**Decorator vs. Proxy**

Although both share a similar UML, but the lave a completely different intent and side effects.

. Decorator can be seen as a pipeline which affects the **output**. The output can be both direct or indirect (like log). Like adding new functionality to calculate tax.

. Proxy **never changes the output** and its concern is only related to object access like, access control, caching or lazy loading or adding circuit breaker. Generally, it doesn’t contain business logic.

**Strategy vs. Bridge**

Both use Behavior encapsulation technique (composing over inheritance).

Behavior encapsulation -> separate behaviors into another hierarchy not in the target class.

Whenever we intend to remove/decrease the hierarchy -> **Strategy**

Whenever we already have hierarchy but we intent to prevent Decartes mutation -> **Bridge**

Instead of making more subtype from the target type, we instead try to encapsulate the new behaviors in other hierarchy and then inject it the target class. In other words, we remove the hierarchy from the target class and instead we extract the behaviors into a separate hierarchy.