Understanding Exception Handling

- An error in a program is called bug. Removing errors from program is called debugging. There are basically three types of errors in the Java program:
- **Compile time errors:** Errors which occur due to syntax or format is called compile time errors. These errors are detected by java compiler at compilation time. Desk checking is solution for compile-time errors.
- **Runtime errors:** These are the errors that represent computer inefficiency. Insufficient memory to store data or inability of the microprocessor to execute some statement is examples to runtime errors. Runtime errors are detected by JVM at runtime.
- **Logical errors:** These are the errors that occur due to bad logic in the program. These errors are rectified by comparing the outputs of the program manually.

Exception:

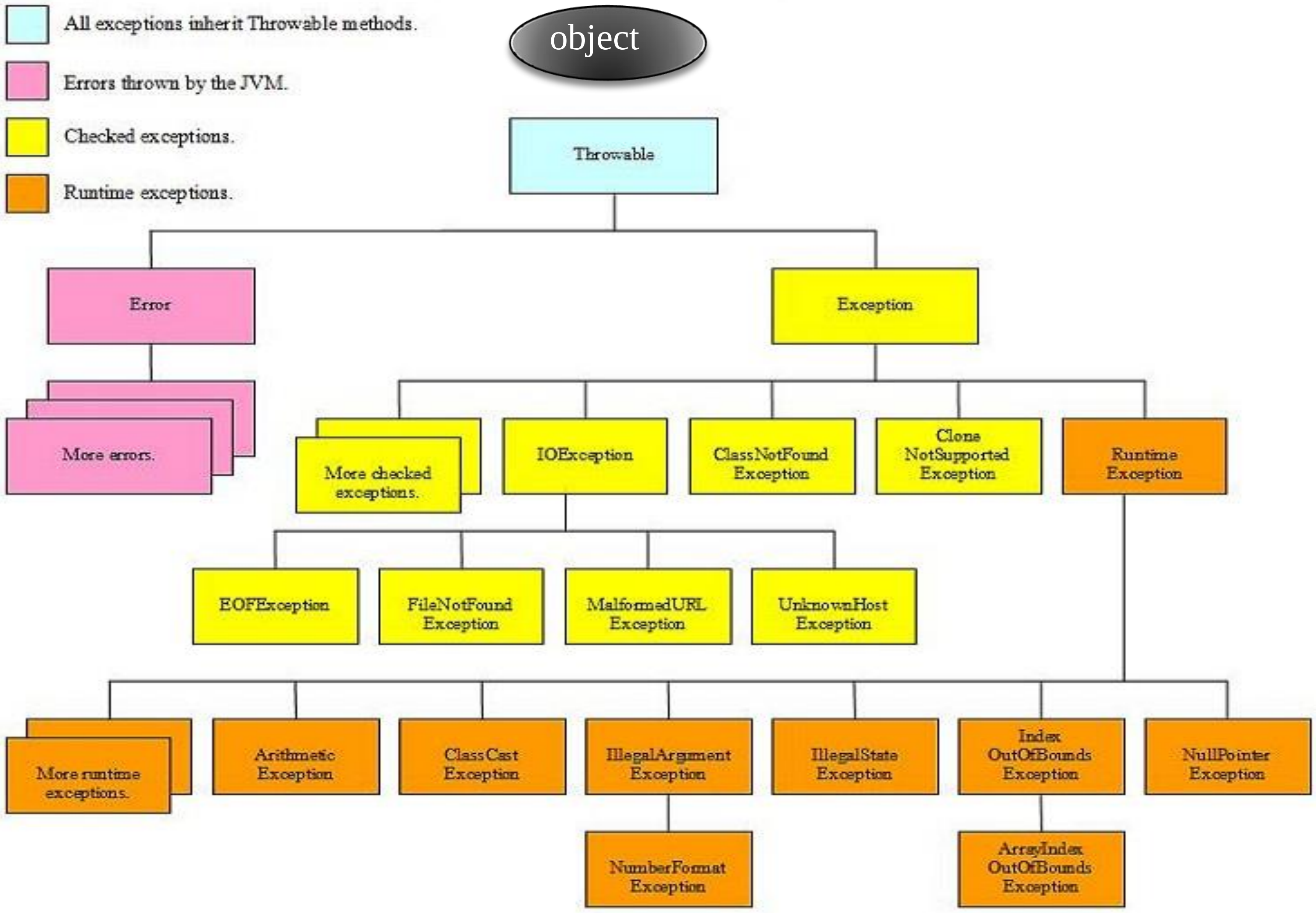
- An abnormal event in a program is called Exception.
- All Exceptions occur at runtime only but some are detected at compile time and some are detected at runtime.
- Exceptions that are checked at compile time by the java compiler are called "Checked exceptions".

eg: ClassNotFoundException, NoSuchMethodException, NoSuchFieldException etc.

- Exceptions that are checked at run time by the JVM are called "Unchecked exceptions".

eg: ArrayIndexOutOfBoundsException, ArithmeticException, NumberFormatException etc.

Exception Hierarchy



Some common exceptions scenarios

- **Scenario where ArithmeticException occurs :**

  If we divide any number by zero, there occurs an ArithmeticException.

  ```
  int a=50/0;//ArithmeticException
  ```

- **Scenario where NullPointerException occurs :**

  If we have null value in any variable, obtaining the length of that variable occurs an NullPointerException.

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

- **Scenario where ArrayIndexOutOfBoundsException occurs**

  If you are inserting any value in the wrong index, it would result ArrayIndexOutOfBoundsException as shown below:

  ```
  int a[]=new int[5];
  a[10]=50; //ArrayIndexOutOfBoundsException
  ```

List of important built-in exceptions

Exception Class	Meaning
ArithmeticException	Thrown when an exceptional condition has occurred in an arithmetic operation
ArrayIndexOutOfBoundsException	Thrown to indicate that an array has been accessed with an illegal index
ClassNotFoundException	Thrown when we try to access a class whose definition is not found
FileNotFoundException	Raised when a file is not accessible or does not open
IOException	Thrown when an input-output operation failed or interrupted
NoSuchFieldException	Thrown when a class does not contain the field(or variable) specified
NullpointerException	Raised when referring to the members of a null object

List of important built-in exceptions

Exception Class	**Meaning**
NumberFormatException	Raised when a method could not convert a string in to a numeric format
RuntimeException	This represents any exceptions which occurs during runtime
StringIndexOutOfBoundsException	Thrown by String class methods to indicate that an index is either negative or greater than the size of the string

Exception Handling

- An exception can be handled by the programmer where as an error cannot be handled by the programmer.
- Exception handling doesn't mean fixing an exception, We need to provide an alternative solution for the free flow of program.

What happens when Exception has occurred?

- ➢ Exception occurs only either inside the block or a method.
- ➢ When exception has raised, that block or method creates an exception object which contains the complete information of that exception including.
  - Name of the Exception
  - Explanation of the Exception
  - State of the Exception (Stack Trace)

- Once object has been created, it passes to JVM, then JVM handles that exception with the help of Default Exception Handler.
- Default Exception Handler checks whether the method contains any exception handling code or not. If method won't contain any handling code then JVM terminates that method abnormally and removes corresponding entry form the stack.
- Default Exception Handler just prints the basic information about the exception which has occurred and terminates the method abruptly.

- When there is an exception the programmer should do the following tasks:
- If the programmer suspects any exception in program statements, he should write them inside try block.

```
try
 {
    statements;
 }
```

- Within the try block if anywhere an exception raised then rest of the try block won't be executed even though we handled that exception.
- When there is an exception in try block JVM will not terminate the program abnormally.
- JVM stores exception details in an exception stack and then JVM jumps into catch block.
- The programmer should display exception details and any message to the user in catch block.

```
catch ( ExceptionClass obj)
    {
        statements;
    }
```

- Programmer should close all the files and databases by writing them inside finally block.
- Finally block is executed whether there is an exception or not.

```
finally
  {
        statements;
    }
```

Performing above tasks is called Exception Handling.