PERTEMUAN 16
ACCESS MODIFIER

A. TUJUAN PEMBELAJARAN

1. Mahasiswa memahami apa itu *access modifier*
2. Mahasiswa dapat membedakan jenis jenis *access modifier*
3. Mahasiswa dapat mengetahui aturan membuat *access modifier*
4. Mahasiswa dapat mempraktekan penggunaan *access modifier*

B. URAIAN MATERI

Access Modifier di Java adalah menentukan aksesibilitas atau cakupan *field*, *method*, *constructor*, atau *class*. Kita dapat mengubah tingkat hak akses *field*, *method*, *constructor*, dan *class* dengan menerapkan pengubah akses di atasnya. Dengan adanya *Access Modifier* kita bisa membatasi hak akses objek, *method* , field ataupun turunannya.

1. Jenis jenis access modifier

Terdapat 4 jenis *access modifier* yaitu :

a. *Default*
b. *Private*
c. *Protected*
d. *public*

Untuk memahami cakupan *access modifier,* marilah kita lihat tabel dibawah ini :

Tabel 16. 1 cakupan access modifier

Access Modifier	**Pada *Class***	**Pada *Package***	***Subclass*** *(package yang sama)*	***Subclass*** *(package yang berbeda)*	***Diluar class***
Default	Y	Y	Y	N	N
Private	Y	N	N	N	N
Protected	Y	Y	Y	Y	N
Public	Y	Y	Y	Y	Y

Keterangan : Y = Dapat diakses , N= Tidak dapat di akses

2. Default access modifier

Ketika kita tidak menentukan *access modifier* apa pun, maka *access modifier* disebut *default*. Cakupan modifier ini terbatas pada package saja. Ini berarti bahwa jika kita memiliki *class* akses default dalam sebuah package, hanya class yang ada dalam package ini yang dapat mengakses *class* nya. Tidak ada class lain di luar package yang dapat mengakses nya. Demikian pula, jika kita memiliki method *default* atau data di *class*, tidak akan terlihat di *class* lain.

Contoh ***default access modifier*** dalam program java

Dalam contoh ini kita memiliki dua kelas, class ***Test*** mencoba untuk mengakses metode default class **Penambahan**, karena kelas ***Test*** berbeda pacakage, program ini akan menimbulkan kesalahan kompilasi, karena ruang lingkup modifier default terbatas pada package yang sama di yang dideklarasikan.

pertambahan.java

```
package package1;

public class pertambahan{

   /* Ketika tidak menentukan akses apapun
```

```
    * ini menjadi akses default.
    */
   int tambah2angka(int a, int b){
     return a+b;
   }
}
```

Test.java

```
package package2;

/* Kita akan meng import pacakage1
 * tapi akan terlihat errorkarena
 * class  mempunyai akses default
 * modifier.
 */
import package2.*;
public class Test {
   public static void main(String args[]){
      Pertambahan obj = new Pertambahan();
        /* It will throw error because we are trying
to access
         * the default method in another package
         */
      obj.tambah2angka(10, 21);
   }
}
```

Output :

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem: The method addTwoNumbers(int, int) from the type Addition is not visible at xyzpackage.Test.main(Test.java:12)
```

3. Private access modifier

Ruang lingkup modifier ***Private*** terbatas pada class saja.data dan method private hanya dapat diakses di dalam class. Class dan interface tidak dapat dideklarasikan sebagai private Jika suatu class memiliki consturctor private maka kita tidak dapat membuat objek class itu dari luar class.

Untuk memahaminya mari kita lihat contoh berikut dibawah ini :

Contoh ini memunculkan kesalahan kompilasi karena kita mencoba mengakses data private dan method class ***ABC*** di kelas ***Contoh***. Data dan *method* private hanya dapat diakses di dalam class.

Contoh *Private access modifier* dalam program java

```
class ABC{
   private double angka= 100;
   private int persegipanjang(int a){
      return a*a;
   }
}
```

```
public class Contoh{
public static void main(String args[]){
ABC obj = new ABC();
System.out.println(obj.angka);
System.out.println(obj.persegipanjang(10));
  }
}
```

Output:

```
Compile - time error
```

4. Protected access modifier

Data dan method yang di Protected hanya dapat diakses oleh class class dari pacakage yang sama dan sub class yang ada dalam pacakage apa pun. Kita juga dapat mengatakan bahwa pengubah akses yang di protected mirip dengan *access modifier* default dengan satu pengecualian bahwa ia memiliki visibilitas di sub class.

Class yang tidak dapat dinyatakan terlindungi, modifier access ini umumnya digunakan dalam hubungan *parent & child*.

Contoh 1 program Protected access modifier

Dalam contoh ini, class **Test** yang ada dalam package lain dapat memanggil method `tambah2angka` (), yang dideklarasikan *protected*. Ini karena kelas Test meng extend class **Penambahan.java** dan modifier yang di protected memungkinkan akses anggota yang di protected dalam sub class (package apa pun).

Pertambahan.java

```
package package1;
public class pertambahan{

   protected int tambah2angka (int a, int b){
      return a+b;
   }
}
```

Test.java

```
package package2;
import package1.*;
```

```
class Test extends pertambahan{
   public static void main(String args[]){
      Test obj = new Test();
      System.out.println(obj.tambah2angka(11, 22));
   }
}
```

Output :

```
33
```

Contoh 2 program *Protected access modifier*

Contoh 2 Protect.java

```
package pack;
public class Contoh2Protect{
protected void msg(){
System.out.println("Hello ini contoh protected");}
}
package mypack;
import pack.*;

class Test2 extends Contoh2Protect{
  public static void main(String args[]){
   Test2 obj = new Test2();
   obj.msg();
  }
}
```

Output :

```
Hello ini contoh protected
```

Contoh 2 private Access *modifier*

A.java

```
class A{
private A(){}//private constructor
void msg(){System.out.println("Hello java");}
}
```

Simple.java

```
public class Simple{
 public static void main(String args[]){
   A obj=new A();//Compile Time Error
 }
}
```

Catatan : class A tidak dapat menjadi private atau diprotect kecuali nested class.

5. Public access modifier

Anggota metode dan class yang dideklarasikan publik dapat diakses dari mana saja. Modifier ini tidak membatasi akses.

contoh program *modifier access* public pada java

Mari kita ambil contoh yang sama yang telah kita lihat di atas tetapi kali ini metode tambah2angka() memiliki modifier public dan class **Test** dapat mengakses method ini bahkan tanpa meng extend class Pertambahan. Ini karena modifier public memiliki visibilitas di mana-mana.

pertambahan.java

```
package package1;


public class Pertambahan{


   public int tambah2angka(int a, int b){
      return a+b;
   }
}
```

Test.java

```
     package package2;
     import  package1.*;
     class Test{
     public static void main(String args[]){
     Addition obj = new Addition();
     System.out.println(obj.addTwoNumbers(100, 1));
     }
     }
```

Output :

```
     101
```

contoh 2 program *modifier access* public pada java

public2.java

```
     package pack;
     public class public2{
     public void msg(){
```

```
     System.out.println("Hello  ini  contoh  2  public
");
     }
     }
```

```
package mypack;
import pack.*;

class Test2{
  public static void main(String args[]){
   public2 obj = new public2();
   obj.msg();
  }
}
```

Output :

```
     Hello ini contoh 2 public
```

6. Access Modifiers dengan Method Overriding

Jika kita mengganti method apa pun, method yang diganti (yaitu yang dideklarasikan dalam sub class) tidak boleh melebih batas.

contoh program *modifier access* dengan method overriding

```
class A{
     protected void msg(){
     System.out.println("Hello java");
     }
     }
```

```
public class Simple extends A{
void msg(){
System.out.println("Hello java");
}//C.T.Error
 public static void main(String args[]){
   Simple obj=new Simple();
   obj.msg();
   }
```

C. LATIHAN/TUGAS

1. Buatlah aplikasi operasi artimatika penjumlahan, pengurangan, perkalian dan pembagian dari dua bilangan A dan B, menggunakan konsep abstract data dan method akses nya adalah di tentukan *protected* Nilai parameter bilangan A = 9.5 dan parameter bilangan B = 2.5
2. Buatlah aplikasi operasi artimatika penjumlahan, pengurangan, perkalian dan pembagian dari dua bilangan A dan B, menggunakan konsep *polymorphism*, data dan method akses nya adalah di tentukan *protected* Nilai parameter bilangan A = 9.5 dan parameter bilangan B = 2.5

D. REFERENSI

Horstmann Cay S., (2011). *Big Java 4t^{h} Edition ,san jose university , united state Of America*. *RRD jefferson city publishing.*

Deitel Paul , Deitel Harvey, (2012) Java how to program eighth edition, pearson education, Boston Massachusetts , *USA, publishing as prentice hall.*

Rose Cristhoper, (2017), Java Succinctly Part 2, Morrisville, NC 27560, USA, Syncfusion, Inc.

Downey Allen B. , Mayfield Chris, (2017), Think Java, Needham, Massachusetts, USA, Green Tea Press

Hayes Helen, (2021), BeginnersBook.com, https://beginnersbook.com/java-tutorial-for-beginners-with-examples/, di akses pada tanggal 21 November 2021.

Sonoo Jaiswal , (2021) JavaTpoint offers college campus training , https://www.javatpoint.com/java-tutorial, diakses pada tanggal 1 Desember 2021