Java Control Statements

shape Description

Java Control Statements, Control statements are  makes the developer to change the flow of code execution that altering the normal program flow to jump directly on some statement. Control statements can be divided as three types as follows.

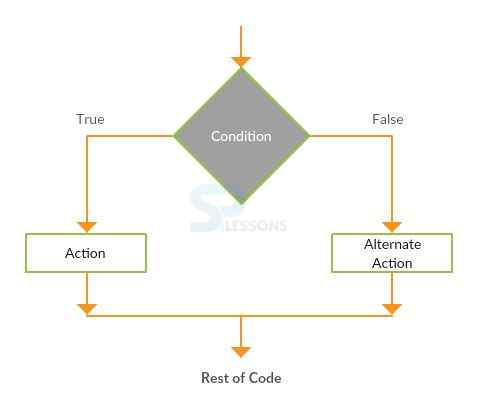
* Decision making statements
* Loop statements
* Branching Statements

shape Description

Java Control Statements, Decision making statements are utilized to test the program condition, following are the some statements.

* If
* If-else
* Switch

If statement is used to return the true value, following is the conceptual figure which can guide simply. In the following figure if condition is true then it goes for action otherwise it checks for the alternative action finally code will be closed.

[](https://cdn.splessons.com/spf/c81e728d9d4c2f636f067f89cc14862c/wp-content/uploads/2016/08/splessons-07-ifelse-corejava.jpg)

Following is an example to understand the **if statement**.  
**leapyear.java**  
To know the logic behind leap year program two rules need to be remember as follows.  
Rule 1: A year is called leap year if it is divisible by 400 such as 1600, 2000 etc.  
Rule 2: If year is not divisible by 400 as well as 100 but it is divisible by 4 then that year are also leap year such as 2004, 2008, 1012 etc.

|  |
| --- |
| import java.util.Scanner;  public class leapyear {      private static Scanner input;        public static void main(String args[])      {          int year;          System.out.println("Enter the year");          Scanner s=new Scanner(System.in);          year = s.nextInt();          if(year%400==0 || year%100 ==0 || year%4==0)//this is the if condition{              System.out.println("Entered number is leap year");          }      }    } |

**Output**

|  |
| --- |
| Enter the year  2016  Entered number is leap year |

Now Splessons will explain the above example with **if else** statement. The following is the syntax for the **if else** condition.

|  |
| --- |
| if(Boolean\_expression) {     // Executes when the Boolean expression is true  }else {     // Executes when the Boolean expression is false  } |

**leapyear.java**

|  |
| --- |
| import java.util.Scanner;  public class leapyear {      private static Scanner input;        public static void main(String args[])      {          int year;          System.out.println("Enter the year");          Scanner s=new Scanner(System.in);          year = s.nextInt();          //if statement results true          if(year%400==0 || year%100 ==0 || year%4==0){              System.out.println("Entered number is leap year");          }          //else statement results false          else{              System.out.println("it is not a leap year");          }      }    } |

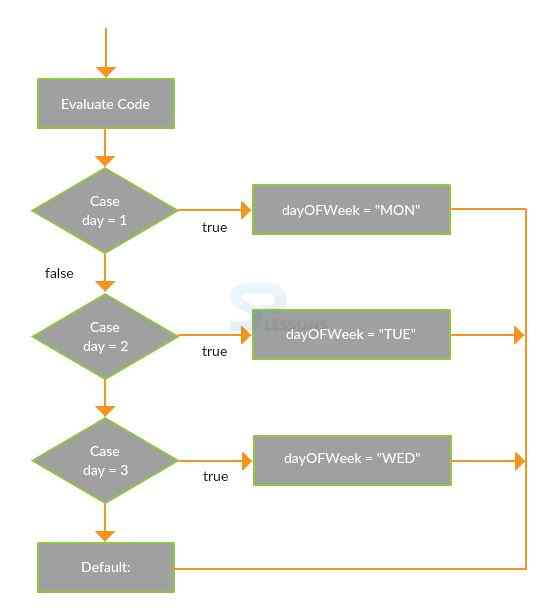
**Output**

|  |
| --- |
| Enter the year  2006  it is not a leap year |

The following are the some differences between **if**and **else if**.

* The **if**condition is mandatory for a conditional construct. An **else if** condition cannot exist without a preceding **if** block.
* User can have only one “if” block but multiple “else if” blocks.
* Both “if” and “else if” are conditional expressions that help in the decision-making  
  process.

The **switch** statement is executes one statement from multiple conditions. Following is the conceptual figure to under stand the switch concept. In the following figure compiler checks the every condition if it is true prints the statement otherwise checks the case until find the solution.

[](https://cdn.splessons.com/spf/c81e728d9d4c2f636f067f89cc14862c/wp-content/uploads/2016/08/splessons-06-switch-corejava.jpg)

Following is an example.

|  |
| --- |
| class Switch{  public static void main(String[] args) {      int number=2;      switch(number){      case 1: System.out.println("100");break;      case 2: System.out.println("200");break;      case 3: System.out.println("300");break;      default:System.out.println("Not in 100, 200 or 300");      }  }  } |

**Output**  
When compile the code following is an example.

|  |
| --- |
| 200 |

The following are the features of **switch case**.

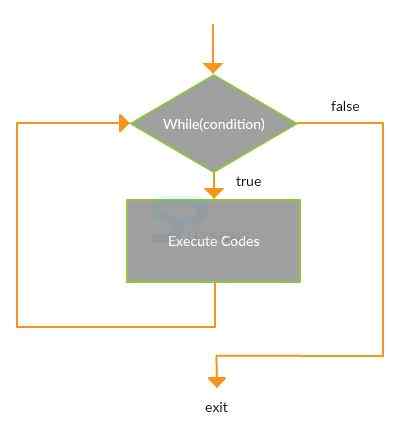
* It assesses the value of an expression and a piece of code is chosen on the premise of that assessed expression.
* Each case alludes back to the first expression.
* The data type that can be utilized as a part of switch expression is integer only.

shape Description

Loop statements are utilized to execute the repeatedly. Following are the Java loop statements.

* While
* DO-While
* For

While loop is utilized to iterate the code several times, in this scenario it will be used. Following is the conceptual figure which explains the while loop functionality.

[](https://cdn.splessons.com/spf/c81e728d9d4c2f636f067f89cc14862c/wp-content/uploads/2016/08/splessons-04-while-corejava.jpg)

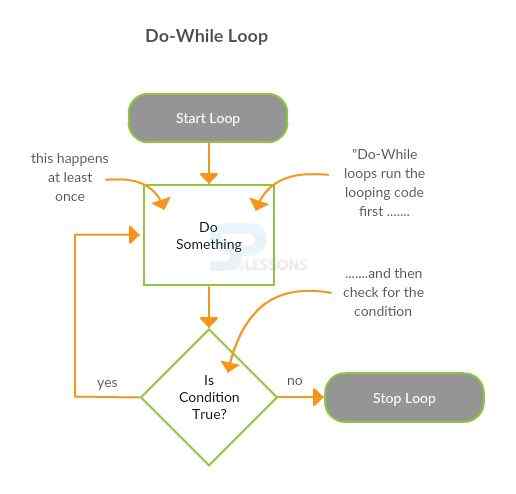
Following is an example which describes more about the while loop.  
**whileloop.java**

|  |
| --- |
| public class whileloop {            public static void main(String[] args) {              int i=1;              while(i<=10){                  System.out.println(i);              i++;              }          }          } |

**Output**  
When compile the code following is the output will be displayed.

|  |
| --- |
| 1  2  3  4  5  6  7  8  9  10 |

Java Control Statements, It is utilized to ensure that body of the loop executes at least once. Following is the conceptual figure which describes the functionality of **Do-While** functionality.

[](https://cdn.splessons.com/spf/c81e728d9d4c2f636f067f89cc14862c/wp-content/uploads/2016/08/splessons-03-dowhile-corejava.jpg)

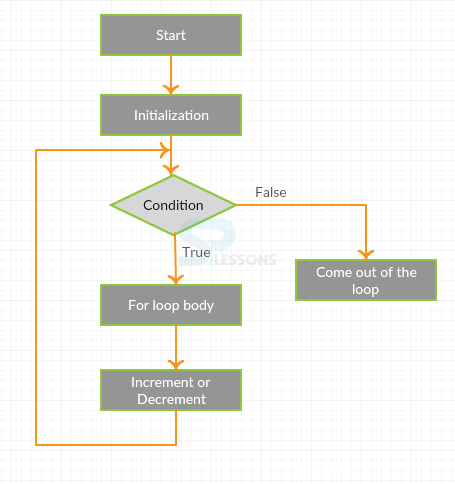
Following is an example which describes the functionality of **Do-While**.

|  |
| --- |
| public class dowhileloop {        public static void main(String[] args) {          int i=1;          do{              System.out.println(i);          i++;          }while(i<=10);      }          } |

**Output**  
When compile the code following the output will be displayed.

|  |
| --- |
| 1  2  3  4  5  6  7  8  9  10 |

For loop is utilized to repeat the particular block of the code several times.

[](https://cdn.splessons.com/spf/c81e728d9d4c2f636f067f89cc14862c/wp-content/uploads/2016/08/splessons-01-corejava-forloop.png)

**forloop.java**

|  |
| --- |
| public class forloop {      public static void main(String[] args) {          for(int i=1;i<=10;i++){              System.out.println(i);          }      }  } |

**Output**  
When compile the code following is the output will be displayed.

|  |
| --- |
| 1  2  3  4  5  6  7  8  9  10 |

The following is an example to print a pyramid by using for loop.

|  |
| --- |
| import java.util.Scanner;    public class Pyramid {      public static void main(String args[])      {          int i, space, rows, k=0;          Scanner scan = new Scanner(System.in);          System.out.print("Enter Number of Rows : ");          rows = scan.nextInt();          for(i=1; i<=rows; i++)          {              for(space=1; space<=(rows-i); space++)              {                  System.out.print("  ");              }              while(k != (2\*i-1))              {                  System.out.print("\* ");                  k++;              }              k = 0;              System.out.println();          }      }  } |

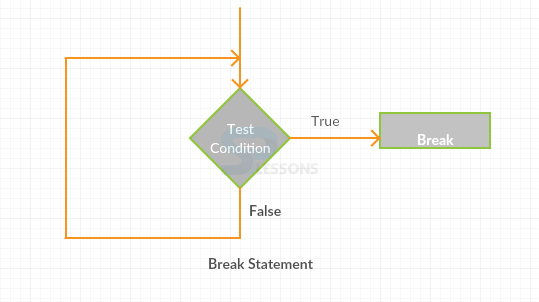
In the above example, the **java.util.Scanner** class is a straightforward content scanner which can parse primitive sorts and strings utilizing expressions.  
**Output:**

|  |
| --- |
| Enter Number of Rows : 5          \*        \* \* \*      \* \* \* \* \*    \* \* \* \* \* \* \*  \* \* \* \* \* \* \* \* \* |

shape Description

Branching statements are used to transfer the control to another point in the code.Java will have three branching statements such as Break, Continue, Return.

Break is used to stop the current flow execution of the code, in generally switch case break will be used. Following is the conceptual figure.

[](https://cdn.splessons.com/spf/c81e728d9d4c2f636f067f89cc14862c/wp-content/uploads/2016/08/splessons-02-corejava-break.png)

**Switch.java**

|  |
| --- |
| class Switch{  public static void main(String[] args) {      int number=2;      switch(number){      case 1: System.out.println("100");      break;      case 2: System.out.println("200");      break;      case 3: System.out.println("300");      break;      default:System.out.println("Not in 100, 200 or 300");      }  }  } |

**Output**  
When compile the code following is the output will be displayed.

|  |
| --- |
| 200 |

Continue statement is utilized to continue the flow of the code. Following is an example.  
**continuestatement.java**

|  |
| --- |
| public class continuestatement {      public static void main(String[] args) {          for(int i=1;i<=10;i++){              if(i==5){                  continue;              }              System.out.println(i);          }      }  } |

**Output**  
Following is the output will be generated when compile the code.

|  |
| --- |
| 1  2  3  4  5  6  7  8  9  10 |

shape Key Points

* Java Control Statements are logical statements
* To repeat the flow of code**for** loop will be used.
* While using multiple conditions **switch** statement will be used.