Взаимодействие объектов

Java Syntax
2 уровень, 1 лекция

ОТКРЫТА

— Привет, Амиго. Сегодня я хочу рассказать тебе, как устроена типичная программа на Java. Главная новость: Каждая программа на Java состоит из классов и объектов.

— Что такое классы, я уже знаю. А что такое объекты?

— Начну с аналогии. Представь, что ты хочешь сделать небольшой корабль. Сначала нужно сделать чертёж, затем отдать его на завод, где по этому чертежу соберут корабль. Или десяток. Да, вообще, сколько угодно кораблей. По одному чертежу строятся десятки идентичных кораблей, вот что важно.

— В программировании на Java все точно так же.

— Программист – он как проектировщик. Только проектировщик рисует чертежи, а Java-программист пишет классы. Затем на основе чертежей создаются детали, а на основе классов – объекты.

— Сначала мы пишем классы (делаем чертежи), а потом, во время исполнения программы, на основе этих классов Java-машина создает объекты. Точно так же, как корабли создаются на основе чертежей. Чертёж один, но кораблей много. Корабли разные, у них разные имена, они возят разные грузы. Но они очень похожие: они все — корабли с идентичной конструкцией, и могут выполнять аналогичные задачи.

— На примере кораблей все понятно. А можно еще пару аналогий, чтобы я точно понял, о чем речь?

— Вот, например, пчелы…

— Хотя нет, что-то с пчелами у меня плохие ассоциации. Возьмем лучше муравейник.

— Муравейник – это хороший пример взаимодействия объектов. В простейшем муравейнике есть три класса муравьёв: королева, воины и рабочие муравьи. Количество муравьёв каждого класса – разное. Королева – одна на весь муравейник, воинов – десятки, а рабочих муравьёв – сотни. Три класса и сотни объектов. Муравьи взаимодействуют друг с другом, с такими же муравьями и муравьями других классов по жёстко заданным правилам.

— Это просто идеальный пример. В типичной программе все точно так же. Есть главный объект, который создаёт объекты всех остальных классов. Объекты начинают взаимодействовать друг с другом и «внешним миром» программы. Внутри этих объектов жёстко запрограммировано их поведение.

— Не совсем понятно. Вернее, совсем не понятно.

— Два этих пояснения – это две стороны одной медали. Истина посередине. Первый пример (про чертеж и корабли) показывает связь между классом и объектами этого класса. Аналогия очень сильная. Второй пример (про муравейник) показывает связь между объектами, которые существуют во время работы программы, и написанными классами.

— Ты хочешь сказать, что сначала мы должны написать классы для всех существующих в программе объектов, а потом ещё и описать их взаимодействие?

— Да, но это легче чем кажется. В Java все сущности во время работы программы являются объектами, а написание программы сводится к описанию различных способов взаимодействия объектов. Объекты просто вызывают методы друг друга и передают в них нужные данные.

— Не совсем очевидно, но почти понятно.

— А как узнать, какие методы вызывать, и какие данные туда передавать?

— У каждого класса есть описание, в котором говорится – для чего он создан. Также и у каждого его метода есть описание: что он делает, и какие данные нужно в него передавать. Чтобы использовать класс, нужно в общих чертах знать, что он делает. А также нужно точно знать, что делает каждый его метод. И совсем не обязательно знать, как он это делает. Такая себе волшебная палочка.

— Хм. Звучит заманчиво.

— Вот посмотри на код класса, который копирует файл:

Копирование файла c:data.txt в файл c:result.txt

```
1  package com.javarush.lesson2;
2  import java.io.FileInputStream;
3  import java.io.FileOutputStream;
4  import java.io.IOException;
5
6  public class FileCopy
7  {
8      public static void main(String[] args) throws IOException
9      {
10         FileInputStream  fileInputStream = new FileInputStream("c:\dat
11         FileOutputStream  fileOutputStream = new FileOutputStream("c:\
12
13         while (fileInputStream.available() > 0)
14         {
15             int data = fileInputStream.read();
16             fileOutputStream.write(data);
17         }
18
19         fileInputStream.close();
20         fileOutputStream.close();
21     }
22 }
```

— Не то, чтобы все понятно, но суть уже улавливаю.

— Отлично. Тогда – до следующего урока.

— Чуть не забыла. Вот тебе задачка от Диего.

Задача Java Syntax, 2 уровень, 1 лекция

РЕШЕНА

★★★★½

Реализуем метод print

Собственные методы — путь к неограниченной свободе, но также большая ответственность. Реализуем методы аккуратно и продумано. В этой задаче нам предстоит написать метод print, который будет выводить на экран некую строку. И не один раз, а сразу четыре. Строка — аргумент метода, то есть будет подаваться на входе.

Открыть

‹ Предыдущая Следующая ›