# História e Ecossistema .NET

Para criamos códigos em C# é preciso utilizar uma máquina virtual. Assim não é necessário reescrever ou adaptar o código, caso precise levar para outro sistema operacional e/ou outra máquina.

A máquina virtual CLR (common language runtime) está preparada para utilizar a Aplicação e biblioteca em MSIL (Microsoft intermediate language). Então podemos escrever o código em C# e então a máquina virtual vai compilar para a aplicação MSIL.



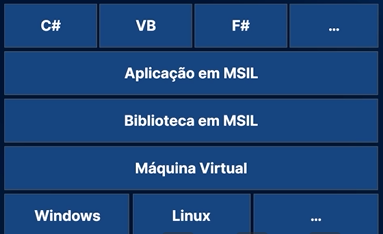
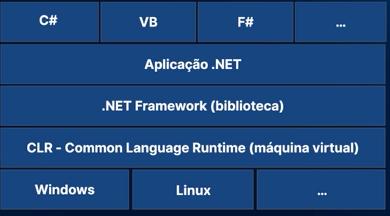
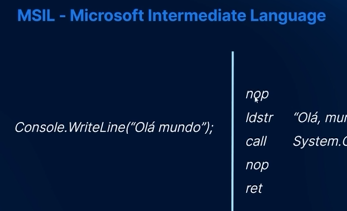
Figura 1 - Processo de Compilação

Figura 2 - Processo de compilação completo



Do Lado direito é como escrevemos em C# e no lado esquerdo é como a máquina compila o que foi colocado no código.

# Primeiro Programa C#

## Console.WriteLine( );

Para mostrar mensagens no console.

Exemplo:

using System;

class Programa

{

static void Main(string[] args)

{

Console.WriteLine("Hello, World!");

}

}

### Compilação Prompt

É possível fazer pelo bloco de notas, assim que salvar este arquivo pelo cmd precisa o caminha da compilação .NET que já tem no computador + o nome do arquivo:

C:\Windows\Microsoft.NET\Framework\v4.0.30319\csc.exe OlaMundo.txt

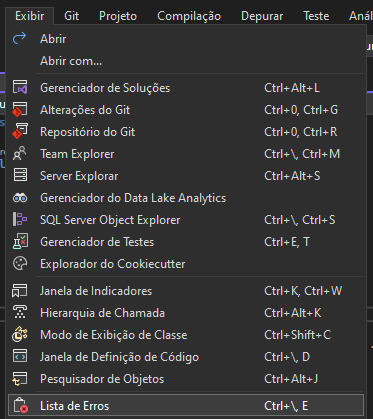
Assim que finalizar a compilação vai ser criado na pasta do TXT um arquivo .EXE, então pelo cmd abra esse arquivo para rodar o código.

Toda vez que fizer uma alteração é necessário fazer essa compilação pelo cmd.

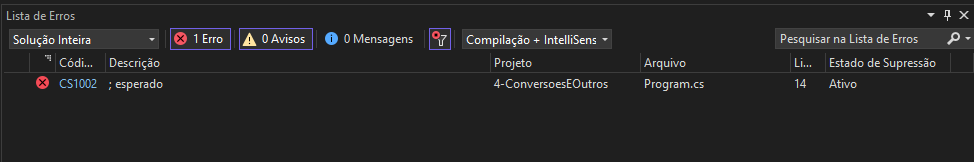
# Verificar erros

No Visual Studio tem uma opção que consegue utilizar para mostrar e localizar seus erros durante a edição.

Para acessar vá em “Exibir” e aperte “Lista de Erros”:



No canto inferior vai abrir uma parte onde consegue visualizar seus erros:



# Primeiro Código VS

### Console.ReadLine();

Aguarda uma entrada pelo teclado do usuário para encerrar o código.

# Criando variáveis Numéricas

Para atribuímos um valor para uma variável, primeiro temos que darmos um tipo a ela.

### int idade

Aceita somente números inteiros, caso tente números decimais ocorre um erro.

### double salario

Aceita tanto número inteiros como decimais, somente fique atento em divisões para garantir que apareça os números decimais coloque .0 mesmo sendo inteiro, para facilitar na hora do resultado.

### long x

Para armazenar números inteiros muito grandes, sendo mais que o *int*.

### short Y

Tipo usado para armazenar poucos números, sendo menos que o *int*.

### float altura

Para número decimais com mais casas, sendo mais preciso que o *double*.

Quanto adicionado um valor com poucos números decimais é recomendado *double*, mas caso queira adicionar com float coloque o f no final do valor.

Ex:

float altura;

altura = 1.62f;

### Converter tipos

Para converter um valor para outro tipo de variável:

double salario;

salario = 3000.12;

int salarioInteiro;

salarioInteiro = (int)salario;

# Caracteres

Para definir uma variável como caractere, é necessário atribuir o **char** e ir adicionando caracteres separados.

Assim que atribuído o valor tem que ser atribuído entre aspas simples ‘ ’.

Ex:

char letra = 'a';

Pode também ser atribuída uma letra em número como na tabela ASCII.

Ex:

letra = (char)65;

Tem como somar os códigos numéricos das letras.

Ex:

letra = (char)(65 + 1);

# Textos

Para adicionar textos em uma variável, é necessário atribuir o tipo string.

Ex:

string primeiraFrase = "Alura = Technology Course";

### Quebra de Texto

Caso precise adicionar uma lista de coisas e precise quebrar a linha para ficar mais visual, é necessário adicionar o @ na frente da variável que está atribuindo.

Ex:

string cursos = @"Curses Alura:

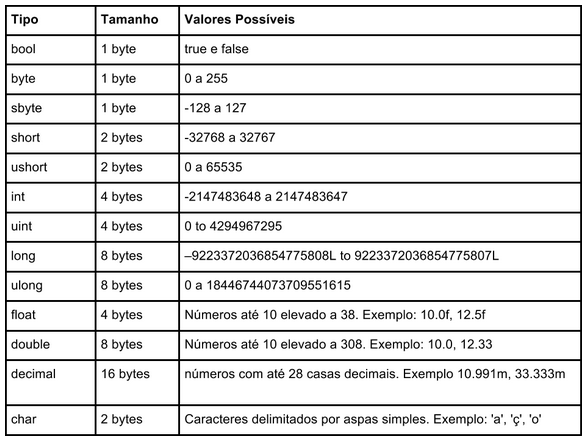
- GO

- C#

- PYTHON

- HTML";

# Tabela de Tipos de Variáveis



Os tipos listados nessa tabela são conhecidos como tipos primitivos ou value types da linguagem C#. Toda vez que atribuímos um valor para uma variável de um tipo primitivo, o C# copia o valor atribuído para dentro da variável.