



Full Automation Computing Technology and Integrated Computer System



## アプリケーション設計ガイド



International Laboratory Corporation



## 製品ならびにパッケージについてのご注意

1. 本製品は日本著作権法により保護されています。ソフトウェアおよびマニュアルの一部または全部を無断で使用、複製することはできません。
2. このソフトウェアおよびマニュアルを運用した結果の影響については、一切責任を負いかねますので、ご了承ください。
3. このソフトウェアの仕様、およびマニュアルに記載されている事柄は、将来予告なしに変更することがあります。

GENWARE、GENIFA は、株式会社アイ・エル・シーの登録商標です。

Microsoft、Windows、Windows Vista、Microsoft Visual C++は 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、記載されている会社名、製品名は、各社の商標および登録商標です。


## 改訂履歴

印刷日付	マニュアル番号	改訂内容
2014 年 1 月	GS186-A	初版作成

## ● 安全上のご注意 ●

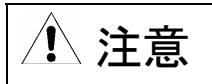
(ご使用前に必ずお読みください)

GENWARE3 および C 言語コントローラのご使用に際しては、各製品に付属しているマニュアルおよび付属マニュアルで紹介している関連マニュアルをよくお読みいただくと共に、安全に対して十分に注意を払って、正しい取扱いをしていただくようお願いいたします。


この「安全上のご注意」では、安全注意事項のランクを「 警告」、「 注意」として区分してあります。



取扱いを誤った場合に、危険な状況が起こりえて、死亡または重傷を受ける可能性が想定される場合。



取扱いを誤った場合に、危険な状況が起こりえて、中程度の傷害や軽傷を受ける可能性が想定される場合および物的損害だけの発生が想定される場合。

なお、 注意に記載した事項でも、状況によっては重大な結果に結びつく可能性があります。いずれも重要な内容を記載していますので必ず守ってください。

本マニュアルは必要なときに読めるよう大切に保管すると共に、必ず最終ユーザまでお届けいただくようお願いいたします。

### 【設計上の注意事項】



- 外部電源の異常や C 言語コントローラユニット本体の故障時でも、システム全体が安全側に働くように C 言語コントローラユニットの外部で安全回路を設けてください。誤出力、誤動作により、事故のおそれがあります。
    - (1) 非常停止回路、保護回路、正転／逆転などの相反する動作のインタロック回路、位置決めの上限／下限など機械の破損防止のインタロック回路などは、C 言語コントローラユニットの外部で回路構成してください。
    - (2) C 言語コントローラユニットは、次の異常状態を検出すると、(a) の場合はユーザプログラムからの出力 (Y) およびバッファメモリへの書込みを禁止し、全出力を OFF にします。(b) の場合はユーザプログラムからの出力 (Y) およびバッファメモリへの書込みを禁止します。  
出力を全点保持するか全点 OFF するかはパラメータで設定します。
      - (a) 電源ユニットの過電流保護装置または過電圧保護装置が働いたとき。
      - (b) C 言語コントローラユニットでウォッチドッグタイマエラーなど自己診断機能により異常を検出したとき。
- また、C 言語コントローラユニットで検出できない入出力制御部分などの異常時は、全出力が ON することがあります。このとき、機械の動作が安全側に働くよう、C 言語コントローラユニットの外部でフェールセーフ回路を構成したり、安全機構を設けてください。



## 【設計上の注意事項】



(3) 出力回路のリレーやトランジスタなどの故障によっては、出力が ON の状態を保持したり、OFF の状態を保持することがあります。重大な事故につながるような出力信号については、外部で監視する回路を設けてください。

- 出力回路において、定格以上の負荷電流または負荷短絡などによる過電流が長時間継続して流れた場合、発煙・発火のおそれがありますので、外部にヒューズなどの安全回路を設けてください。
- C 言語コントローラユニット本体の電源立上げ後に、外部供給電源を投入するように回路を構成してください。外部供給電源を先に立ち上げると、誤出力、誤動作により事故のおそれがあります。
- データリンクが交信異常になったときの各局の動作状態については、各データリンクのマニュアルを参照してください。誤出力、誤動作により事故のおそれがあります。
- CPU ユニットに周辺機器を接続、またはインテリジェント機能ユニットにパソコンなどを接続して、運転中の C 言語コントローラユニットに対する制御（データ変更）を行うときは、常時システム全体が安全側に働くように、ユーザプログラム上でインタロック回路を構成してください。また、運転中の C 言語コントローラユニットに対するその他の制御（プログラム変更、運転状態変更（状態制御））を行うときは、マニュアルを熟読し、十分に安全を確認してから行ってください。特に外部機器から遠隔地の C 言語コントローラユニットに対する上記制御では、データ交信異常により C 言語コントローラユニット側のトラブルに即対応できない場合もあります。ユーザプログラム上でインタロック回路を構成すると共に、データ交信異常が発生時のシステムとしての処置方法などを外部機器と CPU ユニット間で取り決めてください。
- インテリジェント機能ユニットのバッファメモリの中で、書込み禁止のエリアにはデータを書き込まないでください。また、CPU ユニットからインテリジェント機能ユニットに対する出力信号の中で、使用禁止の信号を出力(ON)しないでください。書込み禁止のエリアに対するデータの書込み、使用禁止の信号に対する出力を行うと、C 言語コントローラシステムが誤動作する危険性があります。
- 書込み禁止のエリア、使用禁止の信号については、各インテリジェント機能ユニットのユーザーズマニュアルを参照してください。通信ケーブルが断線した場合は、回線が不安定になり、複数の局でネットワークが交信異常になる場合があります。複数の局でネットワークの交信異常が発生しても、システムが安全側に働くようにユーザプログラム上でインタロック回路を構成してください。誤出力、誤動作により、事故のおそれがあります。
- ネットワーク経由の外部機器からの不正アクセスに対して、C 言語コントローラシステムの安全を保つ必要があるときは、ユーザによる対策を盛り込んでください。また、インターネット経由の外部機器からの不正アクセスに対して、C 言語コントローラシステムの安全を保つ必要があるときは、ファイアウォールなどの対策を盛り込んでください。
- リフレッシュパラメータの設定で、リンク出力 (LY) リフレッシュデバイスおよびリモート出力 (RY) リフレッシュデバイスに“Y”を指定できません。そのため、CPUSTOP 時に、STOP する前のデバイスがそのまま保持されます。

## 【設計上の注意事項】

### 注意

- 制御線や通信ケーブルは、主回路や動力線などと束線したり、近接したりしないでください。100mm 以上を目安として離してください。ノイズにより、誤動作の原因になります。
- 出力ユニットでランプ負荷、ヒータ、ソレノイドバルブなどを制御するとき、出力の OFF→ON 時に大きな電流（通常の 10 倍程度）が流れる場合がありますので、定格電流に余裕のある出力ユニットの選定を行ってください。
- CPU ユニットの電源 OFF→ON またはリセット時、CPU ユニットが RUN 状態になるまでの時間が、システム構成、パラメータ設定、プログラム容量などにより変動します。RUN 状態になるまでの時間が変動しても、システム全体が安全側に働くように設計してください。
- 各種設定を登録中、ユニット装着局の電源 OFF および CPU ユニットのリセット操作を行わないでください。登録中にユニット装着局の電源 OFF および CPU ユニットのリセット操作を行うと、内蔵 ROM の設定データが不定となり、再設定・再登録が必要となります。また、ユニット故障および誤動作の原因となります。
- CPU ユニットまたはリモート I/O ユニットのパラメータを変更したときは、必ず CPU ユニットのリセットしてください。ユニットに変更前のデータが残ることにより、誤動作の原因になります。

## 【取付けの注意事項】



- C 言語コントローラユニットは本マニュアル記載の一般仕様の環境で使用してください。一般仕様の範囲以外の環境で使用すると、感電、火災、誤動作、製品の損傷あるいは劣化の原因になります。
- ユニット下部のユニット装着用レバーを押さえながら、ユニット固定用突起をベースユニットの固定穴に確実に挿入し、ユニット固定穴を支点として装着してください。ユニットが正しく装着されていないと、誤動作、故障、落下の原因になります。振動の多い環境で使用する場合は、ユニットをネジで締め付けてください。また、ユニット固定金具付きのユニットは、ベースユニットへ装着したあと、ユニット固定金具で固定してください。
- ネジの締め付けは、規定トルク範囲で行ってください。ネジの締め付けが緩いと、落下、短絡、誤動作の原因になります。ネジを締め過ぎると、ネジやユニットの破損による落下、短絡、誤動作の原因になります。
- 増設ケーブルは、ベースユニットの増設ケーブル用コネクタに確実に装着してください。装着後に、浮上りがないかチェックしてください。接触不良により、誤入力、誤出力の原因になります。
- SD メモリカードは、装着スロットに確実に装着してください。装着後に浮上りがないかチェックしてください。接触不良により、誤動作の原因になります。
- コンパクトフラッシュカードは、コンパクトフラッシュカード装着スロットに押し付けて確実に装着してください。コンパクトフラッシュカード装着後に、確実に装着されているかチェックしてください。接触不良により、誤動作の原因になります。
- ユニットの着脱は、必ずシステムで使用している外部供給電源を全相遮断してから行ってください。全相遮断しないと製品の損傷のおそれがあります。オンラインユニット交換に対応した CPU ユニットを使用したシステムおよび MELSECNET/H リモート I/O 局は、オンライン中(通電中)でのユニット交換が可能です。ただし、オンライン中(通電中)でのユニット交換が可能なユニットには制限があり、ユニットごとに交換手順が決められています。詳細については、QCPU ユーザーズマニュアル(ハードウェア設計・保守点検編)およびオンラインユニット交換に対応したユニットのマニュアルに記載されているオンラインユニット交換の項を参照ください。
- ユニットの導電部分には直接触らないでください。ユニットの誤動作、故障の原因になります。
- モーション CPU ユニット、モーションユニットを使用するときは、電源を投入する前にユニットの組合せが正しいか必ず確認してください。誤った組合せで使用した場合、製品が損傷するおそれがあります。詳細については、使用するユニットのユーザーズマニュアルを参照してください。

## 【配線上の注意事項】

### 警告

- 取付け、配線作業などは、必ずシステムで使用している外部供給電源を全相遮断してから行ってください。全相遮断しないと、感電あるいは製品の損傷のおそれがあります。
- 配線作業後、通电、運転を行う場合は、必ず製品に付属の端子カバーを取り付けてください。端子カバーを取り付けしないと、感電のおそれがあります。

### 注意

- FG 端子および LG 端子は、C 言語コントローラユニット専用の D 種接地(第三種接地)以上で必ず接地を行ってください。感電、誤動作のおそれがあります。
- 圧着端子は適合圧着端子を使用し、規定のトルクで締め付けてください。先開形圧着端子を使用すると、端子ネジが緩んだ場合に脱落し、故障の原因になります。
- ユニットへの配線は、製品の定格電圧および端子配列を確認した上で正しく行ってください。定格と異なった電圧の入力や、電源を接続、誤配線をすると、火災、故障の原因になります。
- 外部接続用コネクタ、同軸ケーブル用コネクタは、メーカー指定の工具で圧着、圧接または正しくハンダ付けしてください。接続が不完全になっていると、短絡、火災、誤動作の原因になります。
- 制御線や通信ケーブルは、主回路や動力線などと束線したり、近接したりしないでください。100mm 以上を目安として離してください。ノイズにより、誤動作の原因になります。
- ユニットに接続する電線やケーブルは、必ずダクトに納めるか、またはクランプによる固定処理を行ってください。ケーブルをダクトに納めなかったり、クランプによる固定処理をしていないと、ケーブルのふらつきや移動、不注意の引っ張りなどによるユニットやケーブルの破損、ケーブルの接続不良による誤動作の原因となります。
- 端子ネジの締め付けは、規定トルク範囲で行ってください。端子ネジの締め付けが緩いと、短絡、火災、誤動作の原因になります。端子ネジを締め過ぎると、ネジやユニットの破損による落下、短絡、誤動作の原因になります。
- コネクタ取付けネジの締め付けは、規定トルク範囲で行ってください。ネジの締め付けが緩いと、短絡、火災、誤動作の原因になります。ネジを締め過ぎると、ネジやユニットの破損による落下、短絡、火災、誤動作の原因になります。
- コネクタは確実にユニットに取り付けてください。取付けが不確実だと誤動作の原因になります。
- ユニットに接続されたケーブルを取り外すときは、ケーブル部分を手に持って引っ張らないでください。コネクタ付きのケーブルは、ユニットの接続部分のコネクタを手で持って取り外してください。端子台接続のケーブルは、端子台端子ネジを緩めてから取り外してください。ユニットに接続された状態でケーブルを引っ張ると、誤動作またはユニットやケーブルの破損の原因となります。

## 【配線上の注意事項】

### 注意

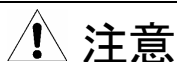
- ケーブル接続は、接続するインタフェースの種類を確認の上、正しく行ってください。異なったインタフェースに接続または誤配線すると、ユニット、外部機器の故障の原因となります。
- ユニット内に、切粉や配線クズなどの異物が入らないように注意してください。火災、故障、誤動作の原因になります。
- ユニットは、配線時にユニット内へ配線クズなどの異物が混入するのを防止するため、ユニット上部に混入防止ラベルを貼り付けています。配線作業中は、本ラベルをはがさないでください。システム運転時は、放熱のために本ラベルを必ずはがしてください。
- C 言語コントローラユニットは、制御盤内に設置して使用してください。制御盤内に設置されたシーケンサ電源ユニットへの主電源配線に関しては、中継端子台を介して行ってください。また、電源ユニットの交換と配線作業は、感電保護に対して、十分に教育を受けたメンテナンス作業者が行ってください。配線方法は、QCPU ユーザーズマニュアル(ハードウェア設計・保守点検編)を参照してください。

## 【立上げ・保守時の注意事項】

### 警告

- 通電中に端子に触れないでください。感電または誤動作の原因になります。
- バッテリコネクタは正しく接続してください。バッテリーに充電、分解、加熱、火中投入、ショート、ハンダ付けなどを行わないでください。バッテリーの取扱いを誤ると、発熱、破裂、発火などにより、ケガ、火災のおそれがあります。
- 清掃、端子ネジ、コネクタ取付けネジ、ユニット固定ネジの増し締めは、必ずシステムで使用している外部供給電源を全相遮断してから行ってください。全相遮断しないと、感電、ユニットの故障や誤動作のおそれがあります。ネジの締め付けが緩いと、落下、短絡、誤動作の原因になります。ネジを締め過ぎると、ネジやユニットの破損による落下、短絡、誤動作の原因になります。

## 【立上げ・保守時の注意事項】



- CPUユニットに周辺機器を接続、またはインテリジェント機能ユニットにパソコンなどを接続して、運転中のC言語コントローラユニットに対する制御(データ変更)を行うときは、常時システム全体が安全側に働くように、シーケンスプログラム上でインタロック回路を構成してください。また、運転中のC言語コントローラユニットに対するその他の制御(プログラム変更、パラメータ変更、強制出力、運転状態変更(状態制御))を行うときは、マニュアルを熟読し、十分に安全を確認してから行ってください。特に外部機器から遠隔地のC言語コントローラユニットに対する上記制御では、データ交信異常によりC言語コントローラユニット側のトラブルに即対応できない場合もあります。シーケンスプログラム上でインタロック回路を構成するとともに、データ交信異常が発生時のシステムとしての処置方法などを、外部機器とCPUユニット間で取り決めてください。
- ユニットの分解、改造はしないでください。故障、誤動作、ケガ、火災の原因になります。
- 携帯電話やPHSなどの無線通信機器は、C言語コントローラユニットの全方向から25cm以上離して使用するようにしてください。誤動作の原因になります。
- ユニットの着脱は、必ずシステムで使用している外部供給電源を全相遮断してから行ってください。全相遮断しないと、ユニットの故障や誤動作の原因になります。オンラインユニット交換に対応したCPUユニットを使用したシステムおよびMELSECNET/HリモートI/O局は、オンライン中(通電中)でのユニット交換が可能です。ただし、オンライン中(通電中)でのユニット交換が可能なユニットには制限があり、ユニットごとに交換手順が決まっています。詳細については、QCPU ユーザーズマニュアル(ハードウェア設計・保守点検編)およびオンラインユニット交換に対応したユニットのマニュアルのオンラインユニット交換の項を参照ください。
- ユニットとベースユニットおよび端子台の着脱は、製品ご使用後、50回以内としてください。(JISB3502に準拠)なお、50回を超えた場合は、誤動作の原因となるおそれがあります。
- ユニットに装着するバッテリーには、落下・衝撃を加えないでください。落下・衝撃によりバッテリーが破損し、バッテリー液の液漏れがバッテリー内部で発生するおそれがあります。落下・衝撃を加えたバッテリーは使用せずに廃棄してください。
- 制御盤内での立上げ・保守作業は、感電保護に対して、十分に教育を受けたメンテナンス作業者が行ってください。また、メンテナンス作業者以外が制御盤を操作できないよう、制御盤に鍵をかけるようにしてください。
- ユニットに触れる前には、必ず接地された金属などに触れて、人体などに帯電している静電気を放電してください。静電気を放電しないと、ユニットの故障や誤動作の原因になります。
- SDメモ리카ードの取付け・取外しは、製品使用後、500回以内としてください。500回を超えた場合は、誤動作の原因となるおそれがあります。

### 【廃棄時の注意事項】



- 製品を廃棄するときは、産業廃棄物として扱ってください。  
バッテリーを廃棄する際には各地域にて定められている法令に従い分別を行ってください。  
(EU 加盟国内でのバッテリー規制についての詳細は QCPU ユーザーズマニュアル(ハードウェア設計・保守点検編)を参照してください。)

### 【輸送時の注意事項】



- リチウムを含有しているバッテリーの輸送時には、輸送規制に従った取扱いが必要となります。(規制対象機種についての詳細は QCPU ユーザーズマニュアル(ハードウェア設計・保守点検編)を参照してください。)

## 表記について

製品を使用する上で必要な以下の項目について記述しています。

### MEMO

操作、解説、設定についての補足情報を記述しています。

#### **MEMO**

- 

### IMPORTANT

製品を使用する上で、使用者が安全を確保するために取扱いに関する注意が必要な項目について記述しています。

操作、解説、設定についての大切な情報を記述しています。

#### **IMPORTANT**

- 

## 用語について

本書では、「GENWARE3 for Q24DHCCPU-VG」を「GENWARE3」と記載します。

「GENIFA3 for Q24DHCCPU-VG」を「GENIFA3」と記載します。

また、「C 言語コントローラ」と記載した場合、以下を指します。

- Q24DHCCPU-VG



## マニュアルの種類と構成

GENWARE3 に関連するマニュアルは、下記のとおりです。

名称	マニュアル番号	内容
GENWARE3 インストールガイド	GS183	GENWARE3 のセットアップ方法
GENWARE3 ユーザーズマニュアル	GS184	GUI デザインツール「CI SKETCH」による画面データ作成方法、および CI SKETCH の操作全般 GUI ライブラリ「GENIFA3」を用いたプログラム開発手順と、GENIFA3 の 仕様
GENIFA3 関数リファレンス	GS185	GUI ライブラリ「GENIFA3」の関数リファレンス
GENWARE3 アプリケーション設計ガイド (本書)	GS186	GENWARE3 を用いてアプリケーションを開発する際の一連の手順説明

インストールガイド、ユーザーズマニュアル、関数リファレンスは、GENWARE3 インストール CD 内に、PDF データとして格納されています。

## 目次

### 第 1 章 はじめに

1-1 本書について .....	1 - 1
1-2 テンプレート画面 .....	1 - 2
1-3 開発環境 .....	1 - 20
1-4 開発の流れ .....	1 - 20

### 第 2 章 CI SKETCHによる画面作成

2-1 プロジェクトの作成 .....	2-1
2-2 リソースの登録 .....	2-5
2-3 パネル/ウィンドウの作成と設定 .....	2-22
2-4 コントロールの作成と設定 .....	2-30
2-5 スクリーンプロパティの設定 .....	2-33
2-6「手動操作画面」の作成 .....	2-34
2-7「モニタ画面」の作成 .....	2-40
2-8「エラー管理画面」の作成 .....	2-46
2-9「運転データ画面」の作成 .....	2-52
2-10「マニュアル管理画面」の作成 .....	2-59
2-11「保守管理画面」の作成 .....	2-62
2-12「操作履歴画面」の作成 .....	2-67
2-13「パラメータ設定画面」の作成 .....	2-70
2-14「メニュー画面」の作成 .....	2-75
2-15「メッセージ画面」の作成 .....	2-77
2-16「テンキー画面」の作成 .....	2-78

## 第 3 章 ソースコード生成

3-1 ソースコード生成 .....	3-1
3-2 生成ファイル .....	3-6

## 第 4 章 ユーザプログラムの追加

4-1 追記ファイル一覧 .....	4-1
4-2 ユーザプログラムの追加 .....	4-2

## 第 5 章 アプリケーションの実行

5-1 C 言語コントローラへの転送 .....	5-1
5-2 アプリケーションの起動 .....	5-2

## 付録 1 プロパティ設定一覧

付 1-1 リソース設定 .....	付 1-1
付 1-2 プロジェクトプロパティ設定 .....	付 1-23
付 1-3 スクリーンプロパティ設定 .....	付 1-23
付 1-4 パネル/ウィンドウプロパティ設定 .....	付 1-24
付 1-5 コントロールプロパティ設定 .....	付 1-28

## 第 1 章 はじめに

1-1 本書について.....	1-1
1-2 テンプレート画面 .....	1-2
1-3 開発環境.....	1-20
1-4 開発の流れ .....	1-20

## 1-1 本書について

「GENWARE3 アプリケーション設計ガイド(以下、本書と称します)」では、GENWARE3 を使ってはじめて C 言語コントローラ用 GUI アプリケーションを開発される方への道しるべとして、サンプルアプリケーション(テンプレート画面)を作り上げるまでの一連の開発手順を通して、基本的な作成方法について説明します。

本書は、GENWARE3 を初めてお使いになる方を対象としていますが、「GENWARE3 ユーザーズマニュアル」には記載されていないユーザープログラムの具体的なプログラミング方法も数多く説明しているため、既にお使いの方にも役立つ内容となっています。

### IMPORTANT

- ◆ 本書で紹介する「テンプレート画面」は、お客様が GUI アプリケーションを作成する際の参考としていただくためのサンプルとして提供していますが、お客様のシステムで、そのままご使用いただくことを想定したアプリケーションではありません。お客様のシステムで、テンプレート画面をそのまま使用しないでください。

### MEMO

- ◆ CI SKETCH3 を使用するにあたって、標準的な Windows デスクトップアプリケーションを操作できる必要があります。またユーザプログラム追記の際には C/C++ 言語、および C 言語コントローラ用エンジニアリングツール CW Workbench の基礎知識が必要です。
- ◆ CI SKETCH3 の操作方法についての詳細説明は「GENWARE3 ユーザーズマニュアル」を、GENIFA3 関数についての詳細説明は「GENIFA3 関数リファレンス」を参照ください。

## 1-2 テンプレート画面

今回、サンプルアプリケーションとして、装置の操作画面をイメージした画面を作成します。

作成する画面は、次の 10 画面です。

画面名称	概要
手動操作画面	ボタンを中心とした操作画面です。ランプ、折れ線グラフも表示します。
モニター画面	装置モニター画面です。コンベア上のワークの移動を表示します。
エラー管理画面	アラーム履歴を一覧表示します。
運転データ管理画面	運転データを表示する画面です。データを数値、棒グラフで表します。
マニュアル管理画面	マニュアルニュアル(画像データ)の表示を行います。
保守管理画面	ユーザセキュリティのログイン画面です。
操作履歴画面	操作履歴の一覧を表示します。
パラメータ設定画面	簡易なレシピ設定を行います。
メニュー画面	画面切換、銘板切換(日・英)を行います。
テンキー画面	テンキー画面です。

これらの画面の中で、一般的な表示器で使用される以下の機能、動作を作成しています。

- ・シーケンサのデバイスに対する読書き
- ・ボタン
- ・ランプ
- ・数値表示、数値入力
- ・グラフ表示
- ・アラーム履歴(履歴の一覧表示、ファイル保存)
- ・操作ログ(履歴の一覧表示、ファイル保存)
- ・データログ(グラフ表示、)
- ・ユーザセキュリティ、パスワード入力
- ・レシピ切換
- ・アニメーション
- ・画面切換、ウィンドウ表示
- ・銘板切換

次に各画面の動作を説明します。

なお、画面内の表示要素(ボタン、ランプ、数値表示など)のプロパティ設定や、シーケンサのデバイスの割り付けについては、「付録」を参照してください。

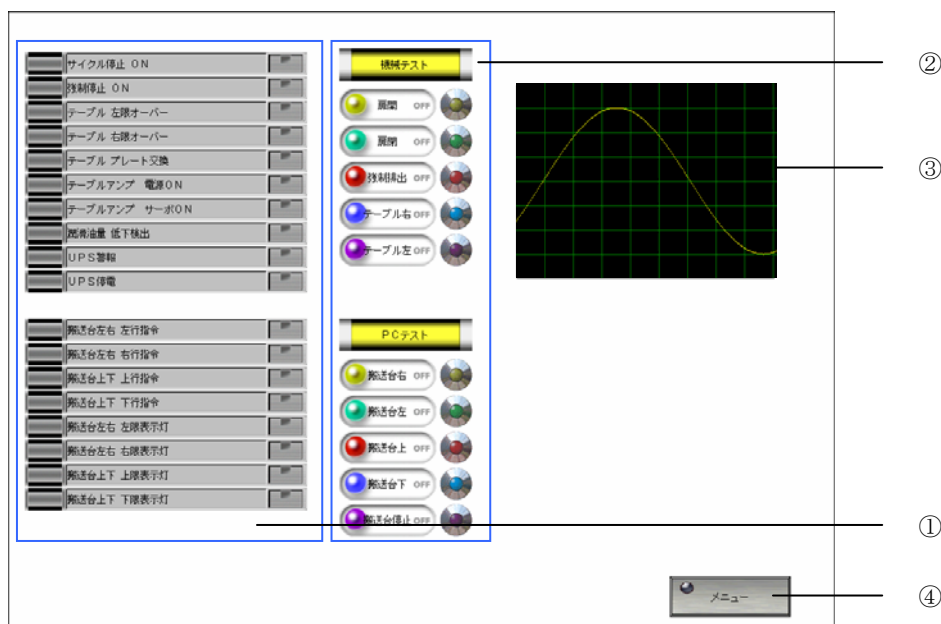
## 手動操作画面

本画面では、シーケンサのビットデバイスに対し、スイッチ、ランプによる読み書きを行っています。

ワードデバイスの値をモニタし、グラフ表示します。

また、各画面共通の動作として、以下も確認できます。

- ・ウィンドウの表示(メニュー画面の表示)
- ・ユーザセキュリティのレベルによるスイッチの操作可・不可(詳細は後述)
- ・銘板切換(表示言語を日・英に切換。変更はメニュー画面から実施)



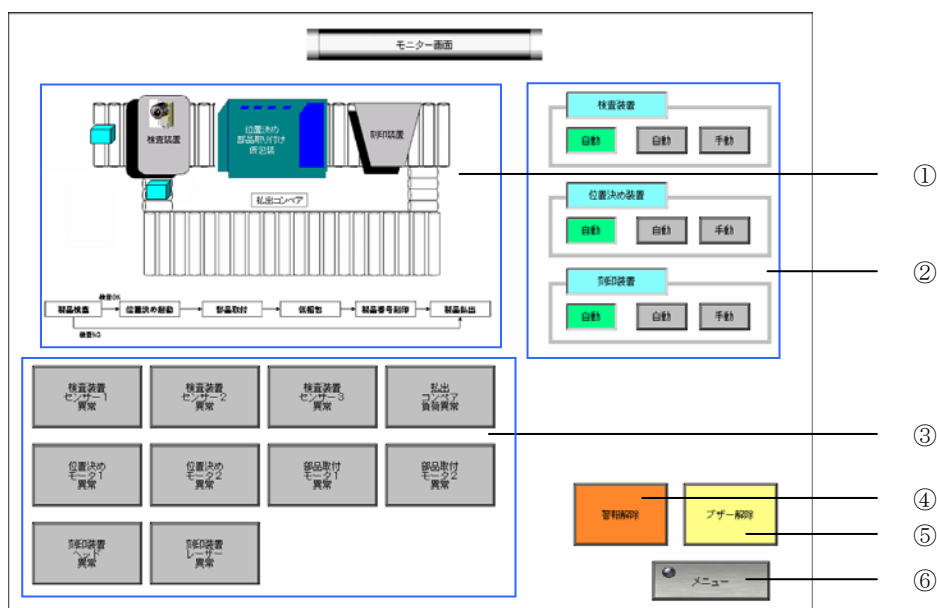
番号	項目	動作	レベル(※)	使用コントロール
①	ランプ	M0 ～ M9 、 M16 ～ M23 の ON/OFF に従い、ランプを ON/OFF します。	—	ピクチャ、 スタティックテキスト
②	スイッチ、ランプ	画面上のスイッチを押すことで、Y40～Y44、Y50～Y54 の各端子へ出力します。 また、X20～X24、X30～X34 への入力状態に従い、ランプを ON/OFF します。	1	ボタン、ピクチャ
③	グラフ表示	折れ線グラフを表示します。 ここではグラフ線の引き方の説明のために、アプリケーション内に自動的に sin カーブを描く処理を作成します。	—	基本コントロール
④	メニューボタン	ボタンを押すと、メニュー画面をウィンドウ表示します。	0	ボタン

※: ユーザセキュリティ機能による操作可能なレベルを示しています。ユーザセキュリティ機能の詳細については、後述の「保守管理画」を参照してください。



## モニター画面

ここでは、コンベア上のワークの流れをモニタする画面を示しています。  
ランプ、スイッチに関しては、前述の「手動操作画面」と同様です。



番号	項目	動作	レベル	使用コントロール
①	コンベア	コンベア上のワークの流れを示します。 ワークが通過した際に、表示色を水色から桃色に変更します。	—	ピクチャ
②	「自動」/「手動」	各装置の現在の設定を、左端のランプで示します。 設定変更を行う場合に、中央、右端のボタンを押します。	2	ボタン、ピクチャ、矩形
③	異常ランプ	それぞれの異常に割り当てられたビットデバイスの ON/OFF に連動して、ランプの表示を ON/OFF します。	—	ボタン
④	警報解除	警報解除指示を行います。ここでは本ボタンに割り付けたビットデバイスを ON します。	1	ボタン

⑤	ブザー解除	ブザー解除指示を行います。ここでは本ボタンに割り付けたビットデバイスを ON します。	0	ボタン
⑥	メニューボタン	ボタンを押すと、メニュー画面をウィンドウ表示します。	0	ボタン

## エラー管理画面

本画面では、アラームの発生・解除の履歴を一覧表示しています。



番号	項目	動作	レベル	使用コントロール
①	エラー履歴	エラーに割り当てられたビットの ON(エラー発生)/OFF(解除)に連動して、エラー一覧を表示します。 エラーレベル、発生日時、エラーコード、異常内容、復旧日時を表示します。 発生中のエラーは赤、解除済みのエラーは青で表示します	0	スタティックテキスト

## GENWARE3 アプリケーション設計ガイド

②	ソート	以下の順でエラー履歴の一覧の並び替えができます。 ・発生日時が古いものから新しい順 ・発生日時が新しいものから古い順 ・レベルの高いものから低い順 ・レベルの低いものから高い順	0	ボタン
③	現在発生中エラー	現在発生中のエラー情報を表示します。 複数エラーが発生している場合は、その中で最新のエラーを 1 つ表示します。 そのエラーが復旧した場合は、その次に新しいエラーを表示します。	—	スタティックテキスト
④	履歴クリア	エラー履歴をクリアします。	1	ボタン
⑤	メニューボタン	ボタンを押すと、メニュー画面をウィンドウ表示します。	0	ボタン

## アラーム管理について

テンプレート画面では、アラームとして登録したビットデバイスの ON/OFF(アラームの発生／解除)を、1000msec 周期で監視しています。

エラー履歴の情報は、C 言語コントローラ内にファイル保存され、本画面のエラー履歴の一覧では、そのファイルの情報を読み出して表示しています。

また、エラー履歴はリングバッファで保存し、最大 100 件まで保存します。

アラーム登録は CSV ファイルで行います(最大 50 件)。

CSV ファイルには、先頭行に変数の ID を設定し、以降の行がアラーム設定となり、

変数インデックス,エラーレベル,エラーコード,文字列リソース ID

を登録個数分、記載します。

変数インデック	アラームに対応した変数の配列番号を指定します。
エラーレベル	エラーレベルには 1～99 の数値を設定します。レベル 1 を一番高いレベルとします。
エラーコード	4 桁の 16 進数値を設定します。
文字列リソース ID	「異常内容」に表示する文字列は、CI SKETCH で文字列リソースとして登録します。その文字列リソース ID を指定します。

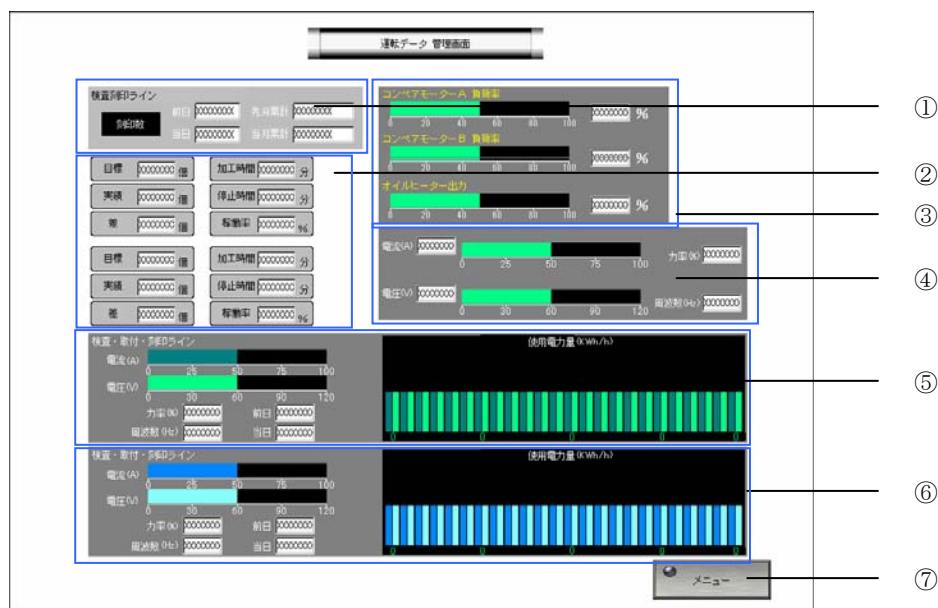
テンプレート画面のアプリケーションが起動するときに、CSV ファイルを読み込みます。

### MEMO

- ◆ アラーム登録件数や履歴数、周期は、ユーザがプログラムを修正することで、変更可能です。
- ◆ テンプレート画面では、エラー履歴をC言語コントローラ内に保存していますが、ユーザがプログラムを修正することで、C言語コントローラに装着したSDカード・USBメモリなど、C言語コントローラの外部の記憶媒体に保存することも可能です。

## 運転データ画面

本画面では、ラインの生産状況などの表示画面イメージを示しています。  
ここでは、ワードデバイスの現在値を、数値および棒グラフにより示しています。  
また、ワードデバイスのロギングも行っています。



番号	項目	動作	レベル	使用コントロール
①	刻印数表示エリア	ワードデバイスの現在値を表示します。	—	ピクチャ、スタティックテキスト、テキストボックス
②	目標・実績値表示エリア	ワードデバイスの現在値を表示します。 表示単位として「%」となっている項目については、現在値×0.1の値を表示します。	—	ピクチャ、スタティックテキスト、テキストボックス
③	コンペアモーター負荷率、オイルヒーター出力表示エリア	ワードデバイスの現在値を数値、および棒グラフで表示します。	—	ピクチャ、スタティックテキスト、テキストボックス、プログレスバー
④	電流・電圧表示エリア	ワードデバイスの現在値を数値、および棒グラフで表示します。	—	ピクチャ、スタティックテキスト、テキストボックス、プログレスバー
⑤	検査・取付・刻印ライン1表示エリア	ワードデバイスの現在値を数値、および棒グラフで表示します。 使用電力量は、1時間ごとにワードデバイスの値をロギングし、そのデータを棒グラフとして表示します。	—	ピクチャ、スタティックテキスト、テキストボックス、プログレスバー
⑥	検査・取付・刻印ライン2表示エリア	ワードデバイスの現在値を数値、および棒グラフで表示します。 使用電力量は、1時間ごとにワードデバイスの値をロギングし、そのデータを棒グラフとして表示します。	—	テキスト、テキストボックス、プログレスバー
⑦	メニューボタン	ボタンを押すと、メニュー画面をウィンドウ表示します。	0	ボタン

### ワードデータのロギングについて

この画面では、ワードデバイスのロギングを、1時間周期で行っています。

ロギングデータは、C言語コントローラ内にファイル保存しています。

本画面の画面右下の「電力使用量」を示す棒グラフは、そのファイルの情報を読み出して表示しています。

また、ロギングデータはリングバッファで保存します。デバイス1点につき、最大100件まで保存します。

**GENWARE3 アプリケーション設計ガイド**

ロギングするデバイスの登録は CSV ファイルで行います (最大 16 件)。

CSV ファイルには、

変数名

を登録個数分、記載します。

変数名	ロギングするデバイスに対応した変数を指定します。
-----	--------------------------

テンプレート画面のアプリケーションが起動するときに、CSV ファイルを読み込みます。

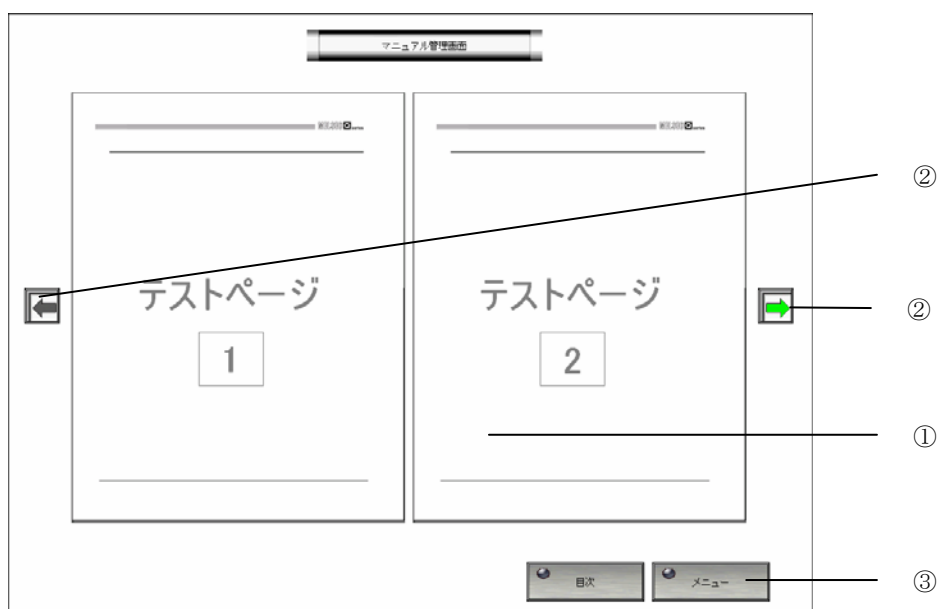
**MEMO**

- ◆ ロギングの登録件数や履歴数、周期は、ユーザがプログラムを修正することで、変更可能です。
- ◆ テンプレート画面では、ログデータを C 言語コントローラ内に保存していますが、ユーザがプログラムを修正することで、C 言語コントローラに装着した SD カード・USB メモリなど、C 言語コントローラの外部の記憶媒体に保存することも可能です。

## マニュアル管理画面

本画面では、マニュアルの表示を行います。

マニュアルの各ページを画像ファイル(bmp ファイル)として 10 ページ分作成し、C 言語コントローラ内に保存しています。



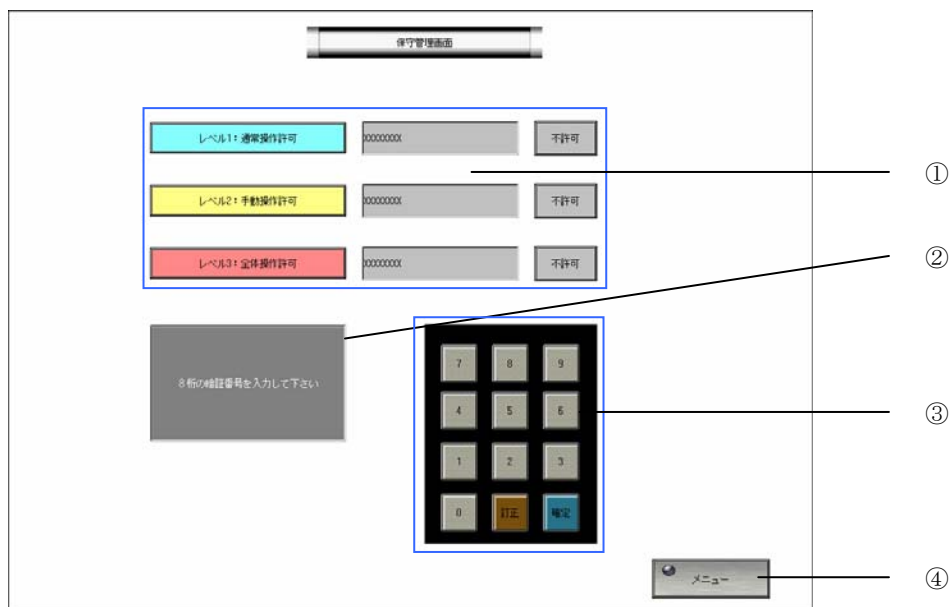
番号	項目	動作	レベル	使用コントロール
①	マニュアル表示	マニュアルの各ページを画像ファイル(bmp ファイル)を表示します。	-	ピクチャ
②	ページ切換ボタン	マニュアルのページを前後に切り替えます。	0	ボタン
③	メニューボタン	ボタンを押すと、メニュー画面をウィンドウ表示します。	0	ボタン

### MEMO

- ◆ テンプレート画面では、bmp データを C 言語コントローラ内に保存していますが、ユーザがプログラムを修正することで、C 言語コントローラに装着した SD カード・USB メモリなど、C 言語コントローラの外部の記憶媒体に保存することも可能です。

## 保守管理画面

本画面では、ユーザセキュリティのログイン画面を表示しています。



番号	項目	動作	レベル	使用コントロール
①	ログイン	操作権限のレベルを変更したい場合に、各レベルのボタンを押します。 ボタンを押すと、パスワード入力欄が入力可能となりますので、テンキーによりパスワードを入力します。	0	ボタン、テキストボックス
②	ガイダンス	メッセージ(固定文字列)を表示します。	-	スタティックテキスト
③	テンキー	パスワード入力用のテンキーです。	0	ボタン、ピクチャ
④	メニューボタン	ボタンを押すと、メニュー画面をウィンドウ表示します。	0	ボタン

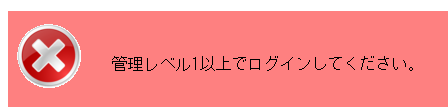


### ユーザセキュリティについて

テンプレート画面の各画面で、入力を受け付ける部品は、あるレベル以上の権限でログインしていないと操作できません。

例えば、「レベル 2」が設定された部品は、レベル 2 以上でログインした場合に操作できますが、レベル1でログインしている場合には操作できません。

レベルの異なる部品を操作した場合は、以下のメッセージが表示されます。



レベル 3 が一番多くの権限を持ち、すべての部品を操作できます。

レベルの番号が小さいほど制約が多くなり、操作できる部品が少なくなります。

本画面でレベルを変更する手順は、以下のとおりです。

1. 変更したいレベルのボタンを押します。

2. パスワードを入力します。

入力は、画面右下のテンキーで行います。入力後、[確定]ボタンを押します。

3. 正しいパスワードが入力されていた場合は、レベルが変更されます。

なお、誤ったパスワードが入力された場合には、「パスワードが違います。」とメッセージが表示され、パスワード入力欄がクリアされます。

3 回連続してパスワードの入力をミスすると、「パスワード入力を 3 回間違えました。」と表示され、パスワード入力が強制的に終了します。この場合、再度レベルボタンを押すことで、パスワード入力が可能です。

本アプリケーションでは、パスワードとして、以下が設定されています。

レベル	パスワード
レベル 0	(なし)
レベル 1	11111111
レベル 2	22222222
レベル 3	33333333

なお、5 分間、何も入力がない場合は、自動的にレベル 0 の状態になります。

**MEMO**

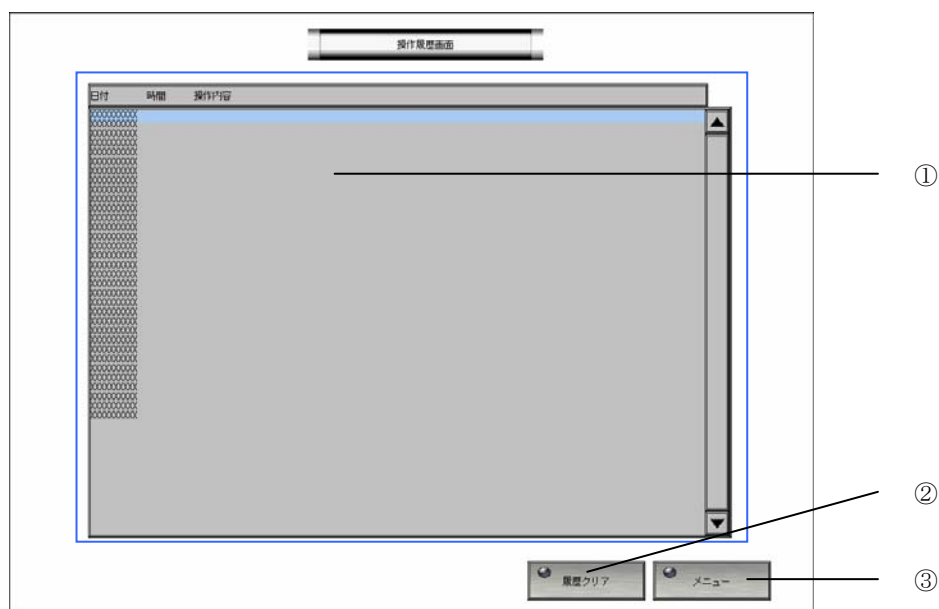
- ◆ パスワードは、ユーザがプログラムを修正することで、変更することができます。

**操作履歴画面**

本画面では、操作履歴の一覧を表示します。

操作履歴の対象となる操作は以下のとおりです。

- ・手動操作画面のボタン操作



## GENWARE3 アプリケーション設計ガイド

番号	項目	動作	レベル	使用コントロール
①	操作履歴	操作履歴を一覧表示します。	0	リスト、スタティックテキスト
②	履歴クリア	履歴をクリアします。	1	ボタン
③	メニューボタン	ボタンを押すと、メニュー画面をウィンドウ表示します。	0	ボタン

**操作履歴について**

操作履歴の情報は、各操作を行った時点で C 言語コントローラ内にファイル保存します。本画面の操作履歴の一覧では、そのファイルの情報を読み出して表示しています。

また、操作履歴はリングバッファで保存され、最大 100 件まで保存します。

操作履歴と画面の「操作内容」に表示する文字列の関連づけは、CSV ファイルで行います。

CSV ファイルには、

ID, 文字列リソース ID

を登録個数分、記載します。

ID	CI SKETCH で作成したコントロール名、パネル名で、操作対象となるものを指定します。
文字列リソース ID	操作履歴一覧に表示する文字列は、CI SKETCH で文字列リソースとして登録します。その文字列リソース ID を指定します。

テンプレート画面のアプリケーションが起動するときに、CSV ファイルを読み込みます。

手動操作画面の各ボタンが押された時に、そのときの日時(年月日、時分秒)と、銘板に設定された文字列を記録します。

**MEMO**

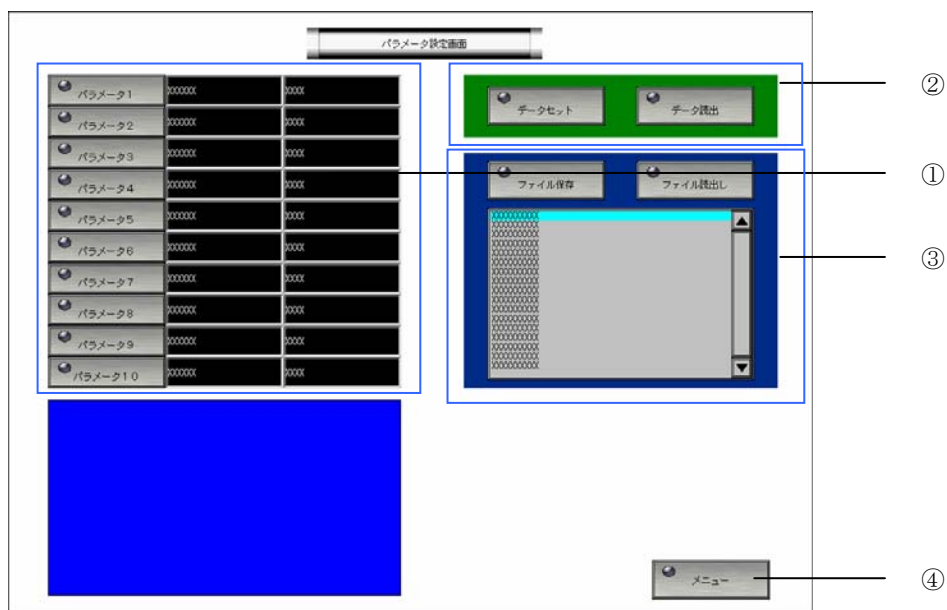
- ◆ 保守管理画面のテンキーの各ボタンはボタン操作としてではなく、文字・数値入力として扱っています。
- ◆ テンプレート画面では、ログデータを C 言語コントローラ内に保存していますが、ユーザがプログラムを修正することで、C 言語コントローラに装着した SD カード・USB メモリなど、C 言語コントローラの外部の記憶媒体に保存することも可能です。

## パラメータ設定画面

本画面では、パラメータの設定、設定保存を行う画面である(簡易なレシピ機能)。パラメータは、以下による変更が可能です。

- ・画面内で数値入力
- ・レシピファイルからの読出し

変更したデータは、シーケンサまたはレシピファイルに書き込むことができます。



番号	項目	動作	レベル	使用コントロール
①	パラメータ	パラメータを設定します。本エリアを押すと、テンキー画面(ウィンドウ)が表示されます。	3	ボタン、テキストボックス
②	データセット／データ読出	「データセット」を押すと、①の値をシーケンサに書き込みます。 「データ読出」を押すと、シーケンサからデータを読み出し①に反映します。	3	ボタン
③	レシピファイル	C 言語コントローラ内に格納されたレシピファイルが一覧表示されます。 レシピファイルを選択し、「ファイ	3	ボタン、リスト

		ル保存」を押すと、①の設定内容を、選択したレシピファイルに書き込みます。 「ファイル読出」を押すと、選択したレシピファイルからデータを読み出し①に反映します。		
④	メニューボタン	ボタンを押すと、メニュー画面をウィンドウ表示します。	0	ボタン

### レシピファイルについて

レシピファイルは、ファイル名は「RecipeXX.csv」(XX=00～09)とし、C 言語コントローラ内に格納します。

各ファイルには、以下のようなフォーマットでパラメータを設定します。

10 進数値,16 進数値

を 10 行、記載します。

### MEMO

- ◆ レシピファイル内で、範囲外の数値があった場合は、その箇所は 0 として読み出します。
- ◆ テンプレート画面を起動し、初めてこの画面を開いた時は、各パラメータの値は 0 で表示します。
- ◆ テンプレート画面では、レシピファイルを C 言語コントローラ内に保存していますが、ユーザがプログラムを修正することで、C 言語コントローラに装着した SD カード・USB メモリなど、C 言語コントローラの外部の記憶媒体に保存することも可能です。

## メニュー画面

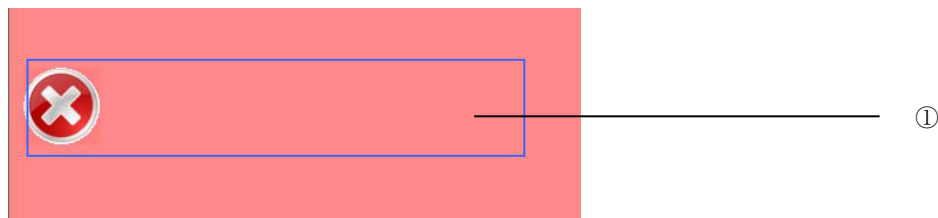
画面切換、表示言語切換を行うメニュー画面です。



番号	項目	動作	レベル	使用コントロール
①	画面切換	選択した各画面に切り換えます。画面切換時、メニュー画面は閉じます。	0	ボタン
②	言語切換	テンプレート画面内で表示する文字列を、日本語／英語に切り換えます。	0	ボタン

## メッセージ画面

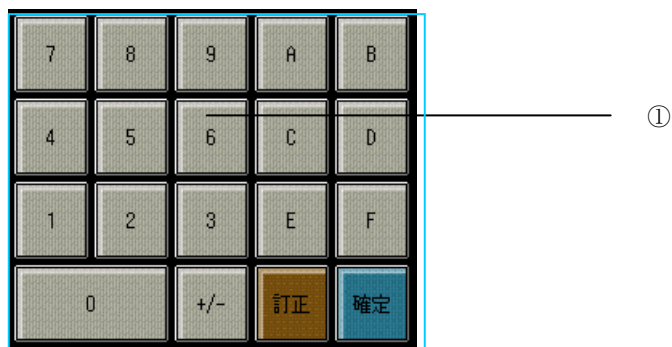
メッセージ表示に使用する画面です。



番号	項目	動作	レベル	使用コントロール
①	メッセージ	ユーザセキュリティ機能で、レベルの違う部品を操作したときのメッセージを表示します。	-	ピクチャ、スタティックテキスト

## テンキー画面

テンキー画面です。



番号	項目	動作	レベル	使用コントロール
①	テンキー	0～9、A～Fを入力できます。 「確定」を押すと、入力が確定します。 「訂正」を押すと、入力した数値をクリアします。	-	ボタン

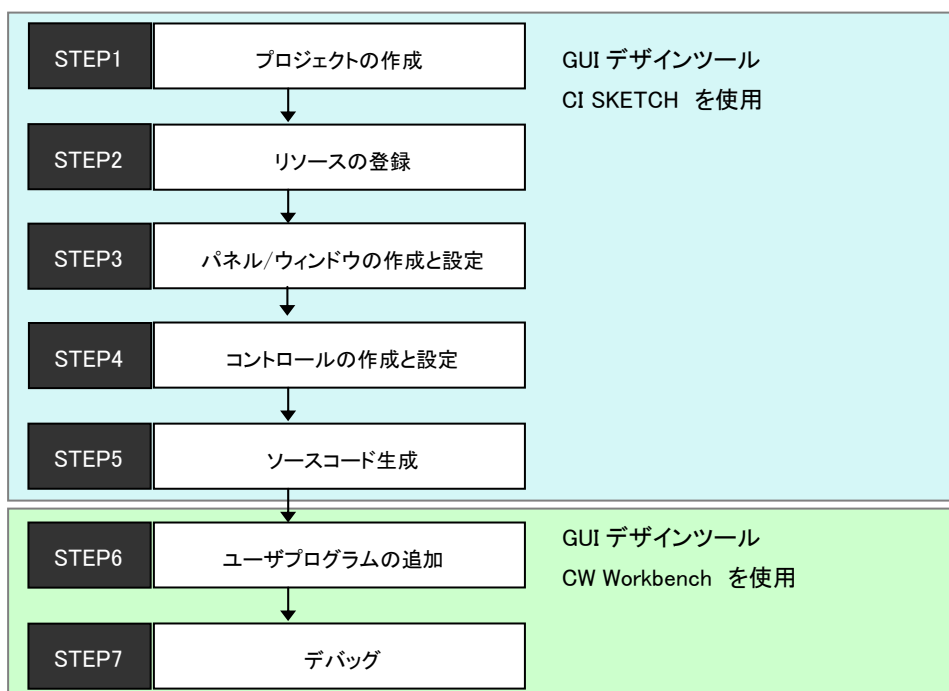
## 1-3 開発環境

テンプレート画面を開発するために必要な開発環境は、以下のとおりです。

項目	ツール
GUI 作成ツール	GENWARE3
コンパイラ	CW Workbench

## 1-4 開発の流れ

GENWARE3 を用いた開発の流れは以下のとおりです。







## 第2章 CI SKETCH による画面作成

2-1 プロジェクトの作成.....	2-1
2-2 リソースの登録.....	2-5
2-3 パネル/ウィンドウの作成と設定 .....	2-22
2-4 コントロールの作成と設定.....	2-30
2-5 スクリーンプロパティの設定.....	2-33
2-6「手動操作画面」の作成 .....	2-34
2-7「モニタ画面」の作成.....	2-40
2-8「エラー管理画面」の作成 .....	2-46
2-9「運転データ画面」の作成.....	2-52
2-10「マニュアル管理画面」の作成 .....	2-59
2-11「保守管理画面」の作成.....	2-62
2-12「操作履歴画面」の作成.....	2-67
2-13「パラメータ設定画面」の作成 .....	2-70
2-14「メニュー画面」の作成.....	2-75
2-15「メッセージ画面」の作成.....	2-77
2-16「テンキー画面」の作成.....	2-78

## 2-1 プロジェクトの作成

プロジェクトを新しく作成して画面の編集を開始するまでの操作方法について説明します。

1. [ファイル]メニューの[プロジェクトの新規作成]、またはツールバーの[プロジェクトの新規作成]ボタンを選択します。



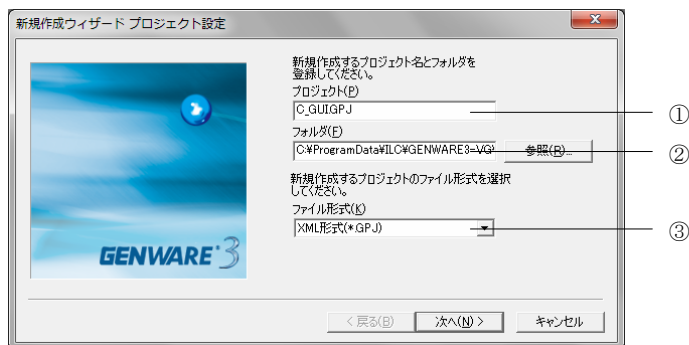
2. プロジェクト新規作成ウィザードが表示されます。ウィザードにしたがってプロジェクトの設定を行います。
3. [完了]ボタンをクリックすると、プロジェクトが作成されます。

### MEMO

- ◆ プロジェクトが作成されると、ページ番号 0 のパネルが自動的に作成されます。
- ◆ プロジェクトを既に編集している状態で、プロジェクトを新規作成すると、編集中のプロジェクトの保存確認メッセージが表示されます。

以下にプロジェクト新規作成ウィザードの設定方法について説明します。

### (1) プロジェクト設定



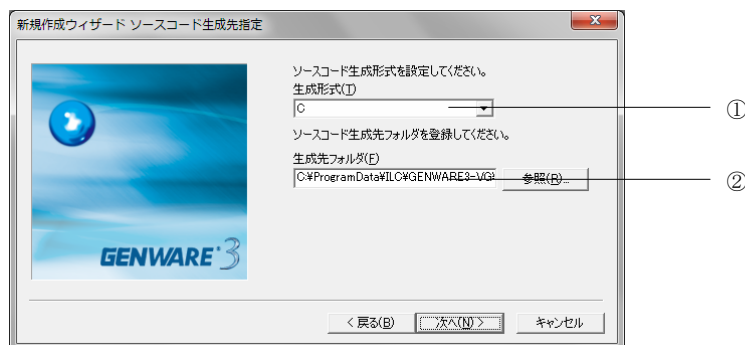
## GENWARE3 アプリケーション設計ガイド

番号	項 目	内 容
①	プロジェクト名	プロジェクト名を入力します。
②	フォルダ	プロジェクトを格納するフォルダをフルパスで指定します。
③	ファイル形式	プロジェクトファイルの形式を「バイナリ形式(*.IPP)」/「XML 形式(*.GPJ)」から選択します。

今回のサンプルアプリケーションでは上記項目をそれぞれ以下のように設定してください。

番号	項 目	設 定 内 容
①	プロジェクト名	C_GUI.GPJ
②	フォルダ	Windows XP の場合 C:¥Documents and Settings¥All Users¥Application Data¥ILC¥ GENWARE3-VG¥ ¥MyProj Windows 7 の場合 C:¥ProgramData¥ILC¥GENWARE3-VG¥MyProj (プロジェクトを保存するフォルダは任意ですので、上記以外のフォルダを指定してもかまいません。)
③	ファイル形式	XML 形式(*.GPJ)

## (2) ソースコード生成先指定

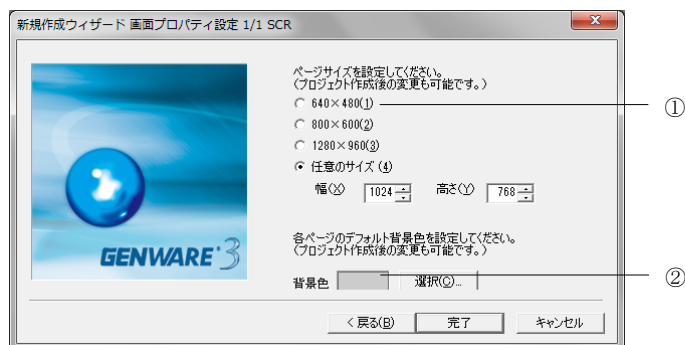


番号	項 目	内 容
①	生成形式	ソースコードの生成形式をリストから選択します。
②	生成先フォルダ	ソースファイルを生成するフォルダをフルパスで指定します。

今回のサンプルアプリケーションでは上記項目を以下のように設定してください。

番号	項 目	設 定 内 容
①	生成形式	C (生成形式として、C++言語の指定も可能です。)
②	生成先フォルダ	C 言語: (プロジェクト作成フォルダ)¥C_GUI¥C C++言語: (プロジェクト作成フォルダ)¥C_GUI¥Cpp ((1)にてプロジェクト名を“C_GUI”と設定した場合、自動的に上記のパスが設定されます。ソースファイルを保存するフォルダは任意ですので、上記以外のフォルダを指定してもかまいません。)

## (3) 画面プロパティ設定



番号	項 目	内 容
①	ページサイズ	ページのデフォルトの画面サイズを選択します。
②	背景色	ページのデフォルトの背景色を指定します。「選択」ボタンをクリックして表示される[色の設定]ダイアログから色を選択してください。

今回のサンプルアプリケーションでは上記項目を以下のように設定してください。

番号	項 目	内 容
①	ページサイズ	「任意のサイズ」を選択し 幅:1024、高さ:768 に設定します。
②	背景色	白色(RGB(255,255,255))

プロジェクト新規作成ウィザードを完了すると、「スクリーン選択」が表示されます。



[OK]ボタンを押下すると、パネルが 1 枚作成されます。[キャンセル]ボタンを押下すると、パネルは作成されません。

## 2-2 リソースの登録

リソースとは画像イメージやコントロールに設定するキャプション文字列やフォントなど、複数のコントロールやページで共通して用いるデータのことです。

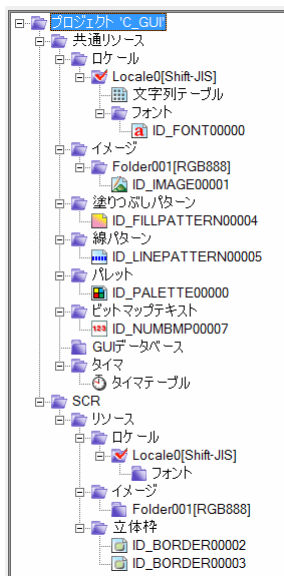
GENWARE3 では画像や文字列などのデータをリソースとしてプロジェクトに登録する仕組みを持っており、リソースは登録されたプロジェクト内で一元管理されます。

リソースの登録や編集、削除は Solution Explorer から行います。

プロジェクト新規作成時、Solution Explorer は以下のように設定されています。

リソースの登録や編集、削除は Solution Explorer から行います。

プロジェクト新規作成時、Solution Explorer は以下のように設定されています。



### MEMO

- ◆ 本サンプルでは、「文字列」、「フォント」、「イメージ」、「GUI データベース」の登録、編集を行います。

## GENWARE3 アプリケーション設計ガイド

リソースとして登録できるデータは以下のとおりです。

項 目		内 容
ロケール	文字列リソース	コントロールで使用する文字列やウィンドウタイトルなどの文字列データ。
	フォントリソース	フォントデータ。
イメージ		背景画やピクチャコントロールで使用する画像データ (BMP、JPG、PNG)。
塗りつぶしパターン		コントロールおよび図形で使用する塗りつぶしパターンデータ。
線パターン		図形で使用する線パターンデータ。
立体枠		コントロールに設定する外枠。
パレット		色指定時に使用するパレット。
ビットマップテキスト		ビットマップテキストコントロールで使用する、デザイン化された一連のビットマップテキストデータ。
GUI データベース		GUI アプリケーションで表示に必要な情報を管理するデータベース。
タイマ		GUI アプリケーションが表示に使用するタイマ。

## MEMO

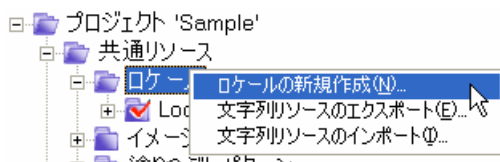
- ◆ 今回のサンプルアプリケーションで登録する各リソースの登録内容は「付録 1-1 リソース設定」を参照してください。



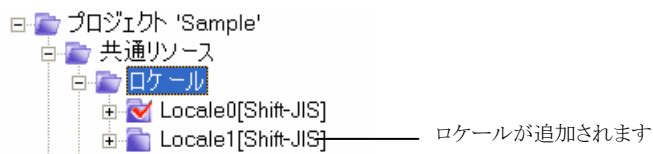
## ロケールの登録

今回のサンプルアプリケーションでは日本語と英語の 2 つのロケールを使用します。  
ロケールの登録方法を以下に説明します。

1. Solution Explorer の「ロケール」にマウスカーソルをあわせ、マウス右クリックで表示されるポップアップメニューから「ロケールの新規作成」を選択します。



2. 新規ロケールが登録されます。



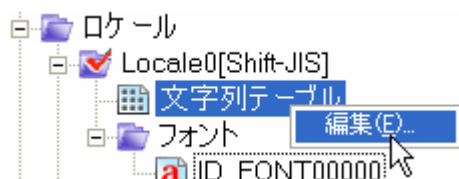
3. 設定を行いたいロケール名をダブルクリック、または、ロケール名にマウスカーソルを合わせ、マウス右クリックで表示されるポップアップメニューから「プロパティ」を選択します。
4. プロパティウィンドウにてロケールの設定ができるようになります。

項 目	内 容
ロケール名	ロケール名を設定します。 ロケール名の先頭の文字は、必ず半角英字(A～Z、a～z)を設定します。 また、2 文字目以降に使用できる文字は、半角英数字または「_」(アンダースコア)です。 設定可能なロケール名は、最大 16 文字です。
文字コード	ロケールの文字コードを選択します。

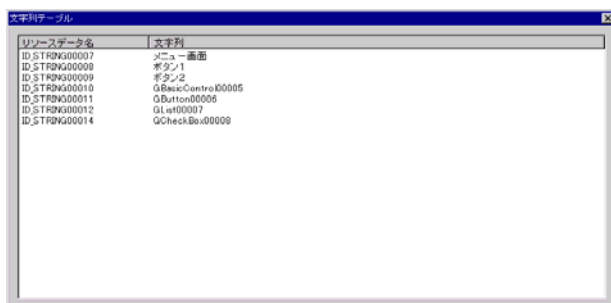
## 文字列リソースの登録

リソースの登録方法は以下のとおりです。

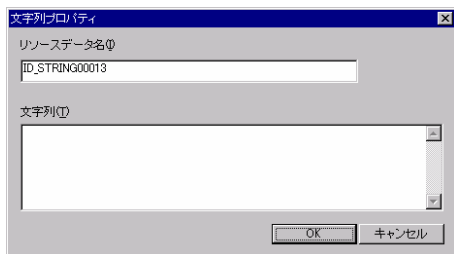
1. Solution Explorer の「文字列テーブル」にマウスカーソルをあわせ、ダブルクリック、または、マウス右クリックで表示されるポップアップメニューから「編集」を選択します。



2. 「文字列テーブル」が表示されます。



3. 新規に文字列リソースを登録する場合は、任意のリソースを選択し、マウス右クリックで表示されるポップアップメニューから「新規作成」を選択するか、リソースが表示されていないエリアをダブルクリックします。  
[文字列プロパティ]ダイアログが表示されます。



4. 「リソースデータ名」、「文字列」を入力し、[OK]ボタンをクリックしてください

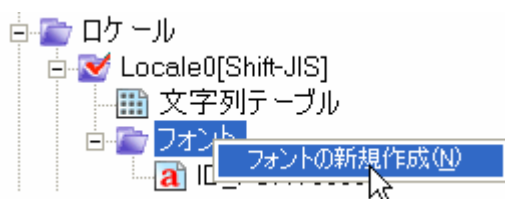
**MEMO**

- ◆ 文字列リソースのリソースデータ名の先頭の文字は、必ず半角英字(A～Z、a～z)を設定します。また、2 文字目以降に使用できる文字は、半角英数字または「\_」(アンダースコア)です。設定可能なリソースデータ名は、最大 32 文字です。
- ◆ 文字列は半角・全角に関わらず 256 文字まで入力できます。
- ◆ 文字列リソースは、以下の場合にも自動的に作成されます。
  - ・コントロールのプロパティ設定で、コントロールに表示する文字列を設定したとき。
- ◆ ロケールを複数登録しているとき、いずれかのロケールで文字列リソースを登録すると、ほかのロケールの文字列テーブルにも、新規作成した文字列リソースデータ名、文字列が反映されます。
- ◆ 文字列リソースは、プロジェクト共通情報の設定です。したがって、マルチユーザ開発を行っている場合、プロジェクト共通情報の編集権を持つユーザだけが文字列リソースを編集できます。

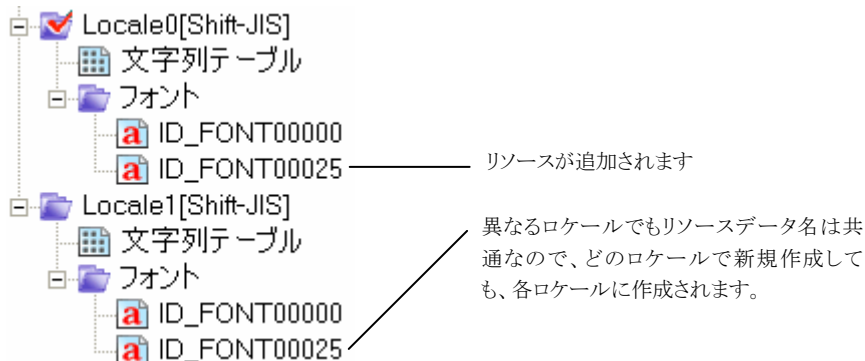
## フォントリソースの登録

フォントリソースの登録方法は以下のとおりです。

1. Solution Explorer の「フォント」にマウスカーソルをあわせ、マウス右クリックで表示されるポップアップメニューから「フォントの新規作成」を選択します。



2. 新規フォントリソースが登録されます。



3. 設定を行いたいフォントリソースデータ名をダブルクリック、または、フォントリソースデータ名にマウスカーソルを合わせ、マウス右クリックで表示されるポップアップメニューから「プロパティ」を選択します。
4. プロパティウィンドウにフォントリソースの設定項目が表示されます。

項 目	内 容
リソースデータ名	リソースデータ名を設定します。 リソースデータ名の先頭の文字には、必ず半角英字(A～Z、a～z)を設定します。 また、2 文字目以降に使用できる文字は、半角英数字または「_」(アンダースコア)です。 設定可能なリソースデータ名は、最大 32 文字です。
フォント名	フォント名を設定します。
フォントサイズ	フォントサイズを設定します。選択したフォントがベクタフォントの場合に有効になります。
倍率 横	文字サイズの横倍率を設定します。
倍率 縦	文字サイズの縦倍率を設定します。
太さ	文字の太さを「THIN」、「NORMAL」、「BOLD」から選択します。
イタリック	イタリック表示を「あり」、「なし」から選択します。
色縁取り	文字縁取りを「あり」、「なし」から選択します。
文字の読み方向	文字の読み方向を「左から右」、「右から左」から選択します。
拡張スタイル	「なし」固定です。
文字間隔	「0」固定です。

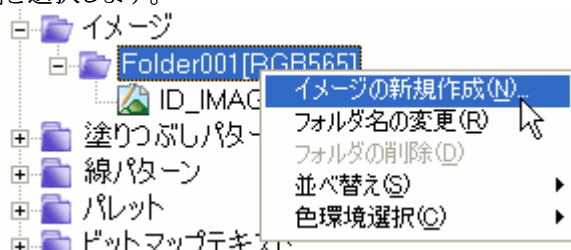
**MEMO**

- ◆ リソースビューで同時に複数のフォントリソースを選択している場合、一括でプロパティの変更ができます。設定を変更した項目は、選択されているすべてのフォントリソースに適用されます。  
ただし、リソースデータ名を変更することはできません。
- ◆ 倍率、太さ、フォントスタイルを設定した場合でも、CI SKETCH の編集画面上はこれらの設定で表示されません。
- ◆ フォントリソースは、プロジェクト共通情報の設定です。したがって、マルチユーザ開発を行っている場合、プロジェクト共通情報の編集権を持つユーザだけがフォントリソースを編集できます。

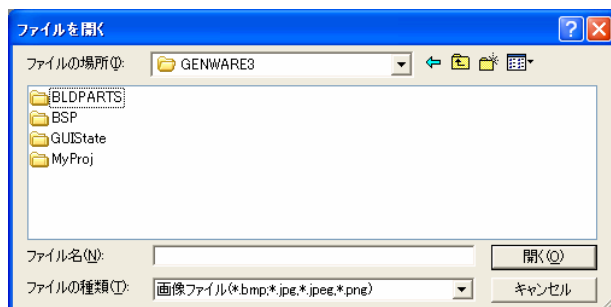
## イメージリソースの登録

イメージリソースの登録方法は以下のとおりです。

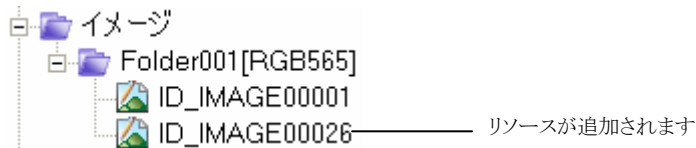
1. Solution Explorer でイメージリソースを作成したいフォルダにマウスカースルを合わせ、マウス右クリックで表示されるコンテキストメニューから[イメージの新規作成]を選択します。



2. 「ファイルを開く」ダイアログが表示されます。



3. イメージリソースとして登録したいイメージファイルを選択し、[開く]ボタンを押下します。イメージファイルを複数選択することもできます。
4. イメージリソースが追加されます。イメージファイルを複数選択した場合には、同時に複数のイメージリソースを追加できます。



## GENWARE3 アプリケーション設計ガイド

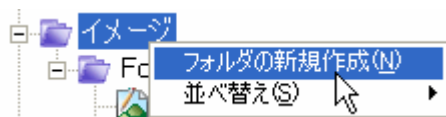
5. 設定を行いたいイメージリソースデータ名をダブルクリック、または、イメージリソースデータ名にマウスカーソルを合わせ、マウス右クリックで表示されるポップアップメニューから「プロパティ」を選択します。
6. プロパティウィンドウにイメージリソースの設定項目が表示されます。

項 目	内 容
リソースデータ名	リソースデータ名を設定します。 リソースデータ名の先頭の文字には、必ず半角英字(A～Z、a～z)を設定します。 また、2 文字目以降に使用できる文字は、半角英数字または「_」(アンダースコア)です。 設定可能なリソースデータ名は、最大 32 文字です。
ファイル名	イメージファイルを設定します。 [...]ボタンをクリックすると、ファイルセレクトが表示されます。 イメージファイルを設定してください。設定可能なファイルは、BMP ファイル、JPG ファイル、PNG ファイルです。 選択したファイルは、プロジェクトフォルダの中にコピーされます。
パレットの指定	イメージリソースで使用するパレットを指定するかしないかを選択します。パレットを指定する場合は「あり」を選択します。指定しない場合は「なし」を選択します。
パレット	イメージリソースで使用するパレットを、パレットリソースから選択します。
色数	イメージリソースのパレットで使用する表示色数を、「2 色」/「4 色」/「16 色」/「256 色」から選択します。
形式	イメージリソースを圧縮するかしないかを選択します。 [圧縮あり]を選択した場合、イメージのデータサイズを小さくできます。
透過	透過機能を使用するかどうかを設定します。
透過色	透過色を設定します。 [透過]を[あり]に設定した場合のみ設定が反映されます。

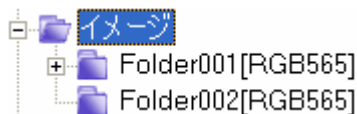
### イメージリソースのフォルダ管理

イメージリソースは数が多くなりやすく管理が大変なため、イメージをグループごとにフォルダに分類して管理する方法を説明します。

1. Solution Explorer で「イメージ」フォルダにマウスカーソルをあわせ、マウス右クリックで表示されるポップアップメニューから「フォルダの新規作成」を選択します。



2. リソースビューに、フォルダが作成されます。

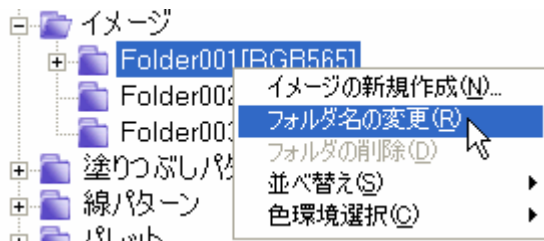


今回のサンプルアプリケーションでは 3 つのフォルダを作成します。

### フォルダ名の変更

イメージリソースのフォルダ名を変更します。

1. 名前を変更したいフォルダにマウスカーソルをあわせ、マウス右クリックで表示されるポップアップメニューから「フォルダ名の変更」を選択します。



2. フォルダ名を入力し、Enter キーを押下します。  
今回のサンプルアプリケーションでは 3 つのフォルダ名にそれぞれ「BackGround」、「Button」、「Icon」と入力します。





### イメージリソースの移動

ほかのフォルダにイメージリソースを移動させます。

1. 移動したいイメージリソースをマウスで選択し、移動先のフォルダにドラッグ&ドロップします。



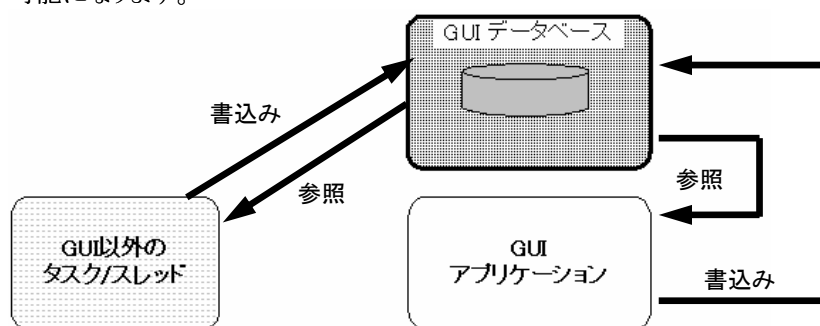
#### MEMO

- ◆ 複数のイメージリソースを選択して移動させることもできます。

## GUI データベースの登録

GUI データベースは、GUI アプリケーションの表示に必要な各種の情報を格納し、管理するための機構です。

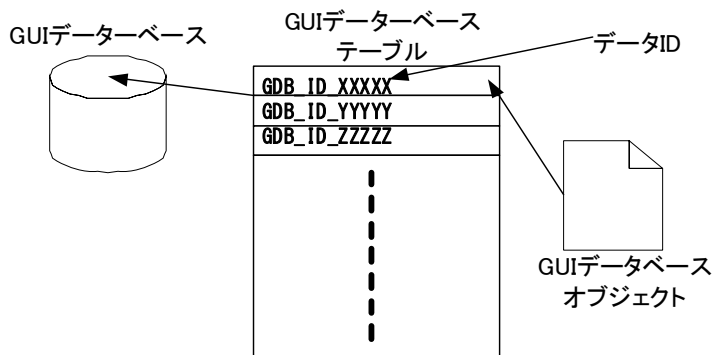
GUI データベースによって、外部のタスク/スレッドと GUI との間で、データの共有が可能になります。



GUI データベースに格納する各種の情報は、GUI データベースオブジェクトという単位で定義します。

GUI データベースオブジェクトは、GUI データベーステーブルに登録され、それぞれにデータ ID が振られます。

このデータ ID を用いて、GUI データベース内の情報を指定してアクセスを行います。



GUI データベースに対して行える機能は以下のとおりです。

- データの書き込み
- データの読み出し
- データ変更通知
- データ変更の有無確認
- 初期化処理と終了処理

### 変数の型を登録する

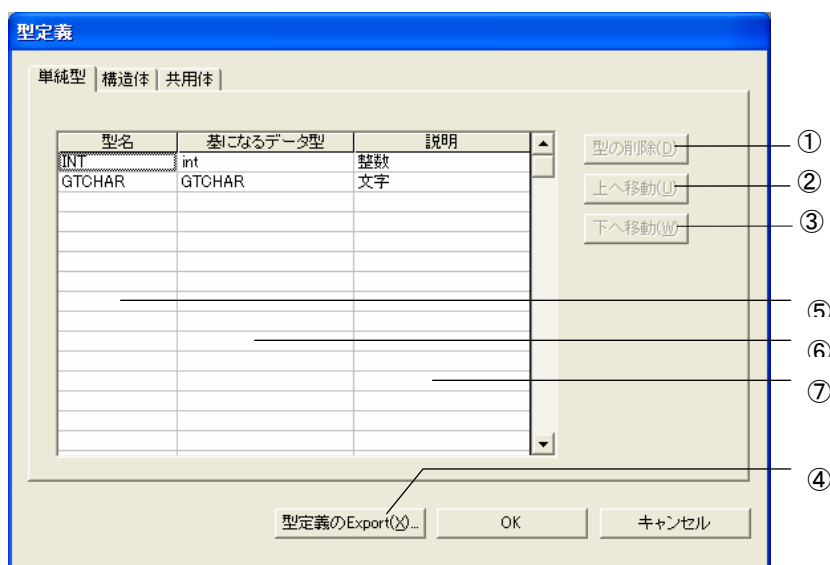
GUI データベースに使用する変数は、ユーザ独自の型や構造体として定義する場合があります。このため、GENWARE3 では変数の型をあらかじめ登録できるような仕組みを用意しています。

#### MEMO

- ◆ 色型定義の登録可能数の範囲は 0～1024 個です。構造体のメンバ変数は1～1024 個の範囲で設定可能です。
- ◆ GENWARE3 には、初期状態で以下が登録されています。

型名	型の種類	基となるデータ型	コメント
INT	SIMPLE	GInt32	整数
GTCHAR	SIMPLE	GTCHAR	文字

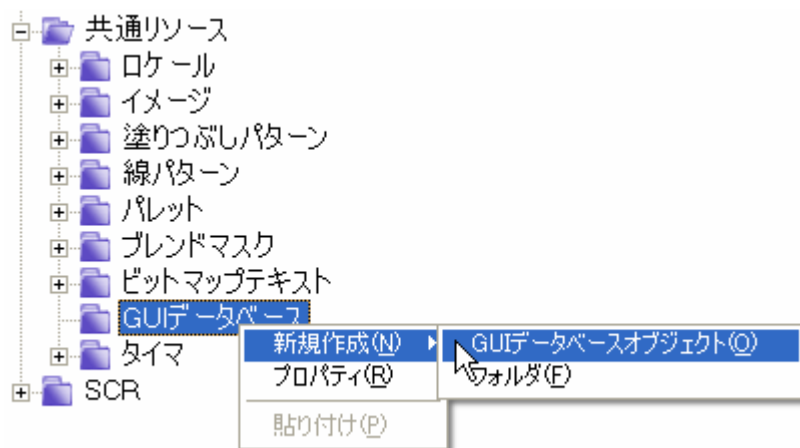
- 1.[ツール]メニューの[GUI データベース]-[型定義]を選択し、[型定義]ダイアログを表示します。



番号	項目	内 容
①	型の削除	選択中の登録されている型を削除します。 ただし、あらかじめ GENWARE3 にて登録された型については削除できません。
②	上へ移動	選択中の型を 1 つ上に移動します。
③	下へ移動	選択中の型を 1 つ下に移動します。
④	型定義の Export	登録された型定義をテキストファイルとしてエクスポートします。
⑤	型名	GUI データベースで使用する変数の型名を表示、設定します。 最大 31 文字まで設定できます。
⑥	基になるデータ型	GUI データベースで使用する変数のデータ型を表示、設定します。 使用するコンパイラに対応した型を設定してください。 最大 31 文字まで設定できます。
⑦	説明	GUI データベースで使用する変数の説明を表示、設定します。 最大 31 文字まで設定できます。

## GUI データベースの登録方法

1. SolutionExplorer の[GUI データベース]にマウскарソルを合わせ、マウス右クリックで表示されるコンテキストメニューから[新規作成]-[GUI データベースオブジェクト]を選択します。

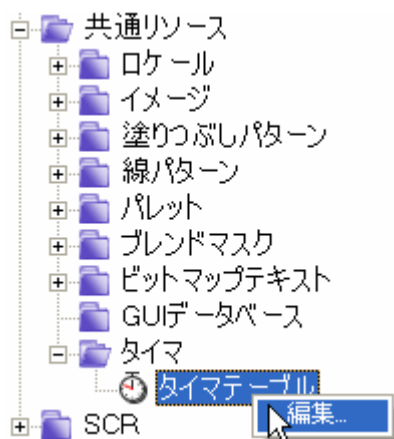


2. プロパティウィンドウに立体枠リソースの設定項目が表示されます。

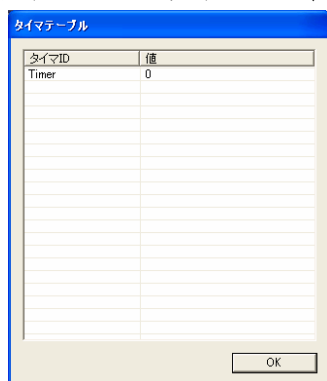
項 目	内 容
オブジェクト名	GUI データベースオブジェクトの名称です。(全半角 31 文字以内。大文字、小文字は区別されます。)
ID	データ内での識別子として使用します。(半角英数字+ '_' 31 文字以内。大文字、小文字は区別されます。)
変数名	static 宣言を「なし」にした場合に、直接呼び出して使用する変数名です。(半角英数字+ '_' 31 文字以内。大文字、小文字は区別されます。)
変数型	CI SKETCH 上で登録された変数型から選択して設定します。
配列サイズ	GUI データベースオブジェクトの配列の要素数を設定します。(1~32768)
static 宣言	static 宣言の「あり」、「なし」を設定します。

## タイマリソースの登録

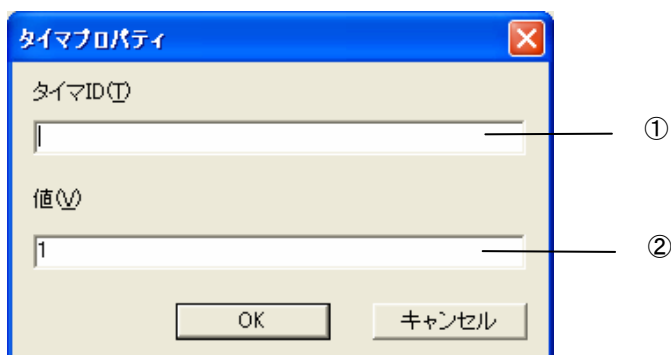
1. SolutionExplorer の[タイマテーブル]をダブルクリックします。またはマウス右クリックで表示されるコンテキストメニューから[編集]を選択します。



2. タイマテーブルダイアログが表示されます。



3. マウス右クリックで表示されるコンテキストメニューから[新規作成]を選択します。または、リソースが表示されていない部分をダブルクリックすると[タイマプロパティ]ダイアログが表示されます。  
[タイマプロパティ]ダイアログ上で、追加するタイマの設定を行い[OK]ボタンを押します。



番号	項 目	内 容
①	タイマ ID	タイマ ID が表示されます。 リソースデータ名の先頭の文字には、必ず半角英字(A～Z、a～z)を設定します。 また、2 文字目以降に使用できる文字は、半角英数字または「_」(アンダースコア)です。 設定可能なリソースデータ名は、最大 32 文字です。
②	値	タイマ ID の値が表示されます。 0～65535 の範囲で未使用の値を設定することができます。

今回のサンプルアプリケーションでは、以下の 5 つを登録します。

タイマ ID	値
ID_TIMER_DATA_LOGGING	100
ID_TIMER_FIVE_MINUTE	101
ID_TIMER_DRAWING	102
ID_TIMER_MESSAGE	103
ID_TIMER_REFRESH_LOGGING_DATA	104

## 2-3 パネル/ウィンドウの作成と設定

GENWARE3 で表示できる画面には、次の 3 種類があります。

項 目	内 容
パネル	スクリーン上に全画面で表示される画面です。
ウィンドウ	スクリーン上にウィンドウの状態が表示される画面です。
サブパネル	他のパネル／ウィンドウ内に表示できる共通画面です。

GENWARE3 ではパネル、ウィンドウ、サブパネルを総称し、ページと呼びます。  
今回のサンプルアプリケーションでは各画面を以下のような構成で作成します。

ページ名	ページの種類	画面名
Panel_ManualOperation	パネル	手動操作画面
Panel_Monitor	パネル	モニター画面
Panel_ErrorManagement	パネル	エラー管理画面
Panel_OperDataManagement	パネル	運転データ管理画面
Panel_ManualManagement	パネル	マニュアル管理画面
Panel_Maintenance	パネル	保守管理画面
Panel_OperationHistory	パネル	操作履歴画面
Panel_ParameterSetting	パネル	パラメータ設定画面
Window_Menu	ウィンドウ	メニュー画面
Window_Message	ウィンドウ	メッセージ画面
Window_NumericKeypad	ウィンドウ	テンキー画面



## パネルの新規作成

今回のサンプルアプリケーションにおいてパネルを用いて作成される画面は「手動操作画面」、「モニター画面」、「エラー管理画面」、「運転データ管理画面」、「マニュアル管理画面」、「保守管理画面」、「操作履歴画面」、「パラメータ設定画面」となります。

パネルの新規作成手順は以下のとおりです。

1. [ファイル]メニューの[パネルの新規作成]またはツールバーの[パネルの新規作成]ボタンを押します。

[ツールバー]



2. パネルが作成されます。

### MEMO

- ◆ 未使用ページ番号の中で、一番若いページ番号が新しいパネルに使用されます。
- ◆ 新規プロジェクト作成時、パネルが1つ自動的に作成されます。

## パネルのプロパティ設定

新規で作成したパネルには何も設定がされていません。作成したパネルの背景画やコールバックなどのプロパティ設定手順は以下のとおりです。

1. [設定]メニューの[パネル/ウィンドウプロパティ]または、マウス右クリックで表示されるポップアップメニューから[パネル/ウィンドウプロパティ]を選択します。
2. プロパティウィンドウにパネルプロパティが表示されます。

パネルのプロパティ設定項目の内容は以下のとおりです。

パネル/ウィンドウ	
パネル名	Panel00000
コメント	
説明	
WIDTH	800
HEIGHT	600
背景色	
背景画	なし
背景イメージ	ID_IMAGE00001
ブリンクOFF時間(ms)	100
ブリンクON時間(ms)	100
OnCreate	なし
PreDelete	なし
PreKeyPress	なし
PreKeyRelease	なし
OnUser	なし
PrePress	なし
PreRelease	なし
PrePaint	なし
OnTimer	なし
OnActive	なし
OnUpdateDB	なし
OnMouseMove	なし

項 目	内 容
パネル名	パネル名を設定します。半角文字列を入力してください。(最大 31 文字) 先頭の文字には、必ず半角英字(A～Z、a～z)を設定します。 また、2 文字目以降に使用できる文字は、半角英数字または「_」(アンダースコア)です。
コメント	パネルのコメントを設定することができます。(最大 32 文字) 全角文字・半角英数・記号・半角カナの入力が可能です。 ソースコードには生成されません。
説明	パネルの説明を設定することができます。(最大 256 文字) 全角文字・半角英数・記号・半角カナの入力が可能です。 ソースコードには生成されません。
WIDTH	パネルの幅をドット単位で設定します。(1～2560)
HEIGHT	パネルの高さをドット単位で設定します。(1～1920)
背景色	背景色を設定します。
背景画	背景画を使用する場合は「あり」、使用しない場合は「なし」を選択します。
背景イメージ	背景画として使用するイメージリソースの ID を選択します。
ブリンク OFF 時間	コントロールのブリンク OFF 状態の表示(通常を表示)時間を ms 単位で設定します。(100～60000)
ブリンク ON 時間	コントロールのブリンク ON 状態の表示時間を ms 単位で設定します。(100～60000)
OnCreate	表示パネル上にあるコントロールの Create 関数の処理を行った後に呼び出されるコールバックです。

項 目	内 容
PreDelete	表示パネル上にあるコントロールの Delete 関数の処理を行う前に呼び出されるコールバックです。
PreKeyPress	表示パネル上にあるコントロールの KeyPress 関数の前に呼び出され、処理を追加するためのコールバックです。戻り値によりコントロールの KeyPress 関数の実行有無を制御します。
PreKeyRelease	表示パネル上にあるコントロールの KeyRelease 関数の前に呼び出され、処理を追加するためのコールバックです。戻り値によりコントロールの KeyRelease 関数の実行有無を制御します。
OnUser	パネルへユーザイベントが呼ばれた後、ユーザ独自の処理を追加するためのコールバックです。
PrePress	表示パネル上にあるコントロールの LButtonPress 関数の前に呼び出され、処理を追加するためのコールバックです。戻り値によりコントロールの LButtonPress 関数実行の有無を制御します。
PreRelease	表示パネル上にあるコントロールの LButtonRelease 関数の前に呼び出され、処理を追加するためのコールバックです。戻り値によりコントロールの LButtonRelease 関数実行の有無を制御します。
PrePaint	表示パネル上にあるコントロールの Paint 関数の前に呼び出され、処理を追加するためのコールバックです。戻り値によりコントロールの PrePaint 関数実行の有無を制御します。
OnTimer	パネルへタイマイベントが呼ばれた後に実行する処理を追加するためのコールバックです。
OnActive	パネルの表示完了時に呼び出されるコールバックです。
OnUpdateDB	データベース変更通知が送られると呼び出されるコールバックです。
OnMouseMove	パネル上で座標位置が変わるたびに呼び出されるコールバックです。

### 3. 各項目の設定を行います。

#### MEMO

- ◆ 今回のサンプルアプリケーションのパネルプロパティ設定内容は「付録 1-4 パネル/ウィンドウプロパティ設定」を参照してください。

## ウィンドウの新規作成

今回のサンプルアプリケーションにおいてウィンドウを用いて作成される画面は「メニュー画面」、「メッセージ画面」、「テンキー画面」となります。

ウィンドウの新規作成手順は以下のとおりです。

1. [ファイル]メニューの[ウィンドウの新規作成]またはツールバーの[ウィンドウの新規作成]ボタンを選択します。



2. ウィンドウが作成されます。

### MEMO

- ◆ 未使用ページ番号の中で、一番若いページ番号が新しいウィンドウに使用されます。

## ウィンドウのプロパティ設定

ウィンドウにもパネルと同様に背景画を設定します。

登録したリソースをウィンドウの背景画のプロパティとして設定する手順は以下のとおりです。

1. [設定]メニューの[パネル/ウィンドウプロパティ]または、マウス右クリックで表示されるポップアップメニューから[パネル/ウィンドウプロパティ]を選択します。
2. プロパティウィンドウにウィンドウプロパティが表示されます。

ウィンドウのプロパティ設定項目の内容は以下のとおりです。

パネル/ウィンドウ	
ウィンドウ名	Window00002
コメント	
説明	
X	0
Y	0
WIDTH	320
HEIGHT	240
背景色	
背景画	なし
背景イメージ	ID_IMAGE00001
タイトル文字列	
タイトルバーの有無	あり
クローズボタンの有無	あり
ウィンドウ枠の有無	あり
ブリンクOFF時間(ms)	100
ブリンクON時間(ms)	100
レイヤ	レイヤ1
OnCreate	なし
PreDelete	なし
PreKeyPress	なし
PreKeyRelease	なし
OnUser	なし
PrePress	なし
PreRelease	なし
PrePaint	なし
OnTimer	なし
OnActive	なし
OnUpdateDB	なし
OnMouseMove	なし

項 目	内 容
ウィンドウ名	ウィンドウ名を設定します。(最大 31 文字) 先頭の文字には、必ず半角英字(A～Z、a～z)を設定します。 また、2 文字目以降に使用できる文字は、半角英数字または「_」(アンダースコア)です。
コメント	ウィンドウのコメントを設定することができます。(最大 32 文字) 全角文字・半角英数・記号・半角カナの入力が可能です。 ソースコードには生成されません。
説明	ウィンドウの説明を設定することができます。(最大 256 文字) 全角文字・半角英数・記号・半角カナの入力が可能です。 ソースコードには生成されません。
X	ウィンドウ左上の表示位置の X 座標をドット単位で設定します。(0～2559)
Y	ウィンドウ左上の表示位置の Y 座標をドット単位で設定します。(0～1919)
WIDTH	ウィンドウの幅をドット単位で設定します。(1～2560)
HEIGHT	ウィンドウの高さをドット単位で設定します。(1～1920)
背景色	背景色を設定します。
背景画	背景画を使用する場合は「あり」、使用しない場合は「なし」を選択します。

項 目	内 容
背景イメージ	背景画として使用するイメージリソースの ID を選択します。
タイトル文字列	ウィンドウのタイトルバーに表示する文字列を、文字列リソースから選択、または、新規の文字列を入力します。
タイトルバーの有無	ウィンドウの表示において、タイトルバーをつけて表示する場合は、「あり」、表示しない場合は「なし」を選択します。
クローズボタンの有無	ウィンドウの表示において、タイトルバーにクローズボタンを表示する場合は、「あり」、表示しない場合は「なし」を選択します。
ウィンドウ枠の有無	ウィンドウの表示において、ウィンドウ枠を表示する場合は、「あり」、表示しない場合は「なし」を選択します。
ブリンク OFF 時間	コントロールのブリンク OFF 状態の表示(通常の表示)時間を ms 単位で設定します。(100～60000)
ブリンク ON 時間	コントロールのブリンク ON 状態の表示時間を ms 単位で設定します。(100～60000)
OnCreate	表示パネル上にあるコントロールの Create 関数の処理を行った後に呼び出されるコールバックです。
PreDelete	表示パネル上にあるコントロールの Delete 関数の処理を行う前に呼び出されるコールバックです。
PreKeyPress	表示パネル上にあるコントロールの KeyPress 関数の前に呼び出され、処理を追加するためのコールバックです。戻り値によりコントロールの KeyPress 関数の実行有無を制御します。
PreKeyRelease	表示パネル上にあるコントロールの KeyRelease 関数の前に呼び出され、処理を追加するためのコールバックです。戻り値によりコントロールの KeyRelease 関数の実行有無を制御します。
OnUser	パネルへユーザイベントが呼ばれた後、ユーザ独自の処理を追加するためのコールバックです。
PrePress	表示パネル上にあるコントロールの LButtonPress 関数の前に呼び出され、処理を追加するためのコールバックです。戻り値によりコントロールの LButtonPress 関数実行の有無を制御します。
PreRelease	表示パネル上にあるコントロールの LButtonRelease 関数の前に呼び出され、処理を追加するためのコールバックです。戻り値によりコントロールの LButtonRelease 関数実行の有無を制御します。
PrePaint	表示パネル上にあるコントロールの Paint 関数の前に呼び出され、処理を追加するためのコールバックです。戻り値によりコントロールの PrePaint 関数実行の有無を制御します。
OnTimer	パネルへタイマイベントが呼ばれた後に実行する処理を追加するためのコールバックです。
OnActive	パネルの表示完了時に呼び出されるコールバックです。
OnUpdateDB	データベース変更通知が送られると呼び出されるコールバックです。
OnMouseMove	パネル上で座標位置が変わるたびに呼び出されるコールバックです。

3. 各項目の設定を行います。

**MEMO**

◆ 作成したパネル/ウィンドウの保存方法は以下のようになります。

1. [ファイル]メニューの[パネル/ウィンドウの保存]またはツールバーの[パネル/ウィンドウの保存]ボタンを選択します。

[ツールバー]



2. 現在表示されているページが保存されます。

## 2-4 コントロールの作成と設定

### コントロールの配置

今回のサンプルアプリケーションでは、パネル/ウィンドウ内にコントロールの配置を行います。

コントロール配置の例として、「手動操作」画面の「扉開」のボタンコントロールの配置方法を説明します。

「扉開」ボタンに必要な設定は以下のとおりです。



[プロパティ設定]

ボタンタイプ、表示タイプ


OFF 時/ON 時意匠、

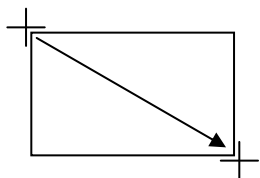
キャプション有無、文字列、

OnClick(コールバック関数)

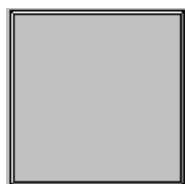
### コントロールの配置



1. コントロールツールバーからボタン  を選択します。
2. 形状が+ (十字型) に変わったマウスポインタを画面の任意の場所へ持っていき、作成したい部品の大きさまでマウスをドラッグします。



3. ボタンが作成されます。





### プロパティ設定

1. 作成したボタンを選択しプロパティウィンドウにボタンのプロパティを表示させます。
2. 「扉開」ボタンのプロパティ設定内容は「付録 1-5 コントロールプロパティ設定」を参照してください。
3. コールバック関数を設定します。

コールバック関数とは「コントロールをクリックしたとき」や「コントロールが表示されたとき」などのイベントが発生したときに実行される関数のことです。CI SKETCH では各イベントごとに実行されるコールバック関数を各コントロールのプロパティにて設定します。コールバック関数が設定された場合は、ソースコード生成時に設定したコールバック関数がソースコードとして生成されます。

OnPress	なし
OnRelease	なし
OnClick	あり
OnDraw	なし
OnTimer	なし

コールバック関数を  
設定してソースコード  
を生成

コールバック関数  
が生成されます

```

GInt32 GPanel_ManualOperation_BTNHIRAKISEKIONEOnClick(GPanel_ManualOperation *pSelf,
    GUInt16 usMessage, GLParam ILParam, GUParam IUPParam)
{
    //[_GUI_SCRIPT_FUNCTION_CALL(GPANEL_MANUALOPERATION_BTNHIRAKISEKIONEOnClick)
    //]_GUI_SCRIPT_FUNCTION_CALL(GPANEL_MANUALOPERATION_BTNHIRAKISEKIONEOnClick)

    GButton *pButton;
    GUInt8 ucStatus;
    unsigned int gUIntDBstr;
    . . .

```

今回のサンプルアプリケーションではマウスクリック時に実デバイスと通信する処理を行うため OnClick の設定を「あり」にします。

同様にしてほかのコントロールも作成してください。

**MEMO**

- ◆ 今回のサンプルアプリケーションのパネルプロパティ設定内容は「付録 1-3 パネル / ウィンドウプロパティ設定」を参照してください。
- ◆ コールバック関数の詳細に関しては「GENWARE3 ユーザーズマニュアル」の「第 26 章 コールバック」を参照してください。

## 2-5 スクリーンプロパティの設定

スクリーンプロパティ([設定]メニューの[スクリーンプロパティ])では、以下のコールバック関数を使用します。

表示する画面によらずに実施する処理を行います。

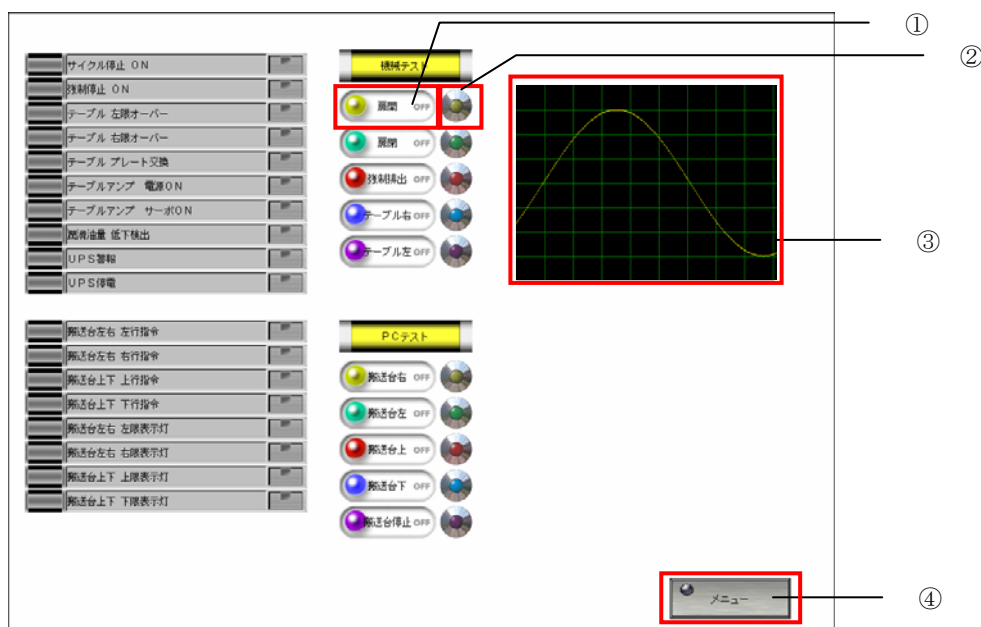
項目	設定	内容
OnTimer	あり	<p>タイマイベント(ID_TIMER_DATA_LOGGING, 60x60x1000ms)により、以下の GUI データベースを使用して実デバイスからデータを読み出します。 読出し後、LoggingDevice.log ファイルに書き込みます。</p> <ul style="list-style-type: none"> <li>•GDB_ELEC_IMPRINT</li> <li>•GDB_VOL_IMPRINT</li> <li>•GDB_ELEC_DELIVERY</li> <li>•GDB_VOL_DELIVERY</li> </ul> <p>本処理は運転データ管理画面において 1 時間ごとにロギングしたデータを棒グラフとして表示するために実施します。</p> <p>タイマイベント(ID_TIMER_FIVE_MINUTE,1000ms)により、連続して5分間、画面の操作がない場合に、ユーザセキュリティ機能のレベル 1～3 をいずれも不許可にする処理を行います。</p> <p>タイマイベント(ID_TIMER_FIVE_MINUTE,1000ms)により、アラームの発生、解除の監視を行うため、以下の GUI データベースを使用して実デバイスからデータを読み出します。</p> <ul style="list-style-type: none"> <li>•GDB_ERR_LIST</li> </ul>
OnUpdateDB	あり	<p>アラームの発生、解除を 1000msec 周期で監視します。</p> <p>値変化がある GUI データベースの内容を読み出しエラー管理画面用のログデータ管理変数に格納を行います。また ErrorHistory.log ファイルへの保存も実施します。</p>

## 2-6 「手動操作画面」の作成

ここでは手動操作画面の作成について、説明します。

基本的なコントロールの作成手順は「2-4 コントロールの作成と設定」と同じですので、本画面の特徴的な部分を説明します。

また、生成したソースコードに対し追加する処理は、「4 章」を参照してください。



### 画面プロパティ設定

本画面のプロパティでは、以下のコールバック関数を使用します。

項目	設定	内容
OnCreate	あり	<p>初期画面表示の為に以下の GUI データベースを使用して実デバイスからデータを読み出します。</p> <ul style="list-style-type: none"> <li>• GDB_SIGNAL_STS1</li> <li>• GDB_SIGNAL_STS2</li> <li>• GDB_MACHINE_TEST2</li> <li>• GDB_PC_TEST2</li> </ul>

項目	設定	内容
OnUpdateDB	あり	以下から値変化がある GUI データベースの内容を読み出し画面に反映を行います。 ・ID_GDB_SIGNAL_STS1 ・ID_GDB_SIGNAL_STS2 ・ID_GDB_MACHINE_TEST2 ・ID_GDB_PC_TEST2

## GUI データベース

本画面では、以下の GUI データベースオブジェクトを使用します。

オブジェクト名	ID	変数型	配列サイズ	内容
GDB_SIGNAL_STS1	ID_GDB_SIGNAL_STS1	USHORT	1	各信号(上段)
GDB_SIGNAL_STS2	ID_GDB_SIGNAL_STS2	USHORT	1	各信号(下段)
GDB_MACHINE_TEST1	ID_GDB_MACHINE_TEST1	USHORT	1	機械テストボタン用
GDB_MACHINE_TEST2	ID_GDB_MACHINE_TEST2	USHORT	1	機械テストランプ用
GDB_PC_TEST1	ID_GDB_PC_TEST1	USHORT	1	PC テストボタン用
GDB_PC_TEST2	ID_GDB_PC_TEST2	USHORT	1	PC テストランプ用

## GDB\_SIGNAL\_STS1 データ構造

実デバイス	内容	
M0	サイクル停止 ON	0:OFF、1:ON
M1	強制停止 ON	0:OFF、1:ON
M2	テーブル 左限オーバー	0:OFF、1:ON
M3	テーブル 右限オーバー	0:OFF、1:ON
M4	テーブル ブレード交換	0:OFF、1:ON
M5	テーブルアンプ 電源 ON	0:OFF、1:ON
M6	テーブルアンプ サーボ ON	0:OFF、1:ON
M7	潤滑油量 低下検出	0:OFF、1:ON
M8	USP 警報	0:OFF、1:ON
M9	USP 停電	0:OFF、1:ON

## GDB\_SIGNAL\_STS2 データ構造

実デバイス	内容	
M16	搬送台左右 左行指令	0:OFF、1:ON
M17	搬送台左右 左行指令	0:OFF、1:ON
M18	搬送台上下 上行指令	0:OFF、1:ON
M19	搬送台上下 下行指令	0:OFF、1:ON
M20	搬送台左右 左限表示灯	0:OFF、1:ON
M21	搬送台左右 右限表示灯	0:OFF、1:ON
M22	搬送台上下 上限表示灯	0:OFF、1:ON
M23	搬送台上下 下限表示灯	0:OFF、1:ON

## GDB\_MACHINE\_TEST1 データ構造

実デバイス	内容	
Y40	扉開	0:OFF、1:ON
Y41	扉閉	0:OFF、1:ON
Y42	強制排出	0:OFF、1:ON
Y43	テーブル右	0:OFF、1:ON
Y44	テーブル左	0:OFF、1:ON

## GDB\_MACHINE\_TEST2 データ構造

実デバイス	内容	
X20	扉開	0:OFF、1:ON
X21	扉閉	0:OFF、1:ON
X22	強制排出	0:OFF、1:ON
X23	テーブル右	0:OFF、1:ON
X24	テーブル左	0:OFF、1:ON

## GDB\_PC\_TEST1 データ構造

実デバイス	内容	
Y50	搬送台右	0:OFF、1:ON
Y51	搬送台左	0:OFF、1:ON
Y52	搬送台上	0:OFF、1:ON
Y53	搬送台下	0:OFF、1:ON
Y54	搬送台停止	0:OFF、1:ON

## GDB\_PC\_TEST2 データ構造

実デバイス	内容	
X30	搬送台右	0:OFF、1:ON
X31	搬送台左	0:OFF、1:ON
X32	搬送台上	0:OFF、1:ON
X33	搬送台下	0:OFF、1:ON
X34	搬送台停止	0:OFF、1:ON

## コントロール

## ①「扉開」スイッチ(ボタンコントロール)

「扉開」スイッチは、以下の動作をするボタンです。

動作	スイッチの ON/OFF により、Y40 を ON/OFF します。オルタネート動作をします。 ユーザセキュリティのレベルが1以上のとき、操作可能です。
----	---

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。

(主な変更点のみ記載)

項目	設定	内容
コントロール名	btnHirakiSekiOne	-
ボタンタイプ	オルタネート	ボタンの動作種別は「ボタンタイプ」で設定します。
表示タイプ	イメージ	ボタンにイメージデータを表示する場合は、「表示タイプ」に「イメージ」を設定します。「ON 時意匠」、「OFF 時意匠」に、表示するイメージリソースの ID を設定します。
ON 時意匠	img_Onoff01_ON	
OFF 時意匠	img_Onoff001_OFF	
文字列	ID_STRING_DoorOpen	表示する文字列リソース ID を設定します。
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

## ②「扉開」ランプ(ピクチャコントロール)

「扉開」ランプは、以下の動作をするピクチャです。

動作	X20 の ON/OFF により、表示するイメージを切換えます。
----	----------------------------------

画面上に、ピクチャコントロールを配置後、次のようにプロパティを設定します。  
(主な変更点のみ記載)

項目	設定	内容
コントロール名	pctHirakiSekiOneLight	-
表示タイプ	イメージ	イメージデータを表示する場合は、「表示タイプ」に「イメージ」を設定します。ON、OFF の 2 つの状態を表示するために、「状態数」を 2 とします。
状態数	2	
状態 0 意匠	img_Onoff_Round001_OFF	
状態 1 意匠	img_Onoff_Round001_OFF	「状態 0 意匠」、「状態 1 意匠」に、表示するイメージリソースの ID を設定します。

## ③折れ線グラフ(基本コントロール)

「扉開」スイッチは、以下の動作をするボタンです。

表示	手動操作画面表示中に、サインカーブを表示します。
----	--------------------------

画面上に、基本コントロールを配置後、次のようにプロパティを設定します。  
(主な変更点のみ記載)

項目	設定	内容
コントロール名	bacCtlSinLine	-
OnTimer	あり	グラフの描画を、画面表示後、周期的に行うため、OnCreate、OnTimer を「あり」に設定します。
OnCreate	あり	



## ④メニュー表示ボタン(ボタンコントロール)

「メニュー表示ボタン」は、以下の動作をするボタンです。

動作	ボタン押下時に、「メニュー画面」を表示します。
----	-------------------------

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。  
(主な変更点のみ記載)

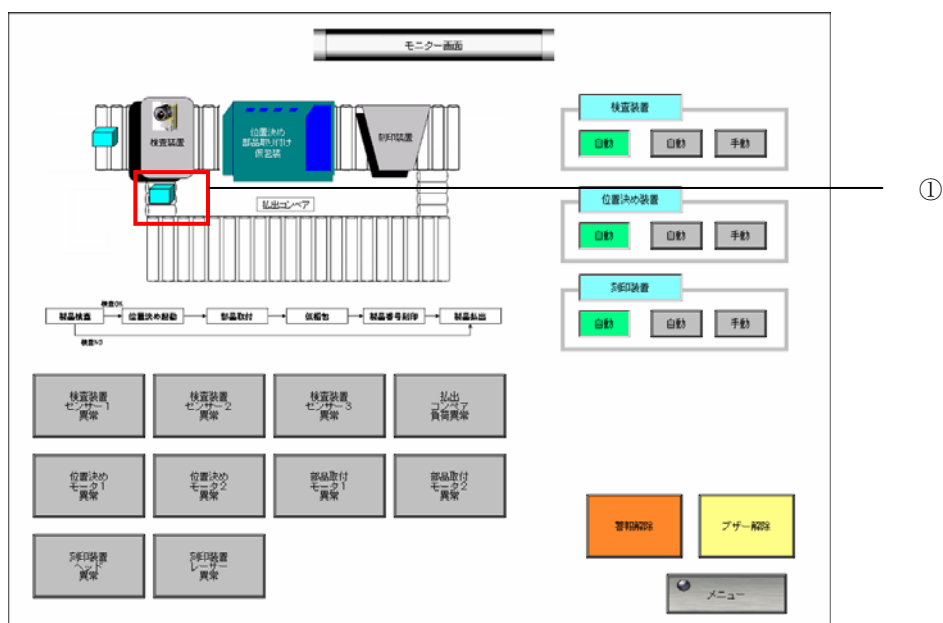
項目	設定	内容
コントロール名	btnManualOperationMenu	-
ボタンタイプ	モーメンタリ	ボタンの動作種別は「ボタンタイプ」で設定します。
表示タイプ	イメージ	ボタンにイメージデータを表示する場合は、「表示タイプ」に「イメージ」を設定します。「ON 時意匠」、「OFF 時意匠」に、表示するイメージリソースの ID を設定します。
ON 時意匠	img_MenuButton_ON	
OFF 時意匠	img_MenuButton_OFF	
文字列	ID_STRING_Menu	表示する文字列リソース ID を設定します。
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

## 2-7 「モニタ画面」の作成

ここではモニタ画面の作成について、説明します。

基本的なコントロールの作成手順は「2-4 コントロールの作成と設定」と同じですの  
で、本画面の特徴的な部分を説明します。

また、生成したソースコードに対し追加する処理は、「4 章」を参照してください。



### 画面プロパティ設定

本画面のプロパティでは、以下のコールバック関数を使用します。

項目	設定	内容
OnCreate	あり	以下の処理を実施します。 ・初期画面表示のために以下の GUI データベースからデータを読み出し画面に反映。 ・アニメーション再生を実行。 ・タイマイイベント(ID 番号 10,1000ms)起動。

項目	設定	内容
OnTimer	あり	タイマイイベント(ID 番号 10,1000ms)を使用し、以下の GUI データベースを使用して実デバイスからデータを読み出します。 ・GDB_ERR_DEVICE_TYPE ・GDB_DEVICE_TYPE ・GDB_WARNING_CLEAR ・GDB_BUZZER_CLEAR
OnUpdateDB	あり	以下から値変化がある GUI データベースの内容を読み出し画面に反映を行います。 ・ID_GDB_ERR_DEVICE_TYPE ・ID_GDB_DEVICE_TYPE

## GUI データベース

本画面では、以下の GUI データベースオブジェクトを使用します。

オブジェクト名	ID	変数型	配列サイズ	内容
GDB_ERR_DEVICE_TYPE	ID_GDB_ERR_DEVICE_TYPE	USHORT	1	異常内容
GDB_DEVICE_TYPE	ID_GDB_DEVICE_TYPE	USHORT	1	装置種類
GDB_WARNING_CLEAR	ID_GDB_WARNING_CLEAR	USHORT	1	警報解除
GDB_BUZZER_CLEAR	ID_GDB_BUZZER_CLEAR	USHORT	1	ブザー
GDB_CHECK_RESULT	ID_GDB_CHECK_RESULT	USHORT	1	検査結果

## GDB\_ERR\_DEVICE\_TYPE データ構造

実デバイス	内容	
M32	検査装置センサー1 異常	0:OFF、1:ON
M33	検査装置センサー2 異常	0:OFF、1:ON
M34	検査装置センサー3 異常	0:OFF、1:ON
M35	払出コンベア異常	0:OFF、1:ON
M36	位置決めモータ1 異常	0:OFF、1:ON
M37	位置決めモータ2 異常	0:OFF、1:ON
M38	部品取付モータ1 異常	0:OFF、1:ON
M39	部品取付モータ2 異常	0:OFF、1:ON
M40	刻印装置ヘッド異常	0:OFF、1:ON
M41	刻印装置レーザー異常	0:OFF、1:ON

## GDB\_DEVICE\_TYPE データ構造

実デバイス	内容	
M48	検査装置	0:自動、1:手動
M49	位置決め装置	0:自動、1:手動
M50	刻印装置	0:自動、1:手動

## GDB\_WARNING\_CLEAR データ構造

実デバイス	内容	
M64	警報解除	0:信号 OFF、1:信号 ON

## GDB\_BUZZER\_CLEAR データ構造

実デバイス	内容	
M80	ブザー解除	0:信号 OFF、1:信号 ON

## GDB\_DEVICE\_TYPE データ構造

実デバイス	内容	
—	製品到着	0:未着、1:到着
—	製品検査 OK	0:検査中、1:OK
—	製品検査 NG	0:検査中、1:NG
—	位置決め起動	0:起動中、1:完了
—	部品取付	0:取付中、1:完了
—	仮梱包	0:梱包中、1:完了
—	製品番号刻印	0:刻印中、1:完了
—	製品払出	0:払出中、1:完了

## コントロール

### ①「ワーク」(ピクチャコントロール)

以下の動作をします。

動作	「検査装置」で NG と判定されたワークが、コンベア上を時間経過とともに移動します。 (アニメーション機能を使用)
----	--

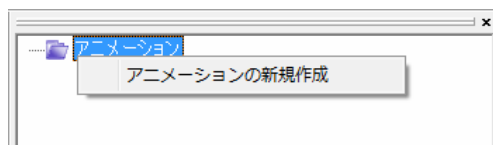
画面上に、ピクチャコントロールを配置後、次のようにプロパティを設定します。  
(主な変更点のみ記載)

項目	設定	内容
コントロール名	pctMoveBody2	-
表示タイプ	イメージ	イメージデータを表示する場合は、「表示タイプ」に「イメージ」を設定します。ON、OFFの2つの状態を表示するために、「状態数」を2とします。
状態数	2	
状態 0 意匠	img_MoveBodyOK	
状態 1 意匠	img_MoveBodyNG	「状態 0 意匠」、「状態 1 意匠」に、表示するイメージリソースの ID を設定します。

次に、ピクチャコントロールを移動させるアニメーションを設定します。

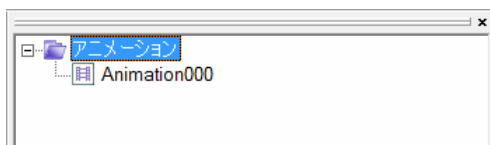
なお、アニメーション機能の詳細については、「GENWARE3-VG ユーザーズマニュアル」(GS-184)の「第 10 章 アニメーション機能」を参照してください。

1. CI SKETCH で[表示]メニューの[アニメーションビュー]を選択します。
2. アニメーションフォルダを選択した状態でマウス右クリックし、表示されるメニューから[アニメーションの新規作成]を選択します。



## GENWARE3 アプリケーション設計ガイド

3. [プロパティウインドウ]でアニメーションのプロパティを以下のように設定します。  
(主な変更点のみ記載)



4. [プロパティウインドウ]でアニメーションのプロパティを以下のように設定します。

アニメーション	
アニメーション名	Animation000
フレーム数	54
時間優先/コマ数優先	時間優先
繰り返し再生	繰り返し
再生回数	2
OnNextFrame	なし
OnEndFrameAnimation	なし

5. [タイムラインビュー]でピクチャコントロール(pctMoveBody2)の移動を設定します。  
pctMoveBody2 の「フレーム 2」をダブルクリックします。

コントロール/ウインドウ		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
軌跡																	
sextPositioningShowChar	<input type="checkbox"/>	*															
stextMarkingShowChange	<input type="checkbox"/>	*															
pctMoveBody2	<input checked="" type="checkbox"/>	*		*													

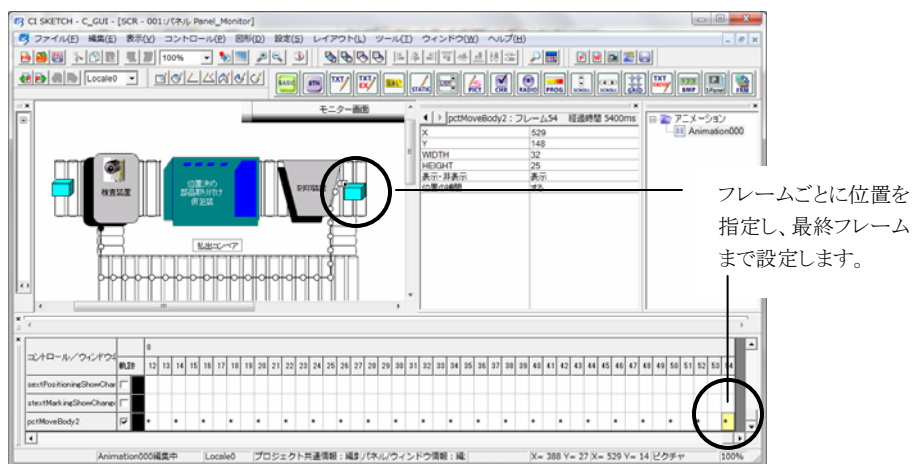
ダブルクリック

プロパティウインドウで、移動先の X 座標、Y 座標を指定します。

pctMoveBody2 : フレーム 2 経過時間 200ms	
X	175
Y	240
WIDTH	32
HEIGHT	25
表示・非表示	表示
位置の補間	する

1 回目の移動場所を指定

6. 各フレームに対し、手順5と同じ操作を行い、pctMoveBody2がコンベアの左端の位置まで移動するように設定します。



## 2-8 「エラー管理画面」の作成

ここではエラー管理画面の作成について、説明します。

基本的なコントロールの作成手順は「2-4 コントロールの作成と設定」と同じですの  
で、本画面の特徴的な部分を説明します。

また、生成したソースコードに対し追加する処理は、「4 章」を参照してください。



### 画面プロパティ設定

本画面のプロパティでは、以下のコールバック関数を使用します。

項目	設定	内容
OnCreate	あり	以下の処理を実施します。 ・現在のエラー一覧を表示更新。 ・現在発生中のエラー内容を表示。
OnUpdateDB	あり	以下から値変化がある GUI データベースの内容を読み出し画面に反映を行います。 ・ID_GDB_ERR_LIST



## GUI データベース

本画面では、以下の GUI データベースオブジェクトを使用します。

オブジェクト名	ID	変数型	配列サイズ	内容
GDB_ERR_LIST	ID_GDB_ERR_LIST	USHORT	50	アラーム

### GDB\_ERR\_LIST データ構造

実デバイス	内容	
D1000	現在値リード通信エラー	0: OFF、1: ON
D1001	湿度計自己診断エラー	0: OFF、1: ON
D1002	ネットワークケーブルエラー	0: OFF、1: ON
D1003	安定化電源異常	0: OFF、1: ON
D1004	通信異常	0: OFF、1: ON
D1005	PLC バッテリ異常	0: OFF、1: ON
D1006	安全 PLC 故障	0: OFF、1: ON
D1007	安全 PLC 通信異常	0: OFF、1: ON
D1008	ファン異常	0: OFF、1: ON
D1009	サイクルタイムオーバー	0: OFF、1: ON
D1010	起動不良異常	0: OFF、1: ON
D1011	回転不良	0: OFF、1: ON
D1012	モーター過負荷	0: OFF、1: ON
D1013	地絡検知	0: OFF、1: ON
D1014	断線検知	0: OFF、1: ON
D1015	シリンダ異常	0: OFF、1: ON
D1016	電圧不足	0: OFF、1: ON
D1017	圧力上限異常	0: OFF、1: ON
D1018	圧力下限異常	0: OFF、1: ON
D1019	ローラー挿入異常	0: OFF、1: ON
D1020	データ無し異常	0: OFF、1: ON
D1021	パラメータ設定不良	0: OFF、1: ON
D1022	コンベア非常停止	0: OFF、1: ON
D1023	非常停止	0: OFF、1: ON
D1024	セーフティドア SW	0: OFF、1: ON
D1025	人体侵入 LC	0: OFF、1: ON
D1026	後退オーバーラン	0: OFF、1: ON
D1027	前進オーバーラン	0: OFF、1: ON
D1028～D1049	Error No.XX	0: OFF、1: ON

## コントロール

### ①「エラー履歴」(スタティックテキストコントロール)

エラー履歴一覧の各行は、スタティックテキストを用いて作成しています。

以下の動作をします。

動作	エラーに割り当てられたデバイスの ON(エラー発生)/OFF(解除)に連動して、エラー一覧を表示します。 エラーレベル、発生日時、エラーコード、異常内容、復旧日時を表示します。 発生中のエラーは赤、解除済みのエラーは青で表示します。
----	--

画面上に、スタティックテキストコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	testErrorList01	-
横位置	左寄せ	-

### ②「スクロールバー」(垂直スクロールバーコントロール)

エラー履歴一覧のスクロールバーは、垂直スクロールバーコントロールを用いて作成しています。

以下の動作をします。

動作	エラー履歴一覧の各行に表示された内容をスクロールさせます。
----	-------------------------------

画面上に、垂直スクロールバーコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	scrollBarError	-
スクロール最大数	49	-
1 ページサイズ	15	-
OnScroll	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はスクロールを行うときに実行するため、「OnScroll」を「あり」とします。

## ③「ソートボタン(OLD→NEW)」(ボタンコントロール)

ソートボタン(OLD→NEW)は、ボタンコントロールを用いて作成しています。  
以下の動作をします。

動作	エラー履歴一覧を古い順に表示します。
----	--------------------

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。  
(主な変更点のみ記載)

項目	設定	内容
コントロール名	btnOldToNew	-
ボタンタイプ	モーメンタリ	ボタンの動作種別は「ボタンタイプ」で設定します。
表示タイプ	イメージ	ボタンにイメージデータを表示する場合は、「表示タイプ」に「イメージ」を設定します。「ON 時意匠」、「OFF 時意匠」に、表示するイメージリソースの ID を設定します。ここでは同じイメージデータを設定しています。
ON 時意匠	img_errorOrderBotton	
OFF 時意匠	img_errorOrderBotton	
文字列	ID_STRING_OrderOldToNew	表示する文字列リソース ID を設定します。
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

## ④「現在発生中エラー」(スタティックテキストコントロール)

現在発生中エラーは、スタティックテキストを用いて作成しています。  
以下の動作をします。

動作	現在発生中のエラー情報を表示します。 複数エラーが発生している場合は、その中で最新のエラーを 1 つ表示します。 そのエラーが復旧した場合は、その次に新しいエラーを表示します。
----	--

画面上に、スタティックテキストコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	textErrorInfo	-
背景色	RGB=(128,255,255)	-
横位置	左寄せ	-

### ⑤「履歴クリアボタン」(ボタンコントロール)

履歴クリアボタンは、スタティックテキストを用いて作成しています。

以下の動作をします。

動作	エラー履歴の情報をクリアします。 ユーザセキュリティのレベルが1以上のとき、操作可能です。
----	--

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	btnHistoryClear	-
ボタンタイプ	モーメンタリ	ボタンの動作種別は「ボタンタイプ」で設定します。
表示タイプ	イメージ	ボタンにイメージデータを表示する場合は、「表示タイプ」に「イメージ」を設定します。「ON 時意匠」、「OFF 時意匠」に、表示するイメージリソースの ID を設定します。
ON 時意匠	img_MenuButton_ON	
OFF 時意匠	img_MenuButton_OFF	
文字列	ID_STRING_HistoryClear	表示する文字列リソース ID を設定します。
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

## アラーム登録

アラーム登録は CSV ファイル「ErrorInfo.csv」で行います。

以下の情報を CSV ファイルに記載してください。

設定例

0	1	32FA	161	
1	3	10C1	162	
2	1	10C2	163	

番号	項目	内容
①	変数インデックス	アラームに対応した変数の配列番号を指定します。
②	エラーレベル	1～99。レベル 1 を一番高いレベルとします。
③	エラーコード	4 桁の 16 進数値を設定します。
④	文字列リソース ID	CI SKETCH で登録した文字列リソースで、「異常内容」に表示する文字列を指定します。

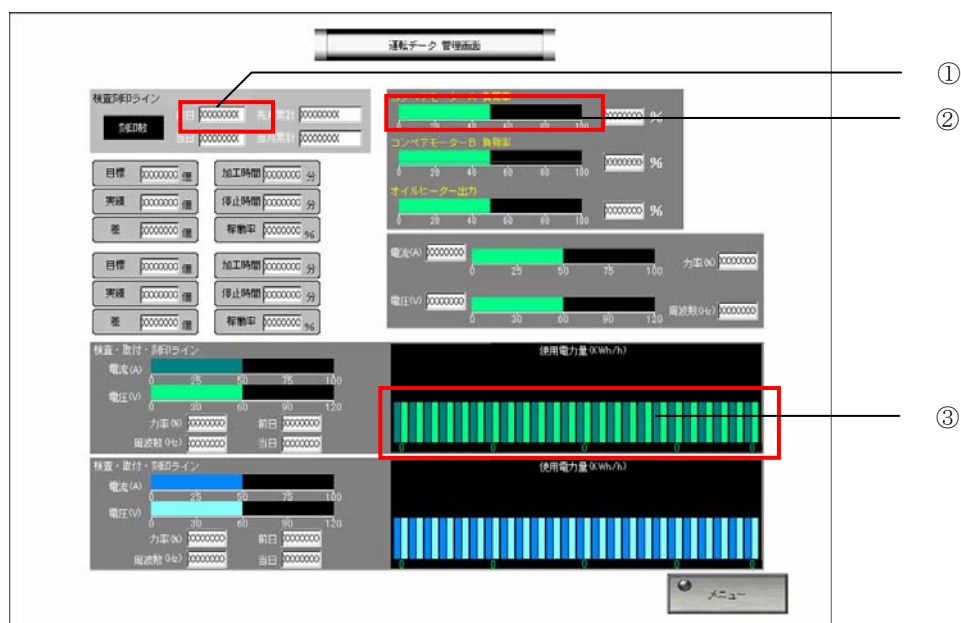
作成した ErrorInfo.csv は、C 言語コントローラの ROM フォルダに格納します。

## 2-9 「運転データ画面」の作成

ここでは運転データ画面の作成について、説明します。

基本的なコントロールの作成手順は「2-4 コントロールの作成と設定」と同じですので、本画面の特徴的な部分を説明します。

また、生成したソースコードに対し追加する処理は、「4 章」を参照してください。



### 画面プロパティ設定

本画面のプロパティでは、以下のコールバック関数を使用します。

項目	設定	内容
OnCreate	あり	以下の処理を実施します。 ・運転データ管理画面用 GUI データベースを使用して実デバイスからデータを読み出します。 ・読み出したデータを元に画面表示。 ・タイマイイベント (ID_TIMER_REFRESH_LOGGING_DATA、60x60x1000msec) 起動。

## GENWARE3 アプリケーション設計ガイド

PreDelete	あり	タイマイイベント (ID_TIMER_REFRESH_LOGGING_DATA) を停止します。
OnTimer	あり	タイマイイベント (ID_TIMER_REFRESH_LOGGING_DATA、60x60x1000msec) により、以下の処理を行います。 ・運転データ管理画面用 GUI データベースを使用して実デバイスからデータを読み出します。 ・ロギングデータの更新処理を実施します。
OnUpdateDB	あり	値変化がある GUI データベースの内容を読み出し画面に反映を行います。

## GUI データベース

本画面では、以下の GUI データベースオブジェクトを使用します。

オブジェクト名	ID	変数型	配列サイズ	内容
GDB_CONVEYOR_A	ID_GDB_CONVEYOR_A	USHORT	1	コンベアモーターA 負荷率
GDB_CONVEYOR_B	ID_GDB_CONVEYOR_B	USHORT	1	コンベアモーターB 負荷率
GDB_OIL_HEATER	ID_GDB_OIL_HEATER	USHORT	1	オイルヒーター出力
GDB_ELEC	ID_GDB_ELEC	USHORT	1	電流
GDB_VOL	ID_GDB_VOL	USHORT	1	電圧
GDB_POWER	ID_GDB_POWER	USHORT	1	力率
GDB_CYCLE	ID_GDB_CYCLE	USHORT	1	周波数
GDB_ELEC_IMPRINT	ID_GDB_ELEC_IMPRINT	USHORT	1	電流 (刻印ライン)
GDB_VOL_IMPRINT	ID_GDB_VOL_IMPRINT	USHORT	1	電圧 (刻印ライン)
GDB_POWER_IMPRINT	ID_GDB_POWER_IMPRINT	USHORT	1	力率 (刻印ライン)
GDB_CYCLE_IMPRINT	ID_GDB_CYCLE_IMPRINT	USHORT	1	周波数 (刻印ライン)
GDB_YESTER_IMPRINT	ID_GDB_YESTER_IMPRINT	USHORT	1	先日 (刻印ライン)
GDB_TODAY_IMPRINT	ID_GDB_TODAY_IMPRINT	USHORT	1	当日 (刻印ライン)
GDB_ELEC_DELIVERY	ID_GDB_ELEC_IMPRINT	USHORT	1	電流 (払い出しライン)
GDB_VOL_DELIVERY	ID_GDB_VOL_DELIVERY	USHORT	1	電圧 (払い出しライン)
GDB_POWER_DELIVERY	ID_GDB_POWER_DELIVERY	USHORT	1	力率 (払い出しライン)

## GENWARE3 アプリケーション設計ガイド

GDB_CYCLE_DELIVERY	ID_GDB_CYCLE_DELIVERY	USHORT	1	周波数(払い出しライン)
GDB_YESTER_DELIVERY	ID_GDB_YESTER_DELIVERY	USHORT	1	先日(払い出しライン)
GDB_TODAY_DELIVERY	ID_GDB_TODAY_DELIVERY	USHORT	1	当日(払い出しライン)
GDB_YESTER_IMPRINT	ID_GDB_YESTER_DELIVERY	USHORT	1	先日(検査刻印ライン)
GDB_TODAY_IMPRINT	ID_GDB_TODAY_DELIVERY	USHORT	1	当日(検査刻印ライン)
GDB_TOTAL_PREMONTH	ID_GDB_TOTAL_PREMONTH	USHORT	1	先月累計(検査刻印ライン)
GDB_TOTAL_CURMONTH	ID_GDB_TOTAL_CURMONTH	USHORT	1	当月累計(検査刻印ライン)
GDB_TARGET_UP	ID_GDB_TARGET_UP	USHORT	1	目標(上図)
GDB_PRACTICAL_UP	ID_GDB_PRACTICAL_UP	USHORT	1	実績(上図)
GDB_WORK_TIME_UP	ID_GDB_WORK_TIME_UP	USHORT	1	加工時間(上図)
GDB_STOP_TIME_UP	ID_GDB_STOP_TIME_UP	USHORT	1	停止時間(上図)
GDB_TARGET_DOWN	ID_GDB_TARGET_DOWN	USHORT	1	目標(下図)
GDB_PRACTICAL_DOWN	ID_GDB_PRACTICAL_DOWN	USHORT	1	実績(下図)
GDB_WORK_TIME_DOWN	ID_GDB_WORK_TIME_DOWN	USHORT	1	加工時間(下図)
GDB_STOP_TIME_DOWN	ID_GDB_STOP_TIME_DOWN	USHORT	1	停止時間(下図)
GDB_DIFF_UP	ID_GDB_DIFF_UP	USHORT	1	差(上図)
GDB_RATIO_UP	ID_GDB_RATIO_UP	USHORT	1	稼働率(上図)
GDB_DIFF_DOWN	ID_GDB_DIFF_DOWN	USHORT	1	差(下図)
GDB_RATIO_DOWN	ID_GDB_RATIO_DOWN	USHORT	1	稼働率(下図)

## 各データ構造

オブジェクト名	ID	実デバイス	内容
GDB_CONVEYOR_A	ID_GDB_CONVEYOR_A	D2000	コンベアモーターA 負荷率
GDB_CONVEYOR_B	ID_GDB_CONVEYOR_B	D2001	コンベアモーターB 負荷率
GDB_OIL_HEATER	ID_GDB_OIL_HEATER	D2002	オイルヒーター出力
GDB_ELEC	ID_GDB_ELEC	D2003	電流
GDB_VOL	ID_GDB_VOL	D2004	電圧



GDB_POWER	ID_GDB_POWER	D2005	力率
GDB_CYCLE	ID_GDB_CYCLE	D2006	周波数
GDB_ELEC_IMPRINT	ID_GDB_ELEC_IMPRINT	D2007	電流(刻印ライン)
GDB_VOL_IMPRINT	ID_GDB_VOL_IMPRINT	D2008	電圧(刻印ライン)
GDB_POWER_IMPRINT	ID_GDB_POWER_IMPRINT	D2009	力率(刻印ライン)
GDB_CYCLE_IMPRINT	ID_GDB_CYCLE_IMPRINT	D2010	周波数(刻印ライン)
GDB_YESTER_IMPRINT	ID_GDB_YESTER_IMPRINT	D2011	先日(刻印ライン)
GDB_TODAY_IMPRINT	ID_GDB_TODAY_IMPRINT	D2012	当日(刻印ライン)
GDB_ELEC_DELIVERY	ID_GDB_ELEC_IMPRINT	D2013	電流(払い出しライン)
GDB_VOL_DELIVERY	ID_GDB_VOL_DELIVERY	D2014	電圧(払い出しライン)
GDB_POWER_DELIVERY	ID_GDB_POWER_DELIVERY	D2015	力率(払い出しライン)
GDB_CYCLE_DELIVERY	ID_GDB_CYCLE_DELIVERY	D2016	周波数(払い出しライン)
GDB_YESTER_DELIVERY	ID_GDB_YESTER_DELIVERY	D2017	先日(払い出しライン)
GDB_TODAY_DELIVERY	ID_GDB_TODAY_DELIVERY	D2018	当日(払い出しライン)
GDB_YESTER_IMPRINT	ID_GDB_YESTER_DELIVERY	D2019	先日(検査刻印ライン)
GDB_TODAY_IMPRINT	ID_GDB_TODAY_DELIVERY	D2020	当日(検査刻印ライン)
GDB_TOTAL_PREMONTH	ID_GDB_TOTAL_PREMONTH	D2021	先月累計(検査刻印ライン)
GDB_TOTAL_CURMONTH	ID_GDB_TOTAL_CURMONTH	D2022	当月累計(検査刻印ライン)
GDB_TARGET_UP	ID_GDB_TARGET_UP	D2023	目標(上図)
GDB_PRACTICAL_UP	ID_GDB_PRACTICAL_UP	D2024	実績(上図)
GDB_WORK_TIME_UP	ID_GDB_WORK_TIME_UP	D2025	加工時間(上図)
GDB_STOP_TIME_UP	ID_GDB_STOP_TIME_UP	D2026	停止時間(上図)
GDB_TARGET_DOWN	ID_GDB_TARGET_DOWN	D2027	目標(下図)
GDB_PRACTICAL_DOWN	ID_GDB_PRACTICAL_DOWN	D2028	実績(下図)
GDB_WORK_TIME_DOWN	ID_GDB_WORK_TIME_DOWN	D2029	加工時間(下図)
GDB_STOP_TIME_DOWN	ID_GDB_STOP_TIME_DOWN	D2030	停止時間(下図)
GDB_DIFF_UP	ID_GDB_DIFF_UP	D2031	差(上図)
GDB_RATIO_UP	ID_GDB_RATIO_UP	D2032	稼働率(上図)
GDB_DIFF_DOWN	ID_GDB_DIFF_DOWN	D2033	差(下図)
GDB_RATIO_DOWN	ID_GDB_RATIO_DOWN	D2034	稼働率(下図)

## コントロール

### ①「刻印数 前日」(テキストボックスコントロール)

刻印数を示す数値表示は、テキストボックスを用いて作成しています。

以下の動作をします。

動作	D2019 の値を、符号無し 10 進数として表示します。
----	-------------------------------

画面上に、テキストボックスコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	textDispensingDeliveryYester	-
背景色	RGB(255,255,255)	-
タイプ	GUInt16	ここでは符号なし 16 ビット変数を、10 進数で表示するために、「GUInt16」、「%d」を設定します。
表示フォーマット	%d	
最大文字数	8	-
フォント	ID_FONT00000	-

## ②「コンペアモーターA 負荷率」(プログレスバーコントロール)

コンペアモーターA 負荷率を示す棒グラフは、プログレスバーコントロールを用いて作成しています。

以下の動作をします。

動作	D2000 の値(0~100)を、棒グラフで表示します。
----	------------------------------

画面上に、プログレスバーコントロールを配置後、次のようにプロパティを設定します。  
(主な変更点のみ記載)

項目	設定	内容
コントロール名	progBarConveyor1	-
本体 背景色	RGB(0,0,0)	棒グラフの背景色を黒、塗りつぶし色を緑に設定します。
本体 前景色	RGB(0,0,0)	
バー 背景色	RGB(0,255,128)	
バー 前景色	RGB(0,255,128)	
方向	左→右	塗りつぶしの方向を設定します。
最小値	0	棒グラフで表示する最小値、最大値を設定します。
最大値	100	

## データログ登録

アラーム登録は CSV ファイル「LoggingDevice.csv」で行います。

以下の情報を CSV ファイルに記載してください。(1 列のみ)

設定例

9	①
10	
15	
16	

番号	項目	内容
①	変数インデックス	ロギング対象のデバイスに対応した変数の配列番号を指定します。 9:ID_GDB_ELEC_IMPRINT 10:ID_GDB_VOL_IMPRINT 15:ID_GDB_ELEC_DELIVERY 16:ID_GDB_VOL_DELIVERY

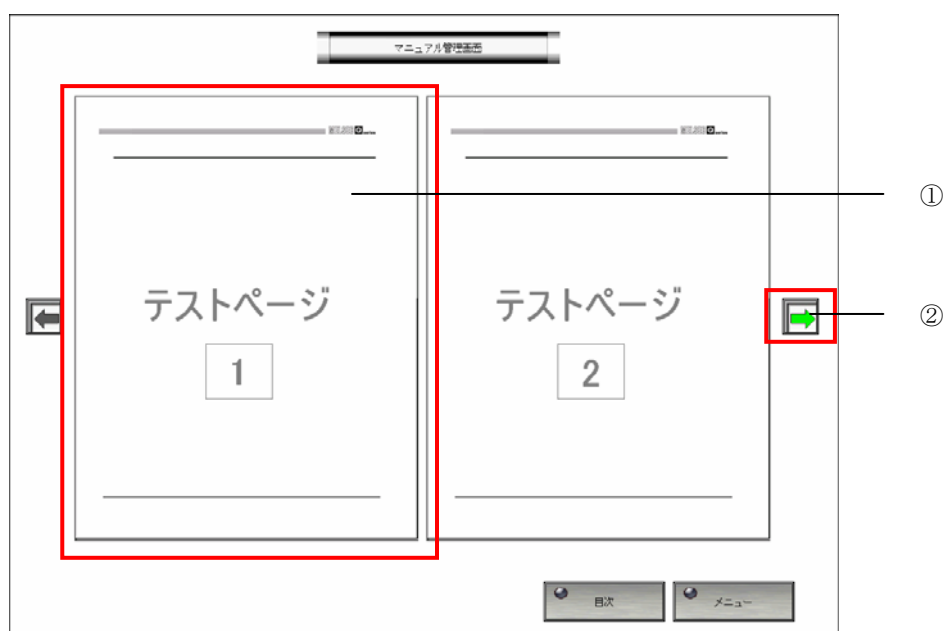
作成した ErrorInfo.csv は、C 言語コントローラの ROM フォルダに格納します。

## 2-10 「マニュアル管理画面」の作成

ここではマニュアル管理画面の作成について、説明します。

基本的なコントロールの作成手順は「2-4 コントロールの作成と設定」と同じですの  
で、本画面の特徴的な部分を説明します。

また、生成したソースコードに対し追加する処理は、「4 章」を参照してください。



### 画面プロパティ設定

本画面のプロパティでは、以下のコールバック関数を使用します。

項目	設定	内容
OnCreate	あり	画面の初期表示に使用します。

## ①「マニュアル表示(左)」(ピクチャコントロール)

マニュアル表示は、ピクチャコントロールを用いて作成しています。

以下の動作をします。

動作	マニュアル(bmp データ)の 1～9 ページ目までを表示します。 「前ページ」、「次ページ」ボタンの押下に合わせて、表示するページが切り換わり ます。
----	--

画面上に、ピクチャコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	GPic_Left	-
状態の数	9	1～9 ページを表示します。
状態 0 意匠	img_Page01	1～9 ページに対応する画像をイメージリソースとして登録します。 本コントロールには、そのイメージリソースを設定します。
状態 1 意匠	img_Page02	
状態 2 意匠	img_Page03	
状態 3 意匠	img_Page04	
状態 4 意匠	img_Page05	
状態 5 意匠	img_Page06	
状態 6 意匠	img_Page07	
状態 7 意匠	img_Page08	
状態 8 意匠	img_Page09	

## ②「次ページボタン」(ボタンコントロール)

次ページボタンは、ボタンコントロールを用いて作成しています。

以下の動作をします。

動作	本ボタンをクリックすることで、マニュアル表示で次のページを表示します。 「マニュアル表示(右)」が 10 ページ目を表示しているときには、本ボタンをクリックしても無処理となります。
----	---

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

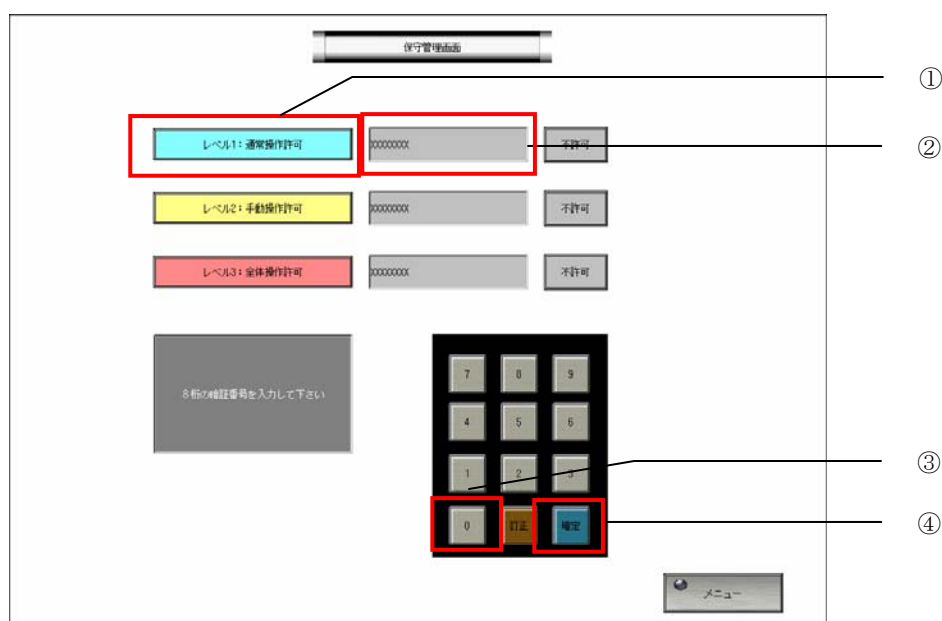
項目	設定	内容
コントロール名	GButton_Right	－
ボタンタイプ	モーメンタリ	ボタンの動作種別は「ボタンタイプ」で設定します。
表示タイプ	イメージ	ボタンにイメージデータを表示する場合は、「表示タイプ」に「イメージ」を設定します。「ON 時意匠」、「OFF 時意匠」に、表示するイメージソースの ID を設定します。
ON 時意匠	img_RightMove	
OFF 時意匠	img_RightMove	
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

## 2-11 「保守管理画面」の作成

ここでは保守管理画面の作成について、説明します。

基本的なコントロールの作成手順は「2-4 コントロールの作成と設定」と同じですの  
で、本画面の特徴的な部分を説明します。

また、生成したソースコードに対し追加する処理は、「4 章」を参照してください。



## 画面プロパティ設定

本画面のプロパティでは、以下のコールバック関数を使用します。

項目	設定	内容
OnCreate	あり	以下の処理を実施します。 ・パスワード登録ファイル (PasswordInfo.csv) から各階層のパスワードの読み出し ・画面の初期表示



## ①「レベル1 ログインボタン」(ボタンコントロール)

レベル1 ログインボタンは、ボタンコントロールを用いて作成しています。  
以下の動作をします。

動作	ボタンをクリックすると、パスワード入力欄(②)にフォーカスに移ります。
----	-------------------------------------

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	btnMessageLevel1	-
ボタンタイプ	モーメンタリ	ボタンの動作種別は「ボタンタイプ」で設定します。
表示タイプ	矩形	ボタンをシンプルな矩形で表示する場合は、「表示タイプ」に「矩形」を設定します。「ON 時背景色」、「OFF 時背景色」に、表示する色を設定します。
ON 時 背景色	RGB(128,255,255)	
OFF 時 背景色	RGB(128,255,255)	
文字列	ID_STRING_MaintenanceLevelMessage1	表示する文字列リソース ID を設定します。
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

## ②「レベル1 パスワード入力欄」(テキストボックスコントロール)

レベル1 パスワード入力欄は、テキストボックスコントロールを用いて作成しています。  
以下の動作をします。

動作	テンキーボタン(③)から、パスワードの入力を受け付けます。 入力された文字は「*」で表示されます。
----	--

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	textPasswordLevel1	–
タイプ	文字列	ボタンをシンプルな矩形で表示する場合は、「表示タイプ」に「矩形」を設定します。「ON 時背景色」、「OFF 時背景色」に、表示する色を設定します。
表示フォーマット	%s	
最大文字数	8	
パスワード設定	あり	「あり」を設定すると、入力された文字列が「*」で表示されます。

### ③「0 ボタン」(ボタンコントロール)

0 ボタンは、ボタンコントロールを用いて作成しています。

以下の動作をします。

動作	パスワード入力欄に、文字列 0 を入力します。
----	-------------------------

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	btnNum0	–
ボタンタイプ	モーメンタリ	ボタンの動作種別は「ボタンタイプ」で設定します。
表示タイプ	イメージ	ボタンにイメージデータを表示する場合は、「表示タイプ」に「イメージ」を設定します。「ON 時意匠」、「OFF 時意匠」に、表示するイメージリソースの ID を設定します。
ON 時 意匠	img_NumButton_ON	
ON 時 意匠	img_NumButton_OFF	
文字列	ID_STRING_MaintenanceNum0	表示する文字列リソース ID を設定します。
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

## ④「確定ボタン」(ボタンコントロール)

確定ボタンは、ボタンコントロールを用いて作成しています。

以下の動作をします。

動作	パスワード入力欄に入力された文字列が正しいパスワードかどうかを判定します。 正しいパスワードが入力されていた場合は、そのレベルでログインします。 不正なパスワードの場合は、パスワードが違うことを告げるメッセージを表示します。3 回続けて不正なパスワードを入力した場合には、3 回続けて間違えたことを告げるメッ セージを表示します。
----	---

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。(主な  
変更点のみ記載)

項目	設定	内容
コントロール名	btnNumOk	—
ボタンタイプ	モーメンタリ	ボタンの動作種別は「ボタンタイプ」で設定しま す。
表示タイプ	イメージ	ボタンにイメージデータを表示する場合は、「表 示タイプ」に「イメージ」を設定します。「ON 時意 匠」、「OFF 時意匠」に、表示するイメージリソ ースの ID を設定します。
ON 時 意匠	img_ConfigButton_ON	
ON 時 意匠	img_ConfigButton_OFF	
文字列	ID_STRING_Maintenance Config	表示する文字列リソース ID を設定します。
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成 するソースコードに追記します。その処理はボ タンをクリックしたときに実行するため、 「OnClick」を「あり」とします。

## パスワードの設定

パスワードは CSV ファイル「PasswordInfo.csv」で行います。

以下の情報を CSV ファイルに記載してください。(1 列のみ)

設定例

11111111	①
22222222	
33333333	

番号	項目	内容
①	パスワード	パスワードとして設定する文字列を入力します。

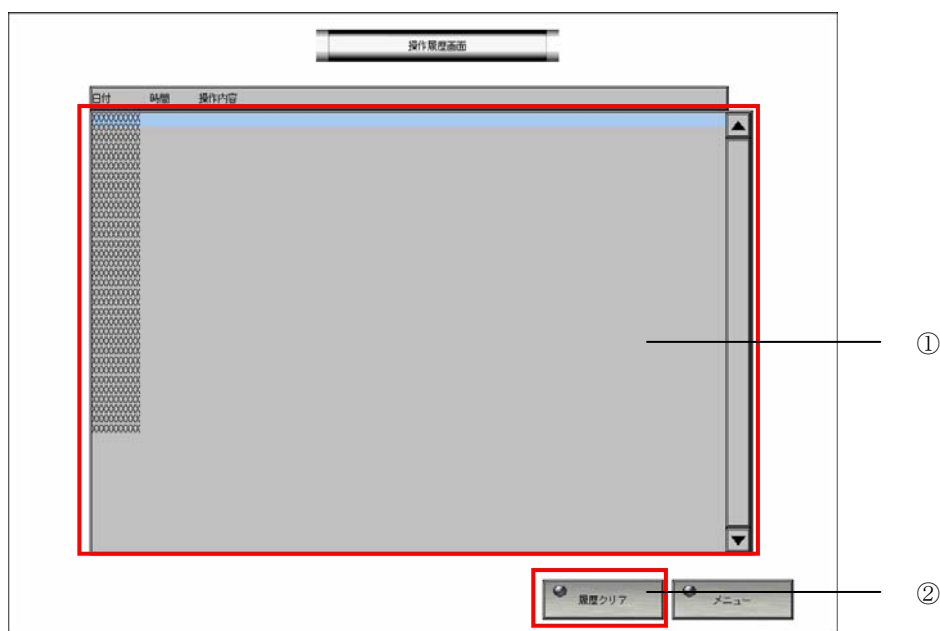
作成した PasswordInfo.csv は、C 言語コントローラの ROM フォルダに格納します。

## 2-12 「操作履歴画面」の作成

ここでは操作履歴画面の作成について、説明します。

基本的なコントロールの作成手順は「2-4 コントロールの作成と設定」と同じですので、本画面の特徴的な部分を説明します。

また、生成したソースコードに対し追加する処理は、「4 章」を参照してください。



### 画面プロパティ設定

本画面のプロパティでは、以下のコールバック関数を使用します。

項目	設定	内容
OnCreate	あり	以下の処理を実施します。 ・操作履歴ログの内容を一覧表示

## ①「履歴一覧」(リストコントロール)

履歴一覧は、リストコントロールを用いて作成しています。

以下の動作をします。

動作	操作履歴を一覧表示します。
----	---------------

画面上に、リストコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	listOperationHistoryInfo	-
最大行数	100	操作履歴は 100 件まで保存されるので、最大数を 100 とします。

## ②「履歴クリアボタン」(ボタンコントロール)

履歴クリアボタンは、ボタンコントロールを用いて作成しています。

以下の動作をします。

動作	ボタンをクリックすると、それまで記録されていた操作履歴をクリアします。 ユーザセキュリティのレベルが1以上のとき、操作可能です。
----	---

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	btnHistoryClear	-
ボタンタイプ	モーメンタリ	ボタンの動作種別は「ボタンタイプ」で設定します。
表示タイプ	イメージ	ボタンにイメージデータを表示する場合は、「表示タイプ」に「イメージ」を設定します。「ON 時意匠」、「OFF 時意匠」に、表示するイメージリソースの ID を設定します。
ON 時 意匠	img_MenuButton_ON	
ON 時 意匠	img_MenuButton_OFF	
文字列	ID.STRING_HistoryClear	表示する文字列リソース ID を設定します。
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

## 操作履歴

本アプリケーションでは、手動操作画面の各ボタンの操作を、操作履歴として記録します。

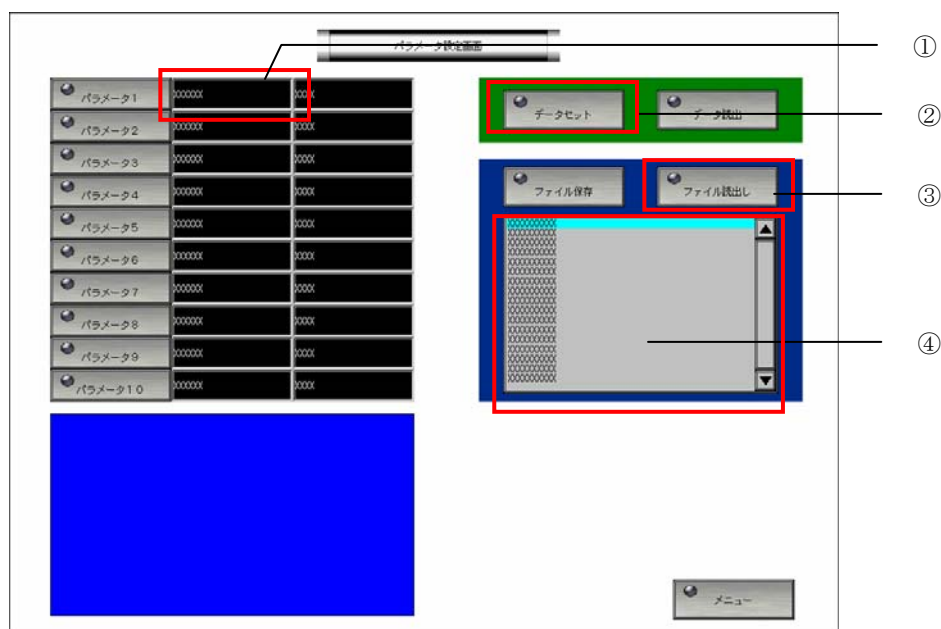
履歴一覧に表示する文字列は、ボタン操作時(OnClick)に、銘板文字列を取得し表示するよう、コールバック関数の中で処理が記述されています。

## 2-13 「パラメータ設定画面」の作成

ここではパラメータ設定画面の作成について、説明します。

基本的なコントロールの作成手順は「2-4 コントロールの作成と設定」と同じですので、本画面の特徴的な部分を説明します。

また、生成したソースコードに対し追加する処理は、「4 章」を参照してください。



### 画面プロパティ設定

本画面のプロパティでは、以下のコールバック関数を使用します。

項目	設定	内容
OnCreate	あり	以下の処理を実施します。 ・ファイル一覧に、/ROM フォルダにあるファイル名”Recipe*.*”を検索し、該当するファイル名を表示 ディレクトリが存在する場合は<ディレクトリ名>として表示 ・パラメータ 1～10 の表示エリアに初期値 0 を表示



## GUI データベース

本画面では、以下の GUI データベースオブジェクトを使用します。

オブジェクト名	ID	変数型	配列サイズ	内容
GDB_DECIMAL	ID_GDB_DECIMAL	USHORT	10	パラメータ第一列データ
GDB_HEXADECIMAL	ID_GDB_HEXADECIMAL	USHORT	10	パラメータ第二列データ

### GDB\_DECIMAL データ構造

実デバイス	配列番号	内容
D0	0	パラメータ 1 第一列データ
D1	1	パラメータ 2 第一列データ
D2	2	パラメータ 3 第一列データ
D3	3	パラメータ 4 第一列データ
D4	4	パラメータ 5 第一列データ
D5	5	パラメータ 6 第一列データ
D6	6	パラメータ 7 第一列データ
D7	7	パラメータ 8 第一列データ
D8	8	パラメータ 9 第一列データ
D9	9	パラメータ 10 第一列データ

### GDB\_DECIMAL データ構造

実デバイス	配列番号	内容
D10	0	パラメータ 1 第二列データ
D11	1	パラメータ 2 第二列データ
D12	2	パラメータ 3 第二列データ
D13	3	パラメータ 4 第二列データ
D14	4	パラメータ 5 第二列データ
D15	5	パラメータ 6 第二列データ
D16	6	パラメータ 7 第二列データ
D17	7	パラメータ 8 第二列データ
D18	8	パラメータ 9 第二列データ
D19	9	パラメータ 10 第二列データ

## ①「パラメータ1」(テキストボックスコントロール)

パラメータ1は、テキストボックスリストコントロールを用いて作成しています。  
以下の動作をします。

動作	入力欄をクリックすると、テンキー画面(ウィンドウ画面: Window_NumericKeypad)を表示し、テンキー画面からの入力を受け付けます。 ユーザセキュリティのレベルが3のとき、操作可能です。
----	---

画面上に、テキストボックスコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	tBoxDecimal01	–
タイプ	文字列	テンキー画面から4文字まで、入力を受け付けます。
表示フォーマット	%s	
最大文字数	4	
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

## ②「データセットボタン」(ボタンコントロール)

データセットボタンは、ボタンコントロールを用いて作成しています。  
以下の動作をします。

動作	ボタンをクリックすると、パラメータ1～10に設定されたデータを、D0～D19に書き込みます。 ユーザセキュリティのレベルが3以上のとき、操作可能です。
----	--

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	btnDataSet	-
ボタンタイプ	モーメンタリ	ボタンの動作種別は「ボタンタイプ」で設定します。
表示タイプ	イメージ	ボタンにイメージデータを表示する場合は、「表示タイプ」に「イメージ」を設定します。「ON 時意匠」、「OFF 時意匠」に、表示するイメージリソースの ID を設定します。
ON 時 意匠	img_MenuButton_ON	
ON 時 意匠	img_MenuButton_OFF	
文字列	ID_STRING_DataSet	表示する文字列リソース ID を設定します。
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

### ③「ファイル読出しボタン」(ボタンコントロール)

ファイル読出しボタンは、ボタンコントロールを用いて作成しています。

以下の動作をします。

動作	ボタンをクリックすると、ファイル一覧で選択した CSV ファイルの内容を読み出し、パラメータ 1～10 のテキストボックスにデータをセットします。
----	---

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	btnLoad	-
ボタンタイプ	モーメンタリ	ボタンの動作種別は「ボタンタイプ」で設定します。
表示タイプ	イメージ	ボタンにイメージデータを表示する場合は、「表示タイプ」に「イメージ」を設定します。「ON 時意匠」、「OFF 時意匠」に、表示するイメージリソースの ID を設定します。
ON 時 意匠	img_MenuButton_ON	
ON 時 意匠	img_MenuButton_OFF	
文字列	ID_STRING_FileLoad	表示する文字列リソース ID を設定します。
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

#### ④「ファイル一覧」(リストコントロール)

ファイル一覧は、リストコントロールを用いて作成しています。

以下の動作をします。

動作	C 言語コントローラ内に保存されたレシピファイルおよびフォルダを一覧表示します。 ファイルをクリックすると、[ファイル保存]ボタン、[ファイル読出し]ボタンにより、データの読書きの対象となります。 フォルダ名をクリックすると、表示するフォルダが切り換わります。
----	--

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

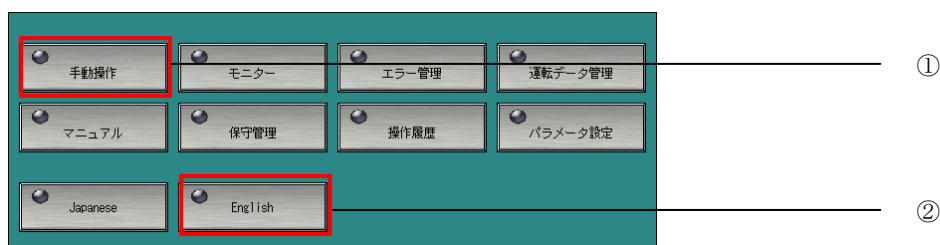
項目	設定	内容
コントロール名	list.ShowFileInfo	-
最大行数	20	-
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

## 2-14 「メニュー画面」の作成

ここではメニュー画面の作成について、説明します。

基本的なコントロールの作成手順は「2-4 コントロールの作成と設定」と同じですの  
で、本画面の特徴的な部分を説明します。

また、生成したソースコードに対し追加する処理は、「4 章」を参照してください。



### 画面プロパティ設定

本画面のプロパティでは、以下のコールバック関数を使用します。

項目	設定	内容
OnCreate	あり	以下の処理を実施します。 ・現在選択されている画面に対応するボタンにフォーカス移動 ・現在選択されている言語に対応するボタンにフォーカス移動

### ①「手動操作ボタン」(ボタンコントロール)

手動操作ボタンは、ボタンコントロールを用いて作成しています。

以下の動作をします。

動作	ボタンをクリックすると、表示する画面を「手動操作画面」に切り換え、メニュー画面を閉じます。
----	---

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	btnManualOperation	–
ボタンタイプ	モーメンタリ	ボタンの動作種別は「ボタンタイプ」で設定します。
表示タイプ	イメージ	ボタンにイメージデータを表示する場合は、「表示タイプ」に「イメージ」を設定します。「ON 時意匠」、「OFF 時意匠」に、表示するイメージリソースの ID を設定します。
ON 時 意匠	img_MenuButton_ON	
ON 時 意匠	img_MenuButton_OFF	
文字列	ID_STRING_ManualOperation	表示する文字列リソース ID を設定します。
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

## ②「English ボタン」(ボタンコントロール)

English ボタンは、ボタンコントロールを用いて作成しています。

以下の動作をします。

動作	ボタンをクリックすると、表示するロケールを、英語に切り換えます。
----	----------------------------------

画面上に、ボタンコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

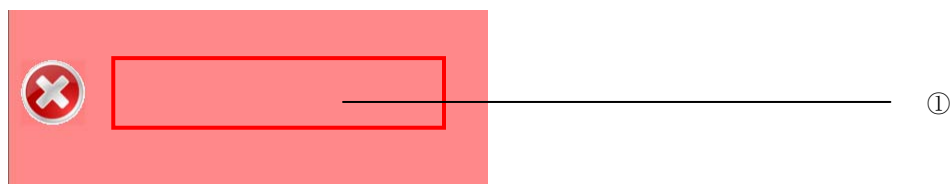
項目	設定	内容
コントロール名	btnEnglish	–
ボタンタイプ	モーメンタリ	ボタンの動作種別は「ボタンタイプ」で設定します。
表示タイプ	イメージ	ボタンにイメージデータを表示する場合は、「表示タイプ」に「イメージ」を設定します。「ON 時意匠」、「OFF 時意匠」に、表示するイメージリソースの ID を設定します。
ON 時 意匠	img_MenuButton_ON	
ON 時 意匠	img_MenuButton_OFF	
文字列	ID_STRING_English	表示する文字列リソース ID を設定します。
OnClick	あり	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。その処理はボタンをクリックしたときに実行するため、「OnClick」を「あり」とします。

## 2-15 「メッセージ画面」の作成

ここではメッセージ画面の作成について、説明します。

基本的なコントロールの作成手順は「2-4 コントロールの作成と設定」と同じですので、本画面の特徴的な部分を説明します。

また、生成したソースコードに対し追加する処理は、「4 章」を参照してください。



### ①「メッセージ」(スタティックテキストコントロール)

メッセージは、スタティックテキストコントロールを用いて作成しています。

以下の動作をします。

動作	ユーザセキュリティ機能に関するエラーメッセージを表示します。 ・保守管理画面でパスワードを間違えたときに表示されるメッセージ ・操作権限のない部品を操作しようとしたときに表示されるメッセージ また、メッセージは、表示から 1.5 秒経過すると、自動的に閉じます。
----	--

画面上に、スタティックテキストコントロールを配置後、次のようにプロパティを設定します。(主な変更点のみ記載)

項目	設定	内容
コントロール名	bstcErrorMessage	-
コールバック関数	OnTimer	表示する文字列リソース ID を設定します。
	OnCreate	「動作」に記載した処理は、CI SKETCH が生成するソースコードに追記します。表示するメッセージ(文字列)を切り換える処理、およびメッセージを閉じる処理を行います。

## 2-16 「テンキー画面」の作成

ここではテンキー画面の作成について、説明します。

テンキーの作成は、[保守管理画面]でも実施しているため、コントロールについては「2-10 「保守管理画面」の作成」を参照してください。



### 画面プロパティ設定

本画面のプロパティでは、以下のコールバック関数を使用します。

項目	設定	内容
OnCreate	あり	以下の処理を実施します。 ・パラメータ設定画面の10 進入力エリアからの起動が16 進入力エリアからの起動かを判断し、10 進入力エリアの場合は「A」～「F」ボタンを無効に、16 進入力エリアの場合は有効に切り替える。



## 第 3 章 ソースコード生成

本章では、作成した画面データからソースコードを生成するまでの手順を説明します。

3-1 ソースコード生成.....	3-1
3-2 生成ファイル.....	3-6

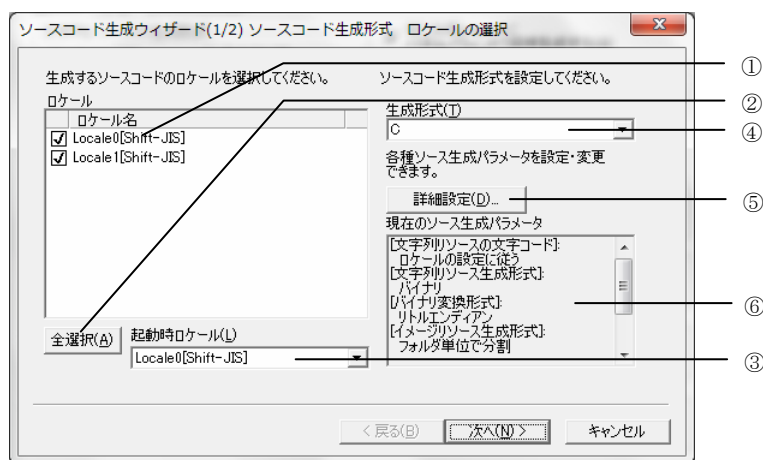
## 3-1 ソースコード生成

GENSKETCH3 では、作成した画面データをそのまま C 言語や C++言語のソースコードとして生成することができます。

本項では C 言語のソースコードの生成方法と、生成されるファイルについて説明します。

### ソースコード生成方法

1. [ファイル]メニューの[ソースコード生成]を選択します。
2. [ソースコード生成ウィザード]が表示されます。



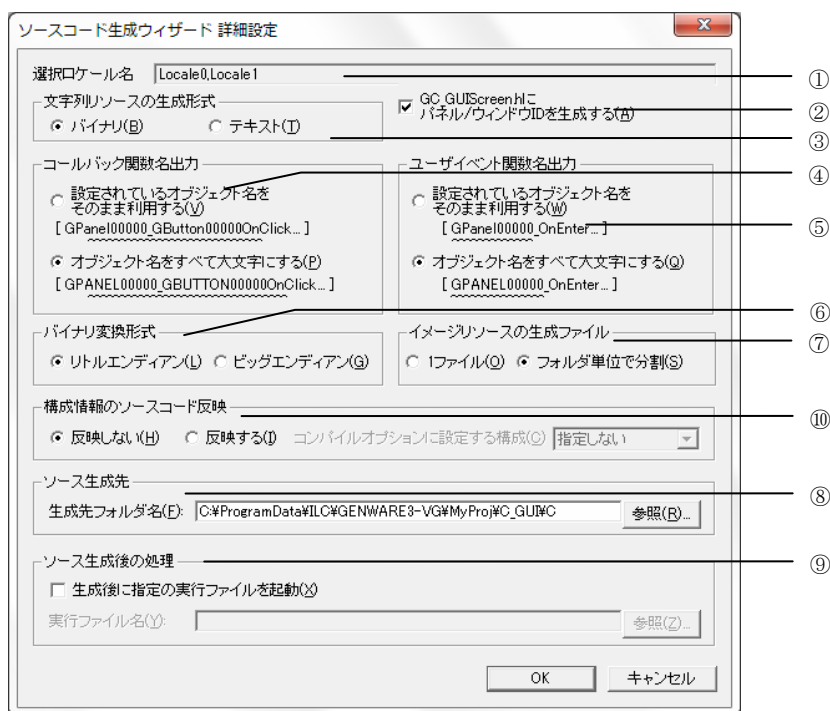
番号	項 目	内 容
①	ロケール	生成するロケールを選択します。
②	全選択	ロケール一覧に表示されているロケールを全選択します。
③	起動時ロケール	実行モジュール起動時の初期ロケールを選択します。
④	生成形式	C、C++など、生成形式を選択します。
⑤	[詳細設定]ボタン	ボタンを押下すると、ソースコード生成時のさまざまなパラメータを設定するための「詳細設定」ダイアログを表示します。
⑥	現在のソース生成パラメータ	ソースコード生成時のパラメータとして、設定されている情報を一覧表示します。パラメータを変更したい場合には、④の[詳細設定]ボタンを押下します。

## GENWARE3 アプリケーション設計ガイド

今回のサンプルアプリケーションでは上記の項目を以下のように設定してください。  
ほかの設定はデフォルトのまま使用します。

番号	項目	内 容
①	ロケール	Locale_JPN[Shift-JIS]、Locale_ENG[Shift-JIS]

3. 設定完了後、[詳細設定]ボタンをクリックし、生成するソースコードの詳細情報を設定します。

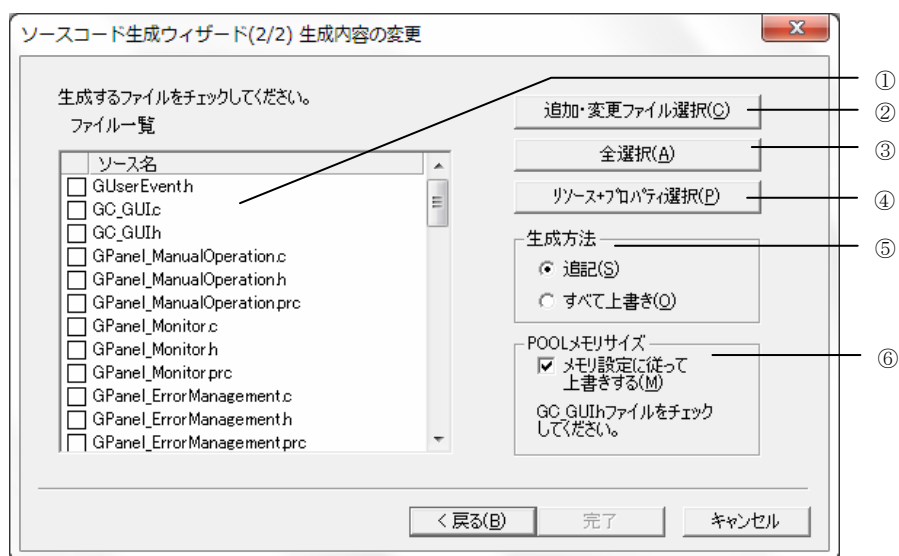


番号	項 目	内 容
①	選択ロケール名	ソースコード生成されるロケール名がすべて表示されます。
②	G「プロジェクト名」Screen.h にパネル/ウィンドウ ID を生成する	「パネル/ウィンドウ」の ID をソースコード中に出力する/しないを設定します。
③	文字列リソースの生成形式	文字列リソースの生成形式を「バイナリ」、「テキスト」から選択します。ロケールを複数選択している場合は、「バイナリ」のみ選択可能です。
④	コールバック関数名出力	ソースコード生成時に、コールバック関数名のオブジェクト名をそのまま生成するか、すべて大文字にするかを設定します。
⑤	ユーザーイベント関数名出力	ソースコード生成時に、ユーザーイベント関数名のオブジェクト名をそのまま生成するか、すべて大文字にするかを設定します。
⑥	バイナリ変換形式	バイナリ変換形式を「リトルエンディアン」、「ビッグエンディアン」から選択します。
⑦	イメージリソースの生成ファイル	イメージリソースをソースコード生成するときに、1 ファイルにまとめて生成するか、フォルダごとに 1 ファイルとして生成するかを設定します。
⑧	ソース生成先	ソースファイルを生成するフォルダをフルパスで指定します。
⑨	ソース生成後の処理	ソースコード生成後に、指定した実行ファイルを起動します。
⑩	構成情報のソースコード反映	構成機能を使用しない場合は「反映しない」を選択します。

今回のサンプルアプリケーションでは、「コールバック関数名出力」を「設定されているオブジェクト名をそのまま利用する」、「イメージリソースの生成ファイル」を「1 ファイル」に設定します。その他の項目はデフォルトのまま使用します。

4. 設定完了後、「OK」ボタンをクリックしてダイアログを閉じます。

5. [次へ]をクリックします。



番号	項 目	内 容
①	ファイル一覧	生成するファイルの種類を選択します。 ファイルを選択し、□をクリックすると、チェック ON、チェック OFF が切り換わります。
②	追加・変更ファイル 選択	クリックすると、前回生成時から追加・変更があったファイルを選択状態にします。 なお、「ファイル一覧」の選択状態は変化しません。
③	全選択	全てのファイルを選択状態にします。 なお、「ファイル一覧」の選択状態は変化しません。
④	リソース+プロパティ 選択	クリックすると、リソースとプロパティのファイルを選択状態にします。 なお、「ファイル一覧」の選択状態は変化しません。
⑤	生成方法	「追記」を選択すると、前回ソースコード生成後に変更されたデータのみがソースファイルに追記されます。 「全て上書き」を選択すると、ソースファイルの内容全てを上書きします。
⑥	POOL メモリサイズ	「メモリ設定に従って上書きする」をチェック ON すると、「メモリ容量の確認と最適化」で最適化したメモリ容量がソースコードに生成されます。メモリ容量は「G プロジェクト名.h」ファイルに記述されますので、「G プロジェクト名.h」ファイルは必ずチェック ON してください。 「メモリ容量の確認と最適化」につきましては、「12-2 メモリ容量の確認と最適化」をご参照ください。

今回のサンプルアプリケーションでは、「全選択」で「全て上書き」を選択してください。

その後データ変更時に応じて「追記」でソースファイルを追記してください。

6. [完了]をクリックすると、ソースコードが自動生成されます。

生成完了後、完了を通知するダイアログが表示されます。[OK]ボタンをクリックします。

生成されたソースコードは、プロジェクト作成時に指定した生成先フォルダに各種ファイルとして保存されます。

## 3-2 生成ファイル

ソースコードファイル生成用のフォルダに以下のようにファイルが生成されます。

### ①メインプログラム

ファイル名	内 容
GC_GUI.c	メインプログラムのソースファイルです。
GC_GUI.h	メインプログラムのヘッダファイルです。

### ②リソースファイル

ファイル名	内 容
GResource.c	リソースデータファイルです。
GResource.h	リソースのヘッダファイルです。
GResStringLocale_JPN.c	日本語ロケールのデータファイルです。
GResStringLocale_ENG.c	英語ロケールのデータファイルです。
GResImage_Folder001.c	イメージリソースのデータファイルです。
GSCRResImage_Folder001.c	イメージリソースのデータファイルです。
GDBObject.c	GUI データベースオブジェクトのソースファイルです。
GDBObject.h	GUI データベースオブジェクトのヘッダファイルです。
GDBObject.prc	GUI データベースオブジェクトのデータファイルです。
GControllID.h	コントロール一覧を定義したヘッダファイルです。
GUserEvent.h	ユーザイベント ID のヘッダファイルです。
GAction.c	システム用のソースファイルです。
GAction.h	システム用のヘッダファイルです。
GScript.c	システム用のソースファイルです。
GScript.h	システム用のヘッダファイルです。
GScript_def.h	システム用のヘッダファイルです。

**MEMO**

◆ ソースコード生成を行うと、「GDBObject.prc」には、以下の変数が作成されます。

```
/*{GDB_OBJECT_UPDATE_VARIABLE_*/
static GUInt8 GDBUpdateReadFlag[7];
static GUInt8 GDBUpdateWriteFlag[7];
/*}_GDB_OBJECT_UPDATE_VARIABLE_*/
```

これらはユーザアプリケーションで GUI データベースの変数の更新有無を判断するために準備されている変数です。

ユーザアプリケーション内で使用しない場合、このままコンパイルを行うとワーニングが表示されますので、その場合は、削除またはコメントアウトしてください。

## ③スクリーン作成プログラム

ファイル名	内 容
GC_GUIScreen.c	スクリーン作成プログラムのソースファイルです。
GC_GUIScreen.h	スクリーン作成プログラムのヘッダファイルです。
GC_GUIScreen.prc	スクリーン作成プログラムのデータファイルです。

## ④各画面プログラム

ファイル名	内 容
GPanel_ManualOperation.c	手動操作画面のソースファイルです。
GPanel_ManualOperation.h	手動操作画面のヘッダファイルです。
GPanel_ManualOperation.prc	手動操作画面のデータファイルです。
GPanel_Monitor.c	モニター画面のソースファイルです。
GPanel_Monitor.h	モニター画面のヘッダファイルです。
GPanel_Monitor.prc	モニター画面のデータファイルです。
GPanel_ErrorManagement.c	エラー管理画面のソースファイルです。
GPanel_ErrorManagement.h	エラー管理画面のヘッダファイルです。
GPanel_ErrorManagement.prc	エラー管理画面のデータファイルです。
GPanel_OperDataManagement.c	運転データ管理画面のソースファイルです。
GPanel_OperDataManagement.h	運転データ管理画面のヘッダファイルです。
GPanel_OperDataManagement.prc	運転データ管理画面のデータファイルです。
GPanel_ManualManagement.c	マニュアル管理画面のソースファイルです。



ファイル名	内 容
GPanel_ManualManagement.h	マニュアル管理画面のヘッダファイルです。
GPanel_ManualManagement.prc	マニュアル管理画面のデータファイルです。
GPanel_Maintenance.c	保守管理画面のソースファイルです。
GPanel_Maintenance.h	保守管理画面のヘッダファイルです。
GPanel_Maintenance.prc	保守管理画面のデータファイルです。
GPanel_OperationHistory.c	操作履歴画面のソースファイルです。
GPanel_OperationHistory.h	操作履歴画面のヘッダファイルです。
GPanel_OperationHistory.prc	操作履歴画面のデータファイルです。
GPanel_ParameterSetting.c	パラメータ設定画面のソースファイルです。
GPanel_ParameterSetting.h	パラメータ設定画面のヘッダファイルです。
GPanel_ParameterSetting.prc	パラメータ設定画面のデータファイルです。
GWindow_Menu.c	メニュー画面のソースファイルです。
GWindow_Menu.h	メニュー画面のヘッダファイルです。
GWindow_Menu.prc	メニュー画面のデータファイルです。
GWindow_Message.c	メッセージ画面のソースファイルです。
GWindow_Message.h	メッセージ画面のヘッダファイルです。
GWindow_Message.prc	メッセージ画面のデータファイルです。
GWindow_NumericKeypad.c	テンキー画面のソースファイルです。
GWindow_NumericKeypad.h	テンキー画面のヘッダファイルです。
GWindow_NumericKeypad.prc	テンキー画面のデータファイルです。

## 第 4 章 ユーザプログラムの追加

CI SKETCH で生成したソースコードに対し、ユーザプログラムの追加を行います。

4-1 追記ファイル一覧 .....	4-1
4-2 ユーザプログラムの追加 .....	4-2

## 4-1 追記ファイル一覧

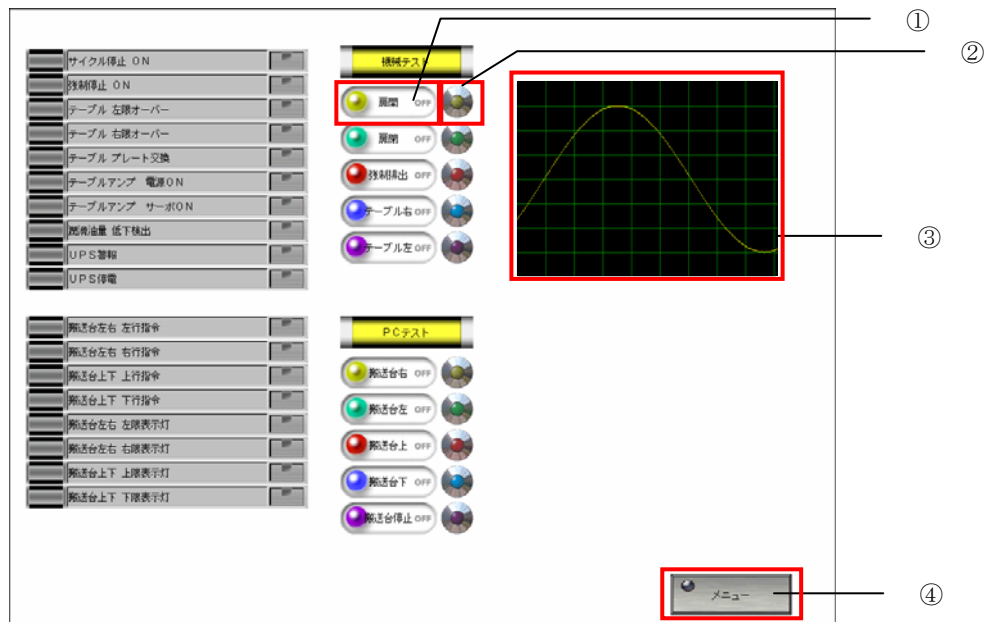
以下のファイルに対し、ユーザプログラムを追加します。

ファイル名	説 明
GPanel_ManualOperation.c	手動操作画面のソースファイル
GPanel_Monitor.c	モニター画面のソースファイル
GPanel_ErrorManagement.c	エラー管理画面のソースファイル
GPanel_OperDataManagement.c	運転データ管理画面のソースファイル
GManualManagement.c	マニュアル管理画面のソースファイル
GPanel_Maintenance.c	保守管理画面のソースファイル
GPanel_OperationHistory.c	操作ログ画面のソースファイル
GPanel_ParameterSetting.c	パラメータ設定画面のソースファイル
GWindow_Menu.c	メニュー画面のソースファイル
GWindow_Message.c	メッセージ画面のソースファイル
GWindow_NumericKeypad.c	テンキー画面のソースファイル

## 4-2 ユーザプログラムの追加

### 4-2-1 GPanel\_ManualOperation.c

ここでは 3 章で説明した画面のコールバック関数、および①～④のコントロールに関する処理について、説明します。



#### 画面のコールバック関数 OnUpdateDB

画面のコールバック関数 OnUpdateDB に追加したソースコードを以下に示します。

なお、②のランプ表示は、この OnUpdateDB の中で処理を行っています。

```
void GPanel_ManualOperation_OnUpdateDB( void *pSelf, void *pParam )
{
    GDBChgMgr *pGDBChgMgr;
    unsigned short gUShortDBstr;

    pGDBChgMgr = (GDBChgMgr*)pParam;
    //テーブル→GDB_SIGNAL_STS1 の値に変化があるか確認
    if( GDBChgMgr_IsChangeData( pGDBChgMgr, ID_GDB_SIGNAL_STS1 ) ){
```

```

        //変化があればデータを読み出し
        GWDBReadData( ID_GDB_SIGNAL_STS1, &gUShortDBstr, sizeof( gUShortDBstr ) );
        //データに該当するコントロールの表示を更新
        updatePanelOnShow(pSelf, gUShortDBstr, ID_GDB_SIGNAL_STS1);
    }
    //データ GDB_SIGNAL_STS2 の値に変化があるか確認
    if( GDBChgMgr_IsChangeData( pGDBChgMgr, ID_GDB_SIGNAL_STS2 ) ){
        //変化があればデータを読み出し
        GWDBReadData( ID_GDB_SIGNAL_STS2, &gUShortDBstr, sizeof( gUShortDBstr ) );
        //データに該当するコントロールの表示を更新
        updatePanelOnShow(pSelf, gUShortDBstr, ID_GDB_SIGNAL_STS2);
    }
    //データ GDB_MACHINE_TEST2 の値に変化があるか確認
    if( GDBChgMgr_IsChangeData( pGDBChgMgr, ID_GDB_MACHINE_TEST2 ) ){
        //変化があればデータを読み出し
        GWDBReadData( ID_GDB_MACHINE_TEST2, &gUShortDBstr,
sizeof( gUShortDBstr ) );
        //データに該当するコントロールの表示を更新
        updatePanelOnShow(pSelf, gUShortDBstr, ID_GDB_MACHINE_TEST2);
    }
    //データ GDB_PC_TEST2 の値に変化があるか確認
    if( GDBChgMgr_IsChangeData( pGDBChgMgr, ID_GDB_PC_TEST2 ) ){
        //変化があればデータを読み出し
        GWDBReadData( ID_GDB_PC_TEST2, &gUShortDBstr, sizeof( gUShortDBstr ) );
        //データに該当するコントロールの表示を更新
        updatePanelOnShow(pSelf, gUShortDBstr, ID_GDB_PC_TEST2);
    }
    return;
}

//変化したデータに該当するコントロールの表示を更新する。(ユーザ作成関数)
void updatePanelOnShow(void *pSelf, unsigned short gUShortDBstr, short sID)
{
    GPicture *pPicture;
    int i;

    //変化のあったデータを確認する(サイクル停止 ON-UPS 停電)
    if(sID == ID_GDB_SIGNAL_STS1){
        //データの登録個数分ループする
        for(i = NUM_INIT_0; i < NUM_FLG_10; i++){
            //ヒュークコントロールのオブジェクトを取得(左側ランプ)

```

```

        pPicture = (GPicture*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
(PCTCYCLESTOPONLIGHT - i) );
        //テ-タ-の値を確認して状態ステ-スを変更させる
        if(((gUShortDBstr >> i) % NUM_MOD_2) == NUM_FLG_1){
            GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture,
NUM_STATUS_1 );
        } else {
            GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture,
NUM_STATUS_0 );
        }
        //ヒ-クチャコントロールのオブ-ジェクトを取得(右側ランプ)
        pPicture = (GPicture*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
(PCTCYCLESTOPON + i) );
        //テ-タ-の値を確認して状態ステ-スを変更させる
        if(((gUShortDBstr >> i) % NUM_MOD_2) == NUM_FLG_1){
            GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture,
NUM_STATUS_1 );
        } else {
            GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture,
NUM_STATUS_0 );
        }
    }
    //変化のあったテ-タ-を確認する(搬送台左右左行指令-搬送台上下下限表示灯)
} else if(sID == ID_GDB_SIGNAL_STS2){
    //テ-タ-の登録個数分ル-プする
    for(i = NUM_INIT_0; i < NUM_FLG_8; i++){
        //ヒ-クチャコントロールのオブ-ジェクトを取得(左側ランプ)
        pPicture = (GPicture*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
(PCTCARRIAGELDIRECTIVELIGHT - i) );
        //テ-タ-の値を確認して状態ステ-スを変更させる
        if(((gUShortDBstr >> i) % NUM_MOD_2) == NUM_FLG_1){
            GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture,
NUM_STATUS_1 );
        } else {
            GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture,
NUM_STATUS_0 );
        }
        //ヒ-クチャコントロールのオブ-ジェクトを取得(右側ランプ)
        pPicture = (GPicture*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
(PCTCYCLESTOPON + i + 10) );
    }
}

```

```

//テストケースの値を確認して状態ステータスを変更させる
if(((gUShortDBstr >> i) % NUM_MOD_2) == NUM_FLG_1){
    GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture,
NUM_STATUS_1 );
} else {
    GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture,
NUM_STATUS_0 );
}
}
//変化のあったテストケースを確認する(機械テスト)
}else if(sID == ID_GDB_MACHINE_TEST2){
    //テストケースの登録個数分ループする
    for(i = NUM_INIT_0; i < NUM_FLG_5; i++){
        //ヒューマンコントロールのオブジェクトを取得
        pPicture = (GPicture*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
(PCTHIRAKISEKIONELIGHT - i) );
        //テストケースの値を確認して状態ステータスを変更させる
        if(((gUShortDBstr >> i) % NUM_MOD_2) == NUM_FLG_1){
            GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture,
NUM_STATUS_1 );
        } else {
            GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture,
NUM_STATUS_0 );
        }
    }
//変化のあったテストケースを確認する(PCテスト)
}else if(sID == ID_GDB_PC_TEST2){
    //テストケースの登録個数分ループする
    for(i = NUM_INIT_0; i < NUM_FLG_5; i++){
        //ヒューマンコントロールのオブジェクトを取得
        pPicture = (GPicture*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
(PCTCARRIAGERIGHTLIGHT - i) );
        //テストケースの値を確認して状態ステータスを変更させる
        if(((gUShortDBstr >> i) % NUM_MOD_2) == NUM_FLG_1){
            GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture,
NUM_STATUS_1 );
        } else {
            GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture,
NUM_STATUS_0 );
        }
    }
}

```

```

    }
}
}

```

### ①ボタン

OnClick に追加したソースコードを以下に示します。

```

GInt32 GPANEL_MANUALOPERATION_BTNHIRAKISEKIONeOnClick(GPanel_ManualOperation *pSelf,
GUInt16 usMessage, GLParam lParam, GUParam lUParam)
{
    GButton *pButton;
    GUInt8 ucStatus;
    unsigned short gUShortDBstr;

    //操作無し状態を監視するカウンタをリセット
    CompareOperateTime();
    //操作履歴テーブルに該当ボタンの操作を追加
    DoOperate(BTNHIRAKISEKIONE, pSelf);
    //セキュリティレベルを確認(レベル1以上で無ければ警告を表示)
    if(G_TRUE == CheckOperateLevel(pSelf, Label_1)){
        //テーブルから該当ボタンの ON/OFF 状態を取得
        GWDBReadData( ID_GDB_MACHINE_TEST1, &gUShortDBstr, sizeof( gUShortDBstr ) );
        //取得した ON/OFF 状態を反転(オルタネートボタン動作)
        if((gUShortDBstr & HIRAKISEKIONE) == HIRAKISEKIONE){
            //ON→OFF
            gUShortDBstr -= HIRAKISEKIONE;
        } else {
            //OFF→ON
            gUShortDBstr += HIRAKISEKIONE;
        }
        //テーブルに ON/OFF 状態を書き込み
        GWDBWriteData( ID_GDB_MACHINE_TEST1, &gUShortDBstr, sizeof( gUShortDBstr ),
G_TRUE );
        //実行バースへの反映を行うため DataAccessWrite 関数に渡す情報を格納
        sprintf(stGDBID[0].cObjectName, "%s", "GDB_MACHINE_TEST1");
        stGDBID[0].usGDBID = ID_GDB_MACHINE_TEST1;
        stGDBID[0].sDataSize = 1;
        //テーブルの内容を実行バースに反映
    }
}

```



```

        DataAccessWrite(stGDBID, 1);
    }else{
        //管理パネルの場合にはボタンの状態を元に戻す
        pButton = (GButton*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
        BTNHIRAKISEKIONE );
        //現在のボタン ON/OFF 状態を取得
        ucStatus = GWGetVmt( GButton_VMT, pButton )->GetStatus( pButton );
        //ON/OFF 状態を反転
        ucStatus = NUM_DATA_1 - ucStatus;
        //ボタンステータスに状態を反映
        GWGetVmt( GButton_VMT, pButton )->SetStatus( pButton, ucStatus );
    }
    return G_TRUE;
}

```

### ③グラフ表示

グラフ表示はベースとなる基本コントロールに OnCreate 及び OnTimer コールバック関数を使用して描画を実施しています。

OnCreate 関数にてグラフ描画の初期化を実施し、OnTimer にて描画を更新します。

```

GInt32 GPanel_ManualOperation_BACCTLSINLINEOnCreate(GPanel_ManualOperation
*pSelf, GUInt16 usMessage, GLParam lParam, GUParam lUParam)
{
    double  dX, dY;
    GInt16  nLoop;
    GWDC    *pDC;
    GBasicControl *pBasic;
    HGDRW   hDraw;
    GRect    gRect;

    //デバイスコンテキスト取得
    pDC = (GWDC *)GWGetVmt( GScreen_VMT, GWGetScreen() )->GetWDC( GWGetScreen() );
    //描画ハンドル取得
    hDraw = GWGetVmt( GWDC_VMT, pDC )->GetHGDRW( pDC );
    //基本コントロールオブジェクト取得
    pBasic = (GBasicControl*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
    BACCTLSINLINE );
}

```

```

//基本コントロールの描画領域取得
GWGetVmt(GBasicControl_VMT, pBasic)->GetBounds( pBasic, &gRect );
//描画領域クリップ
GDSetClipRect( hDraw, &gRect );
//グラフ描画開始位置を指定
dX = NUM_DATA_2*PI/MAX_DEGREE;
dY = MAX_SCALE/NUM_DATA_2;
//サイン波データの初期値を設定
for( nLoop=0; nLoop<MAX_DEGREE; nLoop++ ){
    pSelf->gGraphPrev[nLoop].nX = nLoop + NUM_POINT_630;
    pSelf->gGraphPrev[nLoop].nY = NUM_NEGATIVE_1 + NUM_DATA_90;
    pSelf->gGraphPoint[nLoop].nX = nLoop + NUM_POINT_630;
    pSelf->gGraphPoint[nLoop].nY = (GInt16)( SCR_HEIGHT - dY * sin(nLoop * dX)
- dY ) + NUM_DATA_90;
}
//100ms タイマイベントを定義
GSetTimer( pBasic, ID_TIMER_DRAWING, NUM_TIMER_100 );
pSelf->nCounter = NUM_INIT_0;
showLevel = NUM_INIT_0;
return G_TRUE;
}

```

```

GInt32 GPanel_ManualOperation_BACCTLSINLINEOnTimer(GPanel_ManualOperation *pSelf,
GUInt16 usMessage, GLParam lParam, GUParam lUParam)
{
    HGDRAW hDraw;
    GRect gRect;
    GWDC *pDC;
    GBasicControl *pBasic;
    GInt16 nLoop;
    double dX, dY;

    //デバイスコンテキスト取得
    pDC = (GWDC *)GWGetVmt( GScreen_VMT, GWGetScreen() )->GetWDC( GWGetScreen() );
    //描画ハンドル取得
    hDraw = GWGetVmt( GWDC_VMT, pDC )->GetHGDRAW( pDC );
    //基本コントロールオブジェクト取得
    pBasic = (GBasicControl*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
BACCTLSINLINE );
    //基本コントロールの描画領域取得

```

```
GWGetVmt(GBasicControl_VMT, pBasic)->GetBounds( pBasic, &gRect );
//描画領域クリップ
GDSetClipRect( hDraw, &gRect );
//描画開始
GDBeginDraw( hDraw );
//ペン色を黒色に設定
GDSetPenColor( hDraw, RGB16(0x00, 0x00, 0x00));
//背景色に黒を設定
GDSetBackColor( hDraw, RGB16(0x00, 0x00, 0x00));
//短形描画
GDDrawRect( hDraw, &gRect );
//ペン色を緑に設定
GDSetPenColor( hDraw, RGB16(0x00, 0x80, 0x00));
//ペン位置を移動
GDMoveTo( hDraw, NUM_DRAW_665, NUM_DRAW_90 );
//目盛り線の描画
GDLineTo( hDraw, NUM_DRAW_665, NUM_DRAW_330 );
//ペン位置を移動
GDMoveTo( hDraw, NUM_DRAW_700, NUM_DRAW_90 );
//目盛り線の描画
GDLineTo( hDraw, NUM_DRAW_700, NUM_DRAW_330 );
//ペン位置を移動
GDMoveTo( hDraw, NUM_DRAW_735, NUM_DRAW_90 );
//目盛り線の描画
GDLineTo( hDraw, NUM_DRAW_735, NUM_DRAW_330 );
//ペン位置を移動
GDMoveTo( hDraw, NUM_DRAW_770, NUM_DRAW_90 );
//目盛り線の描画
GDLineTo( hDraw, NUM_DRAW_770, NUM_DRAW_330 );
//ペン位置を移動
GDMoveTo( hDraw, NUM_DRAW_805, NUM_DRAW_90 );
//目盛り線の描画
GDLineTo( hDraw, NUM_DRAW_805, NUM_DRAW_330 );
//ペン位置を移動
GDMoveTo( hDraw, NUM_DRAW_840, NUM_DRAW_90 );
//目盛り線の描画
GDLineTo( hDraw, NUM_DRAW_840, NUM_DRAW_330 );
//ペン位置を移動
GDMoveTo( hDraw, NUM_DRAW_875, NUM_DRAW_90 );
//目盛り線の描画
```

```
GDLineTo( hDraw, NUM_DRAW_875, NUM_DRAW_330 );  
//ペン位置を移動  
GDMoveTo( hDraw, NUM_DRAW_910, NUM_DRAW_90 );  
//目盛り線の描画  
GDLineTo( hDraw, NUM_DRAW_910, NUM_DRAW_330 );  
//ペン位置を移動  
GDMoveTo( hDraw, NUM_DRAW_945, NUM_DRAW_90 );  
//目盛り線の描画  
GDLineTo( hDraw, NUM_DRAW_945, NUM_DRAW_330 );  
//ペン位置を移動  
GDMoveTo( hDraw, NUM_DRAW_630, NUM_DRAW_120 );  
//目盛り線の描画  
GDLineTo( hDraw, NUM_DRAW_950, NUM_DRAW_120 );  
//ペン位置を移動  
GDMoveTo( hDraw, NUM_DRAW_630, NUM_DRAW_150 );  
//目盛り線の描画  
GDLineTo( hDraw, NUM_DRAW_950, NUM_DRAW_150 );  
//ペン位置を移動  
GDMoveTo( hDraw, NUM_DRAW_630, NUM_DRAW_180 );  
//目盛り線の描画  
GDLineTo( hDraw, NUM_DRAW_950, NUM_DRAW_180 );  
//ペン位置を移動  
GDMoveTo( hDraw, NUM_DRAW_630, NUM_DRAW_210 );  
//目盛り線の描画  
GDLineTo( hDraw, NUM_DRAW_950, NUM_DRAW_210 );  
//ペン位置を移動  
GDMoveTo( hDraw, NUM_DRAW_630, NUM_DRAW_240 );  
//目盛り線の描画  
GDLineTo( hDraw, NUM_DRAW_950, NUM_DRAW_240 );  
//ペン位置を移動  
GDMoveTo( hDraw, NUM_DRAW_630, NUM_DRAW_270 );  
//目盛り線の描画  
GDLineTo( hDraw, NUM_DRAW_950, NUM_DRAW_270 );  
//ペン位置を移動  
GDMoveTo( hDraw, NUM_DRAW_630, NUM_DRAW_300 );  
//目盛り線の描画  
GDLineTo( hDraw, NUM_DRAW_950, NUM_DRAW_300 );  
//ペン位置を移動  
GDMoveTo( hDraw, NUM_DRAW_630, NUM_DRAW_330 );  
//目盛り線の描画
```

```

GDLineTo( hDraw, NUM_DRAW_950, NUM_DRAW_330 );
//ペン位置を移動
GDMoveTo( hDraw, pSelf->gGraphPoint[NUM_BUFF_0].nX,
pSelf->gGraphPoint[NUM_BUFF_0].nY );
//ペン色を黄色に設定
GDSetPenColor( hDraw, RGB16(0xFF, 0xFF, 0x00));
//グラフ描画
for( nLoop=0; nLoop<SCR_WIDTH; nLoop++ ){
    GDSetPixel( hDraw, pSelf->gGraphPoint[nLoop].nX,
pSelf->gGraphPoint[nLoop].nY );
}
//グラフ描画開始位置を指定
dX = NUM_DATA_2*PI/MAX_DEGREE;
dY = MAX_SCALE/NUM_DATA_2;
//サインカーブデータを更新
for( nLoop=0; nLoop<MAX_DEGREE; nLoop++ ){
    pSelf->gGraphPrev[nLoop].nY = pSelf->gGraphPoint[nLoop].nY;
    pSelf->gGraphPoint[nLoop].nY = (GInt16 )(SCR_HEIGHT - dY *
sin((nLoop-pSelf->nCounter) * dX) - dY);
}
//描画回数を加算 (sin 関数の角度変化に使用)
pSelf->nCounter++;
//強制描画
GDFlushScreen( hDraw, gRect.nXmin, gRect.nYmin,
(GInt16 )( gRect.nXmax - gRect.nXmin+1 ),
(GInt16 )( gRect.nYmax - gRect.nYmin + 1 ));
//描画終了
GDEndDraw(hDraw);

//現状のデバイス状態を読み出すため情報を格納
sprintf(stGDBID[0].cObjectName, "%s", "GDB_SIGNAL_STS1");
stGDBID[0].usGDBID = ID_GDB_SIGNAL_STS1;
stGDBID[0].sDataSize = 1;
sprintf(stGDBID[1].cObjectName, "%s", "GDB_SIGNAL_STS2");
stGDBID[1].usGDBID = ID_GDB_SIGNAL_STS2;
stGDBID[1].sDataSize = 1;
sprintf(stGDBID[2].cObjectName, "%s", "GDB_MACHINE_TEST2");
stGDBID[2].usGDBID = ID_GDB_MACHINE_TEST2;
stGDBID[2].sDataSize = 1;
sprintf(stGDBID[3].cObjectName, "%s", "GDB_PC_TEST2");

```

```

        stGDBID[3].usGDBID = ID_GDB_PC_TEST2;
        stGDBID[3].sDataSize = 1;
        //実データ読み出し
        DataAccessRead(stGDBID, 4);

        return G_TRUE;
    }

```

#### ④メニューボタン

OnClick に追加したソースコードを以下に示します。

```

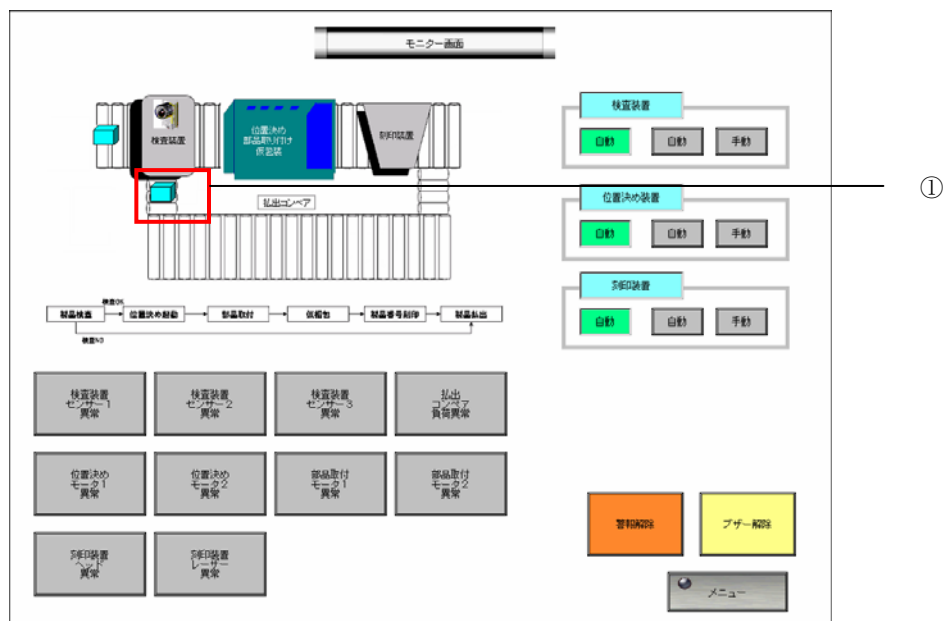
GInt32
GPANEL_MANUALOPERATION_BTNMANUALOPERATIONMENUOnClick(GPanel_ManualOperation
*pSelf, GUInt16 usMessage, GLParam lParam, GUParam lUParam)
{
    GBasicControl *pBasic;

    //操作無し状態を監視するカウンタをリセット
    CompareOperateTime();
    //基本コントロールオブジェクトを取得
    pBasic = (GBasicControl*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
BACCTLSINLINE );
    //タイマーを削除
    GEKillTimer( pBasic, ID_TIMER_DRAWING);
    //操作履歴ログに該当ボタンの操作を追加
    DoOperate(BTNMANUALOPERATIONMENU, pSelf);
    //メニューウィンドウを表示
    GActScreenOpenWindow( pSelf, SHOWGWINDOW_MENU );
    return G_TRUE;
}

```

## 4-2-2 GPanel\_Monitor.c

ここでは 3 章で説明した画面のコールバック関数に関する処理について、説明します。



## 画面のコールバック関数 OnCreate

①のピクチャコントロールはアニメーション機能により移動します。

アニメーションの起動は、画面のコールバック関数 OnCreate で実施しています。

```
void GPanel_Monitor_OnCreate( void *pSelf, GBaseWindow *pParent, GInt16
*pReturnValue )
{
    GInt16 nResult; /*結果*/
    unsigned short gUShortDBstr;

    //データベース読み出し (GDB_ERR_DEVICE_TYPE)
    GWDBReadData( ID_GDB_ERR_DEVICE_TYPE, &gUShortDBstr, sizeof( gUShortDBstr ) );
}
```

```
//データ内容を画面に反映（異常表示ハナ）  
updatePanelErrorButton(pSelf, gUShortDBstr);  
//データ読み出し（GDB_DEVICE_TYPE）  
GWDBReadData( ID_GDB_DEVICE_TYPE, &gUShortDBstr, sizeof( gUShortDBstr ) );  
//データ内容を画面に反映（操作ボタン）  
updatePanelDeviceButton(pSelf, gUShortDBstr);  
//フレームアニメーションを再生する  
nResult = GWGetVmt( GFrameAnimationPanel_VMT,  
pSelf )->StartFrameAnimation( pSelf );  
//画面表示更新用にタイマーを起動  
GSetTimer(pSelf, 10, 1000);  
return;  
}
```



## 4-2-3 GPanel\_ManualManagement.c

ここでは 3 章で説明した画面のコールバック関数、および①～⑤のコントロールに関する処理について、説明します。



## 画面のコールバック関数 OnUpdateDB

①、④のエラー情報の表示は画面のコールバック関数 OnUpdateDB で実施しています。

```
void GPanel_ErrorManagement_OnUpdateDB( void *pSelf, void *pParam )
{
    GDBChgMgr *pGDBChgMgr;
    GStaticText *pTextNowErr;
    GTCHAR szNowErrString[ERROR_ITEM_STRING_SIZE];

    pGDBChgMgr = (GDBChgMgr*)pParam;
    //ID_GDB_ERR_LIST が変更されているか確認する
```

```

    if( GDBChgMgr_IsChangeData( pGDBChgMgr, ID_GDB_ERR_LIST ) ){
        //エラー一覧の更新
        RefreshErrorList(pSelf);
        //文字列変数を初期化
        memset(szNowErrString, 0, sizeof(szNowErrString));
        //現在発生中エラー文字列を取得
        GetNowErrString(szNowErrString);
        //取得した文字列を現在発生中エラー表示欄へ反映
        pTextNowErr = (GStaticText*)GWGetVmt( GPanel_VMT,
pSelf )->GetChild( pSelf, TEXTERRORINFO );
        GWGetVmt( GStaticText_VMT, pTextNowErr )->SetString( pTextNowErr,
szNowErrString );
    }
    return;
}

void RefreshErrorList(void *pSelf)
{
    GUInt8          ucIndex;
    GStaticText      *pSText;
    GTCHAR           szErrInfo[ERROR_ITEM_STRING_SIZE];
    ERROR_HISTORY_NODE *pHistoryNode;
    GCaption          *pCaption;

    //エラーメッセージ内容分ループ
    for(ucIndex = NUM_INIT_0, pHistoryNode = g_pRecodeHeader_Error->pTop; (ucIndex
< ERROR_LIST_ITEM_COUNT) && (NULL != pHistoryNode); ucIndex++, pHistoryNode =
pHistoryNode->pNext) {
        memset(szErrInfo, NUM_INIT_0, sizeof(szErrInfo));
        //表示対象行のステックテキストオブジェクトを取得
        pSText = (GStaticText*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
(TESTERRORLIST01 + ucIndex) );
        //オブジェクトのキャプションの値を取り出し（文字色変更のため）
        pCaption = GWGetVmt( GStaticText_VMT, pSText )->GetCaption( pSText );
        //エラー発生
        if (pHistoryNode->data.ucErrSts == ERROR_STS_TRIGGER) {
            //エラー発生として文字列を生成
            // 2   2013/12/10  15:30:12   01FA   通信エラーが発生しました。
            sprintf(szErrInfo, ERROR_STRING_FORMAT_TRIGGER,
pHistoryNode->data.ucErrLevel,

```

```

        pHistoryNode->data.date.usYear, pHistoryNode->data.date.usMonth,
pHistoryNode->data.date.usDay,
        pHistoryNode->data.timeTrigger.usHour,
pHistoryNode->data.timeTrigger.usMinute,
pHistoryNode->data.timeTrigger.usSecond,
        pHistoryNode->data.usErrCode,
GRCLoadString(pHistoryNode->data.usStrID));
        //文字色を赤に設定
        pCaption->gcColor = RGB16(0xFF, 0x00, 0x00);
    } else {
        //エラー復旧
        //エラー発生として文字列を生成
        // 2 2013/12/10 15:30:12 01FA 通信エラーが発生しました。
17:30:56
        sprintf(szErrInfo, ERROR_STRING_FORMAT_REPAIR,
pHistoryNode->data.ucErrLevel,
        pHistoryNode->data.date.usYear, pHistoryNode->data.date.usMonth,
pHistoryNode->data.date.usDay,
        pHistoryNode->data.timeTrigger.usHour,
pHistoryNode->data.timeTrigger.usMinute,
pHistoryNode->data.timeTrigger.usSecond,
        pHistoryNode->data.usErrCode,
GRCLoadString(pHistoryNode->data.usStrID),
        pHistoryNode->data.timeRepair.usHour,
pHistoryNode->data.timeRepair.usMinute,
pHistoryNode->data.timeRepair.usSecond);
        //文字色を青に設定
        pCaption->gcColor = RGB16(0x00, 0x00, 0xFF);
    }
    //文字列を反映
    GWGetVmt( GStaticText_VMT, pSText )->SetString(pSText, szErrInfo);
}
}

```

## ②スクロールバー

スクロールバーを操作したとき①の表示更新を、OnScroll により実施しています。

```
GInt32 GPanel_ErrorManagement_SCROLLBARERROROnScroll(GPanel_ErrorManagement
*pSelf, GUInt16 usMessage, GLParam lParam, GUParam lUParam)
{
    GScrollBarEx *pScrollBarEx;
    GInt32 lPosition;
    GStaticText *pSText;
    GInt16 ucIndex;
    GInt16 nErrSts;
    GTCHAR szHistoryStr[ERROR_ITEM_STRING_SIZE];
    GCaption *pCaption;

    //操作無し状態を監視するカウンタをリセット
    CompareOperateTime();
    //スクロールバーオブジェクトを取得
    pScrollBarEx = (GScrollBarEx*)GWGetVmt(GPanel_VMT, pSelf)->GetChild(pSelf,
    SCROLLBARERROR);
    //現在のスクロール位置を取得
    lPosition = GWGetVmt(GScrollBarEx_VMT,
    pScrollBarEx)->GetScrollPosition(pScrollBarEx);
    //エラー表示行数分ループ
    for(ucIndex=0; ucIndex< ERROR_LIST_ITEM_COUNT; ucIndex++){
        memset(szHistoryStr, NUM_INIT_0, sizeof(szHistoryStr));
        //該当位置のスタティックテキストオブジェクトを取得
        pSText = (GStaticText*)GWGetVmt(GPanel_VMT, pSelf)->GetChild(pSelf,
        (TESTERRORLIST01 + ucIndex));
        //該当位置のエラー内容とエラー状態を取得
        nErrSts = GetErrorHistoryString((lPosition + ucIndex), szHistoryStr);
        //該当位置のエラー文字列を反映
        GWGetVmt(GStaticText_VMT, pSText)->SetString(pSText, szHistoryStr);
        //文字列色を変更するためキャプション情報を取得
        pCaption = GWGetVmt(GStaticText_VMT, pSText)->GetCaption(pSText);
        //発生中
        if (ERROR_STS_TRIGGER == nErrSts){
            //文字色を赤に設定
            pCaption->gcColor = RGB16(0xFF, 0x00, 0x00);
            //復旧
```

## GENWARE3 アプリケーション設計ガイド

```

    }else if (ERROR_STS_REPAIR == nErrSts){
        //文字色を青に設定
        pCaption->gcColor = RGB16(0x00, 0x00, 0xFF);
    }
}

return G_TRUE;
}

```

## ③「OLD→NEW」ボタン

「OLD→NEW」ボタンの OnClick で、①の内容をソートする処理を実施しています。

```

GInt32 GPanel_ERRORMANAGEMENT_BTNOLDTONEWOnClick(GPanel_ErrorManagement *pSelf,
GUInt16 usMessage, GLParam lLParam, GUParam lUParam)
{
    //操作無し状態を監視するカウンタをリセット
    CompareOperateTime();
    //操作履歴中に該当ボタンの操作を追加
    DoOperate(BTNOLDTONEW, pSelf);
    //ロギングデータのソート処理
    SortRecode(SORT_TYPE_OLD_TO_NEW);
    //ロギングデータの表示処理
    RefreshErrorList(pSelf);
    //ファイルに書き込む
    WRITE_RECODE(g_pRecodeHeader_Error, ERROR_HISTORY_NODE,
ERROR_HISTORY_FILE_NAME);
    return G_TRUE;
}

//リングバッファをソートする
void SortRecode(ERROR_SORT_TYPE sortType)
{
    ERROR_HISTORY_NODE *pNextNode;
    ERROR_HISTORY_NODE *pNowNode;
    ERROR_HISTORY_HEADER *pNewErrHead;

    //リングバッファソート用にメモリ確保
    pNewErrHead = (ERROR_HISTORY_HEADER*)malloc(sizeof(ERROR_HISTORY_HEADER));
    //ソート用バッファの初期化

```

```

memset(pNewErrHead, 0, sizeof(ERROR_HISTORY_HEADER));
pNewErrHead->pTail = NULL;
pNewErrHead->pTop = NULL;
pNewErrHead->ucCount = NUM_INIT_0;
//リングバッファ個数分ループ
while(g_pRecodeHeader_Error->ucCount > 0) {
    //データポインタ取得
    pNextNode = g_pRecodeHeader_Error->pTop->pNext;
    pNowNode = g_pRecodeHeader_Error->pTop;
    //データがあればソート処理を実行
    while (NULL != pNextNode) {
        //ソート条件毎で該当する内容を取得
        if (SORT_TYPE_OLD_TO_NEW == sortType) {
            if (0 < CompareTime(pNowNode, pNextNode)) {
                //現在ノードが次のノードより新しい
                pNowNode = pNextNode;
            }
        }
        }else if (SORT_TYPE_NEW_TO_OLD == sortType) {
            if (0 > CompareTime(pNowNode, pNextNode)) {
                //現在ノードが次のノードより古い
                pNowNode = pNextNode;
            }
        }
        }else if (SORT_TYPE_HIGH_TO_LOW == sortType) {
            //エラーレベルの一番高いノードを探す
            if (pNowNode->data.ucErrLevel < pNextNode->data.ucErrLevel) {
                pNowNode = pNextNode;
            }
        }
        }else if (SORT_TYPE_LOW_TO_HIGH == sortType) {
            //エラーレベルの一番低いノードを探す
            if (pNowNode->data.ucErrLevel > pNextNode->data.ucErrLevel) {
                pNowNode = pNextNode;
            }
        }
    }
    pNextNode = pNextNode->pNext;
}
//先頭のノードではない場合
if (NULL != pNowNode->pPrev) {
    pNowNode->pPrev->pNext = pNowNode->pNext;
} else {
    //残ったノードが一つ以上

```

```

        if (NULL != pNowNode->pNext) {
            g_pRecodeHeader_Error->pTop = pNowNode->pNext;
            g_pRecodeHeader_Error->pTop->pPrev = NULL;
        }

    }
    //最後のノードではない場合
    if (NULL != pNowNode->pNext) {
        pNowNode->pNext->pPrev = pNowNode->pPrev;
    } else {
        //残ったノードが一つ以上
        if (NULL != pNowNode->pPrev) {
            g_pRecodeHeader_Error->pTail = pNowNode->pPrev;
            g_pRecodeHeader_Error->pTail->pNext = NULL;
        }
    }
    //ノード数を減算
    g_pRecodeHeader_Error->ucCount -= NUM_DATA_1;
    pNowNode->pNext = NULL;
    pNowNode->pPrev = NULL;
    //新しいリンクリストに追加
    PUSH_RECODE(pNewErrHead, pNowNode, ERROR_HISTORY_NODE,
RECODE_COUNT_MAX_ERROR);
}
//古いエラーヘッダー領域を開放する
free(g_pRecodeHeader_Error);
//新しい領域を指す
g_pRecodeHeader_Error = pNewErrHead;
}

```

### ⑤「履歴クリア」ボタン

「履歴クリア」ボタンの OnClick で、エラー履歴をクリアする処理を実施しています。

```

GInt32 G_PANEL_ERRORMANAGEMENT_BTNHISTORYCLEAROnClick(GPanel_ErrorManagement
*pSelf, GUInt16 usMessage, GLParam ILParam, GUPParam IUPParam)
{
    GStaticText *pSText;
    GUInt8 ucIndex;

```

```

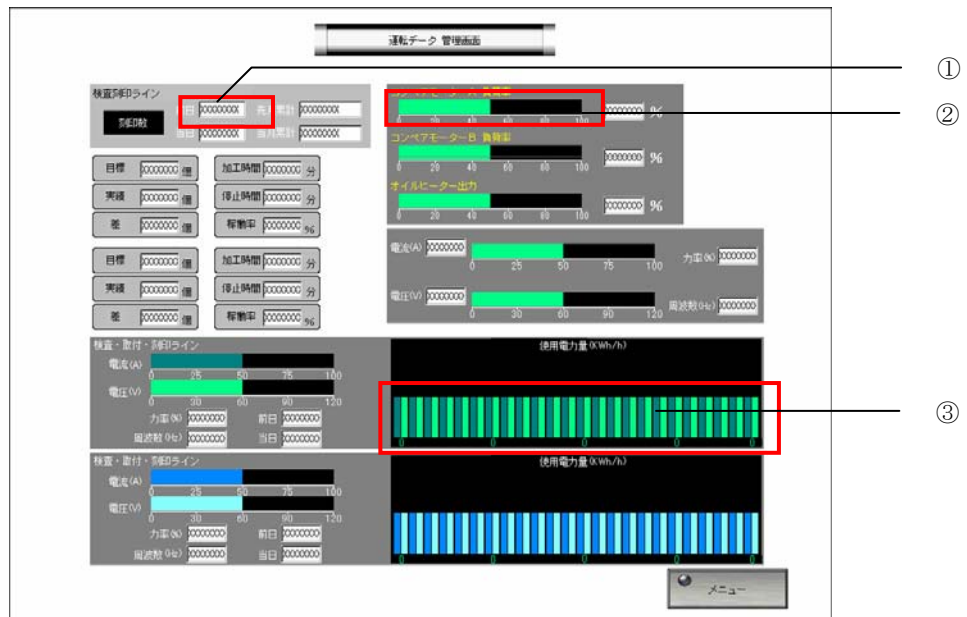
//操作無し状態を監視するカウンタをリセット
CompareOperateTime();
//操作履歴中に該当ボタンの操作を追加
DoOperate(BTNHISTORYCLEAR, pSelf);
//セキュリティレベルを確認(レベル1以上で無ければ警告を表示)
if(G_TRUE == CheckOperateLevel(pSelf, Level_1)){
    //エラー一覧リストをクリア
    for(ucIndex=0; ucIndex< ERROR_LIST_ITEM_COUNT; ucIndex++){
        //各行のスタティックテキストオブジェクトを取得
        pSText = (GStaticText*)GWGetVmt(GPanel_VMT, pSelf)->GetChild(pSelf,
(TESTERRORLIST01 + ucIndex));
        //文字列に NULL を格納
        GWGetVmt(GStaticText_VMT, pSText)->SetString(pSText, "");
    }
    //リングバッファをクリア
    CLEAR_RECODE(g_pRecodeHeader_Error, ERROR_HISTORY_NODE);
    //履歴ファイルをクリア
    CLEAR_FILE(g_pRecodeHeader_Error, ERROR_HISTORY_NODE,
ERROR_HISTORY_FILE_NAME);
    //発生中エラー表示欄をクリア
    pSText = (GStaticText*)GWGetVmt(GPanel_VMT, pSelf)->GetChild(pSelf,
TEXTERRORINFO);
    GWGetVmt(GStaticText_VMT, pSText)->SetString(pSText, "");
}
return G_TRUE;
}

```



## 4-2-4 GPanel\_OperDataManagement.c

ここでは 3 章で説明した画面のコールバック関数、および①～③のコントロールに関する処理について、説明します。



## 画面のコールバック関数 OnUpdateDB

①のテキストボックス、④のプログレスバーの表示更新は画面のコールバック関数 OnUpdateDB で実施しています。値に変化のあった場合にのみ、表示更新します。

```
void GPanel_OperDataManagement_OnUpdateDB( void *pSelf, void *pParam )
{
    GUInt8      ucIndex;
    USHORT      usData;
    GDBChgMgr   *pGDBChgMgr;
    GValue      gvValue;
    GTextBox     *pText;
    GProgressBar *pProgBar;
}
```

## GENWARE3 アプリケーション設計ガイド

```

GUIInt16    usValue;

pGDBChgMgr = (GDBChgMgr*) pParam;
//値変化を確認する GDB 分ループ
for (ucIndex = NUM_INIT_0; ucIndex <
sizeof(gdbOperDataIds)/sizeof(gdbOperDataIds[NUM_BUFF_0]); ucIndex++) {
    //該当 GDB で値変化があるか確認
    if (GDBChgMgr_IsChangeData(pGDBChgMgr,
gdbOperDataIds[ucIndex].usGdbID)) {
        //値変化があれば該当する GDB から値を読み出し
        GWDBReadData(gdbOperDataIds[ucIndex].usGdbID, &usData,
sizeof(USHORT));
        //単位が%である (単位により値を計算)
        if (gdbOperDataIds[ucIndex].ucPercent) {
            //%表示であれば 1/10 する
            gvValue.fValue = (GFloat)usData / NUM_DATA_10;
            usValue = usData / NUM_DATA_10;
        }else{
            //直値で表示
            gvValue.usValue = usData;
            usValue = usData;
        }
        //表示対象がスタティックテキスト
        if (gdbOperDataIds[ucIndex].usSTextCtrlID) {
            //表示するスタティックテキストのオブジェクトを取得
            pText = (GTextBox*) GWGetVmt(GPanel_VMT, pSelf)->GetChild(pSelf,
gdbOperDataIds[ucIndex].usSTextCtrlID);
            if (NULL != pText) {
                //値を格納
                GWGetVmt(GTextBox_VMT, pText)->SetValue(pText, gvValue);
            }
        }
        //表示対象がプログレスバー
        if (gdbOperDataIds[ucIndex].usProgCtrlID) {
            //プログレスバーのオブジェクトを取得
            pProgBar = (GProgressBar*) GWGetVmt(GPanel_VMT,
pSelf)->GetChild(pSelf, gdbOperDataIds[ucIndex].usProgCtrlID);
            if (NULL != pProgBar) {
                //表示範囲と現在値を格納
                GWGetVmt(GProgressBar_VMT, pProgBar)->SetRange(pProgBar, 0,

```

```

gdbOperDataIds[ucIndex].usProgMaxValue);
                                GWGetVmt(GProgressBar_VMT, pProgBar)->SetValue(pProgBar,
usValue);
                                }
                                }
                                }
                                }
                                return;
                                }

```

### 画面のコールバック関数 OnTimer

③のプログレスバーは、ロギングデータを表示しています。表示更新は OnTimer により実施しています。

```

void GPanel_OperDataManagement_OnTimer( void *pSelf, GUInt16 usTimerID )
{
    //実デバイス読み出し及び GDB 反映
    DataAccessRead(stGDBID, 35);
    //プログレスバー表示反映
    RefreshLoggingData(pSelf);
    return;
}

```

```

void RefreshLoggingData(void *pSelf)
{
    GUInt8          ucHistoryIndex;
    GUInt8          ucDeviceIndex;
    GUInt8          ucTimeIndex;
    GInt8           cTime;
    DEVICE_HISTORY_NODE *pNode;
    DEVICE_CSV_NODE  *pCsvNode;
    GProgressBar     *pProgBar;
    GStaticText      *pSText;
    GTCHAR           szTime[3];

    //ロギング対象が未登録であれば無処理
    if(g_pCsvHeader_Device == NULL) {
        return;
    }
}

```

```

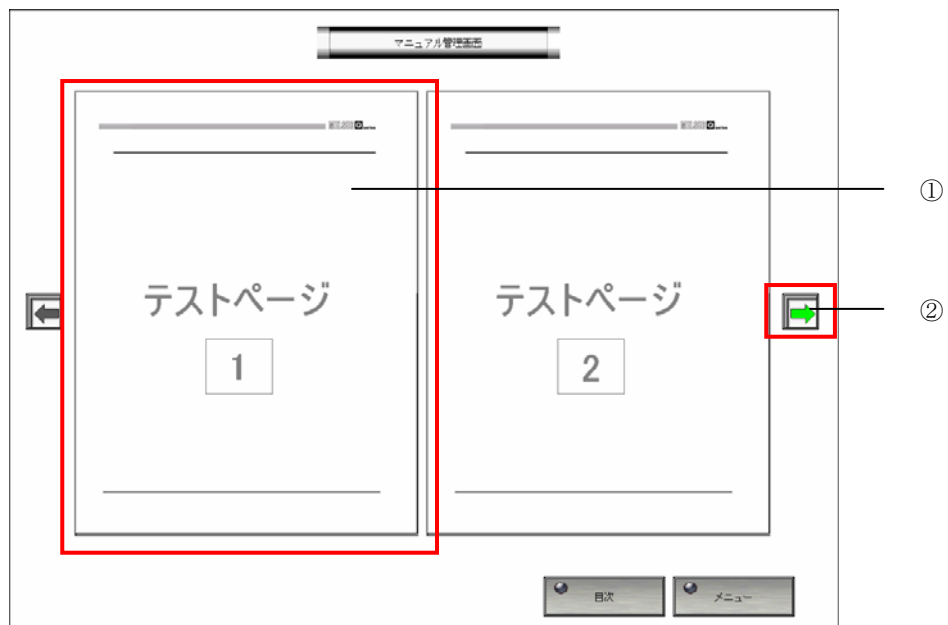
    }
    //ログインするデバイスをループ
    for (pCsvNode = g_pCsvHeader_Device->pTop, ucDeviceIndex = 0; pCsvNode != NULL;
        pCsvNode = pCsvNode->pNext, ucDeviceIndex++) {
        //最後(最新)のノードから取得
        for (pNode = g_pRecodeHeader_Device->pTail, ucHistoryIndex = 0;
            ucHistoryIndex < 24 && NULL != pNode; pNode = pNode->pPrev, ucHistoryIndex++) {
            //該当プログレスバーのオブジェクトを取得
            pProgBar = (GProgressBar*)GWGetVmt(GPanel_VMT,
                pSelf->GetChild(pSelf, progTeble[ucDeviceIndex][ucHistoryIndex]));
            //プログレス更新 最新データは常に最右端に表示する
            if (NULL != pProgBar) {
                GWGetVmt(GProgressBar_VMT, pProgBar)->SetValue(pProgBar,
                    pNode->data.device[ucDeviceIndex]);
            }
        }
    }
    //ログインデータがあるか確認
    pNode = g_pRecodeHeader_Device->pTail;
    if (pNode) {
        //最新ログイン時間を取得
        cTime = pNode->data.usHour;
    } else {
        //ログインデータがない
        cTime = 24;
    }
    ucTimeIndex = 0;
    //時間表示更新するため5回分ループ
    while(ucTimeIndex < 5) {
        memset(szTime, 0, sizeof(szTime));
        sprintf(szTime, "%d", cTime);
        //プログレス下の時間表示用スタティックテキストオブジェクトを取得
        pSText = (GStaticText*)GWGetVmt(GPanel_VMT, pSelf)->GetChild(pSelf,
            stcUpTimeIds[ucTimeIndex]);
        if (NULL != pSText) {
            //プログレス下の時間表示を更新(上段)
            GWGetVmt(GStaticText_VMT, pSText)->SetString(pSText, szTime);
        }
        //プログレス下の時間表示用スタティックテキストオブジェクトを取得
        pSText = (GStaticText*)GWGetVmt(GPanel_VMT, pSelf)->GetChild(pSelf,

```

```
stcDownTimeIds[ucTimeIndex]);  
    if (NULL != pSText) {  
        //プログレス下の時間表示を更新（下段）  
        GWGetVmt(GStaticText_VMT, pSText)->SetString(pSText, szTime);  
    }  
    //カウンタ減算  
    ucTimeIndex++;  
    //24時間表示  
    cTime -= 6;  
    if (cTime < 0){  
        cTime += 24;  
    }  
}  
}
```

## 4-2-5 GManualManagement.c

ここでは 3 章で説明した画面のコールバック関数、および①～②のコントロールに関する処理について、説明します。



## ①マニュアル表示(左)、②ページ切換ボタン

①のマニュアル表示は、②ページ切換ボタンの Onclick の中で表示切換を行います。

```
GInt32 GPanel_ManualManagement_GBUTTON_RIGHTOnClick (GPanel_ManualManagement
*pSelf, GUInt16 usMessage, GLParam lParam, GUParam lUParam)
{
    //操作無し状態を監視するカウンタをリセット
    CompareOperateTime();
    //表示内容を変更（ページ捲り操作）
    PIC_STATUS1++;
    //表示内容更新処理
```

**GENWARE3 アプリケーション設計ガイド**

```

        setManualPicture(pSelf);
        return G_TRUE;
    }

void setManualPicture( GPanel_ManualManagement *pSelf )
{
    GButton *pButton;
    GDesign *pDesign;
    GPicture *pPicture;

    //ページを 0-8 で繰り返すため、状態番号の折り返し対応を行う
    if(PIC_STATUS1 < NUM_FLG_0){
        PIC_STATUS1 = NUM_INIT_0;
    }else if (PIC_STATUS1 > NUM_FLG_8){
        PIC_STATUS1 = NUM_INIT_8;
    }
    if(PIC_STATUS1 == NUM_FLG_0){
        //表示ページが先頭の場合の処理
        //左ボタンのオブジェクトを取得
        pButton = (GButton*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
GBUTTON_LEFT );
        //ON 時意匠イメージを設定
        pDesign = GWGetVmt( GButton_VMT, pButton )->GetOnDesign( pButton );
        pDesign->usImageID = img_LeftNothing;
        GWGetVmt( GButton_VMT, pButton )->SetOnDesign( pButton, pDesign );
        //OFF 時意匠イメージを設定
        pDesign = GWGetVmt( GButton_VMT, pButton )->GetOffDesign( pButton );
        pDesign->usImageID = img_LeftNothing;
        GWGetVmt( GButton_VMT, pButton )->SetOffDesign( pButton, pDesign );
        //右ボタンのオブジェクトを取得
        pButton = (GButton*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
GBUTTON_RIGHT );
        //ON 時意匠イメージを設定
        pDesign = GWGetVmt( GButton_VMT, pButton )->GetOnDesign( pButton );
        pDesign->usImageID = img_RightMove;
        GWGetVmt( GButton_VMT, pButton )->SetOnDesign( pButton, pDesign );
        //OFF 時意匠イメージを設定
        pDesign = GWGetVmt( GButton_VMT, pButton )->GetOffDesign( pButton );
        pDesign->usImageID = img_RightMove;
        GWGetVmt( GButton_VMT, pButton )->SetOffDesign( pButton, pDesign );
    }
}

```

```

} else if (PIC_STATUS1 == NUM_FLG_8) {
    //表示ページが最後の場合の処理
    //左ボタンのオブジェクトを取得
    pButton = (GButton*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
GBUTTON_LEFT );
    //ON 時意匠イメージを設定
    pDesign = GWGetVmt( GButton_VMT, pButton )->GetOnDesign( pButton );
    pDesign->usImageID = img_LeftMove;
    GWGetVmt( GButton_VMT, pButton )->SetOnDesign( pButton, pDesign );
    //OFF 時意匠イメージを設定
    pDesign = GWGetVmt( GButton_VMT, pButton )->GetOffDesign( pButton );
    pDesign->usImageID = img_LeftMove;
    GWGetVmt( GButton_VMT, pButton )->SetOffDesign( pButton, pDesign );
    //右ボタンのオブジェクトを取得
    pButton = (GButton*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
GBUTTON_RIGHT );
    //ON 時意匠イメージを設定
    pDesign = GWGetVmt( GButton_VMT, pButton )->GetOnDesign( pButton );
    pDesign->usImageID = img_RightNothing;
    GWGetVmt( GButton_VMT, pButton )->SetOnDesign( pButton, pDesign );
    //OFF 時意匠イメージを設定
    pDesign = GWGetVmt( GButton_VMT, pButton )->GetOffDesign( pButton );
    pDesign->usImageID = img_RightNothing;
    GWGetVmt( GButton_VMT, pButton )->SetOffDesign( pButton, pDesign );
} else {
    //表示ページが 1-7 の場合の処理
    //左ボタンのオブジェクトを取得
    pButton = (GButton*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
GBUTTON_LEFT );
    //ON 時意匠イメージを設定
    pDesign = GWGetVmt( GButton_VMT, pButton )->GetOnDesign( pButton );
    pDesign->usImageID = img_LeftMove;
    GWGetVmt( GButton_VMT, pButton )->SetOnDesign( pButton, pDesign );
    //OFF 時意匠イメージを設定
    pDesign = GWGetVmt( GButton_VMT, pButton )->GetOffDesign( pButton );
    pDesign->usImageID = img_LeftMove;
    GWGetVmt( GButton_VMT, pButton )->SetOffDesign( pButton, pDesign );
    //右ボタンのオブジェクトを取得
    pButton = (GButton*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
GBUTTON_RIGHT );

```

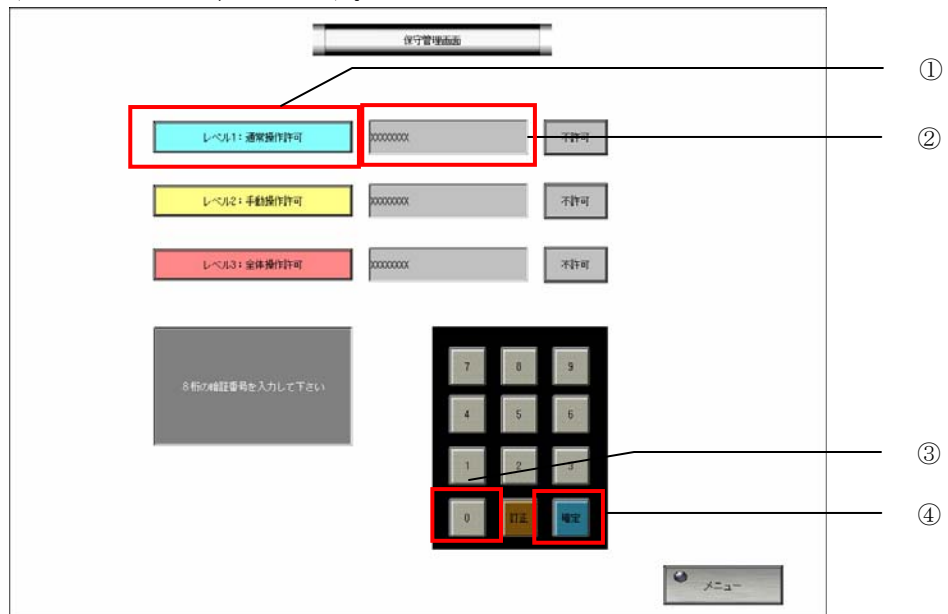


```
//ON 時意匠イメージを設定
pDesign = GWGetVmt( GButton_VMT, pButton )->GetOnDesign( pButton );
pDesign->usImageID = img_RightMove;
GWGetVmt( GButton_VMT, pButton )->SetOnDesign( pButton, pDesign );
//OFF 時意匠イメージを設定
pDesign = GWGetVmt( GButton_VMT, pButton )->GetOffDesign( pButton );
pDesign->usImageID = img_RightMove;
GWGetVmt( GButton_VMT, pButton )->SetOffDesign( pButton, pDesign );
}

//表示イメージ (左) を現在の状態番号で描画
pPicture = (GPicture*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
GPIC_LEFT );
GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture, PIC_STATUS1 );
//表示イメージ (右) を現在の状態番号で描画
pPicture = (GPicture*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
GPIC_RIGHT );
GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture, PIC_STATUS1 );
}
```

## 4-2-6 GPanel\_Maintenance.c

ここでは 3 章で説明した画面のコールバック関数、および①～④のコントロールに関する処理について、説明します。



## ①セキュリティレベルボタン

①のボタンの OnClick により、②のテキストボックスの有効／無効を切り換える処理を実施しています。

```
GInt32 G_PANEL_MAINTENANCE_BTNMESSAGELEVEL1OnClick(GPanel_Maintenance *pSelf,
GUInt16 usMessage, GLParam lParam, GUParam lUParam)
{
    GTextBox *pText;

    //操作無し状態を監視するカンマをリセット
    CompareOperateTime();
    //操作履歴に該当ボタンの操作を追加
    DoOperate(BTNMESSAGELEVEL1, pSelf);
}
```

```

//セキュリティレベルを確認(レベル1以上で無ければ警告を表示)
if (G_TRUE == CheckOperateLevel(pSelf, Lebel_NON)) {
    //セキュリティレベルが2もしくは3であればテキストボックスを無効とする
    if ((g_ucEditLevel == Lebel_2) || (g_ucEditLevel == Lebel_3)) {
        //テキストボックスのオブジェクトを取得
        pText = (GTextBox*)GWGetVmt(GPanel_VMT, pSelf)->GetChild( pSelf,
        PasswordTextIDs[g_ucEditLevel] );
        //入力欄を無効とする
        GWGetVmt(GTextBox_VMT, pText)->SetFocusStatus(pText, G_FALSE);
        GWGetVmt(GTextBox_VMT, pText)->SetDisableColor(pText, RGB16(0xC0,
        0xC0, 0xC0));
    }
    //テキストボックスのオブジェクトを取得
    pText = (GTextBox*)GWGetVmt(GPanel_VMT, pSelf)->GetChild( pSelf,
    TEXTPASSWORDLEVEL1 );
    //入力欄を有効とする
    GWGetVmt(GTextBox_VMT, pText)->SetFocusStatus(pText, G_TRUE);
    GWGetVmt(GTextBox_VMT, pText)->SetDisableColor(pText, RGB16(0xFF, 0xFF,
    0xFF));
    g_ucEditLevel = Lebel_1;
}
return G_TRUE;
}

```

### ③テンキーの「0」ボタン

③のボタンの OnClick により、②などのテキストボックスに 0 を表示します。

```

GInt32 GPanel_MAINTENANCE_BTNNUM0OnClick(GPanel_Maintenance *pSelf, GUInt16
usMessage, GLParam lParam, GUPParam lUParam)
{
    //操作無し状態を監視するカウンタをリセット
    CompareOperateTime();
    //テキストボックスに0を追加表示
    ShowPassword(NUM_KEY_0, pSelf);
    return G_TRUE;
}

void ShowPassword(GUInt8 ucNum, GPanel_Maintenance *pSelf)
{

```

```

GTextBox *pText;
GCHAR szTextOld[NUM_BUFF_10];
GCHAR szTextNew[NUM_BUFF_10];

//セキュリティレベルが選択されていなければ無処理
if (g_ucEditLevel == Lebel_NON) {
    return;
}

//テキストボックスのオブジェクトを取得
pText = (GTextBox*)GWGetVmt(GPanel_VMT, pSelf)->GetChild( pSelf,
PasswordTextIDs[g_ucEditLevel] );
//現在表示（入力）されている文字列を取得
GWGetVmt( GTextBox_VMT, pText )->GetString(pText, szTextOld,
sizeof(szTextOld));
//「O」を文字追加
sprintf(szTextNew, "%s%d", szTextOld, ucNum);
//テキストボックスに反映
GWGetVmt( GTextBox_VMT, pText )->SetString(pText, szTextNew);
}

```

#### ④テンキーの「確定」ボタン

④のボタンの OnClick により、②などのテキストボックスに入力された文字列を確定し、パスワードとして設定された文字列と一致するか判定します。

```

GInt32 GPANEL_MAINTENANCE_BTNNUMOKOnClick(GPanel_Maintenance *pSelf, GUInt16
usMessage, GLParam lParam, GUPParam lUPParam)
{
    GTextBox *pText;
    GButton *pButton;
    GDesign *pDesign;
    GCHAR szPassword[NUM_BUFF_9];
    GInt8 ucIndex;
    static GInt8 ucFailedCnt = 0;

    //操作無し状態を監視するカウンタをリセット
    CompareOperateTime();
    //セキュリティレベルが選択されているか確認
    if (g_ucEditLevel != Lebel_NON) {

```

```

//テキストボックスのオブジェクトを取得
pText = (GTextBox*)GWGetVmt(GPanel_VMT, pSelf)->GetChild( pSelf,
PasswordTextIDs[g_ucEditLevel] );
//入力されている文字列を取得
GWGetVmt( GTextBox_VMT, pText )->GetString(pText, szPassword,
sizeof(szPassword));
//パスワード判別 (正しい場合)
if (strncmp(szPassword, szPasswordTbl[g_ucEditLevel], NUM_FLG_8) == 0) {
//パスワード入力欄を Disable にする
GWGetVmt( GTextBox_VMT, pText )->SetFocusStatus(pText, G_FALSE);
GWGetVmt( GTextBox_VMT, pText )->SetDisableColor( pText, RGB16( 0xC0,
0xC0, 0xC0 ) );
//現在レベルより以下のレベル許可に表示
for(ucIndex = g_ucEditLevel; ucIndex > NUM_FLG_0; ucIndex--) {
//ボタンのオブジェクトを取得し属性を変更
pButton = (GButton*)GWGetVmt(GPanel_VMT, pSelf)->GetChild( pSelf,
PermitBtnIDs[ucIndex] );
GWGetVmt( GButton_VMT, pButton )->SetStringID(pButton,
ID_STRING_MaintenancePermission );
pDesign = GWGetVmt( GButton_VMT,
pButton )->GetOffDesign(pButton);
pDesign->gbBrush.gcBackColor = RGB16(0x00, 0xFF, 0xFF);
GWGetVmt( GTextBox_VMT, pText )->SetString(pText, "");
GWGetVmt( GButton_VMT, pButton )->SetOffDesign(pButton, pDesign);
}
//現在レベルより上のレベルを不許可に表示
for(ucIndex = g_ucEditLevel + NUM_INIT_1; ucIndex <= Lebel_3;
ucIndex++) {
//ボタンのオブジェクトを取得し属性を変更
pButton = (GButton*)GWGetVmt(GPanel_VMT, pSelf)->GetChild( pSelf,
PermitBtnIDs[ucIndex] );
GWGetVmt( GButton_VMT, pButton )->SetStringID(pButton,
ID_STRING_MaintenanceNotpermitted );
pDesign = GWGetVmt( GButton_VMT,
pButton )->GetOffDesign(pButton);
pDesign->gbBrush.gcBackColor = RGB16(0xC0, 0xC0, 0xC0);
GWGetVmt( GTextBox_VMT, pText )->SetString(pText, "");
GWGetVmt( GButton_VMT, pButton )->SetOffDesign(pButton, pDesign);
}
//各変数の初期化

```

```

        ucFailedCnt = NUM_INIT_0;
        g_ucPermitLevel = g_ucEditLevel;
        g_ucEditLevel = Lebel_NON;
    }else{
        //不正なパスワード
        GWGetVmt( GTextBox_VMT, pText )->SetString(pText, "");
        ucFailedCnt++;
        //3 回連続入力ミス
        if(ucFailedCnt > NUM_FLG_2){
            //パスワード入力欄を Disable にする
            pText = (GTextBox*)GWGetVmt(GPanel_VMT, pSelf)->GetChild( pSelf,
PasswordTextIDs[g_ucEditLevel] );
            GWGetVmt( GTextBox_VMT, pText )->SetFocusStatus(pText, G_FALSE);
            GWGetVmt( GTextBox_VMT, pText )->SetDisableColor( pText,
RGB16( 0xC0, 0xC0, 0xC0 ) );
            g_ucEditLevel = Lebel_NON;
            ucFailedCnt = NUM_INIT_0;
            //パスワード入力を 3 回間違えました。
            memset(g_szErrorMsg, NUM_INIT_0, sizeof(g_szErrorMsg));
            strcpy(g_szErrorMsg,
GRCLoadString(ID_STRING_PasswordErrorThreeTime));
        }else{
            //パスワードが間違えます。
            memset(g_szErrorMsg, NUM_INIT_0, sizeof(g_szErrorMsg));
            strcpy(g_szErrorMsg, GRCLoadString(ID_STRING_PasswordError));
        }
        //メッセージを表示
        GActScreenOpenWindow( pSelf, SHOWGWINDOW_MESSAGE );
    }
}
return G_TRUE;
}

```

## 4-2-7 GPanel\_OperationHistory.c

ここでは 3 章で説明した画面のコールバック関数、および①～④のコントロールに関する処理について、説明します。



## 画面のコールバック関数 OnCreate

①の操作履歴一覧(リストコントロール)への文字列挿入は、OnCreate により実施しています。

```
void GPanel_OperationHistory_OnCreate( void *pSelf, GBaseWindow *pParent, GInt16
*pReturnValue )
{
    OPERATE_HISTORY_NODE* pNode;
    GUIInt8  szText[NUM_BUFF_128] = {NUM_INIT_0};
    GList *pList;

    //リストコントロールのオブジェクトを取得
```

```

    pList = (GList*)GWGetVmt(GPanel_VMT, pSelf)->GetChild(pSelf,
LISTOPERATIONHISTORYINFO);
    //操作履歴バッファの先頭ポインタ
    pNode = g_pRecodeHeader_Operate->pTop;
    //操作履歴情報を表示
    while(pNode) {
        //表示内容を成型
        sprintf((char *)szText, "%d/%02d/%02d¥t¥t%02d:%02d:%02d¥t¥t%s",
            pNode->data.date.usYear, pNode->data.date.usMonth,
pNode->data.date.usDay,
            pNode->data.time.usHour, pNode->data.time.usMinute,
pNode->data.time.usSecond,
            GRCLoadString(pNode->data.usStrID));
        //リストコントロールに文字列を追加
        GWGetVmt(GList_VMT, pList)->AddString(pList, (char *)szText);
        //次の履歴データ用ポインタを取得
        pNode = pNode->pNext;
        //文字列バッファクリア
        memset(szText, 0, sizeof(szText));
    }
    return;
}

```

## ②履歴クリアボタン

操作履歴のクリアは、②のボタンの OnClick により実施しています。

```

GInt32 GPanel_OperationHistory_BTNHISTORYCLEAROnClick(GPanel_OperationHistory
*pSelf, GUInt16 usMessage, GLParam lParam, GUParam lUParam)
{
    GList *pList;

    //操作無し状態を監視するカウンタをリセット
    CompareOperateTime();
    //セキュリティレベルを確認(レベル1以上で無ければ警告を表示)
    if(G_TRUE == CheckOperateLevel(pSelf, Lebel_1)) {
        //リストコントロールのオブジェクトを取得
        pList = (GList*)GWGetVmt(GPanel_VMT, pSelf)->GetChild(pSelf,
LISTOPERATIONHISTORYINFO);
    }
}

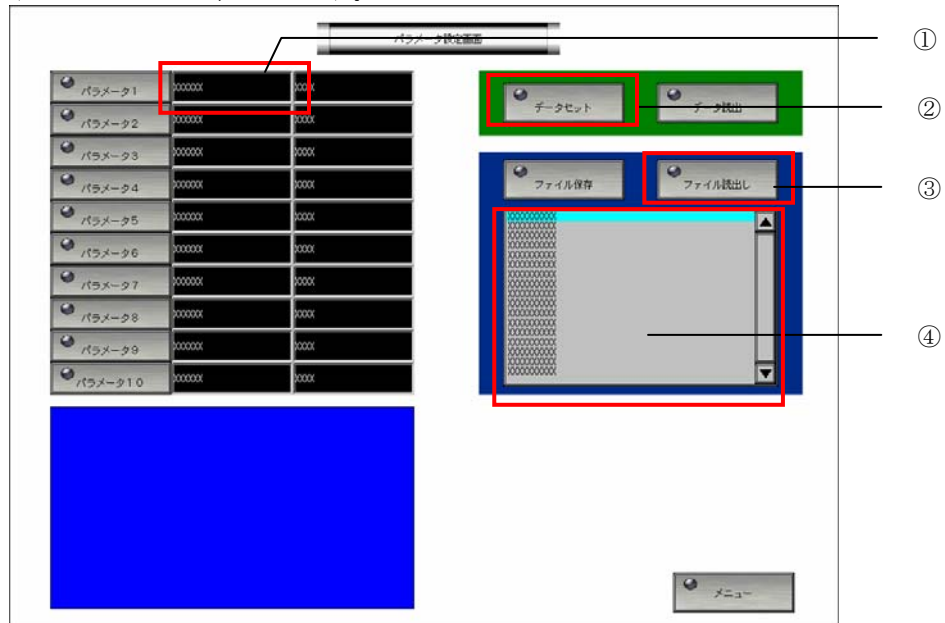
```



```
//リスト文字列を全てクリア
GWGetVmt (GList_VMT, pList)->RemoveAllStrings(pList);
//操作履歴ファイル内容も削除
CLEAR_RECODE(g_pRecodeHeader_Operate, OPERATE_HISTORY_NODE);
CLEAR_FILE(g_pRecodeHeader_Operate, OPERATE_HISTORY_NODE,
OPEPATE_HISTORY_FILE_NAME);
}
return G_TRUE;
}
```

## 4-2-8 GPanel\_ParameterSetting.c

ここでは 3 章で説明した画面のコールバック関数、および①～④のコントロールに関する処理について、説明します。



## ①パラメータ 1

①のテキストボックスの OnClick で、フォーカスをテキストボックスにあて、テンキー画面を表示する処理を実施しています。

```
GInt32 GPANEL_PARAMETERSETTING_TBOXDECIMAL010nClick (GPanel_ParameterSetting
*pSelf, GUInt16 usMessage, GLParam lParam, GUParam lUParam)
{
    GTextBox *pText;
    GButton *pButton;

    //操作無し状態を監視するカウンタをリセット
    CompareOperateTime();
    //セキュリティレベルを確認(レベル3で無ければ警告を表示)
```

```

if(G_TRUE == CheckOperateLevel(pSelf, Lebel_3)){
    //未選択状態であるか確認
    if(INPUT_FLAG == NUM_FLG_0){
        //パラメータ1の10進を選択
        TACH_FLAG = NUM_INIT_1;
        //テキストボックスのオブジェクトを取得
        pText = (GTextBox*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
(TBOXDECIMAL01 + TACH_FLAG - NUM_DATA_1));
        //文字列をクリア
        GWGetVmt( GTextBox_VMT, pText )->SetString(pText, "");
        //フォーカスを当てる
        GWGetVmt( GTextBox_VMT, pText )->SetFocusColor( pText, RGB16( 0xFF,
0xFF, 0xFF ) );
        //パラメータ1ボタンのオブジェクトを取得
        pButton = (GButton*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
(BTNPARAMETERS01 + (TACH_FLAG - NUM_DATA_1) % NUM_MOD_10) );
        //ボタンをON状態とする
        GWGetVmt( GButton_VMT, pButton )->SetStatus( pButton, GBTN_STATE_ON );
    }
    //10キーウィンドウを表示
    GActScreenOpenWindow( pSelf, SHOWGWINDOW_NUMERICKEYPAD );
}
return G_TRUE;
}

```

## ②データセットボタン

②データセットボタンの OnClick で、パラメータ1～10 に設定された数値を、実デバイスに書き込む処理を行います。

```

GInt32 G_PANEL_PARAMETERSETTING_BTNDATASETOnClick(GPanel_ParameterSetting *pSelf,
GUInt16 usMessage, GLParam ILParam, GUParam IUParam)
{
    int i;
    unsigned short gtDBDecima[NUM_BUFF_10];
    unsigned short gtDBParameters[NUM_BUFF_10];
    GTextBox *pText;
    GTCHAR tempText[NUM_BUFF_8];

    //操作無し状態を監視するカウンタをリセット

```

## GENWARE3 アプリケーション設計ガイド

```

CompareOperateTime();
//操作履歴に該当の操作を追加
DoOperate(BTNDATASET, pSelf);
//セキュリティレベルを確認(レベル3で無ければ警告を表示)
if(G_TRUE == CheckOperateLevel(pSelf, Lebel_3)){
    //GDB読み出し
    GWDBReadData( ID_GDB_DECIMAL, &gtDBDecima, sizeof( gtDBDecima ));
    GWDBReadData( ID_GDB_HEXADECIMAL, &gtDBParameters,
sizeof( gtDBParameters ));
    //読み出した内容を10進/16進入力テキストボックスに反映
    for(i = NUM_INIT_0; i < NUM_FLG_10; i++){
        //テキストボックスのオブジェクトを取得
        pText = (GTextBox*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
(TBOXDECIMAL01 + i));
        //入力内容を取り出し
        GWGetVmt( GTextBox_VMT, pText )->GetString(pText, tempText,
sizeof( tempText ));
        //データエリアに格納(10進用)
        gtDBDecima[i] = (unsigned short)atoi(tempText);
        //テキストボックスのオブジェクトを取得
        pText = (GTextBox*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
(TBOXPARAMETERS01 + i));
        //入力内容を取り出し
        GWGetVmt( GTextBox_VMT, pText )->GetString(pText, tempText,
sizeof( tempText ));
        //データエリアに格納(16進用)
        gtDBParameters[i] = (unsigned short)strtol(tempText, NULL, 16);
    }
    //GDBに書き込み
    GWDBWriteData( ID_GDB_DECIMAL, &gtDBDecima, sizeof( gtDBDecima ),
G_TRUE );
    GWDBWriteData( ID_GDB_HEXADECIMAL, &gtDBParameters,
sizeof( gtDBParameters ), G_TRUE );
    //実デバイスに反映
    sprintf(stGDBID[0].cObjectName, "%s", "GDB_DECIMAL");
    stGDBID[0].usGDBID = ID_GDB_DECIMAL;
    stGDBID[0].sDataSize = NUM_BUFF_10;
    sprintf(stGDBID[1].cObjectName, "%s", "GDB_HEXADECIMAL");
    stGDBID[1].usGDBID = ID_GDB_HEXADECIMAL;
    stGDBID[1].sDataSize = NUM_BUFF_10;

```

## GENWARE3 アプリケーション設計ガイド

```

        DataAccessWrite(stGDBID, 2);
    }
    return G_TRUE;
}

```

## ③ファイル読出しボタン

③ファイル読出しボタンの OnClick で、ファイル一覧で選択されたファイルからデータを読み出し、パラメータ 1～10 の表示に反映します。

```

GInt32 GPANEL_PARAMETERSETTING_BTNLOADOnClick(GPanel_ParameterSetting *pSelf,
GUInt16 usMessage, GLParam lParam, GUParam lUParam)
{
    GUInt16 i, j, strLen;
    GTextBox *pText;
    GTCHAR parametersNum[NUM_BUFF_10][NUM_BUFF_5];
    GTCHAR tempText[NUM_BUFF_8];
    int decimalNum[NUM_BUFF_10];
    GList *pList;
    GInt16 nResult;
    GPicture *pPicture;

    //操作無し状態を監視するカウンタをリセット
    CompareOperateTime();
    //操作履歴ログに該当ボタンの操作を追加
    DoOperate(BTNLOAD, pSelf);
    //セキュリティレベルを確認(レベル3で無ければ警告を表示)またパラメータ1～10の何れかが選択されている
    if((G_TRUE == CheckOperateLevel(pSelf, Lebel_3)) && (selectLine < NUM_FLG_10
&& selectLine > - NUM_DATA_1)){
        //ファイル(Recipe?.csv)を読み出し
        ReadRecipeCsvFile(selectLine, decimalNum, parametersNum);
        //読み出し内容をテキストボックスに反映
        for(i = NUM_INIT_0; i < NUM_FLG_10; i++){
            //テキストボックスのオブジェクトを取得
            pText = (GTextBox*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
(TBOXDECIMAL01 + i));
            //値の範囲チェック。範囲を超えている場合は0とする
            if(decimalNum[i] > NUM_NEGATIVE_32769 && decimalNum[i] <
NUM_FLG_32768){

```

```

        //10 進文字列に変換してテキストボックスに反映
        sprintf(tempText, "%d", decimalNum[i]);
        GWGetVmt( GTextBox_VMT, pText )->SetString(pText, tempText);
    }else{
        //0 をテキストボックスに反映
        GWGetVmt( GTextBox_VMT, pText )->SetString(pText, "0");
    }
    //テキストボックスのオブジェクトを取得
    pText = (GTextBox*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
(TBOXPARAMETERS01 + i));
    strLen = (GUInt16)strlen(parametersNum[i]);
    //格納文字列が0～9、A～Fの範囲か確認
    for(j = NUM_INIT_0; j < strLen; j++){
        //範囲外があれば処理中断
        if( !((parametersNum[i][j] > NUM_FLG_47 && parametersNum[i][j] <
NUM_FLG_58) ||
            (parametersNum[i][j] > NUM_FLG_64 && parametersNum[i][j] <
NUM_FLG_71) ||
            (parametersNum[i][j] > NUM_FLG_96 && parametersNum[i][j] <
NUM_FLG_103))) break;
        }
        //文字列をテキストボックスに反映。不正文字があれば0を反映
        if(j == strLen){
            GWGetVmt( GTextBox_VMT, pText )->SetString(pText,
parametersNum[i]);
        }else{
            GWGetVmt( GTextBox_VMT, pText )->SetString(pText, "0");
        }
    }
    //リストのオブジェクトを取得
    pList = (GList*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
LISTSHOWFILEINFO );
    //選択されている行を取得
    nResult = GWGetVmt( GList_VMT, pList )->SetCurrentSelect( pList,
selectLineBak );
    //選択行に相当するイメージの表示
    pPicture = (GPicture*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
PCTRECIPEIMAGE );
    GWGetVmt( GPicture_VMT, pPicture )->SetStatus( pPicture, selectLine+1 );
}
return G_TRUE;
}

```

## ④ファイル一覧

④ファイル一覧の OnClick で、読み書きの対象とするファイルの選択をします。ディレクトリを選択した場合は、リストを更新し、選択したディレクトリに含まれるファイルを一覧表示します。

```
GInt32 GPanel_ParameterSetting_LISTSHOWFILEINFOOnClick(GPanel_ParameterSetting
*pSelf, GUInt16 usMessage, GLParam lParam, GUParam lUParam)
{
    GList *pList;
    GTCCHAR szText[128];          // 文字列
    short sLen;
    GTCCHAR recString[100][NUM_BUFF_40];
    int i;
    short sFileCnt;
    short sPathSize;

    memset(recString, 0, sizeof(recString));
    //操作無し状態を監視するカウンタをリセット
    CompareOperateTime();
    //セキュリティレベルを確認(レベル3で無ければ警告を表示)
    if(G_TRUE == CheckOperateLevel(pSelf, Lebel_3)){
        //リストコントロールオブジェクトを取得
        pList = (GList*)GWGetVmt(GPanel_VMT, pSelf)->GetChild( pSelf,
LISTSHOWFILEINFO );
        //選択行を取得
        selectLineBak = (GUInt16)GWGetVmt( GList_VMT,
pList )->GetCurrentSelect( pList );
        //未選択であれば処理終了
        if(selectLineBak < 0){
            selectLine = -1;
            return G_TRUE;
        }
        //リストコントロールの文字列取得
        GWGetVmt( GList_VMT, pList )->GetString( pList, selectLineBak, szText,
sizeof( szText ) );
        //ディレクトリを選択中の場合
        if(strncmp("<..>", szText, 4) == 0){
            //一つ上のディレクトリパスを生成
            sPathSize = strlen(cFullIPath);
```

```

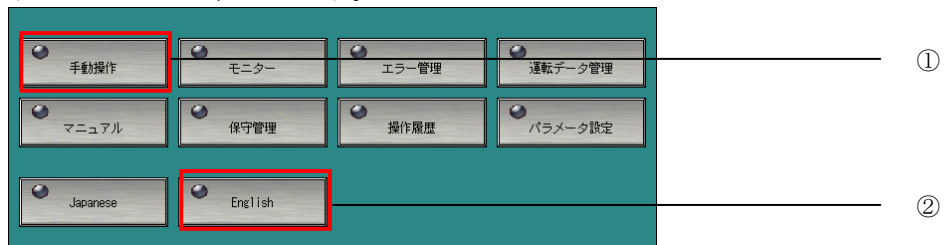
        for(i = sPathSize-1; i >= 0; i--){
            if(cFullPath[i] == '/') {
                cFullPath[i] = 0;
                break;
            }
        }
        //リスト文字列を消去
        GWGetVmt( GList_VMT, pList )->RemoveAllStrings( pList );
        //生成パスにあるレシピファイルを検索
        sFileCnt = LoadRecipeCsvFile( recString );
        //リストに追加
        for(i = NUM_INIT_0; i < sFileCnt; i++){
            GWGetVmt( GList_VMT, pList )->AddString( pList, recString[i] );
        }
        selectLine = -1;
        //Recipexx.csv 選択の場合
    }else if(strncmp("Recipe", szText, 6) == 0){
        //レシピ番号を取り出し
        selectLine = szText[7] - '0';
    }else{
        //ディレクトリ選択された場合
        sLen = strlen(szText);
        szText[sLen-1] = 0;
        //選択ディレクトリを含むフルパスを生成
        sprintf(cFullPath, "%s/%s", cFullPath, &szText[1]);
        //リスト文字列を消去
        GWGetVmt( GList_VMT, pList )->RemoveAllStrings( pList );
        //選択ディレクトリ内のレシピファイルを検索
        sFileCnt = LoadRecipeCsvFile( recString );
        //リストに追加
        for(i = NUM_INIT_0; i < sFileCnt; i++){
            GWGetVmt( GList_VMT, pList )->AddString( pList, recString[i] );
        }
        selectLine = -1;
    }
}
return G_TRUE;
}

```



## 4-2-9 GWindow\_Menu.c

ここでは 3 章で説明した画面のコールバック関数、および①～④のコントロールに関する処理について、説明します。



## ①手動操作ボタン

①のボタンの OnClick で、パネルを手動操作画面に切り換えます。またメニュー画面を閉じます。

```
GInt32 GWINDOW_MENU_BTNMANUALOPERATIONOnClick(GWindow_Menu *pSelf, GUInt16
usMessage, GLParam lParam, GUPParam lUPParam)
{
    //メニュー切替処理
    SelectMenuBtn(pSelf, BTNMANUALOPERATION, SHOWGPANEL_MANUALOPERATION);

    return G_TRUE;
}

void SelectMenuBtn(void *windowObject, GUInt16 controlId, GInt32 changePageID)
{
    GFrame *pFrame = NULL;
    GWindow *pWindow = NULL;

    //フレームオブジェクト取得
    pFrame = (GFrame*)GWGetVmt(GScreen_VMT,
GWGetScreen())->GetFrame(GWGetScreen());
    //操作無し状態を監視するカウンタをリセット
    CompareOperateTime();
    //操作履歴に該当ボタンの操作を追加
    DoOperate(controlID, windowObject);
    //画面切替
    GActScreenShowPanel(windowObject, changePageID);
}
```

```

        if(pFrame != NULL) {
            //ウィンドウオブジェクト取得
            pWindow = (GWindow*)GWGetVmt(GFrame_VMT, pFrame)->GetChild(pFrame,
ID_GWINDOW_MENU + 1);
            //メニューウィンドウを破棄
            if(pWindow != NULL) {
                GWGetVmt(GFrame_VMT, pFrame)->DeleteChild(pFrame, pWindow);
            }
        }
    }
}

```

## ②English ボタン

②のボタンの OnClick で、ロケールを英語に切り換える処理を実施します。

```

GInt32 GWINDOW_MENU_BTNENGLISHOnClick(GWindow_Menu *pSelf, GUInt16 usMessage,
GLParam lParam, GUParam lUParam)
{
    GPanel *pPanel;
    GButton *pButton1;
    GButton *pButton2;

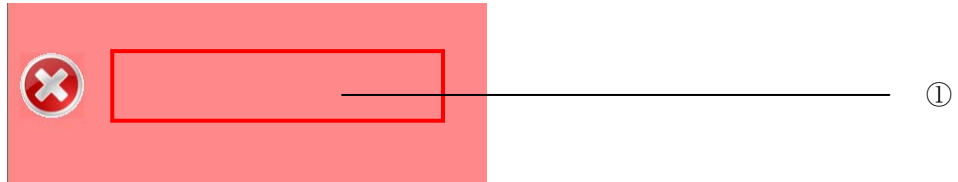
    //操作無し状態を監視するカウンタをリセット
    CompareOperateTime();
    //操作履歴カウンタに該当ボタンの操作を追加
    DoOperate(BTNENGLISH, pSelf);
    //パネルオブジェクトを取得
    pPanel = GWGetVmt(GScreen_VMT, GWGetScreen())->GetPanel(GWGetScreen());
    //ロケールを英語に切替
    GRCSetLocale(ID_LOC_LOCALE1);
    //ボタンオブジェクトを取得
    pButton1 = (GButton*)GWGetVmt(GWindow_Menu_VMT, pSelf)->GetChild(pSelf,
BTNENGLISH);
    //「English」ボタンにフォーカスセット
    GWGetVmt(GButton_VMT, pButton1)->SetFocusStatus(pButton1, G_TRUE);
    //ボタンオブジェクトを取得
    pButton2 = (GButton*)GWGetVmt(GWindow_Menu_VMT, pSelf)->GetChild(pSelf,
BTNJAPANESE);
    //「Japanese」ボタンにフォーカス無効

```

```
GWGetVmt( GButton_VMT, pButton2 )->SetFocusStatus( pButton2, G_FALSE );  
//強制再描画発行  
GWGetVmt(GScreen_VMT, GWGetScreen() )->AddRefreshRect( GWGetScreen(), NULL,  
GW_REFRESH_BGNOERASE );  
  
return G_TRUE;  
}
```

## 4-2-10 GWindow\_Message.c

ここでは 3 章で説明した画面のコールバック関数、および①～④のコントロールに関する処理について、説明します。



## ①メッセージ

①のスタティックテキストの OnCreate によりメッセージ文字列の表示、タイマイイベントの発行を実施し、OnTimer によりタイマイイベントを破棄し、メッセージ画面を閉じる処理を実施しています。

```
GInt32 GWINDOW_MESSAGE_STCERRORMESSAGEOnCreate(GWindow_Message *pSelf, GUInt16
usMessage, GLParam lParam, GUParam lUParam)
{
    GStaticText *pSText;

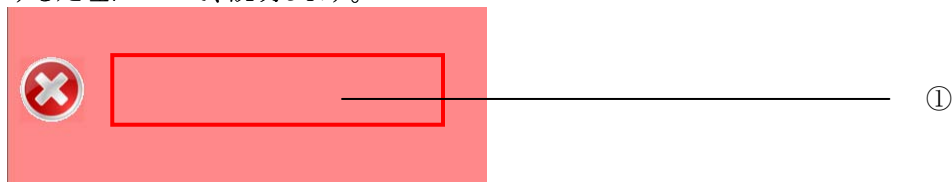
    //テキストボックスのオブジェクトを取得
    pSText = (GStaticText*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
STCERRORMESSAGE );
    //メッセージ文字列をテキストボックスに反映
    GWGetVmt(GStaticText_VMT, pSText )->SetString(pSText, g_szErrorMsg);
    //タイマイイベントを発行 (1500ms)
    GSetTimer( pSText, ID_TIMER_MESSAGE, NUM_TIMER_1500 );
    return G_TRUE;
}

GInt32 GWINDOW_MESSAGE_STCERRORMESSAGEOnTimer (GWindow_Message *pSelf, GUInt16
usMessage, GLParam lParam, GUParam lUParam)
{
    GStaticText *pSText;
    GFrame *pFrame = NULL;
    GWindow *pWindow = NULL;
```

```
//フレームオブジェクトを取得
pFrame = (GFrame*)GWGetVmt(GScreen_VMT,
GWGetScreen())->GetFrame(GWGetScreen());
//テキストボックスのオブジェクトを取得
pSText = (GStaticText*)GWGetVmt(GPanel_VMT, pSelf )->GetChild( pSelf,
STCERRORMESSAGE );
//タイマイベントを破棄
GKillTimer( pSText, ID_TIMER_MESSAGE);
if(pFrame != NULL) {
    //ウィンドウオブジェクトを取得
    pWindow = (GWindow*)GWGetVmt(GFrame_VMT, pFrame)->GetChild(pFrame,
ID_GWINDOW_MESSAGE + 1);
    if(pWindow != NULL) {
        //ウィンドウを破棄
        GWGetVmt(GFrame_VMT, pFrame)->DeleteChild(pFrame, pWindow);
    }
}
return G_TRUE;
}
```

## 4-2-11 GWindow\_Message.c

ここでは 3 章で説明した画面のコールバック関数、および①～④のコントロールに関する処理について、説明します。



## ①メッセージ

①のスタティックテキストの OnCreate では、保守管理画面でパスワードを間違えたときに表示されるメッセージ、操作権限のない部品を操作しようとしたときに表示されるメッセージの表示と、タイマイベントの軌道を行っています。

また、Ontimer により、ウィンドウ表示から 1.5 秒経過すると、自動的にウィンドウを閉じる処理を行っています。

```
GInt32 GWINDOW_MESSAGE_STCERRORMESSAGEOnCreate(GWindow_Message *pSelf, GUInt16
usMessage, GLParam ILParam, GUParam IUParam)
{
    GStaticText *pSText;

    //テキストボックスのオブジェクトを取得
    pSText = (GStaticText*)GWGetVmt( GPanel_VMT, pSelf )->GetChild( pSelf,
STCERRORMESSAGE );
    //メッセージ文字列をテキストボックスに反映
    GWGetVmt(GStaticText_VMT, pSText )->SetString(pSText, g_szErrorMsg);
    //タイマイベントを発行 (1500ms)
    GESetTimer( pSText, ID_TIMER_MESSAGE, NUM_TIMER_1500 );
    return G_TRUE;
}

GInt32 GWINDOW_MESSAGE_STCERRORMESSAGEOnTimer(GWindow_Message *pSelf, GUInt16
usMessage, GLParam ILParam, GUParam IUParam)
{
    GStaticText *pSText;
    GFrame *pFrame = NULL;
```

```
    GWindow *pWindow = NULL;

    //フレームオブジェクトを取得
    pFrame = (GFrame*)GWGetVmt(GScreen_VMT,
GWGetScreen())->GetFrame(GWGetScreen());
    //テキストボックスのオブジェクトを取得
    pSText = (GStaticText*)GWGetVmt(GPanel_VMT, pSelf )->GetChild( pSelf,
STCERRORMESSAGE );
    //タイミイベントを破棄
    GEKillTimer( pSText, ID_TIMER_MESSAGE);
    if(pFrame != NULL) {
        //ウィンドウオブジェクトを取得
        pWindow = (GWindow*)GWGetVmt(GFrame_VMT, pFrame)->GetChild(pFrame,
ID_GWINDOW_MESSAGE + 1);
        if(pWindow != NULL) {
            //ウィンドウを破棄
            GWGetVmt(GFrame_VMT, pFrame)->DeleteChild(pFrame, pWindow);
        }
    }
    return G_TRUE;
}
```

## 4-2-12 GWindow\_NumericKeypad.c

テンキー画面の処理については、「4-2-6 GPanel\_Maintenance.c」に記載した処理を参照してください。





## 第 5 章 アプリケーションの実行

本章では、作成した GUI アプリケーションを C 言語コントローラ上で実行するまでの手順を説明します。

5-1 C 言語コントローラへの転送 .....	5-1
5-2 アプリケーションの起動 .....	5-2

## 5-1 C 言語コントローラへの転送

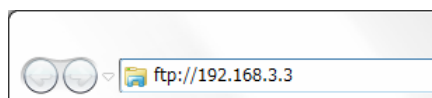
パソコンと C 言語コントローラと ftp 接続し、CW Workbench で作成した out ファイルを、C 言語コントローラに転送します。

ftp 接続には、Windows に添付のエクスプローラや、市販の ftp ツールを使用してください。

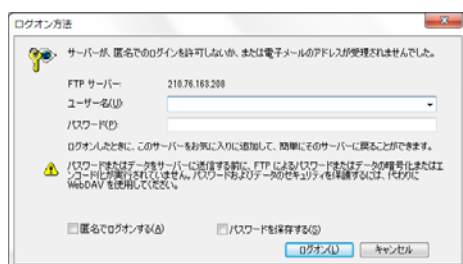
以下では、エクスプローラを例に手順を説明します。

1. C 言語コントローラとパソコンを Ethernet で接続します。
2. エクスプローラのアドレス入力欄に、「ftp://」と入力し、続けて C 言語コントローラの IP アドレスを入力します。

例:ftp://192.168.3.3



3. ユーザー名とパスワードを求めるダイアログが表示されますので、それぞれ入力します。



4. エクスプローラに、C 言語コントローラ内のフォルダが表示されます。
5. out ファイルを、C 言語コントローラの任意のフォルダにコピーします。

### MEMO

- ◆ C 言語コントローラ内の ROM フォルダに out ファイルを保存することをお勧めします。  
C 言語コントローラの電源を OFF しても、ROM フォルダ内のファイルは消えません。
- ◆ 上記の手順は、Windows7 上で実施した場合の手順です。使用する OS やツールにより表示されるダイアログイメージなどは異なります。

## 5-2 アプリケーションの起動

Telnet や市販のターミナルソフトを使用し、アプリケーションを起動します。

ここでは、Telnet を例に手順を説明します。

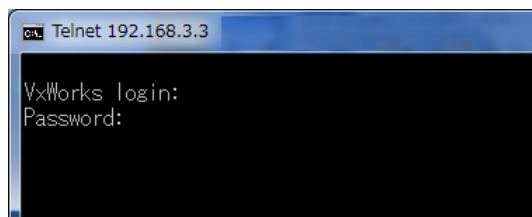
1. Windows のスタートメニューから、コマンドプロンプトを起動します。
2. 「telnet」と入力し、[Enter]キーを押します。「Microsoft Telnet クライアントへようこそ」と表示されます。
3. 「open」と入力し、スペースを入力した後、C 言語コントローラの IP アドレスを入力します。

例: open 192.168.3.3



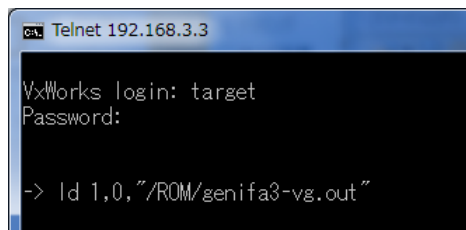
```
ca. コマンド プロンプト - telnet
Microsoft Telnet クライアントへようこそ
エスケープ文字は 'CTRL+]' です
Microsoft Telnet> open 192.168.3.3
```

4. 「VxWorks login:」と表示されるので、ユーザ名を入力してください。続けて、「Password:」と表示されるので、パスワードを入力してください。



```
ca. Telnet 192.168.3.3
VxWorks login:
Password:
```

5. 「ld 1,0,"/ROM/genifa3-vg.out"」と入力します。



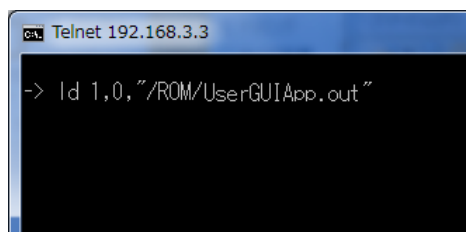
```
C:\> Telnet 192.168.3.3

VxWorks login: target
Password:

-> ld 1,0,"/ROM/genifa3-vg.out"
```

6. 「ld 1,0,"out ファイル"」と入力します。「out ファイル」の部分には、out ファイルをフルパスで指定します。

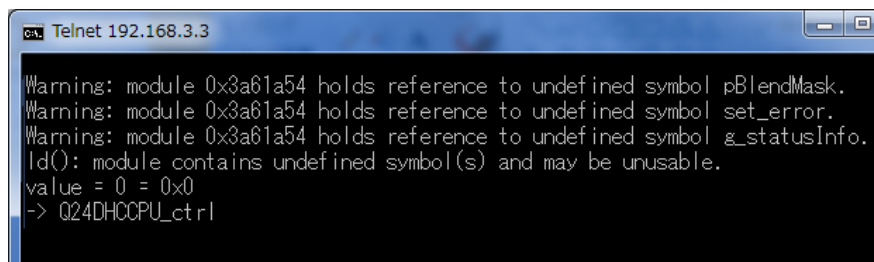
例:ld 1,0,"/ROM/UserGUIApp.out"



```
C:\> Telnet 192.168.3.3

-> ld 1,0,"/ROM/UserGUIApp.out"
```

7. メイン関数名を入力します。  
アプリケーションが起動します。



```
C:\> Telnet 192.168.3.3

Warning: module 0x3a61a54 holds reference to undefined symbol pBlendMask.
Warning: module 0x3a61a54 holds reference to undefined symbol set_error.
Warning: module 0x3a61a54 holds reference to undefined symbol g_statusInfo.
ld(): module contains undefined symbol(s) and may be unusable.
value = 0 = 0x0
-> Q24DHCCPU_ctrl
```

### MEMO

- ◆ Windows7では、初期状態では Telnet がインストールされていません。Telnet をインストールする場合は、[コントロール パネル]の[プログラムと機能]の[Windows の機能の有効化または無効化]から、[Telnet クライアント]をチェック ON してください。
- ◆ Telnet についての詳細は、Windows ヘルプなどを参照してください。

## 付録 1 プロパティ設定一覧

本書において GENSKETCH3 を使用して作成するサンプルアプリケーションのプロパティ設定について説明します。

付 1-1 リソース設定 .....	付 1-1
付 1-2 プロジェクトプロパティ設定 .....	付 1-23
付 1-3 スクリーンプロパティ設定 .....	付 1-23
付 1-4 パネル/ウィンドウプロパティ設定 .....	付 1-24
付 1-5 コントロールプロパティ設定 .....	付 1-28

### MEMO

- ◆ 各プロパティ設定は、初期設定の値からの変更箇所のみを記載しています。
- ◆ 「コールバック関数」欄には、設定を「あり」とする関数名を記載します。

## 付 1-1 リソース設定

## 文字列リソース

## 文字列

リソースデータ名	文字列(ロケール: 日本語)	文字列(ロケール: 英語)
ID_STRING_ManualOperation	手動操作	ManualOperation
ID_STRING_Monitor	モニター	Monitor
ID_STRING_ErrorManagement	エラー管理	ErrorManagement
ID_STRING_OperDataManagement	運転データ管理	OperDataManagement
ID_STRING_ManualManagement	マニュアル	ManualManagement
ID_STRING_Maintenance	保守管理	Maintenance
ID_STRING_OperationHistory	操作履歴	OperationHistory
ID_STRING_Japanese	Japanese	Japanese
ID_STRING_English	English	English
ID_STRING_Menu	メニュー	Menu
ID_STRING_MonitorScreenTitle	モニター画面	Monitor screen
ID_STRING00036	%s	%s
ID_STRING_CycleStopON	サイクル停止 ON	Cycle stop ON
ID_STRING_ForcedStopON	強制停止 ON	Forced stop ON
ID_STRING_TableLeftOverLimit	テーブル 左限オーバー	Table left over limit
ID_STRING_TableRightOverLimit	テーブル 右限オーバー	Table right over limit
ID_STRING_TablePlateExchange	テーブル プレート交換	Table plate exchange
ID_STRING_TableAmpPowerON	テーブルアンプ 電源ON	Table amplifier power ON
ID_STRING_TableAmpServoON	テーブルアンプ サーボON	Table amplifier Servo ON
ID_STRING_OilDropDetection	潤滑油量 低下検出	Oil Drop Detection
ID_STRING_UPSAlarm	UPS警報	UPS alarm
ID_STRING_UPSBlackout	UPS停電	UPS power failure
ID_STRING_CarriageLDirective	搬送台左右 左行指令	Carriage Left Directive
ID_STRING_CarriageRDirective	搬送台左右 右行指令	Carriage Right Directive
ID_STRING_CarriageUDirective	搬送台上下 上行指令	Carriage Up Directive
ID_STRING_CarriageDDirective	搬送台上下 下行指令	Carriage Down Directive
ID_STRING_CarriageLLight	搬送台左右 左限表示灯	Carriage Left Light
ID_STRING_CarriageRLight	搬送台左右 右限表示灯	Carriage Right Light
ID_STRING_CarriageULight	搬送台上下 上限表示灯	Carriage Up Light
ID_STRING_CarriageDLight	搬送台上下 下限表示灯	Carriage Down Light

**GENWARE3 アプリケーション設計ガイド**

ID_STRING_MachineTest	機械テスト	Machine test
ID_STRING_PCTest	PCテスト	PC test
ID_STRING_MonitorInspectionEquipment	検査装置	Inspection equipment
ID_STRING_MonitorAuto	自動	Auto
ID_STRING_MonitorManual	手動	Manual
ID_STRING_MonitorPositioningDevice	位置決め装置	Positioning device
ID_STRING_MonitorMarkingApparatus	刻印装置	Marking apparatus
ID_STRING_MonitorSensor1Abnormality	検査装置 センサー1 異常	Inspection equipment sensor 1 Abnormal
ID_STRING_MonitorSensor2Abnormality	検査装置 センサー2 異常	Inspection equipment sensor 2 abnormal
ID_STRING_MonitorSensor3Abnormality	検査装置 センサー3 異常	Inspection equipment sensor 3 abnormal
ID_STRING_MonitorAbnormalLoad	払出 コンペア 負荷異常	Payout compare abnormal load
ID_STRING_MonitorMotor1Abnormality	位置決め モータ1 異常	Positioning motor 1 Abnormal
ID_STRING_MonitorMotor2Abnormality	位置決め モータ2 異常	Positioning motor 2 Abnormal
ID_STRING_DoorOpen	扉開	Door Open
ID_STRING_ForcedOut	強制排出	Forced Out
ID_STRING_TableRight	テーブル右	Table Right
ID_STRING_TableLeft	テーブル左	Table Left
ID_STRING_CarriageRight	搬送台右	Carriage Right
ID_STRING_CarriageLeft	搬送台左	Carriage Left
ID_STRING_CarriageUp	搬送台上	Carriage Up

付録 1 プロパティ設定一覧

付 1-1 リソース設定

GENWARE3 アプリケーション設計ガイド

ID_STRING_CarriageDown	搬送台下	Carriage Down
ID_STRING_CarriageStop	搬送台停止	Carriage Stop
ID_STRING_MountingMotor1Abnormality	部品取付 モータ1 異常	Parts mounted motor 1 Abnormal
ID_STRING_MountingMotor2Abnormality	部品取付 モータ2 異常	Parts mounted motor 2 Abnormal
ID_STRING_MonitorHeadAbnormality	刻印装置 ヘッド 異常	Marking apparatus abnormal head
ID_STRING_MonitorLaserAbnormality	刻印装置 レーザー 異常	Engraving equipment laser malfunction
ID_STRING_MonitorAlarmRelease	警報解除	Alarm release
ID_STRING_MonitorBuzzerRelease	ブザー解除	Buzzer release
ID_STRING_ErrorManagementScreen	エラー管理画面	Error Management screen
ID_STRING_HistoryClear	履歴クリア	History clear
ID_STRING_OrderOldToNew	OLD→NEW	OLD→NEW
ID_STRING_OrderNewToOld	NEW→OLD	NEW→OLD
ID_STRING_OrderHighToLow	HIGH→LOW	HIGH→LOW
ID_STRING_OrderLowToHigh	LOW→HIGH	LOW→HIGH
ID_STRING_DoorClose	扉閉	Door Close
ID_STRING_OperDataManagementScreen	運転データ 管理画面	OperDataManagementScreen
ID_STRING_NowError	現在発生中エラー	Currently occurring error
ID_STRING_ParameterSettings	パラメータ設定	ParameterSettings
ID_STRING_PasswordError	パスワードが違います。	The password is incorrect.
ID_STRING_PasswordErrorThreeTime	パスワード入力を 3 回間違えました。	I was wrong password three times input.
ID_STRING_MaintenanceScreen	保守管理画面	MaintenanceScreen
ID_STRING_ErrorManagementListTitle	エラーレベル 発生時間 エラーコード 異常内 容 復旧	ErrorLevel GenerationTime ErrorCode AbnormalContent Restoration
ID_STRING_MaintenanceNum0	0	0



付録 1 プロパティ設定一覧

付 1-1 リソース設定

GENWARE3 アプリケーション設計ガイド

ID_STRING_MaintenanceNum1	1	1
ID_STRING_MaintenanceNum2	2	2
ID_STRING_MaintenanceNum3	3	3
ID_STRING_MaintenanceNum4	4	4
ID_STRING_MaintenanceNum5	5	5
ID_STRING_MaintenanceNum6	6	6
ID_STRING_MaintenanceNum7	7	7
ID_STRING_MaintenanceNum8	8	8
ID_STRING_MaintenanceNum9	9	9
ID_STRING_MaintenanceNumA	A	A
ID_STRING_MaintenanceNumB	B	B
ID_STRING_MaintenanceNumC	C	C
ID_STRING_MaintenanceNumD	D	D
ID_STRING_MaintenanceNumE	E	E
ID_STRING_MaintenanceNumF	F	F
ID_STRING_MaintenanceCorrection	訂正	Correction
ID_STRING_MaintenanceConfig	確定	Settlement
ID_STRING_MaintenanceInfo	8桁の暗証番号を入力して下さい	Please enter the personal identification number of 8-digit
ID_STRING_MaintenancePermission	許可	Permission
ID_STRING_MaintenanceNotpermitted	不許可	Notpermitted
ID_STRING_MaintenanceLevelMessage1	レベル 1:通常操作許可	Level 1:Normal operation permission
ID_STRING_MaintenanceLevelMessage2	レベル 2:手動操作許可	Level 2:Manual operation permission
ID_STRING_MaintenanceLevelMessage3	レベル 3:全体操作許可	Level 3:The entire operation permission
ID_STRING_ManualToMenu	手動操作画面からメニュー画面に切り替える	to the menu screen from manual operation screen
ID_STRING_MonitorToMenu	モニター画面からメニュー画面に切り替える	to the menu screen of the monitor screen
ID_STRING_ErrorToMenu	エラー管理画面からメニュー画面に切り替える	to the menu screen from the error management screen
ID_STRING_OperDataToMenu	運転データ管理画面からメニュー画面に切り替える	to the menu screen from the operating data management screen
ID_STRING_ManualMngToMenu	マニュアル管理画面からメニュー画面に切り替える	to the menu screen from the manual management screen

## GENWARE3 アプリケーション設計ガイド

ID_STRING_MaintenanceToMenu	保守管理画面からメニュー画面に切り替える	to the menu screen from the maintenance screen
ID_STRING_OperHistoryToMenu	操作履歴画面からメニュー画面に切り替える	to the menu screen from the operation history screen
ID_STRING_ParamSetToMenu	パラメータ設定画面からメニュー画面に切り替える	to the menu screen from the parameter setting screen
ID_STRING_MenuToManual	メニュー画面から手動操作画面に切り替える	I switched to manual operation screen from the menu screen
ID_STRING_MenuToMonitor	メニュー画面からモニター画面に切り替える	to the monitor screen from the menu screen
ID_STRING_MenuToError	メニュー画面からエラー管理画面に切り替える	to error management screen from the menu screen
ID_STRING_MenuToOperData	メニュー画面から運転データ管理画面に切り替える	to run data management screen from the menu screen
ID_STRING_MenuToManualMng	メニュー画面からマニュアル管理画面に切り替える	to manual control screen from the menu screen
ID_STRING_MenuToMaintenance	メニュー画面から保守管理画面に切り替える	to the maintenance screen from the menu screen
ID_STRING_MenuToOperHistory	メニュー画面から操作履歴画面に切り替える	to the operation history screen from the menu screen
ID_STRING_MenuToParamSet	メニュー画面からパラメータ設定画面に切り替える	to the parameter setting screen from the menu screen
ID_STRING_ParameterSetting	パラメータ設定画面	Parameter setting screen
ID_STRING_Parameters1	パラメータ1	Parameter 1
ID_STRING_Parameters2	パラメータ2	Parameter 2
ID_STRING_Parameters3	パラメータ3	Parameter 3
ID_STRING_Parameters4	パラメータ4	Parameter 4
ID_STRING_Parameters5	パラメータ5	Parameter 5
ID_STRING_Parameters6	パラメータ6	Parameter 6
ID_STRING_Parameters7	パラメータ7	Parameter 7
ID_STRING_Parameters8	パラメータ8	Parameter 8
ID_STRING_Parameters9	パラメータ9	Parameter 9
ID_STRING_Parameters10	パラメータ10	Parameter 10
ID_STRING_DataSet	データセット	Data set
ID_STRING_DataRead	データ読出	Clear
ID_STRING_FileSave	ファイル保存	File storage
ID_STRING_FileLoad	ファイル読出し	File read
ID_STRING_ManualManagementScreen	マニュアル管理画面	Manual management screen

## GENWARE3 アプリケーション設計ガイド

ID_STRING_Contents	目次	contents
ID_STRING_OperationHistoryTitle	操作履歴画面	Operation history screen
ID_STRING_PositiveOrNegative	+/-	+/-
ID_STRING_OperationHistroyListTitle	日付      時間      操作内容	Day pay      Time Operating content
ID_STRING00219	検査刻印ライン	Inspection stamping line
ID_STRING00220	刻印数	Engraved number
ID_STRING00221	前日	The day before
ID_STRING00222	当日	The day
ID_STRING00223	先月累計	Last month total
ID_STRING00224	当月累計	Month total
ID_STRING00225	目標	Goal
ID_STRING00226	実績	Performance
ID_STRING00227	差	Difference
ID_STRING00228	加工時間	Machining time
ID_STRING00229	停止時間	Stop time
ID_STRING00230	稼働率	Capacity utilization
ID_STRING00231	コンペアモーターA 負荷率	Compare motor load factor A
ID_STRING00232	コンペアモーターB 負荷率	Compare motor load factor B
ID_STRING00233	オイルヒーター出力	Oil heater output
ID_STRING00234	電流	Current
ID_STRING00235	電圧	Voltage
ID_STRING00236	力率	Power factor
ID_STRING00237	周波数	Frequency
ID_STRING00238	検査・取付・刻印ライン	Inspection, installation or engraved line
ID_STRING00239	使用電力量	Electric power consumption
ID_STRING00240	払い出しライン	Dispensing line
ID_STRING00242	個	Individual
ID_STRING00243	分	Minute
ID_STRING00247	%d	%d
ID_STRING00248	%.1f	%.1f
ID_STRING00250	コンペア	Conveyor

## フォント

ID\_FONT00000

項 目	設 定 内 容
リソースデータ名	ID_FONT00000
フォント名	STD_JA_12
フォントサイズ	12
倍率横	1 倍
倍率縦	1 倍
太さ	THIN
イタリック	なし
色縁取り	なし
文字の読み方向	左から右
拡張スタイル	なし
文字間隔	0

ID\_FONT00101

項 目	設 定 内 容
リソースデータ名	ID_FONT00101
フォント名	STD_JA_12
フォントサイズ	10
倍率横	1 倍
倍率縦	1 倍
太さ	THIN
イタリック	なし
色縁取り	なし
文字の読み方向	左から右
拡張スタイル	なし
文字間隔	0

## ID\_FONT00241

項 目	設 定 内 容
リソースデータ名	ID_FONT00241
フォント名	STD_JA_12
フォントサイズ	12
倍率横	1 倍
倍率縦	1 倍
太さ	NORMAL
イタリック	なし
色縁取り	なし
文字の読み方向	左から右
拡張スタイル	なし
文字間隔	0

## イメージリソース

項 目	フ ァ イ ル 名	
img_ConfigButton_OFF	pass-num-3.bmp	
img_ConfigButton_ON	pass-num-2.bmp	
img_CorrectionButton_OFF	pass-num-5.bmp	
img_CorrectionButton_ON	pass-num-4.bmp	
img_CurrentVoltage	Pnl-2-4_back.bmp	
img_DispensingLineScreen	Pnl-2-5_back.bmp	
img_Figure	Pnl-2-2_back.bmp	
img_InspectionMountingCarvedSeal	Pnl-2-5_back.bmp	
img_InspectionStampingLine	Pnl-2-1_change.bmp	
img_LeftMove	Left-2.bmp	
img_LeftNothing	Left-1.bmp	
img_LoadFactor	Pnl-2-3_back.bmp	
img_MenuButton_OFF	b-1.bmp	
img_MenuButton_ON	b-2.bmp	
img_MoveBodyNG	2panl_Pic_Body_2.bmp	
img_MoveBodyOK	2panl_Pic_Body.bmp	
img_MoveBottom	2panl_Pic.bmp	
img_NumButton0_OFF	pass-num-1-0.bmp	
img_NumButton0_ON	pass-num-0-0.bmp	
img_NumButton_OFF	pass-num-1.bmp	
img_NumButton_ON	pass-num-0.bmp	
img_Onoff001_OFF	ok-2-1.bmp	
img_Onoff001_ON	ok-2-2.bmp	
img_Onoff002_OFF	ok-3-1.bmp	
img_Onoff002_ON	ok-3-2.bmp	
img_Onoff003_OFF	ok-4-1.bmp	
img_Onoff003_ON	ok-4-2.bmp	
img_Onoff004_OFF	ok-1-1.bmp	
img_Onoff004_ON	ok-1-2.bmp	
img_Onoff005_OFF	ok-5-1.bmp	
img_Onoff005_ON	ok-5-2.bmp	
img_Onoff_Round001_OFF	test-4-1.bmp	
img_Onoff_Round001_ON	test-4-2.bmp	
img_Onoff_Round002_OFF	test-3-1.bmp	
img_Onoff_Round002_ON	test-3-2.bmp	

## GENWARE3 アプリケーション設計ガイド

項 目	ファイル名	
img_Onoff_Round003_OFF	test-2-1.bmp	
img_Onoff_Round003_ON	test-2-2.bmp	
img_Onoff_Round004_OFF	test-1-1.bmp	
img_Onoff_Round004_ON	test-1-2.bmp	
img_Onoff_Round005_OFF	test-5-1.bmp	
img_Onoff_Round005_ON	test-5-2.bmp	
img_Onoff_Square_OFF	led-3.bmp	
img_Onoff_Square_ON	led-1.bmp	
img_Page01	test-page-01.bmp	
img_Page02	test-page-02.bmp	
img_Page03	test-page-03.bmp	
img_Page04	test-page-04.bmp	
img_Page05	test-page-05.bmp	
img_Page06	test-page-06.bmp	
img_Page07	test-page-07.bmp	
img_Onoff_Round005_ON	test-5-2.bmp	
img_Page08	test-page-08.bmp	
img_Page09	test-page-09.bmp	
img_Page10	test-page-10.bmp	
img_RightMove	Right-2.bmp	
img_RightNothing	Right-1.bmp	
img_Show_OFF	tt-0.bmp	
img_Show_ON	tt-2.bmp	
img_Test_Title	title-1.bmp	
img_TitleBack	Pnl-title.bmp	
img_errorOrderBotton	error-botton.bmp	
img_Page08	test-page-08.bmp	

## GUIデータベース

## ErrorManagement

項 目	設 定 内 容
オブジェクト名	GDB_ERR_LIST
ID	ID_GDB_ERR_LIST
変数名	gdbErrList
変数型	UCHAR
配列サイズ	50
static 宣言	あり

## ManualOperation

項 目	設 定 内 容
オブジェクト名	GDB_CURVE_DATA
ID	ID_GDB_CURVE_DATA
変数名	gdbCurveData
変数型	INT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_SIGNAL_STS
ID	ID_GDB_SIGNAL_STS
変数名	gdbSignalSts
変数型	UINT
配列サイズ	1
static 宣言	あり

## Monitor

項 目	設 定 内 容
オブジェクト名	GDB_BUZZER_CLEAR
ID	ID_GDB_BUZZER_CLEAR
変数名	gdbBuzzerClear
変数型	UCHAR
配列サイズ	1
static 宣言	あり



## GENWARE3 アプリケーション設計ガイド

項 目	設 定 内 容
オブジェクト名	GDB_CHECK_RESULT
ID	ID_GDB_CHECK_RESULT
変数名	gdbCheckResult
変数型	UCHAR
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_DEVICE_TYPE
ID	ID_GDB_DEVICE_TYPE
変数名	gdbDeviceType
変数型	UCHAR
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_ERR_DEVICE_TYPE
ID	ID_GDB_ERR_DEVICE_TYPE
変数名	gdbErrDeviceType
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_WARNING_CLEAR
ID	ID_GDB_WARNING_CLEAR
変数名	gdbWarningClear
変数型	UCHAR
配列サイズ	1
static 宣言	あり

## GENWARE3 アプリケーション設計ガイド

## OperDataManagement

項 目	設 定 内 容
オブジェクト名	GDB_CONVEYOR_A
ID	ID_GDB_CONVEYOR_A
変数名	gdbConveyor1
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_CONVEYOR_B
ID	ID_GDB_CONVEYOR_B
変数名	gdbConveyor2
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_CYCLE
ID	ID_GDB_CYCLE
変数名	gdbCycle
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_CYCLE_DELIVERY
ID	ID_GDB_CYCLE_DELIVERY
変数名	gdbCycle_Delivery
変数型	USHORT
配列サイズ	1
static 宣言	あり

## GENWARE3 アプリケーション設計ガイド

項 目	設 定 内 容
オブジェクト名	GDB_CYCLE_IMPRINT
ID	ID_GDB_CYCLE_IMPRINT
変数名	gdbCycleImprint
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_DIFF_DOWN
ID	ID_GDB_DIFF_DOWN
変数名	gdbDiffDown
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_DIFF_UP
ID	ID_GDB_DIFF_UP
変数名	gdbDiffUp
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_ELEC
ID	ID_GDB_ELEC
変数名	gdbElec
変数型	USHORT
配列サイズ	1
static 宣言	あり

## GENWARE3 アプリケーション設計ガイド

項 目	設 定 内 容
オブジェクト名	GDB_ELEC_DELIVERY
ID	ID_GDB_ELEC_DELIVERY
変数名	GDBElecDelivery
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_ELEC_IMPRINT
ID	ID_GDB_ELEC_IMPRINT
変数名	gdbElecImprint
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_OIL_HEATER
ID	ID_GDB_OIL_HEATER
変数名	gdbOilHeater
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_POWER
ID	ID_GDB_POWER
変数名	gdbPower
変数型	USHORT
配列サイズ	1
static 宣言	あり

## GENWARE3 アプリケーション設計ガイド

項 目	設 定 内 容
オブジェクト名	GDB_POWER_DELIVERY
ID	ID_GDB_POWER_DELIVERY
変数名	gdbPowerDelivery
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_POWER_IMPRINT
ID	ID_GDB_POWER_IMPRINT
変数名	gdbPowerImprint
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_PRACTICAL_UP
ID	ID_GDB_PRACTICAL_UP
変数名	gdbPracticalUP
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_PRACTICAL_DOWN
ID	ID_GDB_PRACTICAL_DOWN
変数名	gdbPracticalDown
変数型	USHORT
配列サイズ	1
static 宣言	あり

## GENWARE3 アプリケーション設計ガイド

項 目	設 定 内 容
オブジェクト名	GDB_RATIO_DOWN
ID	ID_GDB_RATIO_DOWN
変数名	gdbRatioDown
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_RATIO_UP
ID	ID_GDB_RATIO_UP
変数名	gdbRatioUp
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_STOP_TIME_UP
ID	ID_GDB_STOP_TIME_UP
変数名	gdbStopTimeUP
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_STOP_TIME_DOWN
ID	ID_GDB_STOP_TIME_DOWN
変数名	gdbStopTimeDOWN
変数型	USHORT
配列サイズ	1
static 宣言	あり

## GENWARE3 アプリケーション設計ガイド

項 目	設 定 内 容
オブジェクト名	GDB_TARGET_DOWN
ID	ID_GDB_TARGET_DOWN
変数名	gdbTargetDOWN
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_TARGET_UP
ID	ID_GDB_TARGET_UP
変数名	gdbTargetUP
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_TODAY_DELIVERY
ID	ID_GDB_TODAY_DELIVERY
変数名	gdbToday_Delivery
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_TODAY_IMPRINT
ID	ID_GDB_TODAY_IMPRINT
変数名	gdbTodayImprint
変数型	USHORT
配列サイズ	1
static 宣言	あり

## GENWARE3 アプリケーション設計ガイド

項 目	設 定 内 容
オブジェクト名	GDB_TODAY_IMPRINT_DELIVERY
ID	ID_GDB_TODAY_IMPRINT_DELIVERY
変数名	gdbTodayImprint
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_TOTAL_CURMONTH
ID	ID_GDB_TOTAL_CURMONTH
変数名	gdbTotalCurMonth
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_TOTAL_PREMONTH
ID	ID_GDB_TOTAL_PREMONTH
変数名	gdbTotalPreMonth
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_VOL
ID	ID_GDB_VOL
変数名	gdbVol
変数型	USHORT
配列サイズ	1
static 宣言	あり



## GENWARE3 アプリケーション設計ガイド

項 目	設 定 内 容
オブジェクト名	GDB_VOL_IMPRINT
ID	ID_GDB_VOL_IMPRINT
変数名	gdbVollkprint
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_VOL_DELIVERY
ID	ID_GDB_VOL_DELIVERY
変数名	gdbVolDelivery
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_WORK_TIME_UP
ID	ID_GDB_WORK_TIME_UP
変数名	gdbWorkTimeUP
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_WORK_TIME_DOWN
ID	ID_GDB_WORK_TIME_DOWN
変数名	gdbWorkTimeDOWN
変数型	USHORT
配列サイズ	1
static 宣言	あり

## GENWARE3 アプリケーション設計ガイド

項 目	設 定 内 容
オブジェクト名	GDB_YESTER_DELIVERY
ID	ID_GDB_YESTER_DELIVERY
変数名	gdbYester_Delivery
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_YESTER_IMPRINT
ID	ID_GDB_YESTER_IMPRINT
変数名	gdbYesterImprint
変数型	USHORT
配列サイズ	1
static 宣言	あり

項 目	設 定 内 容
オブジェクト名	GDB_YESTER_IMPRINT_DELIVERY
ID	ID_GDB_YESTER_IMPRINT_DELIVERY
変数名	gdbYesterDelivery
変数型	USHORT
配列サイズ	1
static 宣言	あり

## ParameterSetting

項 目	設 定 内 容
オブジェクト名	GDB_DECIMAL
ID	ID_GDB_DECIMAL
変数名	gdbDecimal
変数型	INT
配列サイズ	10
static 宣言	あり

**GENWARE3 アプリケーション設計ガイド**

項 目	設 定 内 容
オブジェクト名	GDB_HEXADECIMAL
ID	ID_GDB_HEXADECIMAL
変数名	gdbHexadecimal
変数型	INT
配列サイズ	10
static 宣言	あり

追加する型定義([ツール]-[GUI データベース]-[型定義]メニュー)

型名	基になるデータ型	設 定 内 容
UINT	unsigned int	整数
USHORT	unsigned short	整数
UCHAR	unsigned char	8ビット無符号整数

## 付 1-2 プロジェクトプロパティ設定

項 目	設 定 内 容
プロジェクトコメント	—
システムフォント	ID_FONT00000
表示倍率	100
フレーム更新間隔(fps)	10

## 付 1-3 スクリーンプロパティ設定

「コールバック関数」欄には、設定を「あり」とする関数名を記載します。

項 目	設 定 内 容
スクリーン WIDTH	1024
スクリーン HEIGHT	768
実行開始ページ No.	1
パネルのデフォルト背景色	RGB(192,192,192)
ベースタイム更新間隔	80
カーソル方向	縦 2dot
コールバック関数	OnTimer OnUpdateDB

## 付 1-4 パネル/ウィンドウプロパティ設定

手動操作画面 (パネル名 : Panel\_ManualOperation)

項 目	設 定 内 容
パネル名	Panel_ManualOperation
WIDTH	1024
HEIGHT	768
背景色	RGB(255,255,255)
コールバック関数	OnCreate OnUpdateDB

モニター画面 (パネル名 : Panel\_Monitor)

項 目	設 定 内 容
パネル名	Panel_Monitor
WIDTH	1024
HEIGHT	768
背景色	RGB(255,255,255)
コールバック関数	OnCreate OnUpdateDB

エラー管理画面 (パネル名 : Panel\_ErrorManagement)

項 目	設 定 内 容
パネル名	Panel_ErrorManagement
WIDTH	1024
HEIGHT	768
背景色	RGB(255,255,255)
コールバック関数	OnCreate OnUpdateDB

**GENWARE3 アプリケーション設計ガイド****運転データ管理画面 (パネル名 : Panel\_OperDataManagement)**

項 目	設 定 内 容
パネル名	Panel_OperDataManagement
WIDTH	1024
HEIGHT	768
背景色	RGB(255,255,255)
コールバック関数	OnCreate OnUpdateDB

**マニュアル管理画面 (パネル名 : Panel\_ManualManagement)**

項 目	設 定 内 容
パネル名	Panel_ManualManagement
WIDTH	1024
HEIGHT	768
背景色	RGB(255,255,255)
コールバック関数	OnCreate

**保守管理画面 (パネル名 : Panel\_Maintenance)**

項 目	設 定 内 容
パネル名	Panel_Maintenance
WIDTH	1024
HEIGHT	768
背景色	RGB(255,255,255)
コールバック関数	OnCreate

**操作履歴画面 (パネル名 : Panel\_OperationHistory)**

項 目	設 定 内 容
パネル名	Panel_OperationHistory
WIDTH	1024
HEIGHT	768
背景色	RGB(255,255,255)
コールバック関数	OnCreate

**GENWARE3 アプリケーション設計ガイド****パラメータ設定画面(パネル名:Panel\_ParameterSetting)**

項 目	設 定 内 容
パネル名	Panel_ParameterSetting
WIDTH	1024
HEIGHT	768
背景色	RGB(255,255,255)
コールバック関数	OnCreate

**メニュー(ウィンドウ名:Window\_Menu)**

項 目	設 定 内 容
ウィンドウ名	Window_Menu
X	300
Y	90
WIDTH	650
HEIGHT	240
背景色	RGB(64,128,128)
タイトルバーの有無	なし
クローズボタンの有無	なし
ウィンドウ枠の有無	なし
コールバック関数	OnCreate

**メッセージ画面(ウィンドウ名:Window\_Message)**

項 目	設 定 内 容
ウィンドウ名	Window_Message
X	500
Y	400
WIDTH	480
HEIGHT	180
背景色	RGB(255,128,128)
タイトルバーの有無	なし
クローズボタンの有無	なし
ウィンドウ枠の有無	なし
コールバック関数	なし

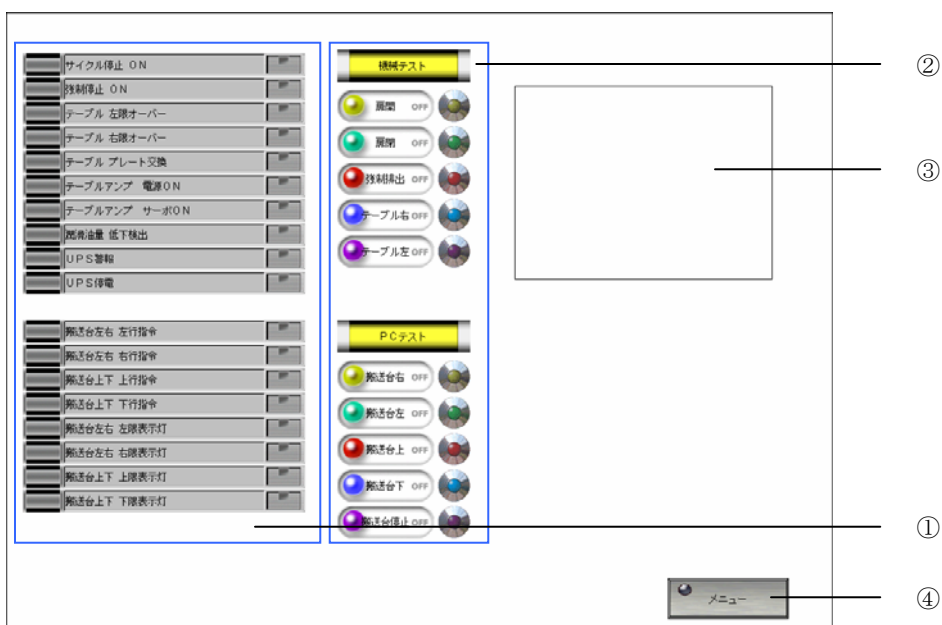
## テンキー画面(ウインドウ名:Window\_NumericKeypad)

項 目	設 定 内 容
ウインドウ名	Window_NumericKeypad
X	550
Y	450
WIDTH	255
HEIGHT	213
背景色	RGB(0,0,0)
タイトルバーの有無	なし
クローズボタンの有無	なし
ウインドウ枠の有無	なし
コールバック関数	OnCreate



## 付 1-5 コントロールプロパティ設定

手動操作画面(パネル名: Panel\_ManualOperation)



番号	項目	使用コントロール
①	ランプ	ピクチャ×36 スタティックテキスト×18
②	スイッチ、ランプ	ボタン×10 ピクチャ×10 (ランプ部分) ピクチャ×2 (タイトル部分)
③	グラフ表示	基本コントロール
④	メニューボタン	ボタン

## GENWARE3 アプリケーション設計ガイド

## ①ランプ 1(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCycleStopON
X	20
Y	50
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## ランプ 2(ピクチャ)

項 目	設 定 内 容
コントロール名	pctForcedStopON
X	20
Y	80
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## ランプ 3(ピクチャ)

項 目	設 定 内 容
コントロール名	pctTableLeftOverLimit
X	20
Y	110
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## GENWARE3 アプリケーション設計ガイド

## ランプ 4(ピクチャ)

項 目	設 定 内 容
コントロール名	pctTableRightOverLimit
X	20
Y	140
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## ランプ 5(ピクチャ)

項 目	設 定 内 容
コントロール名	pctTablePlateExchange
X	20
Y	170
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## ランプ 6(ピクチャ)

項 目	設 定 内 容
コントロール名	pctTableAmpPowerON
X	20
Y	200
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## GENWARE3 アプリケーション設計ガイド

## ランプ 7(ピクチャ)

項 目	設 定 内 容
コントロール名	pctTableAmpServoON
X	20
Y	230
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## ランプ 8(ピクチャ)

項 目	設 定 内 容
コントロール名	pctOilDropDetection
X	20
Y	260
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## ランプ 9(ピクチャ)

項 目	設 定 内 容
コントロール名	pctUPSAlarm
X	20
Y	290
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## GENWARE3 アプリケーション設計ガイド

## ランプ 10(ピクチャ)

項 目	設 定 内 容
コントロール名	pctUPSBlackout
X	20
Y	320
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## ランプ 11(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageLDirective
X	20
Y	380
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## ランプ 12(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageRDirective
X	20
Y	410
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## GENWARE3 アプリケーション設計ガイド

## ランプ 13(ピクチャ)

項 目	設 定 内 容
コントロール名	PctCarriageUDirective
X	20
Y	440
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## ランプ 14(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageDDirective
X	20
Y	470
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## ランプ 15(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageLLight
X	20
Y	500
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## GENWARE3 アプリケーション設計ガイド

## ランプ 16(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageRLight
X	20
Y	530
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## ランプ 17(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageULight
X	20
Y	560
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## ランプ 18(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageDLight
X	20
Y	590
WIDTH	50
HEIGHT	28
状態の数	2
状態 0 意匠	img_Show_OFF
状態 1 意匠	img_Show_ON

## GENWARE3 アプリケーション設計ガイド

## ランプ 19(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCycleStopONLight
X	319
Y	51
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## ランプ 20(ピクチャ)

項 目	設 定 内 容
コントロール名	pctForcedStopONLight
X	319
Y	81
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## ランプ 21(ピクチャ)

項 目	設 定 内 容
コントロール名	pctTableLeftOverLimitLight
X	319
Y	110
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON



## GENWARE3 アプリケーション設計ガイド

## ランプ 22(ピクチャ)

項 目	設 定 内 容
コントロール名	pctTableRightOverLimitLight
X	319
Y	141
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## ランプ 23(ピクチャ)

項 目	設 定 内 容
コントロール名	pctTablePlateExchangeLight
X	319
Y	171
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## ランプ 24(ピクチャ)

項 目	設 定 内 容
コントロール名	pctTableAmpPowerONLight
X	319
Y	201
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## GENWARE3 アプリケーション設計ガイド

## ランプ 25(ピクチャ)

項 目	設 定 内 容
コントロール名	pctTableAmpServoONLight
X	319
Y	231
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## ランプ 26(ピクチャ)

項 目	設 定 内 容
コントロール名	pctOilDropDetectionLight
X	319
Y	261
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## ランプ 27(ピクチャ)

項 目	設 定 内 容
コントロール名	pctUPSAlarmLight
X	319
Y	291
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## GENWARE3 アプリケーション設計ガイド

## ランプ 28(ピクチャ)

項 目	設 定 内 容
コントロール名	pctUPSBlackoutLight
X	319
Y	321
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## ランプ 29(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageLDirectiveLight
X	319
Y	381
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## ランプ 30(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageRDirectiveLight
X	319
Y	411
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## GENWARE3 アプリケーション設計ガイド

## ランプ 31(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageUDirectiveLight
X	319
Y	441
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## ランプ 32(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageDDirectiveLight
X	319
Y	471
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## ランプ 33(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageLLightLight
X	319
Y	501
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## GENWARE3 アプリケーション設計ガイド

## ランプ 34(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageRLightLight
X	319
Y	531
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## ランプ 35(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageULightLight
X	319
Y	561
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## ランプ 36(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarriageDLightLight
X	319
Y	591
WIDTH	49
HEIGHT	25
状態の数	2
状態 0 意匠	img_Onoff_Square_OFF
状態 1 意匠	img_Onoff_Square_ON

## GENWARE3 アプリケーション設計ガイド

## 項目名 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtCycleStopON
X	70
Y	50
WIDTH	300
HEIGHT	28
文字列	ID_STRING_CycleStopON サイクル停止 ON
横位置	左寄せ
縦位置	中央

## 項目名 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtForcedStopON
X	70
Y	80
WIDTH	300
HEIGHT	28
文字列	ID_STRING_ForcedStopON 強制停止 ON
横位置	左寄せ
縦位置	中央

## 項目名 3(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtTableLeftOverLimit
X	70
Y	110
WIDTH	300
HEIGHT	28
文字列	ID_STRING_TableLeftOverLimit テーブル 左限オーバー
横位置	左寄せ
縦位置	中央

## GENWARE3 アプリケーション設計ガイド

## 項目名 4(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtTableRightOverLimit
X	70
Y	140
WIDTH	300
HEIGHT	28
文字列	ID_STRING_TableRightOverLimit テーブル 右限オーバー
横位置	左寄せ
縦位置	中央

## 項目名 5(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtTablePlateExchange
X	70
Y	170
WIDTH	300
HEIGHT	28
文字列	ID_STRING_TablePlateExchange テーブル プレート交換
横位置	左寄せ
縦位置	中央

## 項目名 6(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtTableAmpPowerON
X	70
Y	200
WIDTH	300
HEIGHT	28
文字列	ID_STRING_TableAmpPowerON テーブルアンブ 電源ON
横位置	左寄せ
縦位置	中央

## GENWARE3 アプリケーション設計ガイド

## 項目名 7(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtTableAmpServoON
X	70
Y	230
WIDTH	300
HEIGHT	28
文字列	ID_STRING_TableAmpServoON テーブルアンプ サーボON
横位置	左寄せ
縦位置	中央

## 項目名 8(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtOilDropDetection
X	70
Y	260
WIDTH	300
HEIGHT	28
文字列	ID_STRING_OilDropDetection 潤滑油量 低下検出
横位置	左寄せ
縦位置	中央

## 項目名 9(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtUPSAlarm
X	70
Y	290
WIDTH	300
HEIGHT	28
文字列	ID_STRING_UPSAlarm UPS警報
横位置	左寄せ
縦位置	中央



## GENWARE3 アプリケーション設計ガイド

## 項目名 10(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtUPSBlackout
X	70
Y	320
WIDTH	300
HEIGHT	28
文字列	ID_STRING_UPSBlackout UPS停電
横位置	左寄せ
縦位置	中央

## 項目名 11(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtCarriageLDirective
X	70
Y	380
WIDTH	300
HEIGHT	28
文字列	ID_STRING_CarriageLDirective 搬送台左右 左行指令
横位置	左寄せ
縦位置	中央

## 項目名 12(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtCarriageRDirective
X	70
Y	410
WIDTH	300
HEIGHT	28
文字列	ID_STRING_CarriageRDirective 搬送台左右 右行指令
横位置	左寄せ
縦位置	中央

## GENWARE3 アプリケーション設計ガイド

## 項目名 13(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtCarriageUDirective
X	70
Y	440
WIDTH	300
HEIGHT	28
文字列	ID_STRING_CarriageUDirective 搬送台上下 上行指令
横位置	左寄せ
縦位置	中央

## 項目名 14(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtCarriageDDirective
X	70
Y	470
WIDTH	300
HEIGHT	28
文字列	ID_STRING_CarriageDDirective 搬送台上下 下行指令
横位置	左寄せ
縦位置	中央

## 項目名 15(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtCarriageLLight
X	70
Y	500
WIDTH	300
HEIGHT	28
文字列	ID_STRING_CarriageLLight 搬送台左右 左限表示灯
横位置	左寄せ
縦位置	中央

## GENWARE3 アプリケーション設計ガイド

## 項目名 16(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtCarriageRLight
X	70
Y	530
WIDTH	300
HEIGHT	28
文字列	ID_STRING_CarriageRLight 搬送台左右 右限表示灯
横位置	左寄せ
縦位置	中央

## 項目名 17(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtCarriageULight
X	70
Y	560
WIDTH	300
HEIGHT	28
文字列	ID_STRING_CarriageULight 搬送台上下 上限表示灯
横位置	左寄せ
縦位置	中央

## 項目名 18(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcTxtCarriageDLight
X	70
Y	590
WIDTH	300
HEIGHT	28
文字列	ID_STRING_CarriageDLight 搬送台上下 下限表示灯
横位置	左寄せ
縦位置	中央

## GENWARE3 アプリケーション設計ガイド

## ②スイッチ 1(ボタン)

項 目	設 定 内 容
コントロール名	btnTableLeft
X	410
Y	275
WIDTH	120
HEIGHT	40
ボタンタイプ	オルタネート
表示タイプ	イメージ
ON 時意匠	img_Onoff005_ON
OFF 時意匠	img_Onoff005_OFF
文字列	ID_STRING_TableLeft テーブル左
フォント	ID_FONT00101
立体枠の有無	なし
コールバック関数	OnClick

## スイッチ 2(ボタン)

項 目	設 定 内 容
コントロール名	btnTableRight
X	410
Y	230
WIDTH	120
HEIGHT	40
ボタンタイプ	オルタネート
表示タイプ	イメージ
ON 時意匠	img_Onoff04_ON
OFF 時意匠	img_Onoff004_OFF
文字列	ID_STRING_TableRight テーブル右
フォント	ID_FONT00101
立体枠の有無	なし
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## スイッチ 3(ボタン)

項 目	設 定 内 容
コントロール名	btnForcedOut
X	410
Y	185
WIDTH	120
HEIGHT	40
ボタンタイプ	オルタネート
表示タイプ	イメージ
ON 時意匠	img_Onoff03_ON
OFF 時意匠	img_Onoff003_OFF
文字列	ID_STRING_ForcedOut 強制排出
フォント	ID_FONT00101
立体枠の有無	なし
コールバック関数	OnClick

## スイッチ 4(ボタン)

項 目	設 定 内 容
コントロール名	btnHirakiSekiTwo
X	410
Y	140
WIDTH	120
HEIGHT	40
ボタンタイプ	オルタネート
表示タイプ	イメージ
ON 時意匠	img_Onoff02_ON
OFF 時意匠	img_Onoff002_OFF
文字列	ID_STRING_DoorClose 扉閉
フォント	ID_FONT00101
立体枠の有無	なし
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## スイッチ 5(ボタン)

項 目	設 定 内 容
コントロール名	btnHirakiSekiOne
X	410
Y	95
WIDTH	120
HEIGHT	40
ボタンタイプ	オルタネート
表示タイプ	イメージ
ON 時意匠	img_Onoff01_ON
OFF 時意匠	img_Onoff001_OFF
文字列	ID_STRING_DoorOpen 扉開
フォント	ID_FONT00101
立体枠の有無	なし
コールバック関数	OnClick

## スイッチ 6(ボタン)

項 目	設 定 内 容
コントロール名	btnCarriageStop
X	410
Y	610
WIDTH	120
HEIGHT	40
ボタンタイプ	オルタネート
表示タイプ	イメージ
ON 時意匠	img_Onoff005_ON
OFF 時意匠	img_Onoff005_OFF
文字列	ID_STRING_CarriageStop 搬送台停止
フォント	ID_FONT00101
立体枠の有無	なし
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## スイッチ 7(ボタン)

項 目	設 定 内 容
コントロール名	btnCarriageDown
X	410
Y	565
WIDTH	120
HEIGHT	40
ボタンタイプ	オルタネート
表示タイプ	イメージ
ON 時意匠	img_Onoff04_ON
OFF 時意匠	img_Onoff004_OFF
文字列	ID_STRING_CarriageDown 搬送台下
フォント	ID_FONT00101
立体枠の有無	なし
コールバック関数	OnClick

## スイッチ 8(ボタン)

項 目	設 定 内 容
コントロール名	btnCarriageUp
X	410
Y	520
WIDTH	120
HEIGHT	40
ボタンタイプ	オルタネート
表示タイプ	イメージ
ON 時意匠	img_Onoff03_ON
OFF 時意匠	img_Onoff003_OFF
文字列	ID_STRING_CarriageUp 搬送台上
フォント	ID_FONT00101
立体枠の有無	なし
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## スイッチ 9(ボタン)

項 目	設 定 内 容
コントロール名	btnCarriageLeft
X	410
Y	475
WIDTH	120
HEIGHT	40
ボタンタイプ	オルタネート
表示タイプ	イメージ
ON 時意匠	img_Onoff02_ON
OFF 時意匠	img_Onoff02_OFF
文字列	ID_STRING_CarriageLeft 搬送台左
フォント	ID_FONT00101
立体枠の有無	なし
コールバック関数	OnClick

## スイッチ 10(ボタン)

項 目	設 定 内 容
コントロール名	btnCarriageRight
X	410
Y	430
WIDTH	120
HEIGHT	40
ボタンタイプ	オルタネート
表示タイプ	イメージ
ON 時意匠	img_Onoff01_ON
OFF 時意匠	img_Onoff01_OFF
文字列	ID_STRING_CarriageRight 搬送台右
フォント	ID_FONT00101
立体枠の有無	なし
コールバック関数	OnClick



## GENWARE3 アプリケーション設計ガイド

## ③グラフ(基本コントロール)

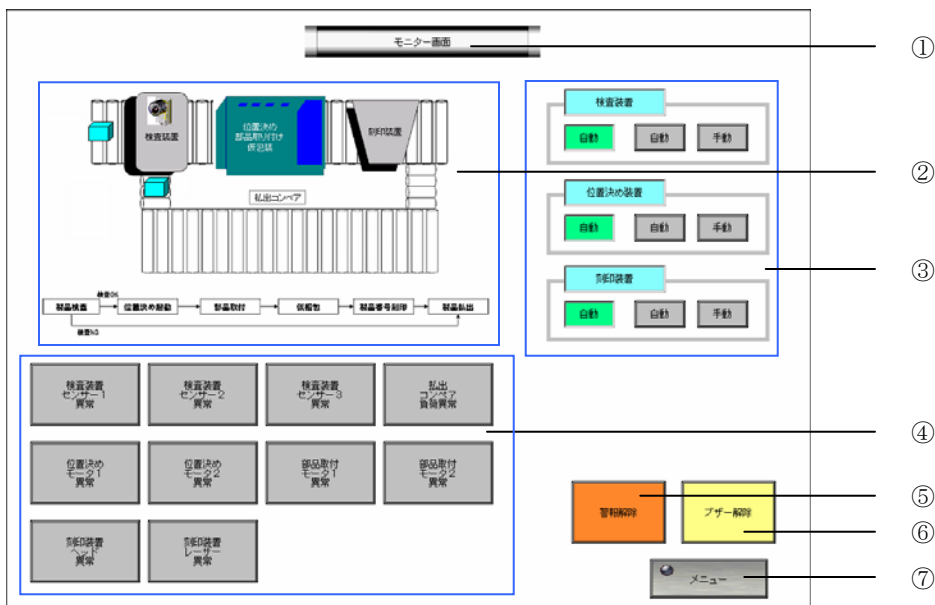
項 目	設 定 内 容
コントロール名	bacCtlSinLine
X	630
Y	90
WIDTH	320
HEIGHT	240
コールバック関数	OnTimer
	OnCreate

## ④メニューボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnManualOperationMenu
X	820
Y	700
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Menu
	メニュー
フォント	ID_FONT00101
立体枠の有無	なし
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## モニター画面(パネル名:Panel\_Monitor)



番号	項目	使用コントロール
①	タイトル	ピクチャ
②	コンベア	ピクチャ×3
③	「自動」/「手動」	ボタン×6 ピクチャ×6 矩形×3
④	異常ランプ	ボタン×10
⑤	警報解除	ボタン
⑥	ブザー解除	ボタン
⑦	メニューボタン	ボタン

## ①タイトル(ピクチャ)

項 目	設 定 内 容
コントロール名	pctMonitorTitle
X	380
Y	20
WIDTH	300
HEIGHT	40
状態の数	1
状態 0 意匠	img_TitleBack
文字列	ID_STRING_MonitorScreenTitle モニター画面
フォント	ID_FONT00000

## ②コンベア背景(ピクチャ)

項 目	設 定 内 容
コントロール名	pctMoveBottom
X	30
Y	100
WIDTH	600
HEIGHT	320
状態の数	1
状態 0 意匠	img_MoveBottom

## ワーク 1(ピクチャ)

項 目	設 定 内 容
コントロール名	pctMoveBody
X	104
Y	142
WIDTH	32
HEIGHT	25
状態の数	2
状態 0 意匠	img_MoveBodyOK
状態 1 意匠	img_MoveBodyNG

## ワーク 2(ピクチャ)

項 目	設 定 内 容
コントロール名	pctMoveBody2
X	175
Y	213
WIDTH	32
HEIGHT	25
状態の数	2
状態 0 意匠	img_MoveBodyOK
状態 1 意匠	img_MoveBodyNG

## ③タイトル 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextInspectionTitle
X	710
Y	100
WIDTH	130
HEIGHT	35
文字色	RGB(128,255,255)
文字列	ID_STRING_MonitorInspectionEquipment 検査装置
横位置	中央
縦位置	中央

## 枠線 1(矩形)

項 目	設 定 内 容
コントロール名	rectInspectionFrame
X	690
Y	120
WIDTH	280
HEIGHT	75
外周線の色	RGB(192,192,192)
外周線の幅	5

## GENWARE3 アプリケーション設計ガイド

## 現在値 1 (スタティックテキスト)

項 目	設 定 内 容
コントロール名	sextInspectionShowChange
X	710
Y	145
WIDTH	65
HEIGHT	35
文字色	RGB(0,255,128)
文字列	ID_STRING_MonitorAuto
	自動
横位置	中央
縦位置	中央

## 自動ボタン 1 (ボタン)

項 目	設 定 内 容
コントロール名	btnInspectionAuto
X	800
Y	145
WIDTH	65
HEIGHT	35
ボタンタイプ	モーメンタリ
表示タイプ	矩形
文字列	ID_STRING_MonitorAuto
	自動
フォント	ID_FONT00000
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## 手動ボタン 1(ボタン)

項 目	設 定 内 容
コントロール名	btnInspectionManual
X	880
Y	145
WIDTH	65
HEIGHT	35
ボタンタイプ	モーメンタリ
表示タイプ	矩形
文字列	ID_STRING_MonitorManual 手動
フォント	ID_FONT00000
コールバック関数	OnClick

## タイトル 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextPositioningTitie
X	710
Y	215
WIDTH	130
HEIGHT	35
文字色	RGB(128,255,255)
文字列	ID_STRING_MonitorPositioningDevice 位置決め装置
横位置	中央
縦位置	中央

## 枠線 2(矩形)

項 目	設 定 内 容
コントロール名	rectPositioningFrame
X	690
Y	235
WIDTH	280
HEIGHT	75
外周線の色	RGB(192,192,192)
外周線の幅	5

## GENWARE3 アプリケーション設計ガイド

## 現在値 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	sextInspectionShowChange
X	710
Y	145
WIDTH	65
HEIGHT	35
文字色	RGB(0,255,128)
文字列	ID_STRING_MonitorAuto
	自動
横位置	中央
縦位置	中央

## 自動ボタン 2(ボタン)

項 目	設 定 内 容
コントロール名	btnPositioningAuto
X	800
Y	260
WIDTH	65
HEIGHT	35
ボタンタイプ	モーメンタリ
表示タイプ	矩形
文字列	ID_STRING_MonitorAuto
	自動
フォント	ID_FONT00000
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## 手動ボタン 2(ボタン)

項 目	設 定 内 容
コントロール名	btnPositioningManual
X	880
Y	260
WIDTH	65
HEIGHT	35
ボタンタイプ	モーメンタリ
表示タイプ	矩形
文字列	ID_STRING_MonitorManual
	手動
フォント	ID_FONT00000
コールバック関数	OnClick

## タイトル 3(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextMarkingTitie
X	710
Y	325
WIDTH	130
HEIGHT	35
文字色	RGB(128,255,255)
文字列	ID_STRING_MonitorMarkingApparatus
	刻印装置
横位置	中央
縦位置	中央

## 枠線 3(矩形)

項 目	設 定 内 容
コントロール名	rectMarkingFrame
X	690
Y	345
WIDTH	280
HEIGHT	75
外周線の色	RGB(192,192,192)
外周線の幅	5



## 現在値 3(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextMarkingShowChange
X	710
Y	370
WIDTH	65
HEIGHT	35
文字色	RGB(0,255,128)
文字列	ID_STRING_MonitorAuto
	自動
横位置	中央
縦位置	中央

## 自動ボタン 3(ボタン)

項 目	設 定 内 容
コントロール名	btnMarkingAuto
X	800
Y	370
WIDTH	65
HEIGHT	35
ボタンタイプ	モーメンタリ
表示タイプ	矩形
文字列	ID_STRING_MonitorAuto
	自動
フォント	ID_FONT00000
コールバック関数	OnClick

## 手動ボタン 3(ボタン)

項 目	設 定 内 容
コントロール名	btnMarkingManual
X	880
Y	370
WIDTH	65
HEIGHT	35
ボタンタイプ	モーメンタリ
表示タイプ	矩形
文字列	ID_STRING_MonitorManual 手動
フォント	ID_FONT00000
コールバック関数	OnClick

## ④異常ランプ 1(ボタン)

項 目	設 定 内 容
コントロール名	btnMonitorSensor1Abnormality
X	30
Y	450
WIDTH	140
HEIGHT	80
ボタンタイプ	なし
表示タイプ	矩形
文字列	ID_STRING_MonitorSensor1Abnormality 検査装置 センサー1 異常
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 異常ランプ 2(ボタン)

項 目	設 定 内 容
コントロール名	btnMonitorSensor2Abnormality
X	180
Y	450
WIDTH	140
HEIGHT	80
ボタンタイプ	なし
表示タイプ	矩形
文字列	ID_STRING_MonitorSensor2Abnormality 検査装置 センサー2 異常
フォント	ID_FONT00000

## 異常ランプ 3(ボタン)

項 目	設 定 内 容
コントロール名	btnMonitorSensor3Abnormality
X	330
Y	450
WIDTH	140
HEIGHT	80
ボタンタイプ	なし
表示タイプ	矩形
文字列	ID_STRING_MonitorSensor3Abnormality 検査装置 センサー3 異常
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 異常ランプ 4(ボタン)

項 目	設 定 内 容
コントロール名	btnMonitorAbnormalLoad
X	480
Y	450
WIDTH	140
HEIGHT	80
ボタンタイプ	なし
表示タイプ	矩形
文字列	ID_STRING_MonitorAbnormalLoad 払出 コンベア 負荷異常
フォント	ID_FONT00000

## 異常ランプ 5(ボタン)

項 目	設 定 内 容
コントロール名	btnMotor1Abnormality
X	30
Y	550
WIDTH	140
HEIGHT	80
ボタンタイプ	なし
表示タイプ	矩形
文字列	ID_STRING_MonitorMotor1Abnormality 位置決め モータ1 異常
フォント	ID_FONT00000

## 異常ランプ 6(ボタン)

項 目	設 定 内 容
コントロール名	btnMotor2Abnormality
X	180
Y	550
WIDTH	140
HEIGHT	80
ボタンタイプ	なし
表示タイプ	矩形
文字列	ID_STRING_MonitorMotor2Abnormality 位置決め モータ2 異常
フォント	ID_FONT00000

## 異常ランプ 7(ボタン)

項 目	設 定 内 容
コントロール名	btnMountingMotor1Abnormality
X	330
Y	550
WIDTH	140
HEIGHT	80
ボタンタイプ	なし
表示タイプ	矩形
文字列	ID_STRING_MountingMotor1Abnormality 部品取付 モータ1 異常
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 異常ランプ 8(ボタン)

項 目	設 定 内 容
コントロール名	btnMountingMotor2Abnormality
X	480
Y	550
WIDTH	140
HEIGHT	80
ボタンタイプ	なし
表示タイプ	矩形
文字列	ID_STRING_MountingMotor2Abnormality 部品取付 モータ2 異常
フォント	ID_FONT00000

## 異常ランプ 9(ボタン)

項 目	設 定 内 容
コントロール名	btnHeadAbnormality
X	30
Y	650
WIDTH	140
HEIGHT	80
ボタンタイプ	なし
表示タイプ	矩形
文字列	ID_STRING_MonitorHeadAbnormality 刻印装置 ヘッド 異常
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 異常ランプ 10(ボタン)

項 目	設 定 内 容
コントロール名	btnLaserAbnormality
X	180
Y	650
WIDTH	140
HEIGHT	80
ボタンタイプ	なし
表示タイプ	矩形
文字列	ID_STRING_MonitorLaserAbnormality 刻印装置 レーザー 異常
フォント	ID_FONT00000

## ⑤警報解除(ボタン)

項 目	設 定 内 容
コントロール名	btnAlarmRelease
X	720
Y	600
WIDTH	120
HEIGHT	80
ボタンタイプ	モーメンタリ
表示タイプ	矩形
文字列	ID_STRING_MonitorAlarmRelease 警報解除
フォント	ID_FONT00000
コールバック関数	OnClick

## ⑥ブザー解除(ボタン)

項 目	設 定 内 容
コントロール名	btnBuzzerRelease
X	860
Y	600
WIDTH	120
HEIGHT	80
ボタンタイプ	モーメンタリ
表示タイプ	矩形
文字列	ID_STRING_MonitorBuzzerRelease ブザー解除
フォント	ID_FONT00000
コールバック関数	OnClick

## ⑦メニューボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnMonitorMenu
X	820
Y	700
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_ON
文字列	ID_STRING_MonitorBuzzerRelease ブザー解除
フォント	ID_FONT00000
コールバック関数	OnClick



## エラー管理画面(パネル名:Panel\_ErrorManagement)



番号	項目	使用コントロール
①	タイトル	ピクチャ
②	エラー履歴	スタティックテキスト×16
③	ソート	ボタン×4
④	現在発生中エラー	スタティックテキスト×2
⑤	履歴クリア	ボタン
⑥	メニューボタン	ボタン

## ①タイトル(ピクチャ)

項 目	設 定 内 容
コントロール名	pctErrorManagementTitle
X	380
Y	20
WIDTH	300
HEIGHT	40
状態の数	1
状態 0 意匠	img_TitleBack
文字列	ID_STRING_ErrorManagementScreen エラー管理画面
フォント	ID_FONT00000

## ②タイトル(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorListTitle
X	110
Y	115
WIDTH	770
HEIGHT	25
文字色	RGB(128,255,255)
文字列	ID_STRING_ErrorManagementListTitle エラーレベル 発生時間 エラーコード 異常内容 復旧
横位置	左寄せ
縦位置	中央

## リスト 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList01
X	110
Y	140
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

リスト 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList02
X	110
Y	165
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

リスト 3(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList03
X	110
Y	190
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

リスト 4(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList04
X	110
Y	215
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

リスト 5(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList05
X	110
Y	240
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

リスト 6(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList06
X	110
Y	265
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

リスト 7(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList07
X	110
Y	290
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

リスト 8(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList08
X	110
Y	315
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

リスト 9(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList09
X	110
Y	340
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

リスト 10(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList10
X	110
Y	365
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

## GENWARE3 アプリケーション設計ガイド

リスト 11(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList11
X	110
Y	390
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

リスト 12(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList12
X	110
Y	415
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

リスト 13(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList13
X	110
Y	440
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

リスト 14(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList14
X	110
Y	465
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

リスト 15(スタティックテキスト)

項 目	設 定 内 容
コントロール名	testErrorList15
X	110
Y	490
WIDTH	770
HEIGHT	25
文字色	RGB(192,192,192)
横位置	左寄せ
縦位置	中央

スクロールバー(垂直スクロールバー)

項 目	設 定 内 容
コントロール名	scrollBarError
X	880
Y	115
WIDTH	30
HEIGHT	400
スクロール最小値	0
スクロール最大値	49
1 ページサイズ	15
コールバック関数	OnScroll

## GENWARE3 アプリケーション設計ガイド

## ③OLD→NEW ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnOldToNew
X	110
Y	522
WIDTH	120
HEIGHT	35
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_errorOrderBotton
OFF 時意匠	img_errorOrderBotton
文字列	ID_STRING_OrderOldToNew OLD→NEW
フォント	ID_FONT00000
コールバック関数	OnClick

## NEW→OLD ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnNewToOld
X	240
Y	522
WIDTH	120
HEIGHT	35
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_errorOrderBotton
OFF 時意匠	img_errorOrderBotton
文字列	ID_STRING_OrderNewToOld NEW→OLD
フォント	ID_FONT00000
コールバック関数	OnClick



## GENWARE3 アプリケーション設計ガイド

## HIGH→LOW ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnHighToLow
X	370
Y	522
WIDTH	120
HEIGHT	35
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_errorOrderBotton
OFF 時意匠	img_errorOrderBotton
文字列	ID_STRING_OrderHighToLow HIGH→LOW
フォント	ID_FONT00000
コールバック関数	OnClick

## LOW→HIGH ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnLowToHigh
X	500
Y	522
WIDTH	120
HEIGHT	35
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_errorOrderBotton
OFF 時意匠	img_errorOrderBotton
文字列	ID_STRING_OrderLowToHigh LOW→HIGH
フォント	ID_FONT00000
コールバック関数	OnClick

## ④タイトル(スタティックテキスト)

項 目	設 定 内 容
コントロール名	textNowError
X	110
Y	598
WIDTH	240
HEIGHT	30
背景色	RGB(255,0,0)
文字列	ID_STRING_NowError 現在発生中エラー
横位置	中央
縦位置	中央
フォント	ID_FONT00000

## 現在発生中エラー(スタティックテキスト)

項 目	設 定 内 容
コントロール名	textErrorInfo
X	110
Y	628
WIDTH	800
HEIGHT	30
背景色	RGB(128,255,255)
横位置	中央
縦位置	中央
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

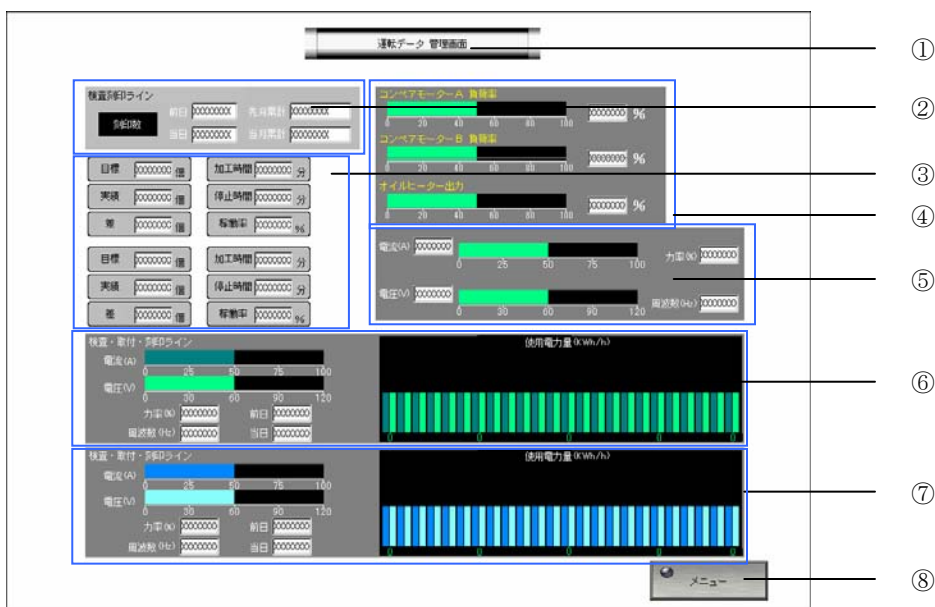
## ⑤履歴クリアボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnHistoryClear
X	660
Y	700
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_HistoryClear 履歴クリア
フォント	ID_FONT00000
コールバック関数	OnClick

## ⑥メニューボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnErrorManagementMenu
X	820
Y	700
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Menu メニュー
フォント	ID_FONT00000
コールバック関数	OnClick

## 運転データ画面(パネル名:Panel\_OperDataManagement)



番号	項目	使用コントロール
①	タイトル	ピクチャ
②	刻印数表示エリア	ピクチャ スタティックテキスト×7 テキストボックス×4
③	目標・実績値表示エリア	ピクチャ×2 スタティックテキスト×22 テキストボックス×12
④	コンペアモーター負荷率、オイルヒーター出力 表示エリア	ピクチャ スタティックテキスト×3 テキストボックス×3 プログレスバー×3
⑤	電流・電圧表示エリア	ピクチャ スタティックテキスト×4 テキストボックス×4 プログレスバー×2

## GENWARE3 アプリケーション設計ガイド

⑥	検査・取付・刻印ライン 1 表示エリア	ピクチャ スタティックテキスト×8 テキストボックス×4 プログレスバー×50
⑦	検査・取付・刻印ライン 2 表示エリア	ピクチャ スタティックテキスト×8 テキストボックス×4 プログレスバー×50
⑧	メニューボタン	ボタン

## ①タイトル(ピクチャ)

項 目	設 定 内 容
コントロール名	pctOperDataMangementTitle
X	380
Y	20
WIDTH	300
HEIGHT	40
状態の数	1
状態 0 意匠	img_TitleBack
文字列	ID_STRING_OperDataManagementScreen 運転データ 管理画面
フォント	ID_FONT00000

## ②タイトル(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionTitle
X	104
Y	96
WIDTH	200
HEIGHT	20
背景色	RGB(192,192,192)
文字列	ID_STRING00219 検査刻印ライン
縦位置	RGB(0,0,0)
横位置	左寄せ
フォント	中央

## 背景(ピクチャ)

項 目	設 定 内 容
コントロール名	pctInspectionMounting
X	100
Y	95
WIDTH	350
HEIGHT	80
状態の数	1
状態 0 意匠	img_InspectionStampingLine

## 刻印数(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionNum
X	117
Y	128
WIDTH	75
HEIGHT	30
背景色	RGB(0,0,0)
文字列	ID_STRING00220
	刻印数
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 刻印数(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionNum
X	117
Y	128
WIDTH	75
HEIGHT	30
背景色	RGB(0,0,0)
文字列	ID_STRING00220 刻印数
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 前日(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionBefore
X	208
Y	118
WIDTH	22
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00221 前日
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 前日の値(テキストボックス)

項 目	設 定 内 容
コントロール名	textDispensingDeliveryYester
X	235
Y	115
WIDTH	60
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## 当日(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionNow
X	208
Y	146
WIDTH	22
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00222
	当日
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央



## GENWARE3 アプリケーション設計ガイド

## 当日の値(テキストボックス)

項 目	設 定 内 容
コントロール名	textDispensingDeliveryToday
X	235
Y	145
WIDTH	60
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## 先月累計(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionBeforeMonth
X	308
Y	118
WIDTH	48
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00223
	先月累計
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## GENWARE3 アプリケーション設計ガイド

## 先月累計の値(テキストボックス)

項 目	設 定 内 容
コントロール名	textTotalPreMonth
X	360
Y	115
WIDTH	80
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## 当月累計(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionNowMonth
X	308
Y	146
WIDTH	48
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00224 当月累計
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## GENWARE3 アプリケーション設計ガイド

## 当月累計の値(テキストボックス)

項 目	設 定 内 容
コントロール名	textTotalCurMonth
X	360
Y	145
WIDTH	80
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## ③背景 1(ピクチャ)

項 目	設 定 内 容
コントロール名	pctOneGroup
X	100
Y	180
WIDTH	290
HEIGHT	110
状態の数	1
状態 0 意匠	img_Figure

## 目標 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionUpGoal
X	120
Y	190
WIDTH	24
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00225
	目標
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 目標値 1(テキストボックス)

項 目	設 定 内 容
コントロール名	textTargetUp
X	162
Y	190
WIDTH	50
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## 個 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionUpInd1
X	215
Y	195
WIDTH	12
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00242
	個
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 実績 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionUpPer
X	120
Y	225
WIDTH	24
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00226 実績
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 実績値 1(テキストボックス)

項 目	設 定 内 容
コントロール名	textPracticalUp
X	162
Y	225
WIDTH	50
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## 個 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionUpInd2
X	215
Y	230
WIDTH	12
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00242 個
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 差 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionUpDiff
X	120
Y	260
WIDTH	24
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00227 差
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 差の値 1(テキストボックス)

項 目	設 定 内 容
コントロール名	textPoorUp
X	162
Y	260
WIDTH	50
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## 個 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionUpInd3
X	215
Y	265
WIDTH	12
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00242
	個
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## GENWARE3 アプリケーション設計ガイド

## 加工時間 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionUpMach
X	265
Y	190
WIDTH	48
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00228
	加工時間
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 加工時間の値 1(テキストボックス)

項 目	設 定 内 容
コントロール名	textWorkTime
X	315
Y	190
WIDTH	50
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000



## 分 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionUpMinute1
X	370
Y	195
WIDTH	12
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00243 分
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 停止時間 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionUpStop
X	265
Y	225
WIDTH	48
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00229 停止時間
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 停止時間の値 1(テキストボックス)

項 目	設 定 内 容
コントロール名	textStopTime
X	315
Y	225
WIDTH	50
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## 分 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionUpMinute2
X	370
Y	230
WIDTH	12
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00243
	分
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## GENWARE3 アプリケーション設計ガイド

## 稼働率 1 (スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionUpCap
X	265
Y	260
WIDTH	48
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00230 稼働率
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 稼働率の値 1 (テキストボックス)

項 目	設 定 内 容
コントロール名	textRatioUp
X	315
Y	260
WIDTH	50
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	Gfloat
表示フォーマット	%.1f
最大文字数	8
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 背景 2(ピクチャ)

項 目	設 定 内 容
コントロール名	pctTwoGroup
X	100
Y	295
WIDTH	290
HEIGHT	110
状態の数	1
状態 0 意匠	img_Figure

## 目標 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionDnGoal
X	120
Y	305
WIDTH	24
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00225 目標
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 目標値 2(テキストボックス)

項 目	設 定 内 容
コントロール名	textTargetDown
X	162
Y	305
WIDTH	50
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## 個 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionDnInd1
X	215
Y	310
WIDTH	12
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00242 個
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 実績 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionDnPer
X	120
Y	340
WIDTH	24
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00226 実績
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 実績値 2(テキストボックス)

項 目	設 定 内 容
コントロール名	textPracticalDown
X	162
Y	340
WIDTH	50
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## 個 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionDnInd2
X	215
Y	345
WIDTH	12
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00242
	個
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## GENWARE3 アプリケーション設計ガイド

## 差 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionDnDiff
X	120
Y	375
WIDTH	24
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00227
	差
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 差の値 2(テキストボックス)

項 目	設 定 内 容
コントロール名	textPoorDown
X	162
Y	375
WIDTH	50
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 個 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionDnInd3
X	215
Y	380
WIDTH	12
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00242 個
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 加工時間 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionDnMach
X	265
Y	305
WIDTH	48
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00228 加工時間
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央



## 加工時間の値 2(テキストボックス)

項 目	設 定 内 容
コントロール名	textWorkTimeDown
X	315
Y	305
WIDTH	50
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## 分 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionDnMinute1
X	370
Y	310
WIDTH	12
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00243
	分
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## GENWARE3 アプリケーション設計ガイド

## 停止時間 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionDnStop
X	265
Y	340
WIDTH	48
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00229 停止時間
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 停止時間の値 2(テキストボックス)

項 目	設 定 内 容
コントロール名	textStopTimeDown
X	315
Y	340
WIDTH	50
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## 分 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionDnMinute2
X	370
Y	345
WIDTH	12
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00243 分
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 稼働率 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperInspectionDnCap
X	265
Y	375
WIDTH	48
HEIGHT	18
背景色	RGB(192,192,192)
文字列	ID_STRING00230 稼働率
縦位置	RGB(0,0,0)
横位置	中央
フォント	中央

## 稼働率の値 2(テキストボックス)

項 目	設 定 内 容
コントロール名	textConveyor1
X	740
Y	117
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	Gfloat
表示フォーマット	%.1f
最大文字数	8
フォント	ID_FONT00000

## ④背景(ピクチャ)

項 目	設 定 内 容
コントロール名	pctLoadFactor
X	470
Y	95
WIDTH	370
HEIGHT	175
状態の数	1
状態 0 意匠	img_LoadFactor

## コンペアモーターA 負荷率(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperCompareA
X	475
Y	95
WIDTH	300
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00231 コンペアモーターA 負荷率
縦位置	RGB(255,255,0)
横位置	左寄せ
フォント	中央

## コンペアモーターA 負荷率 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarConveyor1
X	485
Y	114
WIDTH	228
HEIGHT	18
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	左→右
最小値	0
最大値	100

## GENWARE3 アプリケーション設計ガイド

## コンペアモーターA 負荷率 値(テキストボックス)

項 目	設 定 内 容
コントロール名	textConveyor2
X	740
Y	175
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	Gfloat
表示フォーマット	%.1f
最大文字数	8
フォント	ID_FONT00000

## コンペアモーターB 負荷率(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperCompareB
X	475
Y	152
WIDTH	300
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00232 コンペアモーターB 負荷率
縦位置	RGB(255,255,0)
横位置	左寄せ
フォント	中央

## GENWARE3 アプリケーション設計ガイド

## コンペアモーターB 負荷率 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarConveyor2
X	485
Y	171
WIDTH	228
HEIGHT	18
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	左→右
最小値	0
最大値	100

## コンペアモーターB 負荷率 値(テキストボックス)

項 目	設 定 内 容
コントロール名	textConveyor2
X	740
Y	175
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	Gfloat
表示フォーマット	%.1f
最大文字数	8
フォント	ID_FONT00000

## オイルヒーター出力(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperHeaterOutput
X	475
Y	212
WIDTH	300
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00233 オイルヒーター出力
縦位置	RGB(255,255,0)
横位置	左寄せ
フォント	中央

## コンペアモーターB 負荷率 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarConveyor3
X	485
Y	231
WIDTH	228
HEIGHT	18
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	左→右
最小値	0
最大値	100



## GENWARE3 アプリケーション設計ガイド

## コンペアモーターB 負荷率 値(テキストボックス)

項 目	設 定 内 容
コントロール名	textConveyor3
X	740
Y	235
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	Gfloat
表示フォーマット	%.1f
最大文字数	8
フォント	ID_FONT00000

## ⑤背景(ピクチャ)

項 目	設 定 内 容
コントロール名	pctShowBar
X	470
Y	275
WIDTH	470
HEIGHT	118
状態の数	1
状態 0 意匠	img_CurrentVoltage

## 電流(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperCurrent
X	475
Y	290
WIDTH	24
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00234 電流
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## GENWARE3 アプリケーション設計ガイド

## 電流 値(テキストボックス)

項 目	設 定 内 容
コントロール名	textElec
X	518
Y	287
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	Gfloat
表示フォーマット	%.1f
最大文字数	8
フォント	ID_FONT00000

## 電流 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPower
X	576
Y	294
WIDTH	228
HEIGHT	18
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	左→右
最小値	0
最大値	100

## 電圧(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperVoltage
X	475
Y	350
WIDTH	24
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00235 電圧
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 電圧 値(テキストボックス)

項 目	設 定 内 容
コントロール名	textVol
X	518
Y	348
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	Gfloat
表示フォーマット	%.1f
最大文字数	8
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 電流 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarCycle
X	576
Y	354
WIDTH	228
HEIGHT	18
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	左→右
最小値	0
最大値	100

## 力率(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperPowerFactor
X	840
Y	303
WIDTH	24
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00236
	力率
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 力率 値(テキストボックス)

項 目	設 定 内 容
コントロール名	textPower
X	882
Y	298
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	Gfloat
表示フォーマット	%.1f
最大文字数	8
フォント	ID_FONT00000

## 周波数(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperFrequency
X	822
Y	362
WIDTH	36
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00237
	周波数
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## GENWARE3 アプリケーション設計ガイド

## 周波数 値(テキストボックス)

項 目	設 定 内 容
コントロール名	textCycle
X	882
Y	360
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	Gfloat
表示フォーマット	%.1f
最大文字数	8
フォント	ID_FONT00000

## ⑥背景(ピクチャ)

項 目	設 定 内 容
コントロール名	pctCarvedSealMountingInspection
X	100
Y	410
WIDTH	840
HEIGHT	140
状態の数	1
状態 0 意匠	img_InspectionMountingCarvedSeal

## 検査・取付・刻印ライン(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperIIELine
X	105
Y	410
WIDTH	300
HEIGHT	20
背景色	RGB(192,192,192)
文字列	ID_STRING00238 検査・取付・刻印ライン
縦位置	RGB(255,255,255)
横位置	左寄せ
フォント	中央

## 電流 (スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperUpCurrent
X	124
Y	435
WIDTH	24
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00234 電流
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 電流 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarElceImprint
X	176
Y	431
WIDTH	228
HEIGHT	18
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	左→右
最小値	0
最大値	100

## GENWARE3 アプリケーション設計ガイド

## 電圧 (スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperUpVoltage
X	124
Y	470
WIDTH	24
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00235 電圧
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 電圧 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarVollkprint
X	176
Y	464
WIDTH	228
HEIGHT	18
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	左→右
最小値	0
最大値	100



## GENWARE3 アプリケーション設計ガイド

## 力率(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperUpPowerFactor
X	175
Y	503
WIDTH	24
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00236 力率
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 力率 値(テキストボックス)

項 目	設 定 内 容
コントロール名	textPowerImprint
X	220
Y	500
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	Gfloat
表示フォーマット	%.1f
最大文字数	8
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 周波数(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperUpFrequency
X	155
Y	527
WIDTH	36
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00237 周波数
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 周波数 値(テキストボックス)

項 目	設 定 内 容
コントロール名	textCycleImprint
X	220
Y	526
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	Gfloat
表示フォーマット	%.1f
最大文字数	8
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 前日 (スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperUpDayBefore
X	310
Y	503
WIDTH	24
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00221 前日
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 前日 値 (テキストボックス)

項 目	設 定 内 容
コントロール名	textYesterImprint
X	338
Y	500
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 当日 (スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperUpDayNow
X	310
Y	527
WIDTH	24
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00222
	当日
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 当日 値 (テキストボックス)

項 目	設 定 内 容
コントロール名	textTodayImprint
X	338
Y	526
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 使用電力量 (スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperUpElectricPower
X	600
Y	412
WIDTH	120
HEIGHT	18
背景色	RGB(0,0,0)
文字列	ID_STRING00239 使用電力量
縦位置	RGB(255,255,255)
横位置	右寄せ
フォント	中央

## 電流 1 グラフ (プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint01A
X	479
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 1 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint01B
X	488
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 2 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint02A
X	498
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 2 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint02B
X	507
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 3 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint03A
X	517
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 3 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint03B
X	526
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 4 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint04A
X	536
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100



## 電圧 4 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint04B
X	545
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 5 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint05A
X	555
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 5 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint05B
X	564
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 6 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint06A
X	574
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 6 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint06B
X	583
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 7 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint07A
X	593
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 7 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint07B
X	602
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 8 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint08A
X	612
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 8 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint08B
X	621
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 9 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint09A
X	631
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 9 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint09B
X	640
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 10 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint10A
X	650
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 10 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint10B
X	659
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 11 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint11A
X	669
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 11 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint11B
X	678
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 12 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint12A
X	688
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100



## 電圧 12 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint12B
X	697
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 13 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint13A
X	707
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 13 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint13B
X	716
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 14 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint14A
X	726
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 14 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint14B
X	735
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 15 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint15A
X	745
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 15 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint15B
X	754
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 16 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint16A
X	764
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 16 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint16B
X	773
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 17 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint17A
X	783
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 17 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint17B
X	792
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 18 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint18A
X	802
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 18 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint18B
X	811
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 19 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint19A
X	821
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 19 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint19B
X	830
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 20 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint20A
X	840
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100



## 電圧 20 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint20B
X	849
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 21 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint21A
X	859
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 21 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint21B
X	868
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 22 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint22A
X	878
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 22 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint22B
X	887
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 23 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint23A
X	897
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 23 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint23B
X	906
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 24 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint24A
X	916
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,128)
バー 前景色	RGB(0,128,128)
方向	下→上
最小値	0
最大値	100

## 電圧 24 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint24B
X	925
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 時刻 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperUpTimeOne
X	477
Y	537
WIDTH	24
HEIGHT	10
背景色	RGB(0,0,0)
文字列	ID_STRING_MaintenanceNum0
	0
縦位置	RGB(0,255,128)
横位置	中央
フォント	中央

## GENWARE3 アプリケーション設計ガイド

## 時刻 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperUpTimeTwo
X	591
Y	537
WIDTH	24
HEIGHT	10
背景色	RGB(0,0,0)
文字列	ID_STRING_MaintenanceNum0 0
縦位置	RGB(0,255,128)
横位置	中央
フォント	中央

## 時刻 3(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperUpTimeThree
X	705
Y	537
WIDTH	24
HEIGHT	10
背景色	RGB(0,0,0)
文字列	ID_STRING_MaintenanceNum0 0
縦位置	RGB(0,255,128)
横位置	中央
フォント	中央

## 時刻 4(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperUpTimeFour
X	820
Y	537
WIDTH	24
HEIGHT	10
背景色	RGB(0,0,0)
文字列	ID_STRING_MaintenanceNum0 0
縦位置	RGB(0,255,128)
横位置	中央
フォント	中央

## 時刻 5(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperUpTimeFive
X	914
Y	537
WIDTH	24
HEIGHT	10
背景色	RGB(0,0,0)
文字列	ID_STRING_MaintenanceNum0 0
縦位置	RGB(0,255,128)
横位置	中央
フォント	中央

## ⑦背景(ピクチャ)

項 目	設 定 内 容
コントロール名	pctDispensingLine
X	100
Y	555
WIDTH	840
HEIGHT	140
状態の数	1
状態 0 意匠	img_DispensingLineScreen

## 検査・取付・刻印ライン(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperDispensingLine
X	105
Y	555
WIDTH	300
HEIGHT	20
背景色	RGB(128,128,128)
文字列	ID_STRING00238 検査・取付・刻印ライン
縦位置	RGB(255,255,255)
横位置	左寄せ
フォント	中央



## 電流 (スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperDnCurrent
X	124
Y	582
WIDTH	24
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00234 電流
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 電流 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarElceDelivery
X	176
Y	576
WIDTH	228
HEIGHT	18
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	左→右
最小値	0
最大値	100

## GENWARE3 アプリケーション設計ガイド

## 電圧 (スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperDnVoltage
X	124
Y	615
WIDTH	24
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00235 電圧
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 電圧 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarVolDelivery
X	176
Y	609
WIDTH	228
HEIGHT	18
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	左→右
最小値	0
最大値	100

## GENWARE3 アプリケーション設計ガイド

## 力率(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperDnPowerFactor
X	175
Y	647
WIDTH	24
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00236 力率
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 力率 値(テキストボックス)

項 目	設 定 内 容
コントロール名	textPowerDelivery
X	220
Y	644
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	Gfloat
表示フォーマット	%.1f
最大文字数	8
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 周波数(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperDnFrequency
X	155
Y	673
WIDTH	36
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00237 周波数
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 周波数 値(テキストボックス)

項 目	設 定 内 容
コントロール名	textCycleDelivery
X	220
Y	670
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	Gfloat
表示フォーマット	%.1f
最大文字数	8
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 前日 (スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperDnDayBefore
X	310
Y	647
WIDTH	24
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00221 前日
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 前日 値 (テキストボックス)

項 目	設 定 内 容
コントロール名	textYesterDelivery
X	338
Y	644
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 当日 (スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperDnDayNow
X	310
Y	673
WIDTH	24
HEIGHT	18
背景色	RGB(128,128,128)
文字列	ID_STRING00222
	当日
縦位置	RGB(255,255,255)
横位置	中央
フォント	中央

## 当日 値 (テキストボックス)

項 目	設 定 内 容
コントロール名	textTodayDelivery
X	338
Y	670
WIDTH	51
HEIGHT	23
背景色	RGB(255,255,255)
タイプ	GUInt16
表示フォーマット	%d
最大文字数	8
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## 使用電力量 (スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperDnElectricPower
X	600
Y	557
WIDTH	120
HEIGHT	18
背景色	RGB(0,0,0)
文字列	ID_STRING00239 使用電力量
縦位置	RGB(255,255,255)
横位置	右寄せ
フォント	中央

## 電流 1 グラフ (プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery01A
X	479
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## GENWARE3 アプリケーション設計ガイド

## 電圧 1 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery01B
X	488
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 2 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery02A
X	498
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100



## 電圧 2 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery02B
X	507
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 3 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery03A
X	517
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## GENWARE3 アプリケーション設計ガイド

## 電圧 3 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery03B
X	526
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 4 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery04A
X	536
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 4 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery04B
X	545
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 5 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery05A
X	555
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 5 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery05B
X	564
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 6 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery06A
X	574
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 6 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery06B
X	583
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 7 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery07A
X	593
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 7 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery07B
X	602
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 8 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery08A
X	612
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 8 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery08B
X	621
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 9 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery09A
X	631
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 9 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery09B
X	640
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 10 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery10A
X	650
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100



## GENWARE3 アプリケーション設計ガイド

## 電圧 10 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery10B
X	659
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 11 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery11A
X	669
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 11 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery11B
X	678
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 12 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery12A
X	688
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 12 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery12B
X	697
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 13 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery13A
X	707
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 13 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery13B
X	716
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 14 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery14A
X	726
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 14 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarPrint14B
X	735
Y	436
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,255,128)
バー 前景色	RGB(0,255,128)
方向	下→上
最小値	0
最大値	100

## 電流 15 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery15A
X	745
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 15 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery15B
X	754
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 16 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery16A
X	764
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 16 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery16B
X	773
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 17 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery17A
X	783
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 17 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery17B
X	792
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 18 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery18A
X	802
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100



## 電圧 18 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery18B
X	811
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 19 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery19A
X	821
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 19 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery19B
X	830
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 20 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery20A
X	840
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 20 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery20B
X	849
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 21 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery21A
X	859
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 21 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery21B
X	868
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 22 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery22A
X	878
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## GENWARE3 アプリケーション設計ガイド

## 電圧 22 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery22B
X	887
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 23 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery23A
X	897
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## 電圧 23 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery23B
X	906
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 電流 24 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery24A
X	916
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(0,128,255)
バー 前景色	RGB(0,128,255)
方向	下→上
最小値	0
最大値	100

## GENWARE3 アプリケーション設計ガイド

## 電圧 24 グラフ(プログレスバー)

項 目	設 定 内 容
コントロール名	progBarDelivery24B
X	925
Y	581
WIDTH	8
HEIGHT	100
本体 背景色	RGB(0,0,0)
本体 前景色	RGB(0,0,0)
バー 背景色	RGB(128,255,255)
バー 前景色	RGB(128,255,255)
方向	下→上
最小値	0
最大値	100

## 時刻 1(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperDnTimeOne
X	477
Y	683
WIDTH	24
HEIGHT	10
背景色	RGB(0,0,0)
文字列	ID_STRING_MaintenanceNum0
	0
縦位置	RGB(0,255,128)
横位置	中央
フォント	中央

## 時刻 2(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperDnTimeTwo
X	591
Y	683
WIDTH	24
HEIGHT	10
背景色	RGB(0,0,0)
文字列	ID_STRING_MaintenanceNum0 0
縦位置	RGB(0,255,128)
横位置	中央
フォント	中央

## 時刻 3(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperDnTimeThree
X	705
Y	682
WIDTH	24
HEIGHT	10
背景色	RGB(0,0,0)
文字列	ID_STRING_MaintenanceNum0 0
縦位置	RGB(0,255,128)
横位置	中央
フォント	中央



## 時刻 4(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperDnTimeFour
X	820
Y	683
WIDTH	24
HEIGHT	10
背景色	RGB(0,0,0)
文字列	ID_STRING_MaintenanceNum0 0
縦位置	RGB(0,255,128)
横位置	中央
フォント	中央

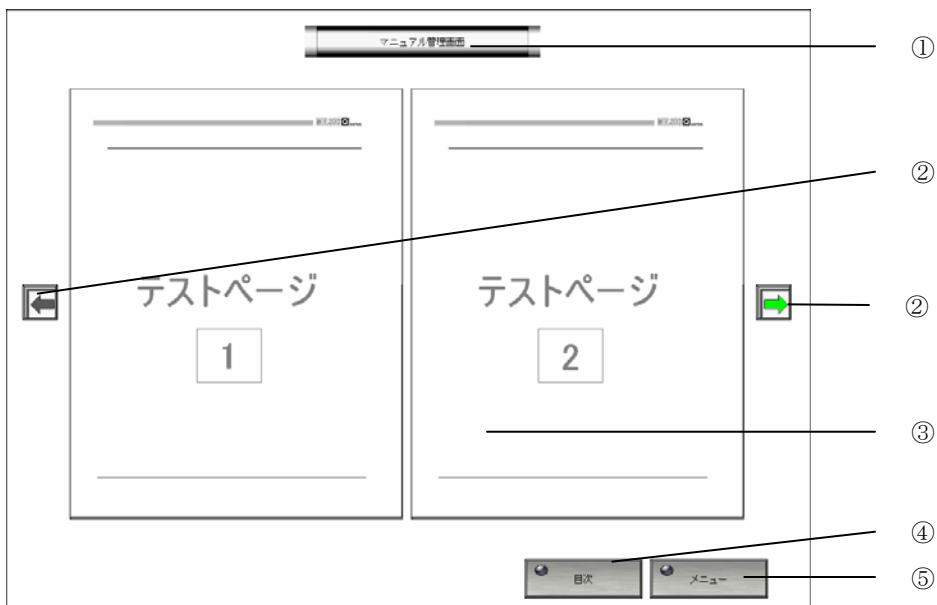
## 時刻 5(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stextOperDnTimeFive
X	914
Y	683
WIDTH	24
HEIGHT	10
背景色	RGB(0,0,0)
文字列	ID_STRING_MaintenanceNum0 0
縦位置	RGB(0,255,128)
横位置	中央
フォント	中央

## ⑧メニューボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnOperDataManagementMenu
X	820
Y	700
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Menu メニュー
フォント	ID_FONT00000
コールバック関数	OnClick

## マニュアル管理画面(パネル名:Panel\_ManualManagement)



番号	項目	使用コントロール
①	タイトル	ピクチャ
②	前／次ページ	ボタン×2
③	マニュアル表示	ボタン
④	目次	ボタン
⑤	メニューボタン	ボタン

## ①タイトル(ピクチャ)

項 目	設 定 内 容
コントロール名	pctManualManagementTitle
X	380
Y	20
WIDTH	300
HEIGHT	40
状態の数	1
状態 0 意匠	img_TitleBack
文字列	ID_STRING_ManualManagementScreen マニュアル管理画面
フォント	ID_FONT00000

## ②前ページボタン(ボタン)

項 目	設 定 内 容
コントロール名	GButton_Left
X	20
Y	350
WIDTH	45
HEIGHT	45
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
フォント	ID_FONT00000
コールバック関数	OnClick

## 次ページボタン(ボタン)

項 目	設 定 内 容
コントロール名	GButton_Right
X	955
Y	350
WIDTH	45
HEIGHT	45
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
フォント	ID_FONT00000
コールバック関数	OnClick

## ③マニュアル表示(左)(ピクチャ)

項 目	設 定 内 容
コントロール名	GPic_Left
X	80
Y	100
WIDTH	425
HEIGHT	550
状態の数	9
状態 0 意匠	img_Page01
状態 1 意匠	img_Page02
状態 2 意匠	img_Page03
状態 3 意匠	img_Page04
状態 4 意匠	img_Page05
状態 5 意匠	img_Page06
状態 6 意匠	img_Page07
状態 7 意匠	img_Page08
状態 8 意匠	img_Page09

## マニュアル表示(右)(ピクチャ)

項 目	設 定 内 容
コントロール名	GPic_Right
X	515
Y	100
WIDTH	425
HEIGHT	550
状態の数	9
状態 0 意匠	img_Page02
状態 1 意匠	img_Page03
状態 2 意匠	img_Page04
状態 3 意匠	img_Page05
状態 4 意匠	img_Page06
状態 5 意匠	img_Page07
状態 6 意匠	img_Page08
状態 7 意匠	img_Page09
状態 8 意匠	img_Page10

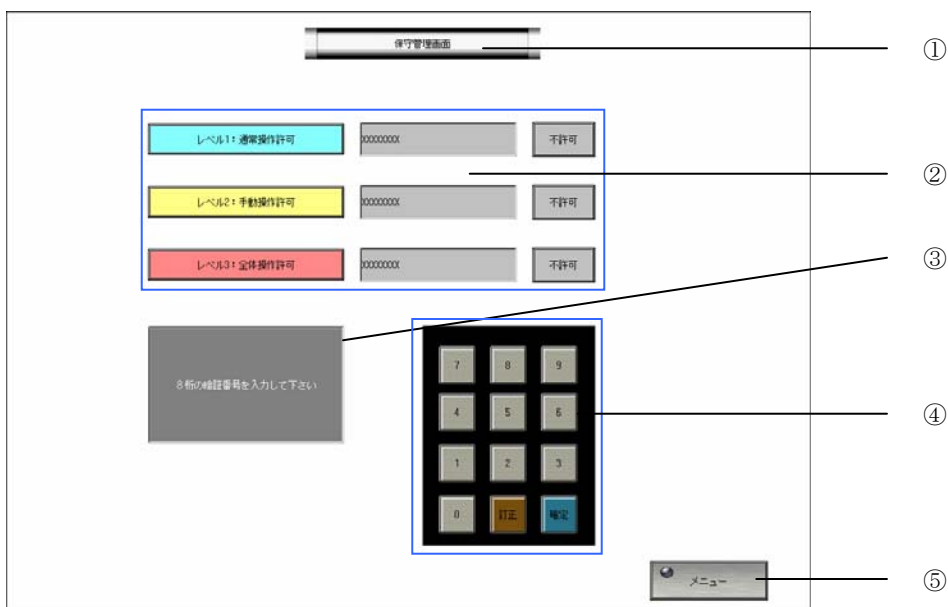
## ③目次ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnManualManagementMenu
X	820
Y	700
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Menu メニュー
フォント	ID_FONT00000
コールバック関数	OnClick

## ④メニューボタン(ボタン)

項 目	設 定 内 容
コントロール名	GButton_Contents
X	660
Y	700
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Contents 目次
フォント	ID_FONT00000
コールバック関数	OnClick

## 保守管理画面(パネル名:Panel\_Maintenance)



番号	項目	使用コントロール
①	タイトル	ピクチャ
②	ログイン	ボタン×6 テキストボックス×3
③	ガイダンス	スタティックテキスト
④	テンキー	ボタン×11 ピクチャ
⑤	メニューボタン	ボタン

## ①タイトル(ピクチャ)

項 目	設 定 内 容
コントロール名	pctMaintenanceTitle
X	380
Y	20
WIDTH	300
HEIGHT	40
状態の数	1
状態 0 意匠	img_TitleBack
文字列	ID_STRING_MaintenanceScreen 保守管理画面
フォント	ID_FONT00000

## ②レベル 1 ログインボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnMessageLevel1
X	180
Y	142
WIDTH	250
HEIGHT	40
ボタンタイプ	モーメンタリ
表示タイプ	矩形
ON 時 前景色	RGB(128,255,255)
ON 時 背景色	RGB(128,255,255)
OFF 時 前景色	RGB(128,255,255)
OFF 時 背景色	RGB(128,255,255)
文字列	ID_STRING_MaintenanceLevelMessage1 レベル 1: 通常操作許可
フォント	ID_FONT00000
コールバック関数	OnClick



## GENWARE3 アプリケーション設計ガイド

## レベル 1 パスワード入力欄(テキストボックス)

項 目	設 定 内 容
コントロール名	textPasswordLevel1
X	450
Y	140
WIDTH	200
HEIGHT	44
背景色	RGB(192,192,192)
タイプ	文字列
表示フォーマット	%s
最大文字数	8
パスワード設定	あり
フォント	ID_FONT00000

## レベル 1 ログイン状態(ボタン)

項 目	設 定 内 容
コントロール名	btnShowLevel1
X	670
Y	140
WIDTH	80
HEIGHT	45
ボタンタイプ	モーメンタリ
表示タイプ	矩形
ON 時 前景色	RGB(192,192,192)
ON 時 背景色	RGB(192,192,192)
OFF 時 前景色	RGB(192,192,192)
OFF 時 背景色	RGB(192,192,192)
文字列	ID_STRING_MaintenanceNotpermitted
	不許可
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## レベル 2 ログインボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnMessageLevel2
X	180
Y	222
WIDTH	250
HEIGHT	40
ボタンタイプ	モーメンタリ
表示タイプ	矩形
ON 時 前景色	RGB(255,255,128)
ON 時 背景色	RGB(255,255,128)
OFF 時 前景色	RGB(255,255,128)
OFF 時 背景色	RGB(255,255,128)
文字列	ID_STRING_MaintenanceLevelMessage2 レベル 2: 手動操作許可
フォント	ID_FONT00000
コールバック関数	OnClick

## レベル 2 パスワード入力欄(テキストボックス)

項 目	設 定 内 容
コントロール名	textPasswordLevel2
X	450
Y	220
WIDTH	200
HEIGHT	44
背景色	RGB(192,192,192)
タイプ	文字列
表示フォーマット	%s
最大文字数	8
パスワード設定	あり
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## レベル 2 ログイン状態(ボタン)

項 目	設 定 内 容
コントロール名	btnShowLevel2
X	670
Y	220
WIDTH	80
HEIGHT	45
ボタンタイプ	モーメンタリ
表示タイプ	矩形
ON 時 前景色	RGB(192,192,192)
ON 時 背景色	RGB(192,192,192)
OFF 時 前景色	RGB(192,192,192)
OFF 時 背景色	RGB(192,192,192)
文字列	ID_STRING_MaintenanceNotpermitted
	不許可
フォント	ID_FONT00000

## レベル 3 ログインボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnMessageLevel3
X	180
Y	302
WIDTH	250
HEIGHT	40
ボタンタイプ	モーメンタリ
表示タイプ	矩形
ON 時 前景色	RGB(255,128,128)
ON 時 背景色	RGB(255,128,128)
OFF 時 前景色	RGB(255,128,128)
OFF 時 背景色	RGB(255,128,128)
文字列	ID_STRING_MaintenanceLevelMessage3
	レベル 3: 全体操作許可
フォント	ID_FONT00000
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## レベル 3 パスワード入力欄(テキストボックス)

項 目	設 定 内 容
コントロール名	textPasswordLevel3
X	450
Y	300
WIDTH	200
HEIGHT	44
背景色	RGB(192,192,192)
タイプ	文字列
表示フォーマット	%s
最大文字数	8
パスワード設定	あり
フォント	ID_FONT00000

## レベル 3 ログイン状態(ボタン)

項 目	設 定 内 容
コントロール名	btnShowLevel3
X	670
Y	300
WIDTH	80
HEIGHT	45
ボタンタイプ	モーメンタリ
表示タイプ	矩形
ON 時 前景色	RGB(192,192,192)
ON 時 背景色	RGB(192,192,192)
OFF 時 前景色	RGB(192,192,192)
OFF 時 背景色	RGB(192,192,192)
文字列	ID_STRING_MaintenanceNotpermitted
	不許可
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## ③ ガイダンス(スタティックテキスト)

項 目	設 定 内 容
コントロール名	textPasswordMessage
X	180
Y	400
WIDTH	250
HEIGHT	150
背景色	RGB(128,128,128)
文字列	ID_STRING_MaintenanceInfo 8桁の暗証番号を入力して下さい
文字色	RGB(255,255,255)
横位置	中央
縦位置	中央
フォント	ID_FONT00000

## ④0 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnNum0
X	550
Y	615
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum0 0
フォント	ID_FONT00000
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## 1 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnNum1
X	550
Y	550
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum1 1
フォント	ID_FONT00000
コールバック関数	OnClick

## 2 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnNum2
X	615
Y	550
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum2 2
フォント	ID_FONT00000
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## 3 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnNum3
X	680
Y	550
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum3 3
フォント	ID_FONT00000
コールバック関数	OnClick

## 4 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnNum4
X	550
Y	485
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum4 4
フォント	ID_FONT00000
コールバック関数	OnClick

## 5 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnNum5
X	615
Y	485
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum5 5
フォント	ID_FONT00000
コールバック関数	OnClick

## 6 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnNum6
X	680
Y	485
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum6 6
フォント	ID_FONT00000
コールバック関数	OnClick



## 7 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnNum7
X	550
Y	425
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum7 7
フォント	ID_FONT00000
コールバック関数	OnClick

## 8 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnNum8
X	615
Y	425
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum8 8
フォント	ID_FONT00000
コールバック関数	OnClick

## 9 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnNum9
X	680
Y	425
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum9 9
フォント	ID_FONT00000
コールバック関数	OnClick

## 訂正ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnNumCancel
X	615
Y	615
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_CorrectionButton_ON
ON 時 意匠	img_CorrectionButton_OFF
文字列	ID_STRING_MaintenanceCorrection 訂正
フォント	ID_FONT00000
コールバック関数	OnClick

## 確定ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnNumOk
X	680
Y	615
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_ConfigButton_ON
ON 時 意匠	img_ConfigButton_OFF
文字列	ID_STRING_MaintenanceConfig 確定
フォント	ID_FONT00000
コールバック関数	OnClick

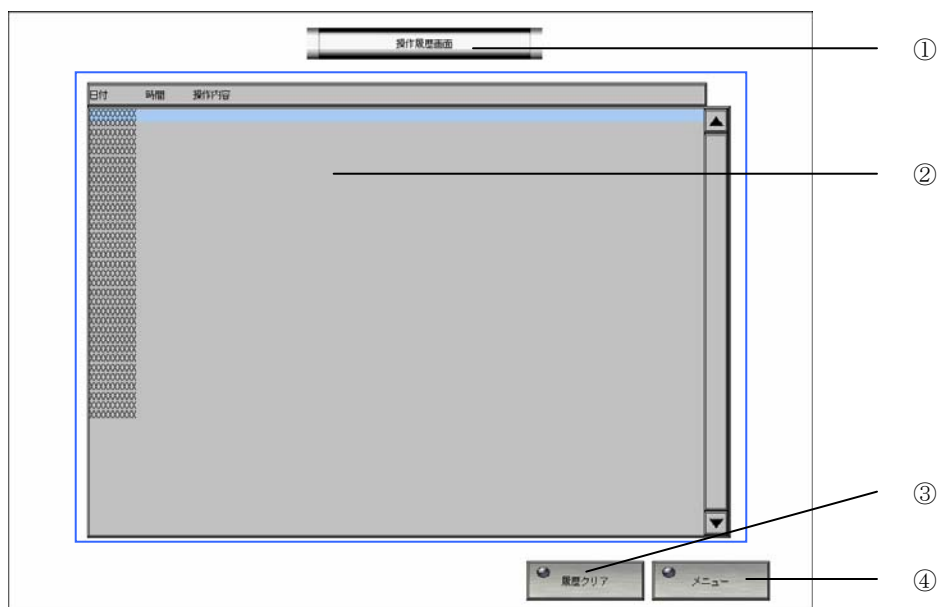
## 背景(ピクチャ)

項 目	設 定 内 容
コントロール名	pctNumBack
X	530
Y	400
WIDTH	220
HEIGHT	280
状態の数	1
状態 0 背景色	RGB(0,0,0)

## ⑤メニューボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnOperDataManagementMenu
X	820
Y	700
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Menu メニュー
フォント	ID_FONT00000
コールバック関数	OnClick

## 操作履歴画面(パネル名:Panel\_OperationHistory)



番号	項目	使用コントロール
①	タイトル	ピクチャ
②	操作履歴	スタティックテキスト リスト
③	履歴クリア	ボタン
④	メニューボタン	ボタン

## ①タイトル(ピクチャ)

項 目	設 定 内 容
コントロール名	pctOperDataMangementTitle
X	380
Y	20
WIDTH	300
HEIGHT	40
状態の数	1
状態 0 意匠	img_TitleBack
文字列	ID_STRING_OperationHistoryTitle 操作履歴画面
フォント	ID_FONT00000

## ②項目名(スタティックテキスト)

項 目	設 定 内 容
コントロール名	textTableHead
X	100
Y	90
WIDTH	790
HEIGHT	30
背景色	RGB(192,192,192)
文字列	ID_STRING_OperationHistroyListTitle 日付      時間      操作内容
文字色	RGB(0,0,0)
縦位置	左寄せ
横位置	中央
フォント	ID_FONT00000

## 履歴一覧(リスト)

項 目	設 定 内 容
コントロール名	listOperationHistoryInfo
X	100
Y	120
WIDTH	820
HEIGHT	550
スクロールバー 幅	30
最大行数	100

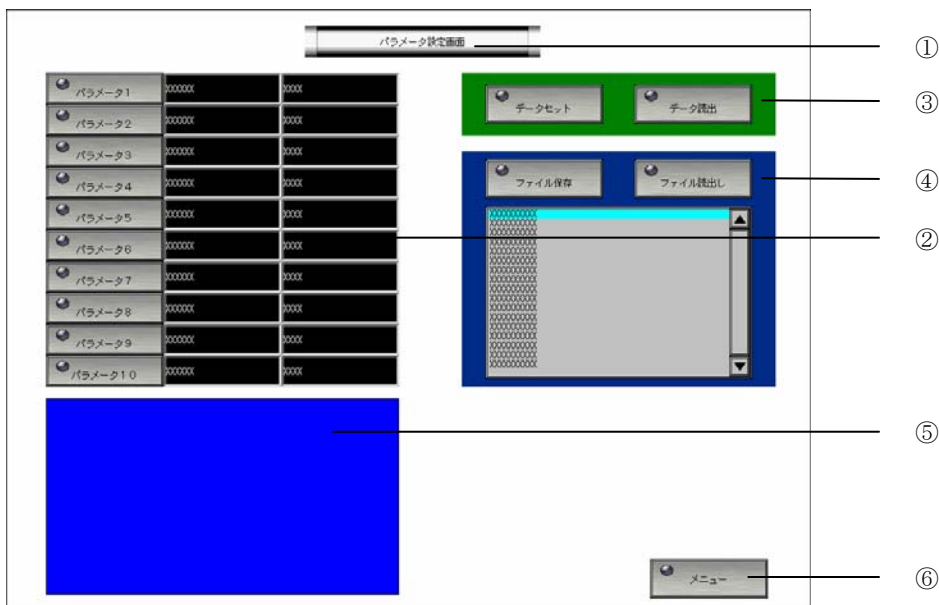
## ③履歴クリアボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnHistoryClear
X	660
Y	700
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_HistoryClear 履歴クリア
フォント	ID_FONT00000
コールバック関数	OnClick

## ④メニューボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnOperationHistoryMenu
X	820
Y	700
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Menu メニュー
フォント	ID_FONT00000
コールバック関数	OnClick

## パラメータ設定画面(パネル名:Panel\_ParameterSetting)



番号	項目	使用コントロール
①	タイトル	ピクチャ
②	パラメータ	ボタン×20 テキストボックス×20
③	シーケンサとの読書き	ボタン×2 ピクチャ
④	レシピファイル	ボタン×2 ピクチャ
⑤	イメージ表示エリア	矩形 ピクチャ
⑥	メニューボタン	ボタン



## ①タイトル(ピクチャ)

項 目	設 定 内 容
コントロール名	pctManualManagementTitle
X	380
Y	20
WIDTH	300
HEIGHT	40
状態の数	1
状態 0 意匠	img_TitleBack
文字列	ID_STRING_ParameterSetting パラメータ設定画面
フォント	ID_FONT00000

## ②パラメータ 1(ボタン)

項 目	設 定 内 容
コントロール名	btnParameters01
X	50
Y	80
WIDTH	150
HEIGHT	40
ボタンタイプ	なし
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Parameters1 パラメータ1
フォント	ID_FONT00000
コールバック関数	なし

## GENWARE3 アプリケーション設計ガイド

## パラメータ 1-1(テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxDecimal01
X	200
Y	80
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## パラメータ 1-2(テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxParameters01
X	350
Y	80
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## パラメータ 2 (ボタン)

項 目	設 定 内 容
コントロール名	btnParameters02
X	50
Y	120
WIDTH	150
HEIGHT	40
ボタンタイプ	なし
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Parameters2 パラメータ2
フォント	ID_FONT00000
コールバック関数	なし

## パラメータ 2-1 (テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxDecimal02
X	200
Y	120
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## パラメータ 2-2(テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxParameters02
X	350
Y	120
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## パラメータ 3(ボタン)

項 目	設 定 内 容
コントロール名	btnParameters03
X	50
Y	160
WIDTH	150
HEIGHT	40
ボタンタイプ	なし
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Parameters3 パラメータ3
フォント	ID_FONT00000
コールバック関数	なし

## パラメータ 3-1(テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxDecimal03
X	200
Y	160
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## パラメータ 3-2(テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxParameters03
X	350
Y	160
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## パラメータ 4 (ボタン)

項 目	設 定 内 容
コントロール名	btnParameters04
X	50
Y	200
WIDTH	150
HEIGHT	40
ボタンタイプ	なし
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Parameters4 パラメータ4
フォント	ID_FONT00000
コールバック関数	なし

## パラメータ 4-1 (テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxDecimal04
X	200
Y	200
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## パラメータ 4-2(テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxParameters04
X	350
Y	200
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## パラメータ 5(ボタン)

項 目	設 定 内 容
コントロール名	btnParameters05
X	50
Y	240
WIDTH	150
HEIGHT	40
ボタンタイプ	なし
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Parameters5 パラメータ5
フォント	ID_FONT00000
コールバック関数	なし

## パラメータ 5-1(テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxDecimal05
X	200
Y	240
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## パラメータ 5-2(テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxParameters05
X	350
Y	240
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000



## GENWARE3 アプリケーション設計ガイド

## パラメータ 6 (ボタン)

項 目	設 定 内 容
コントロール名	btnParameters06
X	50
Y	280
WIDTH	150
HEIGHT	40
ボタンタイプ	なし
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Parameters6 パラメータ6
フォント	ID_FONT00000
コールバック関数	なし

## パラメータ 6-1 (テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxDecimal06
X	200
Y	280
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## パラメータ 6-2(テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxParameters06
X	350
Y	280
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## パラメータ 7(ボタン)

項 目	設 定 内 容
コントロール名	btnParameters07
X	50
Y	320
WIDTH	150
HEIGHT	40
ボタンタイプ	なし
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Parameters7 パラメータ7
フォント	ID_FONT00000
コールバック関数	なし

## パラメータ 7-1(テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxDecimal07
X	200
Y	320
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## パラメータ 7-2(テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxParameters07
X	350
Y	320
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## パラメータ 8 (ボタン)

項 目	設 定 内 容
コントロール名	btnParameters08
X	50
Y	360
WIDTH	150
HEIGHT	40
ボタンタイプ	なし
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Parameters8 パラメータ8
フォント	ID_FONT00000
コールバック関数	なし

## パラメータ 8-1 (テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxDecimal08
X	200
Y	360
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## パラメータ 8-2 (テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxParameters08
X	350
Y	360
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## パラメータ 9 (ボタン)

項 目	設 定 内 容
コントロール名	btnParameters09
X	50
Y	400
WIDTH	150
HEIGHT	40
ボタンタイプ	なし
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Parameters9 パラメータ9
フォント	ID_FONT00000
コールバック関数	なし

## GENWARE3 アプリケーション設計ガイド

## パラメータ 9-1 (テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxDecimal09
X	200
Y	400
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## パラメータ 9-2 (テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxParameters09
X	350
Y	400
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## パラメータ 10(ボタン)

項 目	設 定 内 容
コントロール名	btnParameters10
X	50
Y	440
WIDTH	150
HEIGHT	40
ボタンタイプ	なし
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Parameters10 パラメータ10
フォント	ID_FONT00000
コールバック関数	なし

## パラメータ 10-1(テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxDecimal10
X	200
Y	440
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## GENWARE3 アプリケーション設計ガイド

## パラメータ 10-2(テキストボックス)

項 目	設 定 内 容
コントロール名	tBoxParameters10
X	350
Y	440
WIDTH	150
HEIGHT	40
背景色	RGB(0,0,0)
Focus 時 背景色	RGB(0,0,0)
Disable 時 背景色	RGB(0,0,0)
タイプ	文字列
表示フォーマット	%s
最大文字数	4
フォント	ID_FONT00000

## ③データセット(ボタン)

項 目	設 定 内 容
コントロール名	btnDataSet
X	610
Y	95
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_DataSet データセット
フォント	ID_FONT00000
コールバック関数	OnClick



## データ読出(ボタン)

項 目	設 定 内 容
コントロール名	btnDataLoad
X	800
Y	95
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_DataRead データ読出
フォント	ID_FONT00000
コールバック関数	OnClick

## 背景(ピクチャ)

項 目	設 定 内 容
コントロール名	pctFunctionArea
X	580
Y	80
WIDTH	400
HEIGHT	80
状態の数	1
状態 0 パターン	背景塗りつぶし
状態 0 背景色	RGB(0,128,0)

## GENWARE3 アプリケーション設計ガイド

## ④ファイル保存(ボタン)

項 目	設 定 内 容
コントロール名	btnSave
X	610
Y	190
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_FileSave ファイル保存
フォント	ID_FONT00000
コールバック関数	OnClick

## ファイル読出(ボタン)

項 目	設 定 内 容
コントロール名	btnLoad
X	800
Y	190
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_FileLoad ファイル読出し
フォント	ID_FONT00000
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## ファイル一覧(リスト)

項 目	設 定 内 容
コントロール名	listShowFileInfo
X	610
Y	250
WIDTH	340
HEIGHT	220
スクロールバー 幅	25
選択バー 色	RGB(0,255,255)
最大行数	20
コールバック関数	OnClick

## 背景(ピクチャ)

項 目	設 定 内 容
コントロール名	pctShowArea
X	580
Y	180
WIDTH	400
HEIGHT	300
状態の数	1
状態 0 パターン	背景塗りつぶし
状態 0 背景色	RGB(0,64,128)

## ⑤イメージ(ピクチャ)

項 目	設 定 内 容
コントロール名	pctRECIPEIMAGE
X	55
Y	500
WIDTH	440
HEIGHT	240
状態の数	11

## 背景(矩形)

項 目	設 定 内 容
コントロール名	GSRect00074
X	50
Y	495
WIDTH	450
HEIGHT	250
状態の数	1
塗りつぶし背景色	RGB(0,0,255)

## ⑥メニューボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnParameterSettingMenu
X	820
Y	700
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
文字列	ID_STRING_Menu メニュー
フォント	ID_FONT00000
コールバック関数	OnClick

## メニュー画面(ウィンドウ名:Window\_Menu)



番号	項目	使用コントロール
①	画面切換ボタン	ボタン×8
②	言語切換ボタン	ボタン×2

## ①手動操作ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnManualOperation
X	10
Y	30
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
Foucus 時意匠	img_MenuButton_ON
文字列	ID_STRING_ManualOperation 手動操作
フォント	ID_FONT00000
コールバック関数	OnClick

## モニターボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnMonitor
X	170
Y	30
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
Foucus 時意匠	img_MenuButton_ON
文字列	ID_STRING_Monitor モニター
フォント	ID_FONT00000
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## エラー管理ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnErrorManagement
X	330
Y	30
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
Foucus 時意匠	img_MenuButton_ON
文字列	ID_STRING_ErrorManagement エラー管理
フォント	ID_FONT00000
コールバック関数	OnClick

## 運転データ管理ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnOperDataManagement
X	490
Y	30
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
Foucus 時意匠	img_MenuButton_ON
文字列	ID_STRING_OperDataManagement 運転データ管理
フォント	ID_FONT00000
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## マニュアルボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnManualManagement
X	10
Y	90
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
Foucus 時意匠	img_MenuButton_ON
文字列	ID_STRING_ManualManagement マニュアル
フォント	ID_FONT00000
コールバック関数	OnClick

## 保守管理ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnMaintenance
X	170
Y	90
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
Foucus 時意匠	img_MenuButton_ON
文字列	ID_STRING_Maintenance 保守管理
フォント	ID_FONT00000
コールバック関数	OnClick



## GENWARE3 アプリケーション設計ガイド

## 操作履歴ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnOperationHistory
X	330
Y	90
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
Foucus 時意匠	img_MenuButton_ON
文字列	ID_STRING_OperationHistory 操作履歴
フォント	ID_FONT00000
コールバック関数	OnClick

## パラメータ設定ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnParameterSettings
X	490
Y	90
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
Foucus 時意匠	img_MenuButton_ON
文字列	ID_STRING_ParameterSettings パラメータ設定
フォント	ID_FONT00000
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

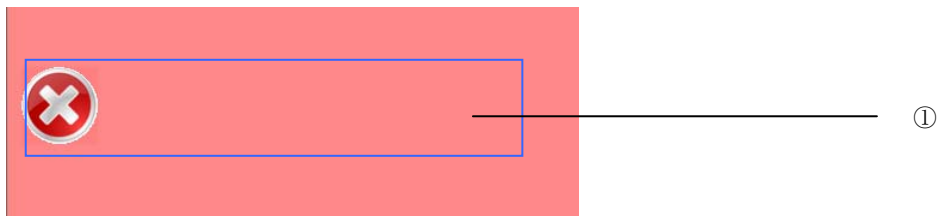
## ②Japanese ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnJapanese
X	10
Y	170
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
Foucus 時意匠	img_MenuButton_ON
文字列	ID_STRING_Japanese
	Japanese
フォント	ID_FONT00000
コールバック関数	OnClick

## English ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnEnglish
X	170
Y	170
WIDTH	150
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時意匠	img_MenuButton_ON
OFF 時意匠	img_MenuButton_OFF
Foucus 時意匠	img_MenuButton_ON
文字列	ID_STRING_English
	English
フォント	ID_FONT00000
コールバック関数	OnClick

## メッセージ画面(パネル名:Window\_Message)



番号	項目	使用コントロール
①	メッセージ表示エリア	ピクチャ、スタティックテキスト

## GENWARE3 アプリケーション設計ガイド

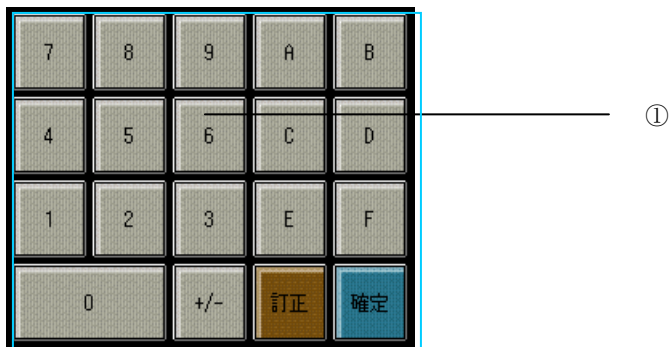
## ①メッセージ表示エリア(スタティックテキスト)

項 目	設 定 内 容
コントロール名	stcErrorMessage
X	80
Y	58
WIDTH	324
HEIGHT	47
背景色	RGB(255,128,128)
文字色	RGB(0,0,0)
横位置	中央
縦位置	中央
フォント	ID_FONT00000
コールバック関数	OnTimer
	OnCreate

## アイコン(ピクチャ)

項 目	設 定 内 容
コントロール名	GPicture00002
X	12
Y	51
WIDTH	64
HEIGHT	64
状態の数	1
状態 0 意匠	ID_IMAGE_ErrorIcon

テンキー画面(ウィンドウ名:Window\_Menu)



番号	項目	使用コントロール
①	テンキー	ボタン×19

## ① 0 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNum0
X	3
Y	159
WIDTH	98
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum0
	0
フォント	ID_FONT00000
コールバック関数	OnClick

## 1 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNum1
X	3
Y	107
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum1
	1
フォント	ID_FONT00000
コールバック関数	OnClick

## 2 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNum2
X	53
Y	107
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum2
	2
フォント	ID_FONT00000
コールバック関数	OnClick

## 3 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNum3
X	103
Y	107
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum3
	3
フォント	ID_FONT00000
コールバック関数	OnClick

## 4 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNum4
X	3
Y	55
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum4 4
フォント	ID_FONT00000
コールバック関数	OnClick

## 5 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNum5
X	53
Y	55
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum5 5
フォント	ID_FONT00000
コールバック関数	OnClick



## 6 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNum6
X	103
Y	55
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum6 6
フォント	ID_FONT00000
コールバック関数	OnClick

## 7 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNum7
X	3
Y	3
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum7 7
フォント	ID_FONT00000
コールバック関数	OnClick

## 8 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNum8
X	53
Y	3
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum8 8
フォント	ID_FONT00000
コールバック関数	OnClick

## 9 ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNum9
X	103
Y	3
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNum9 9
フォント	ID_FONT00000
コールバック関数	OnClick

## A ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNumA
X	153
Y	3
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNumA A
フォント	ID_FONT00000
コールバック関数	OnClick

## B ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNumB
X	203
Y	3
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNumB B
フォント	ID_FONT00000
コールバック関数	OnClick

## C ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNumC
X	153
Y	55
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNumC
	C
フォント	ID_FONT00000
コールバック関数	OnClick

## D ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNumD
X	203
Y	55
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNumD
	D
フォント	ID_FONT00000
コールバック関数	OnClick

## E ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNumE
X	153
Y	107
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNumE E
フォント	ID_FONT00000
コールバック関数	OnClick

## F ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNumF
X	203
Y	107
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_MaintenanceNumF F
フォント	ID_FONT00000
コールバック関数	OnClick

## GENWARE3 アプリケーション設計ガイド

## +/-ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyPositiveOrNegative
X	103
Y	159
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_NumButton_ON
ON 時 意匠	img_NumButton_OFF
文字列	ID_STRING_PositiveOrNegative +/-
フォント	ID_FONT00000
コールバック関数	OnClick

## 訂正ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNumCorrection
X	153
Y	159
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_CorrectionButton_ON
ON 時 意匠	img_CorrectionButton_OFF
文字列	ID_STRING_MaintenanceCorrection 訂正
フォント	ID_FONT00000
コールバック関数	OnClick

## 確定ボタン(ボタン)

項 目	設 定 内 容
コントロール名	btnKeyNumConfig
X	203
Y	159
WIDTH	48
HEIGHT	50
ボタンタイプ	モーメンタリ
表示タイプ	イメージ
ON 時 意匠	img_ConfigButton_ON
ON 時 意匠	img_ConfigButton_OFF
文字列	ID_STRING_MaintenanceConfig 確定
フォント	ID_FONT00000
コールバック関数	OnClick



### 本製品の適用について

- ・ 本製品は人命にかかわるような状況の下で使用される機器あるいはシステムに用いられることを目的として設計・製造されたものではありません。
- ・ 本製品を、乗用移動体用、医療用、航空宇宙用、原子力用、電力用、海底中継用の機器あるいはシステムなど、特殊用途への適用をご検討の際には、当社の営業窓口までご照会ください。
- ・ 本製品は厳重な品質管理の下に製造しておりますが、本製品の故障により重大な事故または損失の発生が予測される設備への適用に際しては、安全装置を設置してください。



## 株式会社 アイ・エル・シー

### 本社

〒100-0005

東京都千代田区丸の内 2 丁目 3 番 2 号 郵船ビルディング 7F

PHONE : 03-3287-7700 FAX : 03-3287-3999

### デザインセンター(HDC:Hiroshima Design Center)

〒732-0824

広島市南区的場町 1 丁目 3 番 6 号 広島場のビル 9F

PHONE : 082-262-7700 FAX : 082-263-4411

### 本厚木オフィス

〒243-0018

神奈川県厚木市中町 4 丁目 9 番 17 号 原田センタービル 6F-A

PHONE : 046-223-7700 FAX : 046-204-5151

### 名古屋オフィス

〒453-0801

名古屋市中村区太閤 3 丁目 1 番 18 号 名古屋 KS ビル 12F

PHONE : 052-452-7700 FAX : 052-453-4400

### 石川オフィス

〒192-0032

東京都八王子市石川町 2956-6 ファインビル 1F

PHONE : 042-643-7710 FAX : 042-645-7011

### 三田オフィス

〒669-1529

兵庫県三田市中央町 4-5 三田ビルディング 4C 号室

PHONE : 079-559-7000 FAX : 079-559-7001

### FACTICS テクニカルセンター

PHONE : 082-262-7799 FAX : 082-263-4411

### インターネットによるアクセス

<http://www.ilc.co.jp> (ILC ホームページ)

[fa@ilc.co.jp](mailto:fa@ilc.co.jp) (FACTICS テクニカルサポート)

- 本書および、本ソフトウェアの内容は、製品改良のため予告なしに変更することがあります。
- 本書の内容の一部、または全部を無断転写することを禁止します。
- コンピュータ本体、CRT、プリンタ、記憶装置などの機器類に関しての故障については、当社は、一切の責任を負いません。異常があった場合は、各々のメーカーにご相談ください。
- 本書および、本ソフトウェアの内容には、万全を期しておりますが、万一ご不審な点がございましたらご連絡ください。
- 本書によって、工業所有権その他の利益の実施に対する保証、または実施権を許諾するものではありません。また本書の記載内容の使用により起因する工業所有権上の諸問題については、当社は一切の責任を負うことができません。
- 記載されている他社製品は各社の商標または登録商標です。