3.5

Оператор switch… case

00:00—01:18

Привет! Вы уже знакомы с условными операторами if, else и их комбинацией elseif, но есть ещё один условный оператор, который называется switch case. Его тоже важно знать, чтобы, прежде всего, понимать уже написанный код и оптимизировать свой код, в котором используется множество условий.
В переводе с английского switch означает «переключать», а case означает, собственно, кейс, вариант или случай. Этот оператор используется для переключения между различными вариантами. Вариантами чего именно мы и рассмотрим в этом видеоматериале.

Этот оператор необходим для того, чтобы заменить множество операторов elseif и сделать код более понятным. Возможно, это не кажется делом первой необходимости, однако лучше сразу приучать себя к хорошему — умение писать чистый и понятный код является одним из ключевых навыков при приёме на работу программистом, особенно если вы идёте работать в команду. Поэтому владение оператором switch case будет ещё одним вашим преимуществом.

Давайте сразу перейдём к среде разработки и посмотрим, как его использовать.

01:18—05:31

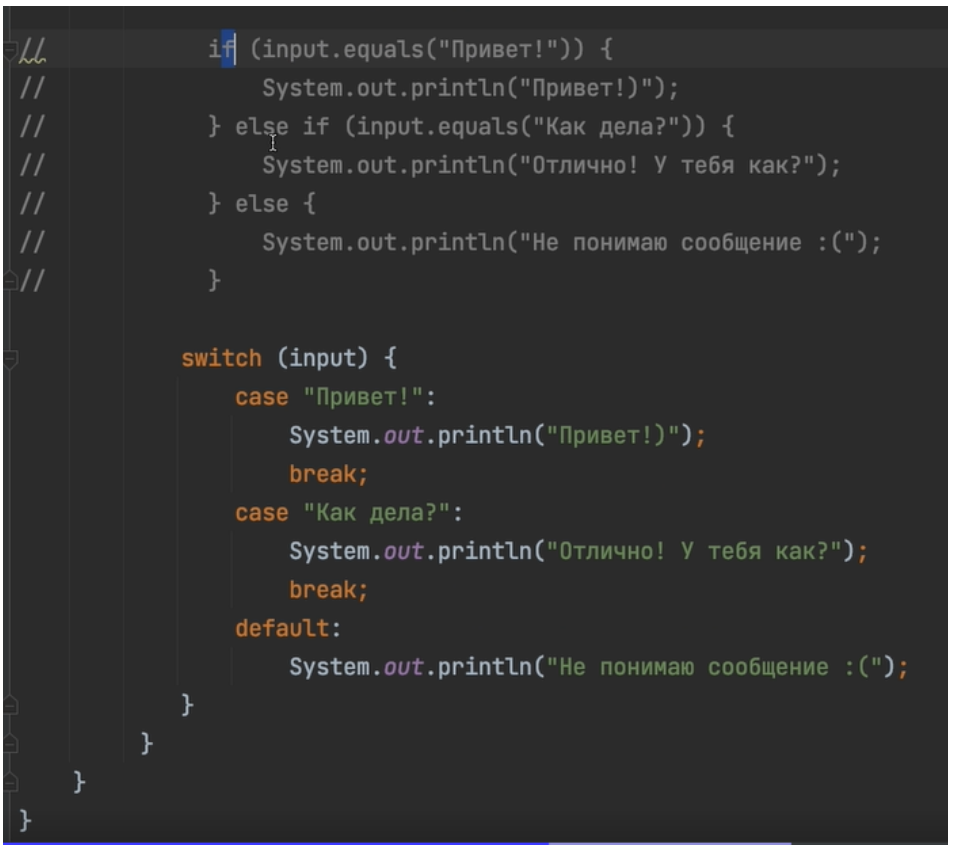
Для этого сначала напишем бесконечный цикл, который будет каждый раз просить ввести новое сообщение.
Введём переменную, назовём её input и напишем код, который будет реагировать на то или иное сообщение пользователя.

- If Input=Привет — в этом случае пользователю ответим: «Привет!»

- Если Input=Как дела? — ответ будет: «Отлично! У тебя как?»
- Если Input будет каким-то ещё, то программа честно признается, что она не поняла нашего сообщения.

```java
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        while (true) {
            System.out.println("Введите сообщение:");
            String input = new Scanner(System.in).nextLine();

            if (input.equals("Привет!")) {
                System.out.println("Привет!)");
            } else if (input.equals("Как дела?")) {
                System.out.println("Отлично! У тебя как?");
            } else {
                System.out.println("Не понимаю сообщение :(");
            }
        }
    }
}
```

Если сообщений, на которые программа должна реагировать корректно, станет слишком много, то код будет выглядеть не очень аккуратно из-за очень длинной цепочки операторов.
В таких случаях принято писать код с использованием оператора switch case, с которым он станет более приятным, читабельным и содержащим меньшее количество фигурных скобок.
Давайте уберём предыдущий код в комментарии и ниже напишем код с использованием switch case.
Здесь мы пишем значение, которое мы, соответственно, проверяем, далее ставятся фигурные скобки и для каждого кейса, то есть для каждого варианта этого значения, со сдвигом вправо пишем код, который должен выполняться в этом случае, затем обязательно пишем оператор break, иначе будут проверяться все кейсы.
Получаем такой код:



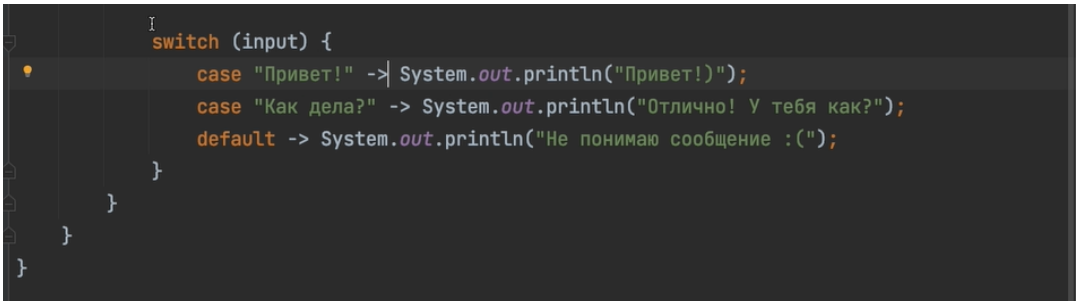
```
//            if (input.equals("Привет!")) {
//                System.out.println("Привет!)");
//            } else if (input.equals("Как дела?")) {
//                System.out.println("Отлично! У тебя как?");
//            } else {
//                System.out.println("Не понимаю сообщение :(");
//            }

            switch (input) {
                case "Привет!":
                    System.out.println("Привет!)");
                    break;
                case "Как дела?":
                    System.out.println("Отлично! У тебя как?");
                    break;
                default:
                    System.out.println("Не понимаю сообщение :(");
            }
        }
    }
}
```

Эти два кода абсолютно идентичны друг другу, но второй вариант выглядит несколько лучше и понятнее. Но, всё-таки, такой код длиннее, чем цепочка операторов if-else, и он кажется немного громоздким.

Мы стремимся писать код более профессионально и понятно, а, с этой точки зрения, наш код надо бы ещё немного улучшить.

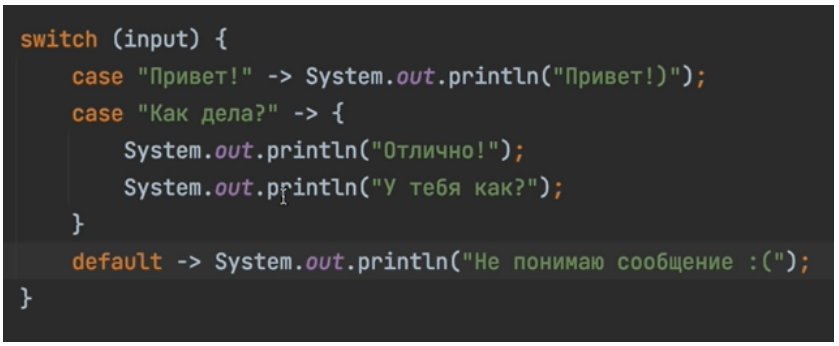
Этот пример — пример старого варианта использования оператора switch case, и он не очень удобен. Во-первых, кода пока ещё много, а, во-вторых, надо каждый раз писать break.

К счастью, языки программирования эволюционирует, чтобы нам было легче и приятнее с ними работать. В языке Java 14-й версии стало возможным писать этот же код короче с помощью стрелки — лямбды выражения.



Говоря упрощённо, этот код считается таким образом: если переменная равна кейсу, то выполняется только одна определённая строка кода. Это всё тот же оператор switch case, но код стал значительно короче и проще: один кейс — одна строка.
Мы ещё изучим лямбда-выражения подробнее в одном из следующих модулей, а пока расскажу о ещё одном моменте.

05:31—06:10
Что делать, если вам нужно выполнить в одном кейсе не одну строку кода, а несколько?
Если вам нужно выполнить в одном кейсе не одну строку кода, а несколько, например, наша программа должна ответить пользователю и задать ему встречный вопрос, то достаточно вывести вопрос в отдельной строке и поставить фигурные скобки.



06:10—06:31

Итак, в этом видеоматериале вы узнали, как выглядит и для чего используется оператор switch case, познакомились с его старыми и новыми вариантами использования.
С этого момента помните, что человека встречают по одёжке, а разработчика — по опрятности его кода.