Konular

- Struct
- Exception Handling
- Enumeration
- Attributes
- Reflection

Struct (Yapı)

- C#, bir yapı değer tipli (value type) bir veri tipidir.
- Özellikle ilişkili farklı veri tiplerini tek değişkende tutmakta yardımcı olur. (Sınıfa benziyor.)
- Struct anahtar kelimesi ile bir yapı kurulur.
- Yapılar kayıtların içeriklerini belirlemek için kullanılır. Örneğin bir kütüphanedeki kitaplar (books) için bir uygulama düşünelim. Bu uygulamada aşağıdaki içerikleri ön plana çıkar:
  - Title (Başlık)
  - Author (Yazar)
  - Subject (Konu)
  - Book ID (Kitap ID)

```
struct Books
{
    public string title;
    public string author;
    public string subject;
    public int book_id;
};
```

Struct

```
struct Books
{
    public string title;
    public string author;
    public string subject;
    public int book_id;
};
```

```
public class testStructure
{
    public static void Main(string[] args)
    {

        Books Book1;    /* Book1 tanımlandı */
        Books Book2;    /* Book2 tanımlandı */

        /* Book1 özellikleri */
        Book1.title = "C Programming";
        Book1.author = "Nuha Ali";
        Book1.subject = "C Programming Tutorial";
        Book1.book_id = 6495407;

        /* Book2 özellikleri */
        Book2.title = "Telecom Billing";
        Book2.author = "Zara Ali";
        Book2.subject = "Telecom Billing Tutorial";
        Book2.book_id = 6495700;

        /* Book1 içeriği */
        Console.WriteLine("Book 1 title : {0}", Book1.title);
        Console.WriteLine("Book 1 author : {0}", Book1.author);
        Console.WriteLine("Book 1 subject : {0}", Book1.subject);
        Console.WriteLine("Book 1 book_id :{0}", Book1.book_id);

        /* Book2 içeriği */
        Console.WriteLine("Book 2 title : {0}", Book2.title);
        Console.WriteLine("Book 2 author : {0}", Book2.author);
        Console.WriteLine("Book 2 subject : {0}", Book2.subject);
        Console.WriteLine("Book 2 book_id : {0}", Book2.book_id);

        Console.ReadKey();
    }
}
```

C# Struct Özellikleri

- C#'ta Yapılar C ve C++ gibi dillere göre farklılıklar gösterir. C# Yapılarının özellikleri şöyledir:
- Yapılar metotlar (methods), alanlar (fileds), özellikler (properties), operator metotları (operator methods) ve olaylar (events) içerebilir.
- Yapılara constructor tanımlanabilir, ama destructor (yıkıcı) tanımlanamaz. Ancak, yapılara geçerli bir constructor tanımlanmaz. İllaki parametreli bir constructor olmalı.
- Bir struct yaratmada new operatörü kullanıldığı zaman, yaratılmış constructor çağrılmış olur. Sınıflardan farklı olarak, new operatörü kullanmadan da struct'ları kullanmak mümkündür.
- Eğer new operatörü kullanılmazsa, alanlar tanımlanmamış kalır ve nesne tüm alanlar tanımlanmadan kullanılamaz.

C# Struct Özellikleri

- Structures can have defined constructors, but not destructors. However, you cannot define a default constructor for a structure. The default constructor is automatically defined and cannot be changed.
- Sınıflardan farklı olarak yapılar diğer sınıf veya yapılardan kalıtım uygulanamaz. Diğer bir deyişle, yapılar diğer yapılara veya sınıflara base olarak kullanılamaz.
- Bir yapı bir veya birden fazla interface'e uygulayabilir. (DateTime'ı hatırla)
- Yapı üyeleri abstract, virtual veya protected olarak belirlenemez.

Struct Örneği

```
public struct ogrenci_s
{
    public string AdSoyad;
    private int _Not;

    public ogrenci_s(string a, int n)
    {
        AdSoyad = a;
        _Not = n;
        //Not değişkeni kullanılamıyor ama class kullanılabiliyor...
    }

    //structta parametresiniz constructor olmaz.
    //public ogrenci_s(){
    //    AdSoyad = "Test";
    //    _Not = 0;
    //}

    public int Not {
        get { return _Not; }
        set {
            _Not = value;
            if (_Not < 0)
                _Not = 0;
            if (_Not > 100)
                _Not = 100;
        }
    }

    public void Bilgi()
    {
        Console.WriteLine("Adı Soyadı: (struct)" + AdSoyad + " : " + Not);
    }
}//struct sonu
```

Sadece public ve private erişim belirteçlerini destekler.

Constructor

Property

Property

Class vs. Struct

```
public class ogrenci_c
{
    public string AdSoyad;
    private int _Not;
    //Sadece class için tanımlanabilir
    public ogrenci_c(){
        AdSoyad = "Guest";
        Not = 0;
    }

    //struct ile aynı
    public ogrenci_c(string a, int n){
        AdSoyad = a;
        Not = n;//struct'tan farklı olabiliyor
    }

    //struct ile aynı
    public int Not
    {
        get { return _Not; }

        set {
            _Not = value;
            if (_Not < 0)
                _Not = 0;
            if (_Not > 100)
                _Not = 100;
        }
    }
    //struct ile aynı
    public void Bilgi()
    {
        Console.WriteLine("Adı Soyadı: (Class) " + AdSoyad + " : " + Not);
    }
}
```

- Struct'tan farklı olarak Class'ta boş constructor tanımlanabilir.
- Erişim belirteçlerinin tümü kullanılabilir.

Struct ve Class Yaratma

- Main bölümünden önceki struct ve class'ları çağıralım.

```
class Program
{
    static void Main(string[] args)...
}
```

```
ogrenci_s _s1 = new ogrenci_s(); //boş constructor yok ama çalışıyor...
_s1.AdSoyad = "Deneme Öğrencisi";
_s1.Not = 45;
_s1.Bilgi();
ogrenci_s _s2 = new ogrenci_s("Deneme Öğrencisi 2", 55);
_s2.Bilgi();
ogrenci_s _s3;
_s3.AdSoyad = "1111";
Console.WriteLine(_s3.AdSoyad);
//_s3.Not = 34;
//_s3.Bilgi(); //??? struct ile çalışmaz new ihtiyaç var
//strcutları dizi olarak tanımlama
ogrenci_s[] _sdizi = new ogrenci_s[5];
_sdizi[0].AdSoyad = "Deneme 1";
_sdizi[0].Not = 450;
_sdizi[1].AdSoyad = "Deneme 2";
_sdizi[1].Not = -5;
_sdizi[2].AdSoyad = "Deneme 3";
_sdizi[2].Not = 55;

for (int i = 0; i < 5; i++)
    _sdizi[i].Bilgi();
```

```
public string AdSoyad;
private int _Not;
```

Adsoyad çağırılabilir ama _Not çağırılamaz çünkü private. _Not'u çağırabilmek için constructoru new ile yaratmak gerek

Struct İşlemleri

```
ogrenci_c _c1 = new ogrenci_c(); //eğer constructor tanımlamazı yoksa çalışmaz
_c1.AdSoyad = "Deneme Öğrencisi";
_c1.Not = 45;
_c1.Bilgi();

ogrenci_c _c2 = new ogrenci_c("Deneme Öğrencisi 2", 65);
_c2.Bilgi();

/*ogrenci_c _c3;
_c3.AdSoyad = "1111";
Console.WriteLine(_c3.AdSoyad);*/
//Kesinlikle çalışmaz... Nesne yaratılmadığı için (New kullanılması lazım)
//class'ımız statik olsaydı new işlemine gerek olmazdı.
//geçmiş örneklere bakın...
```

Class İşlemleri