Decorator tasarım deseni, structural bir tasarım desenidir. Nesnelerin nesnelerin farklı kombinasyonlar ile farklı özelliklere sahip olabilmesini sağlamak için kullanılır. Yapmak içinse farklı sınıflar oluşturmaktansa her bir özellik için bir sınıf tanımlayarak, var olan nesnenin üzerine bu sınıfın özelliklerini ekleyerek yeni nesne oluşturulmasını sağlar. Bu tasarım desenin kullanarak yazılım karmaşasının önüne geçilir ve hata risk oranı azalır.

Mesela örnek olarak kahve siparişi senaryosunu bir kafeye gittiniz ve menüden bir kahve siparişi vermek istiyorsunuz. Sade Kahve istediniz ama aklınıza şöyle seçenekler de geldi

Süt, çikolata veya şurup tarzı maddeler eklemek geldi. Bu kombinasyonlar için ayrı bir kahve çeşidi menüde tutmak karmaşaya yol açar. Bunun yerine, Decorator deseniyle bu durumu yönetebiliriz.

Temel Sınıf: SadeKahve Sınıfı

Decorator Sınıflar: SütDecorator, CikolataDecorator, SrubDecorator

Müşteri bir sade kahve isterse sadece SadeKahve sınıfı kullanılır. Süt eklemek isterse SütDecorator sınıfı ile kahveye süt eklenir.

Decorator deseni, temel sınıfı (SadeKahve) ve dekoratör sınıflarını (SütDecorator, ÇikolataDecorator) aynı interface veya soyut sınıftan türetir. Bu, temel nesne ve dekoratörlerin birbirlerinin yerine geçebilir olmasını sağlar.

Bir dekoratör sınıfı, kendisine geçilen temel nesneyi bir alt sınıf gibi sarar. Bu işlem sayesinde temel nesne üzerinde işlem yapılabilir, ardından yeni özellikler eklenmiş şekilde kullanıma devam edilir.

Avantajları

Açık-Kapalı Prensibi (Open/Closed Principle): Decorator deseni, yazılımdaki bir nesneyi değiştirmek yerine, ona yeni özellikler ekleyerek genişletmemizi sağlar. Böylece nesneler değişime kapalı ama genişlemeye açık olur. Bu, yazılımın bakımı ve geliştirilmesi açısından önemlidir.

Yeniden Kullanılabilirlik: Aynı dekoratör sınıfı farklı nesneler üzerinde kullanılabilir. Örneğin, bir SutDecorator sadece kahve için değil, başka içecekler için de kullanılabilir. Bu da kodun tekrar kullanılabilirliğini artırır.

Dinamik Olması

Dezavantajları

Çok Fazla Sınıf: Her bir özellik için ayrı bir dekoratör sınıfı oluşturmak, çok sayıda sınıf yazmaya sebep olabilir. Bu durum, projenin karmaşıklığını artırabilir. Ancak bu karmaşıklık, kombinasyonları yönetmenin getireceği karmaşaya kıyasla daha yönetilebilirdir

Işıkay ÇETİN - 1220505031.