Hexagonal Architecture (Ports and Adapters)

Ali Emre Pamuk

April 2025

Giriş

Modern yazılım geliştirme süreçlerinde, uygulamaların dış dünyaya olan bağımlılıklarını azaltmak ve test edilebilirliği artırmak temel hedeflerden biridir. **Hexagonal Architecture**, bu hedefe ulaşmak için geliştirilen mimari yaklaşımlardan biridir. İlk kez **Alistair Cockburn** tarafından tanıtılmıştır ve sıklıkla *Ports and Adapters* mimarisi olarak da anılır.

Hexagonal Architecture Nedir?

Hexagonal mimari, uygulamayı merkezi bir çekirdek (core) etrafında organize eder. Dış dünya ile olan tüm etkileşimler **portlar** ve bu portlara bağlanan **adapter'lar** aracılığıyla gerçekleştirilir. Böylece domain mantığı dış sistemlerden izole edilir.

- **Core (İç Katman)**: Uygulamanın iş mantığını ve domain modelini içerir.
- **Ports (Bağlantı Noktaları)**: Uygulamanın dış dünya ile iletişime geçtiği arabirimlerdir.
- **Adapters (Uyarlayıcılar)**: Portları dış sistemlere bağlayan somut implementasyonlardır (veritabanı, HTTP, mesaj kuyruğu vb.).

Katmanlar ve Bağımlılıklar

- İç Katman dış katmana **asla bağımlı değildir**.
- Tüm dış bağımlılıklar (ör: framework, veritabanı) adaptörler aracılığıyla soyutlanır.
- Testler doğrudan iç katmana uygulanabilir çünkü dış sistemlere bağımlı değildir.

E-Ticaret Örneği: Sipariş Oluşturma

Bir e-ticaret uygulamasında müşteri sipariş verdiğinde sistemin hexagonal mimariye göre nasıl organize edilebileceğine bakalım:

İç Katman (Core)

- **SiparişService**: İş mantığını içerir (stok kontrolü, sipariş kaydı vs).
- **Sipariş**: Domain modeli (Entity).
- **SiparişRepositoryPort**: Siparişin kaydedilmesi için arabirim.

Giriş Adaptörleri (Inbound Adapters)

- REST API: `POST /siparis` ile sipariş alma
- CLI veya Event Listener da olabilir

Çıkış Adaptörleri (Outbound Adapters)

- Veritabanı adaptörü: `JpaSiparisRepository`
- E-posta adaptörü: Sipariş sonrası bilgilendirme

Örnek Süreç

1. REST API üzerinden sipariş alınır (Inbound Adapter).
2. `SiparişService.createOrder()` çağrılır.
3. Sipariş doğrulanır ve `SiparişRepositoryPort.save()` çağrılır.
4. Bu port, veritabanına bağlanan adapter tarafından uygulanır.

Avantajları

- Test edilebilirlik artar (core bağımsızdır).
- Framework bağımlılığı en aza indirgenir.
- Yeni dış sistemler kolayca entegre edilebilir (yeni adapter ekleyerek).

Hexagonal ve DDD Uyumu

Hexagonal mimari, Domain-Driven Design (DDD) ile mükemmel uyumludur:

- Domain modeli core katmanda yer alır.
- Domain servisleri dış dünyadan izole çalışır.
- Repository, EventPublisher gibi arabirimler port olarak tanımlanır.

Sonuç

Hexagonal Architecture, uygulama iç mantığını dış dünyadan izole ederek, sürdürülebilir, modüler ve test edilebilir sistemler geliştirmenin önünü açar. E-ticaret gibi karmaşık senaryolarda bu mimari yaklaşım, yüksek esneklik ve uzun vadede düşük bakım maliyeti sunar.