Java 13 Updations:
---
1. Text Blocks
2. New Methods in String Class for Text Blocks
3. Switch Expressions Enhancements
4. Reimplement the Legacy Socket API
5. Dynamic CDS Archive
6. ZGC: Uncommit Unused Memory
7. FileSystems.newFileSystem() Method

1. Text Blocks:

Upto Java12 version, if we want to write String data in more than one line then we have to use \n chacter in middle of the String.
EX:

```
package java13features;

public class Test {

	public static void main(String[] args) {
		String address = "\tDurga Softweare Solutions\n"+
				 "\t202, HMDA, Mitryvanam\n"+
				 "\tAmeerpet, Hyd-38";
		System.out.println(address);
	}

}
```

OP:

```
	Durga Softweare Solutions
	202, HMDA, Mitryvanam
	Ameerpet, Hyd-38
```

In the above code, to bring string data into new line wqe have to use \n character and to provide space before each and every line we have to use \t, these escape symbols may or may not support by all the operating systems.

To overcome the above problems, JAVA13 version has provided a new preview feature called as "Text Blocks", where we can write String data in multiple lines with out providing \n and \t escape symbopls.
Syntax:

```
String varName = """
                 String Data in line1
                 String Data in line2
                 ----
                 """
```

EX:

```
package java13features;

public class Test {
```

```
        public static void main(String[] args) {
                String address = """
                                Durga Softweare Solutions
                                202, HMDA, Mitryvanam\n
                                Ameerpet, Hyd-38""";
                System.out.println(address);
        }

}
```

OP:

```
Durga Softweare Solutions
202, HMDA, Mitryvanam
Ameerpet, Hyd-38
```

Note: If we want to execute preview features we have to enable preview Features first before execution, for thsi we have to use the following approaches.

1. If we want to execute program through Command prompt then use the following commands.

```
    javac --enable-preview --release 13 Test.java
    java --enable-preview Test
```

2. In Eclipse IDE:

```
     1) Right Click on Project
     2) Properties
     3) Java Compiler
     4) Uncheck "Use Compliance from Execution Evenvironment JAVASE13.
     5) Select "use --release option"
     6) Unselect "Use default complaince Settings"
     7) Select "Enable preview Features on Java13 version".
```

--> In Text Block, we have to provide data in the next line after """ and we can provide end """ either in the same line of data or in the next line, Data must not be provided in the same line of """.

EX:

```
String str1 = """Durga Software Solutions"; --> Invalid.
String str2 = """
               Durga Software Solutions"""; ---> Valid
String str3 = """
               Durga Software Solutions
              """; --------------------------> Valid
```

--> In Java, If we provide same data in normal "" and the data in """ """ then we are able to compare by using equals() and == operators , in both the cases we are able to get true value.

EX:

```
package java13features;

public class Test {

        public static void main(String[] args) {
                String data = "Durga Software Solutions";
```

```
                String newData = """
                                Durga Software Solutions""";
                System.out.println(data);
                System.out.println(newData);
                System.out.println(data.equals(newData));
                System.out.println(data == newData);
        }
}
```

OP:

```
---
Durga Software Solutions
Durga Software Solutions
true
true
```

--> It is possible to perform concatination operation between the strings which areepresented in "" and """ """.

EX:

```
---
package java13features;

public class Test {

        public static void main(String[] args) {
                String str1 = "Durga ";
                String str2 = """
                                Softare""";
                String str3 = " Solutions";
                String result1 = str1+str2+str3;
                String result2 = str1.concat(str2).concat(str3);
                System.out.println(result1);
                System.out.println(result2);
        }
}
```

OP:

```
---
Durga Softare Solutions
Durga Softare Solutions
```

--> It is possible to pass Text Blocks as parameters to the methods and it is possible to return Text Blocks from the methods.

EX1:

```
----
package java13features;

public class Test {

        public static void main(String[] args) {
                System.out.println("""
                                Durga Software Solutions
                                202, HMDA, Mitryvanam
                                Ameerpet,Hyd-38
                                """);
        }
}
```

OP:

Durga Software Solutions
202, HMDA, Mitryvanam
Ameerpet,Hyd-38

EX:

```
package java13features;

class Employee{
        public String getEmpDetails(String str) {
                String result ="""
                                Employee Details:
                                ------------------------
                                """+str+"""
                                ESAL   : 15000
                                EADDR  : Hyd
                                EEMAIL : durga@dss.com
                                EMOBILE: 91-9988776655
                                """;
                return result;
        }
}
public class Test {

        public static void main(String[] args) {
                String str = """
                                EID    : E-111
                                ENAME  : Durga
                                """;
                Employee emp = new Employee();
                String result = emp.getEmpDetails(str);
                System.out.println(result);
        }
}
```

OP:

```
Employee Details:
------------------------
EID    : E-111
ENAME  : Durga
ESAL   : 15000
EADDR  : Hyd
EEMAIL : durga@dss.com
EMOBILE: 91-9988776655
```

Text Block is very much usefull feature to represent Html or JSON code in Java applications.
EX:

```
package java13features;


public class Test {

        public static void main(String[] args) {
```

```
                String html = """
                                <html>
                                        <body>
                                                <p>Durg Software
Solutions</p>.
                                        </body>
                                </html>
                                """;
                System.out.println(html);
        }
}

OP:
----
<html>
        <body>
                <p>Durg Software Solutions</p>.
        </body>
</html>

IN the above Html code, how much spaces are provided at each and every
line the same no of spaces will be provided in the output.
EX:
---
package java13features;


public class Test {

        public static void main(String[] args) {
                String html = """
                                        <html>
                                                <body>
                                                        <p>Durg
Software Solutions</p>.
                                                </body>
                                        </html>
                                """;
                System.out.println(html);
        }
}

OP:
---
        <html>
                <body>
                        <p>Durg Software Solutions</p>.
                </body>
        </html>

--> On Text Blocks we can apply intend() method inorder to mange
spaces.
EX:
---
package java13features;
```

```
public class Test {

        public static void main(String[] args) {
                String data1 = """
                                Durga Software Solutions
                                """;
                System.out.println(data1);
                System.out.println(data1.indent(5));
                System.out.println();
                String data2 = """
                                     Durga Software Solutions
                                """;
                System.out.println(data2);
                System.out.println(data2.indent(-5));

        }
}
```

OP:

```
Durga Software Solutions

     Durga Software Solutions


     Durga Software Solutions
```

Durga Software Solutions

--> It is possible to provide ' '[Single qutations] and ""[Double quotations] directly, but, we can provide """[Triple Quotations] with \.

EX:

```
package java13features;


public class Test {

        public static void main(String[] args) {
                String str = """
                                'Durga Software Solutions'
                                "Ameerpet"
                                \"""Hyderabad\"""
                                """;
                System.out.println(str);

        }
}
```

OP:

```
'Durga Software Solutions'
"Ameerpet"
"""Hyderabad"""
```

2. New Methods in String Class for Text Blocks

For Text Blocks, JAVA13 version has provided the following three methods in String class.

1. public String formatted(Object ... values)
2. public String stripIndent()
3. public Sltring translateEscapes()

1. public String formatted(Object ... values):

--->In Text Blocks we can provide data formating by using the following method.

```
public String formatted(val1,val2,...val_n)
```

EX:

```
package java13features;

public class Test {

        public static void main(String[] args) {

                String empDetails = """
                                Employee Details:
                                ----------------------
                                Employee Number   : %d
                                Employee Name     : %s
                                Employee Salary   : %f
                                Employee Address  : %s
                                """;

System.out.println(empDetails.formatted(111,"Durga",25000.0f,"Hyd"));
                System.out.println();

System.out.println(empDetails.formatted(222,"Anil",35000.0f,"Hyd"));
                System.out.println();

System.out.println(empDetails.formatted(333,"Ravi",45000.0f,"Hyd"));
        }
}
```

OP:

```
Employee Details:
----------------------
Employee Number   : 111
Employee Name     : Durga
Employee Salary   : 25000.000000
Employee Address  : Hyd


Employee Details:
----------------------
Employee Number   : 222
Employee Name     : Anil
Employee Salary   : 35000.000000
Employee Address  : Hyd
```

```
Employee Details:
----------------------
Employee Number   : 333
Employee Name     : Ravi
Employee Salary   : 45000.000000
Employee Address  : Hyd
```

2. public String stripIndent():
--> If we have any spces at begining and Ending of String.
EX:

```
package java13features;

public class Test {

        public static void main(String[] args) {
                String data1 = "\t\tDurga Software Solutions \t\t";
                String data2 = "\t\tHyderabad\t\t";
                System.out.println(data1+data2);

System.out.println(data1.stripIndent()+data2.stripIndent());

        }
}
```
OP:

```
                Durga Software Solutions
Hyderabad
Durga Software SolutionsHyderabad
```

3. public Sltring translateEscapes()
---> It able to remove Escape symbols like \ in our Srting data.
EX:

```
package java13features;

import java.io.FileInputStream;
import java.io.FileOutputStream;

public class Test {

        public static void main(String[] args)throws Exception {

                String str1 = "Durga\nSoftware\nSolutions";
                String str2 = "Durga\\nSoftware\\nSolutions";
                System.out.println(str1);
                System.out.println();
                System.out.println(str2);
                System.out.println();
                System.out.println(str2.translateEscapes());
        }
}
```
OP:

```
Durga
Software
```

```
Solutions

Durga\nSoftware\nSolutions

Durga
Software
Solutions
```

3. Switch Expressions Enhancements:

Switch Expression in JAVA12 was introduced as preview feature, in JAVA13 also it was very same except break statement. In JAVA13 version, we can use "yield" in place of break statement to return a value.

Note:

In JAVA13 version, we are able to use switch as statement and as Expression.
In Switch statement we will use "yield" keyword to return a value in place of break statement.
IN Switch Expression, we will use -> to return value.

EX:

```
public class Hello {
    public static void main(String[] args) {
        var val = "SUNDAY";
        var result = switch (val){
            case "MONDAY","TUESDAY","WENSDAY","THURSDAY","FRIDAY":
                yield "Week Day";
            case "SATURDAY","SUNDAY":
                yield "Weekend";
            default:
                yield  "Not a Day";
        };
        System.out.println(result);
    }
}
```

OP:

```
OP:
Weekend
```

EX:

```
public class Hello {
    public static void main(String[] args) {
        var val = "SUNDAY";
       var result = switch (val){
         case "MONDAY","TUESDAY","WENSDAY","THURSDAY","FRIDAY" ->
"Week Day";
         case "SATURDAY","SUNDAY" -> "Weekend";
         default -> "Not a Day";
       };
        System.out.println(result);
```

```
    }
}
OP:
---
Weekend
```

4. Reimplement the Legacy Socket API:

Upto JAVA12 version, Socket API contains old library which was provided before JDK1.0 version which uses java.net.Socket and java.net.ServerSocket, JAVA13 version has provided new Implementation for Socket API which is based on NioSocketImpl inplace of PlainSocketImpl.

New Socket API used java.util.concurrent locks rather than synchronized methods.
If you want to use the legacy implementation, use the java option -Djdk.net.usePlainSocketImpl for backword compatibility.

5. Dynamic CDS Archive:

Class Data Sharing was introduced in JAVA10 version, it was difficult to prepare cds, but, JAVA13 version simplifies CDS creation by using the following command.

```
java -XX:ArchiveClassesAtExit=my_app_cds.jsa -cp my_app.jar

java -XX:SharedArchiveFile=my_app_cds.jsa -cp my_app.jar
```

6. ZGC: Uncommit Unused Memory:

Z Garbage Collector was introduced in JAVA11 version, it has provide small span of time before clean up memory and the unused memory was not returned to Operating System, but, JAVA13 version has provided an enhancement over
Z garbage Collector, it will return uncommitted and unused memory to Operating System.

7. FileSystems.newFileSystem() Method

Java1e3 version has provided three new methods have been added to the FileSystems class to make it easier to use file system providers, which treats the contents of a file as a file system.

```
newFileSystem(Path)
newFileSystem(Path, Map<String, ?>)
newFileSystem(Path, Map<String, ?>, ClassLoader)
```