**Exception Handling In Java**

**What is an Exception in Java?**

The exception in Java is an event that causes the program’s usual flow to be disrupted. It’s a type of object that’s thrown at runtime. To maintain the normal flow of the program we use Exception handling.

**What causes an exception to occur?**

There are various reasons why a program might throw an exception. For instance, opening a non-existing file in your program, a network connection error, or incorrect input data provided by the user, to name a few examples.

**Hierarchy of Java Exception classes**

The java.lang package. The Throwable class is the root of the Java Exception hierarchy, with two subclasses: Exception and Error inheriting from it.

Diagram, schematic

Description automatically generated

**What is Error?**

An error occurs when a user performs an unauthorized action that causes the program to behave abnormally. Errors in programming are frequently unnoticed until the program is compiled or run. Some of the errors make it impossible to compile or run the software. As a result, errors should be removed before compiling and running the program. There are three varieties of it.

* Compile-time
* Run-time
* Logical

**What is Exception handling?**

Exception Handling is a framework for dealing with runtime issues like ClassNotFoundException, IOException, SQLException, and RemoteException, among others. The Exception can be handled but Error can not be. We can to the type of Exception.

* Checked Exception.
* Unchecked Exception.

**Syntax**

try {  
 // Protected code  
} catch (ExceptionName e1) {  
 // Catch blo**ck  
}**

**Checked Exception**

Checked exceptions are all exceptions other than Runtime Exceptions that the compiler checks during compilation to verify if the programmer has handled them or not. Compilation errors will occur if these exceptions are not handled/declared in the program. SQLException, IOException, ClassNotFoundException, and so on.

**Unchecked Exception**

Unchecked Exceptions and Runtime Exceptions are two terms for the same thing. Because these exceptions aren’t verified at compile-time, the compiler can’t tell if the programmer has handled them or not. It’s up to the programmer to handle them and give a safe exit. ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException, and so on are examples of exceptions.

**Keywords in exception handling**

Text

Description automatically generated

**The syntax of java try-catch**

try{   
//code that may throw exception   
}catch(Exception\_class\_Name ref){}

**The syntax of a try-finally block**

try{   
//code that may throw exception   
}finally{}

**Nested try block**

In Java, a nested try block is a try block within a try block.

**2. catch Block**

To handle the Exception, a catch block in Java is utilized. Only after the try block should it be utilized. With a single try, you can use several catch blocks.

Syntax:

try  
{  
 //code that cause exception;  
}  
catch(Exception\_type e)  
{  
 //exception handling code  
}

**3. throw Keyword**

The throw statement allows you to explicitly throw an exception from your program. The following is an example of a general throw:

throw ThrowableInstance;

ThrowableInstance must be of type Throwable or a subclass of Throwable in this case.

**4. throws Keyword**

To define an exception in Java, use the throws keyword. It informs the programmer that an exception may occur.

**The syntax of java throws**

return\_type method\_name() throws exception\_class\_name{   
//method code   
}

**5. finally Block**

The finally block in Java is used to run critical code such as shutting connections, streams, and so on.

Whether or not an exception is handled, the Java finally block is always run.

Finally, the try or catch block in Java is followed by the finally block.

There can be zero or more catch blocks for each attempt block, but only one finally block.

If the application exits (either by calling System. exit() or by introducing a fatal error that causes the process to abort), the finally block will not be executed.