Disciplina: Desenvolvimento de Sistemas de Informação Avançados I

Professor: Rafael Marinho e Silva

CRONOGRAMA

- OBJETIVOS DA DISCIPLINA
- AVALIAÇÃO
- CONTEÚDO DA DISCIPLINA

EMENTA E OBJETIVOS DA DISCIPLINA

- **Conhecimento das bibliotecas e funções básicas e avançadas em programação utilizando C#**
- **Estudo e aplicação de programação em camadas utilizando a Plataforma .NET, com ênfase em ASP.NET MVC**

- **Criação de aplicações web e desktop através do uso da linguagem C#**
  - **proporcionando o conhecimento através de conceitos e exemplos práticos**

CRONOGRAMA

- OBJETIVOS DA DISCIPLINA
- **AVALIAÇÃO**
- CONTEÚDO DA DISCIPLINA

AVALIAÇÃO

DISTRIBUIÇÃO DOS PONTOS

Atividades	Datas
1º Lote - 10 pts	até dia 02/09
2º Lote - 20 pts	até dia 21/10
3º Lote - 10 pts	até dia 22/11
Avaliação Colegiada (AC) - 20 pts	de 02/12 a 06/12
Avaliação Integradora (AVIN) - 20 pts	16/12

AVALIAÇÃO

DISTRIBUIÇÃO DOS PONTOS

Atividades	Pontuação	Datas
1ª prova	12 pts	dia __/__
2ª prova	12 pts	dia __/__
Prova Colegiada	20 pts	dia __/__
3 listas de APS	- 3 pts - 3 pts - 3 pts	dia __/__ dia __/__ dia __/__
3 atividade práticas (sala de informática)	- 2 pts - 2 pts - 3 pts	dia __/__ dia __/__ dia __/__

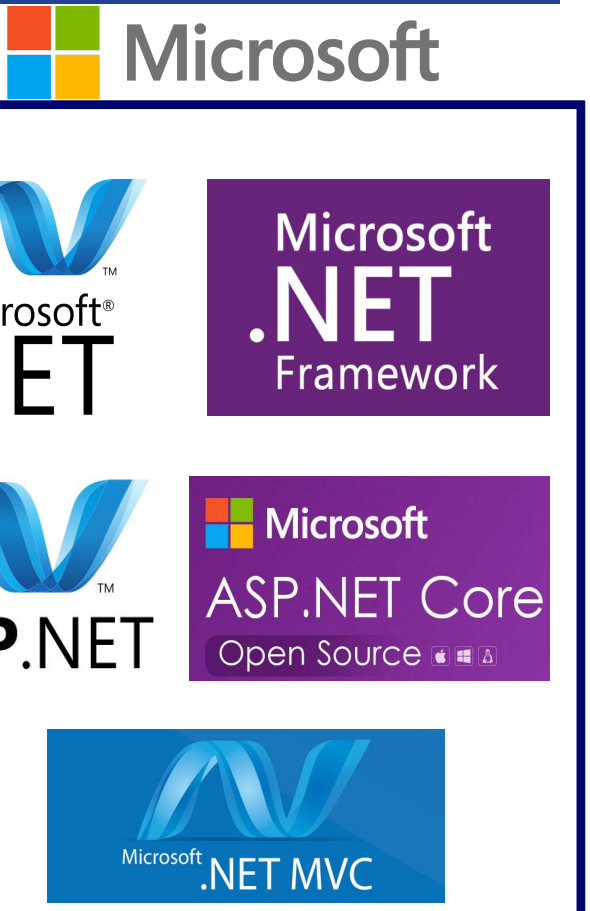
CRONOGRAMA

- OBJETIVOS DA DISCIPLINA
- AVALIAÇÃO
- **CONTEÚDO DA DISCIPLINA**

PLATAFORMA .NET

.NET

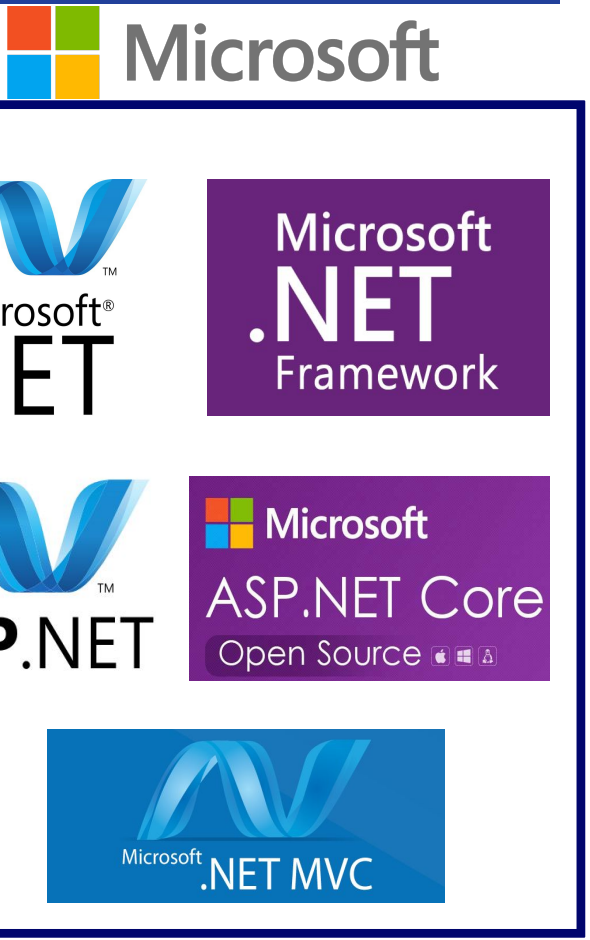
- .NET Framework (1999 - 2000 -> Microsoft)
  - SDK (Software Development Kit - Kit de desenvolvimento de software)
  - Runtime
- .NET Core (2015 - 2016 -> Microsoft)
  - Windows, Mac e Linux
- .NET 5 (2020)
  - Unificação do .NET Framework e .NET Core
- .NET 8 - Link
- TIOBE - Link



PLATAFORMA .NET

.NET

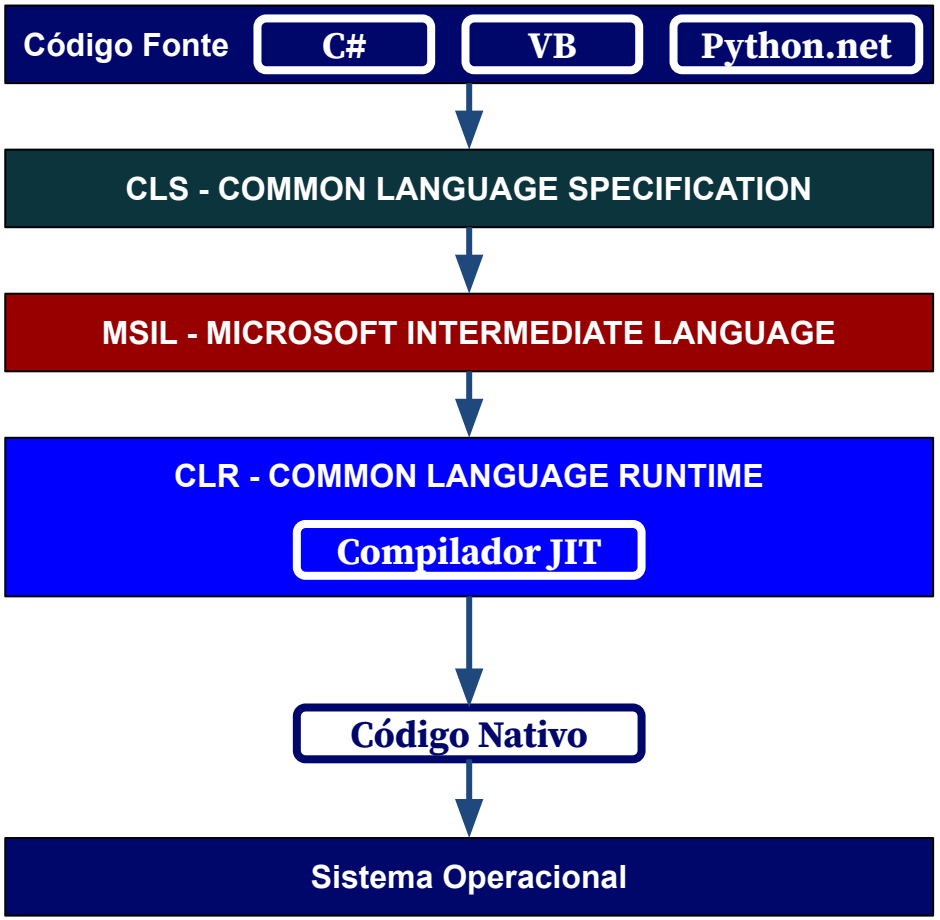
- Desenvolvedor(a) .NET
  - Plataforma suporta diferentes tipos de linguagens
    - C#, F#, VB.NET, COBOL.NET, ...
  - ASP.NET
    - Tecnologia para desenvolvimento de aplicações Web



PLATAFORMA .NET

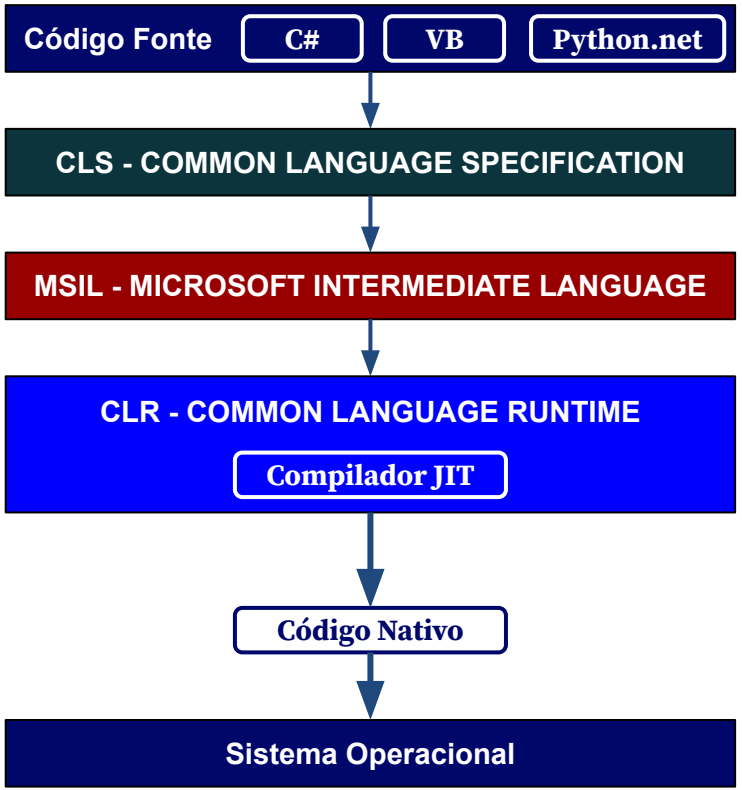
DOCUMENTAÇÃO .NET

- MSIL - MICROSOFT INTERMEDIATE LANGUAGE
- CLR - COMMON LANGUAGE RUNTIME
- CTS - COMMON TYPE SYSTEM
- CLS - COMMON LANGUAGE SPECIFICATION
- BCL - BASE CLASS LIBRARY



PLATAFORMA .NET

- Os sistemas desenvolvidos em .NET são auto-explicativos;
- Cada programa compilado contém em si informações necessárias para que o runtime não precise procurar as informações no registro do Windows, ou de qualquer SO;
- Quando um programa é criado, ele pode ser executado em qualquer máquina (que suporte a .NET).

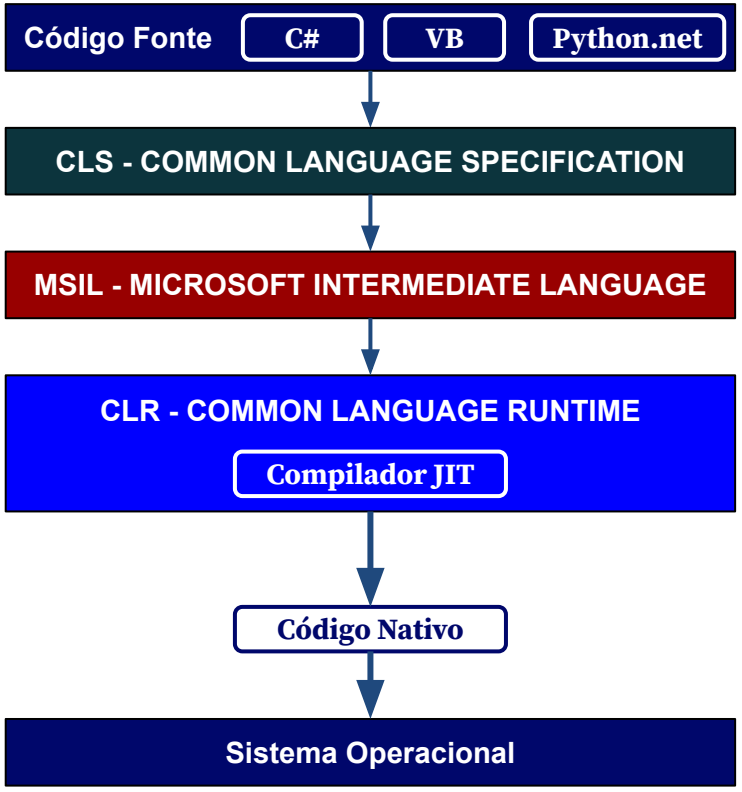


PLATAFORMA .NET

MSIL - MICROSOFT INTERMEDIATE LANGUAGE DOS PONTOS

MSIL

- Linguagem intermediária gerada no processo de compilação do código fonte;
- Código formado por um conjunto de instruções em linguagem intermediária e por metadados;
- Informações essenciais para execução do código, tais como definição de tipos e controle de versão;
- Os programas são multiplataformas, independente de linguagem, seguros, versionados;
- Não é qualquer linguagem que pode ser compilada em .NET;
- Ela deve aderir às especificações da CLS e CTS
  - Ex.: Python.NET, Cobol.NET etc.

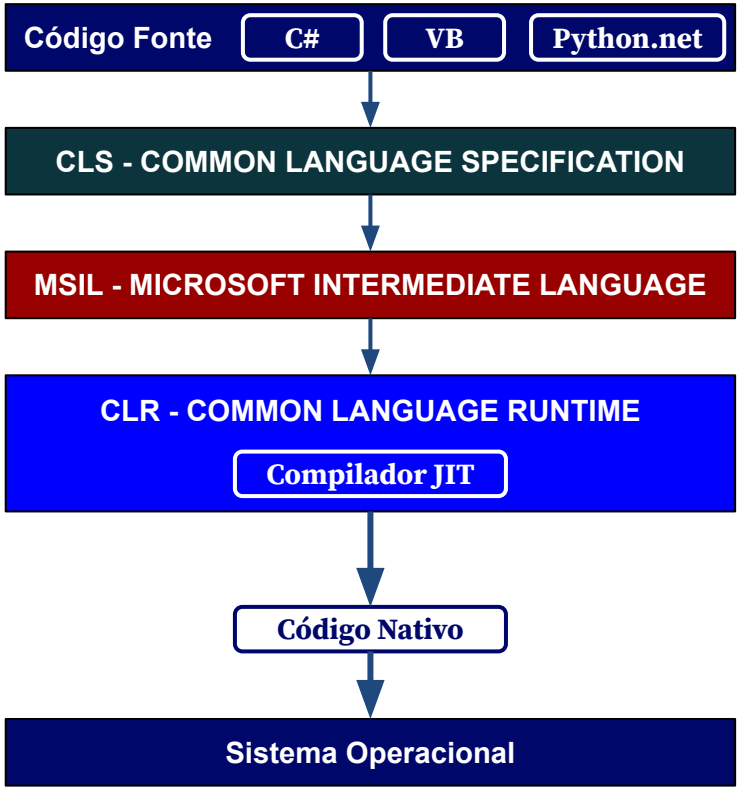


PLATAFORMA .NET

CLR - COMMON LANGUAGE RUNTIME

CLR

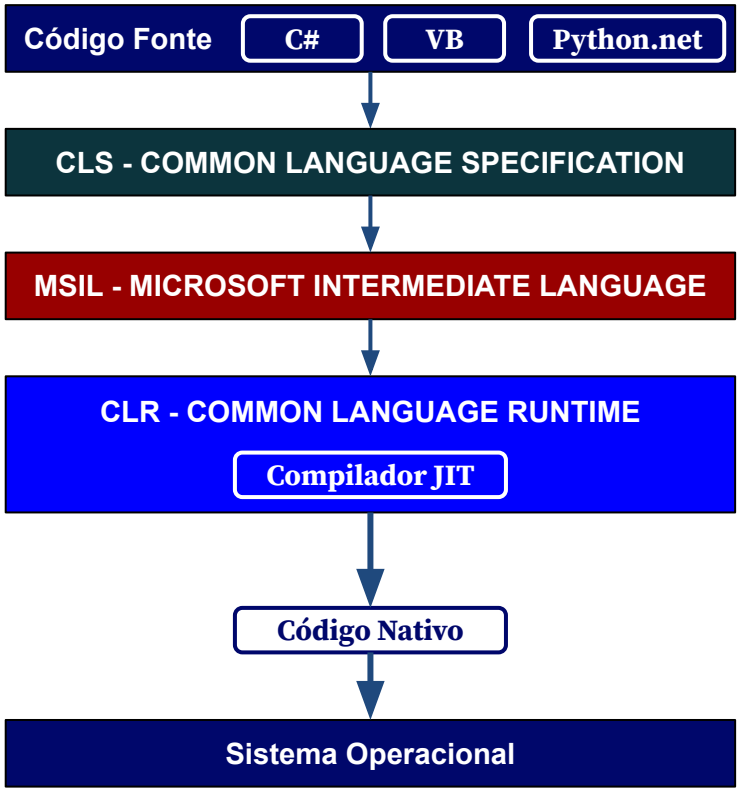
- Ambiente de execução das aplicações .NET, que facilita o processo de desenvolvimento;
- Responsável pelo gerenciamento de memória através do GC (Garbage Collector - Coletor de Lixo);
- Este gerenciamento de memória torna o programa menos propício a erros.
- Responsável pela conversão da sua linguagem para IL;
- Possui o compilador JIT (Just-in-Timer), responsável por interpretar a IL e gerar a linguagem de máquina.



CTS - COMMON TYPE SYSTEM

CTS

- Definição de tipos de dados, onde tudo é um objeto e deriva da classe System.Object, que é o núcleo do sistema de tipos.
- Tipos Valor: variáveis alocadas na pilha e têm como classe base System.ValueType, que por sua vez deriva da System.Object.
  - Estruturas e Tipos Enumerados
- Tipos Referência: variáveis alocadas na memória heap e têm a classe System.Object como classe base.
  - Objeto, Interface e Ponteiros

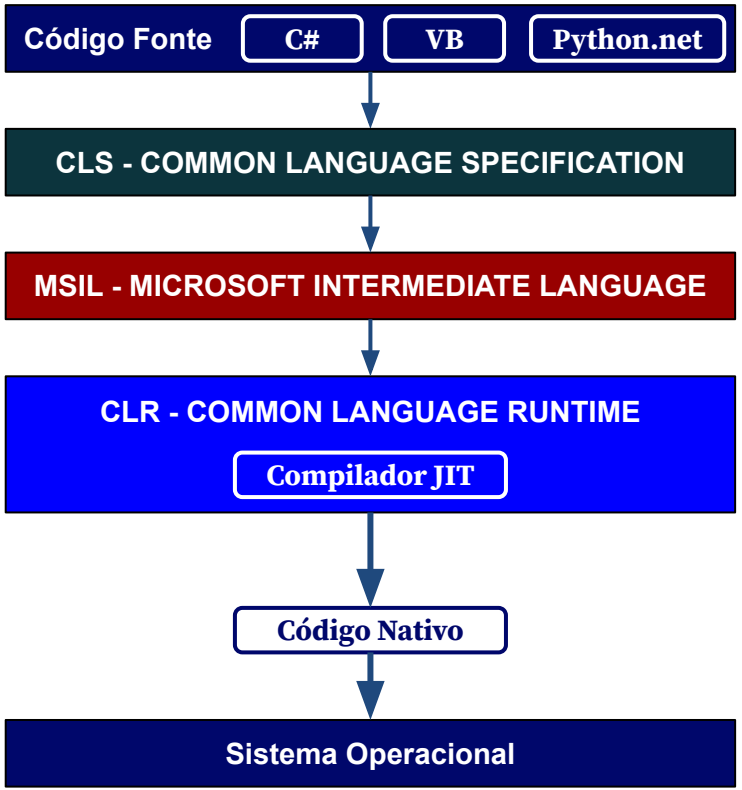


PLATAFORMA .NET

CLS - COMMON LANGUAGE SPECIFICATION

CLS

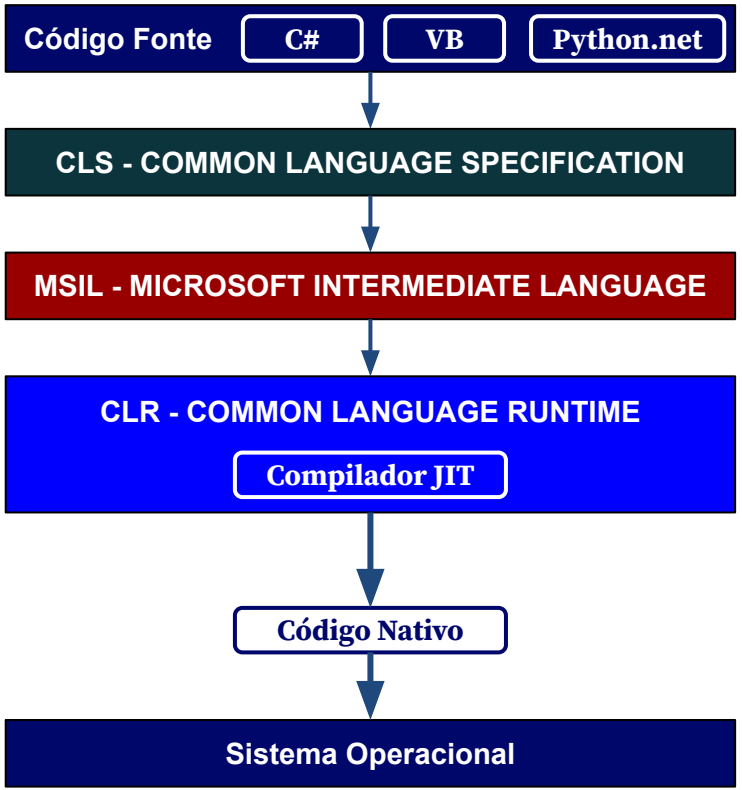
- Define um conjunto de regras que as linguagens que implementam a .NET devem seguir para que a CLR possa gerar a IL;
- Possibilidade de criar sistemas em diferentes linguagens e interagir entre elas dentro da .NET;
- No momento da compilação é gerado um código único intermediário (IL) e todas essas linguagens suportadas pela .NET seguem as regras da CLS, para que depois de gerado a IL, ela seja interpretada corretamente pela CLR.



BCL - BASE CLASS LIBRARY

BCL

- Biblioteca de classe base coma sistema de janelas, biblioteca de entrada/saída de dados, sockets, gerenciamento de memória, etc;
- Biblioteca organizada em uma estrutura conhecida como namespace



PLATAFORMA .NET - C#

DELEGATES

Exemplo de código em C# que utiliza delegates para realizar operações matemáticas básicas (adição, subtração, multiplicação e divisão) com duas classes: Calculadora e Program.

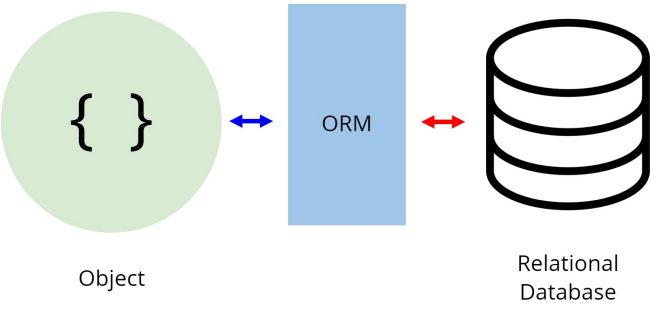
Neste exemplo: Criamos um delegate chamado OperacaoMatematica, que representa uma função que recebe dois números (double) como entrada e retorna um resultado (double).

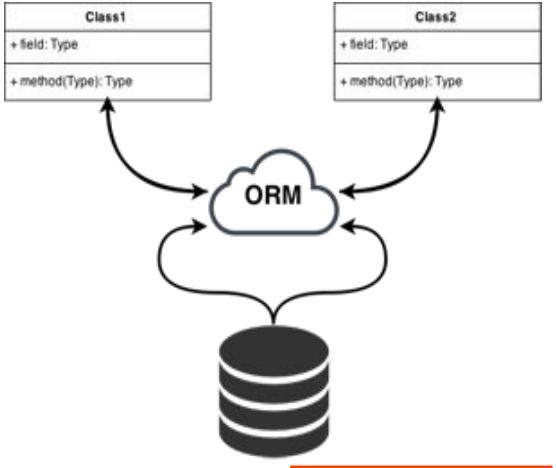
- A classe Calculadora possui quatro métodos que correspondem às operações matemáticas básicas: **Adicao**, **Subtracao**, **Multiplicacao** e **Divisao**.
- No método **Main** da classe **Program**, criamos uma instância da Calculadora e declaramos quatro delegados para representar cada operação.
- Usamos os delegados para realizar operações matemáticas, passando os números necessários como argumentos.
- Exibimos os resultados das operações na tela.

Este exemplo demonstra como usar delegates para encapsular e chamar métodos em diferentes classes de forma dinâmica, tornando o código mais flexível e reutilizável.

MAPEAMENTO OBJETO-RELACIONAL - ORM

- Técnica de conversão das classes da aplicação para tabelas do banco de dados e vice-versa;
  - Conversão entre os objetos da aplicação e as linhas da tabela
  - No código: classes, objetos e métodos
  - No BD: tabelas e registros/linhas

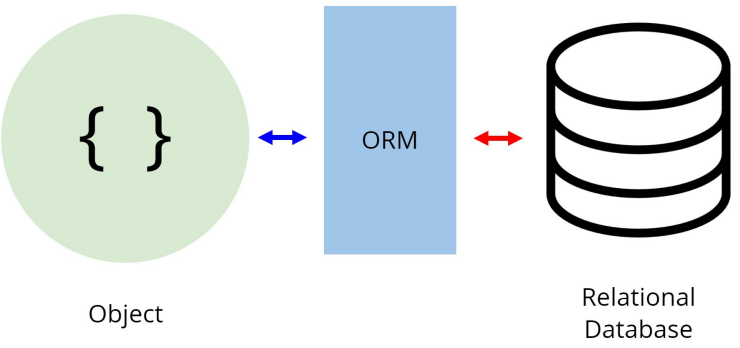


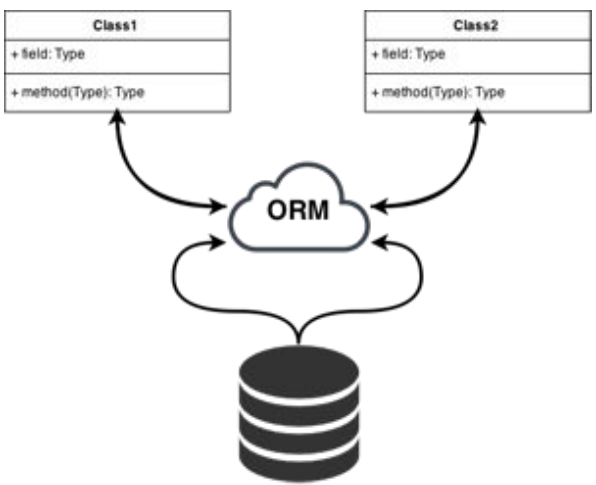


MAPEAMENTO OBJETO-RELACIONAL - ORM

ENTITY FRAMEWORK

- MIGRAÇÕES
- CRUD
  - Create + Read + Update + Delete
  - + Conection + Queries
  - Métodos básicos para interagir entre aplicações e o banco de dados

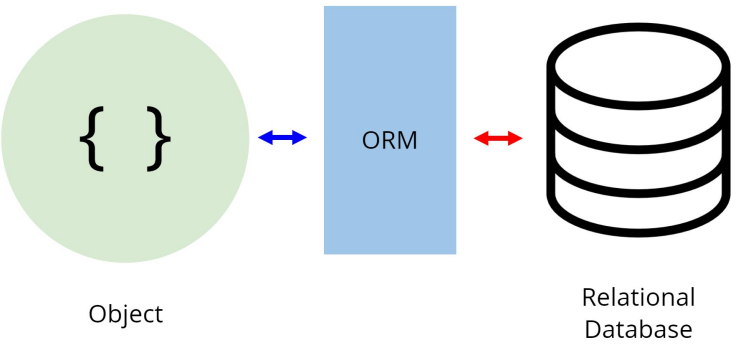


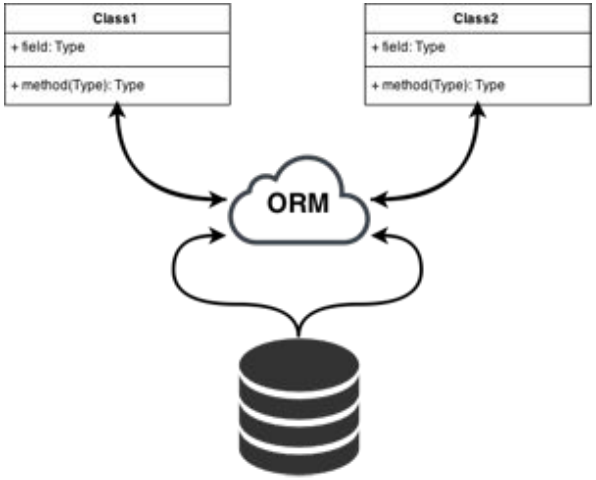


MAPEAMENTO OBJETO-RELACIONAL - ORM

ENTITY FRAMEWORK

- DATABASE FIRST
  - O banco já está criado
  - Mapeamento do banco que já existe para os novos objetos criados
- CODE FIRST
  - Modelo First
  - Começa pelo código
  - Gera o banco automaticamente via Migração

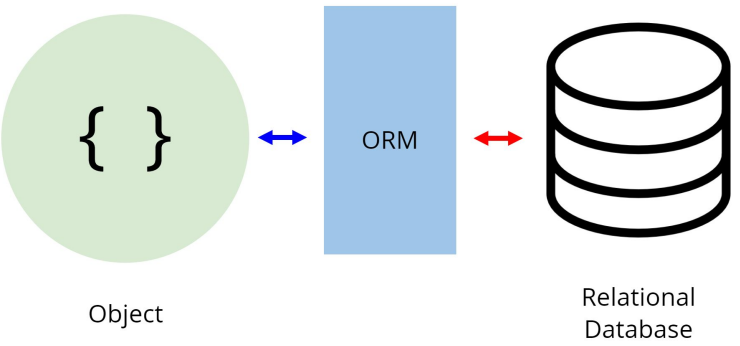


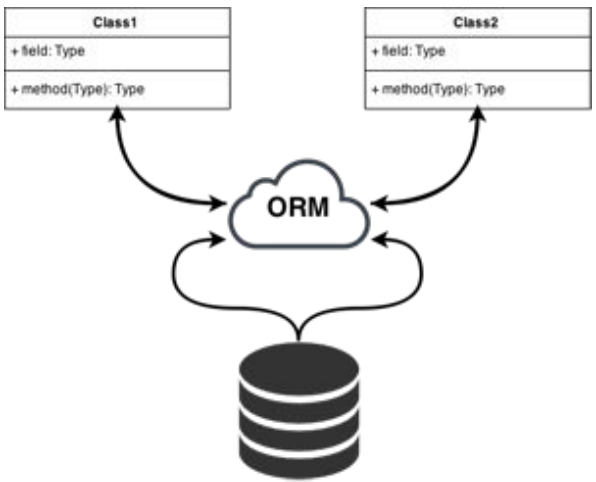


ENTITY FRAMEWORK

DATA CONTEXT

- CONTEXTOS
  - Único objeto que o Entity Framework precisa;
  - Representação do “banco de dados” em memória;
  - Composto por subconjuntos de dados;
    - Propriedades chamados DbSet;



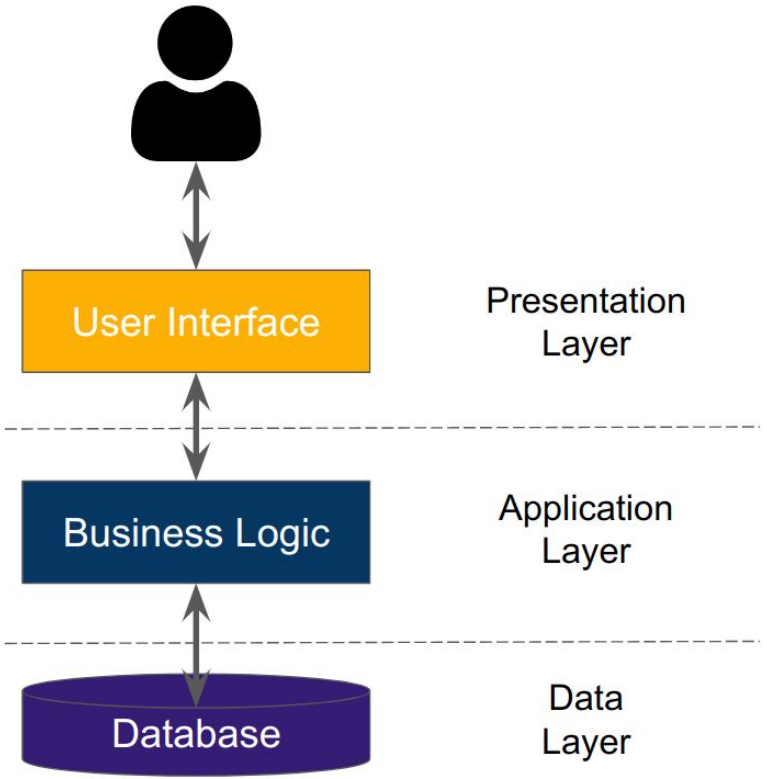


- DATA CONTEXT representa um banco;
- DbSet representa as tabelas;

PROGRAMAÇÃO EM CAMADAS

CAMADA DE APRESENTAÇÃO/INTERFACE
PRESENTATION LAYER - UI

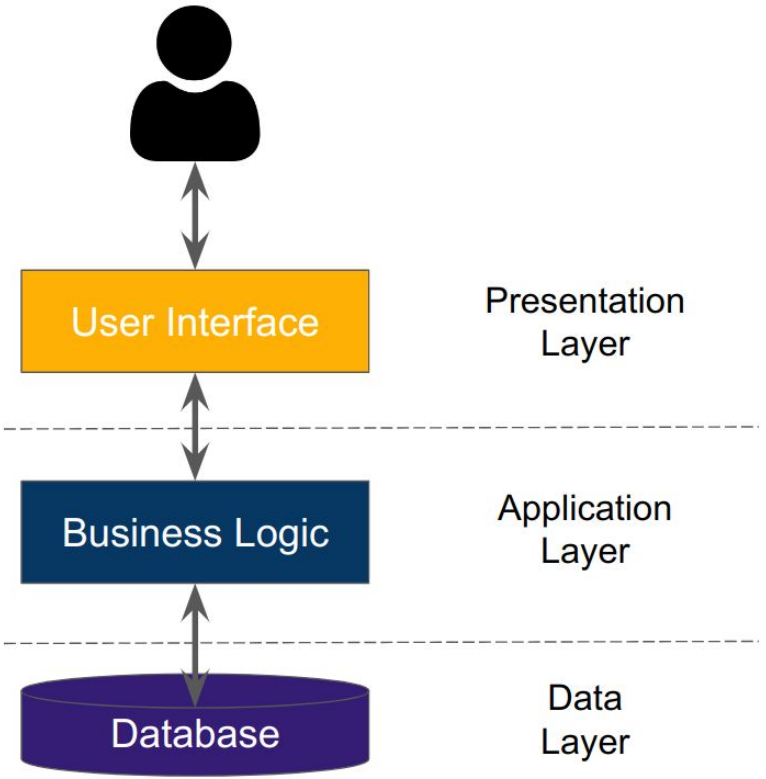
- Apresentação do sistema para o usuário
  - Primeira camada, ou camada superior
  - Comunica-se somente com a BLL (Busines Logic Layer)
  - Não faz acesso direto à DAO (Data Access Object)



PROGRAMAÇÃO EM CAMADAS

CAMADA DE NEGÓCIO - BUSINESS LOGIC LAYER

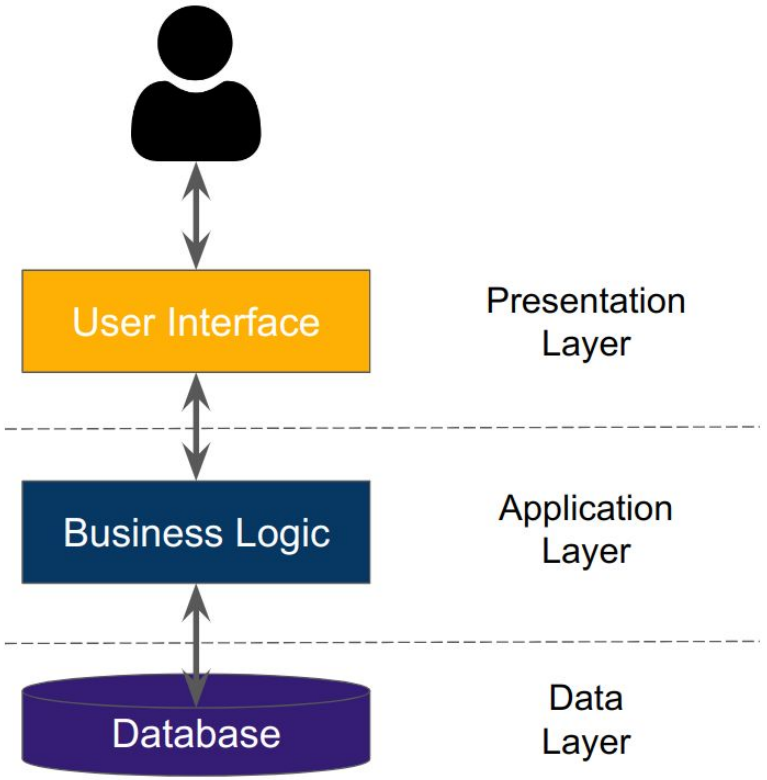
- Camada intermediária
  - Contém toda a lógica de negócios do sistema/aplicativo
  - Regras da interação entre os objetos de negócios
  - Permite a comunicação indireta entre a camada de apresentação e a camada de acesso a dados



PROGRAMAÇÃO EM CAMADAS

CAMADA DE ACESSO AOS DADOS
DATA ACCESS LAYER -DAL

- Mapeamento e persistência de Dados
  - Centralização de todo o acesso ao banco de dados do seu sistema
  - Impõe regras relativas ao acesso aos dados
  - Fornece acesso simplificado aos dados
  - Armazenamento persistente como o SQL Server



MVC EM APLICAÇÕES WEB

Model-View-Controller

- 1979 - Trygve Reenskaug, funcionário da Xerox PARC, nascimento do padrão MVC
- Artigo: *Applications Programming in Smalltalk-80: How to use Model-View-Controller*
  - Padrão de arquitetura com o objetivo de separar o projeto em três camadas independentes.

- Aumento de produtividade;
- Uniformidade na estrutura do software;
- Redução da complexidade no código;
- Estabelece um vocabulário comum de projeto entre desenvolvedores;
- Permite a reutilização de módulos do sistema em outros sistemas;
- Construção de softwares confiáveis com arquiteturas testadas;
- Reduz o tempo de desenvolvimento de um projeto.

ASP.NET MVC

Model

- Responsável por representar as entidades da lógica de negócios da aplicação

View

- Responsável por apresentar uma interface para o usuário

Controller

- Realiza o controle entre os elementos, fornecendo uma ligação entre eles

MVC EM APLICAÇÕES WEB

Model-View-Controller

- 1979 - Trygve Reenskaug, funcionário da Xerox PARC, nascimento do padrão MVC
- Artigo: *Applications Programming in Smalltalk-80: How to use Model-View-Controller*
  - Padrão de arquitetura com o objetivo de separar o projeto em três camadas independentes.

- Aumento de produtividade;
- Uniformidade na estrutura do software;
- Redução da complexidade no código;
- Estabelece um vocabulário comum de projeto entre desenvolvedores;
- Permite a reutilização de módulos do sistema em outros sistemas;
- Construção de softwares confiáveis com arquiteturas testadas;
- Reduz o tempo de desenvolvimento de um projeto.

Model

- Responsável por representar as entidades da lógica de negócios da aplicação
- Ele é responsável por recuperar, armazenar e processar os dados
- Os modelos geralmente mapeiam as tabelas do banco de dados ou outras fontes de dados
- Eles não têm conhecimento sobre a interface do usuário

ASP.NET MVC

View

- Responsável pela apresentação dos dados ao usuário.
- Representa a interface do usuário e a aparência da aplicação
- As visões não contêm lógica de negócios, apenas exibem os dados fornecidos pelo Modelo
- No .NET, as Visualizações são frequentemente criadas usando tecnologias como Razor para ASP.NET MVC ou XAML para aplicativos Windows Presentation Foundation (WPF).

Controller

- O Controlador atua como um intermediário entre o Modelo e a Visualização
- Ele recebe as solicitações do usuário, processa essas solicitações, interage com o Modelo para obter ou atualizar dados e, em seguida, decide qual Visualização deve ser exibida
- Os Controladores contêm a lógica que responde às ações do usuário, como cliques de botão ou solicitações da web

.NET

ASP.NET MVC

- É um framework da Microsoft para desenvolvimento web que segue o padrão MVC. Ele é usado para criar aplicativos web usando o ASP.NET.

ASP.NET Core MVC

- A versão mais recente do ASP.NET MVC, projetada para ser multiplataforma e mais modular.

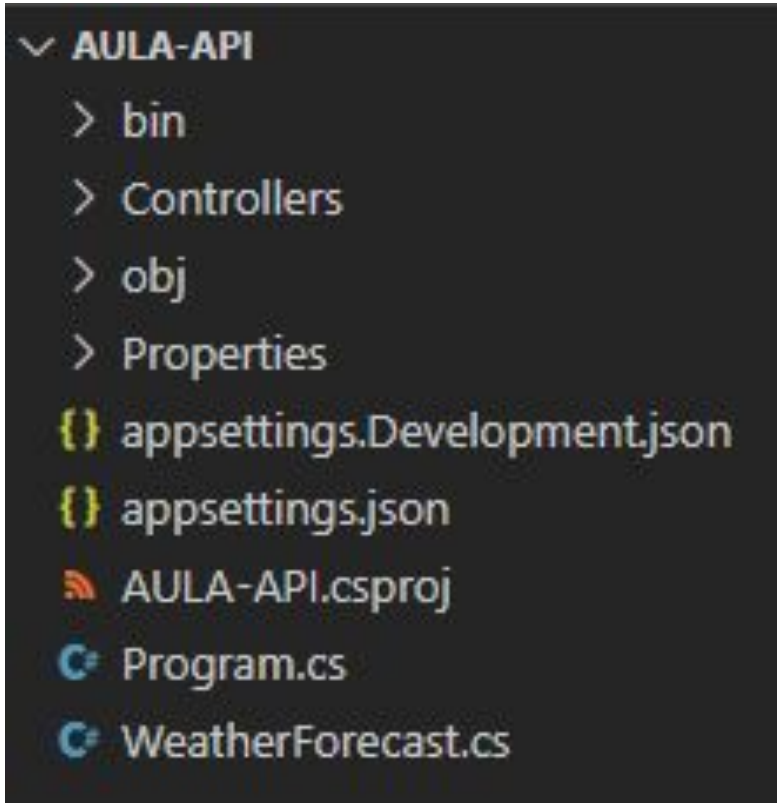
WPF (Windows Presentation Foundation)

- Uma tecnologia da Microsoft para criar aplicativos desktop do Windows que segue o padrão MVVM (Model-View-ViewModel), uma variação do padrão MVC.

Blazor

- Uma tecnologia para desenvolver aplicativos web interativos no .NET que segue o padrão MVVM, semelhante ao WPF.

ESTRUTURA DE UM PROJETO WEBAPI



ASP.NET MVC

- É um framework da Microsoft para desenvolvimento web que segue o padrão MVC. Ele é usado para criar aplicativos web usando o ASP.NET.

ASP.NET Core MVC

- A versão mais recente do ASP.NET MVC, projetada para ser multiplataforma e mais modular.

WPF (Windows Presentation Foundation)

- Uma tecnologia da Microsoft para criar aplicativos desktop do Windows que segue o padrão MVVM (Model-View-ViewModel), uma variação do padrão MVC.

Blazor

- Uma tecnologia para desenvolver aplicativos web interativos no .NET que segue o padrão MVVM, semelhante ao WPF.

Agradecimentos

Obrigado.

Rafael Marinho e Silva
rafaelmarinho@unipam.edu.br