

# Değişkenler Tür Dönüşümü

Adem AKKUŞ

Bilgisayar Mühendisi

Bilişim Teknolojileri Öğretmeni

Eğitmen

# Değişkenler

- \* Program içerisinde daha sonra kullanılmak üzere bilgi saklanan alanlardır.
- \* Program içerisinde saklanacak bilgiler RAM üzerinde depolanır.

# Değişkenler

- \* RAM üzerinde saklanan C# türleri ikiye ayrılır.
  1. Bellekte kapladıkları alan belli olan değer türleri,
  2. Bellekte kapladıkları alan değişkenlik gösterebilen (genişleyebilen) referans türleridir.

# Değer Tipleri (Value Types)

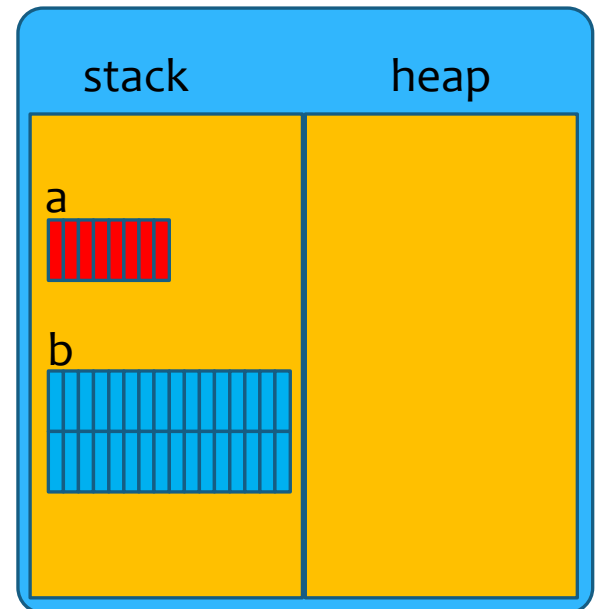
\* Bellekte kapladıkları alan bellidir.

* Byte	(8 bit)	Tamsayı
* Int	(32 bit)	Tamsayı
* Long	(64 bit)	Tamsayı
* Char	(16 bit)	Tamsayı
* Bool	(True veya False)	
* Float	(32 bit)	Reel Sayı
* Double	(64 bit)	Reel Sayı
* Decimal	(128 bit)	Reel Sayı

# Değer Tipleri (Value Types)

- \* Belleğin stack alanında saklanırlar.
- \* Stack alanında saklanan verilere erişim çok hızlıdır.
- \* Saklanan verinin boyutu belirli olmalıdır.

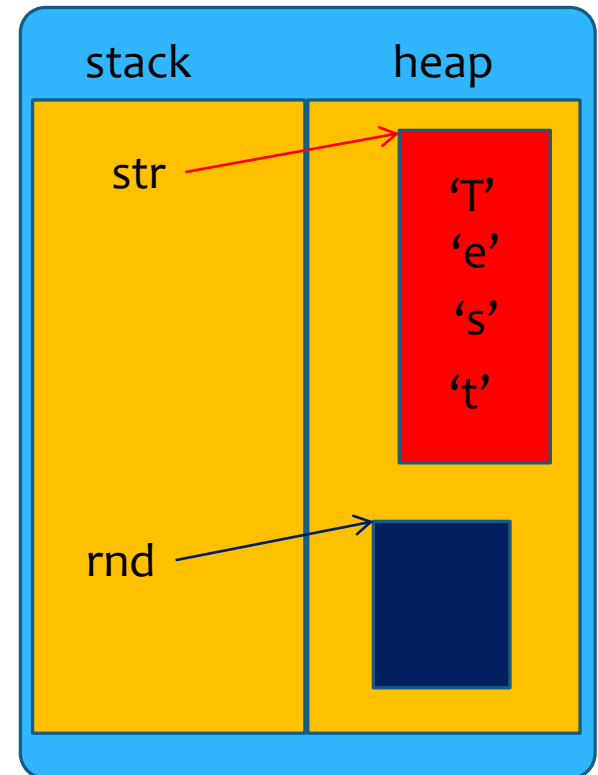
```
int a;  
int b;
```



# Referans Türleri (Reference Types)

```
string str = "Test";  
Random rnd=new Random();
```

- \* Bellekte kapladıkları alan belli değildir.
- \* Kapladıkları alan dinamik olarak değiştirilebilir.
- \* Belleğin heap alanında saklanırlar.
- \* new anahtar kelimesi ile oluşturulurlar.



# var Tipi Değişkenler

var tipi değişken tanımlarsanız eğer değişkenin tipini siz değil derleyici tanımlayacaktır. Derleyicinin tanımladığı tipe göre o veri RAM üzerin de o tipe saklanacaktır.

```
1 | var a = 10.6; //a değişkeninin tipi double dır.  
2 | var b = 20; //b değişkeninin tipi int dir.  
3 | var c = "asd"; //c değişkeninin tipi string dir.  
4 | var d = true; //d değişkeninin tipi booldur.
```

# var Tipi Değişkenler

Not: **var** değişken tipi, diller arası, veritabanları arası entegrasyonu sağlarken veri tipleri uyumsuzluğunu gidermek için oluşturulmuş bir tiptir. Yani C#'ta int ile tanımlanan bir değişken Delphi 'de başka türlü tanımlanabilir. var değişken tipide bütün dillerde evrensellik özelliği taşımaktadır. Sizlere tavsiyem normal kodlamada verinizin tipine göre normal değişken tanımlayınız.

```
1 | var a = 10.6; //a değişkeninin tipi double dır.  
2 | var b = 20; //b değişkeninin tipi int dir.  
3 | var c = "asd"; //c değişkeninin tipi string dir.  
4 | var d = true; //d değişkeninin tipi booldur.
```



# object Tipli Değişkenler

**object** olarak tanımlanan değişken, her tipteki veriyi içinde saklayabilir. Fakat object tipteki değişkene atanan değer Boxing(Kutulama) işlemine uğrar. Yani, object tipiyle oluşturduğumuz bir **string** değişken, RAM'e string olarak değil de **object** olarak kaydedilir. Tanımlanan object değişkeni kullanmak istediğimiz de içinde saklanan verinin tipini elde ederek kullanmalıyız. Bunun içinde **Unboxing**(Kutudan Çıkarma) işlemi yapılır. Bu işlem sonucunda object içindeki asıl tipini almış oluruz.

```
1 | object o1 = "ali";  
2 | object o2 = 12.5f;  
3 | object o3 = 'c';  
4 | object o4 = true;
```

# object Tipli Değişkenler

Yukarıda object tipiyle oluşturulan değişkenler bulunmaktadır. Bu değişkenlere verilen değerlerin hiç biri RAM'e tipine göre isimlendirilerek kaydedilmemiştir. **object** olarak kaydedilmiştir. İşte bu işleme **Boxing**(Kutulama) denir.

```
1 | object b=10; //Boxing
```

```
1 | object ahmet1 = "ahmet";//Boxing  
2 | string ahmet2 = (string)ahmet1;//Unboxing
```

# object Tipli Değişkenler

Son olarak object tipli değişkenler de değişkenin içinde ki verinin tipini dinamik olarak öğrenmek istiyorsak aşağıdaki metod işimizi görecektir.

```
1 | object dd=TİPİBİLİNMEYENBİRDEĞER;  
2 | dd.GetType();/*Bu metod sayesinde  
3 | dd isimli object değişkeninin  
4 | içinde sakladığı değişkenin tipini  
5 | öğrenebiliriz.*|
```

# string Değeri Sayısal Türlerle Dönüştürme

C#'ta string'i sayısal (integer) türlere dönüştürmenin üç yolu vardır.

1. **Parse()** metodu kullanmak
2. **Convert** class kullanmak
3. **TryParse()** metodu kullanmak (önerilen)

# 1. Parse() Metodu Kullanmak

`Parse()` tüm ilkel veri türleri için kullanılabilir. Dizeden tamsayıya dönüştürmenin en kolay yolu budur.

Ayrıştırma yöntemleri 16, 32, 64 bit işaretli tamsayı türleri için mevcuttur:

## Method Overloads

```
Parse(string s)  
Parse(string s, numberstyle style)  
Parse(String s, NumberStyles style, IFormatProvider provider)
```

# 1. Parse() Metodu Kullanmak

```
Int16.Parse("100"); // returns 100
```

```
Int16.Parse("(100)", NumberStyles.AllowParentheses); // returns -100
```

```
int.Parse("30,000", NumberStyles.AllowThousands, new CultureInfo("en-au"));  
// returns 30000
```

```
int.Parse("$ 10000", NumberStyles.AllowCurrencySymbol); //returns 10000
```

```
int.Parse("-100", NumberStyles.AllowLeadingSign); // returns -100
```

```
int.Parse(" 100 ", NumberStyles.AllowLeadingWhite|NumberStyles.AllowTrailingWhite);  
// returns 100
```

```
Int64.Parse("2147483649"); // returns 2147483649
```

# 1. Parse() Metodu Kullanmak

//geçersiz dönüşümler

```
int.Parse(null); //throws FormatException
```

```
int.Parse(""); //throws FormatException
```

```
int.Parse("100.00"); // throws FormatException
```

```
int.Parse("100a"); //throws formatexception
```

```
int.Parse(2147483649); //throws overflow exception use Int64.Parse()
```

## 2. Convert Class Kullanmak

**short** değişkeni 16 bitlik işaretli tamsayı saklar.

`short x = Convert.ToInt16( Console.ReadLine() );`

16 bitlik  
işaretli  
tamsayı

16 bitlik  
işaretli  
tamsayı

string döndürür

Yukarıdaki atama geçersizdir. Çünkü tama işleminde her iki taraftaki türde aynı olmalıdır.



## 2. Convert Class Kullanmak

**ushort**

değişkeni 16 bitlik işaretli tamsayı saklar.

```
ushort x = Convert.ToUInt16( Console.ReadLine() );
```



16 bitlik  
işaretsiz  
tamsayı



16 bitlik  
işaretsiz  
tamsayı



string döndürür

Yukarıdaki atama geçerli olacaktır.

## 2. Convert Class Kullanmak

**int**

değişkeni 32 bitlik işaretli tamsayı saklar.

int x =



32 bitlik  
işaretli  
tamsayı

Convert.ToInt32( Console.ReadLine() );



32 bitlik  
işaretli  
tamsayı



string döndürür

## 2. Convert Class Kullanmak

Sayısal Tip	Metod
decimal	ToDecimal(String)
float	ToSingle(String)
double	ToDouble(String)
short	ToInt16(String)
int	ToInt32(String)
long	ToInt64(String)
ushort	ToUInt16(String)
uint	ToUInt32(String)
ulong	ToUInt64(String)

## 2. Convert Class Kullanmak

Dizeyi tamsayıya dönüştürmenin başka bir yolu, **static Convert** sınıfını kullanmaktır. **Convert** sınıfı, temel veri türünü başka bir temel veri türüne dönüştüren farklı metotlar içerir. Tüm türler arasında dönüştürme yapabilir.

Convert sınıfı, farklı veri türlerinden int türüne dönüştürmek için aşağıdaki metotları içerir

```
Convert.ToInt16("100");    // returns short
Convert.ToInt16(null);     //returns 0
Convert.ToInt32("23300");  // returns int
Convert.ToInt32("1234", 16); // returns 4660 - Hexadecimal of 1234
Convert.ToInt64("1003232131321321"); //returns long
Convert.ToString(1234);    //returns "1234"
Convert.ToDouble("99.125"); //returns 99.125
Convert.ToSingle("3.14");  //returns 3.14f
Convert.ToChar(97);        //returns 'a'
```

## 2. Convert Class Kullanmak

Dizeyi tamsayıya dönüştürmenin başka bir yolu, **static Convert** sınıfını kullanmaktır. **Convert** sınıfı, temel veri türünü başka bir temel veri türüne dönüştüren farklı metotlar içerir. Tüm türler arasında dönüştürme yapabilir.

Convert sınıfı, farklı veri türlerinden int türüne dönüştürmek için aşağıdaki metotları içerir

// aşağıdaki dönüştürmeler hata verir

```
Convert.ToInt16(""); //throws FormatException
```

```
Convert.ToInt32("30,000"); //throws FormatException
```

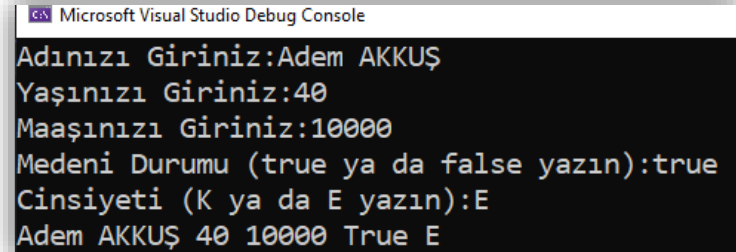
```
Convert.ToInt16("(100)"); //throws FormatException
```

```
Convert.ToInt16("100a"); //throws FormatException
```

```
Convert.ToInt16(2147483649); //throws OverflowException
```

# Convert Class Örnek Uygulama

```
string adi;  
byte yas;  
double maas;  
bool medeniDurum;  
char cinsiyet;  
Console.Write("Adınızı Giriniz:");  
adi = Console.ReadLine();  
Console.Write("Yaşınızı Giriniz:");  
yas =Convert.ToByte(Console.ReadLine());  
Console.Write("Maaşınızı Giriniz:");  
maas =Convert.ToDouble(Console.ReadLine());  
Console.Write("Medeni Durumu (true ya da false yazın):");  
medeniDurum=Convert.ToBoolean(Console.ReadLine());  
Console.Write("Cinsiyeti (K ya da E yazın):");  
cinsiyet=Convert.ToChar(Console.ReadLine());  
Console.WriteLine(adi+ " "+yas+" "+maas+" "+medeniDurum+" "+ cinsiyet);
```



Microsoft Visual Studio Debug Console

```
Adınızı Giriniz:Adem AKKUŞ  
Yaşınızı Giriniz:40  
Maaşınızı Giriniz:10000  
Medeni Durumu (true ya da false yazın):true  
Cinsiyeti (K ya da E yazın):E  
Adem AKKUŞ 40 10000 True E
```

# 3. TryParse() MetoduKullanmak

string `TryParse()` çağıran veri türüne dönüştürmek için tüm ilkel türler için metotlar mevcuttur. `string`'i bir tamsayıya dönüştürmek için önerilen yol budur.

Yöntem `TryParse()`, bir sayının dize gösterimini 16, 32 ve 64 bit işaretli tamsayı eşdeğerine dönüştürür. Dönüştürmenin başarılı mı yoksa başarısız mı olduğunu gösteren `bool` değerini döndürür ve bu nedenle asla istisnalar oluşturmaz

## Method Overloads

```
bool Int32.TryParse(string s, out int result)  
bool Int32.TryParse(string s, NumberStyle style, IFormatProvider provider, out int result)
```

# 3. TryParse() MetoduKullanmak

string `TryParse()` çağıran veri türüne dönüştürmek için tüm ilkel türler için metotlar mevcuttur. `string`'i bir tamsayıya dönüştürmek için önerilen yol budur.

Yöntem `TryParse()`, bir sayının dize gösterimini 16, 32 ve 64 bit işaretli tamsayı eşdeğerine dönüştürür. Dönüştürmenin başarılı mı yoksa başarısız mı olduğunu gösteren `bool` değerini döndürür ve bu nedenle asla istisnalar oluşturmaz

## Method Overloads

```
bool Int32.TryParse(string s, out int result)
bool Int32.TryParse(string s, NumberStyle style, IFormatProvider provider, out int result)
```



# 3. TryParse() Metodu Kullanmak

```
static void Main(string[] args)
{
    string numberStr = "123456";
    int sayi;

    bool donustumu = Int32.TryParse(numberStr, out sayi);

    if (donustumu)
        Console.WriteLine(sayi);
    else
        Console.WriteLine("Tür dönüşümü yapılamadı.");
}
```

Microsoft Visual Studio Debug Console

123456

# 3. TryParse() Metodu Kullanmak

```
static void Main(string[] args)
{
    string numberStr = "123456as";
    int sayi;

    bool donustumu = Int32.TryParse(numberStr, out sayi);

    if (donustumu)
        Console.WriteLine(sayi);
    else
        Console.WriteLine("Tür dönüşümü yapılamadı.");
}
```

Microsoft Visual Studio Debug Console

Tür dönüşümü yapılamadı.

# Değişkenlerin Faaliyet Alanları

```
0 references
static void Main(string[] args)
{
    {
        int a = 5;
    }
    Console.WriteLine(a);
}
```

**a** değişkeni ekrana yazdırıldığı yerde erişilebilir değildir.

# Değişkenlerin Faaliyet Alanları

```
0 references
static void Main(string[] args)
{
    int a = 5;
    {
        int b = 3;
        {
            Console.WriteLine(a);
            Console.WriteLine(b);
        }
    }
}
```

**a** ve **b** değişkenleri değişkeni ekrana yazdırıldığı yerde erişilebilirdir.