## **Задание 6**

Распакуйте архив “Задание 6.zip” в Ваш рабочий каталог. В Вашем проекте после этого должны появиться классы классы TestComboBox, TestCursor, TestDesktop, TestListBox, TestPoint, TestRadioButton, TestRectButton, TestRectButton3D, TestRectWindow, TestRoundButton, TestRoundWindow, TestTypes, TestWindow и TestWindowFactory в каталоге test/java в пакете net.thumbtack.school.windows.v4

На этом занятии мы создадим четвертую версию наших оконных классов, изменив их поведение, и добавим обработку ошибок.

Создайте пакет net.thumbtack.school.windows.v4 в каталоге main/java и скопируйте в него все классы из пакета net.thumbtack.school.windows.v3. Не забудьте изменить в каждом классе package!

1. Состояние окна.

В ходе разработки выяснилось, что двух вариантов для состояния окна (активно или пассивно) недостаточно. Окно может находиться еще и в состоянии “разрушено”, в него окно переходит, когда его закрывают, но оно еще окончательно не закрыто. Поэтому вместо boolean active нам придется ввести поле, которое может принимать 3 значения : ACTIVE, INACTIVE и DESTROYED. При этом надо иметь в виду, что переходы из состояния ACTIVE или INACTIVE в другое состояние разрешены, а вот переход из состояния DESTROYED в любое иное состояние не разрешается - такое окно “оживить” уже нельзя. Кроме того, нельзя создавать окно с состоянием DESTROYED.

Создайте в пакете net.thumbtack.school.windows.v4.base enum WindowState со следующими значениями (порядок произвольный)

* ACTIVE
* INACTIVE
* DESTROYED

Измените Ваши оконные классы таким образом, чтобы вместо поля **boolean active** в них хранилось одно из значений этого enum. Аналогично измените все конструкторы, сеттеры и геттеры. Вместо isActive/getActive введите setState/getState.

Кроме того, необходимо сделать, чтобы состояние окна могло передаваться не только одной из enum констант, но и одной из трех текстовых строк : “ACTIVE”, “INACTIVE”, “DESTROYED”. Для этого в enum WindowState добавьте метод

**public static WindowState fromString(String stateString)**

Возвращает WindowState по переданной текстовой строке

Напишите новые конструкторы в каждом классе, для которых вместо WindowState передается такая текстовая строка.

1. Форма курсора

Аналогично, для курсора вместо числового номера введем несколько форм

* ARROW // стрелка, направленная на северо-запад
* UPARROW // вертикальная стрелка
* CROSS // перекрестие
* HELP // стрелка с вопросительным знаком
* WAIT // крутящееся колесико или песочные часы

Создайте в пакете net.thumbtack.school.windows.v4.cursors enum CursorForm с вышеприведенными значениями (порядок произвольный) и замените в классе Cursor поле целого типа на поле типа CursorForm.

1. Обработка ошибок

Теперь у нас состояние окна передается с помощью enum, а значит, в качестве этого параметра может быть передан null. В рамках данной задачи передачу null для WindowState мы разрешать не будем - это не имеет смысла. Аналогично, null не может быть допустимым значением для текстовой строки в методе WindowState.fromString. В обоих случаях должно выбрасываться исключение.

Для класса Cursor мы допустим значение поля типа CursorForm, равное null - это будет означать, что форма курсора не определена. Задавать форму курсора текстовой строкой мы не будем, поэтому установка поля CursorForm не будет вызывать исключений.

Исключение должно выбрасываться и в ряде других случаев, например, при передаче методу WindowState.fromString текстовой строки, не входящей в вышеприведенный список, при попытке создать окно в состоянии DESTROYED, получить из ListBox строку с номером, большим чем (количество строк - 1), установить в ComboBox номер выбранной строки, больший чем (количество строк - 1) и др.

Создайте в пакете net.thumbtack.school.windows.v4.base enum WindowErrorCode со следующими значениями (порядок произвольный)

|  |  |
| --- | --- |
| Значение | Примечание |
| WRONG\_STATE | При создании окна передается WindowState.DESTROYED или null. При изменении состояния состояние устанавливается в null или окно, находящееся в WindowState.DESTROYED, переводится в иное состояние. |
| WRONG\_INDEX | Передан недопустимый индекс для массива строк. |
| EMPTY\_ARRAY | Массив строк равен null. |

В этом enum создайте текстовое поле errorString - понятное для человека описание ошибки и следующие конструктор и методы : конструктор, принимающий текстовую строку и записывающий ее в поле errorString и геттер для errorString. Обращаем внимание, что атрибут public для этого конструктора ставить нельзя- для enum все конструкторы могут быть только private.

Теперь создайте класс WindowException (наследник от класса Exception) - класс исключения, экземпляр которого должен выбрасываться при ошибке. В этом классе должно быть поле типа enum WindowErrorCode, конструктор, принимающий enum WindowErrorCode и геттер (getWindowErrorCode) для него. Разместите этот класс также в пакете net.thumbtack.school.window.base. Все методы, при выполнении которых возможна ошибка, должны выбрасывать это исключение.

Измените конструкторы и сеттеры ранее разработанных классов таким образом, чтобы при передаче им недопустимых значений выбрасывалось исключение.

Измените следующие методы, чтобы они удовлетворяли нижеприведенным требованиям.

**Класс ListBox**

1. **public String getLine(int number)**

Возвращает строку с номером index. Если массив строк равен null, выбрасывается исключение WindowException с кодом ошибки EMPTY\_ARRAY.Если строки с таким номером нет, выбрасывается исключение WindowException с кодом ошибки WRONG\_INDEX.

1. **public void setLine(int index, String line)**

Заменяет строку с номером index. Если массив строк равен null, выбрасывается исключение WindowException с кодом ошибки EMPTY\_ARRAY.Если строки с таким номером нет, выбрасывается исключение WindowException с кодом ошибки WRONG\_INDEX.

1. **public String[] getLinesSlice(int from, int to)**

Возвращает набор строк ListBox, начиная со строки “from” и до строки (“to”- 1) включительно. Если массив строк равен null, выбрасывается исключение WindowException с кодом ошибки EMPTY\_ARRAY. Если “from” < 0 или в ListBox строк меньше, чем “to” , или “from” > (“to” - 1), выбрасывается исключение WindowException с кодом ошибки WRONG\_INDEX.

**Класс ComboBox**

1. **Конструкторы класса ComboBox**

Если параметр lines равен null, параметр “selected” может быть только null, иначе выбрасывается исключение WindowException с кодом ошибки EMPTY\_ARRAY

Если делается попытка установить “selected”, больший чем (число строк - 1), выбрасывается исключение WindowException с кодом ошибки WRONG\_INDEX,

1. **public String getLine(int number)**

Возвращает строку с номером index. Если строки с таким номером нет, выбрасывается исключение WindowException с кодом ошибки WRONG\_INDEX. Если массив строк равен null, выбрасывается исключение WindowException с кодом ошибки EMPTY\_ARRAY

1. **public void setLine(int index, String line)**

Заменяет строку с номером index. Если строки с таким номером нет, выбрасывается исключение WindowException с кодом ошибки WRONG\_INDEX. Если массив строк равен null, выбрасывается исключение WindowException с кодом ошибки EMPTY\_ARRAY

1. **public void setSelected(Integer selected)**

Выбрасывается исключение WindowException с кодом ошибки EMPTY\_ARRAY, если текущее значение lines равно null, а значение параметра selected не равен null.

Выбрасывается исключение WindowException с кодом ошибки WRONG\_INDEX, если делается попытка установить номер, больший чем (число строк - 1).

1. **public void setLines(String[] lines)**

В результате выполнения этого метода номер выбранной строки всегда устанавливается в null.

1. **public String[] getLinesSlice(int from, int to)**

Возвращает набор строк ListBox, начиная со строки “from” и до строки (“to”- 1) включительно . Если “from” < 0 или в ListBox строк меньше, чем “to” , или “from” > (“to” - 1), выбрасывается исключение WindowException с кодом ошибки WRONG\_INDEX. Если массив строк равен null, выбрасывается исключение WindowException с кодом ошибки EMPTY\_ARRAY

Теперь запустите тесты с плагином Coverage , и определите, какие методы из разработанных классов не покрыты тестами. Добавьте в тестовые классы дополнительные тесты, проверяющие работу этих методов. Проверять работу метода hashCode не надо.

Примечание. При запуске тестов с Coverage , если у Вас Windows и имя каталога профиля (то есть Вашего каталога в C:\Users) содержит русские буквы, возможно исключение ArrayIndexOutOfBoundsException в самом плагине Coverage. В этом случае нужно

* Добавить строку

-Djava.io.tmpdir=C:\Temp

в файлах idea.exe.vmoptions и idea64.exe.vmoptions (файлы находятся в "C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 201x.y\bin"\)

* Убедиться, что каталог C:\Temp существует и доступен для записи
* Перезапустить IDEA

Проверьте работу тестов в консольном окне, запишите все классы на сервер (не забудьте изменить текст сообщения в git commit!) и убедитесь, что на сервере все тесты также проходят успешно (см. Занятие 1, п.15-19)