

2015



C#/ .NET Programming Basics

The logo features a stylized profile of a human head in grey, with a yellow brain inside. The brain is depicted with white lines representing neural connections.

C#/.NET PROGRAMMING BASICS

Module 1. Basic principles of c#, clr

Training program

- **Block 1.** C# programming fundamentals
- **Block 2.** Windows application development
- **Block 3.** Service-oriented and web application development
- **Block 4.** Application architecture and design patterns
- **Block 5.** Certification

Block 1 contents

1. Basic principles of C#, CLR
2. Object oriented fundamentals
3. Exception handling
4. Advanced programming (Delegates, events, lambdas. Generics. Collections)
5. Assembly management and application debug
6. Multithreading and asynchronous processing
7. Data access
8. Unsafe code and pointers. .Net Framework security

C# & CLR basics

- Basic principles of C#, CLR
 - C# & CLR basics
 - Data types
 - Operators
 - Array, Structure, Enum
 - System.Console

Lecture contents

- C# & CLR basics
 - C# history
 - .Net Framework
 - CLR functionality and benefits
 - C# elements overview
 - Visual Studio IDE
 - Visual Studio templates
 - First simple program
 - Fixing compilation errors
 - Second simple program
 - Debugging
 - Solution vs Project

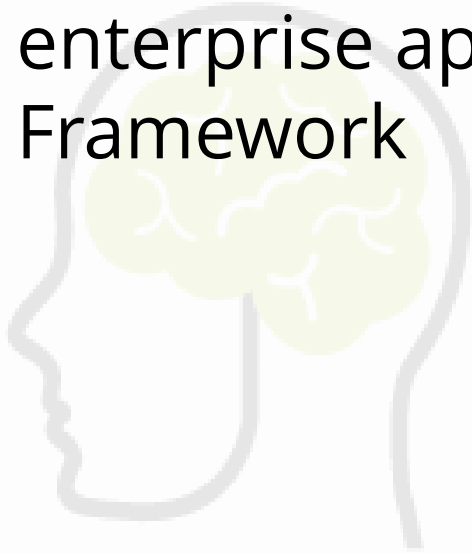
C# history

- C# & CLR basics
 - C# history
 - .Net Framework
 - CLR functionality and benefits
 - C# elements overview
 - Visual Studio IDE
 - Visual Studio templates
 - First simple program
 - Fixing compilation errors
 - Second simple program
 - Debugging
 - Solution vs Project

C# history (1/4)



C# - a programming language designed for building enterprise applications that run on the .NET Framework



BRAIN
ACADEMY

C# history (2/4)

- C# - evolution of C and C++
- Type safe
- Object oriented
- Developed by Anders Hejlsberg
- In 1999-2001

C# history (3/4)

- 2002 - C# 1.0
- 2003 - C# 1.2
- 2005 - C# 2.0: generics, partial types, anonymous methods, nullable types, delegates, static classes
- 2007 - C# 3.0: anonymous types, auto-implemented properties, extension methods, query expressions (LINQ), lambda expressions


C# history (4/4)

- 2010 – C# 4.0: dynamic binding, named and optional arguments, generic co/ contrvariance, embedded iterop types
- 2013 – C# 5.0: asynchronous methods, caller info attributes
- Announced 2015 – C# 6.0: compiler-as-a-service (Roslyn), exception filters, await in catch/ finally blocks, expression bodied members, null propagator (succinct null checking), string interpolation, nameof operator, parameterless struct constructors

.Net Framework

- C# & CLR basics
 - C# history
 - .Net Framework
 - CLR functionality and benefits
 - C# elements overview
 - Visual Studio IDE
 - Visual Studio templates
 - First simple program
 - Fixing compilation errors
 - Second simple program
 - Debugging
 - Solution vs Project

.NET Framework (1/3)

 **.NET Framework** – an integral Windows component that supports building, deploying, and running the next generation of applications and XML Web services.

It provides a highly productive, standards-based, multilanguage environment for integrating existing investments with next generation applications and services, as well as the agility to solve the challenges of deployment and operation of Internet-scale applications.

.NET Framework (2/3)

- The .NET Framework consists of three main parts:
 - the common language runtime (CLR)
 - a hierarchical set of unified class libraries
 - a componentized version of ASP called ASP.NET



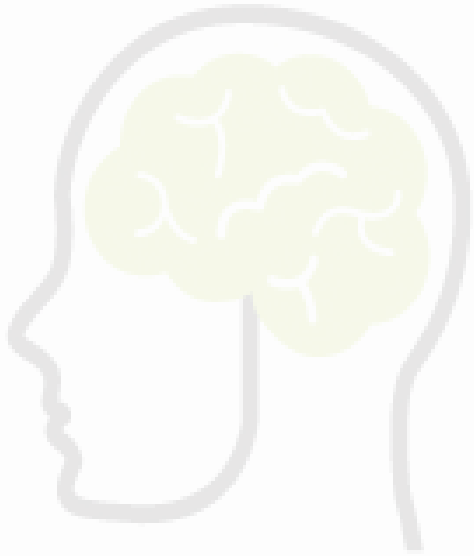
BRAIN
ACADEMY

.NET Framework (3/3)

- 2000 - .Net Framework
- 2012 - .Net Framework 4.5:
 - .Net compiler platform (Roslyn)
 - Microsoft .Net Native
 - New APIs for ASP.NET apps
 - Resizing in Windows Forms control
 - New workflow feature
 - Profiling improvements
 - Debugging improvements
 - Event tracing changes

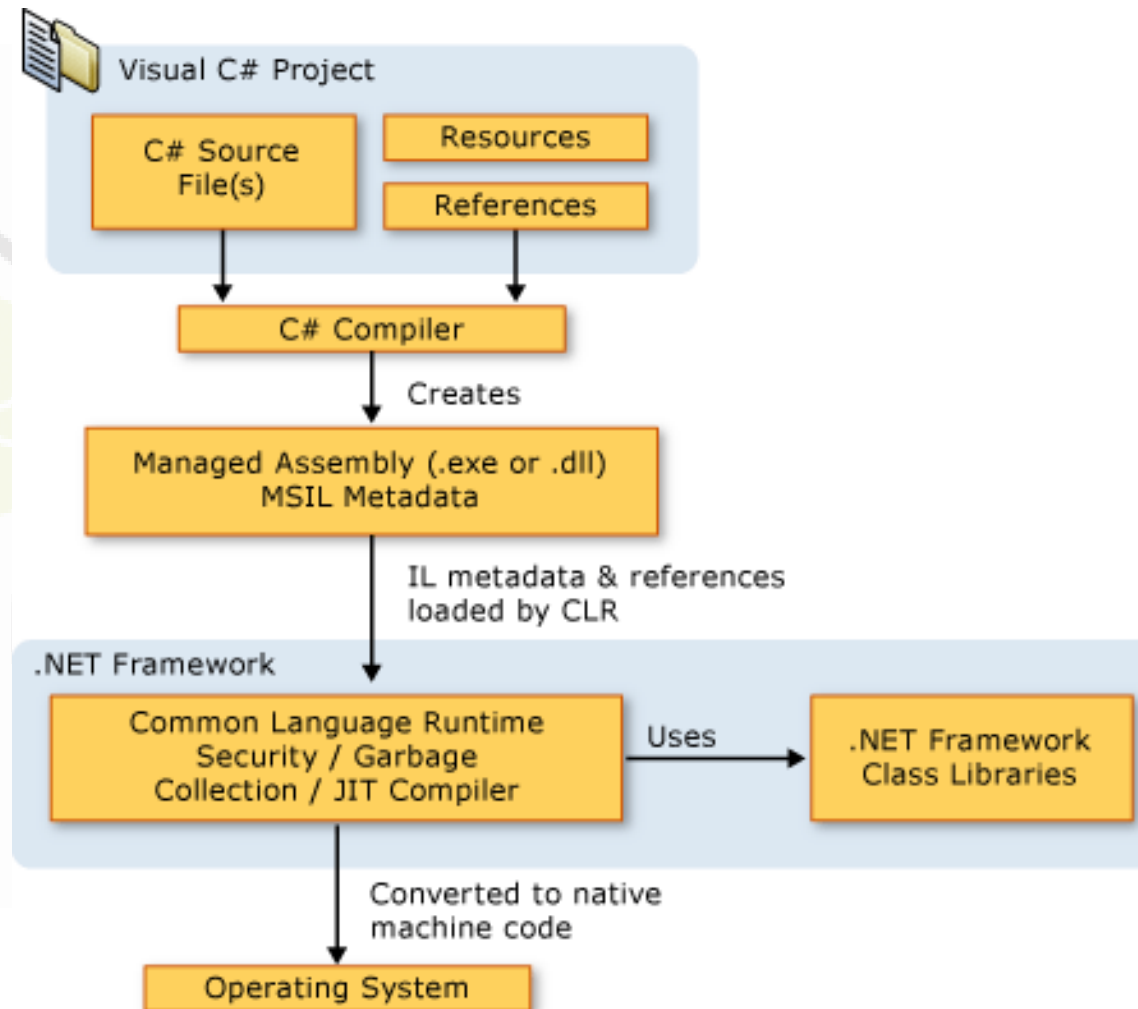
C# & .NET Framework

- C# applications run on .Net Framework



BRAIN
ACADEMY

.NET Framework platform architecture




• <https://msdn.microsoft.com/en-us/library/z1zx9t92.aspx>

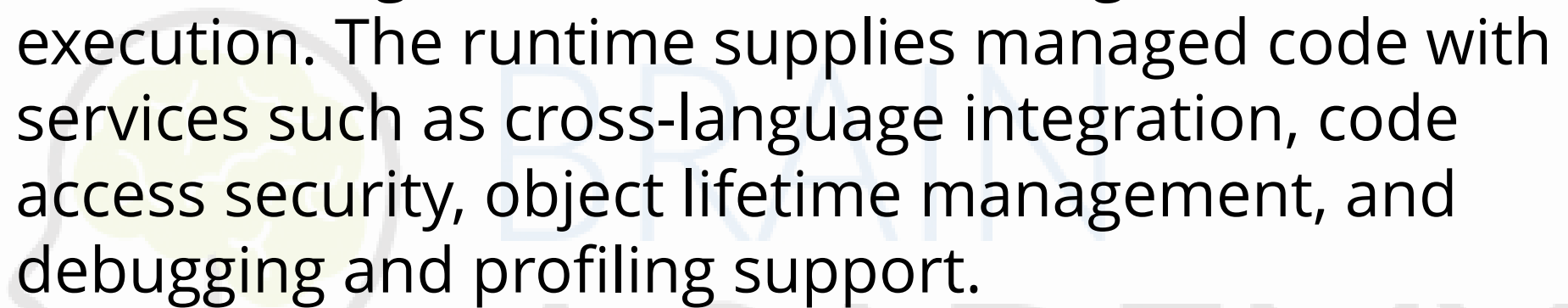
CLR functionality and benefits

- C# & CLR basics
 - C# history
 - .Net Framework
 - CLR functionality and benefits
 - C# elements overview
 - Visual Studio IDE
 - Visual Studio templates
 - First simple program
 - Fixing compilation errors
 - Second simple program
 - Debugging
 - Solution vs Project

CLR functionality (1/2)

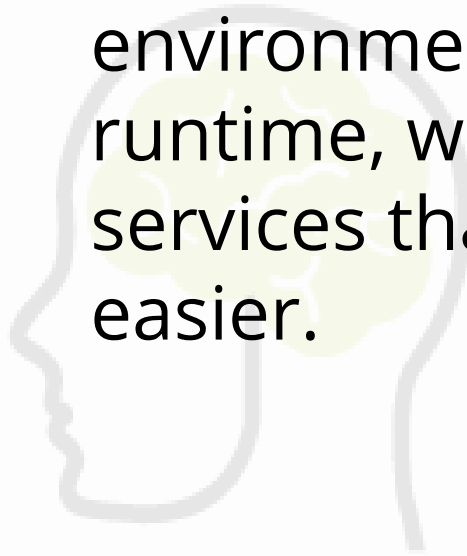


CLR - the engine at the core of managed code execution. The runtime supplies managed code with services such as cross-language integration, code access security, object lifetime management, and debugging and profiling support.



CLR functionality (2/2)

- The .NET Framework provides a run-time environment called the common language runtime, which runs the code and provides services that make the development process easier.



BRAIN
ACADEMY




CLR benefits

- Performance improvements.
- The ability to easily use components developed in other languages.
- Extensible types provided by a class library.
- Language features such as inheritance, interfaces, and overloading for object-oriented programming.
- Support for explicit free threading that allows creation of multithreaded, scalable applications.
- Support for structured exception handling.
- Support for custom attributes.
- Garbage collection.
- Use of delegates instead of function pointers for increased type safety and security

C# elements overview

- C# & CLR basics
 - C# history
 - .Net Framework
 - CLR functionality and benefits
 - **C# elements overview**
 - Visual Studio IDE
 - Visual Studio templates
 - First simple program
 - Fixing compilation errors
 - Second simple program
 - Debugging
 - Solution vs Project

C# elements overview 1/2

-  **C#** - programming language. It has syntax and semantics.
-  **Syntax** - the rules governing the formation of a command-line statement, including the order in which a command must be typed, and the elements that follow the command.
-  **Semantics** – expressions in the code

C# elements overview 2/2

- Every program based on statements, expressions and operators.
- The actions that a program takes are expressed in statements.
- A statement can consist of a single line of code that ends in a semicolon, or a series of single-line statements in a block. A statement block is enclosed in {} brackets and can contain nested blocks.

Simple C# statements

- Declaration statements
- Expression statements
- Selection statements
- Iteration statements
- Jump statements
- Exception handling statements

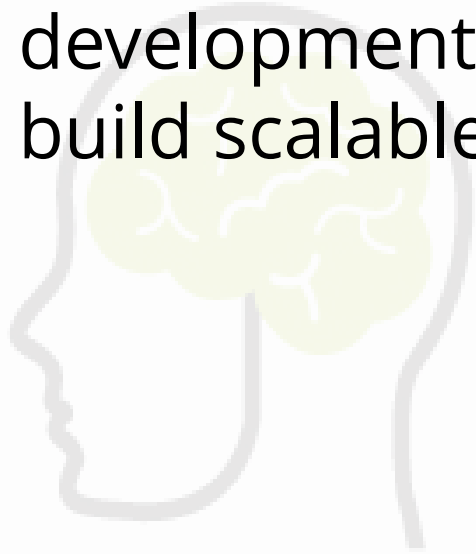
Visual Studio IDE

- C# & CLR basics
 - C# history
 - .Net Framework
 - CLR functionality and benefits
 - C# elements overview
 - Visual Studio IDE
 - Visual Studio templates
 - First simple program
 - Fixing compilation errors
 - Second simple program
 - Debugging
 - Solution vs Project

Visual Studio IDE



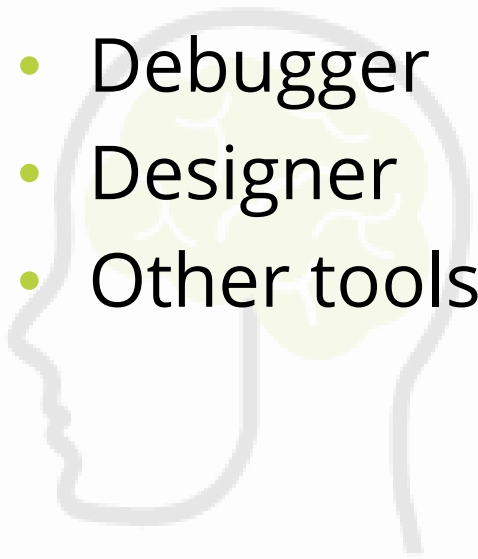
Microsoft Visual Studio - an integrated development environment for helping developers to build scalable applications and Web services.



BRAIN
ACADEMY

Visual Studio IDE features

- Code editor
- Debugger
- Designer
- Other tools



BRAIN
ACADEMY

Visual Studio IDE editions

- Express
- Professional
- Community
- Premium
- Ultimate

BRAIN
ACADEMY

Visual Studio templates

- C# & CLR basics
 - C# history
 - .Net Framework
 - CLR functionality and benefits
 - C# elements overview
 - Visual Studio IDE
 - Visual Studio templates
 - First simple program
 - Fixing compilation errors
 - Second simple program
 - Debugging
 - Solution vs Project

Visual Studio templates

- Windows
- Web
- Office/ SharePoint
- Windows Store
- Cloud
- LightSwitch
- Reporting
- Silverlight
- Test
- WCF
- Windows Phone
- Workflow

First simple program

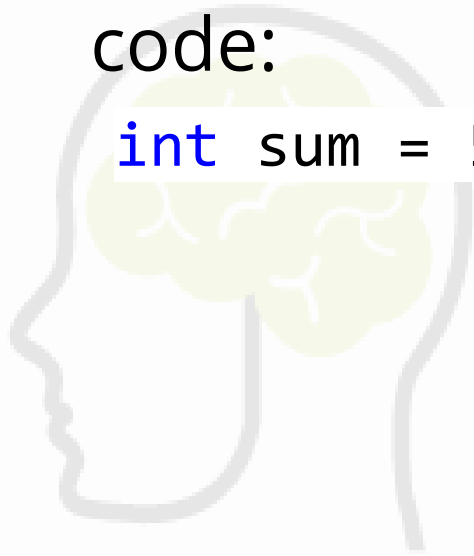
- C# & CLR basics
 - C# history
 - .Net Framework
 - CLR functionality and benefits
 - C# elements overview
 - Visual Studio IDE
 - Visual Studio templates
 - **First simple program**
 - Fixing compilation errors
 - Second simple program
 - Debugging
 - Solution vs Project



First simple program

- Put in the “Main” block between brackets {}, next code:

```
int sum = 5 + 6;
```



BRAIN
ACADEMY

Fixing compilation errors

- C# & CLR basics
 - C# history
 - .Net Framework
 - CLR functionality and benefits
 - C# elements overview
 - Visual Studio IDE
 - Visual Studio templates
 - First simple program
 - **Fixing compilation errors**
 - Second simple program
 - Debugging
 - Solution vs Project

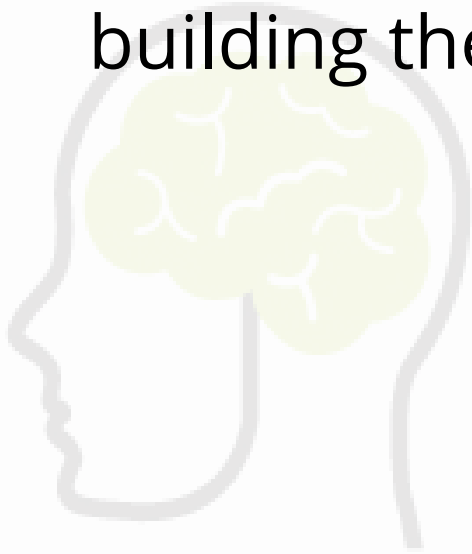
Fixing compilation errors

- When you compile the project, an error message indicates what the problem is and where it occurred.
- There are 2 error message windows:
 - **Output** window
 - **Error List** window



Fixing compilation errors using the IDE (1/2)

- A message in the **Output** window indicates that building the project failed.



BRAIN
ACADEMY



Fixing compilation errors using the IDE (2/2)

- A message in the **Error List** window shows list of errors with description, line and column.



BRAIN
ACADEMY

Second simple program

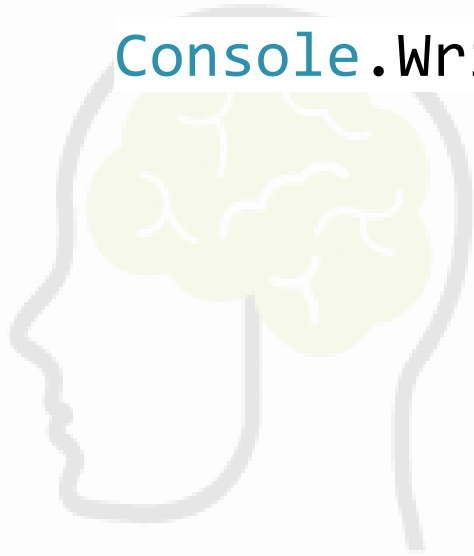
- C# & CLR basics
 - C# history
 - .Net Framework
 - CLR functionality and benefits
 - C# elements overview
 - Visual Studio IDE
 - Visual Studio templates
 - First simple program
 - Fixing compilation errors
 - **Second simple program**
 - Debugging
 - Solution vs Project



Second simple program

- Add code

```
Console.WriteLine(sum);
```



BRAIN
ACADEMY

Debugging

- C# & CLR basics
 - C# history
 - .Net Framework
 - CLR functionality and benefits
 - C# elements overview
 - Visual Studio IDE
 - Visual Studio templates
 - First simple program
 - Fixing compilation errors
 - Second simple program
 - **Debugging**
 - Solution vs Project



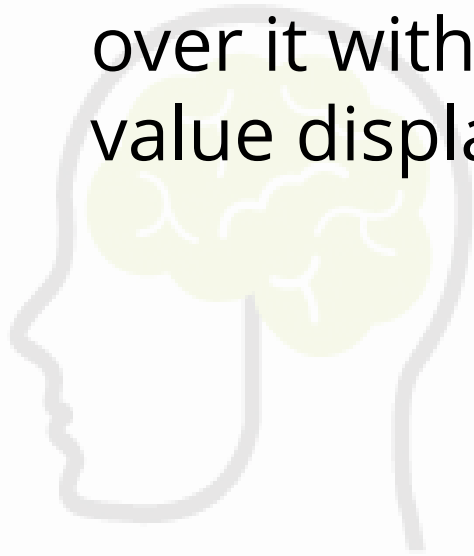
Debugging (1/2)

- Running a program in Debug mode enables you to use breakpoints to pause the program to examine the state of variables and objects.
- A red circle appears to the left of a line with a breakpoint set.
- When the program reaches the line with the breakpoint, execution stops temporarily. A yellow arrow to the left of a line of code indicates that is the next line to be executed.



Debugging (2/2)

- To examine the value of the sum variable, hover over it with mouse. The variable name and its value displayed in a tooltip window.



BRAIN
ACADEMY

Solution vs project

- C# & CLR basics
 - C# history
 - .Net Framework
 - CLR functionality and benefits
 - C# elements overview
 - Visual Studio IDE
 - Visual Studio templates
 - First simple program
 - Fixing compilation errors
 - Second simple program
 - Debugging
 - Solution vs Project

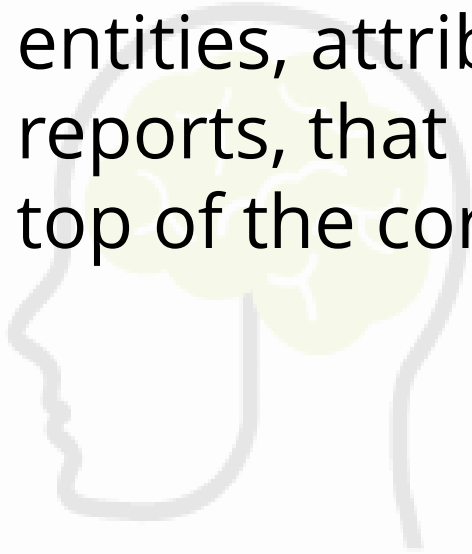
Solution vs project

- Visual Studio provides two containers to help to manage the items that are required by development effort.
- These containers are called **solutions** and **projects**.
- **Solution Explorer** is used to view and manage projects and solutions and their associated items.

Solution (1/3)



Solution – a set of components, for example, entities, attributes, relationships, workflows, and reports, that provide a specific set of functionality on top of the core CRM platform.



BRAIN
ACADEMY

Solution (2/3)

- Solutions contain items that you need in order to create your application.
- Solution includes one or more projects, files and metadata that help define the solution as a whole.
- Visual Studio automatically generates a solution when you create a new project. Visual Studio stores the definition for a solution in two files: .sln and .suo.

Solution (3/3)

- The solution definition file (.sln) stores the metadata that defines your solution, including:
 - The projects that are associated with the solution.
 - The items that are not associated with a particular project.
 - The build configurations that determine which project configurations to apply in each type of build.
- The metadata stored in the .suo file as you construct a solution and set its properties is used to customize the IDE whenever the solution is active.

Project (1/2)



A set of source files and related metadata, such as component references and build instructions, that helps you organize and perform common tasks on the items that you are developing. Projects are contained within a solution.

BRAIN
ACADEMY

Project (2/2)

- Projects are used in a solution to logically manage, build, and debug the items that make up your application. The output of a project is usually an executable program (.exe), a dynamic-link library (.dll) file or a module, among others.
- Visual Studio provides several pre-defined project templates.