OOP bir yaklaşımdır.

Nesne = class

Gerçek hayatın karşılığını sunuyor

Referans türlü değerlerdir. Stack de heap e erişen referanslar tanımlayıp(new) ardından stackden heap e erişiyoruz.

OOP de bir obje oluşturabilmek için oncelikle o objenin modellenmesi gerekir.

Bir objenin modelinin tanımını oluşturabilmek için sınıf yapısını kullanırız.

Classlarda nesnelerin ortak alan tanımlanmaları yapılır.

Classs lar bir referans türdür.

Class lar namespace içerisinde-dışarısında ve class içersinde oluşturulur.(içi içe class). Namespace dışından erişmek için kendi namespaceini de kullanıyoruz.

FIELD nesne içerisinde veri tuttuğumuz alandır.

Class ın içerisinde tanımlanmış değişkendir.

Sadece field larda default değer atanır.

Property bir metottur. sette property e değer atanır ve değer okunmak istendiğinde get metodu

kullanılır .

propertylerin bu durumuna kapsülleme Encapsulation denir.

NESNE NEDİR?

Nesne canlı bir varlıktır. Complex type dır. Sadece class ile nesne oluşturabiliriz.

REFERANS NEDİR?

Referans ramin stack bölgesinde tanımlanan heap bölgesindeki nesneleri işaretleyen değişkenlerdir.

Nesneler sadece class lardan oluşturulur Heap bölgesinde. Ama referans etmek için Interface ve Abstarct Class dan da stack bölgesinde değişken oluşturulup Polymorphisim kavramı ile heap bölgesindeki nesnelerimizi işaret edebiliriz.



Nesne somutlaştırılmadığında NullReferance hatası verir.

Heap te referanssız nesneler C# ta Garbage Collector tarafından temizlenmektedir.

Object Initializer ile nesne ilk oluşturulurken değer atama. Süslü parantezi açıp memberlarına değer atanır. Sadece field ile propertylere değer atanır burada metotlar zaten değer atanmaz bir işlemdir. Metotlar çağrılmalıdır.

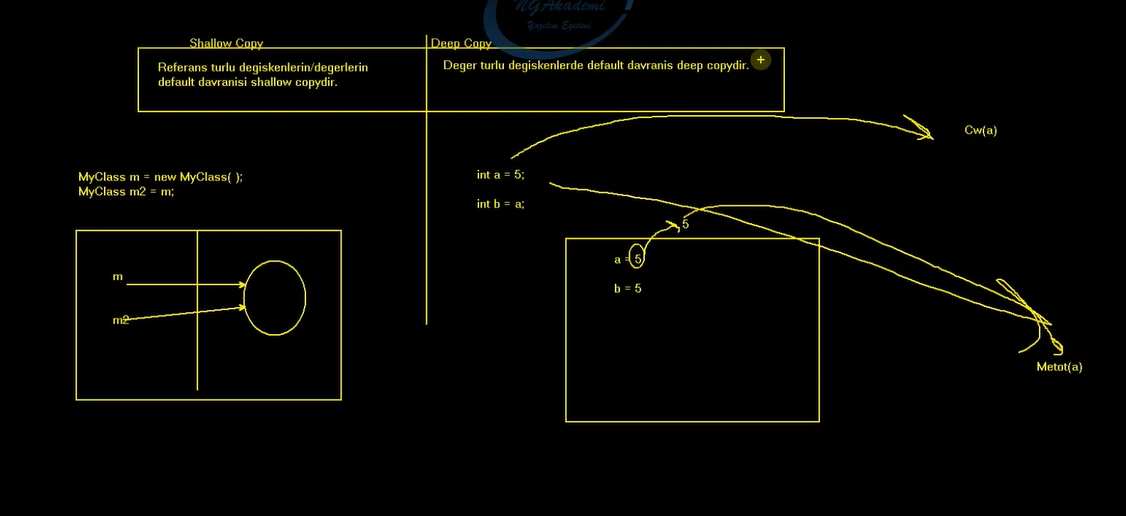
Deep And Shallow Copy

Deep copy de veri çoğaltılır. Shallow Copy de veri 1 tanedir o referans edilir.

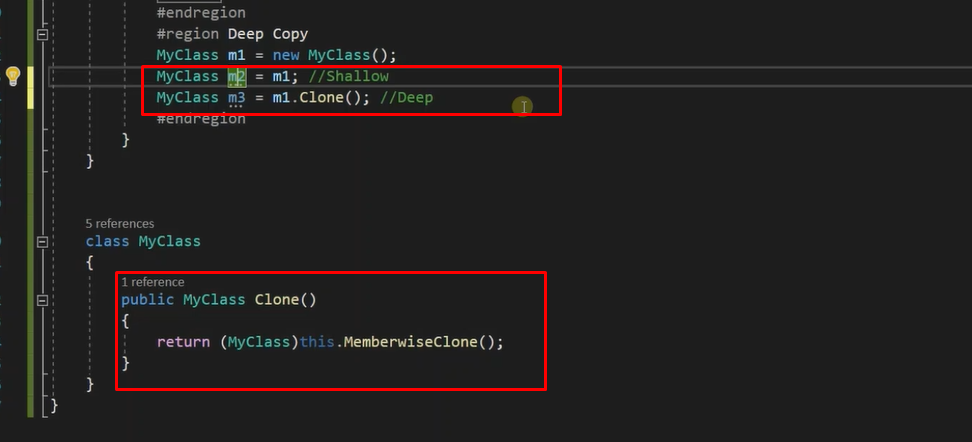
Shallow Copy

Referans türlü değişkenlerin default durumu shallow Copy dir.

Burası baya önemli



MemberwiseClone o nesne üzerinden üretilmiş olan nesneyi klonlamamızı sağlayan metottur.

 Bir nesneden Deep Copy yapmak istersek o nesneye bunu yapacağımızı Clone metotu ile söylememiz gerekiyor. Zaten bu Clone metotu bu işi yapıyor.

ENCAPSULATION

Nesnelerimizin fieldlarına kontrollü bir şekilde erişmek için kullanırız.

Eskiden encapsulation field isminin get ve set metotları ayrı ayrı yazılırdı. Ama artık ile bu işlemi gerçekleştriyoruz.

RECORDS

Bir nesnenin bütün değerleri değişmez olacaksa kullanılır. Sadece bir tanesi değişmez olduğunda biz init only prop ile bu işi yapıyoruz. 