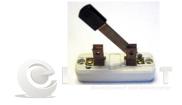


Switch statements



Switch statements (1)

Sometimes if statements become just too complex and hard to maintain.

```
if (expression == 1) {
    // Do this if 1
} else {
    if (expression == 2) {
        // Do this if 2
} else {
        if (expression == 3) {
            // Do this if 3
        } else {
            // Do this as default
}
```

Switch statements (2)

More idiomatic code might look like this. (Much easier to read)

```
if (expression == 1) {
    // Do this if 1
} else if (expression == 2) {
    // Do this if 2
} else if (expression == 3) {
    // Do this if 3
} else {
    // Do this as default
}
```



Switch statements (3)

Writing this sort of code using **if** statements quickly becomes difficult to maintain.

```
if (expression == 1) {
                                               switch (expression) {
                                                   case 1:
   // Do this if 1
                                                       // Do this in case 1
                                                       break;
} else if (expression == 2) {
  // Do this if 2
                                                       // Do this in case 2
                                                       break;
} else if (expression == 3) {
  // Do this if 3
                                                       // Do this in case 3
                                                       break;
} else {
  // Do this as default
                                                       // Do this if expression is
                                                       // neither 1, 2 or 3
}
                                                       break;
                                               }
```

Instead, we can replace it with a **switch** statement, which is much easier to understand.

Switch statements (4)

The expression is no longer necessarily interpretable as a boolean value.

Instead, we have different cases:

Anatomy of the switch statement

```
Expression to apply the switch to
           switch keyword
                                (Note: Doesn't need to be an integer)
                   switch (expression) {
                       case 1:
                          // Do this in case 1
                           break; ←
                                                  - This will exit the switch
if expression == 1
                       case 2:
                          // Do this in case 2
                           break;
if expression == 2
                          // Do this in case 3
                           break;
if expression == 3
                       default:
if the expression
                          // Do this if expression is
isn't covered by
                           // neither 1, 2 or 3
                          break;
any case
```

Switch vs Nested If

Switch is easier to understand since it works on one expression.

```
if (expression == 1) {
                                               switch (expression) {
                                                  case 1:
                                                       // Do this in case 1
   // Do this if 1
                                                       break;
} else if (expression == 2) {
  // Do this if 2
                                                       // Do this in case 2
                                                       break;
} else if (expression == 3) {
  // Do this if 3
                                                       // Do this in case 3
                                                       break;
} else {
                                                   default:
                                                      // Do this if expression is
  // Do this as default
                                                       // neither 1, 2 or 3
}
                                                      break;
                                               }
```

Switch only works on integers, strings, or enumerated values.

Falling through

IF we forget a **break** statement, we will keep executing conditions until we hit a **break** statement.

```
int a=1;
switch (a) {
   case 1:
                                                               Output
       System.out.println("Got a=1");
                                                                Got a=1
       System.out.println("Got a=2");
                                                                Got a=2
    case 3:
       System.out.println("Got a=3");
                                                                Got a=3
       break;
    default:
       System.out.println("Not 1, 2, or 3");
       break;
}
```

This can be useful in some cases to let the execution fall through the cases.

Flow control example



Flow control: Example (1)

Example: We want to create a small menu with user interaction.

- For instance, a (very simple) interface for an ATM.
- For now, the application will just output a message, telling us which action it took.

Select an option:

1) Withdraw from account
2) Deposit to account
3) Exit



Flow control: Example (2)

What should our program do?

Step by step, we want to:

- 1. Show a menu.
- 2. Let the user select an item in the menu.
- Make a choice based on the input.
- 4. Print a message telling us what happened.



Flow control: Example (3)

1. Show a menu

```
public static void main(String[] args) {
    // Print main menu
    System.out.println("Select an option:");
    System.out.println("-----");
    System.out.println();
    System.out.println("1) Withdraw from account");
    System.out.println("2) Deposit to account");
    System.out.println("3) Exit");
    System.out.println();
}
```

```
Select an option:
------

1) Withdraw from account
2) Deposit to account
3) Exit
```



Flow control: Example (4)

2. Let the user select an item in the menu

```
public static void main(String[] args) {
    // Print main menu
    System.out.println("Select an option:");
    System.out.println("-----");
    System.out.println();
    System.out.println("1) Withdraw from account");
    System.out.println("2) Deposit to account");
    System.out.println("3) Exit");
    System.out.println();

    Scanner in = new Scanner(System.in);
    int input = Integer.parseInt(in.nextLine());
}
```



Flow control: Example (5)

3. Make a choice based on the input

```
public static void main(String[] args) {
   // Menu code omitted
   Scanner in = new Scanner(System.in);
    int input = Integer.parseInt(in.nextLine());
   switch (input) {
       case 1:
            // The user selected "Withdraw from account"
           break;
        case 2:
            // The user selected "Deposit to account"
           break;
        default:
           // The user selected "Exit"
           // This could also be used to check for erroneous input
           break;
   }
}
```



Flow control: Example (6)

4. Print a message telling us what happened

```
public static void main(String[] args) {
   // Menu code omitted
   Scanner in = new Scanner(System.in);
    int input = Integer.parseInt(in.nextLine());
    switch (input) {
        case 1:
            // The user selected "Withdraw from account"
            System.out.println("Withdrawing 100 SEK from the account");
            break;
        case 2:
            // The user selected "Deposit to account"
            System.out.println("Depositing 100 SEK to the account");
        default:
            // The user selected "Exit"
// This could also be used to check for erroneous input
            System.out.println("Bye!");
            break;
    }
}
```

Exercise 6

Lets do exercise 6

