**Design Pattern**

* Design Patterns are nothing but documented and tested solutions for recurring problems in a given context.
* They are reusable solutions to the problems that as a developer we encounter in our day to day programming.
* One thing you need to remember is that the design patterns are for projects and projects are not for patterns.
* Many developers are enforcing the design pattern into their project which makes the project messy. So, use only when required.

**Types**

1. [**Creational Design Pattern**](https://dotnettutorials.net/lesson/creational-design-pattern/) (object creation and initialization)
2. [**Structural Design Pattern**](https://dotnettutorials.net/lesson/structural-design-pattern/) (Structural Changes of class and interfaces, and the relationship between classes)
3. [**Behavioural Design Pattern**](https://dotnettutorials.net/lesson/behavioral-design-pattern/) (Communication Between Objects)
4. **Creational Design Pattern**

* Creational Design Patterns are concerned with the way in which objects are created.
* They reduce complexities and instability by creating objects in a controlled manner.
* The Creational Design Pattern helps us to centralize the object creation logic and depending upon the condition, it will create, initialize, and returns the appropriate object to the client.
* There are 6 Creational Design Patterns –

1. Singleton DP
2. Factory DP
3. Abstract Factory DP
4. Fluent Interface DP
5. Builder DP
6. Prototype DP