# Design patterns and principles

**Design patterns** are projectual solutions to common problems.   
Example of patterns are:   
- Singleton   
- Factory method   
- Abstract factory   
- Builder   
- MVC   
  
If you like to be a good programmer you have to know at least the GOF (Gang of Four) patterns (read the book Design Patterns: Elements of Reusable Object-Oriented Software - written by Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides).

**Design principles** are ideas that a programmer must follow to develop a good software.   
  
Example of design principles are:   
- Separation of concerns   
- Single Responsibility principle   
- Principle of Least Knowledge   
- Don’t repeat yourself (DRY)   
- Minimize upfront design   
  
You have to follow all them (so known all them) to be a good programmer.   
Note that following the design principles brings you to apply design patterns in many situations: as an example MVC is relative to separation of concerns.

* program to an interface, not an implementation
* loose coupling
* favor object composition over class inheritance ; delegate to member
* use parameterized types (generics) (>< raw types)
* KISS principle (Keep It Simple, Stupid)
* design patterns = logical ; framework = physical
* Bekijk eens: **Core design principles for software developers** (~2u30): <https://www.youtube.com/watch?v=llGgO74uXMI>