Solid Principles Cheatsheet

Single Responsibility

- "A class should have only one reason to change".
- A class or module should have responsibility over a single part of functionality provided by the software
- That responsibility should be entirely encapsulated by the class

Open/Closed

- "software entities (classes, modules, functions, etc.) should be open for extension, but closed for modification"
- An entity can allow its behaviour to be extended without modifying its source code

Liskov substitution

- Supposing object S is a subtype of object T, then objects of type T may be replaced with objects of type S without altering any of the desirable properties of T.
- A particular definition of a subtyping relation, called (strong) behavioral subtyping

Interface segregation

- No client should be forced to depend on methods it does not use.
- ISP splits interfaces that are very large into smaller and more specific ones so that clients will only have to know about the methods that are of interest to them.

Dependency inversion principle

- High-level modules should not depend on low-level modules. Both should depend on abstractions.
- Abstractions should not depend on details. Details should depend on abstractions.

Solid Principles Cheatsheet

More Information at http://egkatzioura.com