

# JobGen Plus ユーザマニュアル

For Microsoft Windows Version 5.8

本書の情報はお断りなく変更することがあります。本書の一部または全部をユニテック社の許可なしに複製または転送してはいけません。

Microsoft Windows と Windows 95/98/Me/NT/2000/XP は Microsoft Corporationの登録商標です。

オリジナル: © 1996-2001 Unitech America Incorporation. All rights reserved. Document Revision 5.0, May 2001 翻訳: ユニテック・ジャパン株式会社、2005 年 12 月

E	

次

JOBGEN PLUS ユーザマニュアル1
第1章 概要
第2章 インストール
第3章 JOBGEN PLUS を使う
1. ファイルメニュー
5. $E \mu F y = 2^{-1}$
第4章 ジョブのプログラミング
<b>数の範囲</b>
第6章 ジョブの作成

第7章	ジョブのシュミレーション	7
第8章	言語サポート	B

JobGen Plus の基礎

このセクションでは以下を学習します:

- JobGen Plus の概要
- JobGen Plus のインストールと必要なシステム
- JobGen Plus の動作環境

### 第1章 概要

#### JobGen Plus の概要

JobGen Plus は皆様が独自のデータ収集アプリケーションを作ることができるプログラミング ツールです。これは紙にプランを書くようにできるだけ簡単にデータ収集のアプリケーション を開発することができるようにするという考えに沿って設計されています。JobGen Plus はユ ーザフレンドリーなプログラミング環境、完全なプログラミングツールのセット、そしてプロ グラマでない方にでも多彩なデータ収集のソリューションを提供いたします。

JobGen Plus は Windows アプリケーションなので、Windows の良く知られたユーザフレンドリ ーな操作環境の利点を受けることができます。ユーザは単にマウスボタンをクリックするだけ で、JobGen Plus のプログラミングツールのすべてにアクセスすることができます。この強力 なプログラミングツールのセットは、データ収集アプリケーションの設計に含まれるすべての ステップを深く検討することによって開発されました。これは JobGen Plus ユーザの誰もが、 工業、学校、小売店舗、倉庫 – ほとんどどこでも – 何らプログラミングの経験なしに、御自 分のデータ収集アプリケーションを容易に開発できるようにします。

また、JobGen Plus はより複雑なデータ収集アプリケーションを必要とする上級ユーザにプロ グラミング言語オプションを提供します。このオプションはデータ収集アプリケーションで他 の開発ツールによって加えられた制限からプログラマを解放します。しかし、ほとんどのユー ザは、各種のデータ収集アプリケーション、例えば、倉庫管理から売り上げの記録までを、 JobGen Plus だけを使用することによって開発することができます。

#### JobGen Plus のコンポーネント

JobGen Plus によって作られたデータ収集アプリケーションは二つの主要なコンポーネントを 含んでいます:最初の一つは**ノード**と呼ばれ、データ収集アプリケーション中の**プロセス**を表 します。

二つ目は**リンク**と呼ばれ、あるプロセスから他のプロセスへの**移動**を表します。これらの JobGen Plus アプリケーションの主要なエレメントは、以下の章で詳しく説明します。 このセ クションでは、ノードとリンクの概念を紹介することにとどめておきます。

各ノード はデータ収集における一つのタスクを表します。データ収集アプリケーションにお けるタスクの例は、データ収集のメニューの選択; データコレクタに項目番号を入力; あるい はデータコレクタからホストコンピュータへ収集した在庫情報を送信する等を含んでいます。 JobGen Plus はデータ収集アプリケーションにおける各種のタスクを実行するために 9 つのタ イプのノードを持っています。これらは、コメントノード、メニューノード、収集ノード、演 算ノード、編集ノード、消去ノード、アップロードノード、プログラムノード、そしてジョブ 実行ノードです。

コメントノードはフローチャートに直接注記を貼り付けることができます。

**メニューノード**はメッセージ、指示あるいはユーザが選択するアプリケーションの主要 な機能の表示のためのスペースを提供します。 **収集ノード**はデータコレクタでユーザによって入力されたバーコードスキャナまたはキ ーパッドからの情報を保存するために使用されます。

演算ノードはデータの計算に使用されます。

**編集ノード**はデータコレクタに記録されたデータの表示あるいは変更を可能にします。

消去ノードはデータコレクタに記録されたデータを削除します。

アップロードノードはデータコレクタからホストコンピュータへ収集した情報を送信します。

**プログラムノード**はアプリケーション開発者がC言語モジュールを追加することによっ てプロジェクトを強化することを可能にします。

ジョブ実行ノードは他の JobGen Plus ジョブを読み込んで実行するために使用します。

**リンク**は二つあるいはそれ以上のノード(プロセス)間の移動を行います。リンクはすべてのプロセスを一つの JobGen Plus アプリケーションに接続するだけではなく、一つのプロセス(ノード) から他へパスと方向をも提供します。どのようにして、そして何時移動が起こるかについての情報は各リンクの中に記載されます。

# 第2章 インストール

### 必要なシステム

以下は JobGen Plus のインストールに必要な最低条件です。:

- DOS/V 互換コンピュータ、Intel Pentium 互換 CPU 以上
- 128 MB メモリ
- Hard Disk に 10 MB 以上の空き容量があること
- VGA モニター
- 1シリアルポート(COM1、COM2、COM3等々)
- 1マウス(あるいは MS Windows を操作するためのポインティングデバイス)
- Windows Me/2000/XP

#### MS Windows Me/2000/XP へのインストール

- 1. JobGen Plus の CD-ROM を CD-ROM ドライブに入れて下さい。または、ユニテック社の Web サイトから JobGen Plus プログラムダウンロードして適当なフォルダに置いて下さい。
- 2. スタートをクリックし、そしてファイル名を指定して実行を選択して下さい。
- 3. JobGen Plus CD-ROM の入っているドライブまたはフォルダをタイプして、 "Setup"をタイ プして下さい。
- 4. Enter キーを押し、 OK をクリックして下さい。
- 5. JobGen Plus は設定の初期化を実行し、JobGen Plus システムファイルを保存するディレ クトリ名を入力するように求めます。標準のディレクトリ名は JGPLUS5 です。
- 6. セットアッププログラムはファイルを指定したディレクトリヘコピーを始めます。
- 7. セットアッププログラムが終了したときに、JobGen Plus のプロクラムフォルダとアイコ ンを作成します。インストールが終わります。

## 第3章 JobGen Plus を使う

MS Windows Me/2000/XP から JobGen Plus を実行

1. タスクバーのスタートボタンをクリックして、プログラムを選択します。

2. マウスカーソルを JobGen Plus プログラムを含むプログラムグループに移動します。

3. プログラムを実行するために JobGen Plus アイコンをクリックします。

(あるいは、デスクトップにショートカットアイコンを作っておき、これをダブルクリックします。)



### <u>第3章 JobGen Plus を使う</u>

🍣 JobGen Plus - [Job1]				
📆 ファイル(E) 編集(E) ノード(N) ま	表示(⊻) ビルド(B)	ツール① ウインドウѠ) へ	ルプ(田)	_ 7 ×
D 🗳 🔒 📮 🐰 🖻	8 👌 💰		👍 🔒 🕌	ă
			2	
Job     Portable     Records     Comment     Excluded     Nodes, Menu     Nodes, Collect     Nodes, Collect     Nodes, Erase     Nodes, Erase     Nodes, Program     Nodes, Run-Job     Links     Links, Key Conditions     Links, Other Conditions				
				· · · · · · · · · · · · · · · · · · ·
ヘルプは、F1 を押して下さい。				NUM

ここで、JobGen Plus はデータ収集のアプリケーション開発の準備ができました。

JobGen Plus の基礎

### JobGen Plus の作業環境



# 1. ファイルメニュー

3	JobGen Plus - [Job1]		
12	ファイル( <u>F)</u> 編集(E) ノード(N) 表示	♡ ビルド® ツール① ウインドウ铋 ヘルプЮ	- 8 ×
] [	♪ 新規(N) Ctrl+N	🗅 ≼ メ 🗖 🗖 🖉 🖧 💂	i i
[	☑ 開(@) Ctrl+O	🏬 🗞 💷 🖾 🗊 📶 🍢	
- 	閉じる( <u>C</u> )		🔼
	□ 保存⑤ Ctrl+S		
	名前を付けて保存(A)		· · · · · · =
	🗐 全部保存(1)		· · · · · ·
	🚑 フローチャート印刷(P)		· · · · · · <b>–</b>
	□ 印刷プレビュー(⊻)		· · · · · ·
	印刷設定(4)		
	<u>1</u> Job1 2 invent		· · · · · ·
	<u>3</u> tutorial		
	4 rm_asset		
	int: 1 \ <u>X</u> /		

新規	新しいジョブを作成する。
開く	すでにあるジョブファイルを開く。
閉じる	現在のジョブウインドウを閉じる。
保存	現在のジョブの内容を保存する。
名前を付けて保存	現在のジョブを別なファイル名で保存する。
全部保存	すべての開いたジョブの内容を保存する。
-	
フローチャート印刷	現在のジョブのフローチャートを印刷する。
印刷プレビュー	印刷出力をスクリーン上で見る。
印刷設定	印刷オプションの設定。
-	
最近使ったファイル	最近使ったファイルをリストする。
-	

### 2. 編集メニュー

-

-

🄏 JobGen P	lus -	[Job1]																			×
🕎 ファイル(E)	編集	( <u>E)</u> ノード( <u>N</u> )	表示₩	ビルト	«( <u>Β</u> )	ツ-	'nŒ	) '	ケイン	ドウ(	W	∧ル:	¢( <u>H</u> )	1					-	٦	×
]	×.	定義( <u>D</u> )		Enter	ĸ		<u>د ا</u>	T	1		X	4	5	9	L	₽			Ë	j	
🛃 🚮	4	ジョブ定義(標準	售値)		ab	X		7	fo	-		₹.									
<b>⊕</b> Job ⊕ Portable		ジョブ定義(」)						 				 	 			· ·		 			^
Records Comment		ポータブル定義	( <u>P</u> )			· ·	:	· ·	· ·			 	· ·	· ·	•	· ·	:	· ·	· ·	•	
Excluded Nodes, Meni		レコード定義(B	)			· ·	:	 	· ·	· ·		 	 	· · · ·		· ·		 	· ·		≡
Nodes, Colle Nodes, Math		ジョブ保存時に	ノードID更	新	- · · ·	•••		· ·	· ·			· ·	· ·	· · · ·	:	· ·	:	· ·	· · · ·		
Nodes, Edit Nodes, Eras Nodes, Unio	Ô	新規レコード(N	)			· · ·		· · · ·	· · · ·	· · ·		· ·	· · · ·	· · · ·		· · · ·		· · · ·	· · · ·		
- Nodes, Prog Nodes, Run-	*	削除		Del			•	· ·	· · ·			 	 	· · ·	•	· · · ·	•	· ·	· · ·	•	
Links Links, Key C		メッセージウイン	ドウクリア(			 		 	 	 		 	 	 		 		 	 	•	
Links, Other	R	検索(F)		F3	-	· ·	•	· ·	· · ·			· ·	· ·	· · ·	•	· · · ·	•	· ·	· · ·	•	
l					 	· ·	:	 	· ·			 	 	· ·	:	· ·	:	 	· ·	•	~
			<																	>	
																		NUM	Ī		1

定義	ノード、リンク、あるいはレコードのいずれかの現在選択されている
	オブジェクトを定義する。

	ジョブ定義(標準値)	ジョブについて、	標準値の設定を定義する	5.
--	------------	----------	-------------	----

**ジョブ定義** 現在のジョブを定義する。

- **ポータブル定義** ポータブルターミナルの設定を定義する。
- **レコード定義** レコード設定を定義する。
- 新規レコード 新しいレコードを作成する。JobGen Plus は複数レコードをサポート している。
- **削除** ノード、リンク、あるいはレコードのいずれかの現在選択されている オブジェクトを削除する。

9

**メッセージウインドウ** メッセージウインドウに表示されているすべてのメッセージをクリア **クリア** します。

検索 現在のジョブ中のテキストを検索する。結果はメッセージスクリーン 検索に表示されます。 "枠 2 へ出力"がチェックされた場合、すべて の結果はメッセージスクリーン検索 2 に表示されます。メッセージウ インドウで見つかった結果をダブルクリックすると関連の定義ダイア ローグ・ウインドウが開きます。

検索	
Total Total	nodes: 0 links: 0
検索:	
	□ 大文字/小文字→致 □ 枠2へ出力
	OK キャンセル

# 3. ノードメニュー

🄏 JobGen Plus - [Jo	lob1]	
🕎 ファイル(E) 編集(E)	ノード(N) 表示(V) ビルド(B) ツール(T) ウインドウ(M) ヘルプ(H)	- 8 ×
D 🖻 🔒 🗐	- 19 😽 🗖 🔁 🖬 🐄 😽	Ť
🛛 🖬 🛛 🛃	א 🗉 אבע-א-א(ש) 🔯 🗊 🖬 🌄	
I ⊕ Job	□ ₩₩ Ψ集ノード© □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	
Portable     Records     Comment	通算ノード(a)         ご こ こ こ こ こ こ こ こ こ こ こ こ こ こ こ こ こ こ こ	
- Excluded Nodes Menu	・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	· · · · · · =
- Nodes, Collect [ - Nodes, Math - Nodes, Edit [ - Nodes, Erase - Nodes, Upload [ - Nodes, Program	◎ 消去ノード(2) パー・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	
	■ アップロードノード(型)	💻
	<b>麺</b> プログラムノード(P) (1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.	· · · · · ·
Nodes, Run-Job Links	□ ジョブ実行ノード(R)     □	· · · · · ·
- Links, Key Conditions Links, Other Condition	🐾 υ>σω	· · · · · ·
l		
		• • • • • 🗸
1		

コメント(ノード)	コメントノードを作成する。
メニューノード	メニューノードを作成する。
収集ノード	収集ノードを作成する。
演算ノード	演算ノードを作成する。
編集ノード	編集ノードを作成する。
消去ノード	消去ノードを作成する。
アップロードノード	アップロードノードを作成する。
プログラムノード	プログラムノードを作成する。
ジョブ実行ノード	ジョブ実行ノードを作成する。
-	

リンク リンクを作成する。

#### 4. 表示メニュー



**ツールバー** このコマンドがチェックされた場合、ツールバーを表示する。

**ステータスバー** このコマンドがチェックされた場合、ステータスバーを表示する。

出力 メッセージウインドウを開く。

**プロパティパネル** プロパティパネルを開く。

**詳細なプロパティ** プロパティパネルで詳細な情報の表示を可能にする。

フローチャート再描画 全体のフローチャートを再描画する。

\_

JobGen Plus の基礎

# 5. ビルドメニュー

🌃 JobGen Plus – [C:¥Docum	ents ar	id Set	ting	s¥YAMA	Shigeru	¥My Do	cumen	ts¥Ur	nitec	h		×
🕎 ファイル(E) 編集(E) ノード(N)	表示♡	ビル	۲( <u>B)</u>	ツール①	ウインド	ל₩) ^	ルプ(日)				- 8	×
D 🖻 🔒 🗐 🕺 🖻	B	í 👕	೮∋ರ	"作成( <u>M</u> )		F4	4	8	₽		Ť	
3 🖏 🛃 🔤		<del>ا</del>	יבע	ードファイルフ	<sup>7</sup> ップロード	Ē	ь					
Job     Portable	<b>∧</b> [ ] ]											^
Model: PT-630										· · ·	· · ·	
		• •							· ·			
– Records ASSETMGT (default)							· · ·	· · ·	· ·	· · ·	· · ·	
Comment												_
Excluded			E JA	<u></u>		÷			• •	· · ·		
i ⊶ Nodes, Menu I ➡- Nodes, Collect			Ealt			Secti	on					
Nodes, Math		• •	• •			···↑			• •			
i Modes, Edit												
- Nodes, Erase - Nodes, Upload		• • •							· ·			
Nodes, Program										· · ·	· · ·	
Nodes, Run-Job						• • • •	L	<b>.</b>				
ten Links Ten Links, Key Conditions								· • •				$\mathbf{M}$
										KILIKA	2	J
										NUM		11.

**ジョブ作成** ジョブの実行コードを生成し、そしてコードとすべてのデータファイル を含むファイルをポータブルターミナルにダウンロードする。

レコードファイル レコードファイルをアップロードする。 アップロード

### 6. ツールメニュー



PTComm Manager

PTComm マネージャ(通信管理)アプリケーションを実行する。

ジョブシュミレーション シュミレーション終了 ジョブシュミレーションを開始する。 ジョブシュミレーションを終了する。 JobGen Plus の基礎

-

# 7. ウインドウメニュー

🌾 JobGen Plus - [Job1]																		$\mathbf{X}$
📆 ファイル(E) 編集(E) ノード(N) ま	表示☑	ビルト	∹( <u>B</u> )	ツーノ	ΓŪ	ウイ	ンドウ	Ŵ	Λŀ.	⊅( <u>H</u> )							- 6	×
D 🛎 🔒 🎒 🐰 🖻	6 (	<mark>ר ר</mark>	s.	*		C	重ね	て表	π©			א	Ŧ	۶		ĺ	<b>*</b> j	
🛛 🖏 🛛 🛃 🖉 🗉		٩	QD	X	E		並べ	て表	πD									
<ul> <li>Job</li> <li>Portable</li> <li>Records</li> <li>Comment</li> <li>Excluded</li> <li>Nodes, Menu</li> <li>Nodes, Collect</li> <li>Nodes, Math</li> <li>Nodes, Edit</li> <li>Nodes, Edit</li> <li>Nodes, Erase</li> <li>Nodes, Upload</li> <li>Nodes, Run-Job</li> <li>Links</li> <li>Links, Key Conditions</li> </ul>				· · · · · · · · · · · · · · · · · · ·	<ul> <li>.</li> <li>.</li></ul>		1 Jo	<b>bb1</b>	· · · · · · · · · · · · · · · · · · ·	· · · · · · · · · · · · · · · · · · ·	· · · · ·		•••••••••••••••••••••••••••••••••••••••		· · · · · · · · · · · · · · · · · · ·		· · · · · · · · · · · · · · · · · · ·	
Links, Other Conditions		· · ·		· · · ·	· · · ·	· ·		· ·	· ·		:		:				: :	~
	<												<b>_</b>		NU	M	>	

**重ねて表示** すべてのウインドウを重ねて表示する。

**並べて表示** すべてのウインドウを並べて表示する。

**開かれたウインドウ** すべての開かれたウインドウのタイトルをリストする。

# 8. ヘルプメニュー

🍣 JobGen Plus - [Job1]													[			×
🕎 ファイル(E) 編集(E) ノード(N) 🛃	転∞	ビル	۲:( <u>B</u> )	ツール	Ð	ウイン	·ドウ(W)	A/V	⊅( <u>H</u> )					-	. 8	×
D 📽 日 🗐   X 🖻	ß	<b>^</b>	«K	*		3		x 😵	クイッ	ウスター	-Ւ©	3)				
	<b>••••</b>	۵.	۵b	X	Ē	fo			へルフ	የトピッ	りス(	D				
⊞- Job ⊕- Portable				• •					ヘルナ	の内容	容( <u>H</u>	)			F1	
Records Comment Excluded			· · ·	· · ·	· ·	· · ·			Micro	osoft (	っ言	語^	いげ	)( <u>C</u> )		
- Nodes, Menu Nodes, Collect		•••		• •	· ·			<u>~</u>	Abou	t Job	Gen	Plu	s 5			
Nodes, Math Nodes, Edit		· · ·	· · ·	· · ·	· ·	· · ·		· · · ·	· · · · · ·	· · ·		· ·	· · ·	•	· ·	
Nodes, Erase Nodes, Upload				• •	· ·			• •		• •				•	• •	
Nodes, Program Nodes, Run-Job			 	· ·	 			· · · ·	 	· ·		· ·	· ·	•	· ·	
– Links – Links, Key Conditions			 	· · · ·	 	· · ·	· · ·	· · · ·	 	· · · ·	• •	· ·	· ·		· ·	
Links, Other Conditions			 	· ·	 	· · ·	· · ·	· · · ·	 	· · · ·	· ·		· ·	:	· ·	~
	<														>	
													NU	M		1

- **クイックスタート** JobGen Plus の概要を表示する。
- **ヘルプトピックス** ヘルプのトピックスを表示する。
- **ヘルプの内容** JobGen Plus のヘルプを表示する。

Microsoft C 言語 Microsoft C Language Run-Time Reference のヘルプを表示する。 ヘルプ

About JobGen Plus 5 JobGen Plus のバージョン番号と版権を表示する。

-

# ジョブの作成

このセクションでは以下を学習します:

- ジョブの設計の仕方
- ジョブの作成方法
- 実行可能なジョブの作成方法

#### 第4章 ジョブの設計

# 第4章 ジョブの設計

ジョブを作成する前に、ユーザは作成に必要な各種のタスク(仕事)のすべてについてまず分析 をしなければなりません。タスクの例としては、データの収集、選択を行う、データを送信す る、データのチェック、メッセージの表示などです。次に、ユーザはこれらのタスクをどのよ うに実行するかのプランを立てなければなりません。あるリストされたタスクは、タスクから タスクへの移動を伴う順序を持った手順で構成されます。一方では、あるタスクはジョブ全体 のサポート機能として独立して働くこともあるでしょう。最後ステップは JobGen Plus ですべ てのタスクと、タスク間の移動を実装することです。

タスクの実装とタスク間の移動は簡単です。ユーザは JobGen Plus のノードに合った各タスク を単に実装するだけです。例えば、データ収集のタスクは収集ノードによって実装することが できます。 以下の表は JobGen Plus で相当するノードによって実装することのできるタスク のタイプを示しています。

### <u>ノード</u>実装することのできるタスク

- **コメント** 注記と説明を表示
- **メニュー** ユーザに対する選択を表示
  - エラーメッセージを表示
  - 命令・指示を表示

### キーボードからデータを入力

- バーコードスキャナからデータを入力
- データコレクタのタイマーからデータを入力
- ルックアップファイルからデータを入力
- 式からデータを入力
- データ入力の確認
- 式からデータを入力
  - データの計算
  - データの比較

• 収集したデータ内部のレコードを削除

**アップロード** • データコレクタからホストコンピュータへデータを送信

収集

演算

消去

# ジョブの作成

編集	•	ポータブルターミナルのデータファイルに保存されたデ ータを表示
	•	データを検索
	•	データの変更
プログラム	•	上記にリストされたノードによって実装することのでき ないタスクの種類
ジョブ実行	•	他のジョブを実行

あるタスクから別のタスクへの移動は JobGen Plus ではリンクによって行うことができます。

### 第5章 ジョブのプログラミング

# 第5章 ジョブのプログラミング

ジョブ<sup>1</sup>の構築はマウスボタンをクリックするように簡単です。JobGen Plus は MS Windows の 利点を利用しており、したがってほとんどのユーザは JobGen Plus によって提供されているツ ールの使用を視覚的に行うことができます。この章はジョブの作成に必要なすべてのプロセス を詳しく説明しています。

### ジョブ設定の標準値を定義する

ジョブ設定の標準値定義のダイアローグ・ウインドウを開くために、メニューの編集 > 書部 定義(標準値)を選択して下さい。これらの設定はすべての作業中のジョブに影響します。

### フォントプロパティ

ジョブ定義 (global settings)		
Font Color Language Make Job Simulate Job Portable	フォント: タイブ: インデックス: MS ゴシック ノード名: MS ゴシック ブログラム: MS ゴシック ポータブルスクリーン: MS ゴシック コメント: MS ゴシック	サイズ: 9 >> 9 >> 9 >> 9 >> 9 >> 9 >> 9 >>
	OK キャンセル ヘルプ	

左のプロパティウインドウ、ノード名、プログラムエディタ、そしてコメントを表示するフォ ントを定義します。

<sup>1</sup> ジョブはデータ収集アプリケーションです。

# カラープロパティ

ジョブ定義 (global settings)	×				
Font Language Make Job Simulate Job Portable	<ul> <li>ノードとリンクの色:</li> <li>メニュー・</li> <li>収集・</li> <li>演算・編集・</li> <li>消去・</li> <li>アップロード・</li> <li>プログラム・</li> <li>ジョブ実行・</li> <li>コメント・</li> <li>開始ノード・</li> <li>遅択ノード・</li> <li>ノードテキスト・</li> <li>リンク・</li> <li>遅択リンク・</li> <li>隆外・</li> <li>スクリーン・</li> <li>* 色の上で右クリックすると名前の前景色を変更</li> </ul>				
OK キャンセル ヘルプ					

ノード、リンク、そして選択についてのカラーを定義します。表示されているのはノードについてのカラー設定の標準値です。それぞれのノードの色はその基本プロパティ設定で変更する ことができます。

Language(言語)プロパティ

ジョブ定義 (global settings)	
Font Color Make Job Simulate Job Portable	<ul> <li>言語:</li> <li>○ 英語</li> <li>○ 中国語</li> <li>○ スペイン語</li> <li>○ フランス語</li> <li>○ ドイツ語</li> <li>○ イタリア語</li> <li>○ 日本語</li> </ul>
	OK キャンセル ヘルプ

ジョブの言語を選択します。

日本語フォントを選択します。

Select Japanese Font	
フォント(E): Draws Serif MS Sans Serif MS Serif MS UI Gothic TMS UI Sothic TMS Signore MS Sans Serif MS Serif MS Sans Serif	): サイズ( <u>S</u> ): <u>OK(Q)</u> 16 11 12 14 16 マ
サンブル 文字あい, AaBbYyZz	16×16

### ジョブ作成プロパティ

ジョブ定義 (global settings)	
Font Color Language Make Job Simulate Job Portable	<ul> <li>□ ルックアップファイル用のバイナリサーチテーブル作成スキップ</li> <li>□ 中国語/日本語フォント作成をスキップ</li> <li>□ ジョブ名ファイル作成をスキップ</li> <li>□ JobGen Plus バージョン 4 で作られたジョブを作成</li> <li>□ ルックアップデータファイルをダウンロードしない</li> <li>□ 全体のジョブを自動的にダウンロードしない</li> <li>□ ダウンロード/アップロードに高速転送を使用</li> </ul>
	OK キャンセル ヘルプ

**ルックアップファ** ルックアップファイルの内容が変わらない場合、ジョブを作成 イル用のバイナリ するたびにコンパイルする必要はありません。これはジョブ作 サーチテーブル作 成のプロセスをスピードアップします。 成スキップ

中国語/日本語フ ルックアップファイルとデータファイルに含まれるすべてのテ
 オント作成をスキ キストが変わらない場合、そのたびごとに漢字フォントを作る
 ップ 必要はありません。

ジョブ名ファイル ジョブファィル名がどのジョブを実行するかを決めるためにジ ョブエンジンに対して使用されます。ジョブの実行可能な名前 を選択することによってジョブを実行する場合、ジョブ名ファ イルは不要になります。

JobGen Plus バー古い JobGen Plus のバージョンでは動いていた古いジョブで問ジョン 4 で作ら題が起こった場合、このオプションをチェックして、ジョブをれたジョブを作成再度作成して下さい。これは以前のバージョンとの互換性を持たせることができます。

**ルックアップデー** すべてのルックアップファイルの内容が変更されない場合、そ **タファイルをダウ** のたびごとにダウンロードする必要はありません。

ンロードしない

### 第5章 ジョブのプログラミング

全体のジョブを自 JobGen Plus は、ジョブ実行コードを正しく生成できたときに ホータブルターミナルに実行可能ジョブをダウンロードするこ とができます。ダウンロードを停止するためにはこのオプショ ンをチェックして下さい。JobGen Plus は後でダウンロードす るために Job エンジン (JENG. exe) をジョブ・フォルダにコ ピーします。

**ダウンロード/ア** JobGen Plus は PTComm マネージャが最も速い速度で通信する ップロードに高速 ようにします。

転送を使用

ジョブの作成

### ジョブ設定の標準値を定義

ジョブ定義設定ウインドウを開くために、ジョブの左側のプロパティウインドウで Job > Define を選択します。



### Barcode (default) プロパティ

ジョブ定義		
Basic - Comment - Flowchart - <mark>Barcode (default)</mark> ⊕ Job Prompts	<ul> <li>▼ この収集ノードについてパーコードシンボルをセット * 各個々の設定を変更するためにダブルクリック</li> <li>✓ Codabar *, Send Start/Stop Characters</li> <li>✓ Code 128</li> </ul>	
	<ul> <li>✓ Code 32 *, Send Leading Character 'A'</li> <li>✓ Code 39 *, Send Check Digit</li> <li>Code 93</li> <li>✓ EAN 128</li> </ul>	
	EAN 13 *, Send Leading Digit, Send Check Digit EAN 8 *, Send Check Digit	
	全収集ノードにセット     全部クリア	_
	OK キャンセル ヘルプ	

### 第5章 ジョブのプログラミング

すべての収集ノードについて標準のバーコードシンボル設定を定義します。それぞれの収集ノードについての設定は、Input > Barcode プロパティで変更することができます。

Define Node [Collect] - Sec	stion	×
Define Node [Collect] - Sec Screen Position Input Initial Value Device Cookup File Formula Verify Lookup File Qualifier Function Keys Links, in C- Asset_number, #2 C- Edit, #4 Links, out C- Sec Sec Sec Sec Sec Sec Sec Sec	etion 「この収集ノードについてバーコードシンボルをセット * 各個々の設定を変更するためにダブルクリック 「Codabar *, Send Start/Stop Characters Code 128 Code 32 *, Send Leading Character 'A' Code 39 *, Send Check Digit Code 93 EAN 128 EAN 13 *, Send Leading Digit, Send Check Digit EAN 8 *, Send Check Digit Caracter *	
	OK キャンセル ヘルプ	

# Flowchart(フローチャート)プロパティ

ジョブ定義	
Basic Comment Flowchart Barcode (default) ⊕ Job Prompts	ページサイズ: 幅: 1100 手 高さ: 850 手 バードサイズ: 幅: 60 手 高さ: 30 手
	OK キャンセル ヘルプ

フローチャートのページサイズとすべてのノードサイズを変更します。

ジョブの作成

### Language (言語) プロパティ

このジョブの言語設定を変更します。

ジョブ定義 Language		K
	言語: ○ 英語 ○ 中国語 ○ スペイン語 ○ フランス語 ○ ドイツ語 ○ イタリア語 ○ 日本語 Define Japanese Font Japanese Font Share Japanese Font With Font File:	
	OK キャンセル ヘルプ	

## ポータブルターミナルの定義

デバイスの選択は、メニューノード、収集ノード、演算ノード そしてリンクの表示に影響が あるので JobGen Plus では最も重要です。メニューノード、収集ノードと演算ノードの "Screen"(スクリーン)部分のサイズは選択したデバイスの表示と完全に同じサイズにセットす る必要があります。また、リンクのキー入力は、選択したデバイスと同じキーパッドレイアウ トを表示します。JobGen Plus は選択したジョブを適当なポータブルターミナルにダウンロー ドします。

ポータブルターミナルのモデルを選択するには、ジョブのプロパティパネルの、プロパティの 項: Portable > Model を開きます。ウインドウをポップアップするために Model の項目をダ ブルクリックします。 ポータブルターミナルのタイプを定義するために Model プロパティを 再度選択します。

### ポータブル定義プロパティ

### モデルの定義:

ポータブル 定義	
Setting Keymap	<ul> <li>ポータブルモデルの変更は、新しいモデルでキーマップ等が 失われ、いくつかの設定が動かなくなる可能性があります。</li> <li>ポータブルモデル:</li> <li>HT-636</li> <li>PT-630</li> <li>PT-600</li> </ul>
	OK キャンセル ヘルプ

ポータブル
 例えば、PT630のようにターゲットとするポータブルターミ
 モデル
 ナルのモデルを選択します。PT-630 はポータブルターミナ
 ルの標準のモデルです。

ポータブルモデル設定の変更はジョブのあるプロパティ設定に影響します。例えば、PT-630 は PT600 よりも大きなスクリーンなので、PT-630 から PT600 ヘポータブルモデルの変更を行う と、スクリーンのレイアウトはマニュアルで変更する必要があるか、あるいは小さなスクリー ンに合うように再設計する必要があります。

# ポータブルターミナル設定(Setting)

ボータブル定義		×
Model Setting Keymap	-スクリーン: ○ 小型フォント ○ 大型フォント - スキャナタイブ: ○ 自動 ○ ペン - ブザーボリューム: ○ 高 ○ 中 ○ 低 ○ オフ	
	OK キャンセル ヘルプ	

- **スクリーン** ディスプレィのフォントサイズを定義。日本語を表示させる 場合は大型フォントを選択して下さい。
- **スキャナタイプ** スキャナタイプの定義。標準は自動です。
- ブザーボリューム ブザーの音量を定義。

# キーパッド設定の定義(Keymap):

ボータブル定義	
Model Setting Keymap	+−パッド >> Γ マイキーマップを使用 キーマップリセット
	OK キャンセル ヘルプ

**キーマップリセット** ポータブルターミナルの標準設定を使用。

キーパッド キーマップの設定を定義するためにウインドウをポップアップ。

マイキーマップを使用 キーパッド設定を適用することをジョブエンジンに伝える。

ジョブの作成

### 標準値からキーマップを変更する方法:



キーマップを変更したいキーボタンをクリックします。キーの名前が右側のキーボックスに表示され、そして ASCII コードが印刷可能な文字でキー名の下の、二つのキーコードボックスに表示されます。

キーマップを変更するために、キーの ASCII コードの値を変更し、そして "セット" をクリッ クします。ジョブの実行時にカスタマイズしたキーマップ設定を適用するためには"マイキー マップを使用"をチェックしなければなりません。

### 第5章 ジョブのプログラミング

### メニューノード)の作成

メニューノード)は、メッセージの表示方法を提供します。メニューノード を作成した後で、 ユーザはターゲットデバイスのディスプレィに表示するメッセージをタイプインすることがで きます

メニューノードの作成方法: ノードツールバーのメニューアイコンをクリックすることによって、メニューノードとなるノードタイプを選択するか、あるいはノードメニューでメニューノードを選択します。そして、フローチャートに新しいノードを置きたい場所を指すためにカーソルを使用して、ダブルクリックして下さい。新しいメニューノードが現れます。

あるいは、より簡単な方法では:フローチャートに新しいノードを置きたい場所を指すために カーソルを使用して、コンテキストメニューを表示するために右クリックし、そしてメニュー ノードを選択します。新しいメニューノードが作成されます。

他のノードタイプの作成には、操作は同じです – 別なノードタイプを選択するだけです。

そのプロパティを定義するためにメニューノードをダブルクリックして下さい。

### メニューノードの定義

各ノードは独自のプロパティグループを持っており – そして、basic、 comment、 function keys、links in、 そして links out プロパティなどの共通なプロパティがあります。

### Basic プロパティ

Define Node [Menu] - menu	il second se	×
Basic Comment	This is a menu node.	
Function Keys Links, in Links, out	名前: menu1	
	□ 開始ノード: ジョブはこのノードから実行を開始します	
	▶ * ジョブ作成時にこのノードを外す	
	OK キャンセル ヘルプ	
名前	ノードの名前。アルファベット、数字と下線文 字のみを名前に使用できます。	
----	--	
	収集または演算ノードの場合、これらは実際に データを持っているので、ノード名にプリフィ ックス"_"(下線)を追加し、データにアクセス するために C プログラムによって参照できるよ うにします。	

開始ノード このボックスをチェックすることによって、このノードは開始ノードになります (JobGen Plus アプリケーションの最初のポイント)。

各 JobGen Plus アプリケーションは一つだけの 開始ノードを持つことができます。

**色** ノードの色を定義します。

* ジョブ作成時	このノードはまだ作っている最中であることを
にこのノードを	JobGen Plus に知らせます。その設定はまだ終
外す	わっていないので、JobGen Plus はこのノード
	を実行コードに生成しません。

Comment (コメント)プロパティ

Define Node [Menu] - menu	d.	×
Basic Comment Screen Function Keys Links, in Links, out	コメント:: () () () () () () () () () ()	
	OK キャンセル ヘルプ	

**コメント** コメントを書きます。 ボックス

# Screen(スクリーン)プロパティ

Define Node [Menu] - menu1	
Basic Comment Screen Function Keys Links, in Links, out Comment Screen: rows: 16, columns: 4 文字 務 メニュー 1. 入庫 2. 出庫 3. 返品 4. 棚卸 5. 発注 6. 設定 Coxニューノードを終了するときにスクリーンをクリ フーンルを隠す	ノアしない ノーンをクリアしない
OK キャンセル ヘルブ	Ĵ

画面表示 実際の画面で見える通りに表示するウインドウ です。これはポータブルターミナルの表示と同 じように表示します。 表示したいテキストをウインドウに直接タイプ します。 ディスプレィの大きさは、ポータブル定義で選 択したポータブルターミナルによって決まりま す。 メッセージを表 前のスクリーンからのメッセージを保持しま 示する前にスク す。 リーンをクリア しない このメニューノ 次のスクリーンにメッセージを保持します。 ードを終了する ときにスクリー ンをクリアしな 11

## Function Keys(ファンクションキー)プロパティ

Define Node [Menu] - menu	il second se	×
Basic Comment Screen Function Keys Links, in Links, out	もし実行しているノードがトリガ状態の時に同じファンクショ ンキーとリンクしていない場合、、以下に定義される選択され たファンクションキーを押すとこのノードを実行します。 - ファンクションキー・ ○ F1 ○ F2 ○ F3 ○ F4 ○ F5 ○ F6 ○ F7 ○ F8 - 矢印キー: ○ 左 ○ 右 ○ 上 ○ 下 その他キー: ○ 終了 (Esc) ○ なし	
OK キャンセル ヘルプ		

ジョブを実行中に、このノードで定義したファンクションキーを押すことによりこのノードの 実行フローを指示します。しかし、現在のノードがこのファンクションキーのキー入力状態に リンクされている場合、このリンクはその相手先ノードを実行するために高い優先度を持ちま す。

特定ノードへファンクションキーを定義することは大変簡単なフローチャートの設計です。 方、一つのノードへ同じファンクションキーの入力状態を指示している多数のリンクがあり、 これらのリンクは相手先ノードのファンクションキー定義によって置き換えることができます。

# Links In(リンクイン)プロパティ

Define Node [Menu] - men	u1	×
Basic Comment Screen Function Keys <mark>Links, in</mark> Links, out	Incoming links	
	リンクを作成: リンクツールを選択, リンク元のノードにマ ウスを置き、クリックしてリンク先ノードにドラッグアンド ドロップ。	
	自己ルーブリンクの作成:同じノードの片側から別なところ ヘマウスをドラッグアンドドロップします。	
	OK キャンセル ヘルブ	

# Links Out(リンクアウト)プロパティ

Define Node [Menu] - menu1 🛛 🛛 🔀		
Basic Comment Screen Function Keys Links, in Links, out	Outgoing links	
	自己ループリンクの作成:同じノードの片側から別なと ころへマウスをドラッグアンドドロップします。	
OK キャンセル ヘルブ		

Links In/Out(リンクイン/リンクアウト)プロパティは、このノードに入る、または出るすべ てのリンクをリストします。Define Link(リンク定義)ダイアローグを開くためにリンク上を クリックして下さい。

#### 収集ノードの作成

収集ノードはユーザが情報の入力するための最初の場所です。ユーザからの入力を必要とする JobGen Plus アプリケーションの任意のポイントは、収集ノードによって表されます。

収集ノードの定義 Basic プロパティ Comment プロパティ Screen プロパティ Function Keys プロパティ Links In プロパティ Links Out プロパティ

これらのプロパティはメニューノードと同じです。

# Screen > Position プロパティ

Define Node [Collect] - col	lect2	×
Basic Comment Screen Input Input Device Barcode Lookup File Formula Verify Lookup File Qualifier Function Keys Links, in Links, out	<ul> <li>ブロンブト表示:</li> <li>● 最後の印字に続いて印字、必要ならスクロールアップ</li> <li>● 画面に直接印字</li> <li>■ 指定した場所から画面の最後までスクリーンをクリア Row</li> <li>● Column</li> <li>● 位置マークを表示しない 位置マーク:</li> <li>● 初まの位置</li> <li>● 指定した位置: Row (03):</li> <li>● Column</li> <li>● 「スワードとして(各入力文字を*で印字)</li> <li>■ 自動的に生成されたデータについてはプロンプトと入力を表示しない</li> </ul>	
OK キャンセル ヘルプ		

# プロンプト表示

最後の印字に続い 現在のカーソル位置で行ごとに表示し、そしてスクリ て表示、必要なら ーンをスクロールアップします。 スクロールアップ

画面に直接印字 スクリーンに表示されたテキストをポータブルターミ ナルのスクリーンに表示します。すべてのテキストは その指定された位置になければなりません。

**指定した場所から** 表示する前にスクリーンの一部または全部をクリアし **画面の最後までス** ます。

**クリーンをクリア** row と column ボックスで行と列を指定します。

位置マークを表示 新しい入力のスタート時に、前の入力がブランクにな しない ります。現在の入力は以降の入力が始まるまでスクリ ーン上に残ります。

位置マーク データ入力のためにブランクのスペースを埋めるため に使用される英文字。ユーザがデータを入力したら、 これらの文字は入力データに置き換わります。

# 入力表示

現在の位置	現在のカーソル位置に表示します。	
指定した位置	Row(行)と column(列)で指定された位置に表示しま す。	
	row と column ボックスで行と列を指定します。	
パスワードとして	* を表示することによって入力を隠します。	

自動的に生成され 入力はユーザとの会話なしに入れられます。

たデータについて はプロンプトと入 力を表示しない

# Input(入力)プロパティ



**データタイプ** すべてのデータは、数値または文字列の二つの基本 データタイプです。数値データは整数、 倍精度整数 と浮動小数点の三つのタイプに分けられます。

> 数値データは数値演算を実行するために使用されま す。 三つの数値タイプは整数の範囲と小数点以下の 数があるかどうかが異なっています。

倍精度整数を表すために、数の最後に'L' を追加 します。例えば: 100000L。(100000 は整数の制限 値を越えています。)

浮動小数点を表すために、小数点の右に常に入力が あります。例えば: 10.0。

文字列はテキストです。

データを前のレ
 データは最後のレコードからのデータで作られ(コピ
 コードと同じも
 ーされ)、そして入力はすぐに自動的に完了します。
 のにセット、そ
 言い換えると、新しい入力は前のレコードと完全に
 して入力を終了
 同じです。

**ホワイトスペー** 空白文字(ASCII コード、0x20)でデータの後部を自 **スで後ろを埋め** 動的に埋めます。これは固定長データを生成するた **る** めのものです。

空のデータを許 データ入力のスキップを可能にします。レコード中 のこの特別なフィールドの値はヌルで、何もデータ はありません。標準では、すべてのデータは1また はそれ以上の文字を含まなければなりません。

# 数の範囲

整数:	-32767	から 32767
倍精度整数:	-2147483647	から 2147483647
浮動小数点:	1.175494351e-38F	から 3.402823466e+38F

# Input > Initial Value プロパティ

Define Node [Collect] - col	lect2
Basic Comment Screen Position Input Device Barcode Lookup File Formula Verify Lookup File Qualifier Function Keys Links, in Links, out	<ul> <li>■ 最後のレコードからコピー</li> <li>● 声動によって:</li> <li>● データノードからのコピーによって:</li> <li>● 式によって:</li> <li>● 式によって:</li> <li>● TILITフラグのチェックされたリンク入力によりアクティブにされる</li> </ul>
	OK キャンセル ヘルプ

最後のレコードか データの値は最後のレコードからのデータで(コピーで)
 らコピー 埋められます。データはこの後変更することができます。このオプションは最後の結果からデータを受け継ぎたい場合に使用されます。

新しい値をセット 新しい値をセットします。

**定数によって** データに定数を指定します。

**データノードから** 他のデータノード(収集ノードまたは演算ノード)からデ のコピーによって ータをコピーします。数値タイプの収集ノードだけがデ

Iピーによって ータをコピーします。数値タイプの収集ノードだけがデ ータソースとして選択することができることに注意して 下さい。

式によって 値を指定するのに式を使用。

 INIT フラグのチェ
 新しい値をセットの初期値は INIT フラグのチェックさ

 ックされたリンク
 れたリンク入力のチェックによってオンにされた場合に

 入力によりアクテ
 のみ処理されます。

ィブにされる

Input > Device プロパティ

Define Node [Collect] - col	lect2	
Basic Comment Screen Position Input Barcode Lookup File Formula Verify Lookup File Qualifier Function Keys Links, in Links, out	<ul> <li>▶ 時計:</li> <li>● 日付のみ ● 時刻のみ ● 日付と時刻</li> <li>● 日付タイブ: WM/DD/YYYY ▼</li> <li>● スキャナ:</li> <li>■ 入力無いときに自動トリガ</li> <li>■ 入力データ検証のために二回スキャン</li> <li>■ バーコード読み取り後にスキャナをオフにする</li> <li>● 禁字と + - ● 英字 ■ 記号</li> <li>● Enterで終了</li> </ul>	
OK キャンセル ヘルプ		

ポータブルターミナル装置へのデータ入力方法が三つあります。時計、スキャナそしてキー パッドです。複数の選択が可能な場合、実際の入力は一度に一つの装置からだけ入ります。

### • 時計

自動的に時計からデータを得て、入力を埋めます。三つのデータフォーマットがあります。こ れらは 日付のみ(MM/DD/YYYY)、時刻のみ(HH:MM:SS) と日付と時刻(MM/DD/YYYY HH:MM:SS)で す。

#### • スキャナ

スキャナの入力を可能にします。バーコードシンボルは <u>Barcode プロパティページ</u>でセット することができます。個々の収集ノードはバーコードシンボルについて独自の設定を持つこと ができます。ジョブ中のすべての収集ノードについての標準のバーコードシンボル設定はジョ ブ定義、Barcode (default) プロパティページで定義されます。

### • バーコード読み込み後スキャナをオフにする

バーコードを正しく読んだ後でスキャナをオフにするためにこのオプションをチェックします。 いつオンにするかを気にする必要はありません – 収集ノードはこれが選択した装置の一つで ある場合にスキャナを自動的にオンにします。

通常は、スキャナは収集ノードでバーコードを読んだ後オン状態のままですので、リリースし てスキャナのトリガを再度引くことなしには以降の収集ノードでデータのスキャニングをしま せん。各スキャンはトリガを引くことによってオンにしなければなりません。しかし、オン状 態のままのスキャナはデータ入力のない時間に間違ったスキャンをすることになります。ジョ ブエンジンは現在の実行ノードが収集ノードあるいは演算ノード、またはプログラムノードで なければ、スキャナを自動的にオフにします。

#### ・ キーパッド

入力のためにキーパッドを使用可能にします。

### • 数字と + - .

数字キー: 0 – 9、小数点:. 、そして正負符号 + と – のみがキー入力として受け入れられます。

#### 英字

大文字と小文字の英字キー: a – z 、 A – Z、とスペースがキー入力として受け入れられます。

#### 記号

\$、#、!、等の記号だけがキー入力として受け入れられます。

### • Enter で終了

入力を終了するためには、Enter キーを押さなければなりません。これをチェックしない場合、 入力が最大長に達したときに自動的に終了します。

キーは三つの種類:通常キー(0から9数字キー、そしてAからZ英字キー)、ファンクションキー(F1からF8、そして矢印キー)および特殊キー(CMD、スキャンと電源キー)に分けられます。

ファンクションキーはデータ入力の操作で特別な機能を持っています。入力操作は、ノード、 あるいは現在のノードに入る、あるいは出るリンクに対して定義されたファンクションキーが 押された場合キャンセルするようにすることができます。通常は、ファンクションキーは特別 なノードによる操作手順の変更に使用されます。

通常キーとファンクションキーは<u>キーマップ</u>でコードを再定義することによって他のキーに 割り当て直すことができます。

Input > Barcode プロパティ

Define Node [Collect] - co	lect2	
Basic Comment Screen Position Input Device Barcode Cookup File Formula Verify Lookup File Qualifier Function Keys Links, in Links, out	<ul> <li>✓ この収集ノードについてパーコードシンボルをセット * 各個々の設定を変更するためにダブルクリック</li> <li>✓ Codabar *, Send Start/Stop Characters</li> <li>✓ Code 128</li> <li>✓ Code 32 *, Send Leading Character 'A'</li> <li>✓ Code 39 *, Send Check Digit</li> <li>Code 93</li> <li>✓ EAN 128</li> <li>✓ EAN 13 *, Send Leading Digit, Send Check Digit</li> <li>✓ EAN 8 *, Send Check Digit</li> <li>✓ EAN 8 *, Send Check Digit</li> <li>✓ EAN 9 *, Send Check Digit</li> </ul>	
	OK キャンセル ヘルブ	

選択された場合、それぞれの収集ノードは、それ自身のバーコードシンボル設定を持っている ので、別な収集ノードは別なスキャナの動作をします。

アスタリスク(\*)が付く名前のバーコードは詳細な設定が必要です。設定ウインドウを開く ために名前をダブルクリックして下さい。チェックされたサブ設定は名前の次に表示されます。 標準のバーコードシンボル設定は Job Define、 Barcode (default)ページで行われます。

# Input > Lookup File プロパティ

Define Node [Collect] - co	lect2	X
Basic Comment Screen Position Input Device Barcode Cookup File Formula Verify Lookup File Qualifier Function Keys Links, out	<ul> <li>データはルックアップファイルから自動的に検索されます。 これはルックアップファイルデータ検証が定義される収集 ノードで定義されます。</li> <li>収集ノード:</li> <li>ルックアップファイル:</li> <li>フィールド番号:</li> <li>① (1からカウント)</li> <li>「ルックアップファイルからデータ検索に失敗した場合標準値を適用:</li> <li>標準値:</li> </ul>	
OK キャンセル ヘルプ		

多くの場合、ルックアップ検証における以前の成功を元にしたルックアップファイルからデー タを検索する必要があり、関連するデータをベースにしてデータを処理することができます。

ルックアップファイルを定義するためにルックアップ検証を実行するすべての収集ノードから 名前を選択します。成功したルックアップ検証を必要とするのでルックアップファイルを直接 指定することはできないことに注意して下さい。そして、フィールド番号を指定します。レコ ード番号は、ノードのルックアップ検証から一致したレコード番号によってジョブ実行時に決 定されます。

収集ノードがルックアップ検証に失敗した場合、ルックアップファイルからの入力は終わらず、 そして入力方法がキーボード入力に変更されます。

X

Define Node [Collect] - collect2 データは下に示す式に従って自動的に生成されます。 そして、すぐにデータ検証に進みます。 Basic Comment 🚊 Screen Position  $\overline{}$ 🚊 Input Initial Value Device Barcode Lookup File -Formula - Verify Lookup File Qualifier -Function Keys Links, in V Links, out  $\leq$ > OK キャンセル ヘルプ

Input > Formula(式) プロパティ

データを生成するための式として C 言語で記述して下さい。

収集ノードは実際に他によってアクセスすることのできるデータの変数を持っています。アク セスするには、下線('\_')のプリフィックスを持つノード名であるその変数名を使用します。

# Verify (検証)プロパティ

Define Node [Collect] - co	llect2	×	
Basic Comment Screen Input Initial Value Device Barcode Lookup File Formula Verify Lookup File Qualifier Function Keys Links, in Links, out	長さ:         From:       To:         範囲:         From:       To:         「Vクチャ:         ビクチャ:         「検証パスしなかった場合再入力しない」         検証パスしなかった場合モラー記号(文字 *)を印字しない		
OK キャンセル ヘルプ			

範囲	入力データの値の範囲を指定します。	これはその
	文字の ASCII コードと比較されます。	

**ピクチャ** 入力文字タイプのマスクパターンを定義します。

検証パスしなか 普通は、基本的な検証、長さ、範囲とピクチャは 入力データの基本的なチェックとして扱われま す。入力データは、プロセスを進める前に、これ らの三つの状態に合わなければなりません。そう でなければ、ユーザはデータが正しく渡されるま で再入力させられます。

検証パスしなか 検証が失敗した場合、ユーザの警告のためにブザ
 った場合ビープ ーが鳴ります。検証が失敗した場合にブザーを切
 音を出さない るためにこのオプションをチェックして下さい。

検証パスしなか 検証に失敗した場合、間違ったデータ入力のエラ
った場合、エラ 一記号はスクリーンに表示されません。
一記号(文字\*)を

印字しない

# <u>第5章 ジョブのプログラミング</u>

ピクチャパターン

а	英文字 <b>(A-Z, a-z)</b> のみ可
n	数字(0-9) と '+', '-', '.' (ドット), 'E' と 'e'のみ
0	数度 (0-9) と '+', '-' のみ
р	数字 (0-9)のみ
I	小文字 <b>(a-z)</b> のみ
u	大文字 <b>(A-Z)</b> のみ
x	印刷可能な文字 (0x320x7f)
z	全 ASCII コード
_	全 ASCII コード
#	文字を削除
@	@ と @ 間の文字列
%	インジケータ、パターンにシンボルを追加
?	インジケータ、非固定長のパターン
!	インジケータ、検証結果を逆に
(space)	パターンのターミネータ

ピクチャ検証の例:

ピクチャ	入力データ	結果
ррр-рррр	1112222	1112222
ррр-рррр	111-2222	1112222
ррр-рррр	111.2222	エラー
ррр-рррр	11-22222	エラー
aaa-aaaa	1112222	エラー
#рр-рррр	1112222	112222
#рр-рррр	111-2222	112222
aa#aaaa	PC-WAND	PCWAND
###-zzzz	1112222	2222
###-zzzz	111-2222	2222

pp@bcd@pp	11bcd22	11bcd22
pp@bcd@pp	11abc22	エラー
\$ррр-рр	11133	11133
\$ррр-рр	\$11133	11133
\$ррр-рр	111-33	11133
\$ррр-рр	\$111-33	11133
%ppp-pppp	1112222	111-2222
%ppp-pppp	111-2222	111-2222
%"ppp-pppp"	1112222	"111-2222"
%\$ppp.pp	11133	\$111.33
%\$ppp.##	11133	\$111.
?р	1	1
?р	111	111
?ppp	11	エラー
?aaap	abc123	abc123
?aaap	abc	エラー
?%\$ppp	123	\$123
?%\$ppp	12345	\$12345
?%\$ppp	12	エラー

Define Node [Collect] - col	lect2	
Basic Comment Screen Input Initial Value Device Barcode Lookup File Formula Verify Qualifier Function Keys Links, in Links, out	<ul> <li>データファイル検索によって検証:</li> <li>ルックアップファイル:</li> <li>フィールド番号:</li> <li>ロ</li> <li>フィールド合計:</li> <li>ワ</li> <li>フィールド分離子:</li> <li>, (Comma)</li> <li>★</li> <li>★</li> <li>株式開始:</li> <li>● 始めから</li> <li>● 現在位置から</li> <li>○ 完全キー一致</li> <li>■ 高速バイナリ検索法適用</li> </ul>	
	OK キャンセル ヘルプ	

Verify > Lookup File (ルックアップファイル)プロパティ

これはデータファイル(データベース)の入力データをチェックするのに使用される便利な機能です。

フィールド番号は1からカウントが始まります。これがパスしたとき、リンク状態に使用され る成功フラグが上がります。そうでなければ、異常フラグが上がります。

JobGen Plus はジョブファイルとルックアップファイルを自動的にダウンロードします。これ が大きなファイルで、ジョブ開発中に変更されない場合は、毎回ダウンロードする必要はあり ません。 <u>ジョブ定義(標準設定)</u>、ジョブ作成ページで、ルックアップファイルの自動ダウン ロードを禁止して下さい。

#### • 検索開始

最初から、もしくは現在位置から、のいずれかからデータファイルを検索します。

### 完全キー一致

完全なデータの一致を指定します。例えば: "abc"は "abc"に完全に一致します; "abc"は "abcdefg"と完全には一致しません; "abc"は "1abc23"には一致しません。

#### 高速バイナリ検索法適用

10万件のデータを含むような大きなデータベースを検索する場合、高速検索を可能にするため にこのオプションをチェックして下さい。JobGen Plus は自動的にファイル – バイナリ検索 テーブル - を追加し、そして元のデータファイルとジョブファイルと共にダウンロードしま す。

バイナリ検索の欠点は、データファイルを実行時に変更できないことです。データファイルの 変更はデータファイル中ですべてのレコードの正確な位置を記録しているレコードであるバイ ナリ検索テーブルを壊してしまいます。バイナリ検索テーブルの間違った情報で、検索は予期 しない結果が戻ります。

しかし、ある規則を加えることによって、バイナリ検索中にデータファイル中のあるデータの 変更が可能になります。この規則は以下の通りです:

- キーフィールドは変更することができない。
- レコードの長さは一定。各レコードは同じ長さである必要はありません。レコードを変更した後で、実際のレコード長が変更していないことを保証します。

Define Node [Collect] - co	llect2	×
Basic Comment Screen	□ 以下で定義されたクォリファイアによって検証	
Position 	_RESULT_ =	
	OKキャンセル ヘルブ	

Verify > Qualifier (クォリファイア)プロパティ

これは C の式です。事前定義変数 \_RESULT\_ に値がゼロでなければ True(真)、そうでなけれ ば False(偽)を結果にセットします。これがパスした場合、リンク状態として使用される成功 フラグが上がります。そうでなければ、異常フラグが上がります。

#### 演算ノードの作成

演算ノードは、数値の処理または単にデータフィールドに定数をセットするために使用されま す。収集ノードと演算ノードの両方はデータノードで、これは実際にデータを保持し、そして レコードのフィールドとして選択することができます。

Basic プロパティ

Comment プロパティ

Screen プロパティ

Screen > Position プロパティ

Calculate > Formula プロパティ

Compare > Qualifier プロパティ

Function Keys プロパティ

Links In プロパティ

Links Out プロパティ

これらのプロパティは収集ノードのものと同じです。

# Initial Value > Data Type プロパティ

演算ノードは計算を目的としているので 数値データのみを持つことができます。違うデータ タイプをいっしょに扱う場合、結果は自動的に大きなデータタイプに変換されます。

# Calculate プロパティ

Define Node [Math] - math	3	×
Basic     Comment     Consent     Position     Initial Value     Calculate     Formula     Compare     Function Keys     Links, in     Links, out	Calculate with numerical data 演算子: + (add) マ オペランド: ② 定数: ⑤ 数値タイプのデータノード: 『	
	」 の 速 が漢 M 具 のKキャンセルへルプ	ß

計算の演算子とオペランドを定義します。

• 演算子:

+(add), -(minus), \*(multiply), /(divide),  $\geq$  None( $\exists L$ ).

# • オペランド:

# • 定数:

オペランドとして定数を指定。

• 数値タイプのデータノード:

データノード、これは収集ノードまたは演算ノードからデータを得ます。注意 – 数値 データタイプの収集ノードだけをここで選択することができます。

# Compare (比較)プロパティ

Define Node [Math] - math	3	
Basic Comment Screen Initial Value Calculate Qualifier Function Keys Links, in Links, out	Compare with numerical data 濱算子: None オペランド: の定数: の数値タイプのデータノード:	
		i 加漢 🏿 🗐 🧭

比較の演算子とオペランドを定義します。比較の結果はリンクの成功または失敗状態として使 用することができます。

• 演算子:

```
<(less)、<=(less or equal)、>(greater)、>=(greater or equal)、==(equal)、
!=(not Equal)、と None(なし)。
```

- オペランド:
  - 定数:

オペランドとして定数を指定。

# • 数値タイプのデータノード:

データノード、これは収集ノードまたは演算ノードからデータを得ます。 注意 - 数値データタイプの収集ノードだけをここで選択することができます。

### 編集ノードの作成

編集ノードは、実行時にポータブルターミナルのデータファイルに含まれるデータを表示し、 変更するために使用します。

#### 編集ノードの定義

Basic プロパティ

Comment プロパティ

Function Keys プロパティ

Links In プロパティ

Links Out プロパティ

これらのプロパティはメニューノードのものと同じです。

Edit プロパティ

レコード定義			$\mathbf{X}$
⊡-New Record Separator EOR Links Comment	レコード名: PAR_ データファイル名: PAR_ 「 これは標準のレコード データノード: collect2 math3	TEXT TEXT  *またはデータファイル シン くく	
	OK +	+ンセル ヘルプ	連

- **レコード名** データファイル名を定義するためにレコードを 選択します。
- **データファイル名** 実行時に編集ノードで操作するファイルを定義 します。

レコード定義				
⊡-New Record EOR Links Comment	ブリアンブル: ポストアンブル: フィールド分離子: レコード終端子:	, (Comma) CR-LF		
	OK	キャンセル	ヘルブ	]

フィールド分離子 フィールド区切り文字を定義します。

編集ノードは、いくつかの異なる作業ノードを持っています。 これはスタート時に**ブラウズ** モード(表示モード)に入り、そして最初のレコードの、最初のフィールドにあるデータを表示 します。スクリーンは<sup>"</sup>B, R:xx, F:xx<sup>"</sup>でユーザに入力要求(プロンプト)を示し、ここで B は ブラウズ(Browse)モード、そして R: と F: はそれぞれ現在のフィールドの、レコードとフィ ールドの座標を表します。

行(レコード)中のフィールドを前方または後方に移動するには右または左の矢印キーを押して下さい。列(フィールド)中のフィールドを移動するには、上(または F1 ファンクションキー)あるいは下(または F2 ファンクションキー)を押して下さい。

フィールドに直接移動(ジャンプ)するには、新しいR:とF:の座標を入力するために F3 ファ ンクションキーを押して下さい。Rの後の、かっこ中の数はファイル中のレコード数の合計で す。

ブラウズ(表示)モードで、**編集モード**に入るために Enter を押して下さい。これは入力要求を "E, R:xx, F:xx"に変えます。E は編集(Edit)モードを表します。編集モードでのみユーザはフ ィールドデータを変更することができます。ファイルに変更を保存するためにデータを変更 した後で再度 Enter を押して下さい。編集ノードの選択はブラウズ(表示)モードに戻ります。

データを変更中に、左矢印キーはバックスペースキーのように働き、前の文字を削除します。 右矢印キーを押すと、すべての削除した文字を復元します。このファンクションは、データの 内容を覚えなくてもよいので変更を容易にします。

**検索モード**に入るために F4 ファンクションキーを押して下さい。検索は現在のフィールド番号で定義された欄について検索を現在のレコードから開始します。入力要求は "S, R:xxx, F:xxx"を表示し、ここで R は開始レコード番号を表し、そして F は列番号を表します。入力要求の後で、ユーザに対して検索するための入力データ(標準値は現在の表示データ) を待ちます。

検索は、最初からの部分的なデータが入力データと全く同じ場合に一致します。データを見つけた後で、SEARCH NEXT? を質問します。YES の答えは次のレコードから続けて検索を始め、 あるいは NO は、検索を終了して、一致しているフィールドを表示するために自動的にブラウズ(表示)モードに戻ります。 データが見つからなかった場合、入力要求(プロンプト)は "Not found" を表示し、そして次の検索のための入力待ちに戻ります。検索モードをやめるには ESC を押して下さい。

編集ノードを終了するためにはブラウズモードで ESC キーを押します。

# 消去ノードの作成

消去ノードはデータファイルからレコードを削除するため、あるいはデータファイル全体を消 去するために使用されます。

消去ノード定義

Basic プロパティ

Comment プロパティ

Function Keys プロパティ

Links In プロパティ

Links Out プロパティ

これらのプロパティはメニューノードのものと同じです。

# Erase プロパティ

Define Node [Erase] - eras	25	
Basic Comment Frase Function Keys Links, in Links, out	<ul> <li>● データファイル全体を削除</li> <li>● 最後のレコードを削除</li> <li>レコード名: PAR_TEXT</li> <li>データファイル名: PAR_TEXT</li> </ul>	
	OK キャンセル ヘルプ	

データファイル全 体を削除	ポータブルターミナルからデータファイル全体を消去 します。
最後のレコードを 削除	データファイルから最後のレコードを削除します。
レコード名	データファイル名を定義するために、レコードを選択 します。

**データファイル名** ファイルを定義します。これは実行時に消去ノードで 操作されます。

### アップロードノードの作成

アップロードノードは、ターゲットのポータブルターミナルからホストコンピュータへデータファイルを送信するために使用されます。

アップロードノード定義

Basic プロパティ

Comment プロパティ

Function Keys プロパティ

Links In プロパティ

Links Out プロパティ

これらのプロパティはメニューモードのものと同じです。

# Upload プロパティ

Define Node [Upload] - uplo	ad6	×
Basic Comment Upload Function Keys Links, in Links, out	<ul> <li>○ データファイル中の全レコードをアップロード 現在のレコードをアップロード</li> <li>レコード名: PAR_TEXT ▼</li> <li>アップロードの最後でファイル終端子 (Ctrl-Z) 送信</li> <li>ポータブルのCOMポートセット: 転送速度: Default ▼</li> <li>プロトコル: Muliti-point ▼</li> </ul>	
	OK キャンセル ヘルプ	

アップロードノードの機能は、マルチポイントプロトコル でポータブルターミナルからホス トヘデータを転送することです。アップロードノードの実行で、送るために定義されたすべて のデータは連続的にデータをポーリングしているホストアプリケーションに即座に送られます。

**データファイル** ファイル中のすべてのレコードをアップロード。

注の全レコード

をアップロード

現在のレコード 現在のレコードをアップロード。

をアップロード

**レコード名** アップロードするレコードを選択。レコードはどのフ ィールドをアップロードするかを定義します。

**アップロードの** アップロードの最後でファイル終端子(Ctrl-Z) を送

**最後でファイル** 信。<sup>②</sup>

終端子送信

### プログラムノードの作成

プログラムノードはCプログラムを書くためのスペースをユーザに提供します。

プログラムノードの定義 Basic プロパティ Comment プロパティ Function Keys プロパティ Links In プロパティ Links Out プロパティ これらのプロパティはメニューノードのものと同じです。

### Basic プロパティ

プログラムノードは Basic プロパティに追加の選択があります。

# ・ グローバルアクセス

このプログラムで定義された機能に他からアクセス(呼び出す)することを可能にします。例えば、入力式の、検証修飾子(クォリファイア)の、あるいはリンクの他のプログラムノードによって呼ばれます。すべてのグローバルアクセスプログラムノードは左のプロパティパネルの Job ¥ Common Define エントリにリストされています。

C 関数は呼ばれる前に定義されなければなりません。 JobGen Plus はコードを生成するとき にすべての、他のノードの前に常にグローバルアクセスノードを置きます。そしてアルファベ ット順に処理されます。

# Program プロパティ

Define Node [Program] - pr	gram7	×
Basic Comment Function Keys Links, in Links, out		
	OK キャンセル ヘルプ	

プログラムノードは、Cコードで皆さん独自の関数を書くことのできる場所を実際に提供しま す。編集ボックスにコードを書くか、あるいは大きな編集ウインドウを開くために右下の矢印 ボタンをクリックすることができます。戻るには単にウインドウを閉じます。

各プログラムノードは、関数のエントリを持っており、このプログラムノードの実行時に呼び 出されます。エントリはプリフィックスの下線('\_')が付いたプログラムノードの名前です。例 えば、プログラムノード名が "ring"の場合、"\_ring()'の名前の関数がこのプログラムの実行 時に呼び出されます。

多数の関数がプログラムノードで呼び出されます。これらは、ジョブエンジン関数、ある標準のCライブラリ関数、そしてファームウェア関数の三種類に分けられます。

ジョブエンジン関数の詳細な説明については、ヘルプの内容の <u>関数ライブラリ</u>をチェック して下さい。標準C関数については *Microsoft C 言語ヘルプ*のランタイムルーチンをチェッ クして下さい。ファームウェア関数の詳細な説明については、ポータブルターミナルのプログ ラミングマニュアル をご覧下さい。

### ジョブ実行ノードの作成

一つのジョブは他のジョブの中で実行することができます。大きなジョブを複数の小さな独立 したもの(モジュール)に分割するためにこのテクニックを使用して下さい。小さなジョブはメ インテナンスが簡単であることを意味します。

注意、ジョブ実行は、ジョブ呼び出しとは違っています。ジョブ実行は呼び出した側には戻り ません。ポータブルターミナルのジョブエンジンは一度にメモリ中に一つのジョブだけを持っ ています。現在のジョブはジョブ実行が新しいジョブを読み込んだときに消えます。ジョブ実 行を実行する前にすべてのデータファイルを保存することを忘れないで下さい。

サブジョブは呼び出したジョブを実行することによって呼び出したジョブに戻ることができま す。開始ノード名を指定することによって、ジョブエンジンは、元の開始ノードに代わって指 定したノードからスタートすることができます。

プログラムノード定義

Basic プロパティ

Comment プロパティ

Function Keys プロパティ

Links In プロパティ

Links Out プロパティ

これらのプロパティはメニューノードのものと同じです。

Run Job (ジョブ実行)プロパティ

Define Node [RunJob] - rur	ijob8	
Basic Comment Function Keys Links, in Links, out	現在のジョブはメモリから消されます、言い換えると、ひと つだけのJobGen Plus ジョブが一度にボータブルで実行可能 です。他のジョブを読み込むためにこのノードを実行する前 にすべてのデータファイル(レコードとステータスデータ)を 保存して下さい。 ジョブファイル名: 開始ノード名:	
	OK キャンセル ヘルプ	

- **ジョブファイル名** ジョブ実行ファイル名の定義
- 開始ノード名
  開始ノード名の定義。標準では、ジョブエンジンは ジョブで定義された開始ノードから実行します。これは別なノード名を入力することによって変更できるので、ジョブは指定したノードから開始します。

# コメントノードの作成

コメントノードはフローチャート上に直接コメントを貼り付けるために使用されます。



### リンクの作成

リンクは JobGen Plus アプリケーションを構築する基本的なコンポーネントです。これらはあるプロセス(ノード)から他へのパスだけではなく、プログラム実行のフローです。

**リンクの作成方法:** ツールバーのリンクアイコンをクリックするか、あるいはノードメニュー でリンクを選択することによってリンクツールを選択します。開始ノードを指して、マウスカ ーソルを相手先ノードにドラッグして、マウスボタンを離します。

**自己ループするリンクを作る方法**: ノードボタンを指して、カーソルを一方の側に近づけ、 ノードボタン中の別の側にマウスをドラッグして、マウスボタンを離します。自己ループが作 成されます。

ノードは複数の出ていくリンクと入ってくるリンクがあります。同じノードに複数のリンクが あるだけでなく、各リンクは異なる条件を持ちます。

接続元のノードから複数のリンクを決めるリンクの条件が選択されます。リンクはリンクの条件が TRUE と評価された時に選択されます。

# リンク定義プロパティ

Define Node [Edit] - Edit		×
Basic Comment Edit Function Keys ⊡-Links, in ⊆-Cinks, out Links, out Section, #4	<ul> <li>○ キーパッド入力:</li> <li>○ 入力データ:</li> <li>○ ノードステータス</li> <li>○ 常にパス ○ 成功 ○ 失敗</li> <li>○ 式:</li> <li>□ FESULT_=</li> <li>□ INIT, 初期化動作</li> <li>□ RRLI, 全ノードについてリセット "最後の入力繰り返し"</li> <li>□ アラーム</li> <li>□ * ジョブ作成時に除く</li> </ul>	
	OK キャンセル ヘルプ	

# • キーパッド入力

このオプションを選択するためにクリックし、そしてポータブルキーパッドウインドウを立ち 上げるためにその編集ボックスをクリックします。それをクリックすることによってキーを選 択します。リンクについては一つのキーだけを受け入れることができます。ジョブが接続元の ノードを終了して、リンク選択についての入力を待っているとき、ここで定義したキーを押す とこのリンク状態が TRUE(真)にセットされます。
#### 入力データ

このオプションは接続元のノードが収集ノードまたはカウントタイプの場合にのみ有効です。 これは個々で指定したデータとノードのデータの値を比較して、これらが同じなら結果を TRUE(真)にセットします。

### • ノードステータス

このオプションは、接続元ノードの実行結果チェックのためのものです。メニュー、消去そし て編集ノードについては、その実行は常に成功です。収集ノードについては、その実行は入力 データが検証をパスした場合にのみ成功です。アップロードノードについては、タイムアウト が起こった場合、その実行結果は失敗です。

検証は4つのオプション常に、成功、失敗、そしてなしを提供しています:

- **常にパス** 接続元ノードの実行結果のステータスには関 係ない-これは常に 真(TRUE)
- **成功** 接続元ノードの実行結果が成功の場合にのみ この条件は真(**TRUE**)になる。
- 失敗接続元ノードの実行結果が失敗の場合にのみ<br/>この条件は真(TRUE)になる。
- 式

これは C の式です。これは収集ノードの検証定義のクォリファイア(<u>Qualifier</u>) と同じフォ ーマットです。事前に定義された変数\_RESULT\_の値は条件の結果として評価されます。

#### EOR、レコード書き込み(End of Record, レコード終了)

現在の指定したレコードの入力が完了し、そしてこのリンクが実行された時にデータファイル に保存されることを示すためにこのオプションをクリックします。レコードが定義された場合 EOR を定義したリンクが少なくとも一つあります。複数の EOR(これは複数のリンクが EOR 選択 を持っている)はレコードを終了するために異なる条件によって適用されます。

ジョブエンジンは、実行があるノードでループしている場合には終了やレコードの保存をしま せん。ジョブエンジンがレコードを保存するただ一つの方法はリンクの実行時に EOR に合うこ とです。

どのレコードがデータファイルに保存されるかをジョブエンジンに知らせるためにレコード名 を指定して下さい。

#### 第5章 ジョブのプログラミング

#### • INIT, 初期化動作

収集と演算ノードの初期処理で、新しい値をセットオプションを有効にする時間であることを 示すためにこのオプションをクリックします。

#### • RRLI、全ノードについてリセット、 "最後の入力繰り返し "

次のレコードが新しい入力を必要としていることを示すためにこのオプションをクリックしま す。これは収集ノードの入力セッションで、最後の入力を繰り返す設定と一緒に使用されます。 ジョブエンジンが新しい入力レコードのチェックされたリンクに会った場合、最後の入力繰り 返しのチェックされた収集ノードを実行する次の時に新しいデータの入力を行います。新しい 入力レコードの機能は次のレコードのみに影響し、最後の入力繰り返しを持つ収集ノードの後、 リンクで再度新しい入力レコードに会うまで繰り返すことができます。これは欄中の繰り返さ れたフィールドが繰り返し新しい入力を得ることができる一つの方法です。

### • アラーム

このリンクを実行しているときにブザー音をならすためにこのオプションをクリックします。

#### リンク条件の優先度

ジョブがノードでの実行を終えた後で(この時点で、このノードは接続元のノード)、どのノードを次に実行するノードかを決めます。この決定はすべての出ていくリンクのすべての条件を チェックすることによって、そしてどれがリンク状態に合うかを決定することによって行われます。 これらの条件の評価は以下の優先度順に行われます。

1). ノードが実行の中間で入力を待っている場合に(例えば、収集モードでのキーパッド入力)、<u>特殊キー</u>を押すと現在の実行を終了します。ジョブエンジンは、ちょうど押された特殊 キーがこれらのリンクの一つで**キーパッド入力**条件で定義されている場合に決定するために このノードから接続されているすべてのリンクをチェックします。見つけた場合、このリンク が選択され、そして実行フローはこのリンクの相手先ノードに行きます。このようなキーがこ れらのリンクで定義されていない場合、特殊キーを押すと無視され、ノードの実行が続きます。

**注意**:特殊キーはこれが別なノードで定義されたものかどうかを見るためにもチェックされま す。ジョブエンジンがこの特別なファンクションキー定義されているノードを見つけた場合、 ノードは実行する次のノードとして選択されます。

2). 接続元ノードの実行が終わった場合、ジョブ・エンジンはノードステータス条件でアクティブにセットされた成功または失敗があるかどうかを決めるためにノードから接続されているすべてのリンクをチェックします。リンクが失敗に定義されており、そして接続元ノードの実行結果が成功の場合、このリンクは選択されません。しかし、もし結果が失敗の場合、このリンクが選択され、そして実行はこのリンクの相手先ノードに行きます。

3).式の条件があり、そして論理表現の計算結果が真(TRUE)、そしてリンクが選択された場合、 実行はこのリンクの相手先ノードに行きます。

4). 入力データ条件が定義されており、そしてデータが定義と一致し、そしてこのリンクが選択された場合、実行はこのリンクの相手先ノードに行きます。

5). **常にパス**オプションが**ノードステータス**)条件で定義されている場合、ジョブはこのリン クの相手先ノードに行きます。常にパスオプションを定義しているリンクがない場合、このス テップではリンクは選択されず、そしてジョブは入力を待ちます。

6). **キーパッド入力**条件が定義されており、そして押されるキーがこの定義と一致し、そして このリンクが選択された場合、実行はこのリンクの相手先ノードに行きます。

リンクを選択する6つの優先度があり、そしてすべてのリンクはリンクのシーケンス番号順に 評価されます。リンクが選択された場合、実行は低い優先度の条件をチェックしないで直ちに 相手先ノードに行きます

これらの優先度のステップを実行した後で、リンクが見つからなかった場合、ジョブはここで 停止します。これが不適切なプログラミングの結果起こった場合、ジョブを終了するために ESC キーを押して下さい。

このノードから接続されているリンクが全くない場合、ジョブの実行は自動的に終了します。

### 第5章 ジョブのプログラミング

# レコードの定義

各ジョブは複数のレコードを持つことができます。レコードはフィールド、レコード区切り、 プリアンブルとポストアンブルで構成されています。

レコード定義		×		
⊡-ASSETMGT Separator EOR Links Comment	レコード名: ASSETMGT データファイル名: MSSETMGT.TXT ▼ これは標準のレコードまたはデータファイル データノード: レコードフィールド: Asset_number Section			
OK キャンセル ヘルプ				

レコードの属性を書くフィールドは、収集ノードです。<u>収集ノードを定義して、</u>そしてレコー ドフィールドのチェックボックスを選択した場合、 このノードは自動的にフィールドになり ます。標準では、JobGen Plus はすべての定義された収集ノードをレコードのフィールドとし てセットし、そしてレコード中のフィールドの順序はノードの定義の時刻 – 最初に定義され た収集ノードが最初のフィールド、次の収集ノードが二番目、等々 - で決定されます。フィ ールド分離子はフィールドの分離のために使用され、そしてレコード終端子はレコードを分離 するために使用されます。プリアンブルとポストアンブルは文字列です。プリアンブルはレコ ードのデータの前にセットされ、ポストアンブルはレコードのデータの後にセットされます。

各レコードは、データを保存するために TXT ファイルを持っています。データファイル名はユ ーザによって定義することができ、そしてその標準名はジョブ名に拡張子.TXT が付いていま す。例えば、 demo .txt はジョブ demo のデータファイルです。データファイルは<u>データファ</u> <u>イルフォーマット</u>で定義された多数のレコードを保存します。

レコードを定義する二つの方法があります。最初の方法は収集ノードの定義時に行われますが、 レコードのフィールドだけがこの方法で定義されます。他の方法はレコード定義ダイアローグ ボックスを出すために編集メニューからレコード定義を選択することです。

レコード定義		×		
- ASSETMGT Separator - EOR Links - Comment	レコード名: ASSETMGT データファイル名: MSSETMGT.TXT ▼ これは標準のレコードまたはデータファイル データノード: レコードフィールド: Asset_number Section			
OK キャンセル ヘルプ				

レコード定義ダイアローグボックスには、二つのスクロール・ウインドウ: **データノード**:、 左のウインドウと**レコードフィールド**:、右のウインドウがあります。ジョブで作成されたす べての収集ノードは、データノード:ウインドウにリストされ、そしてレコード中のすべての フィールドは、レコードフィールド:ウインドウにリストされます。これらの二つのウインド ウ間には追加(>>) と 削除(<<) ボタンがあります。

収集ノードをレコードのフィールドになるようにしたい場合は、選択するためにデータノー ド: ウインドウでノードをクリックし、そして 追加(>>) ボタンをクリックします。このノー ドはレコードのフィールドとなり、そして レコードフィールド: ウインドウに現れます。

レコードのフィールドを削除するには、レコードフィールド: ウインドウのノードをクリック し、そして 削除(<<) ボタンをクリックし、そしてフィールドは削除され、そしてウインドウ から除かれます。

フィールド と レコード分離子(区切り文字)を定義するための二つのプルダウンウインドウが あります。カンマ、セミコロン、タブ(ASCII 9)、そして LF&CR 文字のみがフィールドまたは レコード区切り文字として選択することができます。一つの文字を同時にフィールドとレコー ドの両方を区切り文字として選択することはできません。

フィールドの順序はフィールドが作成される順に従っています。

プリアンブルとポストアンブルは適当な編集ボックスで文字列をタイプすることによって定義 することができます。

最終的に、作業を保存するために OK をクリックして下さい。

レコードフィールドチェックボックスで変更の効果を見るために、Define Collect (収集定義)ダイアローグボックスを立ち上げるために変更した収集ノードをクリックします。

ジョブエンジンは、実行があるノードの輪でループしている場合、レコードの完了と保存しません。これは単に古いデータに上書きします。ジョブエンジンがレコードを保存するただ一つの方法はリンクの実行中で EOR (End Of Record) に出会ったときです。

## 第5章 ジョブのプログラミング

# データファイルフォーマット

データはレコードごとにデータファイルに保存されます。レコードの最大長は1024 バイトで す。レコード・フォーマットは以下の通りです:

## [Pre]<f1><fd><f2><fd>...<fn>[<fd>[Post]]<EOR>[CR&LF]

ここで:

<f1><f2><fn></fn></f2></f1>	フィールド1、フィールド2、フィールドn
:	フィールドの最大長は 64 バイトです。
	N はレコードのフィールド番号です。最大値はフ ィールドの長さ、プリアンブルとポストアンブル の長さに依存します。
< <b>fd&gt;</b> :	ユーザ定義のフィールド区切り文字(分離子)。
[ <b>Pre</b> ]:	ユーザ定義のプリアンブル。
[ <fd>[Post]].</fd>	「Post] けューザ定義のポストアンブル ポストフ

[<fd>[Post]]: [Post] はユーザ定義のポストアンブル。ポストア ンブルがある場合、fd はその前に挿入しなければ なりません。

**<EOR>**: ユーザ定義のレコード区切り文字(終端子)。

[CR&LR]: EOR が CR&LF でない場合、JobGen Plus はレコー ドの最後に自動的に CR&LF を追加します。EOR が CR&LF の場合、 JobGen Plus はそれを無視しま す。レコードの最後には CR&LF がひとつだけでな ければなりません。

# 第6章 ジョブの作成

すべてのノードとリンクを作成した後で、次のステップは実行可能なジョブを作ることです。

JobGen Plus は Job の実行形式コードを作るために二つのステップを行います。最初は、フロ ーチャートですべての設定に従ったソースコードを生成します。メッセージウインドウにエラ ーと警告があれば表示します。そして、最終的に実行可能なコードを生成するためにソースコ ードをコンパイルします。メッセージウインドウにまたエラーと警告があれば表示します。

エラーがあった場合(赤文字でメッセージウインドウに表示される)、その上を単にダブルクリ ックすると、関係する定義ダイアローグボックスがポップアップします。問題を修正するため に設定を変更して、**ジョブ実行**を再度実行して下さい。

**ジョブ実行**の途中で、JobGen Plus はバイナリインデックスサーチ・インデックステーブルの 構築)または二バイト文字フォントファイルの構築などいくつかの他の操作を起動します。こ れらすべての動作はメッセージウインドウにレポートされます。

ジョブの実行形式が正しく生成された後で、JobGen Plus はジョブをポータブルターミナルに ダウンロードするために PTComm マネージャを起動します。ダウンロードは、すべてのルック アップファイル、バイナリサーチインデックステーブル、そして二バイトフォントファィルな どのすべての関連するファイルを含んでいます。

🥸 JobGen Plus – Output – C:¥JGPlus5¥assetmet.jep 📃				
<u>ファイル(E)</u> 編集(E) ノード(N) 表示(V) ビルド(B) ツール(T) ウインドウ(W) ヘルプ(H)				
😤 C:¥JGPlus5¥assetmet.jep				
Records Output - C:¥JGPlus5¥assetmgt.jpp				
Comment Build job Exclude Job: "C:¥JGPlus5¥assetmgt.jgp" Nodes, <u>? Error: link, Edit -&gt; Section, </u> #4: not defined Nodes, Record, ASSEIMGT: set as the default record				
Nodes, Record, ASSEIMGI: save file to the default folder: C:¥JGPlusb¥ASSEIMGI.IXT → Nodes, Done: 1 error(s), 0 warning(s) → Nodes, Error!				
Nodes, Nodes, ⊥Inks ⊥Inks,	-			
$\bullet$ Links, Build (Simulate ) Find1 ) Find2 /				
ヘルプは、F1 を押して下さい。 NUM	- //.			

## 第6章 ジョブの作成

実行形式がターゲットのポータブルターミナルに正しくダウンロードダウンロードした後で、 ユーザがポータブルターミナルで "RUN" コマンドの実行ができる状態となり、JobGen Plus のジョブアプリケーションを開始します。

# 第7章 ジョブのシュミレーション

ジョブが設計され、定義された後で、これはホスト PC でシュミレーションをすることができ ます。シュミレーションをスタートするにはツール > ジョブシュミレーションを選択するか、 あるいはツールバーのシュミレートアイコンをクリックします。

ジョブシュミレーションは実際のポータブルターミナルがなくてもジョブの確認をする容易な 方法を提供します。 シミュレーションはジョブで選択したポータブルターミナルのモデルに よく似たポータブルターミナルのウインドウを表示、そしてジョブと Windows の下で DOS セッ ションのジョブエンジンを実行します。これはすべての入力と出力をシュミレーションします。 すべての操作は実際のポータブルターミナルとほとんど同じです。

ジョブのシュミレーションにはある制限があります。ルックアップ検証におけるファイル検索 はファイルの最初から 64K バイトを越えることはできません。

ジョブシュミレーションはジョブの開発の助けになります。実際のポータブルターミナルにダ ウンロードすることによってすべてのジョブのテストを行い、そして各機能の実行をされるこ とを推奨いたします。これはジョブが正しいことを確認するただ一つの方法です。

PT-630			×
部署コ  F2:デ	-		
(Esc) ABC 1 #	Scan DEF 2 \$	Ent GHI 3	  x
JKL 4 : STU 7 -	MN0 5 = VWX 8 +	PQR 6 YZ_ 9	/
Cir ¥	0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0	;.	1
Alp < Pwr U F5 Fi F1 F	C > Dn PgUp P Dn 6 F7 2 F3	Spc Cmd F8 F4	

# 第8章 言語サポート

JobGen Plus はポータブルターミナルのスクリーンで複数の言語をサポートしています。イン ストール時に言語を選択して下さい。JobGen Plus はフォントのビットマップファイルなどの サポートファイルを自動的に生成し、またジョブを作成し、そしてジョブ実行ファイルと共に ポータブルターミナルにこれらをダウンロードします。

日本語の表示は 大型フォントのみをサポートしています。ジョブのプロパティパネルの、プロ パティの項: Portable >> Setting を開きます。スクリーンから大型フォントを選択します。

## 漢字サポート



JobGen Plus はWindows 日本語版のフォントライブラリから漢字のビットマップを探し、コードとビットマップの両方を含むファイルを作成します。このファイルはジョブの実行形式と一緒にポータブルターミナルにダウンロードされます。フォント・タイプ、スタイル、そしてサイズを選択することができます。ポータブルターミナルに表示される漢字のビットマップサイズは 16 x 16 ピクセルです。ビットマップの文字位置のレイアウトを調整することもできます。

漢字ファイルはジョブで共有することができます。複数のジョブが一つの漢字ファイルを使用 することができるので、システムのメモリを節約することができます。ジョブエンジンはジョ ブの漢字ファイル(jobname.ccb). を最初に開こうとします。これに失敗すると、標準の CCB ファイルを開こうとします(jeng.ccb)。