# C# Notları

## Terminoloji:

* Namespace: Class topluluğu
* Decimal, float, double, int, string (küçük harf olanlar kullanılmalı)
* paramType, Camel case ile yazılmalı
* Adlandırmada type ön eki olmamalı, strName
* SabitDeğerleri, BÜYÜK harf yerine pascal yazım tarzında yaz
* Pre defined değişken tiplerini kullan (“string”, “int”), “String”, “Int32” kullanma
* Sadece uzun veya kısa süreli kullanılan değişken tiplerinde “var” kullan
* “Var” değerleri is ile kontrol edilir

## Tip Kontrolleri:

* Object: Compile sırasında
* Dynamic: Runtime sırasında

## Tip Değiştirme

* <var>.To<Type>
* Convert.To<Type>

## Değişkenler

* <type> a, b, c;
* <type> a=1, b=2;
* <type>? a = null;
* <name> = <name> ?? <val>
  + Null değilse değer atama
  + Null ise <val> atanır
  + ?? thow Exception ile kullanılıyor
* Var
  + Sadece uzun değişken tiplerinde tercih edilmelidir
  + Kısa süreli kullanılan değişkenler için de tercih edilebilir
* Özellikleri
  + const: Sabit
  + static: Statik
  + readonly: Sadece okunabilir
  + internal

## Fonksiyonlar

* <type> Func(<out | ref> <param>)
  + Out ile değişkenlere değer atanır, out ile çağırılır
  + Settings.GetValues(out a, out b);
  + Ref ile değişkenlerin adresleri fonksiyona aktarılır, ref ile çağırılır
* <type> Func(params <type>[] <name>)
  + , ile ayrılan argümanları verebilmeyi sağlar
* Method isimleri büyük
* Default olarak private olur
* Func(int) func(bool) gibi Polymorophisim destekler
* Özellikler
  + Override, var olan metodu yeniden tanımlama
  + Virtual ile için sonradan doldurulacak metotlar tanımlanır public virtual int() {}
  + Extern, harici kaynak kodlarında çalışan fonksiyonlar için tanımlanır
* Operator overloading
  + Operatör <operator> {}
  + Public static Box operatör + (Box a, Box b) {}

## Döngüler

* For (<init>;<condition>;<inc>){}
* For(;;){} İnfinite loop
* While (<condition>) {}
* Do {} while (<condition>)
* Foreach (<type> <name> in <array>){}

## Diziler

* <type>[] <name> = new <type>[<size>]
* <type>[]<name> = {<val>, <val>}
* Özellikleri
  + Length, uzunluk bilgisi
  + Rank, boyut bilgisini verir
  + Sort, sıralar
* <array> = <array2>
  + Adres ataması yapılır
  + <array2> değişirse <array1> de değişir
* <type>[ , ] = {{}, {}}
  + Çok boyutlu dizi tanımlama
* Func(<type>[] <name>)
* Func(params <type>[] <name>) {}
  + Func (1, 2, 3, 4, ….)

## String

* Char[] <name> = {‘h’, ‘e’}
* String <name> = new string(<name>)
* $”{<var>} formatlama”
* String.Join(<delimeter>, <array>)
* Özellikler
  + Chars(<index>)
  + Length()
  + <int> Compare(<string>)
  + Contains(<str>)

## Struct

* Struct kullanılır, Class gibi ama değer odaklıdır
  + <var> = <var2> durumunda değerleri kopyalanır, bağımsızlardır
  + Adres odaklı olanlarda adresler kopyalanır, değişikler diğerini de değiştirir
* Metotları da olabilir
* Constructure ve Deconstructure olmaz

## Enum

* Enum <name> { <val>, <val> … };
  + Noktalı virgül gerektirir
* Enum Days {Mon, Sun}; (int)Days.Mon == 1;
  + Takılama işlemleri gerçekleşebilir

## Class

* Default olarak “İnterval” olur
* <Class> : <Class2> ile extend edilir
* “Namespace” içlerine yazılırlar
* Namespace BoxApp { class Box {} class BoxTester {}}
* <Access> <class\_name>() {}
  + Constructure yapısı
  + Deconstructure için ~ kullanılır
    - Uygulama kapanması durumunda tetikle
* Özellikler:
  + Sealed, extend edilemez
  + Abstract, objesi oluşturulamaz