Curso de Extensão

Tecnologias Microsoft

UNICAMP

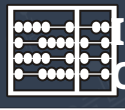
INF-0990

Programação em C#
Aula 1

Prof. Dr. Ricardo Ribeiro Gudwin

gudwin@unicamp.br

27 de Agosto de 2022



INSTITUTO DE COMPUTAÇÃO

Introdução: Origem do C#

Origem do C#

- Linguagens de Máquina
  - Código numérico das instruções de máquina de processadores
  - Programas podem ser carregados diretamente para a memória
- Linguagens de Baixo Nível
  - *Assembly*
  - Mnemônicos das instruções de máquina
  - Pseudo-instruções para inicializações de memória e outras funções
  - Programas precisam ser inicialmente ***montados***, por um **montador**, ou **assembler**, para que possam ser carregados na memória
  - Normalmente essas linguagens são totalmente **dependentes do processador** e seu conjunto de instruções
- Linguagens de Alto Nível
  - Programas precisam ser ***compilados***, por um **compilador**, ou **interpretados**, por um **interpretador**, para que possam ser carregados na memória
  - Linguagens **independentes do processador**

Origem do C#

- Linguagem BCPL
  - *Basic Combined Programming Language*
  - Criada por Martin Richards, da Universidade de Cambridge, nos Estados Unidos, em 1967
  - Linguagem de programação de alto nível, procedural, originalmente desenvolvida para a criação de compiladores para outras linguagens
  - Primeiro compilador do BCPL foi criado para o IBM 7094, sendo depois portado para diferentes computadores, incluindo Honeywell 635, Honeywell 645, IBM 360, PDP-9 e PDP-10, além de outros.
- Linguagem B:
  - Origem no BCPL, foi criada no Bell Labs em 1969 como uma linguagem não tipada de alto nível para o desenvolvimento de aplicações independentes de máquina.

Origem do C#

- Linguagem C:
  - Teve origem na linguagem B, com sua transformação em uma linguagem tipada.
  - Foi criada entre 1972 e 1973, por Dennis Ritchie, também no Bell Labs, e utilizada para o desenvolvimento do sistema operacional Unix.
    - Em 1971, Ritchie começou a melhorar a **linguagem B**, para utilizar recursos do PDP-11.
    - Dentre as inovações, estavam a introdução do tipo “char” para representar caracteres.
    - Ele chamou sua versão modificada de B de “**Novo B**”.
    - Thompson começou a usar o “**Novo B**” para escrever o kernel do Unix, mas os requisitos do projeto foram criando cada vez mais modificações.
    - Por volta de 1972, outros tipos mais ricos foram introduzidos no “Novo B”, incluindo-se arrays de int e char, ponteiros, e a habilidade de gerar ponteiros para os diversos tipos, arrays de tipos e ponteiros para funções.
    - Um novo compilador foi escrito, e a linguagem foi renomeada para **linguagem C**.
  - É uma das linguagens de alto nível mais utilizadas até hoje.

Origem do C#

- Linguagem C++:
  - Foi uma linguagem, originada na linguagem C, para introduzir o conceito de classes da orientação a objetos,
  - Desenvolvida por Bjarne Stroustrup a partir de 1982.
  - Em 1985 foi disponibilizado o primeiro compilador comercial para C++, mas a linguagem só foi standartizada em 1998 com a norma ISO/IEC 14882:1998.
  - Diversos criticismos
    - Expansão excessiva de recursos, tornando a linguagem desnecessariamente complicada (**feature creep**)
      - → A linguagem conta com diversos recursos sofisticados, tais como **herança múltipla**, **overload de operadores** e outros, que podem gerar código de difícil compreensão.
      - → Na prática, os programadores simplesmente não usam esses recursos, utilizando somente parte da linguagem em seus programas.
    - Inexistência de alguns recursos comuns em outras linguagens de programação modernas, como por exemplo, mecanismos de **reflexão** e **garbage collection**.

Origem do C#

- Linguagem Java:
  - Fortemente inspirada no C e no C++, a linguagem Java foi desenvolvida por James Gosling e disponibilizada em 1995 pela **Sun Microsystems**, com a filosofia "**Write Once, Run Everywhere**".
  - A linguagem Java era uma linguagem orientada a objeto (como o C++), mas que trazia um grande número de inovações, principalmente para solucionar diversas das críticas que haviam sido feitas ao C++.
    - Ao invés de ser uma linguagem compilada em código executável, o Java era traduzido em uma linguagem intermediária chamada de **Bytecode**, que correspondia a um tipo de linguagem de máquina de uma máquina virtual.
  - Foram criadas máquinas virtuais Java (**JVM**) para diversas grandes plataformas disponíveis na época, incluindo-se Windows, MacOs, Linux e Solaris.
    - Com isso, um mesmo código de programa conseguia rodar programas virtualmente em qualquer plataforma que contasse com uma **JVM**.
  - Em 1997, com o sucesso e a popularidade crescente do Java, seguindo a estratégia "**Embrace, Extend, and Extinguish (EEE)**", a Microsoft implementa sua própria JVM, com extensões proprietárias que rodavam somente no Windows.
    - Segue-se uma grande batalha judicial entre a Sun e a Microsoft, que a Microsoft perde em 2001. O abandono da JVM Microsoft acaba levando à criação da plataforma **.NET**.

Origem do C#

- Linguagem J++:
  - Como a JVM Microsoft não passava nos testes de conformidade da linguagem Java, a Microsoft acabou criando sua própria linguagem Java, que chamou de J++, pois estava impedida de usar o nome Java.
  - O J++ foi introduzido em 1996 e descontinuado em Janeiro de 2004.
  - O J++ não implementava certas funcionalidades do Java, como RMI e o JNI.
    - Além disso, incluiu algumas extensões, como **callbacks** e **delegates**, além de manipulação de eventos.
  - Aplicações desenvolvidas em J++, entretanto, não rodavam em outras JVMs que não a da própria Microsoft.

Origem do C#

- Linguagem J#:
  - Com a criação do Framework .NET, em 2000, a Microsoft acabou mudando o nome da linguagem J++ para indicar sua compatibilidade com o Common Language Runtime (**CLR**), o que seria o equivalente a uma **JVM**, mas descolada necessariamente do nome Java.
  - Diversas outras linguagens acabaram usando o sufixo # para indicar compatibilidade com o Framework .NET.
  - O J# foi introduzido em 2002 e descontinuado em 2007.
  - Diferentemente do J++, o J# não compilava o código fonte para bytecodes, mas para a linguagem intermediária do CLR.

Origem do C#

- Linguagem C#:
  - Em 1996, No meio da batalha judicial com a Sun, pelo uso do nome Java, a Microsoft contratou Anders Hejlsberg, engenheiro da Borland, que havia colaborado no desenvolvimento do **Turbo Pascal** e do **Delphi**, para criar uma linguagem que pudesse ser como o Java, mas diferente do Java
  - Em 1999 ele inicia o desenvolvimento de uma nova linguagem, que com o início do desenvolvimento do **Framework .NET** em 2000, acaba sendo chamada de C#, supostamente uma linguagem inspirada no C++ e no C (como também era o Java).
  - A primeira implementação do C# torna-se disponível em 2002.
    - As similaridades do C# com o Java, em sua versão 1.0 eram tantas, que o criador da linguagem Java, James Gosling, acusou o C# de ser uma “imitação” do Java.
    - Segundo ele, o C# era “uma espécie de Java em que a confiabilidade, a produtividade e a segurança haviam sido suprimidas”.
    - Outros especialistas em C++ como Klaus Kreft e Angelika Langer também afirmaram que Java e C# eram linguagens de programação quase idênticas.
    - Em 2000, Hejlsberg afirmou que o C# não era um clone do Java, mas muito mais próximo do C++ em seu design.
  - A partir de sua versão 2, em 2005, entretanto, tanto o Java como o C# acabam tomando rumos distintos de desenvolvimento, tornando-se de fato linguagens distintas com diferenças bem marcantes.
  - O C# foi uma das linguagens mais utilizadas para o desenvolvimento do Framework .NET, e é hoje uma das principais linguagens, junto com o VisualBasic, para o desenvolvimento de aplicativos para a plataforma .NET.

A Plataforma .NET

A Plataforma .NET

- O que é o .NET
  - .NET é uma plataforma de desenvolvimento open-source e gratuita, criada pela Microsoft para o desenvolvimento de diversos tipos de aplicações
  - Disponibilizada para diversos sistemas operacionais, incluindo o Windows, o MacOS e o Linux (com diferentes funcionalidades, dependendo do S.O.)
  - Múltiplas Linguagens
    - C#, F#, Visual Basic
  - Múltiplos Editores/IDEs
    - VisualStudio Code, VisualStudio, etc.
  - Bibliotecas para diferentes tipos de aplicações:
    - Web, Mobile, Desktop, MicroServiços, Cloud, Machine Learning, Desenvolvimento de Jogos, IoT, etc.

A Plataforma .NET

- Funcionamento do .NET
  - Semelhante ao Framework Java
  - Programas em C#, F# ou VisualBasic são compilados em uma linguagem intermediária (IL: Intermediary Language), semelhante aos bytecodes do Framework Java
  - Código em IL é interpretado por uma máquina virtual chamada CLR: Common Language Runtime
    - Existem diferentes versões do CLR, dependendo do sistema operacional hospedeiro: Windows, MacOS,Linux,IOS,Android, etc.

A Plataforma .NET

- Diferentes Implementações
  - .NET Framework
    - Primeira versão da plataforma
    - Disponível só para ambiente Windows, de maneira nativa
    - Mono: versão do .NET Framework disponível para Linux e MacOS
  - .NET Core (ou somente .NET)
    - Versão multi-plataforma: Windows, MacOS, Linux
    - Incompatível com o .NET Framework
    - Precisa ser instalado na máquina host
  - .NET Standard
    - Especificação formal das APIs do .NET disponíveis em múltiplas implementações do .NET

- Versões do .NET Framework

Version	Release date	Visual Studio
1.0	2002-01-15	Visual Studio .NET (2002)
1.0 SP1	2002-03-19	
1.0 SP2	2002-08-07	
1.0 SP3	2004-08-30	
1.1	2003-04-09	Visual Studio .NET 2003
1.1 SP1	2004-08-30	
2.0	2005-10-27	Visual Studio 2005
2.0 SP1	2007-11-19	
2.0 SP2	2008-08-11	
3.0	2006-11-06	Visual Studio 2008
3.0 SP1	2007-11-19	
3.0 SP2	2008-08-11	
3.5	2007-11-19	Visual Studio 2008

Version	Release date	Visual Studio
3.5 SP1	2008-08-11	Visual Studio 2008 SP1
4.0	2010-04-12	Visual Studio 2010
4.5	2012-08-15	Visual Studio 2012
4.5.1	2013-10-17	Visual Studio 2013
4.5.2	2014-05-05	Visual Studio 2015
4.6	2015-07-20	Visual Studio 2015
4.6.1	2015-11-30	Visual Studio 2015 Update 1
4.6.2	2016-08-02	Visual Studio 2017 15.0
4.7	2017-04-05	Visual Studio 2017 15.1
4.7.1	2017-10-17	Visual Studio 2017 15.5
4.7.2	2018-04-30	Visual Studio 2017 15.8
4.8	2019-04-18	Visual Studio 2019 16.3
4.8.1	2022-08-09	Visual Studio 2022 17.3

A Plataforma .NET

- Versões do .NET Core (ou só .NET)

Version	Release date	Released with
.NET Core 1.0	2016-06-27	Visual Studio 2015 Update 3
.NET Core 1.1	2016-11-16	Visual Studio 2017 Version 15.0
.NET Core 2.0	2017-08-14	Visual Studio 2017 Version 15.3
.NET Core 2.1	2018-05-30	Visual Studio 2017 Version 15.7
.NET Core 2.2	2018-12-04	Visual Studio 2019 Version 16.0
.NET Core 3.0	2019-09-23	Visual Studio 2019 Version 16.3
.NET Core 3.1	2019-12-03	Visual Studio 2019 Version 16.4
.NET 5	2020-11-10	Visual Studio 2019 Version 16.8
.NET 6	2021-11-08	Visual Studio 2022 Version 17.0

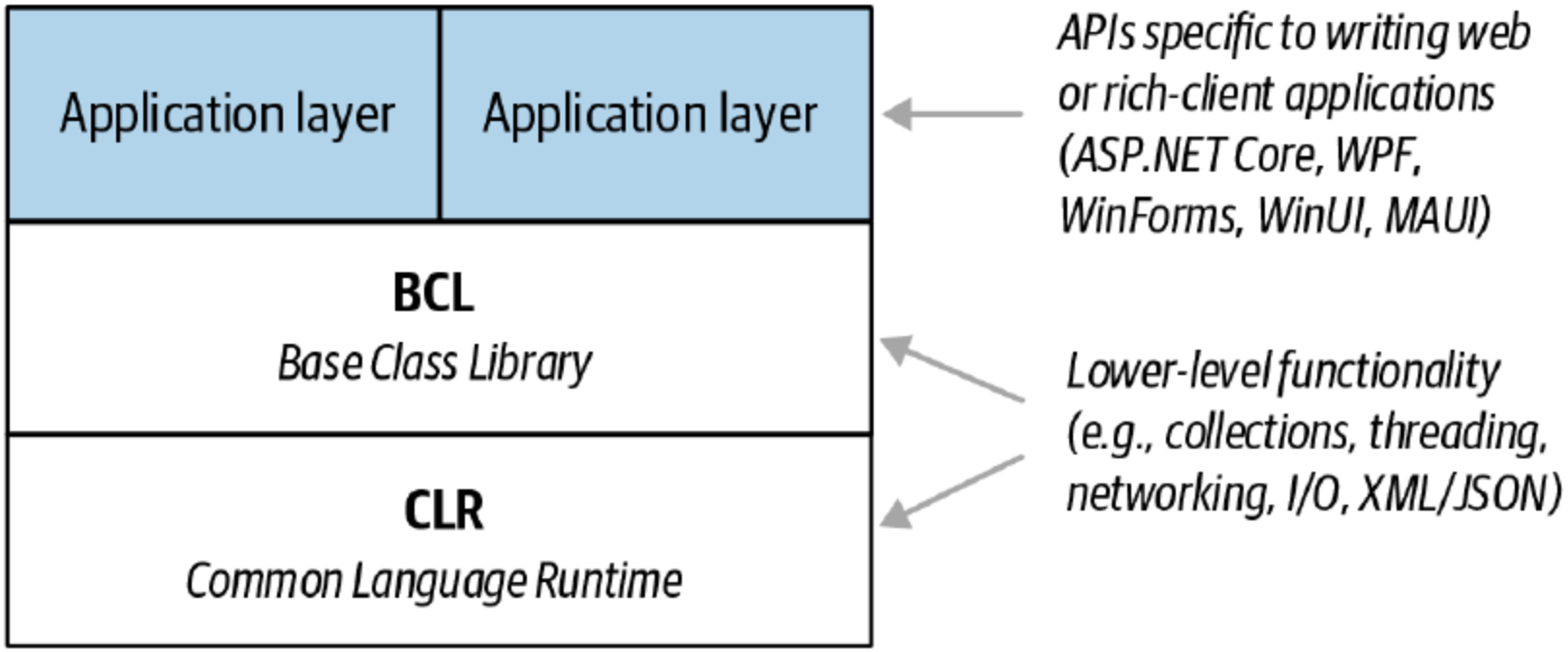
A Plataforma .NET

- Versões do .NET Standard

Implementation	Versions of the .NET Standard								
	1	1.1	1.2	1.3	1.4	1.5	1.6	2	2.1
.NET Core ou .NET	1.0, 1.1, 2.0, 2.1, 2.2, 3.0, 3.1, 5.0, 6.0	1.0, 1.1, 2.0, 2.1, 2.2, 3.0, 3.1, 5.0, 6.0	1.0, 1.1, 2.0, 2.1, 2.2, 3.0, 3.1, 5.0, 6.0	1.0, 1.1, 2.0, 2.1, 2.2, 3.0, 3.1, 5.0, 6.0	1.0, 1.1, 2.0, 2.1, 2.2, 3.0, 3.1, 5.0, 6.0	1.0, 1.1, 2.0, 2.1, 2.2, 3.0, 3.1, 5.0, 6.0	1.0, 1.1, 2.0, 2.1, 2.2, 3.0, 3.1, 5.0, 6.0	2.0, 2.1, 2.2, 3.0, 3.1, 5.0, 6.0	3.0, 3.1, 5.0, 6.0
.NET Framework	4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8	4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8	4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8	4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8	4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8	4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8	4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8	4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8	N/A

A Plataforma .NET

Application layer	CLR/BCL	Program type	Runs on...
ASP.NET	.NET 6	Web	Windows, Linux, macOS
Windows Desktop	.NET 6	Windows	Windows 7–10+
MAUI (early 2022)	.NET 6	Mobile, desktop	iOS, Android, macOS, Windows 10+
WinUI 3 (early 2022)	.NET 6	Win10	Windows 10+ desktop
UWP	.NET Core 2.2	Win10 + Win10 devices	Windows 10+ desktop & devices
(Legacy) .NET Framework	.NET Framework	Web, Windows	Windows 7–10+



Exemplos de Application Layers

- Console
  - Aplicações que interagem com o usuário por meio do Console (Command Shell)
  - Funciona em Windows, Linux e MacOS
- ASP.NET Core
  - Sucessor do ASP.NET (do .NET Framework), é utilizado para a criação de aplicações Web baseadas em REST e microserviços.
  - Pode ser utilizada em conjunto com frameworks the frontend (Angular, React, Vue), para a criação de web-apps.
  - Blazor: código do lado do cliente pode ser desenvolvido em C# ao invés de JavaScript
  - Funciona em Windows, Linux e MacOS
- WPF e Windows Forms
  - Para o desenvolvimento de aplicações com uma GUI
  - Só funcionam em ambiente Windows
- UWP e WinUI 3
  - Para o desenvolvimento de aplicações com GUI e touch first
  - Suportadas em um grande número de *devices* não convencionais que rodam Windows 10, incluindo Xbox, Surface Hub, e HoloLens.
- MAUI
  - Para o desenvolvimento de aplicações mobile em IOS e Android

Mapeamento C# e .NET

Version	Date	.NET	Visual Studio
C# 1.0	01/01/02	.NET Framework 1.0	Visual Studio .NET 2002
C# 1.1 C# 1.2	04/01/03	.NET Framework 1.1	Visual Studio .NET 2003
C# 2.0	11/01/05	.NET Framework 2.0 .NET Framework 2.0	Visual Studio 2005 Visual Studio 2008
C# 3.0	11/01/07	.NET Framework 2.0 (Except LINQ) .NET Framework 3.0 (Except LINQ) .NET Framework 3.5	Visual Studio 2008
C# 4.0	04/01/10	.NET Framework 4	Visual Studio 2010
C# 5.0	08/01/12	.NET Framework 4.5	Visual Studio 2012 Visual Studio 2013
C# 6.0	07/01/15	.NET Framework 4.6 .NET Core 1.0 .NET Core 1..NET Framework 4.6	Visual Studio 2015
C# 7.0	03/01/17	.NET Framework 4.7	Visual Studio 2017 version 15.0
C# 7.1	08/01/17	.NET Core 2.0	Visual Studio 2017 version 15.3
C# 7.2	11/01/17		Visual Studio 2017 version 15.5
C# 7.3	05/01/18	.NET Core 2.1 .NET Core 2.2 .NET Framework 4.8	Visual Studio 2017 version 15.7
C# 8.0	09/01/19	.NET Core 3.0 .NET Core 3.1	Visual Studio 2019 version 16.3
C# 9.0	11/01/20	.NET 5.0	Visual Studio 2019 version 16.8
C# 10.0	11/01/21	.NET 6.0 .NET 6.0.1	Visual Studio 2022 version 17.0

BCL: Base Class Library

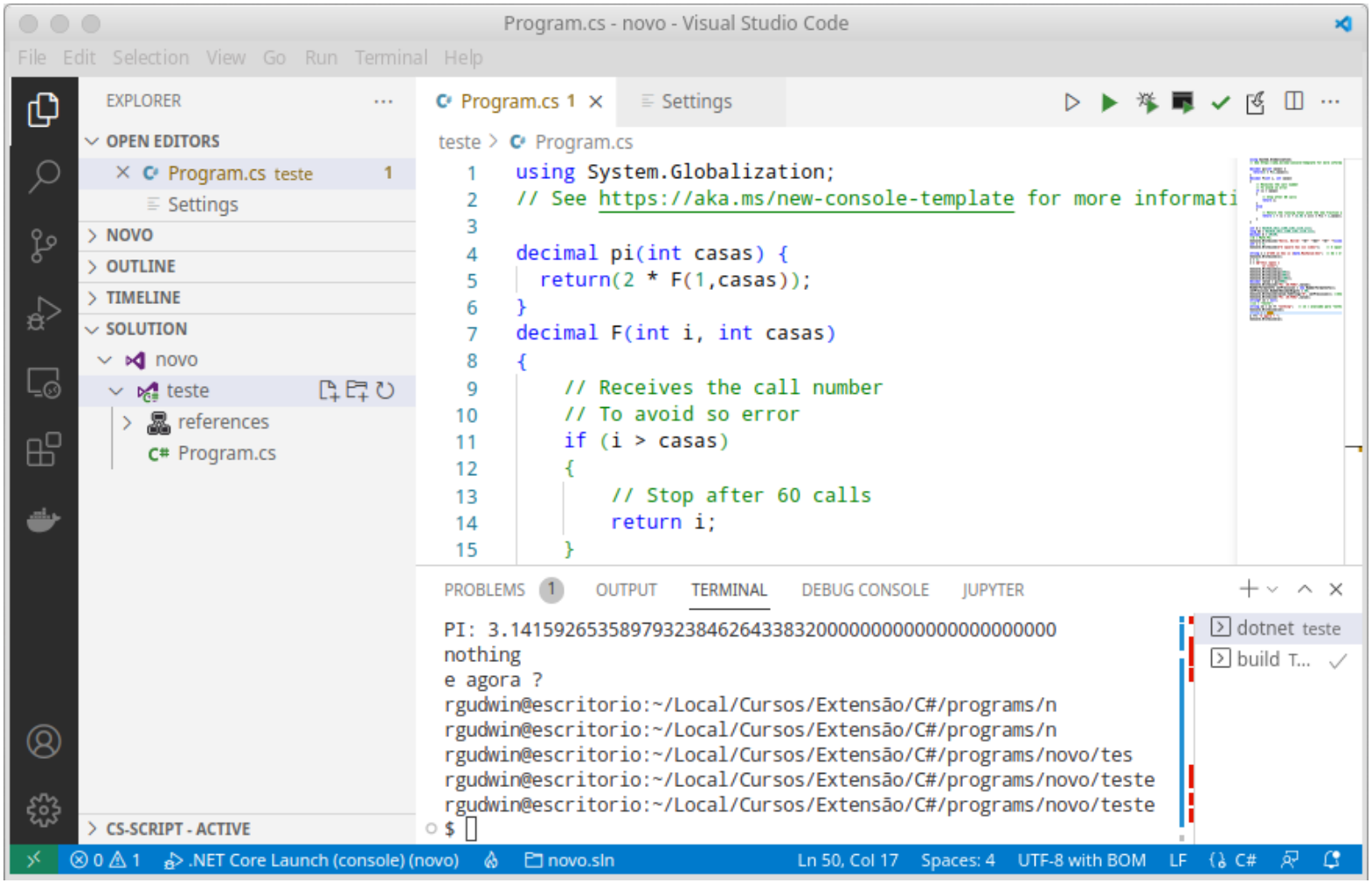
- Processamento de Texto
- Coleções
- Queries para Bancos de Dados
- XML e JSON
- Concorrência e Assincronia
- Streams e Input/Output
- Networking
- Processamento de Assemblies e Reflexão
- Programação Dinâmica
- Criptografia
- Multi-Thread e Programação Paralela
- Expressões Regulares e Serialização

Desenvolvimento em .NET

- .NET SDK (Software Development Toolkit)
  - .NET runtime e biblioteca padrão: CLR e BCL
  - .NET CLI: Command Line Interface
    - Comandos para serem usados no console para diversas finalidades
      - → Criação de *Solutions* e *Projects*
      - → Compilação de Código e geração de Assemblies
      - → Instalação e Gerenciamento de Pacotes Externos
      - → Execução de programas em .NET
- IDEs
  - Visual Studio Code
  - Visual Studio

Desenvolvimento em .NET

- Visual Studio Code



Desenvolvimento em .NET

- Solution
  - É um container, utilizado pelo MS Visual Studio para organizar um ou mais projetos relacionados
  - Definida por um arquivo com sufixo .sln contendo informações sobre os projetos envolvidos e as configurações possíveis para serem construídas
  - Configurações podem definir construções para Debug ou Release e suas plataformas ou CPUs suportadas
  - Pode ser criada pelo CLI do .NET com o comando:
    - `dotnet new sln`

Desenvolvimento em .NET

- Projeto
  - Agrupa um conjunto de classes que funciona de maneira unificada, integrando diversos arquivos com o código fonte necessário para sua compilação
    - Assembly: arquivo .DLL ou .EXE
  - Definido por um arquivo XML com o sufixo .csproj
  - Pode ser uma biblioteca de classes ou um App executável
  - Na criação de um projeto, deve-se especificar um **tipo**, que inicializa o projeto com um template default para o tipo
    - Tipos comuns: “Console App”, “Class Library”, “ASP.NET Web App”, “Test Project”, etc.
  - Opções de Projeto:
    - Self-contained x framework-dependent
    - Especificação do runtime: win-x64, linux-x64, osx-x64, android-arm64, ios-arm64, etc.
    - Executável single-file
    - Compilação ReadyToRun (R2R): torna a execução mais rápida, apesar de ser mais demorado para compilar - JIT compiler.
    - Trimming: excluindo bibliotecas que não são necessárias para o programa

Desenvolvimento em .NET

- NuGet: O Package Manager do .NET
  - `https://www.nuget.org/`
  - Novos pacotes podem ser instalados facilmente com o uso do CLI:
    - `dotnet new --install <NUGET_ID>`
    - Projetos em formato NuGet podem ser criados e disponibilizados no site central do NuGet
  - Posteriormente, esses pacotes podem ser referenciados no projeto, utilizando o comando CLI:
    - `dotnet add package <PACK_NAME>`

Desenvolvimento em .NET

- .NET API Browser
  - Documentação, disponibilizada pela Microsoft, detalhando todas as classes do BCL, que podem ser utilizadas pelos programas em C# para as diversas funcionalidades que podem ser aproveitadas na construção do programa.
    - `https://docs.microsoft.com/en-us/dotnet/api/?view=net-6.0`