Micro Focus Visual COBOL チュートリアル

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. 前提条件

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

OS

Windows 11 Enterprise Edition (64bit)

COBOL 開発環境製品 Micro Focus Visual COBOL 8.0J for Visual Studio

本資料では Visual Studio 2022 を使用しています。Visual Studio のバージョンにより、設定画面への遷移方法やレイア ウトなどが異なる場合があります。

また、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

MICRO[®] FOCUS

内容

- 1. 目的
- 2. 前提条件
- 3. チュートリアル手順の概要
 - 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の概要

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

3)

- 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。

Visual Studio 2022	
最近開いた項目(R)	開始する
Visual Studio を使用するとき、ユーザーが開くプロジェクト、フォルダー、ファイルはここに表示されるので、すばやく アクセスできます。 頻繁に開く項目は、ピン留めして常に一覧の先頭に表示することができます。	↓ リポジトリのクローン(C) GitHub や Azure DevOps などのオンライン リボジトリ からコードを取得します
	プロジェクトやソリューションを開く(P) ローカルの Visual Studio プロジェクトまたは.sln ファイル を開きます
	ビーカル フォルダーを開く(F) 任意のフォルダー内のコードに移動して編集します
	新しいプロジェクトの作成(N) 開始するには、コードスキャフォールディング付きのプロジェ クト テンプルートを選択します
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	
COBOL • Windows • בעיב	JL ~
コンソール アプリケーション (.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。 COBOL Windows コンソール	
こンソール 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プロジェクトです。 COBOL Windows ネーティブ コンソール	
コンソール アプリ (.NET) (Micro Focus) Windows および Linux 上の .NET で実行できるコマンドライン アプリケーション プロジェクトです。	を作成するための
COBOL Common コンソール Linux Windows	

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

WSDL/JSON 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

COBOL Windows Enterprise Server Native コンソール

めのプロジェクトです。

WSDL または JSON 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

4) 「プロジェクト名」に "ManageCOBOLTutorial" を入力し、[作成(C)] をクリックします。

戻る(B)

次へ(N)



新しいプロジェクトを構成します
コンソール アプリケーション COBOL Windows ネーティブ コンソール
プロジェクト名(J)
ManageCOBOLTutorial
場所(L)
C:¥work¥ •
ソリューション名(M) 🚯
ManageCOBOLTutorial
─ ソリュー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)

5) 自動作成された Program1.cbl は不要のため、ソリューションエクスプローラーより、「Program1.cbl」を選択し、マウスの 右クリックからコンテクストメニューを表示した上で、[削除(D)] を選択します。



- 6) チュートリアルファイル解凍フォルダ¥COBOL フォルダ配下の、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配下の DAT フォルダを任意のフォルダにコピーします。フォルダ配下には、書籍情報を管理 する索引ファイルが保存されています。なお、本チュートリアルでは、C:¥ 直下としています。



3.2.2. アプリケーションの動作確認

1) ManageCOBOLTutorial 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。



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)] をクリックしま す。

ManageCOBOLTutorial* 🕂 🗙	
アプリケーション	
SQL	
コピーブック	
プリプロセッサ	出刀の名前: Manage COBOLT the int
COBOL	ManageCoboLittonal
COBOL リンク	
デバッグ	
Micro Focus Code Analysis	O 複数の実行可能ファイル BOOKSCRN
	lbr にパッケージ化 環境(E)

- 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:¥DAT¥BOOKINFO.DAT"

IDE 設定					?	Х
	変数	値				
	BOOKINFO	C:¥DAT¥	BOOKINFO.DA	T	追加	
					削除	
プロジ:	ェクトのビルド時に変数を使用する	5(U)				
			OK(O)		キャンセル(C)

- 4) 追加されたことを確認の上、[OK(O)] をクリックします。
- 5) 「COBOL」タブを選択し、「追加指令」欄に "ASSIGN(EXTERNAL)" を入力します。
- 6) [ファイル(F)] > [すべて保存(L)] で変更を保存します。



7) [デバッグ(D)] > [デバッグの開始(S)] をクリックします。

Lr		(-).			
デル	、ッグ(D) テスト(S) 分析	f(N) ツール(T)	拡張機能(X)	ウィント	
	ウィンドウ(W)			+	
►	デバッグの開始(S)		F5		
₽	テバックなしで開始(H)		Ctrl+F5		
	ManageCOBOLTutorial			-	
	FUNCTION: [_] READ=1,	ADD=2, DELETE:	=3 NEXT=4, GRO)SSSALES=	S, QUIT=9
	ストック番号: [
	タイトル: [
	ジャンル: [
	著者: [
	小売価格: [0] 売上: [000000000]	仕入れ: [0]		
	STATUS: []				

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"

FUNCTION: [1] READ=1, ADD=2, DELETE=3 NEXT=4, GROSSSALES=S,	QUIT=9
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: [2000] 売上: [000001000]	
STATUS: [00]	

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: "FANTASY"
- 著者: "宮部みゆき"
- 小売価格: "3000"
- 仕入れ: "1000"
- 売上: "900"



FUNCTION: [2] READ=1, ADD=2, DELETE=3 NEXT=4, GROSSSALES=S,	QUIT=9
ストック番号: [4444]	
タイトル: [ブレイブストーリー]
ジャンル: [FANTASY]	
著者: [宮部みゆき]	
小売価格: [3000] 仕入れ: [1000] 売上: [000000900]	
 STATUS: [00]	

実行後、READ 機能を用いて、追加した情報が参照できていることを確認してください。 確認した後、以下の入力後、Enter キーを打鍵することで、情報を削除します。 FUNCTION: "3"

ストック番号: "4444"



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。



確認後、FUNCTION=9 でアプリケーションを終了してください。

 後の手順で新たなプロジェクトを作成するため、今回のプロジェクト名を変更します。ソリューションエクスプローラーより、 ManageCOBOLTutorial 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、[名前 の変更(M)]をクリックし、"NativeCOBOL" とプロジェクト名を変更します。

ţţţ	スタートアップ プロジェクトに設定(A) デバッグ(G)	,	•
Q,	プロジェクト詳細		
χ	切り取り(T)	Ctrl+X	
Ĝ	貼り付け(P)	Ctrl+V	
X	削除(V)	Del	
Ē	名前の変更(M)	F2	
			F



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、"ManageCOBOLTutorial" 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。

		コード メトリックスの計算(C)	
新しいプロジェクト(N)		追加(D)	•
既存のプロジェクト(E)	G,	プロジェクト詳細	
既存の Web サイト(B)	ta	ソリューションをソース管理に追加(D)	

2) 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

CBL	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。	l
	COBOL Windows ライブラリ	l
	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	l
	ユニット テスト ライブラリ 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	
2 日 日 日	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。	Ŧ
	次へ(N)	

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。

新しいプロジェクトを構成します	
クラス ライブラリ (.NET Framework) COBOL Window	ws ライブラリ
プロジェクト名(J)	
BookClassLibrary	
- 場所(L)	
C:¥work¥ManageCOBOLTutorial	•
フレームワーク(F)	
.NET Framework 4.7.2	•

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。

Ж	切り取り(T)	Ctrl+X
ŋ	⊐ピ−(Y)	Ctrl+C
×	削除(D)	Del
X	名前の変更(M)	

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。

クリックしまり。

Q,	プロジェクト詳細	
ж	切り取り(T)	Ctrl+X
â	貼り付け(P)	Ctrl+V
×	削除(V)	Del
χ	名前の変更(M)	

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。

X	名前の変更(M)	
	プロジェクトのアンロード(L)	
6	エクスプローラーでフォルダーを開く(X)	
۶	プロパティ(R)	Alt+Enter

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。

アプリケーション	構成(C): アクティブな (Debug))	~	
SQL デバッグ	プラットフォーム(M): アクティブな	t (Any CPU)	~	
コピーブック	- UV 114.	HR. 110.071 CLV0/	v 1/v Lj	
名前空間	最大エラー数:	100	□ 警告をエラーとして処理	
リソース	出力			
設定	出力パス:	.¥bin¥Debug¥		参照
	□ 指令ファイルの生成		□ リストファイルを生成	
	Smart Linkage			
	✓ .NET コードに集団のリン	ンケージ項目を公開	オプション	
	追加指令 —			
	ASSIGN(EXTERNAL)]		
				Ŧ

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。

?ブリケーション	構成(C): Release		~	
QL Struct	プラットフォーム(M): アクティ	イブな (Any CPU)	~	
「ハック]ピーブック		HIK-300042 C	LIND(V NY L)	
名前空間	最大エラー数:	100	□ 警告をエラーとして処理	
OBOL				
リソース	出力 ———			
设定	出力パス:	.¥bin¥Release¥		参照
	□ 指令ファイルの生症	龙	□ リストファイルを生成	
	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)		

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。

 ManageCOBOLTutorial 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)] を選択します。

	新しいプロジェクト(N)	
	既存のプロジェクト(E) 既存の Web サイト(B)	
*	新しい項目(W)	Ctrl+Shift+A
t] *-	既存の項目(G) 新しいソリューション フォルダー(D)	Shift+Alt+A

2) 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

—	ASP.NET Web アプリケーション (.NET Framework) ASP.NET 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。ASP.NET Web Forms、 MVC、または Web API のアプリケーションを作成し、ASP.NET 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 る単体テストを含むプロジェクトです。				
	C# Windows Web テスト			
	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。

新しいプロジェクトを構成します	
ASP.NET Web アプリケーション (.NET Framework)	C# Windows クラウド Web
プロジェクト名(J)	
RESTfulWS	
場所(L)	_
C:¥work¥ManageCOBOLTutorial	•
フレームワーク(F)	
.NET Framework 4.7.2	•

4) 表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。

新しい ASP.NET Web アプリケーションを作成する

 4	空 ASPINET 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。このテンプルートには内容はありません。	認証 なし ・
	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 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 ドロップのイベント駆動モデルを使用して、動物な Web サイトを構築できます。プザイン画面および数百のコントロールとコンボーネントを利用して、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。 MVC ASPNET MVC アブ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 デャを使用してアブリケーションを構成できます。ASPNET MVC には、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 するアプリケーションを構成できための多くの機能が備わっています。	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)
		我知识宁
{ _	Web API ブラウザーやモバイル 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 テンプルート。	■十初期 BX AC ✓ HTTPS 用の構成(C) Docker のサポート
	Web API プラウザーやモバイル 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 アンカート。 シングルレベージ アプリケーション ASPNET Web API を使用してリッチ クライアント (例の JavaScript 駆動型 HTML5 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, Single Page Application では HTML5, CSS3, および JavaScript を使用したクライアント 倒インタラクションが含まれるリッチ ユーザー エクスペリエンスを提供します。	P+F400ス/LC ↓ HTTPS 用の構成(C) ↓ Docker のサポート (Docker のサポート (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) RESTfulWS.Tests

5) 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、[ビルドの依存関係(B)]
 > [プロジェクトの依存関係(S)] を選択します。

プロジェクトの依存関係(S)	ビルドの依存関係(B)	•
プロジェクトのビルド順序(I)	追加(D)	•



6) 「依存関係」 タブより、「BookClassLibrary」 にチェックを行った後、[OK] をクリックします。

プロジェクトの	D依存関係			?	×
依存関係	ビルドの順序				
プロジェクト	~(R):				
RESTfulV	VS				\sim
依存先(D):	_			
🖌 Book	ClassLibrary				
	recobol				
			01	بحرط	te u
			OK	++2	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。
 RESTfulWS
 Connected Services
 Properties
- 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。

参照マネージャー - RESTfulWS						?	×
アセンブリ	ターゲット:	.NET Framework 4.7.2			検索 (Ctrl+E)		ρ-
プレームワーク <u>拡張</u> 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	Y	名前 ADODB CppCodeProvider envdte EnvDTE envdte100 EnvDTE100 envdte80 EnvDTE90 envdte90 EnvDTE90 envdte90 EnvDTE90 EnvDTE90a EventSource Extensibility Json.NET.AD MdsAnalyzer Micro Focus Runtime Micro Focus Runtime Micro Focus Runtime Micro Focus Runtime (Core Tracing) Micro Focus Runtime (Core Tracing) Micro Focus Runtime (Core Tracing)	パージョン 7.0.3300.0 10.0.0 17.0.0 17.0.0 10.0.0 17.0.0 8.0.0 17.0.0 9.0.0 17.0.0 9.0.0 2.7.00 7.0.3300.0 6.0.0 33.1.00 33.1.00 3.8.00 3.8.00 4.0.00	Î	名前: Micro Focus Runtir 作成者: Micro Focus バージョン: 4.0.0.0 ファイル バージョン: 08000.00001.0.63	ne	
		Micro Focus Kuntime (Interop RuntimeServices) Micro Focus XML Extensions Assembly Micro Focus XML Extensions Assembly Micro Focus XML Extensions Interop Assembly MicroFocus.COBOL.MFUNIT	4.0.0.0 5.0.0.0 4.0.0.0 4.0.0.0 1.0.0.12	•			
			1	診照(B)	ОК	キャン	セル
補足)							
上記のように、複数	なのバ-	-ジョンがリストアップされている場合	合は最新	īバー	-ジョンを選択	して	ください。



9) 「プロジェクト」タブを選択し、「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		[
▲ プロジェクト	名前	パス
Alle See	BookClassLibrary	C:¥work¥ManageCOB
ソリューション	NativeCOBOL	C:¥work¥ManageCOB

10) チュートリアルファイル解凍フォルダ¥WebService¥Models フォルダ配下の2ファイルを、RESTfulWS プロジェクト配下の

Model フォルダにドラッグアンドドロップします。



11) チュートリアルファイル解凍フォルダ¥WebService¥Controllers フォルダ配下の2ファイルを RESTfulWS プロジェクト配

下の Controllers フォルダにドラッグアンドドロップします。

4	🗊 RESTfulWS
	🐵 Connected Services
⊳	🎤 Properties
⊳	88 参照
	🛅 App_Data
⊳	App_Start
⊳	🗖 Areas
Þ	Content
-	Controllers
	C# BookSalesController.cs
	C# BooksController.cs
	V C+ HomeController.cs
	C# ValuesController.cs



12) 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。

publ	ic IHttpActionResult Get([FromUri]string id)
1	System.Environment.SetEnvironmentVariable("BOOKINFO", System.Configuration.ConfigurationManager.AppSettings.Get("BOOKINFO"));
	BOOK book = new BOOK(); LnkFunction InkFunction = new LnkFuleStatus(); LnkBDetails InkBDetails = new LnkBDetails();
	RunUnit runUnit = new RunUnit(); runUnit.Add(book); runUnit.Add(InkFinetion); runUnit.Add(InkFinetiatus); InkFunction_LnkFunction = ~1~; InkFunction_LnkFunction = id; book.800K(InkFunction, InkBDetails, InkFileStatus);
	BookResponse bookResponse = new BookResponse(); BookInfo outBookInfo = new BookInfo(); outBookInfo:Infe = InkBDetails-LnkBTyne; outBookInfo.StockNo = InkBDetails-LnkBAuthor; outBookInfo.StockNo = InkBDetails-LnkBAuthor; outBookInfo.StockNo = InkBDetails-LnkBKtokno; outBookInfo.Sonhand = InkBDetails-LnkBKtokIn; outBookInfo.Sold = InkBDetails-LnkBGtoHand; outBookInfo.Sold = InkBDetails-LnkBGtoHand; bookResponse.bookInfo = outBookInfo; bookResponse.bookInfo = outBookInfo; bookResponse.fileStatus = ((int)InkFileStatus.LnkFileStatus.ElementAt(0)).ToString("%") + "/" + ((int)InkFileStatus.LnkFileStatus.ElementAt(1)).ToString("%"); runDnit.StopRun();
}	return Ok(bookResponse);

.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 InkFunction = new LnkFunction();
LnkFileStatus InkFileStatus = new LnkFileStatus();
LnkBDetails InkBDetails = new LnkBDetails();
RunUnit runUnit = new RunUnit();
runUnit.Add(book);
runUnit.Add(InkFunction);
runUnit.Add(lnkFileStatus);
runUnit.Add(InkBDetails);
InkFunction.LnkFunction = "1";
InkBDetails.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。

13) 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。



14) 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。



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



```
<system.webServer>
<handlers>
<remove name="ExtensionlessUrlHandler-Integrated-4.0" />
<!--
<remove name="OPTIONSVerbHandler" />
-->
<remove name="TRACEVerbHandler" />
<add name="ExtensionlessUrlHandler-Integrated-4.0" path="*."
</handlers>
<httpProtocol>
</customHeaders>
<add name="Access-Control-Allow-Origin" value="*" />
<add name="Access-Control-Allow-Methods" value="*" />
<add name="*" />
</add name="*
```



- 15) 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。



16) [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。

	テハック(D)	テスト(S)	分析(N)	ツール(1)	拡張機能()
	ウィンド	ウ(W)			•
Γ	デバッグ	の開始(S)		F5	
	▶ デバック	なしで開始(ト	H)	Ctrl+	-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 X					
0	このプロジェクトは SSL を使用するように構成されています。ブラウザーでの SSL の警告を避 けるには、IIS Express が生成した自己署名証明書を信頼することを選択します。					
	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					
	はいいえ					
さらに、	[はい(Y)] をクリックします。					

291C, [1	はい(Y)」 をクリックします。	
セキュリティ	警告	>
	発行者が次であると主張する証明機関 (CA) から証明書をインストールしようと しています:	
	localhost	
	証明書が実際に "localhost" からのものであるかどうかを検証できません。 "localhost" に連絡して発行者を確認する必要があります。次の番号はこの過 程で役立ちます:	
	拇印 (sha1): 013D99A5 3AF8A4CF 97A8E421 6DFFCBC0 A1805480	
	警告: このルート証明書をインストールすると、この CA によって発行された証明書は自 動的に信頼されます。確認されていない拇印付きの証明書をインストールするこ とは、セキュリティ上、危険です。[はい]をクリックすると、この危険を認識したこと になります。	
	この証明書をインストールしますか?	
	はい(Y) いいえ(N)	



17) ブラウザが起動します。

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。

\leftarrow	С

https://localhost:44370/api/Book × + https://localhost:44370/api/Books/1111

This XML file does not appear to have any style information as:

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。



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



3) [フォルダーの場所]はそのままにして[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



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配下が出力先となります。デフォルトでは、以下のフォルダになります。

C:¥任意の作業フォルダ¥ManageCOBOLTutorial¥RestFulWS

4) [閉じる] をクリックします。



実行プロファイル作成の進行状況

ターゲット	発行プロファイル 'C:¥work¥ManageCOBOLTutorial¥RESTfulWS¥Properties¥PublishProfiles ¥FolderProfile.pubxml' が作成されました。
場所	
完了	
	I
	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]をクリックすると公開が完了した旨のメッセージが表示されます。

RESTfulWS: 公開 → ×	Web.config BookClassLibrary	/	~ ©
概要	FolderProfile.pubxml マ フォルダー		€ 発行(U)
接続済みサ−ビス	上 虻田 るの地のマクション		
公開	〒初焼 その他のアクショク◆		
	① 公開準備が完了しました。		
	設定		
	ターゲットの場所	bin¥app.publish¥ 🗖	
	既存のファイルを削除	false Ø	
	傳 <i>成</i>	Release V	
	すべ(の設定を表示		
2022/09/1	6の21:00に公開が成功しまし	t_ Co	
フォルダーを	開く		

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

C:¥ManageCOBOLTutorialWS

Areas	2022/09/16 21:03	ファイル フォルダー	
📒 bin	2022/09/16 21:03	ファイル フォルダー	
Content	2022/09/16 21:03	ファイル フォルダー	
in fonts	2022/09/16 21:03	ファイル フォルダー	
Scripts	2022/09/16 21:03	ファイル フォルダー	
Views	2022/09/16 21:03	ファイル フォルダー	
Le favicon.ico	2022/09/16 16:48	アイコン	32 KB
🚛 Global.asax	2022/09/16 16:48	ASP.NET Server Appli	1 KB
🖓 Web.config	2022/09/16 21:00	XML Configuration Fi	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。



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。

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、画面右側より [停止] をク リックします。

 ▲ コンピューターの管理(ローカル) ◇ № システムツール 	← → ● ► DESKTOP-2S5E	KOA トサイトト De	fault Web Site 🕨						1	🗄 (0 -
 (1) シノオム ソール (2) タノオム ソール (3) タノオム ノーシュ (3) イベント ビューアー (3) オブオルグー (3) イベント ビューアー (5) イワイ マノス (7) イマノス (7) イワイ マノス (7) イレー (7) イロー (7) イレー (7) イレ (7) イレ<!--</td--><td>接続 DESKTOP-2558K0A (DESKTOP-2558 プアリアーションブール プアリアーションブール プロリーションブール Default Web Site</td><td></td><td>Web Site ホ ・ ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・</td><td>-ム ^{余奈(G)} - Go す SSL 設定 正稿</td><td>(て表示(A) グループ (点) エラーページ 民定のドキュメント</td><td>化: 領域 ディレクトリの参照 山力キャッシュ</td><td> ・ 回・ ・ ・<!--</td--><td>•</td><td> 提作</td><td></td></td>	接続 DESKTOP-2558K0A (DESKTOP-2558 プアリアーションブール プアリアーションブール プロリーションブール Default Web Site		Web Site ホ ・ ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	-ム ^{余奈(G)} - Go す SSL 設定 正稿	(て表示(A) グループ (点) エラーページ 民定のドキュメント	化: 領域 ディレクトリの参照 山力キャッシュ	 ・ 回・ ・ ・<!--</td--><td>•</td><td> 提作</td><td></td>	•	 提作	



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。



11) 以下の入力を行い、[OK] をクリックします。

サイト名: "tut"

物理パス: "C:¥ManageCOBOLTutorialWS"

Web サイトを直ちに開始する: チェックをはずす

Veb サイトの追加			?	>
サイト名(S):	アプリケーショ	ンプール(L):		
tut	tut		選択(E)	
コンテンツ ディレクトリー				
物理パス(P):				
C:¥ManageCOBOLTu	torialWS			
パススルー認証				
接続(C) う	スト設定(G)			
バインド				
種類(T):	IP アドレス(I):	ポート(0):		
http ~	未使用の IP アドレスすべて	~ 80		
ホスト名(H):				
] Web サイトを直ちに開	治する(M)			
		ОК	キャンセ	JL
、下のダイアログには	:、[はい(Y)] をクリックし	ます。		
eb サイトの追加		×		
パインド **:80: は短 ンドを割り当てると するパインドを追加	リのサイトに割り当てられています。この いずれか一方のサイトしか開始できな しますか?	サイトに同じパイ べなります。重複		
	はいの	เงเงิร์(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。

 ▲ コンピューターの管理(ローカル) ✓ № システムツール 	← → 6 → DESKTOP-2558K0A → サ/ト → tut →					
 ● 32,54,37-ル ● 32,74,37-ル ● 32,74,74,74 ■ 47,74,74 ● 47,74,74 ● 17,74,74 ● 17,74,74,74 ● 57,270,92 ● 57,270,92 ● 12,75,71,7-23 ● 12,72,71,7-23 ● 12,72,71,7-23 ● 12,72,71,7-23 ● 12,72,71,72,73 ● 12,72,71,72,73 ● 12,72,71,72,73 ● 12,72,71,72,73 ● 12,72,71,72,73 ● 12,72,71,72,73 	Image: state of the state	tut ホーム Tut ホー				
		コノビューフー ギー セッショノハル ノロハ1フー ハーンおよびコントロ 漫枕文子列 ール	● *:80 (http) 参照			

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

\leftarrow	\rightarrow	С	localhost/api/Books/1111
This	XML f	ile doe	s not appear to have any style information associat
▼ <book ▼ <book < < < <book < <book < <book < <book <book </book <book </book <book </book </br></book </book </book </book </book </book 	kRespon okInfo (Author) (Genre) (Onhand (Retail) (Sold)1 (StockN (Title) pookInfo ileStat	se xmins >TOLKIEN FANTASY >2000 0 15000000>1111 <br LOAD OF 0> us>30/30 nse>	::i="http://www.w3.org/2001/XMLSchema-instance" xmlns="http:// l v(Genre> what http:// d> StockNo THE RING K/fileStatus>

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配下の WebClient フォルダ配下の Search.html を Web ブラウザーで開いてくださ



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。



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。

🖭 » コントロール パネル



3) [Windows の機能の有効化または無効化] をクリックします。

õ	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
6	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」¹

 ¹ チェックを行うと、他の項目に対しても自動的にチェックが行われますが、そのままにしてください。
 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



Windows の機能の有効化または無効化 機能を有効にするには、チェック ボックスをオンにしてください。機能を無効にするには、チェッ クボックスをオフにしてください。塗りつぶされたチェックボックスは、機能の一部が有効になっ ていることを表します。 ■ FTP サーバー ■ ■ Web 管理ツール ■ ■ ■ World Wide Web サービス ■ ■ HTTP 共通機能 _____.NET 拡張機能 3.5 ☑ .NET 拡張機能 4.8 Application Initialization ASP ASP.NET 3.5 ASP.NET 4.8 CGI ____ ISAPI フィルター ISAPI 拡張 WebSocket プロトコル サーバー側インクルード 🛨 💳 セキュリティ □ インターネット インフォメーション サービスのホスト可能な Web コア OK キャンセル

インストール完了後、[閉じる]をクリックします。

必要な変更が完了しました。

閉じる

WHAT'S NEXT

● 本チュートリアルで学習した技術の詳細については製品マニュアルをご参照ください。

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。