SNS COLLEGE OF TECHNOLOGY

Kurumbapalayam (Po), Coimbatore – 641 107

AN AUTONOMOUS INSTITUTION

Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND TECHNOLOGY

23CSB201-OBJECT ORIENTED PROGRAMMING

UNIT IV - EXCEPTION AND MULTITHREADING

TOPIC 2 - Exception types

In Java, exceptions are mainly classified into **three types**:

1. **Checked Exceptions (Compile-Time Exceptions)**
2. **Unchecked Exceptions (Runtime Exceptions)**
3. **Errors (System Exceptions)**

All these come under the root class **Throwable**.

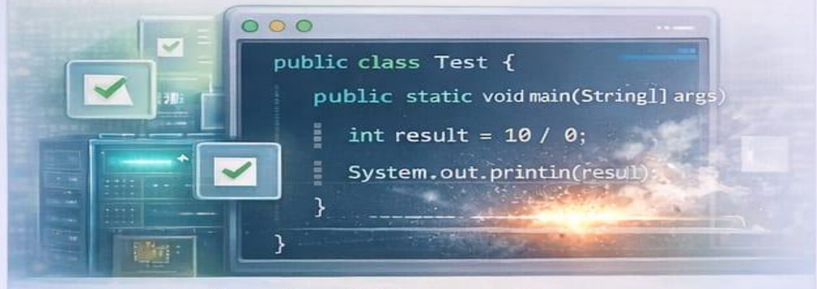
Types of Exceptions

Exceptions are classified based on when they are checked and handled.

Main Types:

Checked Exceptions

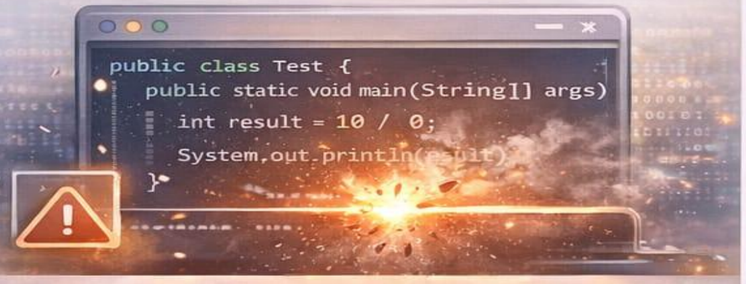
Checked at **compile time**



✓ **Checked Exceptions**
Checked at compile time

Unchecked Exceptions

Checked at **runtime**



• **Unchecked Exceptions**
Checked at runtime

• **Explanation:** Exceptions are classified based on when they are checked and handled.

1. Checked Exceptions (Compile-Time Exceptions):

✓ Definition:

- Checked exceptions are the exceptions that are **checked by the compiler at compile time**.

The programmer must handle these exceptions using:

- try-catch block OR
- throws keyword

Otherwise, the program will not compile.

✓ Characteristics:

- Occur during file handling, database access, etc.
- Mandatory to handle
- Represent recoverable conditions

✓ Examples:

- IOException
- FileNotFoundException
- SQLException
- InterruptedException

✓ Example Program:

```
import java.io.*;
class CheckedExample {
    public static void main(String[] args) {
        try {
            FileReader file = new FileReader("data.txt");
            System.out.println("File opened successfully");
        } catch (FileNotFoundException e) {
            System.out.println("File not found");
        }
    }
}
```

✓ **Key Point:**

- If you don’t handle checked exceptions → **Compile-time error**

2. Unchecked Exceptions (Runtime Exceptions):

✓ **Definition:**

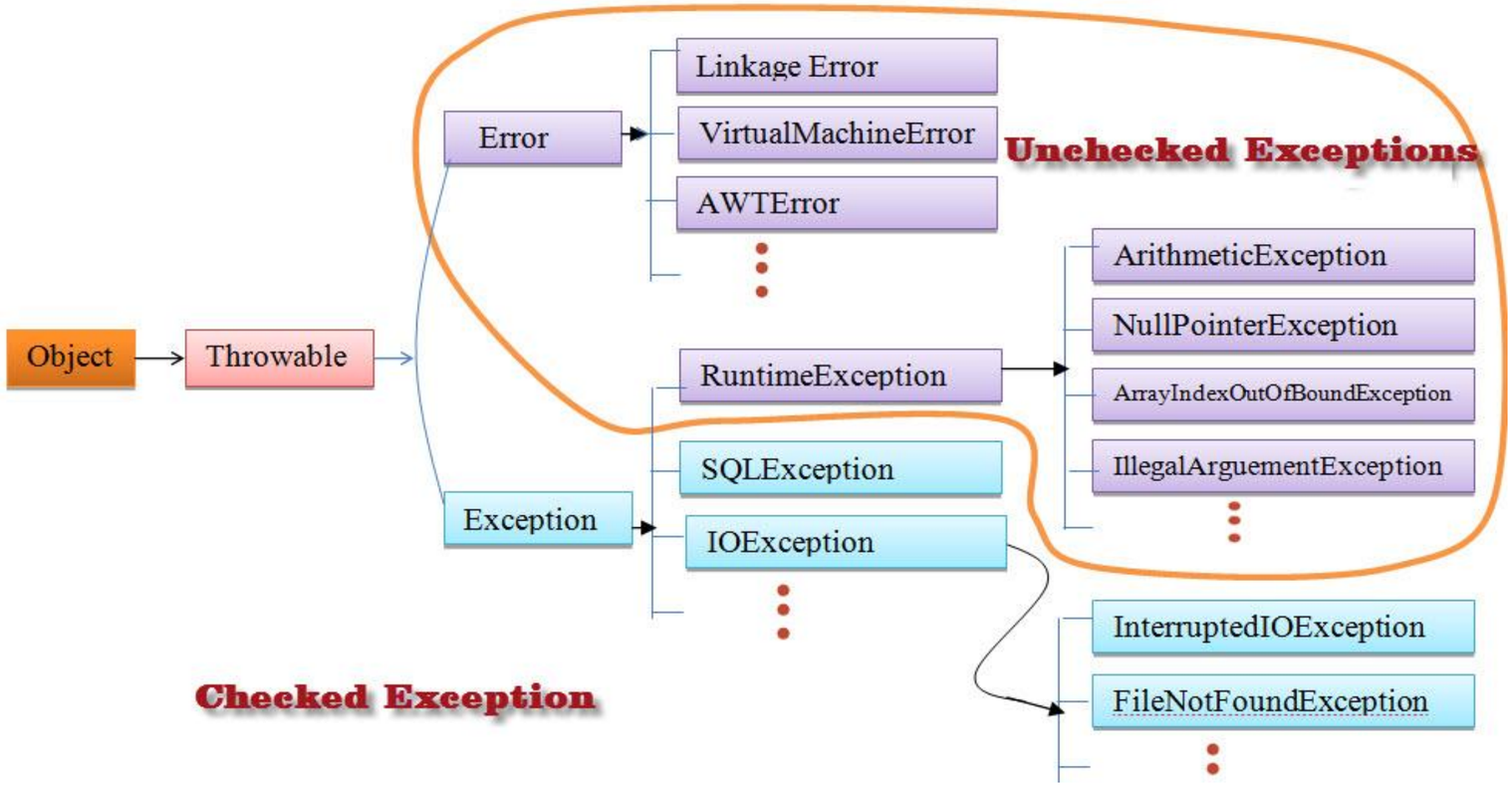
- Unchecked exceptions are the exceptions that occur **during program execution (runtime)** and are **not checked by the compiler**.

✓ Characteristics:

- Occur due to logical errors
- Not mandatory to handle
- Can be avoided by proper coding

✓ Examples:

- ArithmeticException
- NullPointerException
- ArrayIndexOutOfBoundsException
- NumberFormatException



✅ Example Program:

```
class UncheckedExample {
    public static void main(String[] args) {
        int a = 10 / 0;  // ArithmeticException
        System.out.println(a);
    }
}
```

✅ More Example:

```
class Test {
    public static void main(String[] args) {
        String s = null;
        System.out.println(s.length()); // NullPointerException
    }
}
```

✅ Key Point:

- If not handled → **Runtime crash (program terminates)**

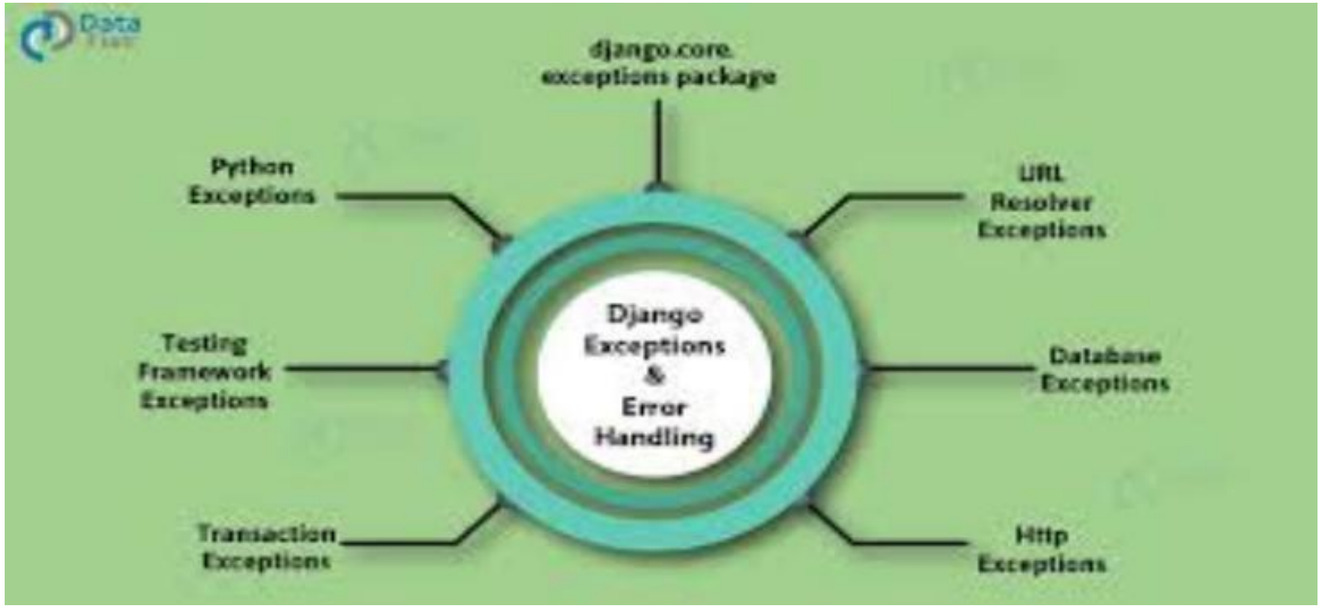
3. Errors (System-Level Exceptions):

✓ Definition:

- Errors are serious problems that occur in the **Java Virtual Machine (JVM)** and are **beyond the control of the programmer**.

✓ Characteristics:

- Cannot be handled normally
- Caused by system failure
- Rare but critical



✓ Examples:

- OutOfMemoryError
- StackOverflowError
- VirtualMachineError

✓ Example

```
class ErrorExample {
    public static void main(String[] args) {
        main(args); // infinite recursion → StackOverflowError
    }
}
```

✓ Key Point:

- Errors are **not meant to be handled** in normal applications

Exception Hierarchy Diagram:

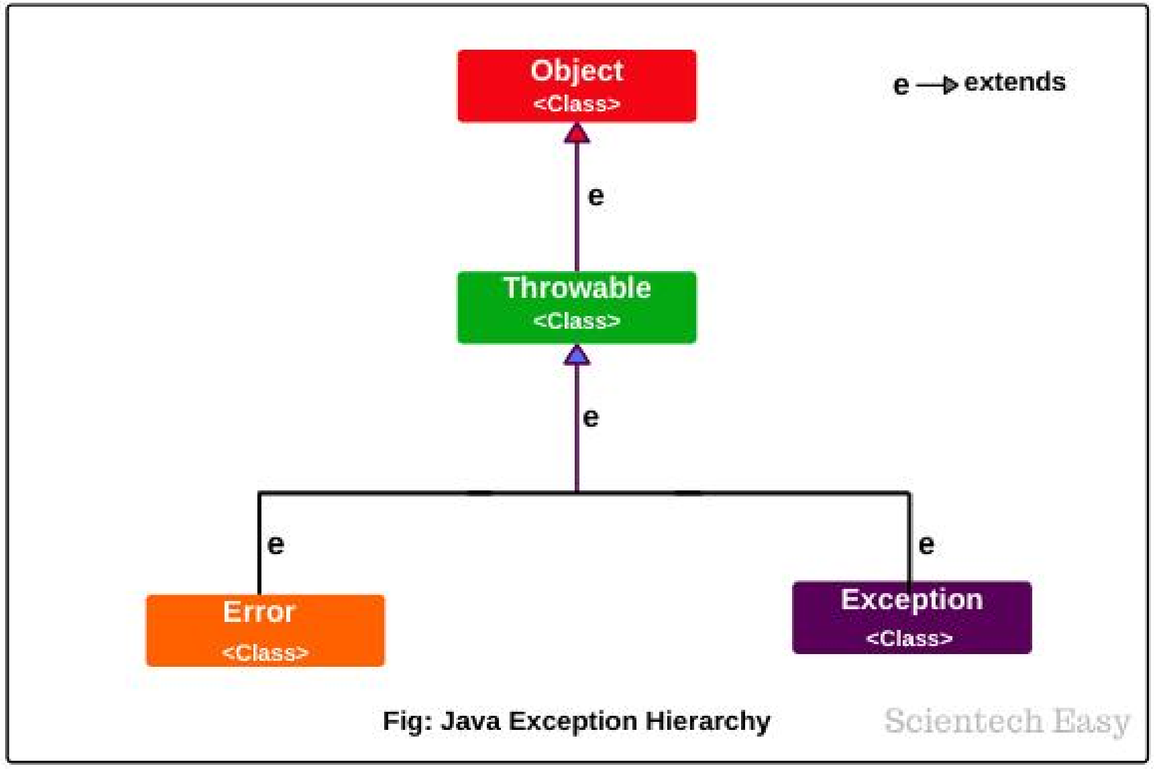
Throwable

/ \

Error Exception

/ \

Checked Exceptions Runtime Exceptions



Difference Between Checked & Unchecked Exceptions:

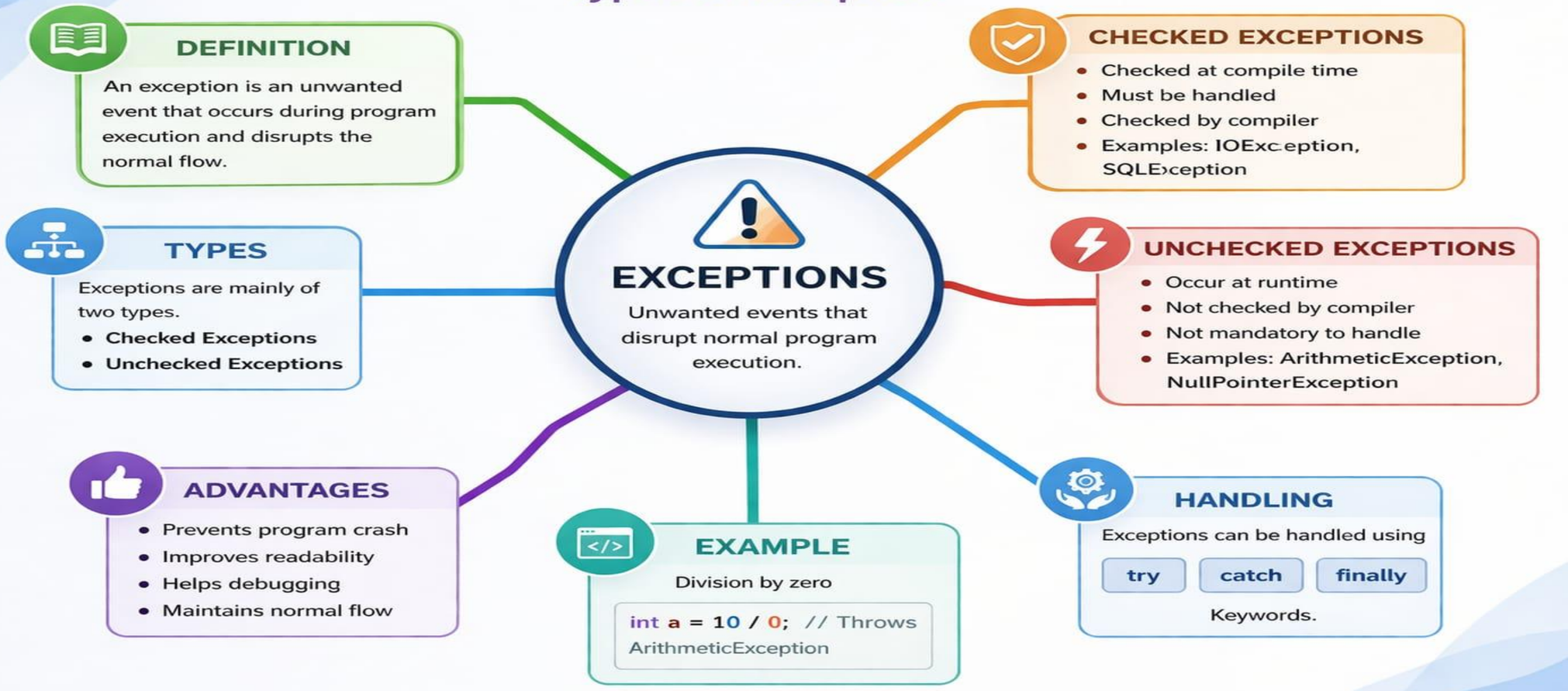
Feature	Checked Exception	Unchecked Exception
Checked by Compiler	Yes	No
Occurs	Compile-time	Runtime
Handling Required	Mandatory	Optional
Cause	External issues	Programming errors
Example	IOException	ArithmeticException

Summary:

- **Checked Exceptions** → Must handle, compile-time
- **Unchecked Exceptions** → Runtime errors, optional handling
- **Errors** → System-level, cannot be handled

MIND MAP

Types of Exception



Assessment

1. How are the keywords final, finally and finalize different from each other?
2. Can we have statements between try, catch and finally blocks?
3. Differentiate between Checked Exception and Unchecked Exceptions in Java.

THANK YOU