

8. DOSYA VE KLASÖR İŞLEMLERİ

8.1. DOSYALAMA İŞLEMLERİ (File)

- **Dosya Oluşturma: File.Create("Dosya Yolu")**

File.Create("C:\\Deneme\\mektup.txt") ya da

File.Create("@C:\\Deneme\\mektup.txt")

- **Dosya Silme : File.Delete("Dosya Yolu")**

File.Delete("C:\\Deneme\\mektup.txt") ya da

File.Delete("@C:\\Deneme\\mektup.txt")

- **Dosya Taşıma : File.Move("Kaynak yolu" , "Hedef yolu")**

File.Move("C:\\Deneme\\mektup.txt" , "D:\\Belgeler\\mektup.txt");

- **Dosya Kopyalama: File.Copy("Kaynak yolu" , "Hedef yolu")**

File.Copy("C:\\Deneme\\mektup.txt" , "D:\\Belgeler\\mektup.txt");

NOT: Taşırken ya da Kopyalarken dosyanın adı da değiştirilebilir.

Örnek: File.Copy("C:\\Deneme\\mektup.txt" , "D:\\Belgeler\\dilekce.txt");

- **Dosyanın var olup olmadığını kontrol etme : File.Exists("Dosya yolu ve adı")**

Bu metodun dönüşü true ya da false olur.

Örnek:

```
if(File.Exists("C:\\belgeler\\mektup.txt")==true)
    MessageBox.Show("Dosya Bulundu...");
else
    MessageBox.Show("Dosya Yok!");
```

- **Dosya içindeki metni okuma : File.ReadAllText("Dosya yolu");**

String yazı = File.ReadAllText("Dosya"); // Dönüşü string olur.

8.2. DOSYA BİLGİLERİNİ ALMA (FileInfo Sınıfı)

FileInfo dosya=new FileInfo ("Dosya yolu + adı");

- Klasör adını öğrenme : dosya.DirectoryName
- Dosyanın uzantısını öğrenme : dosya.Extension
- Dosya boyutunu öğrenme : dosya.Length
- Dosyanın tam yolunu öğrenme : dosya.FullName
- Dosyanın adını öğrenme : dosya.Name

8.3. KLASÖR İŞLEMLERİ (Directory)

- **Klasör Oluşturma : Directory.CreateDirectory("Yol");**

Directory.CreateDirectory("C:\\Evraklar");

- **Bir klasörün altındaki alt klasör adlarını öğrenme: Directory.GetDirectories("Yol");**

Dönüş bir dizi olacağı için bir dizi değişkeni oluşturarak bilgiler alınır.

String[] altklasörler = Directory.GetDirectories("C:\\Denemeler");

Alınan klasör adlarını listelemek için döngü kuruyoruz.

```
foreach(string klasör in altklasörler )
{
    listBox1.Items.Add(klasör);
}
```

8.4. KLASÖR BİLGİLERİNİ ALMA (DirectoryInfo Sınıfı)

```
DirectoryInfo klasor = new DirectoryInfo klasor("C:\\Denemeler");
```

- Klasör altında klasör açma

```
klasor.CreateSubDirectory("altklasor");
```

- Klasör altındaki dosyaların tüm bilgilerini öğrenme

```
FileInfo [ ] dosyalar = klasor.GetFiles();
```

```
foreach(FileInfo dosya in dosyalar)
```

```
{  
listbox1.Items.Add(dosya.Name+ "-" + dosya.CreateOnTime");  
}
```

- Klasör altındaki klasörlerin bilgilerini alma

```
DirectoryInfo[] altklasorler=klasor.GetDirectories()
```

```
Foreach(DirectoryInfo dosya in altklasorler)
```

```
{  
listbox1.Items.Add(dosya)  
}
```

- Klasörün bir üst dizinini öğrenme

```
klasor.Parent.Name
```

- Klasör bilgilerini belli kriterlere göre yapma

Örn : adı V ile başlayanlar.

```
Directory.GetDirectories("Yol", "V*");
```

- .exe dosyasının çalıştığı yolu öğrenmek için;

```
Directory.GetCurrentDirectory();
```

- Bir klasörün altındaki dosyaları listeleme

```
Diziye=Directory.GetFiles("Yol");
```

Dönüş bir dizi olur bir dizi değişkene atanması gerekir. Belli kriterlere göre listeleme yapılabilir.

- Bir klasörün kök (root) dizinini bulmak için;

```
Directory.GetDirectoryRoot("Klasörün yolu");
```

- Bulunulan dizinin Bir üst dizinini bulmak için;

```
Directory.GetParent("Yol");
```

- Sürücülere ulaşmak, sürücülerin listesini almak için ; (Dönüşü dizi olur)

```
String[] suruculer = Directory.GetLogicalDrives();
```

8.5. SÜRÜCÜ İLE İLGİLİ BİLGİLERE ULAŞMA (DriveInfo Sınıfı)

```
DriverInfo[] suruculer = new DriverInfo.GetDrivers();
```

```
foreach (DriverInfo surucu in suruculer)
```

```
{  
listbox1.Items.Add(surucu.Name + surucu.TotalSize + surucu.TotalFreeSpace  
+ surucu.AvailableFreeSpace)  
}
```

```
.DriveFormat //Dosya Formatı
```

```
.Driver.VolumeLabel // Sürücü Etiketi
```

8.6. DOSYAYA BİLGİ YAZMA (StreamWriter Sınıfı)

```
StreamWriter yaz = new StreamWriter("Dosya yolu + adı", true/false);
```

Dosya yolu belirtildikten sonra **true** kullanılırsa daha önce var olan dosya içerisinde bilgiler var ise yeni bilgiler bu dosyanın devamına eklenir. Dosya içindeki bilgiler silinmez. Herhangi bir şey belirtilmezse ya da **false** olarak belirtilirse, her açıldığında dosya yeni oluşturuluyormuş gibi işlem görür ve içindeki bilgiler silinir.

```
yaz.Write(" Bilgi ");           // Dosyaya metni yazar aynı satırda kalır. Bir sonraki metin aynı satırdan devam eder.
yaz.WriteLine(" Bilgi ");      // Dosyaya metni yazar bir alt satıra geçer. Bir sonraki metin alt satırdan devam eder.
yaz.Close();                  // Dosyayı kapatır. Dosya kapatılırken de dosyaya metni kaydetmek için gereken metot (Flush) çağrılır ve metin dosyaya kaydedilmiş olur.
```

- Metni dosyaya kaydet;

```
yaz.Flush();                  // Kendinden önce verilmiş olan metin yazma komutlarının çalışarak dosyaya kaydedilmesini sağlar.
```

- Metni dosyaya otomatik kaydet;

```
yaz.AutoFlush()=true;       // Program içerisinde verilen tüm yazma komutlarını çalıştırarak metinleri dosyaya otomatik kaydeder.
```

8.7. DOSYADAN BİLGİ OKUMA (StreamReader Sınıfı)

```
StreamReader oku = new StreamReader("Dosya yolu + adı");
```

- Dosyadaki tüm metni okuma

```
String veriler = oku.ReadToEnd();
textBox1.text=veriler;
oku.Close()
```

- Dosyadan satır satır okuma

```
textbox1.text=oku.ReadLine();
```

//Belli bir satırdan sonrasını okuma ya da satır satır ardı ardına okuma yapmak için aşağıdaki yöntem kullanılır. Burada **EndOfStream** dosya sonuna gelinip gelinmediğini tespit eder. ! – değil (not) işlemi yapar.

```
While(!oku.EndOfStream)
{
textBox2.text +=oku.ReadLine()+"\r\n";
}
```

8.8. DİLLERİN METİN KODLARI (Encoding ve EncodingInfo Sınıfları)

- Kullanılan tüm dil kodlarını listelemek için;

```
EncodingInfo[] kodlar = Encoding.GetEncodings();
foreach(EncodingInfo kod in kodlar)
{
listBox1.Items.Add(kod.CodePage + kod.DisplayName + kod.Name)
}
```

- Metinleri formatlı okumak için;

```
Encoding trk = Encoding.GetEncoding("Windows-1254");
StreamReader oku = new StreamReader("Dosya", trk);
```