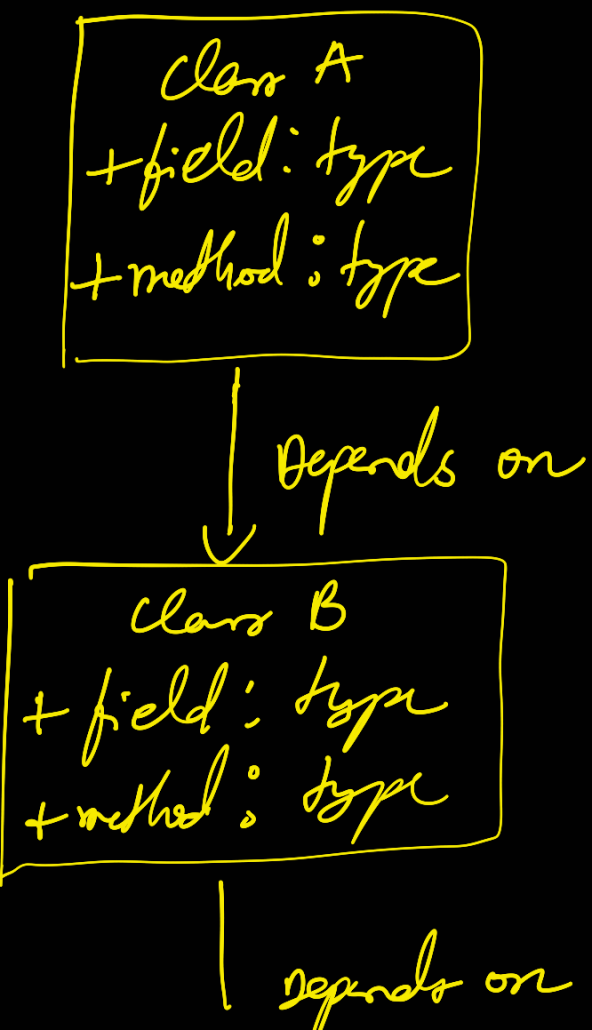
5. Dependency Inversion

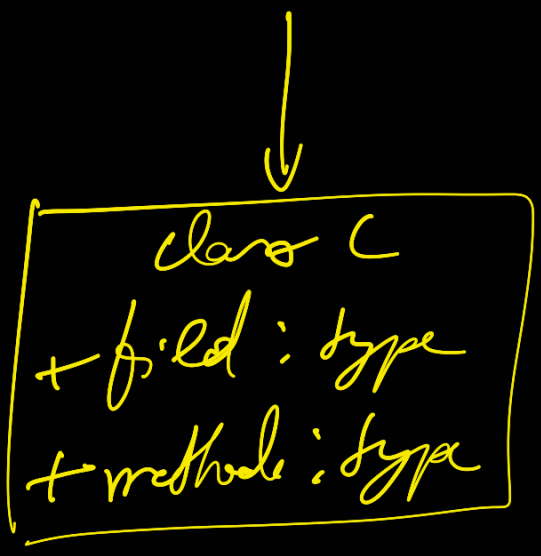
⇒ "High level module or low level modules in your code should not depend on the actual implementation.

They should depend on abstraction

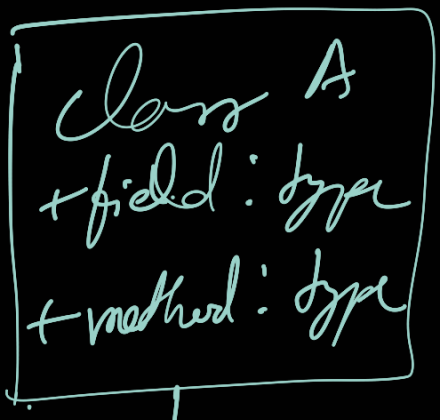
⇓

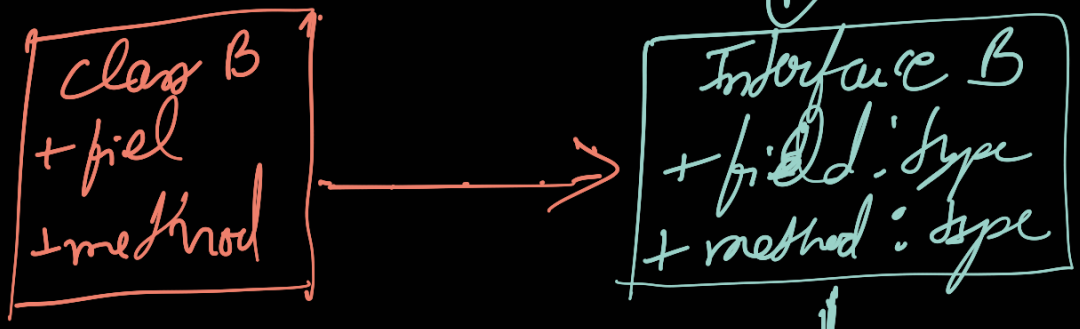
interfaces

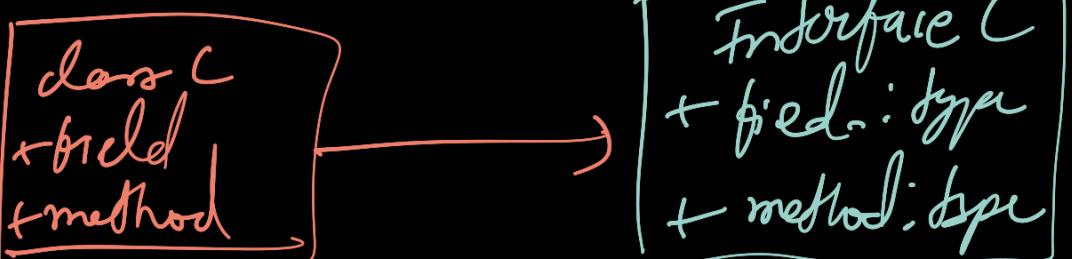




* Any change in function of class C will affect class B which intern affect class A.







* Any changes in methods of class B and C won't affect, as the interface remains same.
* This is good for writing test cases.

* All these SOLID principles ensure the code is Robust, modular and allows big team of developers to make change Independently