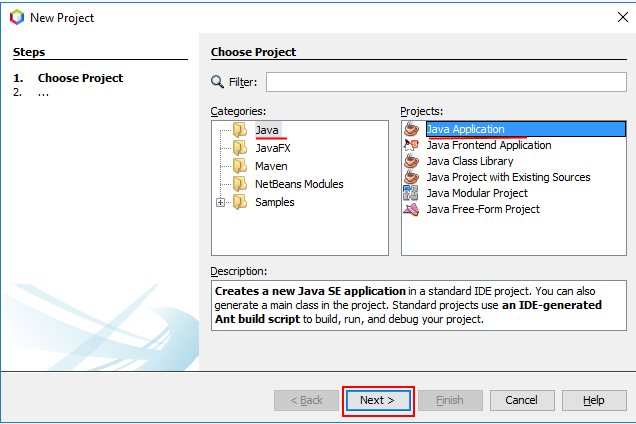
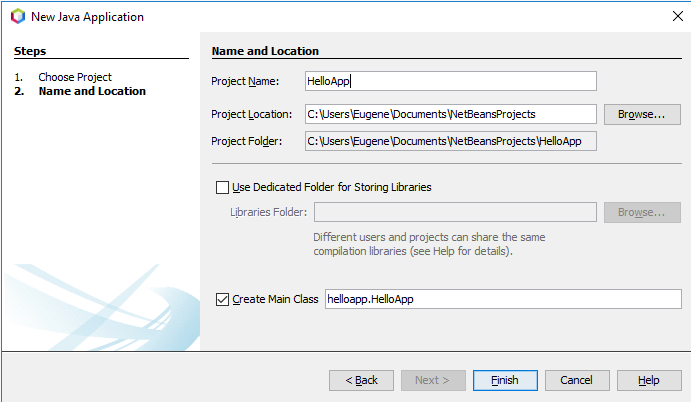
## Основы Java

1. Создание программы в Netbeans

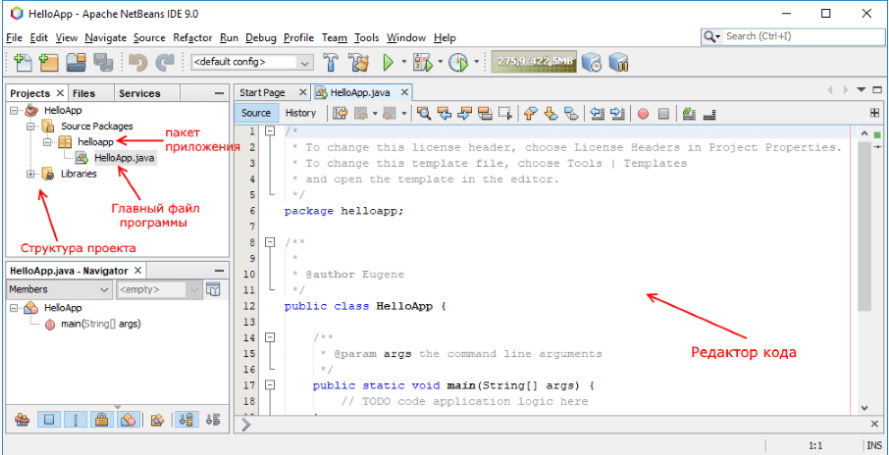


В окне создания нового проекта выберем в левой колонке первый пункт - Java, а в правой в качестве типа проекта - **Java Application**. И нажмем кнопку **Next >**

Затем откроется окно настроек проекта:



В поле Project Name дадим проекту какое-нибудь название (в нашем случае это HelloApp). Для всех остальных полей можно оставить значения по умолчанию. Последнее поле "Create Main Class" указывает, что автоматически в проекте будет создан класс программы HelloApp, который будет находиться в одноименном пакете helloapp. И в завершении создания проекта нажмем на кнопку "Finish". И перед нами откроется новый проект в Netbeans:

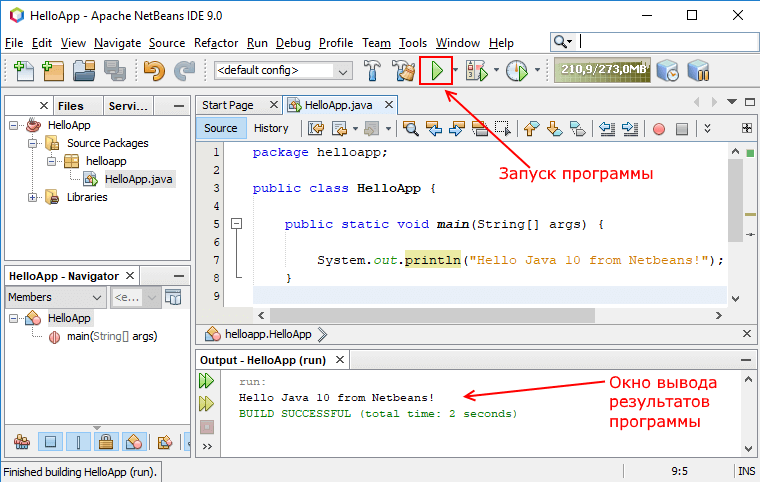


Узкое окно слева отображает все открытые проекты и их структуру. В данном случае у нас открыт пока только один проект. А большое окно справа представляет редактор кода с некоторыми дополнительными функциями, как подцветка кода, интеллектуальная подсказка и т.д. Редактор кода уже имеет некоторое содержание по умолчанию:

Здесь, во-первых, много комментариев, которые нам не нужны и которые мы можем удалить. Во-вторых, в начале файла указывается, что класс будет находиться в пакете helloapp с помощью директивы package: package helloapp;. Затем, как уже рассматривалось в прошлой теме, у нас есть класс программы - класс HelloApp, который имеет метод main.

Изменим код по умолчанию на следующий, чтобы программа выводила простейшую строку на консоль:





1. **Консольный ввод/вывод в Java**

**Вывод на консоль**

Для создания потока вывода в класс System определен объект **out**. В этом объекте определен метод println, который позволяет вывести на консоль некоторое значение с последующим переводом консоли на следующую строку:

|  |  |
| --- | --- |
| 1 | System.out.println("Hello world"); |

В метод println передается любое значение, как правило, строка, которое надо вывести на консоль. При необходимости можно и не переводить курсор на следующую строку. В этом случае можно использовать метод System.out.print(), который аналогичен println за тем исключением, что не осуществляет перевода на следующую строку.

|  |  |
| --- | --- |
|  | System.out.print("Hello world"); |

Но с помощью метода System.out.print также можно осуществить перевод каретки на следующую строку. Для этого надо использовать escape-последовательность *\n*:

|  |  |
| --- | --- |
|  | System.out.print("Hello world \n"); |

Если у нас есть два числа, и мы хотим вывести их значения на экран, то мы можем, например, написать так:

|  |  |
| --- | --- |
|  | int x=5;  int y=6;  System.out.println("x="+x +"; y="+y); |

Но в Java есть также функция для форматированного вывода, унаследованная от языка С: System.out.printf(). С ее помощью мы можем переписать предыдущий пример следующим образом:

|  |  |
| --- | --- |
|  | int x=5;  int y=6;  System.out.printf("x=%d; y=%d \n", x, y); |

В данном случае символы **%d** обозначают спецификатор, вместо которого подставляет один из аргументов. Спецификаторов и соответствующих им аргументов может быть множество. В данном случае у нас только два аргумента, поэтому вместо первого %d подставляет значение переменной x, а вместо второго - значение переменной y. Сама буква **d** означает, что данный спецификатор будет использоваться для вывода целочисленных значений типа int.

Кроме спецификатора *%d* мы можем использовать еще ряд спецификаторов для других типов данных:

* **%x**: для вывода шестнадцатеричных чисел
* **%f**: для вывода чисел с плавающей точкой
* **%e**: для вывода чисел в экспоненциальной форме, например, 1.3e+01
* **%c**: для вывода одиночного символа
* **%s**: для вывода строковых значений

Например:

|  |  |
| --- | --- |
|  | String name = "Иван";  int age = 30;  float height = 1.7f;    System.out.printf("Имя: %s   Возраст: %d лет   Рост: %.2f метров \n", name, age, height); |

При выводе чисел с плавающей точкой мы можем указать количество знаков после запятой, для этого используем спецификатор на %.2f, где .2 указывает, что после запятой будет два знака. В итоге мы получим следующий вывод:

Имя: Иван Возраст: 30 лет Рост: 1,70 метров

**Консольный ввод**

Для получения консольного ввода в классе System определен объект **in**. Однако непосредственно через объект System.in не очень удобно работать, поэтому, как правило, используют класс Scanner, который, в свою очередь использует System.in. Например, создадим маленькую программу, которая осуществляет ввод чисел:

|  |  |
| --- | --- |
|  | import java.util.Scanner;    public class FirstApp {        public static void main(String[] args) {            Scanner in = new Scanner(System.in);          int[] nums = new int[5];          for(int i=0;i < nums.length; i++){              nums[i]=in.nextInt();          }            for(int i=0;i < nums.length; i++){              System.out.print(nums[i]);          }          System.out.println();      }  } |

Так как класс Scanner находится в пакете *java.util*, то мы вначале его импортируем. Для создания самого объекта Scanner в его конструктор передается объект System.in. После этого мы можем получать вводимые значения. Например, чтобы получить введенное число, используется метод in.nextInt();, который возвращает введенное с клавиатуры целочисленное значение.

В данном случае в цикле вводятся все элементы массива, а с помощью другого цикла все ранее введенные элементы массива выводятся в строчку.

Класс Scanner имеет еще ряд методов, которые позволяют получить введенные пользователем значения:

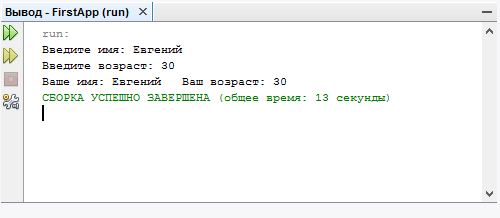
* **next()**: считывает введенную строку до первого пробела
* **nextLine()**: считывает всю введенную строку
* **nextInt()**: считывает введенное число int
* **nextDouble()**: считывает введенное число double
* **hasNext()**: проверяет, было ли введено слово
* **hasNextInt()**: проверяет, было ли введено число int
* **hasNextDouble()**: проверяет, было ли введено double

Кроме того, класс Scanner имеет еще ряд методов nextByte/nextShort/nextFloat/nextBoolean, которые по аналогии с nextInt считывают данные определенного типа данных.

Создадим следующую программу для ввода информации о человеке:

|  |  |
| --- | --- |
|  | import java.util.Scanner;    public class FirstApp {        public static void main(String[] args) {            Scanner in = new Scanner(System.in);          System.out.print("Введите имя: ");          String name = in.nextLine();          System.out.print("Введите возраст: ");          int age = in.nextInt();          System.out.println("Ваше имя: " + name + "   Ваш возраст: " + age);      }  } |

Например, если бы мы запускали проект в NetBeans, то это выглядело бы так:



**Упражнения**

1. Дана точка на плоскости с координатами (х, у). Составить программу, которая выдает одно из сообщений «Да», «Нет», «На границе» в зависимости от того, лежит ли точка внутри заштрихованной области, вне заштрихованной области или на ее границе.

|  |  |
| --- | --- |
| а. | б. |

1. Вывести на экран кубы всех целых чисел из диапазона от А до В (А< В) в обратном порядке.
2. Вывести на экран числа следующим образом

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 |  |  |  |  |
| 2 | 1 |  |  |  |
| 3 | 2 | 1 |  |  |
| 4 | 3 | 2 | 1 |  |
| 5 | 4 | 3 | 2 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| а. | 5 |  |  |  | б. |
|  | 5 | 5 |  |  |  |
|  | 5 | 5 | 5 |  |  |
|  | 5 | 5 | 5 | 5 |  |
|  | 5 | 5 | 5 | 5 | 5 |

1. Разработать метод min(a,b) для нахождения минимального из двух чисел. Вычислить с помощью него значение выражения z=min(3x,2y)+min(x-y,x+y).
2. Разработать метод f(x), который нечетное число заменяет на 0, а четное число уменьшает в два раза. Продемонстрировать работу данного метода на примере.