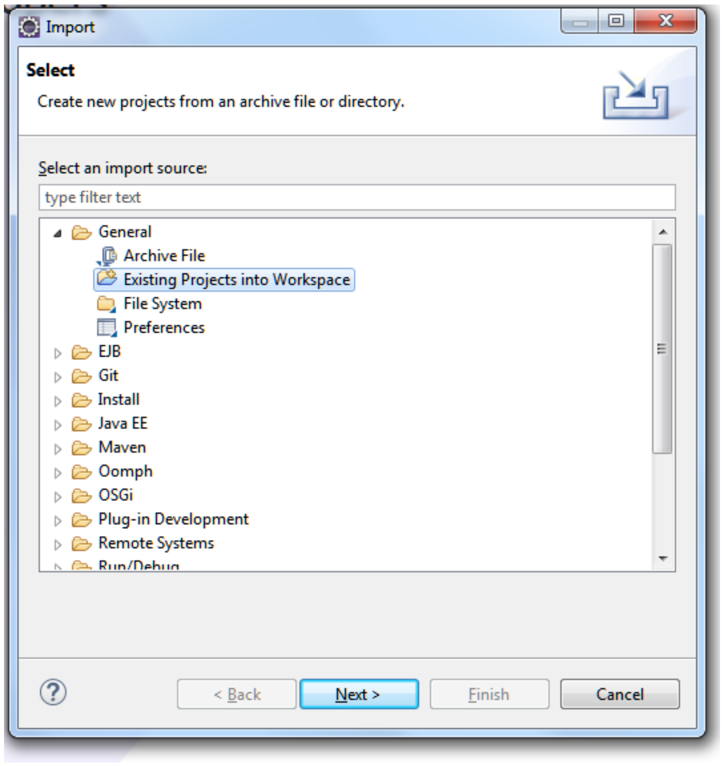
```
cf push ssolab-20150601ka
```

7. Access your app by entering the following URL into your browser: http://ssolab-20150601ka.mybluemix.net

Now that you have the code, it is time to update the application to have some security constraints.

Import into Eclipse

Next, open Eclipse. Select File -> Import.. . Select General -> Existing Projects into Workspace in the Import Select dialog.


Import

Select

Create new projects from an archive file or directory.

Select an import source:

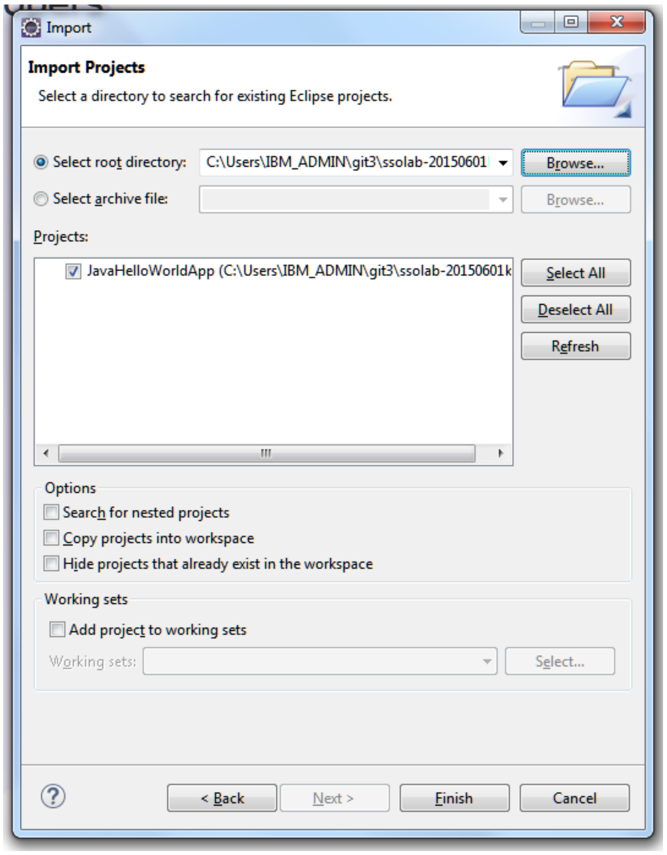
type filter text

- General
  - Archive File
  - Existing Projects into Workspace
  - File System
  - Preferences
- EJB
- Git
- Install
- Java EE
- Maven
- Oomph
- OSGi
- Plug-in Development
- Remote Systems
- Run/Debug

< Back | Next > | Finish | Cancel

Select Next.

In the Import Projects dialog, click the Browse button and browse to the unzipped code you downloaded from Bluemix.


Import

Import Projects

Select a directory to search for existing Eclipse projects.

(●) Select root directory: C:\Users\IBM_ADMIN\git3\ssolab-20150601

( ) Select archive file:

Projects:

[✓] JavaHelloWorldApp (C:\Users\IBM_ADMIN\git3\ssolab-20150601k

Options

[ ] Search for nested projects

[ ] Copy projects into workspace

[ ] Hide projects that already exist in the workspace

Working sets

[ ] Add project to working sets

Working sets:

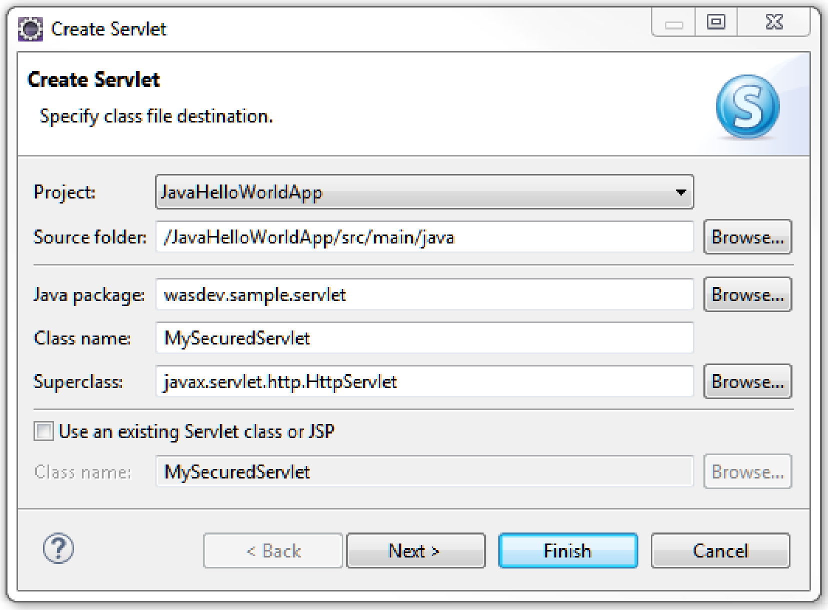
Click Finish.

Add servlet to protect

Let's add a servlet to protect from all users except those authenticated by SSO.

In Eclipse, select File -> New -> Servlet.

Create a servlet names MySecuredServlet.


Create Servlet

Create Servlet

Specify class file destination.

Project: JavaHelloWorldApp

Source folder: /JavaHelloWorldApp/src/main/java

Java package: wasdev.sample.servlet

Class name: MySecuredServlet

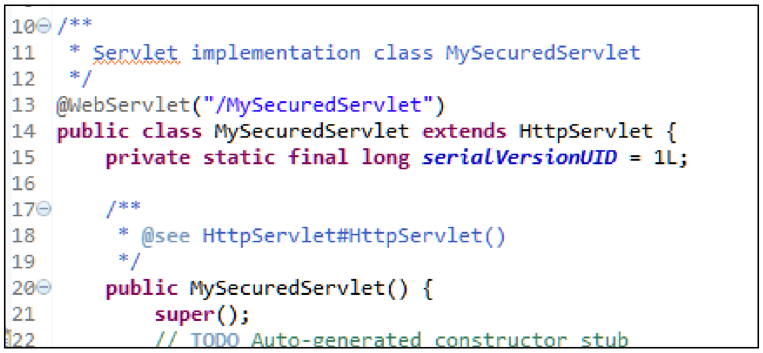
Superclass: javax.servlet.http.HttpServlet

☐ Use an existing Servlet class or JSP

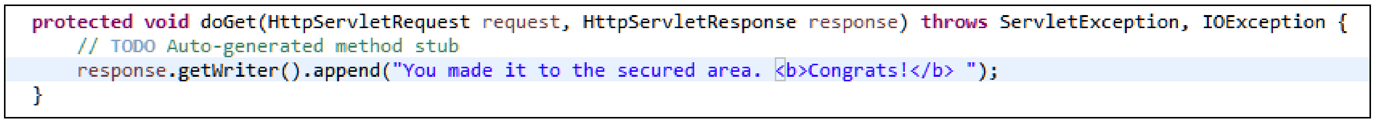
Class name: MySecuredServlet

Click Finish.

The servlet you just created should already have a WebServlet annotation set.

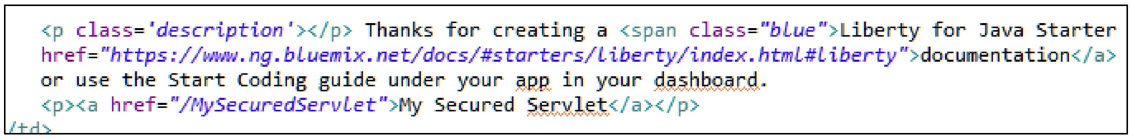

```
/**
 * Servlet implementation class MySecuredServlet
 */
@WebServlet("/MySecuredServlet")
public class MySecuredServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public MySecuredServlet() {
        super();
        // TODO Auto-generated constructor stub
```

Update the response text in the doGet method to be a custom message you want to see.


```
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // TODO Auto-generated method stub
    response.getWriter().append("You made it to the secured area. <b>Congrats!</b> ");
}
```

Next, open index.html.

Add an html link to the new servlet you created.


```
<p class='description'></p> Thanks for creating a <span class="blue">Liberty for Java Starter
href="https://www.ng.bluemix.net/docs/#starters/liberty/index.html#liberty">documentation</a>
or use the Start Coding guide under your app in your dashboard.
<p><a href="/MySecuredServlet">My Secured Servlet</a></p>
</td>
```

This will add a link from the home page to the secured servlet.

Add http constraints

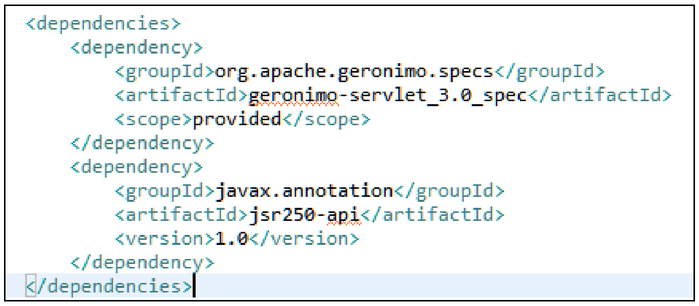
Now let's protect the servlet. We are going to use javax annotations to secure the MySecuredServlet page.

First we need to include the libraries for the annotations.

Open pom.xml

There are several ways to add a dependency in the pom file. I like to add it by straight xml. So, click the pom.xml tab to see the raw xml for the pom file.

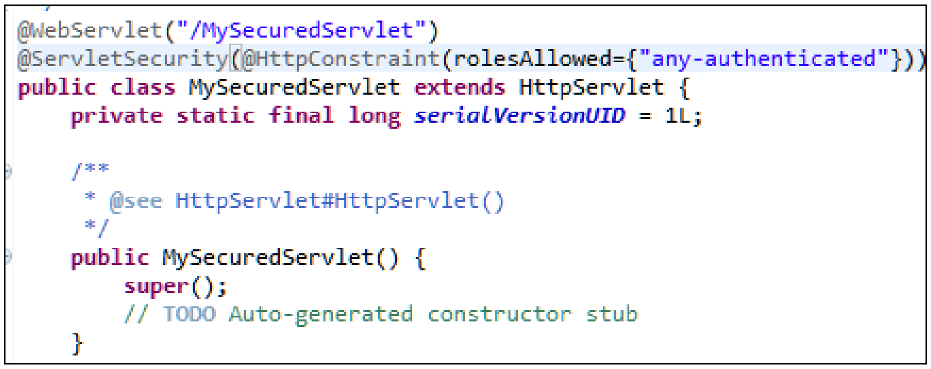
In the dependencies, add a dependency for jsr250-api.



```
<dependencies>
    <dependency>
        <groupId>org.apache.geronimo.specs</groupId>
        <artifactId>geronimo-servlet_3.0_spec</artifactId>
        <scope>provided</scope>
    </dependency>
    <dependency>
        <groupId>javax.annotation</groupId>
        <artifactId>jsr250-api</artifactId>
        <version>1.0</version>
    </dependency>
</dependencies>
```

Next, open MySecuredServlet.java file.

Add a the servlet security and HTTP constraint annotations as shown in the next image.



```
@WebServlet("/MySecuredServlet")
@ServletSecurity(@HttpConstraint(rolesAllowed={"any-authenticated"}))
public class MySecuredServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public MySecuredServlet() {
        super();
        // TODO Auto-generated constructor stub
    }
```

This will protect all access to this servlet except by those authenticated users who have the role of 'any-authenticated'.

Map roles

Next we are going to map the role 'any-authenticated' to a valid subject.

Add a folder named 'resources' under the src\main folder.

Add a folder named 'META-INF' under the newly created resources folder.

Create a new file named ibm-application-bnd.xml

Add the xml you see in the next image

```xml
<?xml version="1.0" encoding="UTF-8"?>
<application-bnd xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://websphere.ibm.com/xml/ns/javaee"
    xsi:schemaLocation="http://websphere.ibm.com/xml/ns/javaee
        http://websphere.ibm.com/xml/ns/javaee/ibm-application-bnd_1_0.xsd"
    version="1.0">

    <security-role name="any-authenticated">
        <special-subject type="ALL_AUTHENTICATED_USERS" />
    </security-role>

</application-bnd>
```

This says that All Authenticated Users get the role of 'any-authenticated'.

Lastly, open the pom.xml again. You will need to inform maven to use the src\main\resources files when it builds the war file. In the build -> pluginManagement -> plugins -> maven war plugin, add the following

```xml
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-war-plugin</artifactId>
    <configuration>
        <failOnMissingWebXml>false</failOnMissingWebXml>
        <webResources>
            <resource>
                <directory>${project.basedir}/src/main/resources</directory>
            </resource>
        </webResources>
        <warName>JavaHelloWorldApp</warName>
    </configuration>
</plugin>
```

Prepare and push your project

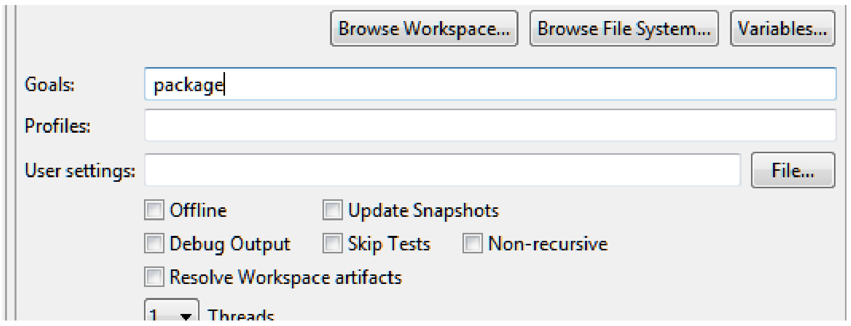
You are almost ready to push the project to Bluemix.

(Kyle note: no manifest change is needed if the reader downloaded the code from Bluemix after binding the app to SSO).

Compile the war by right clicking on pom.xml from Enterprise Explorer view.

Select Run As -> Maven build… from the context menu.

In the goal field, type package.



Goals: package
Profiles:
User settings:
☐ Offline ☐ Update Snapshots
☐ Debug Output ☐ Skip Tests ☐ Non-recursive
☐ Resolve Workspace artifacts
1 Threads

Then click the Run button.

Check the Console view and make sure your build was successful.

Open a command promt.

Go to the directory that contains the project you have been working on.

Using the cloud foundry command, login to bluemix.

Once logged in, push your app. Note, you must be in the directory that has the manifest.yml file.

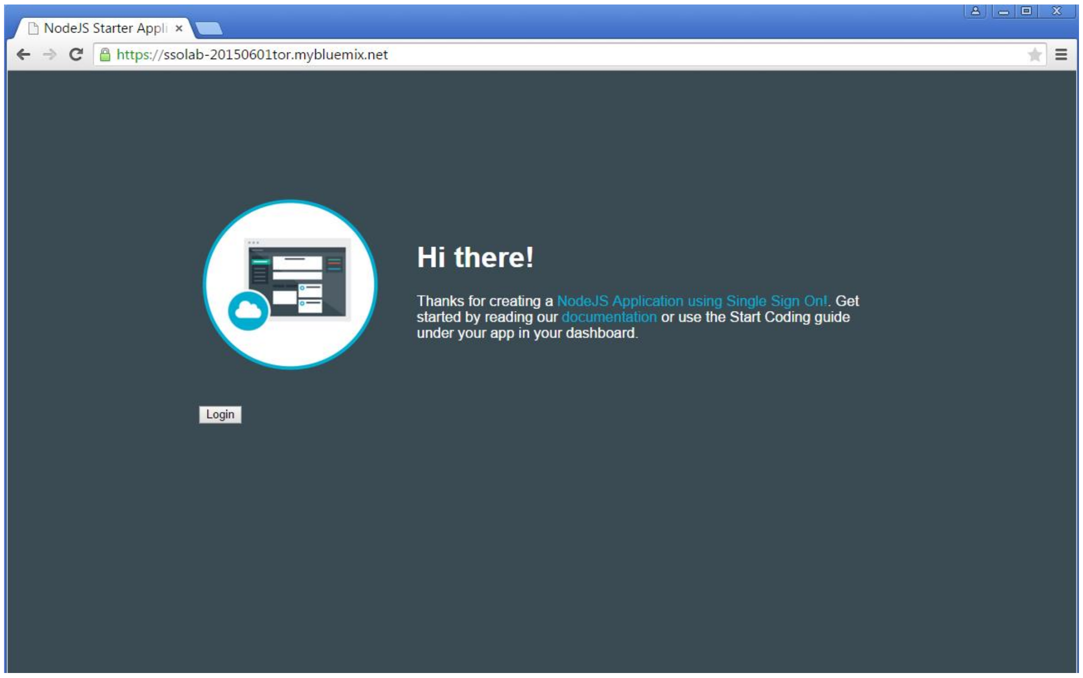
Run the command cf push yourappname, where yourappname is the name of the app on Bluemix.

Once your app is pushed and restarted, you are ready to test.

Testing the application

When testing with the Single Sign On service with the Cloud Directory, it can be easier to use a different browser security and cookie context (for example in a new “private” window in Firefox) to simulate the experience of a new user to your application. Or you can also just use an entirely different browser. Here we'll use a Chrome browser in the screenshots.

Begin by opening your application's url: https://<appname>.mybluemix.net (be sure to use https!):

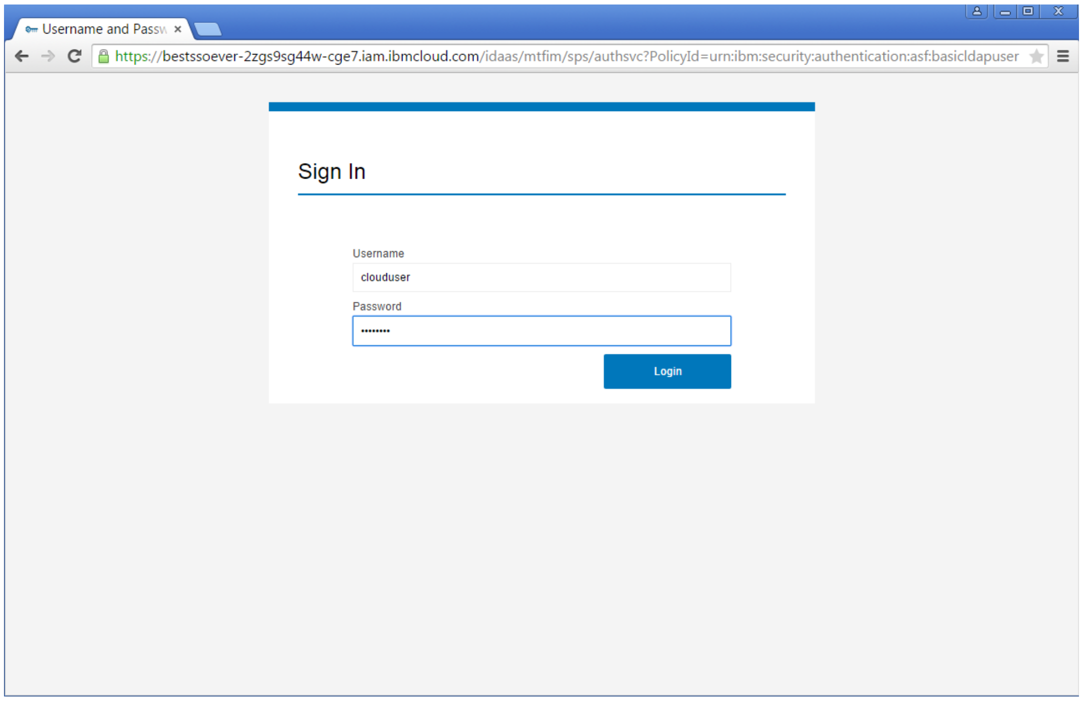


Hi there!

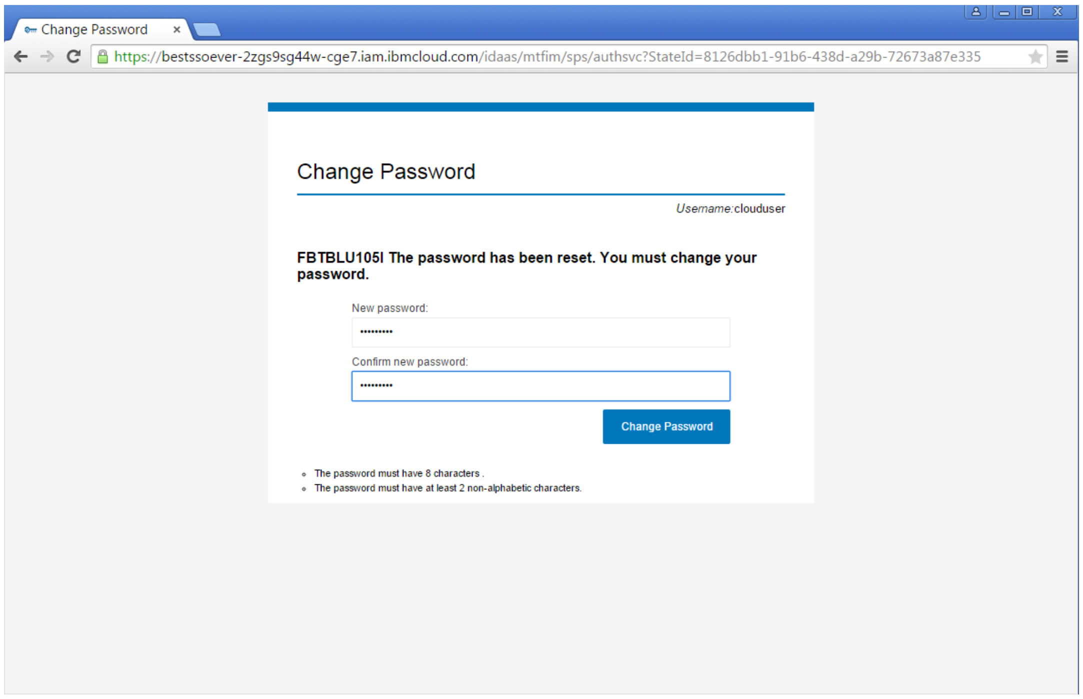
Thanks for creating a NodeJS Application using Single Sign On!. Get started by reading our documentation or use the Start Coding guide under your app in your dashboard.

Note that the My Secured Servlet link is at the end of the paragraph. Click on this link to bring up the Single Sign On identity provider chooser list:

Since the Single Sign On service is configured with only one identity source, only one option is shown. Click on Cloud Directory and sign in as the user you added to the directory previously:


Username and Passw
https://bestssoever-2zgs9sg44w-cge7.iam.ibmcloud.com/idaas/mtfim/sps/authsvc?PolicyId=urn:ibm:security:authentication:asf:basicldapuser
Sign In
Username
clouduser
Password
••••••••
Login

After you click on the Login button – a prompt will be displayed (this happens by policy when using the Cloud Directory) stating that the user password has been reset and must be changed. Note that requirements for the password are displayed in the form. Enter in an updated password:


Change Password

https://bestssoever-2zgs9sg44w-cge7.iam.ibmcloud.com/idaas/mtfim/sps/authsvc?StateId=8126dbb1-91b6-438d-a29b-72673a87e335

Change Password

Username: clouduser

FBTBLU105I The password has been reset. You must change your password.

New password: •••••••••

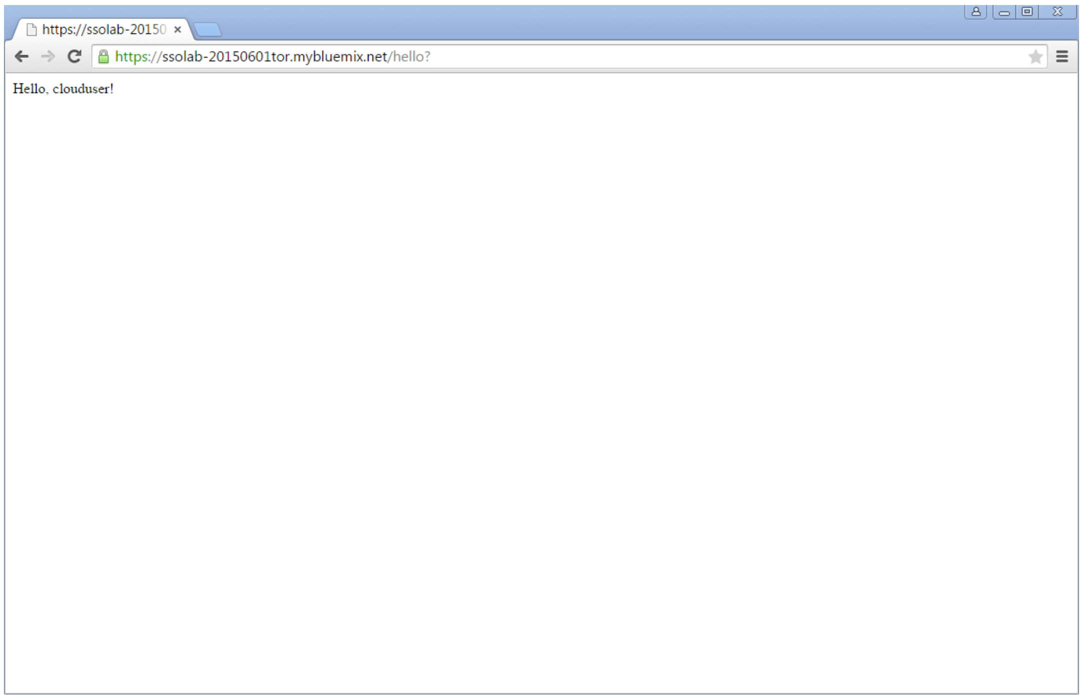
Confirm new password: •••••••••

Change Password

- The password must have 8 characters .
- The password must have at least 2 non-alphabetic characters.

Next, there will be a panel asking for permission (from the application name you set in the INTEGRATE panel) to retrieve your personal information from the external authentication identity provider. You may select either Allow or Allow and Remember. If you select Allow, then each time you authenticate with this identity provider through the Single Sign On service, the prompt will be displayed.

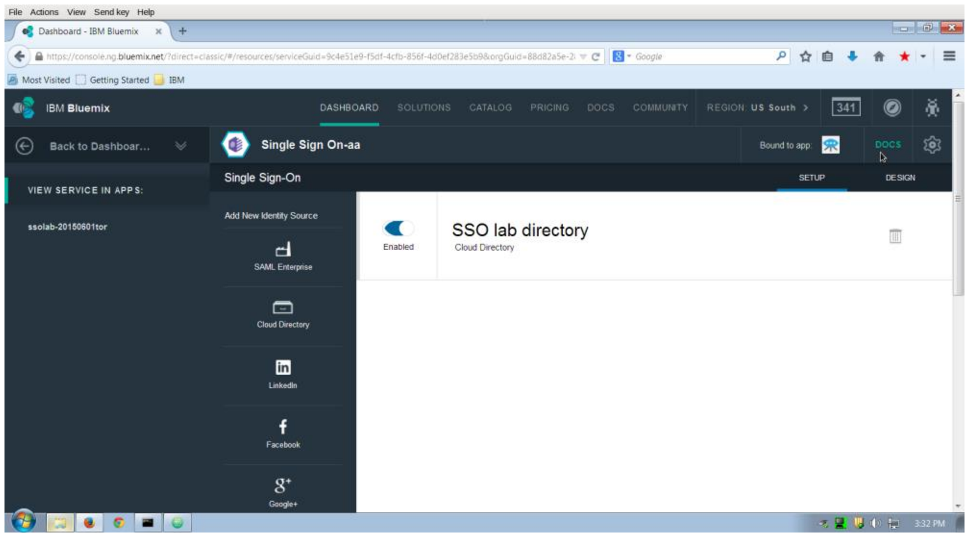
Then the application displays the successful authentication message:


https://ssolab-20150601tor.mybluemix.net/hello?

Hello, clouduser!

Summary and next steps

Now you have an application that can use the Single Sign On service with a simple identity provider. A very good next step would be to continue on with the Bluemix SSO documentation to add a Social Media identity provider. There are step-by-step instructions for this as well as configuration of SAML Enterprise identity sources. To access the documentation, from the configuration panel for the Single Sign On service, click on the DOCS button:


IBM Bluemix
DASHBOARD
SOLUTIONS
CATALOG
PRICING
DOCS
COMMUNITY
REGION US South
Back to Dashboard...
Single Sign On-aa
Bound to app:
DOCS
VIEW SERVICE IN APPS:
ssolab-20150601tor
Single Sign-On
SETUP
DESIGN
Add New Identity Source
SAML Enterprise
Cloud Directory
LinkedIn
Facebook
Google+
Enabled
SSO lab directory
Cloud Directory