What is .NET? The Microsoft Programming Framework

Daniel S. Fowler, dan@tekeye.uk

First Published: 2012-11-23
Latest Updated: 2023-06-12
Online Version: https://tekeye.uk/computing/what-is-dot-net

Introduction

If you are new to writing software for Windows personal computers (PCs) it is not long before you come across Microsoft's *.NET* (pronounced *dot net*). *What is .NET?* The simplest answer is to say that .NET is used to build and run programs on a computer. Microsoft's .NET is a widely used computer programming framework for Windows and other systems. This article provides a brief explanation, as an introduction, to answer the question, what is .NET? It explains the meaning of a computer framework and the role of Microsoft's .NET app environment.

Windows Developers Can Use Microsoft .NET When Programming

Microsoft .NET is an important Windows technology to help with the development of computer programs. It is shipped for free with Microsoft Windows and is thus available on millions of computers around the world. .NET used to be internal to Microsoft, but it is Open Source with a big community. It enables software to be built to run on multiple devices using single sets of code files.

.NET Provides a Framework to Help Write Complex Programs

Whenever you see a building, such as an office block or skyscraper, being constructed you will notice that a steel frame is used to provide the basic structure. On to this frame are fixed walls, windows, floors, ceilings and everything else that makes up a building. The framework is made of steel beams, rivets and welds. Everything that is attached to it is made from common building materials and components. It is the architects and builders that determine the final look of the building using standard components.

Figure 1. Buildings look different with different purposes but are constructed from common components, similarly, programs have different functions but are built from common libraries (image cropped from a Wikimedia Commons image).

A similar thing occurs when computer programs are written, the software developer (a.k.a. computer programmer) decides the final functionality and look of a program but will use an existing set of components to help achieve the end requirement.

Microsoft .NET Libraries

Microsoft .NET is a computer framework that contains hundreds of useful components. The components are gathered together into libraries, to organize the components into categories. Libraries with components to draw on the screen, read input, talk to the Internet, etc. This allows the programmer to concentrate on producing the functionality of the software and not have to work on the code that controls the hardware. Modern software programs are written with a tool called an Integrated Development Environment (IDE). An IDE allows the programmer to write the code in an editor, run the code to test it and use a debugger to fix any errors found during testing. Many IDEs come with a set of common existing components to use, such as text boxes, buttons and image containers. Microsoft provides a *Visual Studio* IDE. Here's an IDE in use:

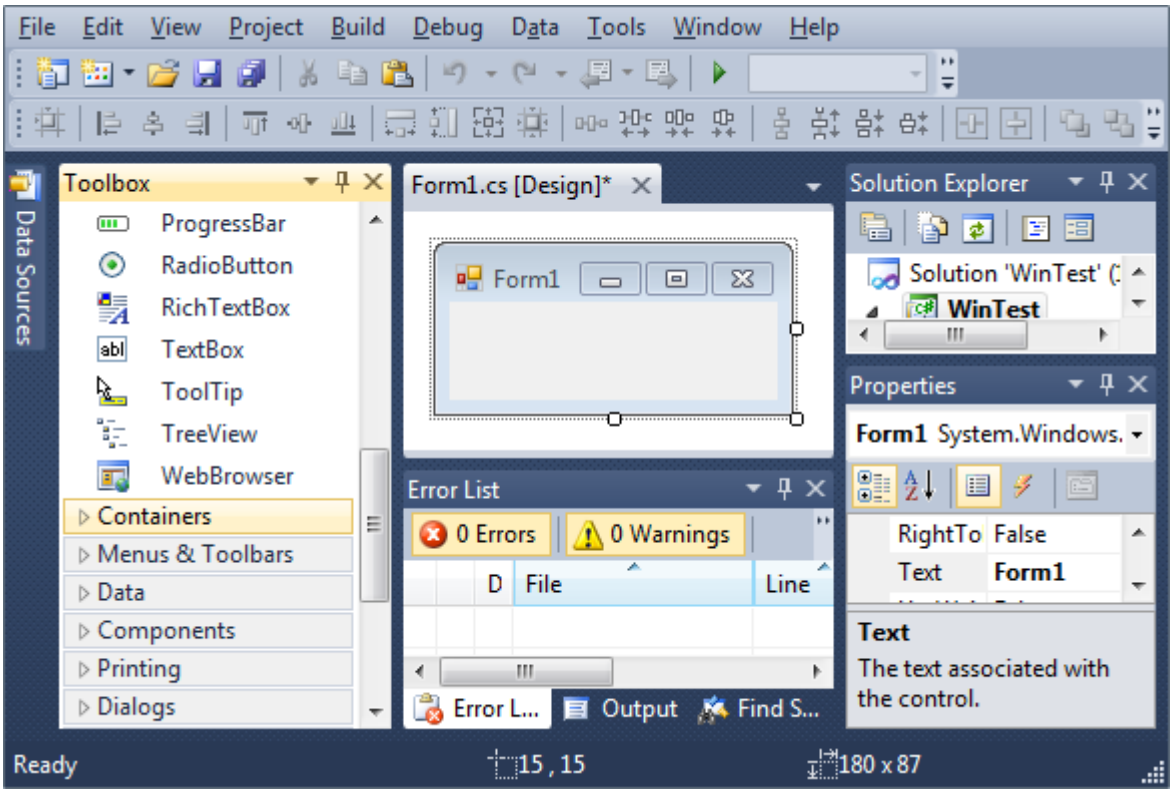


Figure 2. An Integrated Development Environment enables the editing, compiling, and debugging of apps.

With the .NET framework and an IDE, it only takes a few lines of code to get a simple program working. The speed of software development using .NET makes it a very useful framework for developers writing Windows programs.

Programmers and other companies can produce their own components and bundle them into their own libraries, many do. A programmer will often develop a library of routines that they reuse in new software projects. Very large and complex programs will be broken down into different sections, and those sections will usually be grouped into functional libraries to help with a program's design and maintenance. Not only is .NET a comprehensive set of libraries for all of the facilities that a program could need, but it is also an environment in which a program executes.

Microsoft .NET Runtime Engine

Most computer programs are written in a high-level logical computer language. Whereas the computer itself runs code that is stored as binary numbers. The process of converting the programmer's high-level code to the computer's binary code is called compiling. Microsoft .NET adds a layer between the high-level code and the binary code called the Common Language Runtime (CLR). The CLR allows programmers to pick from a variety of high-level languages with which to write software and allows the computers to execute that code efficiently and securely. It means that many types of computers with different types of microprocessors can all run the same .NET program efficiently.

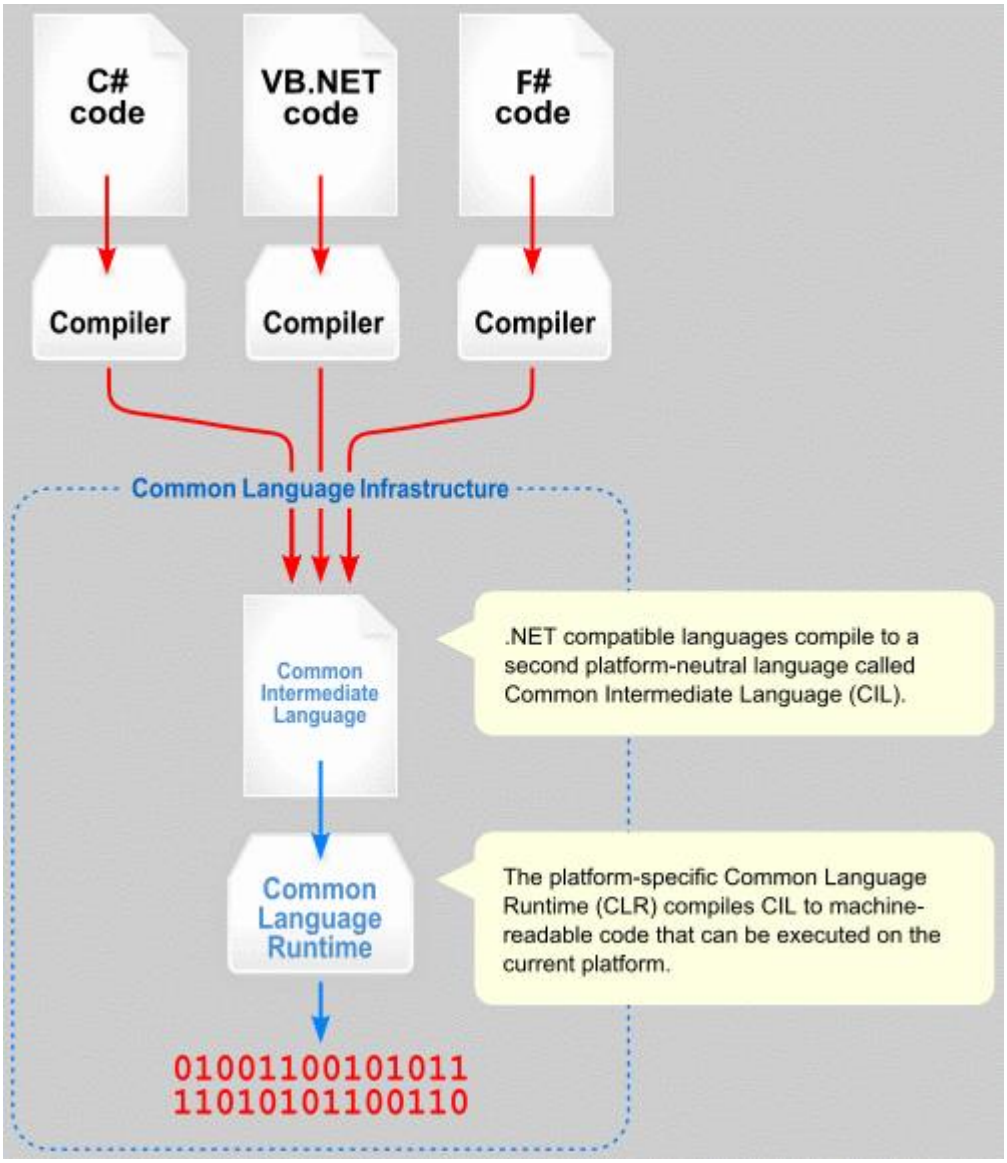
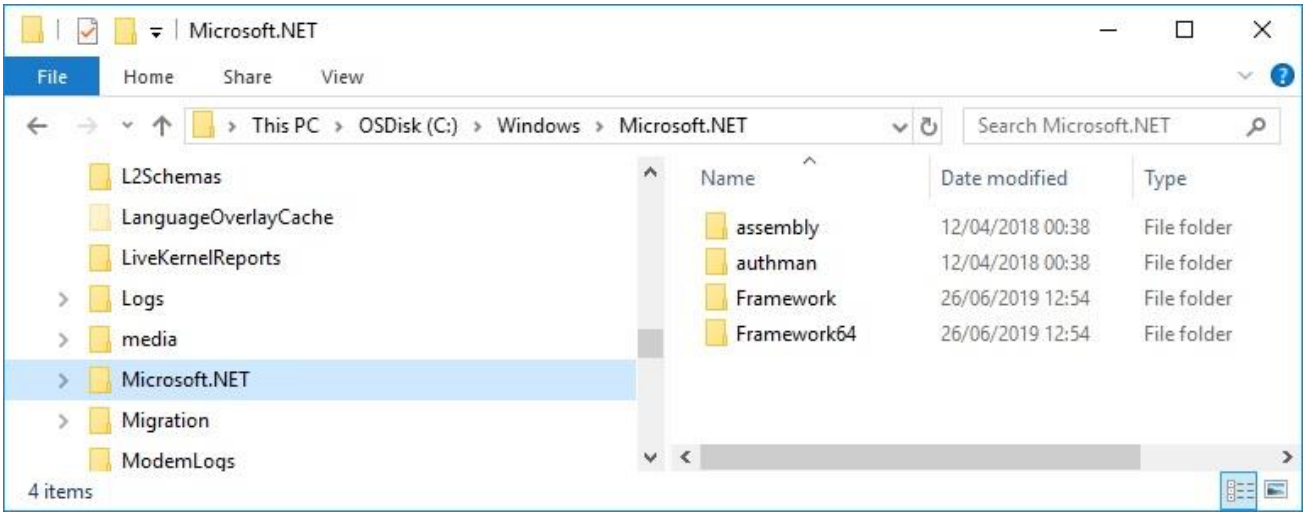


Figure 3. The structural elements within .NET (based on a Wikimedia Commons image).

Multiplatform NET

When Microsoft first produced .NET it was built for Windows PCs. Windows .NET is likely installed in the `C:\Windows\Microsoft.NET` directory:



There can be several versions of .NET stored in the subdirectories under *Framework64* (for 64-bit .NET) and *Framework* (for 32-bit .NET). Each version provides C# and Visual Basic compilers to enable Windows programs to be built. There is a build tool for more complex projects, called *MSBuild*, and other utilities related to .NET programming for PCs. This means that programs can be written for a PC on Windows just using a text editor, e.g. Windows Notepad. However, an IDE or a more powerful text editor, like *Visual Studio Code* does make it easier to write an app.

To enable .NET to support more types of computers, operating systems and servers, Microsoft first developed *.NET Core*. This is a version of .NET allowing .NET apps to be written and run on computers that do not use Windows. This is good for the programmer as the same code can be used across multiple types of devices and environments, much like C code. Microsoft now provides a unified .NET for multiple platforms, see the .NET download website. Furthermore, there is a community of developers providing a version of .NET for embedded computers called the *.NET nanoFramework*.

.NET can be installed by other software, usually by an app installer. Microsoft's IDE, Visual Studio, will install .NET, as can other IDEs and the *.NET Software Development Kit* (SDK). If the .NET SDK is on a Windows PC it can be found in the `C:\Program Files\dotnet` directory. Unlike the earlier versions of .NET, which accessed individual programs for various features, for example, *csc.exe* for the C# compiler, there is now the *dotnet* program to invoke a variety of .NET features for building .NET apps.

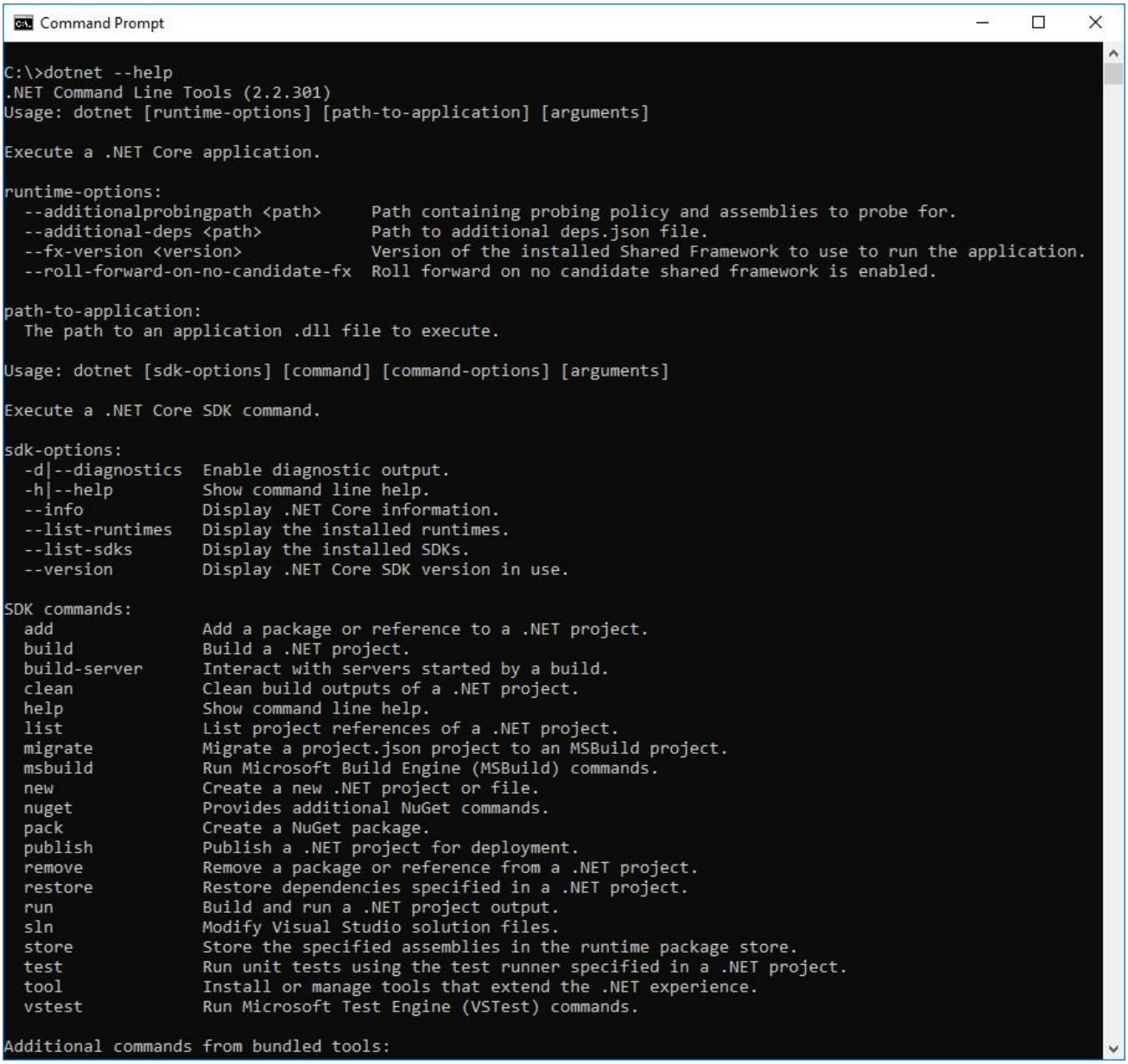


Figure 4. The dotnet command line executes in a terminal window.

Now you know that .NET is on Windows, grab an IDE (https://visualstudio.microsoft.com/) and start building .NET programs. Visual Studio requires the administrator machine account for installation and updates, otherwise try *Visual Studio Code* (https://code.visualstudio.com/), also installable from a Zip file.

Additional Information

This article provides a basic introduction to .NET, for a more in-depth overview see the Wikipedia entry. To learn more about the details of .NET head over to the Microsoft .NET website. See also the *Mono Project* (https://www.mono-project.com/).

For details on installing recent versions of .NET on Windows see the Microsoft Install .NET on Windows article (https://learn.microsoft.com/en-us/dotnet/core/install/windows). It includes information on the current supported versions of .NET and which versions of Windows they support.