**SOLID**

**S**

**Single responsibility principle** = every class should have a single responsibility.

**O**

**Open closed principle** = your class should be open for extension, but closed for modification

= you should be able to extend a classes behavior, without modifying it

= use abstract base classes

= use private variables with getters and setter – ONLY when you need them

**L**

**Liskov substitution principle** = objects in a program would be replaceable with instances of their subtypes WITHOUT altering the correctness of the program

= a square “is a” rectangle, but a rectangle “is not” a square.

= while you extend your classes you are not changing the behavior

**I**

**Interface segregation principle** = many client-specific interfaces are better than one general-purpose interface

= keep your components focused and minimize dependencies between them

**D**

**Dependency inversion principle** = abstraction should not depend upon details