**Access Modifiers**

1. **Private**: The access level of a private modifier is only within the class. It cannot be accessed from outside the class.

private class MyClass {

private int myVar;

private void myMethod() {

// code here

}

}

2. **Default**: The access level of a default modifier is only within the package. It cannot be accessed from outside the package. If you do not specify any access level, it will be the default.

class MyClass {

int myVar;

void myMethod() {

// code here

}

}

1. **Protected**: The access level of a protected modifier is within the package and outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package.

The protected access modifier can be applied on the data member, method and constructor. It can't be applied on the class.

It provides more accessibility than the default modifier.

protected class MyClass {

protected int myVar;

protected void myMethod() {

// code here

}

}

1. **Public**: The access level of a public modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package.

public class MyClass {

public int myVar;

public void myMethod() {

// code here

}

}

1. **Exception and Errors**

|  |  |
| --- | --- |
| **Exception** | **Error** |
| * An Exception is an event that occurs during the program execution and disrupts the normal flow of the program's execution. * It Can be handled * The error indicates trouble that primarily occurs due to the scarcity of system resources * Only checked exceptions are known to the compiler. * It is mainly caused by the application itself. * There are two types of exceptions i.e. **checked** and **unchecked**. * When an error is detected, an exception is thrown * It belongs to java.lang.Exception package. * It can be classified into two categories i.e. Checked and un Checked   E.g.  invalid  Exception in thread "main" java.lang.ArithmeticException: Invalid age  at vig/exceptExamp.Throwexp1.age(Throwexp1.java:10)  at vig/exceptExamp.Throwexp1.main(Throwexp1.java:18) | * An Error is a severe situation generated when the user performs an unexpected operation * It cannot be handled Easily * The exceptions are the issues that can appear at runtime and compile time. 2. It is not possible to recover from an error * Errors will not be known to the compiler. * It is mostly caused by the environment in which the application is running. * An example of errors syntax errors, runtime errors, logical errors, Compile Time Error. * Errors will not be known to the compiler. * It belongs to java.lang.Error package. * All errors in Java are unchecked.   E.g.  Exception in thread "main" java.lang.Error: Unresolved compilation problem:  at vig/exceptExamp.Throwexp1.main(Throwexp1.java:16) |

1. **CHECKED EXCEPTION and UN CHECKED EXCEPTION**

|  |  |
| --- | --- |
| **CHECKED EXCEPTION** | **UN CHECKED EXCEPTION** |
| * Checked exceptions happen at compile time when the source code is transformed into an executable code. * Checked Exceptions include IO Exception, SQL Exception, and File not found Exception, * A checked exception must be handled either by re-throwing or with a try catch block * Java itself will know if might throw an exception and it allows user to handle it * it must specify the exception using the [throws keyword](https://www.geeksforgeeks.org/throw-throws-java/). In checked exceptions * It occurs at compile time or run time.   Syntax:  Void method\_name Throws Exception  E.g.  public void n() throws Exception | * Unchecked exceptions happen at runtime when the executable program starts running. * Unchecked Exceptions include Null Pointer Exception, Array Index Out of Bounds Exception, and Run Time Exception, String Exception * An unchecked exception is a programming error and are fatal, * It will not make compulsory for the user to handle it .if they can .they can handle it or they can leave un checked * it must specify the exception using the [throw keyword](https://www.geeksforgeeks.org/throw-throws-java/). In checked exceptions * It occurs at run time.   Syntax:  Throw new type\_exception(“Message”)  E.g.  throw new ArithmeticException("Invalid age"); |