Abstract Factory Pattern

Abstract Factory Pattern says that just **define an interface or abstract class for creating families of related (or dependent) objects but without specifying their concrete sub-classes.**That means Abstract Factory lets a class returns a factory of classes. So, this is the reason that Abstract Factory Pattern is one level higher than the Factory Pattern.

An Abstract Factory Pattern is also known as **Kit.**

Advantage of Abstract Factory Pattern

* Abstract Factory Pattern isolates the client code from concrete (implementation) classes.
* It eases the exchanging of object families.
* It promotes consistency among objects.

Usage of Abstract Factory Pattern

* When the system needs to be independent of how its object are created, composed, and represented.
* When the family of related objects has to be used together, then this constraint needs to be enforced.
* When you want to provide a library of objects that does not show implementations and only reveals interfaces.
* When the system needs to be configured with one of a multiple family of objects.

UML for Abstract Factory Pattern

* We are going to create a **Bank interface** and a **Loan abstract class** as well as their sub-classes.
* Then we will create **AbstractFactory** class as next step.
* Then after we will create concrete classes, **BankFactory,** and **LoanFactory** that will extends **AbstractFactory class**
* After that, **AbstractFactoryPatternExample** class uses the **FactoryCreator** to get an object of **AbstractFactory** class.
* See the diagram carefully which is given below:

