PROGRAMAÇÃO COM ACESSO A BANCO DE DADOS

C# .NET: Introdução ao C# e Visual Studio 2015

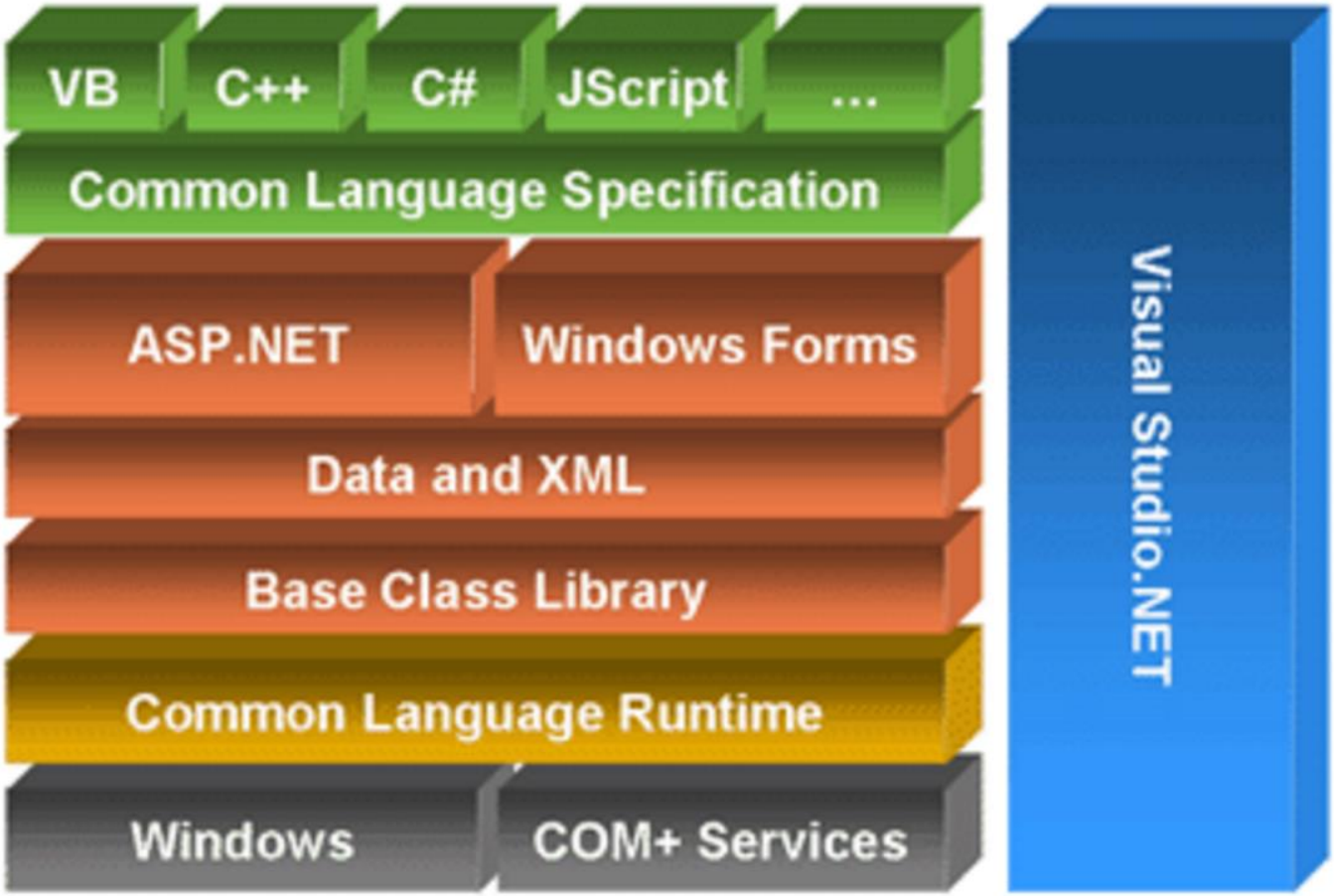
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Msc. Eliezio Soares
eliezio.soares@ifrn.edu.br | https://docente.ifrn.edu.br/elieziosoares

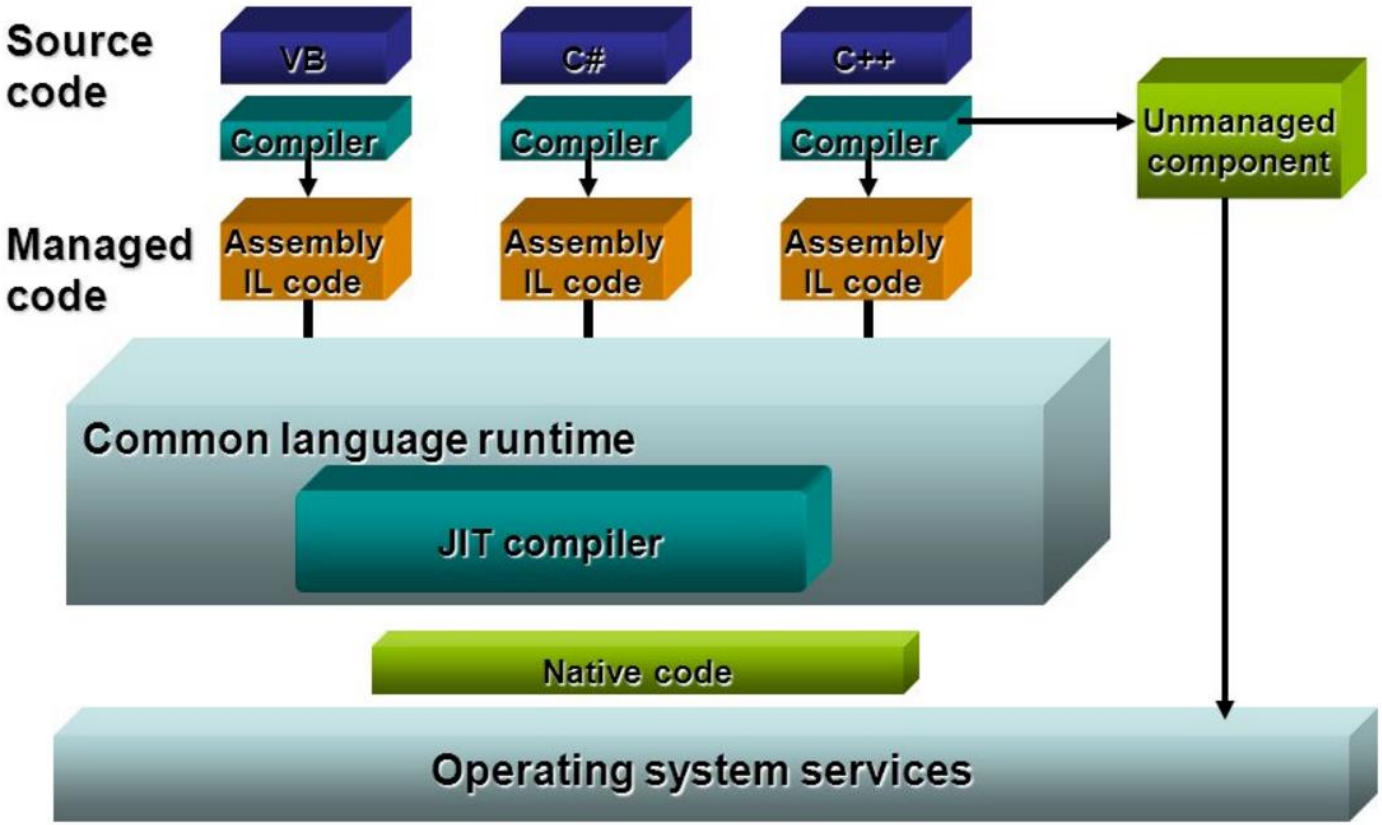
.NET FRAMEWORK

- O .NET Framework é uma plataforma de desenvolvimento para a construção de aplicativos para Windows, Windows Phone ou qualquer plataforma que possua uma implementação do .NET Framework.
- Consiste de uma Common Language Runtime (CLR) e a Biblioteca de Classes .NET que inclui classes, interfaces e tipos.
- Possui um ambiente de execução gerenciada (managed execution environment);
- Visa simplificar:
  - Desenvolvimento de aplicativos.
  - Instalação e distribuição de aplicativos.
  - Integração com outras linguagens de programação (compatíveis com o .NET)

ARQUITETURA.NET



MODELO DE EXECUÇÃO CLR



ARQUITETURA.NET

- Linguagem de Programação

  Não é toda linguagem de programação que pode ser compilada para a linguagem intermediária (Intermediate Language - IL). Ela deve ser compatível com duas especificações, a saber: (i) *CLS - Common Language Specification*, conjunto de recomendações que as linguagens devem seguir e que garante a interoperabilidade entre elas, (ii) *CTS Common Type System,* definição padrão de todos os tipos de dados disponíveis para a *IL*.

- Assembly

  É a unidade de código física resultante da compilação para *IL*. Pode ser representado por um arquivo executável com extensão .EXE, ou por uma biblioteca de ligação dinâmica com extensão .DLL. Todo *assembly* é auto-explicativo através de seus metadados, conforme explicado no item anterior.

- CLR – Common Language Runtime

  É o ambiente de execução das aplicações .NET, gerencia a relação entre o programa e o SO:

  - Gerenciamento de Memória;
  - Mecanismos de Segurança;
  - Tratamento de Exceções;
  - Integração com outras plataformas;

C#
TIPOS DE DADOS

Nome abreviado	Classe do .NET	Type (Tipo)	Width	Intervalo (bits)
byte	Byte	Inteiro sem sinal	8	0 a 255
sbyte	SByte	inteiro com sinal com sinal	8	-128 a 127
int	Int32	inteiro com sinal com sinal	32	-2,147,483,648 to 2,147,483,647
uint	UInt32	Inteiro sem sinal	32	0 a 4294967295
short	Int16	inteiro com sinal com sinal	16	-32.768 a 32.767
ushort	UInt16	Inteiro sem sinal	16	0 a 65535
long	Int64	inteiro com sinal com sinal	64	-9223372036854775508 to 922337203685477507
ulong	UInt64	Inteiro sem sinal	64	0 a 18446744073709551615

C#
TIPOS DE DADOS

Nome abreviado	Classe do .NET	Type (Tipo)	Width	Intervalo (bits)
float	Single	Tipo de ponto flutuante de precisão simples	32	-3.402823e38 para 3.402823e38
double	Double	Tipo de ponto flutuante de precisão dupla	64	-1.79769313486232e308 para 1.79769313486232e308
char	Char	Um único caractere Unicode	16	Unicode símbolos usados no texto
bool	Boolean	Tipo booliano lógico	8	True ou false
object	Object	tipo de base de todos os outros tipos		
string	String	Uma sequência de caracteres		
decimal	Decimal	Preciso tipo fracionário ou integral que pode representar números Decimal com 29 dígitos significativos	128	±1.0 × 10e−28 para ±7.9 × 10e28

C#
DECLARAÇÃO DE VARIÁVEIS

- Linguagem de Programação

  Não é toda linguagem de programação que pode ser compilada para a linguagem intermediária (Intermediate Language - IL). Ela deve ser compatível com duas especificações, a saber: (i) *CLS - Common Language Specification*, conjunto de recomendações que as linguagens devem seguir e que garante a interoperabilidade entre elas, (ii) *CTS Common Type System,* definição padrão de todos os tipos de dados disponíveis para a *IL*.

- Assembly

  É a unidade de código física resultante da compilação para *IL*. Pode ser representado por um arquivo executável com extensão .EXE, ou por uma biblioteca de ligação dinâmica com extensão .DLL. Todo *assembly* é auto-explicativo através de seus metadados, conforme explicado no item anterior.

- CLR – Common Language Runtime

  É o ambiente de execução das aplicações .NET, gerencia a relação entre o programa e o SO:

  - Gerenciamento de Memória;
  - Mecanismos de Segurança;
  - Tratamento de Exceções;
  - Integração com outras plataformas;

C#
ESTRUTURAS DE REPETIÇÃO

- While

```
int i = 0;
while ( i < 5 )
{
        Console.WriteLine ( i );
        ++i;
}
```

Resultado do laço **while**:

```
0
1
2
3
4
```

- Do While

```
int i = 0;
do
{
        Console.WriteLine ( i );

        i++;
}
while ( i < 5 );
```

O laço **Do/While** é quase igual ao laço **While**.

A única diferença é que o código dentro do laço será executado pelo menos uma vez pois a seguir é feita a verificação da condição.

C#
ESTRUTURAS DE REPETIÇÃO

- For

```
int i = 0;
for ( int i = 0; i < 5; i++ )
{
      Console.WriteLine ( i );
}
```

Resultado do laço For:

0
1
2
3
4

- foreach

```
string[] nomes = new string[] {"Programação", "Acesso","Banco","Dados"};
foreach (string n in nomes)
    Console.WriteLine(n);
```

  - O laço foreach itera em uma coleção, como um array, por exemplo.
  - Saída:
    - Programação
    - Acesso
    - Banco
    - Dados

C#
ESTRUTURAS CONDICIONAIS

- If | else if | else

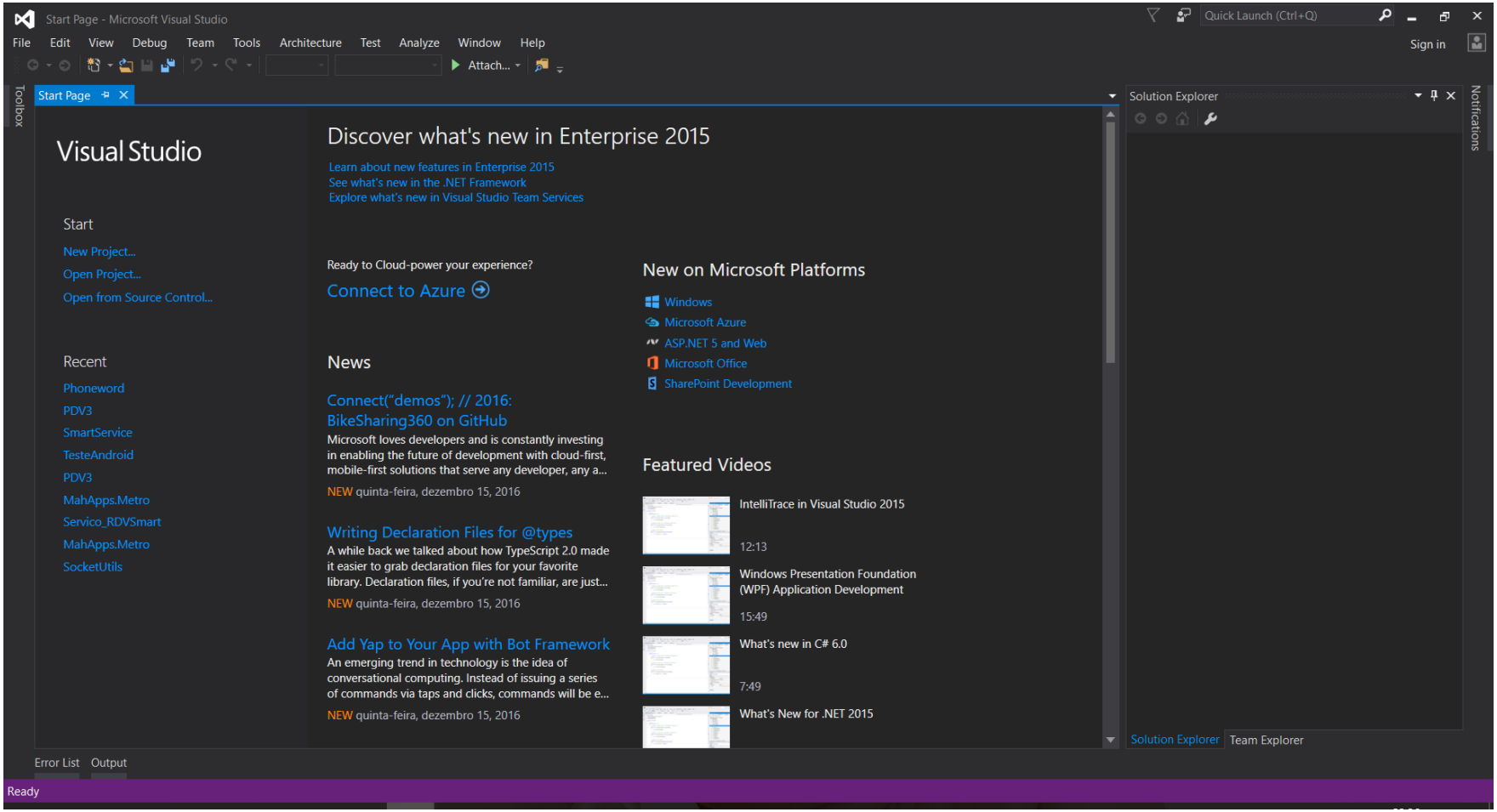
```
string nome = "Programação";
if(nome == "Acesso")
    Console.WriteLine("Bloco IF");
else if(nome == "Banco")
    Console.WriteLine("Bloco else if");
else
    Console.WriteLine("Bloco else");
```

C#
ESCRITA E LEITURA

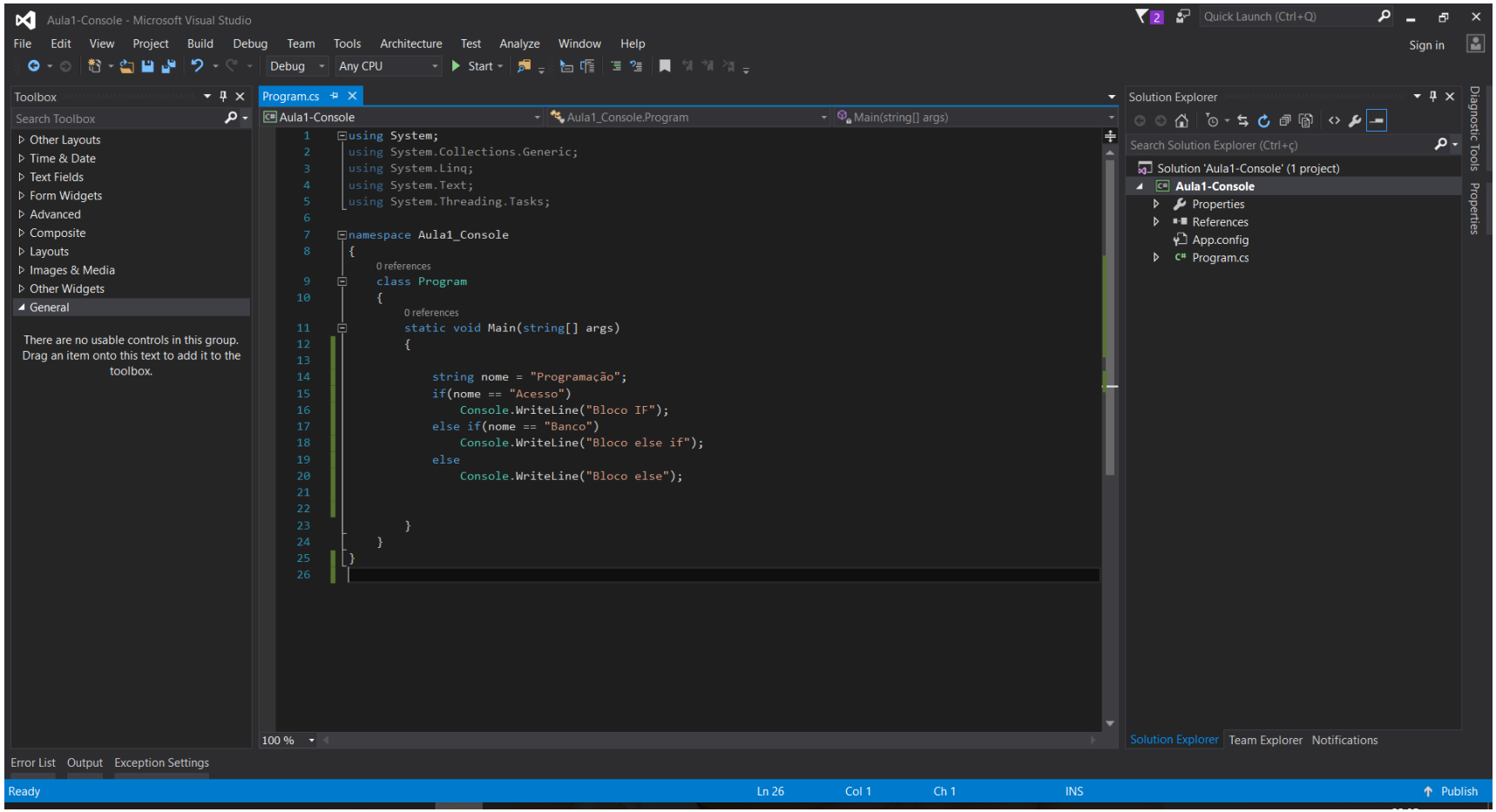
- Console.ReadLine()
  - Efetua a leitura de uma String da entrada padrão.
- Console.WriteLine()
  - Escreve uma String no console.

```
string nome = Console.ReadLine();
Console.WriteLine(nome);
```

VISUAL STUDIO 2015



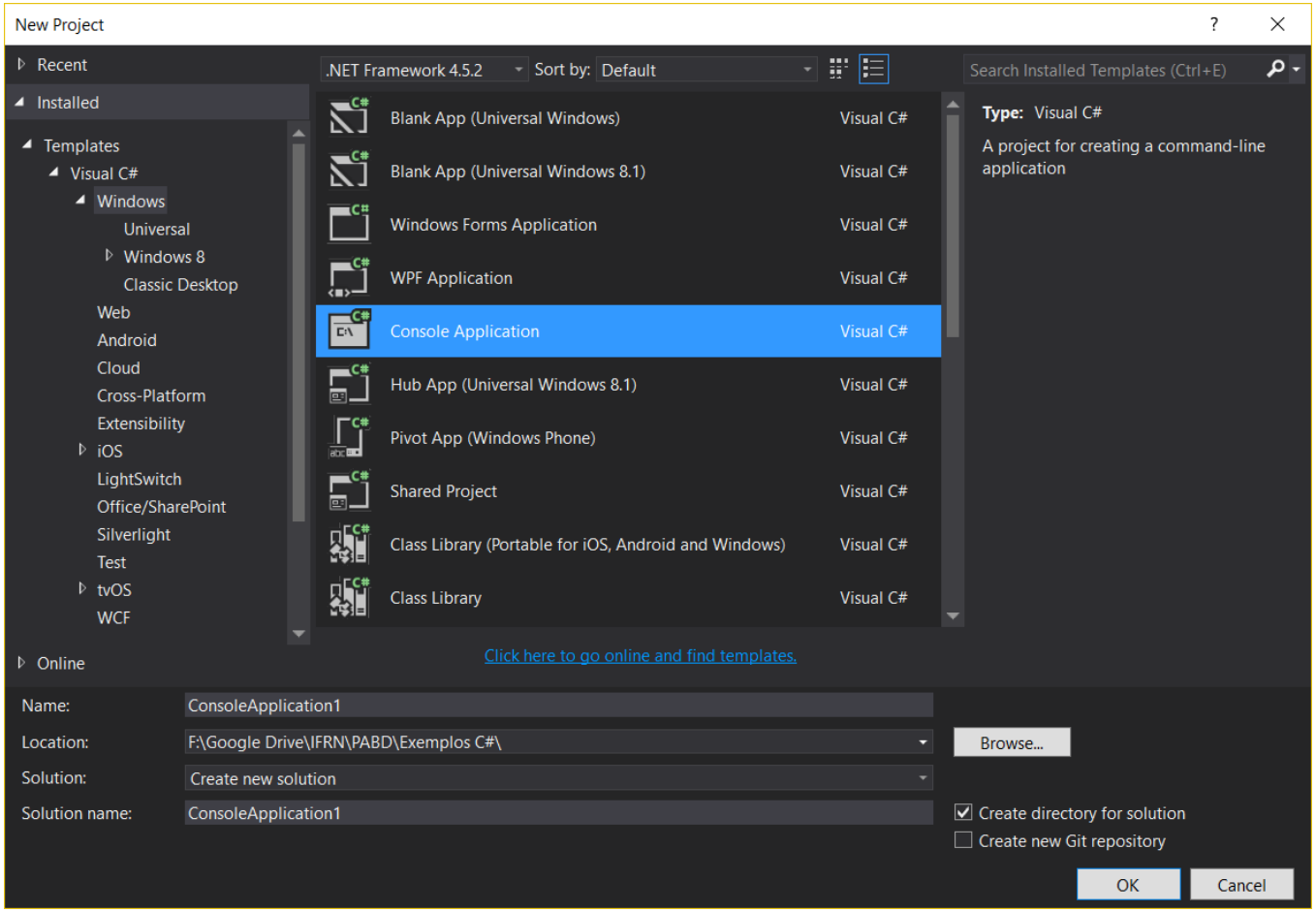
VISUAL STUDIO 2015



EXEMPLO
CONSOLE APPLICATION

- Para criar um projeto:

File > new > Project



EXEMPLO
CONSOLE APPLICATION

A prefeitura de Mossoró abriu uma linha de crédito para os funcionários estatutários. O valor máximo da prestação não poderá ultrapassar 30% do salário bruto. Fazer um programa C# que permita entrar com o salário bruto, o valor do crédito desejado e a quantidade de prestações. A partir dessas informações o programa deve informar se o empréstimo pode ou não ser concedido.

EXEMPLO
CONSOLE APPLICATION

```
float salario, credito, valorParcela;
int parcelas;

Console.WriteLine("Digite o valor do seu salário: ");
salario = float.Parse(Console.ReadLine());
Console.WriteLine("Digite o valor do crédito desejado: ");
credito = float.Parse(Console.ReadLine());
Console.WriteLine("Digite a quantidade de parcelas desejadas: ");
parcelas = int.Parse(Console.ReadLine());

valorParcela = credito / parcelas;
if (valorParcela >= (0.3 * salario))
{
    Console.WriteLine("Não é possível fornecer o empréstimo solicitado.");
    Console.WriteLine("O valor da parcela ultrapassa 30% do seu salário.");
}
else
    Console.WriteLine("Empréstimo concedido.");

string nome = Console.ReadLine();
```

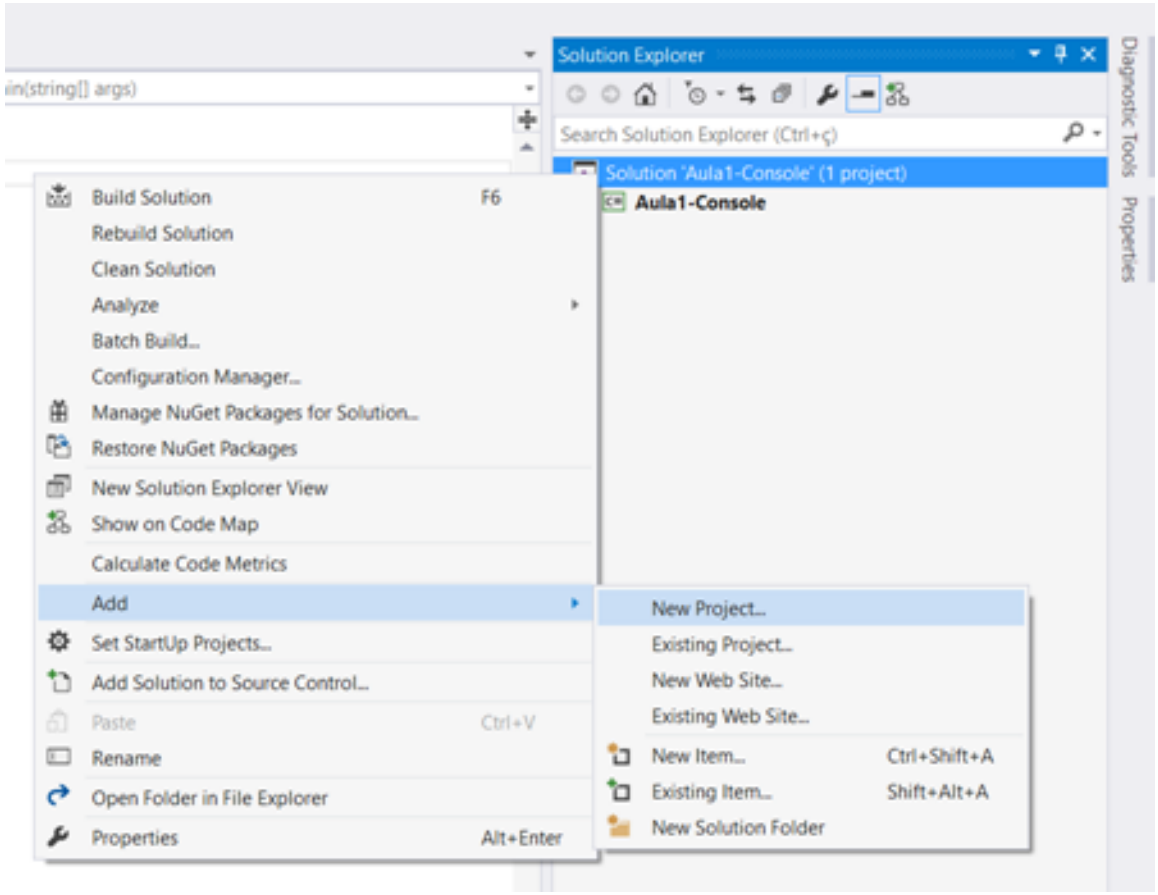
EXEMPLO
WPF - WINDOWS PRESENTATION FOUNDATION

- O WPF *(Windows Presentation Foundation)* é um conjunto de classes que provêem um modelo unificado para construção de aplicações com interface rica, incorporando UI, mídia e documentos.

- Incluída na versão 3.0 em substituição ao Windows Forms;
- Implementa a interface Aero (Windows Vista);
- É composto por:
  - **XAML** (eXtended Aplication Markup Language): Um XML que contém marcações para construção de interfaces gráficas;
  - Código .NET;

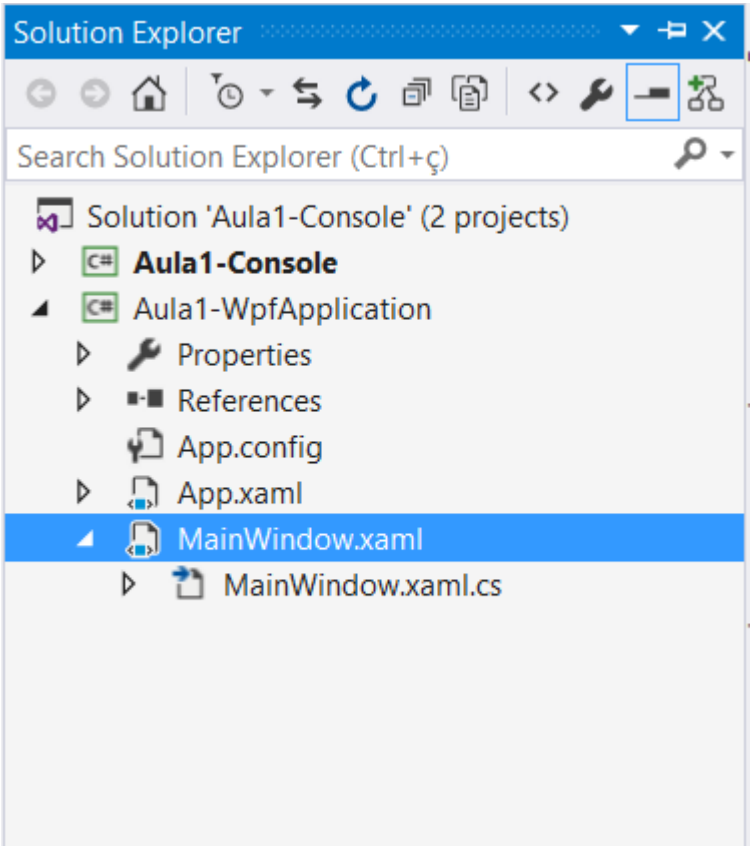
EXEMPLO

WPF - WINDOWS PRESENTATION FOUNDATION



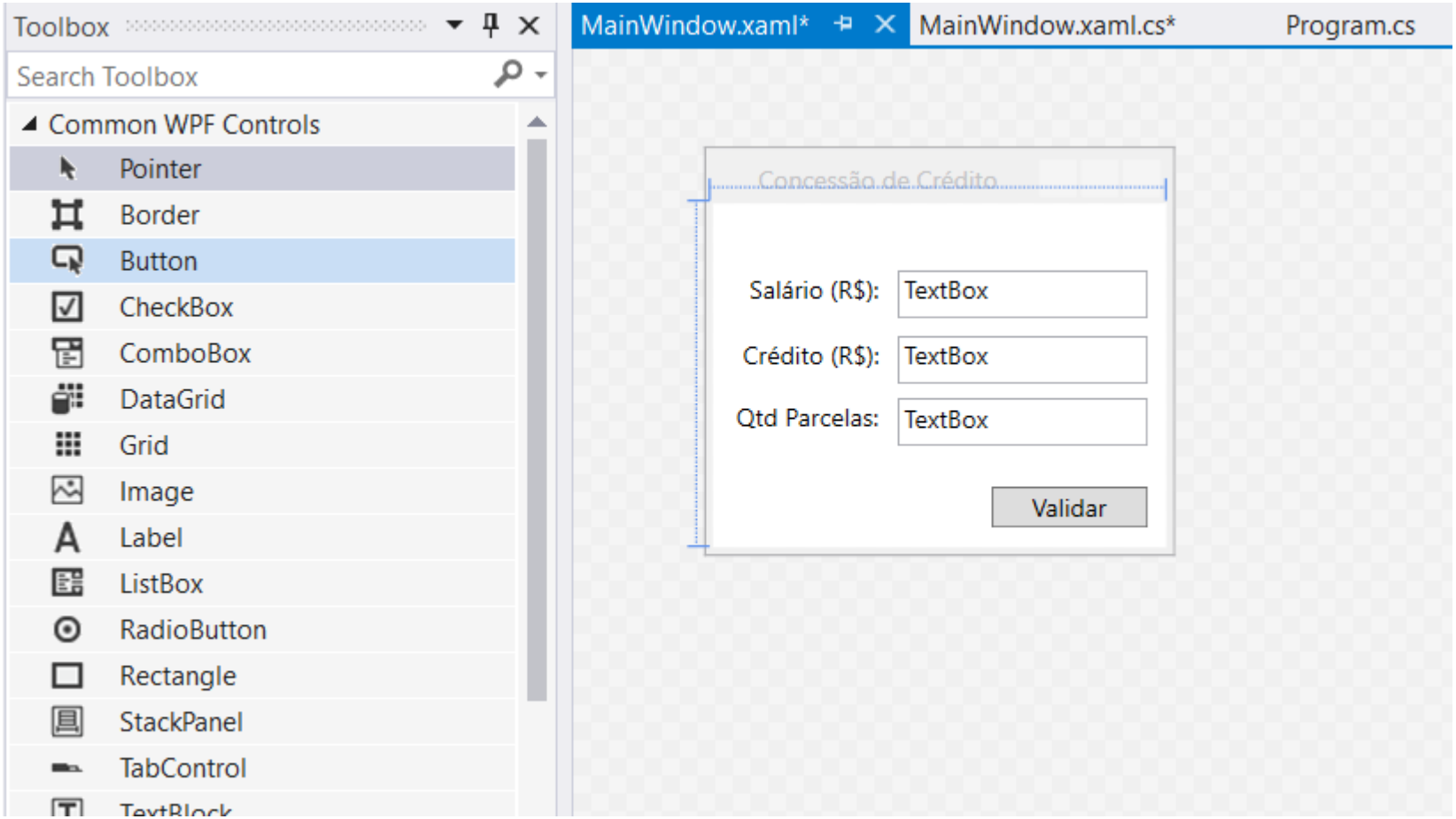
EXEMPLO

WPF - WINDOWS PRESENTATION FOUNDATION



EXEMPLO

WPF - WINDOWS PRESENTATION FOUNDATION



EXEMPLO
WPF - WINDOWS PRESENTATION FOUNDATION

```
private void btValidar_Click(object sender, RoutedEventArgs e)
{
    float salario, credito, valorParcela;
    int parcelas;

    salario = float.Parse(tbSalario.Text);
    credito = float.Parse(tbCredito.Text);
    parcelas = int.Parse(tbQtd.Text);

    valorParcela = credito / parcelas;
    if (valorParcela >= (0.3 * salario))
        MessageBox.Show("Não é possível fornecer o empréstimo solicitado.\nO
    else
        MessageBox.Show("Empréstimo concedido.");
}
```

DÚVIDAS?

EXERCÍCIO

1. Implemente na mesma janela:
   1. Campos para o cadastro de atores e um botão para inserir o registro.
   2. Campos para o cadastro de categoria e um botão para inserir o registro.
   3. Campos para inserção de idioma e um botão para inserir o registro.

- Ao clicar no botão de inserir a aplicação deve mostrar um MessageBox contendo os valores digitados pelo usuário.
- Na próxima aula vamos inserir esses dados no banco de dados.

REFERÊNCIAS BIBLIOGRÁFICAS

Tipos de dados (C# vs. Java). Disponível em: <https://msdn.microsoft.com/pt-br/library/ms228360(v=vs.90).aspx. >Acesso em 15 Dez. 2016.

C# - Sintaxe e Conceitos Básicos. Disponível em: <http://www.macoratti.net/cshp_cb1.htm>. Acesso em 15 Dez. 2016.