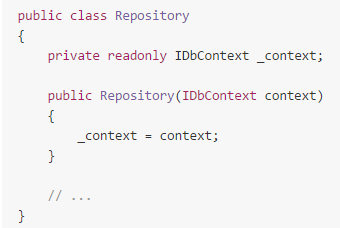
Dependency injection => giving an object its instance variables.

Constructor injection, setter injection, or property injection. Does not have to modify code. OCP. Just adds new implementation and sets it.

In DI, need to use interfaces.. making the code more testable.. we supply mocks etc.



IF CLASS BREAKS WITHOUT THE DEP.. this makes sure the class isn‘t created without.

And since constructed only once.. WILL ALWAYS HAVE SAME DEPENDENCY.

If need to extend the class and override the constructor => trouble.

Setter Injection:



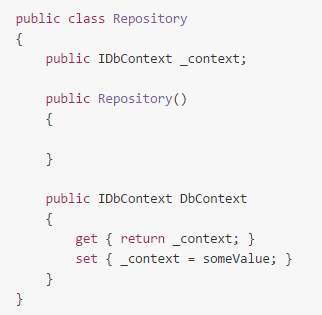
Advantages:

1. If the dependency is not required for the class to work.. much easier to use the setter.. change any time.. dynamic..
2. If you do not need the dependency you do not have to call the setter.

Disadvantages:

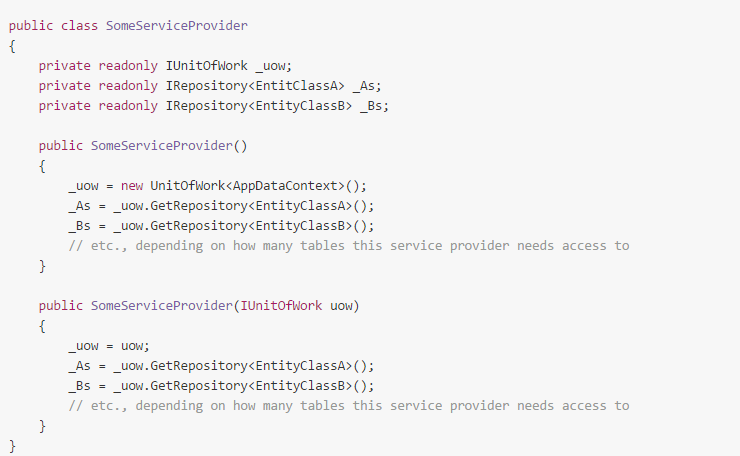
1. Don‘t have to call the setter, you‘ll have to check if the dep is set or not..
2. Setter can be called at any time.. can not be sure the dependency is the same as you intend it ot be at any given time..

Property Injection:



No good way to check whether is legal..

POOR MANS DI, UoW:



1. PASS DEPENDENCY (uow) or b CONSTRUCT A DEFAULT ONE...

Either instiate it with a reference to some IunitOfWork implementation.. or the parameterless.. where a concrete UnitOfWOrk class is used..

**CAN WORK IN Web API (REGULAR CODE) or in a unit test project.. where we MOCK THE DATA SOURCE..**