Факультет «Информатика и системы управления»

Методические указания к лабораторной работе

по курсу «Распределенные системы обработки информации»

*"Создание компонентов JavaBeans"*

Москва 2004 г.

# *Цель работы*

1. Изучить основные понятия и назначение технологии JavaBeans.
2. Научиться создавать jar‑файлы в среде Eclipse.
3. Изучить структуру файлов манифеста.
4. Познакомиться с созданием многопоточных приложений.
5. Освоить простой визуальный редактор Java BeanBox.

# 

# *Задание для домашней подготовки*

Ознакомиться с теоретическим материалом, представленным в приложениях к данным методическим указаниям и примерами программ. Ознакомиться с текстом задания к лабораторной работе, предложить размещение компонентов и функциональность, удовлетворяющую требованиям задания к лабораторной работе, и написать программу.

# *Задание к лабораторной работе*

Необходимо создать компонент JavaBean, работающий в роли таймера. Он должен выводить количество срабатываний с момента активации. Интервал таймера задаётся в свойствах компонента.

# *Содержание отчета*

Отчет должен содержать:

1. Постановку задачи, решаемой отлаженной программой.
2. Руководство пользователя отлаженной программы, содержащее описание интерфейсов всех функций программы.
3. Листинг программы с необходимыми комментариями.

***Контрольные вопросы***

1. Где используются компоненты JavaBeans?
2. Что представляет собой процесс создания компонента?
3. В каком месте данной работы необходимо использование потоков?
4. Что такое файл манифеста?
5. Для чего нужны jar‑архивы?
6. Для чего нужен редактор BeanBox?
7. Как увидеть свой компонент в действии?

# 

# *Литература*

Официальные источники:

1. Кен Арнольд, Джеймс Гослинг, Дэвид Холмс. Язык программирования Java™.
2. Официальный сайт Java – http://java.sun.com/ (есть раздел на русском языке с учебником).
3. Java™ 2 SDK, Standard Edition Documentation – http://java.sun.com/products/jdk/1.5/index.html.
4. Джеймс Гослинг, Билл Джой, Гай Стил. Спецификация языка Java (The Java Language Specification – http://www.javasoft.com/docs/books/jls/). Перевод на русский язык – http://www.uni-vologda.ac.ru/java/jls/index.html
5. Официальный сайт проекта Eclipse – http://www.eclipse.org/.
6. BeanSDK Documentation.

Другое:

1. Дмитрий Рамодин. Начинаем программировать на языке Java.
2. Николай Смирнов. Java 2: Учебное пособие.
3. Картузов А.В. Программирование на языке Java.
4. Вязовик Н.А. Программирование на Java.
5. Алексей Литвинюк. Введение в интегрированную среду разработки Eclipse – http://lib.juga.ru/article/articleview/174/1/69/.

# *Приложение 1: Общие сведения о компонентах JavaBeans*

Официальное определение компонента JavaBean в документации BeanSDK выглядит следующим образом: «Компонент – это повторно используемый элемент программного обеспечения, основанный на спецификации JavaBeans компании Sun Microsystems, которым можно управлять визуально с помощью инструментов интегрированной разработки». Аналогом компонентов JavaBeans являются ActiveX‑компоненты (кнопки, переключатели, флажки, текстовые поля и т.п.), которые изучались в курсе «визуальное программирование на Visual Basic».

Программировать при помощи компонентов JavaBeans очень просто – достаточно разместить необходимые компоненты в окне и описать их взаимодействие. Как видите, это очень похоже на разработку программ в визуальных средах программирования (VB, Delphi, Visual C++ и т.п.). Нас же будет интересовать создание самих компонентов.

По своей сути компонент JavaBean это обычный класс (или несколько) с названиями методов, задаваемыми по специальным правилам. После создания \*.class файлов происходит их упаковка в архив (файл jar – **j**ava **ar**chive). Для задания информации о компоненте используют файл манифеста (см. приложение). После этого компонент готов к распространению.

# *Приложение 2: Требования к именам методов компонента*

Любой компонент JavaBean (это относится в равной степени и к ActiveX компонентам других языков программирования) характеризуется следующими параметрами:

– свойствами

– генерируемыми событиями

Например, для компонента «кнопка» свойствами могут быть текст кнопки, её размеры и т.п.; события – нажатие на кнопку, перевод фокуса.

Свойства:

Для задания свойства X используют два метода, один для чтения содержимого свойства, другой – для изменения. Их имена соответственно: «getX» и «setX». Для создания свойства X с неизменяемым (константным) значением достаточно создать только метод «getX».

Итак, для создания свойства «FileName» необходимо создать методы:

public String getFileName()

{

 // Код метода

}

public void setFileName()

{

 // Код метода

}

Исключением являются свойства, принимающие логические (булевы) значения. Имена их начинаются на «is» для получения значения и «set» для задания. Таким образом для создания свойства Enabled (активно) необходимы следующие методы:

public boolean isEnabled()

{

 // Код метода

}

public void setEnabled()

{

 // Код метода

}

События:

Для создания пользовательского события X, необходимо выполнить следующие шаги:

– Создать класс XEvent (название обязательно должно заканчиваться на Event) на основе класса EventObject

– Создать интерфейс XListener производный от EventListener с методами уведомления.

– Добавить в компонент JavaBean следующие методы:

public void addXListener (XListener listener) {

listenerList.add (XListener.class listener);

}

public void removeXListener (XListener listener) {

listenerList.remove (XListener.class listener);

}

Дополнительно можно создать метод активации события

public void fireXEvent (XEvent event)

{

 // Внутренний код, вызывающий событие XEvent

}

# *Приложение 3: Многопоточность*

Для одновременного выполнения различных процессов, например, взаимодействия с пользователем, интенсивных вычислений, построения графика пользуются потоки. В обычной программе присутствует лишь один поток, но одновременное выполнение требует создания дополнительных.

В Java поток создаётся либо порождением своего класса от класса Thread, либо созданием класса, реализующего интерфейс Runnable. Необходимо определить метод run(), отвечающий за процесс потока. После окончания выполнения метода run() поток перестаёт существовать.

Для задержки выполнения потока на определённое время используется метод Thread.sleep (int time), где time – время задержки в миллисекундах. Чтобы принудительно прервать выполнение потока используют метод pleaseStop().

Для предотвращения ситуации, когда два потока одновременно модифицируют один и тот же объект используется *синхронизация потоков*. Когда поток начинает модификацию объекта, происходит его *блокировка*, так что другие потоки вынуждены ждать окончания процесса модификации. Для создания методов, использующих синхронизацию потоков, в заголовке метода пишется ключевое слово synchronized.

При работе с синхронизированными методами может возникнуть ситуация *взаимоблокировки*(deadlock). Суть проблемы в следующем: два синхронизированных метода могут блокировать друг друга, так что ни один из потоков не может ни снять блокировку, ни продолжить выполнение. Данная ошибка является одной из наиболее часто встречающихся при работе с потоками.

# *Приложение 4: Создание jar‑файлов в Eclipse. Формат файла манифеста*

Файл манифеста хранит в себе информацию о содержимом архива jar. Одной из наиболее важных возможностей, предоставляемых манифестами, является создание автоматически запускающихся архивов.

Применительно к компонентам JavaBeans манифесты используют для информирования среды разработки о наличии компонента в jar архиве.

Формат манифеста для компонента JavaBean выглядит следующим образом:

1. Manifest-Version: 1.0

2. Sealed: true

3. Name: TimerBean.class

4. Java-Bean: True

Обратите внимание, последняя строка обязательно должна быть пустой. Номера строк в тексте файла не пишутся. При сохранении, в файле манифеста отсутствует расширение (или присутствует \*.mf для некоторых версий).

Пример команды для сборки jar‑файла:

> jar cfm NewBean.jar manifest NewBean.class

# *Приложение 5: Использование компонента в среде разработки BeanBox*

В состав BeanSDK 1.1 входит простейшая среда для тестирования JavaBeans компонентов BeanBox. Найти её можно в папке JDK FOLDER\beans\beanbox\

Для тестирования работы компонента необходимо поместить jar файл с подготовленным манифестом в папку JDK FOLDER\beans\jars\. В ней уже есть несколько заранее созданных JavaBeans.

После этого, при загрузке среды в списке доступных компонентов появится тестируемый компонент. Для использования достаточно перетащить его на окно формы.

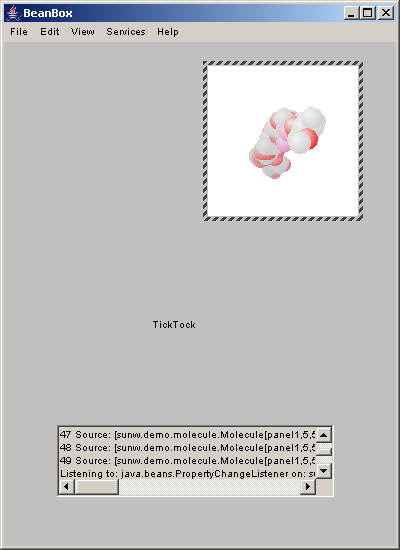


Рис. 1: Окно формы BeanBox