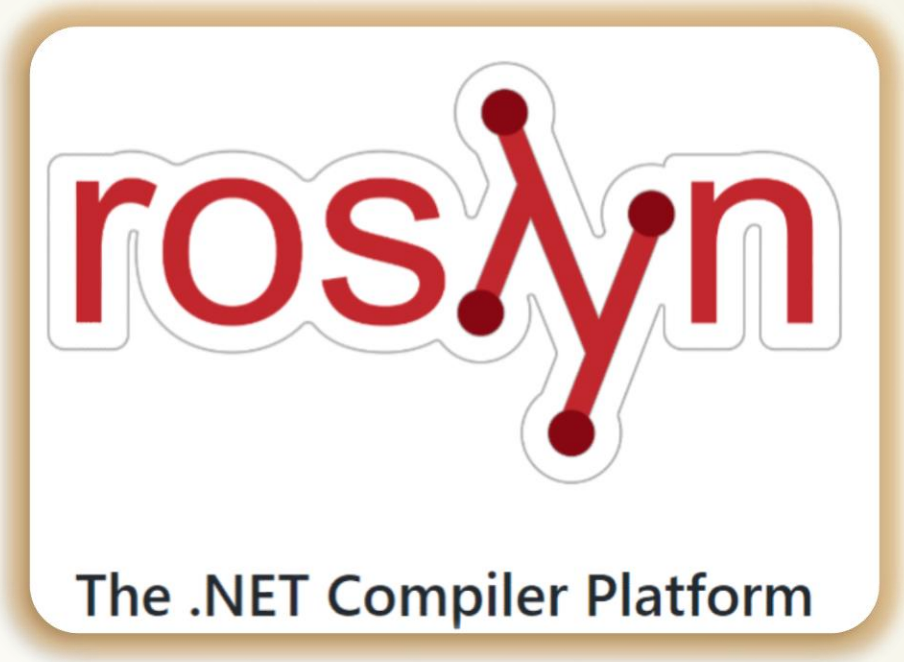
Akademija tehničko-vaspitačkih strukovnih studija

.NET tehnologije

Prof. dr Zoran Veličković, dipl. inž. el.

2022/23.

Prof. dr Zoran Veličković, dipl. inž. el.



.NET kompajler, MSIL kod, JIT kompajler i .NET Web servisi (2)

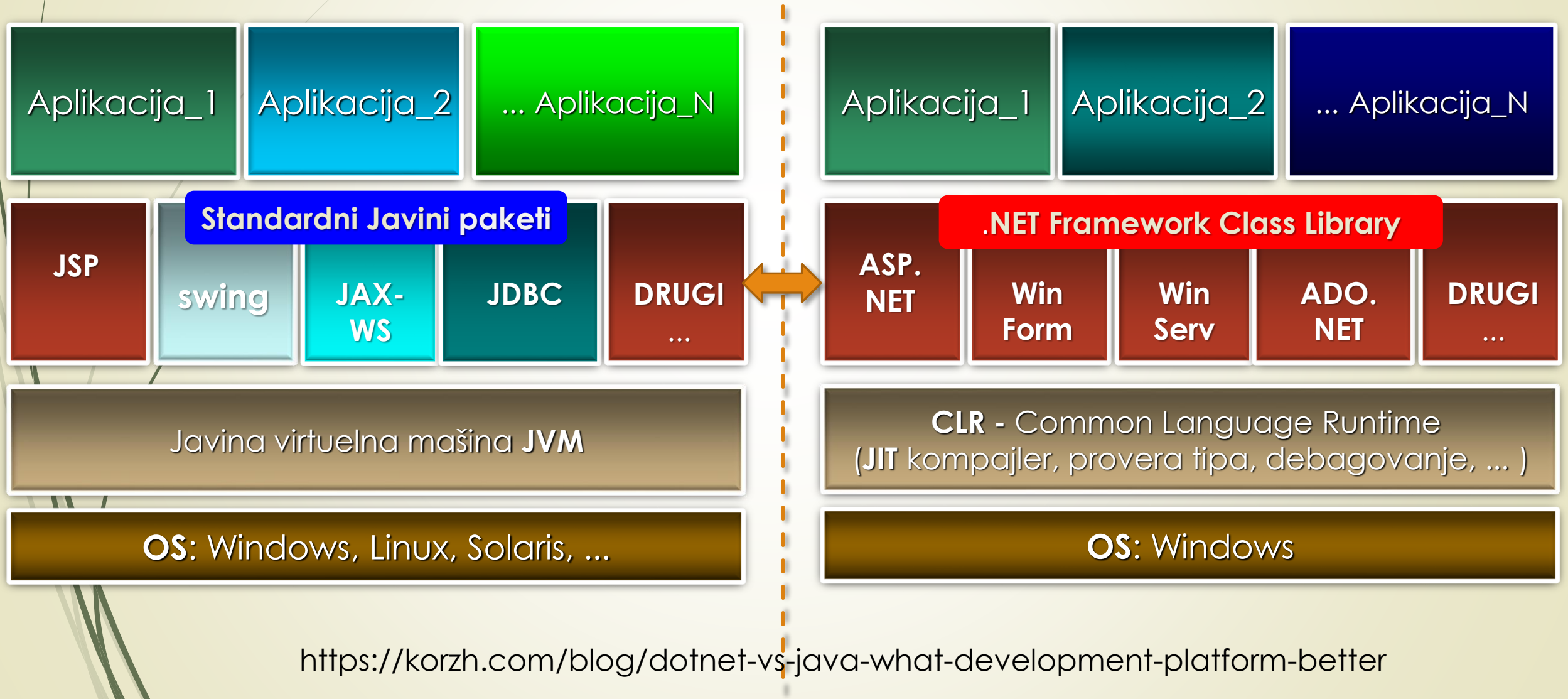
Sadržaj

- **OO Jezici: C# - Java**
  - Kompatibilnost .NET jezika
  - Zajednička specifikacija za .NET jezike - **CLS**
  - Zajednički tipovi .NET-a - **CTS**
- **Kompajliranje programskih kodova u .NET-u**
  - Programski sklopovi
    - Metapodaci sklopa
  - Programski moduli u .NET-u
  - JIT Kompajler
    - Primer rada JIT Kompajlera
- **Izvršavanje programskih kodova i .NET**
  - Upravljani kodovi i .NET
  - Neupravljani kodovi i .NET
- **Web servisi u .NET-u**
  - Arhitektura Web servisa
  - Komponente Web servisa
  - Hijerarhija Web servisa

OO Jezici: C# - Java

- Osnove **OBJEKTNO ORIJENTISANIH** (OO) programskih jezika su obrađene na kursu iz Objektno orijentisanog programiranja na primeru Jave!
- **Java** i **C#** su **OO** jezici naslednici C/C++-a!
- Da li postoji strukturalna razliku između Jave i C# (pogledajte naredne slajdove)?
- **CLR**, odnosno **JVM**, obezbeđuje osnovne servise:
  - Prevođenje koda;
  - Alokacija memorije;
  - Upravljanje nitima;
  - Sakupljanje otpadaka ...
- Sistem zajedničkih tipova **CTS** (engl. *Common Type System*)
  - Obezbeđuje STROGU PROVERU **bezbednost TIPOVA** (o CTS-u ali i o CLS-u nešto kasnije).
  - Sprovodi **bezbednost pristupa** koda.

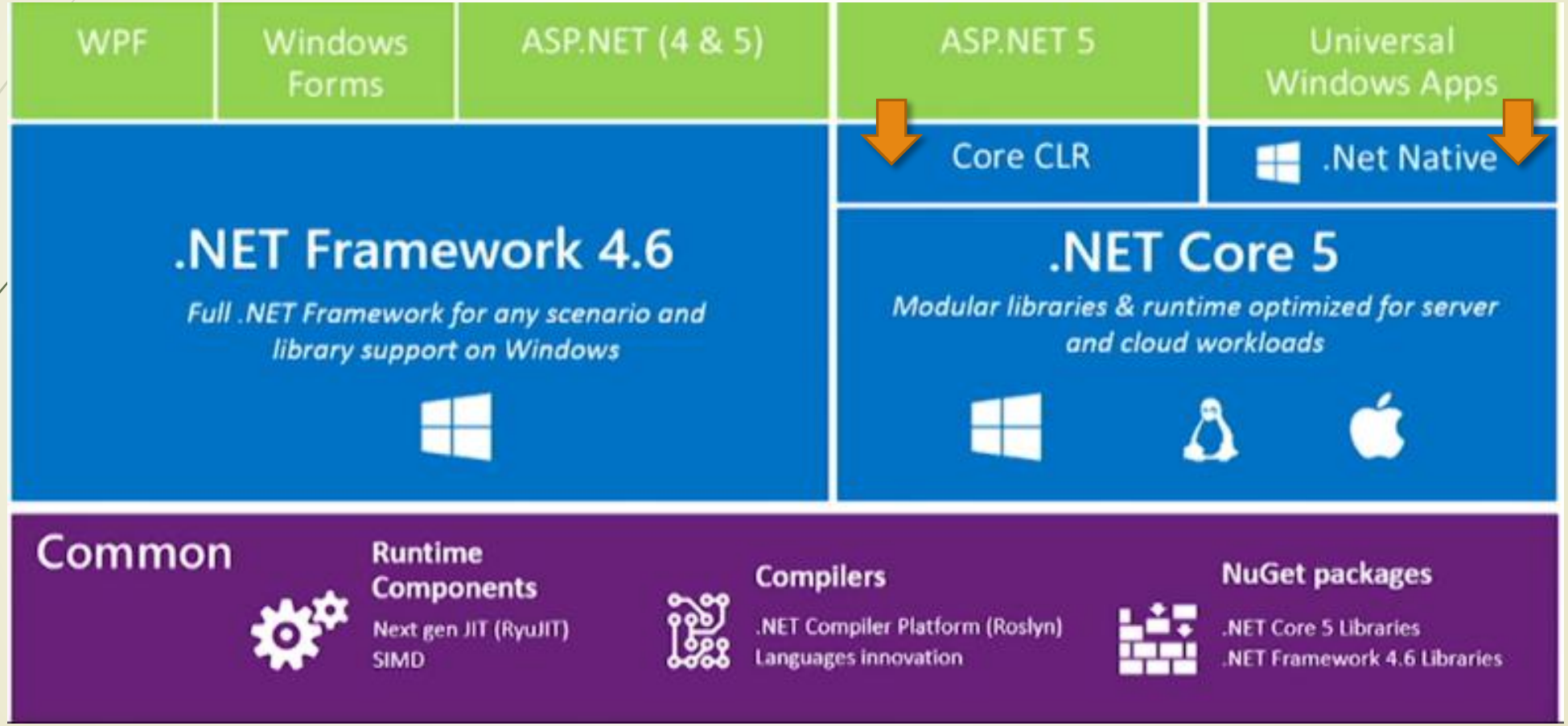
Arhitektura Javinog nasuprot .NET Framework-a



https://korzh.com/blog/dotnet-vs-java-what-development-platform-better

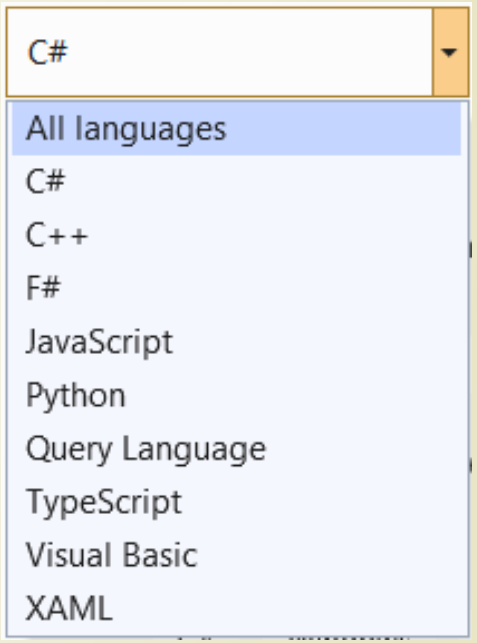
Arhitektura .NET-a 4.6 / .NET Core 5

Tipovi aplikacija



Programski jezici i .NET

- Već znamo, visoki nivo međujezičke kompatibilnosti obezbeđuje **CLR**.
- Aplikacija .NET-a se **PREVODI** u Microsoft-ov posrednički jezik **MSIL/IL** (engl. *MicroSoft Intermediate Language*).
- **MSIL/IL** je jezik **NISKOG NIVOA** koji **CLR** može da pročita i da razume (čitaj: *prevede i izvrši*).
- Zato što **SVI** .NET izvršni programi i datoteke **DLL**-a postoje u formi MSIL-a, mogu slobodno da **RADE ZAJEDNO**!
- Zajednička specifikacija za .NET jezike - **CLS** (engl. *Common Language Specification*) definiše **MINIMALNE STANDARDE** koje moraju da zadovolje prevodioci .NET-a.
- Tako, **CLS** obezbeđuje da svaki izvorni kod koji je uspešno preveden .NET prevodiocem, može da radi u drugom okruženju sa .NET FW.

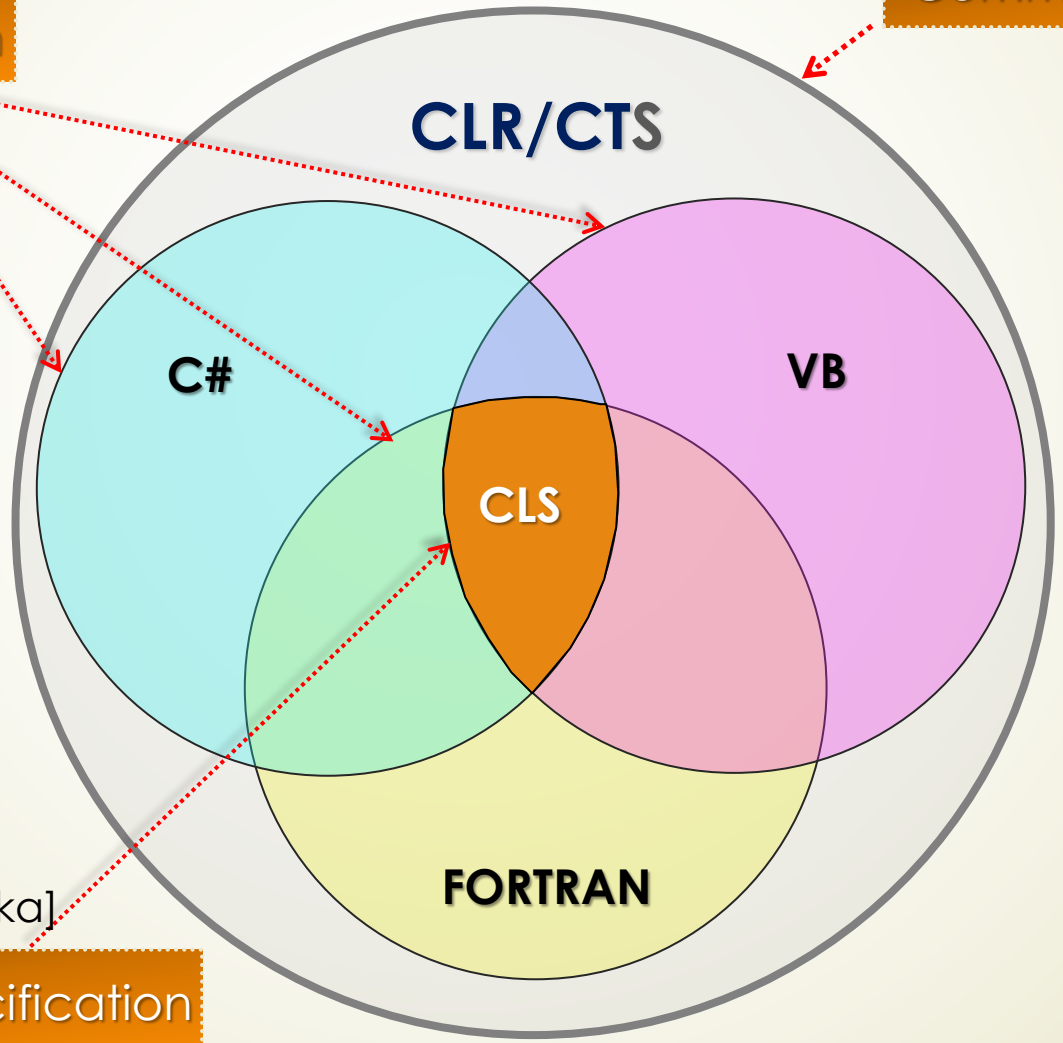


CLR/CTS

[Zajedniči sistem tipova]

Common **T**ype **S**ystem

Jezici podržani **.NET**-om

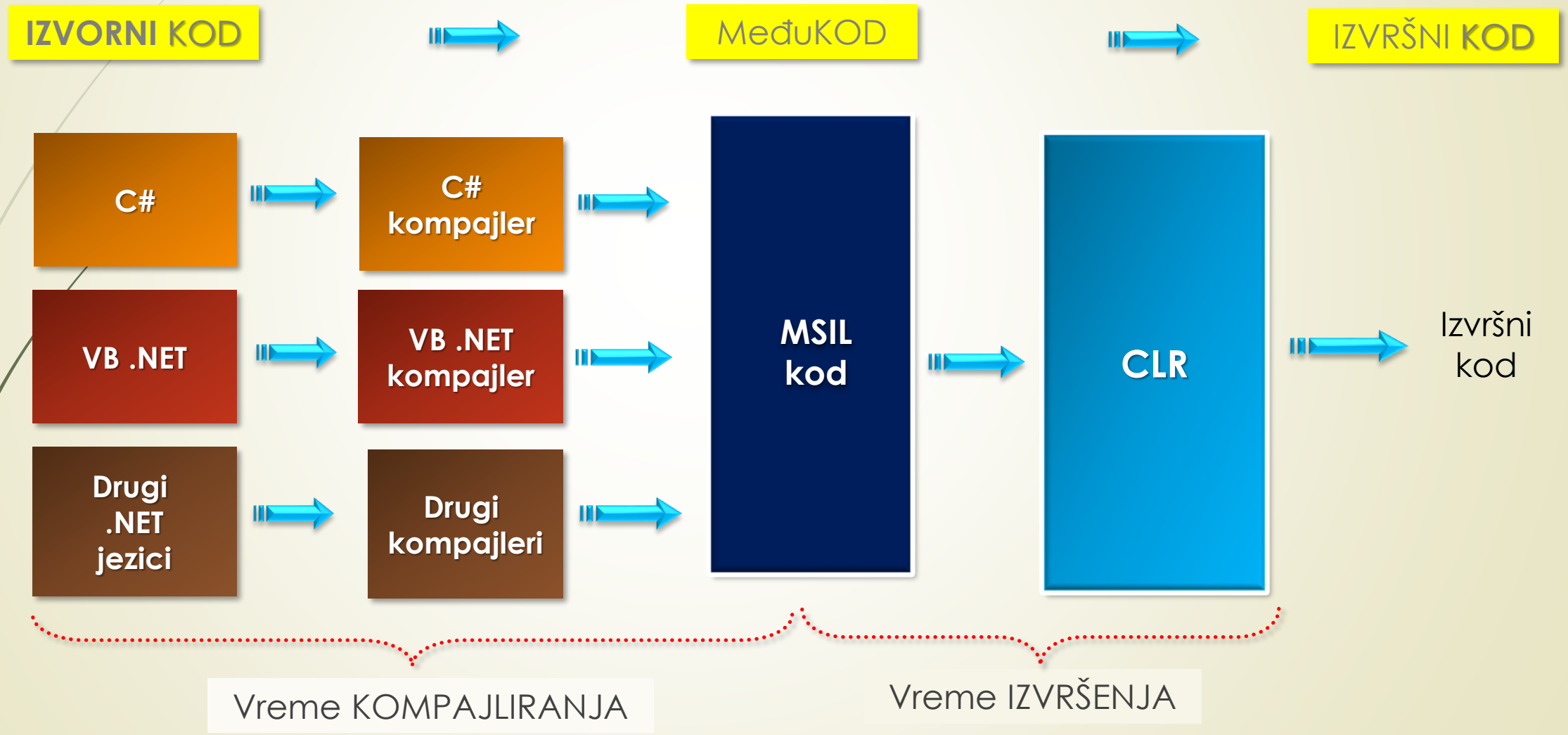


CTS tipovi
Byte
Char
Boolean
SByte
Int16
UInt16
Int32
UInt32
Single
Double
Decimal
Int64

[Zajednička specifikacija jezika]

Common **L**anguage **S**pecification

MSIL i kompatibilnost .NET jezika



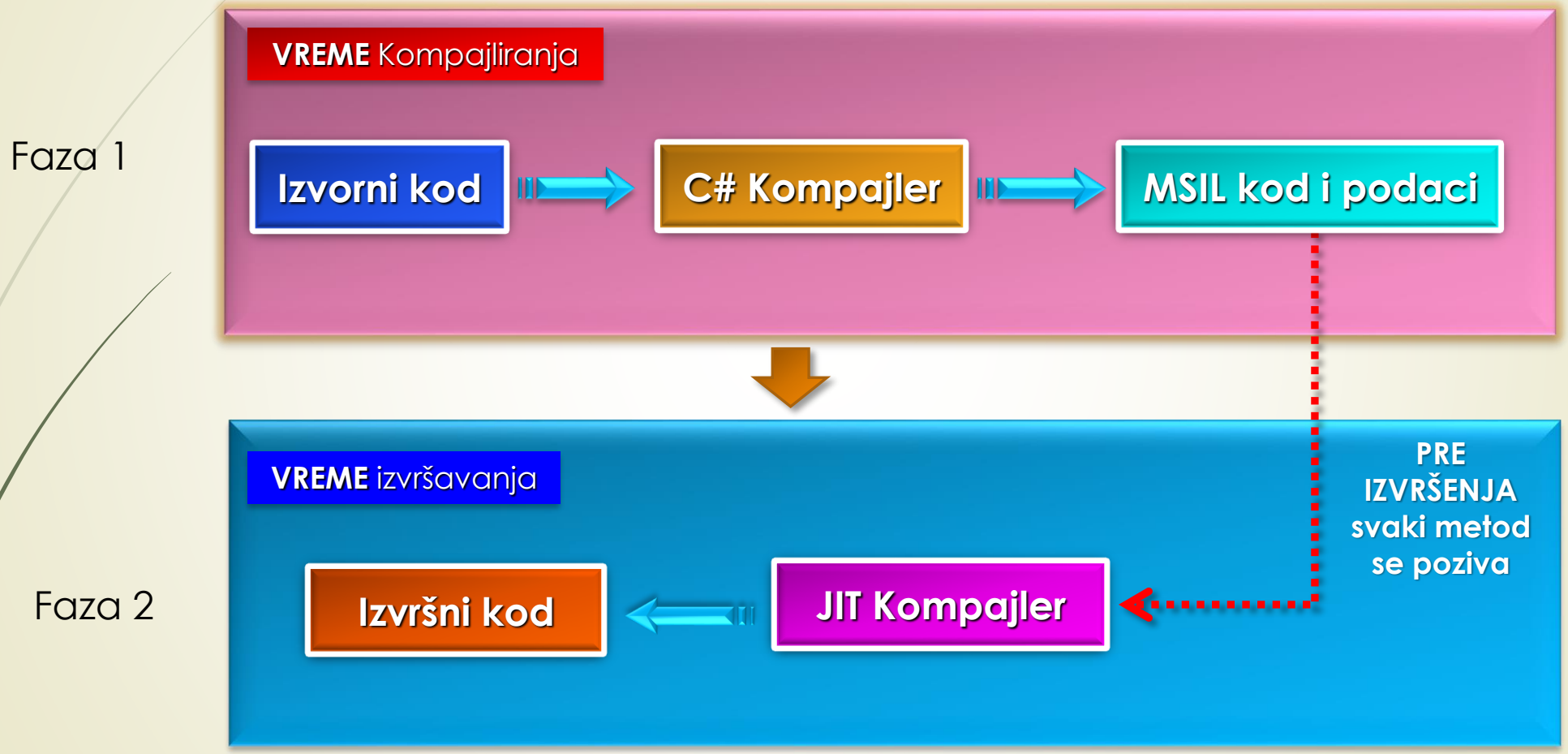
Prevođenje i izvršavanje u .NET-u

- Za razliku od C++, rezultat kompajliranja aplikacije **NIJE BINARNI MAŠINSKI KOD**, već **(MS)IL** kod.
- **MSIL** kod se sastoji od **PROGRAMSKIH SKLOPOVA** (engl. *Assembly*) koji predstavljaju **OSNOVNU JEDINICU PROGRAMIRANJA** u .NET okruženju.
- **BAR JEDAN** od programskih sklopova mora posedovati **IZVRŠNU DATOTEKU** koja je bila određena kao **ULAZNA TAČKA** za aplikaciju.
- Kada započne izvršavanje programa, **PRVI SKLOP** se učitava u memoriju.
- **CLR ISPITUJE OBJAVU SKLOPA** i utvrđuje zahteve neophodne za izvršenje (o programskim sklopovima nešto više u nastavku kursa).
- **CLR ISPITUJE BEZBEDNOST DOZVOLE** zatražene od sklopa i **POREDI** ih sa sistemskim bezbedonsonim merama.
- **MSIL KOD** zahteva **KOMPAJLIRANJE** u **VREME IZVRŠENJA**, a taj posao obavlja KOMPONENTA CLR-a **- JIT** (engl. *Just In Time Compile*r).

Prevođenje i izvršavanje u .NET (2)

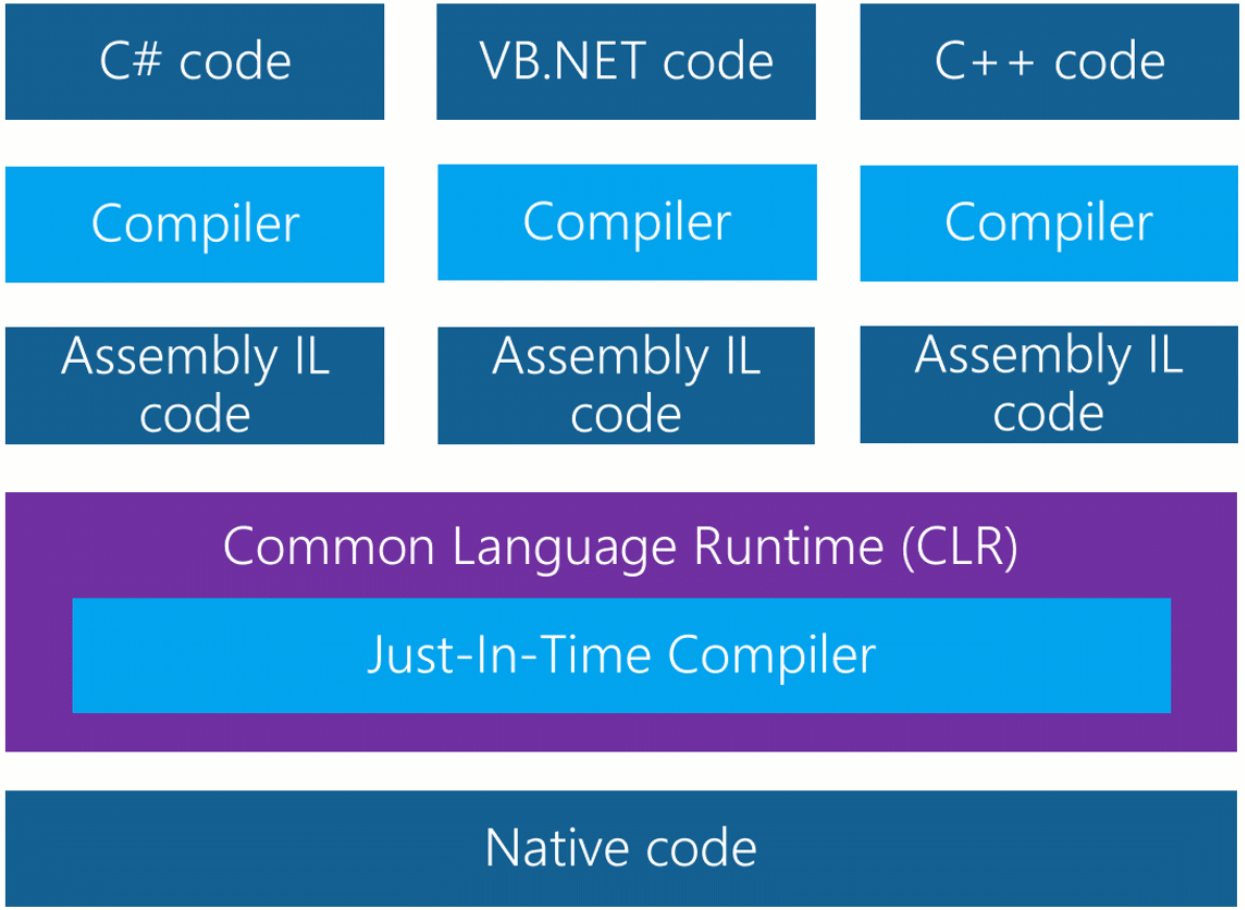
- Ukoliko sistemska bezbedonosna politika **NE OBEZBEĐUJE** zahtevane dozvole – aplikacija se **NEĆE IZVRŠITI**, u suprotnom CLR će izvršiti kod.
- **CLR** pravi **PROCES ZA APLIKACIJU** u kome će se ona izvršiti i započinje njeno izvršenje.
- Prvi delovi koda se **UČITAVAJU** u memoriju i **PREVODE** u prirodni (engl. *native*) binarni kod iz MSIL-a pomoću prevodioca CLR-a koji se zove **Just-InTime kompajler** (JIT).
- Svaki deo koda se prevodi **SAMO JEDNOM** kada se aplikacija izvrši.
- Grananje programa na deo koda koji nije preveden direktno izaziva njegovo **UČITAVANJE** i **PREVOĐENJE**.
- Na ovaj način se performanse aplikacije **MAKSIMIZUJU** jer se prevodi samo onaj deo koda koji će se izvršiti!

Faza kompajliranja i faza izvršavanja



- Postoje tri tipa JIT kompajlera: Pre-JIT, Normal-JIT i Econo-JIT

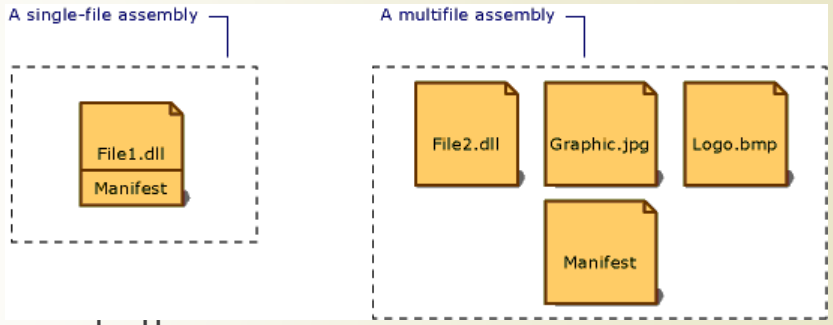
Arhitektura .NET-a - kompajleri



https://faun.pub/demystifying-asp-net-core-aad722ebbd51

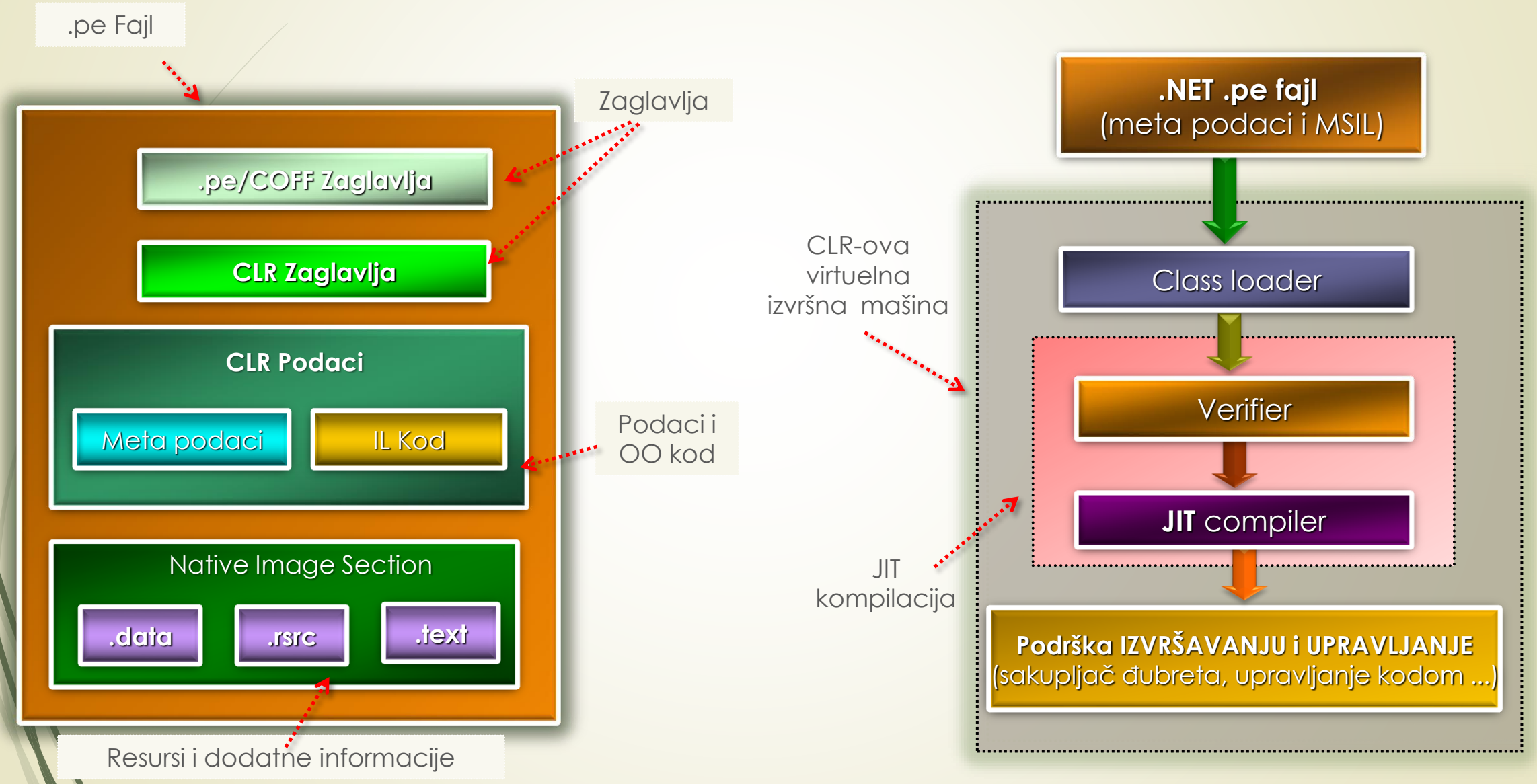
Programski sklopovi (1)

- Kako je već rečeno, **PROGRAMSKI SKLOP** (engl. *Assembly*) je osnovna jedinica programiranja u .NET okruženju i predstavlja **SKUP DATOTEKA** koji izgleda slično kao jedinstvena **.dll** ili **.exe** datoteka.
- Programski sklopovi se sastoje od:
  - **OBJEKTNOG KODA**,
  - **RESURSA** (primer .gif datoteke),
  - **DEFINICIJE TIPOVA** za svaku klasu, kao i druge metapodatke.



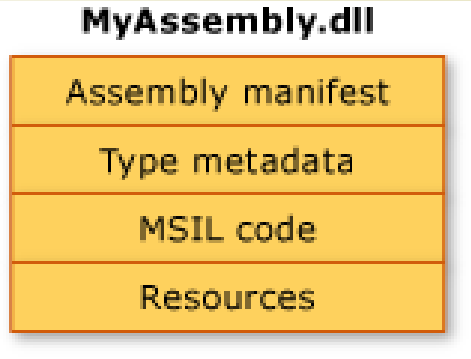
- Na disku, **PROGRAMSKI SKLOPOVI** su predstavljeni kao **PRENOSIVE IZVRŠNE DATOTEKE** (engl. *Portable Executable*) i imaju ekstenziju **.pe**.
- Programski sklopovi se mogu sastojati od **JEDNOG** ili više **MODULA**.
- **MODULI** se **NE MOGU IZVRŠAVATI** nezavisno, već se **MORAJU** kombinovati u **SKLOPOVE**.
- Na sledećem slajdu je prikakazana struktura **.pe** fajlova.

Struktura .NET sklopa / Virtualni izvršni sistem CLR-a



Programski sklopovi (2)

- Dakle, osnovna jedinica aplikacije u .NET-u je **PROGRAMSKI SKLOP**, a njegova glavna osobina je da je to **SAMOOPISUJUĆA KOLEKCIJA** koja se sastoji od:
  - **koda**,
  - **resursa** i
  - **podataka**.



- **DEKLERACIJA SKLOPA** sadrži informacije o tome **OD ČEGA** je sazdan sklop.
- **DEKLERACIJA SKLOPA** poseduje sledeće podatke:
  - Podatke **o identitetu sklopa**, kao što je ime i broj verzije sklopa;
  - Listu **svih tipova** koje sklop sadrži;
  - **Listu svih sklopova** koje zahteva dati sklop (dakle, sklop može sadržavati druge sklopove!);
  - Listu uputstva za **bezbednost pristupa** koda za sklop, što podrazumeva dozvole koje sklop zahteva i dozvole će se odbiti za sklop.
  - Skup **metapodataka** sklopa.

Metapodaci sklopa

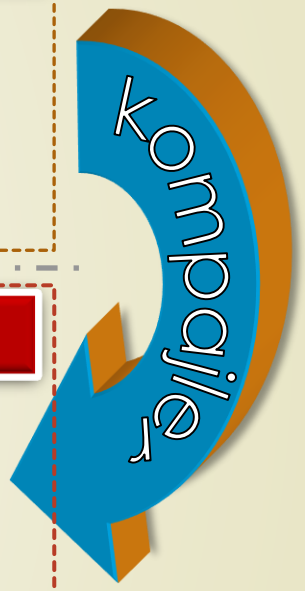
- Podaci o sklopu se objavljuju u **MANIFESTU SKLOPA**.
- Da bi programski sklop mogao da se deli mora biti smešten u **GLOBALNI KEŠ PROGRAMSKIH SKLOPOVA** - **GAC** (engl. *Global Assembly Cache*) koji je CLR predvideo za čuvanje deljenih programskih sklopova.
- **GAC** implementira funkciju **deljene biblioteke** u kojoj različite aplikacije **ponovo koriste** kod smešten u datotekama koje se nalaze u zajedničkom folderu.
- Na sledećem slajdu je prikazan izgled **MANIFESTA** programskog sklopa za program razvijen u C# koji štampa poruku: "Zdravo svete".
- Da bi se dobio sledeći prikaz treba aktivirati **ILDasm** (**dotPeek** u novijim verzijama VS-a) program, koji može prikazati **MANIFEST SKLOPA** u .NET-u.
- Pogledajte izvorni kod ovog "programa" bez ulaženja u detalje, jer će o njima biti reči u narednim predavanjima.

C#/MSIL kod za HelloWord aplikaciju

Izvorni kod

C#

```
using System;
public class MainApp
    { public static void Main( )
        { Console.WriteLine("C# Hello, World!"); }
     }
```

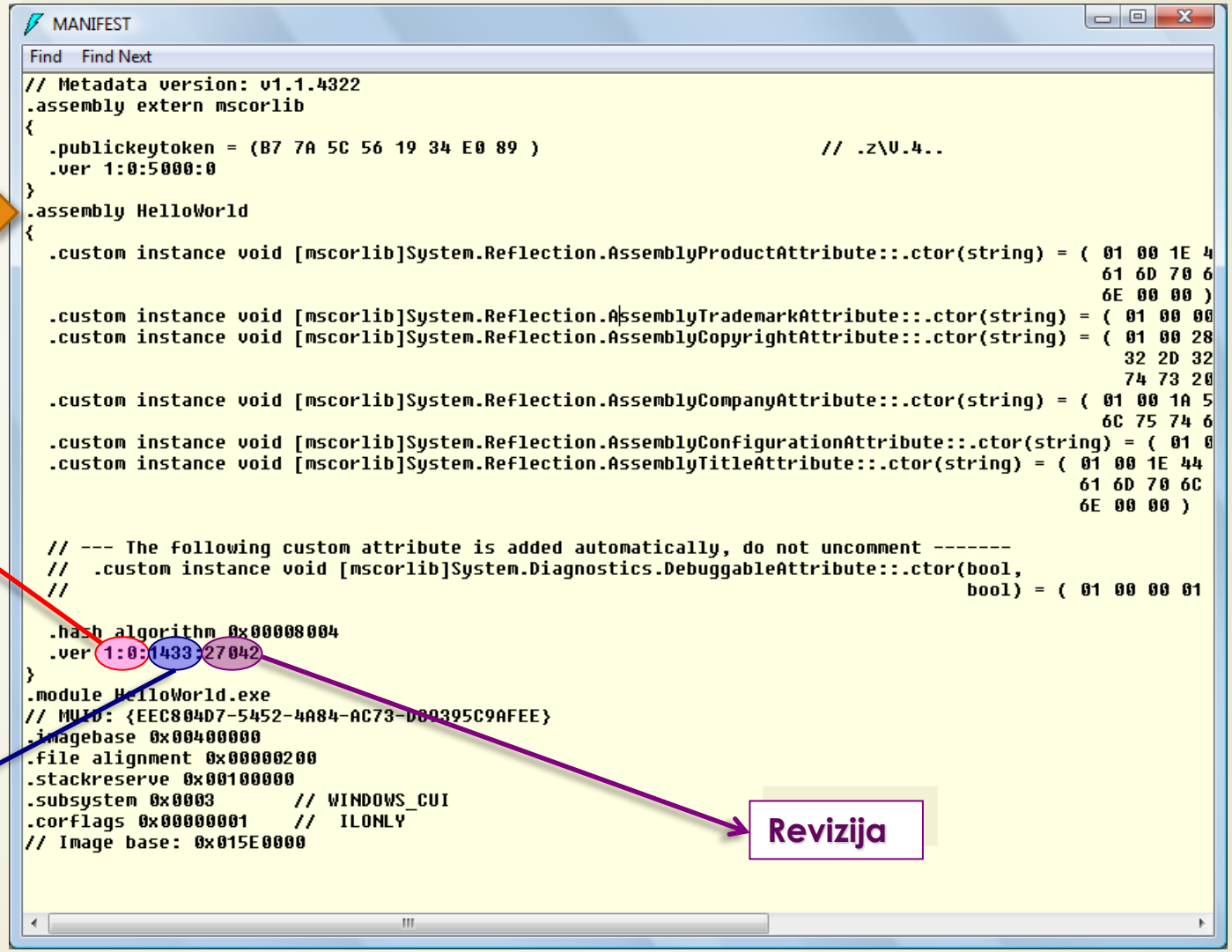


Sadržaj sklopa

MSIL

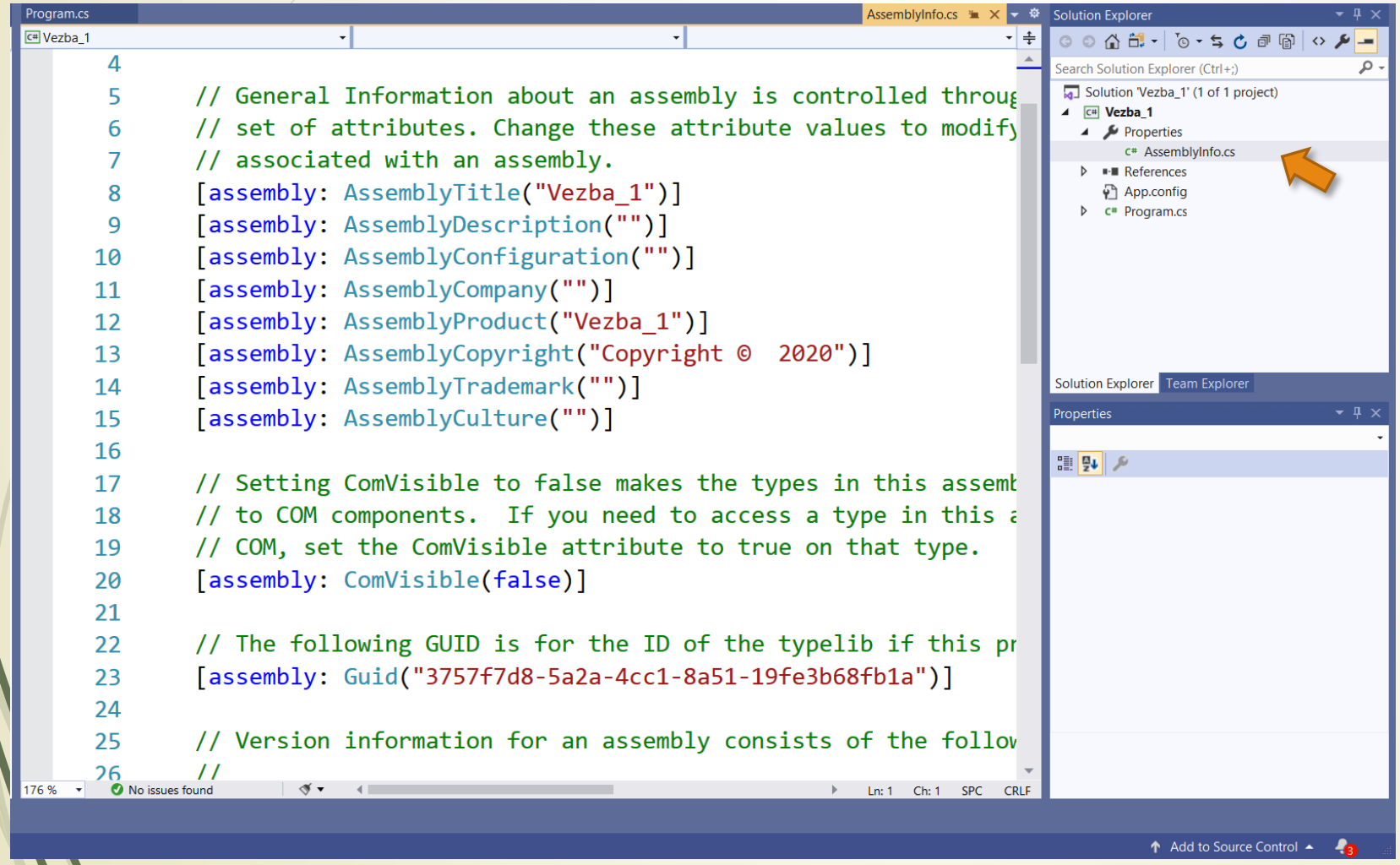
```
.assembly HelloWorld { }
.assembly extern mscorlib { }
.class Program extends [mscorlib]System.Object
    {
        .method static void Main() cil managed
        {
            .entrypoint
            .maxstack 1
            ldstr "Hello, World!"
            call void [mscorlib]System.Console::WriteLine(string)
         ret
        }
    }
```

Manifest sklopa i metapodaci



ILDasm

AssemblyInfo.css u VS 2019



- U datoteci ***AssembliInfo.cs*** čuvaju se informacije o nazivu proizvoda, opisu, zaštitnom znaku, autorskim pravima, …
- .NET sklop omogućava čuvanje ovih podataka kao ***deo sklopa*** posle kompajliranja.
- U vreme izvršavanja mogu se pročitati ove informacije.
- Prikazan je fajl ***AssembliInfo.cs*** iz laboratorijske vežbe Vezba_1.

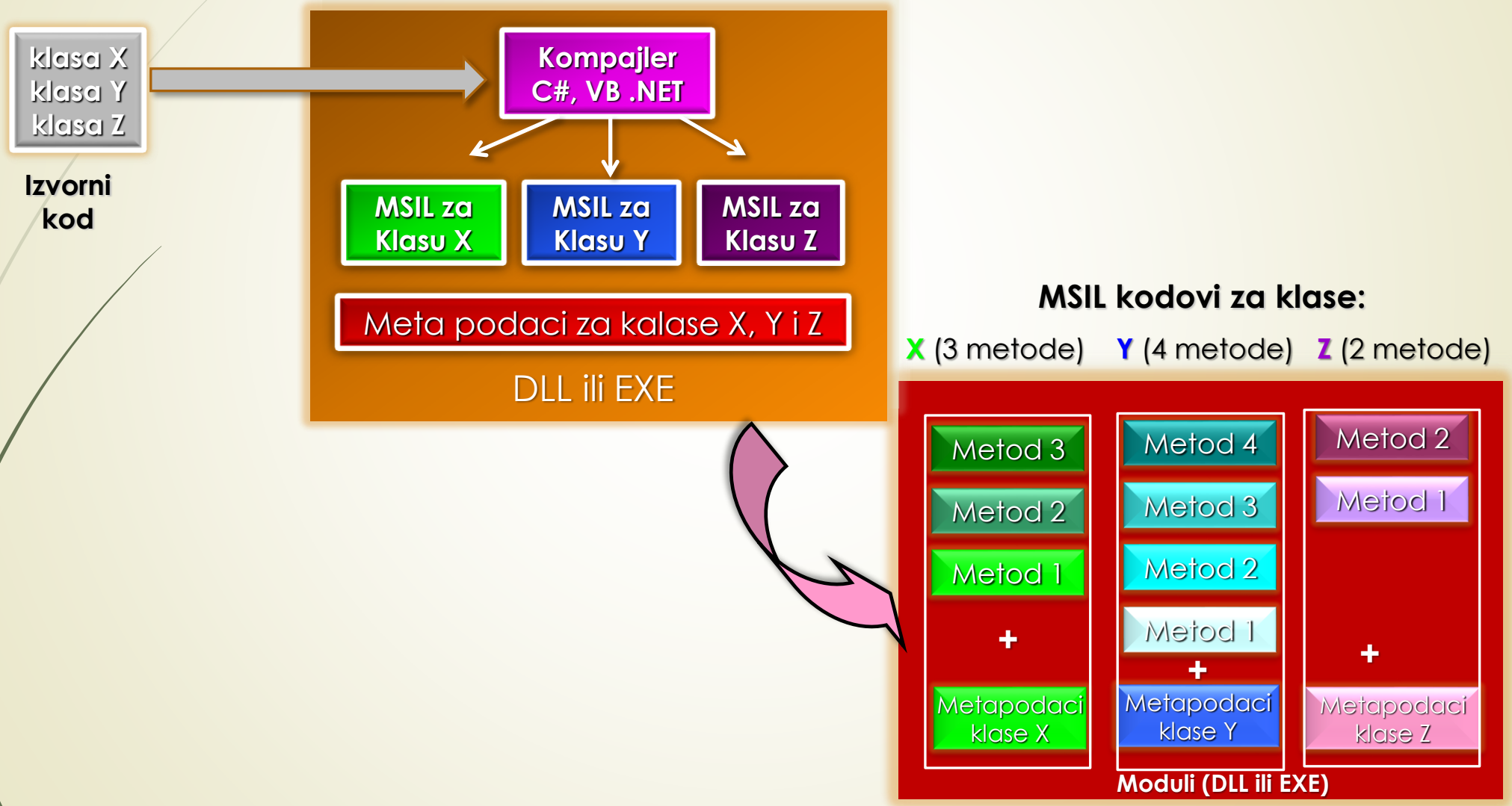
Struktura modula u .NET-u

- Svaki modul (deo sklopa) u .NET-u sadrži određeni **BROJ TIPOVA PODATAKA**.
- **TIPOVI** podataka su **ŠABLONI** koji opisuju **KOLEKCIJU ENKAPSULACIJE PODATAKA** i **FUNKCIONALNOSTI**.
- O tipovima podataka u .NET-u biće održano posebno predavanje, za sada zapamtite da postoje **DVE** vrste tipova:
  - **VREDNOSTNI** tipovi (strukture, int, char,...)
  - **REFERENTNI** tipovi (klase, interfejsi, delegati, ...)
- Svaki tip je opisan u **CLR**-u u objavi sklopa!
- Opis tipa sadrži:
  - **Polja** (za skaldištenje podataka)
  - **Svojstva** (obezbeđuju neku vrstu provere)
  - **Metode** (predstavljaju ponašanje tipa)

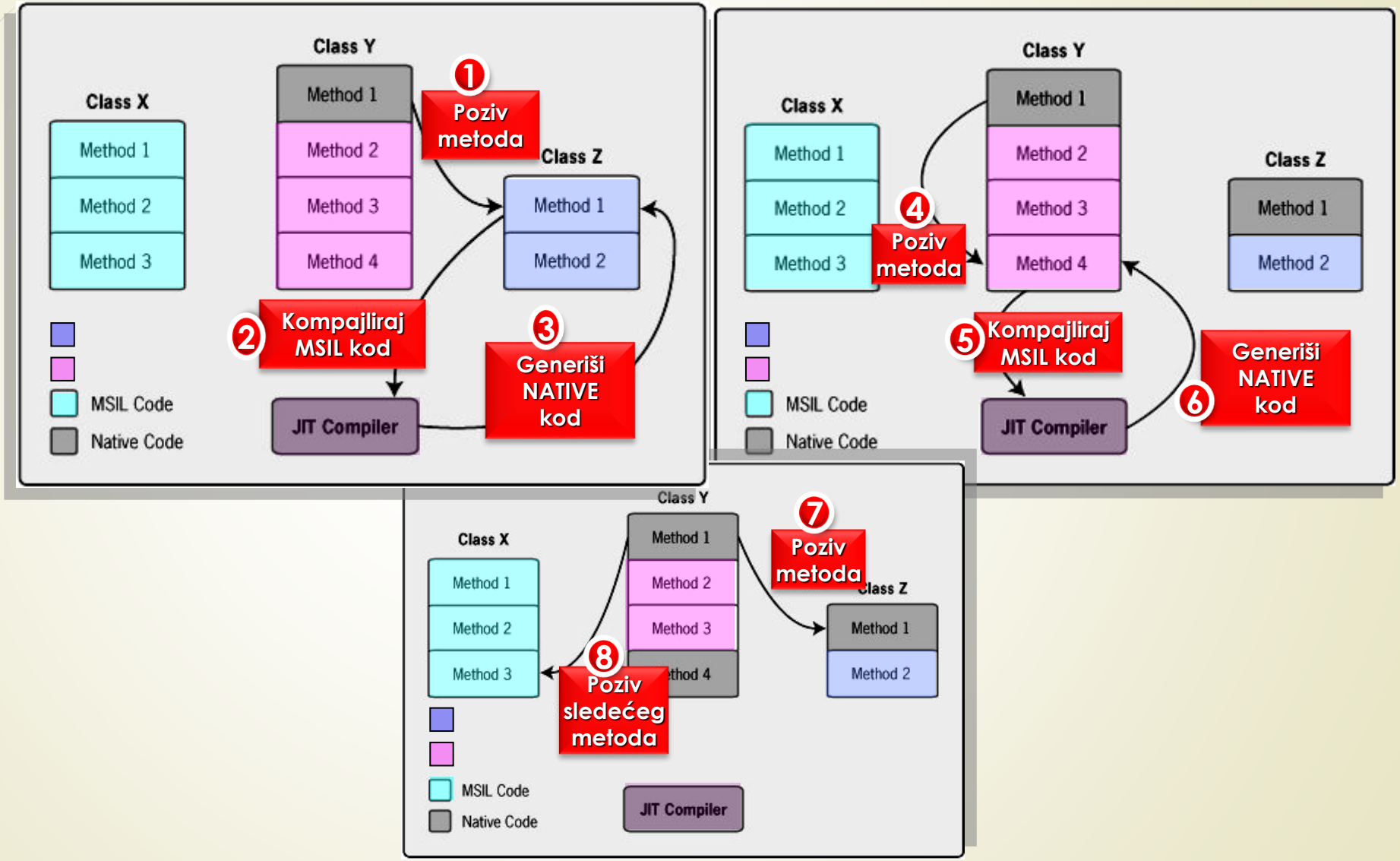
Struktura aplikacija .NET-a

- Svaki sklop ima **JEDNU I SAMO JEDNU DEKLERACIJU**.
- Dekleracija sklopa može da se nalazi u **posebnoj datoteci** ili može biti unutar nekog od modula sklopa.
- Već znamo, **SKLOP** može sadržavati jedan ili više modula.
- Modul sadrži kod koji **sačinjava aplikaciju** ili **BIBLIOTEKU** i **METAPODATKE** koji opisuju taj kod.
- **Prevođenje projekta u sklop** podrazumeva prevođenje koda iz **višeg** .NET programskog jezika u **MSIL**.
- Konvertovanje u **MSIL** kod pre izvršenja je zapravo ključ MEĐUJEZIČKE KOMAPTIBILNOSTI.

Prog. jezici .NET-a i kompajler



Primer rada JIT Kompajlera



Upravljani kodovi i .NET

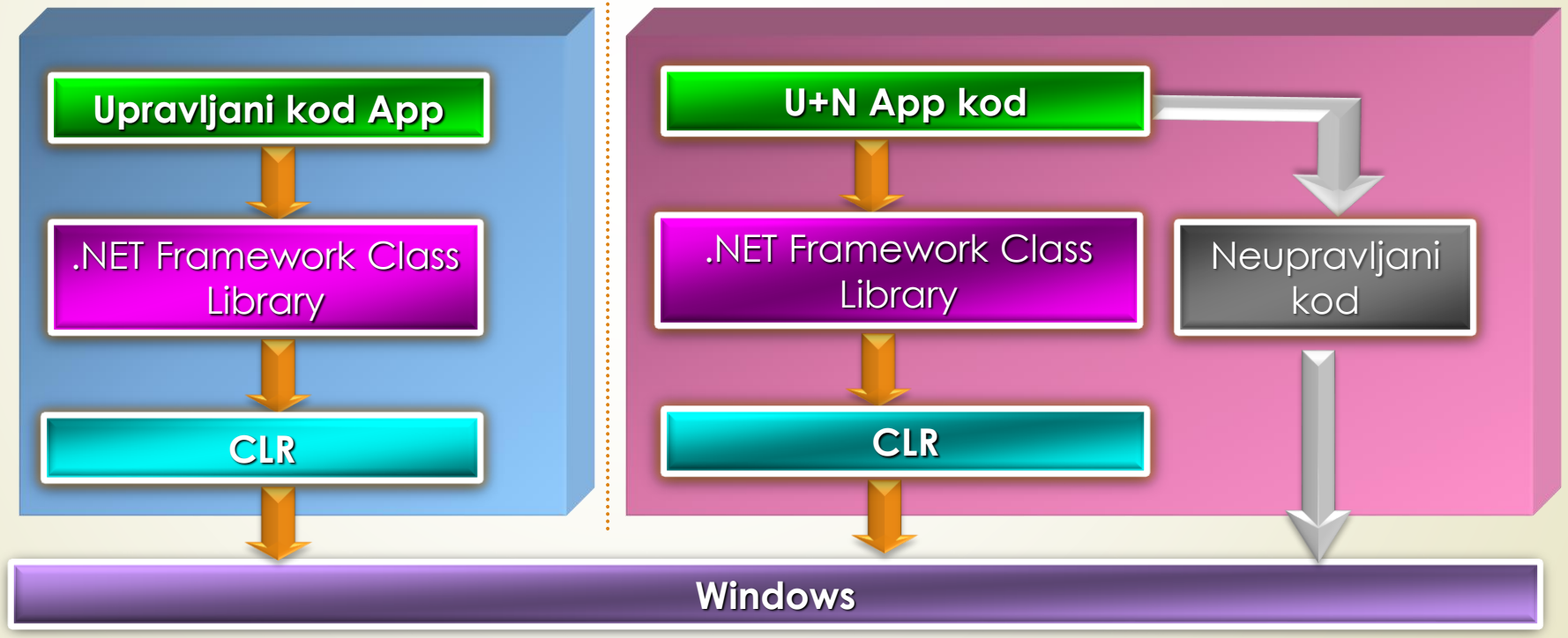
- **UPRAVLJANI** (engl. *managed*) kod je programski kod načinjen da **RADI** pod **CLR**-om koji je **SAMOOPISUJUĆI**.
- Kod napisan npr. u VB6 koji nemaju MSIL i kojima **NE TREBA** CLR za rad nazivaju se **NEUPRAVLJANIM** kodom.
- **CLR** će za **UPRAVLJANE** kodove:
  - Pronaći **metapodatke** koji su u vezi sa metodom u bilo kom vremenskom trenutku;
  - Prošetati **stekom**;
  - Rukovati **izuzecima**;
  - Čuvati i pribavljati informacije o **bezbednosti**.

Upravljani – neuparvljani kodovi (1)

Aplikacija koja koristi OBA koda:

UPRAVLJANI i NEUPRAVLJANI N kod
(engl. *unmanaged*)

Aplikacija koja korista
ISKLJUČIVO UPRAVLJANI
(engl. *managed*) **U kod**



Upravljani – neuparvljani kodovi (2)

UPRAVLJANI KOD	NEUPRAVLJANI KOD
Kod izvršava **CLR** umesto operativnog sistema (**OS)**	Kod **direktno** izvršava operativni sistem
Radno okruženje obezbeđuje **GAC** (engl. *Global Assembly Cache*), upravlja radom izuzetaka, …	**Nema** usluga kao što je GAC, upravljanje izuzecima, …
Kod kompajliran u **MSIL** kod	Kod je kompajliran u **NATIVE** kod

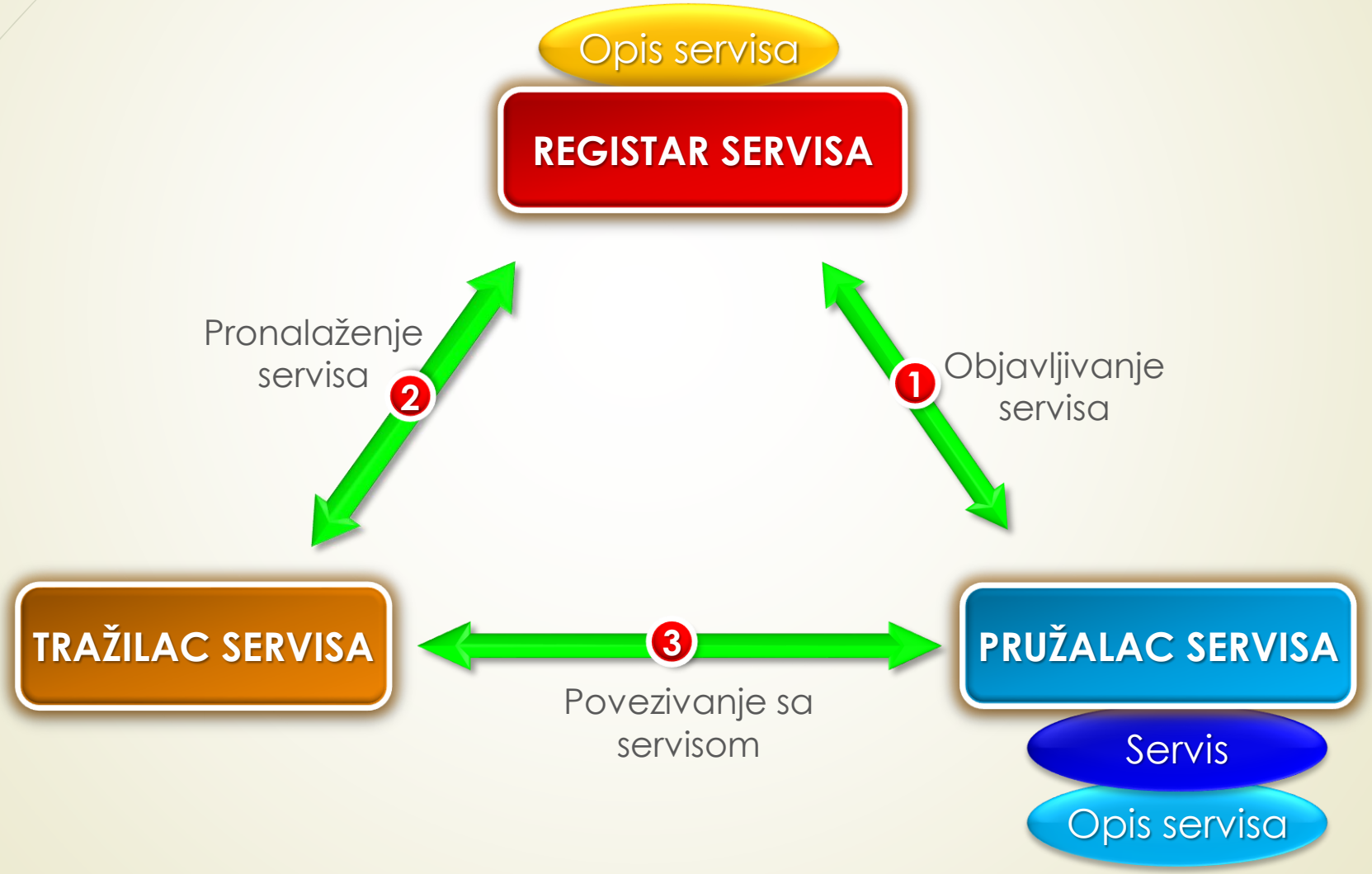
Da se podsetimo - web servisi

- Koncept savremenih aplikacija omogućava istovremeno korišćenju **VIŠE** (XML) **WEB SERVISA** bez obzira na njihovu:
  - LOKALIZACIJU i
  - NAČIN IMPLEMENTACIJE.
- Savremena (Web) aplikacija koja kombinuje više (XML) WEB SERVISA u **jednoj aplikaciji** naziva se **MASHUP** aplikacija.
- Da bi ovaj koncept zaživeo, bilo je potrebno da se u .NET-u kombinuju specifičnosti HTML-a i XML-a, a posebno:
  - **Prezentacione** mogućnosti HTML-a i
  - Izuzetna mogućnosti rada sa **meta** podacima XML-a.
- Na sledećem slajdu je prikazane **BAZNA ARHITEKTURA** XML Web servisa.
- Opis funkcije svih **KOMPONENTATA XML Web servisa** je dat na narednom slajdu.

Komponente Web servisa

- Osnovne **KOMPONENTE** Web servisa su:
  - **PRUŽALAC SERVISA** – Kreator Web servisa je osoba ili organizacija koja realizuje Web servis i čini ga **DOSTUPNIM** preko Internet mreže.
  - **REGISTAR SERVISA** – Centralni **DIREKTORIJUM** Web servisa za publikovanje Web servisa ili pronalaženje već postojećih.
  - **TRAŽILAC SERVISA** – Korisnik Web servisa, osoba ili organizacija koja **KORISTI** Web servis. Korisnik Web servisa uspostavlja vezu sa servisom i šalje zahteve za uslugom u formi XML poruka.
- **ARHITEKTURA** Web servisa je podržana sledećim operacijama:
  - **OBJAVLJIVANJE** - Web servis mora biti OBJAVLJEN da bi bio dostupan! Korisnici ga mogu **OTKRITI** i **POZVATI** na izvršavanje.
  - **PRONALAŽENJE** - Potencijalno zainteresovani korisnik **PRONALAZI** Web servis pretraživanjem registra po zadatom kriterijumu.
  - **POVEZIVANJE** - Posle pronalaženja Web servisa, korisnik **PRISTUPA OPISU** Web servisa i poziva ga na izvršavanje u skladu sa informacijama koje se nalaze u opisu.

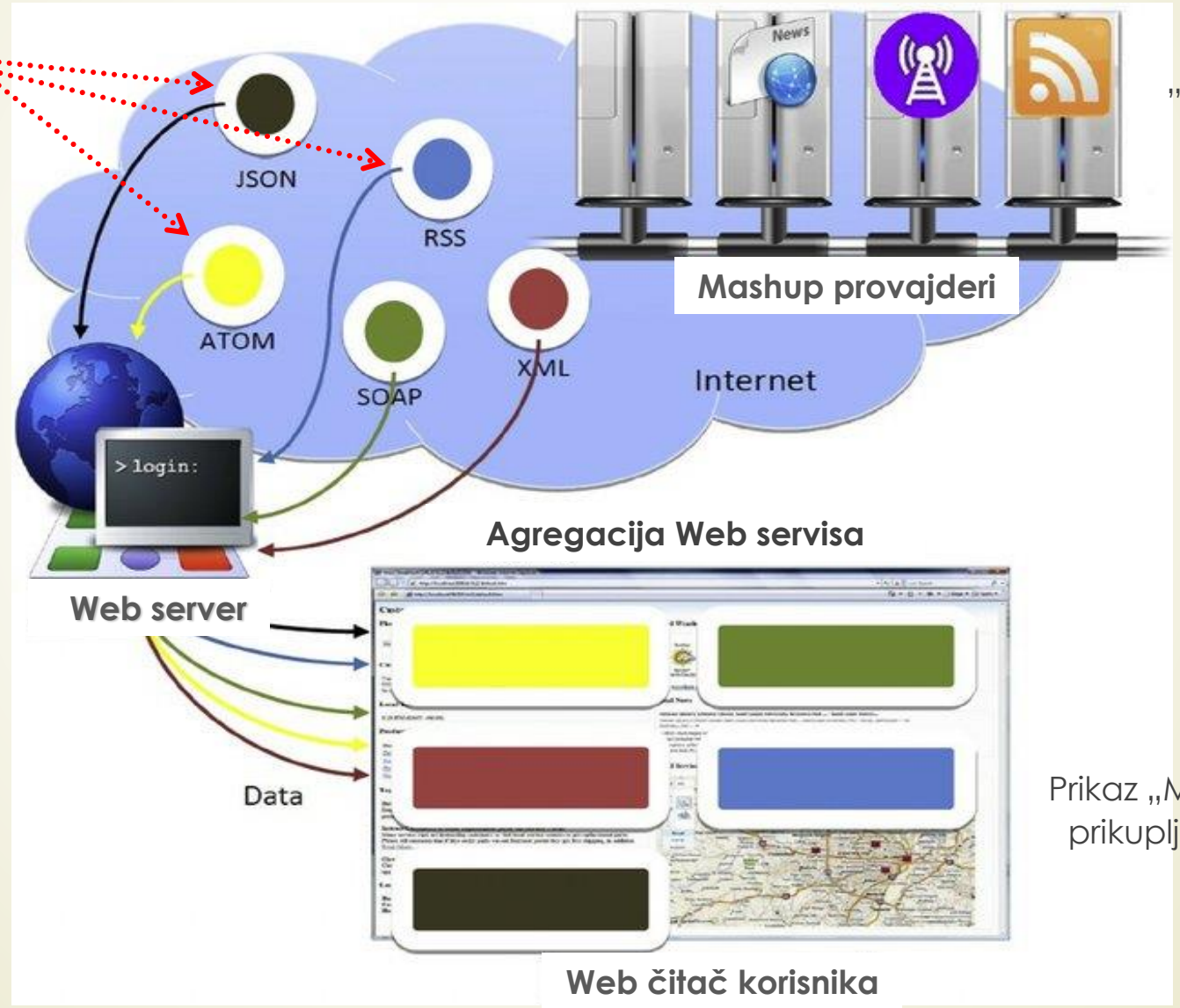
Arhitektura Web servisa



Arhitektura MASHUP aplikacije

Različite forme i protokoli za prikupljanje podataka od „**Mashup**“ provajdera.

Web aplikacija **integriše** „Mashup“ podatke.



„**Mashup**“ provajderi svoje podatke čine dostupnim preko mreže.

Jedan od načina da se podaci učine dostupnim je kreiranje **Web servisa**.

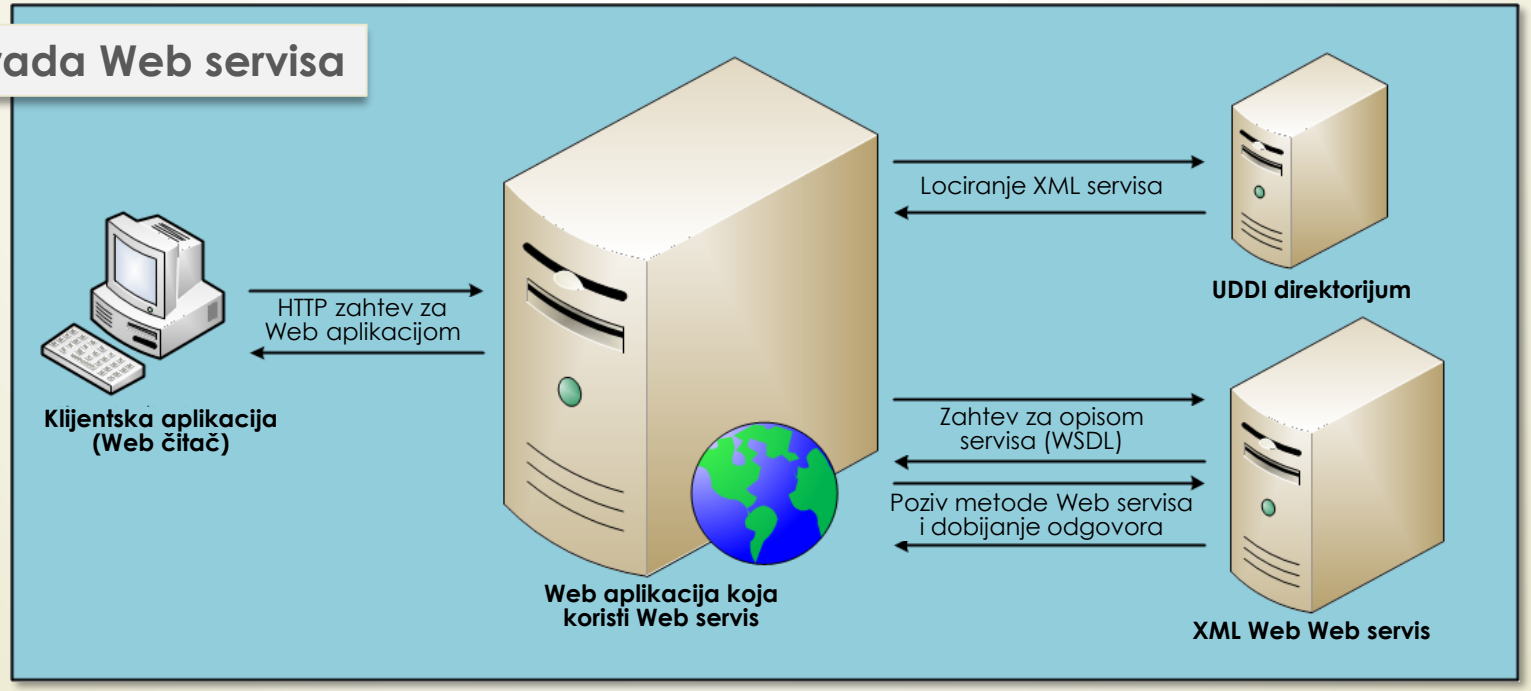
Web servis je skup otvorenih protokola i standarda koji se koriste za razmenu podataka između aplikacija.

Prikaz „Mashup“ podataka prikupljenih iz **više** izvora.

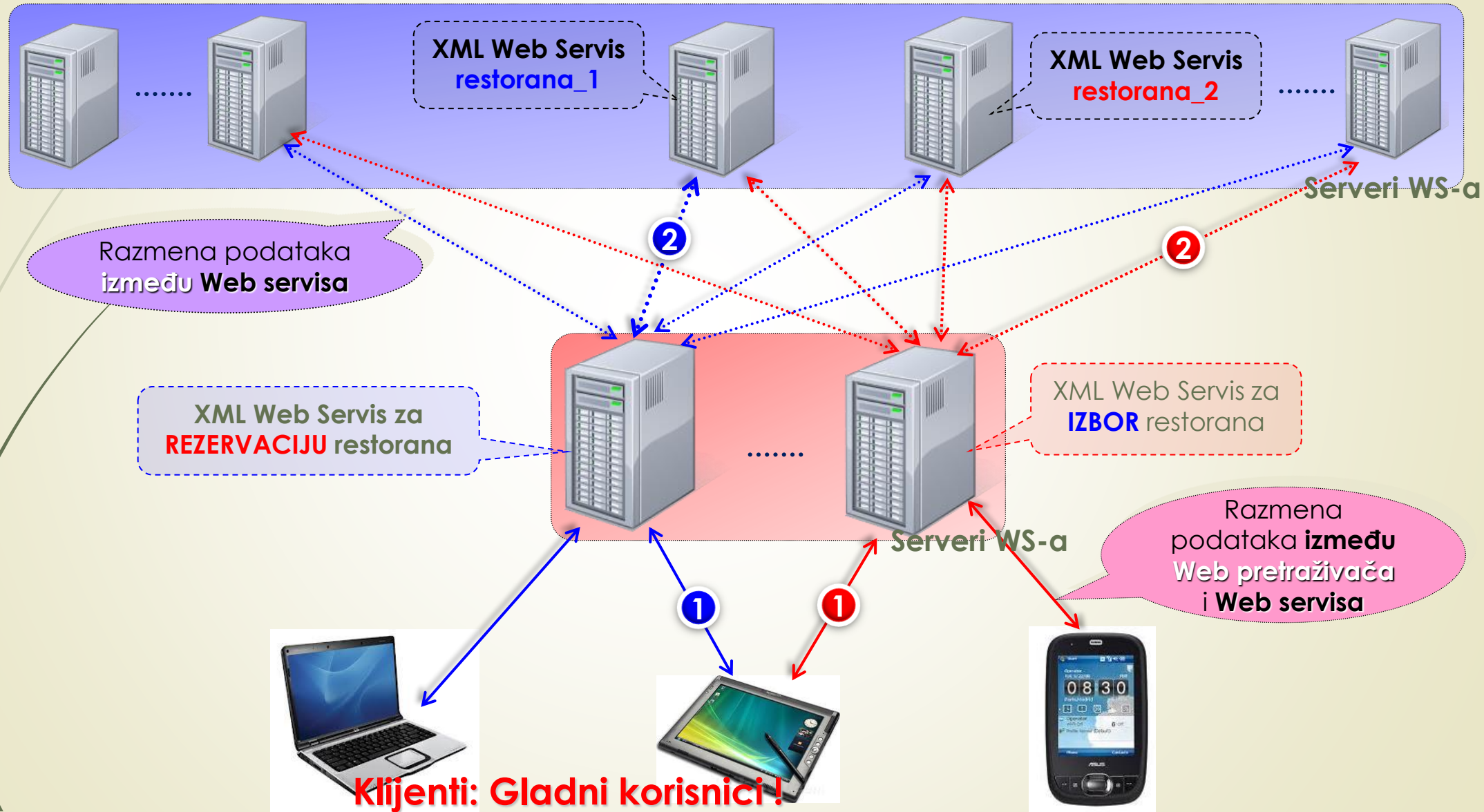
Stek protokola Web servisa

Registar servisa	UDDI - Universal Description, Discovery and Integration
Opis servis	WSDL - Web Services Description Language
Razmena XML poruka	XML-RPC, Remote Procedure Call, SOAP - Simple Object Access Protocol
Transportna usluga	HTTP, SMTP, FTP

Princip rada Web servisa



Hijerarhija Web servisa



Umesto rezimea

- Koje osnovne servise obezbeđuje CLR?
- Objasnite pojam sistem zajedničkih tipova CTS
- Biblioteka klasa Framework-a.
- Objasnite pojam međujezičke kompatibilnosti.
- MSIL/IL kao programski jezik.
- Programski jezici .NET-a.
- Objasnite pojam Just-In-Time (JIT) prevodioca.
- Šta je osnovna jedinica aplikacije u .NET-u?
- Objasniti pojam metapodataka.