**Design Patterns**

Design patterns are solutions for most commonly(frequently) occurred problems while designing a software.

These patterns are mostly evolved rather than discovered.

None of these patterns force us anything in respect to implementation, they are just guidelines to solve a problem – in a particular way, – in a particular  
contexts.

Being so important, lets start learning these design patterns…!!!

We have total 3 kind of design patterns which are also subdivided.

1. Creational Design Patterns.
2. Structural Design Patterns.
3. Behavioral Design Patterns.

* **Creational Design Patterns**

Creational patterns often used in place of direct instantiation with *constructors.*

They make the creation process more *adaptable* and *dynamic*.

In Particular, they can provide a great deal of *flexibility* about which objects are created, how those objects are created & how they are initialized.

Creational Design Patterns are subdivided into below patterns:-

1. Builder
2. Prototype
3. Factory
4. Abstract Factory
5. Singleton

**#** Builder

Builder design pattern is an alternative way to construct complex objects and  
should be used only when we want to build different types of immutable  
objects using same object building process.

**#** Prototype

**#** Factory

**#** Abstract Factory

**#** Singleton

* **Structural Design Patterns**
* **Behavioral Design Patterns**