**All Exception Questions**

**Q: What is Exception handling?**

The exception handling in java is one of the powerful mechanisms to handle the run time errors so that the normal flow of the application can be maintained. There are two type of exception :

1).Checked or compile time exception 2).Unchecked or Run Time exception.

**Q What are the different ways to handle exceptions?** Ans:  There are two ways to handle exceptions:  
 1)By wrapping the desired code in a try block   followed by a catch block to catch the exceptions. And

 2) List the desired exceptions in the throws clause of   the method and let the caller of the method handle those exceptions.

**Q: What are runtime exceptions?**

Runtime exceptions are those exceptions that are not warn by compiler but it thrown at runtime. Ex:StackOverflowException ,MemoryoutException ,ArithmaticException

**What is** regular expressio**n in java?**  
  
Ans:A regular expression is a special sequence of characters that helps you match or find other strings or sets of strings, using a specialized syntax held in a pattern. They can be used to search, edit, or manipulate text and data. Here is some expression:  
   
\A    Matches beginning of string.   
\z    Matches end of string.  
\G    Matches point where last match finished.  
\n    Back-reference to capture group number   
  
**59.What is Unsupported Operation Exception?**

* This exception is thrown to indicate that the requested operation is not supported.
* Example of UnsupportedOperationException:In other words, if you call add() or remove() method on the read-only collection .
* We know readOnly collection cannot be modified . Hence, UnsupportedOperationException will be thrown.

**Q:**

**Q: What are Checked and UnChecked Exception?**  
  
  Ans:The exceptions which are checked by compiler for smooth execution of the programme at runtime are called   checked exception.  
  Example: FileNotFoundException,NoSuchElementException  etc  In the Case of checked exceptions compiler will check   whether we are handling exception if programmer not   handling then we will get compile time error.

The exceptions which are not checked by compiler are called unchecked exception Example: StringIndexOutOfBoundsException, ArithmaticException, NuulPointException.  
 In the case of unchecked exception compiler won't check whether programmer handling exception or not.

**Q:How to handle exception in java?**  
  
By using Try Catch block we can handle exception. Basic syntacs of Try Catch block and combination of Try Catch block are explained below –  
  
  
    First combination:  
  
Try  
  
{  
  
            Any statement;  
  
}  
  
catch(AnyException Class     Any variable name)  
  
{  
  
            syso(“print whatever you want”);  
  
}  
  
  
  
    Second Combination:  
  
  
Try  
  
{  
  
            Any statement;  
  
}  
  
catch (AnyException Class     Any variable   
  
name)  
  
{  
  
            syso(“print whatever you want”);  
  
}  
  
catch (AnyException Class     Any variable   
  
name)  
  
{  
  
            syso(“print whatever you want”);  
  
}  
  
.  
  
.  
  
.  
  
catch (Exception(-parent calss of exception)      
  
 Any variable name)  
  
{  
  
            syso(“print whatever you want”);  
  
}  
  
  
  
  
    Third combination:  
  
  
Try  
  
{  
  
            Any statement;  
  
}  
  
catch (AnyException Class     Any variable   
  
name)  
  
{  
  
            syso(“print whatever you want”);  
  
}  
  
catch (AnyException Class     Any variable   
  
name)  
  
{  
  
            syso(“print whatever you want”);  
  
}  
  
.  
  
.  
  
.  
  
catch (Exception(-parent calss of exception)      
  
 Any variable name)  
  
{  
  
            syso(“print whatever you want”);  
  
}  
  
  
finally  
  
{  
  
            statements;  
  
}  
  
  
  
    Fourth combination  
  
  
Try  
  
{  
  
            Any statement;  
  
}  
  
  
finally  
  
{  
  
            statements;  
  
}  
  
  
  
\*\* finally will always execute no matter what.  
**Q:What is the difference between error and exception?**  
Ans:  
Error means is major problem which developer can not handle such as system error.

Exception means the problem which developer can handle by developer can handle using try catch block or throws keyword.

**Exception and Thread Java Interview Questions**

**Q1. What is difference between Error and Exception?**

An error is an irrecoverable condition occurring at runtime. Such as OutOfMemory error. These JVM errors you can not repair them at runtime.Though error can be caught in catch block but the execution of application will come to a halt and is not recoverable.

While exceptions are conditions that occur because of bad input or human error etc. e.g. FileNotFoundException will be thrown if the specified file does not exist. Or a NullPointerException will take place if you try using a null reference. In most of the cases it is possible to recover from an exception (probably by giving user a feedback for entering proper values etc.

**Q2. How can you handle Java exceptions?**

There are five keywords used to handle exceptions in java:

1. try
2. catch
3. finally
4. throw
5. throws

**Q3. What are the differences between Checked Exception and Unchecked Exception?**

**Checked Exception**

* The classes that extend Throwable class except RuntimeException and Error are known as checked exceptions.
* Checked exceptions are checked at compile-time.
* Example: IOException, SQLException etc.

**Unchecked Exception**

* The classes that extend RuntimeException are known as unchecked exceptions.
* Unchecked exceptions are not checked at compile-time.
* Example: ArithmeticException, NullPointerException etc.

**Q4. What purpose does the keywords final, finally, and finalize fulfill?**

**Final:**

Final is used to apply restrictions on class, method and variable. Final class can’t be inherited, final method can’t be overridden and final variable value can’t be changed. Let’s take a look at the example below to understand it better.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | class FinalVarExample {  public static void main( String args[])  {  final int a=10;   // Final variable  a=50;             //Error as value can't be changed  } |

**Finally**

Finally is used to place important code, it will be executed whether exception is handled or not. Let’s take a look at the example below to understand it better.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | class FinallyExample {  public static void main(String args[]){  try {  int x=100;  }  catch(Exception e) {  System.out.println(e);  }  finally {  System.out.println("finally block is executing");}  }}  } |

**Finalize**

Finalize is used to perform clean up processing just before object is garbage collected. Let’s take a look at the example below to understand it better.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | class FinalizeExample {  public void finalize() {  System.out.println("Finalize is called");  }  public static void main(String args[])  {  FinalizeExample f1=new FinalizeExample();  FinalizeExample f2=new FinalizeExample();  f1= NULL;  f2=NULL;  System.gc();  }  } |

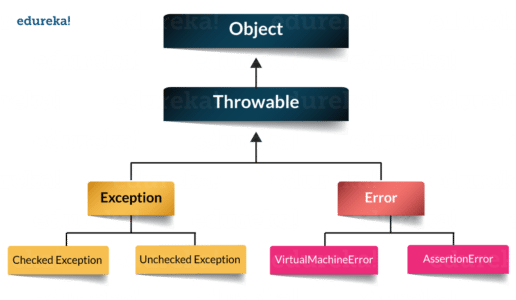
**Q5. What are the differences between throw and throws?**

|  |  |
| --- | --- |
| **throw keyword** | **throws keyword** |
| Throw is used to explicitly throw an exception. | Throws is used to declare an exception. |
| Checked exceptions can not be propagated with throw only. | Checked exception can be propagated with throws. |
| Throw is followed by an instance. | Throws is followed by class. |
| Throw is used within the method. | Throws is used with the method signature. |
| You cannot throw multiple exception | You can declare multiple exception e.g. public void method()throws IOException,SQLException. |

**Q6. What is exception hierarchy in java?**

The hierarchy is as follows:

Throwable is a parent class of all Exception classes. There are two types of Exceptions: Checked exceptions and UncheckedExceptions or RunTimeExceptions. Both type of exceptions extends Exception class whereas errors are further classified into Virtual Machine error and Assertion error.



**Q7. How to create a custom Exception?**

To create you own exception extend the Exception class or any of its subclasses.

* class New1Exception extends Exception { }               // this will create Checked Exception
* class NewException extends IOExcpetion { }             // this will create Checked exception
* class NewException extends NullPonterExcpetion { }  // this will create UnChecked exception

**Q8. What are the important methods of Java Exception Class?**

Exception and all of it’s subclasses doesn’t provide any specific methods and all of the methods are defined in the base class Throwable.

1. **String getMessage()** – This method returns the message String of Throwable and the message can be provided while creating the exception through it’s constructor.
2. **String getLocalizedMessage(**) – This method is provided so that subclasses can override it to provide locale specific message to the calling program. Throwable class implementation of this method simply use getMessage() method to return the exception message.
3. **Synchronized Throwable getCause()** – This method returns the cause of the exception or null id the cause is unknown.
4. **String toString()** – This method returns the information about Throwable in String format, the returned String contains the name of Throwable class and localized message.
5. **void printStackTrace()** – This method prints the stack trace information to the standard error stream, this method is overloaded and we can pass PrintStream or PrintWriter as argument to write the stack trace information to the file or stream.

**Q9. What are the differences between processes and threads?**

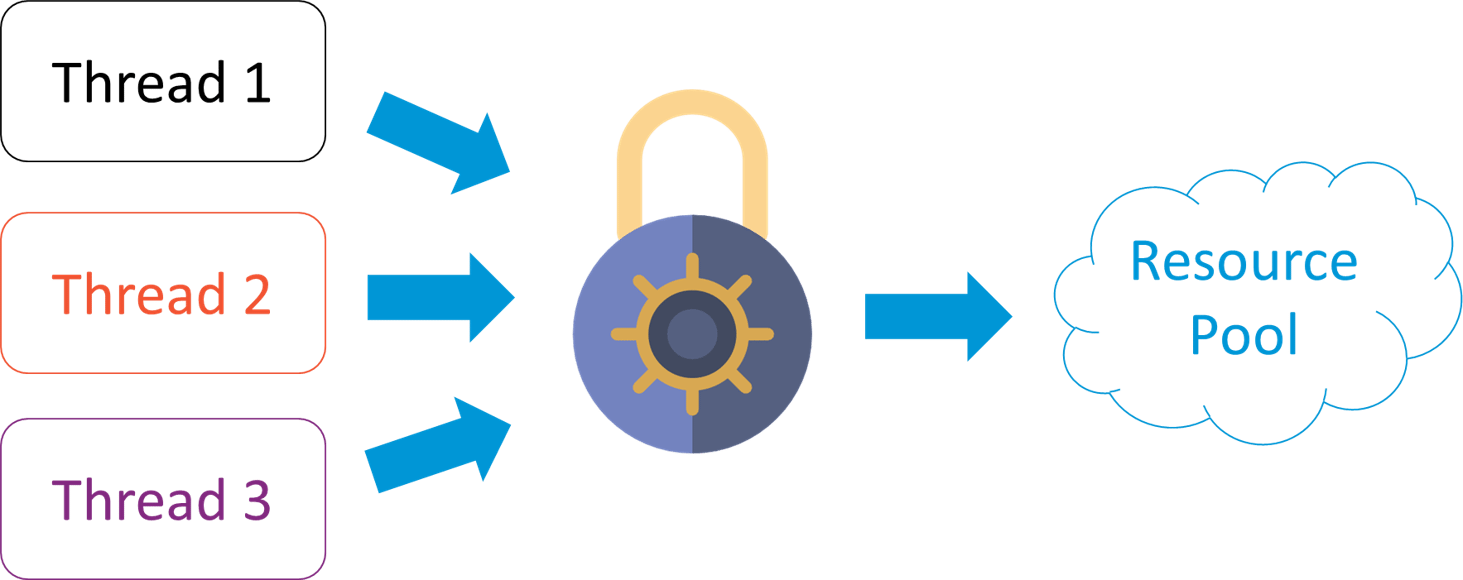
|  |  |  |
| --- | --- | --- |
|  | **Process** | **Thread** |
| **Definition** | An executing instance of a program is called a process. | A thread is a subset of the process. |
| **Communication** | Processes must use inter-process communication to communicate with sibling processes. | Threads can directly communicate with other threads of its process. |
| **Control** | Processes can only exercise control over child processes. | Threads can exercise considerable control over threads of the same process. |
| **Changes** | Any change in the parent process does not affect child processes. | Any change in the main thread may affect the behavior of the other threads of the process. |
| **Memory** | Run in separate memory spaces. | Run in shared memory spaces. |
| **Controlled by** | Process is controlled by the operating system. | Threads are controlled by programmer in a program. |
| **Dependence** | Processes are independent. | Threads are dependent. |

**Q10. What is a finally block? Is there a case when finally will not execute?**

Finally block is a block which always execute a set of statements. It is always associated with a try block regardless of any exception that occurs or not.   
Yes, finally will not be executed if the program exits either by calling System.exit() or by causing a fatal error that causes the process to abort.

**Q11. What is synchronization?**

Synchronization refers to multi-threading. A synchronized block of code can be executed by only one thread at a time. As Java supports execution of multiple threads, two or more threads may access the same fields or objects. Synchronization is a process which keeps all concurrent threads in execution to be in sync. Synchronization avoids memory consistency errors caused due to inconsistent view of shared memory. When a method is declared as synchronized the thread holds the monitor for that method’s object. If another thread is executing the synchronized method the thread is blocked until that thread releases the monitor.



**Q12. Can we write multiple catch blocks under single try block?**

Yes we can have multiple catch blocks under single try block but the approach should be from specific to general. Let’s understand this with a programmatic example.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | public class Example {  public static void main(String args[]) {  try {  int a[]= new int[10];  a[10]= 10/0;  }  catch(ArithmeticException e)  {  System.out.println("Arithmetic exception in first catch block");  }  catch(ArrayIndexOutOfBoundsException e)  {  System.out.println("Array index out of bounds in second catch block");  }  catch(Exception e)  {  System.out.println("Any exception in third catch block");  }  } |

**Q. What are the important methods of Java Exception Class?**

Methods are defined in the base class Throwable. Some of the important methods of Java exception class are stated below.

1. **String getMessage()** – This method returns the message String about the exception . The message can be provided through its constructor.
2. **public StackTraceElement[] getStackTrace() –**This method returns an array containing each element on the stack trace. The element at index 0 represents the top of the call stack whereas the last element in the array represents the method at the bottom of the call stack.
3. **Synchronized Throwable getCause()** – This method returns the cause of the exception or null id as represented by a Throwable object.
4. **String toString()** – This method returns the information in String format. The returned String contains the name of Throwable class and localized message.
5. **void printStackTrace()** – This method prints the stack trace information to the standard error stream.