EPSON
EXCEED YOUR VISION

EPSON RC+ 7.0 オプション

GUI Builder 7.0

Rev.8 JAM231S5539F

翻訳版

EPSON RC+ 7.0 オプション　GUI Builder 7.0 Rev.8

EPSON RC+ 7.0 オプション

GUI Builder 7.0

Rev.8

はじめに

このたびは当社のロボットシステムをお求めいただきましてありがとうございます。
本マニュアルは、マニピュレーターを正しくお使いいただくために必要な事項を記載したものです。
システムをご使用になる前に、本マニュアルおよび関連マニュアルをお読みいただき、正しくお使いください。
お読みになった後は、いつでも取り出せる所に保管し、不明な点があったら再読してください。

当社は、厳密な試験や検査を行い、当社のロボットシステムの性能が、当社規格に満足していることを確認しております。マニュアルに記載されている使用条件を超えて、当社ロボットシステムを使用した場合は、製品の基本性能は発揮されませんのでご注意ください。

本書の内容は、当社が予見する範囲の、危険やトラブルについて記載しています。当社のロボットシステムを、安全に正しくお使いいただくため、本書に記載されている安全に関するご注意は、必ず守ってください。

商標

Microsoft, Windows, Windowsロゴは、米国Microsoft Corporationの米国およびその他の国における登録商標または商標です。その他の社名，ブランド名，および製品名は、各社の登録商標または商標です。

表記について

Microsoft® Windows® 8 operating system 日本語版
Microsoft® Windows® 10 operating system 日本語版
Microsoft® Windows® 11 operating system 日本語版
本取扱説明書では、上記オペレーティングシステムをそれぞれ、Windows 8, Windows 10, Windows 11と表記しています。また、Windows 8, Windows 10, Windows 11を総称して、Windowsと表記することがあります。

ご注意

本書取扱説明書の一部、または全部を無断で複製、転載することはできません。
本書に記載の内容は、将来予告なく変更することがあります。
本書の内容について、万一誤り、お気づきの点がありましたら、ご連絡くださいますようお願いいたします。

製造元

セイコーエプソン株式会社

お問い合わせ先

お問い合わせ先の詳細は、以下のマニュアル冒頭 “販売元”に記載しています。

「ロボットシステム 安全マニュアル はじめにお読みください」

1. はじめに

GUI Builder 7.0 は、GUI(グラフィカルユーザーインターフェイス)を SPEL+アプリケーションに簡単に作成できる EPSON RC+オプションです。EPSON RC+開発環境で SPEL+アプリケーション GUI を作成するためにより使いやすくデザインされた統合ツールで、Visual Studio などのサードパーティー製品を使用せずに、シンプルな GUI を必要としているユーザーにとって、GUI Builder は理想的です。また、これまでに GUI を作成したことがなくても、GUI Builder を使用すれば簡単に GUI を作成することができます。

NOTE
より高度な GUI アプリケーションには、EPSON RC+ 7.0 RC+ API オプションを使用することで Visual Studio などのツールと連携したアプリケーションを開発できます。

1.1 特徴

GUI Builder 7.0 には、次のような特徴があります。

- GUI は、EPSON RC+環境に完全に統合されており、簡単デザイン, デバッグ, プログラム実行中の表示ができます。第三者ツールは必要ありません。
- EPSON RC+プロジェクト内で GUI フォームの作成、デバッグが可能です。
- ボタン、ラベル、テキストボックスなどの標準的なコントロールがあります。
  さらに、ビジョン画像の表示や、コントローラーの状態表示、I/O 状態表示も可能です。
- フォームとコントロールのイベントは、SPEL+タスクとして作成されます。これらのタスクには、Normal、NoPause、または NoEmgAbort モードの指定が可能です。
- オペレーターモードでは、EPSON RC+が起動時にユーザーのメインフォームを自動表示します。また、SPEL+プログラムからフォームを表示させることもできます。

2. インストール

本章の指示にしたがって、GUI Builder 7.0 をインストールしてください。

インストールの前に、すべての Windows アプリケーションを終了してください。

GUI Builder のインストール:

1. EPSON RC+ 7.0 をインストールします。GUI Builder 7.0 オプションは、自動的にインストールされます。
2. GUI Builder 7.0 を使用する前に、コントローラーでソフトウェアキーの設定を有効にしておきます。オプションを使用可能にする方法は、「EPSON RC+7.0 ユーザーズガイド」を参照してください。

これで、GUI Builder 7.0 のインストールは終了です。

3. ご使用の前に

GUI Builder 7.0 のご使用の前に、以下の本章をお読みください。

GUI Builder 7.0 オプションが有効になっていることを確認してください。詳細は、「2. インストール」を参照してください。

EPSON RC+ を初めてご使用になる方は、「EPSON RC+ 7.0 ユーザーズ ガイド-プロジェクトとプログラムの作成方法について」をお読みください。

次の項では、いくつかのコンセプトを簡単に説明するチュートリアルを示します。

3.1 GUI Builder チュートリアル

この項では、ロボットのサイクル動作を実行する、簡単な GUI アプリケーションを作成します。次の手順にしたがってください。

- ロボットのサイクル動作を行うファンクションを持つ新しいプロジェクトを作成します。
- ロボットのサイクル動作を開始、停止するためのボタンを含むフォームを作成します。ここでは、フォームの作成や、開始／停止イベントを実行する SPEL$^+$のタスクに関連づけられたボタンの配置方法を示します。
- <一時停止>と<継続実行>ボタンをフォームに追加します。ここでは EventTaskType プロパティーについて説明します。
- セットアップ用のフォームを追加します。このフォームは、ロボットの速度を変更するためのラベルと、テキストボックスを使います。
- セットアップ用のフォームを表示するためのボタンをメインフォームに配置します。ここでは、GShowDialog ステートメントと、DialogResult プロパティーを説明します。

それでは、次の手順にしたがって、プログラムを作成してください。

1 “GUITest”という名前の新しい EPSON RC+7.0 プロジェクトを作成します。

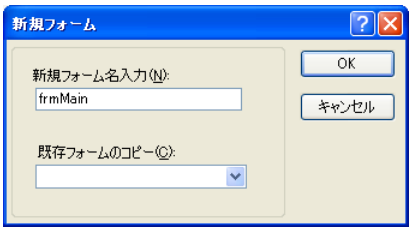
2. ロボットマネージャーを使用し、P0, P1 の 2 つの異なるポイントデータをティーチングします。

3. “Main.prg”に、“main 関数”を作成します。

```
Function main
  Robot1
  Motor On
  Do
    Go P0
    Wait 0.5
    Go P1
    Wait 0.5
  Loop
Fend
```

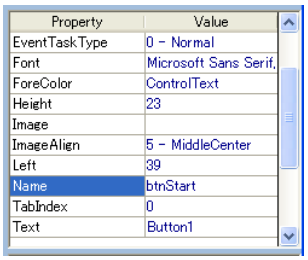
4. ツールメニューから[GUI Builder]を選択し、GUI Builder ウィンドウを表示します。

5. GUI Builder ウィンドウのツールバーにある<新規フォーム>ボタンをクリックし、“frmMain”という名前の新しいフォームを作成し、<OK>ボタンをクリックします。



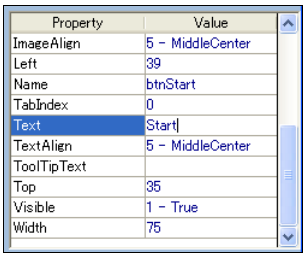
6. GUI Builder ウィンドウのツールバーにある<新規ボタン>ボタンをクリックし、フォーム上でクリックします。クリックした場所に、新しいボタンが作成されます。

7. 新しく作成されたボタンのプロパティーを設定します。
プロパティーのグリッドを下へスクロールして[Name]プロパティーを表示します。名前を“btnStart”に変更します。キーボードの<ENTER>キーを押します。



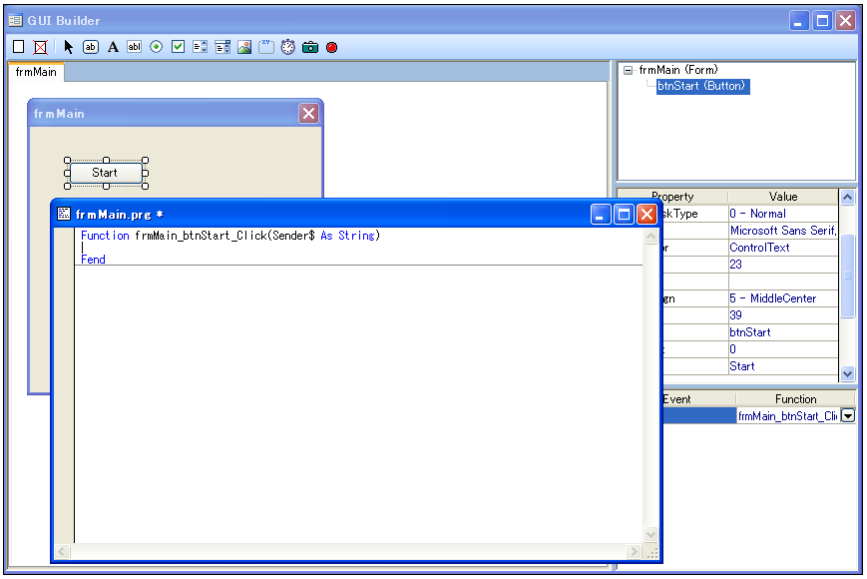
Property	Value
EventTaskType	0 - Normal
Font	Microsoft Sans Serif,
ForeColor	ControlText
Height	23
Image	
ImageAlign	5 - MiddleCenter
Left	39
Name	btnStart
TabIndex	0
Text	Button1

8. プロパティーのグリッドを下へスクロールして、[Text]プロパティーを表示します。“Button1”を“Start”に変更します。<ENTER>キーを押します。



Property	Value
ImageAlign	5 - MiddleCenter
Left	39
Name	btnStart
TabIndex	0
Text	Start
TextAlign	5 - MiddleCenter
ToolTipText	
Top	35
Visible	1 - True
Width	75

9. フォーム上の<開始>ボタンをダブルクリックします。“frmMain.prg”に、<開始>ボタンをクリックしたときに実行する新しい関数が作成されます。

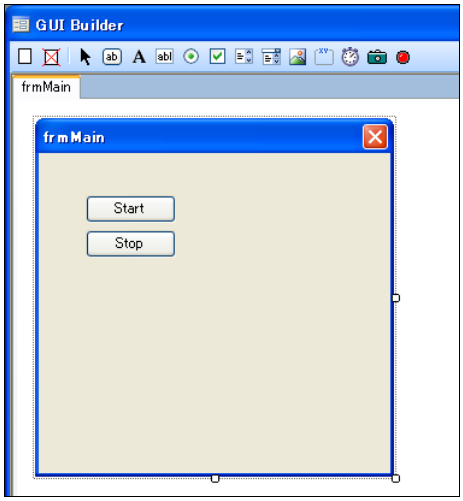


10. “main 関数”を起動させるため、次のように frmMain_btnStart_Click 関数を変更します。これによって、<開始>ボタンが押されたときに、“main 関数”がスタートします。

```
Function frmMain_btnStart_Click(Sender$ As String)
  Xqt main
Fend
```

11. GUI Builder ウィンドウのツールバーにある<新規ボタン>ボタンをクリックし、<開始>ボタンの下の位置をクリックして、別のボタンを作成します。

12. 新しいボタンの[Name]プロパティーを“btnStop”に、[Text]プロパティーを“Stop”に変更します。ここまでの手順で、次のようなフォームになっているはずです。



13. <停止>ボタンをダブルクリックして、イベントハンドラーを作成します。関数を次のように変更します。

```
Function frmMain_btnStop_Click(Sender$ As String)
  Quit main
Fend
```

14. キーボードの<F5>キーを押して、プロジェクトをビルドし、Run ウィンドウを表示します。ここでビルドエラーが発生した場合は、プログラムを修正して、再度<F5>キーを押してビルドを行ってください。

15. Run ウィンドウで<Form>ボタンを選択します。

16. <開始>ボタンをクリックします。

17. frmMain フォームが表示されます。[frmMain]の<開始>ボタンをクリックします。ロボットが、P0 と P1 の間を動作します。

18. [frmMain]フォームの<停止>ボタンをクリックします。ロボットが停止します。

19. 今度は、フォームの右上の<閉じる>ボタンをクリックします。フォームが閉じます。

次に、<一時停止>ボタンと<継続実行>ボタンを GUI に加えてみましょう。

20. ツールバーの<GUI Builder>ボタンをクリックして、もう一度 GUI Builder ウィンドウを開きます。

21. GUI Builder ウィンドウのツールバーにある<新規ボタン>ボタンをクリックします。<開始>ボタンの右側の位置をクリックし、新しいボタンを作成します。

22. 新しいボタンの Name プロパティーを btnPause に、Text プロパティーを Pause に変更します。

23. <一時停止>ボタンの EventTaskType プロパティーを 1 – NoPause に変更します。これにより、イベントハンドラーは他の通常タスクが一時停止しているときに一時停止せずに、Pause ステートメントを実行できます。

24 <一時停止>ボタンをダブルクリックして、イベントハンドラー関数を作成します。次のような Pause ステートメントを追加します。

```
Function frmMain_btnPause_Click(Sender$ As String)
  Pause
Fend
```

25. GUI Builder ウィンドウのツールバー-<新規ボタン>ボタンをクリックします。<停止>ボタンの右側の位置をクリックして、新しいボタンを作成します。

26. 新しいボタンの Name プロパティーを btnCont に、Text プロパティーを Continue に変更します。

27. <継続実行>ボタンの EventTaskType プロパティーを 1 – NoPause に変更します。これにより、イベントハンドラーは、他の通常タスクが一時停止しているときに、Cont ステートメントを実行できるようになります。

28. <継続実行>ボタンをダブルクリックして、イベントハンドラー関数を作成します。次に、次に示すように Cont ステートメントを追加します。

```
Function frmMain_btnCont_Click(Sender$ As String)
  Cont
Fend
```

29. キーボードの<F5>キーを押して、プロジェクトをビルドし、Run ウィンドウを表示します。ここでビルドエラーが発生した場合は、プログラムを修正して、再度 <F5>キーを押してビルドを行ってください。

30. Run ウィンドウの<開始>ボタンをクリックして、作成したフォームを表示します。

31. <開始>ボタンをクリックします。ロボットのサイクル動作が実行されます。

32. <一時停止>ボタンをクリックします。ロボットのサイクル動作が一時停止します。

33. <継続実行>ボタンをクリックします。ロボットのサイクル動作が再開します。

34. <停止>ボタンをクリックします。次にフォーム右上の<閉じる>ボタンをクリックして、フォームを閉じます。

次に、セットアップフォームを追加してみましょう。

35. GUI Builder ウィンドウのツールバーにある<新規フォーム>ボタンをクリックし、frmSetup という名前の新しいフォームを作成します。<OK>ボタンを押すと、新しいタブページに新しいフォームが表示されます。

36. <新規ラベル>ボタンをクリックします。[frmSetup]フォーム上をクリックして、新しいラベルを作成します。

37. Name プロパティーを lblSpeed に変更します。

38. Text プロパティーを Speed:に変更します。

39. <New TextBox>ボタンをクリックします。[frmSetup]フォームの[Speed]ラベルの右側をクリックして、新しいテキストボックスを作成します。

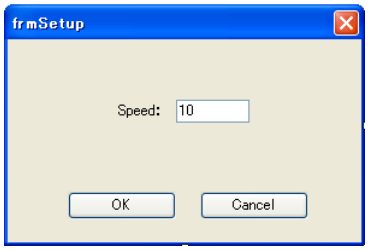
40. Name プロパティーを txtSpeed、Text プロパティーを 10 に変更します。

41. [frmSetup]フォームをダブルクリックすると、フォームの Load イベントハンドラー関数が作成されます。次のようにプログラムを追加してください。

```
Function frmSetup_Load(Sender$ As String)
  GSet frmSetup.txtSpeed.Text, Str$(Speed(1))
Fend
```

42. GUI Builder ウィンドウのツールバーにある<新規ボタン>ボタンをクリックして、新しいボタンを作成します。Name プロパティーを btnOK、Text プロパティーを OK に変更します。

43. GUI Builder ウィンドウのツールバーにある<新規ボタン>ボタンをクリックして新しいボタンを作成します。Name プロパティーを btnCancel、Text プロパティーを Cancel に変更します。この時点で、セットアップフォームは以下のようになります。



44. <OK>ボタンをダブルクリックしてイベントハンドラー関数を作成し、次のようにフォームの DialogResult プロパティーを設定するプログラムを追加します。

```
Function frmSetup_btnOK_Click(Sender$ As String)
  GSet frmSetup.DialogResult, DIALOGRESULT_OK
  GClose frmSetup
Fend
```

45. <Cancel>ボタンをダブルクリックしてイベントハンドラー関数を作成し、次のようにフォームの DialogResult プロパティーを設定するプログラムを追加します。

```
Function frmSetup_btnCancel_Click(Sender$ As String)
  GSet frmSetup.DialogResult, DIALOGRESULT_CANCEL
  GClose frmSetup
Fend
```

46. [frmMain]タブをクリックして、[frmMain]フォームを修正します。

47. GUI Builder ウィンドウのツールバーにある<新規ボタン>ボタンをクリックして新しいボタンを作成します。Name プロパティーを btnSetup、Text プロパティーを Setup に変更します。

48. <Setup>ボタンをダブルクリックしてイベントハンドラー関数を作成し、次のように[frmSetup]ダイアログを表示してロボットの動作速度を設定するプログラムを追加します。

```
Function frmMain_btnSetup_Click(Sender$ As String)
  Integer result
  String value$

  result = GShowDialog(frmSetup)
  If result = DIALOGRESULT_OK Then
    GGet frmSetup.txtSpeed.Text, value$
    Speed Val(value$)
  EndIf
Fend
```

49. <F5>キーを押してプロジェクトをビルドし、Run ウィンドウを表示します。

50. Run ウィンドウの<開始>ボタンをクリックし、作成したフォームを表示します。

51. [frmMain]フォームの<Setup>ボタンをクリックすると、現在のロボットの動作速度をテキストボックスに表示した[frmSetup]ダイアログを表示します。

52. 新しい速度を入力して、<OK>ボタンをクリックします。

53. [frmMain]フォームの<開始>ボタンをクリックすると、設定した動作速度で、ロボットのサイクル動作を実行します。<停止>ボタンをクリックしてロボットを停止すると、動作速度はデフォルトに戻ります。

54. これでこのチュートリアルは終了です。

4. GUI Builder 環境

4.1 概要

本章では、GUI Builder と構成要素を理解していただくために、次のコンセプトと定義について説明します。

- GUI Builder を理解するための基本コンセプト
- GUI Builder ウィンドウの開き方
- GUI Builder ウィンドウのそれぞれの機能
- フォームとコントロールの使い方
- GUI Builder の設定

4.2 GUI Builder を使用するための基本コンセプト

本章をより理解するために、簡単にいくつかの基本コンセプトを説明します。本章を読み進める前に確認してください。

GUI とは?

GUI(グイ)は、**G**raphical **U**ser **I**nterface(グラフィカルユーザーインターフェイス)の略です。SPEL+アプリケーションの実行や各設定などを、より簡単に対話的に行うことができます。GUI の基本部品は、“フォーム”と呼ばれます。

フォームとは?

フォームとは、コントロールを含むウィンドウ、またはダイアログボックスです。フォームが、GUI アプリケーションの基本ユニットになります。実行時にフォームが表示され、フォーム上のコントロールが有効になり、キーボードやマウス操作のイベントを受け取ります。GUI プロジェクトは、1 つまたはいくつかのフォームから構成されます。

コントロールとは?

コントロールとは、フォームに含まれるボタンや、チェックボックス、テキストボックスなどのオブジェクトです。各コントロールに固有のプロパティー、イベントがあります。

イベントとは?

イベントとは、フォームやコントロールのイベントが発生したときに GUI から呼び出される SPEL+関数です。例えばオペレーターがボタンをクリックしたとき、ボタンクリックイベントが、クリックされたときに指定された SPEL+関数を呼び出します。

4.3 GUI Builder ウィンドウの開き方

GUI Builder ウィンドウは、EPSON RC+の開発環境から開くことができます。EPSON RC+が起動している状態から、次の 2 つの方法で GUI Builder ウィンドウを開きます。

ツールバー: ツールバーに GUI Builder アイコン  があります。
GUI Builder アイコンをクリックすると、GUI Builder ウィンドウが開きます。

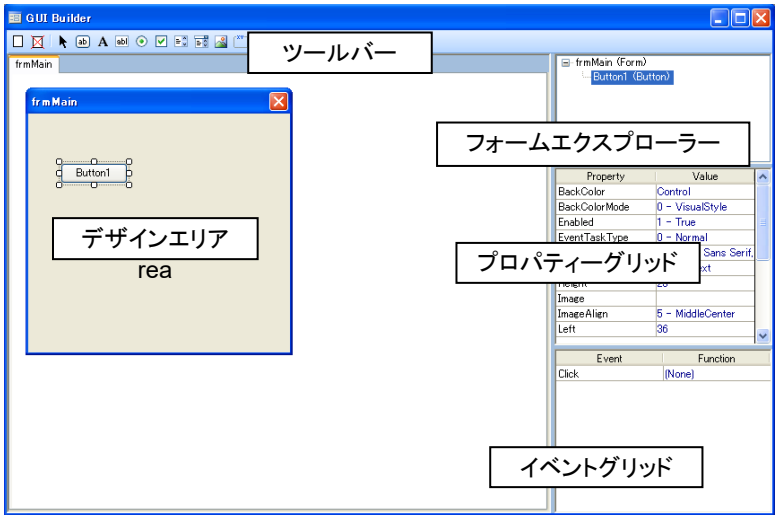
ツールメニュー: ツールメニューの GUI Builder から、GUI Builder ウィンドウを開くことができます。

GUI Builder ウィンドウを開くと、GUI アプリケーションが作成できるようになります。

次に、GUI Builder ウィンドウの各部分について説明していきます。

4.4 GUI Builder ウィンドウのそれぞれの機能

GUI Builder ウィンドウを以下に示します。各部分を次の項で説明します。



4.4.1 デザインエリア

設計時のフォームが表示される場所です。タブごとにそれぞれのフォームが表示されます。タブをクリックするか、フォームエクスプローラーでフォームの名前をダブルクリックすることで、表示されるフォームを切り替えることができます。フォームがデザインエリアより大きい場合、スクロールバーが表示され、フォームのすべてのエリアを操作することができます。

4.4.2 ツールバー

ツールバーには、フォームやコントロールの作成ボタンが含まれます。GUI Builder のツールバーは、GUI Builder ウィンドウの上部にあるタイトルバーのすぐ下に、次のように表示されます。

GUI Builder ツールバー

GUI Builder ツールバーの各ボタンについて説明します。

ボタン	説明
	**新規フォーム:** 新しいフォームを作成します。フォームの名前を入力するためのダイアログが表示されます。
	**フォーム削除:** 現在のプロジェクト内の選択しているフォームを削除します。フォームがない場合は、このボタンは無効になります。
	**ポインター:** 新しいコントロールの追加を中断して、通常の操作に戻します。
	**新規ボタン:** 新しいボタンコントロールを作成します。
	**新規ラベル:** 文字を表示するための、新しいラベルコントロールを作成します。
	**新規テキストボックス:** 文字を入力するための、新しいテキストボックスコントロールを作成します。
	**新規ラジオボタン:** 新しいラジオボタンを作成します。
	**新規チェックボックス:** 新しいチェックボックスコントロールを作成します。
	**新規リストボックス:** 新しいリストボックスコントロールを作成します。
	**新規コンボボックス:** 新しいコンボボックスコントロールを作成します。
	**新規グリッド:** 新しいグリッドコントロールを作成します。
	**新規ピクチャーボックス:** 新しいピクチャーボックスコントロールを作成します。
	**新規グループボックス:** 新しいグループボックスコントロールを作成します。
	**新規ステータスバー:** 新しいステータスバーコントロールを作成します。
	**新規プログレスバー:** 新しいプログレスバーコントロールを作成します。
	**新規トラックバー:** 新しいトラックバーコントロールを作成します。
	**新規タイマー:** 新しいタイマーコントロールを作成します。
	**新規ビデオボックス:** 新しいビデオボックスコントロールを作成します。このコントロールは、Vision Guide オプションの画像を表示します。
	**新規 LED:** 新しい LED コントロールを作成します。このコントロールは、I/O の状態を表示します。このコントロールをダブルクリックすることで、I/O の出力状態を変更するような使い方も可能です。表示色を変えたり、LED の代わりにイメージ画像を表示することも可能です。

4.4.3 フォームエクスプローラー

フォームエクスプローラーは、現在のプロジェクトのフォームや関連するコントロールをツリー表示します。新しいフォームやコントロールが追加されると、このツリーに加えられます。

ツリー上のフォームをダブルクリックすると、デザインエリアにフォームを開きます。選択されているフォームのプロパティーやイベントが、プロパティーグリッド、イベントグリッドに表示されます。フォーム上で右クリックすると、スタートアップフォームの設定、削除、閉じるのメニューが表示されます。

コントロールをクリックすると、関連するフォームが開き、選択したコントロールがフォーカスされます。コントロールのプロパティーとイベントは、プロパティーグリッド、イベントグリッドに表示されます。

4.4.4 プロパティーグリッド

プロパティーグリッドでは、フォームとコントロールのプロパティーを表示、編集します。フォームやコントロールを選択すると、そのプロパティーがグリッド上に表示されます。プロパティーグリッドには、Property と Value の 2 つの列があります。

Property ： プロパティーの名前
Value ： プロパティーの現在値(編集可能)

4.4.5 イベントグリッド

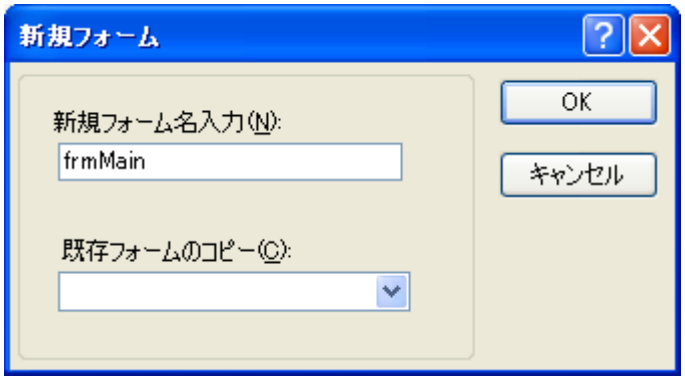
イベントグリッドでは、フォームとコントロールのイベントを表示、編集します。各イベントは、そのイベントで呼び出されるユーザー関数を持っています。

4.5 フォームとコントロールの使い方

4.5.1 フォームの作成

フォームを作成します。

1. GUI Builder ウィンドウを開きます。
2. ツールバーの<新規フォーム>ボタンをクリックします。



3. フォームの名前を入力します。名前の頭に“frm”を付けることをおすすめします。
[新規フォーム]ダイアログからは、フォームリストに表示される他のフォームからコピーしてフォームを作成することも可能です。

4.5.2 フォームの削除

フォームを削除するには、フォームエクスプローラー上で右クリックし、削除メニューを選択します。確認のメッセージが表示されたら、<はい>をクリックしてフォームを削除します。
また、次の方法でもフォームを削除できます。

- ツールバーにある<フォーム削除>ボタンをクリック
- フォームのタブを右クリック - <削除>メニューを選択

4.5.3 フォームの開閉

フォームを開くには、フォームエクスプローラーからフォームをダブルクリックします。

フォームを閉じるには、フォームエクスプローラーを右クリックして、<閉じる>メニューを選択します。または、フォームのタブを右クリックして、<閉じる>メニューを選択します。

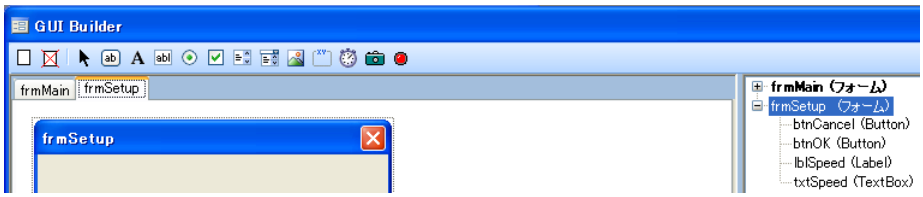
4.5.4 フォームサイズの変更

フォームのサイズを変更するには、次の 2 つの方法があります。

- GUI Builder ウィンドウのデザインエリアのフォームに表示されているフォームの縁をマウスでつかんでドラッグします。
- プロパティーグリッドの Height(高さ)、Width(幅)プロパティーの値を変更するか、SPEL+プログラムから GSet ステートメントを使って値を書き換えます。

4.5.5 複数のフォームの編集

GUI Builder ウィンドウは、タブインターフェースを使って複数のフォームを切り替えます。タブをクリックすることで、現在のフォームを切り替えることができます。



フォームからコントロールをコピーして、他のフォームに貼りつけることもできます。

4.5.6 コントロールの作成

コントロールを作成します。

1. コントロールを追加したいフォームを開きます。
2. GUI Builder ウィンドウのツールバーにあるコントロールボタンをクリックします。カーソルがクロスカーソルに変わります。
3. コントロールを配置したい場所でクリックすると、デフォルトのサイズでコントロールを配置します。マウスでコントロールの外形をドラッグすると、自由にコントロールの大きさを変えられます(タイマー以外)。

4.5.7 コントロールの削除

コントロールを削除します。

1. コントロールを削除したいフォームを開きます。.
2. 削除したいコントロールをクリックします。複数のコントロールを削除する場合、<Ctrl>キーまたは<Shift>キーを押したまま別のコントロールをクリックします。
3. キーボードの<Del>キーを押すと、選択したコントロールが削除されます。

4.5.8 コントロールのサイズ変更と移動

コントロールのサイズを変更します。

- GUI Builder ウィンドウのデザインエリアのフォームに表示されているコントロールの縁をマウスでつかんでドラッグします。
- プロパティーグリッドの Height(高さ)、Width(幅)プロパティーの値を変更するか、SPEL+プログラムから GSet ステートメントを使って値を書き換えます。

コントロールを移動します。

- コントロールをクリックして、選択した状態にします。コントロール上にカーソルを移動させると、カーソルが移動カーソル  に変わります。その状態でコントロールをドラッグし、新しい場所に配置します。
- プロパティーグリッドの Left、Top プロパティーの値を変更するか、SPEL+プログラムから GSet ステートメントを使って値を書き換えます。

4.5.9 コントロールのコピー、切り取り、貼りつけ

コピーと切り取りのために、コントロールを選択します。

まず、コントロールをクリックします。複数のコントロールを選択するには、<Ctrl>キーまたは<Shift>キーを押したまま別のコントロールをクリックします。

選択したコントロールをコピーします。次の 3 つの方法があります。

- <Ctrl> + <C>キーを押す
- ツールバーの<コピー>ボタンをクリックする
- 編集メニューの[コピー]を選択する

選択したコントロールを切り取ります。次の 3 つの方法があります。

- <Ctrl> + <X>キーを押す
- ツールバーの<切り取り>ボタンをクリックする
- 編集メニューの[切り取り]を選択する

選択したコントロールを貼りつけます。次の 3 つの方法があります。

- <Ctrl> + <V>キーを押す
- ツールバーの<貼り付け>ボタンをクリックする
- 編集メニューの[貼り付け]を選択する

4.5.10 プロパティーの編集

プロパティーを編集するには、まずフォームかコントロールをクリックして、プロパティーグリッドにプロパティーを表示させます。

テキスト入力が必要なプロパティーの場合

プロパティーグリッドの変更したいプロパティーをクリックします。新しい値を入力したあと、<Enter>キーを押すか別のプロパティーを選択すると、変更が適用されます。

ドロップダウンリストから選択するプロパティーの場合

プロパティーグリッドの変更したいプロパティーをクリックします。値の右にある▼をクリックし、ドロップダウンリストから新しい値を選択すると変更が適用されます。

ボタンを選択するプロパティーの場合

プロパティーグリッドの変更したいプロパティーをクリックします。値の右にある楕円のボタンを選択します。新しい値をダイアログから選択します。<OK>ボタンをクリックすると変更が適用されます。

4.5.11 イベントハンドラーの使い方

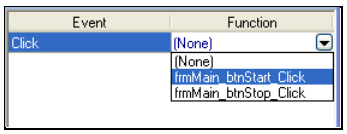
イベントハンドラーは、イベント固有のパラメーターをもった SPEL+関数です。イベントハンドラーは、カレントプロジェクトのどのプログラムファイルにも記述できます。デフォルトでは、フォームを作成するためのプログラムの中に生成されます。例えば、ボタンコントロールのクリックイベントのハンドラーを作成する場合、もしプログラムファイルがなければ自動的に作成し、そのファイルの中に関数を作成します。

イベントハンドラーを作成します。次の 3 つの方法があります。

- フォームやコントロールをダブルクリックし、デフォルトのイベントハンドラーを作成します。例えば、ボタンコントロールをダブルクリックすると、クリックイベントのハンドラーの関数が生成されます。
- イベントグリッドで、イベント名をダブルクリックします。
- イベントグリッドで、イベントの値のドロップダウンリストから関数を選択します。正しいパラメーターで作成された関数が、ドロップダウンリストに表示されます。

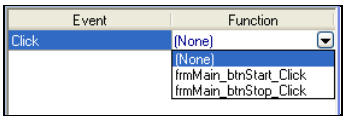
イベントハンドラーを変更します。

イベントの Fuction のドロップダウンリストから、イベントハンドラー関数を変更できます。



イベントハンドラーを無効にします。

イベントハンドラー作成後に、必要なくなったイベントハンドラーを無効にすることができます。イベントハンドラーを無効にするには、[Function]ドロップダウンリストから(None)を選択します。関数は削除されませんが、イベントが発生したときに呼び出されなくなります。



EventTaskType の使い方

イベントが発生した場合、イベントハンドラー関数が SPEL+のタスクとして起動されます。

EventTaskType プロパティーは、タスクの実行されるタイプを選択します。これは、一時停止状態や、非常停止状態でタスクを実行するためにはとても重要です。

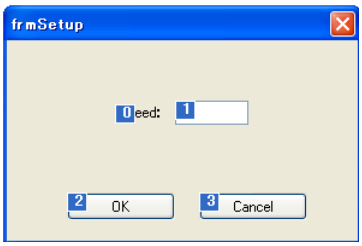
例えば、Pause ステートメントや Cont ステートメントを実行するボタンのイベントハンドラーでは、EventTaskType プロパティーを 1 – NoPause に設定しなくてはなりません。また、非常停止中に Reset ステートメントを行うボタンのイベントハンドラーでは、EventTaskType を 2 – NoEmgAbort に設定します。

4.5.12 タブオーダーの変更

フォームにコントロールを追加した後に、タブオーダーを変更できます。タブオーダーは、プログラム実行中に<Tab>キーを押したときに、フォーカスのあたる順序です。ホットキーを指定したラベルのタブインデックスは関連付けるコントローラーの前に設定する必要があります。

タブオーダーを変更します。

1. タブオーダーを変更したいフォームを開きます。
2. <Tab>キーを押すと、タブオーダーが次のように表示されます。



3. 順序づけしたい順番に、タブオーダーの値をクリックします。
4. もう一度<Tab>キーを押すと、タブオーダーの表示を隠します。

4.5.13 変更の保存

GUI Builder ウィンドウの変更を保存するには、次の 2 つの方法があります。

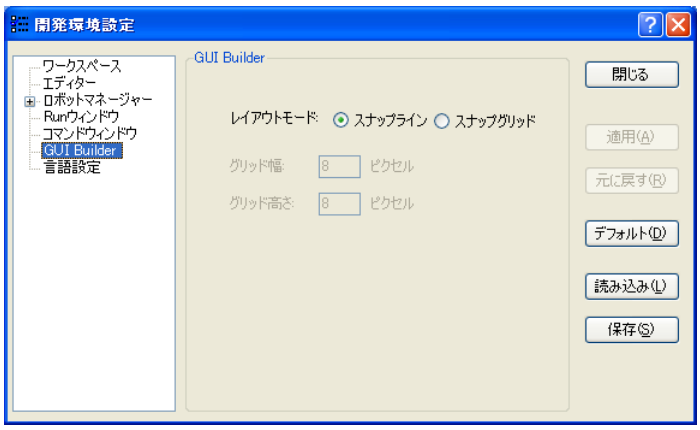
- メニュー-[ファイル]-[保存]を選択するか、<Ctrl>＋<S>キーを押します。

-ツールバー-<プロジェクトの保存>ボタンをクリックします。

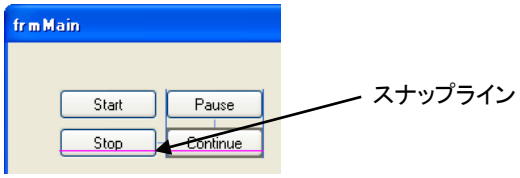
4.6 GUI Builder の設定

GUI Builder ウィンドウのフォーム上でのコントロールの配置方法を、GUI Builder の設定から変えることができます。

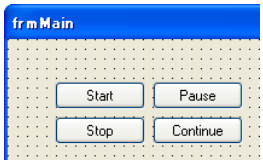
[セットアップメニュー]-[開発環境設定]の GUI Builder を選択します。



[レイアウトモード]を“スナップライン”に設定した場合、GUI Builder は、スナップラインを表示して、コントロールをスナップラインに添って配置します。



[レイアウトモード]を“スナップグリッド”に設定した場合、GUI Builder はグリッドを表示して、コントロールをグリッドの位置に配置します。



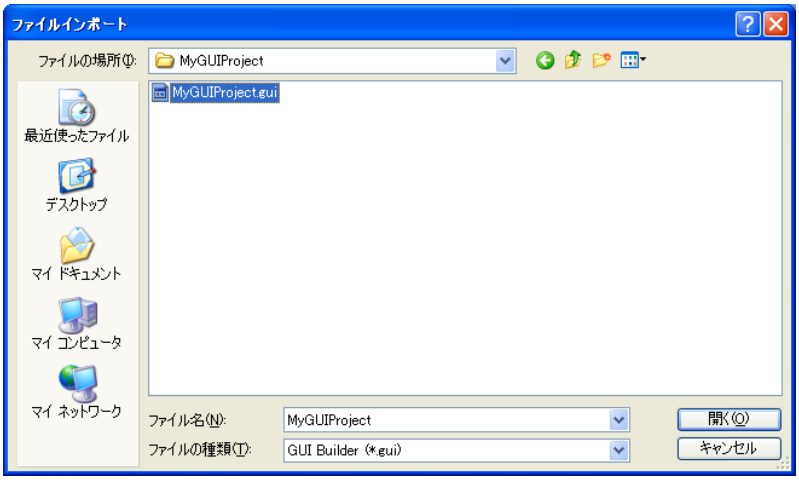
グリッドのサイズ(間隔)を変える場合は、[グリッド幅], [グリッド高さ]の値を変更してください。

GUI Builder の設定を変更して<適用>ボタンをクリックすると、GUI Builder ウィンドウが開かれている場合、変更を保存後、新しい設定でウィンドウを開きます。

4.7 他のプロジェクトからフォームをインポートするには

[ファイル]メニュー-[ファイルインポート]で、他のプロジェクトから GUI フォームをインポートできます。

1. [ファイル]-[ファイルインポート]を選択します。
2. インポート元のプロジェクトフォルダーを指定し、GUI Builder のファイル形式を選択します。



3. GUI のファイル名を選択し、<開く>ボタンをクリックします。
4. インポート元のプロジェクトが持っているフォームの一覧を表示するダイアログが表示されます。

5. インポートしたいフォームを選択し(複数選択可)、<OK>をクリックします。
6. カレントプロジェクトに選択したフォームが追加されます。選択したフォームがすでに追加されていた場合、フォームを上書きするかを、質問されます。

5. GUI Builder の構成要素

本章では、フォームと、フォームの上で使用可能なすべてのコントロールついて説明します。各コンポーネントの項では、使い方、プロパティー、イベントについて説明します。

プロパティー、イベント、ステートメントの詳細は、リファレンスの章を参照してください。

5.1 フォーム

5.1.1 解説

フォームは GUI アプリケーションの基本的な土台となります。オペレーター操作のために、コントロールが配置されたウィンドウやダイアログボックスを表示します。GUI アプリケーションは、1 つまたは複数のフォームを持つことができます。

5.1.2 使い方

フォームを表示します。次の 3 つの方法でフォームを表示します。

1. フォームがスタートアップフォームに設定されている場合は、オペレーターモードでシステムを起動すると自動的に表示されます。スタートアップフォームに設定するためには、フォームエクスプローラーのフォームを右クリックして、“スタートアップフォーム”を選択してください。

   スタートアップフォームは、太字で表示されます。
   また、メニュー-[プロジェクト]-[プロパティー]-[GUI Builder]を選択して、スタートアップフォームに設定することもできます。

2. GShowDialog 関数を使用します。この関数は、フォームをダイアログボックスとして表示し、DialogResult の値を返します。

   ```
   result = GShowDialog(frmSetSpeed)
   ```

   通常、DialogResult の値は、フォームの上の<OK>ボタンや<キャンセル>ボタンなどで設定されます。

3. GShow ステートメントを使用します。このステートメントは、フォームをウィンドウとして表示します。

```
GShow frmIODiags
```

フォームの設定

FormBorderStyle プロパティーを設定します。ダイアログとして使うときは、3 – FixedDialog を設定します。

フォームのタイトルバーの構成は、ControlBox、MaximizeBox、MinimizeBox プロパティーを設定します。

フォームを表示するときの設定は、WindowState プロパティーに、Normal size、Maximized、Minimized を選択してください。

ヘルプの使い方

HelpButton プロパティーを True に設定すると、独自の Help ファイルからヘルプトピックを表示させることができます。その時は、HelpID プロパティーに、ヘルプファイルのトピック ID を設定します。

詳細は、「6.6 ヘルプファイルの使い方」を参照してください。

5.1.3 フォームプロパティー

プロパティー	説明
AcceptButton	ボタン以外のコントロールにフォーカスがあるとき、<Enter>キーを押したときにクリックイベントを実行するボタンを設定します。 デフォルト: None
BackColor	フォームの背景色を設定します。 デフォルト: Control
CancelButton	キーボードの<Esc>キーを押したときに、クリックイベントを実行するボタンを設定します。実行後にフォームを閉じます。 デフォルト: None
ControlBox	タイトルバーのコントロールボックスを表示するかを設定します。 デフォルト: 1 – True
Controls	フォーム上のコントロール配列。
Count	コントロール配列上のコントロールの数を返します。
Dialog Result	フォームを閉じるときの戻り値を設定します。 (SEPL プログラムからのみ有効)
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト: 0 –Normal
FormBorderStyle	フォームのボーダースタイルを設定します。 デフォルト: 3 – FixedDialog
Height	フォームの高さをピクセル数で設定します。
HelpButton	タイトルバーへ、ヘルプボタンを表示するかを設定します。 デフォルト: 0 – False
HelpID	タイトルバーのヘルプボタンを押したときに表示するヘルプトピック ID を設定します。 デフォルト: 0
Left	フォームの表示位置の左の座標をピクセル数で設定します。

プロパティー	説明
MaximizeBox	タイトルバーに最大化ボタンを表示するかを設定します。 デフォルト: 0 – False
MinimizeBox	タイトルバーに最小化ボタンを表示するかを設定します。 デフォルト: 0 – False
Name	フォームの名前を設定します。
StartPosition	フォームの表示位置を設定します。 デフォルト: 1 – CenterScreen
Text	フォームの表示テキストを設定します。 デフォルト:フォームの名前
Top	フォームの表示位置の上の座標をピクセル数で設定/取得します。
Type	コントロールのタイプを返します。
Width	フォームの幅をピクセル数で設定します。
WindowState	フォームのウィンドウの状態を設定します。 デフォルト: 0 – Normal

5.1.4 フォームイベント

イベント	説明
Closed	フォームが閉じられたとき実行されます。
Load	フォームが読み込まれるとき実行されます。
Resize	フォームのサイズが変更されるとき実行されます。

5.2 ボタンコントロール

5.2.1 解説

ボタンコントロールは、オペレーターが何かを実行したいときにクリックするものです。
ボタンコントロールには、テキストと画像の両方を表示させることができます。ボタンがクリックされると、ボタンが押されたように表示されます。

5.2.2 使い方

ボタンコントロールは、オペレーターが行う操作をクリック一つで実行します。

ForeColor(ボタンテキストの表示用)、BackColor、Font、TextAlign、Image、ImageAlign などのプロパティーを設定すると、ボタンの外観を変更できます。
ボタンが押されたときにどのような動作を行うか、クリックイベント関数に記述してください。

5.2.3 ボタンコントロールのプロパティー

プロパティー	説明
BackColor	コントロールの背景色を設定します。 デフォルト: Control
BackColorMode	コントロールの背景色モードを設定します。 デフォルト: 0 – Visual Style
Enabled	コントロールを有効にするかを設定します。 デフォルト: 1 – True
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト:0 – Normal
Font	コントロールのテキストを表示するフォントを設定します。(フォント名、スタイル、サイズ) (設計時のみ有効／SPEL プログラムからは変更できません) デフォルト: Microsoft Sans Serif 8.25 pt
FontBold	現在のフォントを太字にします。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontItalic	現在のフォントをイタリックにします。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontName	現在のフォントの名前を設定します。 (SPEL プログラムからのみ有効) デフォルト: Microsoft Sans Serif
FontSize	現在のフォントのサイズをポイントで設定します。 (SPEL プログラムからのみ有効) デフォルト: 8.25
ForeColor	コントロールのテキストの色を設定します。 デフォルト: ControlText
Height	コントロールの高さをピクセル数で設定します。
Image	コントロールに表示する画像ファイルを設定します。<Delete>キーで削除できます。 デフォルト: 空白
ImageAlign	表示する画像の配置を設定します。 デフォルト: 5 – MiddleCenter
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
Name	コントロールの名前を設定します。 デフォルト: ButtonXX
TabIndex	コントロールのタブインデックスを設定します。
Text	コントロールの表示テキストを設定します。 デフォルト: コントロールの名前
TextAlign	テキストの配置を設定します。 デフォルト: 5 – MiddleCenter
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白
Top	コントロールの表示位置の上の座標をピクセル数で設定します。

プロパティー	説明
Visible	コントロールを表示するか、隠すかを設定します。 デフォルト: 1 – True
Width	コントロールの幅をピクセル数で設定します。

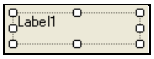
5.2.4 ボタンコントロールのイベント

イベント	説明
Click	コンロトールがクリックされたときに実行されます。

5.3 ラベルコントロール

5.3.1 解説

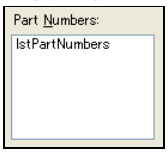
ラベルコントロールは、オペレーターが編集できないテキストやイメージを表示します。
フォーム上のオブジェクトの説明や、実行時の情報表示、アプリケーションの状況などを表示します。



5.3.2 使い方

ラベルコントロールを使用して、フォームにテキストを表示します。

通常、ラベルコントロールは他のコントロールのテキストラベルに使用されます。
例えば、フォームの上にリストボックスコントロールを作成する場合、ラベルコントロールでリスト名をリストボックスの上に配置します。
また、リストボックス名として、Text プロパティーを Part Numbers:としたラベルコントロールの例を下に示します。



ラベルの外観の設定

ラベルの外観を変更するには、次のプロパティーを使用します。
BorderStyle、ForeColor、BackColor、Font、Image、ImageAlign、TextAlign

ホットキーの使用(ニーモニック)

ラベルの Text プロパティーで、ホットキーとなる文字を指定できます。
ラベルコントロールの Text プロパティーの中で、ホットキーになる文字の前にアンパーサンド(&)を付けてください。
例えば、前の説明のラベルで “Part &Numbers:”のように Text プロパティーを設定します。

実行時に、<Alt>＋<N>を押すと、フォーカスがリストボックスに移動します。

ホットキーを使用するとき、ラベルのタブインデックスが、コントロールのタブインデックスの一つ前となっていることを確認してください。(「4.5.12 タブオーダーの変更」を参照)

5.3.3 ラベルコントロールのプロパティー

プロパティー	説明
BackColor	コントロールの背景色を設定します。 デフォルト: Control
BorderStyle	コントロールのボーダースタイルを設定します。 デフォルト: 0 – None
Enabled	コントロールを有効にするかを設定します。 デフォルト: 1 – True

プロパティー	説明
EventTaskType	イベントで起動される関数のタスクタイプを設定します。 デフォルト: 0 – Normal
Font	コントロールのテキストを表示するフォントを設定します(フォント名、スタイル、サイズ)。(設計時のみ有効) デフォルト: Microsoft Sans Serif 8.25 pt
FontBold	現在のフォントをボールド表示します。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontItalic	現在のフォントをイタリック表示します。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontName	現在のフォントの名前を設定します。 (SPEL プログラムからのみ有効) デフォルト: Microsoft Sans Serif
FontSize	現在のフォントのサイズをポイントで設定します。 (SPEL プログラムからのみ有効) デフォルト: 8.25
ForeColor	コントロールのテキストの色を設定します。 デフォルト: ControlText
Height	コントロールの高さをピクセル数で設定します。
Image	コントロールに表示する画像ファイルを設定します。<Delete>キーで削除できます。 デフォルト: 空白
ImageAlign	表示する画像の配置を設定します。 デフォルト: 5 – MiddleCenter
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
Name	コントロールの名前を設定します。 デフォルト: LabelXX
TabIndex	コントロールのタブインデックスを設定します。
Text	コントロールの表示テキストを設定します。 デフォルト: コントロールの名前
TextAlign	テキストの配置を設定します。 デフォルト: 1 – TopLeft
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白
Top	コントロールの表示位置の上の座標をピクセル数で設定します。
Visible	コントロールを表示するか、隠すかを設定します。 デフォルト: 1 – True
Width	コントロールの幅をピクセル数で設定します。

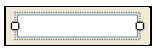
5.3.4 ラベルコントロールのイベント

イベント	説明
Click	コンロトールがクリックされたときに実行されます。
DblClick	コンロトールがダブルクリックされたときに実行されます。

5.4 テキストボックスコントロール

5.4.1 解説

テキストボックスコントロールは、オペレーターが入力したり、テキストを表示するために使われます。通常、テキストボックスは編集可能なテキストのために使われます。(書き換え不可にすることも可能です。)
テキストボックスは、複数行のテキストを表示したり、コントロールのサイズに合わせてテキストを改行します。
テキストボックス内のフォーマットは 1 つのみ設定できます。一部の文字を太字にするなどはできません。



5.4.2 使い方

テキストボックスは、1 行または複数行のモードで使用します。

1 行のテキストボックス

デフォルトでは、テキストボックスは 1 行モードで作成されます。コントロールの幅のみ変更可能です。オペレーターは、文字を 1 行入力することができます。

複数行のテキストボックス

複数行モードで使うときは、Multiline プロパティーを True に設定します。このモードでは、コントロールの高さと幅を変更できます。
Scrollbars プロパティーの設定によって、スクロールバーを表示することもできます。
水平スクロールバーを表示するには、WordWrap プロパティーを False に設定します。

テキストボックスの外観の設定

テキストボックスの外観を変更するには、次のプロパティーを使用します。
BorderStyle、ForeColor、BackColor、Font、TextAlign

SPEL^{+}のグローバル変数の表示方法

Variable プロパティーを設定することによって、自動的に SPEL+グローバル変数の値を表示することができます。

注意
Variable プロパティー設定で SPEL+のグローバル変数が表示されない場合、プロジェクトをビルドしてください。

5.4.3 テキストボックスコントロールのプロパティー

プロパティー	説明
AppendText	テキストボックスにテキストを追加します。 (SPEL プログラムからのみ有効)
BackColor	コントロールの背景色を設定します。 デフォルト: Window
BorderStyle	コントロールのボーダースタイルを設定します。 デフォルト: 2 – Fixed3D
Enabled	コントロールを有効にするかを設定します。 デフォルト: 1 – True
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト: 0 – Normal

プロパティー	説明
Font	コントロールのテキストを表示するフォントを設定します。 (設計時のみ有効) デフォルト: Microsoft Sans Serif 8.25 pt
FontBold	現在のフォントを太字にします。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontItalic	現在のフォントをイタリック表示します。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontName	現在のフォントの名前を設定します。 (SPEL プログラムからのみ有効) デフォルト: Microsoft Sans Serif
FontSize	現在のフォントのサイズをポイントで設定します。 (SPEL プログラムからのみ有効) デフォルト: 8.25
ForeColor	コントロールのテキストの色を設定します。 デフォルト: WindowText
Height	コントロールの高さをピクセル数で設定します。
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
Multiline	複数行モードにするかを設定します。 デフォルト: 0 – False
Name	コントロールの名前を設定します。 デフォルト: TextBoxXX
PasswordChar	入力された文字を隠すための文字を設定します。 デフォルト:空白
ReadOnly	オペレーターにテキストを編集させない場合に設定します。 デフォルト: 0 – False
ScrollBars	スクロールバーを表示するときに設定します。 デフォルト: 0 – None
ShowPrint	Print ステートメントの出力を表示するかを設定します。 デフォルト: 0 - False
TabIndex	コントロールのタブインデックスを設定します。
Text	コントロールの表示テキストを設定します。 デフォルト: 空白
TextAlign	テキストの配置を設定します。 デフォルト: 1 – Left
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白
Top	コントロールの表示位置の上の座標をピクセル数で設定します。
Update	テキストボックスコントロールの表示を更新します。 (SPEL プログラムからのみ有効)
Variable	表示させたい SPEL+のグローバル変数を設定します。 デフォルト: None
Visible	コントロールを表示するか、隠すかを設定します。 デフォルト: 1 – True

プロパティー	説明
Width	コントロールの幅をピクセル数で設定します。
WordWrap	テキストを改行して表示するかどうかを設定します。 デフォルト: 1 – True

5.4.4 テキストボックスコントロールのイベント

イベント	説明
KeyPress	コントロールにフォーカスがあり、さらにキーが押された場合に実行されます。

5.5 ラジオボタンコントロール

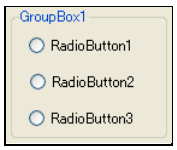
5.5.1 解説

ラジオボタンコントロールは、オペレーターが 2 つまたはそれ以上の選択操作を行うためのコントロールです。

ラジオボタンとチェックボックスは同じような機能に見えますが、大きな違いがあります。ラジオボタンの場合、一つ選択すると同じグループの他のラジオボタンは選択できなくなります。

5.5.2 使い方

ラジオボタンコントロールは、グループボックスコントロールとともに使用します。
オペレーターがラジオボタンの 1 つをクリックすると、他のボタンは選択が解除されます。



オペレーターがラジオボタンをクリックしたことは、クリックイベントで判断できます。ラジオボタンがチェックされているかどうかは、Checked プロパティーで判断できます。

各ラジオボタンにイベントハンドラーを作成するより、グループのラジオボタンすべてにひとつのイベントハンドラーを作成するほうが簡単な場合があります。

イベントハンドラーの Sender$パラメーターを使うと、どのラジオボタンがクリックされたかを判断することができます。Sender$はイベントを送ったコントロールの名前が設定されています。

```
Function frmSetup_OptionsClick(Sender$ As String)
  Boolean checked
  GGet frmSetup.Sender$.Checked, checked
  If checked Then
    Select Sender$
      Case "RadioButton1":
        g_Option1 = True
      Case "RadioButton2":
        g_Option2 = True
    Send
  EndIf
Fend
```

ラジオボタンの外観の設定

ラジオボタンの外観を変更するには、次のプロパティーを使用します。
BorderStyle、ForeColor、BackColor、Font、Image、ImageAlign、TextAlign

5.5.3 ラジオボタンコントロールのプロパティー

プロパティー	説明
BackColor	コントロールの背景色を設定します。 デフォルト: Control
Checked	ラジオボタンの選択状態の初期値を設定します。 デフォルト: 0 – False
Enabled	ラジオボタンを有効にするかどうかを設定します。 デフォルト: 1 – True
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト: 0 – Normal
Font	コントロールのテキストを表示するフォントを設定します。 (設計時のみ有効) デフォルト: Microsoft Sans Serif 8.25 pt
FontBold	現在のフォントをボールド表示します。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontItalic	現在のフォントをイタリック表示します。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontName	現在のフォントの名前を設定します。 (SPEL プログラムからのみ有効) デフォルト: Microsoft Sans Serif
FontSize	現在のフォントのサイズをポイントで設定します。 (SPEL プログラムからのみ有効) デフォルト: 8.25
ForeColor	コントロールのテキストの色を設定します。 デフォルト: ControlText
Height	コントロールの高さをピクセル数で設定します。
Image	コントロールに表示する画像ファイルを設定します。<Delete>キーで削除できます。 デフォルト: 空白
ImageAlign	表示する画像の配置を設定します。 デフォルト: 5 – MiddleCenter
Left	フォームの表示位置の左の座標をピクセル数で設定します。
Name	コントロールの名前を設定します。 デフォルト: RadioButtonXX
TabIndex	コントロールのタブインデックスを設定します。
Text	コントロールの表示テキストを設定します。 デフォルト: コントロールの名前

プロパティー	説明
TextAlign	テキストのアライメント方法を設定します。 デフォルト: 4 – Middle Left
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白
Top	コントロールの表示位置の上の座標をピクセル数で設定します。
Visible	コントロールを表示するかどうかを設定します。 デフォルト: 1 – True
Width	コントロールの幅をピクセル数で設定します。

5.5.4 ラジオボタンコントロールのイベント

イベント	説明
Click	コンロトールがクリックされたときに実行されます。

5.6 チェックボックスコントロール

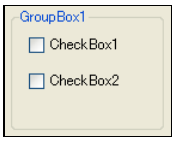
5.6.1 解説

チェックボックスコントロールは、オンかオフの状態を入力するコントロールです。主に、はい／いいえ、True／False を選択するのに使われます。
チェックボックスコントロールをグループ化して、1 つまたは複数を選択するためのコントロールにすることもできます。

ラジオボタンコントロールと似ていますが、チェックボックスコントロールでは複数のボタンの選択が可能です。

5.6.2 使い方

チェックボックスコントロールは、各設定のオン／オフ操作を行えます。
通常、チェックボックスコントロールはグループボックスコントロールに追加していきます。
Checked プロパティーでチェックボックスがチェックされているかを確認できます。



チェックボックスの外観の設定

チェックボックスの外観を変更するには、次のプロパティーを使います。
BorderStyle、ForeColor、BackColor、Font、Image、ImageAlign、TextAlign

5.6.3 チェックボックスコントロールのプロパティー

プロパティー	説明
BackColor	コントロールの背景色を設定します。 デフォルト: Control
Checked	チェックボックスの選択状態の初期値を設定します。 デフォルト: 0 –False
Enabled	コントロールを有効にするかどうかを設定します。 デフォルト: 1 –True
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト: 0 –Normal
Font	コントロールのテキストを表示するフォントを設定します。 (設計時のみ有効) デフォルト: Microsoft Sans Serif 8.25 pt
FontBold	現在のフォントを太字にします。 (SPEL プログラムからのみ有効) デフォルト: 0 –False
FontItalic	現在のフォントをイタリックにします。 (SPEL プログラムからのみ有効) デフォルト: 0 –False
FontName	現在のフォントの名前を設定します。 (SPEL プログラムからのみ有効) デフォルト: Microsoft Sans Serif
FontSize	現在のフォントのサイズをポイントで設定します。 (SPEL プログラムからのみ有効) デフォルト: 8.25
ForeColor	コントロールのテキストの色を設定します。 デフォルト: ControlText
Height	コントロールの高さをピクセル数で設定します。
Image	コントロールに表示する画像ファイルを設定します。<Delete>キーで削除できます。 デフォルト: 空白
ImageAlign	表示する画像の配置を設定します。 デフォルト:5 –MiddleCenter
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
Name	コントロールの名前を設定します。 デフォルト: CheckBoxXX
TabIndex	コントロールのタブインデックスを設定します。
Text	コントロールの表示テキストを設定します。 デフォルト: コントロールの名前
TextAlign	テキストの配置を設定します。 デフォルト: 4 – MiddleLeft
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白
Top	コントロールの表示位置の上の座標をピクセル数で設定します。

プロパティー	説明
Visible	コントロールを表示するかどうかを設定します。 デフォルト: 1 –True
Width	コントロールの幅をピクセル数で設定します。

5.6.4 チェックボックスのコントロールのイベント

イベント	説明
Click	コンロトールがクリックされたときに実行されます。

5.7 リストボックスコントロール

5.7.1 説明

リストボックスコントロールは、オペレーターが選択する 1 つまたは複数のアイテムを表示するコントロールです。

5.7.2 使い方

リストボックスコントロールを使って、オペレーターが選択できるアイテムのリストを作成します。

リストボックスにアイテムを追加

AddItem プロパティーを使用して、リストボックスにアイテムを加えます。通常は、フォームのロードイベントで行います。

```
GSet frmSetup.lstModels.AddItem, "Model1"
GSet frmSetup.lstModels.AddItem, "Model2"
GSet frmSetup.lstModels.AddItem, "Model3"
```

Sorted プロパティーを True に設定すると、アイテムをソート(整列)することができます。

選択されたアイテムの確認

SelectedIndex プロパティーを使用して、オペレーターが選択したアイテムを確認できます。どのアイテムも選択されていなければ、SelectedIndex プロパティーの値は–1 です。

```
Integer index
GGet frmSetup.lstModels.SelectedIndex, index
```

リスト配列

List Array プロパティーを使用すると、リストの中のアイテムのすべてにアクセスが可能です。

```
Integer i, count
String item$
GGet frmSetup.lstModels.ListCount, count
For i = 0 To count - 1
  GGet frmSetup.lstModels.List(i), item$
Next i
```

List プロパティーに空の文字列を設定することで、アイテムを削除できます。

```
GSet frmSetup.lstModels.List(0), ""
Setting ListBox Appearance
Use the BorderStyle, ForeColor, BackColor, Font,
properties to change the appearance of the listbox.
```

リストボックスの外観の設定

リストボックスの概観を変更するには、次のプロパティーを使います。
BorderStyle、ForeColor、BackColor、Font

5.7.3 リストボックスコントロールのプロパティー

プロパティー	説明
AddItem	リストボックスにアイテムを追加します。 (SPEL プログラムからのみ有効)
BackColor	コントロールの背景色を設定します。 デフォルト: Window
BorderStyle	コントロールのボーダーのスタイルを設定します。 デフォルト: 2 – Fixed3D
Enabled	コントロールを有効にするかを設定します。 デフォルト: 1 – True
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト: 0 – Normal
Font	コントロールのテキストを表示するフォントを設定します。 (設計時のみ有効) デフォルト: Microsoft Sans Serif 8.25 pt
FontBold	現在のフォントを太字にします。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontItalic	現在のフォントをイタリックにします。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontName	現在のフォントの名前を設定します。 (SPEL プログラムからのみ有効) デフォルト: Microsoft Sans Serif
FontSize	現在のフォントのサイズをポイントで設定します。 (SPEL プログラムからのみ有効) デフォルト: 8.25
ForeColor	コントロールのテキストの色を設定します。 デフォルト: WindowText
Height	コントロールの高さをピクセル数で設定します。
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
List	リストボックスのアイテムにアクセスします。 (SPEL プログラムからのみ有効)
ListCount	アイテムの数を取得します。
Name	コントロールの名前を設定します。 デフォルト: ListBoxXX
SelectedIndex	選択されたアイテムの番号を取得します。 (SPEL プログラムからのみ有効)
Sorted	リストのアイテムをソートするかを設定します。 デフォルト: 0 – False
TabIndex	コントロールのタブインデックスを設定します。
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白
Top	コントロールの表示位置の上の座標をピクセル数で設定します。
Visible	コントロールを表示するか、隠すかを設定します。 デフォルト: 1 – True
Width	コントロールの幅をピクセル数で設定します。

5.7.4 リストボックスコントロールのイベント

イベント	説明
Click	コンロトールがクリックされたときに実行されます。
DblClick	コンロトールがダブルクリックされたときに実行されます。

5.8 コンボボックスコントロール

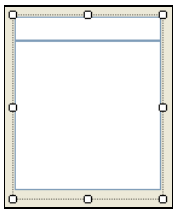
5.8.1 解説

コンボボックスコントロールは、データをドロップダウンコンボボックスに表示します。
デフォルトでは 2 つの部分に分かれます。上部分はテキストボックスで、リストのアイテムを入力できます。下部分はリストボックスで、オペレーターは表示されているアイテムから 1 つを選択できます。

5.8.2 使い方

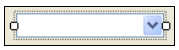
DropDownStyle プロパティーの値によって、コンボボックスコントロールは 3 種類のモードを持ちます。

DropDownStyle = Simple



Simple モードでは、リストボックスの上にテキストボックスがつながっています。オペレーターは、テキストボックスに入力するか、リストから選択します。

DropDownStyle = DropDown



DropDown モードでは、テキストボックスの右側の▼をクリックするとリストボックスが表示されます。テキストボックスで、選択されたテキストを編集することができます。

DropDownStyle = DropDownList

DropDownList モードでは、テキストボックス内のテキスト編集ができません。リストにあるアイテムのみ選択可能です。

コンボボックスコントロールのリストボックス部分にアイテムを追加する方法は、リストボックスの AddItem、List、ListCount、SelectedIndex プロパティーの説明を参照してください。

コンボボックスの外観の設定

ForeColor、BackColor、Font プロパティーを使って、コンボボックスの外観を変更します。

5.8.3 コンボボックスコントロールのプロパティー

プロパティー	説明
AddItem	コンボボックスにアイテムを追加します。 (SPEL プログラムからのみ有効)
BackColor	コントロールの背景色を設定します。 デフォルト: Window
DropDownStyle	コンボボックスのドロップダウンのモードを設定します。 デフォルト: 1 – DropDown
Enabled	コントロールを有効にするかを設定します。 デフォルト: 1 – True
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト: 0 – Normal
Font	コントロールのテキストを表示するフォントを設定します。 (設計時のみ有効) デフォルト: Microsoft Sans Serif 8.25 pt
FontBold	現在のフォントを太字にします。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontItalic	現在のフォントをイタリックにします。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontName	現在のフォントの名前を設定します。 (SPEL プログラムからのみ有効) デフォルト: Microsoft Sans Serif
FontSize	現在のフォントのサイズをポイントで設定します。 (SPEL プログラムからのみ有効) デフォルト: 8.25
ForeColor	コントロールのテキストの色を設定します。 デフォルト: WindowText
Height	コントロールの高さをピクセル数で設定します。
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
List	リストボックスのアイテムにアクセスします。 (SPEL プログラムからのみ有効)
ListCount	アイテムの数を取得します。 (SPEL プログラムからのみ有効)
Name	コントロールの名前を設定します。 デフォルト: ComboBoxXX
SelectedIndex	選択されたアイテムの番号を取得します。 (SPEL プログラムからのみ有効)
Sorted	リストのアイテムをソートするかを設定します。 デフォルト: 0 – False
TabIndex	コントロールのタブインデックスを設定します。
Text	コントロールの表示テキストを設定します。 デフォルト: 空白

プロパティー	説明
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白
Top	コントロールの表示位置の上の座標をピクセル数で設定します。
Visible	コントロールを表示するか、隠すかを設定します。 デフォルト: 1 – True
Width	コントロールの幅をピクセル数で設定します。

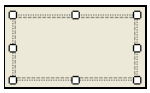
5.8.4 コンボボックスコントロールのイベント

イベント	説明
Click	コンロトールがクリックされたときに実行されるイベントです。

5.9 ピクチャーボックスコントロール

5.9.1 解説

ピクチャーボックスコントロールは、ウィンドウズビットマップ、GIF、JPEG、PNG、WMF、ウィンドウズメタファイル、アイコンを表示します。



5.9.2 使い方

ピクチャーボックスコントロールを使って、選択したイメージファイルを表示できます。
イメージを選択するのは、設計時または実行時どちらでも可能です。

イメージの設定

設計時には Image プロパティーを使用し、ファイル選択ダイアログでイメージファイルを選択することができます。

実行時にイメージファイルを設定するには、イメージファイルのフルパスの指定が必要です。

イメージのサイズ変更

SizeMode プロパティーを使ってイメージのサイズを変更することが可能です。

ピクチャーボックスの外観の設定

BackColor、BorderStyle、SizeMode を使ってピクチャーボックスの外観を変更します。

5.9.3 ピクチャーボックスコントロールのプロパティー

プロパティー	説明
BackColor	コントロールの背景色を設定します。 デフォルト: Control
BorderStyle	コントロールのボーダーのスタイルを設定します。 デフォルト: 0 – None
Enabled	コントロールを有効にするかを設定します。 デフォルト: 1 – True

プロパティー	説明
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト: 0 – Normal
Height	コントロールの高さをピクセル数で設定します。
Image	コントロールに表示する画像ファイルを設定します。<Delete>キーで削除できます。
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
Name	コントロールの名前を設定します。 デフォルト: PictureBoxXX
SizeMode	イメージのサイズの変更方法を設定します。 デフォルト: 0 – Normal
TabIndex	コントロールのタブインデックスを設定します。
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白
Top	コントロールの表示位置の上の座標をピクセル数で設定します。
Visible	コントロールを表示するか、隠すかを設定します。 デフォルト: 1 – True
Width	コントロールの幅をピクセル数で設定します。

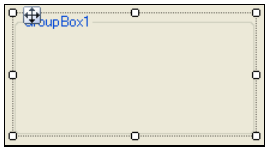
5.9.4 ピクチャーボックスコントロールのイベント

イベント	説明
Click	コンロトールがクリックされたときに実行されます。
DblClick	コンロトールがダブルクリックされたときに実行されます。

5.10 グループボックスコントロール

5.10.1 解説

グループボックスコントロールは、アイテムをグループ化して他のコントロールと分けます。コントロールを機能ごとに分類する場合などに使用されます。グループボックスでコントロールを分類すると、視覚的に分かりやすくなります。



5.10.2 使い方

グループボックスコントロールを使って、選択項目をラジオボタンやチェックボックスごとに分類してください。他のコントロールとともに分類することもできます。

まず、フォーム上にグループボックスを配置します。

次に、グループボックス上に新しいコントロールを作成するか、既存のコントロールをグループボックスの上にドラッグします。

グループボックスの Text プロパティーにグループの名前を設定します。

グループボックスの外観の設定

BackColor、ForeColor、Font プロパティーを使い、グループボックスの外観を変更します。

5.10.3 グループボックスコントロールのプロパティー

プロパティー	説明
BackColor	コントロールの背景色を設定します。 デフォルト: Control
Enabled	コントロールを有効にするかを設定します。 デフォルト: 1 – True
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト:0 – Normal
Font	コントロールのテキストを表示するフォントを設定します。 (設計時のみ有効) デフォルト: Microsoft Sans Serif 8.25 pt
FontBold	現在のフォントを太字にします。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontItalic	現在のフォントをイタリックにします。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontName	現在のフォントの名前を設定します。 (SPEL プログラムからのみ有効)

プロパティー	説明
FontSize	現在のフォントのサイズをポイントで設定します。 (SPEL プログラムからのみ有効) デフォルト: 8.25
ForeColor	コントロールのテキストの色を設定します。 デフォルト: ControlText
Height	コントロールの高さをピクセル数で設定します。
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
Name	コントロールの名前を設定します。 デフォルト: GroupBoxXX
TabIndex	コントロールのタブインデックスを設定します。
Text	コントロールの表示テキストを設定します。 デフォルト: コントロールの名前
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白
Top	コントロールの表示位置の上の座標をピクセル数で設定します。
Visible	コントロールを表示するか、隠すかを設定します。 デフォルト: 1 – True
Width	コントロールの幅をピクセル数で設定します。

5.10.4 グループボックスコントロールのイベント

イベント	説明
Click	コンロトールがクリックされたときに実行されます。

5.11 タイマーコントロール

5.11.1 解説

タイマーコントロールは、一定間隔でイベントを実行します。
非常停止や他の通常タスクが一時停止しているときも実行します。

5.11.2 使い方

タイマーコントロールを使って、定期的に SPEL+プログラムを実行できます。
例えば、タイマーコントロールを使って 2 秒毎にステータスラベルを更新できます。

タイマーコントロールの使い方

1. フォーム上にタイマーコントロールを追加します。
   このコントロールは、実行時には見えません。コントロールはデザインエリアのフォームの下に表示されます。
2. 実行する時間の間隔を、Interval プロパティーにミリ秒単位で設定します。
3. タイマーイベントを常に実行する場合は、Enabled プロパティーを True に設定します。または、実行時に、Enabled プロパティーを True に設定します。
4. Tick イベントにイベントハンドラーを追加して、SPEL+プログラムを実行します。

NOTE Tick イベントハンドラーの実行中は、同じタイマーコントロールからの Tick イベントは無視されます。

5.11.3 タイマーコントロールのプロパティー

プロパティー	説明
Enabled	コントロールを有効にするかを設定します。 デフォルト: 0 – False
Interval	イベントを実行する間隔を、ミリ秒単位で設定します。 デフォルト: 100
Name	コントロールの名前を設定します。 デフォルト: TimerXX

5.11.4 タイマーコントロールのイベント

イベント	説明
Tick	設定された時間間隔に達したときにイベントが実行されます。

5.12 ビデオボックスコントロール

5.12.1 解説

ビデオボックスコントロールは、フォームに VisionGuide オプションの画像を表示します。

5.12.2 使い方

フォームの上にビデオボックスコントロールを使って、簡単にアプリケーションにビデオウィンドウを表示できます。ビジョンシーケンスを実行する場合に、シーケンス結果をウィンドウに表示できます。

以下のステップで、ビジョンウィンドウを作成します:

1. フォーム上の表示したい場所に、ビデオボックスコントロールを配置します。
   コントロールのサイズは、フルサイズまで変更が可能です。
2. VideoEnabled プロパティーを True に設定します。
3. 画像処理の経過を表示する場合は、GraphicsEnabled プロパティーを True に設定します。
4. Camera プロパティーのデフォルト値は 0 です。
   画像処理シーケンス実行時に、カメラ画像を表示します。Camera プロパティーにカメラの番号を設定した場合、そのカメラの処理結果を表示します。

カメラ画像は、ビデオボックスコントロールのサイズに合わせて、自動的にスケーリングされます。ビデオボックスコントロールの高さや幅を変更するとき、その縦横比は一定に維持されます。

ビデオボックスコントロールの外観の設定

BorderStyle プロパティーを使って、ビデオボックスコントロールの外観を変更します。

5.12.3 ビデオボックスコントロールのプロパティー

プロパティー	説明
BorderStyle	コントロールのボーダーのスタイルを設定します。 デフォルト: 0 –None
Camera	画像を表示するカメラの番号を設定します。 デフォルト: 0
Enabled	コントロールを有効にするかを設定します。 デフォルト: 1 –True
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト: 0 –Normal
GraphicsEnabled	画像処理結果を表示するかを設定します。 デフォルト: 0 –False
Height	コントロールの高さをピクセル数で設定します。
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
Name	コントロールの名前を設定します。 デフォルト: VideoBoxXX
TabIndex	コントロールのタブインデックスを設定します。
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白

プロパティー	説明
Top	コンとロールの表示位置の上の座標をピクセル数で設定します。
VideoEnabled	画像を表示させるかを設定します。 デフォルト: 0 - False
Visible	コントロールを表示するか、隠すかを設定します。 デフォルト: 1 -True
Width	コントロールの幅をピクセル数で設定します。

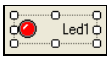
5.12.4 ビデオボックスコントロールのイベント

イベント	説明
Click	コンロトールがクリックされたときに実行されます。
DblClick	コンロトールがダブルクリックされたときに実行されます。

5.13 LED コントロール

5.13.1 解説

LED コントロールは、I/O ステータス表示に使用します。



5.13.2 使い方

LED コントロールを使用します。

1. Text プロパティーにステータスの名前を設定します。
2. IOType プロパティーを設定します。input、output、memoryI/O(入力、出力、メモリー I/O)から選択できます。モニターする I/O の入出力タイプを設定します。
3. IOBit プロパティーを設定します。ステータスを表示するビット番号を設定します。

出力の場合に、AllowStateChange プロパティーを True に設定すると、コントロールをダブルクリックしたときに出力を反転させることができます。

LED コントロールの外観の設定

LED コントロールの外観を変更するには、次のプロパティーを使います。
BackColor、BorderStyle、ForeColor、Font、ImageAlign、TextAlign

また、ImageOn、ImageOff プロパティーを使って、LED 表示が組み込まれたカラーイメージから選択したり、ユーザー独自のイメージを表示することができます。

5.13.3 LED コントロールのプロパティー

プロパティー	説明
AllowStateChange	LED をダブルクリックしたとき、出力を反転させる動作を許可するかを設定します。 デフォルト: 0 – False
BackColor	コントロールの背景色を設定します。 デフォルト: Control
BorderStyle	コントロールのボーダーのスタイルを設定します。 デフォルト: 0 – None
Enabled	コントロールを有効にするかを設定します。 デフォルト: 1 – True
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト: 0 – Normal
Font	コントロールのテキストを表示するフォントを設定します。 (設計時のみ有効) デフォルト: Microsoft Sans Serif 8.25 pt
FontBold	現在のフォントを太字にします。 (SPEL プログラムからのみ有効) デフォルト: 0 – False

プロパティー	説明
FontItalic	現在のフォントをイタリック表示します。 (SPEL プログラムからのみ有効) デフォルト: 0 – False
FontName	現在のフォントの名前を設定します。 (SPEL プログラムからのみ有効) デフォルト: Microsoft Sans Serif
FontSize	現在のフォントのサイズをポイントで設定します。 (SPEL プログラムからのみ有効) デフォルト: 8.25
ForeColor	コントロールのテキストの色を設定します。 デフォルト: ControlText
Height	コントロールの高さをピクセル数で設定します。
ImageAlign	コントロールに表示する画像の配置を設定します。 デフォルト: 4 - MiddleLeft
ImageOff	I/O がオフのときの表示を設定します。 デフォルト: LedOff.ico
ImageOn	I/O がオンのときの表示を設定します。 デフォルト: LedRed.ico
IOBit	範囲: 0～9999 デフォルト: 0
IOType	モニターする I/O の入出力タイプを設定します。 デフォルト: 0 – Input
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
Name	コントロールの名前を設定します。 デフォルト: LedXX
TabIndex	コントロールのタブインデックスを設定します。
Text	コントロールの表示テキストを設定します。 デフォルト:コントロールの名前
TextAlign	テキストの配置を設定します。 デフォルト: 6 – MiddleRight
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白
Top	コントロールの表示位置の上の座標をピクセル数で設定します。
Visible	コントロールを表示するか、隠すかを設定します。 デフォルト: 1 – True
Width	コントロールの幅をピクセル数で設定します。

5.13.4 LED コントロールのイベント

イベント	説明
DblClick	コンロトールがダブルクリックされたときに実行されます。

5.14 ステータスバーコントロール

5.14.1 解説

ステータスバーコントロールは、テキストパネルの表示や、オプションパネル(日時, 非常停止状態, 安全扉状態, ロボット情報)の表示に使用します。

5.14.2 使い方

ステータスバーコントロールを使用して、任意のテキストや、その他の組み込みステータスを表示します。テキストパネルは常に表示されます。Text プロパティーを使用して、テキストパネルに表示する文字列を設定します。他のステータスパネル (日時, 非常停止状態, ロボット情報, 安全扉状態)はデフォルトでは表示されません。
ShowDateTime, ShowEStop, ShowRobot, および ShowSafeguard プロパティーを使用して、これらのパネルを表示または非表示にします。これらのパネルは、システムによって自動的に更新されます。

5.14.3 ステータスバーコントロールのプロパティー

プロパティー	説明
BackColor	コントロールの背景色を設定します。 デフォルト: Control
BorderStyle	コントロールのボーダーのスタイルを設定します。 デフォルト: 0 – None
Dock	コントロールのドッキング位置を設定します。 デフォルト: 2 –Bottom
Enabled	コントロールを有効にするかを設定します。 デフォルト: 1 – True
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト: 0 – Normal
Font	コントロールのテキストを表示するフォントを設定します。 (設計時のみ有効) デフォルト: Microsoft Sans Serif 8.25 pt
ForeColor	コントロールのテキストの色を設定します。 デフォルト: ControlText
Height	コントロールの高さをピクセル数で設定します。
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
Name	コントロールの名前を設定します。 デフォルト: StatusBarXX
RobotNumber	ShowRobot に使用するロボット番号を設定します。
ShowDateTime	現在日時表示の有無を設定します。 デフォルト: 0 – False
ShowEStop	EStop 状態表示の有無を設定します。 デフォルト: 0 – False
ShowRobot	RobotNumber で指定したロボットの表示有無を設定します。 デフォルト: 0 – False
ShowSafeguard	安全扉状態表示の有無を設定します。 デフォルト: 0 – False
TabIndex	コントロールのタブインデックスを設定します。

プロパティー	説明
Text	コントロールの表示テキストを設定します。 デフォルト: コントロールの名前
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白
Top	コントロールの表示位置の上の座標をピクセル数で設定します。
Visible	コントロールを表示するか、隠すかを設定します。 デフォルト: 1 – True
Width	コントロールの幅をピクセル数で設定します。

5.14.4 ステータスバーコントロールのイベント

イベント	説明
Click	コンロトールがクリックされたときに実行されます。

5.15 プログレスバーコントロール

5.15.1 解説

プログレスバーコントロールは、処理の進行状況表示に使用します。

5.15.2 使い方

プログレスバーコントロールを使用して、実行時間の長い操作のステータスや変数の値を表示します。

SPEL+ のグローバル変数の表示方法

Variable プロパティーを設定することによって、自動的に SPEL+グローバル変数の値を表示することができます。

注意
Variable プロパティー設定で SPEL+のグローバル変数が表示されない場合、プロジェクトをビルドしてください。

5.15.3 プログレスバーコントロールのプロパティー

プロパティー	説明
BackColor	コントロールの背景色を設定します。 デフォルト: Control
BorderStyle	コントロールのボーダーのスタイルを設定します。 デフォルト: 0 – None
Dock	コントロールのドッキング動作を設定します。 デフォルト 0 –None
Enabled	コントロールを有効にするかを設定します。 デフォルト: 1 – True
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト: 0 – Normal
ForeColor	コントロールのテキストの色を設定します。 デフォルト: ControlText
Height	コントロールの高さをピクセル数で設定します。
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
Maximum	コントロールが処理している範囲の上限を設定します。 デフォルト: 100
Minimum	コントロールが処理している範囲の下限を設定します。 デフォルト: 0
Name	コントロールの名前を設定します。 デフォルト: ProgressBarXX
Orientation	コントロールの向きを設定します。 デフォルト: 0 – Horizontal
ProgressBarStyle	コントロールのスタイルを設定します。 デフォルト: 0 – Blocks
Step	コントロールの現在の値をインクリメントする量を設定します。 デフォルト 10
TabIndex	コントロールのタブインデックスを設定します。

プロパティー	説明
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白
Top	コントロールの表示位置の上の座標をピクセル数で設定します。
Value	コントロールの現在位置を表す値を設定します。 デフォルト: 50
Variable	表示させたい SPEL+のグローバル変数を設定します。 デフォルト: None
Visible	コントロールを表示するか、隠すかを設定します。 デフォルト: 1 – True
Width	コントロールの幅をピクセル数で設定します。

5.15.4 プログレスバーコントロールのイベント

イベント	説明
Click	コンロトールがクリックされたときに実行されます。
DblClick	コンロトールがダブルクリックされたときに実行されます。

5.16 トラックバーコントロール

5.16.1 解説

トラックバーコントロールは、スライダーをドラッグして値を設定して、その設定値を表示するために使用します。

5.16.2 使い方

値をグラフィカルに変更したい場合に、トラックバーコントロールを使用します。

SPEL+のグローバル変数の表示方法

Variable プロパティーを使用すると、トラックバーの値に従って、自動で SPEL+グローバル変数の値を設定できます。

注意
Variable プロパティー設定で、SPEL+のグローバル変数が更新されない場合、プロジェクトをビルドしてください。

5.16.3 トラックバーコントロールのプロパティー

プロパティー	説明
BackColor	コントロールの背景色を設定します。 デフォルト: Control
Enabled	コントロールを有効にするかを設定します。 デフォルト: 1 – True
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト: 0 – Normal
Height	コントロールの高さをピクセル数で設定します。
LargeChange	マウスのクリックや、Page Up および Page Down キーを押す時、スライダーが移動するポジションの数を設定します。 デフォルト: 5
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
Maximum	コントロールが処理している範囲の上限を設定します。 デフォルト: 10
Minimum	コントロールが処理している範囲の下限を設定します。 デフォルト: 0
Name	コントロールの名前を設定します。 デフォルト: TrackBarXX
Orientation	コントロールの向きを設定します。 デフォルト: 0 – Horizontal
SmallChange	キーボードの入力 (方向キー)に対して、スライダーが移動するポジションの数を設定します。 デフォルト: 1
TabIndex	コントロールのタブインデックスを設定します。
TickFrequency	目盛りマーク間の位置の数を設定します。 デフォルト: 1
TickStyle	目盛りをトラックバーのどこに表示するかを設定します。 デフォルト: 2 - BottomRight

プロパティー	説明
ToolTipText	コントロールのツールチップで表示されるテキストを設定します。 デフォルト: 空白
Top	コントロールの表示位置の上の座標をピクセル数で設定します。
Value	コントロールの現在位置を表す値を設定します。 デフォルト: 0
Variable	表示させたい SPEL+のグローバル変数を設定します。 デフォルト: None
Visible	コントロールを表示するか、隠すかを設定します。 デフォルト: 1 – True
Width	コントロールの幅をピクセル数で設定します。

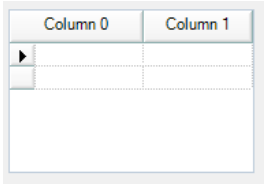
5.16.4 トラックバーコントロールのイベント

イベント	説明
Scroll	トラックバーのスライダーが移動したときに実行されます。

5.17 グリッドコントロール

5.17.1 解説

グリッドコントロールは、データをスプレッドシート方式で表示・編集するために使用します。



5.17.2 使い方

グリッドコントロールには、行と列のデータを含むセルがあります。オペレーターは、行と列を選択でき、指定した範囲のセルを自由に編集できます。

デザイン時:

1. GridEditor プロパティーをクリックし、[GridEditor]ダイアログを開きます。
2. 適切な値にプロパティーを設定し、グリッドを設計します。
3. ダイアログを閉じます。

実行時:

- CellTex プロパティーを使って、セルの文字列を読み書きできます。
- AddRow プロパティーと、RemoveRow プロパティーを使って、行の追加や削除ができます。
- CellForeColor プロティーと、CellBackColor プロパティーを使って、セルの ForeColor と BackColor を変更できます。

5.17.3 グリッドコントロールのプロパティー

プロパティー	説明
AddRow	実行時に行を追加します。
BorderStyle	コントロールの背景色を設定します。 デフォルト: Control
CellBackColor	セルの背景色を設定・取得します。
CellForeColor	セルの ForeColor を設定・取得します。
CellText	セル内の文字列を設定・取得します。
Enabled	コントロールを有効にするかを設定します。 デフォルト: 1 - True
EventTaskType	イベントで起動するタスクタイプを設定します。 デフォルト: 0 - Normal
GridEditor	[GridEditor]ダイアログを開いて、グリッドコントロールを設定します。
Height	コントロールの高さをピクセル数で設定します。
Left	コントロールの表示位置の左の座標をピクセル数で設定します。
Name	コントロールの名前を設定します。 デフォルト: GridXX
RemoveRow	行を削除します。

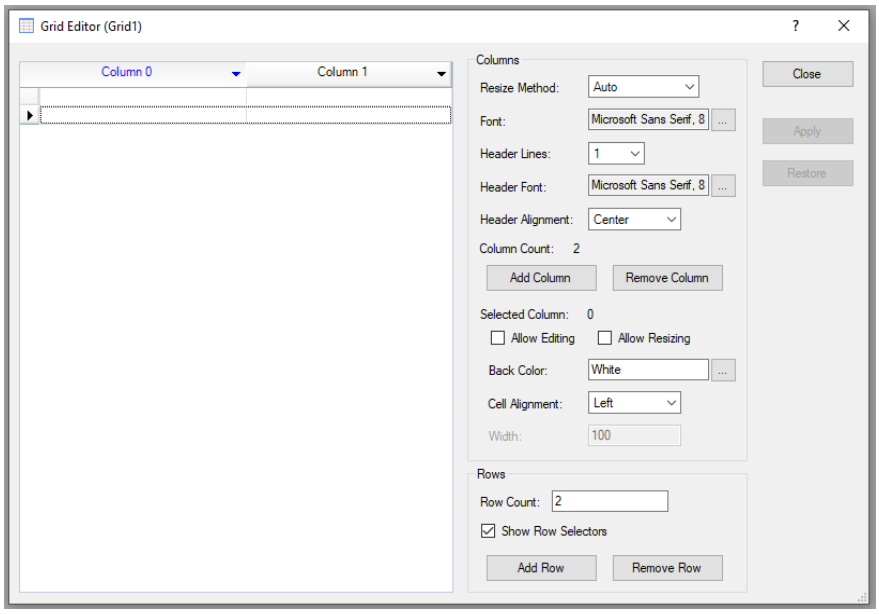
RowCount	行数を設定・取得します。
ScrollBars	スクロールバーを表示するときに設定します。 デフォルト: 0 - None
SelectedIndex	選択した行を設定・取得します。
Top	コントロールの表示位置の上の座標をピクセル数で設定します。
Visible	コントロールの表示・非表示を設定します。 デフォルト: 1 - True
Width	コントロールの幅をピクセル数で設定します。

5.17.4 グリッドコントロールのイベント

イベント	説明
CellChanged	変更したセルを離れたときに実行されます。
Click	コントロールがクリックされたときに実行されます。
DblClick	コントロールがダブルクリックされたときに実行されます。

5.17.5 グリッドエディター

グリッドエディターは、ユーザーがグリッドコントロールを修正・設定できるダイアログです。デザイン時のみ、使用できます。最適な設定にするために、多くのプロパティーがあります。グリッドエディターの詳細は、GridEditor プロパティーを参照してください。



6. 操作

6.1 概要

本章では、以下の項目を説明します:

- プログラムモードで GUI 開発
- オートモードで GUI をスタートアップに登録する方法
- 一時停止、継続実行の操作
- 非常停止の操作
- ヘルプファイルの使い方

6.2 プログラムモードで GUI 開発

GUI アプリケーションの開発とデバッグはプログラムモードで行います。

まず、プログラムモードで EPSON RC+ 7.0 を起動してください。

6.2.1 GUI 設計

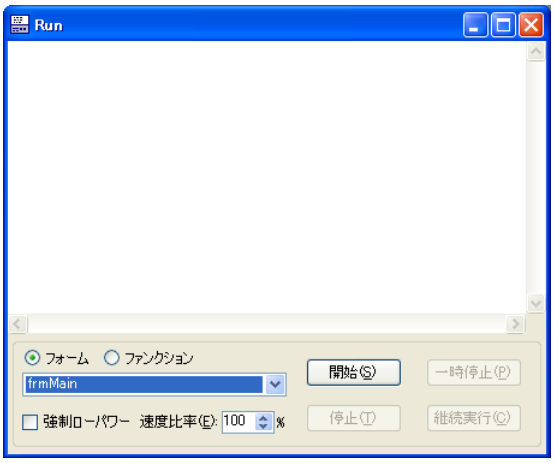
GUI アプリケーションを設計するために、簡単なガイドラインを示します。

1. アプリケーションのために 1 つ以上のフォームを作成してください。
   通常、そのうちの 1 つをメインフォームとして設計します。
2. GUI をどのように起動するかを決めてください。
   スタートアップフォームとして、ひとつのフォームを指定することができます。このフォームは、EPSON RC+をオペレーターモードで起動すると自動的に表示されます。
   起動時にスタートアップフォームを表示したくないときは、オペレーターウィンドウが表示された後、SPEL+プログラムから GShow ステートメントまたは GShowDialog 関数を使ってフォームを表示することができます。
3. フォームにコントロールを追加します。
4. コントロールに必要なイベントを追加します。
5. フォームとコントロールの使い方については、「4.5 フォームとコントロールの使い方」を参照してください。

6.2.2 デバッグ

Run ウィンドウから、プロジェクト内のフォームの実行が可能です。

1. Run ウィンドウを開きます。



2. ドロップダウンリストから実行したいフォームを選択します。
3. <開始>ボタンをクリックすると、選択されたフォームが表示され、コントロールを使うことができます。
4. 停止するには、フォームを閉じるか、Run ウィンドウの<停止>ボタンをクリックします。

SPEL+タスクで実行される GUI イベントハンドラーは、イベントハンドラー中にブレークポイントの設定や、プログラムのステップ実行、変数の値の確認ができます。

6.3 オペレーターモード

アプリケーションをオペレーターモードで実行するには、EPSON RC+のセットアップが必要です。コントローラー起動時に、GUI フォームの 1 つを表示するか、オペレーターウィンドウを表示するか選択した後、プログラムから GUI フォームを表示するかを選択できます。

EPSON RC+をオペレーターモードで起動する

1. セットアップメニューから[システム設定]を選択します。
2. [スタートモード]画面で、<オペレーターモード>ラジオボタンをチェックし、<適用>、<閉じる>ボタンをクリックします。

起動時に表示するフォームの設定

1. GUI Builder ウィンドウを開きます。
2. フォームエクスプローラーから、起動時に表示したいフォームを右クリックし、[スタートアップフォーム]を有効にします。
3. ツールバーの<プロジェクトの保存>ボタンをクリックします。

6.4 一時停止、継続実行の操作

一時停止や継続実行を、GUI アプリケーションから操作することもできます。
通常、フォーム上に一時停止ボタンや継続実行ボタンを配置します。ボタンが有効に動作するように、EventTaskType プロパティーを 1 – NoPause に設定してください。

これは、コントローラの一時停止状態の中で、イベントハンドルのプログラムを実行する必要があるためです。

```
Function frmMain_btnPause_Click(Sender$ As String)
  Pause
Fend

Function frmMain_btnCont_Click(Sender$ As String)
  Cont
Fend
```

6.5 非常停止の操作

非常停止が入力された後に、ロボットを再度動作させるためには、非常停止状態をリセットする必要があります。

そのためには、リセットボタンの EventTaskType プロパティーを 2 – NoEmgAbort に設定してください。

```
Function frmMain_btnReset_Click(Sender$ As String)
  Reset
  If EstopOn Then
    MsgBox "EStop could not be reset"
  EndIf
Fend
```

6.6 ヘルプファイルの使い方

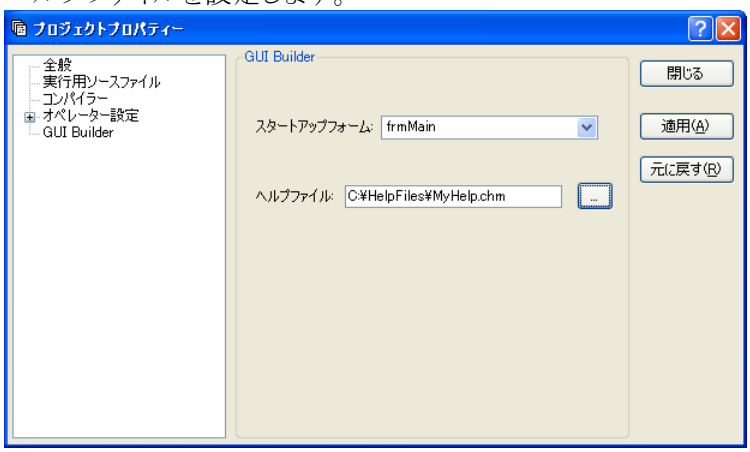
GUI Builder は分かりやすいヘルプ機能をサポートしています。オペレーターがヘルプボタンをクリックしたときに表示するヘルプファイル、トピックを設定します。

ヘルプファイルは、マイクロソフト HTML ヘルプフォーマットで作成してください。

ヘルプファイルを使用するために

プロジェクトメニュー-[プロパティー]-[GUI Builder]を選択します。

ヘルプファイルを設定します。



各フォームからヘルプを表示するには、HelpButton プロパティーを True にして、HelpID プロパティーに表示したいトピックの ID を設定します。

7. GUI Builder リファレンス

7.1 概要

本章では、GUI Builder のフォーム、コントロールのプロパティーとイベントについて説明します。また、GUI Builder に関係するすべての SPEL⁺コマンドについても説明します。

GUI Builder の使用についての詳しい情報は、GUI Builder マニュアルを参照してください。

7.2 GUI Builder プロパティーとイベント書式

GUI Builder のプロパティーとイベントを以降のページで挙げています。

プロパティーやリザルトリファレンスに関する説明は次のとおりです。

適用	プロパティーやイベントを適用できる GUI Builder オブジェクトを表示します。 (例. Button、Label、CheckBox…) プロパティーやイベントがフォームで使用されるときは、Form を表示します。
解説	各プロパティーやイベントについての簡潔な説明です。
用法	SPEL⁺言語からプロパティーやイベントにアクセスする方法です。
値	プロパティーに設定できる値、戻り値の範囲が記載されています。 デフォルト値がある場合、その値も記載されます。
詳細説明	解説の内容より詳しい説明です。それぞれのプロパティー特有の注意事項や特記事項が記載されていますので、そのプロパティーをご使用になる前に必ずこの項目をお読みください。
参照	関連するプロパティー、リザルト、オブジェクト、トピックをあげています。
プログラム実行時のみ	プロパティーまたはリザルトがプログラム実行中のみに適用しているとき、プロパティー名かリザルト名の下に記載されています。プログラム実行中のみのプロパティーとリザルトは、GUI Builder からはアクセスできません。SPEL⁺プログラムからのみアクセスすることができます。

AcceptButton プロパティー

適用

Form

説明

Enter キーが押されたときにクリックイベントを実行するボタンを設定・取得します。

使い方

GGet *Form*.**AcceptButton**, *var*

GSet *Form*.**AcceptButton**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

var　　プロパティーの値を取得する文字列変数を指定します。

value　　プロパティーに設定する文字列式を指定します。

値

フォーム上のボタンの名前

デフォルト: None

注記

設計した GUI 上で、Enter キーが押されたときに実行するデフォルトの動作を指定できます。カレントフォーム上、またはカレントフォーム上のコンテナにあるボタンを指定します。

このプロパティーを指定することで、ユーザーがマウス操作で操作を終了するかわりに、Enter キーを押すことで操作することを可能にします。

コントロールによっては、Enter キー入力を別の操作が優先する場合があります。例えば複数行入力のテキストボックスは、Enter キーが押された場合、新しい行を追加します。

参照

Form, Button, CancelButton

使用例

```
GSet frmMain.AcceptButton, "btnOK"
```

AddItem プロパティー

適用

ListBox, ComboBox

説明

リストボックスやコンボボックスコントロールに、アイテムを追加します。

使い方

GSet *Form.Control.***AddItem***, value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

value　プロパティーに設定する文字列式を指定します。

値

リストに追加する文字列です。

参照

ComboBox, ListBox, List, ListCount, Sorted

使用例

```
GSet frmMain.lstModels.AddItem, "Model1"
```

AddRow プロパティー

適用

Grid

説明

グリッドコントロールに、行を追加します。

使い方

GSet *Form.Control.***AddRow***, value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

value　新しく追加した行のインデックスを表す整数値です。“-1”を指定すると、新しい行が最終行の次に追加されます。それ以外の場合、新しい行は、指定した行のインデックスに追加されます。

参照

Grid, RemoveRow, RowCount

使用例

```
GSet frmMain.Grid01.AddRow, 0
```

AppendText プロパティー

適用

TextBox

説明

テキストボックスコントロールに文字列を追加します。

使い方

GSet *Form.Control.***AppendText***, value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

value　プロパティーに設定する文字列式を指定します。

値

コントロールの最後に追加される文字列です。

参照

TextBox

使用例

```
GSet frmMain.txtStatus.AppendText, "Cycle Complete"
```

AllowStateChange プロパティー

適用

LED

説明

LED コントロールをダブルクリックしたとき、出力を反転させる動作を許可するかどうかを設定、取得します。

使い方

GGet *Form.Control.***AllowStateChange**, *var*

GSet *Form.Control.***AllowStateChange**, *value*

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

var プロパティーの値を取得する論理変数を指定します。

value プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

注記

IOType プロパティーが 0 – Input に設定されている場合は、AllowStateChange プロパティーを 1 – True に設定しても、出力を反転できません。出力を反転させるには、IOType プロパティーを 1 – Output または 2 – Memory に設定してください。

参照

LED, IOType

使用例

```
GSet frmMain.Led1.AllowStateChange, True
```

BackColor プロパティー

適用

Form, Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, PictureBox, GroupBox, LED, ProgressBar, TrackBar, StatusBar

説明

フォーム、またはコントロールの背景色を設定・取得します。

使い方

GGet *Form.Control*.**BackColor**, *var*

GSet *Form.Control*.**BackColor**, *value*

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var プロパティーの値を取得する文字列変数を指定します。

value プロパティーに設定する文字列式を指定します。

値

フォーム、またはコントロールの背景色に指定する色の名前です。

デフォルト: Control (Form, Button, Label, RadioButton, CheckBox, PictureBox, GroupBox, LED)
Window (TextBox, ListBox, ComboBox)

参照

BackColorMode, Form, Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, PictureBox, GroupBox, LED, ProgressBar, TrackBar, StatusBar

使用例

```
GSet frmMain.lblStatus.BackColor, "Red"
```

BackColorMode プロパティー

適用

Button

説明

ボタンコントロールの背景色モードを設定・取得します。

使い方

GGet *Form.Control*.**BackColorMode**, *var*

GSet *Form.Control*.**BackColorMode**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する整数変数を指定します。

value　　プロパティーに設定する整数式を指定します。

値

0 – VisualStyle　　システムで設定されている背景色を使います。

1 – User　　ユーザーが指定する背景色を使います。

デフォルト: 0 – Visual Style

注記

BackColorMode が 0 のとき、Windows の画面のプロパティーで設定された背景色になります。

BackColorMode が 1 のとき、背景色は BackColor プロパティーで指定された色になります。

参照

BackColor, Button

使用例

```
GSet frmMain.btnOK.BackColorMode, BACKCOLORMODE_USER
```

BorderStyle プロパティー

適用

Label, TextBox, ListBox, PictureBox, VideoBox, LED, ProgressBar, StatusBar, Grid

説明

コントロールのボーダースタイルを設定・取得します。

使い方

GGet *Form.Control*.**BorderStyle**, *var*

GSet *Form.Control*.**BorderStyle**, *value*

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var プロパティーの値を取得する整数変数を指定します。

value プロパティーに設定する整数式を指定します。

値

0 – None

1 – FixedSingle

2 – Fixed3D

デフォルト: 0 – None (Label, PictureBox, VideoBox, LED)
2 – Fixed3D (TextBox, ListBox)

注記

コントロールのボーダースタイルを設定するのに使います。例えば、アプリケーションの状態を表示するラベルを、他のラベルと異なるボーダースタイルで表示することができます。

参照

Label, TextBox, ListBox, PictureBox, VideoBox, LED, ProgressBar, StatusBar

使用例

```
GSet frmMain.lblStatus.BorderStyle, BORDERSTYLE_NONE
```

Camera プロパティー

適用

VideoBox

説明

ビデオボックスコントロールのカメラ番号を設定・取得します。

使い方

GGet *Form.Control*.**Camera**, *var*

GSet *Form.Control*.**Camera**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

ビデオボックスに表示するカメラ番号です。値が“0”の時は、VRun コマンドが実行されたときのカメラの画像が表示されます。画像を表示するときは、VideoEnabled プロパティーを “1 – True”に設定してください。

参照

VideoBox, VideoEnabled

使用例

```
GSet frmMain.VideoBox1.Camera, 1
```

CancelButton プロパティー

適用

Form

説明

ESC キーが押されたときにクリックイベントを実行するボタンを設定・取得します。

使い方

GGet *Form*.**CancelButton**, *var*

GSet *Form*.**CancelButton**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

var　　プロパティーの値を取得する文字列変数を指定します。

value　　プロパティーに設定する文字列式を指定します。

値

フォーム上のボタンの名前

デフォルト: None

注記

キャンセルボタンは、ユーザーが Esc キーを押すときにクリックされるボタンコントロールです。カレントフォーム上、またはカレントフォーム上のコンテナにあるボタンを指定します。

設計した GUI 上で Esc キーが押されたときに実行するデフォルトの動作を指定できます。このプロパティーを指定することで、ユーザーがマウス操作で操作を終了するかわりに、Esc キーを押すことで操作することを可能にします。

コントロールによっては、Esc キー入力を別の操作が優先する場合があります。例えばコンボボックスのドロップダウンリストが開いている場合は、Esc キーが押された場合、コンボボックスのドロップダウンリストを閉じます。

キャンセルボタンに割りあてたボタンを可視状態にしなければ、Esc キーの入力は無視されます。

参照

Form, Button, AcceptButton

使用例

```
GSet frmMain.CancelButton, "btnCancel"
```

CellBackColor プロパティー

適用

Grid

説明

グリッドセルの背景色を取得・設定します。

使い方

GGet *Form.Control*.**CellBackColor** (*row*, *column*), *var*

GSet *Form.Control*.**CellBackColor** (*row*, *column*), *value*

Form	フォームの名前、またはフォームの名前を含む文字列変数を指定します。
Control	コントロールの名前、またはコントロールの名前を含む文字列を指定します。
row	セル行の整数式を指定します。最初の行のインデックスは“0”です。
column	セル列の整数式を指定します。最初の列のインデックスは“0”です。
var	プロパティーの値を取得する文字列変数を指定します。
value	プロパティーに設定する文字列式を指定します。

値

指定したグリッドセルの、背景色の名称を含む文字列です。

参照

Grid, CellForeColor, CellText

使用例

```
GSet frmMain.gridMyData.CellBackColor(1, 1), "Red"
```

CellChanged イベント

適用

Grid

説明

変更したセルを離れたときに、発生するイベントです。

使い方

*Form_Control_***CellChanged** (*sender$* As String, *CellText$* As String, *RowNumber* As Integer, *ColumnNumber* As Integer)

Sender$	イベントを発生させたコントロールの名前が設定されます。
CellText$	変更したセルのテキストが設定されます。
RowNumber	変更したセルの行番号が設定されます。
ColumnNumber	変更したセルの列番号が設定されます。

注記

CellChanged イベントは、ユーザーが変更したセルを離れたときに、応答するために使われます。Sender$パラメーターを使用し、イベントを送信したコントロールを判別します。CellText$パラメーターを使用し、セルの現在値を確認します。RowNumber パラメーターと ColumnNumber パラメーターを使用して、イベントのトリガーであるセルを判別します。これを使うことで、特定のセルや任意のセルが、いつ変更されたかを確認するときに役に立ちます。

参照

Grid

使用例

```
Function frmMain_Grid1_CellChanged(Sender$ As String, CellText$ As
String, RowNumber As Integer, ColumnNumber As Integer)

Xqt main
Fend
```

CellForeColor プロパティー

適用

Grid

説明

グリッドセルのテキストの色を設定・取得します。

使い方

GGet *Form.Control*.**CellForeColor** (*row*, *column*), *var*

GSet *Form.Control*.**CellForeColor** (*row*, *column*), *value*

Form	フォームの名前、またはフォームの名前を含む文字列変数を指定します。
Control	コントロールの名前、またはコントロールの名前を含む文字列を指定します。
row	セル行の整数式を指定します。最初の行のインデックスは 0 です。
column	セル列の整数式を指定します。最初の列のインデックスは 0 です。
var	プロパティーの値を取得する文字列変数を指定します。
value	プロパティーに設定する文字列式を指定します。

値

指定したグリッドセルの、テキストの色の名称を含む文字列です。

参照

Grid, CellBackColor, CellText

使用例

```
GSet frmMain.gridMyData.CellForeColor(1, 1), "Red"
```

CellText プロパティー

適用

Grid

説明

グリッドセルのテキストを取得・設定します。

使い方

GGet *Form.Control*.**CellText** (*row*, *column*), *var*

GSet *Form.Control*.**CellText** (*row*, *column*), *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

row　セル行の整数式を指定します。最初の行のインデックスは 0 です。

column　セル列の整数式を指定します。最初の列のインデックスは 0 です。

var　プロパティーの値を取得する文字列変数を指定します。

value　プロパティーに設定する文字列式を指定します。

値

特定のグリッドセルのテキストを含む文字列です。

参照

Grid, CellForeColor, CellBackColor

使用例

```
GSet frmMain.gridMyData.CellText(1, 1), "Value1"
```

Checked プロパティー

適用

RadioButton, Checkbox

説明

チェックボックスか、ラジオボタンコントロールがチェックされているかどうか設定・取得します。

使い方

GGet *Form.Control*.**Checked**, *var*

GSet *Form.Control*.**Checked**, *value*

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var プロパティーの値を取得する論理変数を指定します。

value プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

注記

このプロパティーは、チェックボックスやラジオボタンコントロールがチェックされているか判断するために使います。チェックボックスやラジオボタンコントロールを使って ON/OFF の状態を表示するために使うこともできます。

参照

RadioButton, CheckBox

使用例

```
GSet frmMain.chkHiPower.Checked, False
```

Click イベント

適用

Button, Label, RadioButton, Checkbox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, ProgressBar, StatusBar , Grid

説明

コントロールがクリックされたときに実行されるイベントです。

使い方

*Form_Control*_**Click** (*Sender$* As String)

Sender$　　イベントを発生させたコントロールの名前が設定されます。

注記

クリックイベントは、ユーザーがコントロールをクリックしたときの応答をするために使われます。Sender$パラメーターにより、どのコントロールがイベントを発生させたか判断できます。これを使うことで、ラジオボタンやチェックボックスのように、複数のコントロールを 1 つのファンクション内で使用したいときに役に立ちます。

ハンドリングを、1 つの関数で作成することが容易になります。

参照

Button, Label, RadioButton, Checkbox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, ProgressBar, StatusBar

使用例

```
Function frmMain_btnStart_Click(Sender$ As String)
  Xqt main
Fend
```

Closed イベント

適用

Form

説明

フォームが閉じられたときに実行されるイベントです。

使い方

*Form*_**Closed** (*Sender$*As String)

Sender$　　イベントを発生させたフォームの名前が設定されます。

注記

このイベントは、フォームが閉じられたあとタスクを実行するときに使用します。

参照

Form、Load、Resize

使用例

```
Function frmMain_Closed(Sender$ As String)

  Print "frmMain was closed"
Fend
```

ControlBox プロパティー

適用

Form

説明

フォームのキャプションバーに、コントロールボックスを表示するかどうかを設定・取得します。

使い方

GGet *Form*.**ControlBox**, *var*

GSet *Form*.**ControlBox**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

var　　プロパティーの値を取得する論理変数を指定します。

value　　プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 1 – True

注記

ControlBox プロパティーが 1 – True に設定された場合、コントロールボックスがフォームのタイトルバーの左上に表示されます。コントロールボックスは、フォームを閉じるのに使うことができます。

参照

Form, MaximizeBox, MinimizeBox

使用例

```
GSet frmMain.ControlBox, False
```

Controls プロパティー

適用

Form

説明

フォーム上のコントロールの配列。インデックスを使用してコントロールのプロパティーにアクセスする場合に使用します。

使い方

GGet *Form*.**Controls(***Index* As Integer).*Property*, *var*

GSet *Form*. **Controls(***Index* As Integer).*Property*, *value*

Form	フォームの名前、またはフォームの名前を含む文字列変数を指定します。
Index	フォーム上のコントロールのインデックスを含む整数式を指定します。
Property	アクセスするコントロールのプロパティーの名前を指定します。
var	プロパティーの値を取得する変数を指定します。データ型は指定するプロパティーによって異なります。
value	プロパティーに設定する式を指定します。データ型は指定するプロパティーによって異なります。

注記

Controls プロパティーは、インデックスを使用してフォーム上のコントロールにアクセスします。これにより、コントロールの集合を順に処理し、共通するプロパティーを設定または取得することが出来ます。

参照

Count, Type

使用例

```
Integer i, count
String type$

GGet frmMain.Controls.Count, count
For i = 1 To count
   GGet frmMain.Controls(i).Type, type$
   If type$ = "Button" Then
      GSet frmMain.Controls(i).Enabled, False
   EndIf
Next i
```

Count プロパティー

適用

Form.Controls

説明

フォーム上のコントロールの数を取得します。

使い方

GGet *Form*.Controls.**Count**, *var*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

var　　プロパティーの値を取得する変数を指定します。

値

フォーム上のコントロールの数を返します。

注記

Count プロパティーは、フォーム上のコントロールの数を特定するために使用します。これにより、Controls プロパティーを使用して、すべてのコントロールを順に処理することが出来ます。

参照

Controls, Type

使用例

```
Integer count

GSet frmMain.Controls.Count, count
For i = 1 To count
   GGet frmMain.Controls(i).Type, type$
   If type$ = "Button" Then
      GSet frmMain.Controls(i).Enabled, False
   EndIf
Next i
```

DblClick イベント

適用

Label, ListBox, Led, PictureBox, VideoBox, LED, ProgressBar , Grid

説明

コントロールがダブルクリックされたときに実行されるイベントです。

使い方

Form_Control_**DblClick** (*Sender$* As String)

Sender$　　イベントを発生させたコントロールの名前が設定されます。

注記

クリックイベントは、ユーザーがコントロールをクリックしたときの応答をするために使われます。Sender$パラメーターにより、どのコントロールがイベントを発生させたか判断できます。

参照

Label, ListBox, PictureBox, VideoBox, LED, ProgressBar

使用例

```
Function frmMain_lstModels_DblClick(Sender$ As String)
  Integer index

  GGet frmMain.lstModels.SelectedIndex, index

  GGet frmMain.lstModels.List(index), g_CurrModel$
Fend
```

DialogResult プロパティー

適用

Form

説明

フォームのダイアログ結果を設定・取得します。

使い方

GGet *Form*.**DialogResult**, *var*

GSet *Form*.**DialogResult**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

var　　プロパティーの値を取得する整数変数を指定します。

value　　プロパティーに設定する整数式を指定します。

値

0 – None

1 – OK

2 – Cancel

デフォルト: 2 – Cancel

注記

DialogResult プロパティーは、ユーザーがフォームを閉じるときに、了解して閉じたか、キャンセルして閉じたかを判断するために使います。通常、ダイアログは、OK ボタン、またはキャンセルボタンを持ちます。ユーザーが OK ボタンを押したとき、ボタンクリックイベントの中で、DialogResult プロパティーに 1 – OK を設定します。<キャンセル>ボタンが押されたときには、2 – Cancel を設定します。

参照

Form

使用例

```
Function frmSetSpeed_btnOK_Click(Sender$ As String)
  GSet frmSetSpeed.DialogResult, DIALOGRESULT_OK
  GClose frmSetSpeed
Fend
Function frmMain_btnSetSpeed_Click(Sender$ As String)
  Integer result
  String speed$
  result = GShowDialog(frmSetSpeed)
  If result = DIALOGRESULT_OK Then
     GGet frmSetSpeed.txtSpeed.Text, speed$
    g_RobotSpeed = Val(speed$)
Fend
```

Dock プロパティー

適用

ProgressBar, StausBar

説明

ステータスバーのドッキング動作を設定・取得します。

使い方

GGet *Form.Control*.**Dock**, *var*

GSet *Form.Control*.**Dock**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する整数変数を指定します。

value　　プロパティーに設定する整数式を指定します。

値

0 – None

1 – Top

2 – Bottom

3 – Left

4 – Right

5 – Fill

デフォルト: ProgressBar: 0 – None, StatusBar: 2 - Bottom

参照

ProgressBar, StatusBar

使用例

```
GSet frmStatus.ProgressBar1.Dock, DOCK_TOP
```

DropDownStyle プロパティー

適用

ComboBox

説明

コンボボックスのドロップダウンスタイルを設定・取得します。

使い方

GGet *Form.Control*.**DropDownStyle**, *var*

GSet *Form.Control*.**DropDownStyle**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する整数変数を指定します。

value　　プロパティーに設定する整数式を指定します。

値

0 – Simple

1 – DropDown

2 – DropDownList

デフォルト: 1 – DropDown

注記

DropDownStyle プロパティーは、ユーザーへのインターフェイスをコントロールします。
次の中から選択します。
- 常にリストを表示し選択できる
- 右側のドロップダウンボタンで表示したリストから選択でき、テキストボックス内のテキストを編集可能（デフォルト）
- 右側のドロップダウンボタンで表示したリストから選択できるが、テキストボックス内のテキストは編集不可

常にリストが表示されて、テキストの編集を許可しないのであれば、ListBox コントロールを使ってください。

参照

ComboBox

使用例

```
GSet frmMain.cmbPartNames.DropDownStyle, DROPDOWNSTYLE_SIMPLE
```

Enabled プロパティー

適用

Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, PictureBox, GroupBox, Timer, VideoBox, LED, ProgressBar, TrackBar, StatusBar , Grid

説明

コントロールが実行時に有効かどうか、設定・取得します。

使い方

GGet *Form.Control*.**Enabled**, *var*

GSet *Form.Control*.**Enabled**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する論理変数を指定します。

value　プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 1 – True (Timer 以外)
0 – False (Timer)

注記

あるケースでは、ボタンのようなコントロールをオペレーターにクリックさせたくない場合があります。例えば、SPEL+のタスクが実行されているときに、<Pause>ボタンを押させたくないときは、Enabled プロパティーを 0 – False に設定してください。

参照

Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, PictureBox, GroupBox, Timer, VideoBox, LED, ProgressBar, TrackBar, StatusBar

使用例

```
GSet frmMain.btnPause.Enabled, False
```

EventTaskType プロパティー

適用

Form, Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar , Grid

説明

イベントで起動するタスクタイプを取得します。

使い方

GGet *Form*.[*Control*].**EventTaskType**, *var*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する整数変数を指定します。

値

0 – Normal

1 – NoPause

2 – NoEmgAbort

デフォルト: 0 – Normal

注記

EventTaskType プロパティーを 1 – NoPause に設定すると、他の通常タスクが一時停止しているときに、イベントハンドラーを実行できるようになります。2 – NoEmgAbort に設定すると、他の通常タスクが一時停止しているときや非常停止状態でも、イベントハンドラーを実行できるようになります。

参照

Form, Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar

使用例

```
Integer Ttype
GGet frmMain.btnPause.EventTaskType, Ttype
```

Font プロパティー

設計 GUI 上のみの機能です。

適用

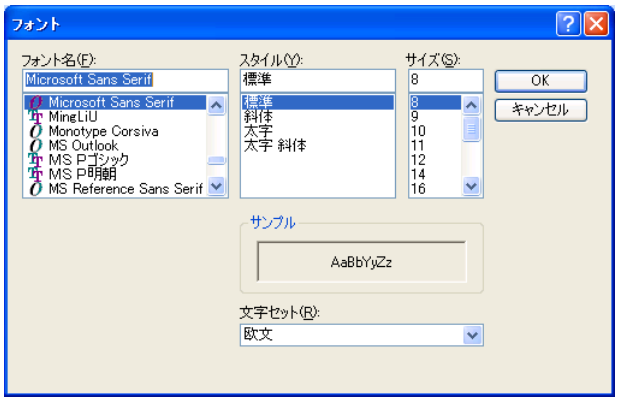
Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, GroupBox, LED, StatusBar

説明

フォーム設計時に、コントロールのフォントを変更することができます。(フォント名、スタイル、サイズ)

使い方

GUI Builder のプロパティーグリッドで、コントロールを選択して、Font プロパティーの値をクリックすると、次のようなダイアログが表示されます。



フォントの設定を変更した後、<OK>ボタンをクリックすると設定が保存されます。

参照

Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, GroupBox, LED, StatusBar

FontBold プロパティー

適用

Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, GroupBox, LED, StatusBar

説明

フォーム設計時に、フォントが“標準”または“太字”かを設定・取得します。

使い方

GGet *Form.Control*.**FontBold**, *var*

GSet *Form.Control*.**FontBold**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する論理変数を指定します。

value　　プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

参照

Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, GroupBox, LED, StatusBar, FontName, FontItalic, FontSize

使用例

```
GSet frmMain.btnOK.FontBold, True
```

FontItalic プロパティー

適用

Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, GroupBox, LED, StatusBar

説明

フォーム設計時に、フォントが“標準”または“イタリック(斜体)”かを設定・取得します。

使い方

GGet *Form.Control*.**FontItalic**, *var*

GSet *Form.Control*.**FontItalic**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する論理変数を指定します。

value　　プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

参照

Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, GroupBox, LED, StatusBar, FontName, FontBold, FontSize

使用例

```
GSet frmMain.btnOK.FontItalic, True
```

FontName プロパティー

適用

Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, GroupBox, LED, StatusBar

説明

コントロールに使われているフォントの名称を設定・取得します。

使い方

GGet *Form.Control*.**FontName**, *var*

GSet *Form.Control*.**FontName**, *value*

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var プロパティーの値を取得する文字列変数を指定します。

value プロパティーに設定する文字列式を指定します。

値

フォントの名称の文字列です。

デフォルト: Microsoft Sans Serif

参照

Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, GroupBox, LED, StatusBar, FontSize, FontItalic, FontBold

使用例

```
GSet frmMain.txtStatus.FontName, "Courier New"
```

FontSize プロパティー

適用

Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, GroupBox, LED, StatusBar

説明

コントロールに使われるフォントのサイズを、設定・取得します。

使い方

GGet *Form.Control*.**FontSize**, *var*

GSet *Form.Control*.**FontSize**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名か、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する実数変数を指定します。

value　プロパティーに設定する実数式を指定します。

値

フォントのサイズのポイント値です。

デフォルト: 8.25

参照

Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, GroupBox, LED, StatusBar, FontName, FontItalic, FontBold

使用例

```
GSet frmMain.btnOK.FontSize, 16
```

ForeColor プロパティー

適用

Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, GroupBox, LED, ProgressBar, StatusBar

説明

コントロールのテキストの色を設定・取得します。

使い方

GGet *Form.Control*.**ForeColor**, *var*

GSet *Form.Control*.**ForeColor**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する文字列変数を指定します。

value　プロパティーに設定する文字列式を指定します。

値

コントロールのテキストの色の名称の文字列です。

デフォルト: Control Text (Button, Label, RadioButton, CheckBox, GroupBox, LED)
Window Text (TextBox, ListBox, ComboBox)

参照

Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, GroupBox, LED, ProgressBar, StatusBar

使用例

```
GSet frmMain.btnOK.ForeColor, "Blue"
```

FormBorderStyle プロパティー

適用

Form

説明

フォームのボーダースタイルを設定・取得します。

使い方

GGet *Form*.**FormBorderStyle**, *var*

GSet *Form*.**FormBorderStyle**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

var　　プロパティーの値を取得する整数変数を指定します。

value　　プロパティーに設定する整数式を指定します。

値

0 – None
1 – Fixed Single
2 – Fixed3D
3 – FixedDialog
4 – Sizeable
デフォルト: 3 – FixedDialog

参照

Form

使用例

```
GSet frmMain.FormBorderStyle, FORMBORDERSYTLE_NONE

GClose
```

GClose ステートメント

適用

Form

説明

フォームを閉じます。

使い方

GClose *Form*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。また、フォーム ID を指定して、以下のシステムウィンドウを閉じることも出来ます。

WIN_IOMON	I/O モニターウィンドウを閉じます。
WIN_TASKMGR	タスクマネージャーウィンドウを閉じます。
WIN_FORCEMON	トルクモニターウィンドウを閉じます。
WIN_SIMULATOR	シミュレーターウィンドウを閉じます。

注記

表示していないフォームに対しては GClose ステートメントは使用しないでください。

参照

GShow, GShowDialog

使用例

```
GClose frmSetup

GClose WIN_TASKMGR
```

GGet ステートメント

説明

SPEL+プログラムからフォームやコントロールのプロパティーを取得するためのステートメントです。

使い方

GGet Form **.**Property, var

GGet Form **.**Control**.**Property, var

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

Property 取得するプロパティーの名前を指定します。

var プロパティーの値を取得する変数を指定します。

注記

GGet ステートメントは、実行時にプロパティーの値を取得するために使います。

参照

GSet

GraphicsEnabled プロパティー

適用

VideoBox

説明

VideoBox コントロールに、画像処理のグラフィックスを表示するかを、設定・取得します。

使い方

GGet *Form.Control*.**GraphicsEnabled**, *var*

GSet *Form.Control*.**GraphicsEnabled**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する論理変数を指定します。

value　　プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

参照

VideoBox, VideoEnabled

使用例

```
GSet frmMain.VideoBox1.GraphicsEnabled, True
```

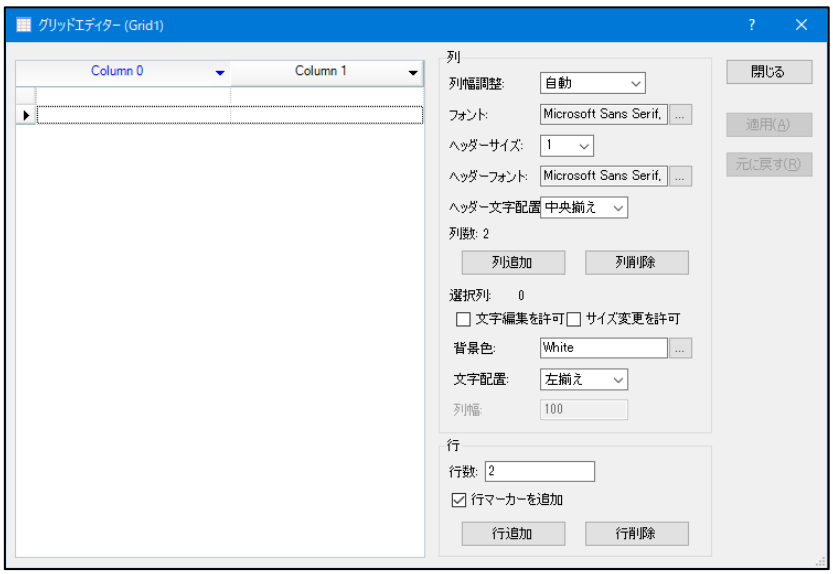
GridEditor プロパティー

適用

Grid

説明

グリッドコントロールをセットアップするため、[GridEditor]ダイアログを開きます。



使い方

GridEditor プロパティーは、デザイン時のみの使用できます。ユーザーの要望に合わせ、グリッドを設定できるようにします。

設定	説明
列追加	グリッドの最後に、列を追加します。
	最小値: 1
	最大値: 25
行追加	グリッドの下部に、行を追加します。
	最小値: 0
	最大値: 10000
文字編集を許可	ユーザーが、選択した列のセルを編集できるかを設定します。
サイズ変更を許可	ユーザーが、選択した列のサイズを変更できるかを設定します。
背景色	選択した列の背景色を設定します。
文字配置	セルテキストの配置を指定します。
	左揃え: セルテキストを列の左側に揃えます。
	中央揃え: セルテキストを列の中央に揃えます。
	右揃え: セルテキストを列の右側に揃えます。
フォント	列のヘッダーのフォントを設定します。
ヘッダー文字配置	列のヘッダーの配置を設定します。
	左揃え: 列名を列の左側に揃えます。
	中央揃え: 列名を列の中央に揃えます。
	右揃え: 列名を列の右側に揃えます。

設定	説明
ヘッダーサイズ	列のヘッダーで、利用できる行数を設定します。 最小値: 1 最大値: 3
列削除	選択した列を削除します。
行削除	選択した行を削除します。
列幅調整	列の追加、削除、サイズ変更するとき、グリッド内の列のサイズ変更方法を決定します。 自動: グリッドコントロールに合わせて、自動で列のサイズを変更します。 手動: 各列をぞれぞれの列幅に変更します。 同期: 列のサイズ変更をするとき、すべての列を同じサイズに変更します。
行数	グリッドの行数を設定・表示します。手動で値を変更するには、該当の行に入力して、<適用>ボタンをクリックします。
行マーカーを表示	グリッドの左側にある、行選択を表示するかを設定します。
列幅	選択した列の幅を設定・表示します。手動で値を変更するには、列幅調整を自動にしないでください。

注記

[GridEditor]ダイアログを使って、グリッドと初回の表示方法を設定します。設計時にセルにテキストを追加したり、データの追加ができます。

列は、ドラッグ&ドロップで移動したい位置に動かせます。列のヘッダーにある、ドロップダウン矢印を使って、他の列と交換することもできます。

列のヘッダーをダブルクリックすると、列の名称を変更できます。

右クリックのメニューとショートカットを使って、切り取り, コピー, 貼り付け, 元に戻す, やり直しの操作ができます。行マーカーを表示が True に設定されている場合は、複数行を選択できます。

参照

Grid

GSet ステートメント

説明

SPEL+プログラムからフォームやコントロールのプロパティーを設定するためのステートメントです。

使い方

GSet Form **.**Property, value

GSet Form **.**Control**.**Property, value

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

Property 設定するプロパティーの名前を指定します。

value プロパティーの値を設定する式を指定します。

注記

GSet ステートメントは、実行時にプロパティーの値を設定するために使います。

参照

GGet

GShow ステートメント

適用

Form

説明

フォームをモードレス ウインドウとして表示します。

使い方

GShow *Form*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。また、フォーム ID を指定して、以下のシステムウィンドウを表示することも出来ます。

WIN_IOMON	I/O モニターウィンドウを開きます。
WIN_TASKMGR	タスクマネージャーウィンドウを開きます。
WIN_FORCEMON	トルクモニターウィンドウを開きます。
WIN_SIMULATOR	シミュレーターウィンドウを開きます。

参照

GClose, GShowDialog

使用例

```
GShow frmIODiags

GShow WIN_TASKMGR
```

GShowDialog ファンクション

適用

Form

説明

フォームをモーダル ダイアログとして表示し、DialogResult の値を返します。

使い方

GShowDialog(*Form*)

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

戻り値

DialogResult プロパティーの値を返します。

参照

GShow, GShowDialog ステートメント

使用例

```
result = GShowDialog(frmSetup)
If result = DIALOGRESULT_OK Then
  Call SaveSettings
EndIf
```

GShowDialog ステートメント

適用

Form

説明

DialogResult の値を返さずに、フォームをモーダル ダイアログとして表示します。

使い方

GShowDialog *Form*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

参照

GShow, GShowDialog Function ファンクション

使用例

```
GShowDialog frmInfoDisplay
```

Height プロパティー

適用

Form, Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar, Grid

説明

フォームやコントロールの高さをピクセル値で設定・取得します。

使い方

GGet *Form*.[*Control*].**Height**, *var*

GSet *Form*.[*Control*].**Height**, *value*

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var プロパティーの値を取得する整数変数を指定します。

value プロパティーに設定する整数式を指定します。

値

高さを表すピクセル整数値です。

参照

Form, Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar, Left, Top, Width

使用例

```
GSet frmMain.btnOK.Height, 48
```

HelpButton プロパティー

適用

Form

説明

フォームのタイトルバーに<Help>ボタンを表示するかを設定・取得します。

使い方

GGet *Form*.**HelpButton**, *var*

GSet *Form*.**HelpButton**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

var　　プロパティーの値を取得する論理変数を指定します。

value　　プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

注記

このプロパティーを 1 – True に設定すると、クエスションマークの小さなボタンが、タイトルバーの<閉じる>ボタンの左に表示されます (MaximizeBox, MinimizeBox プロパティーが False のとき)。このボタンを使って、ヘルプを表示することができます。

参照

Form, HelpID

使用例

```
GSet frmMain.HelpButton, True
```

HelpID プロパティー

適用

Form

説明

ヘルプファイルのトピックの HelpID を設定・取得します。

使い方

GGet *Form*.**HelpID**, *var*

GSet *Form*.**HelpID**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

var　　プロパティーの値を取得する整数変数を指定します。

value　　プロパティーに設定する整数式を指定します。

値

ヘルプファイルのトピック ID の整数値です。(0~999999)

デフォルト: 0

注記

HelpID は、文脈依存のヘルプを表示させるために使います。ヘルプファイルは、EPSON RC+メニュー-[プロジェクト]-[プロパティー]-[GUI Builder]から設定します。

HelpID に 0 以外が設定されている場合、F1 キーを押すか、フォームのヘルプボタンをクリックすると(HelpButton プロパティーが True に設定されているとき)、ヘルプトピックを表示できます。

参照

Form, HelpButton

使用例

```
GSet frmMain.HelpID, 50
```

Image プロパティー

適用

Button, Label, RadioButton, CheckBox, PictureBox

説明

コントロールに表示するイメージを設定・取得します。

使い方

GGet *Form.Control*.**Image**, *var*

GSet *Form.Control*.**Image**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する文字列変数を指定します。

value　プロパティーに設定する文字列式を指定します。

値

イメージファイルのフルパスの文字列です。

デフォルト: 空白

注記

Delete キーで設定したイメージファイルを削除することができます。

参照

Button, Label, RadioButton, Checkbox, PictureBox, ImageAlign

使用例

```
GSet frmMain.btnTools.Image, "c:\Images\Tools.bmp"
```

ImageAlign プロパティー

適用

Button, Label, RadioButton, CheckBox, LED

説明

コントロールに表示するイメージの配置を設定・取得します。

使い方

GGet *Form.Control*.**ImageAlign**, *var*

GSet *Form.Control*.**ImageAlign**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

1 – TopLeft

2 – TopCenter

3 – TopRight

4 – MiddleLeft

5 – MiddleCenter

6 – MiddleRight

7 – BottomLeft

8 – BottomCenter

9 – BottomRight

デフォルト: 5 – MiddleCenter (Button, Label, RadioButton, CheckBox)
4 – MiddleLeft (LED)

参照

Button, Label,RadioButton, Checkbox, LED, Image, TextAlign

使用例

```
GSet frmMain.btnTools.ImageAlign, IMAGEALIGN_MIDDLECENTER
```

ImageOff プロパティー

適用

LED

説明

I/O がオフのときに表示するイメージを設定・取得します。

使い方

GGet *Form.Control*.**ImageOff**, *var*

GSet *Form.Control*.**ImageOff**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する文字列変数を指定します。

value　　プロパティーに設定する文字列式を指定します。

値

イメージファイルのフルパスの文字列です。拡張子が、bmp、ico、jpeg、gif、wmf、png のファイルを指定できます。

デフォルト: LedOff.ico

参照

LED, ImageOn

使用例

```
GSet frmMain.Led1.ImageOff, "c:\EpsonRC70\GUI\Icons\LedOff.ico"
```

ImageOn プロパティー

適用

LED

説明

I/O がオンのときに表示するイメージを設定・取得します。

使い方

GGet *Form.Control*.**ImageOn**, *var*

GSet *Form.Control*.**ImageOn**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する文字列変数を指定します。

value　プロパティーに設定する文字列式を指定します。

値

イメージファイルのフルパスの文字列です。拡張子が、bmp、ico、jpeg、gif、wmf、png のファイルを指定できます。

デフォルト: LedRed.ico

参照

LED, ImageOff

使用例

```
GSet frmMain.Led1.ImageOn, "c:\EpsonRC70\GUI\Icons\LedRed.ico"
```

Interval プロパティー

適用

Timer

説明

タイマー間隔を msec (ミリ秒)単位で設定・取得します。

使い方

GGet *Form.Control*.**Interval**, *var*

GSet *Form.Control*.**Interval**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

タイマー間隔を msec (ミリ秒)で指定します。1~9999 の整数を設定します。

デフォルト: 100

注記

タイマー間隔を秒で取得したいときは、1000 で割ってください。

参照

Timer, Enabled

使用例

```
GSet frmMain.tmrMain.Interval, 500
```

IOBit プロパティー

適用

LED

説明

状態を表示したい I/O ビットを設定・取得します。

使い方

GGet *Form.Control*.**IOBit**, *var*

GSet *Form.Control*.**IOBit**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

I/O ビットを表す整数値です。(範囲: 0~9999)

デフォルト: 0

参照

LED, IOType

使用例

```
GSet frmMain.Led1.IOBit, 10
```

IOType プロパティー

適用

LED

説明

状態を表示したい I/O の入出力タイプを設定・取得します。

使い方

GGet *Form.Control*.**IOType**, *var*

GSet *Form.Control*.**IOType**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する整数変数を指定します。

value　　プロパティーに設定する整数式を指定します。

値

0 - Input

1 – Output

2 – Memory

デフォルト: 0 - Input

参照

LED, IOBit

使用例

```
GSet frmMain.Led1.IOType, IOTYPE_OUTPUT
```

KeyPress イベント

適用

TextBox

説明

キーが押されたときに実行されるイベントです。

使い方

*Form_Control*_**KeyPress** (*Sender$* As String, ByRef *Key$* As String)

Sender$　　イベントを発生させたコントロールの名前が設定されます。

Key$　　押されたキーの文字列です。

注記

Keypress イベントは、ユーザーのキー入力を直接読みたいときに使います。

また、このイベントをフィルターとして使うことも可能です。Key$に他の文字を代入すると、入力されたキー文字列を変更することができます。Key$に空の文字列を代入すると、キー入力をキャンセルすることができます。

参照

TextBox

使用例

```
Function frmMain_txtSpeed_KeyPress(Sender$ As String, ByRef Key$ As String)

  ' 数値のみ入力可能にする
  If Instr("0123456789" Key$) < 0 Then
    Key$ = ""
  EndIf
Fend
```

LargeChange プロパティー

適用

TrackBar

説明

マウスのクリックや、Page Up および Page Down キーを押す時、スライダーが移動するポジションの数を設定・取得します。

使い方

GGet *Form.Control*.**LargeChange**, *var*

GSet *Form.Control*.**LargeChange**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

スライダーが移動するポジションの数を表す整数値です。

参照

TrackBar, SmallChange

使用例

```
GSet frmMain.TrackBar1.LargeChange, 10
```

Left プロパティー

適用

Form, Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar, Grid

説明

フォームやコントロールの左側の座標をピクセル値で設定・取得します。

使い方

GGet *Form*.[*Control*].**Left**, *var*

GSet *Form*.[*Control*].**Left**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

左側の座標を表すピクセル整数値です。

参照

Form, Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar, Top, Height, Width

使用例

```
GSet frmMain.btnOK.Left, 200
```

List プロパティー

適用

ListBox, ComboBox

説明

指定されたコントロールのリストアイテムの値を文字列で設定・取得します。

使い方

GGet *Form.Control.***List**(*index*)*, var*

GSet *Form.Control.***List**(*index*)*, value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

index　0 から始まるリストのインデックス番号を、整数式で指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する文字列式を指定します。

値

リストアイテムの文字列です。

注記

リストボックスやコンボボックスに AddItem プロパティーでアイテムを追加したあとに、アイテムの値を取得したり変更したりすることができます。

空の文字列を設定することで、アイテムを削除することができます。

参照

ComboBox, ListBox, ListCount, AddItem

使用例

```
String part$
GGet frmMain.lstPartNames.List(0), part$
```

ListCount プロパティー

適用

ListBox, ComboBox

説明

リストボックスやコンボボックスのアイテム数を設定・取得します。

使い方

GGet *Form.Control.***ListCount***, var*

GSet *Form.Control.***ListCount**, *value*

Form	フォームの名前、またはフォームの名前を含む文字列変数を指定します。
Control	コントロールの名前、またはコントロールの名前を含む文字列を指定します。 そのコントロールは、指定されたフォーム上に存在しなくてはなりません。.
var	プロパティーの値を取得する整数変数を指定します。
value	プロパティーに設定する文字列式を指定します。

値

リストのアイテム数です。

注記

ListCount を“0”に設定することで ListBox や ComboBox のアイテムを消去できます。ListCount にリストのアイテム数を超える値を設定すると、追加分のアイテムに空のテキストが追加されます。

参照

ComboBox, ListBox, AddItem, List

使用例

```
Integer count

' リスト内のアイテム数を取得
GGet frmMain.lstPartNames.ListCount, count

' リスト内のアイテムを消去
GSet frmMain.lstPartName.ListCount, 0
```

Load イベント

適用

Form

説明

フォームが表示される前に発生するイベントです。

使い方

*Form*_**Load** (*Sender$* As String)

Sender$　　イベントが発生したフォームの名前です。

注記

フォームが表示される前に、SPEL+プログラムを実行するときに使います。

参照

Form, Resize, Closed

使用例

```
Function frmMain_Load(Sender$ As String)

  GSet frmMain.txtSpeed.Text, Str$(g_RobotSpeed)
Fend
```

MaximizeBox プロパティー

適用

Form

説明

フォームのタイトルバーに<最大化>ボタンを表示するかを、設定・取得します。

使い方

GGet *Form*.**MaximizeBox**, *var*

GSet *Form*.**MaximizeBox**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

var　　プロパティーの値を取得する論理変数を指定します。

value　　プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

注記

フォームを画面いっぱいに最大化するときに使います。

参照

Form, MinimizeBox

使用例

```
GSet frmMain.MaximizeBox, True
```

Maximum プロパティー

適用

ProgressBar, TrackBar

説明

コントロールの処理範囲の最大値を設定・取得します。

使い方

GGet *Form.Control.***Maximum***, var*

GSet *Form.Control.***Maximum**, *value*

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。.

var プロパティーの値を取得する整数変数を指定します。

value プロパティーに設定する文字列式を指定します。

値

処理範囲の最大値を表す整数値です。

参照

ProgressBar, TrackBar, Minimum, Value

使用例

```
GSet frmMain.TrackBar1.Maximum, 60
```

MinimizeBox プロパティー

適用

Form

説明

フォームのタイトルバーに<最小化>ボタンを表示するかを、設定・取得します。

使い方

GGet *Form*.**MinimizeBox**, *var*

GSet *Form*.**MinimizeBox**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

var　　プロパティーの値を取得する論理変数を指定します。

value　　プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

注記

フォームを最小化するときに使います。

参照

Form, MaximizeBox

使用例

```
GSet frmMain.MinimizeBox, True
```

Minimum プロパティー

適用

ProgressBar, TrackBar

説明

コントロールの処理範囲の最小値を設定・取得します。

使い方

GGet *Form.Control.***Minimum***, var*

GSet *Form.Control.***Minimum**, *value*

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。.

var プロパティーの値を取得する整数変数を指定します。

value プロパティーに設定する文字列式を指定します。

値

処理範囲の最小値を表す整数値です。

参照

ProgressBar, TrackBar, Maximum, Value

使用例

```
GSet frmMain.TrackBar1.Minimum, 10
```

MultiLine プロパティー

適用

TextBox

説明

テキストボックスコントロールを複数行にするかを設定・取得します。

使い方

GGet *Form.Control*.**MultiLine**, *var*

GSet *Form.Control*.**MultiLine**, *value*

Form	フォームの名前、またはフォームの名前を含む文字列変数を指定します。
Control	コントロールの名前、またはコントロールの名前を含む文字列を指定します。 そのコントロールは、指定されたフォーム上に存在しなくてはなりません。
var	プロパティーの値を取得する論理変数を指定します。
value	プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

注記

テキストボックスコントロールを複数行で使うときに、MultiLine プロパティーを 1 – True に設定します。

参照

TextBox, ScrollBars

使用例

```
GSet frmMain.txtStatus.MultiLine, True
```

Name プロパティー

適用

Form, Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, PictureBox, GroupBox, Timer, VideoBox, LED, ProgressBar, TrackBar, StatusBar , Grid

説明

設計時は、フォームやコントロールの名前を設定します。
実行時は、名前を取得します。

使い方

GGet *Form*.[*Control*].**Name**, *var*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する文字列変数を指定します。

値

フォームやコントロールの名前を含んだ文字列です。

参照

Form, Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, PictureBox, GroupBox, Timer, VideoBox, LED, ProgressBar, TrackBar, StatusBar

使用例

```
String name$
GGet frmMain.btnOK.Name, name$
```

Orientation プロパティー

適用

ProgressBar, TrackBar

説明

コントロールの向きを設定・取得します。

使い方

GGet *Form.Control*.**Orientation**, *var*

GSet *Form.Control*.**Orientation**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する文字列変数を指定します。

value　　プロパティーに設定する文字列式を指定します。

値

0 – Horizontal

1 – Vertical

デフォルト: 0 – Horizontal

参照

ProgressBar, TrackBar

使用例

```
GSet frmMain.ProgressBar1.Orientation, ORIENT_VERTICAL
```

PasswordChar プロパティー

適用

TextBox

説明

1 行のテキストボックスで、パスワード入力のときに表示する文字を設定・取得します。

使い方

GGet *Form.Control*.**PasswordChar**, *var*

GSet *Form.Control*.**PasswordChar**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

var　プロパティーの値を取得する文字列変数を指定します。

value　プロパティーに設定する文字列式を指定します。

値

マスクに使う文字を含む文字列です。

注記

パスワード入力時に入力した文字を隠すための文字です。パスワード入力時には、この文字が表示されます。MultiLine プロパティーは、0 – False に設定してください。

参照

TextBox

使用例

```
GSet frmMain.txtPassword.PasswordChar, "*"
```

ProgressBarStyle プロパティー

適用

ProgressBar

説明

プログレスバーのスタイルを設定・取得します。

使い方

GGet *Form.Control*.**ProgressBarStyle**, *var*

GSet *Form.Control*.**ProgressBarStyle**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する文字列変数を指定します。

value　　プロパティーに設定する文字列式を指定します。

値

0 – Blocks

1 – Continuous

2 – Marquee

デフォルト: 0 - Blocks

参照

ProgressBar

使用例

```
GSet frmMain.ProgressBar1.ProgressBarStyle, PROGRESSBAR_STYLE_MARQUEE
```

ReadOnly プロパティー

適用

TextBox

説明

テキストボックスを編集不可にするかどうかを設定・取得します。

使い方

GGet *Form.Control*.**ReadOnly**, *var*

GSet *Form.Control*.**ReadOnly**, *value*

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

var プロパティーの値を取得する論理変数を指定します。

value プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

注記

オペレーターにテキストボックスを編集させたくない場合に使用します。

参照

TextBox

使用例

```
GSet frmMain.txtSpeed.ReadOnly, True
```

RemoveRow プロパティー

適用

Grid

説明

グリッドコントロールから、行を削除します。

使い方

GSet *Form.Control.***RemoveRow***, value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

value　削除される行のインデックスを表す整数値です。

値

削除される行のインデックスを表す整数値です。

参照

AddRow, Grid, RowCount

使用例

```
GSet frmMain.Grid01.RemoveRow, 0
```

Resize イベント

適用

Form

説明

フォームのサイズが変更されたときに発生するイベントです。

使い方

*Form*_**Resize** (*Sender$* As String)

Sender$　　イベントを発生させたフォームの名称が入ります。

参照

Form, Load, Closed

使用例

```
Function frmMain_Resize(Sender$ As String)
  Integer width

  GGet frmMain.Width, width
  GSet frmMain.btnOK.Left, width / 2 - 100
  GSet frmMain.btnCancel.Left, width / 2 + 10
Fend
```

RobotNumber プロパティー

適用

StatusBar

説明

ShowRobot プロパティーが有効の場合、表示したいロボット情報のロボット番号を設定・取得します。

使い方

GGet *Form.Control.***RobotNumber***, var*

GSet *Form.Control.***RobotNumber**, *value*

Form	フォームの名前、またはフォームの名前を含む文字列変数を指定します。
Control	コントロールの名前、またはコントロールの名前を含む文字列を指定します。 そのコントロールは、指定されたフォーム上に存在しなくてはなりません。
var	プロパティーの値を取得する整数変数を指定します。
value	プロパティーに設定する整数式を指定します。

値

ロボット番号を表す整数値です。

デフォルト: 1

参照

StatusBar, ShowRobot

使用例

```
GSet frmMain.StatusBar1.RobotNumber, 2
```

RowCount プロパティー

適用

Grid

説明

グリッドコントロールの行数を設定・取得します。

現在の行数より大きな値を設定した場合、新しい行が末尾に追加されます。

現在の行数より小さな値を設定した場合、末尾から行が削除されます。

使い方

GGet Form.Control.**RowCount,** var

GSet *Form.Control.***RowCount***, value*

Form	フォームの名前、またはフォームの名前を含む文字列変数を指定します。
Control	コントロールの名前、またはコントロールの名前を含む文字列を指定します。 そのコントロールは、指定されたフォーム上に存在しなくてはなりません。
var	コントロールの行数を含む整数変数を指定します。
value	コントロールの行の総数を表す整数式を指定します。

値

グリッドの行数を指定する整数値です。

参照

AddRow, Grid, RemoveRow, RowCount

使用例

```
Integer count
GSet frmMain.Grid01.RowCount, 50
GGet frmMain.Grid01.RowCount, count
```

Scroll イベント

適用

TrackBar

説明

スライダーが移動したときに発生するイベントです。

使い方

*Form_Control_***Scroll** (*Sender$* As String)

Sender$　　イベントを発生させたコントロールの名前が設定されます。

参照

TrackBar

使用例

```
Function frmMain_TrackBar1_Scroll(Sender$ As String, Value As Integer)
  Print "The trackbar value after scroll is", Value
Fend
```

ScrollBars プロパティー

適用

TextBox, Grid

説明

複数行のテキストボックスに、スクロールバーを表示させるかを設定・取得します。

使い方

GGet *Form.Control*.**ScrollBars**, *var*

GSet *Form.Control*.**ScrollBars**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

0 – None

1 – Horizontal

2 – Vertical

3 – Both

デフォルト: 0 – None

注記

スクロールバーを使うときは、MultiLine プロパティーを 1 – True に設定してください。

水平スクロールバーを表示するときは、WordWrap プロパティーを 0 – False に設定してください。

参照

TextBox, Multiline, WordWrap

使用例

```
GSet frmMain.txtStatus.Scrollbars, SCROLLBARS_VERT
```

SelectedIndex プロパティー

適用

ListBox, ComboBox, Grid

説明

現在選択されているアイテムのインデックス番号を設定・取得します。

使い方

GGet *Form.Control*.**SelectedIndex**, *var*

GSet *Form.Control*.**SelectedIndex**, *value*

Form	フォームの名前、またはフォームの名前を含む文字列変数を指定します。
Control	コントロールの名前、またはコントロールの名前を含む文字列を指定します。 そのコントロールは、指定されたフォーム上に存在しなくてはなりません。
var	プロパティーの値を取得する整数変数を指定します。
value	プロパティーに設定する整数式を指定します。

値

選択されたアイテムの番号を 0 から始まる整数で返します。
アイテムが選択されていないときは –1 を返します。

注記

SelectedIndex プロパティーは、リストボックスやコンボボックスの選択されているアイテムを判断するときに使います。多くは、コントロールのクリックイベント処理の中で使います。

参照

AddItem, ComboBox, List, ListBox, Grid

使用例

```
Integer index
GGet frmMain.lstParts.SelectedIndex, index
```

ShowDateTime プロパティー

適用

StatusBar

説明

現在日時表示の有無を設定・取得します。

使い方

GGet *Form.Control*.**ShowDateTime**, *var*

GSet *Form.Control*.**ShowDateTime**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

var　プロパティーの値を取得する論理変数を指定します。

value　プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

参照

StatusBar, ShowEStop, ShowRobot, ShowSafeguard

使用例

```
GSet frmMain.StatusBar1.ShowDateTime, True
```

ShowEStop プロパティー

適用

StatusBar

説明

Estop 状態表示の有無を設定・取得します。

使い方

GGet *Form.Control*.**ShowEStop**, *var*

GSet *Form.Control*.**ShowEStop**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

var　プロパティーの値を取得する論理変数を指定します。

value　プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

参照

StatusBar, ShowDateTime, ShowRobot, ShowSafeguard

使用例

```
GSet frmMain.StatusBar1.ShowEStop, True
```

ShowPrint プロパティー

適用

TextBox

説明

Print ステートメントの出力を表示するかを設定・取得します。

使い方

GGet *Form.Control*.**ShowPrint**, *var*

GSet *Form.Control*.**ShowPrint**, *value*

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

var プロパティーの値を取得する論理変数を指定します。

value プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

Default: 0 – False

注記

ShowPrint が True の場合、Spel タスクで実行された Print ステートメントのすべての出力がテキストボックスに表示されます。複数のテキストボックスで ShowPrint が True に設定されている場合、すべてのテキストボックスにプリント出力が表示されます。

参照

TextBox, ScrollBars

使用例

```
GSet frmMain.txtStatus.ShowPrint, True
```

ShowRobot プロパティー

適用

StatusBar

説明

RobotNumber プロパティーで指定したロボット表示の有無を設定・取得します。

使い方

GGet *Form.Control*.**ShowRobot**, *var*

GSet *Form.Control*.**ShowRobot**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

var　プロパティーの値を取得する論理変数を指定します。

value　プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

参照

StatusBar, RobotNumber, ShowDateTime, ShowEStop, ShowSafeguard

使用例

```
GSet frmMain.StatusBar1.ShowRobot, True
```

ShowSafeguard プロパティー

適用

StatusBar

説明

安全扉状態表示の有無を設定・取得します。

使い方

GGet *Form.Control*.**ShowSafeguard**, *var*

GSet *Form.Control*.**ShowSafeguard**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

var　プロパティーの値を取得する論理変数を指定します。

value　プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

参照

StatusBar, ShowDateTime, ShowEStop, ShowRobot

使用例

```
GSet frmMain.StatusBar1.ShowSafeguard, True
```

SizeMode プロパティー

適用

PictureBox

説明

イメージの表示方法を設定、取得します。

使い方

GGet *Form.Control*.**SizeMode**, *var*

GSet *Form.Control*.**SizeMode**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

0 – Normal

1 – StretchImage

2 – AutoSize

3 – CenterImage

4 – Zoom

デフォルト: 0 – Normal

注記

Normal (デフォルト)　:　イメージはピクチャーボックスの左上に表示されます。
ピクチャーボックスよりイメージが大きい場合、その部分は表示されません。

StretchImage　:　イメージは、ピクチャーボックスに合わせて拡大縮小されます。

AutoSize　:　イメージに合わせて、ピクチャーボックスを拡大縮小します。

CenterImage　:　イメージは、ピクチャーボックスの中央に配置されます。

Zoom　:　イメージは、ピクチャーボックスに合わせて縦横比を固定して拡大縮小されます。

参照

PictureBox, Image, ImageAlign

使用例

```
GSet frmMain.picLogo.SizeMode, SIZEMODE_AUTOSIZE
```

SmallChange プロパティー

適用

TrackBar

説明

キーボードの入力 (方向キー)に対して、スライダーが移動するポジションの数を設定・取得します。

使い方

GGet *Form.Control*.**SmallChange**, *var*

GSet *Form.Control*.**SmallChange**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

スライダーが移動するポジションの数を表す整数値です。

参照

TrackBar, LargeChange

使用例

```
GSet frmMain.TrackBar1.SmallChange, 5
```

Sorted プロパティー

適用

ComboBox, ListBox

説明

リストボックスやコンボボックスのアイテムを、アルファベット順にソートするかを設定・取得します。

数字 (全角半角なし)、英字、カタカナ、ひらがな、漢字の順にソートされます。

使い方

GGet *Form.Control*.**Sorted**, *var*

GSet *Form.Control*.**Sorted**, *value*

Form	フォームの名前、またはフォームの名前を含む文字列変数を指定します。
Control	コントロールの名前、またはコントロールの名前を含む文字列を指定します。 そのコントロールは、指定されたフォーム上に存在しなくてはなりません。
var	プロパティーの値を取得する論理変数を指定します。
value	プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

参照

ComboBox, ListBox

使用例

```
GSet frmMain.lstParts.Sorted, True
```

StartPosition プロパティー

適用

Form

説明

フォームを実行するときの開始位置を設定・取得します。

使い方

GGet *Form*.**StartPosition**, *var*

GSet *Form*.**StartPosition**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

var　　プロパティーの値を取得する整数変数を指定します。

value　　プロパティーに設定する整数式を指定します。

値

0 – Manual

1 – CenterScreen

2 – CenterParent

デフォルト: 1 – CenterScreen

参照

Form, WindowState

使用例

```
GSet frmMain.StartPosition, STARTPOSITION_CENTERSCREEN
```

Step プロパティー

適用

ProgressBar

説明

プログレスバーの現在値をインクリメントする量を設定・取得します。

使い方

GGet *Form.Control*.**Step**, *var*

GSet *Form.Control*.**Step**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

現在値をインクリメントする量を表す整数値です。

参照

ProgressBar

使用例

```
GSet frmMain.ProgressBar1.Step, 5
```

TabIndex プロパティー

適用

Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar , Grid

説明

コンテナ内のコントロールのタブオーダーを設定・取得します。

使い方

GGet *Form.Control*.**TabIndex**, *var*

GSet *Form.Control*.**TabIndex**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

コントロールのタブインデックス番号の整数値です。

参照

Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar

使用例

```
GSet frmMain.txtStatus.TabIndex, 3
```

Text プロパティー

適用

Form, Button, Label, TextBox, RadioButton, CheckBox, ComboBox, GroupBox, LED, StatusBar

説明

コントロールに関連づけられたテキストを設定・取得します。

使い方

GGet *Form*.[*Control*].**Text**, *var*

GSet *Form*.[*Control*].**Text**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する文字列変数を指定します

value　　プロパティーに設定する文字列式を指定します。.

値

フォームやコントロールのテキストを含む文字列です。

参照

Form, Button, Label, TextBox, RadioButton, Checkbox, ComboBox, GroupBox, LED, StatusBar, TextAlign

使用例

```
GSet frmMain.lblName.Text, "Name: "
```

TextAlign プロパティー

適用

Button, Label, TextBox, RadioButton, CheckBox, LED

説明

コントロールのテキストの位置情報を設定・取得します。

使い方

GGet *Form.Control*.**TextAlign**, *var*

GSet *Form.Control*.**TextAlign**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

ボタン、ラベル、ラジオボタン、チェックボックスコントロール

1 – TopLeft
2 – TopCenter
3 – TopRight
4 – MiddleLeft
5 – MiddleCenter
6 – MiddleRight
7 – BottomLeft
8 – BottomCenter
9 – BottomRight

デフォルト:　1 – TopLeft (Label 1)
4 – MiddleLeft (RadioButton, CheckBox)
5 – MiddleCenter (Button)
6 – MiddleRight (LED)

テキストボックスコントロール

1 – Left
2 – Center
3 – Right

デフォルト: 1 – Left

参照

Button, Label, TextBox, RadioButton, Checkbox, LED, Text

使用例

```
GSet frmMain.lblName.TextAlign, TEXTALIGN_LEFT
```

Tick イベント

適用

Timer

説明

設定されたタイマー間隔に達したときに発生するイベントです。

使い方

*Form_Control_***Tick** (*Sender$* As String)

Sender$　　イベントを発生したコントロールの名前です。

参照

Timer, Interval, Enabled

使用例

```
Function frmMain_Timer1_Tick(Sender$ As String)

  GSet frmMain.lblDateTime.Text, Date$ + " " + Time$
Fend
```

TickFrequency プロパティー

適用

TrackBar

説明

目盛りマーク間の位置の数を設定・取得します。

使い方

GGet *Form.Control*.**TickFrequency**, *var*

GSet *Form.Control*.**TickFrequency**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

目盛りマーク間の位置の数を表す整数値です。

参照

TrackBar

使用例

```
GSet frmMain.TrackBar1.TickFrequency, 5
```

TickStyle プロパティー

適用

TrackBar

説明

目盛りをトラックバーのどこに表示するかを設定・取得します。

使い方

GGet *Form.Control*.**TickStyle**, *var*

GSet *Form.Control*.**TickStyle**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

0 – None

1 – TopLeft

2 – BottomRight

3 – Both

デフォルト: 2 - BottomRight

参照

TrackBar

使用例

```
GSet frmMain.TrackBar1.TickStyle, TICKSTYLE_BOTH
```

ToolTipText プロパティー

適用

Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar, Grid

説明

コントロールの上でマウスポインタを止めたときのポップアップに表示するテキストを設定・取得します。

使い方

GGet *Form.Control*.**ToolTipText**, *var*

GSet *Form.Control*.**ToolTipText**, *value*

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var プロパティーの値を取得する文字列変数を指定します。

value プロパティーに設定する文字列式を指定します。

値

コントロールのツールチップのテキストを含む文字列です。

デフォルト: 空白

参照

Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar

使用例

```
GSet frmMain.btnStart.ToolTipText, "Click Here to Start"
```

Top プロパティー

適用

Form, Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar, Grid

説明

フォームやコントロールの上側のピクセル座標を設定・取得します。

使い方

GGet *Form.*[*Control*].**Top**, *var*

GSet *Form.*[*Control*].**Top**, *value*

Form	フォームの名前、またはフォームの名前を含む文字列変数を指定します。
Control	コントロールの名前、またはコントロールの名前を含む文字列を指定します。
var	プロパティーの値を取得する整数変数を指定します。
value	プロパティーに設定する整数式を指定します。

値

上側の座標を表すピクセル整数値です。

参照

Form, Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar, Left, Height, Width

使用例

```
GSet frmMain.txtStatus.Top, 200
```

Type プロパティー

適用

All Controls

説明

コントロールのタイプ名を取得します。

使い方

GGet *Form.Control*.**Type**, *var*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する文字列変数を指定します。

値

コントロールのタイプ名を含む文字列。

参照

Controls

使用例

```
GGet frmMain.Controls(index).Type, typeName$
```

Update プロパティー

適用

TextBox

説明

テキストボックスにグローバル変数を表示するように設定されているとき、変数が書き換えられたときに自動的に表示を更新するかどうかを設定・取得します。

使い方

GGet *Form.Control*.**Update**, *var*

GSet *Form.Control*.**Update**, *value*

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var プロパティーの値を取得する論理変数を指定します。

value プロパティーに設定する論理を指定します。

値

0 – False

1 – True

デフォルト: 1 – True

注記

SPEL+のグローバル変数の値をテキストボックスに表示するには、テキストボックスの Variable プロパティーに表示したいグローバル変数名を設定します。Update プロパティを 0 – False に設定した場合、設定したグローバル変数の値が更新されません。

TextBox Variable プロパティーをグローバル変数に設定すると、デフォルトで、RC＋の実行時に TextBox Text 値を変数値に自動で更新します。Update プロパティーのデフォルト値は True です。ユーザーが同じ変数値を変更できるようにしたいときは、Update を False に設定する必要があります。これにより、自動更新は行われなくなり、ユーザーは新しい値を入力できるようになります。

Update が False から True に設定されているとき、RC +は変数値を、ユーザーがテキストボックスに入力した新しい値に設定します。メソッドの 1 つは、「値の変更」、または同様の名前のテキストボックスの上にチェックボックスを追加することです。チェックボックスのクリックイベントで、GGet を使用してチェックボックスのチェック状態を取得してから、GSset を使用して、TextBox Update プロパティーを設定してください。チェックボックスにチェックが入っている場合は、Update を False に設定してください。

参照

TextBox, Variable

使用例

```
GSet frmMain.txtStatus.Update, False
```

Value プロパティー

適用

ProgressBar, TrackBar

説明

コントロールの現在位置を表す値を設定・取得します。

使い方

GGet *Form.Control*.**Value**, *var*

GSet *Form.Control*.**Value**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。

var　プロパティーの値を取得する文字列変数を指定します。

value　プロパティーに設定する文字列式を指定します。

値

現在値を表す整数値です。

参照

ProgressBar, TrackBar, Maximum, Minimum, Variable

使用例

```
GSet form1.ProgressBar1.Value, 75
```

Variable プロパティー

適用

TextBox, ProgressBar, TrackBar

説明

テキストボックスに表示する SPEL+のグローバル変数の名前を設定・取得します。

使い方

GGet *Form.Control*.**Variable**, *var*

GSet *Form.Control*.**Variable**, *value*

Form	フォームの名前、またはフォームの名前を含む文字列変数を指定します。
Control	コントロールの名前、またはコントロールの名前を含む文字列を指定します。 そのコントロールは、指定されたフォーム上に存在しなくてはなりません。
var	プロパティーの値を取得する文字列変数を指定します。
value	プロパティーに設定する文字列式を指定します。

値

SPEL+プログラムで使うグローバル変数の名前です。

デフォルト: None

注記

Variable プロパティーに SPEL+のグローバル変数を設定すると、自動的に変数の値を Text プロパティーに設定、表示します。

TextBox Variable プロパティーをグローバル変数に設定すると、デフォルトでは、RC＋は実行時に TextBox Text 値を変数値に自動で更新します。Update プロパティーのデフォルト値は True です。ユーザーが同じ変数値を変更できるようにするには、Update を False に設定する必要があります。

Update が False から True に設定されているとき、RC +は変数値を、ユーザーがテキストボックスに入力した新しい値に設定します。

方法の 1 つは、「値の変更」、または同様の名前のテキストボックスの上にチェックボックスを追加することです。チェックボックスのクリックイベントで、GGet を使用してチェックボックスのチェック状態を取得してから、GSset を使用して、TextBox Update プロパティーを設定してください。チェックボックスにチェックが入っている場合は、Update を False に設定してください。

ProgressBar Variable プロパティーをグローバル変数に設定すると、プログレスバーに変数値が表示されます。TrackBar Variable プロパティーをグローバル変数に設定すると、変数値はトラックバーによって設定されます。

参照

TextBox, ProgressBar, TrackBar, Value

使用例

```
GSet frmMain.txtStatus.Variable, "g_Status$"
```

VideoEnabled プロパティー

適用

VideoBox

説明

ビデオボックスに画像を表示するかどうかを設定、取得します。

使い方

GGet *Form.Control*.**VideoEnabled**, *var*

GSet *Form.Control*.**VideoEnabled**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する論理変数を指定します。

value　　プロパティーに設定する論理式を指定します。

値

0 – False

1 – True

デフォルト: 0 – False

参照

VideoBox, GraphicsEnabled

使用例

```
GSet frmMain.VideoBox1.VideoEnabled, True
```

Visible プロパティー

適用

Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, StatusBar, TrackBar, Grid

説明

コントロールを実行時に表示するかどうかを設定・取得します。

使い方

GGet *Form.Control*.**Visible**, *var*

GSet *Form.Control*.**Visible**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　　プロパティーの値を取得する論理変数を指定します。

value　　プロパティーに設定する論理式を指定します。

値

1 – True　　コントロールを表示します。

0 – False　　コントロールを表示しません。

デフォルト: 1 – True

注記

GShow、あるいは GShowDialog でフォームをロードしていない状態で GGet を実行した場合、値は False が戻ります。

参照

Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, StatusBar, TrackBar, Enabled

使用例

```
GSet frmMain.txtStatus.Visible, True
```

Width プロパティー

適用

Form, Button, Label, TextBox, RadioButton, CheckBox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar, Grid

説明

フォームやコントロールの幅をピクセルで設定、参照します。

使い方

GGet *Form*.[*Control*].**Width**, *var*

GSet *Form*.[*Control*].**Width**, *value*

Form　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control　コントロールの名前、またはコントロールの名前を含む文字列を指定します。

var　プロパティーの値を取得する整数変数を指定します。

value　プロパティーに設定する整数式を指定します。

値

フォームやコントロールの幅のピクセル整数値です。

参照

Form, Button, Label, TextBox, RadioButton, Checkbox, ListBox, ComboBox, PictureBox, GroupBox, VideoBox, LED, ProgressBar, TrackBar, StatusBar, Left, Top, Height

使用例

```
GSet frmMain.txtStatus.Width, 300
```

WindowState プロパティー

適用

Form

説明

フォームのウィンドウの状態を設定・取得します。

使い方

GGet *Form*.**WindowState**, *var*

GSet *Form*.**WindowState**, *value*

Form　　フォームの名前、またはフォームの名前を含む文字列変数を指定します。

var　　プロパティーの値を取得する整数変数を指定します。

value　　プロパティーに設定する整数式を指定します。

値

0 – Normal

1 – Minimized

2 – Maximized

デフォルト: 0 – Normal

参照

Form

使用例

```
GSet frmMain.WindowState, WINDOWSTATE_MAXIMIZED
```

WordWrap プロパティー

適用

TextBox

説明

複数行のテキストボックスで、文字単位で改行するかどうかを設定、参照します。

使い方

GGet *Form.Control*.**WordWrap**, *var*

GSet *Form.Control*.**WordWrap**, *value*

Form フォームの名前、またはフォームの名前を含む文字列変数を指定します。

Control コントロールの名前、またはコントロールの名前を含む文字列を指定します。
そのコントロールは、指定されたフォーム上に存在しなくてはなりません。.

var プロパティーの値を取得する論理変数を指定します。

value プロパティーに設定する論理式を指定します。

値

1 – True : 複数行テキストボックスのときに、その行に収まらない単語を次の行に送ります。

0 – False : ユーザーの入力がテキストボックスの右端まできたときに改行します。

デフォルト: 1 – True

参照

TextBox, Multiline, ScrollBars

使用例

```
GSet frmMain.txtStatus.WordWrap, True
```