Теоретический материал для индивидуального задания №9

JavaBeans

(продолжение)

Более сложный компонент

В примере 1 показан один компонент - `YesNoPanel`. Этот компонент отображает для пользователя сообщение (при помощи `MultiLineLabel`) и три кнопки. При щелчке мышью на этих кнопках он генерирует событие. Компонент `YesNoPanel` предназначен для использования внутри диалоговых окон, так как он предлагает глобальный спосо6 задать пользователю вопрос «Да/Нет».

Для уведомления объекта `AnswerListener` о том, что пользователь нажал одну из кнопок, компонент `YesNoPanel` использует пользовательский тип `AnswerEvent`.

Компонент `YesNoPanel` не пользуется классами из пакета `java.beans`. Одним из необычных свойств компонентов `JavaBean` является то, что им обычно не нужно использовать какие-либо классы из этого пакета. Трудности для использования этого пакета создают вспомогательные классы, распространяемые вместе с компонентом.

Пользовательские события

Компоненты могут использовать стандартные типы событий, определенные в пакетах `java.awt.event` и `javax.swing.event`, но они не обязаны ограничиваться ими. Наш класс `YesNoPanel` определяет собственный тип событий - `AnswerEvent`. Определить новый класс событий на самом деле достаточно легко.

Вместе с классом `AnswerEvent` в компоненте `YesNoPanel` определяется новый тип интерфейса слушателей событий, `AnswerListener`, содержащий методы, которые обязательно должны быть реализованы любым объектом, заинтересованным в получении уведомлений от компонента `YesNoPanel`.

Предоставление информации о компоненте

Сам класс `YesNoPanel`, как и классы `MultiLineLabel`, `Alignment`, `AnswerEvent` и `AnswerListener`, которыми он пользуется, является необходимой частью нашего компонента. Когда приложение» использующее данный компонент, распространяется, оно должно содержать все пять файлов класса. Однако часто имеются и другие виды классов, связанные с компонентом, но не предназначенные для использования разработчиками приложений. Эти классы используются контейнерным средством, которое манипулирует данным компонентом. Сам класс компонента не ссылается ни на один из этих вспомогательных классов контейнера, поэтому он не зависит от них, и они не обязаны поставляться вместе с компонентом в составе законченного продукта.

Первым из этих необязательных, вспомогательных классов является класс `BeanInfo`. Контейнер обнаруживает экспортируемые компонентом свойства, события и методы с помощью механизма интроспекции, основанного на программном интерфейсе отражения

Java. Разработчик компонента, желающий предоставить дополнительную информацию о компоненте или уточнить информацию, доступную через интроспекцию, должен для предоставления этой информации определить класс, реализующий интерфейс BeanInfo. Класс BeanInfo обычно наследуется от SimpleBeanInfo, который предоставляет пустую реализацию интерфейса BeanInfo. В том случае, когда вы хотите заместить только один или два метода, это проще сделать путем создания класса, производного от SimpleBeanInfo, чем реализовать BeanInfo непосредственно.

Контейнерные инструментальные средства при поиске класса BeanInfo, относящегося к данному компоненту, предполагают использование следующего соглашения об именах: класс BeanInfo должен иметь то же имя, что и компонент, с добавлением в конец строки «BeanInfo».

В нашем примере это класс YesNoPanelBeanInfo. Он определяет для нашего компонента несколько информационных частей:
- Изображение, представляющее компонент.
- Объект BeanDescriptor, содержащий ссылку на класс Customizer, относящийся к компоненту.
- Список поддерживаемых свойств компонента с кратким описанием каждого. Некоторые контейнерные средства показывают эти строки пользователю в некотором удобном для него виде.
- Метод, возвращающий наиболее часто используемое свойство компонента. Оно называется «свойством по умолчанию» (default property).
- Ссылка на класс PropertyEditor для одного из свойств.

Помимо предоставления этой информации, класс BeanInfo может предоставлять информацию о методах, которые в нем определены, и событиях, которые он генерирует. Различные объекты FeatureDescriptor, предоставляющие информацию по свойствам и методам, могут содержать и другую информацию, не предоставляемую объектом YesNoPanelBeanInfo, такую как локализованное имя для отображения, которое отличается от программного имени.

Демонстрационные примеры (в папке Примеры)

Дополнительный материал смотрите в:
- Java-УП-09-1.ppt
- главе 7 книги _Books\corejava2_2.djvu

Литература

1. Хабибуллин И. Ш. Java 7. — СПб.: БХВ-Питербург, 2012
2. Г. Шилдт. Java . Полное руководство, 8-е издание, М.: ООО “И.Д. Вильямс”,2012
3. Кей С. Хорстман. Java2 Основы. Том 1. С.-Питербург. 2007 (_Books\ corejava2_1.djvu)
4. Кей С. Хорстман. Java2 Тонкости программирования. Том 2. С.-Питербург. 2007 (_Books\ corejava2_2.djvu)
5. http://docs.oracle.com/javase/8/docs/