**Видимость переменных**

[Java Syntax](https://javarush.ru/quests/QUEST_JAVA_SYNTAX)

[Уровень 2](https://javarush.ru/quests/lectures?quest=QUEST_JAVA_SYNTAX&level=2), Лекция 4

— Привет, мой любимый ученик. Сейчас я расскажу тебе о видимости переменных.

— А что, бывают и невидимые переменные?

— Нет, под «видимостью переменных» подразумевают места в коде, где к этой переменной можно обратиться. К некоторым переменным можно обращаться отовсюду в программе, к другим только в пределах их класса, к третьим же только внутри одного метода.

— Например, ты не можешь обратиться к переменной раньше, чем она объявлена.

— Логично.

— Вот несколько примеров:

┏

┃public class Variables

┃┏

┃┃{

┃┃ private static String TEXT = "The end.";

┃┃ ┗━━━━━━━━━━━━━━┛

┃┃ public static void main (String[] args)

┃┃ ┏ ┗━━━━━━━┛

┃┃ ┃ {

┃┃ ┃ System.out.println("Hi");

┃┃ ┃ String s = "Hi!";

┃┃ ┃ ┏┗━━━━┛

┃┃ ┃ ┃ System.out.println(s);

┃┃ ┃ ┃ if (args != NULL)

┃┃ ┃ ┃ {

┃┃ ┃ ┃ String s2 = s;

┃┃ ┃ ┃ ┗━━━━┛

┃┃ ┃ ┃ ┏

┃┃ ┃ ┃ ┃ System.out.println(s2);

┃┃ ┃ ┃ ┗

┃┃ ┃ ┃ }

┃┃ ┃ ┃ Variables variables = new Variables();

┃┃ ┃ ┃ System.out.println(variables.classVariables);

┃┃ ┃ ┃ System.out.println(TEXT);

┃┃ ┃ ┗

┃┃ ┃ }

┃┃ ┗

┃┃ public String classVariables;

┃┃ ┗━━━━━━━━━━━━━━━┛

┃┃ public Variables()

┃┃ {

┃┃ classVariables = "Class Variables test.";

┃┃ }

┃┃}

┃┗

┗

1. Переменная, объявленная в методе, существует/видна с начала объявления до конца метода.

2. Переменная, объявленная в блоке кода, существует до конца этого блока кода.

3. Переменные — аргументы метода — существуют везде внутри метода.

4. Переменные класса/объекта существуют все время жизни содержащего их объекта. Их видимость дополнительно регулируется специальными модификаторами доступа: public, private.

5. Статические переменные классов существуют все время работы программы. Их видимость также определяется модификаторами доступа.

— Обожаю картинки, на них все всегда понятно.

— Молодец, Амиго. Всегда знал, что ты сообразительный малый.

— Еще расскажу тебе про «модификаторы доступа». Да не пугайся ты так, ничего сложного в них нет. Это слова **public** и **private**, которые ты видишь.

— А я и не пугаюсь – это у меня глаз дергается.

— Верю, верю. Ты можешь управлять доступом (видимостью) методов и переменных твоего класса из других классов. Каждому методу или переменной можно указывать только один модификатор доступа.

1 Модификатор «**public**».

К переменной, методу или классу, помеченному модификатором public, можно обращаться из любого места программы. Это самая высокая степень открытости — никаких ограничений нет.

2 Модификатор «**private**».

К переменной или методу, помеченному модификатором private, можно обращаться только из того же класса, где он объявлен. Для всех остальных классов помеченный метод или переменная — невидимы и «как бы не существуют». Это самая высокая степень закрытости – только свой класс.

3 Без модификатора.

Если переменная или метод не помечены никаким модификатором, то считается, что они помечены «модификатором по умолчанию». Переменные или методы с таким модификатором (т.е. вообще без какого-нибудь) видны всем классам пакета, в котором они объявлены. И только им. Этот модификатор еще иногда называют «**package**», намекая, что доступ к переменным и методам открыт для всего пакета, в котором находится их класс

Таблица с пояснением:

|  |  |  |  |
| --- | --- | --- | --- |
| **Модификаторы** | **Доступ из…** | | |
| **Своего класса** | **Своего пакета** | **Любого класса** |
| **private** | Есть | Нет | Нет |
| нет модификатора (**package**) | Есть | Есть | Нет |
| **public** | Есть | Есть | Есть |