# Design Pattern Java

This document focuses on the various important design patterns of Java. Below are the examples,

1. **Factory Pattern**

A Factory Pattern or Factory Method Pattern says that just define an interface or abstract class for creating an object but let the subclasses decide which class to instantiate. In other words, subclasses are responsible to create the instance of the class.

The Factory Method Pattern is also known as Virtual Constructor.

1. **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. It is also known as **Factory of factories**.

1. Strategy Pattern

Strategy pattern is a behavioural desi.gn pattern to choose strategy or algorithm at run time

Concepts / Important points about strategy pattern,

* Eliminate conditional statements
* Behaviour encapsulated in the classes
* client is typically aware of strategies
* client chooses strategy
* Example - Java.util.Comparator

Design of Strategy Pattern,

* Interface or Abstract base class
* Concrete class per strategy
* Removes if/else conditionals
* Strategies are independent

In UML diagram of Strategy pattern we will find Context,Stratgey and Concrete Strategy