Escape Sequence Characters

-----------------------------------

- A character preceded by a backslash (\) is an escape sequence.

It has special meaning to the Compiler, JVM and Console softwares.

- Every escape character must be started with \ following by

a single character.

- When we place \ before a character it is not consider as a regular

character, some special functionality will be added.

- For example:

- If we use just n, n is displayed as it is.

System.out.println("Hari n Krishna");

O/P is: Hari n Krishna

- If we place \ before n, its meaning is not n

it is a new line, n is not displayed, rather

new line is displayed, means characters

placed after \n are moved to the next line.

System.out.println("Hari \n Krishna");

O/P is Hari

Krishna

The rule is:

- After \ we can not place every character as we like,

we are allowed to place only few characters

- Those few characters list will be changed from

one language to another language.

- Java supports 9 escape sequence characters

Escape

Sequence Description

\n Insert a newline in the text at this point.

\t Insert a tab space in the text at this point.

\r Insert a carriage return in the text at this point.

\s Insert a space in the text at this point.

\f Insert a formfeed in the text at this point.

\b Insert a backspace in the text at this point.

\' Insert a single quote in the text at this point.

\" Insert a double quote in the text at this point.

\\ Insert a backslash in the text at this point.

In addition to above characters Java also supports

below 8 octal digits after \

\0 - it inserts null character(nothing or empty)

\1 -

\2 -

\3 -

\4 -

\5 -

\6 -

\7 -

Java also supports UNICODE escape sequence

which is starts with \u as below

\uxxxx --> unicode number

Note: The escape sequence \s is available from Java 14v

for supporting Text blocks feature. The remaining all 8 characters

are available from beginning version Java 1.0

Working with \n

===============

- The \n is called new line escape sequence character

- it is used for inserting one new line in this text

- the characters we placed after \n will be

moved to next line and displays in the next line

Program #4: Develop a program to display your name

in two lines with single Sopln

I/P: O/P:

Hari Krishna Hari

Krishna

Working with \t

===============

- The \t is called tab space escape sequence character

- It is used for inserting or display one tab spaces in this text

- It means, in this text it inserts either 4 or 8 spaces

depending on the editor or console we used.

Program #5: Develop a program to display your name

in the below pattern

I/P: O/P:

H H

A A

R R

I I

Escape sequence characters combination usage:

==========================================

- We can also use escape sequence characters combined based on our requirement

Program #6: Develop a program to display the same above output with single Sopln(-)

To develop this program we must use \n and \t combination

Working with \s (available from Java 14v)

=========================================

- The \s is called space escape character

- By using \s we can insert one space in this text

- It is not available from beginning version Java 1.0

- It is added in Java 14v for supporting new feature "text blocks"

- In Java 14v "text blocks" feature is added as preview feature and in

Java 15v it is made as permanent

Program #7: Develop a program to show the usage of \s

I/P O/P

HK H K

//SlashS.java

class SlashS {

public static void main(String[] args) {

System.out.println("Hari\sKrishna");

}

}

//NamePattern.java

class NamePattern {

public static void main(String[] args) {

System.out.println("H");

System.out.println("\sA");

System.out.println("\s\sR");

System.out.println("\s\s\sI");

}

}

Working with \" and \'

=======================

- The \" is called as double quote escape character

- It is used to substitute the double quote(") itself in the text

- The double quote(") has special meaning in Java, it is consider as

string literal open and close.

- For printing double quote(") within the text in another ""

we must place it as \". The \ suppresses its special meaning and

consider it as normal character double quote(")

- The \' is called as single quote escape character

- It is used to substitute the single quote(') itself in the text

- Like double quote("), if we want to print single quote(') in single quote ' '

we must us \' else compiler consider its as char literal open or close,

we will get compile time errors empty character literal

- For placing " in "", we must use \"

- For placing ' in '', we must use \'

Program #8: Develop a program to display your name in "" and in '' on console

I/P O/P

Hari "Hari"

Hari 'Hari'

Working with \\

================

- The \\ is called as backslash escape character

- It is used for substituting \ as it is

- In a text of character \ is consider as escape character prefix

then the next character placed after \ must be a valid escape

character they are n t r s f b ' " \ 0 1 2 3 4 5 6 7. If we place

other these 17 characters, compiler will throw an error illegal

escape character.

- To use \ as \ we must use \\.

- To use or print \ either in "" or in '' we must place \\

Program #9:

Working with \r

================

- The \r is called as carriage return escape character

- In some console windows it work as \n as new line it means

like \n, \r also moves its right side characters

to next line

- But in other console windows it act at return character.It

means it moves its right side characters to the beginning

of the same line and replaces the existing characters

- It replaces the number of characters equals to the number of

characters exists its right side

Program #10:

Working with \b

================

- The \b is called as backspace escape character

- It is used for removing it's left side character

- Actually the remove operation is done with replace operation

- It replaces the left side character with characters

available its right side by moving one location to left

Hari \bKrishna => HariKrishna

#1: if there are no characters right side of \b

it will not remove any character,

returns text as it is

abcdef\b => abcdef

#2: if there is one character right side of \b

it will remove one character its left side,

returns text as it is

abcde\bf => abcdf

#3: if we use multiple \bs, it will not remove its left side

characters sequentially like the keyboard backspace key.

First it moves cursor to the left side number characters

equals to number of \bs, then it starts replacing

characters from this location with its left side

characters

abcde\b\bf => abcfe

abcde\b\b\bf => abfde

abcd\b\b\bef => aefd

Working with \f

================

- The \f is called as form feed escape character

- While printing a document, from the middle of the text

if we want to place as a paragraph, we must use \f

- The \f is not understood to command prompt, hence

it will display some symbol for \f.

- It will be understood by printers.

Program #11: Below program show the usage \f

class SlashF{

public static void main(String[] args){

System.out.println("abcdef\f");

System.out.println("abc\fdef");

}

}

Working with octal digits escape characters \0 to \7

====================================================

- Like \f, these escape charaters are also not used in

regular program.

- These escape characters are also displays some symbols.

- only \0 is used in program it represents null character

means empty character which is the default value of char

variable

Program #12: Below program explains the usage of octal digits

class OctalDigits {

public static void main(String[] args){

System.out.println("a\0b");

System.out.println("a\1b");

System.out.println("a\2b");

System.out.println("a\3b");

System.out.println("a\4b");

System.out.println("a\5b");

System.out.println("a\6b");

System.out.println("a\7b");

//System.out.println("a\8b");

//System.out.println("a\9b");

System.out.println("a\10b");

System.out.println("axyz\10\10b");

}

}

Program #7: Explain all above escape sequence characters functionality with a program

class EscapeSequence {

public static void main(String[] args) {

System.out.println("Hi\fHello");

System.out.println("HiHello\f");

System.out.println("\fHiHello");

System.out.println("HRU?");

System.out.println();

System.out.println("Hi\bHello");

System.out.println();

System.out.println("Hi\nHello");

System.out.println();

System.out.println("Hi\tHello");

System.out.println();

System.out.println("Hi\rHello");

System.out.println("HiHello\rHRU?");

System.out.println("HiHelloHRU?\r");

System.out.println("\rHiHelloHRU?");

System.out.println();

// System.out.println("""); //DQ in DQ is not allowed

// System.out.println('''); //SQ in SQ is not allowed

System.out.println("\"");

System.out.println('\'');

System.out.println('"'); //In SQ DQ is allowed

System.out.println("'"); //In DQ SQ is allowed

// System.out.println("\"); //HERE \ IS CONSIDERED AS ESCAPER

// System.out.println('\'); //HERE \ IS CONSIDERED AS ESCAPER

System.out.println("\\"); //We must use \\ to consider \ as a \

// System.out.println("\\\");

System.out.println("\\\\");

}

}

When an escape sequence is encountered in a print statement, the compiler interprets it accordingly. For example, if you want to put quotes within quotes you must use the escape sequence, \", on the interior quotes. To print the sentence

She said "Hello!" to me.

you would write

System.out.println("She said \"Hello!\" to me.");