

# Java Foundations

## switch Statement



## Objectives

- This lesson covers the following objectives:
  - Create a switch control structure
  - Compare if/else constructs with switch control structures
  - Understand the purpose of the break keyword



# What About Using an if/else Statement?

- Consider the scenario where you need to write a Java program to implement the following:
  - User enters a school grade between 9 to 12 and the program prints the name of the grade
- First, let's start with a solution using an if/else statement

## Solution: if/else Statement

```
Scanner in = new Scanner(System.in);
System.out.println("Enter your grade");
int grade = in.nextInt();
if (grade == 9){
    System.out.println("You are a freshman");
}
else if (grade == 10) {
    System.out.println("You are a sophomore");
}
else if (grade == 11) {
    System.out.println("You are a junior");
}
else if (grade == 12) {
    System.out.println("You are a senior");
}
else {
    System.out.println("Invalid grade");
}//endif
```

Complex  
conditions with a  
chained if  
construct tend to  
be confusing to  
read and hard to  
maintain

# The switch Statement

- The switch statement provides more efficient syntax for choosing among several alternatives

```
switch (<variable or expression>) {  
    case <literal value>: //code_block1  
        [break;]  
    case <literal value>: // code_block2  
        [break;]  
    default: //default_code  
}//end switch
```

## Solution: switch Statement

```
Scanner in = new Scanner(System.in);  
System.out.println("What grade are you in?");  
int grade = in.nextInt();  
switch (grade) {  
    case 9:  
        System.out.println("You are a freshman");  
        break;  
    case 10:  
        System.out.println("You are a sophomore");  
        break;  
    case 11:  
        System.out.println("You are a junior");  
        break;  
    case 12:  
        System.out.println("You are a senior");  
        break;  
    default:  
        System.out.println("Invalid grade");  
}//end switch
```

# The switch statement

- Compared with the if/else statement the switch statement:
  - Is more streamlined than chained if statements
  - Is easier to read and maintain
  - Simplifies the organization of the various branches of code that can be executed
  - Offers better performance
  - Can be used for complex conditions

## When to Use switch Constructs

- Use when you are testing:
  - Equality (not a range)
  - A single value
  - For fixed known values at compile time
  - int, short, byte, char, or String

```
int month = 8;
month = in.nextInt(); Only a single value can be tested

switch (month) {
    case 1: case 3: case 5: case 7: } Known values
    case 8: case 10: case 12: System.out.print("31 days");
                break;
    case 2: if(isLeapYear)){
                ..
```

## String in a switch Statement: Example

```
String typeOfDay;
String dayOfWeekArg = "Thursday";

switch (dayOfWeekArg) {
    case "Monday": typeOfDay = "Start of work week";
                    break;
    case "Tuesday":
    case "Wednesday":
    case "Thursday": typeOfDay = "Midweek";
                     break;
    case "Friday": typeOfDay = "End of work week";
                   break;
    case "Saturday":
    case "Sunday": typeOfDay = "Weekend";
                   break;
    default: System.out.print("Invalid");
}
//end switch
```

## Exercise 1



- Import and open the SwitchEx project
- Modify SwitchEx1.java to implement the following with the switch statement
  - The user enters the month as a number
  - The corresponding month name must be displayed
  - For any invalid month, the output must be displayed as “Invalid month”

## switch Statement: Keywords

- The following keywords are used in a switch statement:
  - **switch**: Specifies the variable to test for value
  - **case**: Compares the value of the switch variable
  - **default**: When the input doesn't match the cases, then the default statement is executed, however, the default statement is optional
  - **break**: Is used as the last statement in each case statement list, a break statement causes control to transfer to the end of the switch statement

## What Is a break Keyword?

- Is used as the last statement in each case statement list and it causes control to transfer outside the switch



# What Is a break Keyword?

```
char option = 'A';
int aCount = 0, bCount = 0, cCount = 0;

switch (option) {
    case 'A': aCount++;
        System.out.println("Count of A " + aCount);
        break;
    case 'B': bCount++;
        System.out.println("Count of B " + bCount);
        break;
    case 'C': cCount++;
        System.out.println("Count of C " + cCount);
        break;
}//end switch
//additional code . . .
```

## Exercise 2



- Import and open the SwitchEx project
- Observe SwitchEx2.java and execute the program
- Observe the output



## Exercise 2

- Modify the switch statement as follows:
- Remove the break statements for case ‘A’
  - Execute the program
  - Observe the output
- Remove the break statements for case ‘A’ and case ‘B’
  - Execute the program
  - Observe the output

## What Is switch Fall Through?

- switch fall through is a condition that occurs if there are no break statements at the end of each case statement
- All statements after the matching case label are executed in sequence, regardless of the expression of subsequent case labels, until a break statement is encountered.

# Understanding switch Fall Through

- Expected Output:

- The values of the count variables are incremented by 1

```
char option = 'A';
int aCount = 0, bCount = 0, cCount = 0;

switch (option) {
    case 'A': aCount++;
                System.out.println("Count of A " + aCount);

    case 'B': bCount++;
                System.out.println("Count of B " + bCount);
                break;
    case 'C': cCount++;
                System.out.println("Count of C " + cCount);
                break;
}//end switch
```

No break statement, so it continues ←  
execution with the next two case statements

## switch Fall Through: Example

```
int month = 12;
switch (month) {
    case 2: System.out.println("28 days (29 in leap years)");
              break;
    case 4:
    case 6:
    case 9:
    case 11: System.out.println("30 days");
              break;
    case 1:
    case 3:
    case 5:
    case 7:
    case 8:
    case 12: System.out.println("31 days");
              break;
    default: System.out.println("Illegal month number");
              break;
}//end switch
```

## Summary

- In this lesson, you should have learned how to:
  - Create a switch control structure
  - Compare if/else constructs with switch control structures
  - Understand the purpose of the break keyword

