Mini projet Framework d'Injection des dépendances

Par Hajar TAJAYOUTI

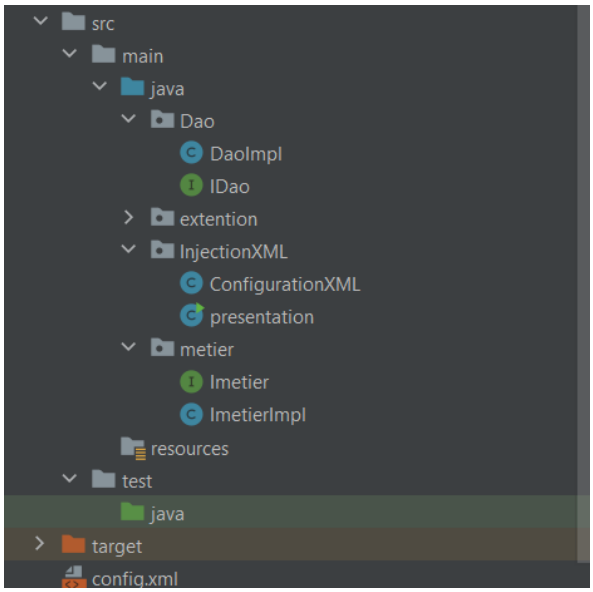
Objectif:

Créer un mini Framework d'injection des dépendances similaire à Spring IOC.
Le Framework doit permettre à un programmeur de faire l'injection des dépendances entre les différents composant de son application en respectant les possibilités suivantes :
1- A travers un fichier XML de configuration

2- En utilisant les annotations
3- Possibilité d'injection via Le constructeur

⇨ Injection des dépendances via XML

- Structure de projet



- La classe DaoImp

```java
public class DaoImpl implements IDao {
    1 usage
    @Override
    public double getData() {

        System.out.println("version base de donnes");
        return Math.random()*10;
    }
}
```

- La classe MetierImp

```java
public class ImetierImpl implements Imetier {
  //  @AmAutowired
    2 usages
    private IDao dao; //couplage faible
    1 usage
    @Override
    public double calcule() { return dao.getData()*540/Math.cos(Math.PI*40); }

 public void setDao(IDao dao) { this.dao = dao; }
}
```

- Fichier XML DOM

```xml
<fwork_TAJAYOUTI>
    <dao>Dao.DaoImpl</dao>
    <metier>metier.ImetierImpl</metier>
</fwork_TAJAYOUTI>
```

- La classe de la configuration :

```java
public class ConfigurationXML {
    private String nomfile;
```

```
    public ConfigurationXML(String nomfile) {
        this.nomfile = nomfile;
    }
    public String getNomfile() {
        return nomfile;
    }
    public void setNomfile(String nomfile) {
        this.nomfile = nomfile;
    }
    public String gestClassDao() throws ParserConfigurationException,
IOException, SAXException {
        DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();

        dbf.setFeature(XMLConstants.FEATURE_SECURE_PROCESSING, true);

        DocumentBuilder db = dbf.newDocumentBuilder();

        Document doc = db.parse(new File(nomfile));
        doc.getDocumentElement().normalize();
        NodeList list = doc.getElementsByTagName("fwork_TAJAYOUTI");
        String firstname = null;
        for (int temp = 0; temp < list.getLength(); temp++) {

            Node node = list.item(temp);

            if (node.getNodeType() == Node.ELEMENT_NODE) {

                Element element = (Element) node;

                firstname =
element.getElementsByTagName("dao").item(0).getTextContent();

            }
        }
        return firstname;
    }

    public String gestClassMetier() throws ParserConfigurationException,
IOException, SAXException {
        DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();

        dbf.setFeature(XMLConstants.FEATURE_SECURE_PROCESSING, true);

        DocumentBuilder db = dbf.newDocumentBuilder();

        Document doc = db.parse(new File(nomfile));

        doc.getDocumentElement().normalize();
        NodeList list = doc.getElementsByTagName("fwork_TAJAYOUTI");
        String metier = null;
        for (int temp = 0; temp < list.getLength(); temp++) {

            Node node = list.item(temp);

            if (node.getNodeType() == Node.ELEMENT_NODE) {

                Element element = (Element) node;
```

```
                metier =
element.getElementsByTagName("metier").item(0).getTextContent();
            }
        }
        return metier;
    }
public Imetier getClasse() throws InstantiationException,
IllegalAccessException, ParserConfigurationException, IOException,
SAXException, ClassNotFoundException, NoSuchMethodException,
InvocationTargetException {
    Class cDao=Class.forName(gestClassDao());
    IDao dao=(IDao) cDao.newInstance();
    Class cmetier=Class.forName(gestClassMetier());
    Imetier metier= (Imetier) cmetier.newInstance();
    Method method=cmetier.getMethod("setDao",IDao.class);
    method.invoke(metier,dao);
    return  metier;

}
}
```

- La class présentation :

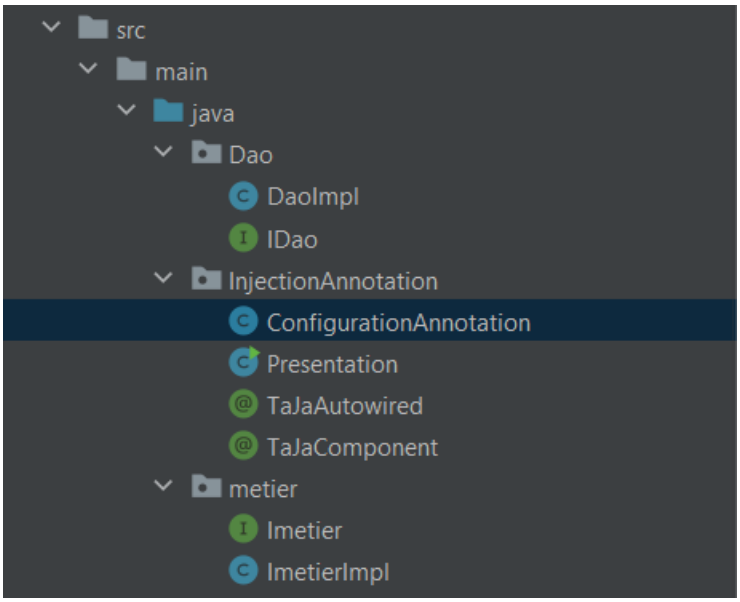
```
import ...

public class presentation {
    public static void main(String[] args) throws ParserConfigurationException, IOException, SAXException, ClassNotFoundException,
        ConfigurationXML fmwrkClass=new ConfigurationXML( nomfile: "config.xml");
        System.out.println(fmwrkClass.getClasse().calcule());
    }
}
```

- Résultat :

```
presentation
"C:\Program Files\Java\jdk-17.0.3\bin\java.exe" "-javaagent:C:\Program Fil
version base de donnes
2132.834926639275

Process finished with exit code 0
```

⇨ En utilisant les annotations

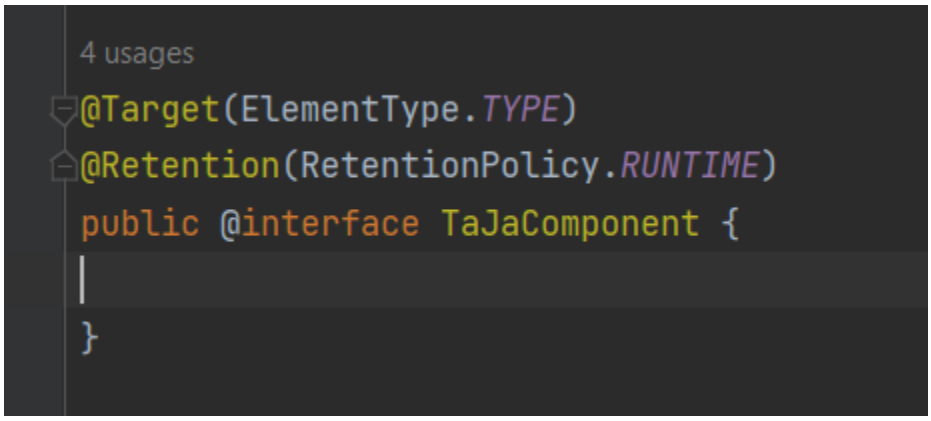
- Structure de projet



- L'annotation TaJaAutowired :

```
2 usages
@Target({ElementType.METHOD,
        ElementType.CONSTRUCTOR,
        ElementType.FIELD})
@Retention(RetentionPolicy.RUNTIME)
public @interface TaJaAutowired {

}
```

- L’annotation TaJaComponent:



```
4 usages
@Target(ElementType.TYPE)
@Retention(RetentionPolicy.RUNTIME)
public @interface TaJaComponent {

}
```

-
- La class ConfigurationAnotation :

```
public class ConfigurationAnnotation {
    HashMap<Class, Object> instances=new HashMap<Class, Object>();
    public void getClasses(String... packages) throws InstantiationException,
IllegalAccessException, NoSuchMethodException, SecurityException,
IllegalArgumentException, InvocationTargetException,
InvocationTargetException {
        ArrayList<Class> classes=new ArrayList<Class>();
        Set<Class<?>> subTypesOf=null;

        for(String packageName : packages) {
            Reflections reflections = new Reflections(new
ConfigurationBuilder()
                    .setScanners(new SubTypesScanner(false ), new
ResourcesScanner())
                    .addUrls(ClasspathHelper.forJavaClassPath())
                    .filterInputsBy(new
FilterBuilder().include(FilterBuilder.prefix(packageName))));
            //prefix(packageName)
          //  new Reflections(new ConfigurationBuilder().filterInputsBy(new
FilterBuilder().includePackage(packageName)).setUrls(ClasspathHelper.forPacka
ge(packageName)).setScanners(new SubTypesScanner(false)));
  /*FilterBuilder<String> filter = new FilterBuilder();
            filter.add("test");*/
            subTypesOf = reflections.getSubTypesOf(Object.class);
            for( Class c :subTypesOf) {
                if(c.toString().contains("class")) {
                    Object o = c.newInstance();
                    instances.put(c.getInterfaces()[0], o);
                    classes.add(c);
                }
            }


        }
        for(Class c : classes) {
            if( c.getAnnotations()[0].toString().contains("TaJaComponent") &&
c.getDeclaredFields().length>0 ) {
```

```
                Field[] fields =c.getDeclaredFields();
                for(Field f : fields) {

if(f.getAnnotations()[0].toString().contains("TaJaAutowired"))
                    {
                        Method method=c.getMethod("setDao",f.getType());
                        method.invoke(instances.get(c.getInterfaces()[0]),
instances.get(f.getType()));
                    }
                }
            }
        }

    }
    public HashMap<Class, Object> getInstances(){
        return instances;
    }
}
```

- La class présentation

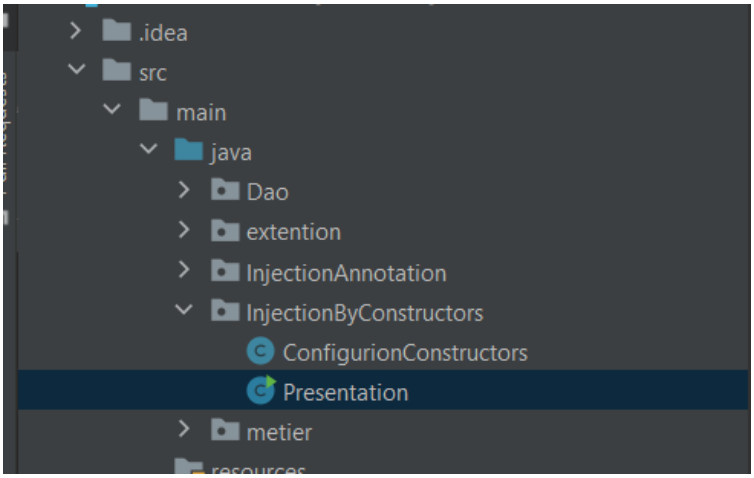
```
import java.lang.reflect.InvocationTargetException;

public class Presentation {
    public static void main(String[] args) throws InvocationTargetException, InstantiationException, IllegalAccessException, NoSu
    ConfigurationAnnotation contx=new ConfigurationAnnotation();
    contx.getClasses( ...packages: "dao", "metier");
    Imetier imetier= (Imetier) contx.getInstances().get(Imetier.class);

        System.out.println(imetier.calcule());
}}
```

- resultat

```
presentation
"C:\Program Files\Java\jdk-17.0.3\bin\java.exe" "-javaagent:C:\Program Fil
version base de donnes
2132.834926639275

Process finished with exit code 0
```

⇨ En utilisant le constructeur

- Structure de projet



- La classe configuration

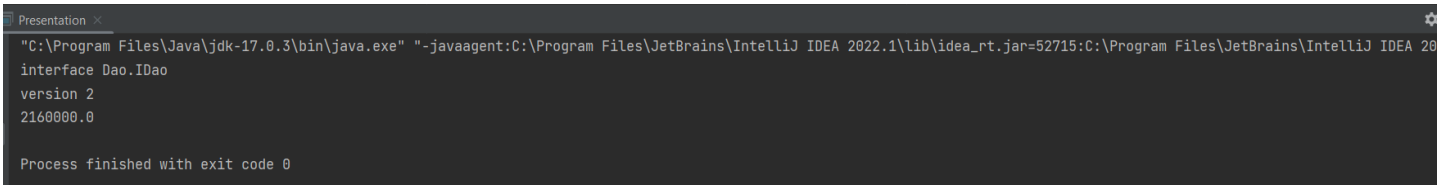
```
public class ConfigurionConstructors {

   private Map<Class,Object> listClass= new HashMap<Class,Object>();
   private List<Class> listClasse=new ArrayList<>();


    public ConfigurionConstructors(List<Class> listClasse) {
        this.listClasse= listClasse;
    }
    public Object getInstance(Class r) throws InstantiationException,
IllegalAccessException {

        for (Class ce:listClass.keySet()) {
          if (ce.getInterfaces()[0].toString().equals(r.toString())){
              System.out.println(ce.getInterfaces()[0].toString());
              return listClass.get(ce);
          }

        }
return null;
    }
    public  void instacierInjection() throws InstantiationException,
IllegalAccessException, NoSuchMethodException, InvocationTargetException {
        for (Class c:listClasse) {
            listClass.put(c,c.newInstance());
        }
        for (Class c:listClasse) {
            if (c.getDeclaredFields()!=null){
                for (Field f:c.getDeclaredFields()) {
                    if ( f.getType().toString().contains("i")){
                        String methodName="setDao";
```

```
                        Method method=c.getMethod(methodName,f.getType());
                        method.invoke(listClass.get(c),
getInstance(f.getType()));

                    }

                }
            }

        }


        }

    public Map<Class, Object> getListClass() {
        return listClass;
    }

    public void setListClass(Map<Class, Object> listClass) {
        this.listClass = listClass;
    }

    public List<Class> getListClasse() {
        return listClasse;
    }

    public void setListClasse(List<Class> listClasse) {
        this.listClasse = listClasse;
    }
}
```

- La classe presentation

```
public class Presentation {
    public static void main(String[] args) throws InvocationTargetException,
InstantiationException, IllegalAccessException, NoSuchMethodException {
        List<Class> list = new ArrayList<>();
        list.add(ImetierImpl.class);
        list.add(DaoImpl2.class);
        ConfigurionConstructors configurionConstructors = new
ConfigurionConstructors(list);
        configurionConstructors.instacierInjection();
              ImetierImpl imt= (ImetierImpl)
configurionConstructors.getListClass().get(ImetierImpl.class);
              System.out.println(imt.calcule());
   }
    }
```

- resultat


Presentation
"C:\Program Files\Java\jdk-17.0.3\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2022.1\lib\idea_rt.jar=52715:C:\Program Files\JetBrains\IntelliJ IDEA 20
interface Dao.IDao
version 2
2160000.0
Process finished with exit code 0