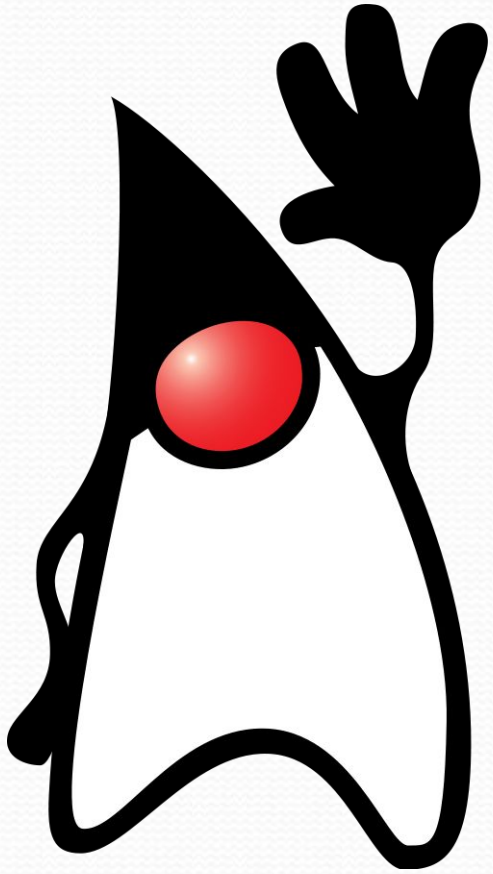
Курс: «Программирование на Java»



Понятие слова «Метод» в Java

- **Метод** — это именованный блок кода, который **объявляется внутри класса** и может быть использован многократно. Таким образом следует то, что –
- **Свойства метода**:

1. МЕТОД ПРИНАДЛЕЖИТ КЛАССУ И НЕ МОЖЕТ БЫТЬ ОБЪЯВЛЕН ВНЕ КЛАССА.
2. Хорошо написанный метод решает одну практическую задачу.
3. Новый метод сначала объявляют и определяют, затем вызывают для нужного объекта или класса.

Для чего нужны методы?

- В программировании методы представляют собой обычные участки кода, которые можно использовать(вызывать) в любом месте программы, класса, если метод в них определен.
- Рассмотрим метод вывода информации на экран – println(); – это метод, который содержит десятков строк кода, но для того, чтобы программисту не пришлось прописывать весь код заново, были придуманы методы.
- Таким образом в методах обычно содержится код или определенный алгоритм, который необходимо использовать в программе более одного раза.

Объявление метода в Java
Ключевые слова “public” и “static”

- Прежде чем начать работу с методам, его необходимо объявить используя ключевые слова public и static
- Ключевое слово – public – сообщает компьютеру о том, что метод является общедоступным и к нему все могут получить доступ.
- Ключевое слово – static – сообщает компьютеру о том, что метод является статическим и его можно использовать без создания экземпляра(объекта) класса.

Объявление метода в Java

- После того, как мы написали ключевые слова public, static и тип возвращаемого значения(например int), необходимо назвать наш метод затем поставить круглые скобки () и после них фигурные скобки {}.
  Имя метода может быть любым.
  Вот как выглядит объявление метода с именем method в Java, который возвращает **целое число**.
- ```
  public static int method(){
    return 3;
  }
  ```
  Прим. Данный метод возвращает целое число – 3.
  ```

Объявление метода в Java
Тип возвращаемого значения

- Любой метод в Java выполняет определенную задачу и по выполнении он обязать что-то сообщить(вернуть) компьютеру. Для определения, что может возвращать метод введено понятие – *тип возвращаемого значения метода.*
  Типов возвращаемых значений может быть множество, но существуют базовые типы:
  Тип – int – Целочисленный тип. Метод возвращает – **целое число**
  Тип – double – Дробный тип. Метод возвращает – **дробное число**
  Тип – String – Строковый тип. Метод возвращает – **строку**
  Тип – void – пустой тип. Метод возвращает – **ничего**

Объявление метода в Java
Ключевое слово “return”

- При объявлении метода использовалось ключевое слово return(вернуть) - данное слово отдает компьютеру значение того, типа, который мы указали после ключевого слова static
- Таким образом методы могут выполнять определенный алгоритм и возвращать определенное значение с помощью ключевого слова return
- МЕТОД НЕ ВОЗВРАЩАЕТ ЗНАЧЕНИЕ В ПУСТОТУ. ОН ОБЯЗАН ВЕРНУТЬ ЕЕ В ПЕРЕМЕННУЮ ИЛИ В ДРУГОЙ МЕТОД. Пример:

  ```
  int a = method();
  ```

  Исключением служит метод с возвращаемым типом void – для него не обязательна переменная.

Объявление метода в Java
Типы “double” и “int”

- Если тип – double – то необходимо вернуть дробное число или переменную дробного типа. Пример:
  return 3.5; или
  return var; (var – переменная дробного типа).
  Если тип – int – то необходимо вернуть целое число или переменную целого типа. Пример:
  return 3; или
  return var1; (var1 – переменная целого типа).

Объявление метода в Java
Типы “string” и “void”

- Если тип – string – то необходимо вернуть строку или переменную строкового типа. Пример:
  return “Строка возвращается в кавычках”;
  или
  return str; (str – переменная строкого типа).
  Если тип – void – то писать ключевое слово return ЗАПРЕЩЕНО, т.е. метод с возвращаемым значением – void – ничего не возвращает и не содержит ключевого слова ”return”

Объявление метода в Java
Примеры.

```
public static double method(){
  return 3.5;          //Возвращаем дробное число(double)
}
                        public static int method(){
//Возвращаем целое число (int)          return 3;
                        }
public static String method(){
  return “Любая строка”;    //Возвращаем строку(String)
}
                        public static void method(){
//Ничего не возвращаем(void)
                        }
```

Принимаемые переменные метода - Параметры.

- При работе с методами им можно передавать различные значения переменных определенного типа(параметры), которые впоследствии можно обработать.
  Передаваемые параметры необходимо объявить в круглых скобках через запятую, указав их тип.
  Пример:

```
public static int method(int a, int b) {
    return 3;
}
```

- //Объявляем метод который обязан принять 2 переменные и вернуть целое число

Работа с методами

Все, что написано внутри фигурных скобок является телом метода и содержит в себе определенную программистом последовательность действий. Напишем метод sum(), который принимает и складывает два числа, а после возвращает их сумму:

```
public static int sum(int a, int b) {   //Принимаем значения a и b
      int summa = a+b;   //Объявим пер.  “summa” и положим a+b
     return summa;        //Возвращаем переменную “summa”
}
```

Или можно сократить данный метод

```
public static int sum(int a, int b) {   //Принимаем значения a и       return a+b;    //Складываем a+b  и возвращаем
}
```

Прим. При возвращении значения поддерживается калькулятор

Работа с методами: Вызов метода.

- Для того чтобы вызвать метод, необходимо написать его имя, поставить круглые скобки и, если требуется, передать в них значения.

Пример:

```
 public static int sum(int a, int b){ //Создаем метод
        return a+b;
}
public static void main(String[] args){
  int summa = sum(5,4); //Вызываем метод для переменной
  System.out.println(sum(5,4));    //Вызываем метод внутри
                                   // другого метода
}
```

Работа с методами
ВАЖНО!!!

- На самом деле - main - тоже является методом. В программировании методы ничего не знаю друг о друге, не знают, какие в них переменные и работают только со своими переменными. Т.е. если мы в main'е объявим переменную, для всех остальных методов данной переменной не будет существовать и в них можно будет создать переменную с точно таким же именем.
- Поэтому когда мы передаем переменные из main'а в другой метод, на самом деле мы передаем лишь значения данных переменных. Сами переменные при работе с ними внутри main'а своих значений не меняют!

Работа с методами
ВАЖНО!!! ПРИМЕРЫ.

- Пусть есть переменная “q” и переменная “z” передадим ее в функцию и посмотрим, что происходит.

```
public static int method(int q){//Создается новая переменная
q=7;//с именем q, которой присваивается значение старой q
System.out.println(q); //Выведется число 7
return q;
}
int q =5;
int z = 6;
z = method(q);
System.out.println(q);//Выведется число 5, т.к. q – старая перем.
System.out.println(z);//Выведется число 7, т.к. z изменилось
```

Работа с методами
ВАЖНО!!! ПРИМЕРЫ.

- Таким образом имена переменных-параметров функции никак не связано с именами переменных в main, значения которых передаются в функцию.

```
public static int method(int a){     //передаем значение, а не
  a=a+5;                             //саму переменную
  return a;
}
int b = 10;
int c = method(b); // можно передавать несовпадающие имена
System.out.println(b); //Выведется число 10
System.out.println(c); //Выведется число 15, т.к. 10+5
```

Пример программы.

```
public class Prog{
  public static int sum(int a, int b){ //Создаем метод
        return a+b;
}
  public static void main(String[] args){
  int p =4;
  int q =5;
  int summa = sum(p,q);//Вызываем и возвращаем значение
  summa = sum(p,5);   //Можно передавать обычные числа
  System.out.println(summa);
  System.out.println(sum(p,q));//Можно вызывать где угодно
}
```

Работа с методами
Тип Void

- Возвращаемый тип void необходим в тех случаях, когда не требуется возвращать куда-либо результат вычислений в функции. Такими случаями могут являться методы, от которых требуется ТОЛЬКО вывод на экран, или методы, которые работают с переменными класса.

Работа с методами
Тип Void. Пример.

- С помощью метода print() вывести значение целочисленной переменной, которая передается в данный метод.

```
public static void print(int a){
  System.out.println(a);
   //ключевое слово return отсутствует
}
public static void main(String[] args){
  int b = 8;
  print(b); //Выведется число 8
}
```