spring®

EraaSoft

🔥 Types of Errors in Java

1. Compile-Time Errors

Errors that the **compiler detects** when you try to compile the code.

- ❌ **Syntax Error**: Incorrect Java syntax.
  - Example: Missing semicolon, unmatched braces.
- ❌ **Type Error**: Using incompatible types.
  - Example: Assigning a `String` to an `int`.

> ✅ You must fix these before running the program.

By Eslam Khder

2. Runtime Errors

Errors that happen when the program is running.

These are further classified into:

2A. ✅ Exceptions (`java.lang.Exception`)

Handled using **try-catch**.

Examples:

- `NullPointerException`
- `ArrayIndexOutOfBoundsException`
- `IOException`

> Used for **recoverable problems** (e.g., file not found, invalid user input).

What is Exception Handling in Java?

Exception Handling in Java is a mechanism that allows you to detect and manage **runtime errors** (exceptions), so your program can continue running instead of crashing.

By Eslam Khder

🔧 How to Handle Exceptions in Java

1. Using try-catch block

java

Copy Edit

```java
try {
    int result = 10 / 0;
    System.out.println(result);
} catch (ArithmeticException e) {
    System.out.println("Cannot divide by zero: " + e.getMessage());
}
```

By Eslam Khder

2. Using finally block

java

Copy Edit

```java
try {
    int[] arr = new int[2];
    arr[3] = 10;
} catch (ArrayIndexOutOfBoundsException e) {
    System.out.println("Array index error");
} finally {
    System.out.println("Finally block always runs");
}
```

By Eslam Khder

3. Using throws keyword

java

Copy Edit

```java
public void readFile() throws IOException {
    FileReader file = new FileReader("file.txt");
}
```

By Eslam Khder

spring®

EraaSoft

4. Custom Exception

java

Copy Edit

```java
class MyException extends Exception {
    public MyException(String message) {
        super(message);
    }
}

public class Test {
    public static void main(String[] args) throws MyException {
        throw new MyException("This is a custom exception");
    }
}
```

By Eslam Khder

⚙️ Exception Class Hierarchy:

php

Copy Edit

```
Object
 └── Throwable
      ├── Exception (checked)
      │     └── RuntimeException (unchecked)
      └── Error
```

By Eslam Khder

Example: Handle Division by Zero

java

Copy Edit

```java
public class TryCatchExample {
    public static void main(String[] args) {
        try {
            int a = 10;
            int b = 0;
            int result = a / b; // This will throw ArithmeticException
            System.out.println("Result: " + result);
        } catch (ArithmeticException e) {
            System.out.println("Error: Cannot divide by zero.");
            System.out.println("Exception message: " + e.getMessage());
        }

        System.out.println("Program continues after exception handling...");
    }
}
```

By Eslam Khder

1. ArrayIndexOutOfBoundsException

java

Copy Edit

```java
public class ArrayExample {
    public static void main(String[] args) {
        try {
            int[] numbers = {1, 2, 3};
            System.out.println(numbers[5]); // Invalid index
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Error: Array index out of bounds.");
        }
    }
}
```

By Eslam Khder

2. NullPointerException

java

Copy Edit

```java
public class NullPointerExample {
    public static void main(String[] args) {
        try {
            String text = null;
            System.out.println(text.length()); // Throws NullPointerException
        } catch (NullPointerException e) {
            System.out.println("Error: Cannot operate on a null object.");
        }
    }
}
```

By Eslam Khder

3. Multiple catch blocks

java

Copy Edit

```java
public class MultipleCatchExample {
    public static void main(String[] args) {
        try {
            int[] arr = new int[3];
            arr[4] = 5; // ArrayIndexOutOfBoundsException
            int result = 10 / 0; // ArithmeticException (won't reach this line)
        } catch (ArithmeticException e) {
            System.out.println("Arithmetic Error: " + e.getMessage());
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Array Error: " + e.getMessage());
        }
    }
}
```

By Eslam Khder

4. Using finally block

java

Copy Edit

```java
public class FinallyExample {
    public static void main(String[] args) {
        try {
            String str = "hello";
            System.out.println(str.charAt(10)); // Out of bounds
        } catch (StringIndexOutOfBoundsException e) {
            System.out.println("String index error: " + e.getMessage());
        } finally {
            System.out.println("This block always executes.");
        }
    }
}
```

By Eslam Khder

5. Checked Exception (File Handling)

java

Copy Edit

```java
import java.io.*;

public class FileReadExample {
    public static void main(String[] args) {
        try {
            FileReader file = new FileReader("example.txt"); // Might throw FileNotFoundException
            BufferedReader reader = new BufferedReader(file);
            System.out.println(reader.readLine());
            reader.close();
        } catch (FileNotFoundException e) {
            System.out.println("File not found.");
        } catch (IOException e) {
            System.out.println("I/O error occurred.");
        }
    }
}
```

By Eslam Khder

✅ 1. Checked Exceptions

These are exceptions that are **checked at compile time**.

Java forces you to **handle them using** `try-catch` **or** `throws`.

🔹 **Examples:**

- `IOException`
- `SQLException`
- `FileNotFoundException`

🔹 **Why use checked exceptions?**

To ensure that the programmer **explicitly handles** conditions that might go wrong — especially in I/O, database, etc.

- **Why use checked exceptions?**

To ensure that the programmer **explicitly handles** conditions that might go wrong — especially in I/O, database, etc.

- **Example:**

java

Copy Edit

```java
import java.io.*;

public class TestChecked {
    public static void main(String[] args) {
        try {
            FileReader file = new FileReader("data.txt"); // Might throw FileNotFoundException
        } catch (FileNotFoundException e) {
            System.out.println("File not found!");
        }
    }
}
```

By Eslam Khder

❌ 2. Unchecked Exceptions

These are exceptions that are **not checked at compile time**, i.e., **runtime exceptions**.

You **can handle them**, but the compiler doesn't force you.

- **Examples:**
- `NullPointerException`
- `ArrayIndexOutOfBoundsException`
- `ArithmeticException`
- `IllegalArgumentException`

- **Why use unchecked exceptions?**

Used for **programming errors** — things that should not happen in well-written code, like null pointer access or dividing by zero.

spring

EraaSoft

- **Example:**

java

Copy Edit

```java
public class TestUnchecked {
    public static void main(String[] args) {
        int[] arr = new int[3];
        System.out.println(arr[5]); // ArrayIndexOutOfBoundsException
    }
}
```

By Eslam Khder

📌 Summary Table:

Feature	Checked Exception	Unchecked Exception
Compile-time check?	✅ Yes	❌ No
Must handle with try/catch or throws?	✅ Yes	❌ No
Inherits from	`Exception`	`RuntimeException`
Example	`IOException`	`NullPointerException`

🔧 How to create your own:

java Copy Edit

```java
// Checked Exception
class MyCheckedException extends Exception { }

// Unchecked Exception
class MyUncheckedException extends RuntimeException { }
```

By Eslam Khder