

*Tür Dönüşümleri*

- *Uygularımızda bazen daha önceden tanımladığımız değişkenlerimiz veri tiplerinde değişiklik yapmamız gerekebilir.*
- *Bu durumlarda var olan değişkenimizin tipini değiştirebiliriz buna tür dönüşümleri denir.*

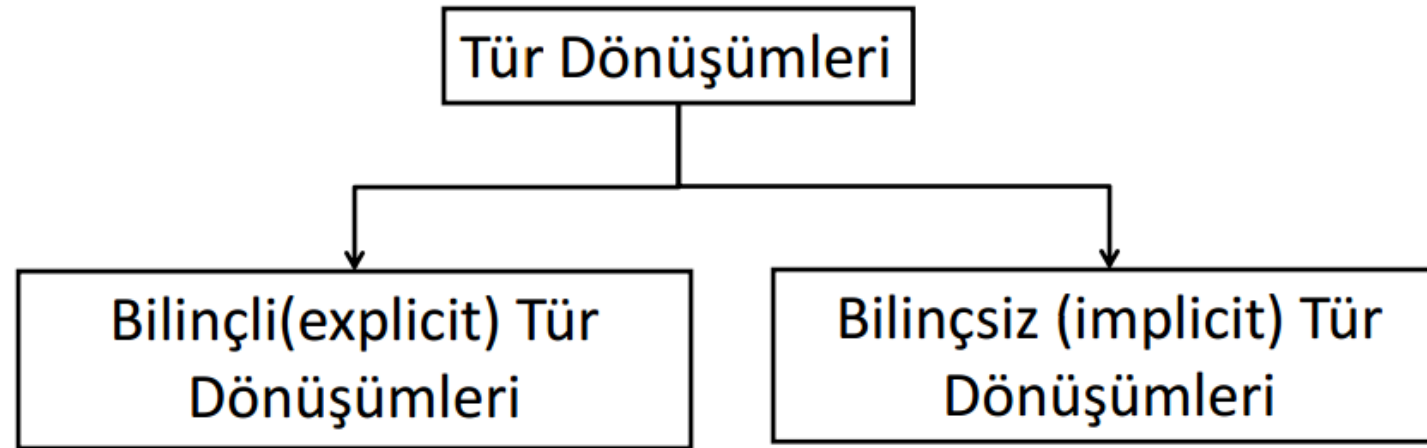
# *Tür dönüşümüne ihtiyaç duyduğumuz durumlar;*

- *En sık kullanılan durum ekrandan yani ister konsol ister form ekranı olsun, okuduğumuz değerler string türündendir, eğer okunan değer bir sayı ve matematiksel hesaplamalarda kullanılacaksa sayısal veri tiplerinden(int, double, decimal vb.) birine dönüştürmek gerekir.*
  - *Bu dönüşüm gelebilecek sayının türüne uygun olarak seçilmelidir.*
  - *Kullanıcıdan farklı değer tiplerinde değerler alırız.(Ürün fiyatı, yaş, finansal değer vb olabilir)*

- . Önceden tanımladığımız bir int değer sonrasında daha çok bir kapasiteye ihtiyaç duyduğumuzda double değişkenine dönüştürülür*
- . Bazen de daha küçük kapasiteli değişkene dönüştürebiliriz.*
- . Farklı türden olan ve tek bir matematiksel işlemde toplanan değişkenleri tek bir tipte toplamak için.*

# *Tür Dönüşümü Çeşitleri*

- *İki çeşit tür dönüşümü vardır*



## *Bilinçsiz Tür Dönüşümü(implicit-gizli, örtülü)*

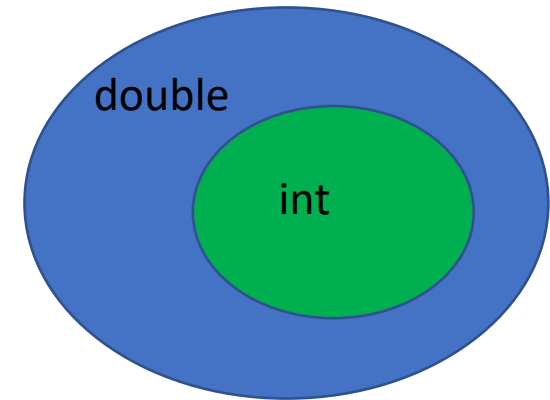
- *Düşük kapasiteli bir değişken, sabit ya da değişken ve sabitlerden oluşan matematiksel ifade daha yüksek kapasiteli bir değişkene atanabilir.*
- *Buna bilinçsiz tür dönüşümü denir, bunun için herhangi bir özel kod gerekmez.*
- *Küçük tür ile büyük tür arasında oluşan boş bitler 0 ile beslenecektir yani herhangi bir veri kaybı olmayacaktır.*

*Double tipi int tipini kapsıyor.*

```
int a=12;
```

```
Double b;
```

```
b=a; //a geçici olarak double a dönüştü
```



## *Birbirini kapsayan veri türleri*

Kaynak	Hedef
sbyte	short, int, float, long, double, decimal
byte	short, ushort, int, uint, long, ulong, float, double, decimal
short	int, long, float, double, decimal
ushort	int, uint, long, ulong, float, double, decimal
int	long, float, double, decimal
uint	long, ulong, float, double, decimal
long, ulong	float, double, decimal
char	ushort, int, uint, long, ulong, float, double, decimal
float	double



# *Bilinçli Tür Dönüşümü(explicit – aşikar, açık)*

- *C# büyük türün küçük türe bilinçsiz dönüşümünü engellemiştir. Bunun sebebi fark edilmeden yapılan bu dönüşümlerde gerçekleşecek veri kaybının önlenmesidir.*
- *Bilinçli tür dönüşümü genellikle **derleyicinin izin vermediği durumlarda** kullanılır.*
- *Büyük bir veri türü daha düşük bir veri türüne dönüştürülebilir bu işlem dikkatli yapılmadığında veri kayıplarına neden olabilir.*

- Zaten bilinçsiz tür dönüşümünün küçük veri tipinden büyük veri tipine dönüşümü izin verip büyükten küçüğe izin vermemesinin nedeni bu veri güvenliği sorunudur.

# *Bilinçli Tür dönüşümünde kullanılan yapılar*

- *parantez() operatörleri kullanarak bilinçli tür dönüşümünü*
- *Parse komutu ile*
- *Convert metodu ile*

*Bu üç yöntemden en çok Convert metodu kullanılır.*

*Null değerlerin dönüşümü gibi durumlarda hata vermez o yüzden **Convert metodunu kullanırız***

# *Parantez operatörü () ile Bilincli Tür Donusumu*

- Burada parantez() operatörleri kullanarak bilinçli tür dönüşümünü yapmayı inceleyeceğiz.*
- İnt türünden byte veri türüne dönüşüm yapacağız. Dönüştürülen değer gittiği veri tipinin değer aralığını aşıya bile dönüşüm gerçekleştirilir ama değer kaybı yaşanır.*
- Byte veri tipi 1 byte'lık veri tutar.            İnt veri tipi ise 4 byte'lık veri türüdür.*
- İlk başta bellek bölgesinde çok yer ayırdığımızı bu kadar alana gerek olmadığını anlayıp int' dan byte'a dönüştürme yapalım.*

- *İlk olarak byte ve int'in min ve max değerlerini hatırlayalım ve ekrana yazdıralım;*

```
byte sayi1 = byte.MaxValue;
```

```
Console.WriteLine(sayi1);
```

```
sayi1 = byte.MinValue;
```

```
Console.WriteLine(sayi1);
```

```
int sayi2 = int.MaxValue;
```

```
Console.WriteLine(sayi2);
```

```
sayi2 = int.MinValue;
```

```
Console.WriteLine(sayi2);
```

- *Sonra sayi2 değişkenine 25 değerini atıyoruz sonra sayi2 değişkenini byte dönüştürüp sayi1' e eşitliyoruz(atıyoruz)*

```
sayi2 = 25;
```

```
sayi1 = (byte)sayi2;//bilinçli tür dönüşümü(byte->int)
```

```
Console.WriteLine(sayi1);
```

```
Console.WriteLine(sayi2);
```

```
Console.ReadLine();
```

## ***Checked ,Unchecked ile Bilinçli Tür Dönüşümü(Kontrol)***

- *Bilinçsiz tür dönüşümüyle yalnızca küçük türler büyük türlere dönüşebiliyordu, yani veri kaybı olması imkansızdı.*
- *Halbuki bilinçli tür dönüşümünde veri kaybı gerçekleşebilir, eğer dönüşümünü yaptığımız değişkenin tuttuğu değer dönüştürülecek türün kapasitesinden büyükse veri kaybı gerçekleşir.*
- *Bu gibi durumlar için C#'ın **checked** ve **unchecked** adlı iki anahtar sözcüğü vardır.*

# ***Checked Bloğu***

- *Dönüşümü kontrol eder veri kaybı varsa hata verir.*
- *Değişkenimizin türünü daha küçük bir tipe çevirirken veri kaybı oluyorsa program çalışma zamanında hata veriyor.*
- *Checked anahtar kelimesi ile yapılan dönüşümlerde eğer bir taşma yani atanan veri tipinden daha büyük bir değerlikli veri dönüştürülecekse değer kaybı yaşanmaması için bize hata verir. Parasal uygulamalarda bizim için bu kontrol önemlidir.*



# ***UnChecked Bloğu***

- *Checked tersi, program hata vermeden yoluna devam ediyor.*
- *Yani değişkenimizin türünü daha küçük bir tipe çevirirken veri kaybı oluyorsa program hata vermeden yolumuza devam edebiliyoruz.*
- *Unchecked yazmamıza gerek yoktur yazmadığımız zamanlarda normal bilinçsiz tür dönüşümü gibi davranır.*

Program.cs

Program.cs

Program.cs

Program.cs

Program.cs

Program.cs

Ornek3.2\_checked\_unchecked

Ornek3.2\_checked\_unchecked.Program

Main

```
16 int sayi2 = 100;
17 sayi2 = 600;
18 sayi1 = (byte)sayi2;
19 Console.WriteLine(sayi1);
20 Console.WriteLine(sayi2);
21 unchecked
22 {
23     sayi2 = 600;
24     sayi1 = (byte)sayi2;
25 }
26 Console.WriteLine(sayi1);
27 Console.WriteLine(sayi2);
28 checked
29 {
30     sayi2 = 600;
31     sayi1 = (byte)sayi2;
32 }
33 Console.WriteLine(sayi1);
34 Console.WriteLine(sayi2);
35 Console.ReadLine();
36
```

Özel Durum İşlenmedi

**System.OverflowException:** 'Aritmetik işlem taşmayla sonuçlandı.'

[Ayrıntıları Görüntüle](#) | [Ayrıntıları Kopyala](#) | [Live Share oturumunu başlat](#)

[Özel Durum Ayarları](#)

C:\Users\Arif\Desktop\DESLER\MASAÜSTÜ UYGULAMA GELİŞTİR

88  
600  
88  
600

5 %

Sorun bulunamadı

5

Otomatikler

Arama (Ctrl+E)

Arama Derinliği: 3

Ad	Değer
(byte)sayi2	88
sayi1	88
sayi2	600

## ***Convert ile Tür Donusumu Dog.Tar.YasıBulma***

- *Convert komutu ile tür dönüşümünü; doğum yılınızı girdiğimizde yaşımızı veren program üzerinden anlatacağız.*
- *Ekrandan alınan(okunan) her şey string dir. Biz bunu matematiksel işlemlerde kullanmak istiyorsak dönüştürmemiz gerekir*

```
string dogumYili;//string=metin
int yil;
int yas;
Console.WriteLine("Doğum yılınızı giriniz");
dogumYili = Console.ReadLine();
yil = Convert.ToInt32(dogumYili);//doğumyılı değeri int yapıp yıl değişkenine atanıyor.
//string int e dönüştü conver sınıfı kullanılarak yapıldı 32 b
yas = 2020 - yil; //yaşı hesaplatıyoruz
Console.WriteLine("yaşınız:" + yas.ToString()); // burada tekrar int değeri string e çeviriyoruz
//çevirirken converte yazmaya gerek kalmadan to
//her değişkenin toString metodu var.

Console.ReadLine();
```

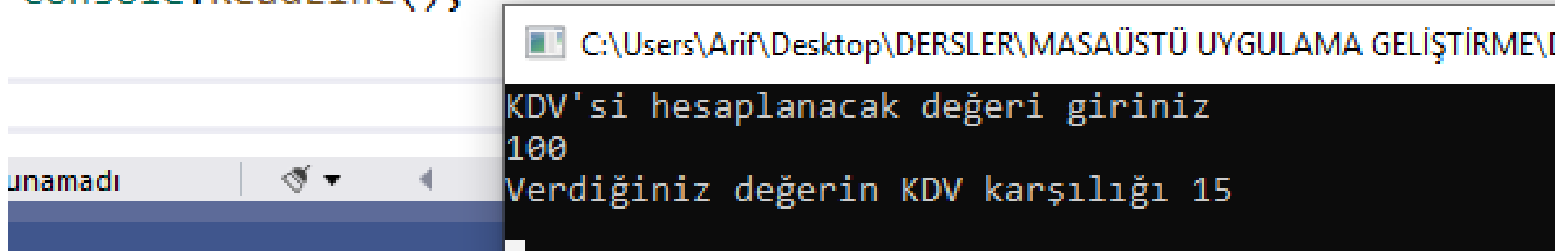
C:\Users\Arif\Desktop\DESLER\MASAÜSTÜ UYGULAMA GELİŞTİRME\Derste Yapılan Örnekler\MasaUstuUygul

```
Doğum yılınızı giriniz
1980
yaşınız:40
```

# Parse İle Tür Donusumu

- *Parse komutu ile tür dönüşümünü inceleyelim.*
- *Kullanıcının girdiği değerin %15 KDV'sini hesaplayan bir örnek üzerinde yapalım*

```
Console.WriteLine("KDV'si hesaplanacak değeri giriniz");  
string okunandeger = Console.ReadLine();  
int hesaplanacakdeger = int.Parse(okunandeger);  
int kdvli = (hesaplanacakdeger / 100) * 15;  
Console.WriteLine("Verdiğiniz değerin KDV karşılığı {0}", kdvli);  
Console.ReadLine();
```



```
C:\Users\Arif\Desktop\DESLER\MASAÜSTÜ UYGULAMA GELİŞTİRME\  
KDV'si hesaplanacak değeri giriniz  
100  
Verdiğiniz değerin KDV karşılığı 15
```

