String Sınıfı – Metotları

Projelerimizde bir çok defa değişken olarak **string** tiplerini kullanmaktayız. Nedir bu string, şimdi bu yazımda bunları irdeleyelim birlikte. String tipler, **System.String** sınıfını temsil etmektedirler. String aslında ard arda gelen karakter dizisidir. String tipler, genel olarak *value* (değer) tipinde kabul edilirler ancak gerçekte *reference* (referans) tip veri türleridir ve bellekte stack alanında değil heap alanında tutulurlar. Tanımlanma şekilde bir çok şekilde olmakla birlikte genel olarak aşağıdaki gibi tanımlanırlar :

string adSoyad = **”Bilgisayar Programlama”;**

string adSoyad;

adSoyad = **”Bilgisayar Programlama”** gibi..

String sınıfının bir çok metotu bulunmaktadır, bunların hepsini tabii ki açıklamak mümkün değil ama en sık kullanılanlara değinmeden geçmek istemiyorum. Gruplayarak anlatalım. Önce açıklamaları yapacağım sonra da örnekle pekiştireceğim.

String Toplama Metotları :

En sık kullanılanı **”+”** operatörüdür. + ile iki string ifade ard arda toplanır. Örneğin :

string ad = **”Bilgisayar”;**

string soyad = **”Programlama”;**

string AdSoyad = **ad + soyad;**

şeklinde işlem yapılabilir, ya da

string adSoyad = **”Bilgisayar”** + **” Programlama”** ; şeklinde de işlem yapılabilir.

Karşılaştırma Metotları :

Compare() : İki stringi karşılaştırmak amacı ile kullanılır.Metotun en sık kullanılan overload edilmiş hallerini de açıklayayım.

“ **Compare (stringifade1, stringifade2) :** Eğer stringifade1 , stringifade2”den büyük ise sıfırdan büyük bir değer geriye döner, eğer stringifade2, stringifade1!den büyük ise bu sefer geriye sıfırdan küçük bir değer döner. Eğer her iki ifade birbirine eşit ise geriye 0 (sıfır) döner.

“ **Compare(stringifade1, stringifade2, boolDurum) :** Üstteki metot ile aynı işi yapmaktadır yalnız ek olarak, boolDurum ile tanımladığım parametre eğer true ise iki ifadeyi karşılaştırırken büyük “ küçük harf duyarlılığını dikkate almaz. Yani “a” ile “A” aynıdır.

“ **Compare (stringifade1, indexno1, stringifade2, indexno2, adet) :** İlk metotla aynı işlemi yapar farkı şudur, stringifade1”in indexno1”inden itibaren adet kadar stringifade2”nin indexno2”den itibaren adet kadar olan kısmını karşılaştırır.

“ **Compare (stringifade1, indexno1, stringifade2, indexno2, adet, boolDurum) :** Üstteki metot ile aynı işlevi yapar tek farkı ikinci metotta olduğu gibi ifadelerin büyük harf küçük harf duyarlılığı durumunu eğer boolDurum true olarak belirtilmiş ise dikkate almaz.

CompareTo() : Karşılaştırma işlemini nesne üzerinde gerçekleştiren statik olmayan bir karşılaştırma metotudur. Herhangi bir hassasiyet olmadan iki stringi karşılaştırır, birinci ifade büyük ise sıfırdan büyük bir değer , ikinci ifade birinci ifadeden büyük ise sıfırdan küçük bir değer eğer her iki ifade birbirine eşit ise bu sefer 0 (sıfır) döner.

CompareOrdinal() : Statik bir sınıftır ve her bir unicode karakterin sayısal gösterimini karşılaştırmaktadır. Böylelikle iki ifadenin alfabetik sıralaması hakkında bilgi alabiliriz. Eğer ilk ifade alfabetik olarak önde ise alfabetik olarak önde ise sıfırdan küçük bir sayı, eğer ikinci ifade alfabetik olarak önde ise bu sefer sıfırdan büyük bir sayı, eğer iki ifade birbirine eşit ise bu sefer 0 (sıfır) döner.
Equals() : İki ifadeyi karşılaştırır geriye bool türünde değer döndürür. Eğer eşitlik var ise geriye true döner, yoksa false döner, aynı işlemi "==" operatörü ile de yapabiliyoruz.
Arama Metotları : Bir string ifade içinde arama yapmak amacı ile kullanılırlar.
IndexOf() : String ifade içinde herhangi bir karakteri ya da bir ifadeyi arar. Aranan karakter yada ifadenin ilk karakteri bulunduğunda geriye buranın indeks numarasını döner. Eğer arama işlemi sonuçsuz ise geriye -1 döner.
LastIndexOf() : Üstteki metot gibidir ancak tek farkı, bu sefer aranılan ifade içinde aranan ifadenin son bulunduğu indeks numarasını döner.
IndexOfAny() : Parametre olarak bir karakter dizisi alır ve bu dizinin herhangi bir elemanının string ifade içindeki ilk bulunduğu indeks numarasını döner. Bulamaz ise -1 döner.
LastIndexOfAny() : Üstteki metotla aynı işlemi yapar yalnızca aranan elemanın en son bulunduğu indeks numarası döner, yoksa -1 döner.
StartsWith() : Bir string ifadenin belirtmiş olduğumuz şekildemi başlayıp başlamadığını kontrol edebiliriz, eğer bu şekilde ise geriye true döner, değilse false döner.
EndsWith () : Bir string ifadenin belirtmiş olduğumuz şekildemi bitip bitmediğini kontrol edebiliriz, eğer bu şekilde ise geriye true döner, değilse false döner.
Birleştirme ve Parçalama Metotları :
Concat() : Bu metot **"+"** operatörü ile aynı işlemi yapmaktadır. **"+"** operatörü daha çok tercih edildiği için kullanılma gereksinim ihtiyacıda dolayısı ile düşmüştür.
Join() : String türündeki bir dizinin elamanlarını tek bir string değişkend toplamaya yarar, ayrıca istenirse elemanlar arasına ayraç, kelime vs eklenebilir.
Split() : Join metotunun tersidir, bir değişkende yer alan string ifadeleri belirmiş olduğumuz ayraç şekline göre parçalayıp string türünde bir diziye atmamıza yarar.
PadLeft() : String bir ifadenin soluna belirli sayıda karakter ekleyerek uzunluğunu belirli bir adete sağlamamızı sağlar.

PadRight() : String bir ifadenin sağına belirli sayıda karakter ekleyerek uzunluğunu belirli bir adete sağlamamızı sağlar.

Trim() : Bu metot ile bir string ifadenin başındaki ve sonundaki boşluk karakterlerini silebiliriz, özellikle üyelik işlemlerinde kullanıcıların kullanıcı adı, şifre vs gibi bilgilerinde yer alan baştaki ve sondaki boşlukları silme gibi işlemleri düşünebiliriz.
Kopyalama ve Değiştirme Metotları :
CopyTo() : String bir değerin belli bir konumundan sonraki karakterlerini konumu ve uzunluğu belirlenmiş bir diziye kopyalama işlemi yapar.
Insert() : Bir string değeri başka bir string içine belirli bir indeks numarasından itibaren ekler.
Replace() : String ifadenin içinde belirtilen bir ifadeyi bir başka string ifade ile değiştirmek için kullanılır.

Substring() : String değerin başlangıç noktası ve uzunluğu belirtilmiş alandaki kısmını elde edebilmemizi sağlar. Eğer elde etmemizi istediğimiz uzunluk belirtilmez ise yani ikinci parametre girilmez ise string değerin başlangıç olarak beğendiğimiz indeks numarasından itibaren sonuna kadar okur.
ToLower() : String ifadeyi küçük harflere çevirir.
ToUpper() : String ifadeyi büyük harflere çevirir.
Tüm bu metotların tabii ki bir çok overload edilmiş hali mevcuttur, bunların hepsini burda anlatmamız imkansız, dolayısı ile kullanımda temel olan şekillerini ele aldım. Şimdi bunları bir örnekte kullanalım nasıl kullanıldıklarını görelim :