Observer

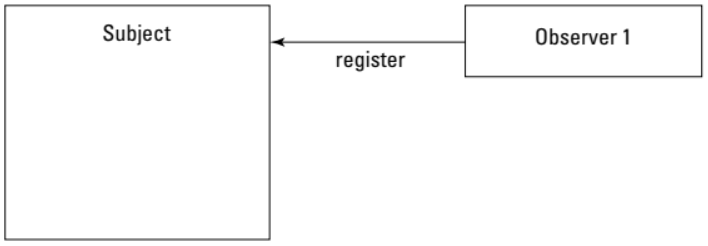
## Intent

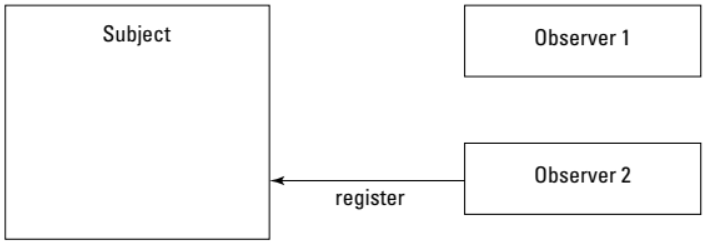
* Define a one-to-many dependency between objects so that when one object changes state, all its dependents are notified and updated automatically.
* Encapsulate the core (or common or engine) components in a Subject abstraction, and the variable (or optional or user interface) components in an Observer hierarchy.
* The "View" part of Model-View-Controller.

## Mechanism Of Action

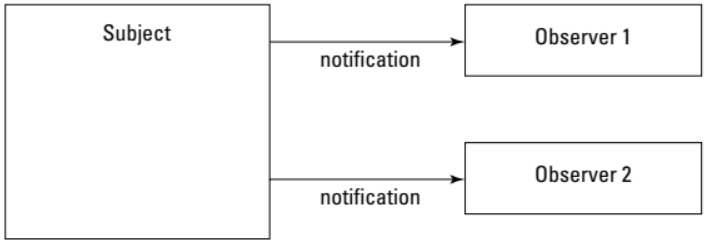
Subject affect the all of registed object.

* Step 1:



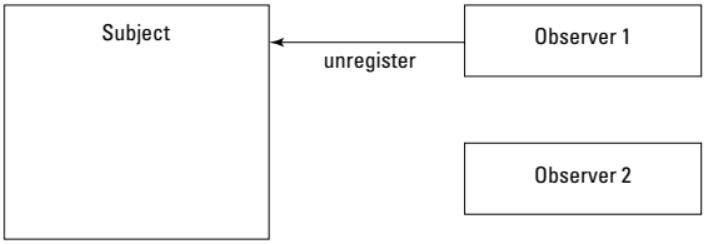


* Step 2:

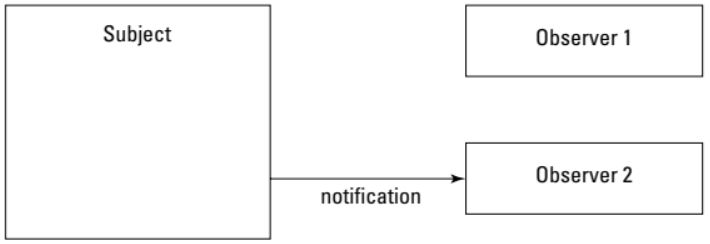


Subject does **not** affect **unregisted** object.

* Step 1:

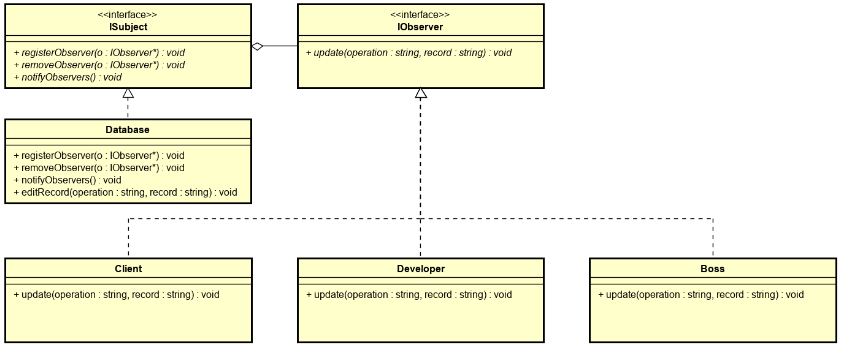


* Step 2:



## Class diagram

Example 1:



## Practical Usage Condition

* Have the **LIST** of classes.
* Each class can register/ unregister sevice with Subject.
* Class still exist when it unregister with Subject.