**Как создавать объекты**

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

[Уровень 5](https://javarush.ru/quests/lectures?quest=QUEST_JAVA_SYNTAX&level=5), Лекция 4

— Так. С классами мы разобрались в прошлый раз. Сегодня я хочу рассказать тебе, как создавать объекты. Это очень просто: пишем ключевое слово new и имя класса, объект которого хотим создать:

|  |
| --- |
| **Пример** |
|  |
| Cat cat = new Cat(); | |
|  | |

|  |
| --- |
| Reader reader = new BufferedReader(new InputStreamReader(System.in)); |
|  |

|  |
| --- |
| InputStream is = new FileInputStream(path); |

— Так я это уже слышал.

— Я знаю, поэтому слушай дальше.

— При создании объекта в скобочках можно передавать различные параметры. Об этом я расскажу сегодня, но чуть попозже. Давай рассмотрим класс Cat:

|  |  |  |
| --- | --- | --- |
| **Код на Java** | **Описание** | |
|  |  | |
|  | |
| class Cat {  public String **name**;  public String getName() {  return **name**;  }  public void setName(String name) {  this.**name** = name;  }  } | | name – это переменная класса. У нее модификатор доступа public, поэтому она видна везде в проекте. | |
| Метод getName – это getter (произносится гЭттэр), т.е. он возвращает значение переменной-класса name. Имя метода строится по принципу get + «имя переменной с большой буквы». | |
| Метод setName – это setter (произносится сЭттэр), т.е. он используется для присваивания нового значения переменной-класса name. Имя метода строится по принципу set + «имя переменной с большой буквы». В этом методе имя параметра совпадает с именем переменной класса, поэтому мы ставим **this** перед именем переменной. | |

— А это что еще за геттеры и сеттеры такие?

— В Java принято скрывать переменные от доступа из других классов. Обычно переменные, объявленные внутри классов, имеют модификатор private.

— Чтобы другие классы могли менять значения таких переменных, для каждой из них создается пара методов: **get** и **set**. Задача метода **get** вернуть текущее значение переменной тому, кто его вызвал. Задача метода **set** установить новое значение переменной.

— А в чем смысл?

— Если мы не хотим, чтобы кто-то менял значения переменных наших объектов, мы можем просто не писать метод **set** для него, или сделать его **private**. Также в этот метод можно добавить дополнительные проверки данных. И если переданное новое значение неверно, то ничего не менять.

— Ясно.

— Т.к. переменных в классе может быть много, то методы get и set обычно имеют в своем имени имя той переменной, с которой работают.

— Если переменная называется name, то методы **setName** и **getName**. И т.д. по аналогии.

— Ясно. Понятный, в общем-то, подход.

— Вот тебе еще пара примеров работы с новосозданным объектом:

|  |  |  |  |
| --- | --- | --- | --- |
| **Шаг** | **Код** | **Описание** | |
| 1 |  |  | |
| new Cat(); | | | Создаём объект типа Cat | |
| 2 | | |  | |

|  |  |
| --- | --- |
| Cat catVaska = new Cat(); | Сохраняем ссылку на объект в переменную catVaska, имеющую тип Cat |
| 3 |  |

|  |  |
| --- | --- |
| catVaska.name = "Vaska";  catVaska.age = 6;  catVaska.weight = 4; | Заполняем объект данными: имя, возраст, вес |
| 4 |  |

|  |  |
| --- | --- |
| catVaska.sleep(); | Вызываем метод объекта |
| 5 |  |

|  |  |
| --- | --- |
| catVaska.fight(catBarsik); | Взаимодействие объектов. |