Parul® University

Generics Programming

Prof. Priya Pati, Assistant Professor
Computer Science & Engineering

Parul® University

DIGITAL LEARNING CONTENT

CHAPTER-4

Generics Programming

Parul® University

DIGITAL LEARNING CONTENT

What is Generics Programming?

- According to dictionary :"That which is not specific or in general"
- In Computer Science : "Type to be specified later"
- The example below showing a cup, which can hold either tea, coffee or water, nothing in particular.But with generic programming which can hold specific type.

Image source : Google

INTRODUCTION

- Similar to Template in C++
- Introduced in J2SE 5 to deal with type safe objects.(Type checking)
- Detects bugs at compile time and makes the code stable.
- Programming with classes and methods which are parameterised with types.
- Any type of objects in the collection can be stored using Generics Programming.

Generics Understanding

```
List list=new ArrayList();
list.add(“Hello”);
list.add(new Float(2.5));
for(Object o : list) { // ClassCastException at run time
String str=(String) obj;
}
List<String>list= newArrayList<String>();
list.add(“Hello”);
list.add(new Float(2.5));//Compile time error
```

Continued…

```
for(String s:list)
{
//avoids ClassCastException
}
```

- After Java 5,collection classes were used as above.
- At the time of creation of List, type of elements of the list is well specified and that is String.
- Any other type of object if added then compile time error will be there.
- No typecasting is allowed hence no ClassCastException too.

Java Generic Class

```
public class Gen<T>{
private T t1;
public T get(){ return this.t1;}
public void set(T t1){ this.t1=t1;}
public static void main(string args[]){
Gen<String> type=new Gen<>();
Type.set("Hello");
Gen ty=new Gen();
ty.set(10); //valid }}
```

Image source : Google

Generic Class

- Generics Type <T> should be parametrized
- Auto boxing is supported.
- If no type is provided then the warning messages will be produced stating that it is raw type.
- As type is not provided it becomes an Object which supports Integer and String objects.

Java Generic Method

```
public class GenMethod {
public static <T> boolean isEqual(GenericsType<T> s, GenericsType<T>
ss){return s.get().equals(ss.get());}
        public static void main(String args[]){
        GenericsType<String> s = new GenericsType<>();
        s.set("Hello");
        GenericsType<String> ss = new GenericsType<>();
        ss.set("Hello");
        boolean isEqual = GenMethod.<String>isEqual(s,ss);
        //above statement can be written simply as
```

Java Generics Method

isEqual = GenMethod.isEqual(s, ss);

//type inferencing is done.

}////Compiler will infer the type that is needed

- If whole class is not to be parameterised, only method can be created.
- Generics type constructor can also be created.
- Declaration of generic method that is enclosed within a non generic class.

Generic Code and Virtual Machine

- Java virtual machine has no concept of generic types or methods.
- Generic classes and methods turn into ordinary classes and methods by erasing the type variables.
- Type variables are erased ,producing a raw type.

Example:

The raw type for Pair<T> is reflected this way:

public class Pair{public Pair(Object first, Object second) {

Generic Code and Virtual Machine

```
this.first = first;

this.second = second;

  }publicObject getFirst() {  return first; }

  publicObject getSecond() {  return second; }

  public void setFirst(Object newValue)  { first = newValue; }

  public void setSecond(Object newValue)  { second = newValue; }

  privateObject first;

privateObject second;
```

Generics

- Generics are powerful extension to Java.
- Creates type safe and reusable code.
- Generic exception classes is not created hence it cannot extend Throwable.
- Static generic methods can be declared.

Parul® University

DIGITAL LEARNING CONTENT

CHAPTER-4

Exception Handling

What are Exceptions?

- A Java exception is an object which describes an exceptional condition.
- The exception object wraps an event which contains the information about the error, the program state when the error occurred and other custom information.
- When an exception occurs, the program halts and JRE looks for the exception to be thrown.
- Thus, an exception object can be thrown and caught.
- It handles the run time errors so that the flow of the application is maintained.

Exception-Handling

To understand exception which can be managed by the following keywords:

- try :Program statements which needs to be examined for error should be contained in the try block.
- catch:If an exception occurs using catch it can be handled.
- throw:To manually throw an exception, throw is used.
- throws:Any exception that is thrown out of a method must be specified using throws.
- finally:The code which is to be executed after try is put in this block.

Exception Handling Mechanism

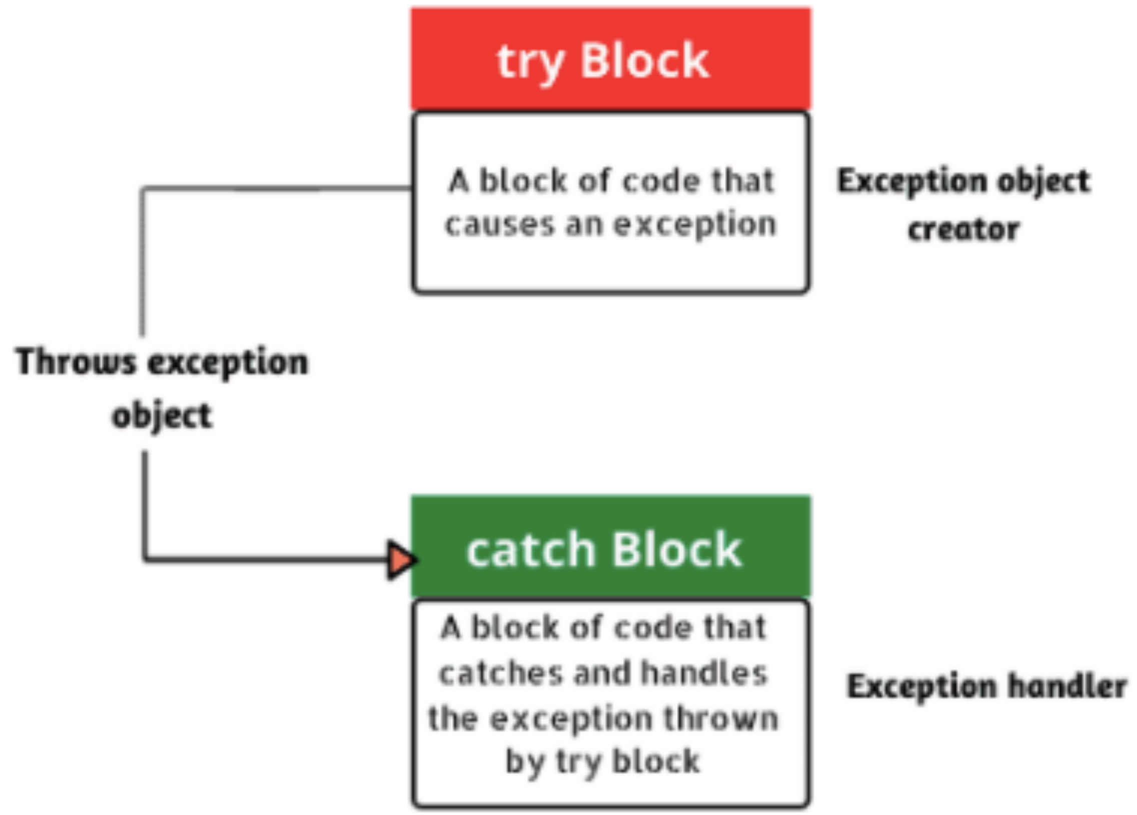

try Block
A block of code that causes an exception
Exception object creator
Throws exception object
catch Block
A block of code that catches and handles the exception thrown by try block
Exception handler

Image source : Google

Parul® University

DIGITAL LEARNING CONTENT

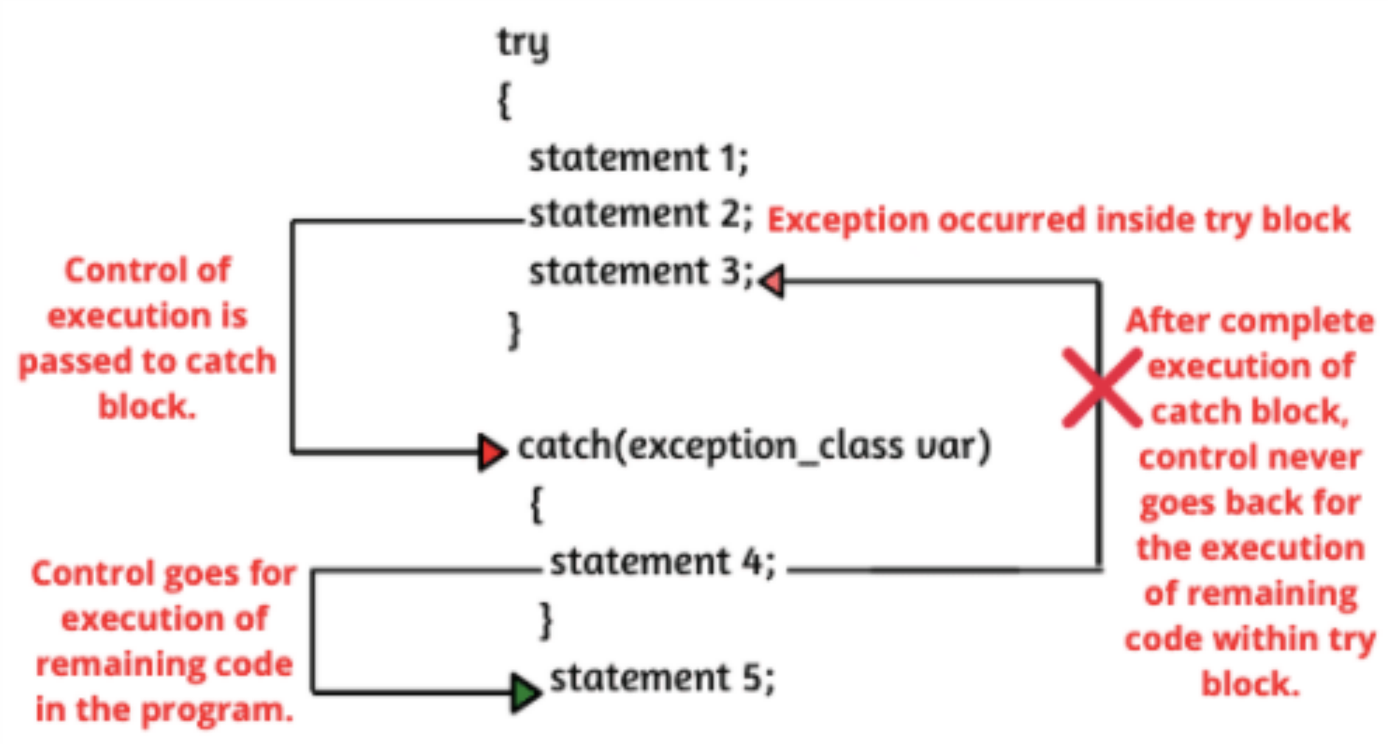
Exception-Handling Block

```
try {
  // A block of code; // generates an exception }
catch(ExceptionType1 O) {
  // Code to be executed when an exception is thrown. }
catch(ExceptionType2 O) {
  // Code to be executed when an exception is thrown. }
finally{
//block of code to be executed after try block ends
}
```

Parul® University

DIGITAL LEARNING CONTENT

Control Flow of try-catch Block



```
try
{
  statement 1;
  statement 2;
  statement 3;
}

catch(exception_class var)
 {
  statement 4;
 }
  statement 5;
```

Exception occurred inside try block
Control of execution is passed to catch block.
After complete execution of catch block, control never goes back for the execution of remaining code within try block.
Control goes for execution of remaining code in the program.

Image source : Google

Exception Hierarchy

- Any exception that when occurs creates an exception object.
- Java exceptions which is based on inheritance are hierarchical.
- The root class is Throwable which has two child objects i.e Error and Exceptions.
- Whereas Exceptions are further categorised into Checked and Unchecked Exception(Runtime Exceptions).

Parul® University

DIGITAL LEARNING CONTENT

Exception Hierarchy

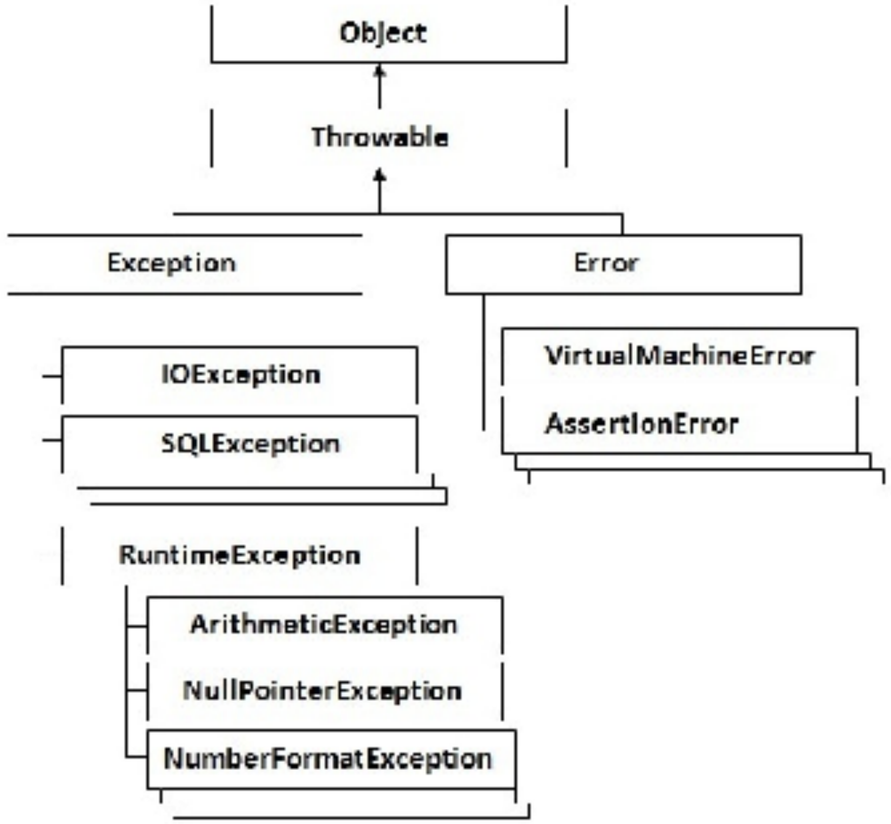


Image source : Google

Exception Hierarchy

- Error:These are out of scope of application and cannot be recovered from them.Out of memory,JVM crash and hardware failure are few conditions.Examples are:OutOfMemoryError and StackOverflowError.
- Checked Exceptions:These can be anticipated also can be recovered from using try catch blocks.Examples:IOException,SQLException
- Unchecked Exceptions:Can be avoided through better programming, it occurs when trying to fetch an element from an array, before checking the length of an array.Examples: ArithmeticException,NullPointerException etc.

Exception Exception

```
public class JavaException{
public static void main(String args[]){
try{
int data=10/0;  //Code that might cause an exception
   }catch(ArithmeticException e){System.out.println(e);}
//rest code of the program
System.out.println("Here we go !!");  }}
```

```
Output:java.lang.ArithmeticException: / by zero
         Here we go !!
```

Commonly found Exceptions are as follows:

- ArithmeticException

```
Int a=1/0; //ArithmeticException
```

- NullPointerException

```
String s=null;
System.out.println(s.length());//NullPointerException
```

- ArrayIndexOutOfBoundsException

```
int a[]=new int[4];
a[8]=100;//ArrayIndexOutOfBoundsException
```

Parul® University

Throwing and Catching Exception

- throw:By using throw statement exception can be thrown explicitly.

Example:

```
class DemoThrow{
static void demo(){
try{throw new NullPointerException(“demo”);}
catch(NullPointerException e){
System.out.println(“Caught inside demo”);
throw e;//rethrow the exception}}
```

Image source : Google

```
public static void main(String args[])
{
try{
demo();
}
catch(NullPointerException e)
{
System.out.println("Recaught:" + e);
}}}
```

1. The previous example showed how to create Java’s standard exception object.

   Ex:*throw new NullPointerException(“demo”)*

2. new is used to construct an instance of NullPointerException.

Parul® University

- throws:throws clause lists the types of exception that a method might throw.It cannot handle Error or RuntimeExceptions or their subclasses.
- The general form of method declaration :

```
  type method-name(parameter-list)
{
          //body of method
}
```

Example:

```
Class DemoThrows{
static void throwOne()throws IllegalAccessException{
Throw new IllegalAccessException(“demo”);}
public static void main(String args[]){
try{
throwOne();}catch(IllegalAccessException  e){System.out.println(“caught
e”); }}}
```

Output:

```
Inside throwOne
caught java.lang.IllegalAccessException:demo
```

1. Firstly needs to have a method which throws the exception.
2. Secondly main should have a try /catch statement to catch the exception.

- finally:The finally block will be executed after a try /catch block has completed and before the code following try/catch block.This block will be executed whether or not an exception is thrown.
- Example of finally block:

```
try{// block of code;}
catch(Exception e){//block of code;}
finally  {
 //block of code;
}
```

```
class FinallyDemo{
 public static void main (String[] args)
 {
 try{
 float y=3.2;}
 catch(Exception e){System.out.println(e);}
 finally{System.out.println(“I will get executed no matter what !!”);}}
Output:illegal character: '\u201c'
         I will get executed no matter what !!
```

Unreachable catch Block

- A block of codes/statements to which control can never reach.
- These unreachable blocks are not supported in Java.
- This condition arises when multiple catch blocks are used.
- The catch blocks should be from most specific to most general i.e.subclass of Exception must be at first and super class be next to it.

```
try{ //statement}
catch(Exception e){ System.out.println(e);}
catch(ArithematicException ae)//unreachable block
{System.out.println(ae);}
```

Stack Trace Elements

- Java Stack is composed of frames where each frame contains the state of one Java method invocation.
- Each stack frame is represented by StackTraceElement.
- The frame at the top of the stack at the zeroth element represents the execution point at which the stack trace was generated.
- The last element of the array represents the bottom of the stack.
- Throwable.getStackTrace() provides programmatic access to the stack trace information.

```
import java.lang.*;
import java.io.*;
import java.util.*;
public class StackElementDemo{
public static void main(String[] arg){
System.out.println("Class name of each thread involved");
for(int i=0;i<2;i++)
{ System.out.println(Thread.currentThread().getStackTrace()
[i].getClassName());}}}
```

Class name of each thread involved:

java.lang.Thread

StackElementDemo

Exceptions Usage

- Exception handling is a powerful mechanism for controlling complex problems arising during runtime.
- Cleaner ways to handle failure modes.
- For non local branching Java's exception handling should not be considered.

DIGITAL LEARNING CONTENT

Parul® University

www.paruluniversity.ac.i