Compte rendu du TP2 :

L'interface Idao qui se trouve dans le package dao,et qui contient la méthode getValue()

```java
package dao;

public interface IDao {
  double getValue();
}
```

La classe DaoImpl qui implemente l'interface Idao

```java
package dao;

import org.springframework.stereotype.Component;

@Component("dao") // le parametre dao c'est pour utiliser Qualifier("dao")
public class DaoImpl implements IDao{

    @Override
    public double getValue() {
        // TODO Auto-generated method stub
        return 5;
    }

}
```

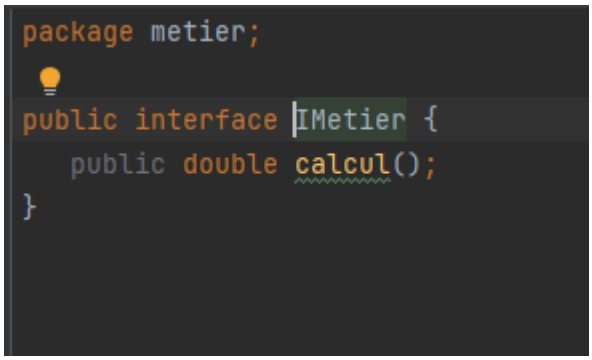
La classe DaoImpl2 qui implemente l'interface Idao

```java
package dao;

import org.springframework.stereotype.Component;

@Component("dao2")
public class DaoImpl2 implements IDao{
    @Override
    public double getValue() { return 6; }
}
```

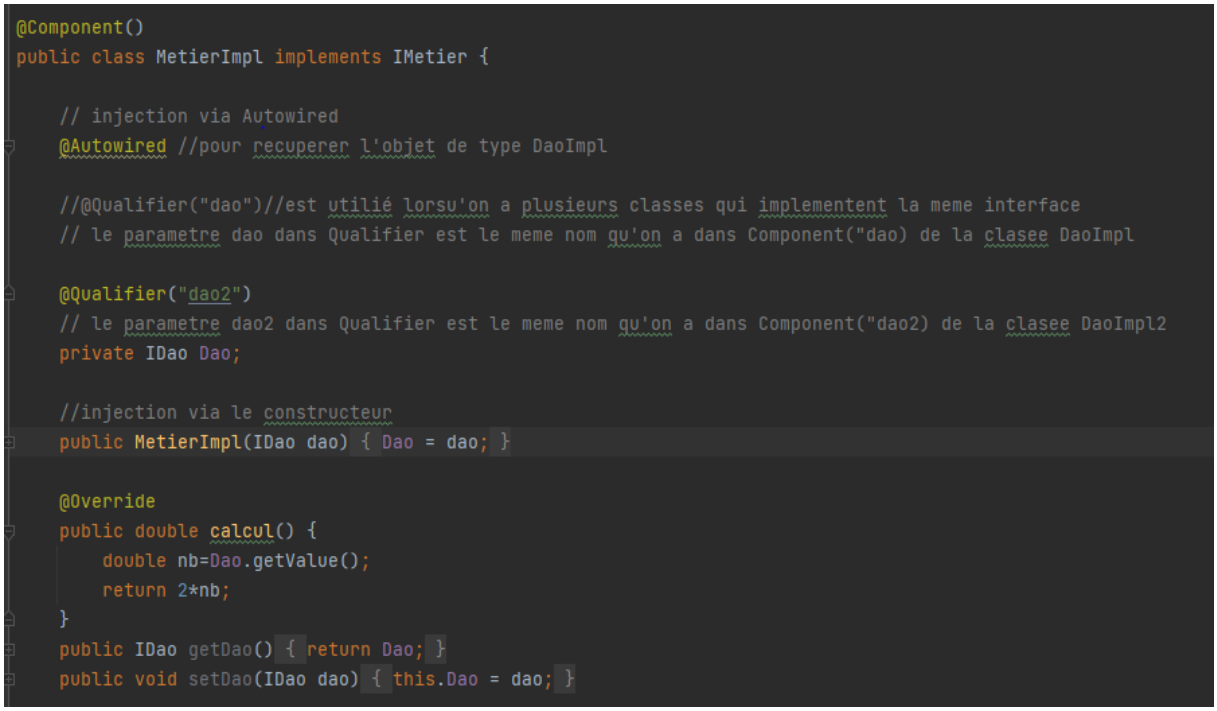
L’interface IMetier qui se trouve dans le package metier,et qui contient la méthode calcul()



```java
package metier;

public interface IMetier {
    public double calcul();
}
```

La classe MetierImpl qui implemente l’interface IMetier,cette classe contient un attribut de type Idao



```java
@Component()
public class MetierImpl implements IMetier {

    // injection via Autowired
    @Autowired //pour recuperer l'objet de type DaoImpl

    //@Qualifier("dao")//est utilié lorsu'on a plusieurs classes qui implementent la meme interface
    // le parametre dao dans Qualifier est le meme nom qu'on a dans Component("dao) de la clasee DaoImpl

    @Qualifier("dao2")
    // le parametre dao2 dans Qualifier est le meme nom qu'on a dans Component("dao2) de la clasee DaoImpl2
    private IDao Dao;

    //injection via le constructeur
    public MetierImpl(IDao dao) { Dao = dao; }

    @Override
    public double calcul() {
        double nb=Dao.getValue();
        return 2*nb;
    }
    public IDao getDao() { return Dao; }
    public void setDao(IDao dao) { this.Dao = dao; }
```

La classe presentation qui contient une instanciation dynamique de deux objets de type Idao et IMetier

```java
package Presentation;

import ...

public class presentation {

    public static void main(String[] args) {
        try {

            Scanner scanner=new Scanner(new File( pathname: "C:/Users/HP/Documents/S4/JEE/TP1/config.txt"));

            String daoClassName=scanner.next();
            String metierClassName=scanner.next();

            Class cdao=Class.forName(daoClassName);
            Class cmetier=Class.forName(metierClassName);

            IDao dao=(IDao) cdao.newInstance();
            IMetier metier=(IMetier) cmetier.newInstance();

            Method meth=cmetier.getMethod( name: "setDao",IDao.class);
            meth.invoke(metier,dao);
            System.out.println(metier.calcul());
        }catch(Exception e) {
            e.printStackTrace();
        }
    }
```

Le fichier pom.xlm qui contient les librairies de spring

```xml
<?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.
    <modelVersion>4.0.0</modelVersion>

    <groupId>org.example</groupId>
    <artifactId>JEE2</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <maven.compiler.source>8</maven.compiler.source>
        <maven.compiler.target>8</maven.compiler.target>
    </properties>

    <dependencies>
        <!-- https://mvnrepository.com/artifact/org.springframework/spring-core -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-core</artifactId>
            <version>5.3.16</version>
        </dependency>
```

Le fichier de configuration de spring version Xml

```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 http://www.springframework.org/schema/beans


    <bean class="dao.DaoImpl" id="dao"></bean> <!-- creer une instance de la classe dao.DaoImpl et la nomer dao-->
    <bean class="metier.MetierImpl" id="metier">
        <!-- on doit avoir necesserement un constructeur par defaut c-à-d sans paramètre-->
        <!-- ref="dao" fait reference à <bean class="dao.DaoImpl" id="dao"></bean> -->
        <!-- name="Dao" signifie la propriété Dao qui se trouve dans la classe MetierImpl va prendre un objet ref=
        <property name="Dao" ref="dao"></property>

        <!-- si on a un consutructeur par paramètre-->
        <constructor-arg ref="dao"></constructor-arg>
    </bean>
</beans>
```

La classe presentation de la version xml du spring

```java
package Presentation;

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

public class presentationAvecSpringXml {
    public static void main(String[] args) {
        ApplicationContext context=new ClassPathXmlApplicationContext( configLocation: "configurationSpring.xml");

        IMetier metier= (IMetier) context.getBean( s: "metier"); // recupere l'objet qui a l'id metier depuis le fic

        System.out.println(metier.calcul());
    }
}
```

La classe presentation de la version annotation du spring

```java
package Presentation;

import metier.IMetier;
import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;

public class presentationAvecSpringAnnotation {

    public static void main(String[] args) {
        ApplicationContext context = new AnnotationConfigApplicationContext( ...basePackages: "dao", "metier");
        // il va scanner les packages dao et metier
        IMetier metier = context.getBean(IMetier.class);
        System.out.println(metier.calcul());
    }
}
```