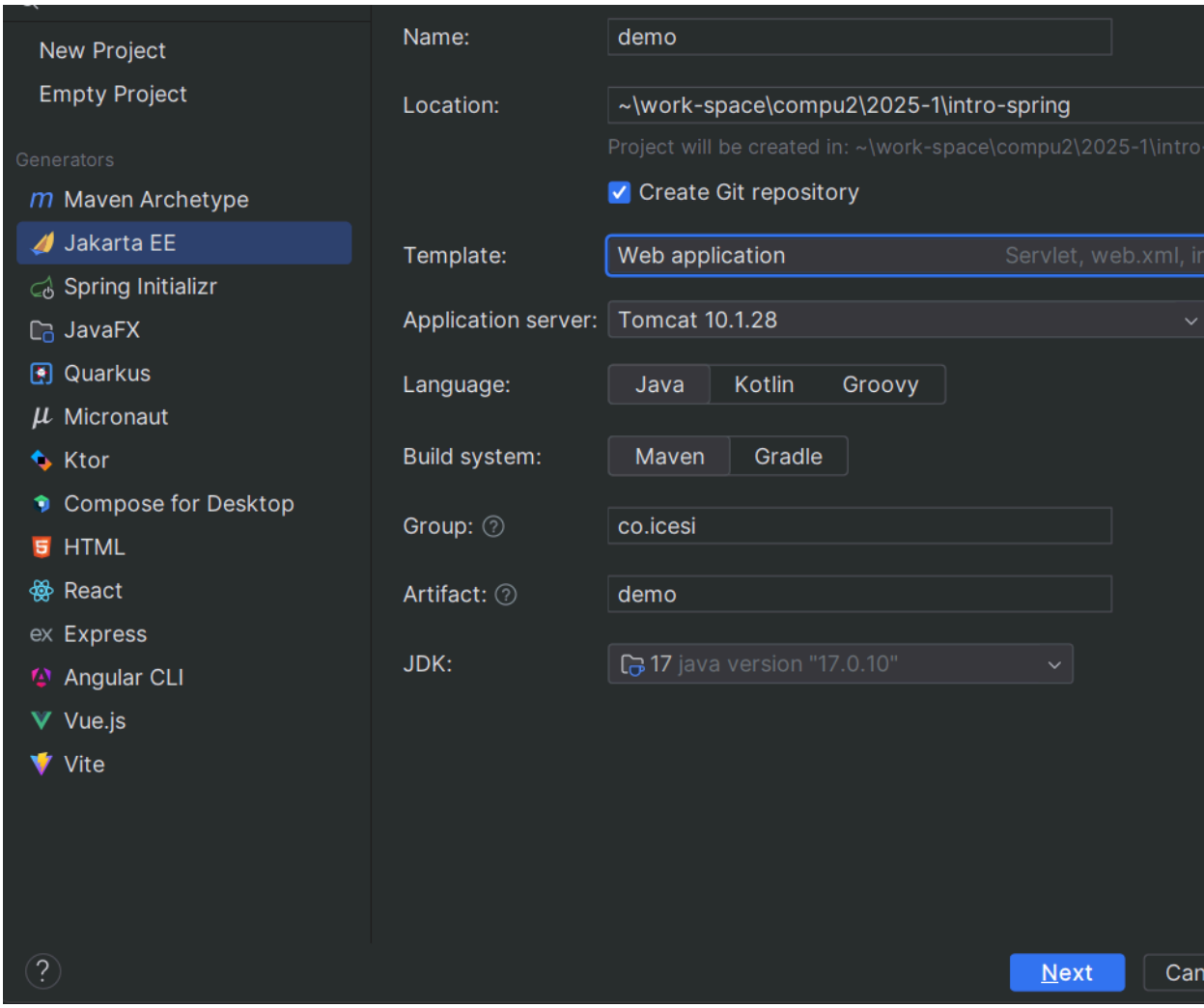
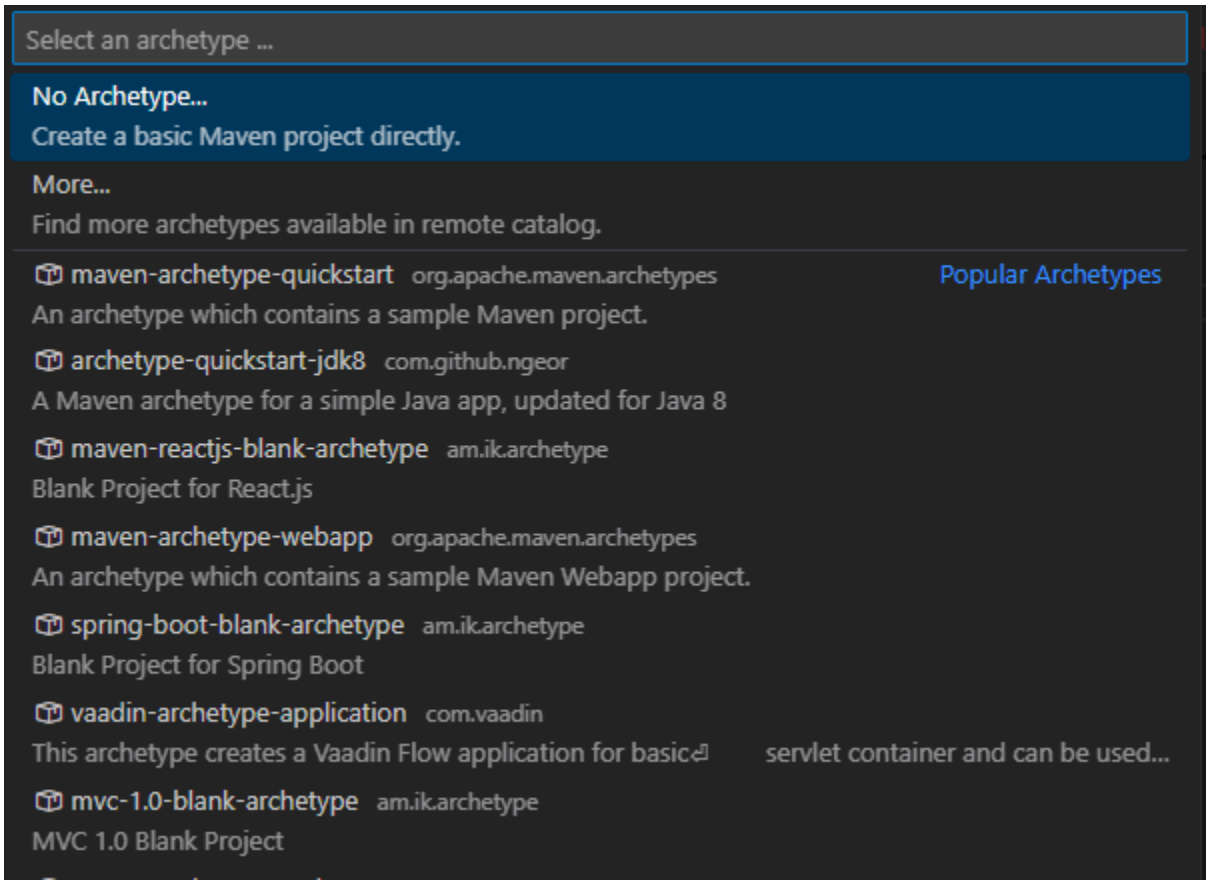
Demo: Developing Dynamic Web Project

1. Instalación y verificación de herramientas:
    a. Java 17: java -version x
    b. IDE: Intellij o vscode. x
    c. Maven: mvn -v x
    d. Tomcat 10: catalina.sh -v

2. Crear proyecto:
    a. Si está usando intellij cree uno con Jakarta EE usando tomcat 10



    b. En vscode: ctrl + p y escriba: > Java: Create java Project, luego seleccione maven Luego seleccione **maven-archetype-webapp,** las demás opciones dependen de ustedes.



Cuando se cree el proyecto agregar esta dependencia al pom.xml, adicionalmente agregue la carpeta para los archivos fuente y resources.

```
<dependency>
     <groupId>jakarta.servlet</groupId>
     <artifactId>jakarta.servlet-api</artifactId>
     <version>5.0.0</version>
     <scope>provided</scope>
   </dependency>
```

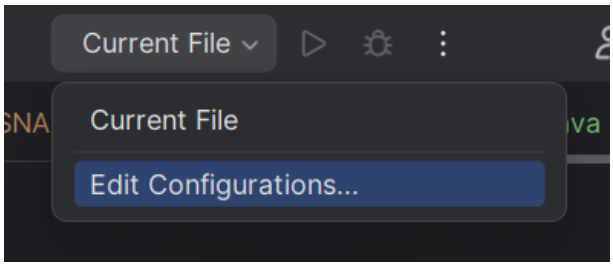
Añadir la clase:

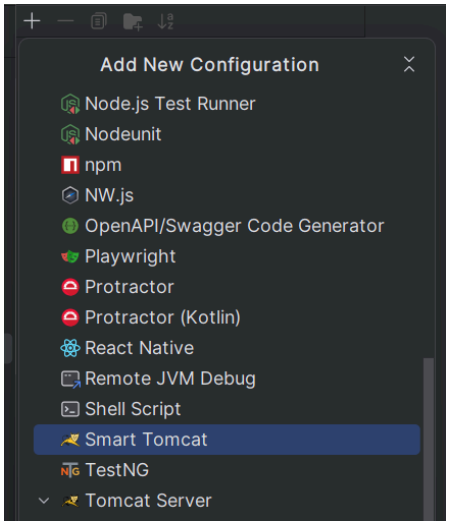
```
import java.io.*;
import jakarta.servlet.http.*;
import jakarta.servlet.annotation.*;

@WebServlet(name = "helloServlet", value = "/hello-servlet")
public class HelloServlet extends HttpServlet {
   private String message;

   public void init() {
       message = "Hello World!";
```

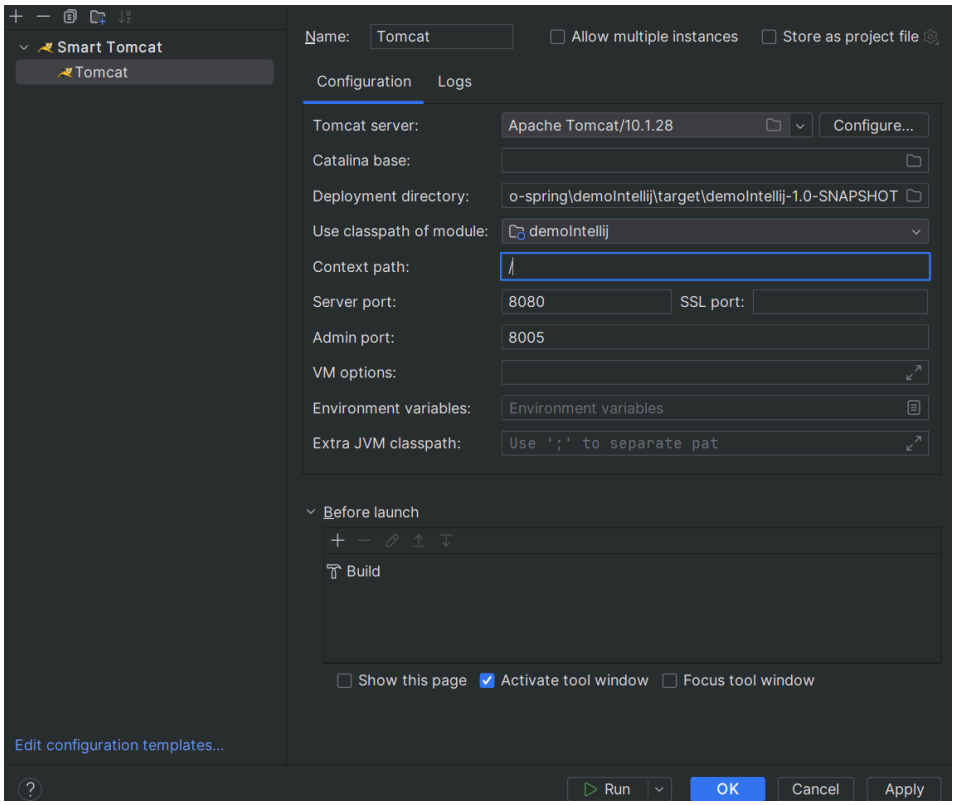
```
    }

    public void doGet(HttpServletRequest request,
HttpServletResponse response) throws IOException {
        response.setContentType("text/html");

        // Hello
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h1>" + message + "</h1>");
        out.println("</body></html>");
    }

    public void destroy() {
    }
}
```

Compile el proyecto para que la carpeta target sea generada: mvn clean compile

3. Seleccionar el servidor Tomcat para el deployment



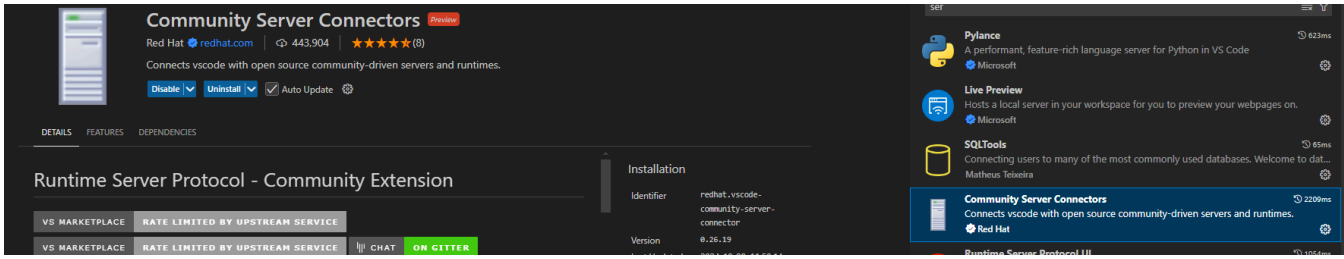


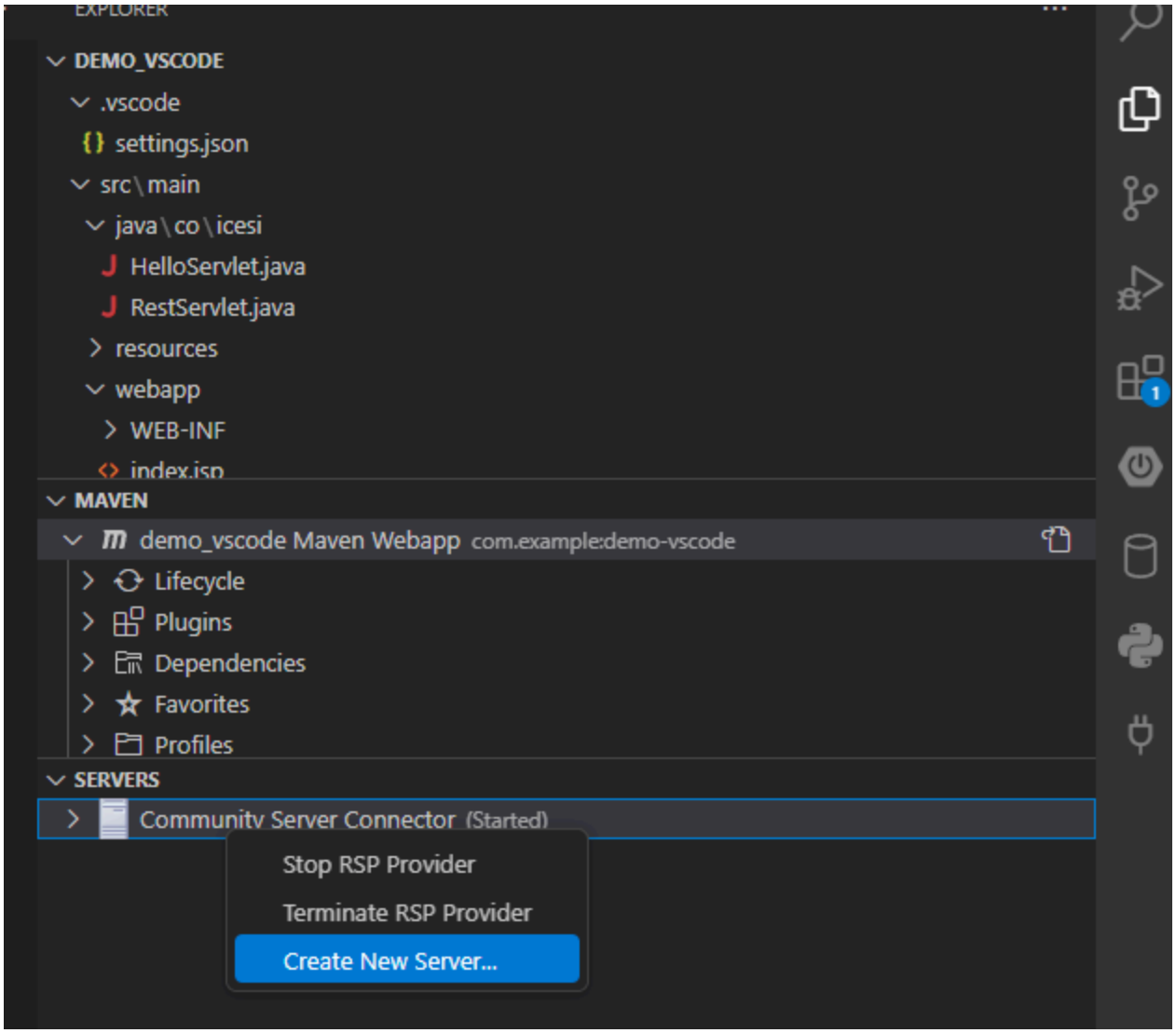


Correr el proyecto con esta configuración.

- **Despliegue con vscode**

Instale la siguiente extensión.

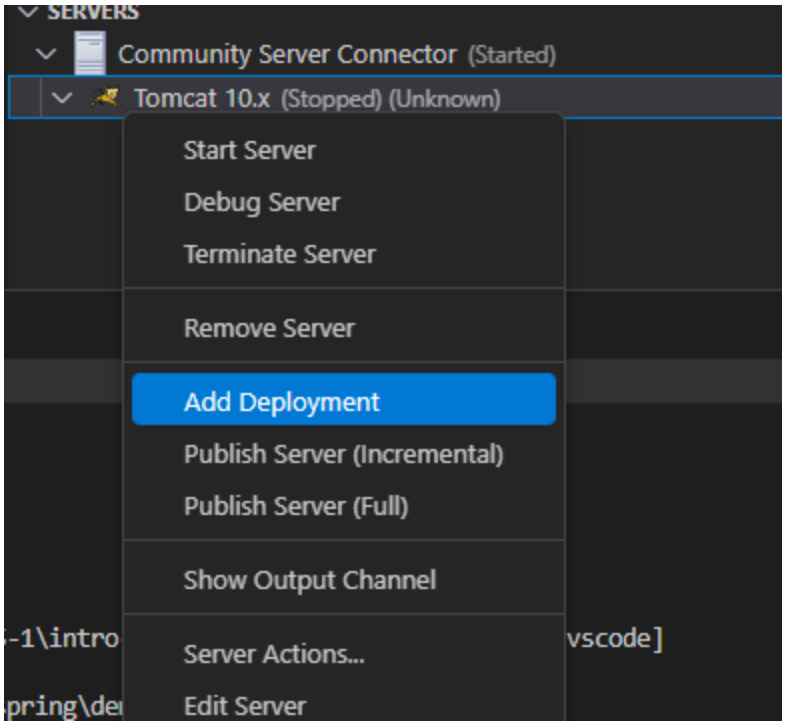




Use el tomcat que debió descargar antes.

Ejecute: mvn clean package.

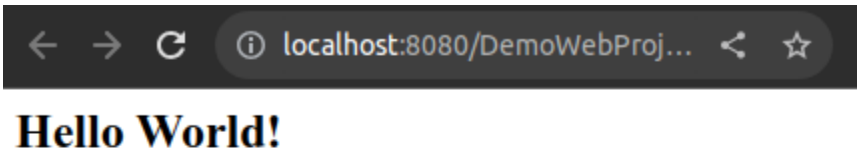
Esto genera la compilación del proyecto en la carpeta target



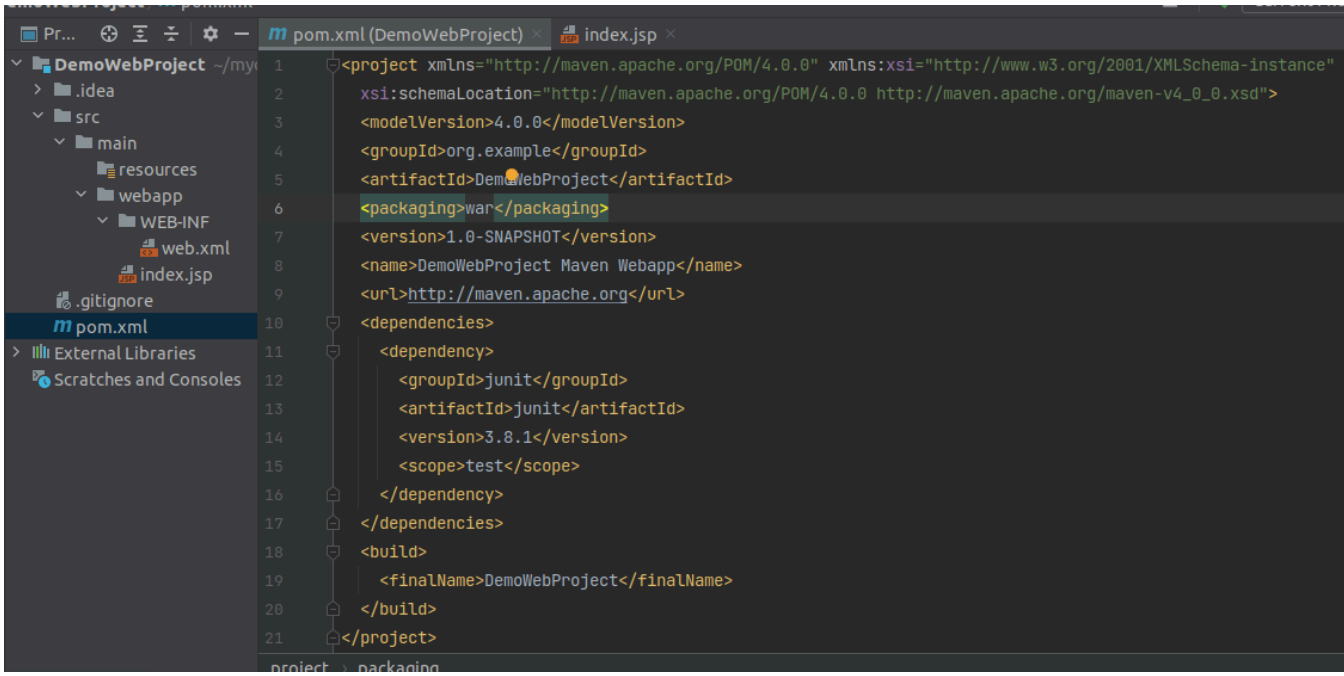
Seleccione la carpeta de binarios generada en la carpeta target.

1. En el panel "SERVERS" de la barra lateral → clic en "+"
2. Selecciona "Apache Tomcat" y elige la versión (descárgala desde ahí mismo o apunta a una instalación existente)
3. Genera el `.war`:
5. Clic derecho en el servidor → "Add Deployment" → selecciona el `.war` de target
6. Clic derecho → "Start"
7. Abre `http://localhost:8080` en el navegador

4. Podemos intentar ejecutar el proyecto No olvide darle clic derecho al deploy y darle start server



5. Demos un vistazo al archivo pom.xml

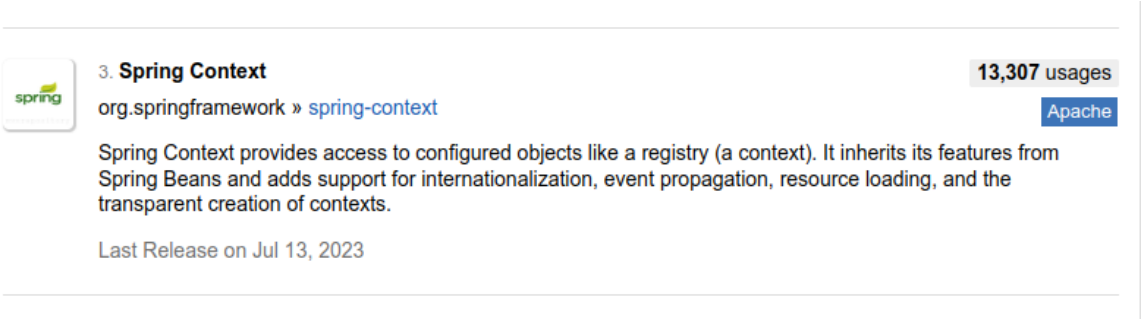


Demo: Developing Spring Web Project (IoContainer)

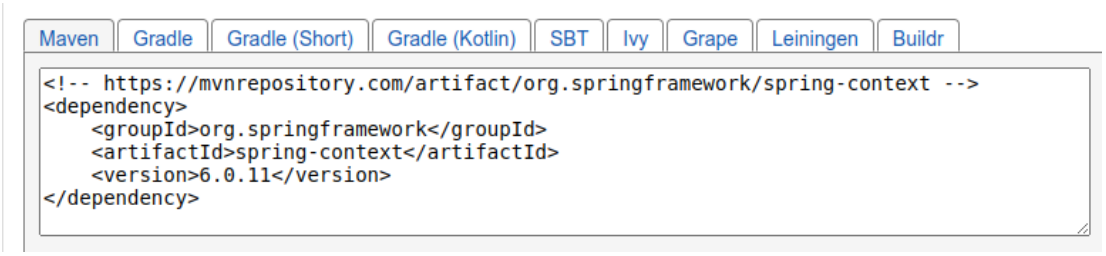
1. Buscamos los artefactos para adicion de dependencias en la siguiente url:

   https://mvnrepository.com/artifact/org.springframework

2. Seleccionamos el Spring Contex



Y copiamos el fragmento xml para adicionar la dependencia.



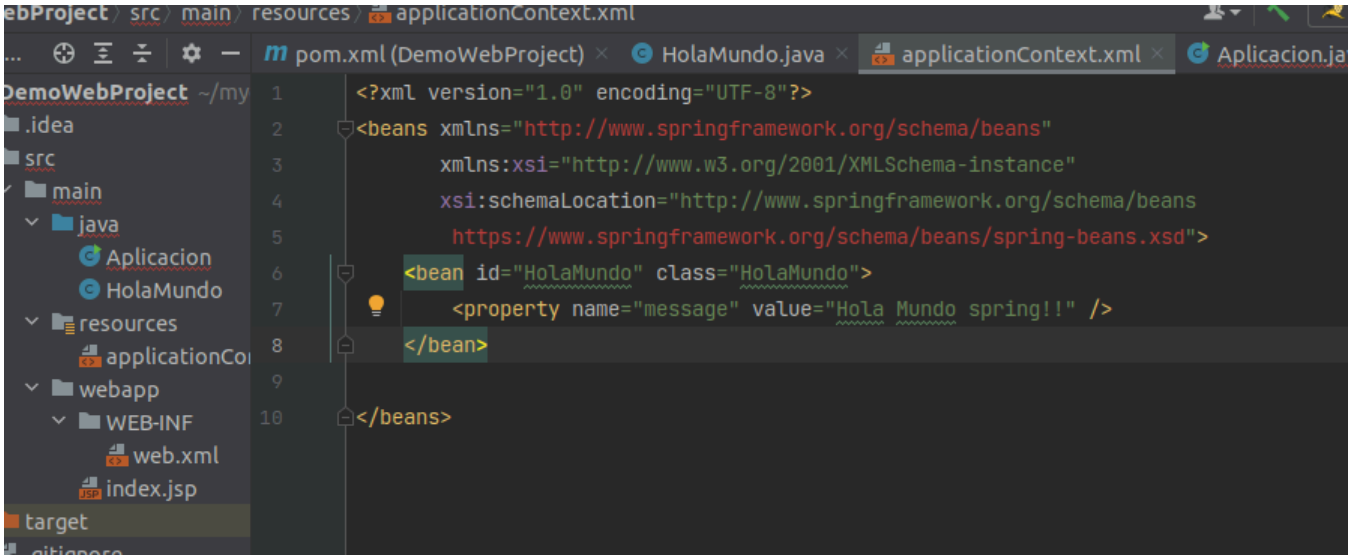
3.

:-$ mvn clean install

4. Adicionar la clase HolaMundo como bean

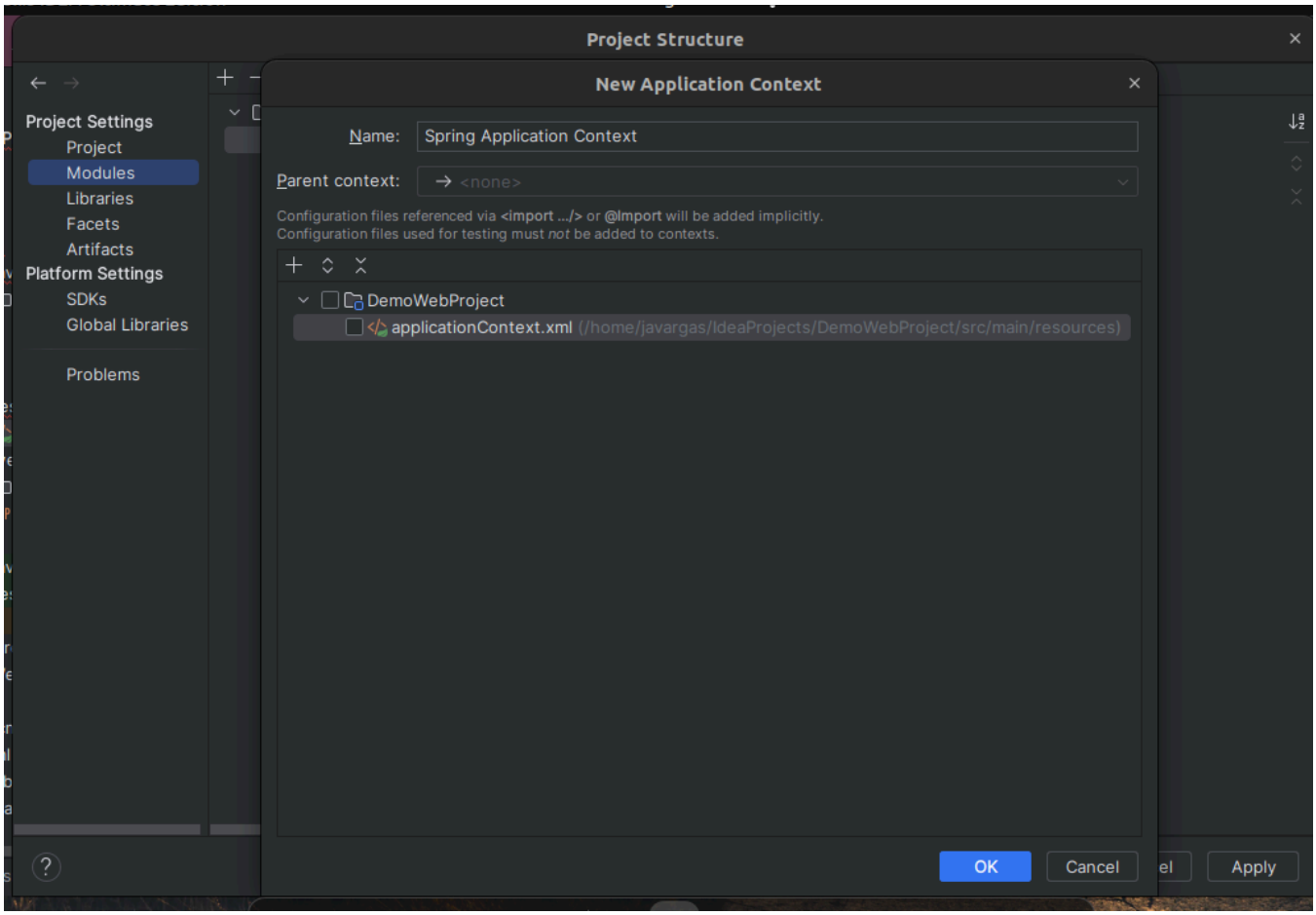
```
2 usages
public class HolaMundo {

    2 usages
    private String message;
    no usages
    public String getMessage() { return message; }

    no usages
    public void setMessage(String message) { this.message = message; }
}
```

5. En el applicationContext.xml



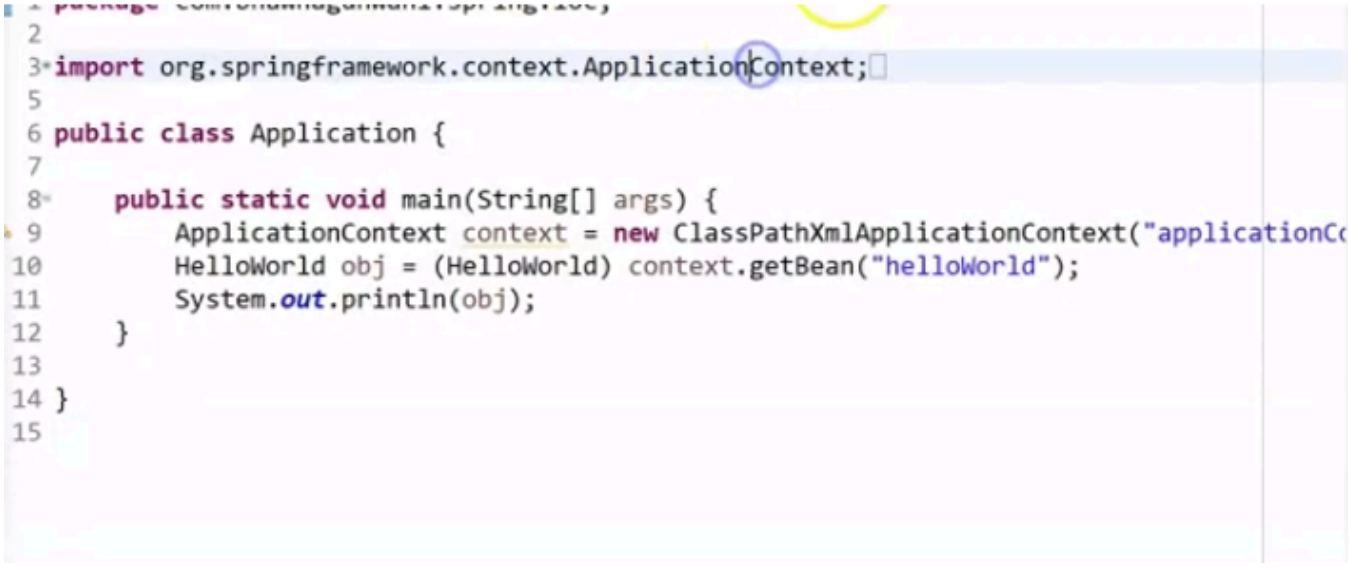

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
        https://www.springframework.org/schema/beans/spring-beans.xsd">
    <bean id="HolaMundo" class="HolaMundo">
        <property name="message" value="Hola Mundo spring!!" />
    </bean>

</beans>
```

File -> Project Structure -> Modules -> Add Spring → Add Spring Application Context



Previous

```
2
3  org.springframework.context.ApplicationContext;
4  org.springframework.context.support.ClassPathXmlApplicationContext;
5
6  lass Application {
7
8  ic static void main(String[] args) {
9   ApplicationContext context = ClassPathXmlApplicationContext("applicationContext.xml")
10
11
12
13
```



```
2
3  import org.springframework.context.ApplicationContext;
5
6  public class Application {
7
8      public static void main(String[] args) {
9          ApplicationContext context = new ClassPathXmlApplicationContext("applicationC
10         HelloWorld obj = (HelloWorld) context.getBean("helloWorld");
11         System.out.println(obj);
12     }
13
14 }
15
```

```
package com.example.demowebproject;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Aplicacion {
    public static void main(String[] args) {
        ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");

        HolaMundo hola  = (HolaMundo) context.getBean("HolaMundo");

    }
}
```

En el index.jsp

```
<%
    ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
    HolaMundo hola  = (HolaMundo) context.getBean("HolaMundo");
%>
```