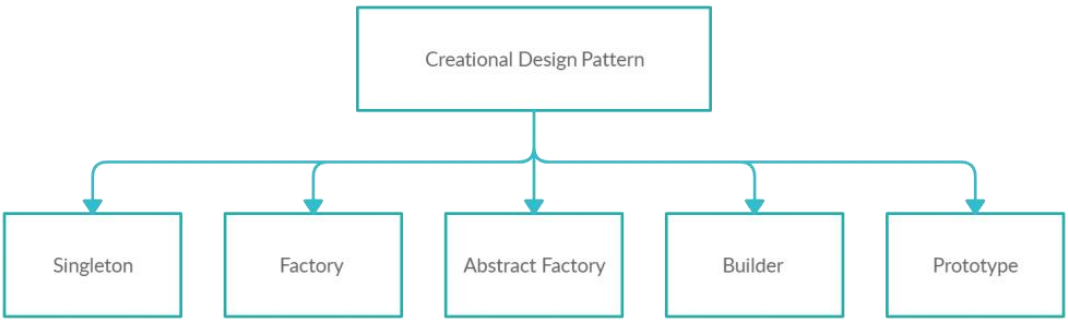
Creational Design Pattern'lar incelenmelidir. Örneklerle anlatınız. (20 PUAN)



Creational Design Pattern'lar nesnelerin nasıl oluşturulacağını ve yönetileceğini kotnrol ederler. Bu kalıplar nesne oluşturma işlemini standartlaştırarak kod okunabilirliğini, tekrar tekrar kullanabilirliğini arttırırlar ve test, bakım işlemlerini kolaylaştırırlar.

5 adet Creational Design Pattern (Yaratımsal Tasarım Kalıbı) vardır. Tek tek örnekler ile inceleyelim.

1-) Fabrika Kalıbı (Factory Pattern)

Bu methodda nesneleri oluşturmak için bir interface oluşturur ve alt sınıflar yardımı ile oluşturulacak nesnenin tipi belirlenir.

Çok fazla alt sınıf olursa kod karmaşıklığı oluşabilir.

Kullanım

- Kullanılacak nesnelerin tipi, bağımlılıkları tam olarak bilinmediğinde kullanılabilir.
- Baştan obje yaratmak yerine zaten yaratılmış objeleri tekrar kullanarak sistem kaynakları daha verimli kullanılabilir.

Shape isimli bir interface tanımlayıp, bu interface'i kullanan kare, daire gibi farklı şekilleri temsil eden sınıflar tanımlayalım. Bu şekilde fabrika metodumuzu oluşturmaya başladık. Shape içinde draw() isimli bir method olsun ve diğer alt sınıflar bu methodu override etsin. ShapeFactory isimili bir sınıf tanımlayıp nesneleri oluşturalım. Oluşturulan nesnelerin draw methodlarını çağırdığımızda farklı sonuçlar göreceğiz fakat bütün şekiller aynı interfaceden geldi. Bu nesne oluşturma işlemine Fabrika methodu denir.

2-) Soyut Fabrika Kalıbı (Abstract Factory Pattern)

Yukarıdaki fabrika methoduna benzer bir method fakat bu methodda direk nesneleri oluşturmak yerine nesneleri farklı ailelere ayırabiliyoruz.

Shape örneğini ele alalım bu örnek için direk kare, dikdörtken, daire yerine Dörtgen Ailesi, Üçgen Ailesi gibi daapsamlı aileler tanımlarsak soyut fabrika yöntemini kullanmış oluruz.

Kullanım

- Kullanılacak nesnelerin kendi aralarında gruplanabiliyor ise ve nesnelerin tam olarak özellikleri bilinmiyor ise bu yöntem kullanılabilir.

3-) Yapıcı Kalıp (Builder Pattern)

Karmaşık bir objeyi daha basit parçalara bölerek adım adım oluştumamızı sağlar.

Bir ev objesi yaratacağımız düşünelim. Direk evi yapmak yerine bu işlemi adımlara bölerek işimizi kolaylaştırabiliriz. Önce tabanı sonra duvarları daha sonra kapı ve pencereleri yapmak gibi.

Kullanım

- Oluşurulacak nesne parçalara bölünebiliyorsa kullanılabilir.

4-) Prototip Kalıp (Prototype Pattern)

Yeni nesneleri baştan yaratmak yerine önceden yaratılmış nesne kopyalanarak üzerinde değişiklikler yapılabilir. Bu işlemlere prototpye pattern denir.

Örnek olarak bir kişinin ad, soyad, yaş bilgilerini tutan bir nesne oluşturup daha sonra ihtiyaç halinde bu nesneyi klonlayıp bu bilgileri değiştirebilir ve farklı bir nesne oluşturmuş oluruz.

Kullanım

- Bir nesnenin birden fazla kopyasının oluşturulması gerektiği durumlarda kullanılabilir.

5-) Tekil Kalıp (Singleton Pattern)

Nesnenin sadece bir defa yaratılması ve bu nesneye global erişim sağlanması durumudur.

Örnek olarak Bir veritabanı bağlantısı sınıfı var ise bu sınıfın sadece bir örneği oluştutulmalıdır birden fazla örnek, birden fazla bağlantı anlamına gelir ve buna gerek yoktur, performansı düşürür.

Kullanım

- Database bağlantısı sınıfı
- Ayarlar sınıfı