プロダクション内での TCP アダプタの使用法

Version 2023.1
2024-01-02

InterSystems Corporation　1 Memorial Drive　Cambridge MA 02142　www.intersystems.com

プロダクション内での TCP アダプタの使用法
InterSystems IRIS Data Platform　Version 2023.1　2024-01-02

インターシステムズ社の製品に関するサポートやご質問は、以下にお問い合わせください：

InterSystems Worldwide Response Center (WRC)
Tel:　+1-617-621-0700
Tel:　+44 (0) 844 854 2917
Email:　support@InterSystems.com

目次

1

TCP 受信アダプタの使用法

このページでは、TCP 受信アダプタのそれぞれ (EnsLib.TCP.CountedInboundAdapter、EnsLib.TCP.CountedXMLInboundAdapter、および EnsLib.TCP.TextLineInboundAdapter) の使用方法について説明します。

Tipヒント InterSystems IRIS® では、TCP アダプタを使用する特殊なビジネス・サービス・クラスとユーザのニーズに適したビジネス・サービス・クラスの 1 つも提供されます。そのため、プログラミングの必要がありません。″相互運用プロダクションの概要″ の “接続オプション” を参照してください。

また、EnsLib.TCP.InboundAdapter またはそのサブクラスのいずれかに基づいて新しい受信アダプタ・クラスを開発することもできます。後述する “カスタム TCP アダプタ・クラスの作成” の節を参照してください。

1.1 TCP 受信アダプタの概要

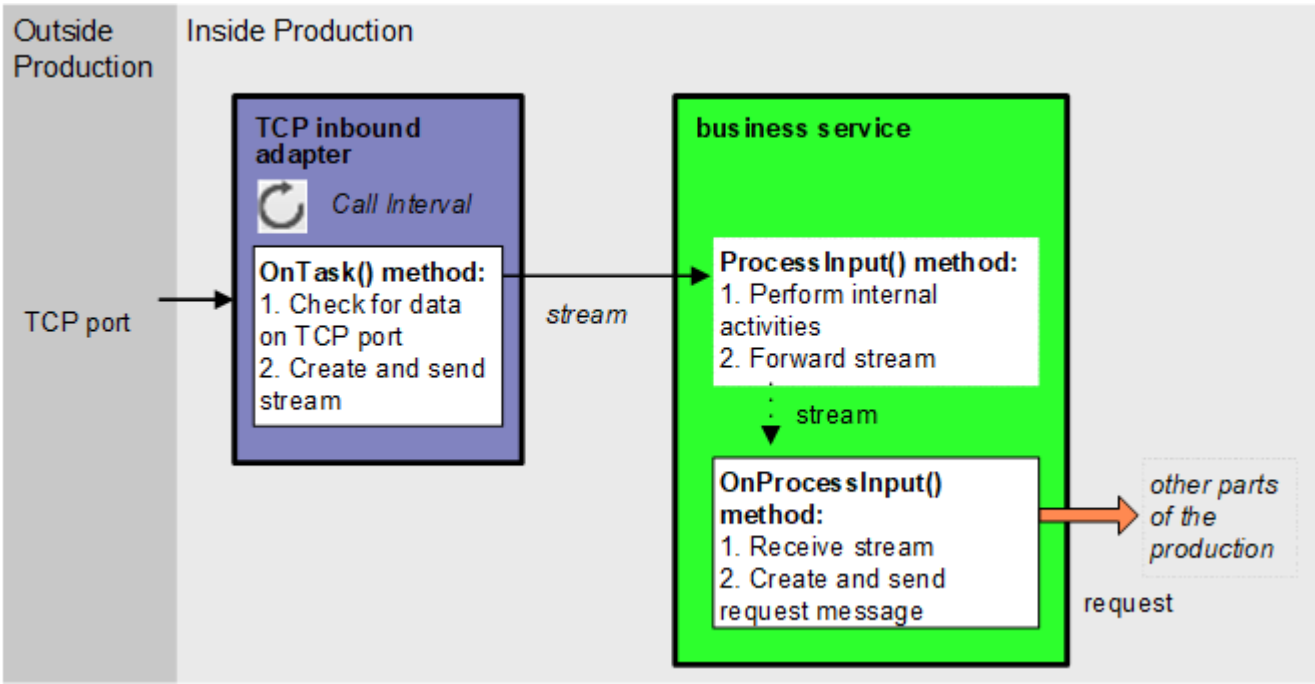
InterSystems IRIS には、すべてが EnsLib.TCP.InboundAdapter のサブクラスである以下の受信 TCP アダプタが用意されています。

- EnsLib.TCP.CountedInboundAdapter は、TCP クライアントと TCP リスナがデータのブロックを交換する受信 TCP 接続をサポートします。このブロック長は、ブロックの最初の 4 バイトで指定されます。アダプタは、クライアント・アプリケーションからのデータの意味のある部分を特定するためにブロック長を使用します。
- EnsLib.TCP.CountedXMLInboundAdapter は、XTE サーバの TCP リスナとして機能します。アダプタは XML データをカウントされた TCP 形式で受信し、XML データを InterSystems IRIS にインポートします。その結果は、インスタンス化された InterSystems IRIS オブジェクトとなります。
- EnsLib.TCP.TextLineInboundAdapter は、TCP クライアントと TCP リスナがデータを行終端文字で終了するテキスト文字列として交換する受信 TCP 接続をサポートします。終端文字のデフォルトは新規行文字 (ASCII 10) です。

  組み込みのビジネス・サービス・クラス EnsLib.TCP.StatusService は、EnsLib.TCP.TextLineInboundAdapter を使用するビジネス・サービスの重要な例です。このクラスにはコマンド行インタフェースが用意されており、これをコンソール・ユーザやコマンド行スクリプトで使用して、実行中のプロダクションから基本ステータス情報を取得できます。詳細は、後述する “TCP ステータス・サービスの追加” を参照してください。

1.2 全般的な動作

それぞれの TCP 受信アダプタが、指定されたポートでデータをチェックして、入力を読み取り、それをストリームとして関連するビジネス・サービスに送信します。ユーザが作成および構成するビジネス・サービスは、このストリームを使用してプロダクションの他の部分と通信します。以下の図は、全体的なフローを示しています。



詳細は、以下のとおりです。

1. アダプタは、構成されたデータ・ソースからの入力を検出するたびに、ビジネス・サービス・クラスの内部 ProcessInput() メソッドを呼び出して、ストリームを入力引数として渡します。
2. ビジネス・サービス・クラスの内部 ProcessInput() メソッドが実行されます。このメソッドは、すべてのビジネス・サービスが必要とする内部情報の保持など、基本的なプロダクション・タスクを実行します。ビジネス・サービス・クラスが継承するこのメソッドは、カスタマイズや上書きを行いません。
3. 次に、ProcessInput() メソッドがカスタム OnProcessInput() メソッドを呼び出して、入力オブジェクトを渡します。このメソッドの要件は、このページの後半を参照してください。

応答メッセージは、同じパスを逆向きにたどります。

1.3 TCP 受信アダプタを使用するビジネス・サービスの作成

プロダクション内で TCP アダプタを使用するには、ここで説明されているように、新しいビジネス・サービス・クラスを作成します。後で、それをプロダクションに追加して、構成します。

存在しなければ、適切なメッセージ・クラスを作成する必要もあります。"プロダクションの開発" の "メッセージの定義" を参照してください。

スタジオには、ビジネス・サービス・スタブを作成するために使用可能なウィザードが用意されています。このウィザードにアクセスするには、[ファイル]→[新規作成] をクリックし、[プロダクション] タブをクリックします。次に [ビジネス・サービス]

をクリックして [OK] をクリックします。このウィザードには、汎用入力引数が用意されています。ウィザードを使用する場合は、選択したアダプタに必要な特定の引数を使用するようにメソッド・シグニチャを編集することをお勧めします。次の節を参照してください。

ビジネス・サービス・クラスの基本要件を以下に列挙します。

- ビジネス・サービス・クラスは Ens.BusinessService を拡張するものでなければなりません。
- クラス内の ADAPTER パラメータは EnsLib.TCP.CountedInboundAdapter、EnsLib.TCP.CountedXMLInboundAdapter、または EnsLib.TCP.TextLineInboundAdapter と一致する必要があります。
- クラスは OnProcessInput() メソッドを実装する必要があります。これについては、“OnProcessInput() メソッドの実装”で説明します。
- クラスは OnConnect() コールバック・メソッドを実装できます。このメソッドは、TCP 受信アダプタがリモート・システムとの新しい接続を確立するたびに、OnTask() メソッドの後に呼び出されます。

  このメソッドを実装する場合は、以下のシグニチャを付与する必要があります。

  ```
  Method OnConnect(pTimeout As %Numeric) As %Status
  ```

  新しい接続が確立されるたびに何らかの処理 (ログオン・シーケンスやハンドシェーク・トークンの送信など) を実行する場合は、OnConnect() を実装します。timeout 引数は、TCP 受信アダプタによって自動的に指定されます。この値には、Read Timeout アダプタ設定の値が入力されます。Read Timeout の詳細は、“設定の参照先”を参照してください。

  OnConnect() がエラー・ステータスを返した場合は、新しい接続が失敗して、アダプタが切り離されます。トラップされない例外が OnConnect() 内で発生した場合、アダプタはその例外をキャッチして、OnConnect() が実装されていなかった場合と同じように処理を継続します。

  その他のオプションと一般情報は、″プロダクションの開発″ の “ビジネス・サービス・クラスの定義”を参照してください。
- クラスでは、OnInit() メソッドを実装して特殊な設定アクションを実行できます。このメソッドは、%OnNew() メソッドから呼び出します。[接続毎のジョブ] 設定を `true` に設定すると、リスナが接続を受け入れるために新しいジョブを生成するたびに、OnInit() が %OnNew() から呼び出されます。
- クラスでは、OnTearDown() メソッドを実装して特殊なティアダウン・アクション (終了処理) を実行できます。このメソッドは、%OnClose() メソッドから呼び出します。具体的には、%OnClose() によって、アダプタの OnTearDown() メソッドが呼び出された後、ビジネス・サービスの OnTearDown() メソッドが呼び出されます。[接続毎のジョブ] 設定を `true` に設定すると、OnTearDown() は、接続ジョブが閉じられる直前に呼び出されます。

  重要　新規のビジネス・サービス・クラスで、OnTearDown() メソッドにカスタム・コードを含める場合、そのビジネス・サービスが適切に機能するように、そのカスタム・コードでスーパークラスの OnTearDown() メソッドを呼び出す必要があります。以下に例を示します。

  ```
  //Sets a node of the ^%zexample global to the ID of the current process
  // and invokes the OnTearDown() method of the superclass
  Method OnTearDown() As %Status
  {
     set ^%zexample($i(^%zexample),"onteardown")=$j
     Return ##super()
  }
  ```

  エラーを処理するために、適宜、`Try/Catch` ロジックを実装する必要もあります。スーパークラス・メソッドを呼び出す方法の詳細は、″##super 構文″ を参照してください。

1.4 OnProcessInput() メソッドの実装

この節では、アダプタによって異なる OnProcessInput() のメソッド・シグニチャとそのメソッドの実装方法について説明します。

1.4.1 EnsLib.TCP.CountedInboundAdapter の OnProcessInput() のシグニチャ

ビジネス・サービス・クラスで EnsLib.TCP.CountedInboundAdapter が使用されている場合は、OnProcessInput() メソッドに次のシグニチャを付与する必要があります。

```
Method OnProcessInput(pInput As %Library.GlobalCharacterStream,
                      Output pOutput As %Library.AbstractStream) As %Status
```

説明：

- pInput には TCP クライアントがアダプタに転送した受信データ・ストリームが含まれています。
- pOutput にはビジネス・サービスが TCP クライアントに提供する可能性のあるすべての応答が含まれています。

1.4.2 EnsLib.TCP.CountedXMLInboundAdapter の OnProcessInput() のシグニチャ

ビジネス・サービス・クラスで EnsLib.TCP.CountedXMLInboundAdapter が使用されている場合は、OnProcessInput() メソッドに次のシグニチャを付与する必要があります。

```
Method OnProcessInput(pInput As %RegisteredObject,
                      Output pOutput As %RegisteredObject) As %Status
```

説明：

- pInput は Accept Class Names アダプタ設定によって指定された任意のオブジェクトにすることができます。

  Accept Class Names の詳細は、“設定の参照先”を参照してください。
- pOutput には XTE サーバに返される可能性のあるすべての応答が含まれています。

1.4.3 EnsLib.TCP.TextLineInboundAdapter の OnProcessInput() のシグニチャ

ビジネス・サービス・クラスで EnsLib.TCP.TextLineInboundAdapter が使用されている場合は、OnProcessInput() メソッドに次のシグニチャを付与する必要があります。

```
Method OnProcessInput(pInput As Ens.StringContainer,
                      Output pOutput As Ens.StringContainer) As %Status
```

説明：

- pInput には受信テキスト行が含まれています。
- pOutput には送信応答文字列 (存在する場合) が含まれています。

1.4.4 OnProcessInput() の実装

すべてのケースにおいて、OnProcessInput() メソッドは以下の一部または全部を実行する必要があります。

1. 入力オブジェクト (pInput) を検査して、その使用方法を決定します。
2. ビジネス・サービスから送信されることになる要求メッセージのインスタンスを作成します。

メッセージ・クラスの作成方法は、"プロダクションの開発" の "メッセージの定義" を参照してください。

3. 要求メッセージの場合は、必要に応じて、入力内の値を使用してそのプロパティを設定します。
4. ビジネス・サービスの適切なメソッドを呼び出して、要求をプロダクション内の宛先に送信します。具体的には、SendRequestSync()、SendRequestAsync()、または (あまり一般的ではない) SendDeferredResponse() を呼び出します。詳細は、"プロダクションの開発" の "要求メッセージの送信" を参照してください。

   これらの各メソッドは、ステータス (具体的には、%Status のインスタンス) を返します。
5. 必ず出力引数 (pOutput) を設定します。通常、受信した応答メッセージと同じように設定します。この手順は必須です。
6. データ・ソースがその入力に対する確認または応答を期待している場合は、OnProcessInput() がこの応答を作成して、アダプタ経由でデータ・ソースに返送する必要があります。
7. 適切なステータスを返します。この手順は必須です。

1.5 EnsLib.TCP.TextLineInboundAdapter の例

EnsLib.TCP.TextLineInboundAdapter を使用するビジネス・サービス・クラスの例を以下に示します。

Class Definition

```
Class TestTCPTextLine.AuthorizationTCPService Extends Ens.BusinessService
{
/// Name of the adapter class
Parameter ADAPTER = "EnsLib.TCP.TextLineInboundAdapter";

Method OnProcessInput(pInput As Ens.StringContainer,
                      pOutput As Ens.StringContainer) As %Status
{
  set $ZT = "EXCEPTION"
  set st = $$$OK

  do {
    if ('$isobject($get(pInput))) { quit }

    // Input must have the following format: 'PatientCode:ProcedureCode'
    set tSubject = pInput.StringValue
    $$$TRACE("received line "_tSubject)

    set req = ##class(TestTCPTextLine.AuthorizationRequest).%New()
    set req.PatientCode = $piece(tSubject,":",1)
    set req.ProcedureCode = $piece(tSubject,":",2)

    set st = ..SendRequestSync("AuthorizationProcess", req, .resp)
    quit:$$$ISERR(st)

    set pOutput=
      ##class(Ens.StringContainer).%New(resp.AuthorizationFlag_
        ":"_resp.AuthorizationCode)
    } while (0)

EXIT
   //do ..Adapter.Disconnect()
   quit st

EXCEPTION
   set $ZT = ""
   set st = $$$EnsSystemError
   goto EXIT
}
}
```

1.6 ビジネス・サービスの追加と構成

ビジネス・サービスをプロダクションに追加するには、管理ポータルを使用して以下の操作を行います。

1. ビジネス・サービス・クラスのインスタンスをプロダクションに追加します。
2. ビジネス・サービスを構成します。設定の詳細は、“設定の参照先”を参照してください。

   ビジネス・サービスを構成するときに、他の設定と一緒に **[許可IPアドレス]** 設定の値を指定します。**[許可IPアドレス]** はアダプタで接続を開始する方法を示していることに注意してください。
3. ビジネス・サービスを有効化します。
4. プロダクションを実行します。

2

TCP 送信アダプタの使用法

このページでは、TCP 送信アダプタのそれぞれ (EnsLib.TCP.CountedOutboundAdapter、EnsLib.TCP.CountedXMLOutboundAdapter、および EnsLib.TCP.TextLineOutboundAdapter) の使用方法について説明します。

Tipヒント InterSystems IRIS® では、TCP アダプタを使用する特殊なビジネス・サービス・クラスとユーザのニーズに適したビジネス・サービス・クラスの 1 つも提供されます。そのため、プログラミングの必要がありません。"相互運用プロダクションの概要" の "接続オプション" を参照してください。

また、EnsLib.TCP.OutboundAdapter またはそのサブクラスのいずれかに基づいて新しい送信アダプタ・クラスを開発することもできます。後述する "カスタム TCP アダプタ・クラスの作成" の節を参照してください。

2.1 TCP 送信アダプタの概要

InterSystems IRIS には、すべてが EnsLib.TCP.OutboundAdapter のサブクラスである以下の送信 TCP アダプタが用意されています。

- EnsLib.TCP.CountedOutboundAdapter には、TCP 接続経由でデータのブロックを読み書きするためのメソッドが含まれています。InterSystems IRIS は、TCP リスナとデータ・ブロックを交換する TCP クライアントとして動作します。ブロック長はブロックの先頭 4 バイトで指定されています。この規則を使用することで、外部 TCP サーバが要求を処理できるように、ビジネス・オペレーションは要求を送信します。
- EnsLib.TCP.CountedXMLOutboundAdapter は、XML エクスポート・オブジェクトをカウントされたバイト・ブロックとして TCP 接続経由で送信し、応答オブジェクトをインポートします。
- EnsLib.TCP.TextLineOutboundAdapter には、TCP 接続経由でテキスト文字列を読み書きするためのメソッドが含まれています。InterSystems IRIS は、TCP リスナとデータ・ブロックを交換する TCP クライアントとして動作します。テキスト文字列の終端文字のデフォルトは新規行文字 (ASCII 10) です。

2.2 全般的な動作

プロダクション内で、送信アダプタは、ユーザが作成および構成するビジネス・オペレーションに関連付けられます。このビジネス・オペレーションはプロダクション内からメッセージを受信し、メッセージ・タイプを調べ、適切なメソッドを実行します。このメソッドは、通常、関連するアダプタのメソッドを実行します。

2.3 TCP 送信アダプタを使用するビジネス・オペレーションの作成

TCP 送信アダプタを使用するビジネス・オペレーションを作成するために、新しいビジネス・オペレーション・クラスを作成します。後で、それをプロダクションに追加して、構成します。

存在しなければ、適切なメッセージ・クラスを作成する必要もあります。"プロダクションの開発" の "メッセージの定義" を参照してください。

ビジネス・オペレーション・クラスの基本要件を以下に列挙します。

- ビジネス・オペレーション・クラスは、Ens.BusinessOperation を拡張するものでなければなりません。
- クラス内の ADAPTER パラメータは EnsLib.TCP.CountedOutboundAdapter、EnsLib.TCP.CountedXMLOutboundAdapter、または EnsLib.TCP.TextLineOutboundAdapter と一致する必要があります。
- クラスの INVOCATION パラメータは、使用する呼び出しスタイルを指定する必要があります。以下のいずれかを使用します。
  - Queue は、メッセージが 1 つのバックグラウンド・ジョブ内で作成され、元のジョブが解放された段階でキューに配置されることを意味します。その後、このメッセージが処理されると、別のバックグラウンド・ジョブがこのタスクに割り当てられます。これは最も一般的な設定です。
  - InProc は、メッセージが、作成されたジョブと同じジョブで生成、送信、および配信されることを意味します。このジョブは、メッセージが対象に配信されるまで送信者のプールに解放されません。これは特殊なケースのみに該当します。
- クラスでは、少なくとも 1 つのエントリを含むメッセージ・マップを定義する必要があります。メッセージ・マップは、以下の構造を持つ XData ブロック・エントリです。

```
XData MessageMap
{
<MapItems>
  <MapItem MessageType="messageclass">
    <Method>methodname</Method>
  </MapItem>
  ...
</MapItems>
}
```

- クラスでは、メッセージ・マップ内で名前が付けられたすべてのメソッドを定義します。これらのメソッドは、メッセージ・ハンドラと呼ばれます。各メッセージ・ハンドラは、以下のシグニチャを持っている必要があります。

```
Method Sample(pReq As RequestClass, Output pResp As ResponseClass) As %Status
```

  ここで、Sample はメソッド名、RequestClass は要求メッセージ・クラスの名前、ResponseClass は応答メッセージ・クラスの名前です。通常、メソッド・コードは、ビジネス・オペレーションのプロパティおよび Adapter プロパティのメソッドを参照します。

  メッセージ・クラスの定義方法は、"プロダクションの開発" の "メッセージの定義" を参照してください。

  メッセージ・ハンドラ・メソッドの定義方法は、このページで後述する "メッセージ・ハンドラ・メソッドの作成" を参照してください。

- クラスは OnConnect() コールバック・メソッドを実装できます。このメソッドは、TCP 送信アダプタがリモート・システムとの新しい接続を確立するたびに呼び出されます。

  OnConnect() メソッドを実装する場合は、以下のシグニチャを付与する必要があります。

```
Method OnConnect(pTimeout As %Numeric) As %Status
```

新しい接続が確立されるたびに何らかの処理 (ログオン・シーケンスやハンドシェーク・トークンの送信など) を実行する必要がある場合は、このメソッドを実装します。timeout 引数は、TCP 送信アダプタによって自動的に指定されます。この引数の値は、**ConnectTimeout** アダプタ設定から取得されます。詳細は、"TCP 送信アダプタに関する設定" を参照してください。

OnConnect() がエラー・ステータスを返した場合は、新しい接続が失敗して、アダプタが切り離されます。トラップされない例外が OnConnect() 内で発生した場合、アダプタはその例外をキャッチして、OnConnect() が実装されていなかった場合と同じように処理を継続します。

- その他のオプションと一般情報は、"プロダクションの開発" の "ビジネス・オペレーション・クラスの定義" を参照してください。

以下の例は、必要となる一般的な構造を示しています。

Class Definition

```
Class ETCP.NewOperation1 Extends Ens.BusinessOperation
{
Parameter ADAPTER = "EnsLib.TCP.CountedOutboundAdapter";

Parameter INVOCATION = "Queue";

Method Sample(pReq As RequestClass, Output pResp As ResponseClass) As %Status
{
  Quit $$$ERROR($$$NotImplemented)
}

XData MessageMap
{
<MapItems>
  <MapItem MessageType="RequestClass">
    <Method>Sample</Method>
  </MapItem>
</MapItems>
}
}
```

注釈　スタジオには、上記のようなビジネス・オペレーション・スタブの作成に使用できるウィザードが用意されています。このウィザードにアクセスするには、**[ファイル]→[新規作成]** をクリックし、**[プロダクション]** タブをクリックします。次に **[ビジネス・オペレーション]** をクリックして [OK] をクリックします。

2.4 メッセージ・ハンドラ・メソッドの作成

TCP 送信アダプタで使用するビジネス・オペレーション・クラスを作成する場合の最大のタスクは、このアダプタで使用されるメッセージ・ハンドラ、つまり、プロダクション・メッセージを受信してファイルを書き込むメソッドの作成です。

各メッセージ・ハンドラ・メソッドは、以下のシグニチャを持っている必要があります。

```
Method Sample(pReq As RequestClass, Output pResp As ResponseClass) As %Status
```

ここで、Sample はメソッド名、RequestClass は要求メッセージ・クラスの名前、ResponseClass は応答メッセージ・クラスの名前です。

通常、このメソッドは以下の操作を実行します。

1. 受信要求メッセージを調べます。
2. 受信要求の情報を使用して、ビジネス・オペレーションの **Adapter** プロパティのメソッドを呼び出します。例えば、Connect() メソッドを呼び出す行は次のようになります。

```
set sc=..Adapter.Connect()
```

“ビジネス・オペレーションからのアダプタ・メソッドの呼び出し”を参照してください。

3. 必ず出力引数 (pOutput) を設定します。通常、応答メッセージと同じように設定します。この手順は必須です。
4. 適切なステータスを返します。この手順は必須です。

2.4.1 ビジネス・オペレーションからのアダプタ・メソッドの呼び出し

Connect()

```
Method Connect(pTimeout As %Numeric) As %Status
```

すべての TCP 送信アダプタに適用されます。

Connect() は、TCP リスナ・マシン上のポートに対して TCP ソケットを開きます。Connect() には、接続試行が成功するのを待つ秒数を指定する %Numeric という 1 つの入力引数があります。Connect() メソッドの一般的な呼び出しでは、構成可能な ConnectTimeout プロパティの値がこの入力引数の値として使用されます。

Stay Connected アダプタ設定が -1 に設定されている場合は、InterSystems IRIS の起動時に、OnInit() メソッド内から自動的に接続が確立されます。“設定の参照先”を参照してください。

Disconnect()

```
Method Disconnect()
```

すべての TCP 送信アダプタに適用されます。

Disconnect() は TCP リスナへの接続を解除します。

SendMessageStream()

```
Method SendMessageStream(pRequestStream As %Stream.Object,
  ByRef pResponseStream As %CharacterStream = "%GlobalCharacterStream")
  As %Status
```

EnsLib.TCP.CountedOutboundAdapter に適用されます。

SendMessageStream() は、ストリーム・コンテンツを TCP ソケット上のカウントされたブロックとして送信します。

SendMessageString()

```
Method SendMessageString(pRequestString As %String,
                Output pResponseString As %String) As %Status
```

EnsLib.TCP.CountedOutboundAdapter と EnsLib.TCP.TextLineOutboundAdapter に適用されます。

SendMessageString() は、文字列コンテンツを TCP ソケット上のカウントされたブロックとして送信します。

SendMessageXMLObj()

```
Method SendMessageXMLObj(pRequest As %RegisteredObject,
                Output pResponse As %RegisteredObject) As %Status
```

EnsLib.TCP.CountedXMLOutboundAdapter に適用されます。

SendMessageXMLObj() は、要求オブジェクトをカウントされた XML ブロックとして TCP リスナに送信します。Get Reply アダプタ設定が真 (True) の場合は、SendMessageXMLObj() が TCP リスナからの応答をカウントされた XML ブロックとして取得します。これは、InterSystems IRIS オブジェクトとしてインポートされます。Get Replyの詳細は、“設定の参照先”を参照してください。

TestConnection()

```
Method TestConnection()
```

すべての TCP 送信アダプタに適用されます。

TestConnection() は、接続状態を反映しているプロパティを修正します。アダプタが接続されている場合は、TestConnection() は True を返します。接続されていない場合は、TestConnection() は Disconnect() を呼び出して、False を返します。

2.5 EnsLib.TCP.CountedOutboundAdapter の例

EnsLib.TCP.CountedOutboundAdapter を参照するビジネス・オペレーション・クラスの例を以下に示します。

Class Definition

```
Class Test.GeneralTCPOperation Extends Ens.BusinessOperation
{
 Parameter ADAPTER = "EnsLib.TCP.CountedOutboundAdapter";
 Parameter MAXREAD=30000;

 Method OnMessage(pReq As Test.GeneralSendRequest,
                  Output pResponse As Test.GeneralSendResponse) As %Status
 {
  Set str=
    pReq.Hr_pReq.DataLth_pReq.BID_pReq.Hr2_pReq.HrVacant_pReq.Cat_pReq.Hr3_pReq.Hr4
  Set tTextStream= ##class(%GlobalCharacterStream).%New()
  Set tSC=tTextStream.Write(str)
  Do pReq.Body2.Read(32) // Throw it away
  Set len=..#MAXREAD
  Set token= pReq.Body2.Read(.len)
  Set i=0
  While(len>0) {
     Set i=i+1
     Do tTextStream.Write(token)
     Set len=..#MAXREAD
     Set token= pReq.Body2.Read(.len)
  }
  Set tSC=
  ..Adapter.SendMessageStream(tTextStream,.tStreamReply) Quit:$$$ISERR(tSC) tSC
  Set pResponse=##class(Test.GeneralSendResponse).%New()
  Do tStreamReply.Rewind()
  Set pResponse.Body = tStreamReply.Read(,.tSC)
  Quit tSC
 }
}
```

2.6 ビジネス・オペレーションの追加と構成

ビジネス・オペレーションをプロダクションに追加するには、管理ポータルを使用して以下の操作を行います。

1. ビジネス・オペレーション・クラスのインスタンスをプロダクションに追加します。
2. ビジネス・オペレーションを構成します。設定の詳細は、“設定の参照先”を参照してください。
3. ビジネス・オペレーションを有効化します。
4. プロダクションを実行します。

3

特殊なトピック

このページでは、TCP アダプタに関連する追加のトピックを取り上げます。

アダプタのコールバック・メソッドのカスタマイズに関する一般情報は、"プロダクションの開発" の "あまり一般的ではないタスク" を参照してください。

Tipヒント InterSystems IRIS® では、TCP アダプタを使用する特殊なビジネス・サービス・クラスとユーザのニーズに適したビジネス・サービス・クラスの 1 つも提供されます。そのため、プログラミングの必要がありません。"相互運用プロダクションの概要" の "接続オプション" を参照してください。

3.1 TCP ステータス・サービスの追加

EnsLib.TCP.StatusService クラスは、受信テキスト文字列を解析して、要求されているプロダクション・ステータス情報のタイプを特定してから、コンソール画面への出力に適した応答文字列を生成します。ユーザは、ステータス・サービスと直接対話したり、ステータス・サービスと通信を行い、コマンドを発し、その後の分析用テキスト・ファイルへの応答を記述するなどのコマンド・スクリプトを記述できます。

開発者は、以下のようにして、ステータス・サービス経由での対話が可能です。

1. EnsLib.TCP.StatusService のインスタンスをプロダクションに追加します。
2. EnsLib.TCP.StatusService のインスタンスを構成して、特定の TCP ポート経由の通信を受け入れます。接続を受け入れる Port 番号と Allowed IP Addresses のリストを設定します。この手順は、ステータス・サービスに関連付けられている TCP テキスト行受信アダプタの構成に有効ないくつかの設定のうちの 2 つにすぎません。詳細は、"設定の参照先" を参照してください。

以上の手順を完了すると、いつでもステータス・サービスを有効化して実行し、ユーザまたはコマンド行スクリプトで Telnet 接続を開始して、Portを構成し、ステータス・サービスにコマンドを送信することができます。各コマンドは、新規行文字 (ASCII 10) で終了するテキスト文字列である必要があります。応答文字列も新規行で終了します。

以下の表は、サポートしているコマンド行と、状況に応じて返されるテキスト文字列の内容を示しています。

コマンド行	返されるテキスト文字列
build	InterSystems IRIS ソフトウェア・バージョンの完全名。

コマンド行	返されるテキスト文字列
configitemstatus name	現在実行中のプロダクション内のいずれかのビジネス・ホストの構成名を使用してこのコマンドを入力すると、ステータス・サービスは、そのビジネス・ホストの現在の状態を表す文字列を返します。 現在、ジョブを実行中のホスト、またはアクティブになっている接続は、返す文字列にそのアクティビティを記録します。識別されたホストが、現在実行中のプロダクションのメンバではない場合、返す文字列にそのことを示します。
exit	文字列は返されません。このコマンドはステータス・サービスを切断します。exit の代わりに x を入力することもできます。
localstarttime	InterSystems IRIS が実行中の場合、現在実行中のプロダクションの開始時刻を現地時間で返します。Ensemble が実行中ではない場合、<UNDEFINED> というメッセージの文字列を返します。
localtime	現地時間による現在時刻です。
namespace	現在実行中のプロダクションが格納されている相互運用対応ネームスペースです。
production	現在実行中のプロダクションの構成名です。
quit	文字列は返されません。このコマンドはステータス・サービスを切断します。quit の代わりに q を入力することもできます。
utcstarttime	InterSystems IRIS が実行中の場合、現在実行中のプロダクションの開始時刻を協定世界時 (UTC) で返します。Ensemble が実行中ではない場合、<UNDEFINED> というメッセージの文字列を返します。
utctime	協定世界時 (UTC) による現在時刻です。
version	InterSystems IRIS ソフトウェア・バージョンの略名です。例: 2018.1.0.514/2171
x	exit と同様です。

3.2 カスタム TCP アダプタ・クラスの作成

TCP アダプタ・クラスのカスタム・サブクラスを作成するには、スタジオで次の手順を実行します。

1. [ファイル] メニューで [新規作成] をクリックしてから [全般] タブをクリックします。
2. [InterSystems IRIS クラス定義]→[OK] の順にクリックして、新しいクラス・ウィザードを開始します。
   a. パッケージとクラス名を入力し、[次へ] をクリックします。

      重要　予約パッケージ名は使用しないでください。“プロダクションの開発” の ″予約パッケージ名″ を参照してください。

   b. [クラスの種類] で、[拡張] をクリックします。
   c. [スーパー・クラス名] の横にある [ブラウズ] をクリックし、サブクラス化する必要のあるクラスに移動します。
   d. [OK] をクリックします。
   e. [終了] をクリックします。

結果は次のようなテンプレート・クラスになりますが、選択したクラスによって異なります。

Class Definition

```
Class MyTest.NewClass1 Extends EnsLib.TCP.InboundAdapter
{

}
```

3. プロパティ、メソッド、クラス・パラメータ、またはクラスまたはその親クラスから継承したその他のクラス・メンバをすべてオーバーライドできます。あるいは、新しいクラス・メンバを追加することもできます。唯一の要件は以下のとおりです。
   - 受信アダプタ・クラスは、OnConnected() メソッドを実装する必要があります。

     実装には、クラスによって継承されるヘルパー・メソッドを使用します。
   - 送信アダプタ・クラスは、TCP 接続の作成、接続の読み込みまたは書き込み、切断を実行するメソッドを定義します。

     これらのメソッドを定義するには、クラスによって継承されるヘルパー・メソッドを使用します。
4. このクラスをコンパイルします。そこでこのクラスが保存されます。

3.3 TCP アダプタ・サブクラスの共通カスタマイズ

以下のリストに、アダプタが必要とするメソッドの実装に加えて、TCP アダプタ・サブクラスの共通カスタマイズをいくつか提案します。

- Terminators プロパティの InitialExpressions に関するさまざまな値を定義できます。

  ASCII 文字を指定するには、ObjectScript 関数の $CHAR (短縮形 : $C) を使用します。
  EnsLib.TCP.TextLineInboundAdapter の以下の例では、Terminators を新規行文字 (ASCII 10) に設定しています。

  Class Member

  ```
  Property Terminators As %String [ InitialExpression = {$C(10)} ];
  ```

  $CHAR などの関数の詳細は、"ObjectScript リファレンス" の “ObjectScript 関数” を参照してください。

  Terminators には、1 つの文字のほか、複数文字の文字列も設定できます。Terminators 値に文字列を指定した場合、InterSystems IRIS は文字列を以下のように使用します。
  - 文字列のいずれか 1 つの文字が入力の終了を実行します。
  - 文字列のすべての文字が出力に付記されます。
- プロパティとメソッドを追加することができます。
- 設定を追加したり、削除したりできます。"プロダクションの開発" の “InterSystems IRIS のプログラミング” のドキュメントで “設定の追加と削除” を参照してください。
- 接続にログイン資格情報を必要とする場合は、プロパティ名 `Credentials` を `SETTINGS` のリストに追加してください。`Credentials` プロパティは、基本クラス Ens.Adapter で %String として既に定義されています。

- (受信アダプタの場合) パラメータ SINGLEPOOLJOB を 1 に設定して、プール・サイズを強制的に 1 にすることもできます。

  ```
  /// Force a single listener job regardless of PoolSize setting
  Parameter SINGLEPOOLJOB = 1;
  ```

  このアダプタ・クラスの任意のサブクラスまたはこのようなアダプタ・クラスを使用するビジネス・サービス・クラスのサブクラスでは、そのクラス・コードでこのパラメータを使用してプール・サイズを詳細に制御できます。

- OnInit() コールバック・メソッドを実装すると、特殊な設定アクションの実行や、任意の構造の初期化が可能になります。

  受信アダプタでは、これらのアクションのいずれかで、設定を検査してアダプタとそれがリッスンする TCP クライアント間の接続を初期化したり、アダプタが使用するすべてのオブジェクト・プロパティを作成したりします。

  送信アダプタでは、これらのアクションのいずれかで、設定を検査して TCP リスナ上のポートへのソケットを開きます。

- 受信アダプタでは、OnConnected() コールバック・メソッドを実装できます。

  構成済み TCP クライアントへの接続が存在する場合は常に、アダプタは OnConnected() を呼び出して TCP データ・ソースからストリームを読み取ります。この読み取り操作は、アダプタの設定によって制御されます。

  OnConnected() は、ヘルパー・メソッドを使用してデータを解析します。

  データを正常に読み取ったら、OnConnected() は、プロダクション・フレームワークの ProcessInput() メソッドを呼び出して、受信したデータ・ストリームを関連付けられたビジネス・サービスに渡します。この呼び出しによって送信ストリームが返された場合は、OnConnected() は、そのデータを TCP クライアントに対する応答として書き込みます。

  CallInterval の期間中に TCP 接続からデータを読み取ることができなかった場合は、OnConnected() は何も実行せずに復帰します。

TCP アダプタ設定

このセクションでは、TCP アダプタの参照情報を提供します。

″プロダクションの管理″ の “すべてのプロダクションに含まれる設定” も参照してください。

TCP 受信アダプタに関する設定

TCP 受信アダプタの設定に関する参照情報を提供します。

概要

TCP 受信アダプタには以下の設定があります。

グループ	設定
基本設定	[ポート]、[呼び出し間隔]
接続設定	[接続毎のジョブ]、[許可IPアドレス]、[OS接続受け付けキューサイズ] (QSize)、[接続を維持]、[読み込みタイムアウト]、[SSL構成]、[ローカル・インターフェース]
追加設定	[受け入れクラス名]、[文字セット]

残りの設定はすべてのビジネス・サービスに共通のものです。詳細は、"プロダクションの構成" の "すべてのビジネス・サービスに含まれる設定" を参照してください。

受け入れクラス名

EnsLib.TCP.CountedXMLInboundAdapter に適用されます。

受信した XML ブロックを基にこのアダプタがインスタンス化するクラスの名前を指定するカンマ区切りのリストです。

許可IPアドレス

すべての TCP 受信アダプタに適用されます。

接続を受け入れるリモート IP アドレスのカンマ区切りのリストです。アダプタは、IPV4 アドレス、IPV6 アドレス、またはドメイン・ホスト・コントローラが解決できるサーバ名を受け入れます。オプションとして、port の指定がサポートされています。例えば、192.168.1.22 または 192.168.1.22:3298 のアドレス形式は両方とも受け入れられます。

注釈　IP アドレス・フィルタリングは、一般アクセスできるシステムではなくプライベート・ネットワーク上のアクセスを制御するための手段です。IP アドレス・フィルタリングを唯一のセキュリティ・メカニズムとして利用することは推奨されません。攻撃者は IP アドレスをスプーフィング (偽装) できるからです。

ポート番号を指定すると、その他のポートからの接続は拒否されます。

この文字列の先頭に感嘆符 (!) が付加されている場合、受信アダプタは、受信接続要求を待つことなく接続を開始します。受信アダプタは指定されたアドレスへの接続を開始し、次にメッセージを待機します。この場合、指定されるアドレスは 1 つだけで、ポートが指定された場合は Port 設定の値より優先されます。それ以外の場合は Port 設定が使用されます。

呼び出し間隔

すべての TCP 受信アダプタに適用されます。

アダプタが構成されたソースからの受信データをリッスンする秒数です。この秒数を超えると、プロダクション・フレームワークからのシャットダウン信号を確認します。

アダプタは、入力を検出すると、データを取得してビジネス・サービスに渡します。ビジネス・サービスはデータを処理し、アダプタは直ちに新規入力の待機を開始します。プロダクションが実行中であり、ビジネス・サービスが有効化され、アクティブになるようにスケジュールされている場合、このサイクルは常に継続されます。

CallInterval のデフォルト値は 5 秒です。最小値は 0.1 秒です。

文字セット

EnsLib.TCP.CountedInboundAdapter、EnsLib.TCP.CountedXMLInboundAdapter、および EnsLib.TCP.TextLineInboundAdapter に適用されます。

受信データの文字セットを指定します。InterSystems IRIS® は、自動的に、文字をこの文字エンコーディングから変換します。この設定値は大文字と小文字が区別されません。`Binary` は、バイナリ・ファイル、新規行文字と改行文字が異なるデータ、または変更しないまま残す必要のあるデータに対して使用します。テキスト・ドキュメントを転送するときは、他の設定が便利な場合があります。選択肢は以下のとおりです。

- `Binary` ― バイナリ転送 (デフォルト)
- `Ascii` ― 文字エンコーディング変換を伴わない Ascii モード FTP 転送
- `Default` ― ローカル InterSystems IRIS サーバのデフォルトの文字エンコード
- `Latin1` ― ISO Latin1 8 ビット・エンコード
- `ISO-8859-1` ― ISO Latin1 8 ビット・エンコード
- `UTF-8` ― Unicode 8 ビット・エンコード
- `UCS2` ― Unicode 16 ビット・エンコード
- `UCS2-BE` ― Unicode 16 ビット・エンコード (ビッグ・エンディアン)
- InterSystems IRIS に NLS (各国言語サポート) をインストールするための、国際文字エンコード規格に基づくその他のエイリアス

InterSystems IRIS の文字変換に関する背景情報は、"サーバ側プログラミングの入門ガイド" の "ローカライズのサポート" を参照してください。

エンディアン

EnsLib.TCP.CountedInboundAdapter と EnsLib.TCP.CountedXMLInboundAdapter に適用されます。

選択項目 [ ] または [ ] は、4 バイト・ブロック・カウントの接頭語のバイト順を示します。[ ] エンディアンは、最上位バイト (MSB) を最初に回線上に送ることを意味し、[ ] エンディアンは、最下位バイト (LSB) を最初に回線上に送ることを意味します。この文字列のデフォルト値は[ ]です。

接続毎のジョブ

すべての TCP 受信アダプタに適用されます。

真のとき、アダプタは受信 TCP 接続ごとに処理を行う新しいジョブを生成して、複数の接続の同時処理を可能にします。偽の場合、各接続に対する新しいジョブは生成されません。X12 接続には、通常は偽を選択するのが適切です。デフォルトは真です。

TCP サービスに対して [JobPerConnection] が真の場合、それぞれの新しい受信ソケット接続は、リスナ・ジョブ自体ではなく新たに生成されたジョブによって処理されるようになります。ただし、一度にリスナになれるのは 1 つのジョブのみであり、なおかつ 1 つのジョブがリスナになっている必要があります。このため、TCP サービスの [**プール・サイズ**] に 2 以上の値が指定されていても、1 つのリスナ・ジョブしか開始されません。しかし、[JobPerConnection] を真に設定すると、このリスナで生成できる接続ジョブの数は無制限になります。

[**プール・サイズ**] 設定を 2 以上に構成すると、同時に存在可能な接続ジョブの数を制限できます。この制限に達すると、既存の接続ジョブが終了するか停止するまで、リスナは接続を受け入れなくなります。イベント・ログの警告メッセージは、この制限に最初に達した時点で表示されます。

ローカル・インタフェース

すべての TCP 受信アダプタに適用されます。

TCP 接続に必要なネットワーク・インタフェースを指定します。リストから値を選択するか、値を入力してください。空の値は、任意のインタフェースが使用できることを意味します。

OS接続受け付けキューサイズ (QSize)

すべての TCP 受信アダプタに適用されます。

このビジネス・サービスのために、オペレーティング・システムで予約しておく必要のある受信接続数を指定します。同時に発生すると想定される接続が 1 つのみで、以降の接続がオペレーティング・システムによって直ちに拒否される場合は 0 に設定します。多数のクライアントが次々と接続する場合には大きな数値を設定します。デフォルトは 100 です。0 ～ 1000 の範囲で設定できますが、最大受信接続数は TCP の実装によって異なります。

注釈　0 に設定した場合は、ビジネス・サービスで処理中の受信接続があると、接続しようとするクライアントは接続を拒否されます。これにより、ビジネス・サービスに接続しているクライアントが接続を解除され、リッスンしているソケットがまだ解放されていない短時間のうちに再接続しようとする状況になることが考えられます。その結果、クライアントは再接続しようとしなくなります。

HL7 と X12 の TCP 受信アダプタの場合は、この既定値が 0 に設定されます。これにより、一度に想定される接続が 1 つのみである、HL7 の先入れ先出し (FIFO) 方針がサポートされます。

ポート

すべての TCP 受信アダプタに適用されます。

アダプタが TCP 要求をリッスンしているローカル・マシン上の TCP ポートを指定します。オペレーティング・システムで一時的な送信接続用に使用されるポート範囲内のポート番号を指定することは避けてください。

読込タイムアウト

すべての TCP アダプタに適用されます。

リモート TCP ポートから初期データを受信した後、次に続く受信 TCP 読み取りオペレーションを待機する秒数です。デフォルトは 5 秒です。指定できる値の範囲は 0 ～ 600 秒 (最大 10 分) です。

SSLConfig

すべての TCP アダプタに適用されます。

この接続の認証に使用する既存の TLS 構成の名前。アダプタから通信が開始されるため、クライアント TLS 構成を選択します。

TLS 構成を作成して管理するには、管理ポータルを使用します。インターシステムズの "TLS ガイド" を参照してください。[SSL/TLS構成を編集] ページの最初のフィールドは [構成名] です。この文字列を [SSL構成] の値として使用します。

接続を維持

すべての TCP アダプタに適用されます。

入力イベント後にアダプタがどれだけの間リモート・システムとの接続を維持するかを指定し、アダプタによる切断処理方法を指定します。

以下のいずれかの値を指定できます。

- 正の値—アダプタは、ここで指定された秒数だけリモート・システムとの接続を維持します。切断はエラーのようには処理されません。
- 0—アダプタは直ちに切断します。切断はエラーのようには処理されません。
- -1—アダプタは、アイドル・タイムも含め、永久的に接続されたままとなります。切断はエラーのように処理されます。

注釈　アダプタは、**[接続を維持]** の値が -1 の場合のみ、起動時に自動接続されます。

デフォルト値は -1 です。

重要　**[接続を維持]** を **-1** に、[接続毎のジョブ] を **False** に設定すると、アダプタは、ネットワークでの事象によって生じた切断を検出しない場合があります。具体的には、以下のような状況が発生する可能性があります。

1. アダプタがリモート・システムからの受信接続を受け入れます。

   ここで、**[接続毎のジョブ]** が **False** に設定されていると、アダプタ上のリスナ・ジョブが各受信接続を処理する (この処理を行う新しいジョブを生成するのではなく) ことを思い出してください。

2. ネットワークでの事象により切断が発生し、アダプタは接続を終了するための TCP FIN または RST パケットを受け取りません。

   例えば、リモート・システムとアダプタとの間のファイアウォールが、一定の時間の経過後に接続を閉じることがあります。

3. リモート・システムは接続を再確立しようとしますが、アダプタ上の待ち受けソケットに接続できません。

4. リモート・システムはエラーを生成し、引き続き接続の再確立を試みます。

この状況を修復するには、そのアダプタを使用しているビジネス・サービスを停止してから再起動する必要があります。

この状況を回避するには、**[接続を維持]** を **-1** 以外の値に設定します。例えば、**[接続を維持]** を **300** に設定すると、アダプタは 5 分後に切断します。さらに、アダプタが要求を順に処理する必要がない場合は、**[接続毎のジョブ]** を **True** に設定し、アダプタが受け取った複数の受信接続要求を一度に処理できるようにします。

TCP 送信アダプタに関する設定

TCP 送信アダプタの設定に関する参照情報を提供します。

概要

TCP 送信アダプタには以下の設定があります。

グループ	設定
基本設定	[IPアドレス]、[ポート]
接続設定	[接続を維持]、[接続タイムアウト]、[再接続試行]、[準備完了]、[応答タイムアウト]、[読込タイムアウト]、[SSL構成]、[ローカル・インタフェース]、[文字セット]、[エンディアン]
追加設定	[文字セット]

残りの設定はすべてのビジネス・オペレーションに共通のものです。詳細は、"プロダクションの構成" の "すべてのビジネス・オペレーションに含まれる設定" を参照してください。

文字セット

EnsLib.TCP.CountedOutboundAdapter、EnsLib.TCP.CountedXMLOutboundAdapter、および EnsLib.TCP.TextLineOutboundAdapter に適用されます。

送信データに適した文字セットを指定します。InterSystems IRIS® は、自動的に、文字をこの文字エンコーディングに変換します。"TCP 受信アダプタに関する設定" の "文字セット" を参照してください。

接続タイムアウト

すべての TCP 送信アダプタに適用されます。

リモート TCP リスナへの接続試行を待機する秒数です。デフォルトは 5 秒です。

エンディアン

EnsLib.TCP.CountedOutboundAdapter と EnsLib.TCP.CountedXMLOutboundAdapter に適用されます。

選択項目 [ ] または [ ] は、4 バイト・ブロック・カウントの接頭語のバイト順を示します。[ ] エンディアンは、最上位バイト (MSB) を最初に回線上に送ることを意味し、[ ] エンディアンは、最下位バイト (LSB) を最初に回線上に送ることを意味します。この文字列のデフォルト値は[ ]です。

準備完了

すべての TCP 送信アダプタに適用されます。

1 (真) のとき、ソケットから返される応答メッセージを読み取るまで待機してから、返します。0 (偽) のときは、待機しません。デフォルト値は 1 です。

IPアドレス

すべての TCP 送信アダプタに適用されます。

TCP 接続先となる IP アドレス。アダプタは、IPV4 アドレス、IPV6 アドレス、またはドメイン・ホスト・コントローラが解決できるサーバ名を受け入れます。

この文字列の先頭に感嘆符 (!) を記述しておくと、送信アダプタは接続を開始せずに、受信接続要求を待ちます。接続要求を承認すると、送信メッセージを送ることができます。感嘆符を単独で指定することも、感嘆符の後ろに IP アドレスのカンマ区切りリストを指定することもできます。感嘆符の後ろに IP アドレスを指定しない場合は、送信 TCP アダプタは

どの IP アドレスからの接続も受け付けます。IP アドレスを指定した場合は、アダプタはその指定アドレスからの接続のみを受け付けます。アダプタは、IPV4 アドレス、IPV6 アドレス、またはドメイン・ホスト・コントローラが解決できるサーバ名を受け入れます。必要に応じて IP アドレスの後ろにコロン (:) とポート番号を記述することで、各 IP アドレス用のポートを指定することもできます。ポートを指定した場合は、アダプタは指定したポートからの接続のみを受け付けます。

注釈　IP アドレス・フィルタリングは、一般アクセスできるシステムではなくプライベート・ネットワーク上のアクセスを制御するための手段です。IP アドレス・フィルタリングを唯一のセキュリティ・メカニズムとして利用することは推奨されません。攻撃者は IP アドレスをスプーフィング (偽装) できるからです。

LocalInterface

すべての TCP 送信アダプタに適用されます。

TCP 接続に必要なネットワーク・インタフェースを指定します。リストから値を選択するか、値を入力してください。空の値は、任意のインタフェースが使用できることを意味します。詳細は、“TCP 受信アダプタに関する設定”の“ローカル・インタフェース”を参照してください。

ポート

すべての TCP 送信アダプタに適用されます。

接続先の TCP ポートを指定します。

QSize

EnsLib.TCP.CountedOutboundAdapter、EnsLib.TCP.CountedXMLOutboundAdapter、および EnsLib.TCP.TextLineOutboundAdapter に適用されます。

このビジネス・オペレーションのために、オペレーティング・システムで予約しておく必要のある受信接続数を指定します。通常、デフォルト値の 0 を変更する必要はありません。デフォルト値では、接続が送信用であるか、ビジネス・オペレーションのプール・ジョブ 1 つについて一度に接続が許可されるクライアントが 1 つのみの排他受信モードをビジネス・オペレーションで使用しています。

読込タイムアウト

すべての TCP アダプタに適用されます。

リモート TCP ポートから初期データを受信した後、次に続く受信 TCP 読み取りオペレーションを待機する秒数です。デフォルトは 5 秒です。指定できる値の範囲は 0 ～ 600 秒 (最大 10 分) です。

再接続試行

すべての TCP 送信アダプタに適用されます。

アダプタが接続を解除して再接続するまでにメッセージの送信を試行する回数。値 0 (ゼロ) は、アダプタが切断することなく永久にメッセージの送信を試行することを意味します。デフォルト値は 5 です。

応答タイムアウト

すべての TCP 送信アダプタに適用されます。

要求を送信後、リモート・システムから応答が返信されるのを待機する秒数。デフォルトは 15 秒です。

SSL構成

すべての TCP アダプタに適用されます。

この接続の認証に使用する既存の TLS 構成の名前。アダプタから通信が開始されるため、クライアント TLS 構成を選択します。詳細は、“TCP 受信アダプタに関する設定”の“SSL 設定”を参照してください。

接続を維持

すべての TCP アダプタに適用されます。

オペレーションの完了後に TCP アダプタがどれだけの間リモート・システムとの接続を維持するかを指定し、アダプタによる切断処理方法を指定します。

以下のいずれかの値を指定できます。

- 正の値ーアダプタは、ここで指定された秒数だけリモート・システムとの接続を維持します。切断はエラーのようには処理されません。
- 0ーアダプタは直ちに切断します。切断はエラーのようには処理されません。
- −1ーアダプタは、アイドル・タイムも含め、永久的に接続されたままとなります。

  注釈　アダプタは、**[接続を維持]** の値が −1 の場合のみ、起動時に自動接続されます。切断はエラーのように処理されます。

デフォルト値は −1 です。