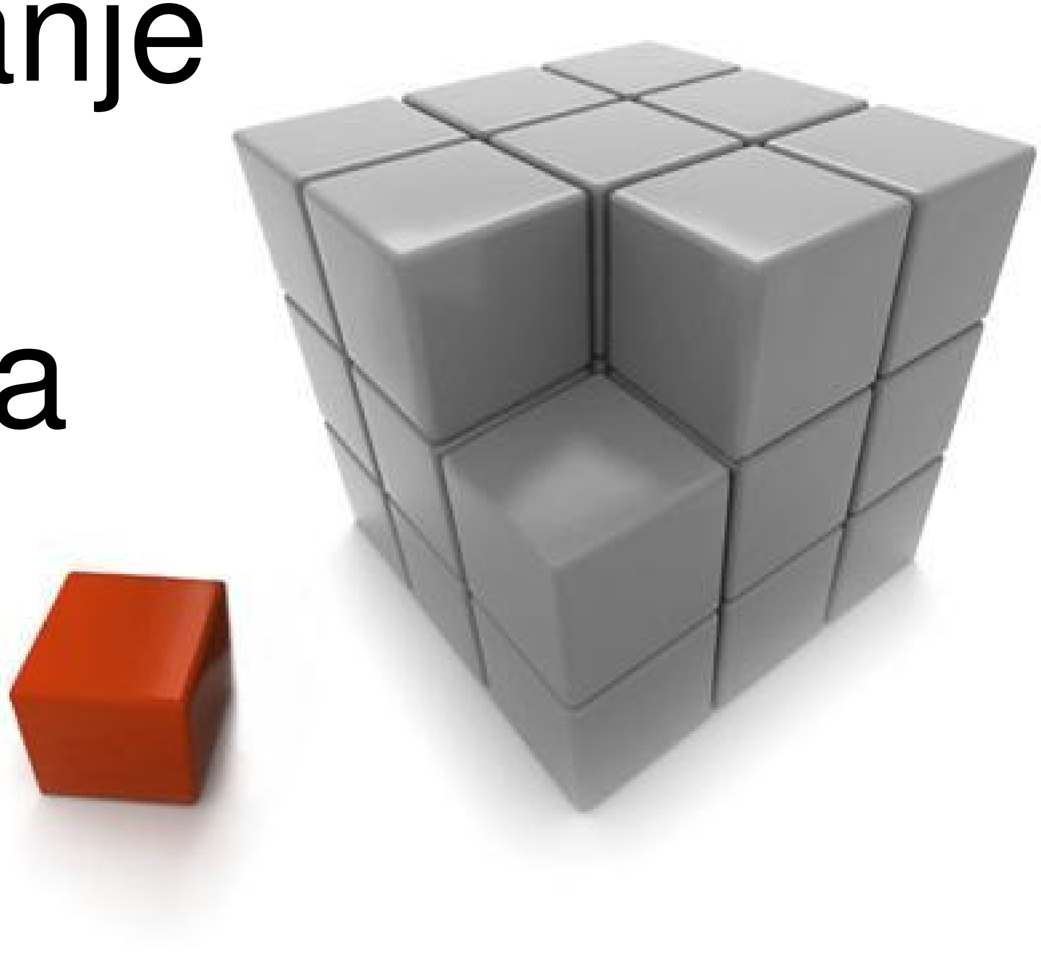
19. hroug
godišnja konferencija

Dinamičko programiranje ADF Aplikacija

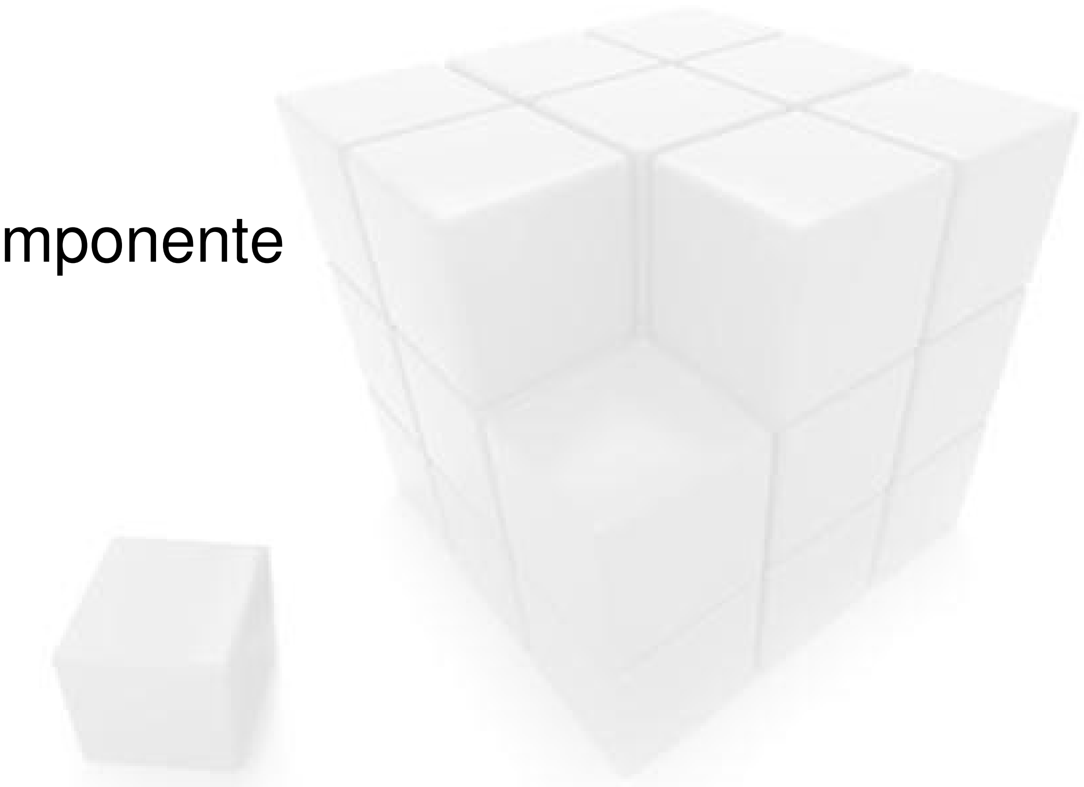
in2



Rovinj, 17.10.2014.

SADRŽAJ

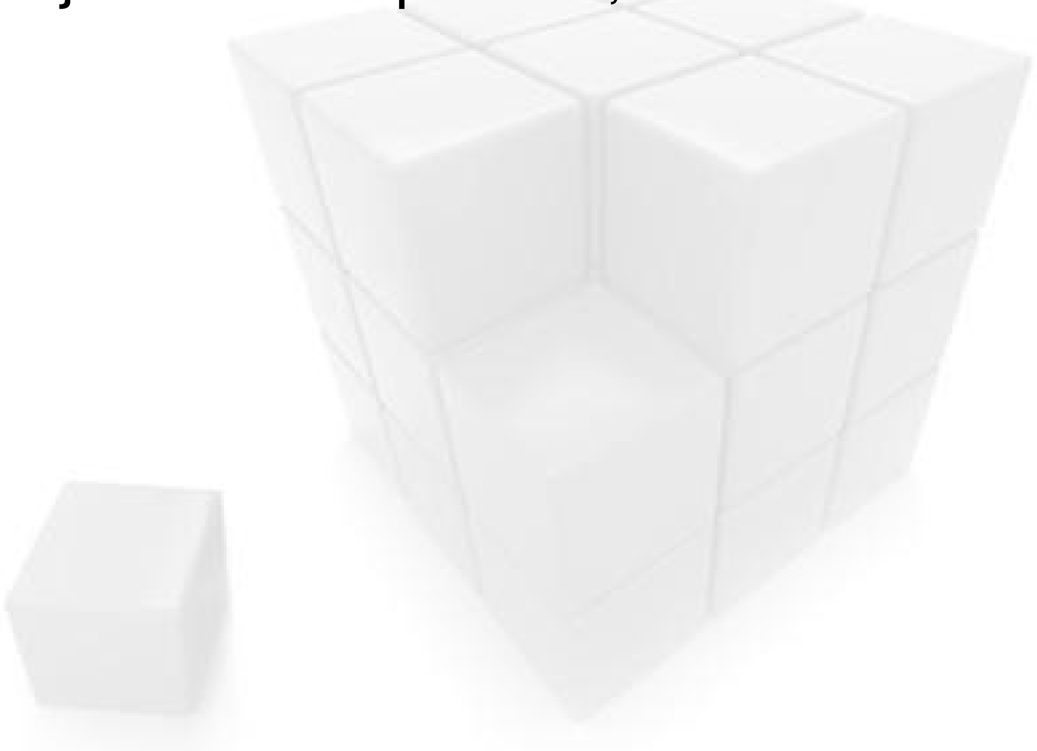
- ❑ Uvod
- ❑ ADF Podrška
- ❑ Implementacija
- ❑ Gumb, input text komponente
- ❑ Lista vrijednosti
- ❑ Demo
- ❑ Zaključak



Uvod

- Oracle ADF

- Dinamičko programiranje ADF komponenti, zašto?

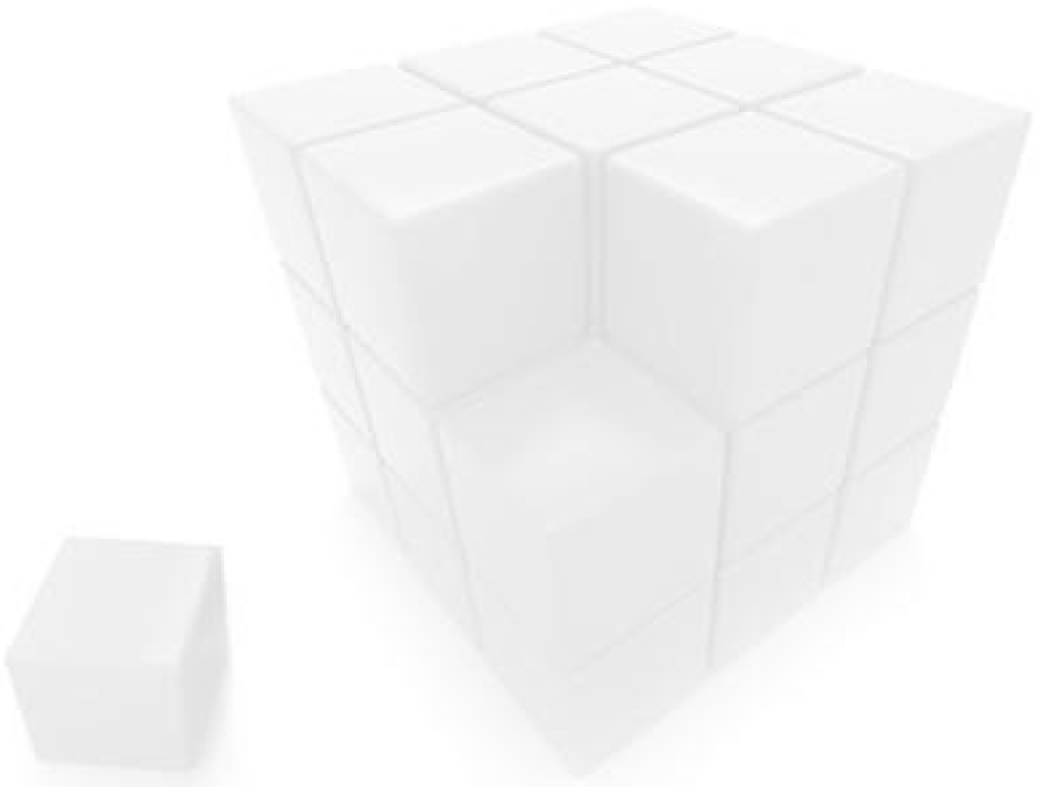


ADF Podrška

❑ Verzija 11.1.2

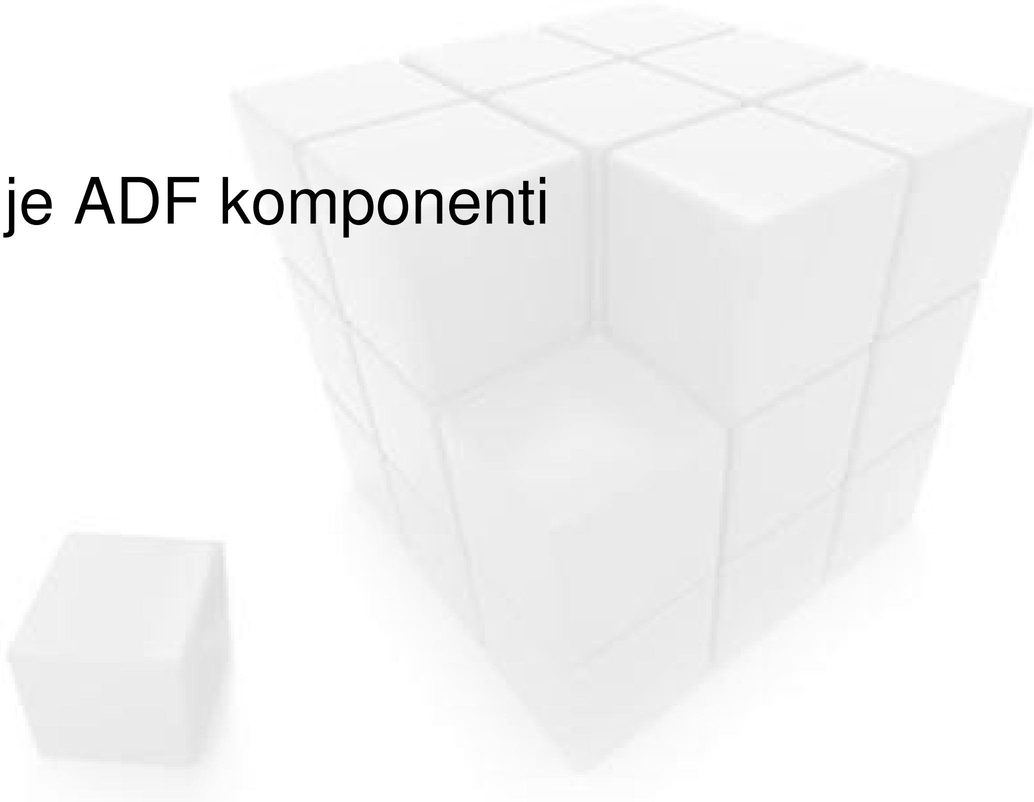
❑ …

❑ Verzija 12.1.3

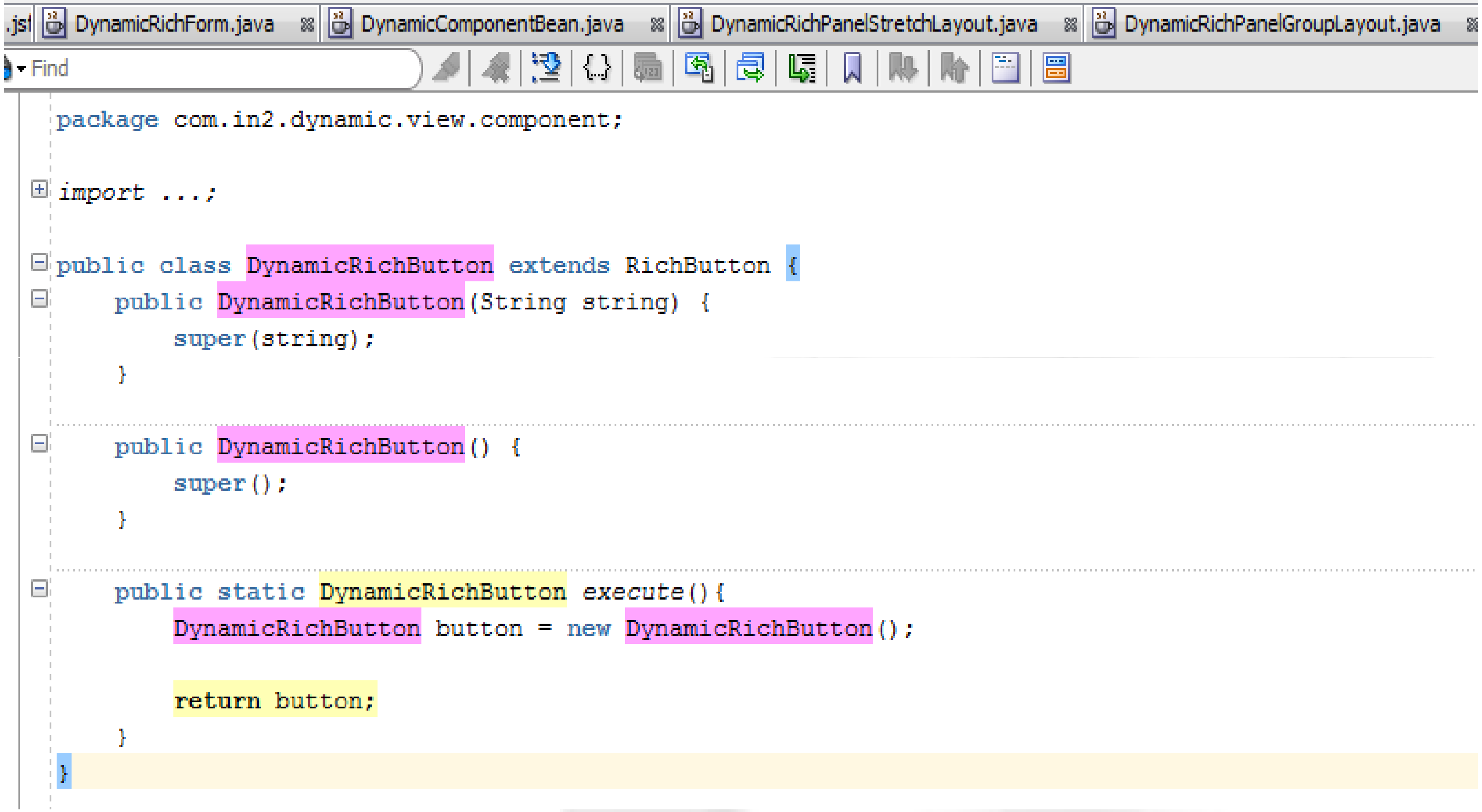


Implementacija

- Jsf
- Binding
- Vlastite implementacije ADF komponenti
- Metamodel



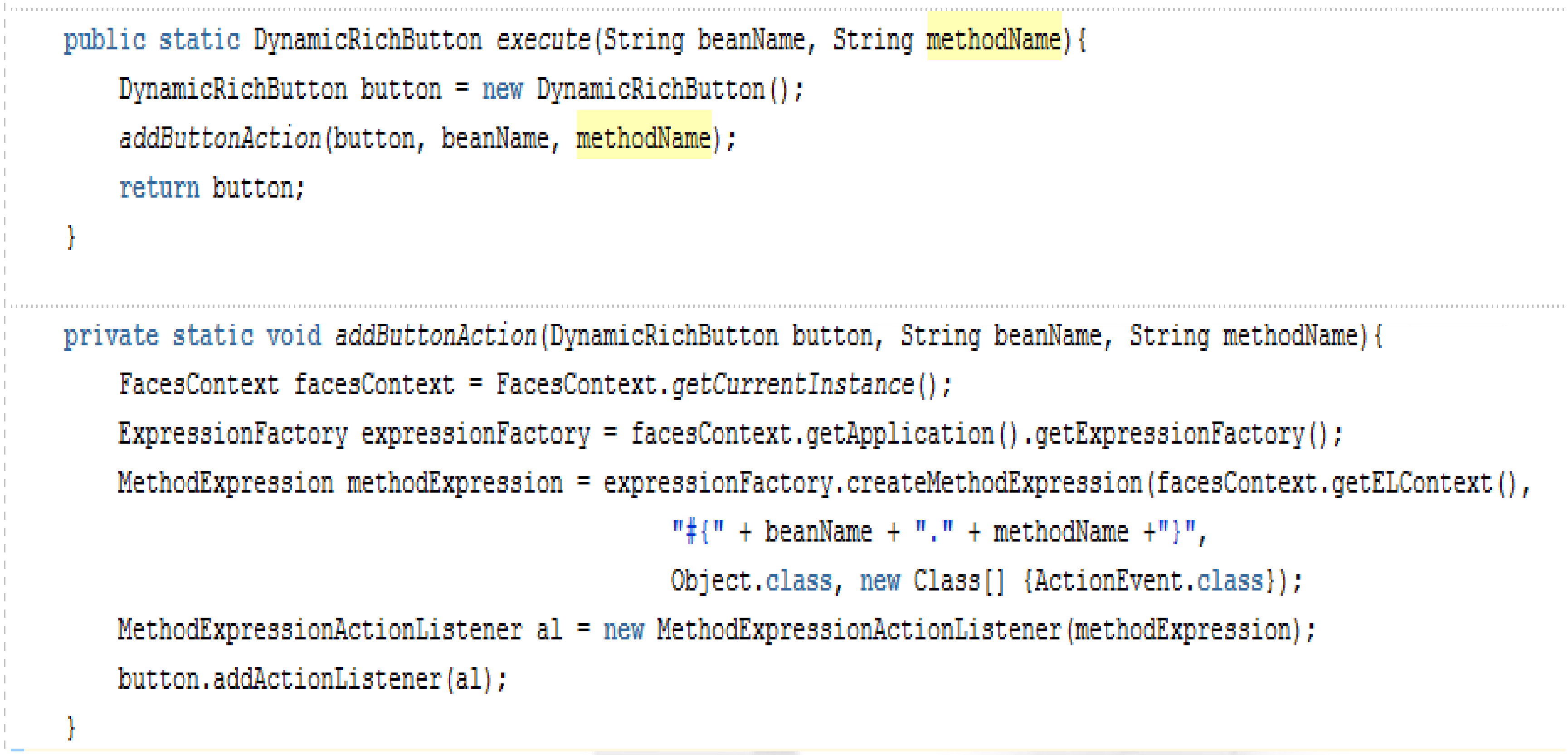
Kreiranje gumba



.jsf
DynamicRichForm.java
DynamicComponentBean.java
DynamicRichPanelStretchLayout.java
DynamicRichPanelGroupLayout.java
Find

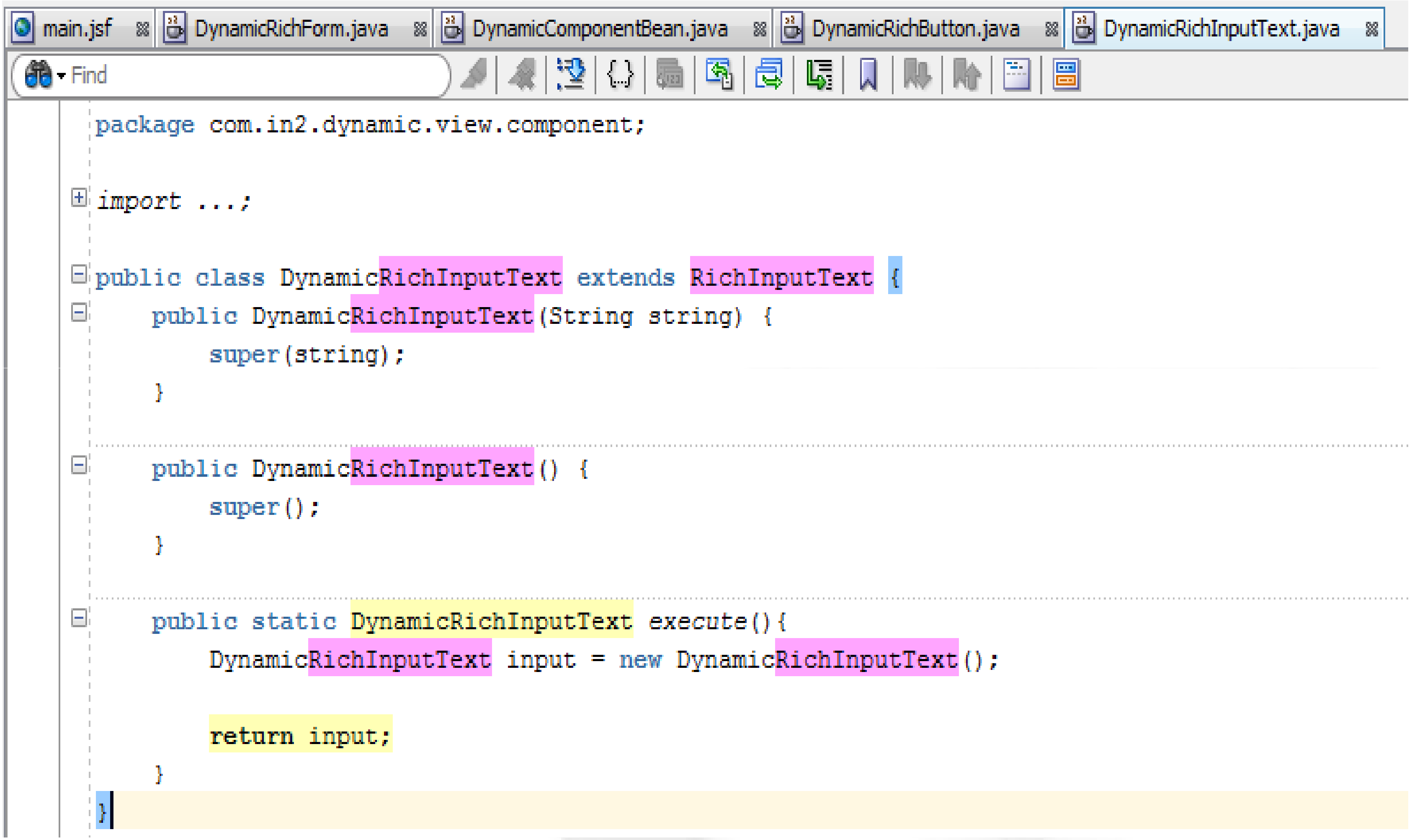
```
package com.in2.dynamic.view.component;

import ...;

public class DynamicRichButton extends RichButton {
    public DynamicRichButton(String string) {
        super(string);
    }

    public DynamicRichButton() {
        super();
    }

    public static DynamicRichButton execute(){
        DynamicRichButton button = new DynamicRichButton();

        return button;
    }
}
```

Dodavanje akcije na gumb



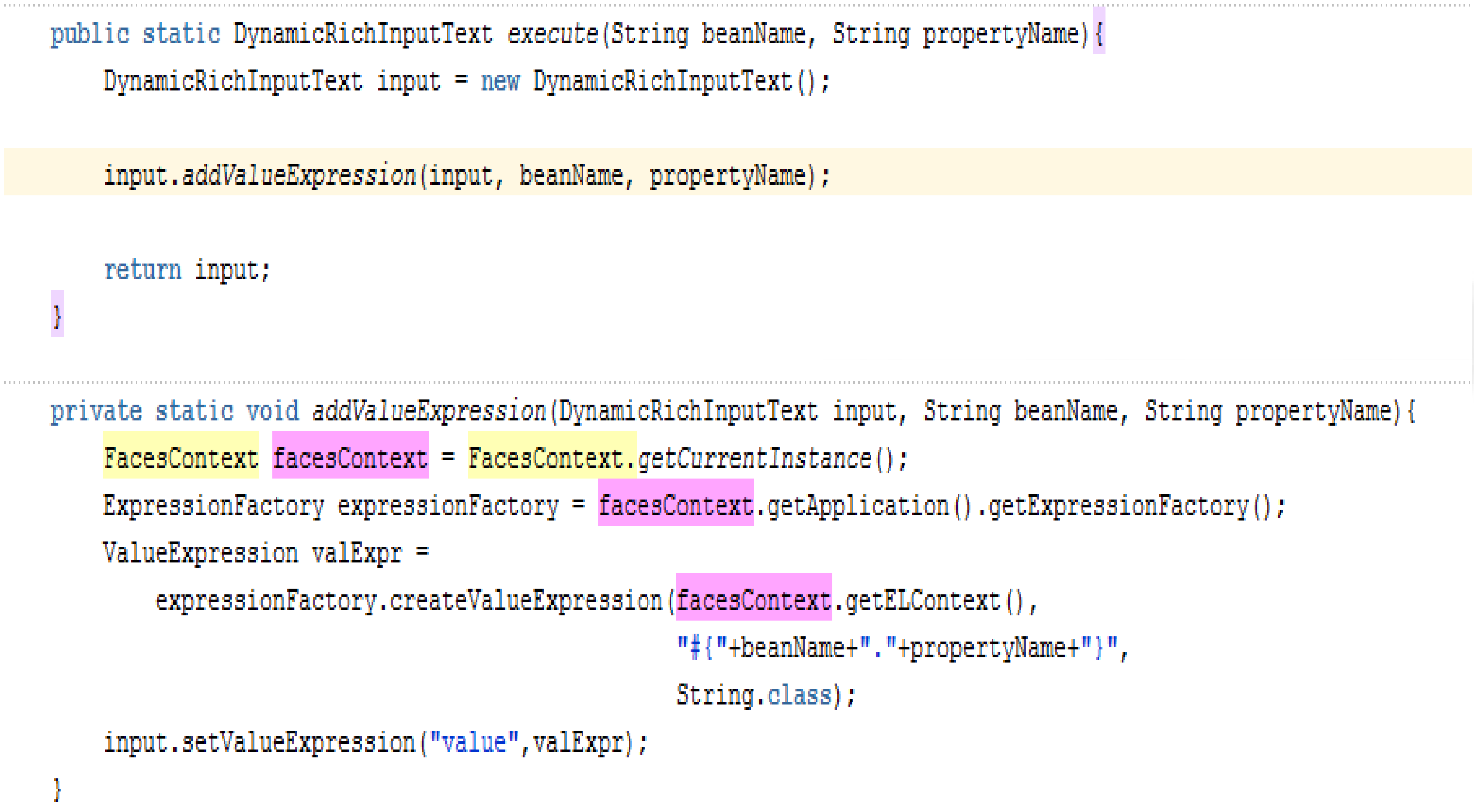
```
public static DynamicRichButton execute(String beanName, String methodName){
    DynamicRichButton button = new DynamicRichButton();
    addButtonAction(button, beanName, methodName);
    return button;
}

private static void addButtonAction(DynamicRichButton button, String beanName, String methodName){
    FacesContext facesContext = FacesContext.getCurrentInstance();
    ExpressionFactory expressionFactory = facesContext.getApplication().getExpressionFactory();
    MethodExpression methodExpression = expressionFactory.createMethodExpression(facesContext.getELContext(),
                                        "#{" + beanName + "." + methodName +"}",
                                        Object.class, new Class[] {ActionEvent.class});
    MethodExpressionActionListener al = new MethodExpressionActionListener(methodExpression);
    button.addActionListener(al);
}
```

Kreiranje input texta



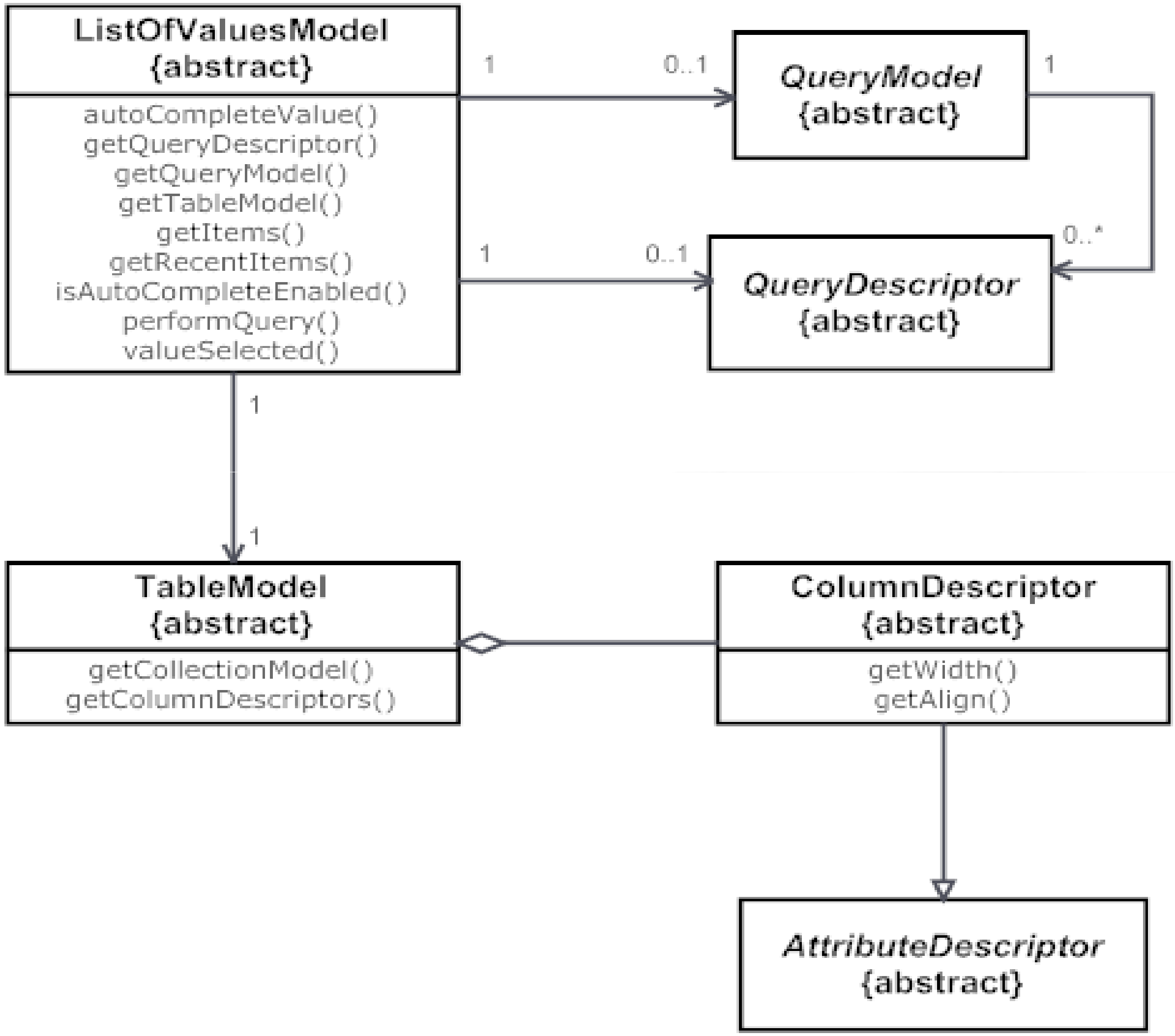
```
package com.in2.dynamic.view.component;

import ...;

public class DynamicRichInputText extends RichInputText {
    public DynamicRichInputText(String string) {
        super(string);
    }

    public DynamicRichInputText() {
        super();
    }

    public static DynamicRichInputText execute(){
        DynamicRichInputText input = new DynamicRichInputText();

        return input;
    }
}
```

Mapiranje vrijednosti input texta

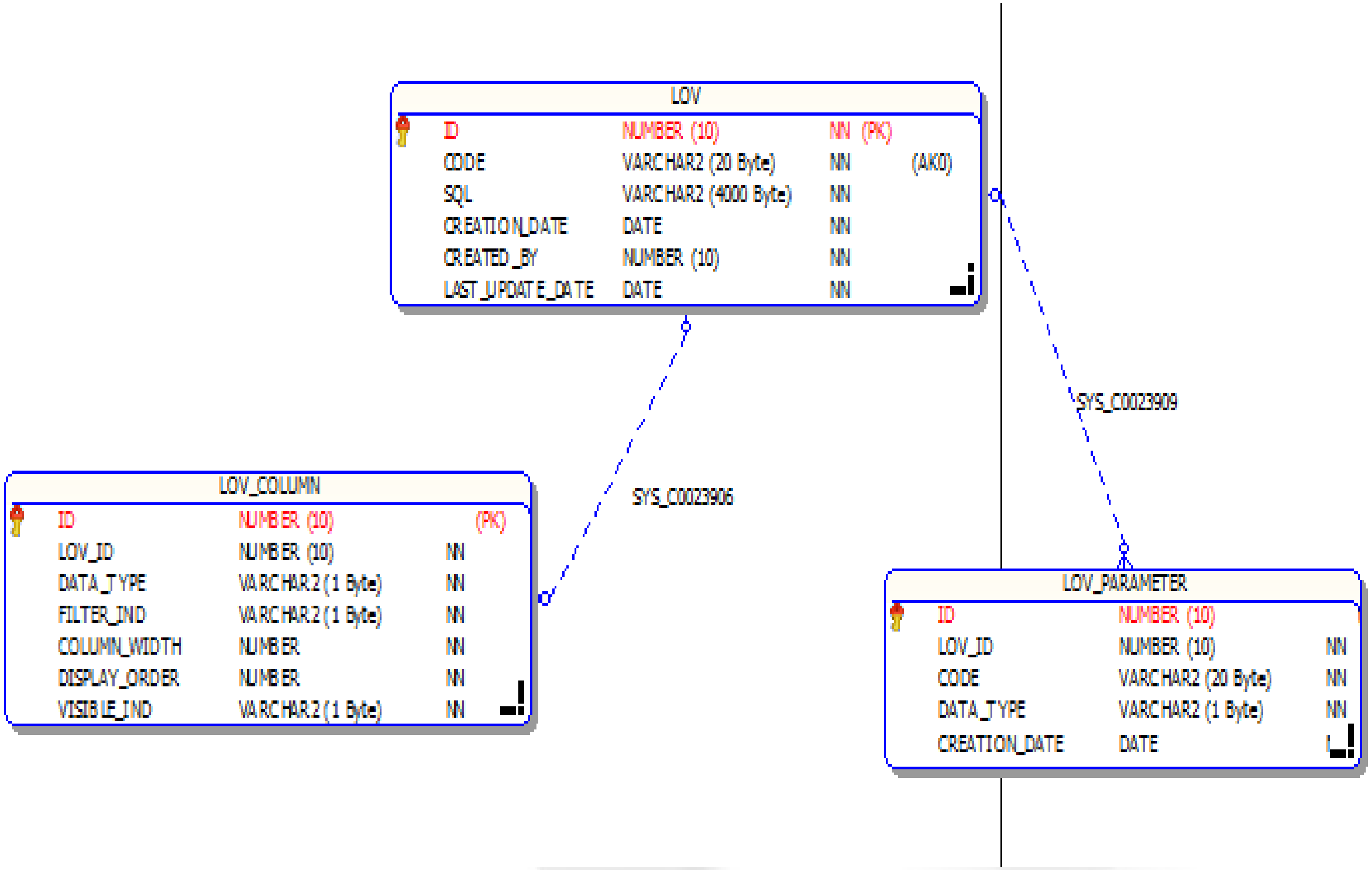


```
public static DynamicRichInputText execute(String beanName, String propertyName){
    DynamicRichInputText input = new DynamicRichInputText();

    input.addValueExpression(input, beanName, propertyName);

    return input;
}

private static void addValueExpression(DynamicRichInputText input, String beanName, String propertyName){
    FacesContext facesContext = FacesContext.getCurrentInstance();
    ExpressionFactory expressionFactory = facesContext.getApplication().getExpressionFactory();
    ValueExpression valExpr =
        expressionFactory.createValueExpression(facesContext.getELContext(),
                                                "#{"+beanName+"."+propertyName+"}",
                                                String.class);
    input.setValueExpression("value",valExpr);
}
```

Lista vrijednosti



Metamodel liste vrijednosti



LOV

ID	NUMBER (10)	NN (PK)	
CODE	VARCHAR2 (20 Byte)	NN	(AK0)
SQL	VARCHAR2 (4000 Byte)	NN	
CREATION_DATE	DATE	NN	
CREATED_BY	NUMBER (10)	NN	
LAST_UPDATE_DATE	DATE	NN	

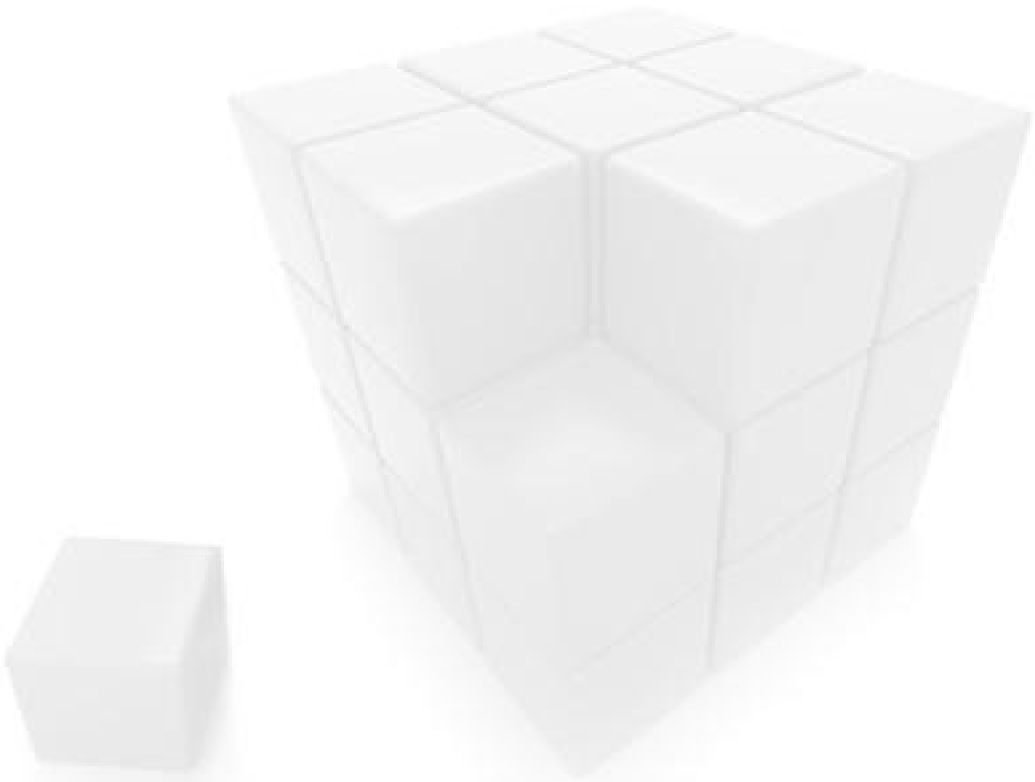
LOV_COLUMN

ID	NUMBER (10)	(PK)
LOV_ID	NUMBER (10)	NN
DATA_TYPE	VARCHAR2 (1 Byte)	NN
FILTER_IND	VARCHAR2 (1 Byte)	NN
COLUMN_WIDTH	NUMBER	NN
DISPLAY_ORDER	NUMBER	NN
VISIBLE_IND	VARCHAR2 (1 Byte)	NN

LOV_PARAMETER

ID	NUMBER (10)	
LOV_ID	NUMBER (10)	NN
CODE	VARCHAR2 (20 Byte)	NN
DATA_TYPE	VARCHAR2 (1 Byte)	NN
CREATION_DATE	DATE	

Demo



ZAKLJUČAK

PITANJA?