2.2 Access Modifiers

Java provides a number of access modifiers to set access levels for classes, variables, methods and constructors.

1) Default Access Modifier - No keyword:

- A variable or method declared without any access control modifier is available to any other class in the same package.
- Visible to all class in its package.
- **Examples:** Variables and methods can be declared without any modifiers,as in the following Examples:

```
String version = "1.5.1"; boolean processOrder()
{
return true;
}
```

2) Private Access Modifier - private:

- Methods, Variables and Constructors that are declared private can only be accessed within the declared class itself.
- Private access modifier is the most restrictive access level.
- Class and interfaces cannot be private.
- Using the private modifier is the main way that an object encapsulates itself and hide data from the outside world.

Examples:
```
private String format;
private void get() { }
```

3) Public Access Modifier - public:

- A class, method, constructor, interface etc declared public can be accessed from any other class.
- Therefore fields, methods, blocks declared inside a public class can be accessed from any class belonging to the Java Universe.
- However if the public class we are trying to access is in a different package, then the public class still need to be imported.
- Because of class inheritance, all public methods and variables of a class are inherited by its subclasses.

Examples:
```
public double pi = 3.14;
{
 public static void main(String[] arguments)
  // ...
 }
```

4) Protected Access Modifier - protected:

- Variables, methods and constructors which are declared protected in a super class can be accessed only by the subclasses in other package or any class within the package of the protected members' class.
- The protected access modifier cannot be applied to class and interfaces.
- Methods, fields can be declared protected, however methods and fields in a interface cannot be declared protected.
- Protected access gives the subclass a chance to use the helper method or variable, while preventing a nonrelated class from trying to use it.

- **Example:** The following parent class uses protected access control, to allow its child class override.

```
protected void show( )
{ }
```

5) private protected:

- Variables, methods which are declared protected in a super class can be accessed only by the subclasses in same package.
- It means visible to class and its subclasses.

Example:

```
private protected void show( )
{ }
```