The Four Access Levels

private, package-private, protected, and public are four widening rings. The habit worth building is starting at private and widening only when something needs the access.

Automation Foundations · ashishautomationlabs.store

Chapter 7 gave you `private` and `public`. Chapter 11 added `protected`. There is a fourth, and it is the one you get by writing nothing at all.

Read it as four widening rings. `private` reaches only the same class. Writing no modifier at all -- **package-private** -- reaches the same class and the same package. `protected` reaches the same class, the same package, _and_ a subclass anywhere. `public` reaches all of that, plus everywhere else in the program. `private` is the tightest, `public` the loosest, and the two in the middle are the ones most people never learn properly.

```
class BasePage {
    public String title = "ShopCart";
    protected String driverName = "chrome";
    String packagePrivateNote = "visible inside this package only";
    private String secret = "session-token-8891";
}
```

```
public    : ShopCart
protected : chrome
default   : visible inside this package only
private   : session-token-8891  (reachable here, inside the class)
```

All four are reachable from inside the class itself -- that is what the output shows. The differences only appear from outside, and they are exactly the four widening rings above.

The habit worth building is to start at `private` and widen only when something needs the access. Most fields should be `private`. A base class shares with its children through `protected`. Package-private is useful for helper classes that exist to serve one package and should not become part of your framework's public surface. And `public` is a promise: once other code depends on it, changing it breaks them. A suite where everything is public has not made a decision. It has skipped one.

> *Which modifier do you get by writing nothing? Can a subclass in a different package read a* `protected` *field? Which is the right default when you are unsure?*

>

> *Answer -- package-private, visible inside the same package only. Yes -- that is exactly what* `protected` *adds over package-private. And* `private` *-- start tight and widen only when something genuinely needs the access.*

Access levels decide who can _reach_ a value. The next problem is a different one entirely: a value that's reachable just fine, but typed out by hand in nine different files.