Назначение лекции

Вы познакомитесь с особенностями ввода данных разного типа с контролем.

Новые слова

**Scanner** – готовый класс для организации ввода данных с клавиатуры, файла и пр.

**hasNextInt** – функция контроля данных целого типа

**hasNextDouble -** функция контроля данных вещественного типа

# Ввод данных с клавиатуры целых чисел

Для написания полноценных программ требуется чтобы программа взаимодействовала с человеком (пользователем программы), т.е. воспринимала от него какую-либо информацию.

В этой лекции мы рассмотрим самый простой ввод данных: с помощью клавиатуры. С помощью клавиатуры человек может вводить цифры (которые формируют числа) и символы, которые формируют строки.

В Java реализация механизма ввода отличается некоторым своеобразием (даже в сравнении с С++). Это своеобразие было продиктовано более строгими требованиями к вводу.

Программы для Java основаны на классах и объектах. Для организации ввода с клавиатуры, также потребуется использовать объект *класса* Scanner.

Принцип работы такой: создается объект класса Scanner, после чего можно использовать этот объект для ввода данных.

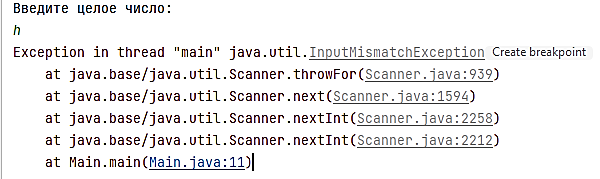
**Scanner cin = new Scanner(System.in);**

где, cin – имя объекта (можно любое другое, например, scn), Scanner – имя класса, System.in – потоковый ввод данных с клавиатуры.

Далее выполняется ввод, например, целых чисел в перемнную n:

**int n = cin.nextInt();**

Особенность ввода заключается в том, что Java сразу проверяет введенные данные и если с клавиатуры введен символ, а не целое число, то программа выдаст сообщение об ошибки и прервет выполнение.



С одной стороны Java строго относится к вводу данных, но сдругой стороны для пользователя такое сообщение малоинформативно.

Для решения такой проблемы java предлагает следующий механизм: ввод данных фактически состоит из двух частей. Первая – собственно прием данных с клавиатуры и вторая – передача данных в программу. Это позволяет выполнить контроль правильности данных еще до их передачи программе.

Конечно, можно ограничится лишь второй частью, но Java не случайно считается надежной системой.

Если вы хотите связать свою профессиональную деятельность с программированием именно на Java, вы должны знать, что большую часть программирования в ней занимает именно обработка ошибок. Например, чтобы написать на Java самый простой код, по открытию файла и считывания информации из него, вам понадобится написать десятка два строк кода, в Python – только одну. Java надежна, безопасная и именно поэтому код многословен.

В нашем случае можно заранее проконтролировать введенные данные с помощью функции **hasNextInt(),** которая принимает значение true если введено целое число:

**Scanner cin = new Scanner(System.*in*);  
System.*out*.println("Введите целое число: ");  
if (cin.hasNextInt())  
{  
 int n = cin.nextInt();  
 System.*out*.println("Вы ввели число: " + n);  
}  
else System.*out*.println("Вы ввели не число");**

Можно немного усовершенствовать программу таким образом, чтобы ввод продолжался до тех пор, пока не будет введено число:

**import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 int n=0;  
 while (true)  
 {  
 Scanner cin = new Scanner(System.*in*);  
 System.*out*.println("Введите целое число: ");  
 if (cin.hasNextInt())  
 {  
 n = cin.nextInt();  
 System.*out*.println("Вы ввели число: " + n);  
 break;  
 }  
 else  
 {  
 System.*out*.println("Вы ввели не число");  
 continue;  
 }  
 }  
 }  
}**

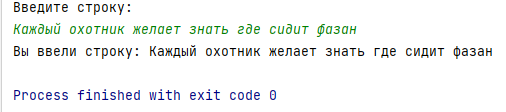
# Ввод данных с клавиатуры данных другого типа

Рассмотрим ввод строк, дробных чисел.

nextLine – ввод строки (можно с пробелами). Например,

**import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 String s;  
 Scanner cin = new Scanner(System.*in*);  
 System.*out*.println("Введите строку: ");  
s = cin.nextLine();  
 System.*out*.println("Вы ввели строку: " + s);  
}  
}**

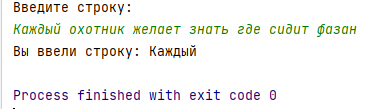
Результат:



**next** вместо **nextLine** – ввод строки до первого пробела. Например,

**import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 String s;  
 Scanner cin = new Scanner(System.*in*);  
 System.*out*.println("Введите строку: ");  
s = cin.next();  
 System.*out*.println("Вы ввели строку: " + s);}  
}**

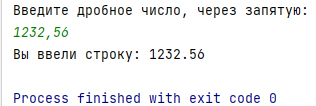
Результат:

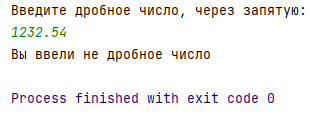


nextDouble – ввод дробного числа. Разделитель – запятая. Например,

**import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 Double d;  
 Scanner cin = new Scanner(System.*in*);  
 System.*out*.println("Введите дробное число, через запятую: ");  
 if (cin.hasNextDouble())  
 { d = cin.nextDouble();  
 System.*out*.println("Вы ввели число: " + d);  
 }  
 else System.*out*.println("Вы ввели не дробное число");  
 }  
}**

Результаты:





# Ввод данных с клавиатуры чисел с контролем и повтором ввода

Разберем вариант повторного ввода после ошибочного значения:

**import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
  
 int n;  
 System.*out*.println("Введите целое положительное число: ");  
 while (true) {  
 Scanner ek = new Scanner(System.*in*);  
 if (ek.hasNextInt()) {  
 n= ek.nextInt();  
 System.*out*.println("Вы ввели число "+n);  
 break;  
 }  
 else System.*out*.println("Вы ввели не число");  
 }  
 }  
}**

Конец лекции