8. Decision-Making Statements

Java

Summer 2008

Instructor: Dr. Masoud Yaghini

Outline

Introduction

Control flow statements

- The statements inside your source files are generally executed from top to bottom, in the order that they appear.
- **Control flow statements**, however, break up the flow of execution and enabling your program to conditionally execute particular

Control flow statements

- **Control flow statements:**
  - the decision-making statements (if-then, if-then-else, switch)
  - the looping statements (for, while, do-while), and
  - the branching statements (break, continue, return)
- This chapter describes the decision-making statements

The if-then Statement

The if-then Statement

- The if-then statement is the most basic of all the control flow statements.
- It tells your program to execute a certain section of code only if a particular test evaluates to True.
- For example, the Bicycle class could allow the brakes to decrease the bicycle's speed only if the bicycle is already in motion.

The if-then Statement

- One possible implementation of the applyBrakes method could be as follows:

```
void applyBrakes() {
    if (isMoving) {            // the "if" clause: bicycle must moving
            currentSpeed--;    // the "then" clause:
                               // decrease current speed
    }
}
```

The if-then Statement

- In addition, the opening and closing braces are optional, provided that the "then" clause contains only one statement:

```
void applyBrakes() {
    if (isMoving)  currentSpeed--;           // same as above,
                                             // but without braces
}
```

- This form is not recommended.

The if-then-else Statement

The if-then-else Statement

- The if-then-else statement provides a secondary path of execution when an "if" clause evaluates to false.
- You could use an if-then-else statement in the applyBrakes method to take some action if the brakes are applied when the bicycle is not in motion.

The if-then-else Statement

- In this case, the action is to simply print an error message stating that the bicycle has already stopped.

```
void applyBrakes() {
    if (isMoving) {
        currentSpeed--;
    } else {
        System.err.println("The bicycle has already stopped!");
    }
}
```

The if-then-else Statement

- The following program assigns a grade based on the value of a test score:

```
class IfElseDemo {
    public static void main(String[] args) {

        int testscore = 76;
        char grade;

        if (testscore >= 90) {
            grade = 'A';
        } else if (testscore >= 80) {
            grade = 'B';
        } else if (testscore >= 70) {
            grade = 'C';
        } else if (testscore >= 60) {
            grade = 'D';
        } else {
            grade = 'F';
        }
        System.out.println("Grade = " + grade);
    }
}
```

The switch Statement

The switch Statement

- The switch statement allows for any number of possible execution paths.
- A switch works with the byte, short, char, and int primitive data types.
- It also works with some other types which discussed in later

The switch Statement

```
class SwitchDemo {
    public static void main(String[] args) {
        int month = 8;
        switch (month) {
            case 1:  System.out.println("January"); break;
            case 2:  System.out.println("February"); break;
            case 3:  System.out.println("March"); break;
            case 4:  System.out.println("April"); break;
            case 5:  System.out.println("May"); break;
            case 6:  System.out.println("June"); break;
            case 7:  System.out.println("July"); break;
            case 8:  System.out.println("August"); break;
            case 9:  System.out.println("September"); break;
            case 10: System.out.println("October"); break;
            case 11: System.out.println("November"); break;
            case 12: System.out.println("December"); break;
            default: System.out.println("Invalid month."); break;
        }
    }
}
```

The switch Statement

- You could also implement the same thing with if-then-else statements:

```
int month = 8;
if (month == 1) {
    System.out.println("January");
} else if (month == 2) {
    System.out.println("February");
}
... // and so on
```

The switch Statement

- The body of a switch statement is known as a ***switch block***.
- Any statement immediately contained by the switch block may be labeled with one or more case or default labels.
- The switch statement evaluates its expression and executes the appropriate case.

The switch Statement

- Deciding whether to use if-then-else statements or a switch statement is sometimes a judgment call. You can decide which one to use based on readability and other factors.
- An if-then-else statement can be used to make decisions based on ranges of values or conditions,
- whereas a switch statement can make decisions based only on a single integer or enumerated value.

break statement

- Each break statement terminates the enclosing switch statement.
- The break statements are necessary because without them, control will flow sequentially through subsequent case statements.

break statement

```
class SwitchDemo2 {
    public static void main(String args[]) {
        int k = 10;

        switch (k) {
            case 5:
                System.out.println(" case k = 5");
                break;
            case 10:
                System.out.println(" case k = 10");
            case 15:
                System.out.println(" case k = 15");
                break;
            default:
                System.out.println(" case default");
        }
    }
}
```

break statement

```
class SwitchDemo3 {
  public static void main(String[] args) {

    int month = 2;
    int year = 2000;
    int numDays = 0;

    switch (month) {
      case 1:
      case 3:
      case 5:
      case 7:
      case 8:
      case 10:
      case 12:
        numDays = 31;
        break;
      case 4:
      case 6:
      case 9:
      case 11:
        numDays = 30;
        break;
      case 2:
        if ( ((year % 4 == 0) && !(year % 100 == 0)) || (year % 400 == 0) )
          numDays = 29;
        else
          numDays = 28;
        break;
      default:
        System.out.println("Invalid month.");
        break;
    }
    System.out.println("Number of Days = " + numDays);
  }
}
```

break statement

- This is the output from the program:

```
Number of Days = 29
```

- Technically, the final break is not required because flow would fall out of the switch statement anyway.
- However, we recommend using a break so that modifying the code is easier.
- The default section handles all values that aren't explicitly handled by one of the case sections.

References

References

- S. Zakhour, S. Hommel, J. Royal, I. Rabinovitch, T. Risser, M. Hoeber, **The Java Tutorial: A Short Course on the Basics**, 4th Edition, Prentice Hall, 2006. (Chapter 3)

The End