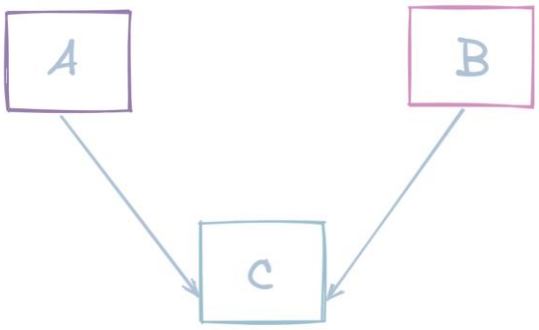
Creational Patterns Neler?

Factory design pattern neden ortaya çıktıyı anlatmadan önce bazı kavramlara değinmem gerek. Dependency ve coupling.

Coupling N tane nesne arasında birbiriyle bağımlı olma durumu.

Dependencey ise daha özele indirilmiş bir tanım 2 nesne arasında oluşan bağımlı olma durumudur.



Örneğin yukarıdaki örnekte A,B,C üzerinden birbirine bağımlı burada coupling var. A'nın C'ye dependency'i var. A değiştiğinde B etkilenmez ama C değiştiğinde çok yüksek ihtimalle A ve B de değişir.

Dependency birlikte değişme problemi ortaya koyar. Aslında gerçek dünyada yazdığımız kodlarda dependencyden kaçış yok. Asıl önemli olan iyi bağımlılıklar kurup dependency'i ve coupling derecesini düşürmek. Bağımlılığı fazla olan bir projeye yeni şeyler eklemek yada bir yeri değiştirmek çok zorlaşır.

Buradaki soruna dependency injection bize yardım ediyor. Dependency injection nesne yaratmayı çözen genel bir patterndir. Bağımlı olunan nesneyi başka bir yerde yaratıp kullanacak olan nesneye verme işlemine dependenct injection denir. Burada bir soru daha sorulması gerek nesneyi kim yaratacak ? İşte tamda bu problemde creational patterns ortaya çıkıyor.

Creational patterns nesne yaratmadan sorumlu bir yapıdır.Nesneleri direkt new ile yaratmak yerine bizim için başka bir yerde yaratılır. Bu patternler uzun yıllar sonucunda ortaya çıkmıştır.

Creational patterns 5 tanedir.

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

Bunlardan factory pattern'ine değineceğim. Nesne yarartırken kullanılabilecek bir tasarım kalıbı. Factory pattern bir interface tanımlar (bir interface yada bir abstract class olabilir) ve alt sınıfların hangi nesneyi başlatılacağına karar verir.Yani nesne yaratılmasından sorumlu bir interface tanımlanır ve alt sınıflara implement edilir. Her alt sınıf bir nesneyi oluşturur.

Kaynakçalar:

https://www.geeksforgeeks.org/factory-method-design-pattern-in-java/https://www.tutorialspoint.com/design_pattern/design_pattern_overview.htm

https://refactoring.guru/design-patterns/factory-method/java/example

https://java-design-patterns.com/patterns/dependency-injection/

https://gokhana.medium.com/dependency-injection-nedir-nas%C4%B1l-uygulan%C4%B1r-kod-%C3%B6rne%C4%9Fiyle-44f4b0d576e4

https://www.tutorialspoint.com/design_pattern/abstract_factory_pattern.htm

https://www.geeksforgeeks.org/abstract-factory-pattern/

https://tugrulbayrak.medium.com/design-patterns-tasarim-kaliplari-3da2018eb9c5#:~:text=1%2D%20Creational%20Patterns%20(Yarat%C4%B1msal%20Kal%C4%B1plar,olu%C5%9Fturmada%20esneklik%20ve%20kolayl%C4%B1k%20sa%C4%9Flar.

https://tugrulbayrak.medium.com/design-patterns-tasarim-kaliplari-3da2018eb9c5

Udemy - Akın Kaldıroğlu Design Pattern ve Spring ile Kurumsal Uygulama Geliştirme:Temeller kurslarından yararlanılıştır