Tutorial Link https://course.testpad.chitkara.edu.in/tutorials/Java : Access Protection with Package/5b1e49e17becc0459deadd99

TUTORIAL

Java : Access Protection with Package

In general, fields in a class are best tagged as private, and methods are usually tagged as public. Any features declared private won't be visible to other classes. a subclass cannot access the private fields of its superclass. There are times, however, when you want to restrict a method to subclasses only or, less commonly, to allow subclass methods to access a superclass field. In that case, you declare a class feature as protected. For example, if the superclass Employee declares the hireDay field as protected instead of private, then the Manager methods can access it directly. However, the Manager class methods can peek inside the hireDay field of Manager objects only, not of other Employee objects. This restriction is made so that you can't abuse the protected mechanism and form subclasses just to gain access to the protected fields.

We already know that access to a private member of a class is granted only to other members of that class. Packages add another dimension to access control. Classes and packages are both means of encapsulating and containing the name space and scope of variables and methods. Packages act as containers for classes and other subordinate packages. Classes act as containers for data and code. The class is Java's smallest unit of abstraction. Because of the interplay between classes and packages, Java addresses four categories of visibility for class members:

- Subclasses in the same package
- Non-subclasses in the same package
- Subclasses in different packages
- Classes that are neither in the same package nor subclasses

The three access specifiers, private, public, and protected, provide a variety of ways to produce the many levels of access required by

these categories. Table below sums up the interactions.

Class member	Private	Public	Protected	No Modifier(Default)
Same class	Yes	Yes	Yes	Yes
Same **package** Subclass	No	Yes	Yes	Yes
Same **package** non-subclass	No	Yes	Yes	Yes
Different **package** subclass	No	Yes	Yes	No
Different Package non-subclass	No	Yes	No	No

While Java's access control mechanism may seem complicated, we can simplify it as follows. Anything declared public can be accessed from anywhere. Anything declared private cannot be seen outside of its class. When a member does not have an explicit access specification, it is visible to subclasses as well as to other classes in the same package. This is the default access. If you want to allow an element to be seen outside your current package, but only to classes that subclass your class directly, then declare that element protected.