

**JAVA EE COURSE**

# **EXERCISE**

## **HELLO WORLD WITH ENTERPRISE JAVA BEANS (EJB' S)**

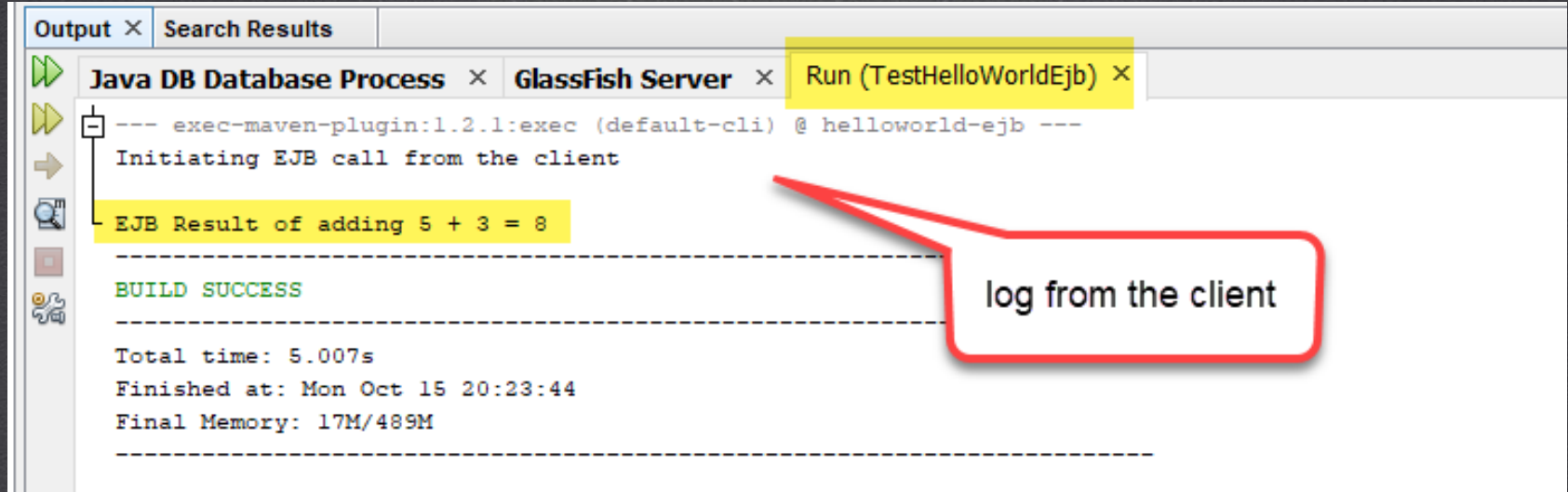


**JAVA EE COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)

# EXERCISE OBJECTIVE

Create a project to create an EJB. The final result is as follows:



```
Output × Search Results
Java DB Database Process × GlassFish Server × Run (TestHelloWorldEjb) ×
--- exec-maven-plugin:1.2.1:exec (default-cli) @ helloworld-ejb ---
Initiating EJB call from the client
EJB Result of adding 5 + 3 = 8
-----
BUILD SUCCESS
-----
Total time: 5.007s
Finished at: Mon Oct 15 20:23:44
Final Memory: 17M/489M
-----
```

log from the client

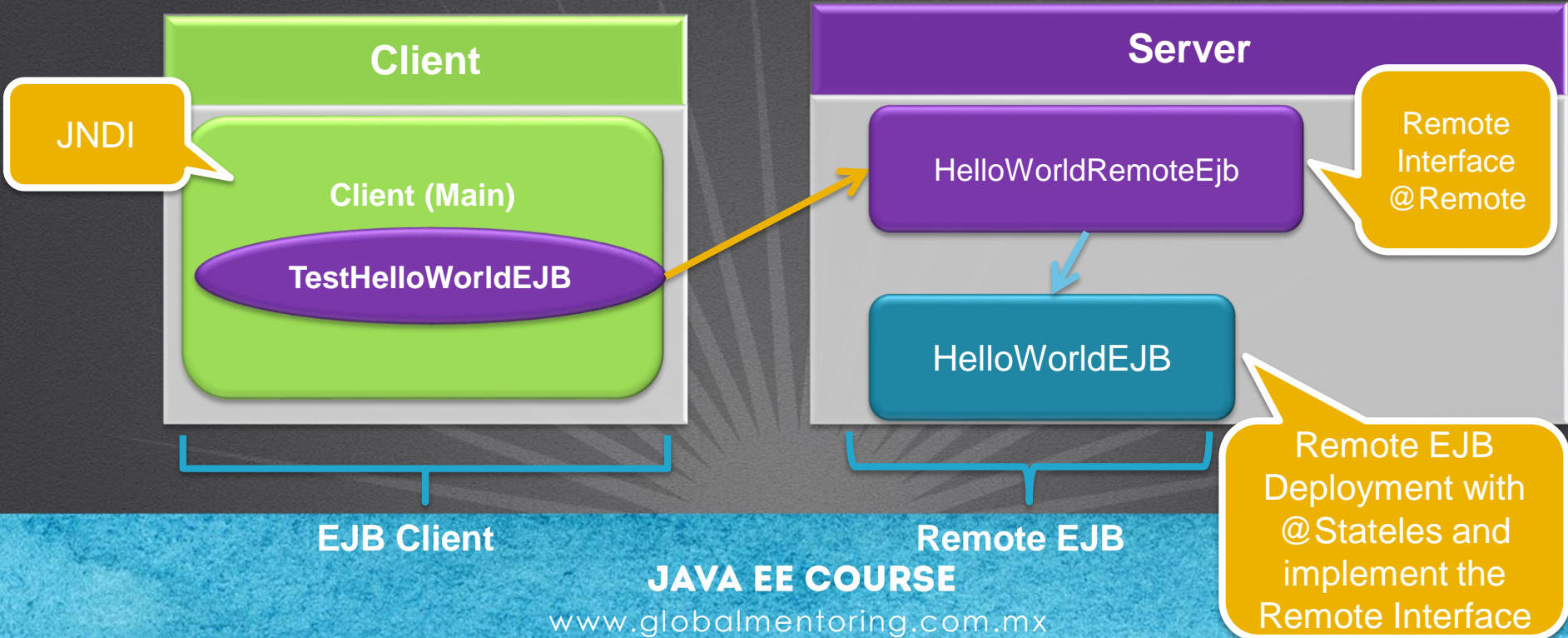
**JAVA EE COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)



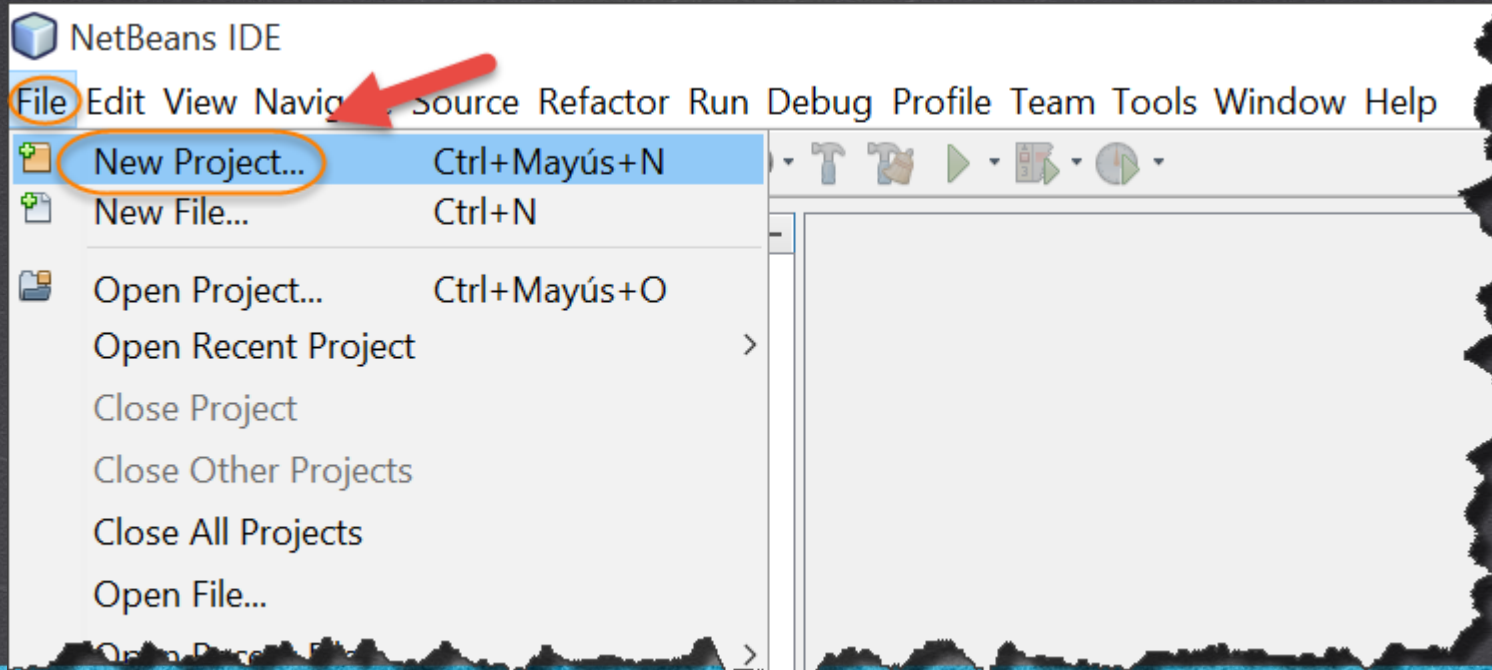
# JAVA EE ARCHITECTURE

In this exercise we are going to add a Remote Session EJB. We will use an interface and a Java class. The interface will add the annotation `@Remote` and the class will implement the interface, we will also add the annotation of `@Stateless` to convert it into an EJB of Stateless type:



# 1. CREATE A NEW PROJECT

We create the helloworld-ejb project:

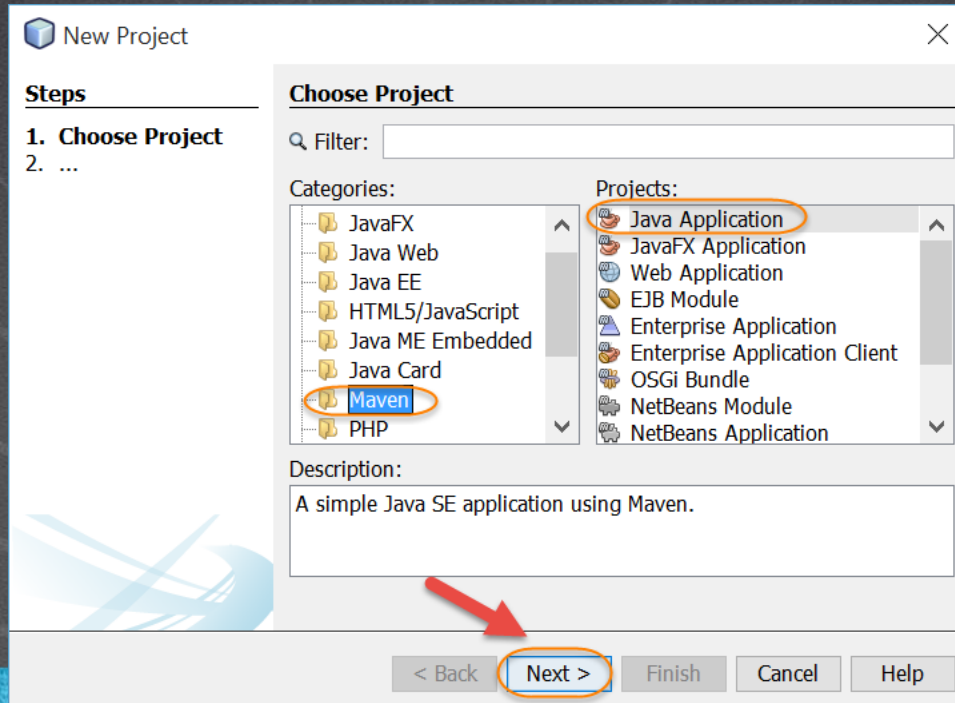


**JAVA EE COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)

# 1. CREATE A NEW PROJECT

We create the helloworld-ejb project:



**JAVA EE COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)



# 1. CREATE A NEW PROJECT

We create the helloworld-ejb project:

**New Java Application**

**Steps**

1. Choose Project
2. **Name and Location**

**Name and Location**

Project Name: helloworld-ejb

Project Location: C:\Courses\JavaEE\Lesson01 Browse...

Project Folder: C:\Courses\JavaEE\Lesson01\helloworld-ejb

Artifact Id: helloworld-ejb

Group Id: beans

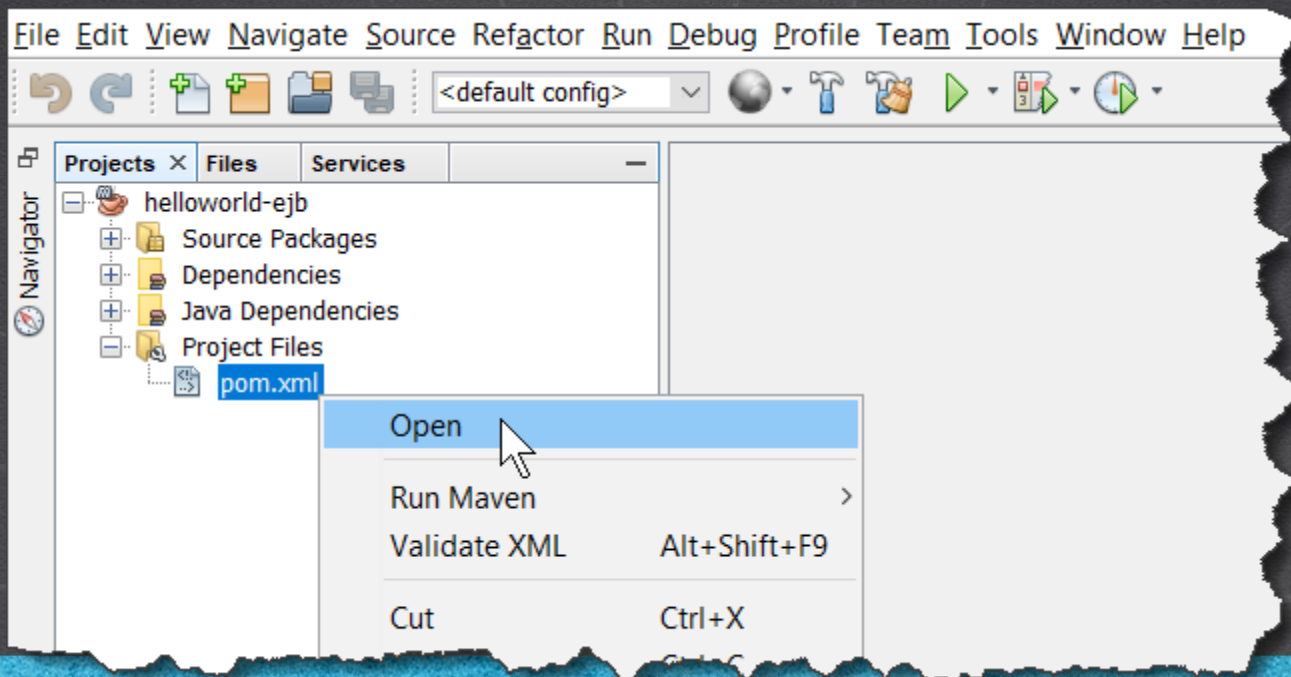
Version: 1

Package: (Optional)

< Back Next > **Finish** Cancel Help

## 2. MODIFY THE FILE

Modify the pom.xml file:



**JAVA EE COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)

## 2. MODIFY THE FILE

[pom.xml:](#)

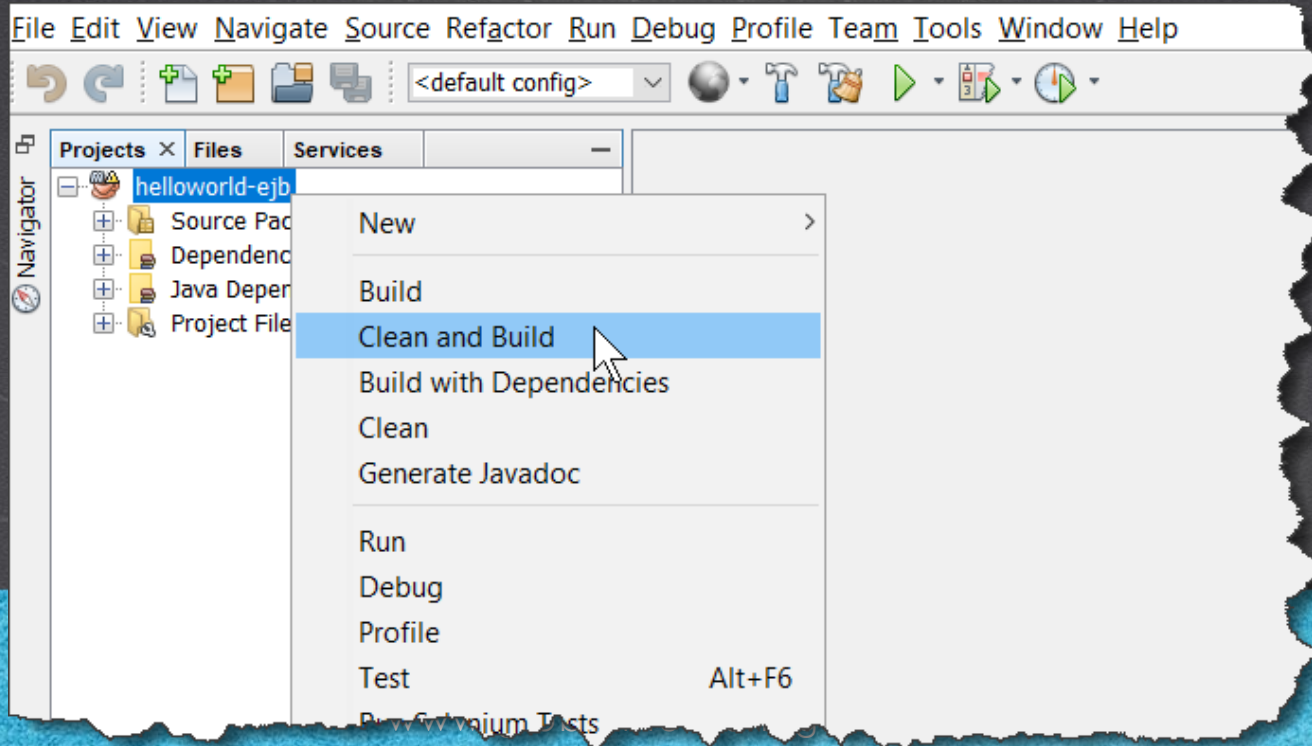
Click to download

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>beans</groupId>
  <artifactId>helloworld-ejb</artifactId>
  <version>1</version>
  <packaging>jar</packaging>
  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
  </properties>
  <dependencies>
    <dependency>
      <groupId>javax</groupId>
      <artifactId>javaee-api</artifactId>
      <version>8.0</version>
      <scope>provided</scope>
    </dependency>
    <!--Library to run the EJB client-->
    <dependency>
      <groupId>org.glassfish.main.appclient</groupId>
      <artifactId>gf-client</artifactId>
      <version>5.0</version>
    </dependency>
  </dependencies>
</project>
```



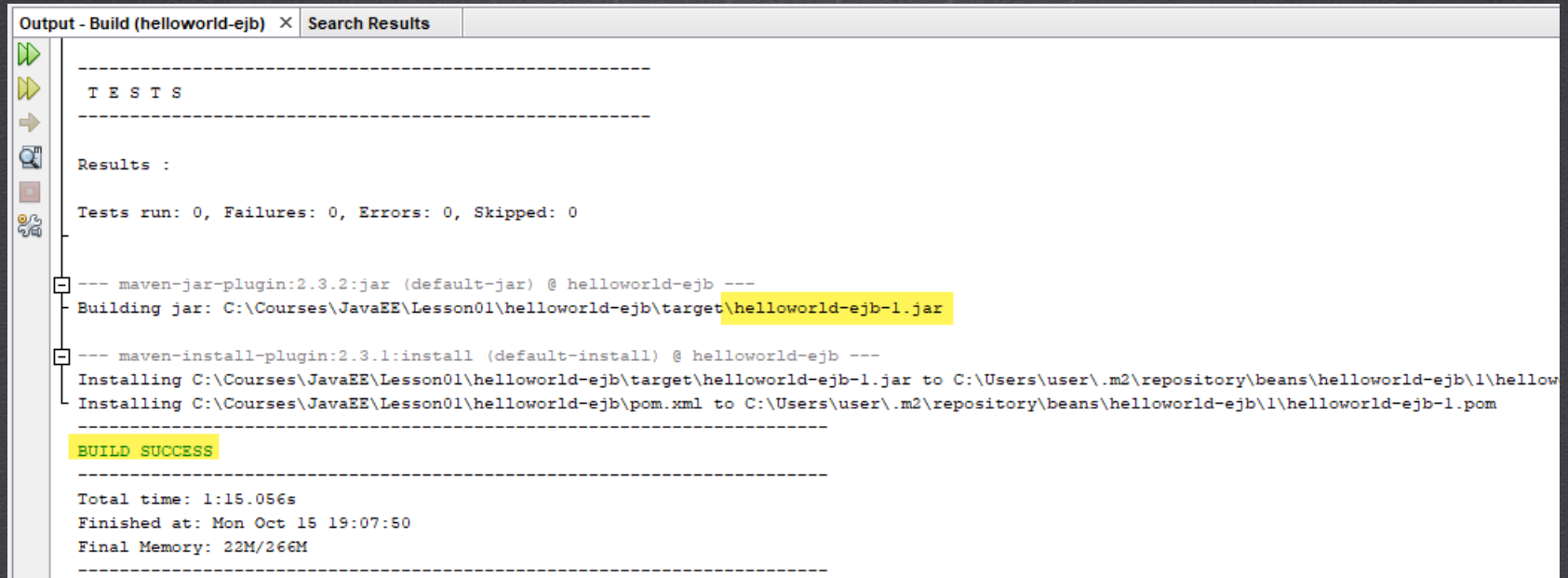
### 3. EXECUTE CLEAN & BUILD

Execute clean & build. Stop any antivirus, firewall, windows defender or any other software that can stop downloading the .jar files:



# 3. EXECUTE CLEAN & BUILD

The result is similar as follows:

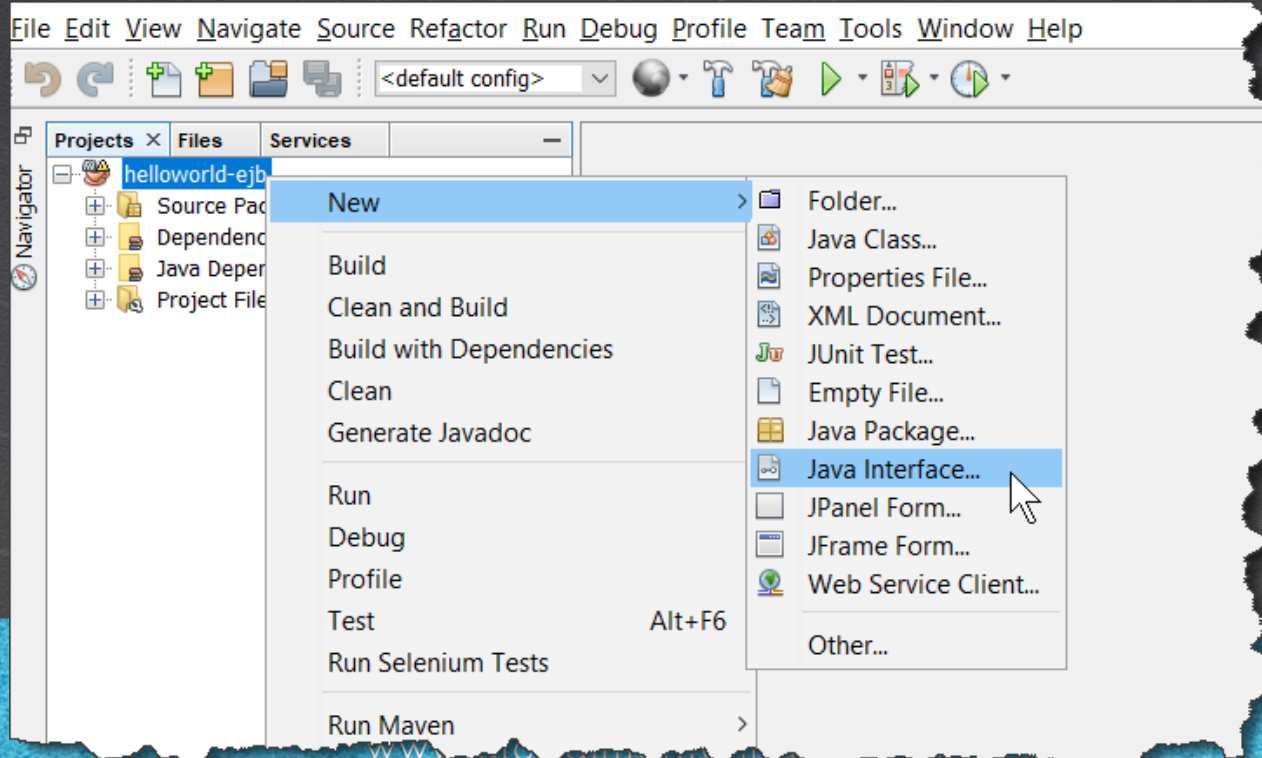


The screenshot shows the 'Output - Build (helloworld-ejb)' window in an IDE. The window has a sidebar on the left with icons for Run, Debug, Test, and Build. The main area displays the following text:

```
-----  
T E S T S  
-----  
  
Results :  
  
Tests run: 0, Failures: 0, Errors: 0, Skipped: 0  
  
--- maven-jar-plugin:2.3.2:jar (default-jar) @ helloworld-ejb ---  
Building jar: C:\Courses\JavaEE\Lesson01\helloworld-ejb\target\helloworld-ejb-1.jar  
  
--- maven-install-plugin:2.3.1:install (default-install) @ helloworld-ejb ---  
Installing C:\Courses\JavaEE\Lesson01\helloworld-ejb\target\helloworld-ejb-1.jar to C:\Users\user\.m2\repository\beans\helloworld-ejb\1\helloworld-ejb-1.jar  
Installing C:\Courses\JavaEE\Lesson01\helloworld-ejb\pom.xml to C:\Users\user\.m2\repository\beans\helloworld-ejb\1\helloworld-ejb-1.pom  
-----  
BUILD SUCCESS  
-----  
  
Total time: 1:15.056s  
Finished at: Mon Oct 15 19:07:50  
Final Memory: 22M/266M  
-----
```

## 4. CREATE AN INTERFACE

We create the interface HelloWorldRemoteEjb.java:





## 4. CREATE AN INTERFACE

We create the interface HelloWorldRemoteEjb.java:

New Java Interface

**Steps**

1. Choose File Type
2. **Name and Location**

**Name and Location**

Class Name: HelloWorldRemoteEjb

Project: helloworld-ejb

Location: Source Packages

Package: beans

Created File: C:\Courses\JavaEE\Lesson01\helloworld-ejb\src\main\java\beans\HelloWorldRemoteEjb.java

< Back   Next >   **Finish**   Cancel   Help

**JAVA EE COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)

## 5. MODIFY THE FILE

HelloWorldRemoteEjb.java:

Click to download

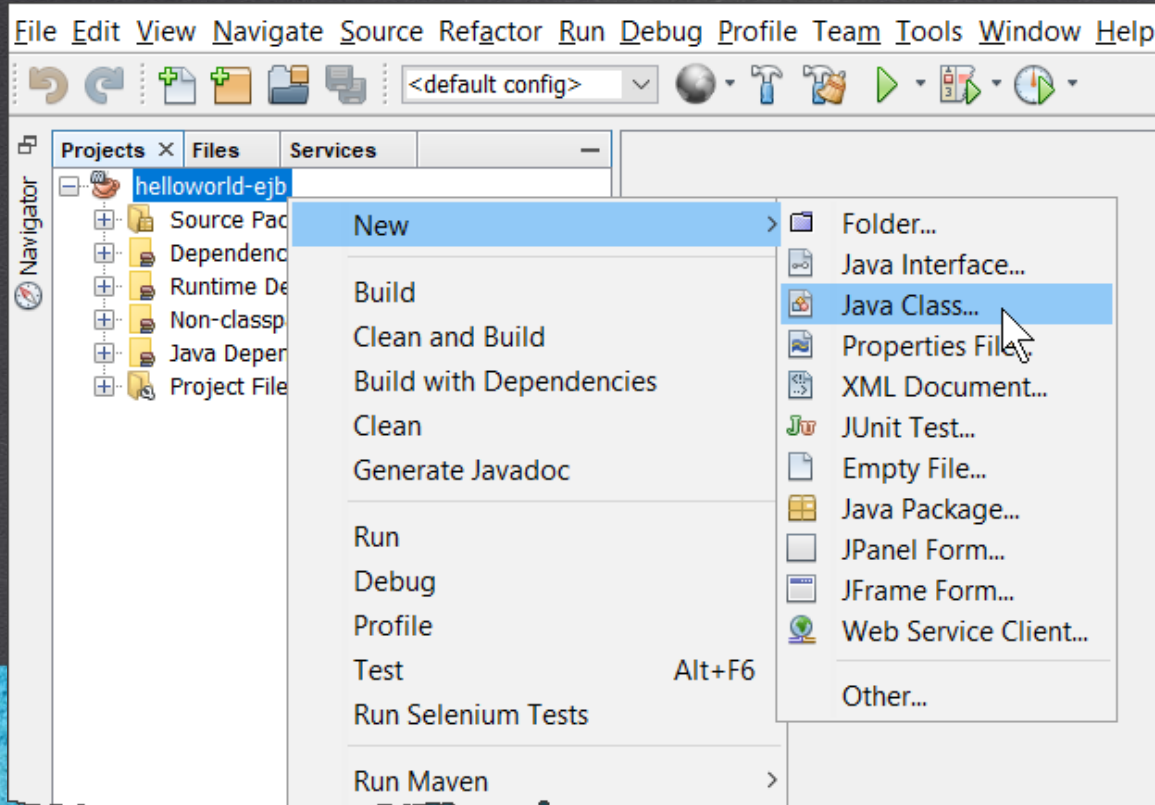
```
package beans;  
  
import javax.ejb.Remote;  
  
@Remote  
public interface HelloWorldRemoteEjb {  
    public int add(int a, int b);  
}
```

**JAVA EE COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)

## 6. CREATE A JAVA CLASS

We create the class HelloWorldEjbImpl.java:





## 6. CREATE A NEW CLASS

We create the class HelloWorldEjbImpl.java:

**New Java Class**

**Steps**

1. Choose File Type
2. **Name and Location**

**Name and Location**

Class Name: HelloWorldEjbImpl

Project: helloworld-ejb

Location: Source Packages

Package: beans

Created File: C:\Courses\JavaEE\Lesson01\helloworld-ejb\src\main\java\beans\HelloWorldEjbImpl.java

< Back Next > **Finish** Cancel Help

**JAVA EE COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)

# 7. MODIFY THE CODE

HelloWorldEjbImpl.java:

Click to download

```
package beans;

import javax.ejb.Stateless;

@Stateless
public class HelloWorldEjbImpl implements HelloWorldRemoteEjb{

    @Override
    public int add(int a, int b) {
        System.out.println("Executing add method on the server");
        return a + b;
    }
}
```

**JAVA EE COURSE**

www.globalmentoring.com.mx

## 8. GENERATE THE .JAR FILE

To be able to execute the EJB, it must be deployed in a Java server. In our case we will use Glassfish to deploy the EJB component that we have created.

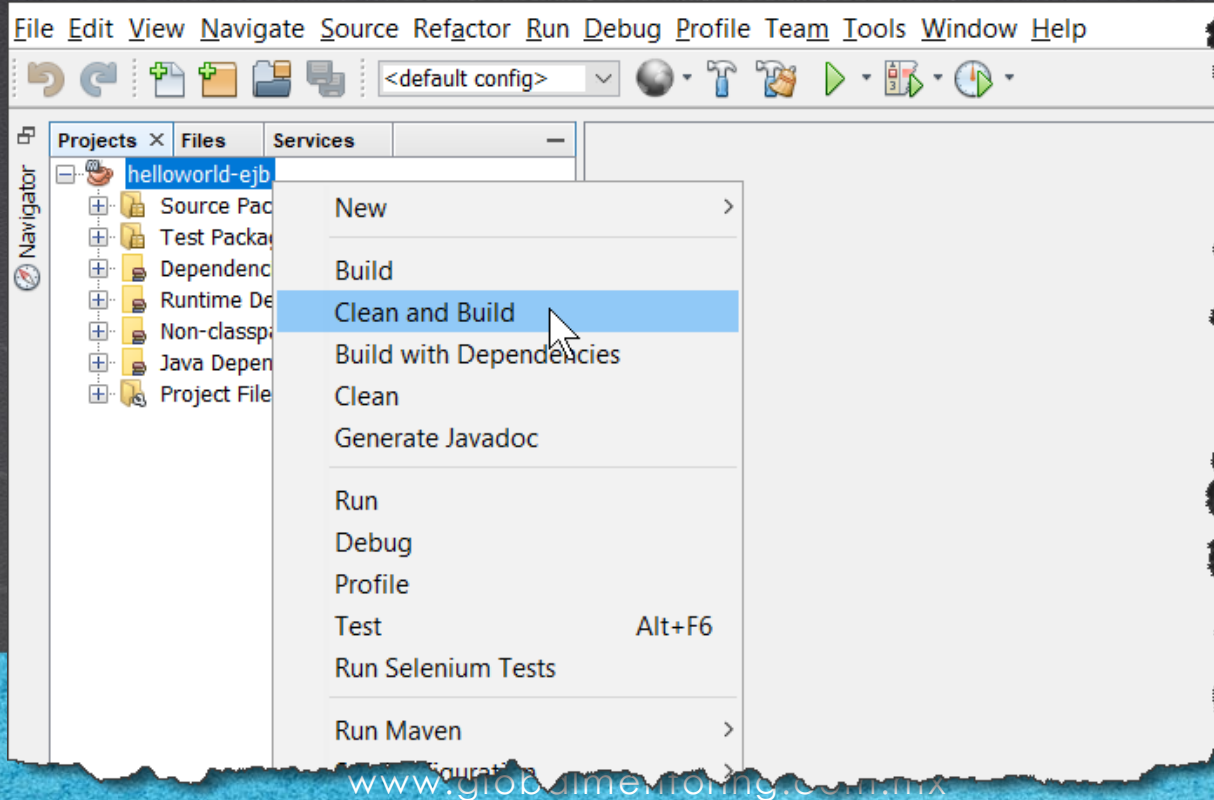
Later we will create an EJB client to access it. That is, through the application server is that an EJB receives all its power (transaction management, security, connection pool, etc), which is why we can not access it directly, but we will access it through the server of Glassfish in our case.

The Glassfish server must be stopped, and then we will start it.



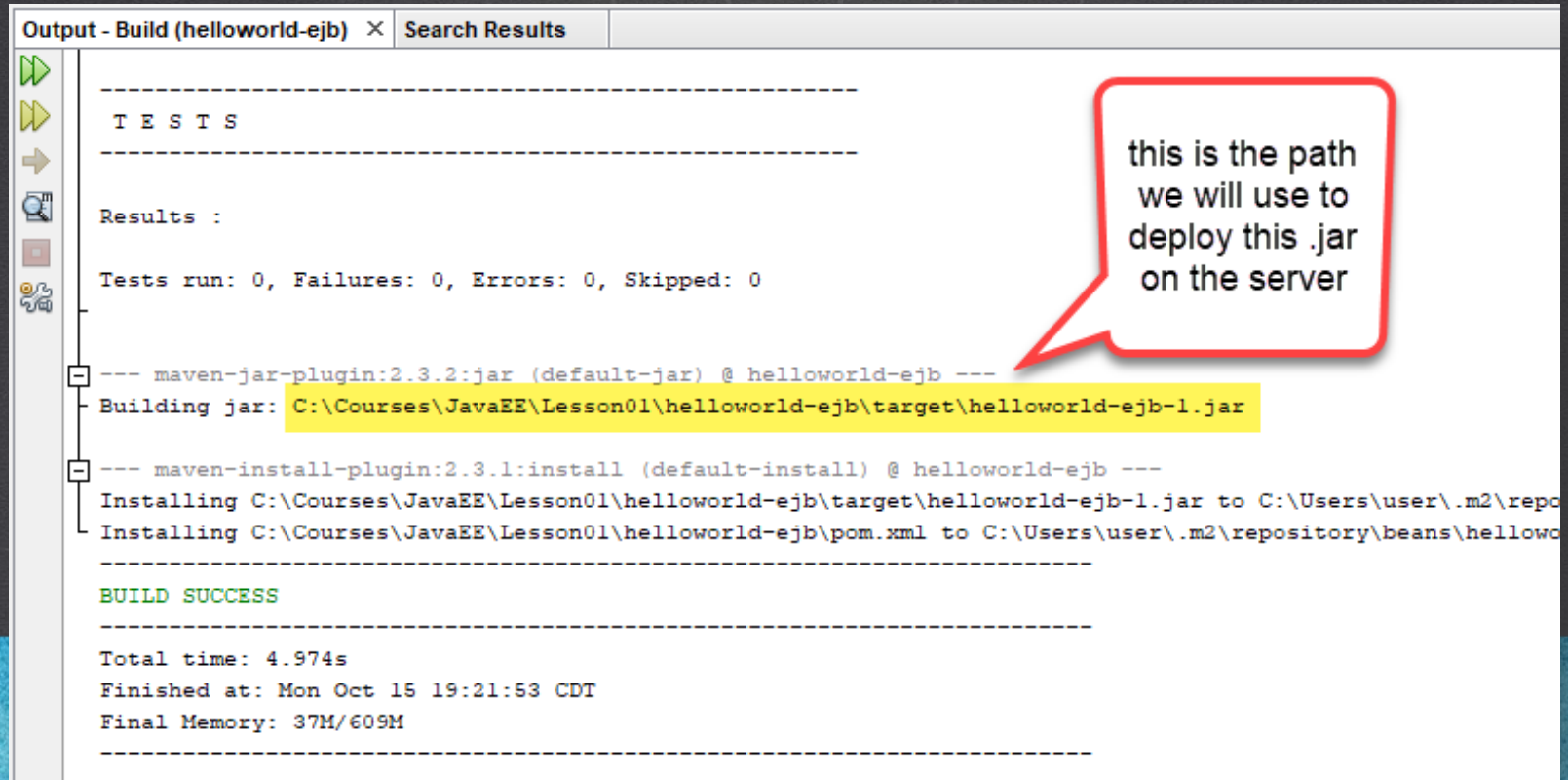
## 8. GENERATE THE .JAR FILE

We generate the project .jar, we do clean & build:



## 8. GENERATE THE .JAR FILE

We see that the helloworld-ejb.jar file was generated.



```
Output - Build (helloworld-ejb) × Search Results

-----
T E S T S
-----

Results :

Tests run: 0, Failures: 0, Errors: 0, Skipped: 0

--- maven-jar-plugin:2.3.2:jar (default-jar) @ helloworld-ejb ---
Building jar: C:\Courses\JavaEE\Lesson01\helloworld-ejb\target\helloworld-ejb-1.jar

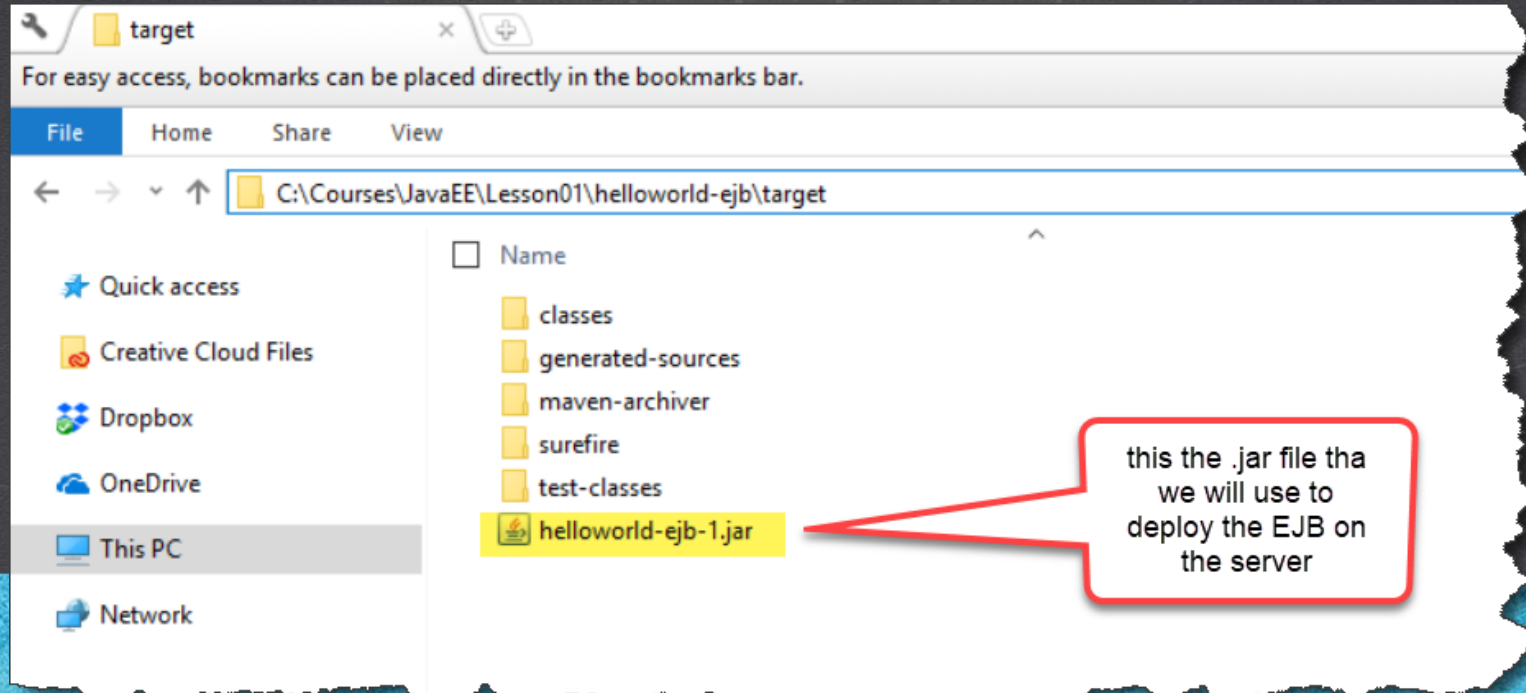
--- maven-install-plugin:2.3.1:install (default-install) @ helloworld-ejb ---
Installing C:\Courses\JavaEE\Lesson01\helloworld-ejb\target\helloworld-ejb-1.jar to C:\Users\user\.m2\repo
Installing C:\Courses\JavaEE\Lesson01\helloworld-ejb\pom.xml to C:\Users\user\.m2\repository\beans\hellowo

-----
BUILD SUCCESS
-----

Total time: 4.974s
Finished at: Mon Oct 15 19:21:53 CDT
Final Memory: 37M/609M
-----
```

## 8. GENERATE THE .JAR FILE

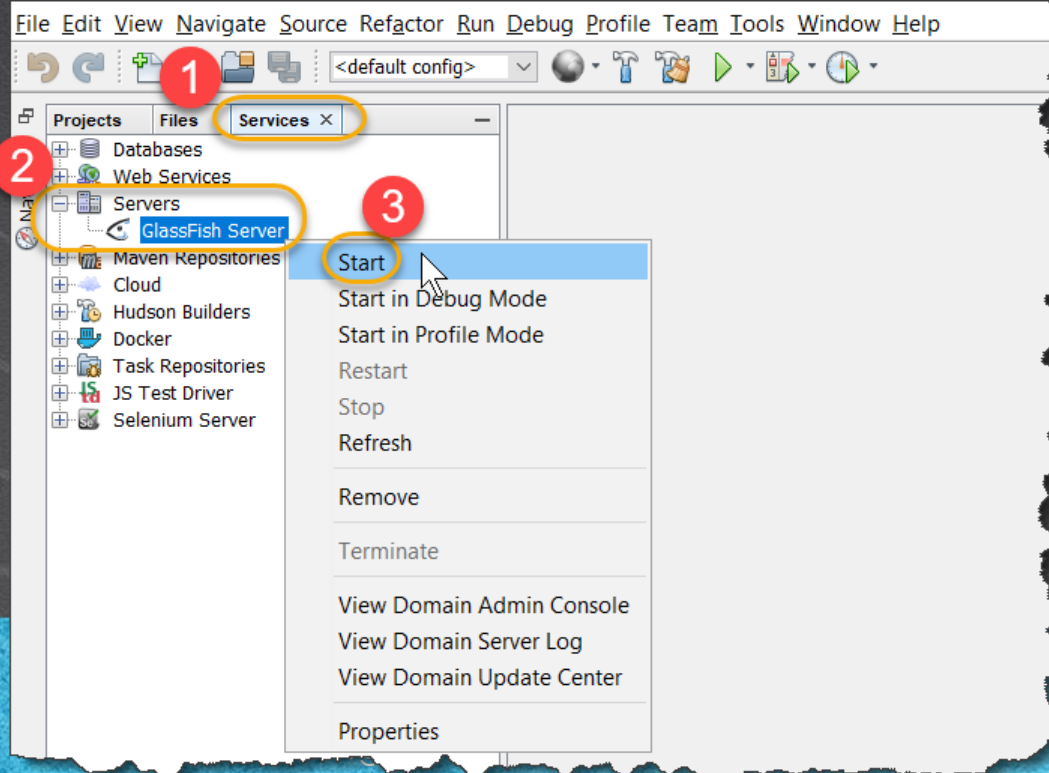
We see that the .jar was generated, we take the route and use it later: C:\Courses\JavaEE\Lesson01\helloworld-ejb\target





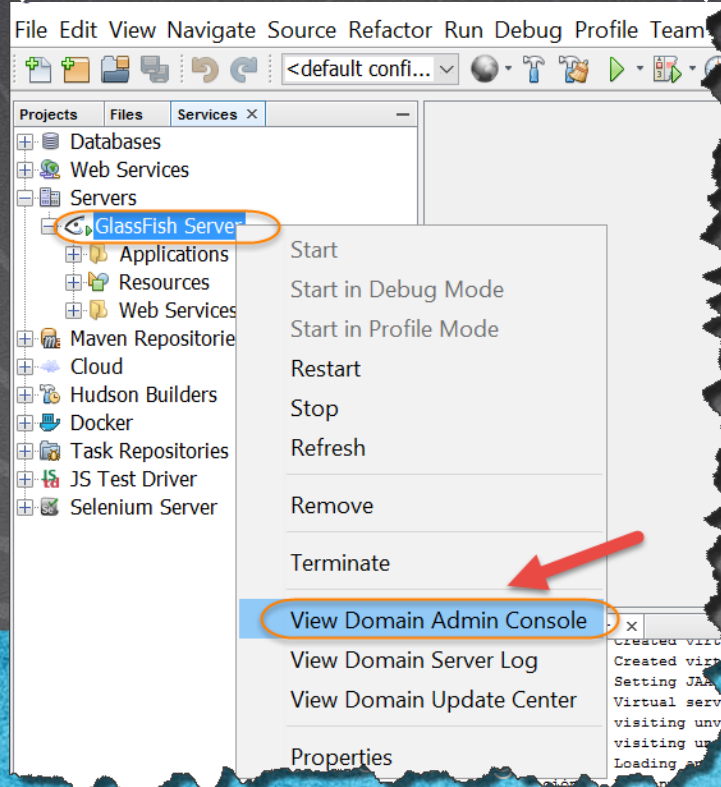
# 9. START UP GLASSFISH

We raise Glassfish to deploy the EJB. We are going to:  
Services -> Servers- Glassfish Server- Start



# 9. START UP GLASSFISH

Once started (it takes about a minute to boot), we enter the Glassfish management console (View Domain Admin Console):



# 10. DEPLOY THE APPLICATION

We deployed the .jar application in Glassfish. We go to:  
Applications -> Deploy:

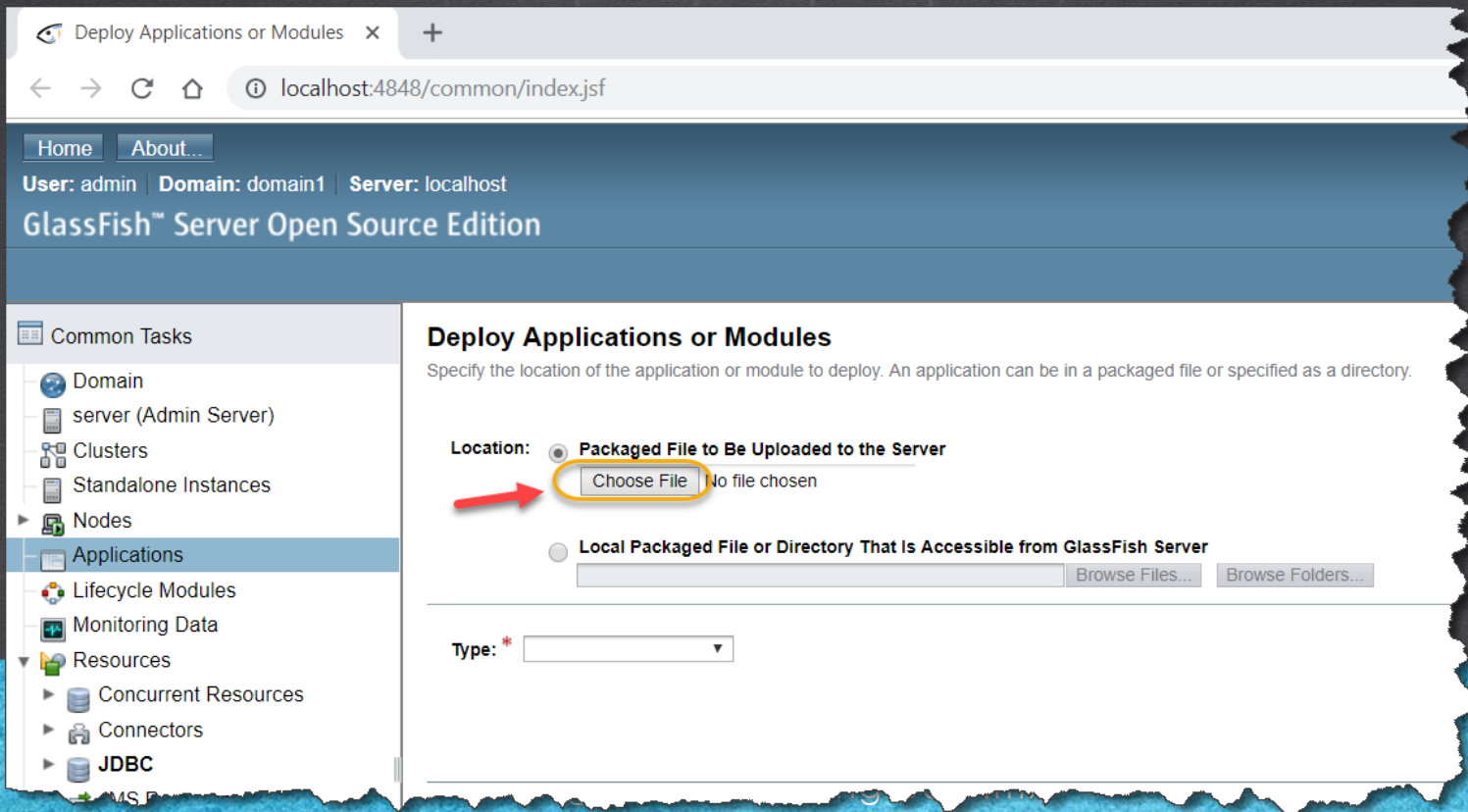
The screenshot shows the GlassFish Server Open Source Edition web console. The browser address bar indicates the URL is `localhost:4848/common/index.jsf`. The page title is "GlassFish™ Server Open Source Edition". The left sidebar contains a tree view of the console's navigation menu. The "Applications" item is selected, highlighted with a red circle and the number "1". The main content area is titled "Applications" and contains a description: "Applications can be enterprise or web applications, or various kinds of modules. Restart an application or module by clicking enabled on." Below this is a section titled "Deployed Applications (0)". This section contains a row of buttons: "Deploy...", "Undeploy", "Enable", and "Disable", followed by a "Filter:" dropdown. The "Deploy..." button is circled in yellow and pointed to by a red arrow, with a red circle and the number "2" next to it. Below the buttons is a table with the following structure:

Select	Name	Deployment Order	Enabled
No items found.			



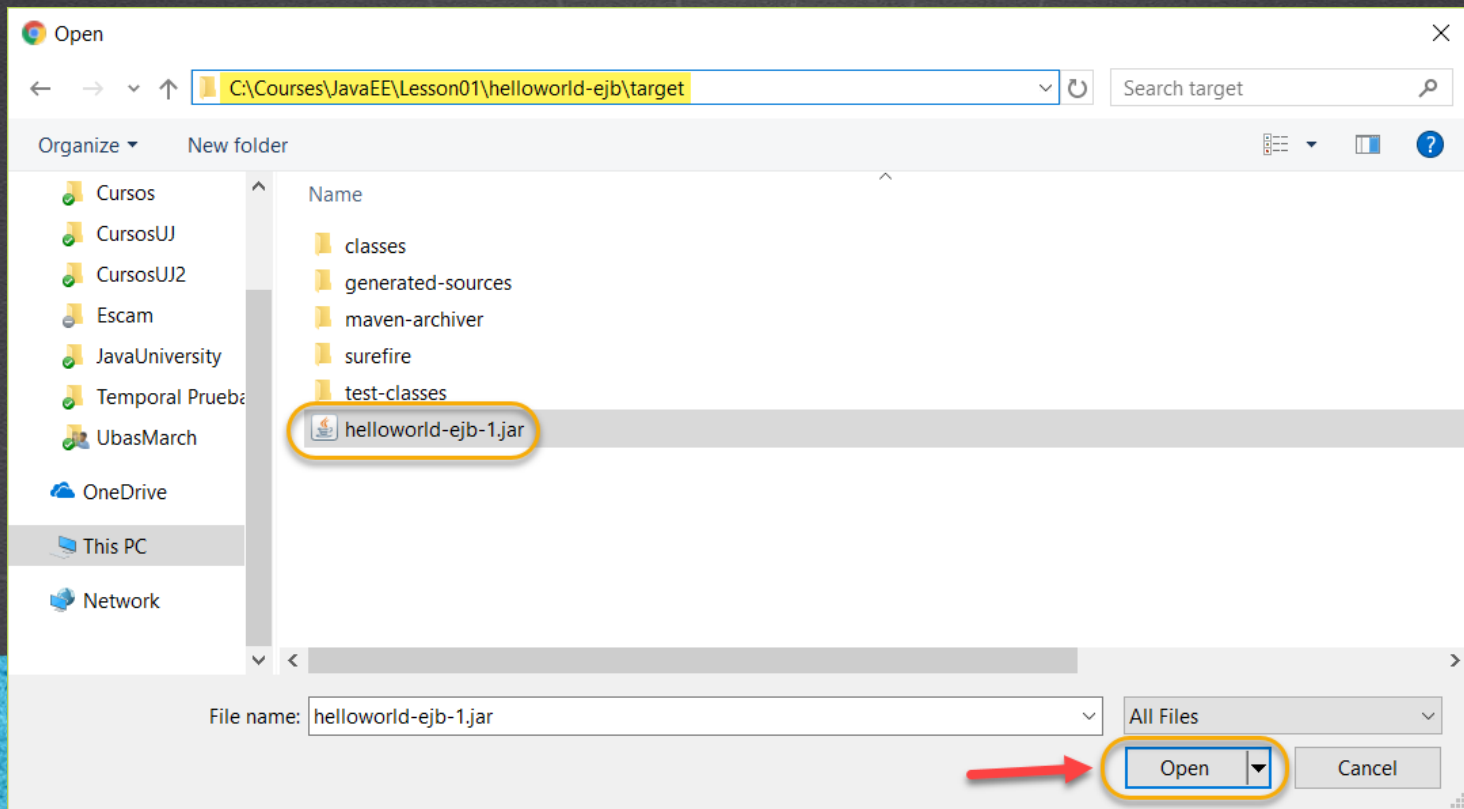
# 10. DEPLOY THE APPLICATION

We deployed the .jar application in Glassfish:



# 10. DEPLOY THE APPLICATION

Select the .jar file that were generated previously:



# 10. DEPLOY THE APPLICATION

Important: We change the name as shown and click on Ok:

Home About... Help

User: admin Domain: domain1 Server: localhost

GlassFish™ Server Open Source Edition

Tree

- Common Tasks
- Domain
  - server (Admin Server)
  - Clusters
  - Standalone Instances
  - Nodes
  - Applications**
  - Lifecycle Modules
  - Monitoring Data
  - Resources
    - Concurrent Resources
    - Connectors
    - JDBC
    - JMS Resources
    - JNDI
    - JavaMail Sessions

### Deploy Applications or Modules

Specify the location of the application or module to deploy. An application can be in a packaged file or specified as a directory.  
\* Indicates required field

Location: ☒ Packaged File to Be Uploaded to the Server

Choose File helloworld-ejb-1.jar

☐ Local Packaged File or Directory That Is Accessible from GlassFish Server

Browse Files... Browse Folders...

Type: \* EJB Jar

Application Name: \* helloworld-ejb

Status: ☒ Enabled  
Allows users to access the application.

Implicit CDI: ☒ Enabled  
Implicit discovery of CDI beans

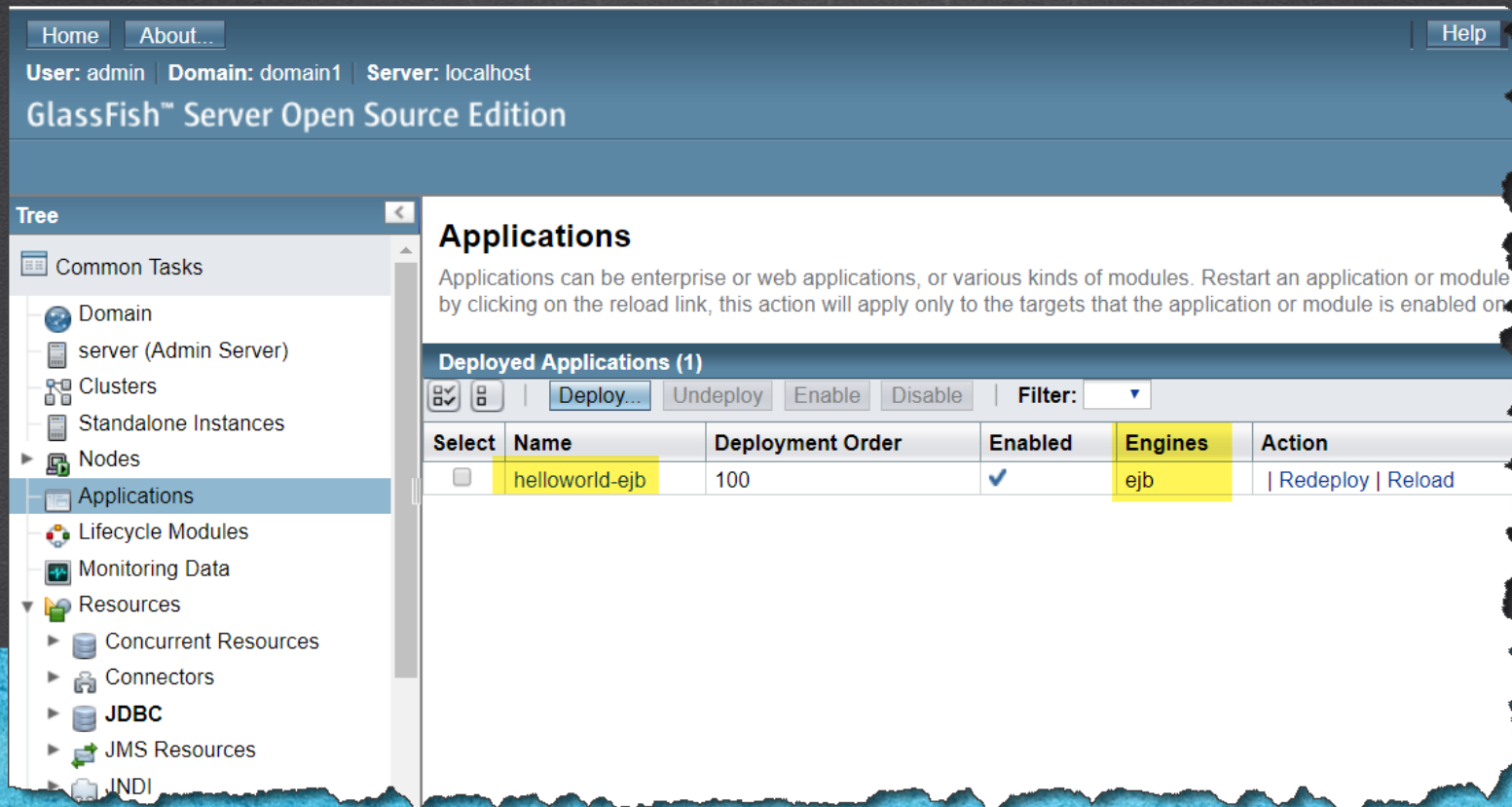
OK Cancel

Important: Change the name as show:



# 10. DEPLOY THE APPLICATION

We deployed the .jar application in Glassfish:



Home About... Help

User: admin Domain: domain1 Server: localhost

GlassFish™ Server Open Source Edition

**Tree**

- Common Tasks
- Domain
  - server (Admin Server)
  - Clusters
  - Standalone Instances
  - Nodes
  - Applications**
  - Lifecycle Modules
  - Monitoring Data
  - Resources
    - Concurrent Resources
    - Connectors
    - JDBC
    - JMS Resources
    - JNDI

**Applications**

Applications can be enterprise or web applications, or various kinds of modules. Restart an application or module by clicking on the reload link, this action will apply only to the targets that the application or module is enabled on.

**Deployed Applications (1)**

☒ ☐ Deploy... Undeploy Enable Disable Filter: ▼

Select	Name	Deployment Order	Enabled	Engines	Action
<input type="checkbox"/>	helloworld-ejb	100	✓	ejb	<a href="#">Redeploy</a>   <a href="#">Reload</a>

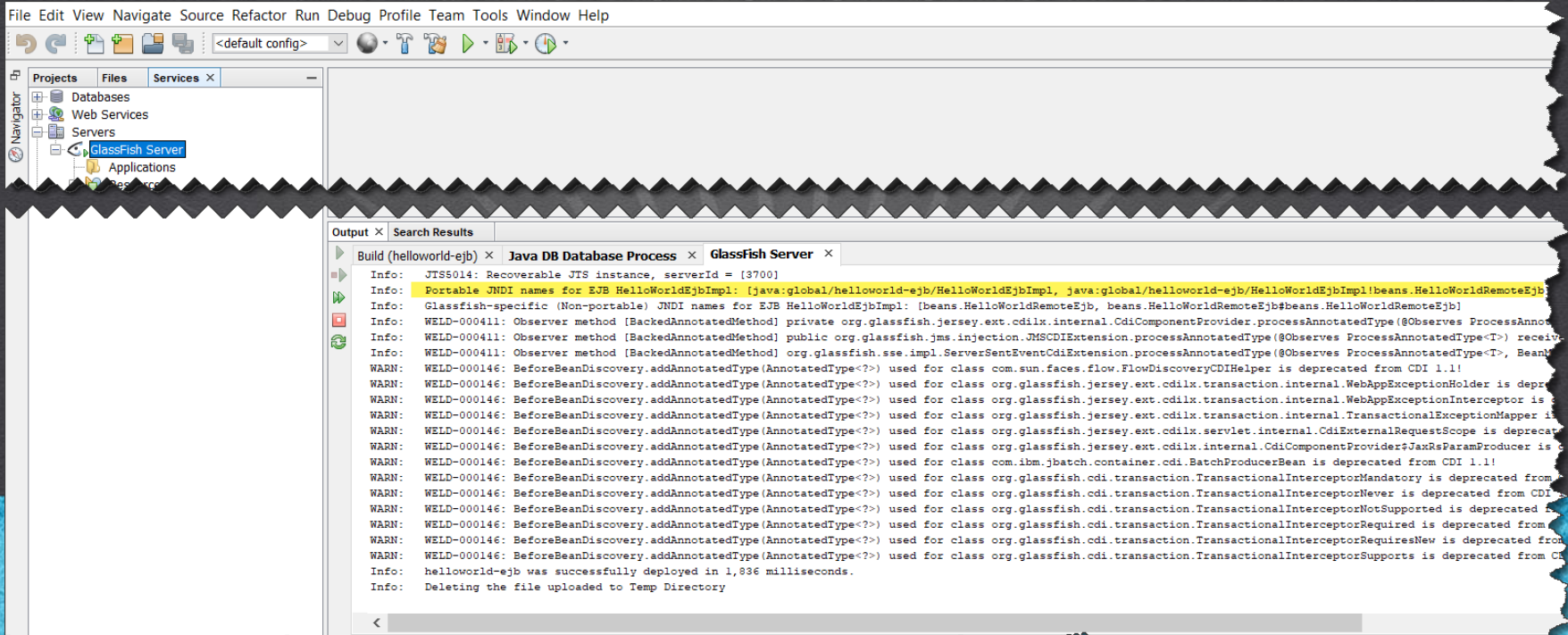
# 11. OBSERVED THE NAMES OF THE EJB

We observe in the log of Glassfish, the names as we can use from a client and thus obtain a reference of the EJB of type Stateless that we have just deployed in Glassfish:

Portable JNDI names for EJB HelloWorldEjbImpl:

java:global/helloworld-ejb/HelloWorldEjbImpl

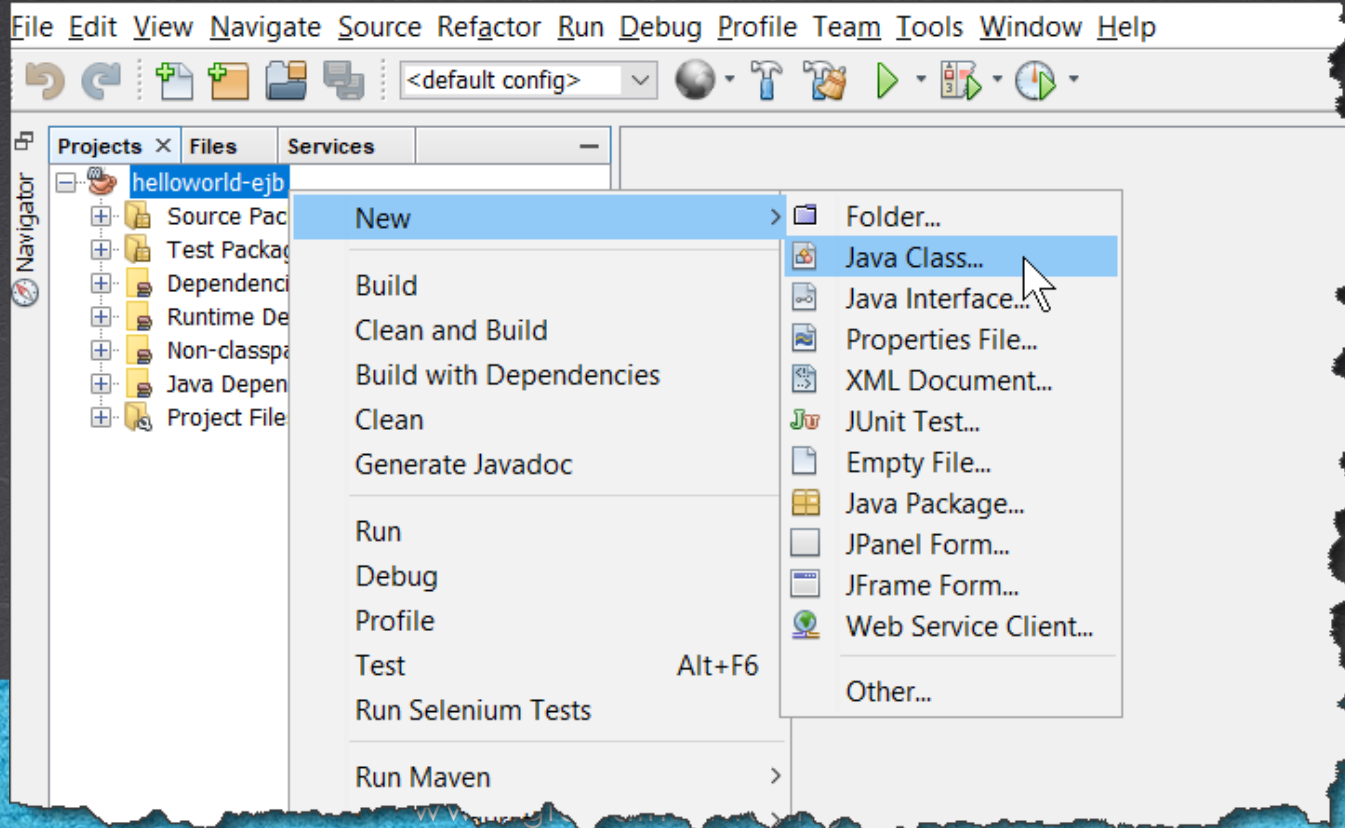
java:global/helloworld-ejb/HelloWorldEjbImpl!beans.HelloWorldRemoteEjb



```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
<default config>
Projects Files Services
Databases
Web Services
Servers
GlassFish Server
Applications
Resources
Output Search Results
Build (helloworld-ejb) Java DB Database Process GlassFish Server
Info: JTSS014: Recoverable JTS instance, serverId = {3700}
Info: Portable JNDI names for EJB HelloWorldEjbImpl: [java:global/helloworld-ejb/HelloWorldEjbImpl, java:global/helloworld-ejb/HelloWorldEjbImpl!beans.HelloWorldRemoteEjb]
Info: Glassfish-specific (Non-portable) JNDI names for EJB HelloWorldEjbImpl: [beans.HelloWorldRemoteEjb, beans.HelloWorldRemoteEjb!beans.HelloWorldRemoteEjb]
Info: WELD-000411: Observer method [BackedAnnotatedMethod] private org.glassfish.jersey.ext.cdilx.internal.CdiComponentProvider.processAnnotatedType(@Observes ProcessAnno
Info: WELD-000411: Observer method [BackedAnnotatedMethod] public org.glassfish.jms.injection.JMSCDIExtension.processAnnotatedType(@Observes ProcessAnnotatedType<T>) receiv
Info: WELD-000411: Observer method [BackedAnnotatedMethod] org.glassfish.sse.impl.ServerSentEventCdiExtension.processAnnotatedType(@Observes ProcessAnnotatedType<T>, Bean
WARN: WELD-000146: BeforeBeanDiscovery.addAnnotatedType(AnnotatedType<?>) used for class com.sun.faces.flow.FlowDiscoveryCDIHelper is deprecated from CDI 1.1!
WARN: WELD-000146: BeforeBeanDiscovery.addAnnotatedType(AnnotatedType<?>) used for class org.glassfish.jersey.ext.cdilx.transaction.internal.WebAppExceptionHandler is depre
WARN: WELD-000146: BeforeBeanDiscovery.addAnnotatedType(AnnotatedType<?>) used for class org.glassfish.jersey.ext.cdilx.transaction.internal.WebAppExceptionHandler is depre
WARN: WELD-000146: BeforeBeanDiscovery.addAnnotatedType(AnnotatedType<?>) used for class org.glassfish.jersey.ext.cdilx.transaction.internal.TransactionalExceptionMapper is depre
WARN: WELD-000146: BeforeBeanDiscovery.addAnnotatedType(AnnotatedType<?>) used for class org.glassfish.jersey.ext.cdilx.servlet.internal.CdiExternalRequestScope is deprecate
WARN: WELD-000146: BeforeBeanDiscovery.addAnnotatedType(AnnotatedType<?>) used for class org.glassfish.jersey.ext.cdilx.internal.CdiComponentProvider$JaxRsParamProducer is depre
WARN: WELD-000146: BeforeBeanDiscovery.addAnnotatedType(AnnotatedType<?>) used for class com.ibm.jbatch.container.cdi.BatchProducerBean is deprecated from CDI 1.1!
WARN: WELD-000146: BeforeBeanDiscovery.addAnnotatedType(AnnotatedType<?>) used for class org.glassfish.cdi.transaction.TransactionalInterceptorMandatory is deprecated from CDI 1.1!
WARN: WELD-000146: BeforeBeanDiscovery.addAnnotatedType(AnnotatedType<?>) used for class org.glassfish.cdi.transaction.TransactionalInterceptorNever is deprecated from CDI 1.1!
WARN: WELD-000146: BeforeBeanDiscovery.addAnnotatedType(AnnotatedType<?>) used for class org.glassfish.cdi.transaction.TransactionalInterceptorNotSupported is deprecated from CDI 1.1!
WARN: WELD-000146: BeforeBeanDiscovery.addAnnotatedType(AnnotatedType<?>) used for class org.glassfish.cdi.transaction.TransactionalInterceptorRequired is deprecated from CDI 1.1!
WARN: WELD-000146: BeforeBeanDiscovery.addAnnotatedType(AnnotatedType<?>) used for class org.glassfish.cdi.transaction.TransactionalInterceptorRequiresNew is deprecated from CDI 1.1!
WARN: WELD-000146: BeforeBeanDiscovery.addAnnotatedType(AnnotatedType<?>) used for class org.glassfish.cdi.transaction.TransactionalInterceptorSupports is deprecated from CDI 1.1!
Info: helloworld-ejb was successfully deployed in 1,836 milliseconds.
Info: Deleting the file uploaded to Temp Directory
```

# 12. CREATE A JAVA CLASS

We create the TestHelloWorldEJB test class:





# 12. CREATE A NEW JAVA CLASS

We create the TestHelloWorldEJB test class:

**New Java Class**

**Steps**

1. Choose File Type
2. **Name and Location**

**Name and Location**

Class Name:

Project:

Location:

Package:

Created File:

< Back   Next >   **Finish**   Cancel   Help

**JAVA EE COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)

# 13. MODIFY THE FILE

## TestHelloWorldEjb.java:

Click to download

```
package test;

import beans.HelloWorldRemoteEjb;
import javax.naming.*;

public class TestHelloWorldEjb {

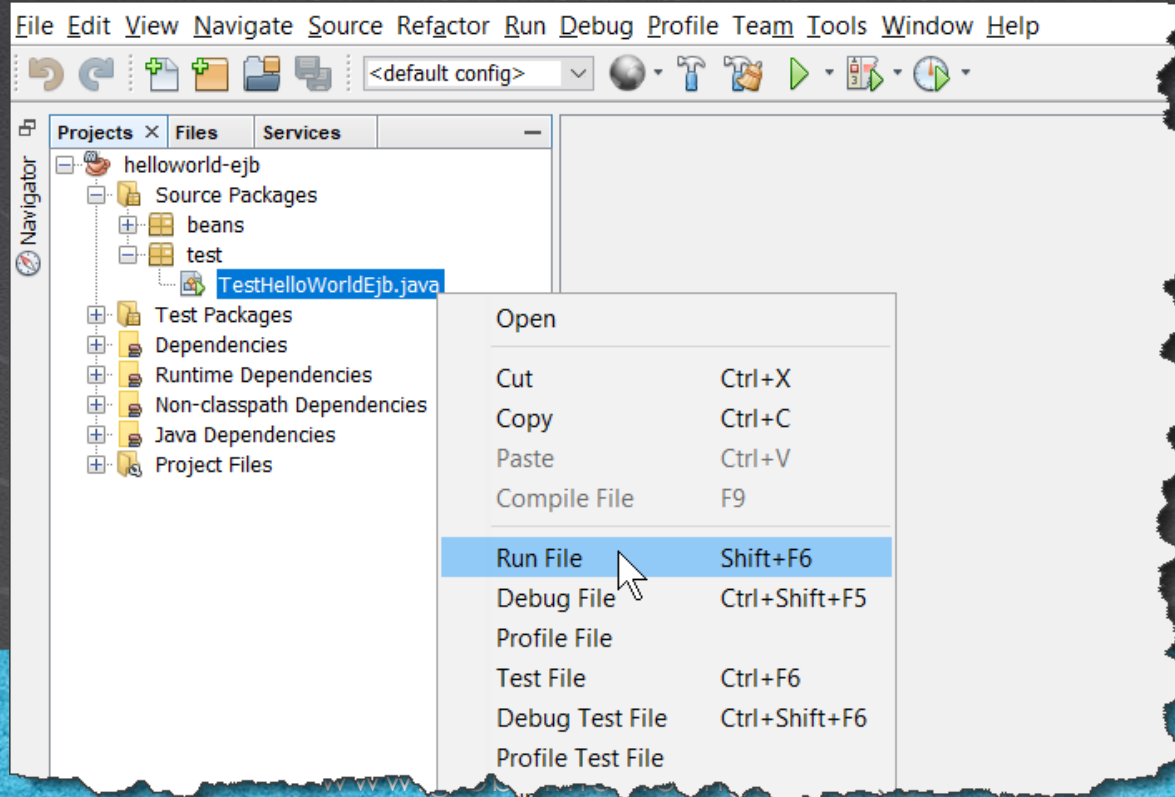
    public static void main(String[] args) {
        System.out.println("Initiating EJB call from the client\n");
        try {
            Context jndi = new InitialContext();
            HelloWorldRemoteEjb helloWorldEjb =
                (HelloWorldRemoteEjb) jndi.lookup("java:global/helloworld-ejb/HelloWorldEjbImpl!beans.HelloWorldRemoteEjb");
            int result = helloWorldEjb.add(5, 3);
            System.out.println("EJB Result of adding 5 + 3 = " + result);
        } catch (NamingException e) {
            e.printStackTrace(System.out);
        }
    }
}
```

**JAVA EE COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)

# 14. EXECUTE THE CLASS

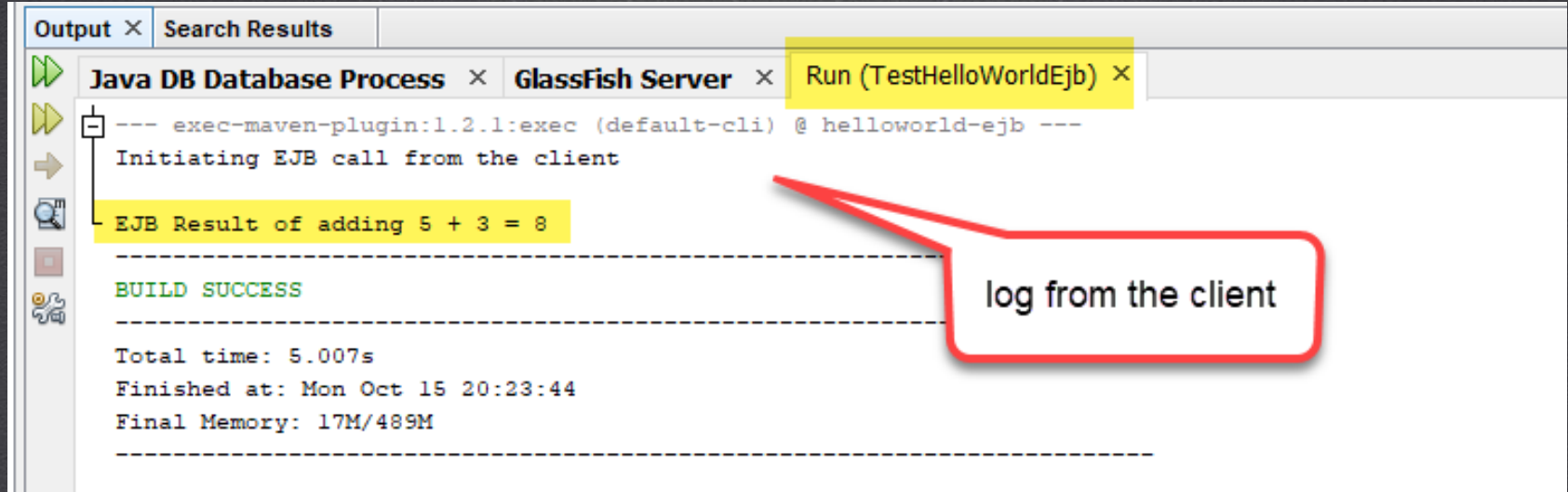
Execute the TestHelloWorldEjb.java. Glassfish must be running:





# 14. EXECUTE THE PROJECT

We execute the project. We see the result of executing the call to the EJB in the log:



```
--- exec-maven-plugin:1.2.1:exec (default-cli) @ helloworld-ejb ---
Initiating EJB call from the client
EJB Result of adding 5 + 3 = 8
-----
BUILD SUCCESS
-----
Total time: 5.007s
Finished at: Mon Oct 15 20:23:44
Final Memory: 17M/489M
-----
```

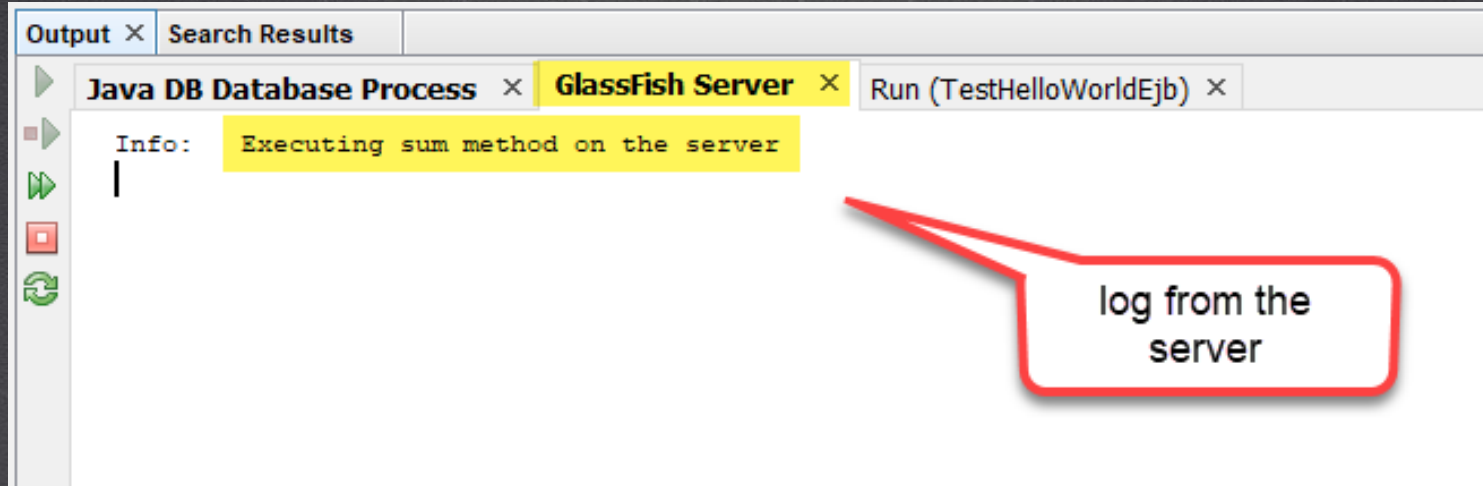
log from the client

**JAVA EE COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)

# 14. EXECUTE THE PROJECT

We execute the project. We see the result of executing the call to the EJB in the log:





# OBSERVATIONS IN CASE OF PROBLEMS

If for some reason the execution of the project does not work, we recommend you perform the following actions:

- 1) The Glassfish server must be running in order to access the remote EJB.
- 2) Remember that any change in the EJB or the interface has to redeploy the .jar file in Glassfish server, since the changes are not published automatically (Select the application and do undeploy and redo deploy in Glassfish)
- 3) Check that in step 10 the application was renamed at the time of deploying it in Glassfish (helloworld-ejb).
- 4) If none of this works, try loading the resolved project, which is 100% functional.



# EXERCISE CONCLUSION

- With this exercise we have put into practice the Hello World with EJB's.
- We use several technologies to simplify this exercise such as Maven and Glassfish Java application server.
- These are some of the tools that we will be using throughout the course, and as we move forward we will become familiar with each of them.



Experiencia y Conocimiento para tu vida

**JAVA EE COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)

**ONLINE**

# **JAVA EE**

By: Eng. Ubaldo Acosta



**JAVA EE COURSE**

[www.globalmentoring.com.mx](http://www.globalmentoring.com.mx)