

Использование аннотаций в J2SE 5

Денис Цыплаков

В данной статье приводится пример использования аннотаций в J2SE 5.0, а также некоторые соображения по поводу области применимости аннотаций.

1. Введение

В конце сентября 2004 вышел релиз платформы J2SE 5.0 "Tiger". Он содержит великое множество нововведений. В принципе о всех о них можно почитать в формальной спецификации или в статьях появляющихся на <http://java.sun.com/>. Статьи эти естественно на английском. Я довольно неплохо могу читать технический английский текст. Но изучать что-то новое зачастую и так непросто, а перевод с неродного языка еще больше осложняет проблему.

Я решил начать планомерное изучение новых возможностей J2SE 5.0 с аннотаций. Аннотации — они же **JSR-175 (Meta-data)**, механизм позволяющий связывать с пакетами, классами (интерфейсами), полями, методами и локальными переменными структурированную информацию. Механизм `java.reflection` позволяет получить к данной мета информации доступ на стадии исполнения программы. Подробнее об аннотациях см. список ссылок в конце статьи. Аннотации интересны прежде всего при разработке различных инструментальных библиотек. Например для библиотек O/R мапинга можно указывать имена таблиц и имена полей, соответствующих классам и полям Java не в отдельных дескрипторах, а непосредственно в коде.

Итак собственно пример использования аннотаций.

2. Постановка задачи

У нас есть некоторый Java объект. Он имеет ряд полей. В примере ограничим диапазон типов полей одним типом — `String`. Мы хотим сделать набор классов (библиотеку если говорить строго), позволяющий автоматически строить Swing диалоги для редактирования полей класса.

3. Ограничения

Мы не ставим задачи сделать готовую библиотеку пригодную для использования в реальных проектах. То, что мы будем делать — прототип демонстрирующий возможности аннотаций. Это значит что некоторые вещи, не связанные с демонстрацией аннотаций, мы будем делать не как правильнее, а как проще. Например, забегая вперед, диалог у нас будет строится довольно неуклюже на основе `GridLayout`. Правильнее было бы использовать `GridBagLayout`, вложенную, `ScrollPane` и т.п. Но в данном случае это для нас не важно.

4. Решение

Мы будем решать задачу в несколько итераций, постепенно улучшая решение.

4.1. Итерация 1

Начнем с того, что опишем аннотации

Файл	ru/yandex/lc/annd/VisualBean.java
Описание	Аннотация класса который мы хотим отображать в диалоговой форме. Служит для указания заголовка формы.

```
package ru.yandex.lc.annd;

import java.lang.annotation.*;

//Аннотация доступна во время исполнения.
//Есть три степени видимости
// SOURCE — аннотация выбрасывается после компиляции.
// CLASS — аннотация сохраняется в .class файле, но выбрасывается при запуске.
```

Использование аннотаций в J2SE 5

```
// RUNTIME — аннотация видна все всегда
@Retention(RetentionPolicy.RUNTIME)

//Аннотация применима только к типам т.е. классам и интерфейсам.
@Target(ElementType.TYPE)

public @interface VisualBean
{
    //Если аннотация содержит только одно поле и мы хотим заполнять его
    //без указания имени поля например: @VisualBean("some value"), то
    //поле должно называться value
    String value();
}
```

Файл	ru/yandex/lc/annd/VisualBeanField.java
Описание	Аннотация класса который мы хотим отображать в диалоговой форме. Служит для указания заголовка формы. Аннотация поля класса. Содержит видимое на диалоговой форме имя класса, значение по умолчанию и признак того что поле скрытое т.е. должно заполняться звездочками.

```
package ru.yandex.lc.annd;

import java.lang.annotation.*;

@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.FIELD)

public @interface VisualBeanField
{
    String name();
    String initialValue();

    //Значение по умолчанию "no"
    String hidden() default "no";
}
```

Теперь опишем собственно класс который мы хотим показывать на форме. В названии

класса есть слово Bean, хотя строго говоря как раз Java Бин он не является. У него нет геттеров и сеттеров. Просто тут мы немного упрощаем себе задачу.

Файл	ru/yandex/lc/annd/TestBean.java
Описание	Собственно аннотированный класс — носитель данных.

```
package ru.yandex.lc.annd;

@VisualBean("Тестовый бин")
public class TestBean
{
    @VisualBeanField(
        name = "Имя пользователя",
        initialValue = "Администратор"
    )
    public String user;

    @VisualBeanField(
        name = "Пароль",
        initialValue = "",
        hidden = "yes"
    )
    public String password;

    //Поле класса, не аннотированно и на форме выводится не будет.
    public String otherField;
}
```

Класс описан и аннотирован. Теперь опишем JDialog который будет эту форму показывать пользователю.

Файл	ru/yandex/lc/annd/VisualBeanDialog.java
Описание	Диалоговая форма автоматически выстраивающая поля для ввода значений класса данных.

```
package ru.yandex.lc.annd;

import javax.swing.*;
```

Использование аннотаций в J2SE 5

```
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.*;
import java.lang.reflect.Method;
import java.lang.reflect.Field;
import java.util.ArrayList;

public class VisualBeanDialog extends JDialog
{
    Object visualBean;

    java.util.List<java.lang.reflect.Field> fList;
    java.util.List<JTextField> textFieldList;

    public void setVisualBean(Object visualBean)
    {
        this.visualBean = visualBean;
        if (!visualBean.getClass().isAnnotationPresent(VisualBean.class))
        {
            throw new Error("visualBean должен иметь аннотацию VisualBean");
        }
        //Устанавливаем заголовок диалога из аннотации визуального бина.
        this.setTitle(
            visualBean.getClass().getAnnotation(VisualBean.class).value());
        fList = new ArrayList<java.lang.reflect.Field>();

        //Бежим по полям и отбираем аннотированные поля типа String
        Field[] fs = visualBean.getClass().getFields();
        for (Field f : fs)
        {
            if (
                (f.isAnnotationPresent(VisualBeanField.class)) &&
                (f.getType().equals(String.class))
            )
            {
                fList.add(f);
            }
        }

        JPanel panel = new JPanel();
```

```
panel.setLayout(new GridLayout(fList.size() + 1, 2));
setContentPane(panel);

//Помещаем поля на форму.
textFieldList = new ArrayList<JTextField>();
for (Field f : fList)
{
    panel.add(
        new JLabel(f.getAnnotation(VisualBeanField.class).name()));
    JTextField textField;
    if (f.getAnnotation(VisualBeanField.class).hidden().equals("yes"))
    {
        textField = new JPasswordField();
    }
    else
    {
        textField = new JTextField();
    }
    try
    {
        if (f.get(vizualBean) == null)
        {
            textField.setText(f.getAnnotation(VisualBeanField.class)
                .initialValue());
        }
        else
        {
            textField.setText(f.get(vizualBean).toString());
        }
    }
    catch (IllegalAccessException e)
    {
        e.printStackTrace();
    }
    textFieldList.add(textField);
    panel.add(textField);
}

//Создаем кнопки
JButton btnCancel = new JButton("Cancel");
```

Использование аннотаций в J2SE 5

```
btnCancel.addActionListener(new CancelListener());
panel.add(btnCancel);
JButton btnOK = new JButton("OK");
btnOK.addActionListener(new OKListener());
panel.add(btnOK);
setLocation(300, 300);
pack();
}

//Обработчик нажатий на кнопку OK
class OKListener implements ActionListener
{

    public void actionPerformed(ActionEvent e)
    {
        for (int i = 0; i < textFieldList.size(); i++)
        {
            try
            {
                fList.get(i).set(visualBean,
                                textFieldList.get(i).getText());
            }
            catch (Throwable e1)
            {
                e1.printStackTrace();
            }
        }
        dispose();
    }
}

//Обработчик нажатий на кнопку Cancel
class CancelListener implements ActionListener
{

    public void actionPerformed(ActionEvent e)
    {
        dispose();
    }
}
```

```
}

```

А теперь опишем класс который все это запускает.

Файл	ru/yandex/lc/annd/Starter.java
Описание	Точка запуска.

```
package ru.yandex.lc.annd;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

public class Starter extends JFrame
{
    static Starter frame;

    TestBean tb = new TestBean();

    public static void main(String[] args)
    {
        frame = new Starter();
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel mainPane = new JPanel();
        mainPane.setLayout(new BorderLayout());
        frame.setContentPane(mainPane);

        JButton btnDo = new JButton("Hy?");
        btnDo.addActionListener(new ShowBeanDialogAction(frame.tb));
        mainPane.add(btnDo);

        frame.pack();
        frame.setLocation(200, 100);
        frame.setVisible(true);
    }

    static class ShowBeanDialogAction implements ActionListener
    {
```

Использование аннотаций в J2SE 5

```
Object bean;

public ShowBeanDialogAction(Object bean)
{
    this.bean = bean;
}

public void actionPerformed(ActionEvent e)
{
    VisualBeanDialog dlg = new VisualBeanDialog();
    dlg.setVisualBean(bean);
    dlg.setVisible(true);
}
}
```

Вот так выглядит сама форма:

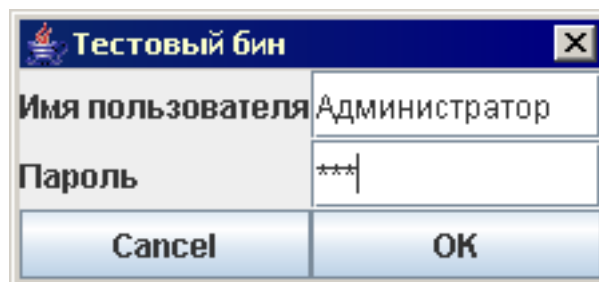


Рисунок 1. Пример диалоговой формы.

4.2. Итерация 2

Усложним задачу. Пусть поля у нас будут 3-х типов. Просто поле ввода, поля для ввода пароля и combobox поле. Далее идут только те классы, код которых менялся.

Начнем с изменения и дополнения аннотаций:

Файл	ru/yandex/lc/annd/FieldTypeEnum.java
Описание	Перечислимое множество поддерживаемых типов полей.

```
package ru.yandex.lc.annd;
```

```
public enum FieldTypeEnum
{
    TEXT_FIELD,
    PASSWORD_FIELD,
    COMBOBOX_FIELD
}
```

Файл	ru/yandex/lc/annd/VisualBeanField.java
Описание	Дополненная аннотация поля.

```
package ru.yandex.lc.annd;

import java.lang.annotation.*;

@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.FIELD)

public @interface VisualBeanField
{
    String name();

    String initialValue();

    //Теперь тип поля мы задаем с помощью перечислимого множества
    FieldTypeEnum fieldType() default FieldTypeEnum.TEXT_FIELD;

    //Если тип поля COMBOBOX_FIELD — то в этом поле — массив
    //с набором допустимых значений
    ComboFieldValue[] comboValues() default {};
}
```

Файл	ru/yandex/lc/annd/ComboFieldValue.java
Описание	Аннотация с допустимым значением combobox поля.

```
package ru.yandex.lc.annd;

public @interface ComboFieldValue
{
    String value();
}
```

Использование аннотаций в J2SE 5

```
}
```

Теперь класс с данными будет выглядеть вот так:

Файл	ru/yandex/lc/annd/TestBean.java
Описание	Собственно класс с данными.

```
package ru.yandex.lc.annd;

@VisualBean("Параметры соединения с БД")
public class TestBean
{
    @VisualBeanField(
        name = "Имя пользователя",
        initialValue = "Администратор"
    )
    public String user;

    @VisualBeanField(
        name = "Пароль",
        initialValue = "",
        fieldType = FieldTypeEnum.PASSWORD_FIELD
    )
    public String password;

    @VisualBeanField(
        name = "Права доступа",
        initialValue = "User",
        fieldType = FieldTypeEnum.COMBOBOX_FIELD,
        comboValues = {
            @ComboFieldValue("Administrator"),
            @ComboFieldValue("User"),
            @ComboFieldValue("Guest")}
    )
    public String accessRights;

    public String otherField;
}
```

Соответственно нам придется усложнить класс с диалоговой формой.

Файл	ru/yandex/lc/annd/VisualBeanDialog.java
Описание	Измененная диалоговая форма.

```

package ru.yandex.lc.annd;

import javax.swing.*;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.*;
import java.lang.reflect.Field;
import java.util.ArrayList;

public class VisualBeanDialog extends JDialog
{
    Object visualBean;

    java.util.List<java.lang.reflect.Field> fList;
    java.util.List<JComponent> textFieldList;

    public void setVisualBean(Object visualBean)
    {
        this.visualBean = visualBean;
        if (!visualBean.getClass().isAnnotationPresent(VisualBean.class))
        {
            throw new Error("visualBean должен иметь аннотацию VisualBean");
        }
        //Устанавливаем заголовок диалога из аннотации визуального бина.
        this.setTitle(
            visualBean.getClass().getAnnotation(VisualBean.class).value());
        fList = new ArrayList<java.lang.reflect.Field>();

        Field[] fs = visualBean.getClass().getFields();
        for (Field f : fs)
        {
            if (
                (f.isAnnotationPresent(VisualBeanField.class)) &&
                (f.getType().equals(String.class))
            )
            {

```

Использование аннотаций в J2SE 5

```
fList.add(f);
    }
}

JPanel panel = new JPanel();
panel.setLayout(new GridLayout(fList.size() + 1, 2));
setContentPane(panel);

textFieldList = new ArrayList<JComponent>();
for (Field f : fList)
{
    panel.add(
        new JLabel(f.getAnnotation(VisualBeanField.class).name()));
    JComponent field;
    VisualBeanField fieldAnnotation =
        f.getAnnotation(VisualBeanField.class);
    //Обратите внимание как выглядит switch по перечислимому типу.
    switch (fieldAnnotation.fieldType())
    {
        case TEXT_FIELD:
        {
            field = new JTextField();
            break;
        }
        case PASSWORD_FIELD:
        {
            field = new JPasswordField();
            break;
        }
        case COMBOBOX_FIELD:
        {
            field = new JComboBox();
            //В цикле бежим по вложенному массиву аннотаций
            for (ComboFieldValue cfv : fieldAnnotation.comboValues())
            {
                ((JComboBox) field).addItem(cfv.value());
            }
            break;
        }
    }
}
```

```
        default :
        {
            throw new Error ( "Что-то неладно с enum");
        }
    }

    try
    {
        if (field instanceof JTextField)
        {
            if (f.get(visualBean) == null)
            {
                ((JTextField) field).setText(
                    f.getAnnotation(VisualBeanField.class)
                        .initialValue());
            }
            else
            {
                ((JTextField) field).setText(f.get(visualBean)
                    .toString());
            }
        }
        else if (field instanceof JComboBox)
        {
            if (f.get(visualBean) == null)
            {
                ((JComboBox) field).setSelectedItem(
                    fieldAnnotation.initialValue());
            }
            else
            {
                ((JComboBox) field).setSelectedItem(f.get(visualBean)
                    .toString());
            }
        }
    }
    catch (IllegalAccessException e)
    {
        e.printStackTrace();
    }
}
```

Использование аннотаций в J2SE 5

```
        textFieldList.add(field);
        panel.add(field);
    }
    JButton btnCancel = new JButton("Cancel");
    btnCancel.addActionListener(new CancelListener());
    panel.add(btnCancel);
    JButton btnOK = new JButton("OK");
    btnOK.addActionListener(new OKListener());
    panel.add(btnOK);
    setLocation(300, 300);
    pack();
}

class OKListener implements ActionListener
{
    public void actionPerformed(ActionEvent e)
    {
        for (int i = 0; i < textFieldList.size(); i++)
        {
            try
            {
                JComponent jc = textFieldList.get(i);
                if (jc instanceof JTextField)
                {
                    fList.get(i).set(visualBean,
                                    ((JTextField) jc).getText());
                }
                else if (jc instanceof JComboBox)
                {
                    fList.get(i).set(visualBean,
                                    ((JComboBox) jc).getSelectedItem());
                }
            }
            catch (Throwable e1)
            {
                e1.printStackTrace();
            }
        }
    }
    dispose();
}
```

```
    }  
}  
  
class CancelListener implements ActionListener  
{  
    public void actionPerformed(ActionEvent e)  
    {  
        dispose();  
    }  
}
```

В новом варианте форма выглядит так:

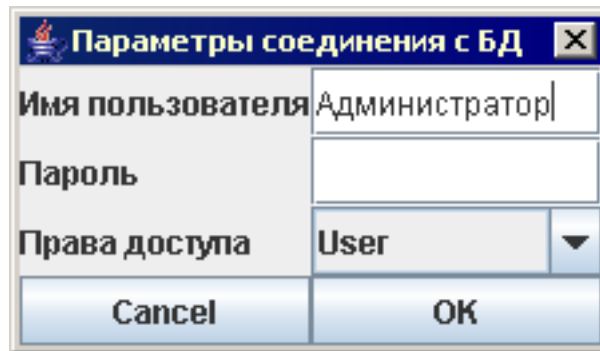


Рисунок 2. Пример диалоговой формы 2.

5. Замечания

Необходимо заметить, что в данной статье не рассматриваются вопросы области применимости аннотаций. Как всякий инструмент аннотации хороши для выполнения определенных задач. Вопрос применимости аннотаций — тема для отдельной статьи.

6. Список ссылок

1. [Дискуссия в эхоконференции fido7.ru.java по поводу данной статьи](#). Самое правильное место чтобы поругать то, что Вам не нравится
2. [Статья с java.sun.com в которой рассматривается использование аннотаций](#). С этой статьи все началось.
3. [Официальный сайт JSR-175 Metadata](#)
4. [Архив с исходными текстами](#)