Java Basic

lecture #4. Method in Java

Mentor: Il'yas Miftakhov

lecture #4. Method in Java

- Основные понятия
- Объявление метода – 6 компонентов
- Типы методов в Java
- Как назвать метод?
- аргументы метода

Основные понятия

- Метод в Java или Java метод, представляет собой набор операторов, которые выполняют определенную задачу и возвращают результат вызывающей стороне.
- Метод Java может выполнять определенную задачу, ничего не возвращая.
- Методы в Java позволяют нам повторно использовать код без повторного ввода кода.
- В Java каждый метод должен быть частью некоторого класса

Объявление метода – 6 компонентов

1. Модификатор: определяет тип доступа к методу, т. е. откуда к нему можно получить доступ в вашем приложении. В Java существует 4 типа спецификаторов доступа.

public: доступен во всех классах вашего приложения

protected: доступен в классе, в котором он определен, и в его подклассах

private: доступен только внутри класса, в котором он определен.

default: объявляется без использования какого-либо модификатора. Доступен в том же классе и пакете, в котором определен его класс.

2. Тип возвращаемого значения: тип данных значения, возвращаемого методом, или void, если значение не возвращается.

3. Имя метода: правила для имен полей применяются и к именам методов.

4. Список параметров: определяется список входных параметров, разделенных запятыми, с предшествующим типом данных в заключенных скобках. Если параметры отсутствуют, необходимо использовать пустые скобки ().

5. Список исключений: Исключения, которые вы ожидаете от метода, вы можете указать эти исключения.

6. Тело метода: заключено в фигурные скобки. Код, который необходимо выполнить для выполнения намеченных операций.

Типы методов в Java

В Java есть два типа методов:

1. **Предопределенный метод**: это метод, который уже определен в библиотеках классов Java, известный как предопределенные методы. Также известен как метод стандартной библиотеки или встроенный метод. Мы можем напрямую использовать эти методы, просто вызывая их в программе в любой момент.

2. **Пользовательский метод:** Метод, написанный программистом, известен как пользовательский метод.

Как назвать метод?

Имя метода обычно представляет собой одно слово, которое должно быть **глаголом** в нижнем регистре или состоять из нескольких слов, которое начинается с глагола в нижнем регистре, за которым следует прилагательное, существительное.

После первого слова первая буква каждого слова должна быть заглавной. (camelCase)

Правила:

- имя метода должно быть глаголом и начинаться со строчной буквы.
- если состоит из более чем двух слов, первое слово должно быть глаголом, за которым следует прилагательное или существительное.
- первая буква каждого слова должна быть прописной, кроме первого слова.
- метод имеет уникальное имя в пределах класса, в котором он определен, но иногда метод может иметь то же имя, что и другие имена методов в том же классе, поскольку в Java разрешена перегрузка методов.

Вызов метода

Метод должен быть вызван для использования его функциональности.

Метод всегда возвращается к коду, вызвавшему его!

При вызове метода могут быть три ситуации:

1. Он завершает все операторы в методе
2. Он достигает оператора возврата
3. Выдает исключение

SOLID

S
Принцип единственной ответственности (single responsibility principle)
Для каждого класса должно быть определено единственное назначение. Все ресурсы, необходимые для его осуществления, должны быть инкапсулированы в этот класс и подчинены только этой задаче.
O
Принцип открытости/закрытости (open-closed principle)
«программные сущности … должны быть открыты для расширения, но закрыты для модификации».
L
Принцип подстановки Лисков (Liskov substitution principle)
«функции, которые используют базовый тип, должны иметь возможность использовать подтипы базового типа не зная об этом». См. также контрактное программирование.
I
Принцип разделения интерфейса (interface segregation principle)
«много интерфейсов, специально предназначенных для клиентов, лучше, чем один интерфейс общего назначения»
D
Принцип инверсии зависимостей (dependency inversion principle)
«Зависимость на Абстракциях. Нет зависимости на что-то конкретное»