What are the Differences between the .NET and the C++ version of Solid Framework?

Author: Roger Dunham

Date: 3rd September 2019

Version   1.0

Introduction

Solid Framework is available as a managed .NET library and as a native C++ library. This document describes the main differences between the two.

.NET Solid Framework	Native C++ Solid Framework
DEPLOYMENT	
Deployed as a single DLL (in 32 bit, 64 bit and AnyCPU versions). This is the only file that needs to be referenced from your project. This DLL contains a compressed archive of many separate Solid Framework files. The first time that Solid Framework is used the archive is automatically unpacked and copied into the user's AppData folder[1].	Deployed as a zip file containing all of the individual files that are needed by Solid Framework. This file needs to be manually unpacked. The SolidFramework.cpp and SolidFramework.h files must be included in your project, and *SolidFramework::Initialize(path)* must be called to load the unpacked files.
SUPPORT FOR CONVERTING MULTIPLE FILES AT THE SAME TIME	
The .NET library directly supports sequential conversions, and also includes SolidFramework.JobHandler.exe which provides support for concurrent processing. Solid Framework is not thread safe, but by passing processing to external applications (the JobHandlers), concurrent conversion can be achieved with minimal development effort. For more details see https://solidframework.net/portfolio-item/jobprocessor-net/	While sequential conversions are allowed, there is no built in support for *concurrent* conversions.

---

[1] Optionally, the user can configure a different folder into which the DLL will be unpacked.

SUPPORT FOR 64 BIT PROCESSES	
The JobProcessor (which creates JobHandlers) can create 64 bit JobHandlers even if the calling application is 32 bit (provided that the operating system supports this). This can allow faster conversion of files, better OCR, and increases the maximum complexity of file that can be converted.	While 64 bit processes can of course be created, if the calling application is 32 bit then *only* 32 bit conversions can occur, increasing the chance of running out of memory if more than 3GB of RAM is required[2].
SUPPORTED OPERATING SYSTEMS	
Windows	Windows Linux® macOS

Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

---

[2] This is a simplification of the memory limits imposed on a 32 bit process; nonetheless, they can cause significantly inferior results to those achievable by a 64 bit process.