

Visual COBOL チュートリアル

1

IIS と .NET COBOL コンポーネントを連携させた RESTful Web サービスの開発

1 目的

Visual COBOL では、他言語・他システムとの連携手法として、「Enterprise Server」を利用したサービス連携だけで はなく、COBOL プログラムに一切の変更を行わずに、 .NET 技術と連携可能な .NET COBOL 機能も提供していま す。本機能を利用することで、C#、VB .NET などの .NET 言語で作成した RESTful Web サービス上で COBOL を利用するといった方法も可能となります。

このドキュメントでは .NET COBOL 機能を利用して C# で実装する RESTful Web サービスと COBOL の連携方法について説明します。

2 前提

本チュートリアルは、下記の環境を前提に作成されています。

- 開発クライアント ソフトウェア
 - Windows 11

COBOL 開発環境製品 Visual COBOL 10.0 for Visual Studio

RESTful Web サービスの実装やテストクライアントなどにおいて、C#、IIS サーバー、jQuery などの技術を利用したチュ ートリアルとなっておりますが、これらの技術に関する説明は本チュートリアルでは行っておりません。別途資料やオンライン文 献などを参照ください。

チュートリアル用サンプルプログラム

下記のリンクから事前にチュートリアル用のサンプルファイルをダウンロードして、任意のフォルダーに解凍してください。 このサンプルプログラムは、COBOL で作成された簡単な書籍情報を管理するアプリケーションであり、索引ファイルを利用し ています。

サンプルプログラムのダウンロード



内容

- 1 目的
- 2 前提
- 3 チュートリアル手順
 - 3.1 今回作成するアプリケーション概要
 - 3.2 プロジェクトの作成と COBOL アプリケーションの確認
 - 3.2.1 プロジェクトの作成
 - 3.2.2 アプリケーションの動作確認
 - 3.3 .NET COBOL プロジェクトの作成
 - 3.4 C# による RESTful Web サービスの作成
 - 3.5 IIS サーバーへのサービス配置
 - 3.6 RESTful Web サービスの確認
- 4 補足
 - 4.1 IIS サーバーのインストール方法

3 チュートリアル手順

3.1 今回作成するアプリケーション概要

従来のコンソールアプリケーションである書籍情報を管理するアプリケーションを .NET COBOL の機能を利用して COBOL プログラムを変更することなく CIL コードを生成します。生成された CIL コードを利用して、Windows の IIS サーバーに RESTful Web アプリケーションとして登録した後、登録したアプリケーションが実際に動作することを確認します。



3.2 プロジェクトの作成と COBOL アプリケーションの確認

3.2.1 プロジェクトの作成

- 1) Visual Studio XXXX (XXXX はバージョン番号です。今回は 2022) を起動します。
- 2) [新しいプロジェクトの作成(N)] をクリックします。

開始する



3) 以下のフィルタを設定し、[コンソールアプリケーション]を選択し、[次へ(N)]をクリックします。

全ての言語: COBOL

全てのプラットフォーム: Windows

全てのプロジェクトの種類: コンソール



CBL			
C:/	コンソール アフリクーション (.NET Framework) コマンドライン アプリケーションを作成するためのプロジェクトです。		
C:\	コンソール アプリケーション ネイティブ コマンドライン アプリケーションを作成するためのづつジェクトです		
	WSDL or JSON/YAML から作成する Web サービスクライアント アプリケーション WSDL または JSON/YAML スキーマ ファイルから Web サービスクライアント アプリケーションを作成 するためのプロジェクトです。		
	COBOL Windows Enterprise Server Native コンソール		
_	コンハー川, アゴロケーション / NET) (Micro Forue)		
. ,	ンティーンティーンティーンティーン (Will for local) Windows および Linux 上で .NET で動作するコマンドライン アプリケーションを作成するためのプロ ジェクトです		
	COBOL Linux macOS Windows Common コンソール		
	探しているものが見つからない場合		
	さらにツールと機能をインストールする		
	戻る(B) 次へ(N)		
プロミン-			
-בעצ	ロクト名:"NativeCOBOL" ・ション名:"dotNETCOBOLREST"		
בעטע ארעב	ロクト名:"NativeCOBOL" -ション名:"dotNETCOBOLREST" レアプリケーション cobol Windows ネーティブ コンソール		
ソリュー ソリュー コンソー」 ^{九ジェクト4}	ロクト名:"NativeCOBOL" ・ション名:"dotNETCOBOLREST" レアプリケーション COBOL Windows ネーティブ コンソール		
ンリン ソリュー コンソー) プロジェクト名 NativeCC	ロクト名:"NativeCOBOL" ・ション名:"dotNETCOBOLREST" レアプリケーション COBOL Windows ネーティブ コンソール 6() 180L		
ン リリュー コンソー 「 加ジェクトネ 私 ivecCo 場所(L)	ロクト名:"NativeCOBOL" ・ション名:"dotNETCOBOLREST" レアプリケーション COBOL Windows ネーティブ コンソール BOL		
ンリノー ソリユー コンソー) プロジェクトネ NativeCC 場所(L) C:¥Usersh	ロクト名: "NativeCOBOL" -ション名: "dotNETCOBOLREST" レアプリケーション coBol Windows ネーティブ コンソール 5(0) 1801		
ソリュー コンソーJ プロジェクト4 NativeCC 場所(L) C:¥Users1 ソリューション	rクト名: "NativeCOBOL" -ション名: "dotNETCOBOLREST" レアプリケーション cobol Windows ネーティブ コンソール s(1) Bol tarot¥source¥repos ・ 生(M) ①		
ソリュー コンソー」 カンジェクト4 NativeCC 場所(L) C:¥Usersà ソリューション dotNETC	rクト名: "NativeCOBOL" ・ション名: "dotNETCOBOLREST" レアプリケーション coBoL Windows ネーティブ コンソール s(/) BOL tarot¥source¥repos		
ソリュー コンソー」 加ジェクト4 NativeCC 場所(L) C:¥Usersà ソリューション dotNETC ソリュー	tクト名: "NativeCOBOL" -ション名: "dotNETCOBOLREST" レアプリケーション COBOL Windows ネーティブ コンソール a() BOL 		
ソリュー コンソー」 プロジェクト4 NativeCC 場所(1) C:¥Usersa ソリューション dotNETCC ソリューション	t クト名: "NativeCOBOL" -ション名: "dotNETCOBOLREST" レアプリケーション coBoL Windows ネーティブ コンソール 5(/) BOL tarotišsource¥repos		
ソリュー コンソー」 プロジェクト4 NativeCC 場所(1) C:¥Usersh yリューション dotNETC ソリューション プロジェクト (tクト名: "NativeCOBOL" -ション名: "dotNETCOBOLREST" レアプリケーション coBoL Windows ネーティブ コンソール 5(0) 180L tarot¥source¥repos ・ *(M) ① OBOLREST ションとプロジェクトを同じディレクトリに配置する(D) は "C#Users¥tarot¥source¥repos¥dotNETCOBOLREST¥NativeCOBOL¥" で作成されます		
ソリュー コンソー」 プロジェクト4 NativeCC 場所(L) C:¥Usersh VJリューション dotNETCC ソリューション プロジェクト1	tクト名: "NativeCOBOL" -ション名: "dotNETCOBOLREST" レアプリケーション coBoL Windows ネーティブ コンソール 5(0) 180L tarot¥source¥repos ・ 年(M) ① CBOLREST ションとプロジェクトを同じディレクトリに配置する(D) は "C#Users¥tarot¥source¥repos¥dotNETCOBOLREST¥NativeCOBOL¥"で作成されます		
ソリュー コンソー」 カジェクト4 NativeCC 場所(L) C:¥Usersà yリューション dotNETC コジェクト1	tクト名: "NativeCOBOL" -ション名: "dotNETCOBOLREST" レアプリケーション coBoL Windows ネーティブ コンソール 5(0) 180L tarot¥source¥repos ・ *4(M) ① DBOLREST ションと力ロジェクトを同じディレクトリに配置する(D) ま "C#Users¥tarot¥source¥repos¥dotNETCOBOLREST¥NativeCOBOL¥" で作成されます		
ソリュー コンソー」 カジェクト4 NativeCC 場所(L) C:¥Users3 yリューション dotNETC カジェクト1	tクト名: "NativeCOBOL" -ション名: "dotNETCOBOLREST" レアプリケーション coBoL Windows ネーティブ コンソール 5(0) 180L 		
ソリュー コンソー」 カジェクト4 NativeCC 場所(L) C:¥Users3 ソリューション dotNETC フリジェクト1	tクト名: "NativeCOBOL" -ション名: "dotNETCOBOLREST" レアプリケーション COBOL Windows ネーティブ コンソール (0) BOL 		
ソリユー コンソー」 カジェクト4 NativeCC 場所(L) C:¥Users* ソリューション dotNETC ソリューション プロジェクト1	tクト名: "NativeCOBOL" -ション名: "dotNETCOBOLREST" レアプリケーション COBOL Windows ネーティブ コンソール 4(0) BOL 		
ソリュー コンソーJ カンジェクト4 NativeCC 場所(L) C:¥Usersh yリューション dotNETC ソリューション カンジェクト	t2クト名: "NativeCOBOL" ・ション名: "dotNETCOBOLREST" レアプリケーション COBOL Windows ネーティブ コンソール は0 BOL tarot¥source¥repos		
ソリュー コンソーJ 力ジェクト4 NativeCC 場所(L) C:¥Users1 グリューション dortNETC ソリュー プロジェクト1	tクト名: "NativeCOBOL" -ション名: "dotNETCOBOLREST" <i>N アプリケーション</i> coBoL Windows ネーティブ コンソール 		
ソリュー コンソー」 プロジェクト4 NativeCC 場所(L) C?¥Users3 dotNETC ツリューション プロジェクト()	tクト名: "NativeCOBOL" -ション名: "dotNETCOBOLREST" レアプリケーション cogol Windows ネーティブ コンソール 	戻る(B)	1

5) 自動作成された Program1.cbl は不要のため、削除してください。

4



6) チュートリアルファイル解凍フォルダー¥COBOL フォルダー配下の「BOOKSCRN.cbl」,「BOOK.cbl」,「BOOK-INFO.cpy」をプロジェクト内にドラッグアンドドロップします。



7) チュートリアルファイル解凍フォルダー配下の DAT フォルダーを任意のフォルダーにコピーします。フォルダー配下には、書籍情報を管理する索引ファイルが保存されています。

本チュートリアルでは、C:¥dotNETManageTutorial¥DAT としています。

3.2.2 アプリケーションの動作確認

1) NativeCOBOL プロジェクト名を選択し、マウスの右クリックにてコンテクストメニューを表示した上で、[プロパティ

(R)] を選択します。			
ソリューション エクスプローラー			- ∓ X 1
 □ □	*	ビルド(U) リビルド(E) クリーン(N)	
MativeCOBOL Properties BOOK.cbl BOOK-IN	•	すべての子孫を折りたたむ ここまでスコープ指定する(S) 新しいソリューション エクスプローラーのビュー(N)	Ctrl+左矢印
BOOKSCF	.0	追加(D) 既存の COBOL 項目を追加 NuGet パッケージの管理(N)	•
	÷	スタートアップ プロジェクトの構成 スタートアップ プロジェクトに設定(A) デバッグ(G)	•
	C,	プロジェクト詳細	
	χ	切り取り(T)	Ctrl+X
	Ĝ	貼り付け(P)	Ctrl+V
	×	削除(V)	Del
	Ē	名前の変更(M)	F2
		指令の確定 すべてのプログラムにプロジェクトのデフォルトを使用する Micro Focus Code Analysis	•
		プロジェクトのアンロード(L) 直接依存関係の読み込み	
		すべての依存関係ツリーの読み込み	
ソリューション エクスプローラー	∂	エクスプローラーでフォルダーを開く(X) ターミナルで開く	
	۶	プロパティ(R)	Alt+Enter



2) 「アプリケーション」 タブを選択し、「エントリポイント」に "BOOKSCRN" を入力した上で、[環境(E)] をクリックし

љ9°		
NativeCOBOL* ↔ ×		
アプリケーション	構成(C): なし	~
SQL デパッグ	プラットフォーム(<u>M</u>): なし	~
コピーブック プリプロセッサ	出力の名前: NativeCOBOL	
COBOL COBOL リンク Micro Focus Code Analysis	出力の種類: コンソール アプリケーション	出力先: ○ 単一の実行可能ファイル
	エントリポイント: BOOKSCRN	○ 複数の実行可能ファイル
	lbr にパッケージ化	環境(<u>E)</u>

- 3) [追加] をクリックし、以下の入力を行ったうえで、[OK(O)] をクリックします。
 - 変数: "BOOKINFO"
 - 值: "C:¥dotNETManageTutorial¥DAT¥BOOKINFO.DAT"

	変数	値		
•	BOOKINFO	C:¥dotNETManageTutorial¥DAT¥BOOKINFO.DAT		追加
			I	削除
- -	つぶっちし のじリ じゅしっか おりた	(古田+7/11)		
	ロンエクトのビルト時に変数を	使用∮る(U)		
			OK(O)	キャンセル(C)

4) 「COBOL」タブを選択し、「追加指令」欄に "ASSIGN(EXTERNAL)" を入力します。

構成(C): アクティブな (Debug) ~ ブラットフォーム(M): アクティブな (x64)	~
出力パス: .¥bin¥x64¥Debug	¥
□ 指令ファイルの生成	□ リストファイルを生成
□ コード カバレッジを有効にする	□ プロファイラを有効にする
追加指令 ————————————————————————————————————	
ASSIGN(EXTERNAL)	
	 構成(C): アクティブな (Debug) ブラットフォーム(M): アクティブな (x64) 出力パス: ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・

- 5) 変更を保存します。
- 6) [デバッグ(D)] > [デバッグなしで開始(H)] をクリックします。



デバ	ッグ(D)	テスト(S)	分析(N)	ツール(T)	拡張機能(X)	
	ウィンド	ウ(W)				۲
	デバッグ	の開始(S)			F5	
\triangleright	デバッグ	なしで開始(ŀ	H)		Ctrl+F5	
2	パフォー	יערשל געד	イラー(F)		Alt+F2	

上記のような画面が表示されます。

以下の入力を行った後、Enter キーを打鍵すると、書籍情報が取得できます。

FUNCTION: "1"

ストック番号: "1111"

lin-	NativeCOBOL ×	+ ~	
	FUNCTION: [1] READ=1	, ADD=2, DELETE=3 NEXT=4, GROSSSALES=	S, QUIT=9
	ストック番号: [111:	1]	
	タイトル: [LOAD OF ⁻	THE RING	1
	ジャンル: [FANTASY]	
	著者: [TOLKIEN]
	小売価格: [1500] 売上: [000001000]	仕入れ:[2000]	
	 STATUS· ΓΩΩ]		

現在、ストック番号: 4444 の書籍は未登録のため、以下の情報を入力し、Enter キーを打鍵することで、情 報を追加します。 FUNCTION: "2" ストック番号: "4444" タイトル: "ブレイブストーリー" ジャンル: "ファンタジー" 著者: "宮部みゆき"

小売価格: "3000" 仕入れ: "1000"

仕入れ: "1000"売上: "900"

FUNCTION: [Þ] READ=1, ADD=2, DELETE=3 NEXT=4, GROSSSALES=S, QUIT=9 ストック番号: [44444] タイトル: [ブレイブストーリー] ジャンル: [ファンタジー] 著者: [宮部みゆき] 小売価格: [3000] 仕入れ: [1000] 売上: [0000000900]



実行後、READ 機能でストック番号: 4444 が削除されていることを確認してください。

FUNCTION=S の GROSSSALES 機能は登録された全書籍情報の売上額を集計します。

<pre>FUNCTION: [S] READ=1, ADD=2, DELETE=3 NEXT=4, GROSSSALES=S, QUIT=9</pre>
ストック番号: []
タイトル: [************************************
ジャンル: [*********************]
著者: [************************************
小売価格: [0] 仕入れ: [0] 売上: [017000000]
STATUS: [00]

確認後、FUNCTION=9 でアプリケーションを終了してください。



3.3 .NET COBOL プロジェクトの作成

本節では、さきほど確認した従来の COBOL プログラムを .NET COBOL としてコンパイルを行います。

1) Visual Studio IDE のソリューションエクスプローラーより、"dotNETCOBOLREST" ソリューション名を選択し、マウスの 右クリックにてコンテクストメニューを表示した上で、[追加(D)] > [新しいプロジェクト(N)] をクリックします。



以下のフィルタを設定し、[クラスライブラリ (.NET Framework)] を選択し、[次へ(N)] をクリックします。
 全ての言語: COBOL

全てのプラットフォーム: Windows

全てのプロジェクトの種類: ライブラリ

COBOL	・ Windows ・ ライブラリ	
	Windows フォーム コントロール ライブラリ (.NET Framework) Windows フォーム アプリケーションで使用するコントロールを作成するプロジェクトです。 COBOL Windows デスクトップ ライブラリ	
	クラス ライブラリ (.NET Framework) 他のアプリケーションで使用するクラスを作成するためのプロジェクトです。 COBOL Windows ライブラリ	
	リンク ライブラリ 他のアプリケーションで使用するクラスを作成するためのネイティブ プロジェクトです。 COBOL Windows ネーティブ ライブラリ	
	ユニット テスト ライプラリ MFUnit テスト ライブラリを作成するためのネイティブ プロジェクトです。 COBOL Windows ネーティブ テスト ライブラリ	
	WPF ユーザー コントロール ライブラリ (.NET Framework) Windows Presentation Foundation ユーザー コントロール ライブラリです。 COBOL Windows デスクトップ ライブラリ	
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	ASPNET サーバー コントロール Web アプリケーションで使用するコントロールを作成するためのプロジェクトです。	
		次へ(N)

© Rocket Software, Inc. or its affiliates 1990–2024. All rights reserved. Rocket and the Rocket Software logos are registered trademarks of Rocket Software, Inc. Other product and service names might be trademarks of Rocket Software or its affiliates.



3) プロジェクト名に "BookClassLibrary" を入力し、[作成(C)] をクリックします。

4) BookClassLibrary プロジェクト作成時に自動作成される Class1.cbl は不要なため、削除してください。



5) NativeCOBOL プロジェクト配下の「BOOK.cbl」,「BOOK-INFO.cpy」を選択し、マウスの右クリックにてコンテクスト メニューを表示した上で、[コピー(Y)] をクリックします。

ソリューション エクスプローラー			▼ ₽×	٦ ۲
🖉 🐚 - 🖛 🗇 🎤 📥	Ŧ			৾৾৴৾৾ঀ
ソリューション エクスプローラー の検索 (Ct	trl+:)		- م	
😡 ソリューション 'dotNETCOBOLRE	ST' (2	2/2 のプロジェクト)		
 BookClassLibrary 				
🔑 Properties				
▶□ 参照設定				
A CBL NativeCOBOL				
Properties				
BOOK.cbl	¢	開<(O)		
BOOK-INFO.cpy		ファイルを開くアプリケーションの選択(N)		
P BOOKSCRIN.CDI		コードのクリーンアップ		,
	0- 	新しいソリューション エクスプローラーのビュー(N)		
	²	選択したものを比較(A)		
		プロジェクトから除外(J)		
		COBOLプロジェクトを作成		
	Х	切り取り(T)	Ctrl+X	
	ď	⊐ピ−(Y)	Ctrl+C	

6) BookClassLibrary プロジェクト名を選択し、マウスの右クリックにてコンテクストメニューを表示した上で、[貼り付け(P)] をクリックします。



ソリューション エクスプローラー				▼ ₽ ×	۲Ľ
🖉 🐻 • ≒ 🗐 🗗 🎤 🗖	6- T	7			९ैन्र
ソリューション エクスプローラー の検索	(Ctrl+	-:)		- م	
Yリューション 'dotNETCOBOL BookClassLibrary Properties Pro 参照設定 MativeCOBOL Properties BOOK.cbl	i.	ビルド(U) リビルド(E) クリーン(N)			
	₽ *∃	すべての子孫を折りたたむ ここまでスコープ指定する(S) 新しいソリューション エクスプローラーのビュー(I	N)	Ctrl+左矢	ÉD
BOOK-INFO.cpy ▷ ⓓ BOOKSCRN.cbi	B	ビルドの依存関係(B) 追加(D) 既存の COBOL 項目を追加 NuGet パッケージの管理(N)			+
	- - - - - - - - - - - - - - - - - - -	スタートアップ プロジェクトの構成 スタートアップ プロジェクトに設定(A) デパッグ(G)			•
	Q, V	プロジェクト詳細		CL V	
	Ĝ	500取り(1) 貼り付け(P)		Ctrl+X Ctrl+V	
	Ŧ				
ソリューション エクスフローラー の検索 (C マリューション 'dotNETCOBOLR	Ctrl +:) EST' (2/2 のプロジェクト)			
▲ BookClassLibrary					
BOOK-INFO.cpy					

- BOOKSCRN.cbl
- 7) BookClassLibrary プロジェクト名を選択し、マウスの右クリックにてコンテクストメニューを表示した上で、[プロパティ(R)] をクリックします。



	×	プロパティ(R)	Alt+Enter
ソリューション エクスプローラー Git 変	۶	ターミナルで開く	
	¢	エクスプローラーでフォルダーを開く(X)	
		すべての依存関係ツリーの読み込み	
		直接依存関係の読み込み	
		プロジェクトのアンロード(L)	
	Ē	名前の変更(M)	F2
	X	削除(V)	Del
	Ĝ	貼り付け(P)	Ctrl+V
	X	切り取り(T)	Ctrl+X
	C,	プロジェクト詳細	
		デバッグ(G)	•
		スタートアップ プロジェクトに設定(A)	
	දිදිදු	スタートアップ プロジェクトの構成	
		NuGet パッケージの管理(N)	
BOOKSCRN.cbl		既存の COBOL 項目を追加	
BOOK-INFO.cpy		追加(D)	•
Properties		ビルドの依存関係(B)	•
▲ MativeCOBOL	*=	新しいソリューション エクスプローラーのビュー(N)	
BOOK-INFO.cpy		ここまでスコープ指定する(S)	
▷ BOOK.cbl	Ξ	すべての子孫を折りたたむ	Ctrl+左矢印
➢ Properties		クリーン(N)	
BookClassLibrary		リビルド(E)	
😽 ソリューション 'dotNETCOBOL	i.	ビルド(U)	

8) 「COBOL」 タブを選択し、[.NET コードに集団のリンケージ項目を公開] にチェックを行い、追加指令に "ASSIGN(EXTERNAL)" を入力します。

BookClassLibrary* 👳	× NativeCOBOL				BOOK-INFO.cpy	%
アプリケーション SQL	構成(C): アクティブな (Debug)	~ 7	^プ ラットフォーム(M):	アクティブな (Any CPU)	~	
デバッグ	出力パス: .¥b	in¥Debug¥			参照.	•
コピーブック 名前空間	□ 指令ファイルの生成		🗌 リストファイ	ルを生成		
COBOL						
リソース 設定	Smart Linkage					
	✓ .NET コードに集団のリンケー	ジ項目を公開	オプション			
	追加指令 ———					
	ASSIGN(EXTERNAL)					•
						-

補足)

[.NET コードに集団のリンケージ項目を公開] にチェックを行うことで、ILSMARTLINKAGE というコンパイラ指令を設定 したことになります。

本指令の詳細については、製品マニュアルトップより、[リファレンス] > [コンパイラ指令] > [.NET COBOL コマンド ライ ン コンパイラ指令] > [コード生成指令] > [ILSMARTLINKAGE] を参照してください。

9) 構成を "Release" に変更し、前手順と同様の設定を行い、保存します。



BookClassLibrary +	× NativeCOBOL				BOOK-INFO.cpy	
アプリケーション SQL	構成(C): Release	\checkmark	プラットフォーム(M):	アクティブな (Any CPU)	~	
デバッグ	出カパス:	.¥bin¥Release¥			参照	
コピーブック 名前空間	□ 指令ファイルの生成		🗌 リストファイ	ルを生成		
COBOL						
リソース 設定	Smart Linkage VIT コードに集団のリン 追加指令	バージ項目を公開	オプション			
	ASSIGN(EXTERNAL)				,	•

10) [ビルド(B)] > [BookClassLibrary のビルド(U)] を選択して、ライブラリのビルドを行います。



生成された BookClassLibrary.dll は Visual COBOL のコンパイラにより、コードの変更なく .NET Framework 上で動作するよう、CIL 形式で DLL を作成しています。このため、C#, VB .NET のような .NET 言語と連携すること が可能になります。

3.4 C# による RESTful Web サービスの作成

さきほど作成した .NET COBOL のライブラリを RESTful Web サービスから呼び出します。RESTful Web サービスは、C# で作成します。

1) Visual Studio IDE のソリューションエクスプローラーより、"dotNETCOBOLREST" ソリューション名を選択し、マウスの 右クリックにてコンテクストメニューを表示した上で、[追加(D)] > [新しいプロジェクト(N)] をクリックします。



以下のフィルタを設定し、[ASP.NET Web アプリケーション (.NET Framework)] を選択し、[次へ(N)] をクリックします。

全ての言語: C#

全てのプラットフォーム: Windows

全てのプロジェクトの種類: Web

C#	✓ Windows ✓ Web
0	Blazor WebAssembly アプリ WebAssembly で実行される Blazor アプリを作成するためのプロジェクト テンプレート。このテンプ レートは、高度でダイナミックなユーザー インターフェイス (UI) を備えた Web アプリに使用できます。
gRPC	C# Linux macOS Windows Blazor クラウド Web ASPINET Core gRPC サービス ASPINET Core を使用して gRPC サービスを作成するプロジェクト テンプレート。native AOT として
	の公開もオノションでサホートされます。 C# Linux macOS Windows クラウド サービス Web NUnit テストプロジェクト Windows, Linux および MacOS 上の .NET で実行できる NUnit テストを含むプロジェクト。
	C# Linux macOS Windows デスクトップ テスト Web
51	ASRNET Web アプリケーション (.NET Framework) ASRNET アプリケーションを作成するためのプロジェクト テンプレートです。ASRNET Web Forms、 MVC、または Web API のアプリケーションを作成し、ASRNET のさまざまな機能を追加できます。 C# Windows クラウド Web
	Web Driver Test for Edge (.NET Core) (Microsoft WebDriver を使用して) Edge ブラウザー内の Web サイトの UI テストを自動化できる 単体テストを含むプロジェクトです。
	C# Windows Web JAP
	次へ(N)



3) プロジェクト名に "RESTfulWS" を入力し、[作成(C)] をクリックします。

ロジェクト	~名(J)	
RESTfu	WS	
所(L)		
C:¥User	s¥tarot¥source¥repos¥dotNETCOBOLREST •	
V-47-	-ク(F)	
NET Fra		
コジェクト	トは "C:¥Users¥tarot¥source¥repos¥dotNETCOBOLREST¥RESTfulWS¥" で作成されます	
		戻る(B) 作成(C)
示。	されたフォームにて「Web API」を選択し、[作成] をクリックします。	
示で 新し	されたフォームにて「Web API」を選択し、[作成]をクリックします。 しい ASP.NET Web アプリケーションを作成する 2 ASPNET アブリケーションを作成するための空のプロジェクト テンプルートには内容はありません。	認証 なし
示。 新し	されたフォームにて「Web API」を選択し、[作成]をクリックします。 しい ASP.NET Web アプリケーションを作成する 2 ASPNET アプリケーションを作成するための空の元ジェクト テンルート。このテンルートには内容はありません。	認証 なし ~
示。 新し 『1	されたフォームにて「Web API」を選択し、[作成]をクリックします。 しい ASP.NET Web アプリケーションを作成する タ ASPNET アガリケーションを作成するための空のカジェクト テンカート、このテンカートには内容はありません。 Web Forms ASPNET Web Forms アブリケーションが作成するためのカジェクト テンカート、ASPNET Web Formsを使用すると 一般のただうか	認証 なし
示 新し 『	されたフォームにて「Web API」を選択し、[作成]をクリックします。 い ASP.NET Web アプリケーションを作成する ASP.NET アプリケーションを作成するための空の加ジェクト テンルート、このテンルートには内容はありません。 Web Forms ASSNET Web Forms アプリケーションを作成するためのプロジェクト テンルート、ASSNET Web Formsを使用すると、一般的なドラッ グアンド ドロップのイベント駆動モデルを使用して、動的な Web サイトを掲載できます。デザイン園面および数百のコントロールとコンボー ないため利用で、データアクレネ構成するためのプロジェクト デンルート、ASSNET Web Formsを使用すると、一般的なドラッ	認証 なし - フォルダーおよびコア参照を追加する
	されたフォームにて「Web API」を選択し、[作成]をクリックします。 こい ASP.NET Web アプリケーションを作成する タ ASP.NET アプリケーションを作成するための空のプロジェクト テンプルート。このテンプルートには内容はありません。 Web Forms ASSNET Web Forms アプリケーションを作成するためのプロジェクト テンプルート。ASP.NET Web Formsを使用すると、一般的なドラッ グ アンド ドロップロイベント駆動モデリルを使用して、動的な Web サイトを得筆できます。デザイン画面および数百のコントロールとコンボー ネットを利用して、データ アクセスを備えた高度で強力な UI 主導のサイトを、短時間で作成できます。	認証 なし フォルダーおよびコア参照を追加する Web フォーム(r) WVC(M)
示 新 し 『 「	されたフォームにて「Web API」を選択し、[作成]をクリックします。 スレ ASP.NET Web アプリケーションを作成する タ ASPNET アブリケーションを作成するための空の力ジェクト アンプレート、このテンプレートには内容はありません。 Web Forms ASPNET Web Forms アブリケーションを作成するためのプロジェクト アンプレート、ASPNET Web Formsを使用すると、一般的なドラッ グアンド ドロップのイベント駆動モデリを使用するためのプロジェクト アンプレート、ASPNET Web Formsを使用すると、一般的なドラッ グアンド ドロップのイベント駆動モデリを使用するためのプロジェクト アンプレート、ASPNET Web Formsを使用すると、一般的なドラッ グアンド ドロップのイベント駆動モデリを使用するためのプロジェクト アンプレート、ASPNET Med Formsを使用すると、一般的なドラッ グアンド ドロップのイベント駆動モデリを使用するためのプロジェクト チンプレート、ASPNET Med Formを使用すると、一般的なドラッ グロントを利用して、データ アクセスを確認するためのプロジェクト ASPNET Med Form Controller アーチョク	認証 なし フォルダーおよびコア参照を追加する ↓ Web フォーム(F) ✓ MVC(M) ✓ Web API(W)
	されたフォームにて「Web API」を選択し、[作成] をクリックします。 こい ASPENET Web アプリケーションを作成するための空のプロジェクト テンプレート。 ASPNET アプリケーションを作成するための空のプロジェクト テンプレート。このテンプレートには内容はありません。 Web Forms ASSNET Web Forms アプリケーションを作成するためのプロジェクト テンプレート。ASPNET Web Formsを使用すると、一般的なドラッ グ アンド トロップロイント駆動モデルを使用して、動めな Web ゲイトを構築できます。デザイン画面および数百のコントロールとコンボー ネットを利用して、データ アクセスを備えた高度で強力な UI 主導のサイトを、短時間で作成できます。 MVC ASPNET MVC アプリケーションを作成するためのプロジェクト テンプレート。 ASPNET MVC では、Model-View-Controller アーキテク デモを使用してアプリケーションを作成するためのプロジェクト テンプレート。 ASPNET MVC では、Model-View-Controller アーキテク デアプリケーションを作成するためのプロジェクト TVTレート。 ASPNET MVC では、Model-View-Controller アーキテク	認証 なし フォルダーおよびコア参照を追加する ↓ Web フォーム(F) ↓ WVC(M) ↓ Web API(W)
	されたフォームにて「Web API」を選択し、[作成]をクリックします。 こい ASP.NET Web アプリケーションを作成するための アプリケーションを作成する タ ASPNET アプリケーションを作成するための 空の プレジェクト テンプルート、 20 テンプルートには 内容は ありません。 Web Forms ASPNET Web Forms アプリケーションを作成するための プロジェクト テンプルート、 ASPNET Web Formsを使用すると、一般的な Form グ アンド ドロップの イベント 駆動 モデル を使用すると、 一般的な Web サイトを構築できます。 デザイン画面 および 数百の コントロールとコンボー ネットを利用して、 データ アクセスを備美できます。 ASPNET MyC では、 Model・View-Controller アーキテク ゲモを使用してアプリケーションを作成するための プロジェクト テンプルート、 ASPNET MyC では、 Model・View-Controller アーキテク ゲモを使用してアプリケーションを作成するための プロジェクト Fy アントート ASPNET MyC では、 Model・View-Controller アーキテク ゲモを使用してアプリケーションを作成するための タンロジェクト ASPNET MyC では、 Model・View-Controller アーキテク ゲモを使用してアプリケーションを作成するための タンロジェクト	認証 な↓ フォルダーおよびコア参照を追加する ○ Web フォ−ム(F) ✓ MVC(M) ✓ Web API(W) 詳細設定
	されたフォームにて「Web API」を選択し、[作成]をクリックします。 ン ASP.NET Web アプリケーションを作成するための空の 加ジェクト テンルート、このテンルートには内容はありません。 Web Forms ASPNET アプリケーションを作成するための アロジェクト テンルート、COFY ルート、COFY ルート、	認証 なし フォルダーおよびコア参照を追加する ○ Web フォーム(F) ○ MVC(M) ○ Web API(W)
	されたフォームにて「Web API」を選択し、[作成]をクリックします。 CV ASPENET Web アプリケーションを作成するための空の カジェクト テンプレート。このテンプレートには内容はありません。 Web Forms ASPNET Web Forms アプリケーションを作成するための プロジェクト テンプレート。ASPNET Web Formsを使用すると、一般的なドラッ グアンド FDップのイベント駆動モデルを使用して、動的な Web ダイトを構築できます。デザイン画面および数百のコントロールとコンボー イントを利用して、データ アクセスを備えた高度で強力な UI 主導のサイトを、短時間で作成できます。 MCC ASPNET MVC アプリケーションを作成するための プロジェクト Fy アレート、ASPNET MVC では、Model- View-Controller アーキテク デャプリケーションを作成するための プロジェクト Fy アレート、ASPNET MVC では、Model- View-Controller アーキテク デャプリケーションを作成するための 多くの機能が備わっています。 Web API プラワザーやモバイル デバイスを含む幅広いクライアントに到達できる RESTFul HTTP サービスを作成するための プロジェクトッー	認証 なし マ フォルダーおよびコア参照を追加する ↓ Web 7オーム(f) ✓ MVC(M) ✓ Web API(W)
	されたフォームにこて「Web API」を選択し、[作成]をクリックします。 CV ASPENET Web アプリケーションを作成するための アプリケーションを作成する ASPNET アプリケーションを作成するための空の プジェクト アンプレート、この アンプレートには 内容はありません。 Web Forms ASPNET Web Forms アプリケーションを作成するための プロジェクト アンプレート、ASPNET Web Formsを使用すると、一般的な ドラッ グッンド ドロップのイベント駆動モデリ を使用するための プロジェクト アンプレート、ASPNET Web Formsを使用すると、一般的な ドラッ グッンド ドロップのイベント駆動モデリ を使用するための プロジェクト アンプレート、ASPNET Web Formsを使用すると、一般的な ドラッ グッンド ドロップのイベント駆動モデリ を使用するための プロジェクト トンプリート 、ASPNET MVC では、Model・View-Controller アーキアク デレを使用して アプリケーションを作成するための プロジェクト トンプレート、ASPNET MVC では、Model・View-Controller アーキアク デレを使用して アプリケーションを作成するための プロジェクト マンプレート 、ASPNET MVC では、Model・View-Controller アーキアク デレを使用して アプリケーションを作成するための プロジェクト トンプレート、ASPNET MVC では、Model・View-Controller アーキアク デレーキャンチャンチャンチャンチャンチャンチャンチャンチャンチャンチャンチャンチャンチャ	認証 なし
	されたフォームにこて「Web API」を選択し、[作成]をクリックします。 CV ASPENET Web アプリケーションを作成するための空のカジェクト テンカート、このテンカートには内容はありません。 PA ASPNET アプリケーションを作成するための空のカジェクト テンカート、COFYカート、COFYカートには内容はありません。 Veb Forms ASSNET Web Forms アプリケーションを作成するためのカビジェクト テンカート、ASSNET Web Formsを使用すると、一般的なドラッ グッンド ドロップのイベント駆動モデルを使用して、動的な Web サイトを構築できます。デザイン画面および数百のコントロールとコンボー クッンド ドロップのイベント駆動モデルを使用して、動的な Web サイトを構築できます。デザイン画面および数百のコントロールとコンボー クッンド ドロップのイベント駆動モデルを使用して、動的な Web サイトを構築できます。デザイン画面および数百のコントロールとコンボー クッンド ドロップのイベント駆動モデルを使用して、動的な Web サイトを構築できます。デザイン画面および数百のコントロールとコンボー クッンド ドロップのケーションを作成するためのカビッジェクト、大切や「Web Formsを使用して、データ P かとを確認する Company Form Form Form Form Form Form Form Form	認証 なし
	されたフォームにこて「Web API」を選択し、[作成]をクリックします。 このASPNET Web アプリケーションを作成するための空のカジェクト アンカート、このアンカートには内容はありません。 そ ASPNET アプリケーションを作成するための空のカジェクト アンカート、このアンカートには内容はありません。 そ ASPNET Web Forms アプリケーションを作成するためのカビジェクト アンカート、ASPNET Web Formsを使用すると、一般的なドラッ クッンド ドロップのイベント駆動モデリルを使用すると、の般的な Web サイトを構築できます。デザイン画面および数百のコントロールとコンポー ペントを利用して、データ アクセンを確成することののカビジェクト アンカート、ASPNET Web Formsを使用すると、一般的なドラッ クッンド ドロップのイベント駆動モデリルを使用すると、の後的な Form クッンド ドロップのイベント駆動モデリルを使用すると、の後的な Form クッンド ドロップのクイント駆動モデリルを使用すると、の後のな Form クッンド ドロップのイベント駆動モデリルを使用すると、の後のな Form クッンド ドロップの イベント駆動モデリルを使用 でた いた ひまつ Controller アーキテク たを使用して アブリケーションを作成するための プロジェクト Fy アントート、ASPNET MVC では、Model-View-Controller アーキテク そを使用して アブリケーションを作成するための プロジェクト Fy アントート、ASPNET MVC では、高速なテスト駆動開発をはじめとした最新の標準技術を使用 アブリケーションを作成するための Controller アーキテク ため Controller アーキテク クリー Pote Full Pote Pote Full Pote Pote Full Pote Full Pote Full Pote Pote Pote Pote Pote Pote Pote Pote	認証 なし ● フォルダーおよびコア参照を追加する ● Web フォーム(F) ● MVC(M) ● Web API(W) 詳細設定 ■ HTTPS 用の構成(C) ■ フンラナーサポート (Docker Desktop が必要) ■ 単体テストのためのプロジェクトも作成する(U) RESTfutWS.Tests

> [プロジェクトの依存関係(S)] を選択します。



- ‡	ソリューション エクスプローラー		ビュー(W)	•
	,∄ [™] 0 • ≒ [™] ⊟ @ /۶		分析とコードのクリーンアップ(Z)	•
	ソリューション エクスプローラー の検索	Ð	発行(B)	
	😡 ソリューション 'dotNETCOBOL		アップグレード	
	BookClassLibrary	* ?	Application Insights の構成(C)	
	🔑 Properties		概要(V)	
		Ξ	すべての子孫を折りたたむ	Ctrl+左矢印
			ここまでスコープ指定する(S)	
	▲ I NativeCOBOL	<u>•</u> =	新しいソリューション エクスプローラーのビュー(N)	
7	ロジェクトの依存関係(S)		ビルドの依存関係(B)	•
7	ロジェクトのビルド順序(I)		追加(D)	•
	BOOKSCRN.cbl		NuGet パッケージの管理(N)…	
	▲ 🗿 RESTfulWS	P	クライアント側のライブラリを管理する(M)	
	Connected Services		ユーザー シークレットの管理(G)	

6) [BookClassLibrary] にチェックを行った後、[OK] をクリックします。

依存関係 ビルドの順序		
プロジェクト(R):		
RESTfulWS		~
依存先(D):		
BookClassLibrary NativeCOBOL		
	ОК	キャンセル

- 7) RESTfulWS プロジェクト配下の参照を選択し、マウスの右クリックにてコンテクストメニューを表示した上で、[参照の追加
 - (R)] を選択します。





8) 「アセンブリ」タブ配下の「拡張」タブを選択し、「Micro Focus Runtime」と「Micro Focus Runtime(Interop RuntimeServices)」のチェックを行います。

フレームワーク 名前 パージョン 名前: 施速 EnvDTE100 10.0.0.0 Rutimeservices 最近使用したファイル EnvDTE100 10.0.0.0 Rutimeservices) アカウト EnvDTE80 80.00 Rutimeservices) サオオカジェクト envdte90 17.0.0.0 Rutimeservices) レ 共有力ジェクト envdte90 17.0.0.0 Rutimeservices) レ 大田クTE90 9.0.0.0 イージョン: レ 大田クTE90 9.0.0.0 アーイル パージョン: レ 大田クTE90 9.0.0.0 Hvak Common Library 10.0.0 レ かいてo Focus Rutime 4.0.0.0 アーイル パージョン: 1000.0000.0.82 レ かいてo Focus Rutime (Azure OSSP) 4.0.0.0 7.7.0.0 1000.0000.0.82 レ かいてo Focus Rutime (Azure OSSP) 4.0.0.0 7.0.0.0 7.0.0.0 Micro Focus Rutime (Core Tracing) 4.0.0.0 7.0.0.0 7.0.0.0 Micro Focus Rutime (Core Tracing) 4.0.0.0 7.0.0.0 7.0.0.0 Micro Focus XIL Extensions Interop Assembly 9.0.0.0 7.0.0.0 7.0.0.0 Microsoft Azure SQL Server Management Libray 0.0.0 7.0.0.0 7.0.0.0 Microsoft Azure SQL Serv	▲ アセンブリ	ターゲット: .NET Framework 4.7.2			検索 (Ctrl+E)
✓ Micro Focus Runtime 4.0.0.0 Micro Focus Runtime (Azure OSSP) 4.0.0.0 Micro Focus Runtime (Core Tracing) 4.0.0.0 Micro Focus Runtime (Core Tracing) 4.0.0.0 Micro Focus Runtime (Interop RuntimeServices) 4.0.0.0 Micro Focus Runtime (Interop RuntimeServices) 4.0.0.0 Micro Focus XML Extensions Interop Assembly 9.00.0 Micro Focus XML Extensions Interop Assembly 2.0.0 Microsoft Azure Common Library 2.0.0.0 Microsoft Azure Resource Management Library 2.0.0.0 Microsoft Azure SQL Server Management Library 0.9.0.0 Microsoft Azure SQL Server Management Library 0.0.0 Microsoft Azure SQL Server Management Library 17.0.0.0 Microsoft Azure SQL Server Management Library 10.0.0 Microsoft Azure SQL Server Management Library 10.0.0 Microsoft Azure SQL Server Manageme	 フレームワーク 拡張 最近使用したファイル プロジェクト 共有プロジェクト COM 参照 	名前 EnvDTE100 envdte80 EnvDTE80 envdte90 EnvDTE90 envdte90a EnvDTE90a EnvDTE90a Extensibility Hvak Common Library	パージョン 10.0.0 17.0.0 8.0.0 17.0.0 9.0.0 17.0.0 9.0.0 7.0.3300.0 1000	•	快来 (Lul+E) 名前: Micro Focus Runtime (Interop RuntimeServices) 作成者: Micro Focus パージョン: 4.0.0 ファイル パージョン: 10000.00000.82
Microsoft.Data.Tools.Com.Interop 17.0.0.0		 Micro Focus Runtime Micro Focus Runtime (Azure OSSP) Micro Focus Runtime (Azure Tracing) Micro Focus Runtime (Core Tracing) Micro Focus Runtime (Interop RuntimeServices) Micro Focus XIVIL Extensions Assembly Micro Focus XIVIL Extensions Assembly Micro Focus XIVIL Extensions Interop Assembly Microsoft Azure Common Library Microsoft Azure Resource Management Library Microsoft Azure SQL Server Management Library Microsoft.AspNet.Scaffolding.12.0 Microsoft.Bcl.AsynChterfaces Microsoft.Bcl.HashCode Microsoft.Data.Entity,Design.Extensibility Microsoft.Data.Tools.Com.Interop 	4.0.0.0 4.0.0.0 4.0.0.0 4.0.0.0 9.00.0 4.0.0.0 7.0.0.0 2.0.0.0 2.0.0.0 0.9.0.0 17.0.0.0 17.0.0.0 17.0.0.0 15.0.0.0 15.0.0.0 17.0.0.0	•	

9) [プロジェクト]を選択し、「BookClassLibrary」にチェックを行った後、[OK]をクリックします。

▶ アセンブリ			検索 (Ctrl+E)
 ✓ プロジェクト ソリューション 	名前 ✓ BookClassLibrary	パス C:¥Users¥tarot¥source	名前: BookClassLibrary
▶ 共有プロジェクト	NativeCOBOL	C.#Osers#tarot#source	
▶ COM			
▶ 参照			
		参照(B)	OK キャンセル

10) チュートリアルファイル解凍フォルダー¥WebService 配下の Models と Controllers フォルダーを RESTfulWS プロ ジェクトにドラッグアンドドロップします。





11) BooksController.cs をダブルクリックして、COBOL 呼出し部の確認を行います。

.NET COBOL 機能により、従来の COBOL プログラムから PROGRAM-ID 名でクラス、および、メソッドが自動作成 されます。今回の BOOK.cbl は PROGRAM-ID が "BOOK" であるため、BOOK クラスとなります。また、前手順で 設定した ILSMARTLINKAGE コンパイラ指令により、LINKAGE SECTION に指定されていた LNK-FUNCTION, LNK-FILE-STATUS, LNK-B-DETAILS に対応するラッパークラスも合わせて作成されています。このラッパークラスを利 用することで、 .NET 言語と COBOL のデータ型の差異を意識することなく、一般的な C# のコーディングで記述でき ていることが確認できます。



BOOK book = new BOOK();	
LnkFunction lnkFunction = new LnkFunction();	
LnkFileStatus InkFileStatus = new LnkFileStatus();	
LnkBDetails lnkBDetails = new LnkBDetails();	
RunUnit runUnit = new RunUnit();	
runUnit.Add(book);	
runUnit.Add(lnkFunction);	
runUnit.Add(InkFileStatus);	
runUnit.Add(InkBDetails);	
InkFunction.LnkFunction = "1";	
lnkBDetails.LnkBStockno = id;	
book.BOOK(InkFunction, InkBDetails, InkFileStatus);	

補足)

多くの COBOL プログラムは、一般的にシングルスレッドでの動作を前提としています。しかし、Web アプリケーションは一 般的にマルチスレッドのため、WORKING-STORAGE SECTION のようなプロセス共有資源のスレッド間衝突による問 題が発生する可能性があります。 MicroFocus.COBOL.RuntimeServices.RunUnit は、これらの共有資源を各ス レッドで独立して利用できるようにします。

- 12) アプリケーションの設定ファイルを修正します。RestfulWS プロジェクト配下の Web.config をダブルクリックします。
- 13) 以下の変更を行った後、保存します。

/configuration/appSettings 配下に、以下を追加

/configuration/system.webServer 配下の以下の行をコメントアウト。

<remove name="OPTIONSVerbHandler"/>





/configuration/system.webServer 配下に、以下を追加。





14) ソリューションエクスプローラーの RESTfulWS プロジェクト名を選択し、マウスの右クリックにてコンテクストメニューを表示した上で、[スタートアッププロジェクトに設定(A)] を選択します。



ソリューション エクスプローラー		Ľа-(W)
		分析とコードのクリーンアップ(Z)
ソリューション エクスプローラー の検索	¢	発行(B)
😡 ソリューション 'dotNETCOBOL		アップグレード
BookClassLibrary	* ?	Application Insights の構成(C)
🔑 Properties		概要(V)
▶ 마□参照設定	Ξ	すべての子孫を折りたたむ
BOOK.cbl		ここでスコープ指定する(S)
NativeCOBOL	°-	新しいソリューション エクスプローラーのビュー(N)
RESTfulWS		
Connected Services		L/ルトの1公子(例(#(B)
Properties		追加(D)
▶ 887 参照		NuGet パッケージの管理(N)…
🛅 App_Data	Ð	クライアント側のライブラリを管理する(M)
App_Start		ユーザー シークレットの管理(G)
Areas	563	スタートアッププロジェクトの構成
Content	~~~~	フカートマップゴロジェクトレージ・ウトロン
Controllers		スツートアップ フロンエクトに設定(A)

15) [デバッグ(D)] > [デバッグの開始(S)] を選択し、デバッグを行います。

デバッグ(D)	テスト(S)	分析(N)	ツール(T)	拡張機能(X)
ウィンド	ל(W)			•
▶ デバッグ	の開始(S)			F5

以下のようなダイアログが表示された場合は、「次回から表示しない」にチェックを行い、[はい]をクリックします。

Microsof	ft Visual Studio		×
0	このプロジェクトは SSL を使用するように構成さ るには、IIS Express が生成した自己署名証明	れています。ブラウザー 月書を信頼することをう	での SSL の警告を避け 選択します。
	IIS Express の SSL 証明書を信頼しますか? 詳細情報		
	☑ 次回から表示しない	はい	เงเงิ
さらに、	[はい(Y)] をクリックします。		
セキュリテ	F1警告		>
1	発行者が次であると主張する証明機関 (CA) しています:	から証明書をインスト	ールしようと
	localhost		
	証明書が実際に "localhost" からのものである "localhost" に連絡して発行者を確認する必引 程で役立ちます:	かどうかを検証できま: 要があります。次の番号	せん。 号はこの過
	拇印 (sha1): DEA597EA 53F1CDE6 FFDA	C877 9CA5B190 21	8C676C
	警告: このルート証明書をインストールすると、この CA 動的に信頼されます。確認されていない拇印付 とは、セキュリティ上、危険です。[はい] をクリッ になります。	によって発行された証 すきの証明書をインスト クすると、この危険を認	明書は自 ールするこ 8職したこと
	この証明書をインストールしますか?		
	Γ	(はい(Y)	L\L\え(N)

© Rocket Software, Inc. or its affiliates 1990–2024. All rights reserved. Rocket and the Rocket Software logos are registered trademarks of Rocket Software, Inc. Other product and service names might be trademarks of Rocket Software or its affiliates.



16) ブラウザーが起動します。

https://localhost:XXXXX/api/Books/1111 にアクセスします。 XXXXX にはポート番号を入力してください。ポート番号は、起動時の URL から確認できます。

上記のように、ストック番号: 1111の情報が戻されることを確認して、ブラウザーをクローズしてください。



This XML file does not appear to have any style information associated with it. The document tree is shown below.



3.5 IIS サーバーへのサービス配置

1) RESTfulWS プロジェクト名を選択し、マウスの右クリックにてコンテクストメニューを表示した上で、[発行(B)] を選択しま



2) 「フォルダー」を選択し、[次へ(N)]をクリックします。



公開 現在公開している場所		
ターゲット		Azure アプリケーションを Microsoft クラウドにホストします
	.	Docker Container Registry アプリケーションを Docker イメージに対応したサポートされている Container Registry に発行します
		フォルダ ー アプリケーションをローカル フォルダーまたはファイル共有に発行します
	\$	FTP/FTPS サーパー アプリケーションを FTP/FTPS サーバーに発行します
		Web サーパー (IIS) Web 配置または Web 配置パッケージを使用してアプリケーションを IIS に発行します
	\hookrightarrow	プロファイルのインボート アプリを配置するための発行設定をインボートします
		戻る(B) 次へ(N) 完了(F) キャンセル(C)

- 3) そのまま [完了(F)] をクリックします。
 - 公開

ローカルまたはネットワーク フォルダーへのパスを指定してください

ターゲット	フォルダーの場所
2 2 2 1	bin¥app.publish¥ 参照(R)
場所	 ローカル フォルダーの場合は、プロジェクトへの完全パスまたは相対パスのいずれかを指定できます。例: publish¥(相対パス) C:¥Users¥Username¥Documents(完全パス) ネットワークフォルダーの場合、¥¥を使用してから、コンピューター名または IP アドレスのいずれかを使用する必要があります。例: ¥¥server1¥fileshare1 ¥¥192.168.1.17¥fileshare1
	戻る(B) 次へ(N) 完了(F) キャンセル(C)

今回は出力先のフォルダーを変更していないため、プロジェクトが保存されたフォルダー配下が出力先となります。

4) そのまま [閉じる] をクリックします。



ターゲット	● 発行プロファイル 'C:¥Users¥tarot¥source¥repos¥dot ¥FolderProfile.pubxml' が作成されました。	NETCOBOLREST¥RESTfulWS¥	Properties¥Publi	shProfiles
場所				
完了				
	成功時に自動的に閉じる			
		戻る(B) 次へ(N)	閉じる	キャンセル(C)

5) [発行(U)] をクリックします。

実行プロファイル作成の進行状況

ller.cs BookCla	assLibrary	NativeCOBOL	
FolderProfile.pubx	ml 👻		🔩 発行(U)
近しいプロファイル その	他のアクション 🗸		
公開準備が完了しました	0		
定 - ゲットの場所 存のファイルを削除 成 ペイののマロチェモニー	bin¥a false Relea	pp.publish¥ [] Ø se Ø	
	FolderProfile.pubxr フォルダー 新しいプロファイル その 公開準備が完了しました 定 -グットの場所 存のファイルを削除 成 べての設定を表示	FolderProfile.pubxml ・ フォルダー 新しいプロファイル その他のアクション ・ 公開準備が完了しました。 定 -ゲットの場所 bin¥a 済のファイルを削除 false 成 Relea べての設定を表示	FolderProfile.pubxml ・ フォルダー ボレいプロファイル その他のアクション・ 公開準備が完了しました。 定 -ゲットの場所 bin¥app.publish¥ ① 店のファイルを削除 false 成 Release べての設定を表示

完了するとメッセージが表示されます。

RESTfulWS: 公開 ♀ × Web.config	BooksController.cs	BookClassLibrary	NativeCOBOL	.
概要 接続済みサービス	FolderPr フォルダー	ofile.pubxml 👻		€ ţ 発行(U)
公開	十 新しいプロファイ	ル その他のアクション	•	
	✓ 2024/08/29 移動	の 15:18 に公開が成功し	Jaluta .	

6) IIS サーバーに公開するため、以下のフォルダーを作成したうえで、前手順で出力されたリソース (bin¥app.publish 配下のフォルダー・ファイル) をコピーしてください。

C:¥dotNETManageTutorial¥aspnetweb



C □ > ··· □-カル ディスク (C:) > dotNETManageTu	ıtorial > aspnetw	eb asj
, 0 🗋 \land 🖻 🛈	↑↓ 並べ替え ~ 🛛 🗮 🗧	表示 ~ •••	
名前	更新日時	種類	サイズ
Areas	2024/08/29 15:21	ファイル フォルダー	
📒 bin	2024/08/29 15:21	ファイル フォルダー	
Content	2024/08/29 15:21	ファイル フォルダー	
Scripts	2024/08/29 15:21	ファイル フォルダー	
Views	2024/08/29 15:21	ファイル フォルダー	
📙 favicon.ico	2024/08/29 14:22	ICO ファイル	32 KB
🚛 Global.asax	2024/08/29 14:22	ASP.NET Server Ap	1 KB
Web.config	2024/08/29 15:18	CONFIG ファイル	5 KB

7) Windows のスタートメニューの上で、右クリックにてコンテクストメニューを表示し、[コンピューターの管理]を選択します。

コンピューターの管理
ターミナル
ターミナル (管理者)
タスク マネージャー
設定
エクスプローラー
検索
ファイル名を指定して実行
シャットダウンまたはサインアウト >
デスクトップ
📕 Q 🧮

8) [サービスとアプリケーション] 配下の [インターネットインフォメーションサービス (IIS) マネージャー] をクリックします。 本項目が表示されていない方は、IIS がインストールされていません。4.1 の手順にてインストールを行ってください。

🌆 コンピューターの管理	
ファイル(F) 操作(A) 表示(V) ヘルプ(H)	
▲ コンピューターの管理 (ローカル)	名前
~ 龍 システム ツール	1 12 システム ツール
> 🕑 タスク スケジューラ	■記信域
> 🚺 イベント ビューアー	□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
> 🔞 共有フォルダー	
> 🌆 ローカル ユーザーとグループ	
> 🔞 パフォーマンス	
📇 デバイス マネージャー	
~ 🔚 記憶域	
📅 ディスクの管理	
~ ビスとアプリケーション	
› 💐 インターネット インフォメーション サービス (IIS) マネージャー	
😘 サービス	
🗃 WMI או-ם-אכב	



9) 本チュートリアル用に新規サイトを作成するため、[サイト] > [Default Web Site] を選択し、マウスの右クリックでコンテ クストメニューを開き、[停止] をクリックします。

ファイル(F) 操作(A) 表示(V) ^	ヽルプ(H)					
 ファビューターの管理(ローカル) コアビューターの管理(ローカル) ジステム ツール ジステム ツール ワスク スケジューラ オペント ビューアー オペント ビューアー オペント ビューアー オペント ビューアー マスト スケックル デバイス マネージルー ご 地域 ディス2の管理 サービスとアブリケーション オットビス マスト マスト<th>ルプ(H)</th><th>マリン サイト フィルクー: 名前 ● Default Web Site</th><th>ID 1</th><th> ▼ 検索(G) - 回すべて表示(A) 秋態 パインド Web サイトの規定値の設定 バインド 基本設定 エクスプローラ- アクセス計可の編集 削除 名前の変更 アプリケーションの表示 仮想ディレクトリの表示 </th><th> グループ化: グループ化な</th><th>ະບ /ໄຊ %Systen</th>	ルプ(H)	マリン サイト フィルクー: 名前 ● Default Web Site	ID 1	 ▼ 検索(G) - 回すべて表示(A) 秋態 パインド Web サイトの規定値の設定 バインド 基本設定 エクスプローラ- アクセス計可の編集 削除 名前の変更 アプリケーションの表示 仮想ディレクトリの表示 	グループ化: グループ化な	ະບ /ໄຊ %Systen
			C	Web サイトの管理 ヘルプ	 ▶ ♥ ● ●	

10) [サイト] を選択し、マウスの右クリックにてコンテクストメニューを表示した上で、[Web サイトの追加] をクリックします。

 ▲ コンピューターの管理(ローカル) ◇ (2) システム ツール ◇ (2) タスクスケジューラ 	← → ○ → WI 接続	N11-V-NA 🕨 サイト 🕨	
 ▲ イベントビューアー 秋石「フオルダー ▲ ローカル ユーザーとグループ ベントマンス デビイフ マスニジャー 	 ✓ ✓	win11-v-na¥tarot) ヨンブール Web サイトの:まれ	-
 → 記憶域 ディスクの管理 → ピスとアプリケーション 			
 > (1) インターネット インフォメーシ ○ サービス (1) サービス (2) WMI コントロール 	4	コノテノッ ビューに切り替え	

11) 以下の入力を行い、[OK] をクリックします。

ファイル(F) 操作(A) 表示(V) ヘルプ(H)

サイト名: "tut"

物理パス: "C:¥dotNETManageTutorial¥aspnetweb"

Web サイトを直ちに開始する: チェックをはずす



Web サイトの追加	? ×
サイト名(S): アプリケーション ブール(L):	
tut	選択(E)
コンテンツ ディレクトリ	
物理パス(P):	
C:¥dotNETManageTutorial¥aspnetweb	
ハススルー認証	
接続(C) テスト設定(G)	
パインド	
種類(T): IP アドレス(I): ポート(O):	
http 、 未使用の IP アドレスすべて 80	
ホスト名(H):	
例: www.contoso.com または marketing.contoso.com	
□ Web サイトを直ちに開始する(M)]	
Web サイトを直ちに開始する(M))	

×

以下のダイアログには、[はい(Y)] をクリックします。



Web サイトの追加

バインド **80: は別のサイトに割り当てられています。このサイトに同じパイン ドを割り当てると、いずれか一方のサイトしか開始できなくなります。重複す るバインドを追加しますか?

	はい(Y)	いいえ(N)
--	-------	--------



3.6 RESTful Web サービスの確認

1) 前手順で使用した "インターネット インフォメーションサービス (IIS) マネージャー" に戻り、"tut" サイトを選択し、画面 右側より [開始] をクリックします。

ファイル(F) 操作(A) 表示(V) ヘルプ(H)

🗢 🏟 🖄 📰 👔			
 ■ コンビューターの管理(ローカル) ● ジステム ツール ● ジステム ツール ● ジステム ツール ● ジステム ツール ● ジステム フール ● ジステム ツール ● ジステム ツール ● ローカル ユーザーとグループ ● ローカル ユーザーとグループ ● ローカル ユーザーとグループ ● バス マネージャー ■ ディスクの管理 ■ サービスとアブリケーション > ● サービス ● サービス ● サービス ● サービス ● サービス ● サービス ● サービス 	そ→ ⑥ ト W 接続 2 ✓- ● WIN11-V-NA ✓- ● WIN11-V-NA ✓- ● サイト ✓- ● サイト ✓- ● Defat ✓- ● tut	//N11-V-NA → サイト → tut → (win11-v-na¥tarot) ション ブール ult Web Site	tut ホール 7/ルター: ASRNET ダー NET グローパリゼーシ ヨン NET ユーザー
		Web サイトの管理 ト	⑦ 重記動
		☑ 最新の情報に更新(R)	▶ 開始
		★ 削除	■ 停止

2) Web ブラウザーを開き、以下の URL にアクセスします。

http://localhost/api/Books/1111

さきほどのデバッグ時と同様、書籍情報が JSON 形式で戻されているため、C# で作成された RESTful Web サービス が COBOL プログラムを正しく呼び出していることが分かります。



3) チュートリアルファイル解凍フォルダー配下の WebClient フォルダー配下の Search.html を Web ブラウザーで開いてく ださい。



こちらは、今回作成した RESTful Web サービスを呼び出す Web 画面となります。さきほど確認したように REST API を介して JSON 形式でデータの送受信を行うモダンなシステムインターフェースとなっています。

例えば、"1111" を入力して、[検索] をクリックすると、以下のように REST API から戻されたデータが表示されます。

書籍情報の検索 ストック番号を入力の上、 (検索)ボタンを押下してください。					
1111	検索				
ストック番号	1111				
タイトル	LOAD OF THE RING				
ジャンル	FANTASY				
著者	TOLKIEN				
小売価格	1500				
仕入れ数	2000				
売り上げ数	1000				

4) "インターネット インフォメーションサービス (IIS) マネージャー" に戻り、"tut" サイトを選択し、画面右側より [停止] を クリックします。

ファイル(F) 操作(A) 表示(V) ^	.ルプ(H)		
🗢 🄿 🙍 📰 👔			
 ヨンピューターの管理(ローカル) ジステムツール タスクスケジューラ 	← → ● → WI 接続	N11-V-NA 🕨 サイト 🕨 tut 🕨	
> 🛃 イベント ビューアー	21		- 🔮 tut 赤一.
 ・ ・	 ~ ♥WIN11-V-NA (win11-v-na¥tarot) □ □ ⑦ ⑦ ⑦ ⑦ ⑦ ○ ○<!--</td--><td>フィルター: ASPNET</td>		フィルター: ASPNET
 ごに、み ディスクの管理 サービスとアプリケーション マ インターネット インフォメーシ 	>- 😂 tut	エクスプローラー アクセス許可の編集…	.NET グローバリゼーション
③ サービス ▲ WMI שורם שורם שורם		 アプリケーションの追加 仮想ディレクトリの追加 バインドの編集 	
		Web サイトの管理 ト	
	6	2 最新の情報に更新(R)	▶ 開始
	2	< 削除	■ 停止



4 補足

4.1 IIS サーバーのインストール方法

- 1) Windows でコントロールパネルを開きます。
- 2) [プログラム] をクリックします。

コンピューターの設定を調整します 表 システムとセキュリティ ユーザー アカウント コンピューターの状態を確認 🗣 アカウントの種類の変更 ファイル履歴でファイルのバックアップ コピーを保存 バックアップと復元 (Windows 7) デスクトップのカスタマイズ ネットワークとインターネット ネットワークの状態とタスクの表示 時計と地域 日付、時刻、数値形式の変更 ハードウェアとサウンド デバイスとプリンターの表示 コンピューターの簡単操作 デバイスの追加 設定の提案の表示 共通で使うモビリティ設定の調整 視覚ディスプレイの最適化 プログラム プログラムのアンインストール

- 3) [Windows の機能の有効化または無効化] をクリックします。
 - プログラムと機能 プログラムのアンインストール | ♥ Windows の機能の有効化または無効化 インストールされた更新プログラムを表示 | 以前のパージョンの Windows 用に作成されたプログラムの実行 | プログラムのインストール方法 既定のプログラム
 - ➡️ メディアまたはデバイスの既定設定の変更
- 4) 以下のチェックを行い、[OK]をクリックします。
 - 「インターネット インフォメーションサービス」 配下の「Web 管理ツール」
 - 「インターネット インフォメーションサービス」配下の「World Wide Web サービス」
 - 「インターネット インフォメーションサービス」>「World Wide Web サービス」>「アプリケーション開発機能」配下の「ASP.NET 4.8」





自動的にチェックされているものはそのままで構いません。

インストール完了後、[閉じる] をクリックします。

免責事項

ここで紹介したソースコードは、機能説明のためのサンプルであり、製品の一部ではございません。ソースコードが実際に動作するか、御社業務に適合するかなどに関しまして、一切の保証はございません。 ソースコード、説明、その他すべてについて、無謬性は保障されません。 ここで紹介するソースコードの一部、もしくは全部について、弊社に断りなく、御社の内部に組み込み、そのままご利用頂いても構いません。 本ソースコードの一部もしくは全部を二次的著作物に対して引用する場合、著作権法の精神に基づき、適切な扱いを行ってください。