C# nedir, .Net Framework nedir?

C# nedir?

- C# , C/C++ ve Java dillerinde türetilmiş,bu dillerin dezavantajlarının elenip iyi yönlerinin alındığı, güçlü basit, esnek, tip-güvenli(type-safe,tür dönüşümlerindeki önlemler,örn: byte=byte+byte olamaz, int=byte+byte).Net platformu için hazırlanmış %100 nesne yönelimli bir dildir.

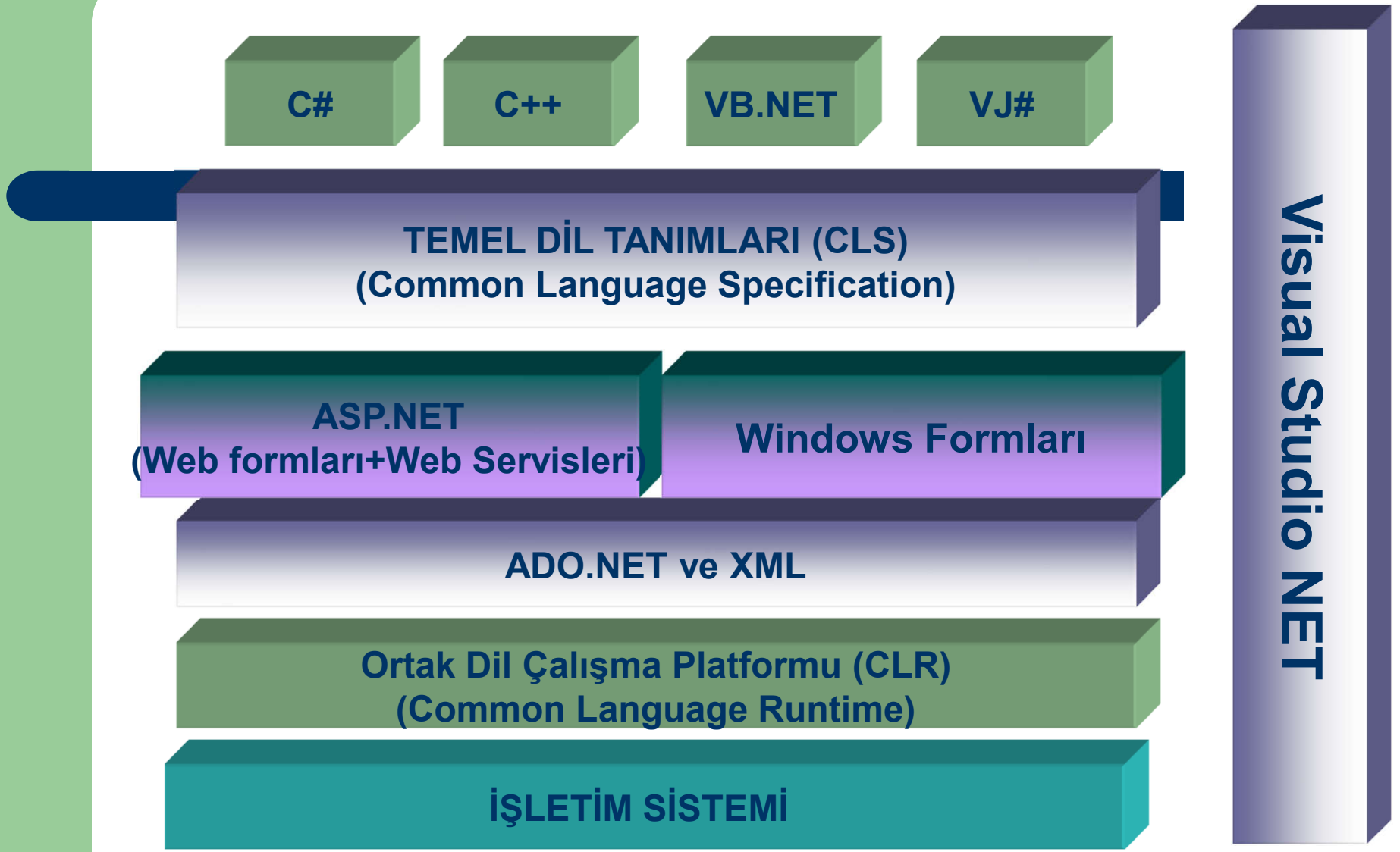
C# nedir?

Script Dilleri	Javascript,VBScript
Yüksek Düzeyli Diller	Vbasic Delphi
Orta Düzeyli Diller	C# Java
Düşük Düzeyli Diller	C/C++
Assebmly(makine dili)	Assembly

C# kullanım alanları

- Konsol uygulamaları
- Windows için program yazma
- Web formları uygulaması
- Web servisleri
- DLL yazma

.NET Bileşenleri



.NET Bileşenleri

Java'dan önce, geliştirilen yazılımlar *direkt* olarak *makine koduna* çevrilirdi.

Java ile program kodu önce *byte code'a* çevrilir. JVM(java virtual machine) bu kodu işletim sisteminin istediği koda çevirir.

.Net içinde çalışma mantığı benzerdir, .NET kodu ilk önce *IL'ye (Intermediate Language-Aradil)* derler, bu IL kodu çalıştırılmak istendiğinde CLR ,JIT derleyicilerini kullanarak kodu makine diline çevirir.

.NET Bileşenleri

CLR makine diline çevrilmiş bu kodu önbellekte tutar, bu performans artışına sebep olurken diğer taraftan sistem hafızasında küçümsenmeyecek yer işgal eder.

Ortak Dil Çalışma Platformu(CLR) (Common Language Runtime)

- CLR .NET altyapısında programların çalışmasını kontrol eden ve işletim sistemi ile programımız arasında yer alan arabirimdir. Normalde yazılan kodlar makine diline çevrilir ve işletim sistemi ile direkt bağlantı kurup çalışırdı.
- Eğer platformdan bağımsız bir ortam istiyorsak, ihtiyaç duyulan şey CLR dir, hangi platformda iseniz (Linux,Mac,Windows) CLR bu noktada devreye girer ve .NET programlarının farklı platformlarda işletim sistemine göre çalıştırır.

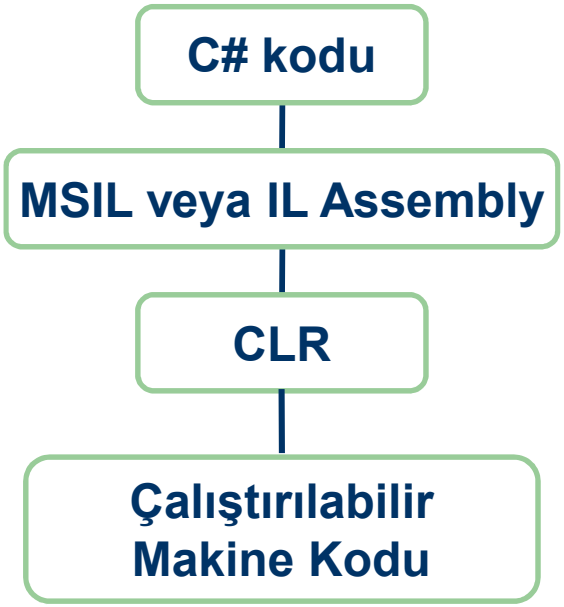
Ortak Dil Çalışma Platformu(CLR) (Common Language Runtime)

- Eğer çok sayıda platform olduğunu düşünürsek, programların bunlar için ayrı ayrı yazılıp derlenmesi gerekir. Bu durum imkansız gibidir.
- Bu durumda çözüm ortak bir ara dil kullanmak ve her bir platform için bu ara dile çevrilmiş kodu çalıştırmaktır.

Ortak Dil Çalışma Platformu(CLR) (Common Language Runtime)

- Managed Code(Yönetilen Kod): Yalnızca CLR yardımları altında çalışan koddur. Bir örnek vermek gerekirse ; Windows'ta çalışan farklı işlemlere sahibiz. Uygulamaların izlemesi gereken kural Windows genel kurallarına uymalarıdır. Managed kodda CLR tarafından Windows'un yaptığı şekilde çalıştırılan koddur.

Ortak Dil Çalışma Platformu(CLR) (Common Language Runtime)



.NET derleme ve çalıştırma

Aradil (IL veya MSIL) (Intermediate Language)

- Herhangi bir C++ veya Vbasic kodu direkt makine koduna çevrilirdi ve çalıştırılırdı. Makine diline çevrilen programlar, işlemciye ve işletim sistemine özel olarak derlenirdi.

  Örn: a ve b sayılarının toplamı için kullanılan bir C++ programı Intel işlemciler için farklı SunSparc işlemciler için farklı derlenirdi.
- Fakat .NET ortamında kodumuzu derlediğimizde elde ettiğimiz IL (aradil) kodu işlemciye bağlı olmaz

Aradil (IL veya MSIL) (Intermediate Language)

- IL içerisinde değişken tanımları, değişkenlerin nasıl saklanacağı, metotların nasıl çalıştırılacağı, aritmetik ve mantıksal işlemler, bellek kullanımı gibi birçok işin nasıl yapılacağı açıklanır.
- Artık IL ile oluşturduğumuz kodumuzun çalıştırılabilir bir program olması için derlememiz gerekiyor. Bunun için JIT (Just in Time) derleyici kullanılır.

JIT Derleyiciler (Just in Time)

- C# ile IL'ye derlediğimiz programı çalıştırırken JIT derleyicileri devreye girerler. Bu derleyiciler programın çalıştırıldığı sistemin ve işlemcinin anlayabileceği makine kodunu oluştururlar.
- Windows ortamı için 3 çeşit JIT mevcuttur

1. Normal JIT
2. Pre-JIT
3. Eco-JIT

JIT Derleyiciler (Just in Time)

- Normal JIT : IL kodu makine koduna çevrilirken default(varsayılan) olarak kullanılan derleyicidir. IL kodunu orijinal makine koduna çevirir ve *önbellekte* tutar. Örneğin ; program içindeki bir derlenmiş bir metot program akışı içinde tekrar çağrılırsa önbellekten çekilir.
- Pre-JIT: Tüm program kodunu makine koduna çevirip sonra çalıştıran JIT. Fazla hafıza gerektirir. Programın daha hızlı çalışmasını sağlar.

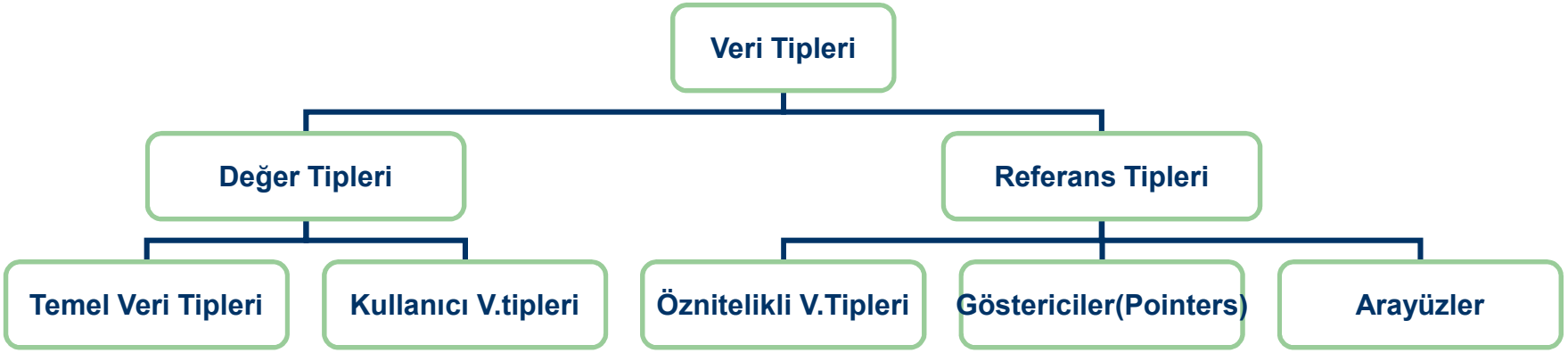
JIT Derleyiciler (Just in Time)

- Eco JIT : Kısıtlı hafıza ve önbellekli sistemlerde .NET programlarının daha iyi çalışmalarını sağlamak için kullanılan derleyicidir.

CTS (Common Type System)

- Bütün veri tiplerinin tanımlı olduğu bir sistem olarak düşünebiliriz. C# dilindeki veri türleri aslında CTS'deki veri türlerine karşılık gelen arayüzlerdir.
- CTS sayesinde .NET platformu için geliştirilen bütün diller aynı veri tiplerini kullanırlar, tek değişen türlerin tanımlama yöntemi ve söz dizimidir.Geliştirilen bir nesnenin diğer dillerde de sorunsuz çalışmasını garanti eder.

CTS (Common Type System)



CTS veri tipleri şeması

CLS (Common Language System) (Temel Dil Tanımları)

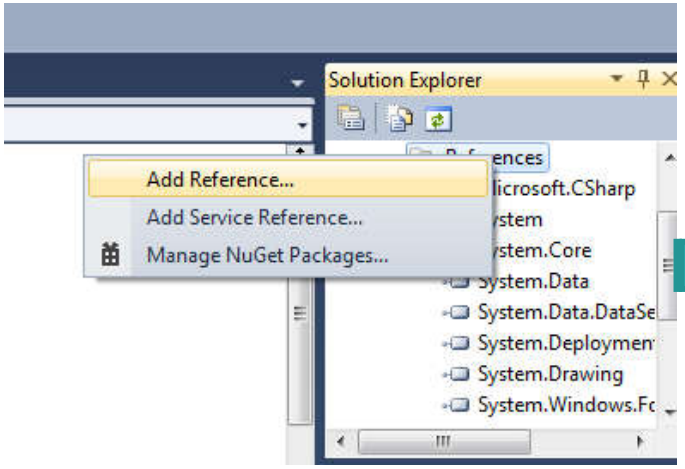
- Dil derleyicisinin uyması gereken kuralları içerir.

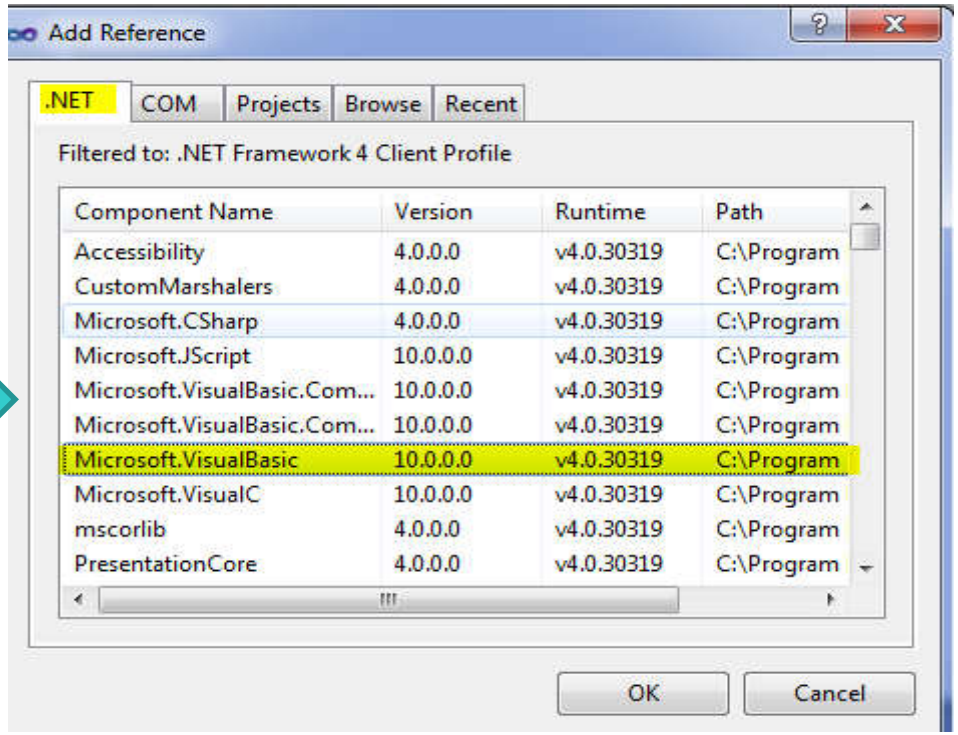
CLS (Common Language System) (Temel Dil Tanımları)

- Dil derleyicisinin uyması gereken kuralları içerir.
- CLS' ye uyan bir dille yazılmış kod ile diller arası iletişim sağlanmış olur
- Örneğin Vbasic.Net içindeki InputBox komutunun C# içinde çağrılması.

CLS (Common Language System) (Temel Dil Tanımları)

- **Projemizin «references» kısmından «Add Reference» ile .NET kısmından VisualBasic referansını eklemeliyiz.**





CLS (Common Language System) (Temel Dil Tanımları)

- Artık proje kodumuzda gerekli namespace'i ekleyerek Vbasic.Net içindeki InputBox komutunu C# içinde kullanabiliriz..

```
using Microsoft.VisualBasic; //başlık kısmına eklenecek namespace

string giris = Microsoft.VisualBasic.Interaction.InputBox("Giris",
"Başlık", "Varsayılan");
```

Namespaces and .NET Class Library (İsim Alanları Sınıf Kütüphanesi)

- Programcıların işlerini kolaylaştırmak için bir takım hazır kütüphaneler vardır fakat C# dili ile gelen hazır bir takım kütüphaneleri yoktur.
- Bunun yerine Framework dediğimiz altyapıda bir takım temel türler ve sınıflar mevcuttur. Bu sınıf ve türleri organize edebilmek için Namespace kavramı kullanılır.

Namespaces and .NET Class Library (İsim Alanları Sınıf Kütüphanesi)

- C# dilinde .NET Framework sınıf kütüphanesi içerisindeki veri türleri ve sınıflar "**using**" sözcüğü ile kullanılır. Diğer dillerde de bu isim alanları farklı şekillerde derleyiciye bildirilir.
- Fakat temelde yapılan iş, .NET Framework Sınıf Kütüphanelerini kullanma hakkı kazanmaktır.
- Program geliştirirken sınıfların birbiri ile ilgili olanlarını aynı isim alanı içine koymalıyız.

Namespaces and .NET Class Library (İsim Alanları Sınıf Kütüphanesi)

- **System** isim alanı : .NET çalışırken gerekli temel sınıfları içerir. Ayrıca diğer tüm sınıf kütüphaneleri de bunu içinde kümelenmiştir. System hiyerarşinin tepesinde bulunur.
- Örneğin tüm veritabanı işlemleri için kullanılacak sınıf kütüphanesi "**System.Data**" dır.
- Bu sınıf kütüphanesi içindeki SQL ile işlemler için "System.Data.SqlClient" isim alanı mevcuttur.

Namespaces and .NET Class Library (İsim Alanları Sınıf Kütüphanesi)

- ***System.Net*** : HTTP ve ağ protokolleri için kullanılır.
- ***System.Xml*** : XML verileri ile çalışmak için
- ***System.IO*** : dosyalara bilgi girişi, dosyadan bilgi okuma, I/O işlemleri için kullanılır.
- ***System.Windows.Forms***: Windows tabanlı uygulamalarda kullanılan zengin grafik arabirimi kontrollerini içerir.