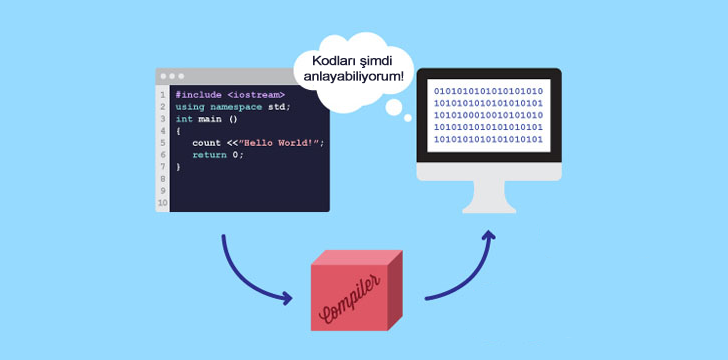
1. Library ile Framework farki nedir?

Framework ve Library 'in ortak noktası, birileri tarafından yazılan ve başkaları tarafından kullanılabilen ortak kod yapılarıdır. İkisinde de başkaları tarafından oluşturulmuş kodların kullanımı söz konusudur. Bu noktada ikisi arasında işlevsel anlamda bir fark mevcut değildir. Bu iki farklı yazılım inşası, temel olarak bir kolaylaştırıcı kod yapısının derlemeleridir. İki farklı yapıda, ortaya çıkaracağınız ürün için bir takım kolaylıklar sağlarlar. Library ile Framework ‘ün ayrıştığı nokta teknik kısımdır. İkisi arasındaki temel teknik fark, kodun nasıl çağrıldığıdır. Library kullanırken, library size bazı özellikler vererek kullanmanızı sağlar, bu şekilde almış olduğunuz kodu kendi sisteminize uygularken size kodu nerede ve ne zaman kullanacağınıza karışmaz veya bir diğer deyişle dikte etmez. Framework ise Library'in tersine kullanacağız özelliğe göre kodu nerede ve ne zaman kullanacağınızı söyler, kullanacağız bu işlev, eğer Framework dokümanın belirtildiği gibi kullanılmaz ise kullanım dışı kalır.

1. compiler nedir?

**Compiler**(Derleyici), geliştiricilerin herhangi bir programlama dilini kullanarak yazdığı kaynak kodu bilgisayarın anlayabileceği makine diline yani 0 ve 1’lere çeviren aracı yazılımdır. Derleyici sayesinde geliştiriciler farklı programlama dillerini kullanarak aynı işlevi yerine getiren yazılımlar üretebilirler. Üstelik Complier’ların varlığı, çok fazla programlama dilinin olmasına ve geliştiricilerin alternatif dillerle çalışmasına yardımcı olmaktadır. Derleyici veya İngilizce adıyla bilinen Compiler, farklı bir dilde oluşturulan kaynak kodun istenilen farklı bir [kod](https://wmaraci.com/nedir/kod) haline dönüştürülmesine yardımcı olan otomatikleştirilmiş programlardır. Derleyici programlar yaygın olarak executable [code](https://wmaraci.com/nedir/code) olarak tanımlanan hemen çalıştırılabilir kodlar üretmektedir.

Derleyiciler sadece aynı seviyedeki programlama dilinde yazılan kodların aynı seviyedeki eşlerine çevrilmesinde görevli değildirler. Bir derleyici, üst seviye bir programlama dilinin kodunu daha alt seviyeli bir programlama diline çevirme görevini üstlenebilirler. Basit bir örnek vermek gerekirse:

Bilgisayarınızda C diliyle hazırlamış olduğunuz bir yazılımı derleyiciler sayesinde makine dili olarak kabul edilen Assembly veya daha alt seviyeli programlama dillerine dönüştürebilirsiniz.

Compiler’ın çalışma mantığı basit bir şekilde özetlememiz gerekirse; C diliyle bir [yazılım](https://wmaraci.com/nedir/yazilim" \t "_blank)geliştirdiniz. Lakin bu yazılımın bilgisayar tarafından kavranıp işleme alınabilmesi için kodunuzun makine dilinde sunulması gerekir. Bilgisayarınız C dilini doğrudan anlayamayacak ancak [PASCAL](https://wmaraci.com/nedir/pascal) ile yapılan bir kodlamayı anlayabilecektir.

Bu aşamada Compiler devreye girip, C kodunuzu PASCAL diline dönüştürecek ve dönüştürme işlemi sırasında meydana gelen hataları size bildirerek kodunuzun sorunsuz dönüştürülmesini ve çalışmasını sağlayacaktır. Compiler’ın oluşturduğu kod çalıştırılabilir kod dizisi olduğu için bilgisayar tarafından işleme alınabilecek ve koşturulabilecektir.

1. interpreter nedir?

**Yorumlayıcı** (*interpreter*), yazılımı kısım kısım ele alarak doğrudan çalıştırır. Yorumlayıcılar standart bir çalıştırılabilir kod üretmezler. Yorumlama işlemi aşama aşama yapılmadığı için genellikle ilk hatanın bulunduğu yerde programın çalışması kesilir.

Derleyicilerin tersine kodun işlenmeyen satırları üzerinden hiç geçilmez ve buralardaki hatalar ile ilgilenilmez.

Yorumlayıcılar genelde kaynak koddan,  makine diline anlık olarak dönüşüm yaptıkları için, derleyicilere göre daha yavaş çalışırlar. Ayrıca kodu iyileştirme (optimizasyon) imkânı da çoğu zaman yoktur.

Yorumlayıcılar, tasarımları itibarıyla, derleyicilere benzer veri yapılarını kullanırlar. Örneğin kelime analizi (lexical analysis) için sembol tablolarından veya sözdizim analizi için (syntactic analysis) BNF tanımlarından faydalanabilirler.

1. Ascii ve unicode nedir?

ASCII karakterler sadece İngilizce üzerinde etkili olurken, Unicode tamamen evrenseldir. Unicode’un farklı sürümleri sayesinde İbranice ve Arapça gibi kompleks diller başta olmak üzere Çince gibi karmaşık diller kolayca dijital ortamlara aktarılabilmektedir. Yalnızca diller değil, Unicode kodlaması sayesinde karmaşık semboller ve karakterler kolayca meydana getirilebilirler.

Yapısal açıdan bilgisayarlar sayılar yardımıyla çalışırlar. Her karakter için hafızlarında bir sayı tutar ve bu sayı yardımıyla karakter, rakam veya sembolün oluşturulmasını sağlarlar. Unicode’un kullanılmaya başlanmasından önce yüzlerce farklı sistem ve binlerce farklı karakter kodlamasının kullanılması gerekiyordu. Üstelik kullanılan bu karakter kodlamalarının hiç biri İngilizce gibi basit diller için bile yeterli değildi. Bir de dünyadaki dil çeşitliliğini düşünürsek karakterlerin bir standarta bağlanması elzem hale gelmişti.

Bilgisayarların erken döneminde kullanılan karakter kodlamaları birbiriyle çelişme özelliğine sahipti. Ortada bir standart olmadığı için bir karakter kodlamasındaki değer farklı bir sembolü oluştururken ötekindeki değer daha farklı bir sembolü meydana getirebiliyordu. Bilgisayarların *(özellikle sunucuların)* bazen birden fazla karakter kodlamasına göre veri işlemesi gerekiyordu. Karakter kodlamalarındaki farklılıklar ve karmaşalar nedeniyle veriler istemsiz olarak değişiyor, bozuluyor ve kullanılamaz hale gelebiliyordu.

Unicode standartı bu sorunların tamamını kesin bir çözüm oldu. Her karakter için benzersiz bir numara kullanılarak platformlar arası karmaşalara çözüm getirildi. Unicode kullanıldığı sürece hangi platformu kullandığınızı hangi cihaz, [yazılım](https://wmaraci.com/nedir/yazilim" \t "_blank) veya dili kullandığınız fark etmiyordu. Yazılım üreticileri kısa süre içinde Unicode standartını kabul ettiler ve yazılımlarını bu standarta bağlı kalarak geliştirdiler ve düzenlediler. Bugün Unicode kodlaması artık her yerde kullanılıyor. Tüm işletim sistemleri, arama motorları, internet tarayıcıları, bilgisayarlar ve hatta akıllı telefonlar bile. [İnternet](https://wmaraci.com/nedir/internet" \t "_blank) dahi Unicode karakter kodlaması üzerinden çalışıyor. URL’ler, HTML, XML, CSS, [JSON](https://wmaraci.com/nedir/json" \t "_blank) vb. diller yine Unicode ile internet dünyasına hayat veriyor.

Unicode standartı hem dijital hem geleneksel medyada içeriklerin standarta bağlanmasına yardımcı oldu. Unicode bugün **Unicode Konsoriyum**’u tarafından kar amacı gütmeden geliştirilmeye devam etmektedir.

**Unicode ile ASCII Farkları**

* ASCII yalnızca Latin alfabesi için kullanılabilir ve Latin alfabelerinde bile çoğu zaman yeteri kadar verimli değildir. Unicode ise evrensel olarak tüm dillerin kullanımına uygun şekilde tasarlanmıştır.
* ASCII’nin kullanımına 1963 yılında başlanırken ASCII’nin yerini alan Unicode’un geliştirilmesine 1980 yılında başlanmıştır.
* Unicode hala Unicode Konsoriyum’u tarafından kar amacı gütmeden geliştirilmeye devam edilmektedir. ASCII’nin geliştirilmesi yıllar önce durdurulmuştur.
* Unicode’un geliştirilmesinin amacı evrensel olması ve platformlar arası yaşanan karmaşaların ortadan kaldırılmasıdır.
* ASCII tam olarak bir standart değilken Unicode tüm dünyada kabül görmeyi başaran bir standarttır.

1. Semantic non-semantic nedir?

Semantik (diğer bir ifade ile anlamsal) elementler, geliştirici ve tarayıcı için anlamlı bir karşılığı olan HTML etiketlerini ifade etmekte. Örneğin, <article>...</article> etiketi anlamlıdır. Çünkü, bu etiketi gördüğümüzde içeriğinde bir makale olduğunu varsayabiliriz.

Bu durum tarayıcılar için de benzer şekilde işler. Dolayısıyla, tarayıcı tarafından gerçekleştirilen işlemlerde bilgileri içerisinde bulundukları etiketlere göre anlamlandırılmaları kolaylaşmaktadır. Tabi bu taramayı internet tarayıcısı dışında arama motoru tarafından gerçekleştirilen taramalar ve hatta [web kazıma](https://ceaksan.com/tr/veri-kazima-data-scraping-nedir) bağlamında da ele alabiliriz. Yukarıdaki görselde de görüldüğü üzere alanların tanımlarını id veya class’lar ile gerçekleştirebilmekteyiz. Id’ler benzersiz olmasına karşın class’ları tekrar tekrar kullanabilmekteyiz ve tahmin edileceği üzere, div ve span gibi semantik olmayan etiketler içerisinde tutulan veri verinin kendisi ele alınmadığı sürece anlamsızdır.

Ancak, aynı yapıyı semantik etiketlerle ele aldığımızda daha anlamlandırılabilir bir yapı ortaya çıkar. Diğer yandan, id ve class tanımlarına olan zorunluluğumuz da ortadan kalkacaktır. Elbette semantik etiketler bizim için yeni bir ifade değil. form, table, ul-ol, img gibi içeriğine dair anlam ifade eden etiketleri uzun zamandır kullanmaktayız. HTML5 bu anlamda bir geliştirme sunmaktadır.

1. Form yapisindaki get/post nedir?

# **HTTP Nedir ?**

Açılımı Hyper Text Transfer Protocol olan HTTP, sunucular ile kullanıcıların birbirileri ile nasıl haberleşeceğini anlatan bir protokoldür(kurallar). Biz google’a girip bir şey arattığımız zaman sunuculara bir istek atarız ve bunun sonucunda sunucu bize bir yanıt gönderir. İşte bu iletişimi sırasında uyulması gereken kurallara HTTP denir.

## HTTP Metotları

* GET
* POST
* DELETE
* HEAD
* PUT
* PATCH
* OPTIONS

**GET** methodu kullanıldığında istekler URL kısmında gönderilir. Gönderilen bilgiler URL’de görüntülendiği için güvenlik riski yüksektir ancak **POST**metoduna göre daha hızlıdır.

/test/demo\_form.php?name1=value1&name2=value2

**POST** metodu ise URL’de görüntülenmesi istemediğimiz istekler için kullanılır. Gönderilen bilgiler URL’de görünmediği için daha güvenlidir ancak **GET**methoduna göre *daha* *yavaştır*. Yani **POST** metodunda form alanları **HTTP REQUEST HEADER** içinde kodlanmış olarak gider, URL’de gözükmez. **POST**işleminde her türlü dosyayı post edebiliriz: XML,Image, File post edebiliriz.

POST /test/demo\_form.php HTTP/1.1  
Host: w3schools.com  
name1=value1&name2=value2

**GET**ile veri göndermede karakter limiti varken **POST**’ta böyle bir limit yoktur. Karakter limiti **GET**metodunda 2048 karakterdir.

**GET**’i ön plana çıkaran bir farktan bahsedelim. **GET** ile değişkenler URL’de yer aldığı için sayfayı değişken değerleri kaybolmadan yer imlerine atmak mümkündür. Yani **GET** ile bir sayfa içeriğini sunucudan almak için tekrar tekrar forma veri girmeye ya da sayfa içi tıklamalar yapılmasına gerek yoktur. Dolayısıyla hazır URL’yi yer imlerine atarak kullanıcı zahmetten kurtarılmış olur. **POST**’ta ise kullanıcı daha önce geldiği aşamaya(sayfa içeriğine) tekrar varabilmek için forma veri girme ya da tıklama zahmetlerine katlanmak zorundadır.

**GET**’in **POST**’a karşı bir diğer dezavantajı şudur: *multi-part binary* türündeki verilerin gönderimi. “multi-part binary” ile kastedilen şey dosyadır. Örneğin; resimdir. **POST**ile forma bir resim koyup sunucuya upload edebiliriz. Fakat **GET**ile bu mümkün değildir.

1. Ajax nedir?

**AJAX** (**Asynchronous**[**JavaScript**](https://tr.wikipedia.org/wiki/JavaScript)**and**[**XML**](https://tr.wikipedia.org/wiki/XML), [Türkçe](https://tr.wikipedia.org/wiki/T%C3%BCrk%C3%A7e): *Eşzamansız JavaScript ve XML*), [İnternet](https://tr.wikipedia.org/wiki/%C4%B0nternet) sayfalarında JavaScript ve [XMLHttpRequest](https://tr.wikipedia.org/wiki/XMLHttpRequest) kullanımı ile etkileşimli uygulamalar yaratan tekniğin adıdır..

En yaygın kullanım alanı, sayfayı yeniden yüklemeye gerek kalmaksızın, sayfada görünür değişiklikler yapmaktır. XMLHttpRequest kullanılarak birden fazla bağımsız işlem yapılabilir. Bazı [bilişim uzmanları](https://tr.wikipedia.org/w/index.php?title=Bili%C5%9Fim_uzman%C4%B1&action=edit&redlink=1), AJAX'ın [HTML](https://tr.wikipedia.org/wiki/HTML) ve [XML](https://tr.wikipedia.org/wiki/XML)'den sonra en yenilikçi İnternet yazılımı olduğunu ve [Web 2.0.](https://tr.wikipedia.org/wiki/Web_2.0.)'ı sonlandırıp, 3. evrenin kapısını açtığını öne sürmüşlerdir.

*Asynchronous JavaScript and XML* sözcüklerinin kısaltması olan **Ajax**, etkileşimli (interaktif) [web uygulamaları](https://tr.wikipedia.org/wiki/Web_uygulamalar%C4%B1) yaratmak için kullanılan bir [web programlama](https://tr.wikipedia.org/w/index.php?title=Web_programlama&action=edit&redlink=1) tekniğidir. Temel amacı arka planda sunucuyla ufak miktarda veri değişimi sayesinde sayfayı daha hızlı güncelleyebilen web sayfaları yapmak, dolayısıyla kullanıcının istediği her anda bütün web sayfasını güncellemek derdinden kurtulmaktır. Bu da web sayfasının etkileşimini, hızını ve kullanılabilirliğini artırmak demektir.

Ajax tekniği aşağıdaki teknolojileri kullanır:

* Bilgiyi biçimlendirmek ve görüntüsünü değiştirmek için [XHTML](https://tr.wikipedia.org/wiki/XHTML) (ya da HTML) ve [CSS](https://tr.wikipedia.org/wiki/Cascading_Style_Sheets).
* Görüntülenecek bilgiyi dinamik olarak göstermek ve onunla etkileşimli çalışmak için özellikle JavaScript
* Web sunucusu ile bilgi değiş tokuşunda kullanılan XMLHttpRequest objesi. Bazı Ajax frameworklerde ve bazı durumlarda web sunucusuyla bilgi alışverişi için XMLHttpRequest yerine [iframe](https://tr.wikipedia.org/w/index.php?title=HTML_elemanlar%C4%B1&action=edit&redlink=1) de kullanılır; diğer uygulamalarda ise dinamik olarak eklenen JavaScript etiketleri (İngilizce: *tag*) de kullanılabilir.
* Kullanıcıyla sunucu arasındaki bilgi alışverişlerinde genelde XML formatı kullanılır. Önceden hazırlanmış HTML, düz metin (İngilizce: *plain text*), [JSON](https://tr.wikipedia.org/wiki/JavaScript_Object_Notation) ve hatta [EBML](https://tr.wikipedia.org/w/index.php?title=Extensible_Binary_Meta_Language&action=edit&redlink=1) dahi olmak üzere herhangi bir format da kullanılabilir. Bu tip dosyalar sunucu tarafında herhangi bir script ([PHP](https://tr.wikipedia.org/wiki/PHP), [ASP](https://tr.wikipedia.org/wiki/ASP) vs.) ile bile üretilmiş olabilir.
* [DHTML](https://tr.wikipedia.org/wiki/DHTML), [LAMP](https://tr.wikipedia.org/w/index.php?title=LAMP_(software_bundle)&action=edit&redlink=1) ve [SPA](https://tr.wikipedia.org/w/index.php?title=Single_page_application&action=edit&redlink=1) gibilerin aksine Ajax kendi başına bir teknoloji değildir, ancak terim birçok teknolojinin bir bütün olarak kullanıldığını ifade eder.

Ajax hemen hemen web’de birçok şey için kullanılır. Bunun bir örneği ise Google Maps. Kullanıcı sayfadakı haritaya tıklayıp resmi çektiği an, bütün sayfa yeniden yüklenmeden bilgi anında gösterilir. Bütün bunlar Ajax’ın senkron özelliği tarafından yapılmaktadır. Ajax aynı zamanda Gmail’de de kullanılmaktadır. Bilginin yüklenmesi için sayfanın tamamının yeniden yüklenmesi gerekmemektedir. Yeni posta geldiğinde, hemen otomatık olarak gelen kutusuna eklenir. Bu kullanıcının yenileme düğmesine sürekli olarak tıklaması ihtiyacını önler. Google Translate'de de Ajax kullanmaktadır. Kullanıcılar bir yazıyı bir dilden başka bir dile tercüme etmek için yazmaya başladıkları zaman, Ajax hemen diğer dildeki anlamı aynı sayfada gösterir.

1. Cdn nedir?

CDN, bir web sitesinin içeriğini en düşük ağ ve işlem gecikmesi ile yani en hızlı bir şekilde kullanıcılara ulaştırmak üzere coğrafi olarak farklı bölgelerde konumlandırılmış sunucu kümesine verilen isimdir.

Birçok e-ticaret ve haber sitesi sayfa açılma süresini düşük tutabilmek için CDN kullanmaktadır. CDN üzerinden kullanıcıdan kullanıcıya genelde pek değişmeyen HTML, JS, CSS, imaj, video ve font dosyaları sunulur. CDN bu tip kaynakları origin olarak adlandırdığı asıl sunucudan (örn: milliyet․com․tr) çekerek belirli bir süre için cache’ler. Bu noktadan sonra CDN’e ilgili kaynaklar için kullanıcıların yaptığı istekler cache’lenen bu kopyadan sağlanır. Cache süresi dolduğunda origin’e yeniden yapılan isteklerle kaynaklar tazelenir.

CDN sağlayıcıları dünya üzerinde pek çok ülkede sunucular barındırarak kullanıcıların yaşayacağı ağ gecikmesini minimize ederler. Ayrıca dağıtık durumdaki sunucular ilgili sitenin veri merkezinin sıkışıklık yaşamasını da engeller.

CDN sağlayıcıları birçok web sitesine aynı sunucular üzerinde CDN hizmeti verdikleri için maliyetleri konsolide ederek düşürebilirler. Ayrıca CDN’de tutulan verinin bozulmasından dolayı bir kayıp yaşanmayacağı için CDN hizmeti için daha düşük maliyetli, merdiven altı denilebilecek veri merkezlerindeki sunucular kullanılıp maliyet iyice düşürülebilir. Web siteleri açısından bakıldığında ise CDN kullanımı ile büyük boyutlu kaynaklar veri merkezlerinden sunulmayıp veri merkezleri ve bulut sağlayıcıya ödenecek bir maliyetten kaçınılmış olur.

Sadece statik içerikten oluşan bir site kolaylıkla CDN üzerinden hizmet verebilir. Sunucu tarafında sayfa oluşturma veya api ile hizmet veren dinamik sitelerde ise CDN entegrasyonu için değişik yöntemler vardır. Uygulanabilecek temel üç yöntem aşağıda detaylandırılmıştır.

Birinci ve muhtemelen en popüler opsiyon web sitesinin ana sayfasını ve dinamik istekleri kendi veri merkezinde karşılamak, ana sayfanın HTML’i içerisinde referans verilen CSS, JS, imaj, vb kaynaklar için ise CDN adresini kullanmaktır.

Bu opsiyonda ana sayfa apex domain ya da www subdomain’inden, dinamik sayfalar www’dan ve web istekleri de api’dan sunulur. Statik kaynaklar cdn veya başka bir subdomain’den sunulabilir. cdn subdomain’inin dns kaydında CDN sağlayıcının IP’si bulunur.

1. Emmet.io nedir?

Emmet, Html ve CSS iş akışını büyük ölçüde geliştiren,biz web geliştiricilerinin sıklıkla kullandığı, kodları daha hızlı yazıp zamandan tasarruf etmek için tercih ettiğimiz bir eklenti diyebiliriz.

**Neden kullanmalıyız?**Emmet’in temel mantığı, yazılımcıya kodlama yaparken zaman kazandırmasıdır. Bu yüzden çoğu yazılımcı text editöründe Emmet eklentisini kullanır. bir örnekle açıklayacak olursak örneğin ;



şeklindeki iç içe girmiş bir kod yapımız olsun, bu kodu tek tek iç içe yazmak yerine;  
div>ul>li\*5>a şeklinde yazarak ekrandaki kod yapısını elde etmiş oluruz.

Emmet birçok Text Editörü ile uyumlu şekilde çalışır. Yapmanız gereken sadece [emmet.io](https://emmet.io/) adresine gidip burada Download sekmesine gittikten sonra ekranda bulunan text editörlerinden size uygun olanı seçip,indirmek. Bu kısaltmalarınızın editörünüzde çalışması için, kısaltılmış kodları yazdıktan sonra editörün tamamlaması için klavyenizden **TAB**tuşuna basmalısınız.

**Child: >** = Elemente alt elemanlar eklemek için kullanılır. (*div>ul>li\*5>a*)

**Sibling: +** = Elementin yanına elamanlar ekler. (*body>div+div+div)*

**Multiplication: \***= Elementin kaç defa üretileceğini belirtir. (*div>ul>li\*5)*

**Class: .**= . karakteri ile oluşturmak istediğiniz elemente class özelliği eklenmiş olur ve girmiş olduğunuz değer atamasıda class özelliğine atanmış olur. (*div.diskutu>ul>li.menu)*

**ID: #**= # karakteri ile oluşturmak istediğiniz elemente id özelliği eklenmiş olur ve girmiş olduğunuz değer atamasıda id özelliğine atanmış olur. (*div#diskutu>p#metin)*

**Item numbering: $** =$ işareti ile sayısal olarak otomatik artan değer üretimi sağlar. (*ul>li.item$\*5)*

**Text: {}**= Belirttiğimiz elementin için metin eklememize yardımcı olur. (*p{Ne Mutlu Türküm Diyene})*

# **Emmet ile HTML 5 İskeleti Oluşturmak**

Bir HTML sayfası oluştururken bunun için yapmanız gereken ilk şey sayfa için bir iskelet oluşturmaktır. Bazen bu iskeleti oluştururken hangi kodun nerede olduğu unutabiliyoruz. İşte Emmet tam burada devreye giriyor ve HTML uzantılı dosyamıza sadece “ **! ”**yazdıktan sonra **TAB**tuşana basmamız yeter. sadece tek bir ünlem işareti ile HTML 5 iskeletimizi oluşturmuş oluyoruz.