Java Exception propagation

An exception is first thrown from the top of the stack and if it is not caught, it drops down the call stack to the previous method,If not caught there, the exception again drops down to the previous method, and so on until they are caught or until they reach the very bottom of the call stack.This is called exception propagation.
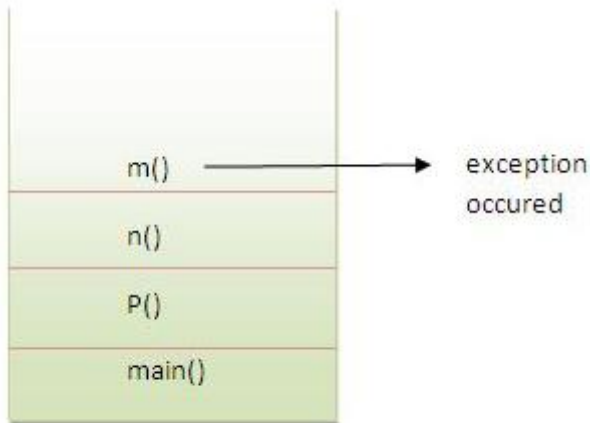
> **Rule: By default Unchecked Exceptions are forwarded in calling chain (propagated).**

Program of Exception Propagation

```java
class TestExceptionPropagation1{
  void m(){
    int data=50/0;
  }
  void n(){
    m();
  }
  void p(){
   try{
    n();
   }catch(Exception e){System.out.println("exception handled");}
  }
  public static void main(String args[]){
   TestExceptionPropagation1 obj=new TestExceptionPropagation1();
   obj.p();
   System.out.println("normal flow...");
  }
}
```

Test it Now

```
Output:exception handled
        normal flow...
```


Call Stack

In the above example exception occurs in m() method where it is not handled,so it is propagated to previous n() method where it is not handled, again it is propagated to p() method where exception is handled.

Exception can be handled in any method in call stack either in main() method,p() method,n() method or m() method.

> **Rule: By default, Checked Exceptions are not forwarded in calling chain (propagated).**

Program which describes that checked exceptions are not propagated

```
class TestExceptionPropagation2{
  void m(){
   throw new java.io.IOException("device error");//checked exception
  }
  void n(){
   m();
  }
  void p(){
   try{
    n();
   }catch(Exception e){System.out.println("exception handeled");}
  }
  public static void main(String args[]){
   TestExceptionPropagation2 obj=new TestExceptionPropagation2();
   obj.p();
   System.out.println("normal flow");
  }
}
```

Test it Now

```
Output:Compile Time Error
```

← prev   next →

Share this page

facebook   twitter   google plus   pinterest   tumbler

Latest 4 Tutorials

CouchDB   Docker

Rails   RichFaces