**Design Patterns: (evolved from object-oriented programming, implementing**

Design patterns are evolved as reusable solutions to the problems that we encounter every day of programming.

**Types of Design Patterns**

**Creational:**

Deals with object creation and initialization, this pattern gives the program more flexibility in deciding which objects need to be created for given case.eg. Singleton, Factory, Abstract Factory etc.

**Structural:**

Deals with class and object composition, this pattern focuses on decoupling interface and implementing of classes and its objects.eg. Adapter, Bridge, etc.

**Behavioral:**

Deals with the communication between Classes and objects.eg. Chain of responsibility, Command, Interpreter, etc.

**Singleton Design Pattern:**

Used when we need to ensure that one object of a particular class need to be created. All further references to the objects are referred to the same underlying instance created.

Advantages of Singleton:

* Singleton controls concurrent access to resource.
* It ensures there is only one object available across the application in a controlled state.

Implementation Guidelines:

* Ensure that only one instance of the class exits
* Provide global access to that instance by
  + Declaring all constructors of the class to be private.
  + Providing static method that return a reference to the instance.
  + The instance is stored as a private static variable.