**Summary of Exception Handling in Java**

**1. Types of Exceptions**

Java has two main types of exceptions:

* **Checked Exceptions**:
  + These exceptions must be handled or declared in the method signature using throws.
  + They are typically caused by external factors like IO issues or database problems.
  + Examples: IOException, SQLException, ClassNotFoundException.
* **Unchecked Exceptions**:
  + These exceptions are not required to be explicitly caught or declared.
  + They generally occur due to programming errors, like accessing an invalid array index or dividing by zero.
  + Examples: NullPointerException, ArrayIndexOutOfBoundsException, ArithmeticException.
  + These exceptions extend RuntimeException.

**2. Using try, catch, and finally Blocks**

* **try**: The block where code that might throw an exception is written.
* **catch**: Catches and handles the exception thrown in the try block.
* **finally**: Always executes, regardless of whether an exception occurred or not. Used for cleanup operations.

**Example**:

java

CopyEdit

public class Main {

public static void main(String[] args) {

try {

int result = 10 / 0; // Throws ArithmeticException

} catch (ArithmeticException e) {

System.out.println("Error: Cannot divide by zero!");

} finally {

System.out.println("This will always be executed.");

}

}

}

**Output**:

vbnet

CopyEdit

Error: Cannot divide by zero!

This will always be executed.

**3. throw and throws Keywords**

* **throw**: Used to explicitly throw an exception within a method.
* **throws**: Declares that a method might throw one or more exceptions, making the caller responsible for handling them.

**Using throw**:

java

CopyEdit

public class Main {

public static void main(String[] args) {

try {

validateAge(15); // This will throw an exception

} catch (Exception e) {

System.out.println(e.getMessage());

}

}

public static void validateAge(int age) {

if (age < 18) {

throw new IllegalArgumentException("Age must be 18 or older.");

}

}

}

**Output**:

CopyEdit

Age must be 18 or older.

**Using throws**:

java

CopyEdit

import java.io.\*;

public class Main {

public static void main(String[] args) {

try {

readFile("example.txt"); // This may throw a checked exception

} catch (IOException e) {

System.out.println("An error occurred while reading the file.");

}

}

public static void readFile(String fileName) throws IOException {

FileReader file = new FileReader(fileName);

BufferedReader fileInput = new BufferedReader(file);

fileInput.readLine();

fileInput.close();

}

}

**4. Creating Custom Exceptions**

Custom exceptions can be created by extending either the Exception class (for checked exceptions) or RuntimeException (for unchecked exceptions).

**Example**:

java

CopyEdit

// Custom checked exception

class InvalidAgeException extends Exception {

public InvalidAgeException(String message) {

super(message);

}

}

public class Main {

public static void main(String[] args) {

try {

checkAge(15); // This will throw our custom exception

} catch (InvalidAgeException e) {

System.out.println(e.getMessage());

}

}

public static void checkAge(int age) throws InvalidAgeException {

if (age < 18) {

throw new InvalidAgeException("Age must be 18 or older to register.");

}

System.out.println("Age is valid.");

}

}

**Output**:

arduino

CopyEdit

Age must be 18 or older to register.

In this example, we created a custom exception InvalidAgeException that extends Exception (a checked exception). The checkAge method throws this custom exception if the age is less than 18.

**Key Points:**

* **Types of Exceptions**:
  + **Checked**: Must be caught or declared (e.g., IOException).
  + **Unchecked**: Do not need to be explicitly handled (e.g., NullPointerException).
* **Exception Handling**:
  + Use try, catch, and finally to handle exceptions.
  + throw is used to throw an exception, and throws is used to declare exceptions in methods.
* **Custom Exceptions**:
  + You can create your own exception classes by extending Exception (for checked exceptions) or RuntimeException (for unchecked exceptions).

4o mini