Apresentação

Curso:

Programação .NET I

Objetivos:

Introdução ao Framework .NET;

Introdução ao Visual Studio;

Conhecimentos básicos em C#.

Introdução

A linguagem de programação Microsoft Visual C# é denominada como “simples, porém poderosa”. Foi criada com o objetivo principal de desenvolvimento de aplicações usando o Framework Microsoft .NET.

Neste curso iremos tratar uma abordagem básica da linguagem, assim como a evolução da mesma e de seu framework-par.

Introdução

A constituição das sintaxes e características da linguagem C# vieram do casamento entre as melhores características do C++ e as facilidades do Visual Basic. A premissa foi de conseguir com essa junção uma linguagem mais limpa e mais lógica.

A evolução da linguagem, assim como a evolução do framework, se dá da seguinte forma:

Introdução

- **Lançamento do C# 1.0 em 2001**.

- **Visual Studio 2005, C# 2.0:**
  - Genéricos;
  - Iteradores;
  - Métodos anônimos.

Introdução

- **Visual Studio 2008, C# 3.0**
  - Extension methods;
  - Lambda expressions;
  - Language-Integrated Query facility (LINQ).

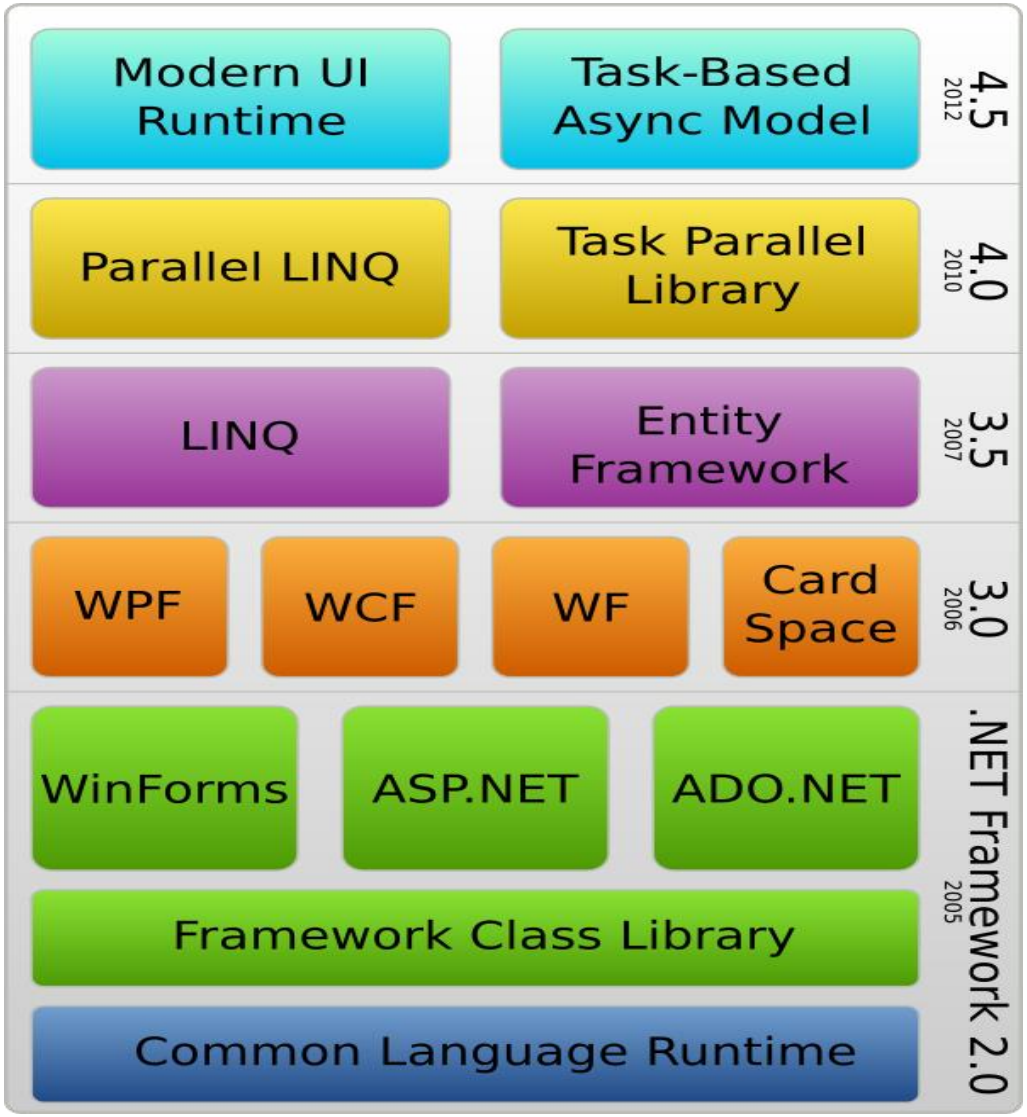
Introdução

- **Visual Studio 2010, C# 4.0:**
  - Maior interoperabilidade entre as linguagens e tecnologias;
  - Inclusão de suporte a nomes e argumentos opcionais;
  - Tipos dinâmicos (Dynamic type);
  - Task Parallel Library (TPL). (Processadores multi-núcleo)

Introdução

- **Visual Studio 2012, C# 5.0:**
  - Método Async (asynchronous task-based processing);
  - Operador “await”.

Introdução



- **WPF:** Windows Presentation Foundation
- **WCF:** Windows Communication Foundation
- **WF:** Windows Workflow Foundation
- **LINQ:** Language Integrated Query

Linguagens .NET

- A#:
  - Departamento da Força Aérea Americana;
  - Portabilidade da linguagem Ada.
- F# (CLI)
  - Desenvolvida pela F# Software Foundation e Microsoft;
  - É uma linguagem open-source. Muito utilizada como *"cross-platform"* e também pode gerar códigos para JavaScript e GPU.
- L# (CLI)
  - Desenvolvida por Rob Blackwell.
  - É uma linguagem dinâmica, criada para ser compilada e executada no Ecma-334 e Ecma-335

Linguagens .NET

- C++ (CLI)
  - Criada pela Microsoft;
  - Destinada a substituir as extensões para C++
- Boo (CLI)
  - Desenvolvida por Rodrigo B de Oliveira;
  - Inspirada na sintaxe do Python.
- Cobra
  - Desenvolvida por Charles Esterbrook;
  - Influenciada pelas linguagens: Objective-C, Eiffel, Python, C#.
  - Open-source (MIT license feb/2008).
- IronLISP
  - Foi uma implementação da linguagem Lisp em Jul/2007.

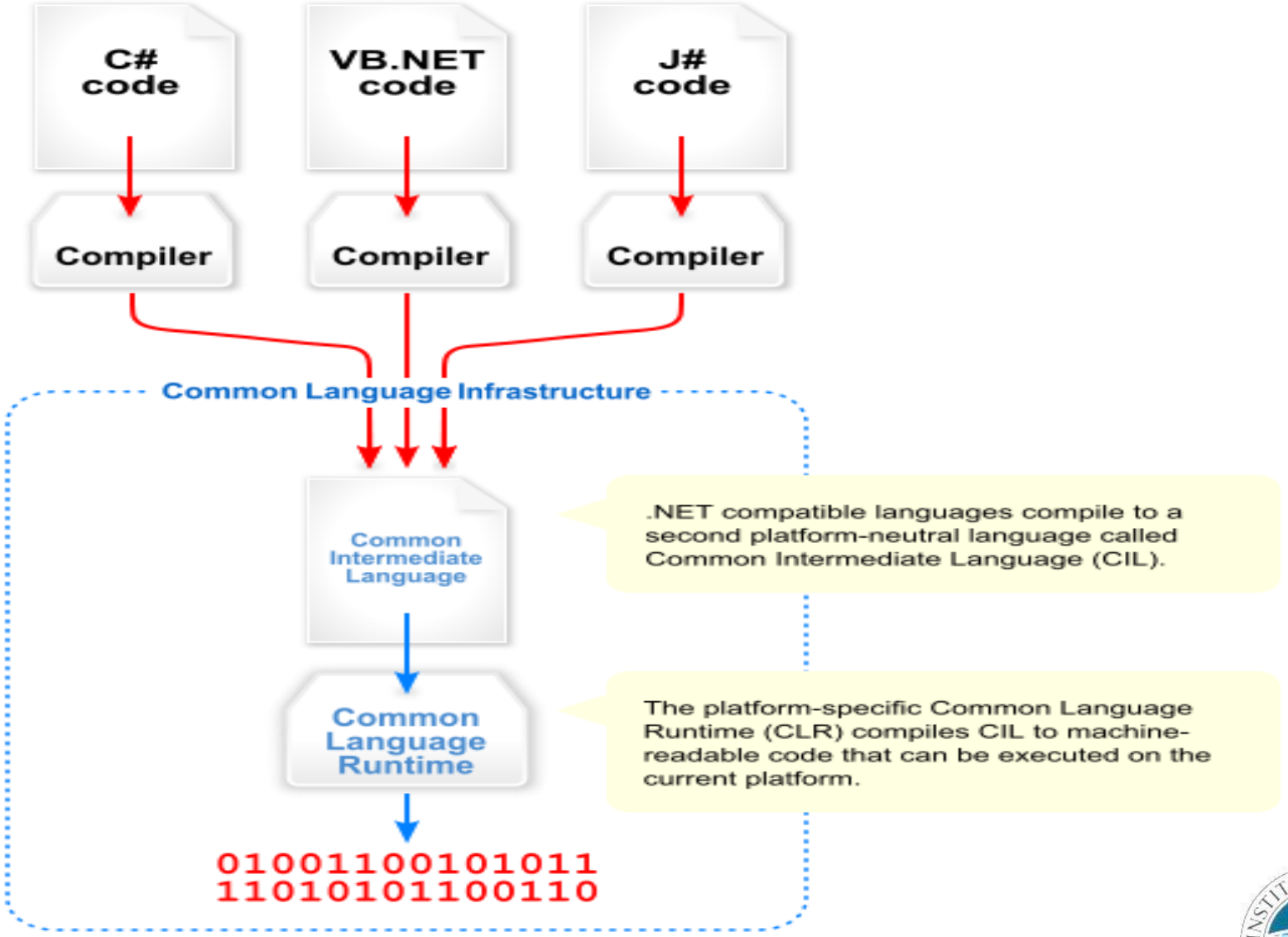
Linguagens .NET

- JScript .NET
  - Desenvolvida pela Microsoft;
  - Fundamentada na tecnologia ActiveX/COM.
- IronPython/IronRuby
  - Criada por Jim Hugunin, que manteve o projeto sozinho até a versão 1.0.
  - Hugunin, junto com uma pequena equipe na Microsoft assumiu o projeto até a versão 2.7 Beta. Foi abandonada no final de 2010 quando Hugunin foi trabalhar na Google.
- Visual Basic .NET (CLI)
  - Criada pela Microsoft;
  - É vista como uma evolução do Visual Basic clássico (baseada em ActiveX/COM).

Arquitetura do .NET

- Common Language Infrastructure (CLI):
  - Desenvolvida pela Microsoft e padronizada pela ECMA e ISO.
  - Foi especificada para definir um ambiente que permita múltiplos níveis de linguagens para ser usada em diversas bases de plataformas (mobile, desktop, internet).
  - É responsável por descrever o código executável e o ambiente de execução, o que forma o núcleo do Framework .NET.

Arquitetura do .NET



Arquitetura do .NET

- Common Language Infrastructure (CLI):

Metadata

Descreve o nível mais alto da estrutura do código. Descrição de todas as classes e os membros das classes que são definidos no *"assembly"*. Também são descritos os membros das classes que serão chamados por outro *"assembly"*.

Arquitetura do .NET

- Common Language Infrastructure (CLI):

Common Language Specification (CLS)

São especificações básicas para que cada linguagem que queira ser CLI-compatível deve estar em conformidade.

Isto garante a interoperabilidade com outra linguagem CLS-compatível. As regras CLS definem um subconjunto da Common Type System.

Arquitetura do .NET

- Common Language Infrastructure (CLI):

Common Type System (CTS)

É o conjunto de tipos de dados e operações que são compartilhadas por todas as linguagens CTS-compatíveis.

Arquitetura do .NET

- Common Language Infrastructure (CLI):

Virtual Execution System (VES)

O Sistema de execução virtual carrega e executa os programas CLI-compatíveis, utilizando os metadados (metadata) para combinar, separadamente, os pedaços de códigos na execução (runtime).

Arquitetura do .NET

- Common Language Infrastructure (CLI):

Assembly

Um *"Assembly"* na CLI é um pedaço de código compilado utilizado para instalação (implantação), versionamento e segurança.

Existem dois tipos de *"Assembly"*: **de processos** (**EXE**) e **de bibliotecas** (**DLL**).

Arquitetura do .NET

- Common Language Infrastructure (CLI):

Assembly

Um *"Assembly"* de processo utilizará uma classe definida numa *"Assembly"* de biblioteca.

O *"assembly"* possui códigos gerados a partir de uma linguagem CLI, e então são traduzidas para a linguagem de máquina no momento da execução pelo compilador em tempo de execução (just-in-time compiler).

Arquitetura do .NET

Pesquisa sobre Segurança e Gerenciamento de Memória na Common Language Infrastructure (CLI):

- Segurança:
  - Principais mecanismos
    - Code Access Security (CAS)
    - Validation and Verification
- Gerenciamento de memória.
  - Automatic Memory management
  - Alocação e liberação de memória

Arquitetura do .NET

- Common Language Runtime (CLR):
  - A *Common Language Runtime* é o componente de máquina virtual do framework .NET e é responsável por gerenciar as execuções dos programas feitos em .NET.
  - Utiliza um processo conhecido como *"Just-In-Time compilation (JIT)"*, os códigos compilados são convertidos em linguagem de máquina para serem executados pelo processador do computador.

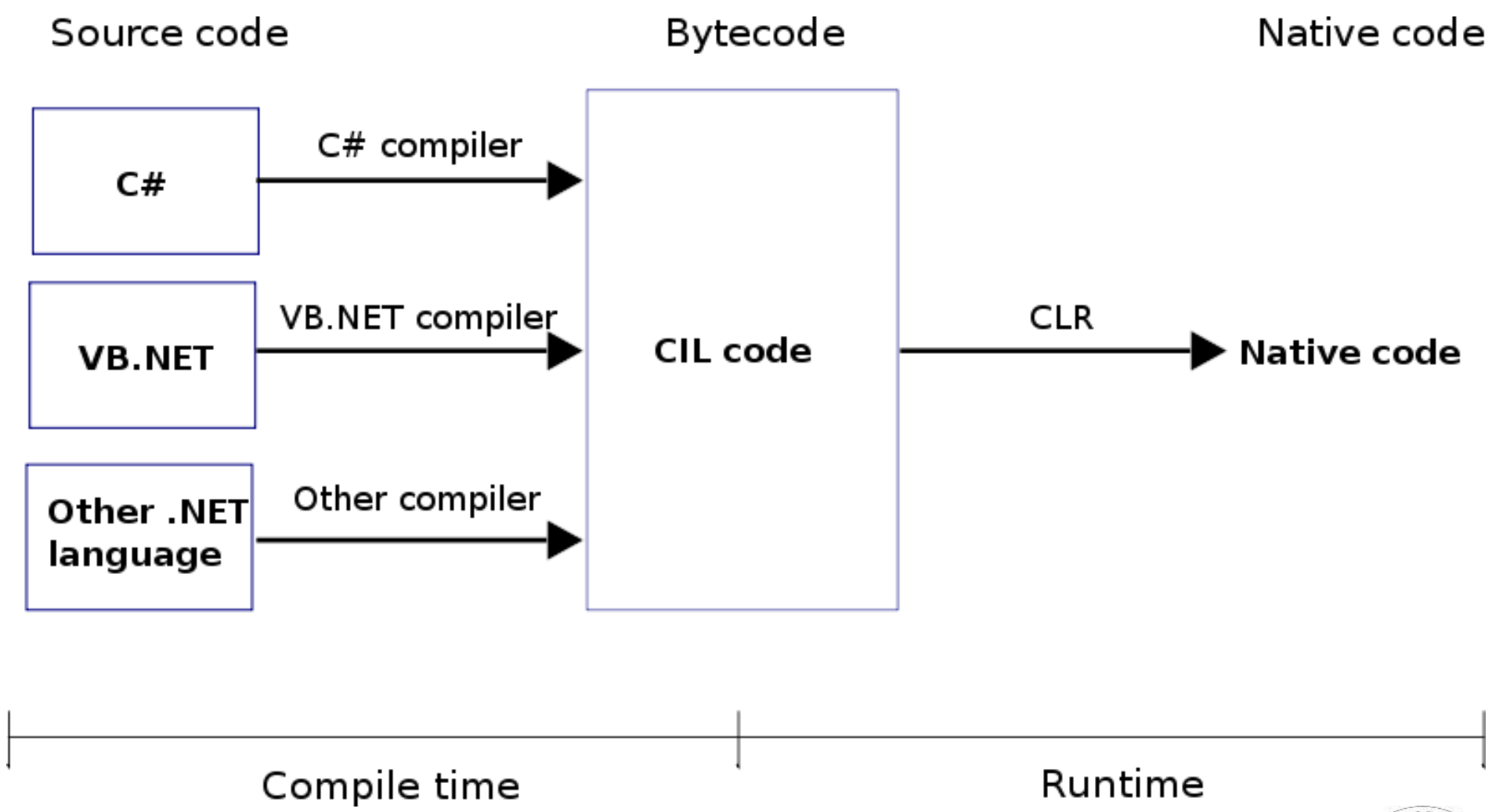
Arquitetura do .NET

- Common Language Runtime (CLR):
  - A CLR fornece serviços adicionais, tais como: gerenciamento da memória, elementos de segurança, gerenciador de exceções, coletor de lixo e gerenciamento de threads.
  - A CLR é a implementação do padrão Common Language Infrastructure (CLI).

Arquitetura do .NET

.NET Framewok version	CLR version
4.6	4
4.5.2	4
4.5.1	4
4.5	4
4	4
3.5	2.0
3.0	2.0
2.0	2.0
1.1	1.1
1.0	1.0

Ciclo de vida .NET



Resumo de Características da C#

- É orientada a objetos.
- Possui um alto nível de abstração.
- Possui coletor de lixo.
- Suporta “tipagem” dinâmica e estática.
    - Dinâmica: Utilizando a palavra *“var”.*
    - Estática: Utilizando o tipo antes do nome da variável.

Primeiros passos no .NET

- O .NET possui uma IDE padrão, conhecida como Visual Studio.
- O Visual Studio oferece diversas facilidades para um melhor desempenho do programador. Isto serve para que o mesmo utilize o seu tempo no que realmente é necessário:
    - Resolver problemas de negócio!

Primeiros passos no .NET

- Para sentir o quanto uma boa IDE influencia, vamos fazer um exercício.

- O nosso exercício consiste em executar uma compilação de um programa em C# sem utilizar o Visual Studio.

Primeiros passos no .NET

- Abra o notepad
- Digite “notepad” no executar (Windows+R) e dê enter

Primeiros passos no .NET

- Digite o seguinte código no notepad:

```
using System;

class HelloWorld
{
    public static void Main()
    {
        Console.WriteLine("Hello World!");
    }
}
```

- E salve o arquivo no “desktop” com o nome “hello.cs”.

Primeiros passos no .NET

- Abra o “console” do Windows.
- Digite “cmd” no executar (Windows+R) e dê enter.
- Digite “desktop” na linha de comando do console e dê enter.

Primeiros passos no .NET

- O compilador do C# está localizado em:

":\Windows\Microsoft.NET\Framework64\v4.0.30319\csc.exe"

Primeiros passos no .NET

- Escreva o seguinte comando:

":\Windows\Microsoft.NET\Framework64\v4.0.30319\csc.exe hello.cs"

Primeiros passos no .NET

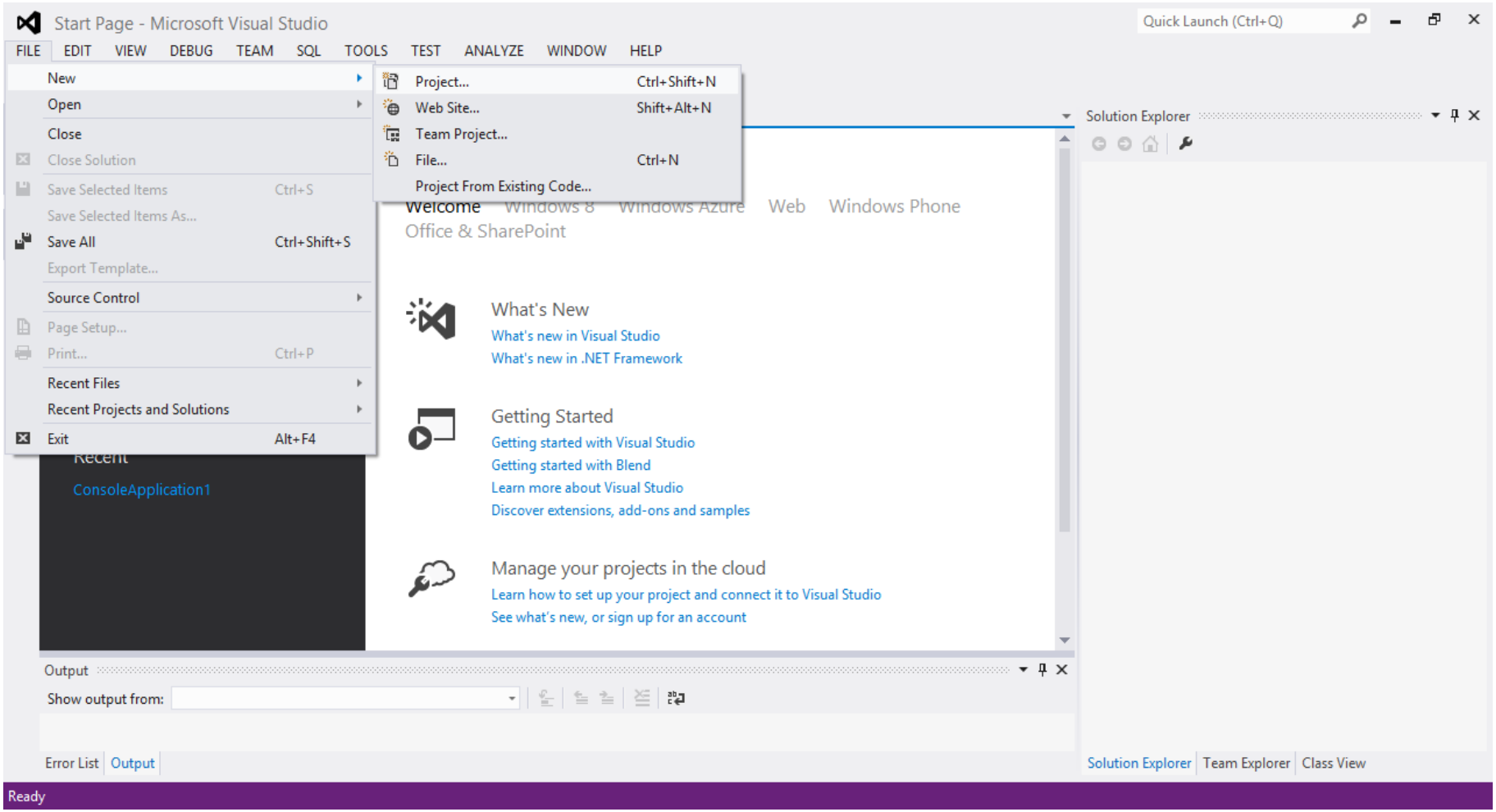
- Execute o seu programa compilado. Digite no prompt de comando:

“hello.exe”

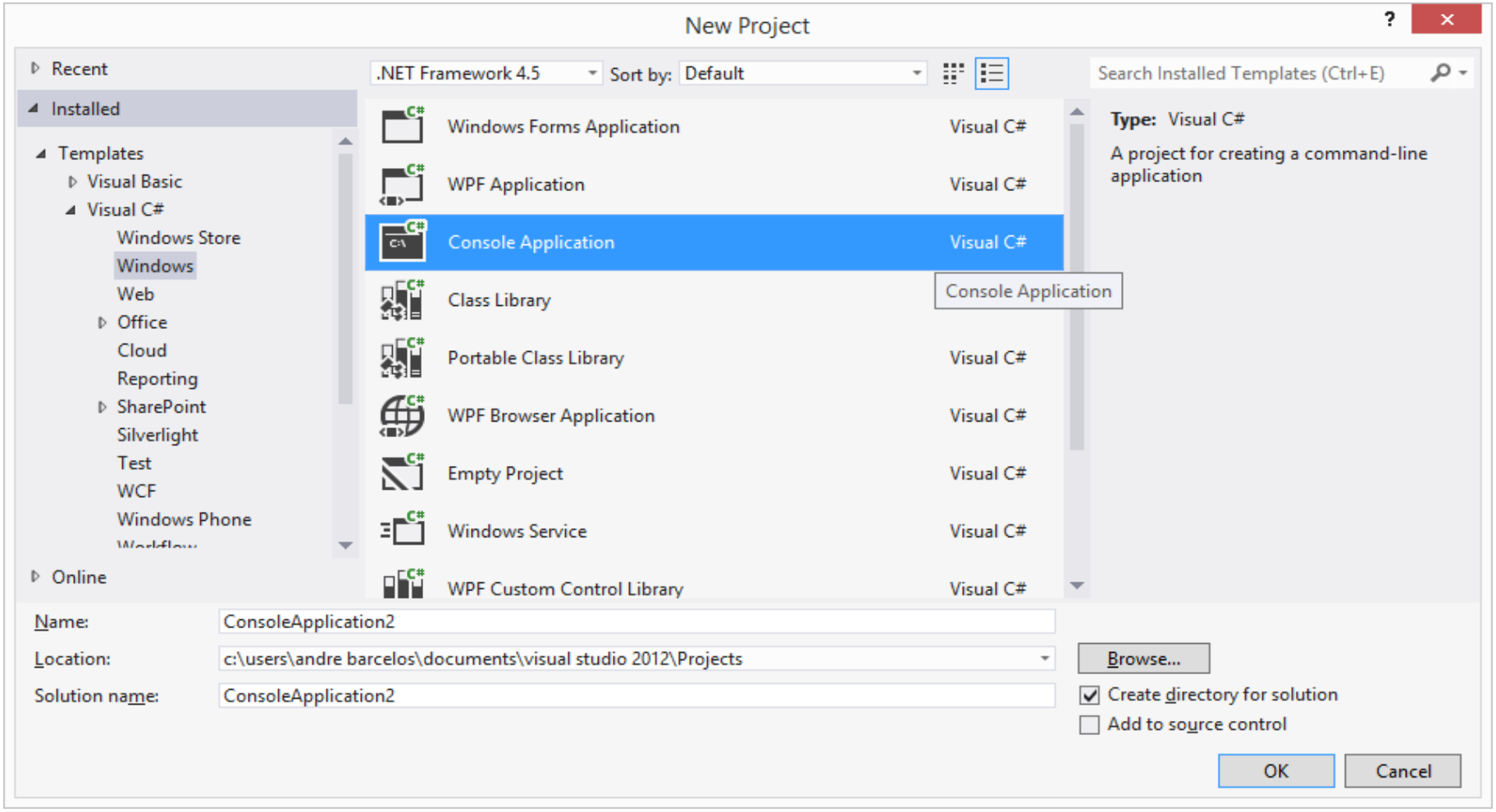
Primeiros passos no .NET

- Agora vamos fazer o mesmo programa, porém utilizando o Visual Studio.
- A versão utilizada neste curso será a 2012.

Primeiros passos (Hello World!)



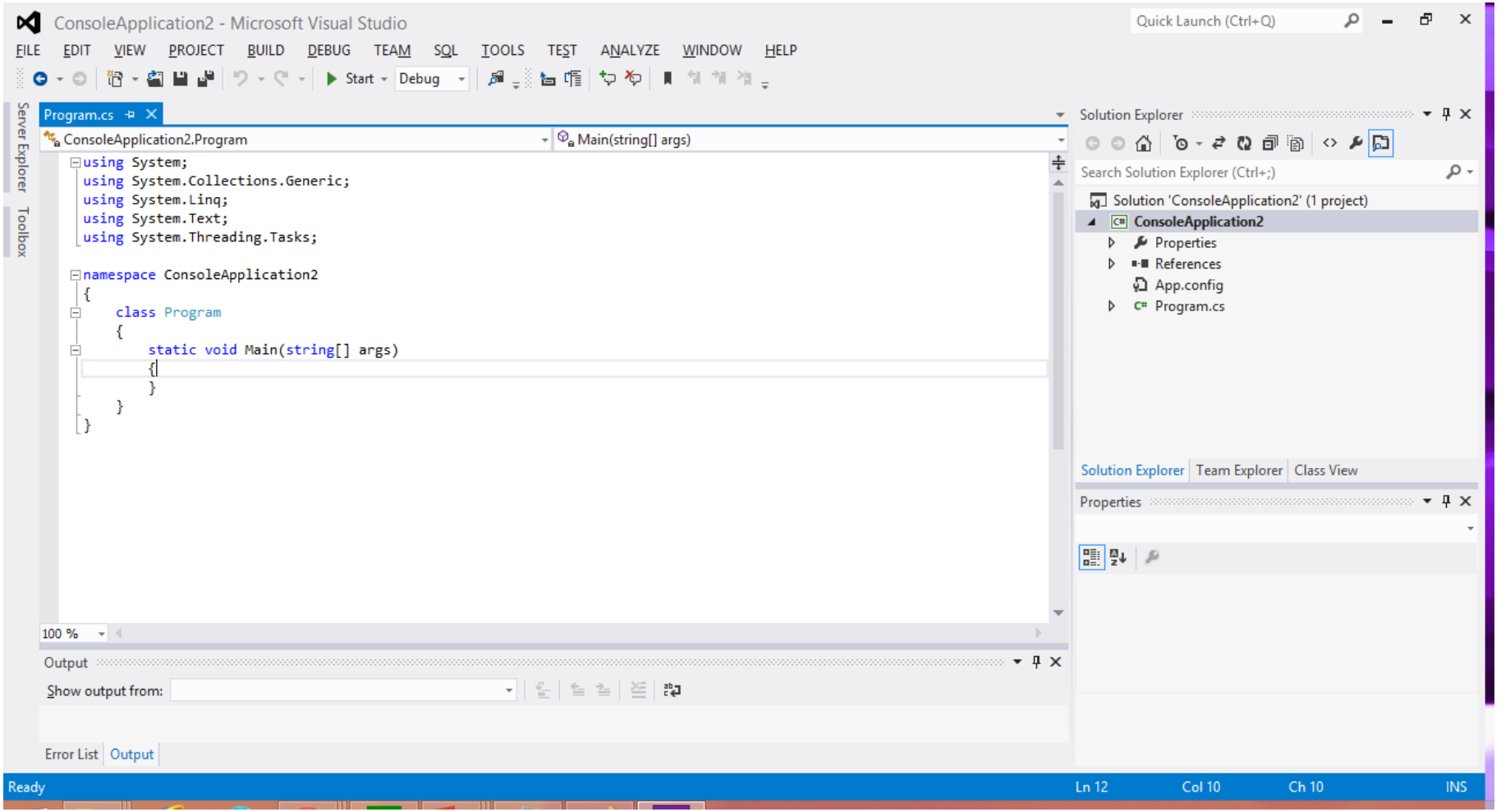
Primeiros passos (Hello World!)



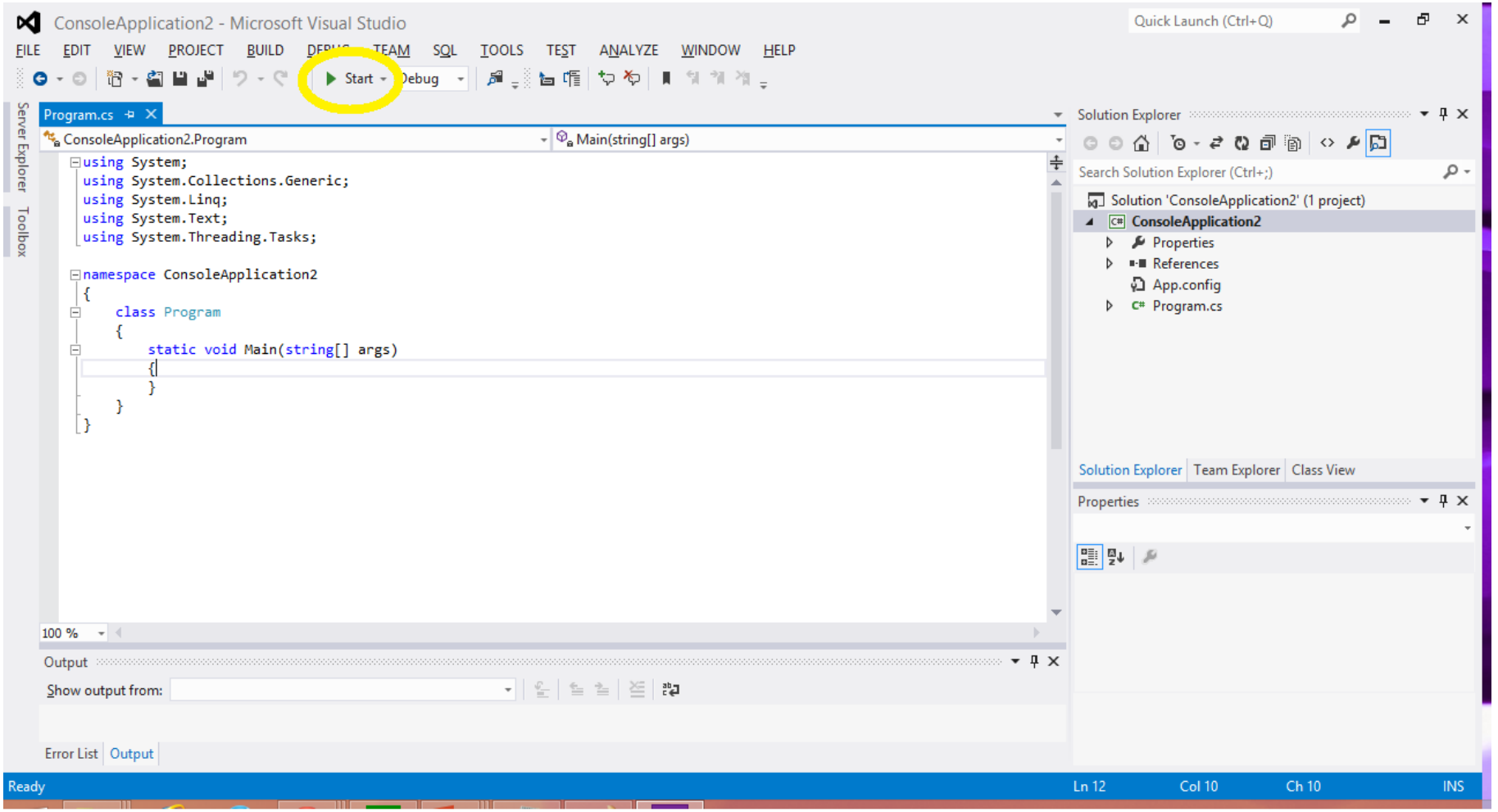
Primeiros passos (Hello World!)

- Agora vamos fazer o mesmo programa, porém utilizando o Visual Studio.
- A versão utilizada neste curso será a 2012.

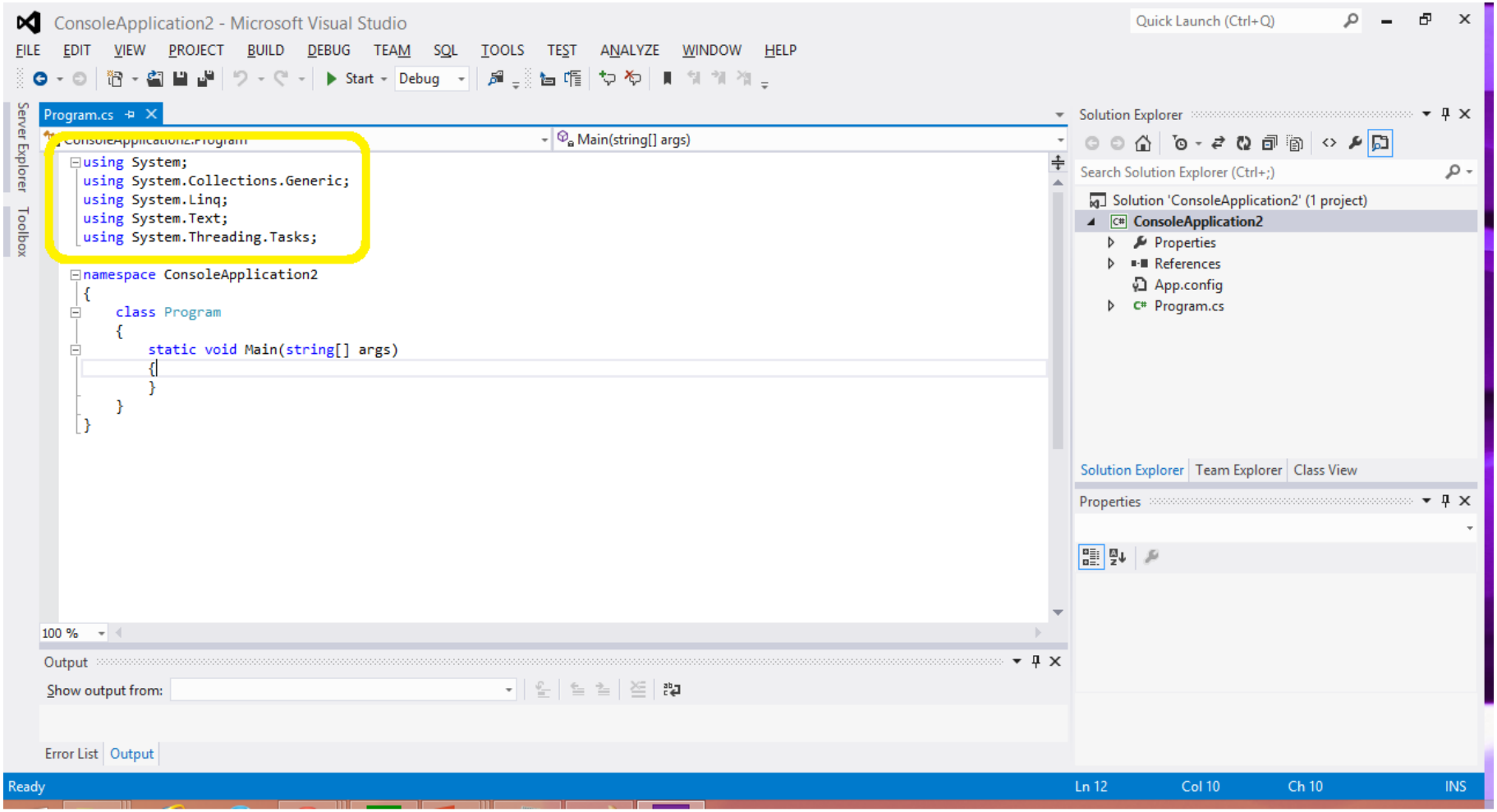
Primeiros passos (Hello World!)



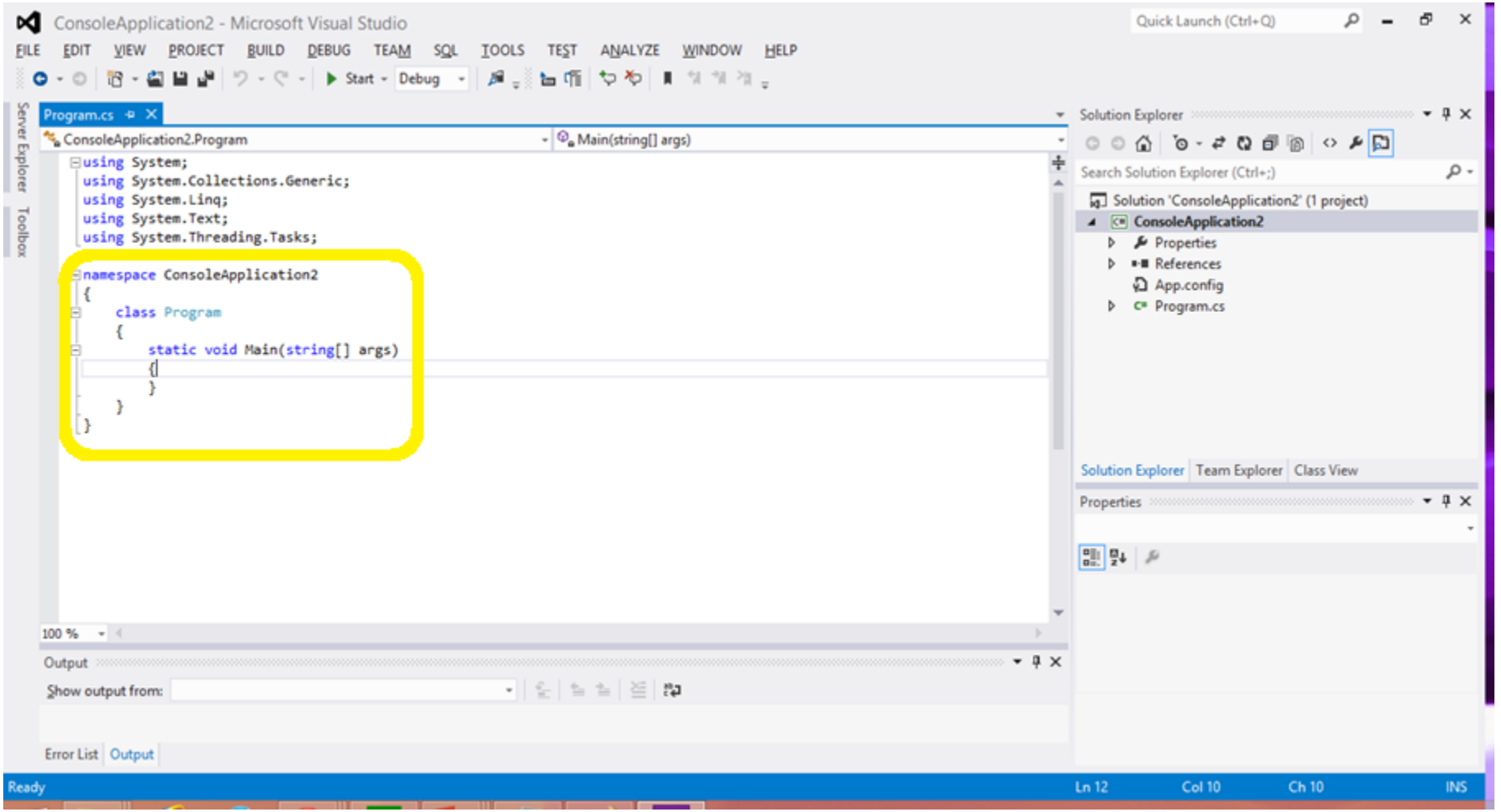
Primeiros passos (Hello World!)



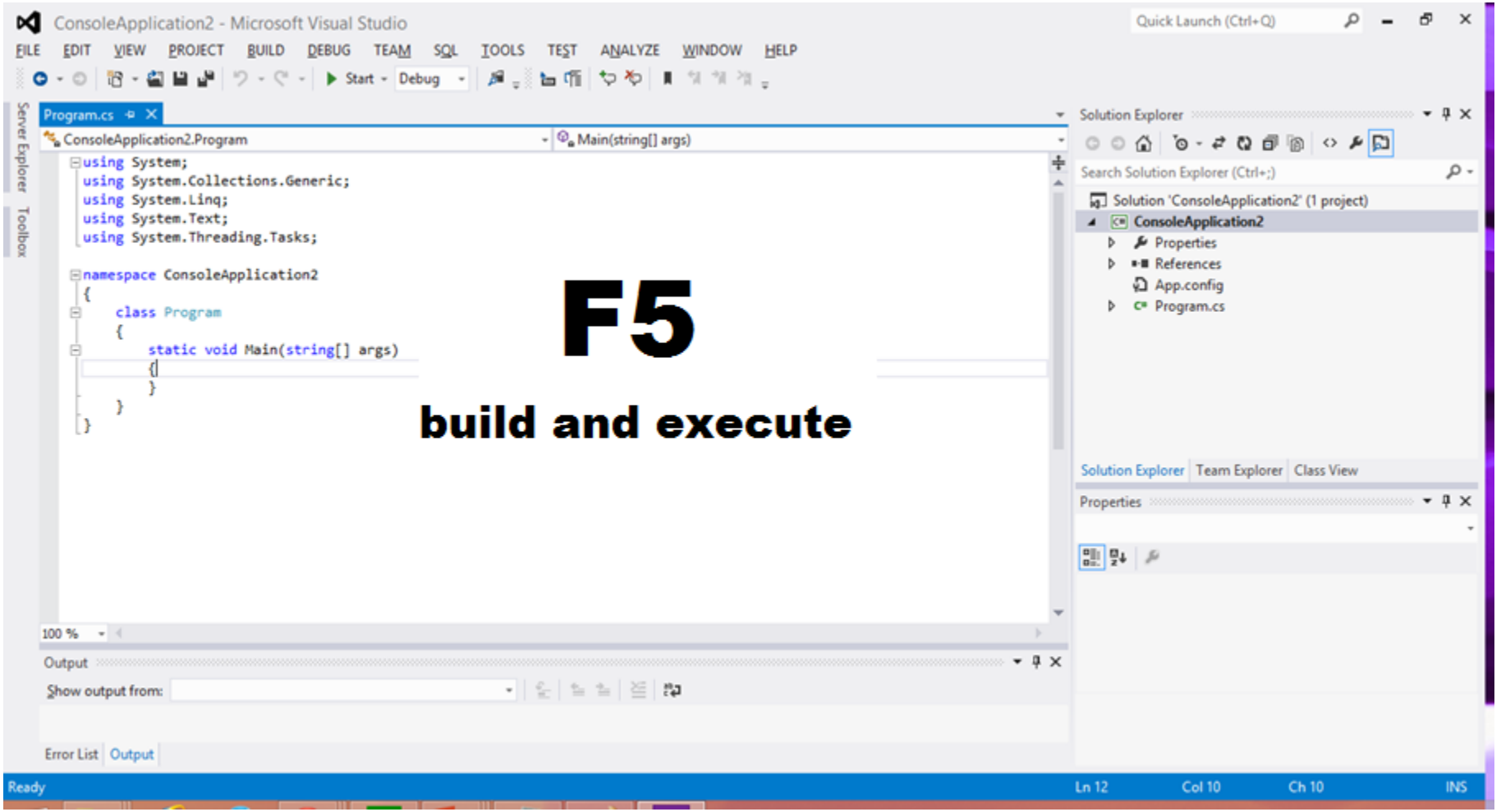
Primeiros passos (Hello World!)



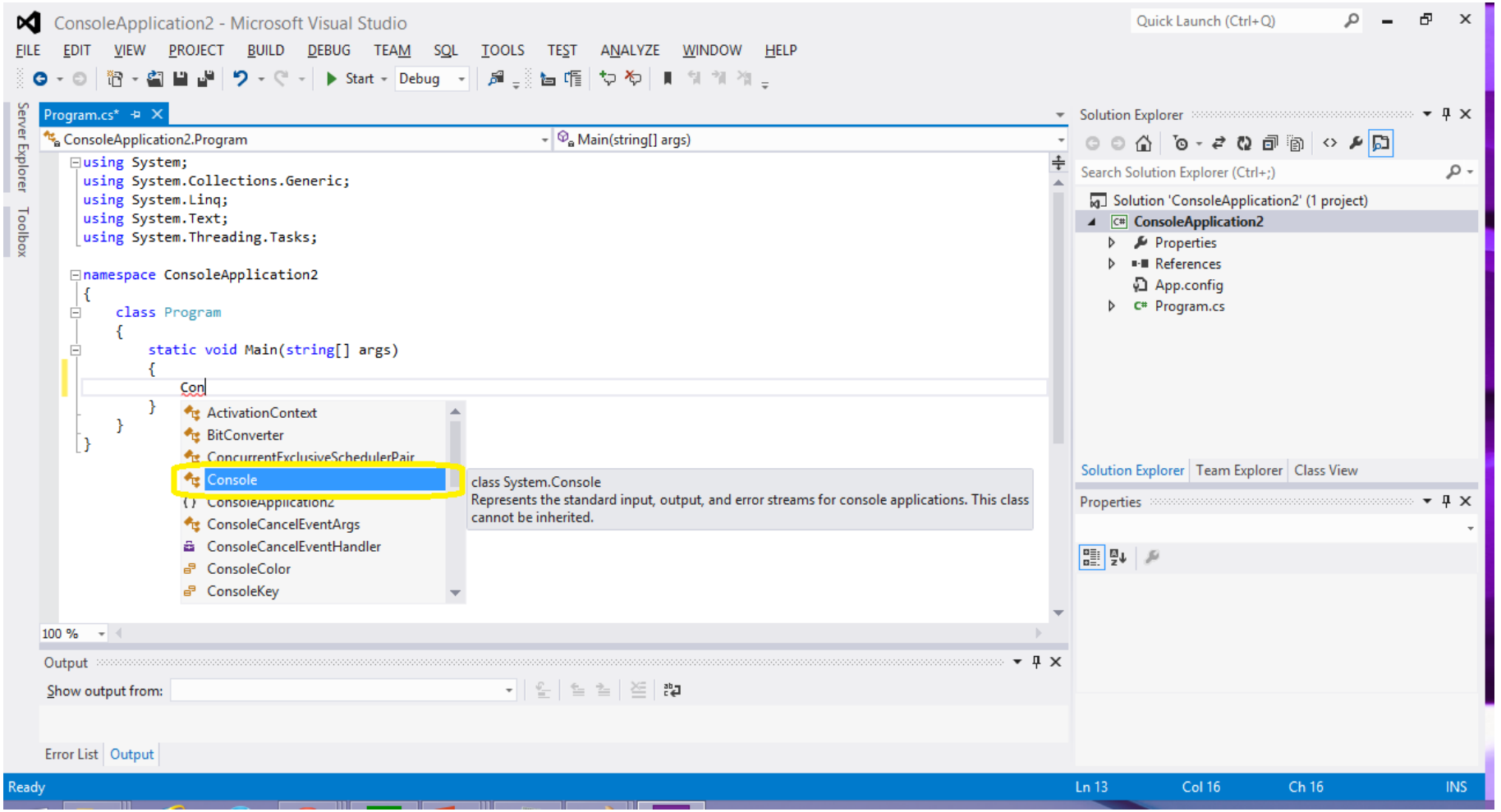
Primeiros passos (Hello World!)



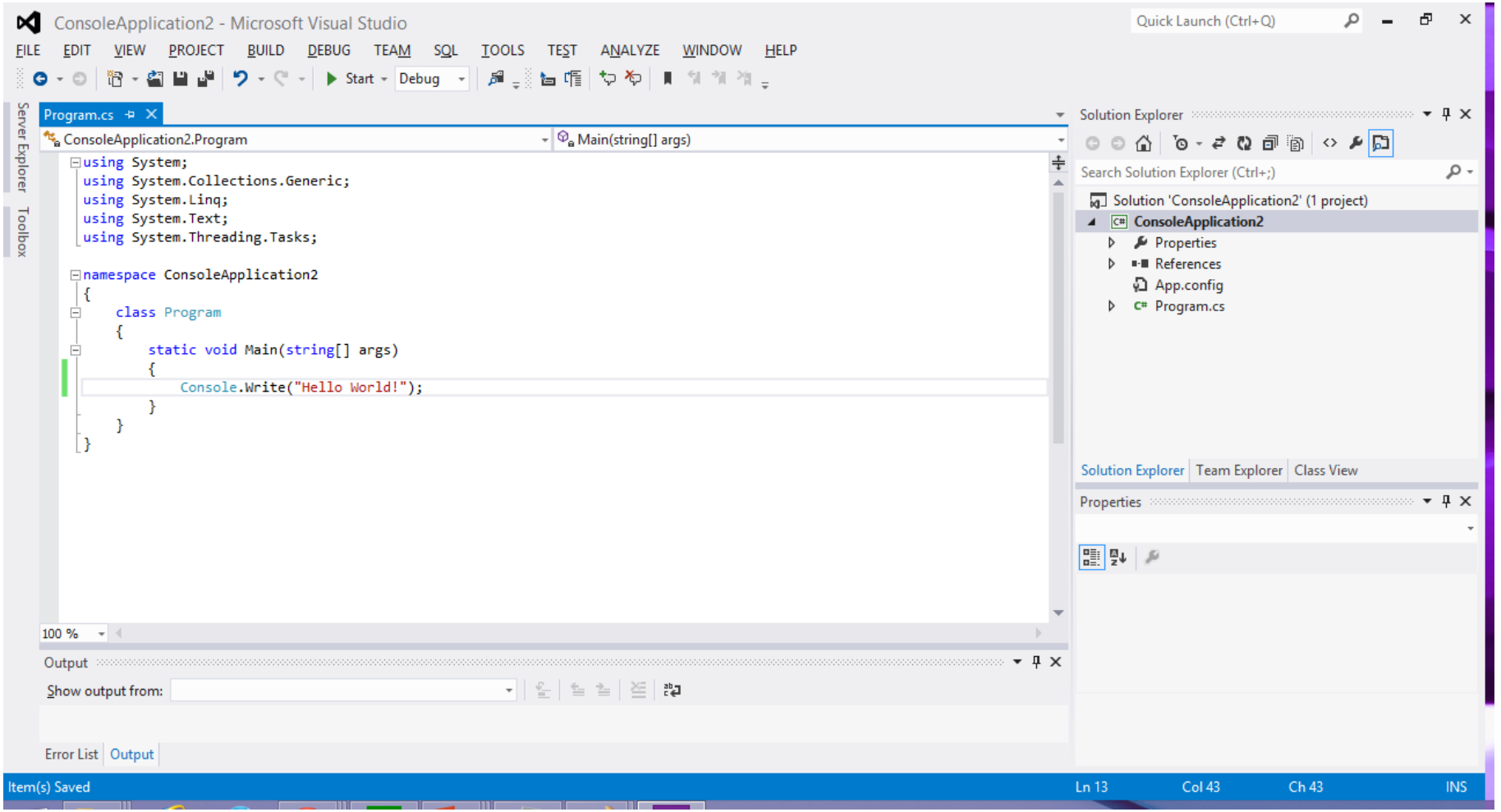
Primeiros passos (Hello World!)



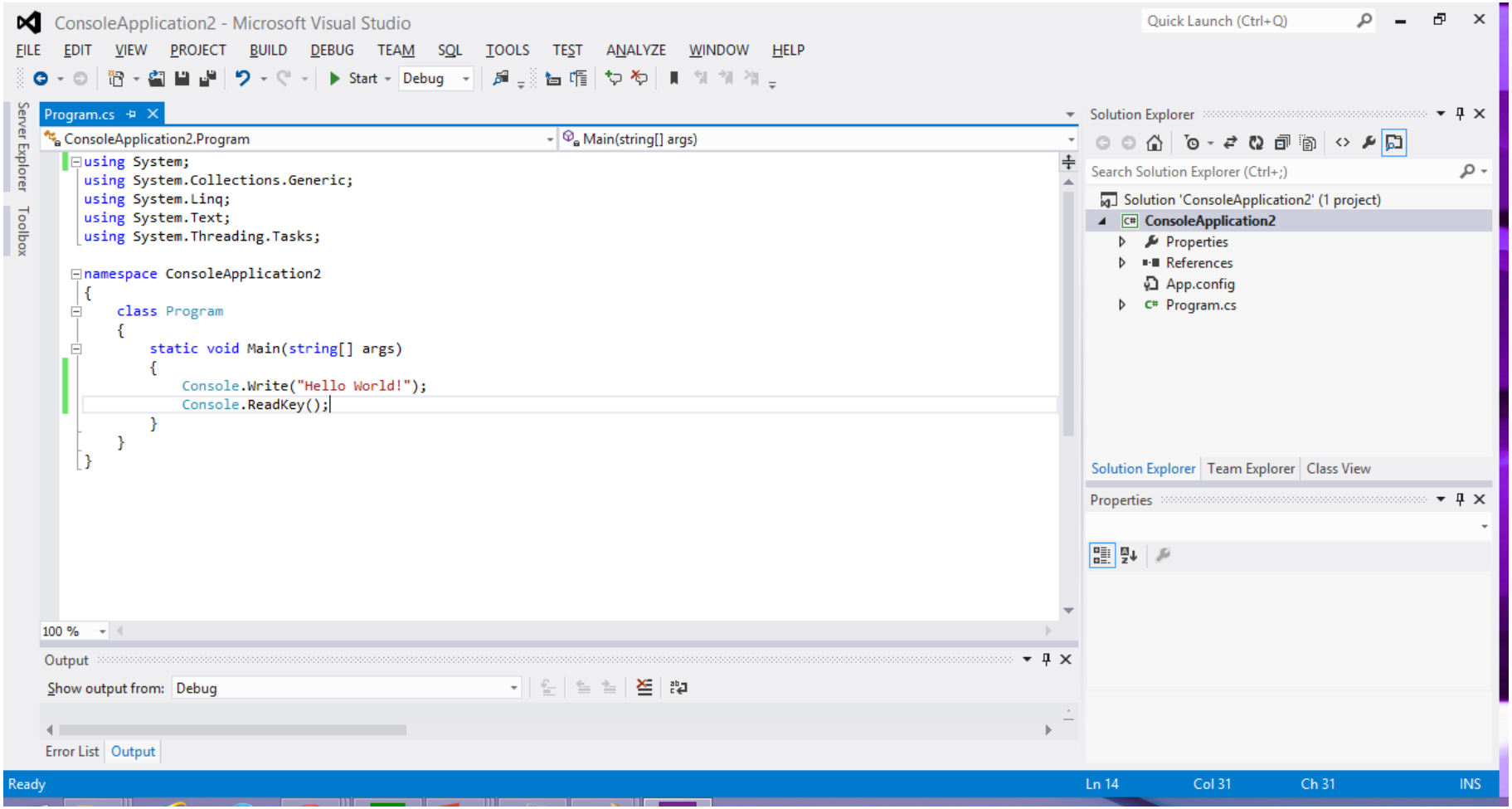
Primeiros passos (Hello World!)



Primeiros passos (Hello World!)



Primeiros passos (Hello World!)



Resumo de aula

- Nesta aula vimos:
  - A evolução do .NET
  - O arranjo estrutural do .NET
    - Common Language Infrastructure (CLI)
    - Metadata
    - Common Language Specification (CLS)
    - Common Type System (CTS)
    - Virtual Execution System (VES)
    - Assembly
    - Implementação de Segurança (Trabalho)
    - Gerenciamento de memória (Trabalho)
  - Common Language Runtime (CLR)
    - Ciclo de vida da aplicação .NET
  - Introdução ao C#
  - Compilação em linha de comando
  - Introdução ao Visual Studio 2012

Resumo de aula

- Mandem as anotações de aula para a tarefa no moodle.
- A tarefa “Trabalho – 01” é correspondente ao trabalho sobre Segurança e Gerenciamento de memória na CLI do .NET