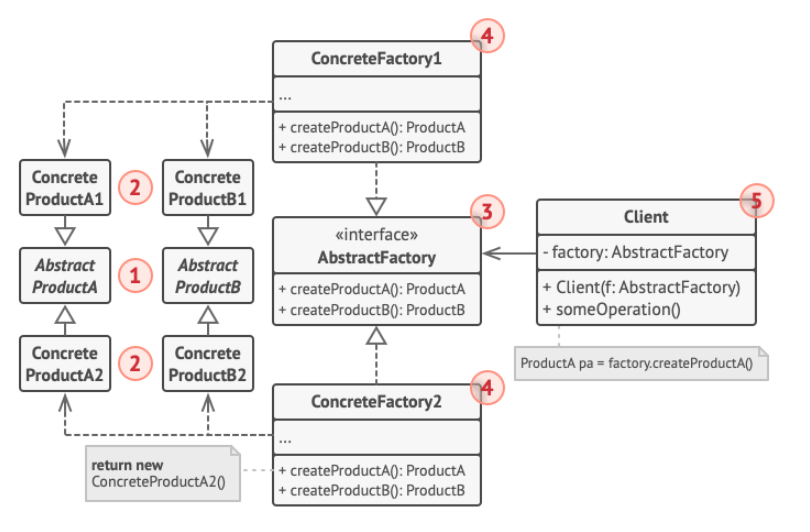
Abstract Factory

Creational Design Pattern

**What is an Abstract Factory:**

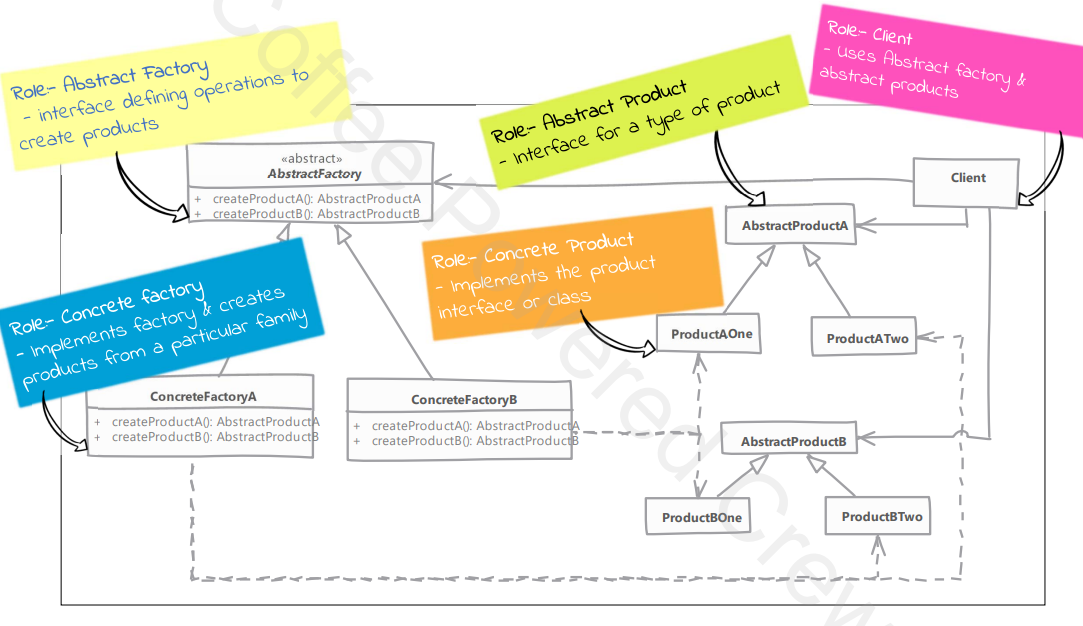
1. Abstract factory is used when we have two or more objects which work together forming a kit or set and there can be multiple sets or kits that can be created by client code.
2. Separate client code from concrete objects forming such a set and also from the code which creates these sets.
3. Abstract Factory can be a Singleton.

**Structure:**



1. Abstract Products declare interfaces for a set of distinct but related products which make up a product family. Example: LandUnit and NavalUnit abstract classes.
2. Concrete Products are various implementations of abstract products, grouped by variants. Each abstract product(chair/sofa) must be implemented in all given variants (Victorian/Modern).
3. The abstract Factory interface declares a set of methods for creating each of the abstract products.
4. Concrete Factories implement creation methods of the abstract factory. Each concrete factory corresponds to a specific variant of products and creates only those product variants.

**UML:**



1. Sets are ProductAOne & ProductBOne and ProductATwo & ProductBTwo

**Difference between Abstract Factory and Factory Method:**

1. Abstract Factory hides factories as well as concrete objects used from the client code.
2. Abstract Factory is suitable when multiple objects are designed to work together & clients must use products from a single family at a time.
3. Factory method hides the concrete objects which are used from the client code.
4. Factory method is concerned with one product & its subclasses. Collaboration of the product itself with other objects is irrelevant.

**Implementation:**

1. Create interfaces for the sets
2. Create concrete classes implementing them (Circle, Square, etc.).

Example: Storage interface, S3Storage & GoogleCloudStorage implementing it & Instance interface, Ec2Instance & Google…Instance implementing it.

1. Create an interface to get factories for the different sets. (AWS & Google)
2. Create Factory classes implementing the interface to generate object of concrete class.