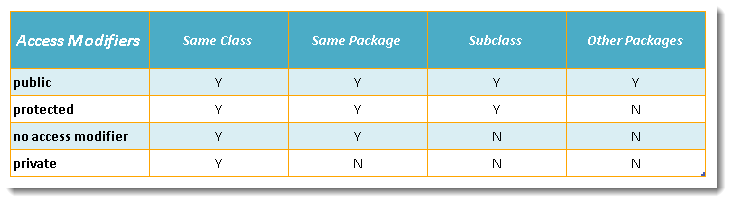
**Access Modifiers**

These are the Java keywords, which are used in declaring *variables, method, constructor or class*. These *modifiers* help to set the level of access of the data or it decides the scope of the data.

For e.g. an ***access modifier*** mentioned in the *declaration* of the class would control what information can be accessible by the other classes. So if a Class is declared as public, then the Class can be freely created or called by other Java objects. If a method is declared as private, then it can only be accessed within the class in which it is declared. In Java there are four types of Access levels - public, protected, private & default (*no access modifier*):



Y - Means Accessible

N - Means Not Accessible

**Access Modifier - default**

When *no access modifier* is given in the declaration, then it is a default ***access modifier***. Classes, members & methods have no access modifier are visible only to the same package. A compile-time error is thrown, when trying to access different packages.

***Example for a default Class***: If the class has *no declared* access modifier, it means that the class is accessible only by the classes with in the same package.

package vehicle;

class Car {

// Code goes here

}

Notice that there is ***no Access Modifier*** in the ***Car*** class declaration and this Car class is in the ***Vehicle package***. In the below code, we will try to create an object of Car class in other classes within and outside the package.

***Same Package***

package vehicle;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

}

}

***Explanation:*** *In the above class, a new Car class object is created as Toyota. The access type of Car class is default and still, it is possible to create its object only because the Car class is in the same Vehicle package.*

***Different Package***

package javaPracticeProgram;

import vehicle.Car;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car(); //Compile Time Error

}

}

***Explanation:*** *In the above class, a new Car class object is created as Toyota but a compile-time error is thrown. The access type of Car class is default, so it is not accessible in any class which is outside the Vehicle package. Notice that this TestModifiers class is in the different JavaPracticeProgram package.*

***Example for default Members***: If the field/member has no declared access *modifier,* then the fields are accessible only by the classes with in the same package.

package vehicle;

public class Car {

String sModel;

int iGear;

int iHighestSpeed;

}

Notice that there is ***no Access Modifier*** in the ***Car*** class members declaration and this Car class is in the ***Vehicle package.*** In the below code, we will try to create an object of Car class in other classes within and outside the package to access its members.

***Same Package***

Package vehicle;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

Toyota.sModel="Camry";

Toyota.iGear = 5;

Toyota.iHighestSpeed = 230;

}

}

***Explanation:*** *In the above class, a new Car class object is created as Toyota. Then assigning the values to the members of Toyota Car. It is possible only because both the Car & TestModifiers classes are in the same Vehicle package.*

***Different Package***

Package javaPracticeProgram;

import vehicle.Car;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

Toyota.sModel="Camry"; //Compile Time Error

Toyota.iGear = 5; //Compile Time Error

Toyota.iHighestSpeed = 230; //Compile Time Error

}

}

***Explanation:*** *In the above class, a new Car class object is created as Toyota. Then assigning the values to the members of Toyota Car but compile-time errors are thrown. The access types of Members of the Car class are default that is why the members/fields are not accessible in any class which is outside the Vehicle package. Notice that this TestModifiers class is in the different JavaPracticeProgram package.*

***Example for default Method***: If the method has *no declared access modifier,* then the method is accessible only by the classes with in the same package.

package vehicle;

public class Car {

public String sModel;

public int iGear;

public int iHighestSpeed;

void DisplayCharacterstics(){

System.out.println("Model of the Car: " + sModel);

System.out.println("Number of gears in the Car: " + iGear);

System.out.println("Max speed of the Car: " + iHighestSpeed);

}

}

Notice that there is ***no Access Modifier*** in the ***DisplayCharacterstics*** method declaration of the *Car* class and this Car class is in the ***Vehicle package***. In the below code, we will try to create an object of the Car class in other classes within and outside the package to use its method.

***Same Package***

package vehicle;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

Toyota.sModel="Camry";

Toyota.iGear = 5;

Toyota.iHighestSpeed = 230;

Toyota.DisplayCharacterstics();

}

}

***Explanation:*** *In the above class, a new Car class object is created as Toyota and assigned the values to the members of Toyota Car. Then call the DisplayCharacterstics method. It is possible only because both the Car & TestModifiers classes are in the same Vehicle package.*

***Different Package***

package javaPracticeProgram;

import vehicle.Car;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

Toyota.sModel="Camry";

Toyota.iGear = 5;

Toyota.iHighestSpeed = 230;

Toyota.DisplayCharacterstics(); //Compile Time Error

}

}

***Explanation:*** *In the above class, a new Car class object is created as Toyota and assigned the values to the members of Toyota Car. Then tried to call the DisplayCharacterstics method but compile-time error is thrown. The access types of the Method of the Car class is default that is why it is not accessible in any class which is outside the Vehicle package. Notice that this TestModifiers class is in the different JavaPracticeProgram package.*

**Access Modifier - public**

When the *public* keyword is mentioned in the declaration, it means least restrictive. Classes, members & methods declared as public are visible to the world, means can be accessed from any packages.

**Example for public Class**: If the class has *public access modifier*, then the class is accessible to any class and any package.

package vehicle;

public class Car {

// Code goes here

}

Notice that ***public Access Modifier*** is used in the ***Car*** class declaration and this Car class is in the ***Vehicle package.*** In the below code, we will try to create an object of the Car class in other classes within and outside the package.

***Same Package***

package vehicle;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

}

}

***Explanation:*** *In the above class, a new Car class object is created as Toyota. The access type of Car class is public, which means that it is accessible in the same package and out site the package as well.*

***Different Package***

package javaPracticeProgram;

import vehicle.Car;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

}

}

***Explanation:*** *In the above class, a new Car class object is created as Toyota in a different package. Notice that the Car class is accessible even in the class which is outside the package. It is possible because the Car class is declared as public.*

***Example for public Members***:  If the members have *public access modifier*, then the members are accessible to any class and in any package.

package vehicle;

public class Car {

public String sModel;

public int iGear;

public int iHighestSpeed;

}

Notice that ***public Access Modifier*** is used in the Members declaration of the ***Car*** class and this Car class is in the ***Vehicle package***. In the below code, we will try to create an object of the Car class in other classes within and outside the package to use its members.

***Same Package***

package vehicle;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

Toyota.sModel="Camry";

Toyota.iGear = 5;

Toyota.iHighestSpeed = 230;

}

}

***Explanation:*** *In the above class, a new Car class object is created as Toyota. The access type of Car class and its members are public, which means that it is accessible in the classes of the same package and out site the packages as well.*

***Different Package***

package javaPracticeProgram;

import vehicle.Car;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

Toyota.sModel="Camry";

Toyota.iGear = 5;

Toyota.iHighestSpeed = 230;

}

}

***Explanation***: *In the above class, a new Car class object is created as Toyota in the different package. Notice that the members of the Car class are accessible even in the class which is outside the package. It is possible because the Car class members are declared as public.*

***Example for a public Method***: If the method has *public access modifier*, then the method is accessible to any class and any package.

package vehicle;

public class Car {

public String sModel;

public int iGear;

public int iHighestSpeed;

public void DisplayCharacterstics(){

System.out.println("Model of the Car: " + sModel);

System.out.println("Number of gears in the Car: " + iGear);

System.out.println("Max speed of the Car: " + iHighestSpeed);

}

}

Notice that ***public Access Modifier*** is used in the ***DisplayCharacterstics*** Method declaration of the *Car* class and this Car class is in the ***Vehicle package***. In the below code, we will try to create an object of the Car class in other classes within and outside the package to use its method.

***Same Package***

package vehicle;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

Toyota.sModel="Camry";

Toyota.iGear = 5;

Toyota.iHighestSpeed = 230;

Toyota.DisplayCharacterstics();

}

}

***Explanation:*** *In the above class, a new Car class object is created as Toyota and assigned values to its members. Then call the DisplayCharacterstics method. The access type of DisplayCharacterstics is public, which means that it is accessible in the classes of the same package and out site the packages as well.*

***Different Package***

package javaPracticeProgram;

import vehicle.Car;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

Toyota.sModel="Camry";

Toyota.iGear = 5;

Toyota.iHighestSpeed = 230;

Toyota.DisplayCharacterstics();

}

}

***Explanation***: *In the above class, a new Car class object is created as Toyota in the different package. Notice that the displayCharacterstics method of the Car class is accessible even in the class which is outside the package. It is possible because the Car class members are declared as public.*

**Access Modifier - private**

\When the *private* keyword is mentioned in the declaration, it means highly restrictive. Methods & members declared as private are visible only with in the same class. This doesn't really make so much sense for classes though. Therefore, the access modifier private is mostly used for fields, constructors and methods. A compile-time error is thrown, when trying to access in a different class even with in the same package.

***Example of private Class***: A Class cannot have the *private access modifier.* In Java programming private keyword cannot be used with classes and interfaces.

package vehicle;

Private class Car { // Compile Time Error

// Code goes here

}

***Explanation :*** *Notice that* ***private Access Modifier*** *is used in the declaration of the* ***Car*** *class but Java compile-time error is thrown.*

***Example of private Members***: If the members have the private access modifier, then the members are only accessible within the class.

package vehicle;

public class Car {

private String sModel;

private int iGear;

private int iHighestSpeed;

public void DisplayCharacterstics(){

System.out.println("Model of the Car: " + sModel);

System.out.println("Number of gears in the Car: " + iGear);

System.out.println("Max speed of the Car: " + iHighestSpeed);

}

}

***Explanation***: *Notice that****private Access****Modifier is used in the****Members declaration****of the Car class. As I said above that private members are only accessible with in the same class, if you see the DisplayCharacterstics method is accessing all the private members.*

***Same Package***

package vehicle;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

Toyota.sModel="Camry"; //Compile Time Error

Toyota.iGear = 5; //Compile Time Error

Toyota.iHighestSpeed = 230; //Compile Time Error

}

}

***Explanation :*** *As the Car class is declared as public, a new Car class object is created as Toyota. When trying to access the members of the Car class, java compile-time error is thrown. This is because the members of the Car class are declared as private and they are not accessible outside the original class even if the class is in the same package.*

***Example of private Method***: If the methods have the ***private access modifier***, then the method is only accessible within the class.

package vehicle;

public class Car {

public String sModel;

public int iGear;

public int iHighestSpeed;

private void DisplayCharacterstics(){

System.out.println("Model of the Car: " + sModel);

System.out.println("Number of gears in the Car: " + iGear);

System.out.println("Max speed of the Car: " + iHighestSpeed);

}

public void Print(){

DisplayCharacterstics();

}

}

***Explanation :*** *Notice that private Access Modifier is used in the Method declaration of the Car class. As I said above that private method is only accessible with in the same class. If you see, the Print method is accessing DisplayCharacterstics method.*

***Same Package***

package vehicle;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

Toyota.sModel="Camry";

Toyota.iGear = 5;

Toyota.iHighestSpeed = 230;

Toyota.DisplayCharacterstics(); //Compile Time Error

Toyota.Print();

}

}

***Explanation***: *As the Car class is declared as public, a new Car class object is created as Toyota. When trying to access the DisplayCharacterstics method of the Car class, java compile-time error is thrown. This is because the method of the Car class is declared as private and they are not accessible outside the original class even if the class is in the same package. But notice that Print method is accessible, as it is declared as public in car class.*

**Access Modifier - protected**

The ***protected access modifier*** is accessible within the package and outside the package but through ***inheritance*** only. The protected access modifier can be applied on the data members, methods and constructors. It can't be applied on the classes and interfaces.

***Example of protected Class***: A Class cannot have the *protected access modifier.* In Java programming protected keyword cannot be used with classes and interfaces.

package vehicle;

Private class Car { // Compile Time Error

// Code goes here

}

***Explanation***: *Notice that protected Access Modifier is used in the declaration of the Car class but Java compile-time error is thrown.*

***Example of protected Members***: If the members have the *protected access modifier,* then the members are only accessible when the original class is extended.

package vehicle;

public class Car {

protected String sModel;

protected int iGear;

public int iHighestSpeed;

}

***Explanation***: *Notice that protected Access Modifier is used in the Members declaration of the Car class. As I said above that protected members are only accessible through Inheritance. In the below code, we will try to create an object of the Car class with extending and non-extending of Car class to check the accessibility of its members.*

***Without Extending***

package javaPracticeProgram;

import vehicle.Car;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

Toyota.sModel="Camry";//Compile Time Error

Toyota.iGear = 5;//Compile Time Error

Toyota.iHighestSpeed = 230;

}

}

***Explanation:*** *In the above class, a new Car class object is created as Toyota. The access types of the members of the Car class are protected, which means that members are not accessible without extending the Car class. A compile-time error is thrown, when trying to access the protected member of the Car class.*

*Notice that iHighestSpeed variable is accessible and not thrown any compile-time error, it is because the variable is declared as public in the Car class.*

***With Extending***

package javaPracticeProgram;

import vehicle.Car;

class SmartCar extends Car {

public static void main(String[] args) {

SmartCar Toyota = new SmartCar();

Toyota.sModel="Camry";

Toyota.iGear = 5;

Toyota.iHighestSpeed = 230;

}

}

***Note:*** *For the sake of this example ignore extends keyword, as we would be explaining this in detail in the later chapters. But on high level, we extend class when we want to inherit the functionalities of the parent class.*

***Explanation:*** *In the above class, a SmartCar extends Car class. Now to access the protected members of the car class, it is required to create an object of SmartCar. Toyota is the object of SmartCar and it is able to access all the protected members of the Car class as well as the public members also like iHighestSpeed.*

***Example of protected Method***: If the method has the protected access modifier, then the method is only accessible when it's original class is extended.

package vehicle;

public class Car {

public String sModel;

public int iGear;

public int iHighestSpeed;

protected void DisplayCharacterstics(){

System.out.println("Model of the Car: " + sModel);

System.out.println("Number of gears in the Car: " + iGear);

System.out.println("Max speed of the Car: " + iHighestSpeed);

}

}

***Explanation***: *Notice that protected Access Modifier is used in the Method declaration of the Car class. As I said above that protected members are only accessible through Inheritance, In the below code, we will try to create an object of the Car class with extending and non-extending to check the accessibility of its members.*

***Without Extending***

package javaPracticeProgram;

import vehicle.Car;

public class TestModifiers {

public static void main(String[] args) {

Car Toyota = new Car();

Toyota.sModel="Camry";

Toyota.iGear = 5;

Toyota.iHighestSpeed = 230;

Toyota.DisplayCharacterstics(); // Compile time error

}

}

***Explanation:*** *In the above class, a new Car class object is created as Toyota. The access type of the method of the Car class is protected, which means that the method is not accessible without extending the Car class. A compile-time error is thrown, when trying to access the protected method of the Car class.*

***With Extending***

package javaPracticeProgram;

import vehicle.Car;

class SmartCar extends Car {

public static void main(String[] args) {

SmartCar Toyota = new SmartCar();

Toyota.sModel="Camry";

Toyota.iGear = 5;

Toyota.iHighestSpeed = 230;

Toyota.DisplayCharacterstics();

}

}

***Explanation:*** *In the above class, a SmartCar extends Car class. Now to access the protected method of the car class, it is required to create an object of SmartCar. Toyota is the object of SmartCar and it is able to access all the protected method of the Car class.*