

Örneklerle Refactoring - 1

2018-04-07T22:43:54+03:00

Contents

Refaktör Etmek Nedir?	1
Diğer Mottomuz: "Keeping Controllers Thin"	1
Sonuç	3
Bağlantılar	4

Refaktör Etmek Nedir?

Refactoring is the process of changing a software system in such a way that it does not alter the external behaviour of the code yet improves the internal structure. It is a diciplined way **to clean up code that minimizes the chances of introducing bugs**. Martin Fowler, Refactoring

Refaktör etmek bir sistemin dış davranışını değiştirmeden, iç yapısını düzelterek bug çıkma ihtimalini düşürme disiplini.

Diğer Mottomuz: "Keeping Controllers Thin"

MVC mimarisinde **controller** üzerinde işlem yapmak, uygulamanın test edilebilirliğini düşürür, modülerlik (reusability) prensibini bozar. İdeal şartlarda **Controller** "veriyi alıp, UI'a iletmelidir" dersek yanlış söylemiş olmayız.

Bu yazımda temel, basit fakat bir o kadar faydalı bir **refactoring** yaparak bir ASP.Net Core View Component'ini **refactor** edeceğiz.

ASP.Net ViewComponent özelliği .NET Core ile geldi. Partial view'ların parametre alabilen, arka tarafında kod çalıştırılabilen daha güçlü hali diyebiliriz.

Bu componentimiz, Restful bir API ile menüyü çekerek **DynamicMenu.cshtml** sayfasına basıyor.

Controller üzerinde işlem yapmamak, business logic çalıştırmamak gibi prensiplerimiz burada da geçerli. Veriyi çekeceğimiz kısmı component dışına taşıyacağız.

```

public class DynamicMenuViewComponent : ViewComponent
{
    public DynamicMenuViewComponent()
    {
    }

    public IViewComponentResult Invoke()
    {
        Menu model = new Menu();
        return View(model);
    }
}

```

Önce anlaşmamızı bir *Interface* üzerinden yapalım.

```

public interface IMenuApi
{
    Menu GetUserMenu();
}

```

Explicit implemantasyonu tercih ediyorum. Bütün metotlarımın Interface üzerinden implemente edilmesini, Interface üzerinden kaldırılan metotların ise kalmamasını istiyorum.

```

public class MenuApi : IMenuApi
{
    Menu IMenuApi.GetUserMenu()
    {
        throw new NotImplementedException();
    }
}

```

Menü verimizi çektiğimiz kısmı, implemente ettiğimiz class'a taşıyoruz.

```

public class MenuApi : IMenuApi
{
    public MenuApi()
    {
    }

    Menu IMenuApi.GetUserMenu()
    {
        Menu model = new Menu();

        return model;
    }
}

```

DI registration işlemimizi yapalım.

```
public class Startup
{
    public void ConfigureServices(IServiceCollection services)
    {
        services.AddTransient<IMenuApi, MenuApi>();
    }
}
```

Constructor Injection ile yeni class'ımızı kullanmaya başlayabiliriz.

```
public class DynamicMenuViewComponent : ViewComponent
{
    private readonly IMenuApi _menuApi;

    public DynamicMenuViewComponent(IMenuApi menuApi)
    {
        this._menuApi = menuApi;
    }

    public IViewComponentResult Invoke()
    {
        var model = _menuApi.GetUserMenu();
        return View(model);
    }
}
```

Sonuç

Ne kazandık? - Öncelikle "single responsibility" prensibini uyguladık. - Test etmesi daha kolay bir yapımız oldu. IMenu üzerinden Mocklama yapıp, dış bir API'ye ihtiyaç duymadan testleri daha hızlı koşturabiliriz. - Menüün veri kaynağı değişebilir. Belki bir JSON dosyası veya veritabanı bağlantısı olabilir. Bunun için mevcut koda dokunmadan IMenu interface'ini implemente edeceğiz. - İleride farklı sayfalarda, role veya kullanıcıya bağlı menü datası için yeni metotlar ekleyebileceğiz. Mevcut durumda Invoke metoduna varsayılan değerleri null olan kullanmadığımız bir çok parametre eklememiz gerekecek ve bir süre sonra if ile yönetmeye başlayacaktık.

```
public IViewComponentResult Invoke(bool isAdmin = false, Guid? userId = null)

    if (isAdmin) {}

    if (userId.HasValue && userId.Value != Guid.Empty) {}
```

Bağlantılar

- <https://jonhilton.net/2016/05/23/3-ways-to-keep-your-asp-net-mvc-controllers-thin/>
- <https://martinfowler.com/bliki/CodeSmell.html>
- <https://refactoring.com/catalog/replaceNestedConditionalWithGuardClauses.html>
- <http://wiki.c2.com/?CodeSmell>
- <https://github.com/lee-dohm/code-smells>
- <https://dzone.com/articles/code-smells-if-statements>
- <https://stackoverflow.com/a/1554691/1766716>