JAVA 4A SAGI – GUI JavaFx avec application Spring (projet Maven)

Cet exemple montre comment exploiter des objets instanciés par Spring au sein dune interface GUI gérée avec JavaFX

- *le contexte Spring crée un Bean de type data.Data*
- *l'interface GUI accède au bean de la classe Data*
- *exemple transposable pour une exploitation des couches Metier/Dao depuis une interface GUI JavaFX*

Source https://stackoverflow.com/ [Integrating Spring with FXML nested controllers]

1) Projet Maven

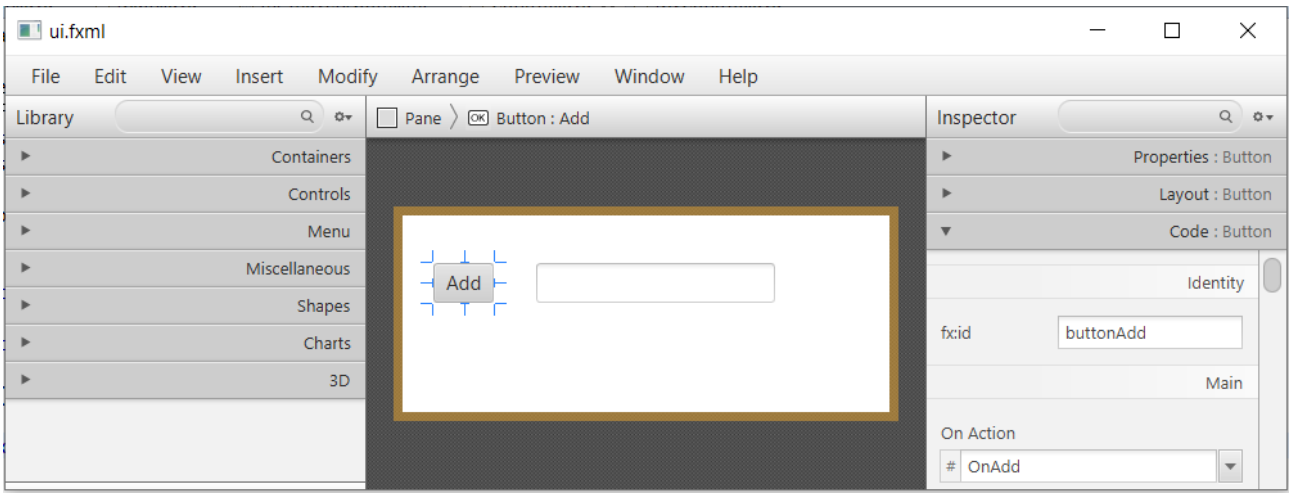
group ID= `pta.sagi` /Artifact ID=`jfx-spring`

2) POM.xml (ajout dépendance spring-context)

```
<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
https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>pta.sagi</groupId>
  <artifactId>jfx-spring</artifactId>
  <version>0.0.1-SNAPSHOT</version>

  <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>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>4.1.7.RELEASE</version>
        </dependency>
</dependencies>
</project>
```

3) Fichier ui.fxml (enregistré dans le dossier du package application)

Controller Class = `application.Control` (classe java de gestion GUI)

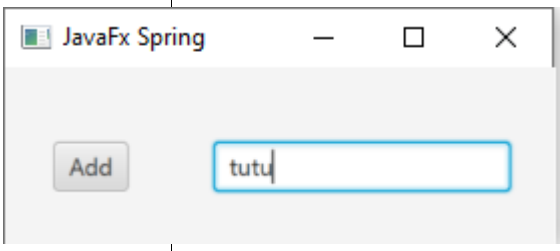


4) data.Data (classe du Bean), config.Config (configuration spring) et classe exécutable

```java
package data;

public class Data
{
 public String str;
 public Data() { str="tutu";}
}
```

```java
package config;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import data.Data;

@Configuration // Annotation Spring
public class Config
{
        @Bean  Data data() {return new Data();}
}
```

```java
package application;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import data.Data;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Button;
import javafx.scene.control.TextField;

public class Control
{
 @Autowired
 Data myData; // injection de dépendance Spring

 @FXML
 TextField textbox;
 @FXML
 void OnAdd(ActionEvent e){  textbox.setText(myData.str); }
}
```




```java
package application;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;
...

public class Main extends Application {

        private AnnotationConfigApplicationContext context;

        @Override
        public void stop(){   context.close();}

        @Override
        public void start(Stage primaryStage) {
        try {
                context = new AnnotationConfigApplicationContext(Config.class);
                FXMLLoader loader = new FXMLLoader(getClass().getResource("ui.fxml"));
                Pane root = (Pane)loader.load();
                Scene scene = new Scene(root,500,300);
                primaryStage.setScene(scene);
                primaryStage.setTitle("JavaFx Spring");
                Control ctrl = loader.getController();
                context.getAutowireCapableBeanFactory().autowireBean(ctrl);
                primaryStage.show();
        } catch(Exception e) {       e.printStackTrace();  }
        }

        public static void main(String[] args){    launch(args);  }
}
```