**Interface and Composition Design**

Composition goes beyond the mechanics of type embedding and is more than just a paradigm. It is the key for maintaining stability in your software by having the ability to adapt to the data and transformation changes that are coming.

**Notes**

* This is much more than the mechanics of type embedding.
* Declare types and implement workflows with composition in mind.
* Understand the problem you are trying to solve first. This means understanding the data.
* The goal is to reduce and minimize cascading changes across your software.
* Interfaces provide the highest form of composition.
* Don't group types by a common DNA but by a common behavior.
* Everyone can work together when we focus on what we do and not what we are.

[Methods, Interfaces and Embedding](https://www.ardanlabs.com/blog/2014/05/methods-interfaces-and-embedded-types.html)   
[Composition In Go](https://www.ardanlabs.com/blog/2015/09/composition-with-go.html)  
[Reducing Type Hierarchies](https://www.ardanlabs.com/blog/2016/10/reducing-type-hierarchies.html)  
[Avoid Interface Pollution](https://www.ardanlabs.com/blog/2016/10/avoid-interface-pollution.html)