

# Continuous delivery pipeline configuration

This file explains how to configure a continuous delivery pipeline for our project using NetBeans, Jenkins, Maven and Glassfish.

We will create three Glassfish servers that represent development, testing and production environments, our application will be deployed to these servers.

## System configuration

The following environment variables must be configured (the paths may be a bit different):

- GLASSFISH\_HOME = C:\Program Files\glassfish-4.1.1
- MAVEN\_HOME = C:\Program Files\NetBeans 8.2\java\maven

Also add the followings to your Path variable:

- %GLASSFISH\_HOME%\bin
- %MAVEN\_HOME%\bin

## Creating GlassFish domains

A GlassFish domain must be created for each environment.

To create the development domain, open the console and use the following command (**change the path to the location where you want to create the domains**):

```
asadmin create-domain --domaindir "C:\Users\vinro\OneDrive\Documents\Efrei\M1\Java EE\Glassfish domains" --adminport 5050 --instanceport 5051 development
```

When prompted, enter "admin" for both user name and password.

```

PS C:\> asadmin create-domain --domainid "C:\Users\vinro\OneDrive\Documents\Efrei\M1\Java EE\Glassfish domains" --
adminport 5050 --instanceport 5051 development
Enter admin user name [Enter to accept default "admin" / no password]>admin
Enter the admin password [Enter to accept default of no password]>
Enter the admin password again>
Using port 5050 for Admin.
Using port 5051 for HTTP Instance.
Using default port 7676 for JMS.
Using default port 3700 for IIOP.
Using default port 8181 for HTTP_SSL.
Using default port 3820 for IIOP_SSL.
Using default port 3920 for IIOP_MUTUALAUTH.
Using default port 8686 for JMX_ADMIN.
Using default port 6666 for OSGI_SHELL.
Using default port 9009 for JAVA_DEBUGGER.
Distinguished Name of the self-signed X.509 Server Certificate is:
[CN=VINCENT-ASUS,OU=GlassFish,O=Oracle Corporation,L=Santa Clara,ST=California,C=US]
Distinguished Name of the self-signed X.509 Server Certificate is:
[CN=VINCENT-ASUS-instance,OU=GlassFish,O=Oracle Corporation,L=Santa Clara,ST=California,C=US]
Domain development created.
Domain development admin port is 5050.
Domain development admin user is "admin".
Command create-domain executed successfully.

```

The development domain is now created.

Use the same command to create domains for testing and production environments:

```

asadmin create-domain --domainid "C:\Users\vinro\OneDrive\Documents\Efrei\M1\Java
EE\Glassfish domains" --adminport 6060 --instanceport 6061 testing

```

```

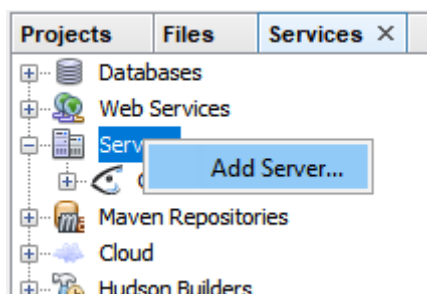
asadmin create-domain --domainid "C:\Users\vinro\OneDrive\Documents\Efrei\M1\Java
EE\Glassfish domains" --adminport 7070 --instanceport 7071 production

```

## GlassFish server configuration

We will first create the development server.

In NetBean's Services tab, right click "Servers", and add a server:



Select "GlassFish Server", give your server a name such as "GlassFish Server (development)", and click "Next":

Add Server Instance

×

Steps

1. Choose Server

2. ...

Choose Server

Server: Apache Tomcat or TomEE  
GlassFish Server  
JBoss Application Server  
Oracle WebLogic Server  
WildFly Application Server

Name: GlassFish Server (development)

< BackNext >FinishCancelHelp

On the next page, set your GlassFish installation location and click “Next”:

Add Server Instance

×

Steps

1. Choose Server

2. Server Location

3. Domain Name/Location

Server Location

Installation Location:  
C:\Program Files\glassfish-4.1.1Browse...  
☒ Local Domain☐ Remote Domain  
Download Now...☐ I have read and accept the license agreement...  
(click)  

⚠ No usable default domain. Use Next to create a personal domain.

< BackNext >FinishCancelHelp

On the next page, set the path to the GlassFish domain you created before, set “DAS Port” to 5050 and “HTTP Port” to 5051:

Add Server Instance

1. Choose Server
2. Server Location
3. **Domain Name/Location**

**Domain Location**

Domain: C:\Users\vinro\OneDrive\Documents\Efrei\M1\Java EE\Glassfish domains\development
Host: localhost
DAS Port: 5050
HTTP Port: 5051
Target:
User Name:
Password:

☒ Loopback
☐ Default

Register existing domain: C:\Users\vinro\OneDrive\Documents\Efrei\M1\Java EE\Glassfish domains\deve

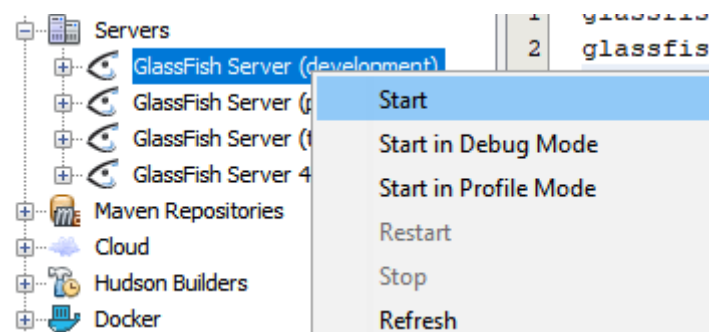
< Back
Next >
**Finish**
Cancel
Help

Click finish and a new GlassFish server will be created.

Repeat these steps to create two other servers for the testing and production environments. Use these settings:

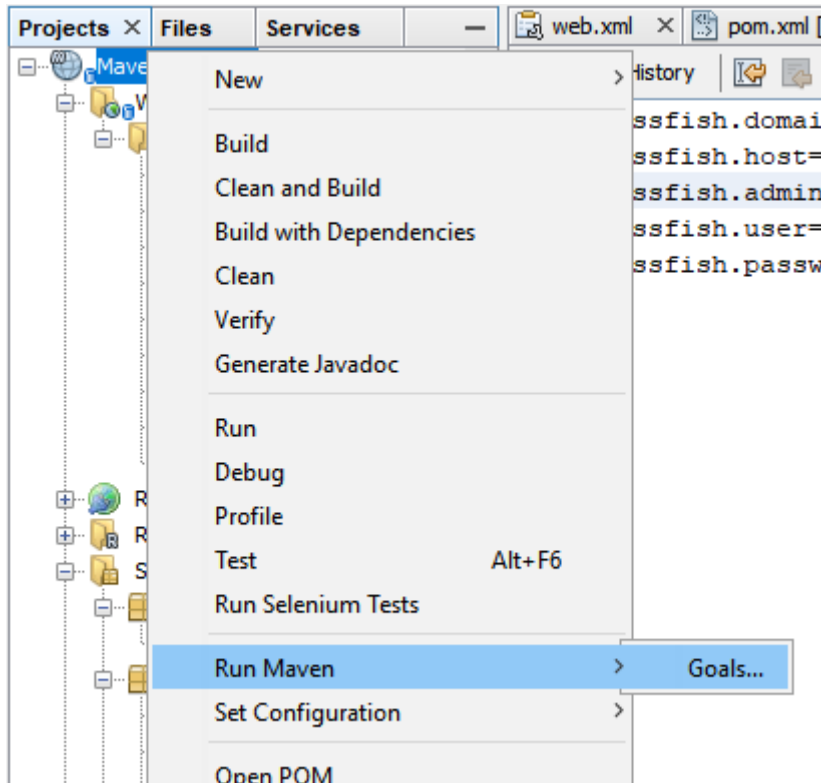
- Testing server:
  - Name: “testing”
  - DAS Port: 6060
  - HTTP Port: 6061
- Production server:
  - Name: “production”
  - DAS Port: 7070
  - HTTP Port: 7071

We will try the development server. Right-click it and select “Start”:



If prompted for credentials, leave the fields empty.

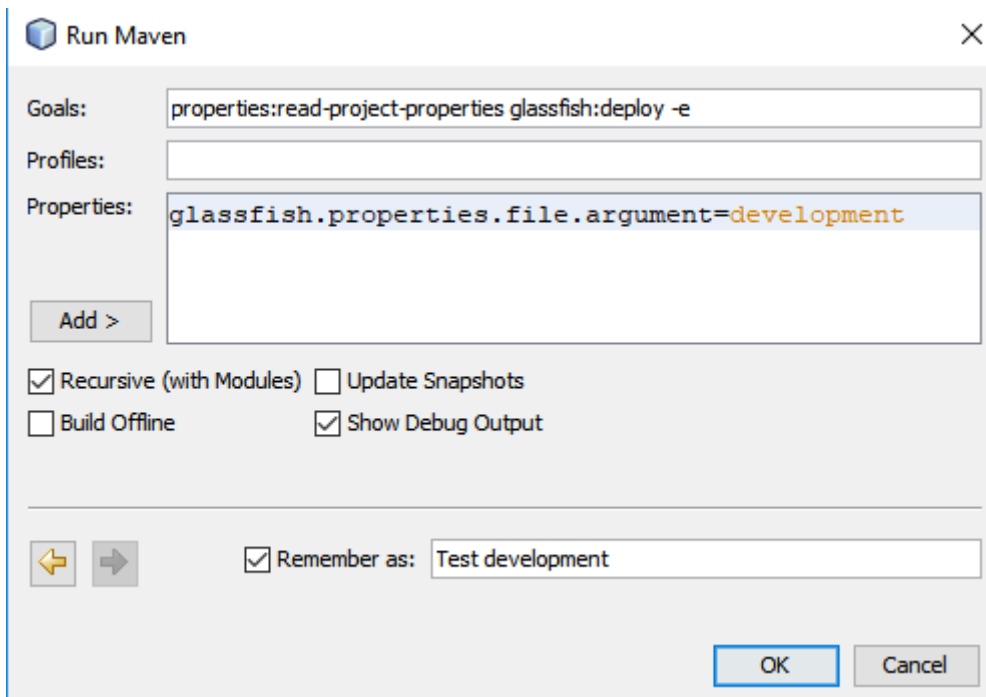
Test your configuration by running a Maven Goal:



Write the following and click OK:

Goals: *properties:read-project-properties glassfish:deploy -e*

Properties: *glassfish.properties.file.argument=development*



After the build ends, connect to your server console at <http://localhost:5050/> and log in using “admin” as both user name and password.

Click “List Deployed Applications”:



Then, click “Launch” on the right of “MavenPROJECT”:

## 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)					
<div><input type="checkbox"/> <input type="checkbox"/>   <a href="#">Deploy...</a> <a href="#">Undeploy</a> <a href="#">Enable</a> <a href="#">Disable</a>   Filter: <input type="text"/></div>					
Select	Name	Deployment Order	Enabled	Engines	Action
<input type="checkbox"/>	<a href="#">MavenPROJECT</a>	100	✓	ejb, web	<a href="#">Launch</a> <a href="#">Redeploy</a>   <a href="#">Reload</a>

GlassFish gives you two links to visit the applications, click on the first one and you should be redirected to the employee management application:

## Web Application Links

If the server or listener is not running, the link may not work. In this event, check the status of the server instance.

**Application Name:** MavenPROJECT

**Links:**  
[server] <http://VINCENT-ASUS:5051/MavenPROJECT-1.0-SNAPSHOT>  
[server] <https://VINCENT-ASUS:8181/MavenPROJECT-1.0-SNAPSHOT>

## Jenkins configuration

### Plugins

Make sure the following plugins are installed:

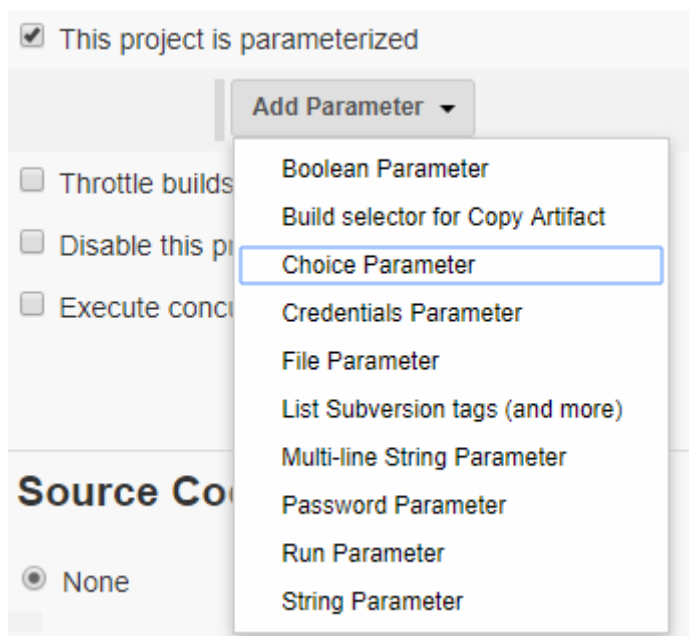
- Build with parameters
- Copy artifact
- GIT plugin
- Parameterized trigger plugin
- Maven integration plugin
- GitHub plugin

### Configure the Jenkins job


Create a new job, give it a name, and select “Freestyle project”.

To select the environment we want to deploy the project to, we add a parameter to the build.

Under “General”, check the “This project is parameterized option” and add a “Choice Parameter”:



Configure the parameter:



The image shows the 'Choice Parameter' configuration window in Jenkins. It has a title bar with a red close button and a help icon. The window contains three main sections: 'Name' with a text field containing 'environment', 'Choices' with a text area containing 'development', 'testing', and 'production', and 'Description' with a text area containing 'Which GlassFish domain to deploy the project to.'. At the bottom, there is a '[Plain text] Preview' link.

Choice Parameter

Name: environment

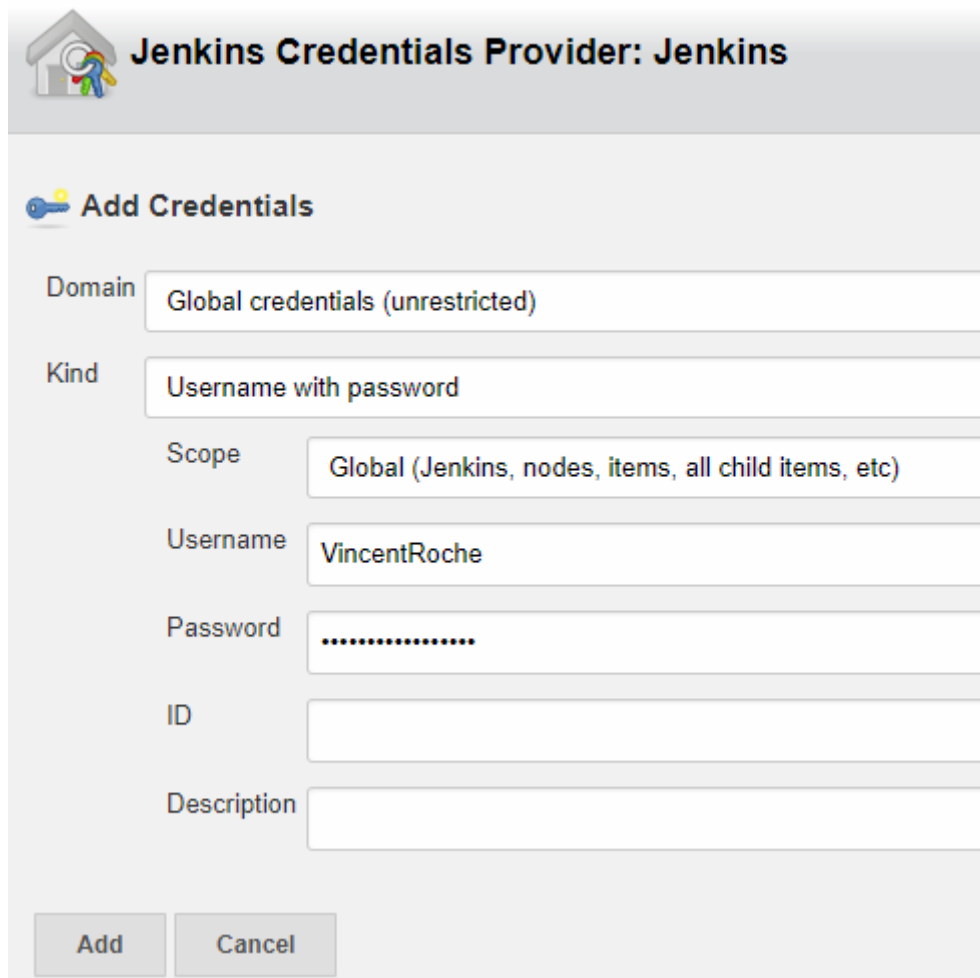
Choices: development, testing, production

Description: Which GlassFish domain to deploy the project to.

[Plain text] Preview

In the “Source Code Management” section, select “Git”.

Set <https://github.com/ClaireH97/JEE2.git> as the repository URL, and click Add → Jenkins to add your GitHub credentials:



The image shows the 'Jenkins Credentials Provider: Jenkins' window. It has a title bar with a house icon and a key icon. The main section is titled 'Add Credentials'. It contains several fields: 'Domain' with a dropdown menu showing 'Global credentials (unrestricted)', 'Kind' with a dropdown menu showing 'Username with password', 'Scope' with a dropdown menu showing 'Global (Jenkins, nodes, items, all child items, etc)', 'Username' with a text field containing 'VincentRoche', 'Password' with a text field containing a series of dots, 'ID' with an empty text field, and 'Description' with an empty text field. At the bottom, there are 'Add' and 'Cancel' buttons.

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain: Global credentials (unrestricted)

Kind: Username with password

Scope: Global (Jenkins, nodes, items, all child items, etc)

Username: VincentRoche

Password: .....

ID:

Description:

Add Cancel

Select the credentials you just created in the dropdown menu and no error should be displaying:



Git

Repositories

Repository URL

Credentials

Advanced...

Add Repository

Branches to build

Branch Specifier (blank for 'any')

Add Branch

Repository browser

Additional Behaviours

In the “Build” section, click “Add build step” and select “Invoke top-level Maven targets”:

**Build**

Add build step ▾

- Conditional step (single)
- Conditional steps (multiple)
- Copy artifacts from another project
- Execute Windows batch command
- Execute shell
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets**
- Run with timeout
- Set build status to "pending" on GitHub commit
- Trigger/call builds on other projects

Click “Advanced...” to view all the settings, and give them these values:

Goals: *properties:read-project-properties glassfish:redploy -e*

POM: *MavenPROJECT\pom.xml*

Properties: *glassfish.properties.file.argument=\${environment}*

Invoke top-level Maven targets

Goals

properties:read-project-properties glassfish:redeploy -e

POM

MavenPROJECT\pom.xml

Properties

glassfish.properties.file.argument=\${environment}

JVM Options

Inject build variables

☐

Use private Maven repository

☐

Settings file

Use default maven settings

Global Settings file

Use default maven global settings

The Jenkins job is now configured! Save and you can now build and deploy the project by clicking “Build with parameters”.