April 4, 2016 By Abhisek Jana — 10 Comments (Edit)

How to deploy Spring Boot application in IBM Liberty and WAS 8.5

We have developed RESTful services using Spring Boot in our last tutorial. Now we will learn How to deploy Spring Boot application in IBM Liberty and WAS 8.5. Many of us are still using IBM Application Servers and it's very important to understand how we can deploy our spring boot code there.

Why ?

Since Spring Boot's main objective is to support Microservices and Cloud Native Applications,it includes embedded Tomcat\Jetty. IBM is also moving towards a very similar direction with their Liberty Profile Application Server and Bluemix. Now Liberty has Maven plugin to create, build and deploy Spring Boot Application to IBM Liberty Profile Application Server, which is either hosted locally or remotely using Bluemix. They have also provided similar support for Cloud Foundry (Could Native). So it has become very crucial to understand how to deploy Spring Boot application in IBM Liberty and WAS 8.5 (Websphere Application Server 8.5).

Moreover, many of our existing infrastructure is already using Websphere Application Server 8.5 / IBM Liberty profile. If we want to enable Spring Boot with our current infrastructure we need to also find a way to deploy the application there.

I am not discussing in detailed on WebLogic/JBOSS (Wildfly) here, the same mechanics would apply there as well.

How ?

There are mainly two ways to configure & deploy Spring Boot application in IBM Liberty Profile Application Server. Here are the approaches we shall experiment with.

1. Configure Spring Boot using spring-boot-starter-parent
2. Configure Spring Boot Application using net.wasdev.wlp.starters.springboot

Once our Spring Boot Application is working in IBM Liberty Profile, we will then deploy the application in IBM WebSphere Application Server 8.5.

P.S – The Java code would stay same across each deployment process, however the configuration and `pom` file would be different.

IBM Liberty Profile

spring-boot-starter-parent

As you have already created some Spring Boot Application by now, you can find that in each pom file has a parent artifact id as `spring-boot-starter-parent`. We will keep that as is and deploy our app.

Make sure you have the latest version (8.5.5.9, at the time to writing this blog) of IBM Liberty Profile, also its good to have the full JEE server. If you are not sure check the read-me file for the server version and run the following command to install the full JEE7 profile.

```
./installUtility install javaee-7.0
```

Changes to Maven Configuration File

Since we were using embedded Tomcat server we were using a jar file to run our app, now we need to create a war file to deploy that to IBM Liberty Server. So at first change packaging to `war` from `jar`

Add the Tomcat starter to exclusion list.

```
org.springframework.boot
spring-boot-starter-tomcat
```

Since we need Servlet 3.1 we need to add the dependency here.

```
javax.servlet
javax.servlet-api
3.1.0
provided
```

In case you are using an old application server and want to use Servlet 3.0, use the following dependency instead.

```
javax
javaee-api
7.0
```

```
 provided
```

Here is my pom file.

```
4.0.0

com.adeveloperdiary
springweb
1.0
war

spring_web
springweb


   org.springframework.boot
   spring-boot-starter-parent
   1.3.3.RELEASE




   com.adeveloperdiary.ServletInitializer
   UTF-8
   1.8



      javax.servlet
      javax.servlet-api
      3.1.0
      provided


      org.springframework.boot
      spring-boot-starter-web


            org.springframework.boot
            spring-boot-starter-tomcat



      org.springframework.boot
      spring-boot-actuator
```

```
            org.springframework.boot
            spring-boot-maven-plugin


                    repackage


                    true




            maven-resources-plugin
```

Change the Code

Still now we had the `main` method as the starting point of our Spring Boot Application, however since we are using a `WAR` file we can't leverage that. We need to change our main class to following. I also have a REST service to test the app.

```
@SpringBootApplication(exclude = MessageSourceAutoConfiguration.class)
public class ServletInitializer extends SpringBootServletInitializer {

   /*public static void main(String[] args) {
      SpringApplication.run(ServletInitializer.class, args);
   }*/

   @Override
   protected SpringApplicationBuilder configure(SpringApplicationBuilder application) {
      return application.sources(ServletInitializer.class);
   }
}

@RestController
class RESTController {

   @RequestMapping("/")
   public String hello() {
      return “
```

```
Spring Boot Application from adeveloperdiary.com
”;
   }
}
```

Liberty Server Configuration

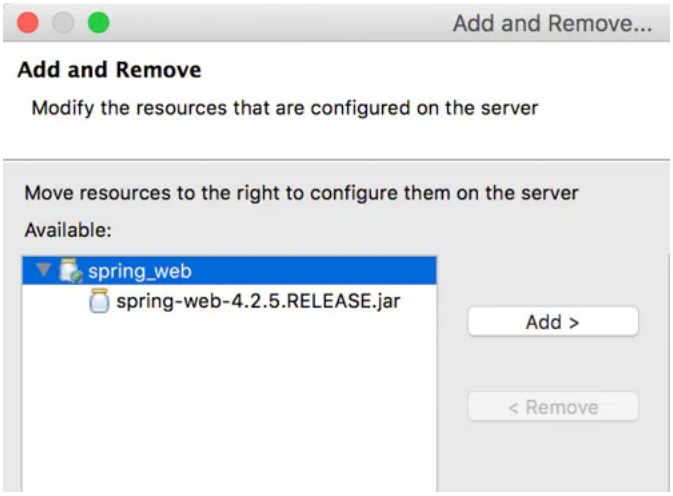
There isn't anything to update/add in the `server.xml` of the Liberty Profile. Here is my version.

```
                javaee-7.0
                adminCenter-1.0
                localConnector-1.0









                admin
```

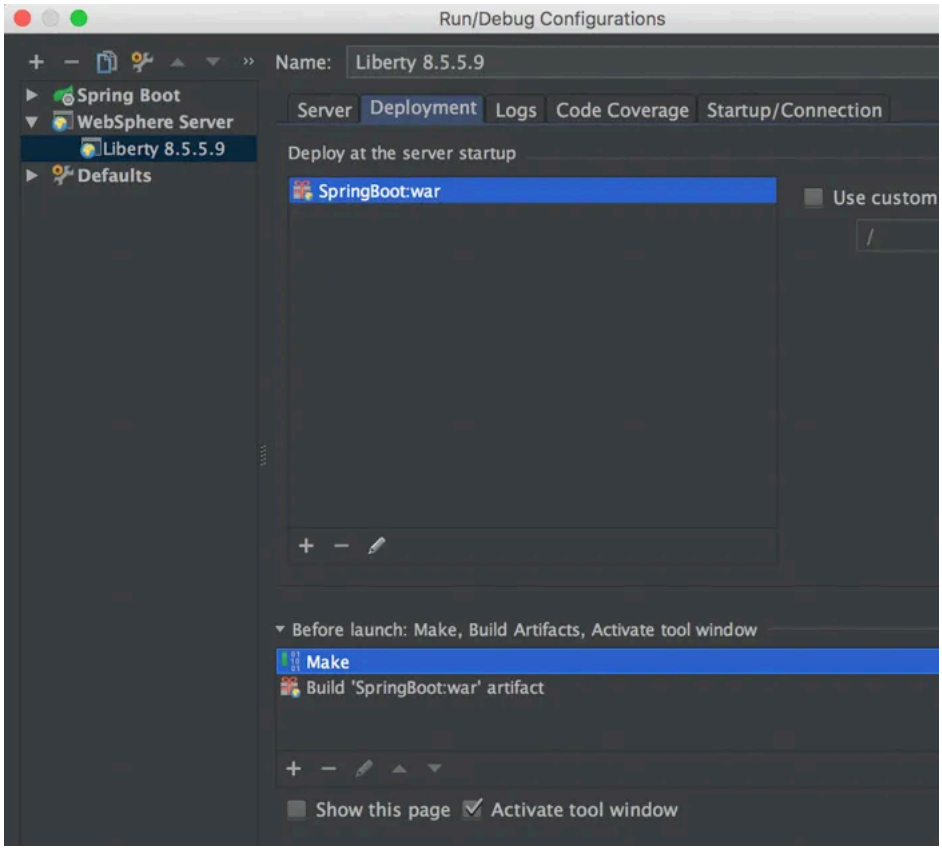
Deploy To Liberty Profile

You can manually drop the war file to the dropins folder once liberty has started it will install the app.

In case you are using eclipse, just add the war to the server, the app will be automatically compiled. One quick tip here is, if you don't create the spring boot app from STS by selecting war packaging option, you may not see the war file listed here.In that case you need to build using Maven and deploy the war manually.



In case you are using IntelliJ IDEA, you need to add a Liberty server profile and add Maven build then deploy artifact. Every time you start the server or redeploy the app it will automatically compile everything and then deploy the war file.



Once you access the URL of the deployed app, Spring Boot will initialize and load the related configurations.

net.wasdev.wlp.starters.springboot

IBM has recently started supported Spring Boot using `net.wasdev.wlp.starters.springboot`. Let me provide you the pom I will be using here.

If you look closely there are many differences of packages here and you can see IBM provided starters has been used here.

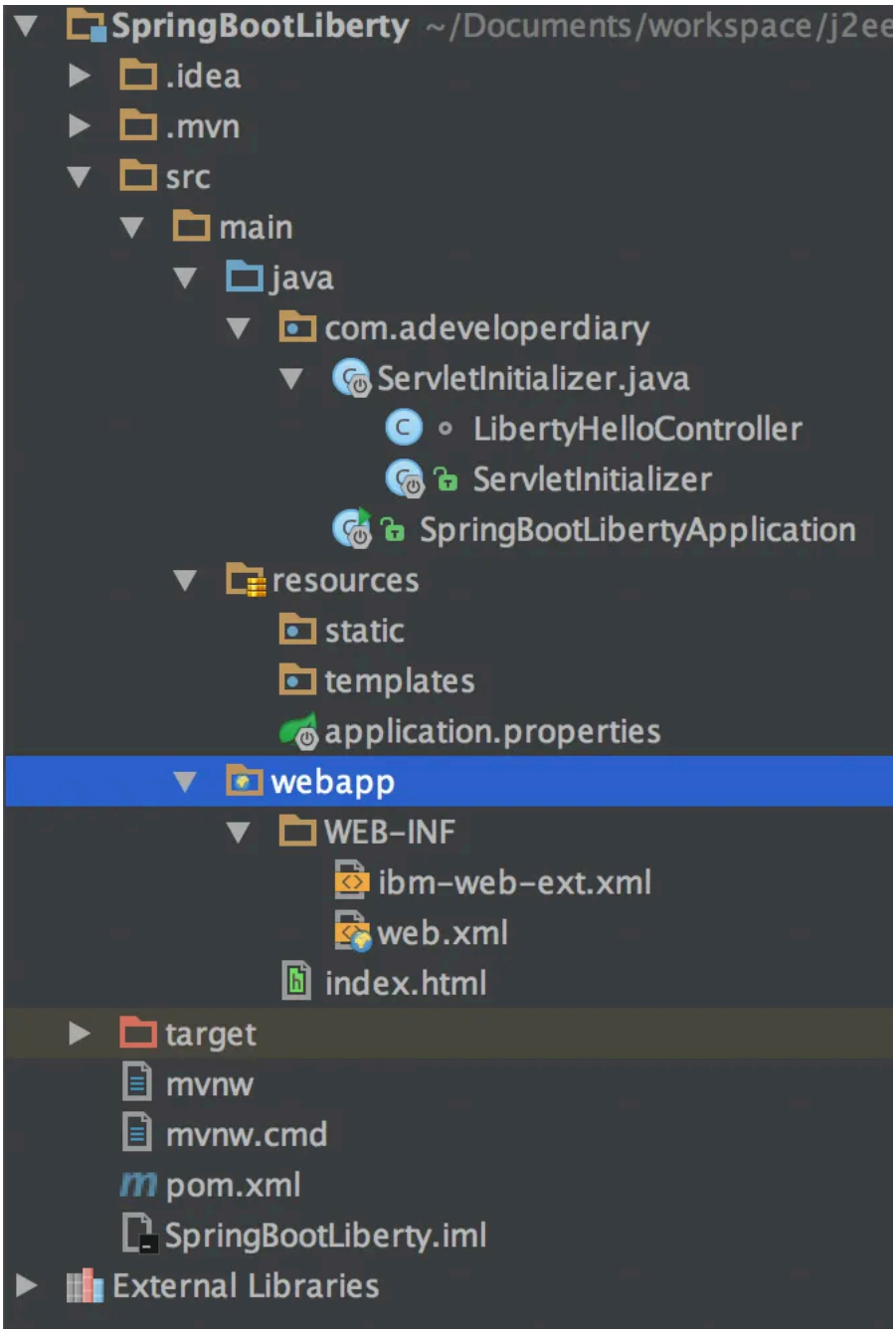
```
    4.0.0
```

```
myParentGroupId
SpringBootLiberty
1.0
war


   scm:git:git@github.com:WASdev/tool.artisan.core.git
   scm:git@github.com:WASdev/tool.artisan.core.git
   git@github.com:WASdev/tool.artisan.core.git



   UTF-8
   UTF-8




      liberty-starter-maven-repo
      liberty-starter-maven-repo
      http://liberty-starter.wasdev.developer.ibm.com/start/api/v1/repo




      net.wasdev.wlp.starters.springboot
      provided-pom
      0.0.1
      pom
      provided


      net.wasdev.wlp.starters.springboot
      runtime-pom
      0.0.1
      pom
      runtime


      net.wasdev.wlp.starters.springboot
      compile-pom
      0.0.1
      pom
      compile


      org.springframework.boot
      spring-boot-actuator
```

```
1.3.3.RELEASE
```

```
org.apache.maven.plugins
maven-war-plugin
2.6

   false
   pom.xml
```

The Java code we changed previously will stay same, however IBM has provided support for defining Context root and `web.xml`. Also you can add any html file. You need to create a folder named `webapp` at the same level of `java` and `resources` and add following files there. Here my directory structure.



You can now define the context root in the `ibm-web-ext.xml` file.

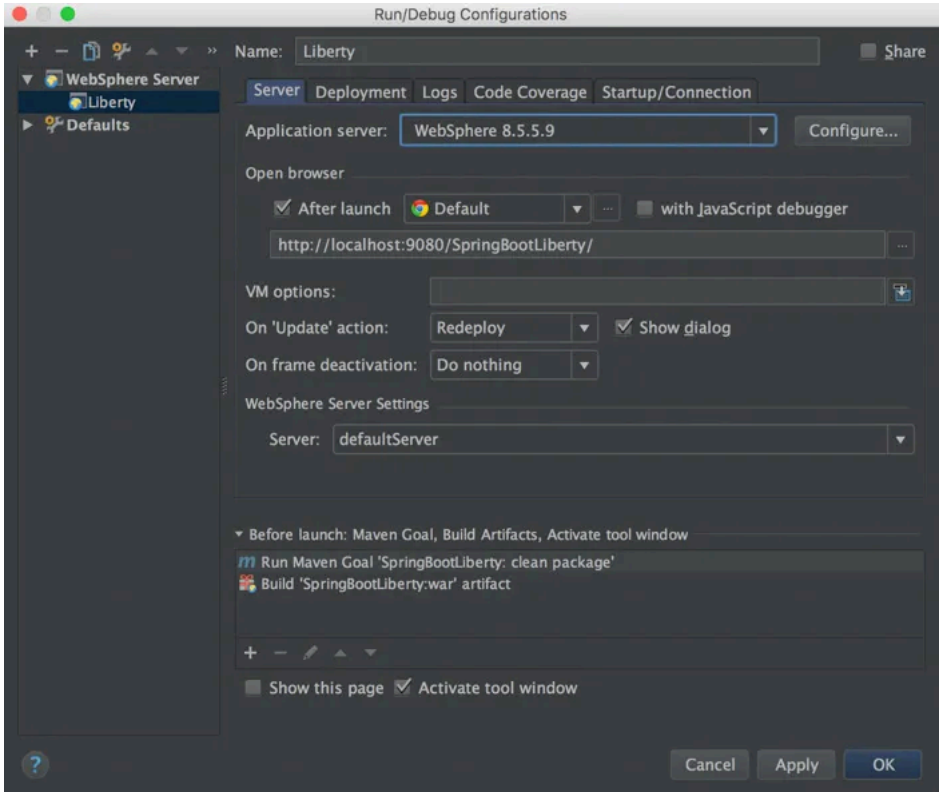
The `web.xml` is a basic one I will be using

```
Spring Boot Liberty

    index.html
```

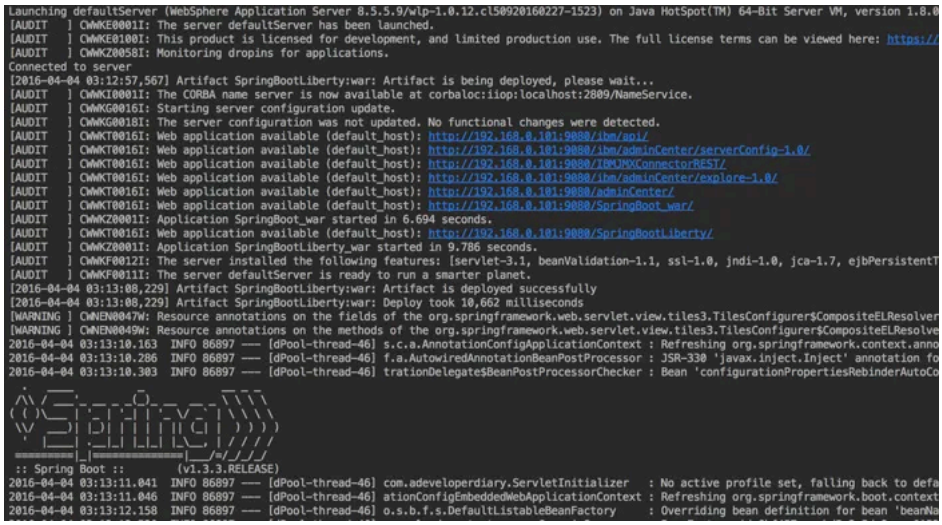
Also added an `index.html` file to make sure its working as expected. When you build your app using the above pom file, it will add the webapp folder in your war file.

I have created the pom specific for development perspective. You can find more details here (http://liberty-app-accelerator.wasdev.developer.ibm.com/).

Here is my IntelliJ configuration.



Once you run the server you will get following logs.



IBM WebSphere Application Server 8.5

In order to deploy Spring Boot Application to IBM WebSphere Application Server 8.5 sever you need to have the latest fix pack installed to get the most compatibility. At the time of writing this post the latest fix pack is `8.5.5.9`, please make to download

and install it.

I have also tested using JDK 6,7 & 8, all of them are working fine. You need to generate the war file using Maven and deploy that to server. Use the above pom files to create the war file.

When using `spring-boot-starter-parent`, you need to manually downgrade the Servlet version to `3.0` since WAS 8.5.5.9 supports only till Servlet 3.0 and not 3.1. You need to use the following dependency.

```
javax
javaee-api
7.0
provided
```

I was able to make both `spring-boot-starter-parent` & `net.wasdev.wlp.starters.springboot` working in IBM WebSphere Application Server 8.5. If you notice we had defined `start-class` in the properties section, WAS will automatically use it to load the Servlet Initializer during server startup.

You will not be able to use Rational Application Developer with the build, you probably need eclipse or IntelliJ IDEA for your development.

Conclusion

You will probably be using embedded Tomcat during your development and use a Maven build to create the WAR specific for Liberty or WAS 8.5. There might be conflicts with older version of the jars when you use WAS 8.5, however Liberty supports most of the latest versions. We will see more compatibility between Spring Boot and Liberty, however may not be so much for WAS 8.5.

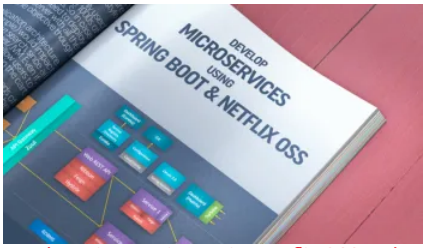
This post is the 4th part of Getting Started with Spring Boot Series. Find all the parts here.

1. An Introduction to Spring Boot
2. How to Create Spring Boot Application Step by Step
3. How to create RESTFul Webservices using Spring Boot
4. How to deploy Spring Boot application in IBM Liberty and WAS 8.5

Related

How to Create Spring Boot Application Step by Step
In "Spring Boot"

An Introduction to Spring Boot
In "Spring Boot"



Develop Microservices using Netflix OSS and Spring Boot
In "Microservice"

Filed Under: Spring Boot | Tagged With: Deploy, IBM, IBM WAS 8.5, liberty, Liberty Profile, Microservices, Spring Boot, WebSphere Application Server

Comments

Sridhar says
April 6, 2016 at 5:07 am

(Edit)

Hi,

Your article was interesting.

I am able to deploy Spring boot app to WAS Liberty Profile.

I am now working on deploying Reverse Proxy Component (Using Spring ZUUL) , Service Registry Component (Using Spring Netflix EUREKA) also to WAS Liberty Profile.

But, I get error when accessing the deployed spring boot service via the ZUUL Proxy.

I believe it has something to do with packaging and deployment to WAS Liberty Profile.

Can you please help me on this.

I have a sample ATOMIC Service, ZUUL Service and EUREKA Service deployed to separate WAS Liberty Profile servers.

I have shared the code for these services in the github repos.

–For ZUUL Reverse Proxy Service
https://github.com/bsridhar123/ZUUL

–For ATOMIC Service which is proxied by ZUUL
https://github.com/bsridhar123/AtomicService

–For EUREKA Service Registry Service
https://github.com/bsridhar123/Eureka

Also, I have the logs folder where I have added the logs obtained when trying to make a lookup of the service via ZUUL Service.

https://github.com/bsridhar123/ZUUL/blob/master/logs/PROXY_ATOMIC.txt

Also, the WAS Libery Profile server.xml for each of the respective services are in the folders below.
–For ATOMIC Service
https://github.com/bsridhar123/AtomicService/blob/master/server-config/server.xml

–For ZUUL Service
https://github.com/bsridhar123/ZUUL/blob/master/server-config/server.xml

–For EUREKA Service
https://github.com/bsridhar123/Eureka/blob/master/server-config/server.xml

Can you please have a look and help me in identifying the cause of the error.

Thanks,
Sridhar
Reply

Ram s says
September 7, 2016 at 11:40 pm

(Edit)

Excellent Article, thanks for putting the this article in various stages. It was very helpful to start from the basics and building the microservices with integration together.

Keep up the good work !!!

Reply

sri123 says
January 23, 2017 at 2:38 pm

(Edit)

Its really very nice articular for beginners, thank you for your post.

Reply

Ramani says
February 22, 2017 at 4:06 pm

(Edit)

nice article step by step

Reply

Stephan Knitelius says
March 27, 2017 at 5:17 am

(Edit)

Why?

You are dealing with fully Java EE 7 (Liberty) / 6 (WAS Traditional) compliant application servers.

Just go ahead and use the provided Java EE stack and deploy a light weight skinny WAR/EAR to your runtime.

The best thing of all – it's all fully integrated and tested for you.

So be a champ and learn about the target environment at hand.

Reply

sulthana says
May 31, 2017 at 5:38 pm

(Edit)

I am deploying this example in WAS 8.5, here is the error I am getting.

Caused by: org.springframework.context.ApplicationContextException: Unable to start EmbeddedWebApplicationContext
due to missing EmbeddedServletContainerFactory bean.
at
org.springframework.boot.context.embedded.EmbeddedWebApplicationContext.getEmbeddedServletContainerFactory(EmbeddedW
at
org.springframework.boot.context.embedded.EmbeddedWebApplicationContext.createEmbeddedServletContainer(EmbeddedWebA
at
org.springframework.boot.context.embedded.EmbeddedWebApplicationContext.onRefresh(EmbeddedWebApplicationContext.java:1

Reply

Aaron Yang says
May 31, 2017 at 11:31 pm

(Edit)

good article, let me quick start spring boot using liberty. Keep up ur great work!

Reply

Aaron Yang says
May 31, 2017 at 11:32 pm

(Edit)

good article, let me quick start spring boot using liberty. Keep up ur great work!

Reply

Thomas says
May 9, 2023 at 6:47 am

(Edit)

Hello,

i was able to start a spring boot application in a liberty (JEE8 Liberty Server in Eclipse) by deploying a war file with the net.wasdev.wlp.starters.springboot approach.

But the port (server.port in application.properties) will not open.
(I think this is because no spring internal tomcat in a liberty server)

How to open the port?
(of course for working REST requests)

Reply

Stefan says
May 16, 2023 at 1:55 am

(Edit)

Hello,

I was able to start two spring boot apps in a liberty with the net.wasdev.wlp.starters.springboot approach.

I can see the “spring logo” in the log twice.

But there a no ports open from the spring boot, no rest controller can be called. How to achieve this?

Kind regards

Reply

Leave a Reply

Logged in as Abhisek Jana. Edit your profile. Log out? Required fields are marked *

Comment *

Post Comment

This site uses Akismet to reduce spam. Learn how your comment data is processed.