**Java 7 Programming Language Enhancements**

Java 7 provides following features for Java Programming:

* **Binary Literals**
* [Switch with String](https://www.javatpoint.com/java-switch-with-string)
* [Java 7 Multi Catch](https://www.javatpoint.com/java-7-catch-multiple-exceptions)
* [Try with Resources](https://www.javatpoint.com/java-try-with-resources)
* [Type Inference](https://www.javatpoint.com/java-type-inference-for-generics)

1. [Numeric Literals](https://www.javatpoint.com/java-numeric-literals-with-underscore): UnderscoreInNumericLiteral(Example ): Java allows you to use underscore in numeric literals.

* [Java 7 JDBC](https://www.javatpoint.com/java-7-jdbc-improvement)

**JAVA NEW FEATURES**

**Java1.7**

1. String in Switch Statement
2. Catch Multiple Exceptions
3. Try with resources(Example)
4. Java 7 JDBC Improvements

**Try with resources(Example)**

Example 3:

import java.io.FileOutputStream;

public class TryWithResources {

public static void main(String args[]){

        // Using try-with-resources

try(FileOutputStream fileOutputStream =newFileOutputStream("/java7-new-features/src/abc.txt")){

String msg = "Welcome to javaTpoint!";

byte byteArray[] = msg.getBytes(); //converting string into byte array

fileOutputStream.write(byteArray);

System.out.println("Message written to file successfuly!");

}catch(Exception exception){

       System.out.println(exception);

}

}

}

**Underscore In Numeric Literal(Example ):**

Example (4):

public class UnderscoreInNumericLiteralExample {

    public static void main(String[] args) {

        // Underscore in integral literal

        int a = 10\_00000;

        System.out.println("a = "+a);

        // Underscore in floating literal

        float b = 10.5\_000f;

        System.out.println("b = "+b);

        // Underscore in binary literal

        int c = 0B10\_10;

        System.out.println("c = "+c);

        // Underscore in hexadecimal literal

        int d = 0x1\_1;

        System.out.println("d = "+d);

        // Underscore in octal literal

        int e = 01\_1;

        System.out.println("e = "+e);

    }

}