Назначение лекции

Знакомство с особенностями ввода целых и вещественных чисел.

Новые слова

**Тернарный оператор** – это условный оператор, который содержит три выражения.

Что было сделано

В предыдущех работе наше приложение MyEvent научилось выдавать список доступных режимов работы и выполнять контроль правильности номера режима. Если номер был введен не верно, то ввод повторялся.

Далее рассматрели реализацию режима «Ввод» в части ввода текущей даты и названия события.

Продолжение

В этой лекции продолжим рассмотрение реализации режима ввода в части ввода количество времени на событие (вещественное число) и определения состояния события.

В программе будет реализовано два состояния: «было» и «будет». Для состояния «будет» понадобится дата этого события.

Для того что бы выполнить ввод количества создадим новый класс WrTimeEv,

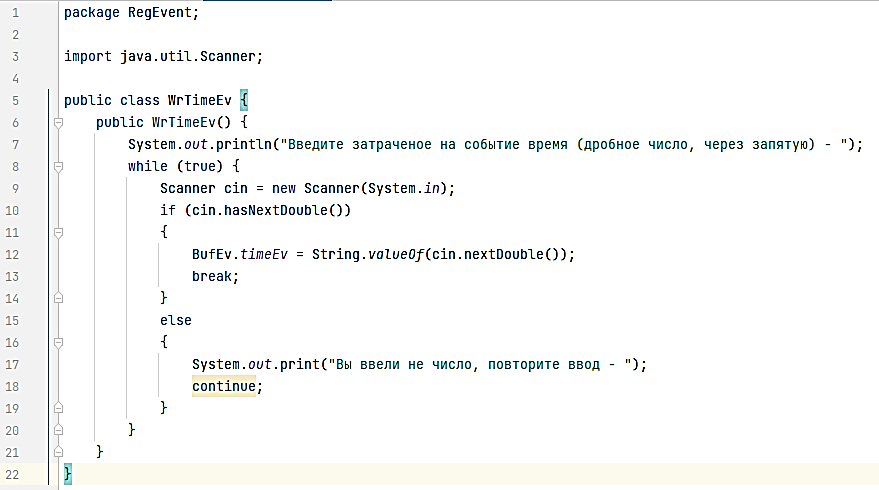
Может возникнуть вопрос: зачем создавать новые классы, если можно все делать в одной программе, например, в классе Main. Раньше так и программировали. Т.е. программа представляла собой единый код. Писать такую программу было легче, но сопровождать, корректировать – очень тяжело.

С появлением объектно-ориентированых языков программирования программная система – это набор объектов. Эти объекты взаимодействуют между собой. Такую программу гораздо легче сопровождать.

Кроме того к разработке можно привлекать нескольких программистов и наконец, появляется возможность распаралелить выполнение приложения в разных потоках при многопоточной обработке.

И наконец удачные классы можно использовать при разработке других приложений.

Итак создаем новый класс WrTimeEv и соответствующий конструктор. Вспомним, что код конструктора автоматически выполняется при создании объекта данного класса.



Прокомментируем каждую строку класса.

package RegEvent;

Заказ пакета , в котором хранятся все классы приложения.

import java.util.Scanner;

Импорт библиотеки для потокового ввода данных

System.*out*.println("Введите затраченое на событие время (дробное число, через запятую) - ");

Выдача на экран приглашения-подсказки. Если это не сделать, то пользователь не будет знать что делать.

while (true) {

Начало «бесконечного» цикла для ввода времени. Если ввод выполнен с ошибкой, то цикл повторяется. Если время введено верно, то цикл будет завершен через *break*

Scanner cin = new Scanner(System.*in*);

Ввод с клавиатуры в переменную cin Напомним, что ввод считается завершенным после нажатия клавиши *Enter*

if (cin.hasNextDouble())

Проверка на правильность ввода вещественного числа, например, введена буква или разделитель неправильный и пр. Выражение принимает значение true или false

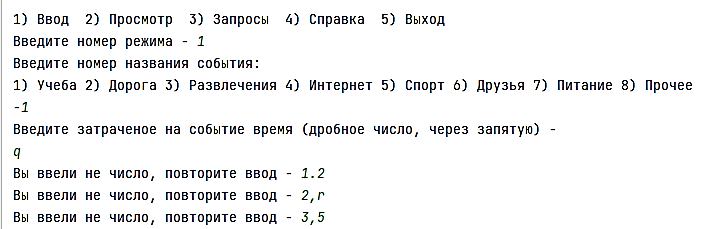
BufEv.*timeEv* = String.*valueOf*(cin.nextDouble());

break;

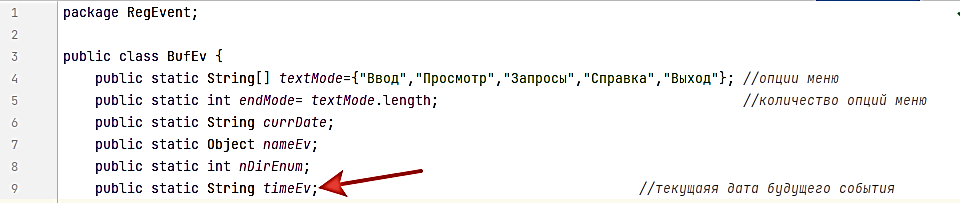
Преобразование числа cin в строку и запись в строковую переменную *timeEv* которая находится в классе BufEv . После этого цикл завершается.

System.*out*.print("Вы ввели не число, повторите ввод - ");  
continue;

Если ввод выполнен с ошибкой, то цикл повторяется. Далее приведен скриншот с ошибочным и верным вводом:

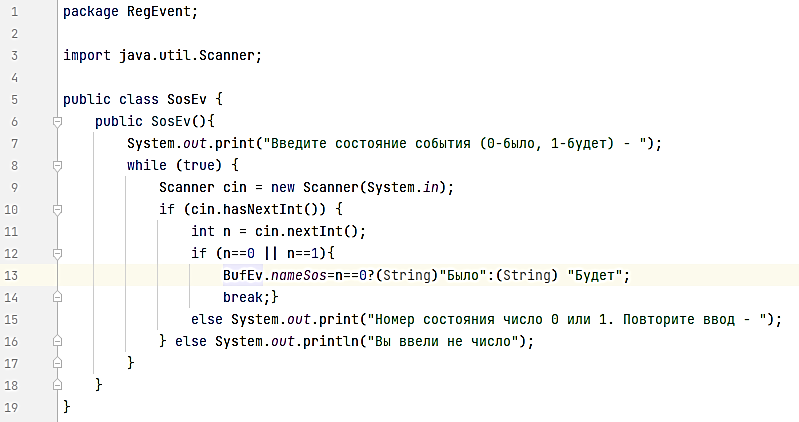


Для правильной работы понадобится дополнить класс BufEv



Далее для работы приложения требуется ввести состояние события. Их два: «Было» и «Будет». Состояние «Было» будем указывать в том случае если оно завершено, а состояние «Будет» - если оно запланировано на какую-либо дату.

Создадим новый класс SosEv и его конструктор:



Прокомментируем каждую строку класса.

package RegEvent;

Заказ пакета , в котором хранятся все классы приложения.

import java.util.Scanner;

Импорт библиотеки для потокового ввода данных

System.*out*.print("Введите состояние события (0-было, 1-будет) - ");

Выдача на экран приглашения-подсказки. Если это не сделать, то пользователь не будет знать что делать. Т.к. у нас консольное приложение, то для избежания лишних ошибок целесообразно при вводе использовать числа вместо текста.

while (true) {

Начало «бесконечного» цикла для ввода времени. Если ввод выполнен с ошибкой, то цикл повторяется. Если состояние введено верно, то цикл будет завершен через *break*

Scanner cin = new Scanner(System.*in*);

Ввод с клавиатуры в переменную cin Напомним, что ввод считается завершенным после нажатия клавиши *Enter*

if (cin.hasNextInt()) {

Проверка на правильность ввода целого числа, например, введена буква или вещественное число. Выражение cin.hasNextInt() принимает значение true или false

int n = cin.nextInt();

преобразование и запись в целую переменную *n* из потока cin

if (n==0 || n==1){

Дополнительная проверка на правильность ввода, т.к. состояние может принимать значение 0 или 1

BufEv.*nameSos*=n==0?(String)"Было":(String) "Будет";

break;}

Состояние будем хранить в строковом виде, поэтому выполним преобразование через тернарный оператор n==0?(String)"Было":(String) "Будет" . Он работает так: если n=0, то значение будет строка «Было», а иначе «Будет». После этого цикл завершается.

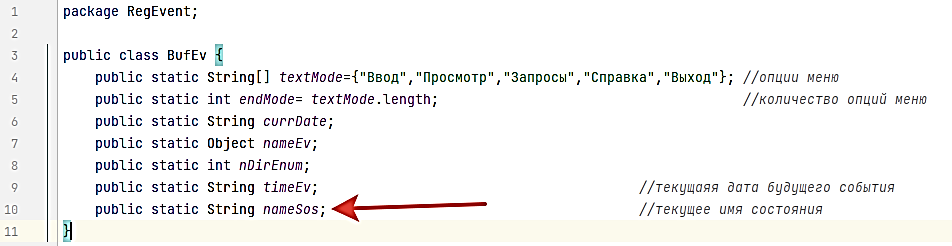
else System.*out*.print("Номер состояния число 0 или 1. Повторите ввод - ");

Неправильный ввод, если не 0 или 1

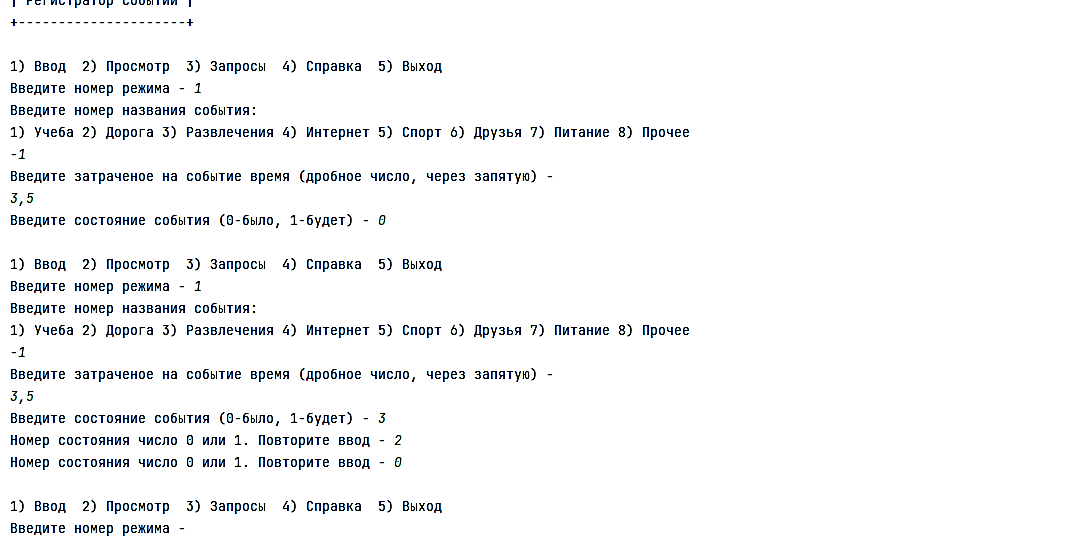
else System.*out*.println("Вы ввели не число");

Неправильный ввод при проверке if (cin.hasNextInt()) {

И для полноты требуется пополнить класс BufEv



Тестовый прогон приложения:



Конец лекции