Java Standard Edition 8

Sandeep Kulange
sandeepkulange@sunbeaminfo.com

Operating System Resources

- Following are the operating system resources that we can use it in the program:

    1. Memory(RAM)
    2. File
    3. Thread
    4. Socket
    5. Connection
    6. IO Devices etc.

- Since OS resources are limited, we should handle it carefully. In other words, we should avoid their leakage.

Resource Type and resource in Java

- AutoCloseable is interface declared in java.lang package.
- Methods:
    1. void close() throws Exception
    2. This method is invoked automatically on objects managed by the try-with-resources statement.

- java.io.Closeable is sub interface of java.lang.AutoCloseable interface.
- Methods:
    1. void close() throws IOException
    2. This method is invoked automatically on objects managed by the try-with-resources statement.

Resource Type and resource in Java

```java
//Class Test => Resource Type
class Test implements AutoCloseable{
    private Scanner sc;
    public Test() {
        this.sc = new Scanner(System.in);
    }
    //TODO
    @Override
    public void close() throws Exception {
        this.sc.close();
    }
}
public class Program {
    public static void main(String[] args) {
        Test t = null;
        t = new Test( );    //Resource
    }
}
```

Resource Type and resource in Java

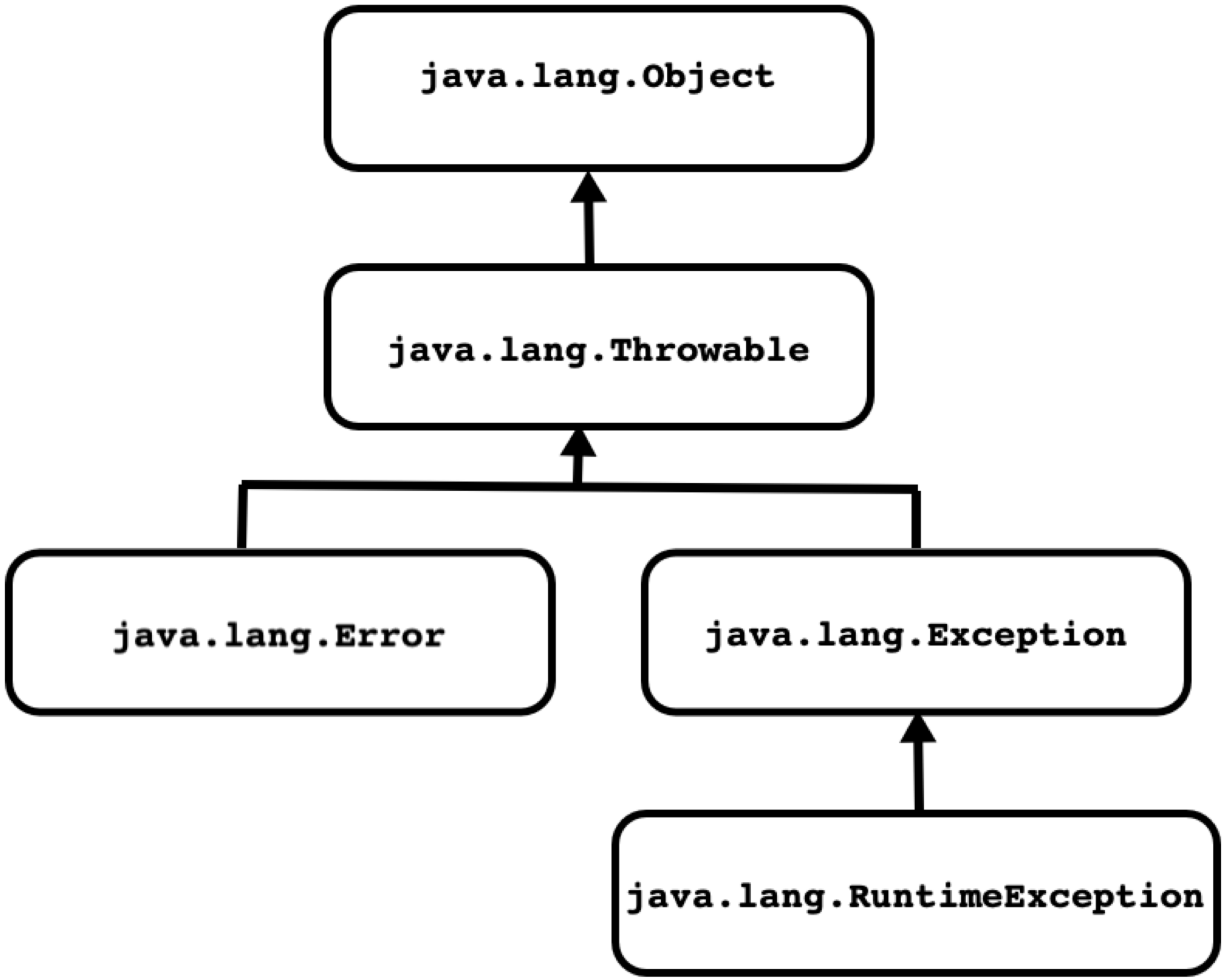
- In the context of exception handling, any class which implements java.lang.AutoCloseable or its sub interface(e.g. java.io.Closeable) is called resource type and its instance is called as resource.
- We can use instane of only resource type inside try-with-resource.
- java.util.Scanner class implements java.io.Closeable interface. Hence Scanner class is called as resource type.

Exception Handling

- **Why we should handle exception**
    1. To handle all runtime errors at single place. It helps developer to reduces maintenance.
    2. To avoid resource leakage/ to manage OS resources carefully.

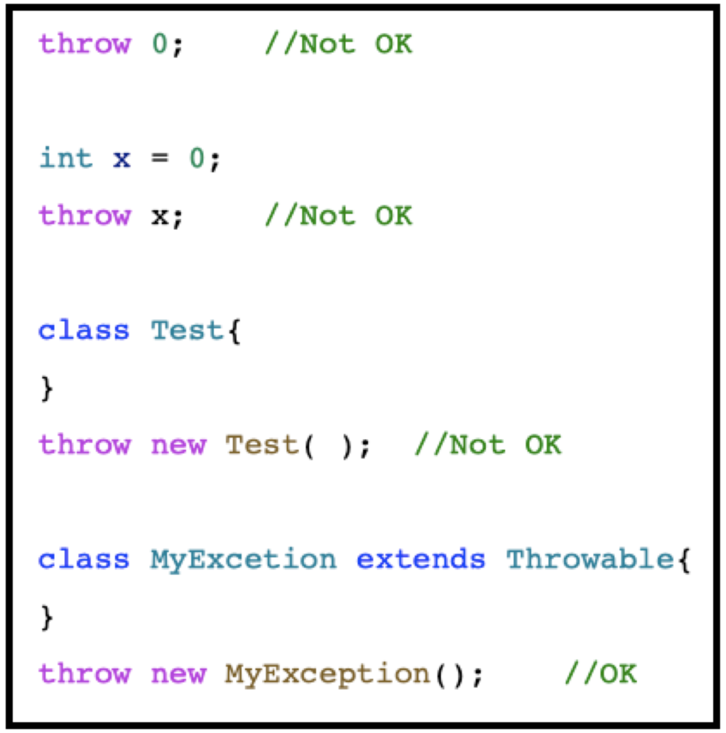
- **How can we handle exception in Java?**
    1. try
    2. catch
    3. throw
    4. throws
    5. finally

Exception Handling



Throwable Class

- It is a class declared in java.lang package.
- The Throwable class is the super class of all errors and exceptions in the Java language.
- Only instances that are instances of Throwable class (or one of its subclasses) are thrown by the Java Virtual Machine or can be thrown by the Java throw statement.



```
throw 0;     //Not OK


int x = 0;
throw x;     //Not OK


class Test{
}
throw new Test( );   //Not OK


class MyExcetion extends Throwable{
}
throw new MyException();     //OK
```

Throwable Class

- Constructors of Throwable class:

1. `public Throwable()`

   ```
   Throwable t1 = new Throwable( );
   ```

2. `public Throwable(String message)`

   ```
   Throwable t1 = new Throwable( "exception message" );
   ```

3. `public Throwable(Throwable cause)`

   ```
   Throwable cause = new Throwable( );
   Throwable t1 = new Throwable( cause );
   ```

4. `public Throwable(String message, Throwable cause)`

   ```
   Throwable cause = new Throwable( );
   Throwable t1 = new Throwable( "exception message", cause );
   ```

Throwable Class

- **Methods of Throwable class:**
    1. public Throwable initCause(Throwable cause)
    2. public Throwable getCause()
    3. public String getMessage()
    4. public void printStackTrace()
    5. public void printStackTrace(PrintStream s)
    6. public void printStackTrace(PrintWriter s)

Error

- java.lang.Error is a sub class of Throwable class.
- It gets generated due to environmental condition/Runtime environment(For Example, problem in RAM/JVM, Crashing HDD etc.).
- We can not recover from error hence we should not try to catch error. But can write try-catch block to handle error.
- Example:
    1. VirtualMachineError
    2. OutOfMemoryError
    3. InternalError
    4. StackOverflowError

Thank you