Controlling Flow Control

Objectives

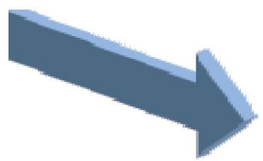
After completing this lesson, you should be able to do the following:

- Use decision-making constructs
- Perform loop operations
- Write `switch` statements

Basic Flow Control Types

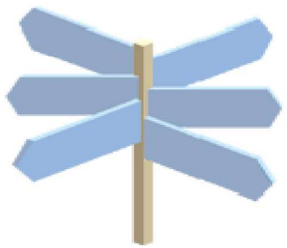
Flow control can be categorized into four types:

Sequential

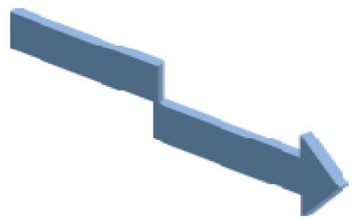


Iteration

Selection



Transfer



Using Flow Control in Java

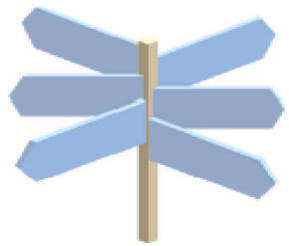
- Each simple statement terminates with a semicolon (;).
- Group statements by using braces { }.
- Each block executes as a single statement in the flow of control structure.

```
{
  boolean finished = true;
  System.out.println("i = " + i);
  i++;
}
```

`if` Statement

General:

```
if ( boolean_expr )
    statement1;
[else
    statement2;]
```



Examples:

```
if (i % 2 == 0)
   System.out.println("Even");
else
   System.out.println("Odd");
…
```

```
if (i % 2 == 0) {
    System.out.print(i);
    System.out.println(" is even");
}
```

Nested `if` Statements

```
if (speed >= 25)
  if (speed > 65)
      System.out.println("Speed over 65");
  else
      System.out.println("Speed >= 25 but <= 65");
  else
    System.out.println("Speed under 25");
```

```
if (speed > 65)
  System.out.println("Speed over 65");
else if (speed >= 25)
        System.out.println("Speed greater… to 65");
     else
        System.out.println("Speed under 25");
```

Guided Practice: Spot the Mistakes

1

```
int x = 3, y = 5;
if (x >= 0)
  if (y < x)
    System.out.println("y is less than x");
else
    System.out.println("x is negative");
```

2

```
int x = 7;
if (x = 0)
    System.out.println("x is zero");
```

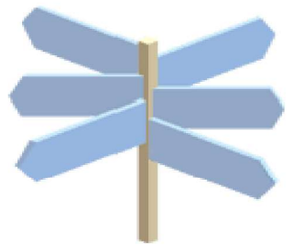
3

```
int x = 14, y = 24;
if ( x % 2 == 0  &&  y % 2 == 0 );
    System.out.println("x and y are even");
```

`switch` Statement

```
switch ( integer_expr ) {

    case constant_expr1:
        statement1;
        break;
    case constant_expr2:
        statement2;
        break;
    [default:
        statement3;]
}
```



- The `switch` statement is useful when selecting an action from several alternative integer values.
- `Integer_expr` must be `byte`, `int`, `char`, `short` or **`String`** (in Java 7).

Strings in Switch

With earlier JDK releases

```
int monthNameToDays(String s,
   int year) {
    if(s.equals("April") ||
       s.equals("June") ||
       ... )
          return 30;
    if(s.equals("January") ||
       s.equals("March") ||
       ... )
          return 31;
    if(s.equals("February"))
       ...
    else
       ...
    }
  }
```

With JDK 7 release

```
int monthNameToDays(String s, int year) {
        switch(s) {
            case "April":
            case "June":
            ...
                return 30;
            case "January":
            case "March":
            ...
                return 31;
            case "February":
                ...
            default
                ...
        }
    }
```

More About the `switch` Statement

- `case` labels must be constants.
- Use `break` to jump out of a switch.
- You should always provide a default.

```
switch (choice) {
  case 37:
     System.out.println("Coffee?");
     break;

  case 45:
     System.out.println("Tea?");
     break;

  default:
     System.out.println("???");
     break;
}
```

Looping in Java

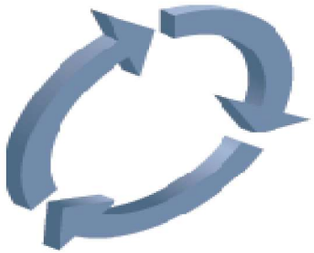
- There are three types of loops in Java:
  - `while`
  - `do…while`
  - `for`
- All (counter-controlled) loops have four parts:
  - Initialization
  - Body
  - Increment
  - Termination

while Loop

`while` is the simplest loop statement and contains the following general form:

```
while ( boolean_expr )
    statement;
```

Example:



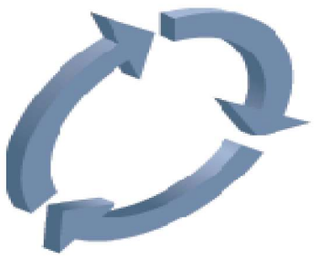
```
int i = 0;
while (i < 10) {
   System.out.println("i = " + i);
   i++;
}
```

do...while Loop

`do...while` loops place the test at the end:

```
do
    statement;
while ( termination );
```

Example:



```
int i = 0;
do {
  System.out.println("i = " + i);
  i++;
} while (i < 10);
```

`for` Loop

`for` loops are the most common loops:

```
for ( initialization; termination; increment )
    statement;
```

Example:

```
for (int i = 0; i < 10; i++)
    System.out.println(i);
```

How could this `for` loop be written as a `while` loop?

More About the `for` Loop

- Variables can be declared in the initialization part of a `for` loop:

```
for (int i = 0; i < 10; i++)
    System.out.println("i = " + i);
```

- Initialization and increment can consist of a list of comma-separated expressions:

```
for (int i = 0, j = 10; i < j; i++, j--) {
    System.out.println("i = " + i);
    System.out.println("j = " + j);
}
```

Guided Practice: Spot the Mistakes

1

```
int x = 10;
while (x > 0);
  System.out.println(x--);
System.out.println("We have lift off!");
```

2

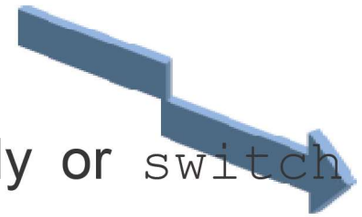
```
int x = 10;
while (x > 0)
  System.out.println("x is " + x);
  x--;
```

3

```
int sum = 0;
for (; i < 10; sum += i++);
System.out.println("Sum is " + sum);
```

`break` Statement

- Breaks out of a loop or `switch` statement
- Transfers control to the first statement after the loop body or `switch` statement
- Can simplify code but must be used sparingly



```
…
while (age <= 65) {
    balance = (balance+payment) * (1 + interest);
    if (balance >= 250000)
        break;
    age++;
}
…
```

continue Statement

- Skips the iteration of a loop
- Moves on to the next one

```
…
for (int i=0; i<=10; i++) {
//skips the print statement if i is not even
       if(i % 2 != 0) {
              continue;
       }
//prints the integer "i" followed by a space
System.out.print(i + ' ');
}
 …
```

Summary

In this lesson, you should have learned the following:

- The primary means of decision-making is the `if` statement, with the optional `else`.
- Java also offers the `switch` statement.
- Java provides three loop statements: `while`, `do…while`, and `for`.
- Use `break` and `continue` sparingly.

Practice : Overview

This practice covers the following topics:

- Performing tests by using `if…else` statements
- Using loops to perform iterative operations
- Using the `break` statement to exit a loop
- Using the `&&`, `||`, and `!` operators in Boolean expressions