Prototype Design Pattern

Prototype allows us to hide the complexity of making new instances from the client. The concept is to copy an existing object rather than creating a new instance from scratch, something that may include costly operations. The existing object acts as a prototype and contains the state of the object. The newly copied object may change same properties only if required. This approach saves costly resources and time, especially when the object creation is a heavy process.

The prototype pattern is a creational design pattern. Prototype patterns is required, when object creation is time consuming, and costly operation, so we create object with existing object itself. One of the best available way to create object from existing objects are **clone() method**. Clone is the simplest approach to implement prototype pattern. However, it is your call to decide how to copy existing object based on your business model.

Prototype Design Participants

1) **Prototype** : This is the prototype of actual object.

2) **Prototype registry** : This is used as registry service to have all prototypes accessible using simple string parameters.

3) **Client** : Client will be responsible for using registry service to access prototype instances.

When to use the Prototype Design Pattern

When a system should be independent of how its products are created, composed, and represented and
When the classes to instantiate are specified at run-time.
For example,
1) By dynamic loading or To avoid building a class hierarchy of factories that parallels the class hierarchy of products or