Java application opens

1. Single Connection with DB.
2. Multiple Connections with Single DB.
3. Multiple Connections with multiple DB’s.
4. Files.

The programmer write logic for opening resources in “try” block. Until java 1.6 version, Logic for closing opened resources to be written in final block. Otherwise It leads to several issues.

Disadvantages:- code length increased and It reduces the readability.

Example:-

BufferedReader br=null;

BufferedReader cr=null;

Try{

Br= new BufferedReader(new FileReader(“abc.txt”);

Cr= new BufferedReader(new FileReader(“data.txt”);

}

Catch(IOException e){

----

}

Finally{

If(br !=null)

Br.close();

If(cr!=null)

Cr.close();

}

**1.Try With Resources:-**

The resource reference variables can be created outside the try block or as parameters of try.

We can pass any number of resource reference variables to try block. But they should be separated with (;).

R1=open connection;

R2=open Connection;

Try(R1;R2;R3,…Rn){

-----

}

* All Resources should be AutoClosable resources. A resource is said to be auto closable if and only if the corresponding class implements java.lang.AutoClosable interface either directly or indirectly.
* All database , file, network related classes implemented AutoClosable Interface.
* AutoClosable Interface introduced in java 1.7 version.
* **Until java 1.6 version “try “ should be followed by either catch or finally . But java 1.7 onwards , try with resources can be with out catch or finally.**

Example:

Try(R1;R2;..Rn){

----

}

Advantage:-

1.Especially ,The programmer does not need to write finally block for placing logic of resources closing. Because Once execution control reaches end of try block either normally or abnormally , the resources which are opened as part of try block will be closed automatically.

Example:-

**class** A **implements** AutoCloseable{

A()

{

System.***out***.println("Resource Created");

}

**public** **void** doProcess() {

System.***out***.println("Processing Resource");

}

**public** **void** close() {

System.***out***.println("Resource Closed");

}

}

**class** suku{

**public** **static** **void** main(String args[])

{

A a1=**new** A();

A a2=**new** A();

**try**(a1;a2){

a1.doProcess();

a2.doProcess();

}

}

}

Output:-

--------

Resource Created

Resource Created

Processing Resource

Processing Resource

Resource Closed

Resource Closed