NAME : SHREYAS MN

list of wonderful features added as part of Java 7 .

1. Diamond Operator
2. Using strings in switch statement
3. Automatic resource management
4. Numeric literals with underscore
5. Improved Exception Handling
6. New File System API (NIO 2.0)

1. Diamond Operator

We used to have declarations like below until java 1.6

```
Map<Object,Object>  mpObj = new HashMap<Object,Object>();
```

Here,if you would observe , there is a repetition of both on the left hand side and right hand side

Hence with Java 1.7 , you don't have to do a repeat on the right hand side

This means you can do something like this on the code

```
Map<Object,Object>  mpObj = new HashMap<>();
```

Because of the symbol hence created , the name "Diamond Operator" was coined in by Ronaldinho

2. Using Strings in switch statements

Until Java 1.6 , we have only int values (convertible) , enum's as contestants of switch cases . Starting

Java 1.7 , we have even java.lang.String was nominated to be a contestant for the switch cases .

for example :

```
String caseContestant= "Apple";

switch(caseContestant){ the location i was talking about is  this position . This position only convertible ints
                                        and enums had the keys to enter into the gates , now in java 1.7 ,duplicate key
                                        has been given to even java.lang.String

    case  "Apple" :
     System.out.print("I am apple");
     break;

    case  "Mango" :
     System.out.print("I am mango");
     break;
  }
```

3. Automatic Resource Management

This means that the management of resources will not be anymore done by developers, but will be
automatically taken care by Java (JVM)

Resource 1 passed as argument Resource 2 passed as argument

```
try ( FileOutputStream fos=new FileOutputStream("movies.txt") ;
DataOutputStream dos= new  DataOutputStream(fos); )
```

This is the way we specify JAVA/JVM , mentioning that we want this list of resources separated by semi colon to be handled .

```
{
   //  try  block code
}
```

Now the question arises How the hell JVM come to know anything about the elements mentioned in the above argument list separated by semi colons. The answer to the question is that all of the above mentioned elements in the list implement an interface called java.lang.AutoCloseable . This interface declares a method called "void close() throws Exception" . This method is completely used for management of resources .

4. Numeric literals with underscores

This means that we can have underscores in between the numbers for readability purposes .

```
int thousand  =   1_000  ;
int tenthousand =  10_000 ;
```

We can introduce underscores in between the numbers for readability purposes; still the value remains the same

5. Improved Exception Handling

In Java 7 , This is technically called multi-catch clause

```
catch(Exception1 |  Exception 2 | Excception 3 | Exception 4 exceptionObject)
```

In the catch block you can pass multiple exceptions separated by " | " separator . If there any exceptions happening fall in this list , it will all be caught by this catch block .

6. New File System API (NIO 2.0)

NIO stands for New Input Output .

There is complete change in the framework when it comes to file handling , input/output operations
and others .

All elements that make up the new framework are all placed under the package java.nio.

The first change is the introduction of the Path (java.nio.file.path) component in the place of File (java.io)
And for Paths , Paths (Java.nio.file.Paths) act as the Component provider .

Now the component provider for the same would be Paths from java.nio.file package

Hence the file object instantiation which had been like

```
File file= new File ("fileName");   Would now look like

Path path= new Paths.get("fileName"); Paths is the Component provider
```

Note that there is something common between the above two statements , both are just object /virtual entities . No links exists between these objects and the storage devices , until some action like create , delete , modification of the file is done .

For example , in the old API , it would be something like the below to create a file.

```
file.createNewFile();
```

In the new API it is something like ,

```
Files.createFile(path);
```

This being the utility class from the java.nio.file package

File Change Notification:

Consider a scenario :

You as a policeman (the program you write) , call out on a FBI (who is the WatchService in java terms)
to monitor a robber (directory/file) . The FBI informs you back using signals (WatchKeys in this case) .

Steps involved in accomplishing the file notification functionality .

1. Create a WatchService .This service consists of a queue to hold WatchKeys.
2. Register the directory/file you wish to monitor with this WatchService .
3. While registering ,specify the types of events you wish to recieve (create , modify or delete events) .
4. You have to start an infinite loop to listen to events .
5. When an event occurs , a WatchKey is placed into the queue.
6. Consume the WatchKey and invoke queries on it.