

JRA－VAN Data Lab. SDK

開発ガイド(イベント VB)

2026年7月31日

第4.0.0版

JRAシステムサービス株式会社

修正履歴

日付	版	項番	種類	内容
2011/3/1	1.00	－	新規	初版作成
2017/6/6	2.00	－	修正	VisualStudio2015 へのバージョン更新
2021/5/13	3.00	－	修正	VisualStudio2019 へのバージョン更新
2021/2/22	3.01	1.1.2	修正	「プロジェクトの作成」手順の誤りを修正
2026/7/31	4.00	－	修正	VisualStudio2026 へのバージョン更新
		1.1.2	追記	プラットフォームの 64bit 変更について説明を追記
		1.1.5	追記	設定ボタンのコーディングを追記



目次

はじめに

- 1. 1 JV-Linkを使ったプログラミング(イベント取得)
 - 1. 1. 1 プログラミングのゴール
 - 1. 1. 2 プロジェクトの作成
 - 1. 1. 3 JV-Linkコントロールの追加
 - 1. 1. 4 フォームの作成
 - 1. 1. 5 設定ボタンのコーディング(JVSetUIProperties)
 - 1. 1. 6 初期化／終了処理のコーディング
 - 1. 1. 7 イベント受信処理のコーディング
 - 1. 1. 8 ビルド&実行



はじめに

平素よりJRA-VANをご利用いただき誠にありがとうございます。

弊社では JRA 公式データをより一層活用できる仕組みとして、「JRA-VAN Data Lab.」サービスを提供しております。本サービスは、データリンクモジュールのJV-Link(※)を通じて、様々な競馬データの利用が可能になるサービスです。

エンドユーザー様、およびソフトウェア作者の皆様には、この「JRA-VAN Data Lab.」サービスの仕組みをご理解いただき、JRA-VANのデータ提供サービスを活用していただきたいと願っております。

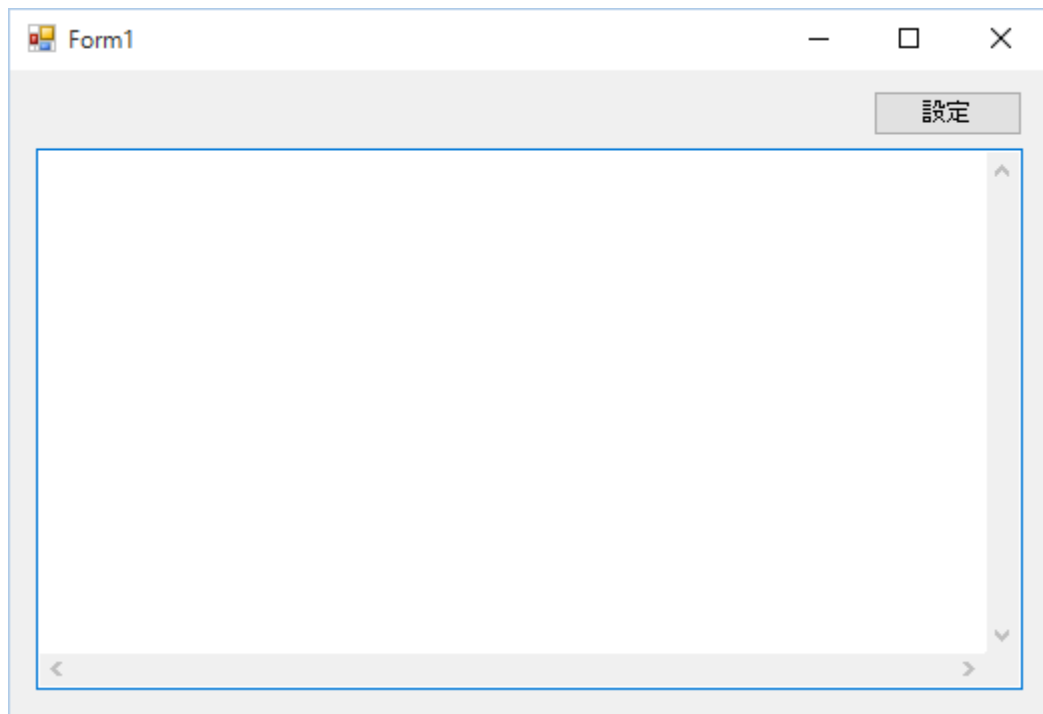
※JV-Linkの詳細については JRA-VAN Data Lab.開発ガイドの本編を参照して下さい。

1.1 JV-Linkを使ったプログラミング(イベント取得)

具体的な例を示すことによって、JV-Linkを使ってイベントを取得する方法について解説していきます。ここではVisual Basic 2026を例に順を追ってプログラミング方法を説明していきます。JV-Linkのインターフェースの詳細については触れませんので、インターフェースの詳細については「JV-Linkインターフェース仕様書」を参照して下さい。

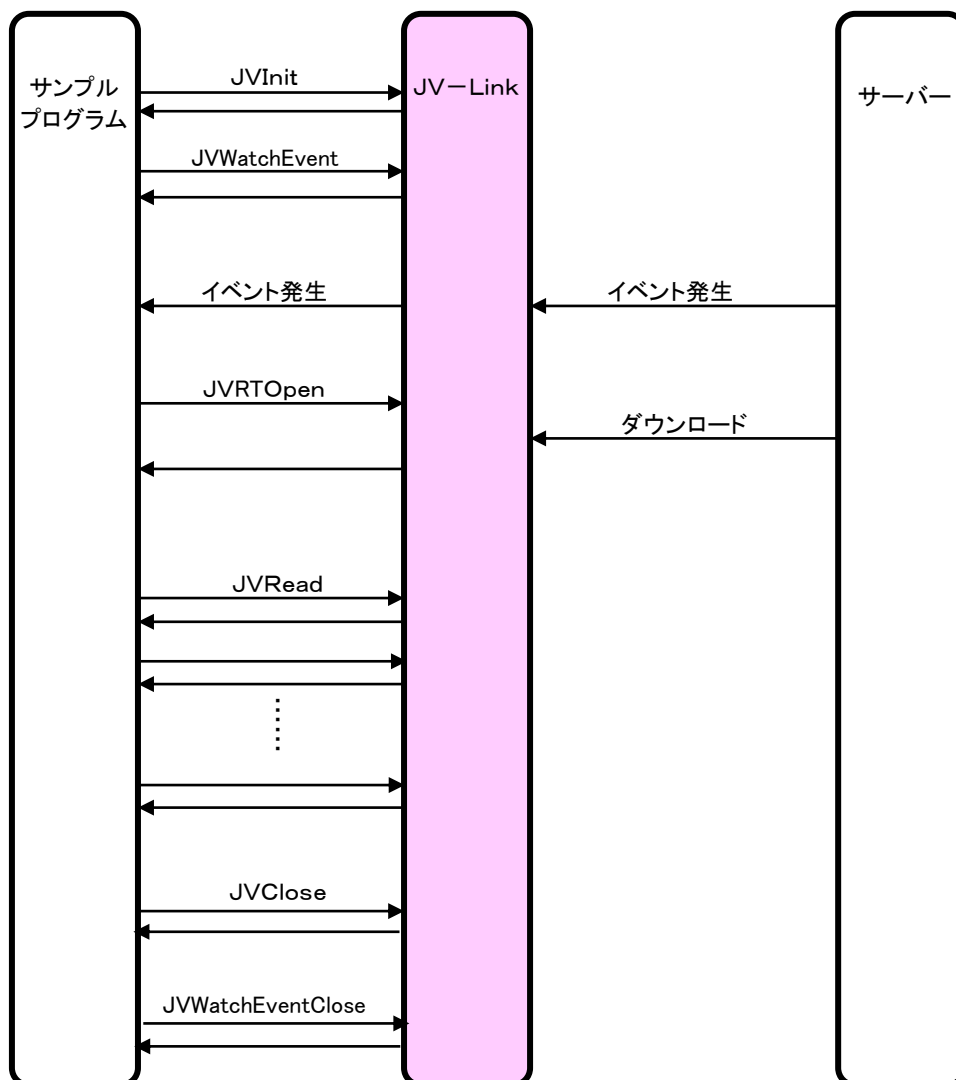
1.1.1 プログラミングのゴール

ここではJV-Linkを使ってイベントを取得して表示する簡単なプログラム(Sample2)を作成します。



このプログラムはボタンが1つとテキストボックスが1つのシンプルな画面を持っています。「設定」ボタンを押すとJV-Linkの **JVSetUIProperties** インターフェースを使ってJV-Linkの各種設定を行なうことができます。イベントが発生すると、その内容がテキストボックスに表示されます。

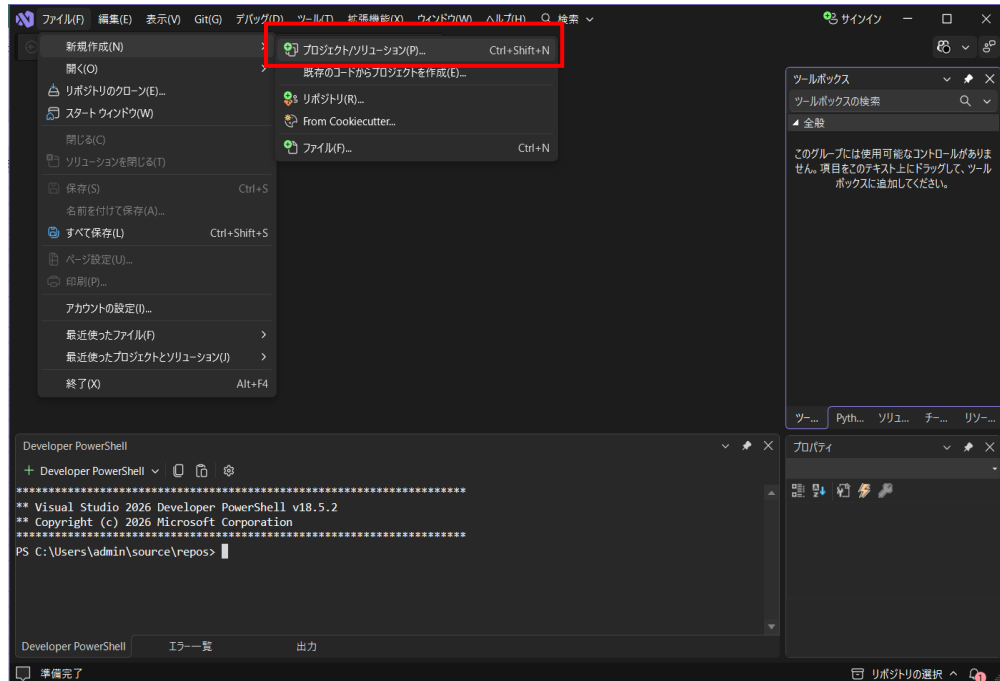
イベント取得で、JV-Link、サーバー間で行なわれる処理は以下のようになります。



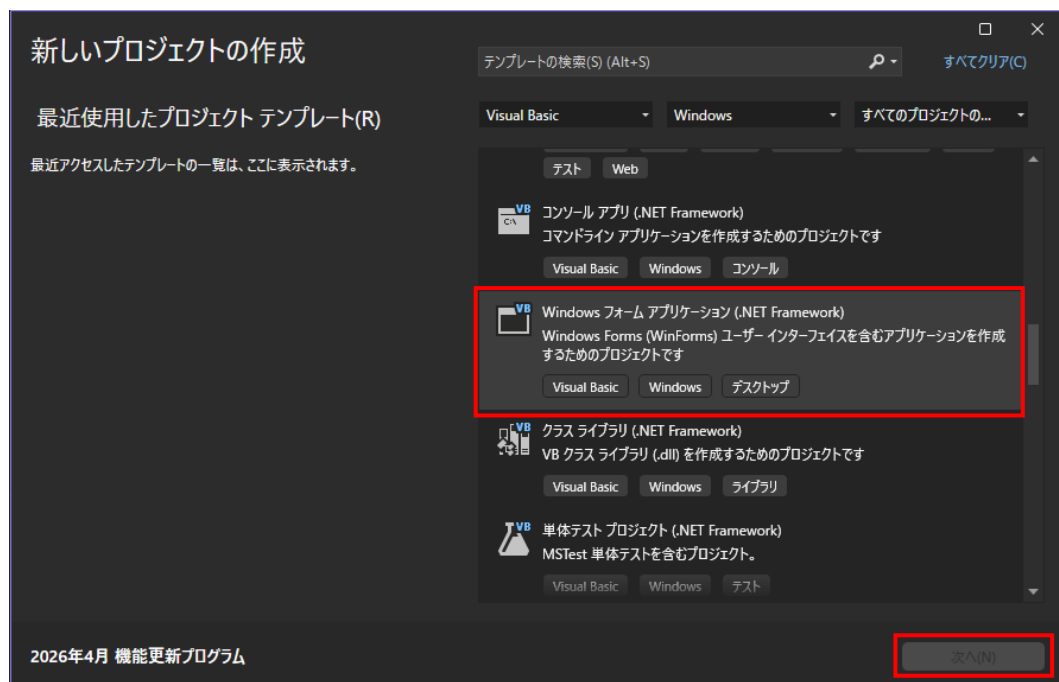
1. 1. 2 プロジェクトの作成

Visual Studio2026 を起動し、「コードなしで続行(W)」リンクをクリックする。

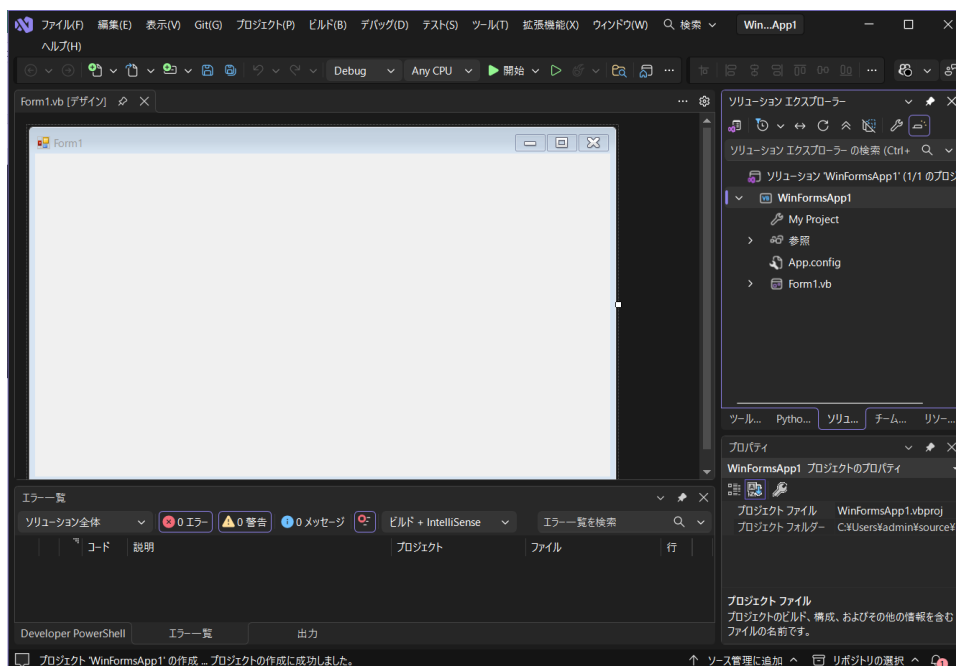
メニューバーのファイル(F)→新規作成(N)→プロジェクト/ソリューション(P) …を選択します。



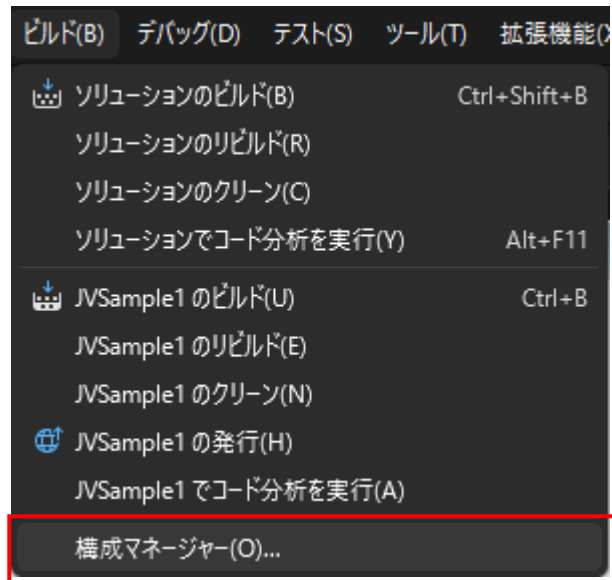
次に、プロジェクトのテンプレートから「Windows フォーム アプリケーション (.NET Framework)」を選択して「次へ(N)」ボタンをクリックします。



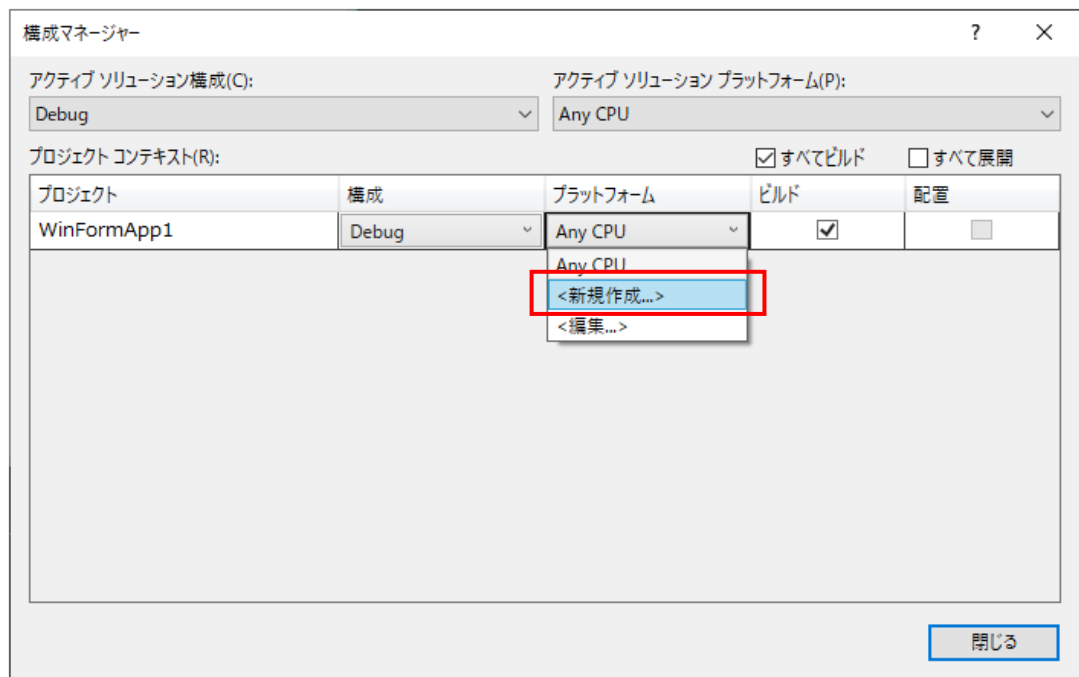
新しいプロジェクトが作成され下の画面になります。



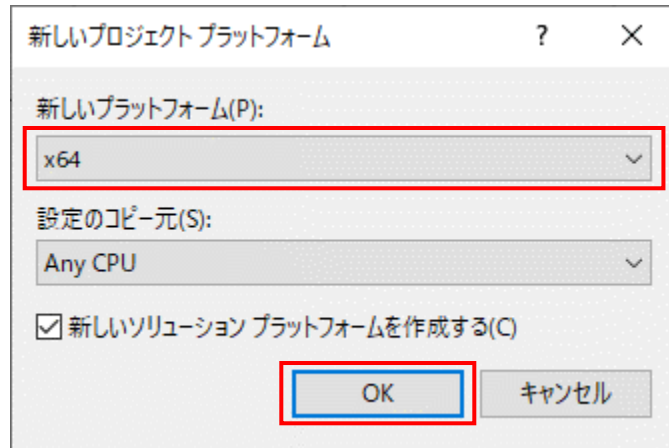
64bit 版のJV-Linkで開発を行う場合、プラットフォームを 64bit に変更します。
メニューバーのビルド(B)→構成マネージャー(O)・・・を選択する。



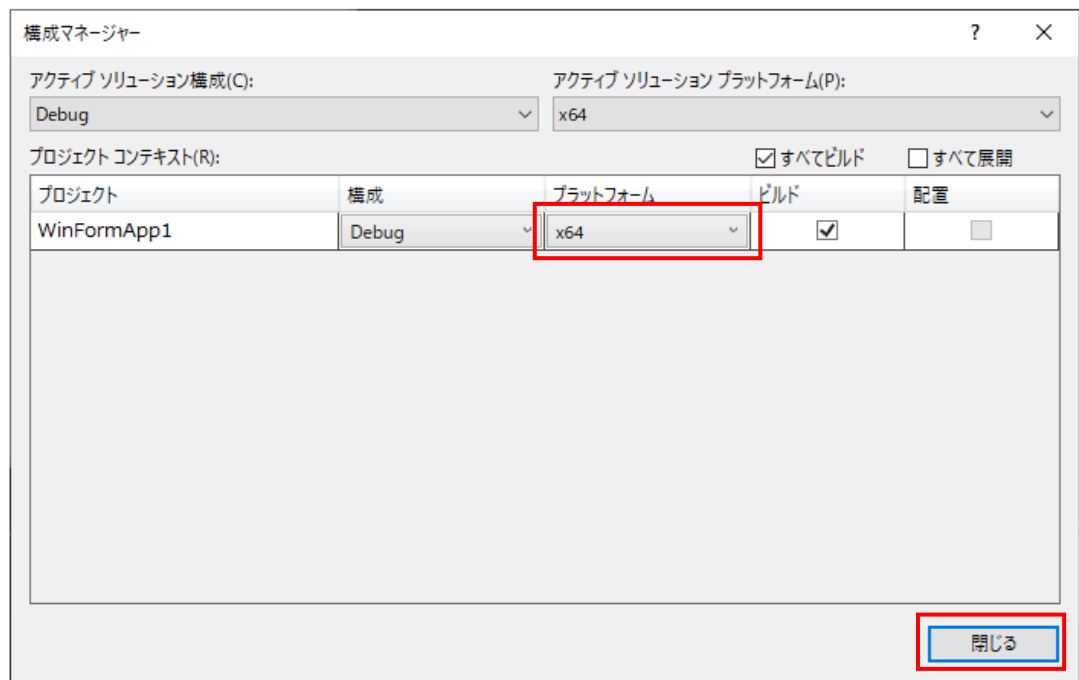
構成マネージャーダイアログのプラットフォームから「新規作成」をクリックします。



新しいプラットフォーム(P)で「x64」を選択して、「OK」ボタンをクリックします。



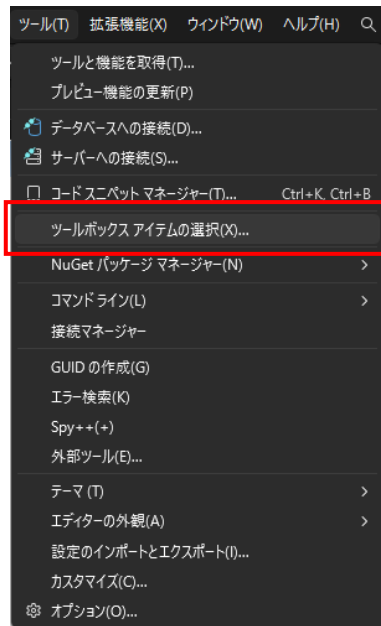
プロジェクトコンテキスト(R)のプラットフォームが「x64」に変更されることを確認し、「閉じる」ボタンをクリックします。



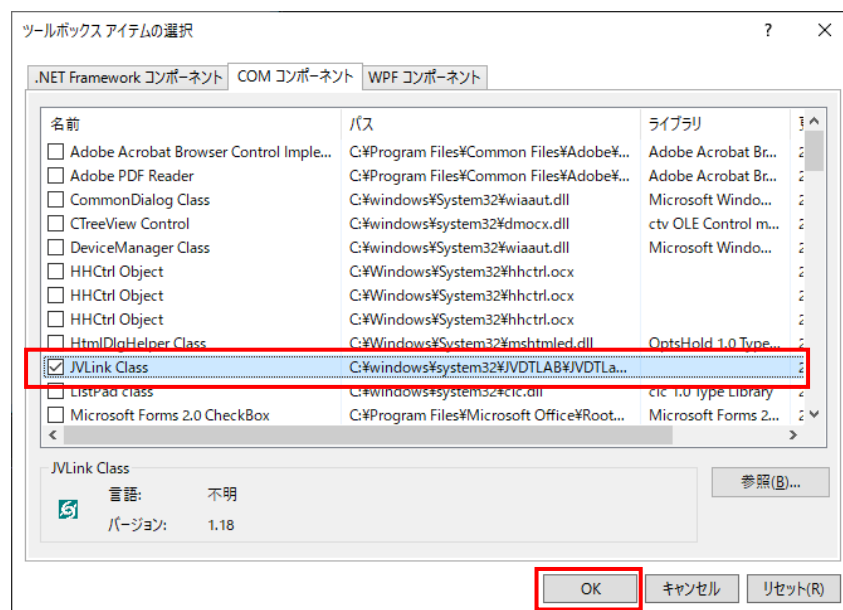
1. 1. 3 JVLinkコントロールの追加

最初に、この開発環境(Visual Studio 2026)でJV-Linkを使用できるように設定を行なう必要があります。

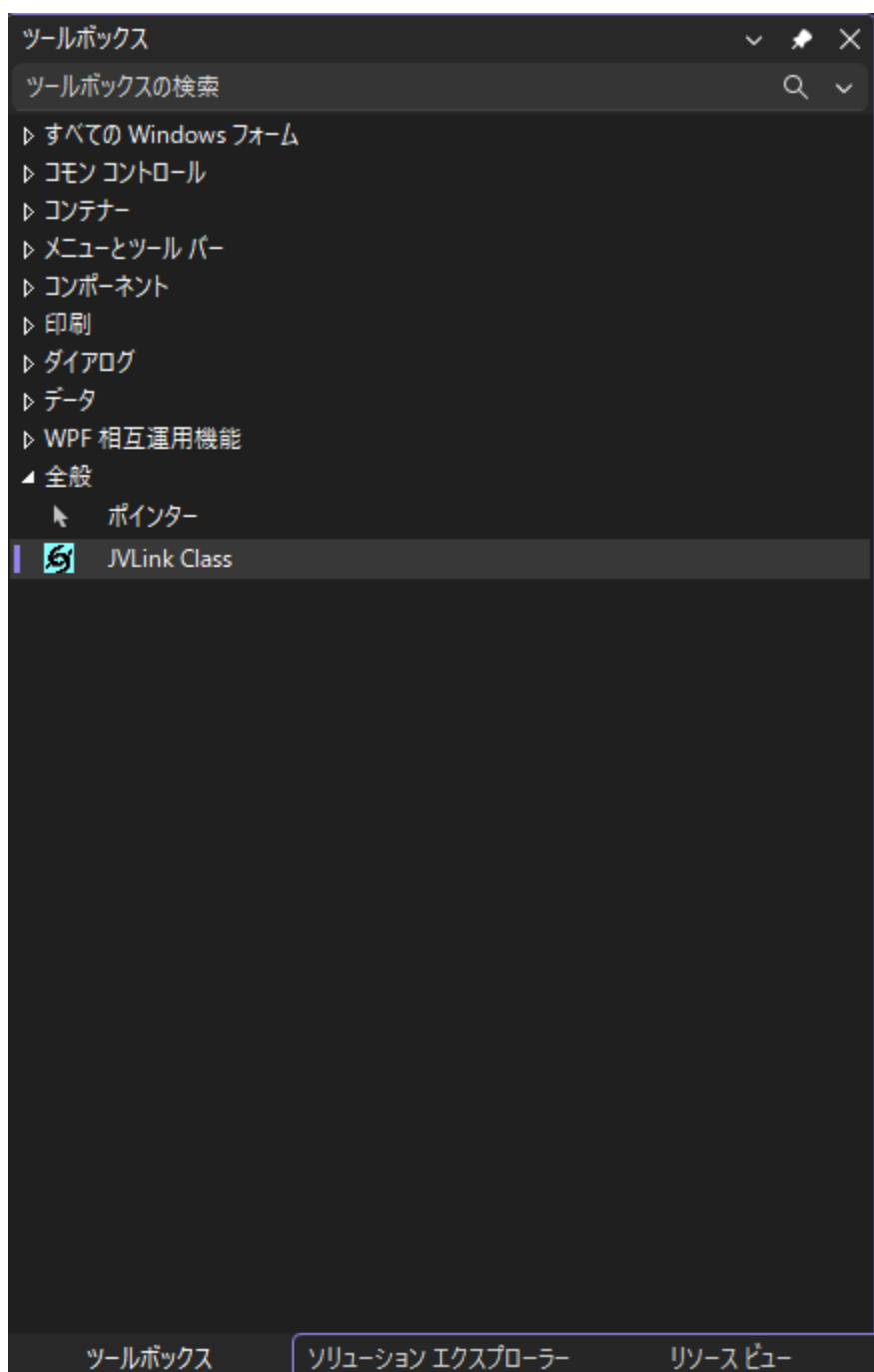
メニューバーのツール(T)→ツールボックス アイテムの選択(X)・・・を選択します。



ツールボックス アイテムの選択ダイアログの COM コンポーネントタブの中から JVLink Class にチェックを付け、OK ボタンをクリックします。

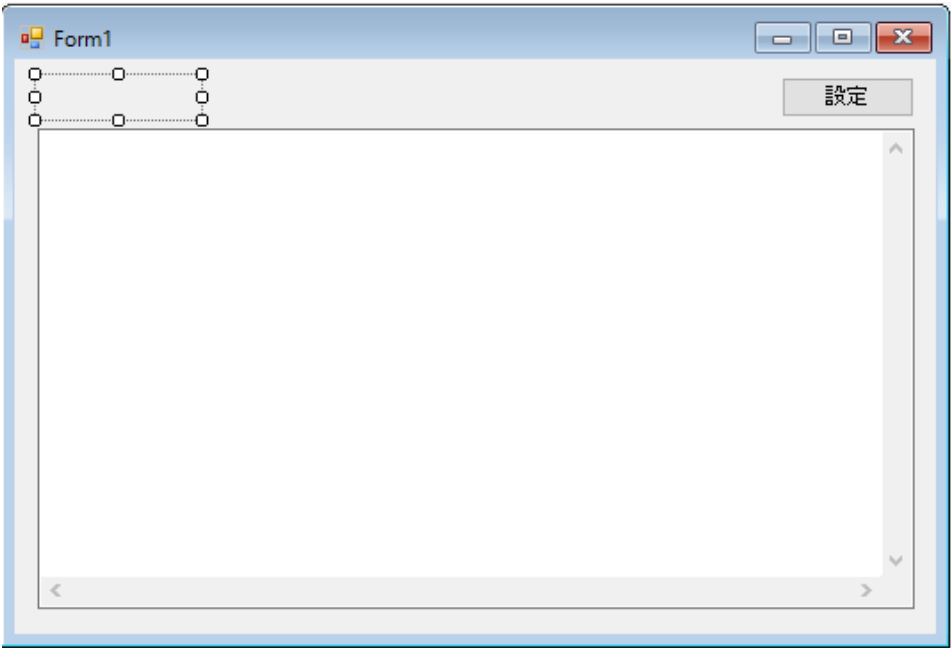


これによりツールボックスに JVLink コントロールが追加されます。



1. 1. 4 フォームの作成

下図のようにフォームにボタンとテキストボックス、および JVLink コントロールを貼り付け各プロパティを設定します。



コントロール	プロパティ	選択／設定
Button	Name Text	Button1 “設定”
TextBox	Name Text ScrollBars Multiline Wordwrap Font	txtOut 空白 Both True False 固定幅フォントの方が見やすいでしょう
JVLink Class	Name	AxJVLink1

1. 1. 5 設定ボタンのコーディング(JVSetUIProperties)

JVSample1 では設定ボタンをクリックすることによって **JVSetUIProperties** インターフェースを呼び出し、JVLinkコントロールが持つ設定ダイアログを起動します。

Button1 の Click イベントハンドラに以下のコードを記述します。

サンプルソース1

```
' -----
' 設定ボタンクリック時の処理
' -----

Private Sub Button1_Click (ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
    ' JVSetUIPropertiesの呼出し
    If (Me.AxJVLink1.JVSetUIProperties() = -100) Then _____ ①
        ' レジストリへの登録に失敗すると-100が返る
        MsgBox("エラーのためJV-L i n kの設定に失敗しました。", MsgBoxStyle.OKOnly)
    End If
End Sub
```

サンプルソース 1 解説

①JVSetUIProperties を呼び出した結果が-100 の場合は設定が何らかのエラーでレジストリに反映できなかったことを示します。

1.1.6 初期化／終了処理のコーディング

初期化処理として以下の処理を、フォーム読込イベントで行います。

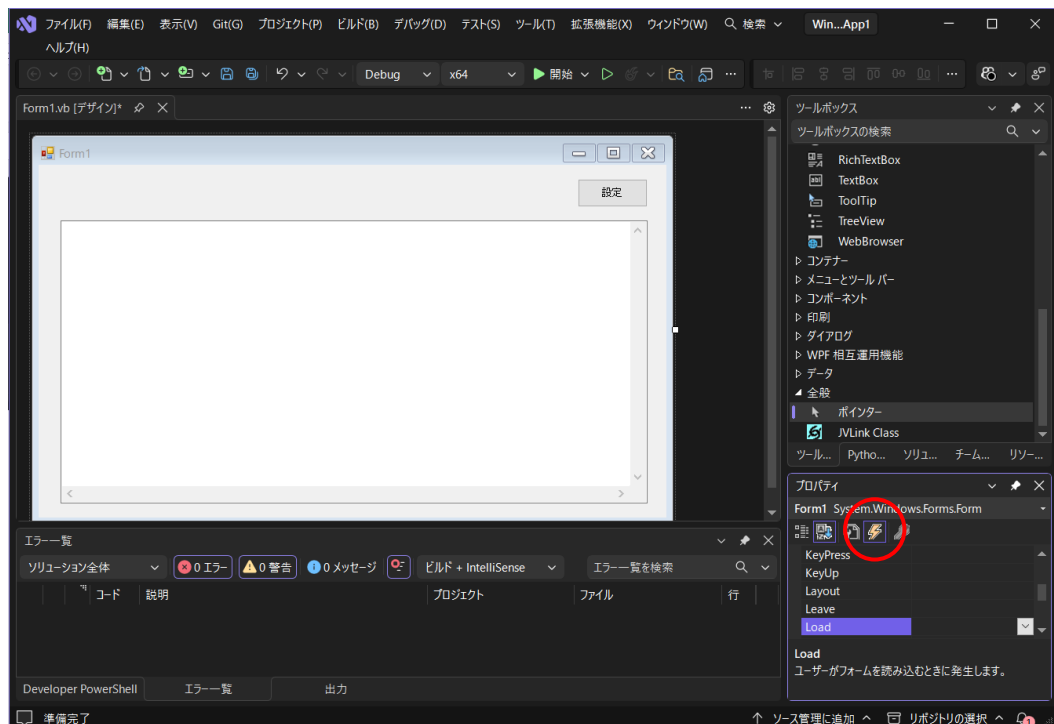
処理	内容
①JVLinkの初期化(JVInit)	JVLinkの初期化を行います。JVWatchEventを呼び出す前に最低1回呼び出す必要があります。
②イベント通知開始(JVWatchEvent)	確定・変更情報が発生した際、イベントを通知するスレッドを開始します。

終了処理として以下の処理を、フォームクローズイベントで行います。

処理	内容
①イベント通知終了(JVWatchEventClose)	イベント通知スレッドを終了します。

デザイン画面を表示し、Form1 の「プロパティ」から「イベント」ボタンを押し、「Load」の関数名入力部分をダブルクリックすると「Form1.vb」にイベント関数が作成されますので、それに対して初期化処理のコードを記述していきます。

同様に「FormClosed」の関数名入力部分をダブルクリックしてイベント関数を作成し、終了処理のコードを記述していきます。



Form1 の Load／Closed イベント関数に以下のコードを記述します。

サンプルソース2

```

' -----
'   初期化
' -----

Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    ' *****
    ' JVLink初期化
    ' *****
    ' ※※※JVInitはJVLinkメソッド使用前(但し、JVSetUIPropertiesを除く)に呼出す
    Dim ReturnCode As Integer = AxJVLink1.JVInit("UNKNOWN") ①

    ' エラー判定
    If ReturnCode <> 0 Then      '' エラー
        txtOut.AppendText("JVInitエラー:" & ReturnCode & ControlChars.CrLf)
    Else                        '' 正常
        ' イベント監視オブジェクト初期化
        ReturnCode = AxJVLink1.JVWatchEvent() ②

        If ReturnCode <> 0 Then      '' エラー
            txtOut.AppendText("JVWatchEventエラー:" & ReturnCode & ControlChars.CrLf)
        End If
    End If
End Sub

' -----
'   フォーム閉じる
' -----

Private Sub Form1_FormClosed(ByVal sender As System.Object, ByVal e As System.Windows.Forms.FormClosedEventArgs) Handles MyBase.FormClosed
    ' イベント監視オブジェクト終了
    Dim ReturnCode As Integer = AxJVLink1.JVWatchEventClose() ③
End Sub

```

サンプルソース2解説

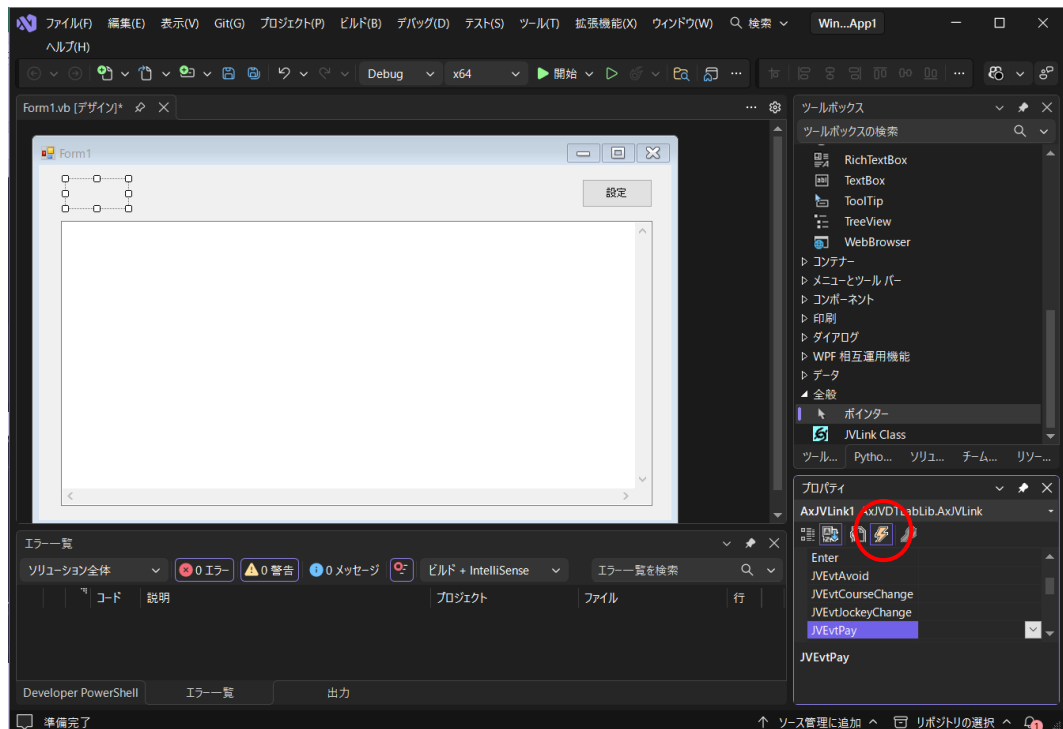
- ①JVInit にはアプリケーションIDを渡します。このアプリケーションIDはサーバーとの通信の際にHTTPヘッダーのUser-Agentとして使用されます。ソフトの名前とバージョン番号を組み合わせてユニークなIDを指定して下さい。開発途中でアプリケーションIDを決められない場合には“UNKNOWN”を指定して下さい。
- ②イベント通知開始をおこなうことで、それ以降に払戻確定、騎手変更、天候馬場状態変更、コース変更、出走取消・競走除外、発走時刻変更、馬体重が発表された際、イベントを受理することが可能になります。
- ③イベント受信処理を終了する場合に JVWatchEventClose を呼び出します。
今回のサンプルではフォームを閉じる時(プログラムが終了する時)に呼び出しています。

1.1.7 イベント受信処理のコーディング

各イベントに対してのイベント関数を作成することにより、対象イベントが発生した際の処理をコーディングすることが可能となります。

ここではサンプルとして、払戻確定イベントが発生した際のコードを載せますが、それ以外のイベントも作り方は同じです。

デザイン画面を表示し、AxJVLink1 の「プロパティ」から「イベント」ボタンを押し、「JVEvtPay」の関数名入力部分をダブルクリックすると「Form1.vb」にイベント関数が作成されますので、それに対してコードを記述していきます。



AxJVLink1 の JVEvtPay イベント関数に以下のコードを記述しさらに読み込み&表示処理の関数を作成します。

このサンプルでは、イベントが発生した元となるデータを取得し画面に表示しています。

処理	内容
①リアルタイム系データの取得要求(JVRTOpen)	リアルタイム系データの取得要求をします。
②JV-Data の読み込み(JVRead)	JV-Data の読み込みをします。
③JV-Data 読み込み処理の終了(JVClose)	JV-Data 読み込み処理の終了をします。

サンプルソース3

```

' -----
'      払戻速報
' -----

Private Sub AxJVLink1_JVEvtPay(ByVal sender As System.Object, ByVal e As
AxJVDTLibLib._IJVLinkEvents_JVEvtPayEvent) Handles AxJVLink1.JVEvtPay
    MsgBox("払戻速報：" & e.bstr, , Text)

    Dim ReturnCode As Integer = AxJVLink1.JVRTOpen("0B12", e.bstr)      ①
    If ReturnCode <> 0 Then
        txtOut.AppendText("JVRTOpenエラー：" & ReturnCode & ControlChars.CrLf)
    Else
        txtOut.AppendText("【速報レース情報(払戻速報) dataspec = " & "0B12" & " key = " & e.bstr
& "】" & ControlChars.CrLf)

        ' 読み込み&表示処理
        ReadData()
    End If
End Sub

' -----
'      読み込み&表示処理
' -----

Private Sub ReadData()
    Dim buffSize As Integer = 150000
    Dim buff As String = New String(vbNullChar, buffSize)
    Dim fName As String = ""
    Dim ReturnCode As Integer

    Do
        ReturnCode = AxJVLink1.JVRead(buff, buffSize, fName)      ②
        Select Case ReturnCode
            Case Is > 0      ' 正常
                txtOut.AppendText(buff)

            Case -1
                ' 何もしない

            Case 0      ' 読込終了
                Exit Do

            Case Else      ' エラー
                txtOut.AppendText("JVReadエラー：" & ReturnCode & ControlChars.CrLf)
                Exit Do
        End Select
    Loop

    ' JVClose
    ReturnCode = AxJVLink1.JVClose()      ③
End Sub

```

サンプルソース3解説

①イベント発生元となったリアルタイム系データの取得要求をします。

第1パラメータはイベント毎に決まっている「データ種別ID」です。

詳細については「JV-Linkインターフェース仕様書」を参照して下さい。

②JVROpen で準備した JV-Data を現在のファイルポインタから1行分読み出します。

正の数が返された場合は、正常にレコードを読み込んでいますので(文末に改行コード付)

データ格納バッファを画面にそのまま表示します。

-1が返された場合は、読み込み対象ファイルの変更を意味しているので、何もせず次のデータを読み込みに入ります。

全てのファイルを読み終わると0が返りますので、読み込みループを終了します。

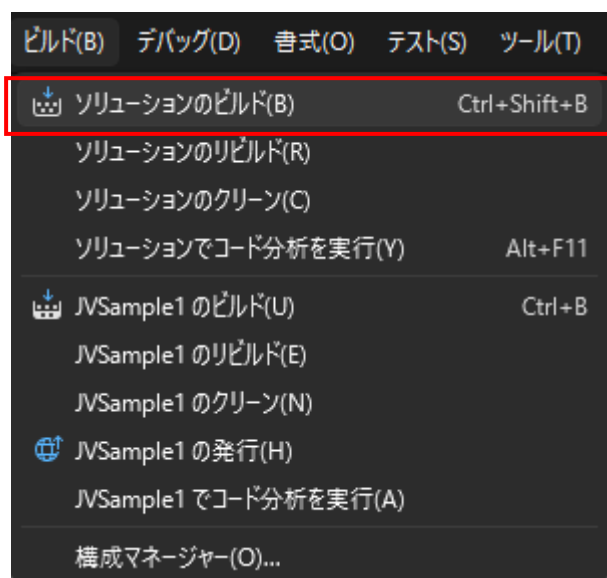
エラーが発生した場合にはエラーの理由コードとして負の数が返されます。

JVRead ではなく JVGets を使って読み出すことも可能です。

③JV-Data読み込み処理を終了します。

1. 1. 8 ビルド&実行

メニューバーのビルド(B)→ソリューションのビルド(B)を選択します。



以上で Sample2 の開発は終了です。ビルドによって作成された.exe ファイルをダブルクリックするかデバッグコマンドにより実行が可能です。

実際にイベントを発生させて、その動作を検証するには DataLab 検証ツールを使います。

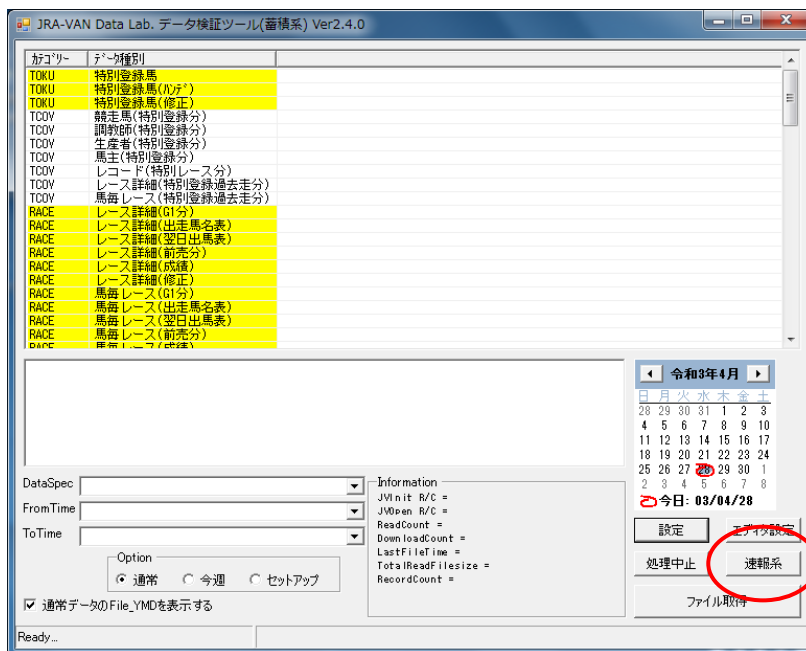
(DataLab 検証ツールはJRA-VAN のホームページの「プログラミングパーツ・開発支援ツール提供コーナー」にありますので、ダウンロードしてください。)

DataLab 検証ツールのインストール及び詳しい使用方法に関しては、DataLab 検証ツール説明書を参照してください。

今回サンプルソースで作成した、払戻確定イベントを発生させるには以下の様に操作します。

① DataLab 検証ツールを起動します。

② 「速報系」ボタンをクリックします。



- ③ 年月日場レースを入力し、「イベントを通知する」をチェックし、「擬似通知(払戻)」をクリックします。

- ④ 払戻結果が Sample2 に表示されます。

DataLab 検証ツールと同じデータが表示されていることを確認してください。