

# Design Pattern & Clean Code

**What are Dependencies?**

**Thomas Ley | @CleanCodeCoach**

# Indirection

"All problems in computer science can be solved by another level of indirection"

Butler Lampson

in *"fundamental theorem of software engineering"*

# Dependencies

- Class (internal, hard)
- Interface (internal, soft)
- Libraries (external, hard/soft).

# Class

## 👉 New is Glue

- "Hard dependency"
- Knows implementation
- Cannot be replaced
- `new ()` as indicator.

# Interface

- "Soft dependency"
- Hides implementation
- Can be replaced
- Subset of functionality [ISP].

# Libraries

- "Hard dependency" or "Soft dependency"
- Knows or hides implementation
- Can partially be replaced
- Depending from 3rd party.

# Independence is Freedom

- Reduce (or remove) hard dependencies
- Use abstractions (e.g. Interfaces, Lambda)
- Watch your dependencies
- Define your dependencies.

# Dependencies can be found

- nuget/references
- namespace usings
- constructor
- `new ()`
- properties.



# Favour Composition

- Favour composition over inheritance
- Legacy Clean Code principle
- Multi-Inheritance (C++)
- Avoid class-inheritance to add functions.