#JavaScript Design pattern

* **Module** helps to create private methods; just accessible by module.

->The Revealing Module pattern came about as Heilmann was frustrated with the fact that he had to repeat the name of the main object when we wanted to call one public method from another or access public variables. He also disliked the Module pattern’s requirement for having to switch to object literal notation for the things he wished to make public.

->we would simply define all of our functions and variables in the private scope and return an anonymous object with pointers to the private functionality we wished to reveal as public.

* The **Singleton pattern** is thus known because it restricts instantiation of a class to a single object.

Singletons differ from static classes (or objects) as we can delay their initialization, generally because they require some information that may not be available during initialization time.

* The **Observer** is a design pattern where an object (known as a subject) maintains a list of objects depending on it (observers), automatically notifying them of any changes to state.

Observer pattern is where we need to maintain consistency between related objects without making classes tightly coupled.

Dynamic relationships can exist between observers and subjects when using either pattern

* **The Mediator Pattern:** The dictionary refers to a mediator as a neutral party that assists in negotiations and conflict resolution. In our world, a mediator is a behavioral design pattern that allows us to expose a unified interface through which the different parts of a system may communicate.

The Mediator promotes loose coupling by ensuring that instead of components referring to each other explicitly, their interaction is handled through this central point.

This can help us decouple systems and improve the potential for component reusability.

A real-world analogy could be a typical airport traffic control system. A tower (Mediator) handles what planes can take off and land because all communications (notifications being listened out for or broadcast) are done from the planes to the control tower, rather than from plane-to-plane. A centralized controller is key to the success of this system and that's really the role a Mediator plays in software design.