

DT-970SDK

アプリケーション開発ガイド

このガイドは DT-970 のアプリケーション開発者向けの開発ガイドブックです。
DT-970 SDK として提供する開発環境を利用した、アプリケーションプログラム開発の概要(プログラムの開発、ダウンロード等)を、開発の流れにしたがって説明します。



ご注意

- この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。
- このソフトウェアおよびマニュアルは、本製品の使用許諾契約書のもとでのみ使用することができます。
- 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、一切の責任を負いかねますのでご了承ください。
- 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。
- このマニュアルの著作権はカシオ計算機株式会社に帰属します。
- 本書中に含まれている画面表示は、実際の画面とは若干異なる場合があります。予めご了承ください。

© 2013-2014 カシオ計算機株式会社

Microsoft, MS, ActiveSync, Active Desktop, Outlook, Windows, Windows NT, および Windows ロゴは、米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。Microsoft 社の製品は、OEM 各社に、Microsoft Corporation の 100%出資子会社である Microsoft Licensing, Inc.によりライセンス供与されています。

目次

1.	はじめに	4
1.1.	概要	4
1.2.	全体像	4
1.3.	動作環境	6
1.4.	必須ソフトウェア	6
1.5.	必須ハードウェア	7
2.	アプリケーション開発方法	8
2.1.	Application 開発準備	9
2.1.1.	開発環境インストールおよびライセンス登録	9
2.1.2.	開発環境構築	9
2.2.	Application 開発	10
2.2.1.	新規に Application を開発する場合	10
2.2.2.	既存の Application を使用する場合	12
2.2.3.	本ツールで開発した Application を使用する場合	12
2.3.	ビルド/テスト	13
3.	ツール詳細機能	14
3.1.	機能一覧	14
3.2.	ファイル構成	14
3.2.1.	ciprj ファイル	15
3.3.	起動	16
3.4.	終了	16
3.5.	画面詳細	17
3.5.1.	ソースツリー領域	18
3.5.2.	プロパティ領域	19
3.5.3.	ビルド結果領域	22
3.6.	メニュー詳細	22
4.	Emulator 機能	26
4.1.	概要	26
4.2.	DT-970 エミュレータでの操作	27
4.2.1.	DT-970 エミュレータ	27
4.2.2.	キー入力機能	27
4.2.3.	表示機能	27
4.2.4.	バーコード入力機能	27
4.2.5.	サウンド機能	27
4.2.6.	ファイル操作	28
4.3.	I/O シミュレータの操作	33
4.3.1.	入力バーコードの登録	33
4.3.2.	ローバッテリー発生/IOBOX 接続動作	35
4.3.3.	LED/バイブレーション/バックライト状態通知	36
4.4.	Emulator での動作差異	37
4.4.1.	ドライブレター	37
4.4.2.	通信機能	37
4.4.3.	システムファイル	37
4.4.4.	副電池	37

4.4.5.	バックライト/コントラスト	37
4.4.6.	DT-930 互換表示モード	37
4.4.7.	ユーザフォント/外字	38
4.4.8.	関数インタフェース	39
5.	その他機能	47
5.1.	フォントコンバータ	47
5.1.1.	機能	49
5.1.2.	画面構成	51
5.1.3.	使用方法	55
5.1.4.	注意事項	64
6.	アプリ開発ステップ詳細	65
6.1.	目的	65
6.2.	全体の流れ	66
6.3.	RX C コンパイラ環境	67
6.4.	DT-970 開発環境	68
6.5.	動作環境設定	69
6.5.1.	PATH 設定	69
6.6.	コード/オブジェクト変換	70
7.	付録	72
7.1.	注意事項	72
7.2.	パッチ	75
7.3.	サンプルアプリケーション	75

1. はじめに

このたびは、CASIO DT-970 SDK をお買い上げありがとうございます。
本書は、DT-970 アプリケーション開発キットを使用して、アプリケーションの開発をするための手順を説明したものです。

1.1. 概要

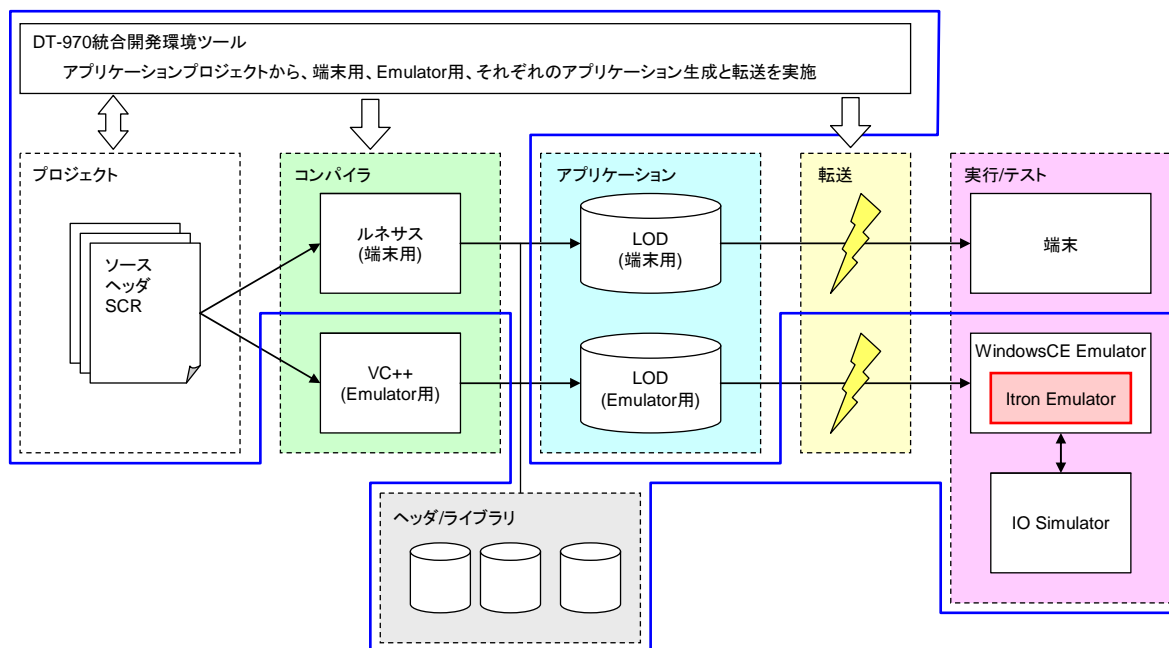
DT-970 アプリケーション開発環境は、DT-970 上で動作するアプリケーションのコンパイル／リンク／転送を Windows PC 上で行うものです。

上記に加えて、Windows CE Emulatorを用いて、Windows PC 上の DT-970 仮想デバイスでの Emulator 実行が可能です。この Emulator 実行には、Windows CE 開発環境が必要となります。

DT-970 アプリケーション開発環境では、これらの操作を、視覚的／直感的な操作で実現することで、コンパイラ等に関しての専門的な知識を必要とせずアプリケーション開発が可能です

1.2. 全体像

DT-970 アプリケーション開発環境の全体像を以下に示します。(青線内を本ツールで提供)



プロジェクト

ソースコード、ヘッダ、およびファイル転送に使用するスクリプトファイル(*.scr)を管理します。
DT-930 からの移行の場合は、上記ファイルを取り込みます。
※ ファイルはすべて Shift-JIS または US-ASCII で記載してください

コンパイラ

アプリケーションを端末で使用する場合はルネサスコンパイラを使用します。
また、アプリケーションを Emulator で使用する場合は VC++コンパイラを使用します。

ライブラリ

DT-970 用ライブラリをリンクします。
使用可能なライブラリの詳細については下記のマニュアルを参照してください。

- デバイス制御ライブラリ リファレンスマニュアル
- 拡張機能ライブラリ リファレンスマニュアル

アプリケーション

アプリケーションを端末で使用する場合は CASIO 独自仕様の LOD ファイルを作成します。
また、アプリケーションを Emulator で使用する場合は DLL を LOD にリネームして使用します。

転送

アプリケーションを端末で使用する場合は LMWIN を使用します。
また、アプリケーションを Emulator で使用する場合は ActiveSync または Windows Mobile デバイスセンタを使用します。

実行/テスト

端末または Emulator でアプリケーションの動作確認を行います。

1.3. 動作環境

ハードウェア

- IBM PC/AT 互換機 (画面解像度 XGA [1024x768]以上)

OS

- Microsoft Windows XP SP3 (x86)
- Microsoft Windows Server 2003 SP2 (x86)
- Microsoft Windows 7 Professional SP1 (x86/x64)
- Microsoft Windows Server 2008 SP2 (x86)
- Microsoft Windows Server 2008 R2 SP1 (x64)

ロケール

- 日本語
- 英語

1.4. 必須ソフトウェア

以下のソフトウェアが必須です。

- Microsoft .NET Framework 3.5 もしくは 3.5.1
- LMWIN

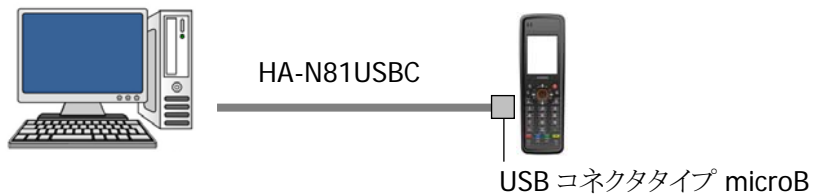
また、Emulator を使用する場合は、以下のソフトウェアが必須です。

- Microsoft Visual Studio 2008 Professional
- Microsoft Device Emulator 3.0 (Visual Studio 2008 には同梱)
- Microsoft ActiveSync または Microsoft Windows Mobile デバイスセンター

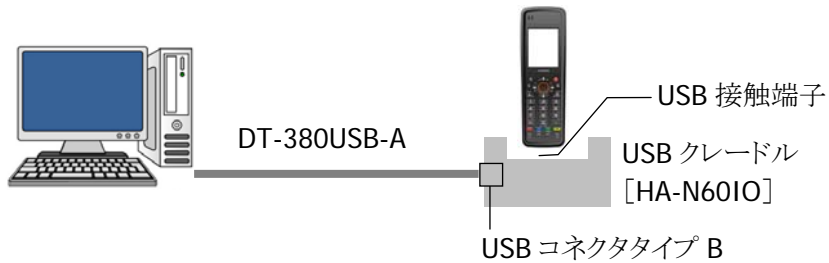
1.5. 必須ハードウェア

DT-970 アプリケーション開発環境で生成したオブジェクトは、LMWIN で DT-970 に転送します。
PC と DT-970 は、以下の何れかの形態で接続して下さい。
推奨する接続形態は、1) 2) 3) です。

1) USB ケーブル[HA-N81USBC]で直結

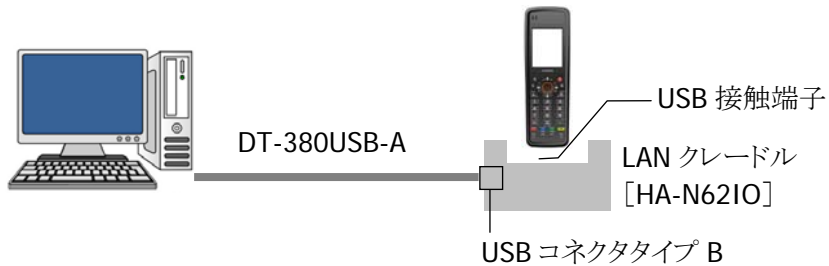


2) USB クレードル[HA-N60IO]で接続

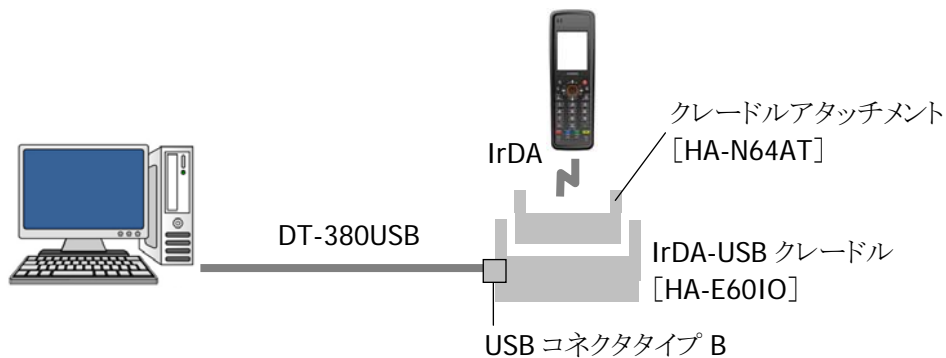


3) LAN クレードル[HA-N62IO]で、USB 接続

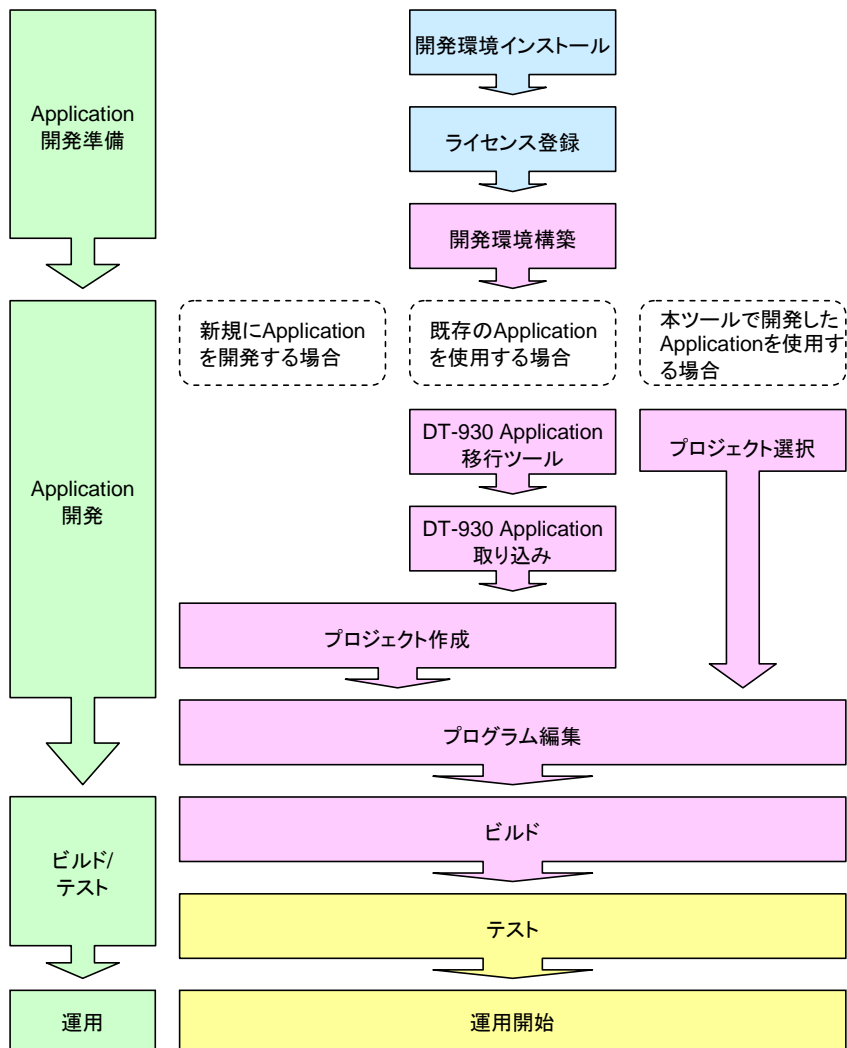
開発環境としては、LAN クレードル利用 LAN 接続でのアプリケーション転送は対応しません。



4) IrDA-USB クレードル[HA-E60IO]、クレードルアタッチメント[HA-N64AT]で接続



2. アプリケーション開発方法



■は、本ツールのインストール CD からの操作になります。

■は、本ツール上で行う操作になります。

■は、端末または Emulator 上で行う操作になります。

2.1. Application開発準備

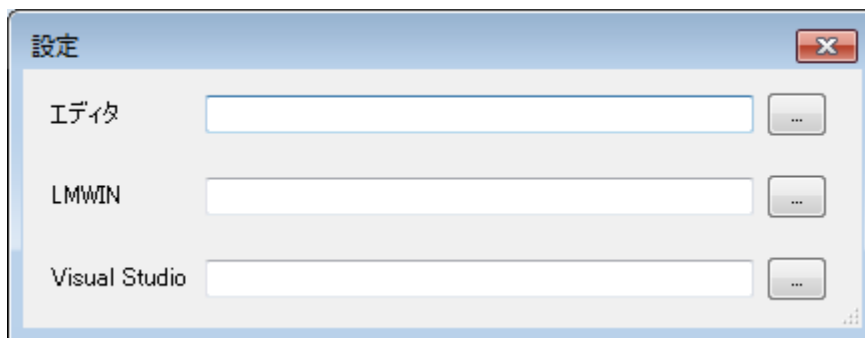
本ツールのインストールや初期設定を行います。

2.1.1. 開発環境インストールおよびライセンス登録

ファーストステップガイドを参照してください。

2.1.2. 開発環境構築

統合開発環境ツールを起動し※、メニューから「ツール」>「設定」を選択してください。
ソースコード等の編集に使用するエディタ、端末へのファイルの転送に使用する LMWIN、および
Emulator 用ビルドに使用する VisualStudio のパス設定を使用します。



エディタが空欄の場合は、メモ帳(Notepad)でソースコードの編集を行います。
VisualStudio2008 がインストールされていない場合は、空欄です。

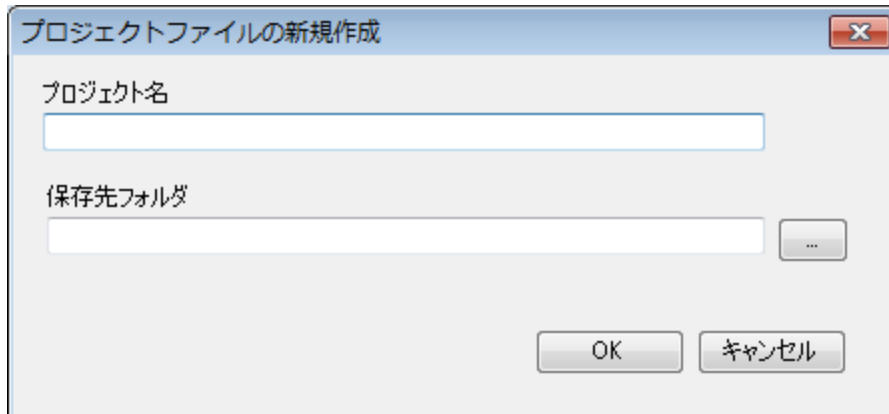
※ 統合開発環境ツールの起動については、「3.3 起動」を参照してください。

2.2. Application開発

プロジェクトの作成やソースコードの編集等を行います。

2.2.1. 新規にApplicationを開発する場合

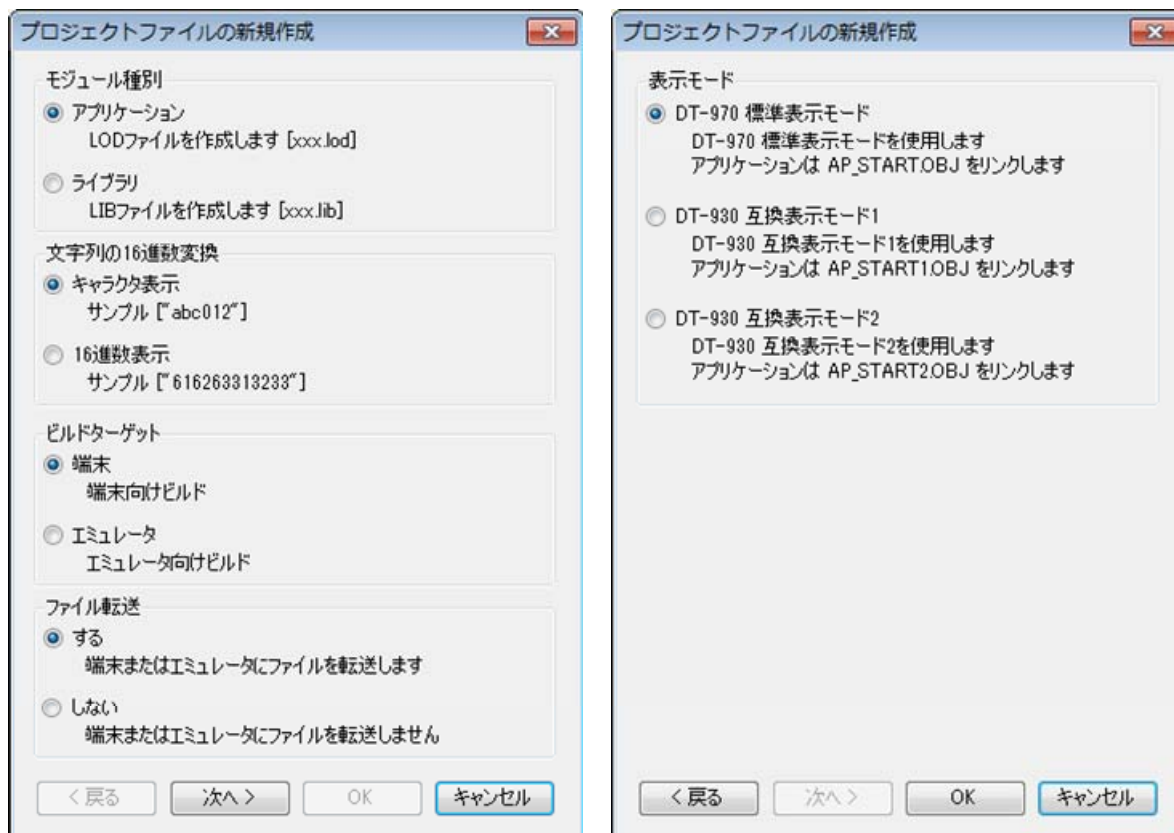
メニューの「ファイル」>「プロジェクトの新規作成」を選択すると、以下のダイアログを表示します。



保存場所 [保存先フォルダ]¥[プロジェクト名]
プロジェクトファイル [保存先フォルダ]¥[プロジェクト名]¥[プロジェクト名].ciprj

初回起動時は、プロジェクト名、保存先フォルダともに空欄です。
2回目以降の場合は、保存先フォルダは前回作成時のパスをデフォルトとして使用します。
また、指定した保存先フォルダ内に指定したプロジェクト名と同じ名称のフォルダが存在する場合は、エラーメッセージを表示します。

次に下記ダイアログが表示されますのでプロジェクトの内容を指定して下さい。
「モジュール種別」以外の項目は、プロパティ領域から変更することができます。

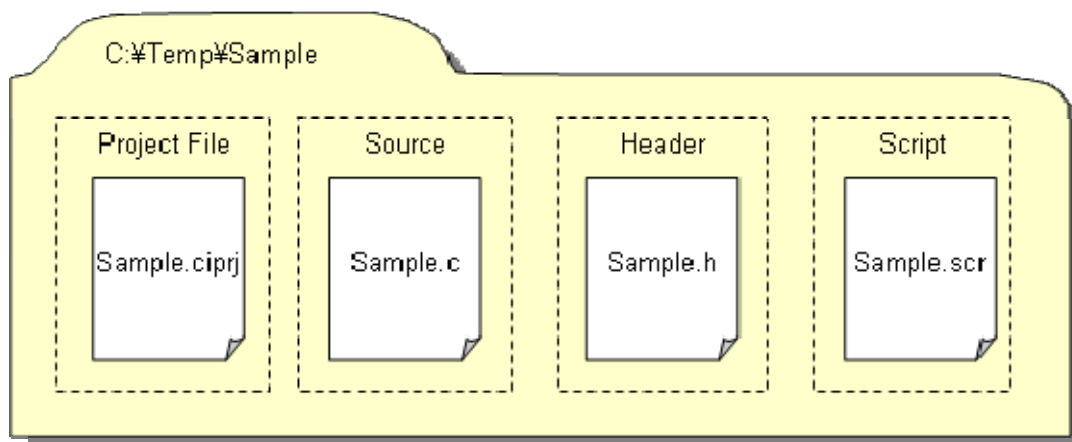


生成ファイルは、プロジェクト名に対して、モジュール種別毎の拡張子を付与したものととなります。

アプリケーション [プロジェクト名].lod

ライブラリ [プロジェクト名].lib

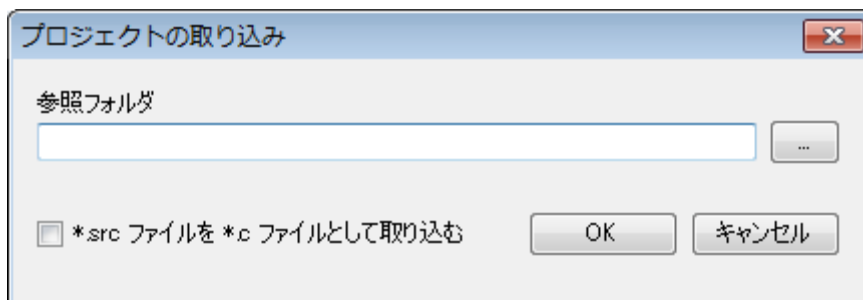
例:C:¥Tempに Sample を作成



2.2.2. 既存のApplicationを使用する場合

既存の DT-930 の Application を使用する場合は、以下の手順で本ツールに取り込むことができます。

1. メニューの「ツール」>「DT-930 アプリケーション移行」を選択し、既存のソースコードを DT-970 用に変換します。
(詳細は DT-900/930/940 アプリケーション移行ガイドを参照してください)
2. メニューの「ファイル」>「DT-930 プロジェクトを取り込む」を選択し、参照フォルダに手順 1. で変換したソースコード等を格納したフォルダを指定してください。

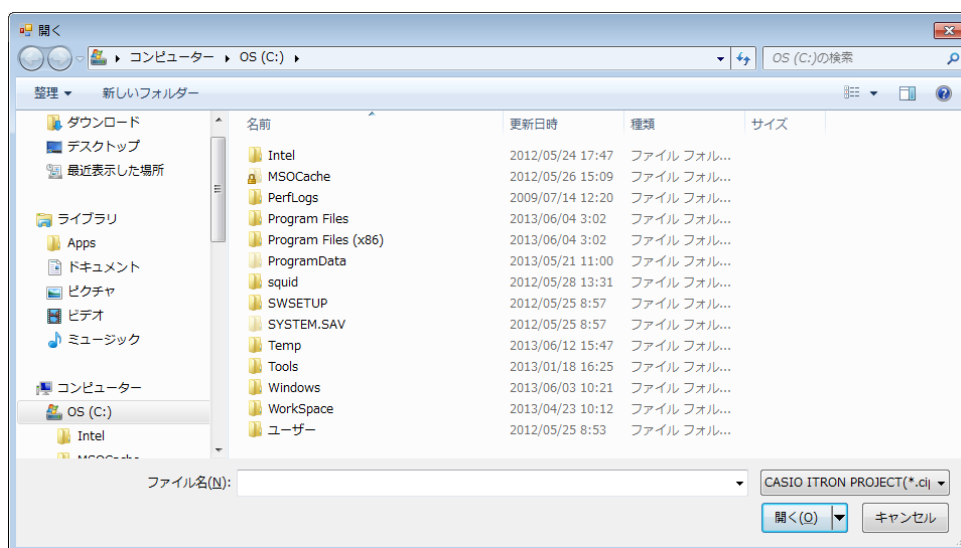


3. ファイルを取り込んだ後は、「2.2.1新規にApplicationを開発する場合」と同様の手順でプロジェクトの設定を行なってください。

2.2.3. 本ツールで開発したApplicationを使用する場合

メニューの「ファイル」>「プロジェクトを開く」を選択すると、以下のダイアログを表示します。

ciprj ファイルを指定すると、そのファイルの情報をもとに、ソースツリー領域およびプロパティ領域を設定します。ファイルのフォーマットなどが異なる場合は、エラーメッセージを表示します。
(デフォルトパス:C:¥、デフォルトファイル名 :なし)



2.3. ビルド／テスト

開発した **Application** を端末上または **Emulator** 上で動作させるためにビルドを行います。メニューの「ビルド」>「端末ビルド」を選択すると端末上で動作する **Application** が、また「ビルド」>「エミュレータビルド」を選択すると **Emulator** 上で動作する **Application** がそれぞれ生成されます。これらの動作テストを行う手順の詳細は、「4.Emulator機能」を参照してください。

3. ツール詳細機能

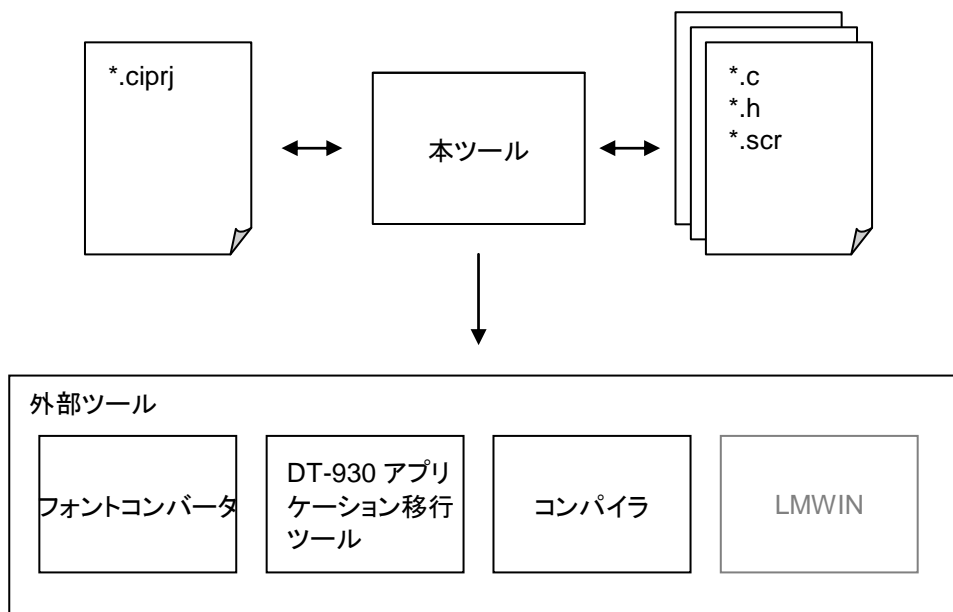
3.1. 機能一覧

本ツールの機能一覧を以下に示します。

No	名称	説明
1	各種ファイル編集	登録したエディタを呼び出し、ソース等の編集を行います。 (デフォルトは Notepad.exe で表示します)
2	ビルド	端末/Emulator 用のアプリケーションをビルドします。
3	ファイル転送	端末/Emulator に対してファイルを転送します。
4	ツール起動	フォントの編集や DT-930 アプリケーションの移行を行います。

3.2. ファイル構成

本ツールのファイル構成を以下に示します。



No	名称	備考
1	ciprj ファイル	プロジェクトの情報を格納します。 (詳細は次節を参照してください)
2	c、h、scr ファイル	
3	フォントコンバータ	DT-930 提供ツールと同等です。
4	DT-930 アプリケーション移行ツール	既存の DT-930 用のソースコードを DT-970 用に移行します。
5	コンパイラ	

3.2.1. ciprjファイル

本ツールで使用するプロジェクト情報を格納します。
ここに保存する情報の詳細については、「3.5.1ソースツリー領域」および「3.5.2プロパティ領域」を参照してください。

```
<?xml version="1.0" encoding="utf-8"?>
<ItronProject xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Base>
    <Version></Version>
    <Path></Path>
    <Name></Name>
  </Base>
  <SourceArray>
    <SourceItem FileName=" " />
  </SourceArray>
  <HeaderArray>
    <HeaderItem FileName=" " />
  </HeaderArray>
  <ScriptArray>
    <ScriptItem FileName=" " />
  </ScriptArray>
  <Setting>
    <Module></Module>
    <Hexa></Hexa>
    <Target></Target>
    <Transfer></Transfer>
  </Setting>
  <Option>
    <HT>
      <Output> </Output>
      <Library> </Library>
      <IncPath> </IncPath>
      <LibPath> </LibPath>
      <Prepro />
    </HT>
    <Emulator>
      <Output> </Output>
      <Library> </Library>
      <IncPath> </IncPath>
      <LibPath> </LibPath>
      <Prepro />
    </Emulator>
  </Option>
</ItronProject>
```

3.3. 起動

本ツールの起動は以下の 5 点です。

1. エクスプローラから IAppBuilder.exe を実行
2. エクスプローラから ciprj ファイルを実行
3. エクスプローラ上で ciprj ファイルを IAppBuilder.exe にドラッグ&ドロップ
4. スタートメニューから「CASIO DT-970 IDE」>「IAppBuilder」を選択
5. Dos から IAppBuilder.exe を実行 (詳細は下記を参照)

```
start IAppBuilder.exe [xxx.ciprj]
[]内は省略可能
```

引数なしで本ツールを起動した場合は、何も設定していない状態で起動します。

ciprj ファイルを引数として指定して本ツールを起動した場合は、ciprj ファイルに格納した情報をもとに各項目を設定した状態で起動します。

3.4. 終了

画面右上の×ボタンを押下すると、本ツールを終了します。

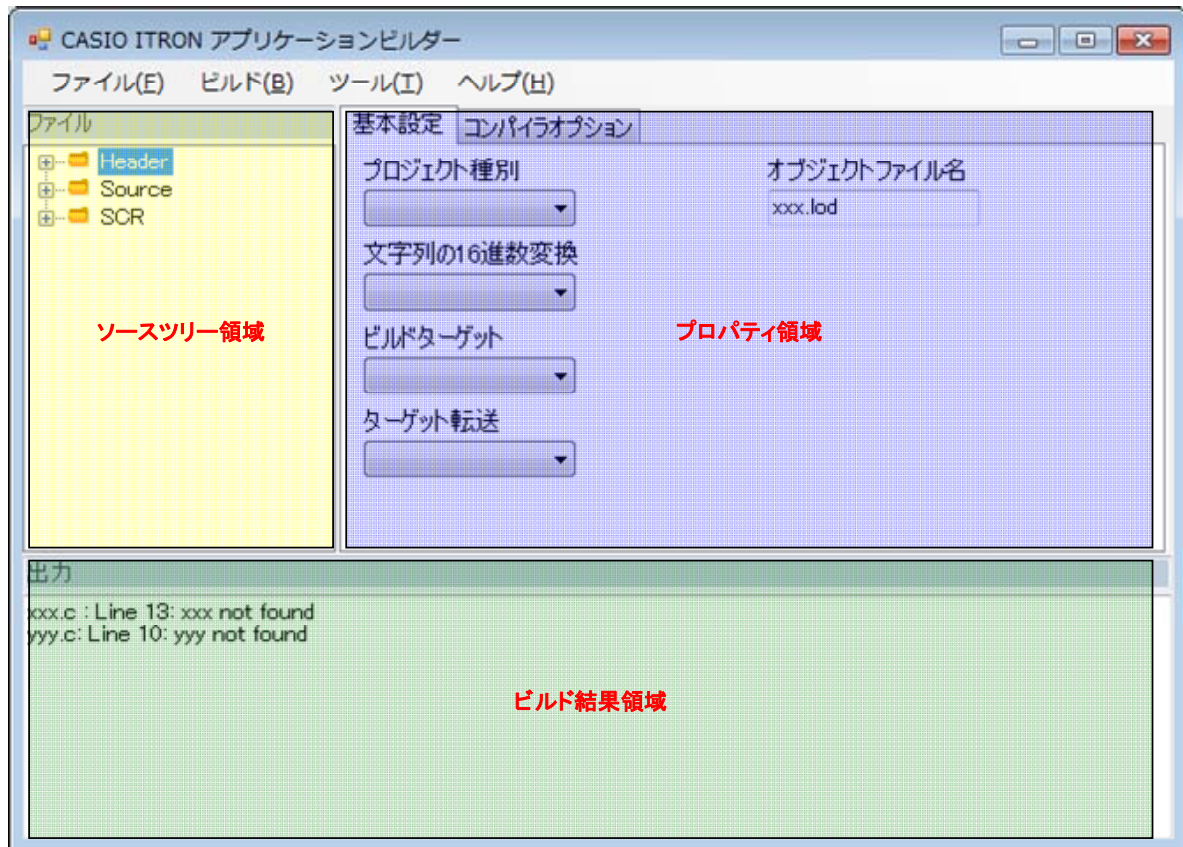
また、メニューの「ファイル」>「終了」を選択すると、本ツールを終了します。

プロジェクトを修正し、その情報を保存していない場合は、上書き保存を行うかどうかをメッセージボックスで表示します。

3.5. 画面詳細

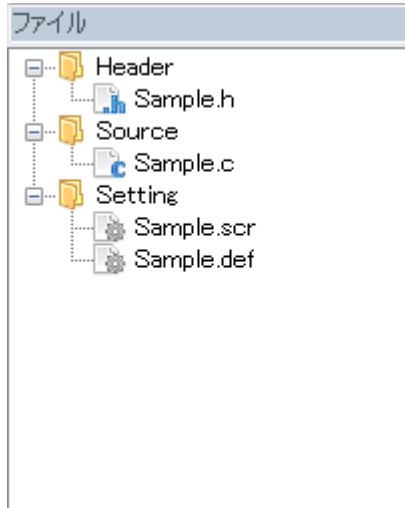
本ツールの画面構成を以下に示します。

本ツールはソースツリー領域、プロパティ領域、ビルド結果領域の 3 つの領域で構成されています。

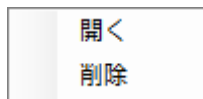


3.5.1. ソースツリー領域

ciprj ファイルに保存してあるソース、ヘッダ、SCR ファイルをそれぞれツリービューで表示し、各ファイルの編集を行います。



ソースファイル等を編集する場合は、対象のファイル名をダブルクリックすることにより、登録したエディタでその内容を表示します。エディタの登録については、「3.6メニュー詳細」を参照してください。また、ソースファイル等の各ファイル上にて右クリックをすると、以下のコンテキストメニューを表示します。



ここで、削除を選択すると、対象ファイルをプロジェクト上から削除します。
(データの削除はしません)

3.5.2. プロパティ領域

コンパイラ等の各種設定を行います。
コンパイラの初期設定は下記のとおりです。

No	項目	内容
1	CPU 選択	-cpu=rx600
2	エンディアン選択	-endian=big
3	ビットフィールド	-bit_order=left
4	符号指定無ビットフィールド型	-signed_bitfield
5	符号指定無 char 型	-signed_char
6	double 型サイズ	-dbl_size=8
7	コンパイルエラーレベル	-change_message=error=20223

これらの設定は、下記の場所にある"IAppBuilderRXC.ini"に定義されています。必要に応じて修正してください。

Windows XP / Windows Server 2003

C:\Documents and Settings\All Users\Application Data\CASIO\DT970\IDE\IAppBuilderRXC.ini

Windows 7 / Windows Server 2008

C:\ProgramData\CASIO\DT970\IDE\IAppBuilderRXC.ini

```
[RXC]
CPU="-cpu=rx600"
Endian="-endian=big"
Bit="-bit_order=left"
SBit="-signed_bitfield"
SChar="-signed_char"
Double="-dbl_size=8"
Prototype=-change_message=error=20223
```

基本設定

基本設定	端末設定	エミュレータ設定
プロジェクト種別	オブジェクトファイル名	
<input type="text"/>	<input type="text"/>	
文字列の16進数変換		
<input type="text"/>		
ビルドターゲット		
<input type="text"/>		
ターゲット転送		
<input type="text"/>		

No	名称	説明
1	プロジェクト種別	アプリケーション(デフォルト) ライブラリ プロジェクトの新規作成時にのみ選択できます
2	オブジェクトファイル名	プロジェクト名.lod またはプロジェクト名.lib プロジェクトの新規作成時にのみ選択できます オブジェクトファイル名を変更する場合は、コンパイラオプションの出力ファイル名を修正してください
3	文字列の16進数変換	する(デフォルト) しない
4	ビルドターゲット	端末(デフォルト) エミュレータ
5	ターゲット転送	する(デフォルト) しない

端末設定／エミュレータ設定

基本設定	端末設定	エミュレータ設定
出力ファイル名		
<input type="text" value="C:\Documents and Settings\CASIO\Sample\ReleaseHT\Sample.lod"/>		
オブジェクト/ライブラリモジュール		
<input and="" c:\documents="" settings\casio\sample\object\casio.obj\""="" type="text" value="\"/>		
追加インクルードのパス		
<input and="" c:\documents="" settings\casio\sample\include\""="" type="text" value="\"/>		
追加ライブラリのパス		
<input and="" c:\documents="" settings\casio\sample\lib\casio.lib\""="" type="text" value="\"/>		
プリプロセッサの定義		
<input type="text" value="DEBUG SAMPLE_HT"/>		

基本設定	端末設定	エミュレータ設定
出力ファイル名		
<input type="text" value="C:\Documents and Settings\CASIO\Sample\ReleaseEmulator\Sample.lod"/>		
オブジェクト/ライブラリモジュール		
<input type="text" value="commctrl.lib itapi970.lib"/>		
追加インクルードのパス		
<input and="" c:\documents="" settings\casio\sample\include\""="" type="text" value="\"/>		
追加ライブラリのパス		
<input and="" c:\documents="" settings\casio\sample\lib\""="" type="text" value="\"/>		
プリプロセッサの定義		
<input type="text" value="DEBUG SAMPLE_EMU"/>		

No	名称	説明
1	出力ファイル名	<p>デフォルトは下記のとおりです。</p> <p>端末: プロジェクトフォルダ¥ReleaseHT¥xxx</p> <p>エミュレータ: プロジェクトフォルダ¥ReleaseEmulator¥xxx</p> <p>(xxx: プロジェクト名.lod またはプロジェクト名.lib)</p>
2	オブジェクト/ ライブラリモジュール	<p>任意</p> <p>デフォルトは下記のとおりです。(環境により異なります)</p> <p>端末: "C:¥Program Files (x86)¥Windows CE Tools¥ wce500¥DT-970¥Object¥Rx600¥AP_START.OBJ"</p> <p>エミュレータ: commctrl.lib coredll.lib corelibc.lib ole32.lib uuid.lib itfw970.lib itapi970.lib</p> <p>※ 複数指定時は、半角スペース区切りです</p> <p>※ パスに半角スペースが含まれる場合は" "で区切ってください</p>
3	追加インクルードのパス	<p>任意</p> <p>デフォルトは下記のとおりです。(環境により異なります)</p> <p>端末: "C:¥Program Files (x86)¥Windows CE Tools¥ wce500¥DT-970¥Include¥Rx600"</p> <p>エミュレータ: "C:¥Program Files (x86)¥Windows CE Tools¥ wce500¥DT-970¥Include¥Armv4i"</p> <p>※ 複数指定時は、半角スペース区切りです</p> <p>※ パスに半角スペースが含まれる場合は" "で区切ってください</p>
4	追加ライブラリのパス	<p>任意</p> <p>デフォルトは下記のとおりです。(環境により異なります)</p> <p>端末: "C:¥Program Files (x86)¥Windows CE Tools¥ wce500¥DT-970¥Lib¥Rx600¥RXCLIB.LIB"</p> <p>エミュレータ: "C:¥Program Files (x86)¥Windows CE Tools¥ wce500¥DT-970¥Lib¥Armv4i"</p> <p>※ 複数指定時は、半角スペース区切りです</p> <p>※ パスに半角スペースが含まれる場合は" "で区切ってください</p> <p>※ 端末には Lib のフルパス、エミュレータには Lib を格納しているフォルダをそれぞれ指定してください</p>
5	プリプロセッサの定義	<p>任意</p> <p>※ 複数指定時は、半角スペース区切りです</p>

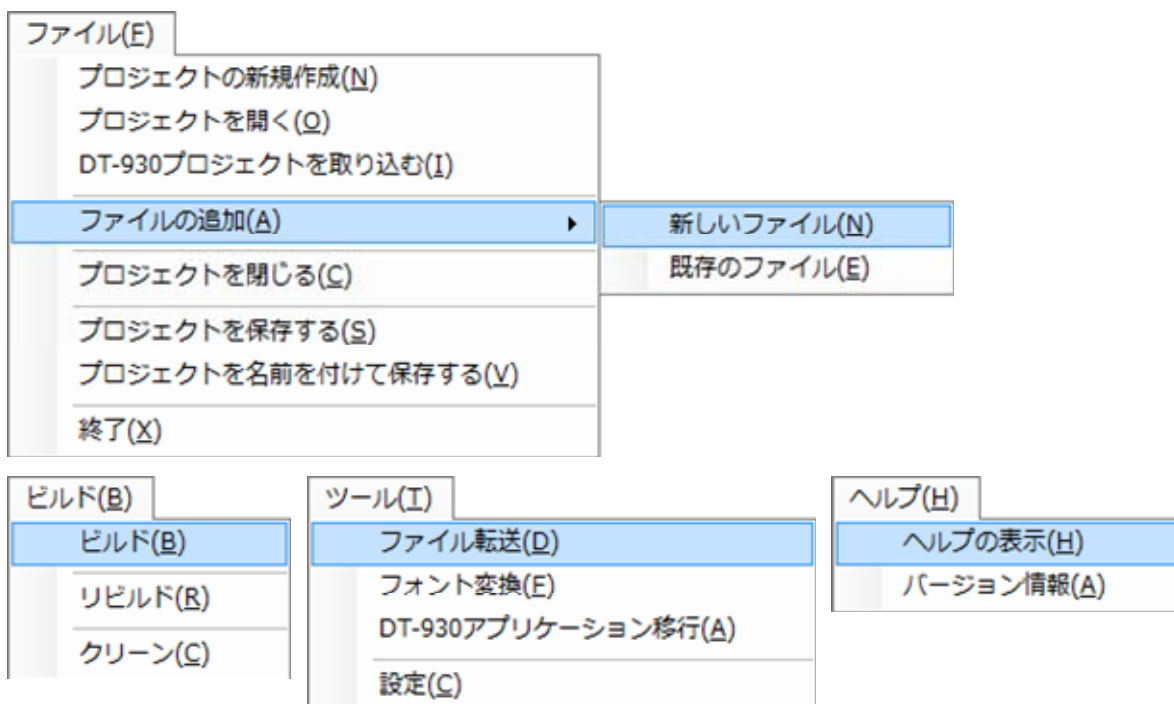
3.5.3. ビルド結果領域

ビルドの結果を表示します。

```
出力
コンパイル中 ...
RX Family C/C++ Compiler V2.00.00 [18 Mar 2013]
RX Family Assembler V2.00.00 [15 Feb 2013]
Renesas Optimizing Linker W1.00.00 [12 Dec 2012]
Copyright (C) 2011, 2013 Renesas Electronics Corporation
Copyright (C) 2003-2012 University of Illinois at Urbana-Champaign.
All rights reserved.
Sample.c:
C:\Users#uesaka\Desktop#Sample#Sample.h(7):F0520005:Could not open source file "itron.h"
```

ビルドエラーが発生した場合は、ファイル名、行数、およびその内容を表示します。

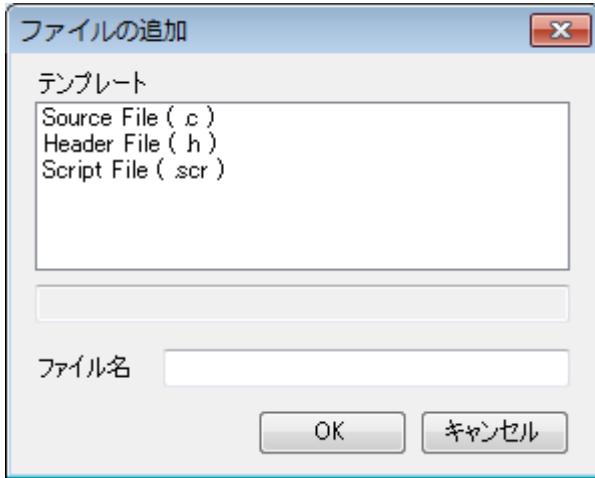
3.6. メニュー詳細



No	項目	説明
	ファイル	
1	プロジェクトの新規作成	新規プロジェクト作成ダイアログを表示します。
2	プロジェクトを開く	プロジェクトファイル選択ダイアログを表示します。
3	DT-930 プロジェクトを取り込む	DT-930 のソースコード等を格納したフォルダ選択ダイアログを表示します。
	ファイルの追加	
4	新しいファイル	新しいファイルをプロジェクトに追加します。
5	既存のファイル	既存のファイルをプロジェクトにコピーします。 (既存ファイルはそのままです)
6	プロジェクトを閉じる	本ツールを初期化します。
7	プロジェクトを保存する	編集中のプロジェクトを上書き保存します。
8	プロジェクトを名前をつけて保存する	ファイル保存ダイアログを表示します。
9	終了	本ツールを終了します。 プロジェクトファイルに変更がある場合は、ファイル保存ダイアログを表示します。
	ビルド	
10	ビルド	ビルドターゲットで設定した対象に対し、ビルドを実行します。
11	リビルド	ビルドターゲットで設定した対象に対し、リビルドを実行します。
12	クリーン	ビルドターゲットで設定した対象に対し、中間ファイルおよび生成物を削除します。
	ツール	
13	ファイル転送	作成したファイルの転送を行います。
14	フォント変換	フォントコンバータを起動します。
15	DT-930 アプリケーション移行	DT-930 アプリケーション移行ツールを起動します。
16	設定	本ツールの各種設定を行います。
	ヘルプ	
17	ヘルプの表示	マニュアルのあるフォルダを開きます。
18	バージョン情報	バージョン情報ダイアログを表示します。

ファイルの追加>新しいファイル

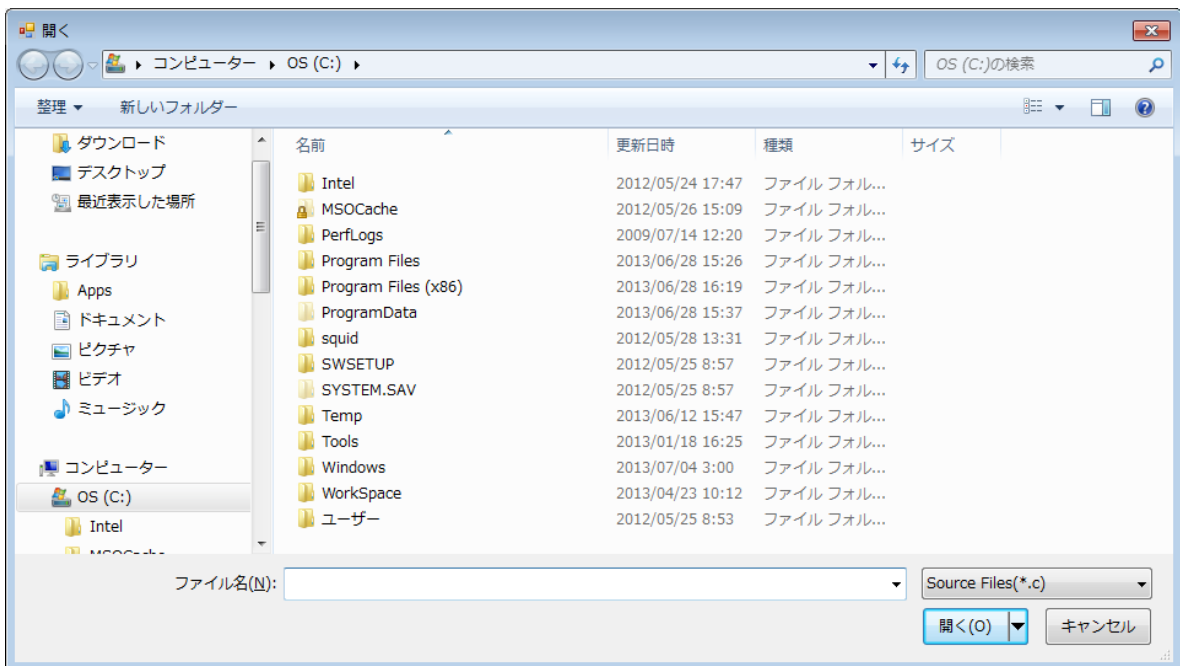
指定したファイルをプロジェクトファイルが存在するフォルダに新規作成します。



テンプレートから追加するファイルの種類を選択し、ファイル名を入力してください。

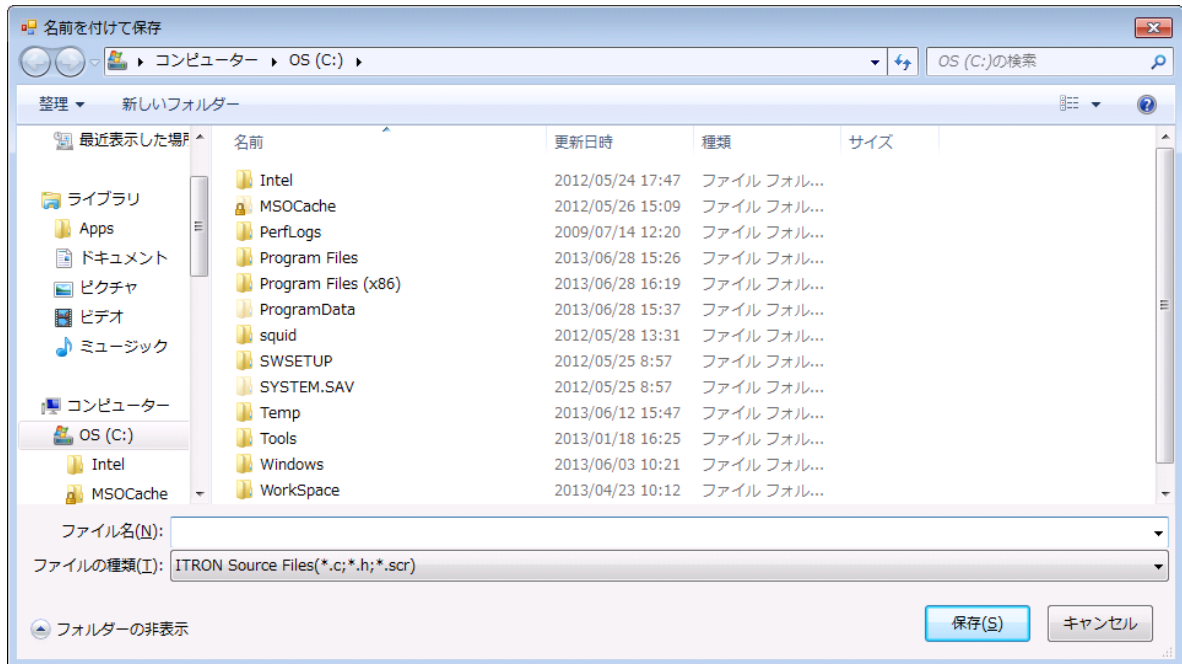
ファイルの追加>既存のファイル

指定したファイルをプロジェクトファイルが存在するフォルダにコピーします。
複数ファイルの選択が可能です。



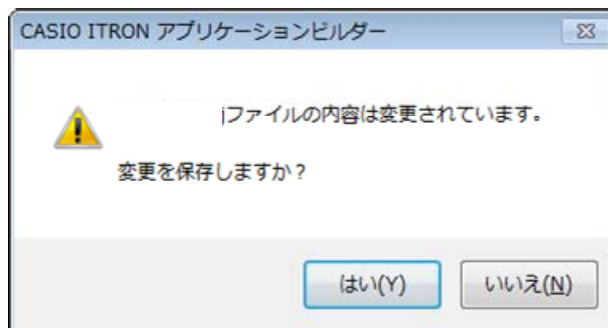
プロジェクトを名前をつけて保存

ファイル保存ダイアログを表示します。



終了

プロジェクトファイル編集後に保存をせずに終了する場合は、以下のメッセージを表示します。

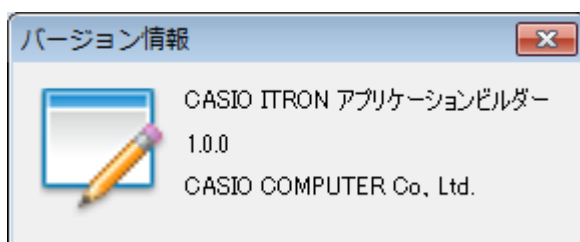


「はい」を選択した場合は、プロジェクトファイルを上書き保存します。

「いいえ」を選択した場合は、プロジェクトファイルを保存せずに本ツールを終了します。

バージョン情報

本ツールのバージョン情報を表示します。



4. Emulator機能

4.1. 概要

本 Emulator 機能は DT-970 エミュレータと I/O シミュレータにより構成されます。

DT-970 エミュレータ

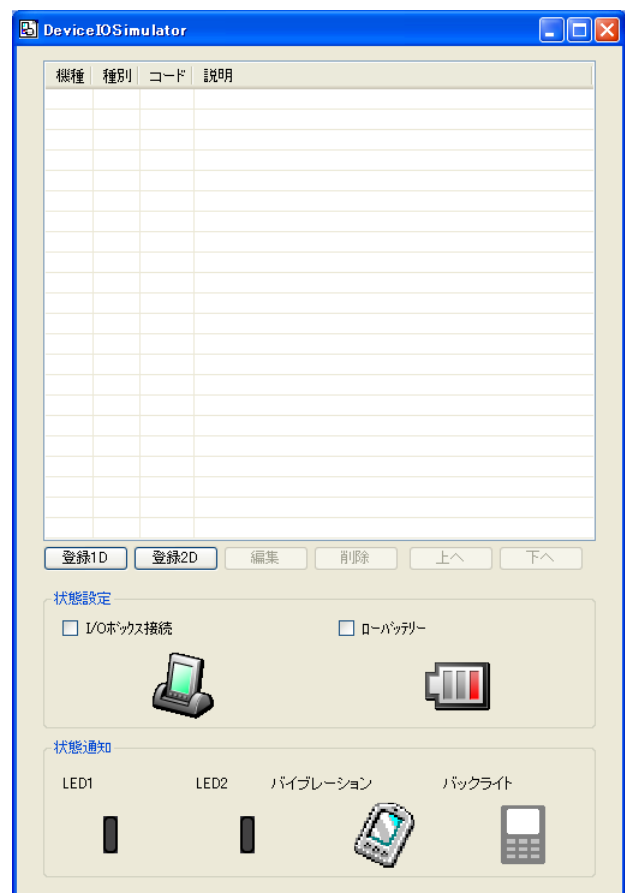
マウスや PC のキーボードによるキー入力や、実行画面の表示等、DT-970 実機上の動作を再現します。



I/O シミュレータ

バーコード入力・ローバッテリー発生・IOBOX 装着の動作を擬似的に行うものです。

また、LED・バイブレーション・バックライトの動作状態を表示します。



4.2. DT-970 エミュレータでの操作

4.2.1. DT-970 エミュレータ

マウスや PC のキーボードによるキー入力や、実行画面の表示等、DT-970 実機上の動作を再現します。



4.2.2. キー入力機能

DT-970 実機と同等のキー入力機能を搭載しています。
画面上のキーをマウスでクリックすることにより入力が可能です。
PC のキーボードからの入力も可能です。

4.2.3. 表示機能

DT-970 実機と同等のフォントイメージを搭載し、実機同等の表示イメージを再現します。

4.2.4. バーコード入力機能

I/O シミュレータに登録されたバーコードデータを、トリガキーの押下等、バーコードの入力操作を行うことにより、指定したバーコードデータを入力することが可能です。

DT-970 エミュレータのトリガキーを押下(マウスクリック)する場合は、必ず 1 秒以上押し続けてください。
押下(マウスクリック)する時間が短いと、正しく入力できない場合があります。

4.2.5. サウンド機能

DT-970 実機と同等の BEEP/SOUND 機能を搭載しています。

4.2.6. ファイル操作

エミュレータ上でファイルの操作(コピーや削除等)を行う場合は、Windows XP/Windows Server 2003 では ActiveSync、Windows Server 2008/Windows 7 では Windows Mobile デバイスセンター(バージョン 6.1 以降)を使用してください。使用手順は以下のとおりです。

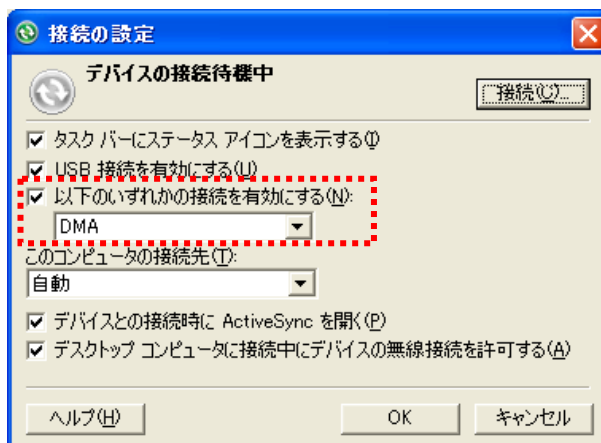
ActiveSync の使用手順

(1) ActiveSync の設定

タスクトレイ内の ActiveSync のアイコンを右クリックし、ポップアップメニューから「接続の設定」を選択します。

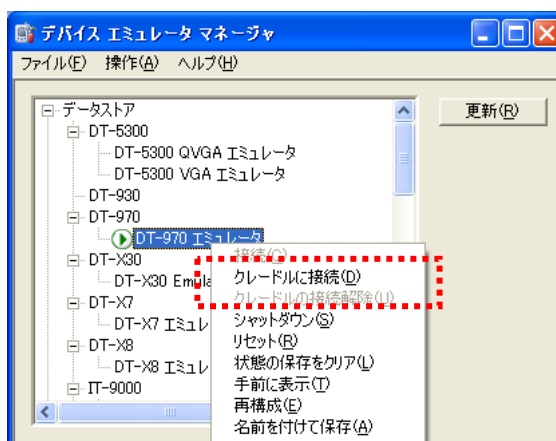


接続の設定ダイアログにおいて、DMA 接続を有効にします。

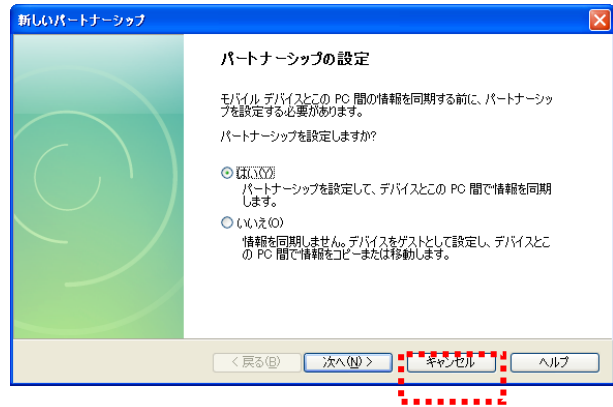


(2) ActiveSync の接続

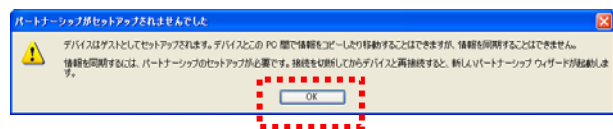
デバイスエミュレータマネージャを起動します。リストから DT-970 エミュレータ上で右クリックし、「クレードルに接続」を選択します。



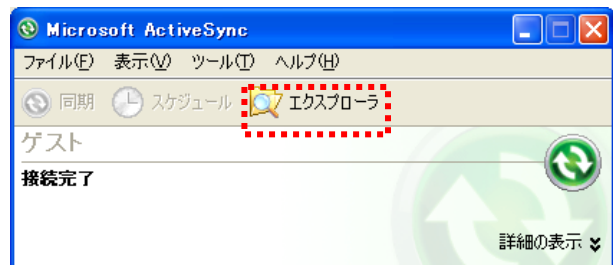
右記の画面が表示されるので、「キャンセル」をクリックします。



続いて右記の画面が表示されるので、「OK」をクリックします。



続いて右記の画面が表示されるので、「エクスプローラ」ボタンをクリックします。



「モバイル デバイス」ウィンドウにより、エミュレータ上のフォルダとファイルが表示されます。



(3) ファイルのコピー／削除

ファイルのコピーや削除は、「モバイル デバイス」ウィンドウ上で行います。コピーや削除の手順は、Windows XPなどに搭載されているファイルエクスプローラと同じ手順になります。なお、エミュレータとDT-970 実機では、ドライブレターに差異があるため、ご注意ください。ドライブレターの差異については、「4.4.1ドライブレター」を参照してください。

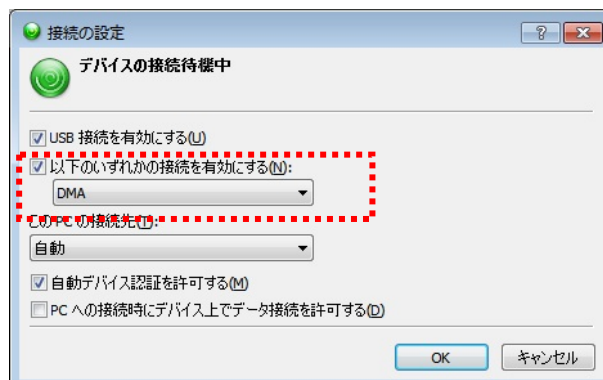
Windows Mobile デバイスセンターの使用手順

(1) Windows Mobile デバイスセンターの設定

スタートメニューから「Windows Mobile デバイスセンター」をクリックします。右記の画面が表示されるので、「接続の設定」をクリックします。

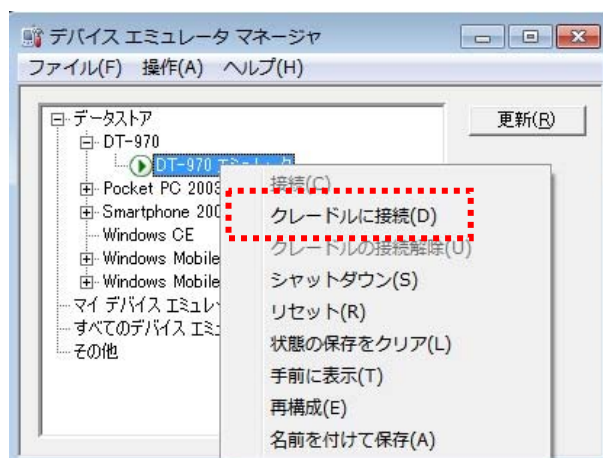


接続の設定ダイアログにおいて、DMA 接続を有効にします。



(2) Windows Mobile デバイスセンターの接続

デバイスエミュレータマネージャを起動します。リストから DT-970 エミュレータ上で右クリックし、「クレードルに接続」を選択します。



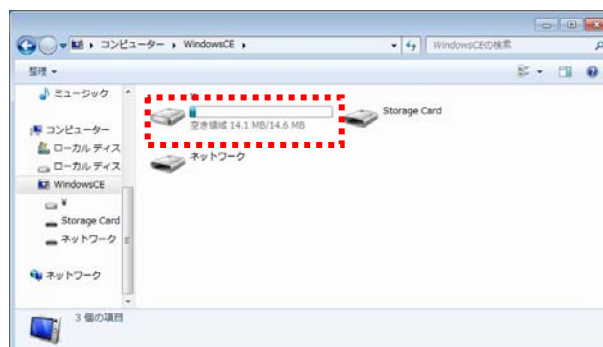
右記の画面が表示されるので、「デバイスを
セットアップしないで接続」をクリックします。



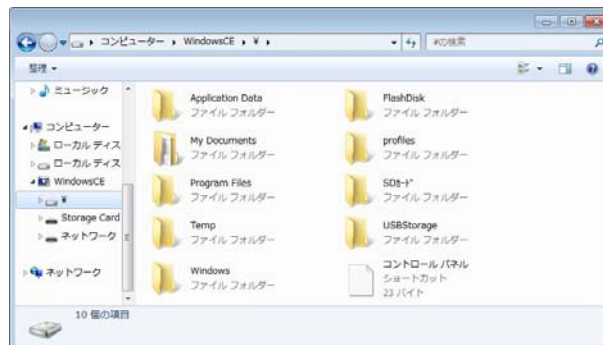
続いて右記の画面が表示されるので、「ファ
イル管理」>「デバイスのコンテンツの参照」
をクリックします。



続いて右記の画面が表示されるので、「▼」の
アイコンをダブルクリックします。



エミュレータ上のフォルダとファイルが表示さ
れます。



(3) ファイルのコピー／削除

ファイルのコピーや削除は、「ファイルエクスプローラ」ウィンドウ上で行います。コピーや削除の手順は、Windows 7などに搭載されているファイルエクスプローラと同じ手順になります。なお、エミュレータとDT-970実機では、ドライブレターに差異があるため、ご注意ください。ドライブレターの差異については、「4.4.1ドライブレター」を参照してください。

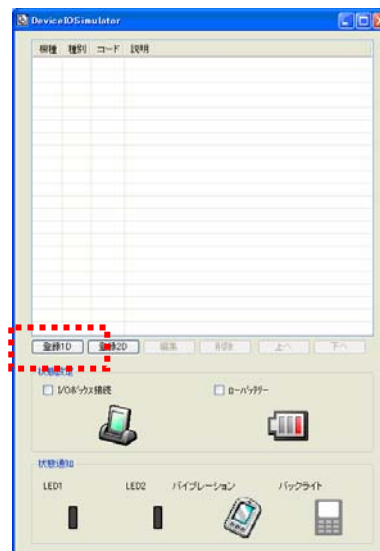
4.3. I/Oシミュレータの操作

入力バーコードの登録・ローバッテリー発生・IOBOX 装着動作を擬似的に行うものです。
また、LED・バイブレーション・バックライトの動作状態を表示します。

4.3.1. 入力バーコードの登録

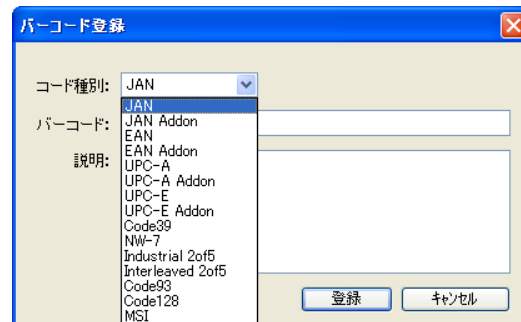
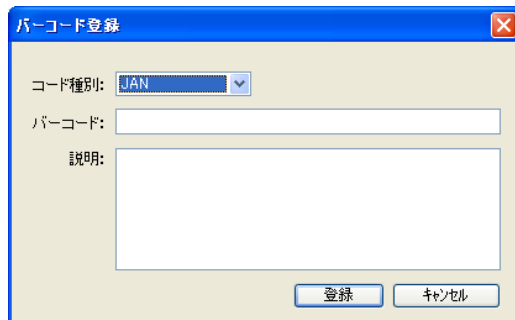
(1) 登録選択

I/O シミュレータの「登録」ボタンをクリックすると、バーコード登録画面に移ります。

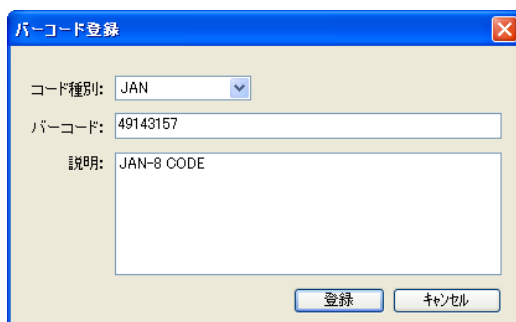


(2) バーコード登録

「コード種別」を選択し、任意のバーコード種別を選択します

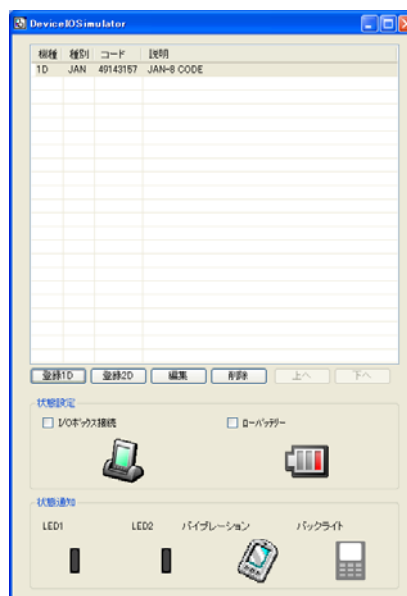


(3) バーコードのデータと説明を登録



(4) 登録完了

登録が完了すると、登録したバーコードが一覧表に表示します。この登録操作を繰り返して、デバッグに必要なバーコードを予め登録しておいてください。

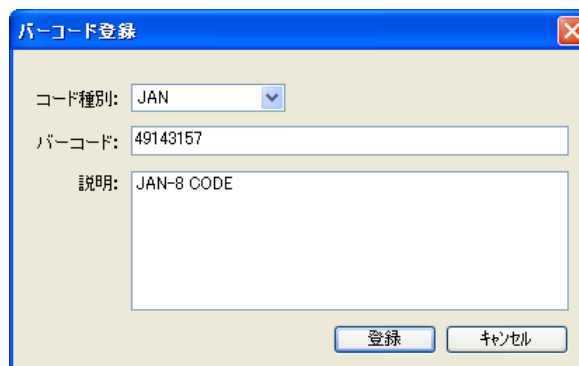


※ バーコード登録時の注意事項

登録可能なバーコードは、DT-970 エミュレータ上で読み取り可能となっているコードに限ります。エミュレータ起動直後はすべてのコードが読み取り可能となっていますが、アプリケーションを起動した場合は、そのアプリケーションで設定した読み取り可能コードのみが、登録可能となります。

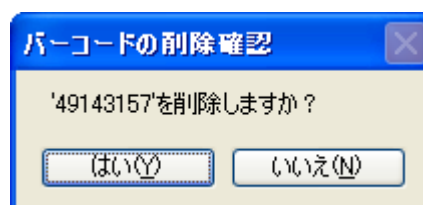
(5) 登録内容の編集

編集したいバーコードを一覧表から選択し、「編集」ボタンをクリックすると、右記の画面を表示しますので、編集してください。



(6) 登録内容の削除

削除したいバーコードを一覧表から選択し、「削除」ボタンをクリックすると、右記の画面を表示しますので、確認後、削除してください。



4.3.2. ローバッテリー発生/IOBOX接続動作

I/O シミュレータの状態設定にある「I/O ボックス接続」、「ローバッテリー」にチェックを入れると、それぞれの動きを擬似的に再現します。

- I/O ボックス接続
DT-970 エミュレータに対して、I/O ボックスが接続されたことを通知します。
アプリケーション側で I/O ボックス接続のイベントを確認することができます。
- ローバッテリー
DT-970 エミュレータに対して、ローバッテリーが発生したことを通知します。
DT-970 エミュレータのローバッテリーアイコンが点灯します。また、アプリケーション側で、ローバッテリー発生イベントを確認することができます。



4.4. Emulatorでの動作差異

DT-970 実機と、DT-970 エミュレータの動作の違いを説明します。

4.4.1. ドライブレター

ドライブレターは DT-970 実機と下記のように異なります。

(1) FAT ファイルモードで使用する関数を実行した場合

DT-970 実機	DT-970 エミュレータ
Aドライブ	¥windows
Bドライブ	¥FlashDisk
Dドライブ	¥SD カード
Eドライブ	¥USBStorage

※ プログラム内では、A:¥data.dat と記載すると、自動的に¥windows¥data.dat と読み替えを行います。

4.4.2. 通信機能

赤外線および Bluetooth 等の通信機能を使用する関数は、エミュレータでは動作しません。該当する関数を実行した場合は、何もせずにエラーコードを返し、次の行に処理が移ります。

4.4.3. システムファイル

下記のシステムファイルは動作しません。

CONFIG.HTS / CONFIG.OBR / CONFIG.ID / CONFIG.PAS / ASTART.HTS / PatchXXX.LOD

4.4.4. 副電池

副電池のローバッテリー検出はできません。

4.4.5. バックライト/コントラスト

バックライトおよびコントラストは変化しません。

4.4.6. DT-930 互換表示モード

DT-930 互換表示モードの登録、および、ビルドはできますが、機能しません。(DT-970 の表示モードで動作します)

4.4.7. ユーザフォント／外字

DT-970 エミュレータでユーザフォントおよび外字フォントを使用する場合は、お手持ちのフォントをあらかじめ TrueType フォントに変換する必要があります。

4.4.8. 関数インタフェース

(1) デバイス制御ライブラリ

関数	実機動作との違い
システムデータ管理	
dat_system	電源 : 値の設定／読出のみ 日本語／英語モード : 値の設定／読出のみ コントラスト設定 : 値の設定／読出のみ 通信 : 値の設定／読出のみ 機器 ID : CONFIG.ID の値を返却／無い場合は「??????」を返却 BIOS バージョン : 固定値 1.0000 を返却 パッチバージョン : 固定値 1.0000 を返却 拡大表示・行間挿入 : 値の設定／読出のみ 電池設定表示 : 値の設定／読出のみ LAN・USB 設定 : 値の設定／読出のみ
dat_OSVer_Read	DT970 エミュレータのバージョン／日付を返却
dat_dealer_chk	Itlaunch.exe の起動パラメータに設定した値が有効 設定しない場合は E_OK を返却
dat_mem_size	-
dat_get_device_id	固定値「123456789AAAA1」を返却
電源管理	
pwr_hold_apo	-
pwr_off	レジューム設定は無効
pwr_ioboxBootMode	E_NG を返却(動作無し)
ブザー鳴動	
s_beep	-
s_sound	-
バイブレータ	
pwr_vibrator	-
日時設定	
s_dateset	-
s_dateget	-
s_timeset	-
s_timeget	-
ファイル管理(ドライブレターが異なります)	
fil_mkdir	-
fil_rmdir	-
fil_remove	-
fil_rename	-
fil_fstat	-
fil_chsize	-
fil_getsize	-
fil_findfirst	-
fil_findnext	-
fil_filesize	-
fil_filefind	-

関数	実機動作との違い
dat_fdir	-
dat_fdel	-
dat_fname	-
dat_F_Search	-
open	-
close	-
read	-
write	-
lseek	-
sbrk	-
イベント通知機能	
pwr_inhabit	LB1 の通知のみ
pwr_inhabit_clr	-
key_fnc_mode	-
s_settimer	-
s_timerend	-
s_settimer2	-
s_timerend2	-
flg_sts	-
clr_flg	-
wai_flg	タイマーフラグ待ちのみ
画面表示	
lcd_cls	-
lcd_csr_set	-
lcd_csr_put	-
lcd_csr_get	-
lcd_char	ESC (画面クリア / カーソル位置設定) 非対応
lcd_string	ESC (画面クリア / カーソル位置設定) 非対応
lcd_string2	ESC (画面クリア / カーソル位置設定) 非対応
lcd_userstr	ESC (画面クリア / カーソル位置設定) 非対応
lcd_line	-
lcd_gaiji	切替により正常終了しますが、指定したフォント表示はできません
lcd_usrfont	切替により正常終了しますが、指定したフォント表示はできません
lcd_romfont	-
lcd_led	-
lcd_el	バックライトの明るさは変化しません
lcd_expand_set	値の設定のみ
lcd_expand_get	値の設定のみ
lcd_expand_pos_set	値の設定のみ
lcd_expand_pos_get	値の設定のみ
ビットマップ表示	
bmp_iDisplayBmpImage	-
bmp_iDisplayBmpData	-

関数	実機動作との違い
キー制御	
key_select	-
key_read	-
key_string	-
key_num	-
key_check	-
key_clear	-
key_fnc	-
OBR 制御	
OBR_open	-
OBR_close	-
OBR_getc	-
OBR_gets	-
OBR_flush	-
OBR_stat	-
OBR_moderd	-
OBR_modewt	-
OBR_chgbuf	-
OBR_trigmode	E_NG を返却(動作無し)
OBR_swing	値の設定／読出のみ
OBR_widenarrow	値の設定／読出のみ
OBR_getadjust	値の読出のみ
OBR_setadjust	値の設定のみ
OBR_getmargincheck	値の読出のみ
OBR_setmargincheck	値の設定のみ
OBR_getfocus	値の読出のみ
OBR_setfocus	値の設定のみ
USB HID 通信	
c_open	E_NG を返却(動作無し)
c_close	E_NG を返却(動作無し)
c_dout	E_NG を返却(動作無し)
c_tmdin	E_NG を返却(動作無し)

関数	実機動作との違い
IrDA 制御	
Ir_Open	E_IRNG を返却(動作無し)
Ir_Close	E_IRNG を返却(動作無し)
Ir_Read	E_IRNG を返却(動作無し)
Ir_Write	E_IRNG を返却(動作無し)
Ir_QueryTx	E_IRNG を返却(動作無し)
Ir_QueryRx	E_IRNG を返却(動作無し)
Ir_EROn	E_IRNG を返却(動作無し)
Ir_EROff	E_IRNG を返却(動作無し)
Ir_RSON	E_IRNG を返却(動作無し)
Ir_RSOff	E_IRNG を返却(動作無し)
Ir_BreakOn	E_IRNG を返却(動作無し)
Ir_BreakOff	E_IRNG を返却(動作無し)
Ir_CheckCD	E_IRNG を返却(動作無し)
Ir_CheckDR	E_IRNG を返却(動作無し)
Ir_CheckCS	E_IRNG を返却(動作無し)
Ir_CheckCI	E_IRNG を返却(動作無し)
Ir_CheckBreak	E_IRNG を返却(動作無し)
Ir_Err_Get	E_IRNG を返却(動作無し)
Ir_State_Set	E_IRNG を返却(動作無し)
Ir_SetPortConfig	E_IRNG を返却(動作無し)
Ir_Init	E_IRNG を返却(動作無し)
Ir_SetWinMode	E_IRNG を返却(動作無し)
Bluetooth 制御	
BT_Start	E_BTNG を返却(動作無し)
BT_Stop	E_BTNG を返却(動作無し)
BT_GetLocalInfo	E_BTNG を返却(動作無し)
BT_SetLocalInfo	E_BTNG を返却(動作無し)
BT_Inquiry	E_BTNG を返却(動作無し)
BT_GetDevInfo	E_BTNG を返却(動作無し)
BT_GetDevName	E_BTNG を返却(動作無し)
BT_SetPassKey	E_BTNG を返却(動作無し)
BT_SelectDev	E_BTNG を返却(動作無し)
BT_Open	E_BTNG を返却(動作無し)
BT_Close	E_BTNG を返却(動作無し)
BT_Read	E_BTNG を返却(動作無し)
BT_Write	E_BTNG を返却(動作無し)
BT_QueryRx	E_BTNG を返却(動作無し)

関数	実機動作との違い
BT_SaveDevInfo	E_BTNG を返却(動作無し)
BT_LoadDevInfo	E_BTNG を返却(動作無し)
BT_Err_Get	E_BTNG を返却(動作無し)
BT_GetPassKey	E_BTNG を返却(動作無し)
BT_SelectProfile	E_BTNG を返却(動作無し)
通信ユーティリティ制御(共通ファンクション)	
cu_stopKeySet	E_NG を返却(動作無し)
cu_setDrive	E_NG を返却(動作無し)
通信ユーティリティ制御(FLNK プロトコル)	
cu_open	E_NG を返却(動作無し)
cu_fileSend	E_NG を返却(動作無し)
cu_fileAdd	E_NG を返却(動作無し)
cu_fileRecv	E_NG を返却(動作無し)
cu_close	E_NG を返却(動作無し)
cu_readErrStat	E_NG を返却(動作無し)
cu_idle	E_NG を返却(動作無し)
cu_cmdRecv	E_NG を返却(動作無し)
cu_fileDelete	E_NG を返却(動作無し)
cu_fileMove	E_NG を返却(動作無し)
cu_makeDir	E_NG を返却(動作無し)
cu_getFileInfo	E_NG を返却(動作無し)
cu_setFileInfo	E_NG を返却(動作無し)
cu_getDiskInfo	E_NG を返却(動作無し)
cu_dateTime	E_NG を返却(動作無し)
cu_getSysInfo	E_NG を返却(動作無し)
cu_msgSend	E_NG を返却(動作無し)
cu_beep	E_NG を返却(動作無し)
cu_setIoboxInfo	E_NG を返却(動作無し)
cu_fchklog_Create	E_NG を返却(動作無し)
cu_fchklog_Check	E_NG を返却(動作無し)
共通関数	
dat_Apload	呼び出し元のアプリケーションは自動終了しません ※関数実行後 ap_start()から直ちに処理を抜けるようにしてください
abort	-
exit	-
wkup_cost	メッセージ BOX が表示されます
wkup_calib	メッセージ BOX が表示されます
ITRON 互換関数	
it2_flg_sts	-
it2_clr_flg	-
it2_wai_flg	-

関数	実機動作との違い
Socket 制御	
net_socket	-1 を返却(動作無し)
net_bind	-1 を返却(動作無し)
net_connect	-1 を返却(動作無し)
net_listen	-1 を返却(動作無し)
net_accept	-1 を返却(動作無し)
net_send	-1 を返却(動作無し)
net_sendto	-1 を返却(動作無し)
net_recv	-1 を返却(動作無し)
net_recvfrom	-1 を返却(動作無し)
net_shutdown	-1 を返却(動作無し)
net_close	-1 を返却(動作無し)
net_select	-1 を返却(動作無し)
net_getsockopt	-1 を返却(動作無し)
net_setsockopt	-1 を返却(動作無し)
net_getsockerr	-1 を返却(動作無し)
net_inet_aton	0 を返却(動作無し)
net_inet_addr	0 を返却(動作無し)
net_inet_ntoa	NULL を返却(動作無し)
net_tcpip_ini	E_NG を返却(動作無し)
net_tcpip_wai_rdy	E_NG を返却(動作無し)
net_ascii_to_ipaddr	0 を返却(動作無し)
net_ipaddr_to_ascii	0 を返却(動作無し)
net_byte4_to_long	0 を返却(動作無し)
net_long_to_byte4	動作無し
net_ping_send	E_NG を返却(動作無し)
net_get_myip_info	E_NG を返却(動作無し)
LAN 制御	
lan_setParam_IPmode	E_NG を返却(動作無し)
lan_setParam_IPaddr	E_NG を返却(動作無し)
lan_setParam_subnet	E_NG を返却(動作無し)
lan_setParam_gateway	E_NG を返却(動作無し)
LAN クレードル制御	
lancradle_get_IPaddr	E_NG を返却(動作無し)
lancradle_set_IPaddr	E_NG を返却(動作無し)
PPP 制御	
ppp_init	E_NG を返却(動作無し)
ppp_setParam_MdmInit1	E_NG を返却(動作無し)
ppp_setParam_MdmInit2	E_NG を返却(動作無し)
ppp_ctrl_call	E_NG を返却(動作無し)
ppp_ctrl_disconnect	E_NG を返却(動作無し)

(2) 拡張機能ライブラリ

関数	実機動作との違い
アプリケーション支援ライブラリ	
ht_CheckDate	-
ht_CheckYear	-
ht_ConvertYear	-
ht_Checksum	-
ht_CalcCRC_ANSI	-
ht_CalcCRC_CCITT	-
ht_CalcCRC_X	-
ht_CalcLRC	-
ht_CheckCD	-
ht_FCWait	-
ht_StrInp	-
ht_NumInp	-
ht_DateInp	-
ht_TimeInp	-
ht_ShiftMode	-
ht_FLNKsend	E_NG を返却(動作無し)
ht_FLNKrecv	E_NG を返却(動作無し)
lr_c_open	E_NG を返却(動作無し)
lr_c_close	E_NG を返却(動作無し)
lr_c_status	E_NG を返却(動作無し)
lr_c_hold	E_NG を返却(動作無し)
lr_c_chkopen	E_NG を返却(動作無し)
lr_c_dout	E_NG を返却(動作無し)
lr_c_din	E_NG を返却(動作無し)
lr_c_tmdin	E_NG を返却(動作無し)
lr_c_out	E_NG を返却(動作無し)
lr_c_break	E_NG を返却(動作無し)
lr_c_trxr	E_NG を返却(動作無し)
lr_c_iobox	E_NG を返却(動作無し)
lr_c_irout	E_NG を返却(動作無し)
lr_c_timer	E_NG を返却(動作無し)
lr_c_rs	E_NG を返却(動作無し)
lr_c_er	E_NG を返却(動作無し)
lr_c_errs	E_NG を返却(動作無し)
lr_c_flush	E_NG を返却(動作無し)
lr_c_bfsts	E_NG を返却(動作無し)
lr_c_errbfing	E_NG を返却(動作無し)
lr_c_r derrsts	E_NG を返却(動作無し)
lr_c_chghdr	E_NG を返却(動作無し)
ht_fileopen	-
ht_fileclose	-

関数	実機動作との違い
ht_fileread	-
ht_filewrite	-
ht_filesize	-
ht_filelof	-
ht_waitmsec	-
ht_dbgsendmsg	動作無し
ht_beep	-
Ir_xy_modem	エラー終了(動作無し)
高速ファイルサーチライブラリ	
iHashAssign	-
iHashRead	-
iHashWrite	-
iHashAdd	-
MobilePrinter 制御ライブラリ	
MPRInit	E_NG を返却(動作無し)
MPRDeinit	E_NG を返却(動作無し)
MPROpen	E_NG を返却(動作無し)
MPRClose	E_NG を返却(動作無し)
MPRSend	E_NG を返却(動作無し)
MPRReceive	E_NG を返却(動作無し)
MPRCalcCRC16	E_NG を返却(動作無し)
MPRSetIrDA	E_NG を返却(動作無し)
MPRGetIrDA	E_NG を返却(動作無し)
MPRSetBluetooth	E_NG を返却(動作無し)
MPRGetBluetooth	E_NG を返却(動作無し)

5. その他機能

5.1. フォントコンバータ

DT-970 SDK ではツールとして「フォントコンバータ」を用意しています。

DT-970 エミュレータと実機ではフォントファイルのフォーマットが異なるため、DT-970 エミュレータでユーザ定義フォント(*1)を使用する場合は、DT-970 実機で使用しているユーザ定義フォントファイルをエミュレータ用のユーザ定義フォントファイルに変換する必要があります。

また、本ツールを使用して、新規にユーザ定義フォントファイル (for エミュレータ) を作成することが可能です。

*1…ユーザ定義フォント

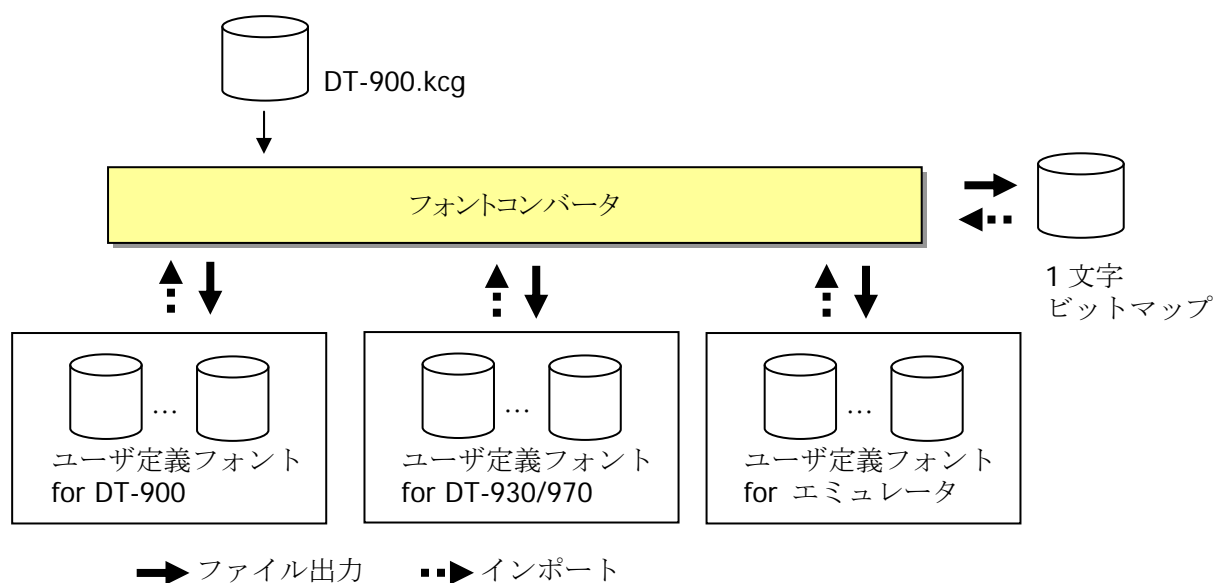
縮小 ANK (6×6)、縮小 ANK (8×8)、縮小 ANK (10×10)

標準 ANK (6×12)、標準 ANK (8×16)、標準 ANK (10×20)

漢字 (12×12)、漢字 (16×16)、漢字 (20×20)

外字 (12×12)、外字 (16×16)、外字 (20×20)

フォントコンバータの構成



DT-970 基本開発環境では、下記フォルダに初期フォントをインストールします。

Windows 種別	格納フォルダ
XP,2003	¥Documents and Settings¥All Users¥Application Data¥CASIO¥Fontcnvw¥Font
7,2008	¥Users¥All Users¥CASIO¥Fontcnvw¥Font

DT-900.kcg

DT-900.kcg は DT-970 基本開発環境でインストールされます。

ユーザ定義フォントファイル (for DT-900、DT-930/970、エミュレータ) を新規に作成する場合は、本ファイルをもとにフォントデータを作成します。

ユーザ定義フォントファイル (for DT-900)

ユーザ定義フォントファイル (for DT-900) は DT-900 実機で使用するユーザ定義フォントファイルです。

既にユーザ定義フォントファイル (for DT-900) がある場合は、その編集および DT-930/970、エミュレータ用のユーザ定義フォントファイルに変換することが可能です。

また、DT-900.kcg を使用して新規に DT-900 用のユーザ定義フォントファイルを作成することが可能です。

ユーザ定義フォントファイル (for DT-930/970)

ユーザ定義フォントファイル (for DT-930/970) は DT-930/970 実機で使用するユーザ定義フォントファイルです。

既にユーザ定義フォントファイル (for DT-930/970) がある場合は、その編集および DT-900、エミュレータ用のユーザ定義フォントファイルに変換することが可能です。

また、DT-900.kcg を使用して新規に DT-930/970 用のユーザ定義フォントファイルを作成することが可能です。

ユーザ定義フォントファイル (for エミュレータ)

ユーザ定義フォントファイル (for エミュレータ) はエミュレータで使用するユーザ定義フォントファイルです。

既にユーザ定義フォントファイル (for DT-900、DT-930/970) がある場合は、それをもとに作成することが可能です。

また、DT-900.kcg を使用して新規にエミュレータ用のユーザ定義フォントファイルを作成することが可能です。

1 文字ビットマップファイル

編集したい 1 文字を選択し、その文字をビットマップファイル (*.bmp) として出力することが可能です。

また、出力したビットマップファイルを外部のビットマップエディタで編集し、各ユーザ定義フォントファイルに反映することが可能です。

5.1.1. 機能

フォントデータ表示／反映

作成または編集したユーザ定義フォントファイルを確認するための Viewer 機能です。

既存のユーザ定義フォントファイル (for DT-900、DT-930/970) を表示する場合は、以下のファイル名および拡張子に修正してください。

- ファイル名

ファイル名	内容
ANK6	縮小 ANK (6×6)
ANK8	縮小 ANK (8×8)
ANK10	縮小 ANK (10×10)
ANK12	標準 ANK (6×12)
ANK16	標準 ANK (8×16)
ANK20	標準 ANK (10×20)
Kanji12	漢字 (12×12)
Kanji16	漢字 (16×16)
Kanji20	漢字 (20×20)
Gaiji12	外字 (12×12)
Gaiji16	外字 (16×16)
Gaiji20	外字 (20×20)

- 拡張子

拡張子	内容
*.ufh	for DT-900
*.ufv	for DT-930/970

フォント変換

作成または編集したユーザ定義フォントファイルを指定した機種用 (DT-900、DT-930/970、エミュレータ) のユーザ定義フォントファイルまたは 1 文字ビットマップファイルに変換します。

DT-900、DT-930/970 用のユーザ定義フォントファイルは上記のファイル名と拡張子で出力します。

また、エミュレータ用のユーザ定義フォントファイルは下記のファイル名で出力します。

• ユーザフォント定義ファイル (for エミュレータ)

ファイル名	内容	タイプフェース名
UserAL6.ttf	縮小 ANK (6×6)	UserFontAL
UserAL6W.ttf	横倍縮小 ANK (6×6)	
UserAL8.ttf	縮小 ANK (8×8)	
UserAL8W.ttf	横倍縮小 ANK (8×8)	
UserAL10.ttf	縮小 ANK (10×10)	
UserAL10W.ttf	横倍縮小 ANK (10×10)	
UserAS6.ttf	標準 ANK (6×12)、漢字 (12×12)	UserFontAS
UserAS6W.ttf	横倍標準 ANK (6×12)、横倍漢字 (12×12)	
UserAS8.ttf	標準 ANK (8×16)、漢字 (16×16)	
UserAS8W.ttf	横倍標準 ANK (8×16)、横倍漢字 (16×16)	
UserAS10.ttf	標準 ANK (10×20)、漢字 (20×20)	
UserAS10W.ttf	横倍標準 ANK (10×20)、横倍漢字 (20×20)	
GaijiAS6.ttf	外字 (12×12)	GaijiFontAS
GaijiAS6W.ttf	横倍外字 (12×12)	
GaijiAS8.ttf	外字 (16×16)	
GaijiAS8W.ttf	横倍外字 (16×16)	
GaijiAS10.ttf	外字 (20×20)	
GaijiAS10W.ttf	横倍外字 (20×20)	

• 1 文字ビットマップファイル

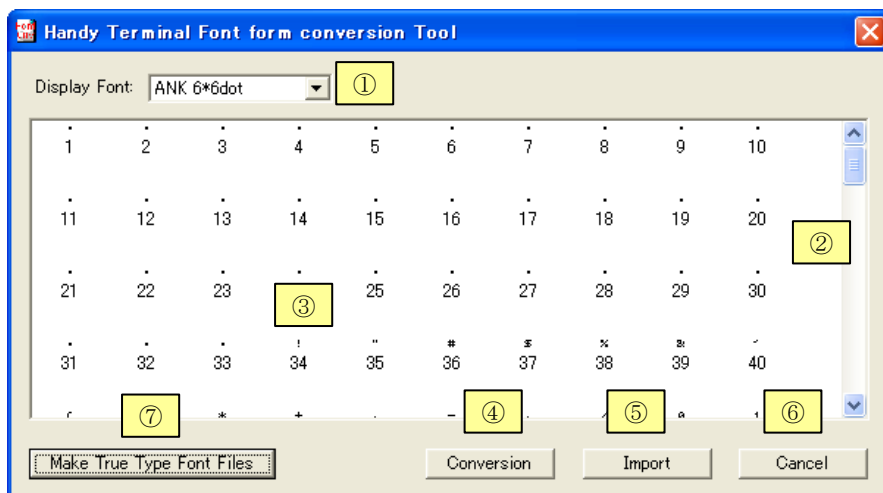
1 文字ビットマップファイルのファイル名は、フォントファイル名_ユーザ定義フォントファイル内のインデックス.bmp となります。

例) 標準 ANK (10×20) の「A」

ANK20_66.bmp

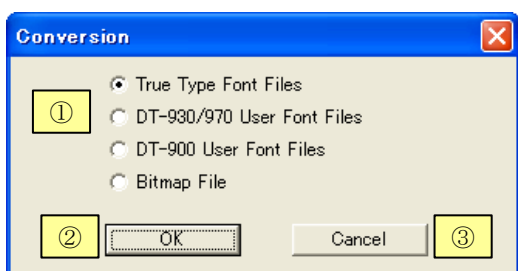
5.1.2. 画面構成

基本画面



No	項目名	説明
①	Display Font	Font 表示領域に表示するフォント種類を指定します。 デフォルトは、ANK 6*6dot です。 フォント種類には、ANK6*6、ANK8*8、ANK10*10、ANK6*12、ANK8*16、ANK10*20、Kanji12*12-1、-2、-3、Kanji16*16-1、-2、-3、Kanji20*20-1、-2、-3、Gaiji12*12、Gaiji16*16、Gaiji20*20 があります。
②	Font 表示領域	①で指定したフォントを表示する領域です。 定義されていない文字は「・」で表示します。
③	選択フォント	Font 表示領域に表示している 1 文字を選択することができます。 選択すると他の文字と区別できるように、インデックスのバックグラウンドカラーが変化します。
④	Conversion ボタン	Conversion 画面を表示します。 詳細は下記の Conversion 画面を参照してください。
⑤	Import ボタン	Import 画面を表示する。 詳細は下記の Import 画面を参照してください。
⑥	Cancel ボタン	フォントコンバータを終了します。
⑦	Make ボタン	Import 画面を表示し、選択したフォントの True Type Font を生成します。 詳細は下記の Import 画面を参照してください。

Conversion 画面



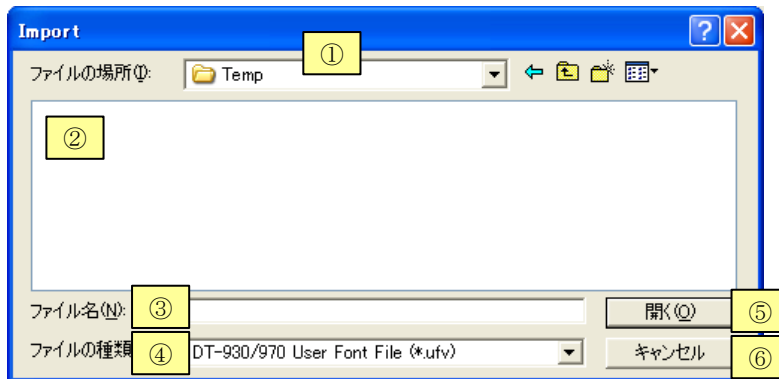
No	項目名	説明
①	出力ファイルの種類	出力するファイル形式を選択します <ul style="list-style-type: none"> • True Type Font Files ユーザ定義フォントファイル (for エミュレータ) • DT-930/970 User Font Files ユーザ定義フォントファイル (for DT-930/970) • DT-900 User Font Files ユーザ定義フォントファイル (for DT-900) • Bitmap File 1 文字ビットマップファイル 出力ファイルは「5.1.1機能」に示したファイル名で保存します
②	OK ボタン	変換ファイルの保存フォルダを指定するダイアログを表示します (下記参照)
③	Cancel ボタン	Conversion 画面を閉じ、基本画面に戻ります



No	項目名	説明
①	保存するフォルダ	ファイルを保存するフォルダを指定します
②	OK ボタン	ファイルを作成し保存します
③	キャンセル ボタン	本ダイアログを閉じ、Conversion 画面に戻ります

※ 本ダイアログで指定したフォルダ内に、出力ファイルと同名のファイルが存在する場合は、確認メッセージを出力します。詳細は下記のメッセージダイアログを参照してください。

Import 画面

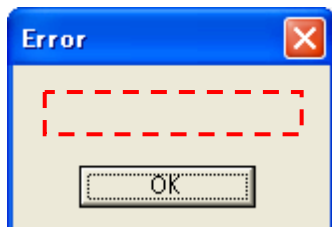


No	項目名	説明
①	ファイルの場所(I)	取り込むファイルが存在するフォルダを選択します
②	ファイル表示領域	①で指定したフォルダ内にある④で指定した拡張子のファイルを表示します
③	ファイル名(N)	取り込むファイルを指定します(複数指定可能) 「5.1.1機能」に示したすべてのファイルを指定しなくても、エラーにはなりません 指定したファイルに関するフォントのみ、基本画面の Font 表示領域を変更します 「5.1.1機能」に示したファイル名と異なるファイルを指定した場合はエラーメッセージを出力し、取り込みを停止します
④	ファイルの種類(I)	取り込むファイルの種類を拡張子で指定します
⑤	開く(O) ボタン	Import ボタンから呼び出した場合： ③で指定したファイルをもとに、基本画面の Font 表示領域を更新し、基本画面に戻ります Make ボタンから呼び出した場合： ③で指定したファイルをもとに、True Type Font を生成し、基本画面に戻ります
⑥	キャンセル ボタン	Import 画面を閉じ、基本画面に戻ります

メッセージダイアログ

- エラーメッセージダイアログ

本ツール使用時に、エラーが発生した場合は、エラーメッセージダイアログを表示します。
 図中の点線内に下表のメッセージのいずれかを表示します。



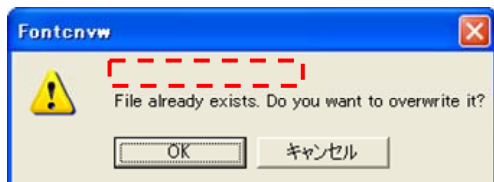
エラーメッセージ	説明
The KCG file is not found. Application will close.	DT-900.kcg ファイルが存在しない場合
The memory which this application uses is short.	メモリが不足している場合
The formats of the KCG file are different. Application will close.	DT-900.kcg のバージョンが異なる場合
File name is different.	*.ufv、*.ufh のファイル名が規定と異なる場合
Please choose 1 character to change.	ビットマップファイルをインポートするときに変更する文字を選択していない場合
Size of 1 character to change is different from a Bitmap file.	ビットマップファイルをインポートするときを選択した文字とビットマップファイルのサイズが異なる場合
The BIN files are not found.	TrueTypeFont ファイルの入力となる BIN ファイルが存在しない場合
This application cannot read an external character file.	外字ファイルを読み込めない場合

- 確認メッセージダイアログ

ユーザフォント定義ファイル保存時に、指定したフォルダ内に同名のファイルが存在する場合は、確認メッセージダイアログを表示します。

ファイルを上書きする場合は「OK」ボタンを、上書きしない場合は「キャンセル」ボタンを押下してください。

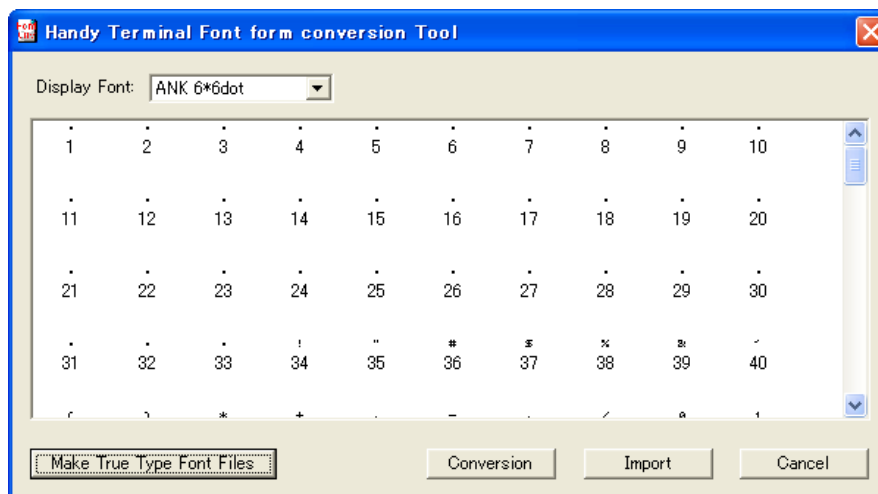
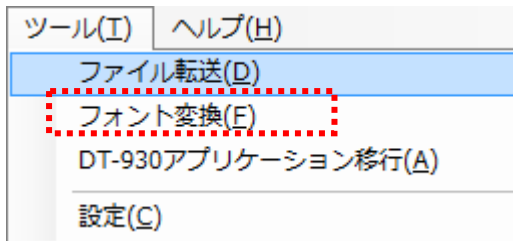
図中の点線内に対象のファイル名を表示します。



5.1.3. 使用方法

(1) 起動方法

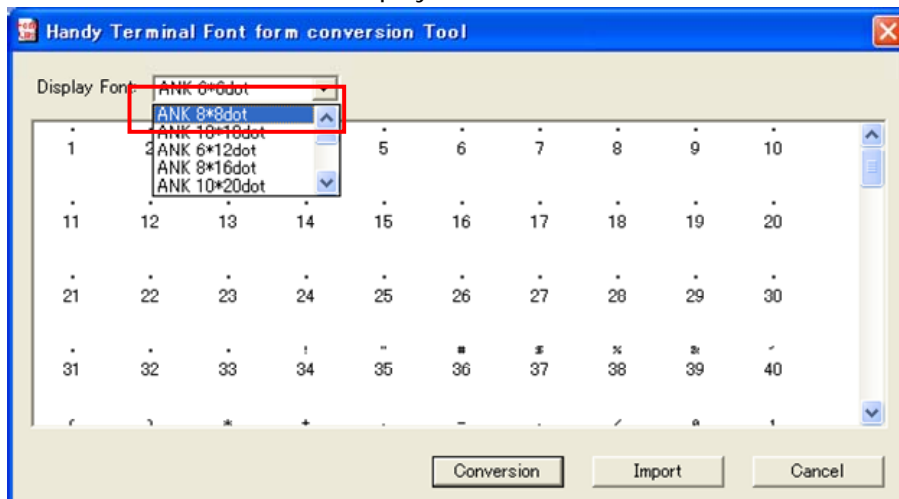
アプリケーション開発環境メニューの「**フォント変換**」をクリックすることにより、「5.1.2画面構成」に示した基本画面が起動します。



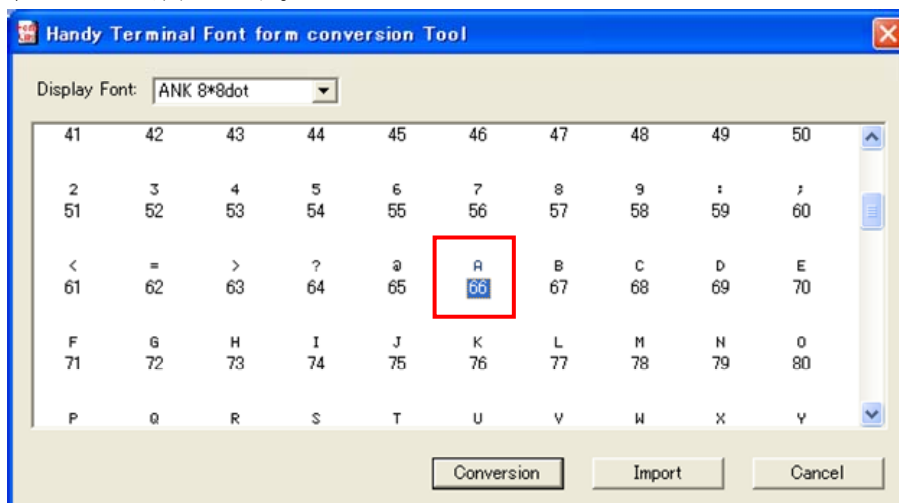
(2) ユーザ定義フォントファイルの新規作成

ユーザ定義フォントファイルを新規作成します。ここでは、縮小 ANK (8×8) の「A」を変更し、ユーザ定義フォントファイル (for エミュレータ) を作成する手順を示します。

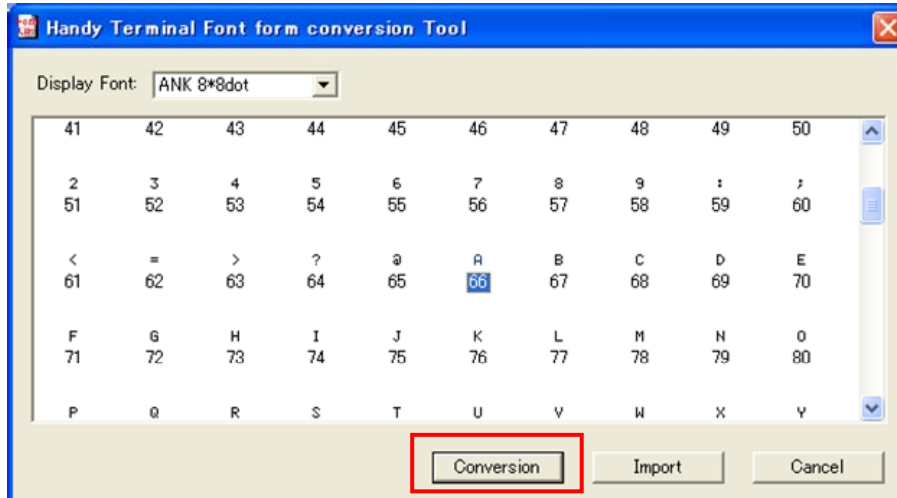
1. フォントコンバータを起動し、Display Font を「ANK 8*8dot」に変更します。



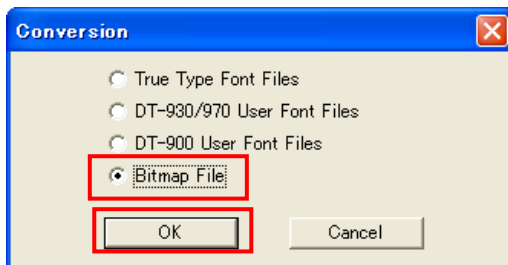
2. Font 表示領域に表示している「A」を左クリックし、インデックスナンバーのバックグラウンドカラーが変化したことを確認します。



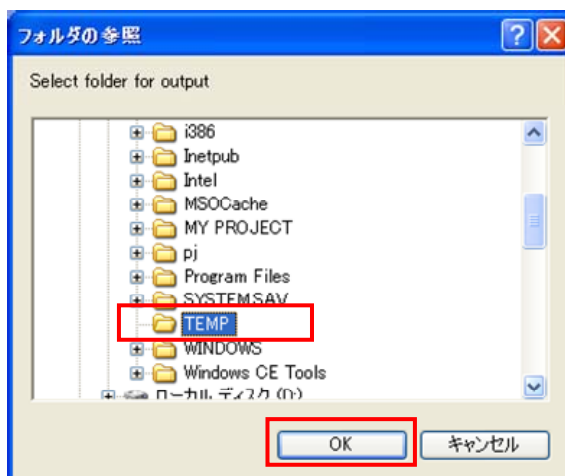
3. 「**Conversion**」ボタンを押下し、Conversion 画面が起動したことを確認します。



4. 「**Bitmap File**」にチェックを入れ、「**OK**」を押下します。

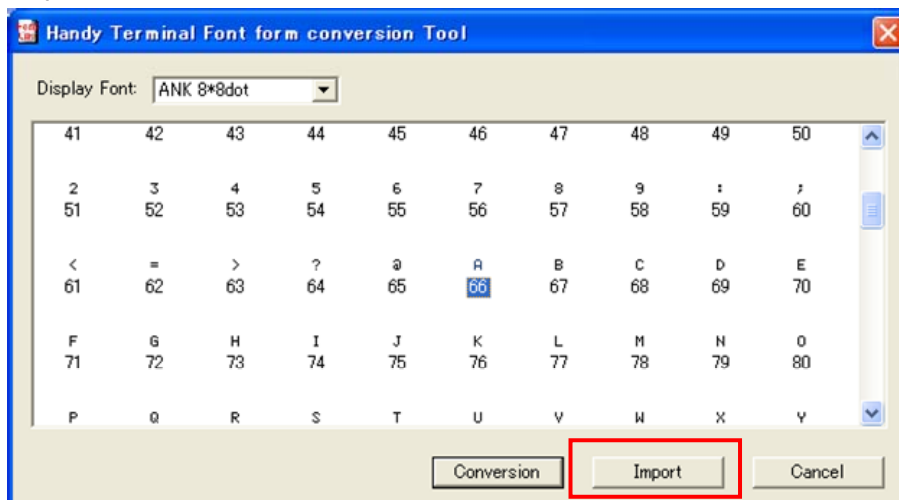


5. 出力する任意のフォルダを選択し、「**OK**」を押下します。ここでは、**C:¥TEMP** に出力します。出力フォルダ内に同名のファイルが存在する場合は上書き確認ダイアログを表示します。上書きしない場合は別のフォルダを指定してください。

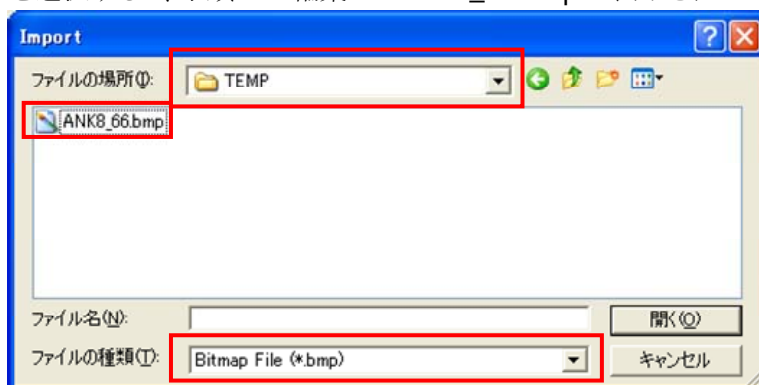


6. **C:¥TEMP** に **ANK8_66.bmp** が存在することを確認し、ビットマップエディタを用いて編集し、**ANK8_66.bmp** を上書き保存します。

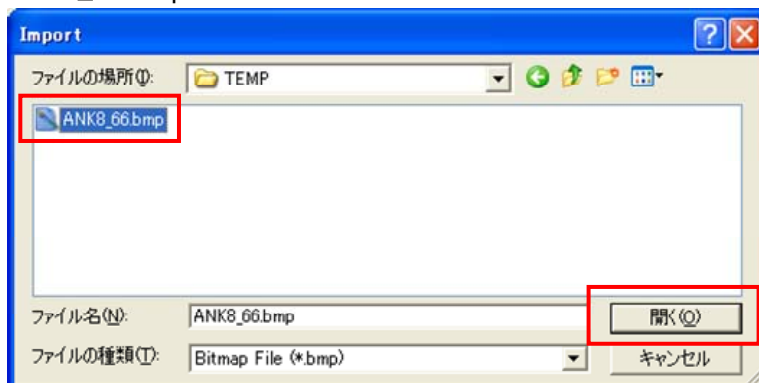
7. 手順 1.および 2.と同様に、「ANK 8*8dot」の「A」を選択した状態で、「Import」ボタンを押下し、Import 画面が起動したことを確認します。



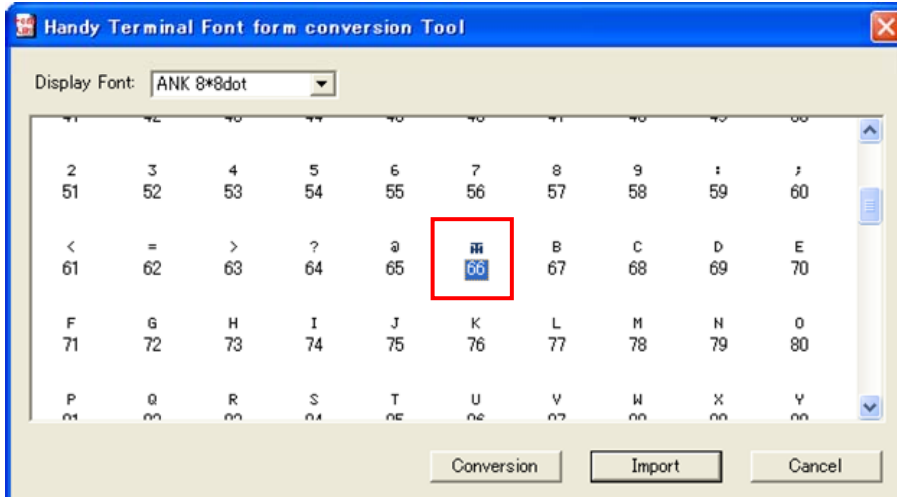
8. 「ファイルの場所」に手順 5.で指定した C:¥TEMP を、「ファイルの種類」に「Bitmap File (*.bmp)」を選択すると、手順 6.で編集した ANK8_66.bmp が表示されていることを確認します。



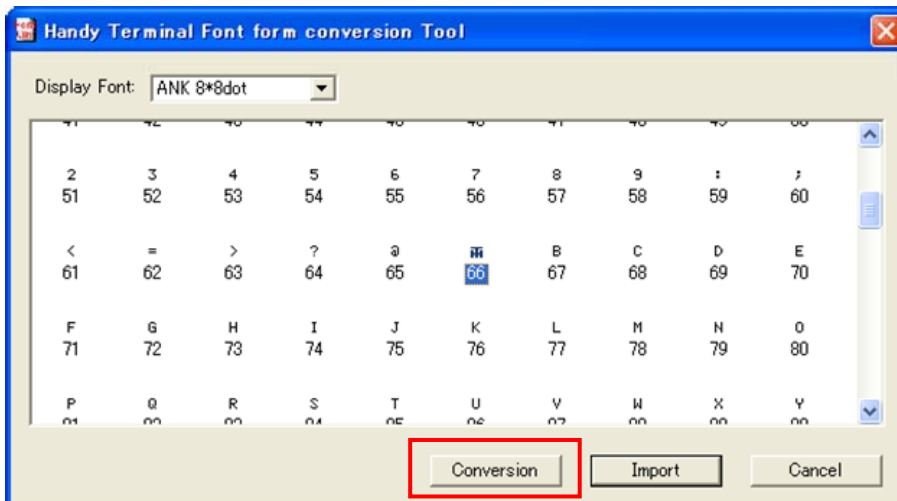
9. ANK8_66.bmp を選択し、「開く」ボタンを押下します。



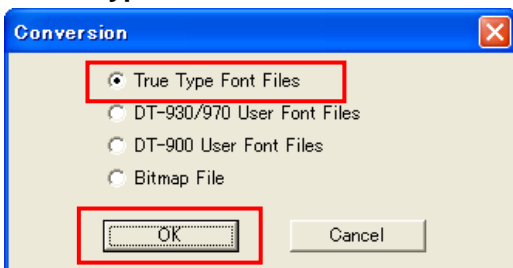
10. Font 表示領域内の「A」が手順 6.で編集した文字に変わっていることを確認します。



11. 「Conversion」ボタンを押下し、Conversion 画面を起動したことを確認します。



12. 「True Type Font Files」にチェックを入れ、「OK」ボタンを押下します。



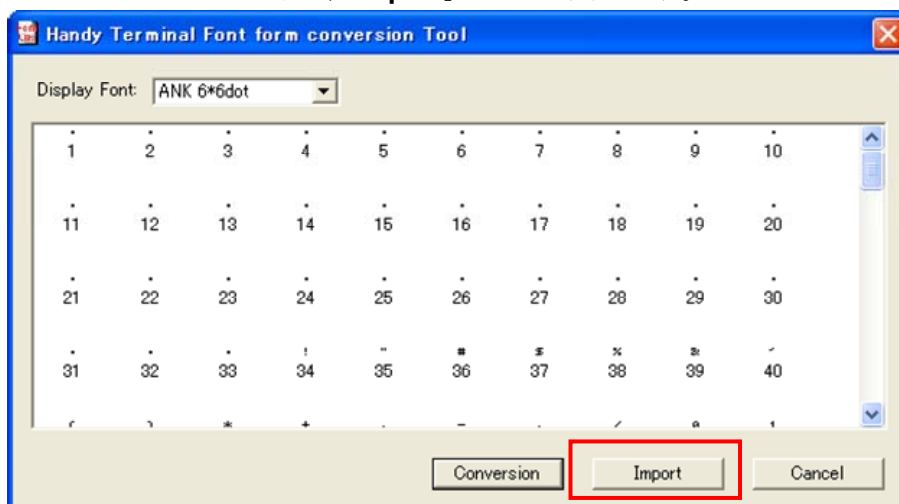
13. 手順 5.と同様に、出力先フォルダに C:¥TEMP を選択し、「OK」ボタンを押下します。
出力フォルダ内に同名のファイルが存在する場合は上書き確認ダイアログを表示します。上書きしない場合は別のフォルダを指定してください。
14. C:¥TEMPフォルダ内に「5.1.1機能」のユーザフォント定義ファイル(forエミュレータ)に示したファイルが存在することを確認します。

ユーザ定義フォントファイル(for DT-900、DT-930/970)を新規作成する場合は、手順 12.で「DT-900 User Font Files」または「DT-930/970 User Font Files」を選択してください。

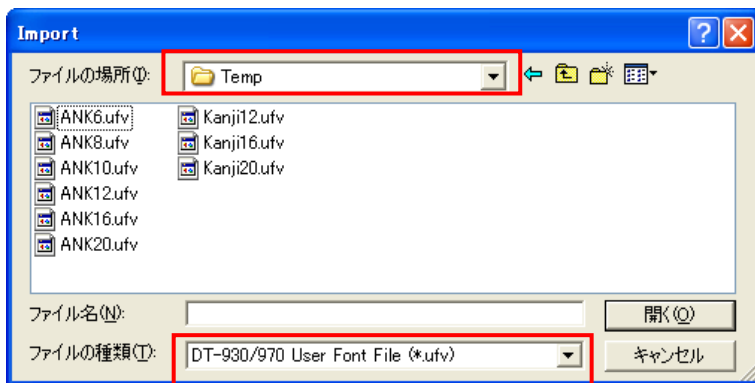
(3) 既存のユーザ定義フォントファイルの編集

既存のユーザ定義フォントファイル(for DT-930/970)を編集します。ここで使用する既存のユーザ定義フォントファイルは、漢字(20×20)の「亜」を「DT930」に変更しているものとします。ここでは、「DT930」を変更し、ユーザ定義フォントファイル(for DT-930/970)を編集する手順を示します。

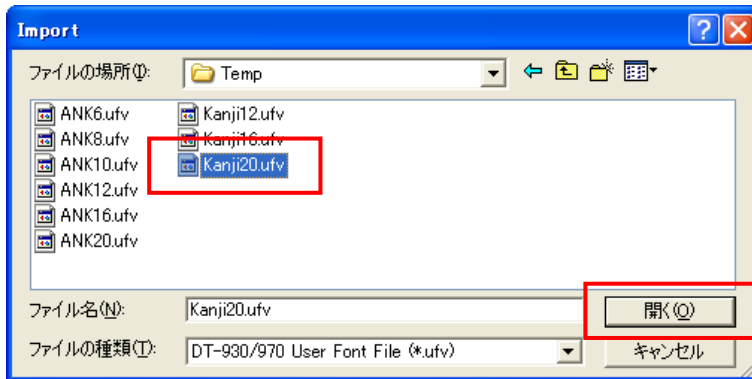
1. 既存のユーザ定義フォントファイルを「5.1.1機能」に示したファイル名および拡張子に変更します。
2. フォントコンバータを起動し、「Import」ボタンを押下します。



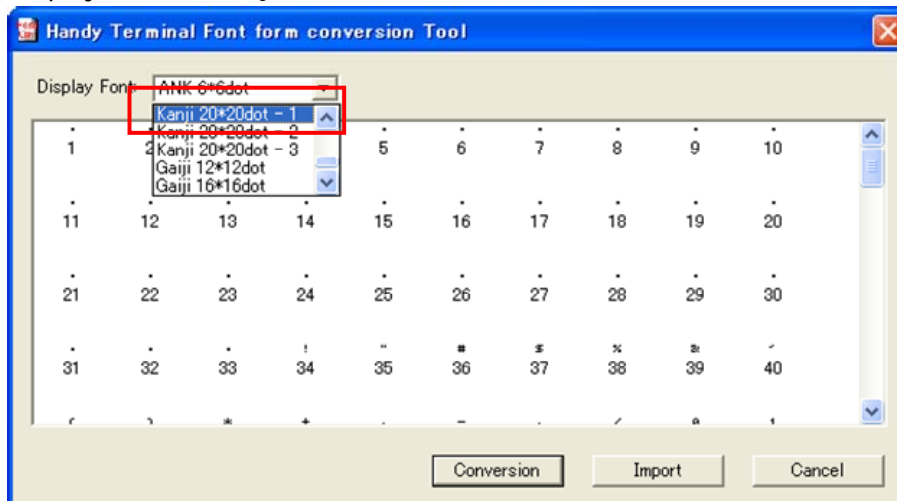
3. 「ファイルの場所」に手順 1.でファイル名および拡張子を変更したファイルが存在するフォルダを、「ファイルの種類」に「DT-930/970 User Font File (*.ufv)」を選択します。ここでは、C:¥TEMP にファイルがあるものとします。



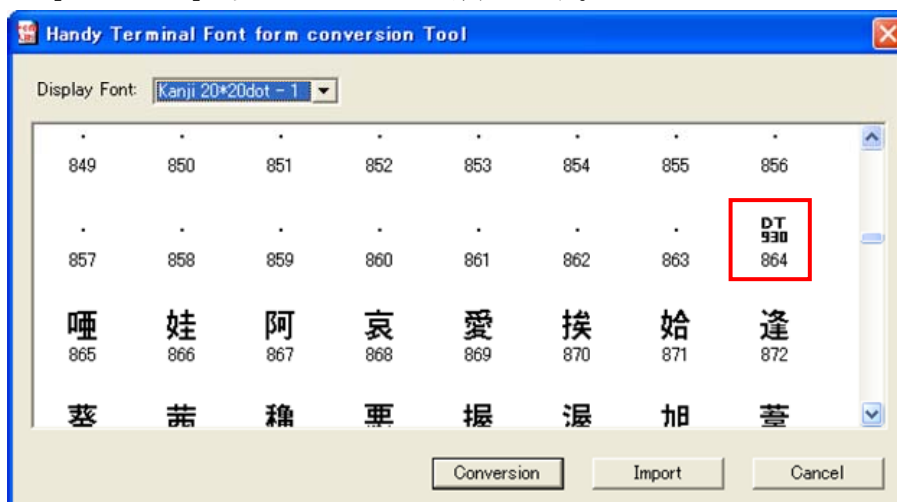
- 「**Kanji20.ufv**」を選択し、「開く」ボタンを押下します。
(ファイルを複数選択することも可能です。ここでは漢字(20×20)を編集するため、Kanji20.ufv のみを選択しています。)



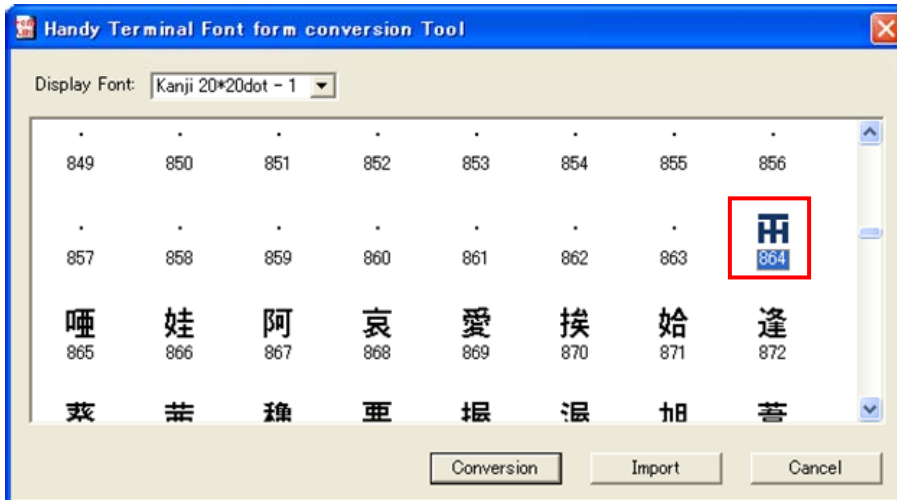
- Display Font を「**Kanji 20*20dot - 1**」に変更します。



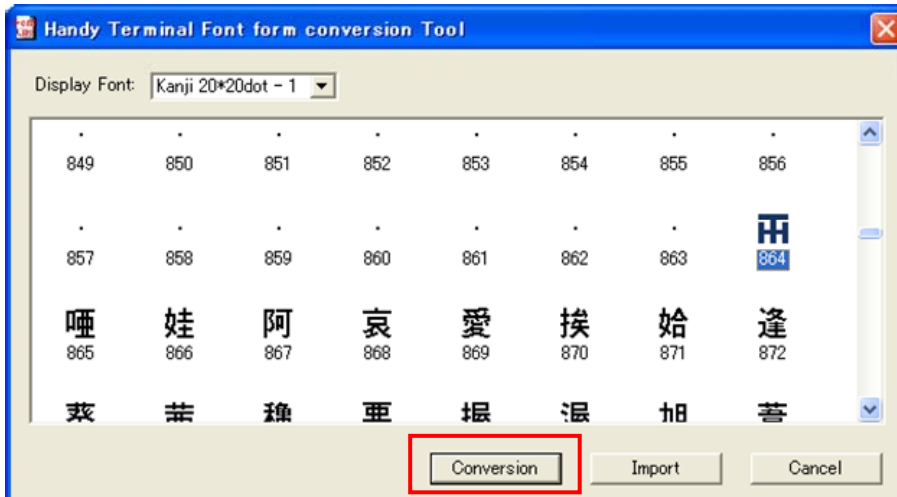
- 「罍」が「**DT930**」に変わっていることを確認します。



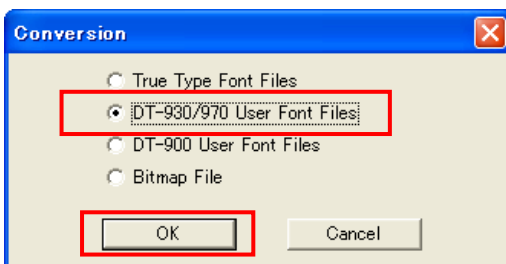
7. 「(2) ユーザ定義フォントファイルの新規作成」の手順 3.~10.と同様に、「DT930」を編集し、その編集した文字が Font 表示領域に表示されていることを確認します。



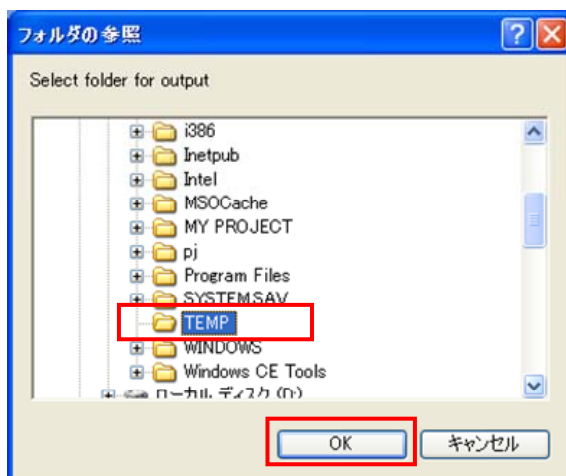
8. 「**Conversion**」ボタンを押下し、Conversion 画面を起動したことを確認します。



9. 「**DT-930/970 User Font Files**」にチェックを入れ、「**OK**」ボタンを押下します。



10. 出力する任意のフォルダを選択し、「OK」を押下します。ここでは、C:¥TEMP に出力します。出力フォルダ内に同名のファイルが存在する場合は上書き確認ダイアログを表示します。上書きしない場合は別のフォルダを指定してください。



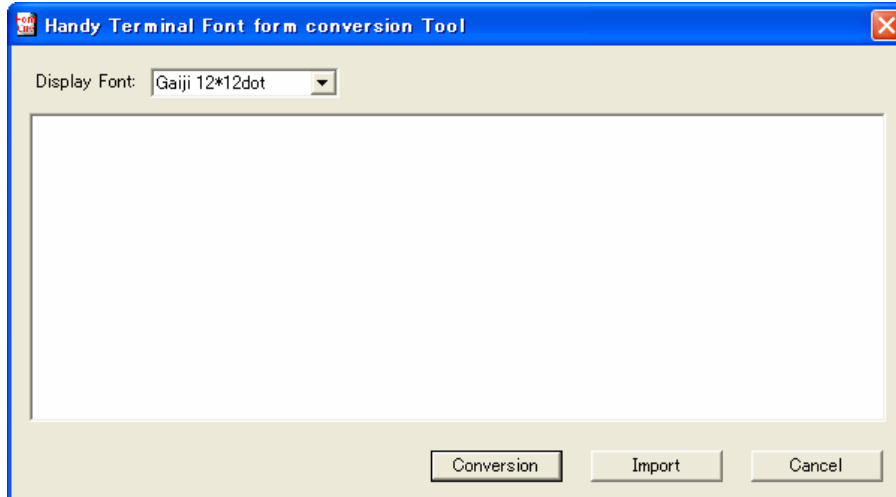
11. C:¥TEMPフォルダ内に「5.1.1機能」のユーザフォント定義ファイル(for DT-930/970)に示したファイルが存在することを確認します。

編集したユーザ定義フォントファイル(for DT-930/970)をエミュレータ、DT-900 で使用する場合は、手順 9.で「**True Type Font Files**」または「**DT-900 User Font Files**」を選択してください。また、手順 3.で「**ファイルの種類**」に「**DT-900 User Font File (*.ufh)**」を選択することにより、DT-900用のユーザ定義フォントファイルの編集が可能となります。

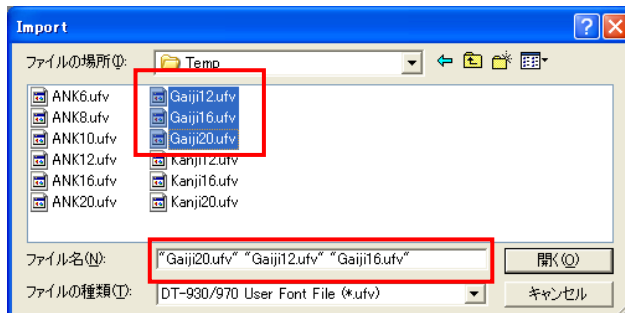
5.1.4. 注意事項

外字について

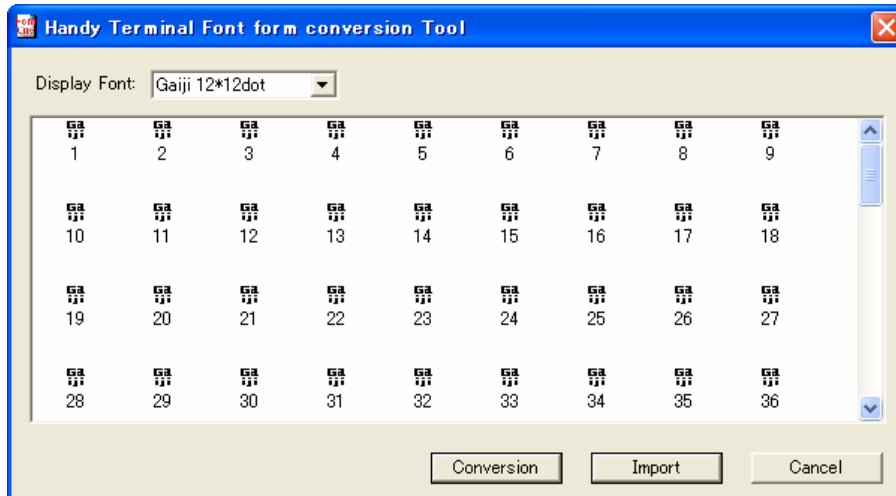
本ツールの初回起動時は、Display Font を「**Gaiji 12*12dot**」、「**Gaiji 16*16dot**」、「**Gaiji 20*20dot**」に設定しても、Font 表示領域に何も表示しません。



外字を編集するためには、「5.1.3使用方法」の「(3) 既存のユーザ定義フォントファイルの編集」の手順3と同様の手順で既存の外字ファイルを取り込む必要があります。



取り込んだ外字ファイルが Font 表示領域に反映されていることを確認してから、編集を行ってください。



6. アプリ開発ステップ詳細

6.1. 目的

DT-970 アプリ開発は、Visual 開発環境(本ツール)を用意することで、視覚的／直感的な操作で、コンパイラ等に関しての専門的な知識を不要としています。

一方で、コンパイラ等の専門的な知識があれば、ツール操作ではなく、アプリ開発手順をバッチ処理化することで、自動化なども可能です。

このような用途に向けて、DT-970 アプリ開発手順のステップ詳細を記載します。

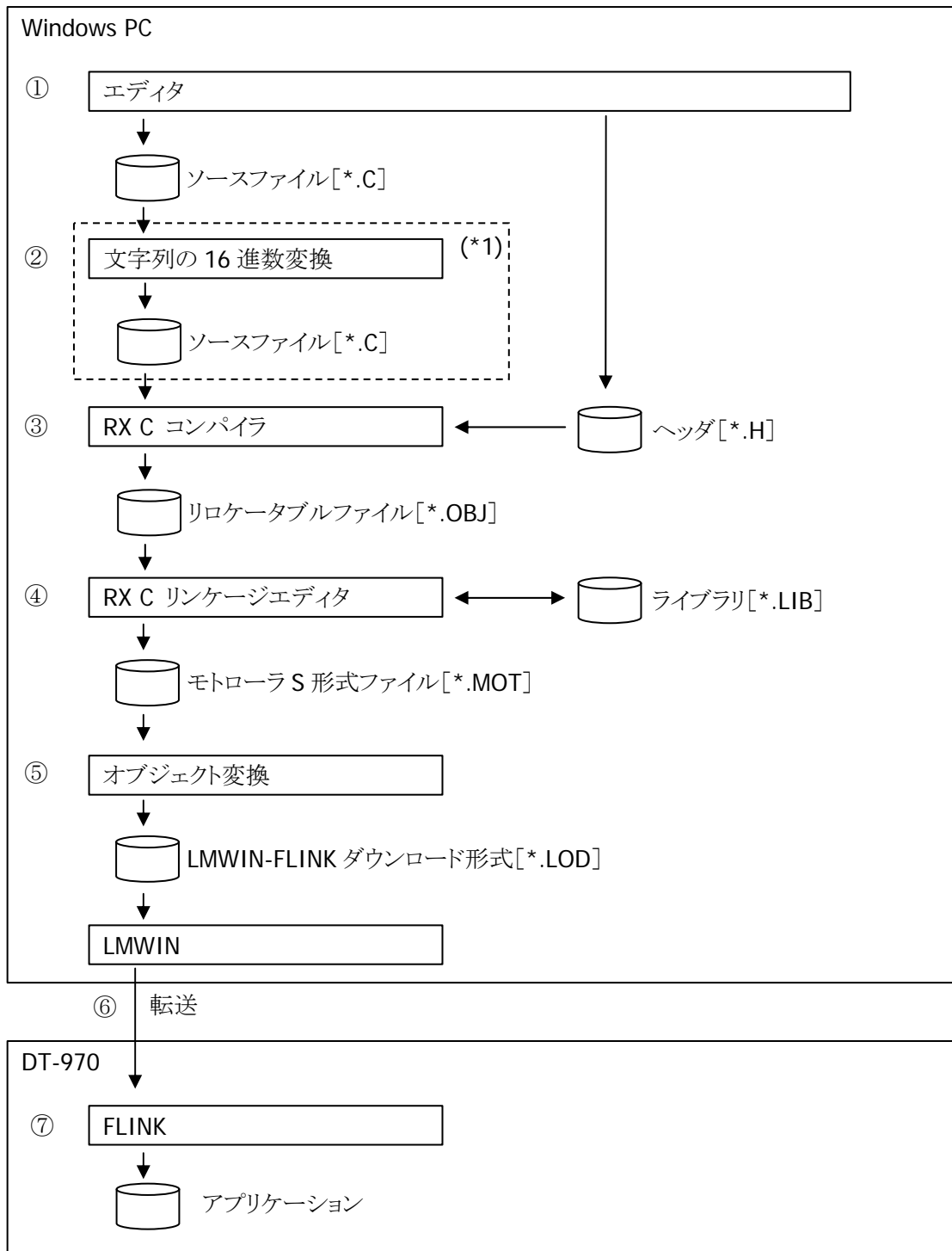
アプリ開発手順のステップは、基本的には、DT-930 と同一ですが、利用するコンパイラ／リンケージエディタが異なるため、これらに対するオプション指定などに差異があります。

DT-900/930/940 では、アプリケーション生成用に、スタートアップ用オブジェクトとして AP_START.OBJ と AP_INIT.OBJ を提供していましたが、DT-970 では、この2つのオブジェクトを統合し AP_START.OBJ のみの提供となります。(AP_INIT.OBJ のリンクは不要となります)

また、DT-930 でサポートしていた「メモリ ライトプロテクト」機能については、DT-970 では未サポートとなっています。このため、DT-930 のアプリ開発手順として実施していた、下記再リンク処理は、DT-970 では実施する必要がありません。

1. リンケージエディタでリンク実行後、B,R セクション再配置(SUBMK.EXE)
2. SUBCOMMAND ファイル更新後、再リンクを実行

6.2. 全体の流れ



*1...C コンパイラのエンコードオプション **-sjis** を利用すれば不要。

No	説明
①	エディタで、C ソース、C ヘッダを作成します。
②	文字列表記を 16 進数に変換します。(KJ_CNVRT) 本処理を実施しない場合には、C コンパイラのエンコードオプションで、対象ファイルのエンコードを正しく指定する必要かあります。
③	C ソースに対して、C ヘッダ(ユーザ作成、及び、SDK 提供)を適応し、オブジェクトを生成します。(ccrx) コンパイラには、各種オプション指定があります。
④	複数のオブジェクトから、ロードモジュール、もしくは、ライブラリを作成します。(rlink) ロードモジュールを作成する場合は、参照するライブラリ(ユーザ作成、及び、SDK 提供)を指定します。 リンケージエディタには、各種オプションがあります。
⑤	LMWIN-FLINK ダウンロード形式への変換(APCNVY)
⑥	LMWIN-FLINK でアプリ転送
⑦	DT-970 で動作確認

6.3. RX Cコンパイラ環境

DT-970 搭載 CPU 用の「RENESAS C コンパイラ」は、「DT-970 基本開発環境」に含まれています。「DT-970 基本開発環境」を標準設定でインストールすると、下記フォルダに各種ファイルがインストールされます。

32 ビット OS: C:\Program Files\Windows CE Tools\wce500\DT-970
64 ビット OS: C:\Program Files (x86)\Windows CE Tools\wce500\DT-970

フォルダ	モジュール	内容
bin	ccrx.exe	C コンパイラ
	rlink.exe	リンケージエディタ
include	*.h	C ヘッダファイル (*1)
lib	*.lib	C ライブラリ (*1)

*1...RENESAS C コンパイラで提供する C 標準関数に関するものです。
アプリケーションのビルドには、後述の「DT-970 SDK C ヘッダ/ライブラリ」も必要となります。

6.4. DT-970 開発環境

「DT-970 基本開発環境」には、前述、「RENESAS C コンパイラ」以外に各種ツールが含まれています。「RENESAS C コンパイラ」以外の各種ツールは、標準設定でインストールすると、下記フォルダにインストールされます。

32 ビット OS: C:\Program Files\Windows CE Tools\wce500\DT-970
 64 ビット OS: C:\Program Files (x86)\Windows CE Tools\wce500\DT-970

バッチビルドでは、上記フォルダ下の下記モジュールを利用します。

- 文字列の 16 進数変換
- オブジェクト変換
- SDK ヘッダ
- SDK ライブラリ

フォルダ	モジュール	内容
bin	KJ_CNVRT.EXE	文字列の 16 進数変換
	APCNVY.EXE	LMWIN-FLINK ダウンロード形式(.LOD)変換
obj	AP_START.OBJ	AP 初期化モジュール
	AP_START1.OBJ	AP 初期化モジュール DT-930 互換表示モード1
	AP_START2.OBJ	AP 初期化モジュール DT-930 互換表示モード2
include\%rx600	*.h	DT-970 SDK C ヘッダファイル
lib\%rx600	*.lib	DT-970 SDK C ライブラリ

上記以外には、下記モジュールも格納されています。

フォルダ	モジュール	内容
bin	Fontcnvw.exe	フォント変換ツール(*1)
	AppConvertor.exe	DT-930 アプリ移行ツール(*2)
include\%armv4i	*.h	Emulator 用 C ヘッダファイル
lib\%armv4i	*.lib	Emulator 用 C ライブラリ

*1...初期 Font は下記フォルダに配置されます。

*2... ツール直下サブフォルダ JA, EN 下に各種設定ファイルがあります。

Windows 種別	格納フォルダ
XP,2003	\Documents and Settings\All Users\Application Data\CASIO\Fontcnvw\Font
7,2008	\Users\All Users\CASIO\Fontcnvw\Font

6.5. 動作環境設定

「RENESAS C コンパイラ」及び「DT-970 Export SDK」で用意されているツール、ヘッダ、ライブラリを利用するために、環境変数の指定が必要となります。

それぞれが標準設定でインストールされている場合

```
32 ビット OS: C:¥Program Files¥Windows CE Tools¥wce500¥DT-970
64 ビット OS: C:¥Program Files (x86)¥Windows CE Tools¥wce500¥DT-970
```

下記設定を実施して下さい。

6.5.1. PATH設定

「RENESAS C コンパイラ」の `ccrx`, `rlink`、及び、「DT-970 Export SDK」の `KJ_CNVRT`, `APCNVY` を利用可能とするために `PATH` に以下のパスを追加して下さい。

```
SET PATH=%PATH%;
C:¥Program Files¥Windows CE Tools¥wce500¥DT-970¥bin
```

※ 実際の指定は改行なしでの指定となります。

※ 上記は 32 ビット OS の場合のパスです。

6.6. コード／オブジェクト変換

KJ_CNVRT

漢字変換ツールは、変換元プログラムの漢字コードを 16 進数値データに変換して、結果を変換先ファイルに出力します。

変換元(foo.c)から、変換先(bar.c)を生成する場合は、以下の記述となります。

```
KJ_CNVRT foo.c bar.c
```

ccrx

ccrx はコンパイルドライバの起動コマンドです。

本コマンド起動により、コンパイル、アセンブル、リンクを行うことができます。

入力ファイルの拡張子が「.s」「.src」「.S」「.SRC」のいずれかである場合、コンパイラはそのファイルをアセンブリ言語ファイル(.src, .s)と解釈して、アセンブラを起動します。

これら以外の拡張子のファイルは、C/C++ 言語ソースファイル(.c, .cpp)としてコンパイルします。

C 言語ソースファイルから、リロケータブルファイル(.obj)を生成する場合は、以下のオプション指定が必要です。

No	項目	内容
1	CPU 選択	-cpu=rx600
2	エンディアン選択	-endian=big
3	ビットフィールド	-bit_order=left
4	符号指定無ビットフィールド型	-signed_bitfield
5	符号指定無 char 型	-signed_char
6	double 型サイズ	-dbl_size=8
7	コンパイルエラーレベル (※1)	-change_message=error=20223
8	リロケータブルファイル出力	-output=obj

※1 関数のプロトタイプ宣言がない場合、エラーとして処理します。

C 言語ソースファイル(foo.c)から、リロケータブルファイル(foo.obj)を生成する場合は、以下の記述となります。

```
ccrx -cpu=rx600 -endian=big -bit_order=left -signed_bitfield  
-signed_char -dbl_size=8 -change_message=error=20223 -output=obj foo.c
```


rlink

rlink は、最適化リンケージエディタの起動コマンドです。
リンク処理だけでなく、以下に挙げる機能も含んでいます。

- リロケータブルファイル結合時の最適化
- ライブラリファイルの作成や編集

リロケータブルファイル(foo.obj、bar.obj)から、ライブラリ(sample.lib)をリンクした上で、モトローラ S 形式ファイル(baz.mot)を生成する場合は、以下の記述となります。

```
rlink -subcommand=cmd.sub -list -SHow=SYmbol
```

※ -list -SHow=SYmbol: マップファイル(baz.map)にシンボル名一覧を出力します

cmd.sub

```
form stype
input AP_START.OBJ,foo.obj,bar.obj
output baz.mot
library sample.lib
rom D=R, D_1=R_1, D_2=R_2
start P,C*,D*,L,W*,B*/5335030
```

リロケータブルファイル(foo.obj、bar.obj)から、ライブラリファイル(baz.lib)を生成する場合は、以下の記述となります。

```
rlink -subcommand=cmd.sub
```

cmd.sub

```
form library=u
input foo.obj,bar.obj
output baz.lib
```

APCNVY

APCNVY を利用して、LMWIN-FLINK ダウンロード形式(*.LOD)に変換します。

モトローラ S 形式ファイル(foo.mot)から、LMWIN-FLINK ダウンロードファイル(foo.lod)を生成する場合は、以下の記述となります。

```
apcnvy foo.mot foo.lod
```

7. 付録

7.1. 注意事項

1) 変数の初期化

条件分岐等で初期化していない変数を使用すると、変数が不定なため、正しく判定できなくなります。変数は必ず初期化して使用してください。

```
ER FuncNG()
{
  ER ret; // 未初期化変数
  ER temp;
  temp = MyFunc1();
  if( temp == 0 )
  {
    ret = MyFunc2();
  }
  if( ret == E_NG )
  {
    // 省略
  }
  else
  {
    // 省略
  }
  return ret;
}
```

```
ER FuncOK()
{
  ER ret = 0; // 初期化
  ER temp;
  temp = MyFunc1();
  if( temp == 0 )
  {
    ret = MyFunc2();
  }
  if( ret == E_NG )
  {
    // 省略
  }
  else
  {
    // 省略
  }
  return ret;
}
```

2) ループの処理時間

ループウェイト(for 文などでの繰り返し処理によるウェイト)は、CPU 性能などの動作環境で消費時間は異なります。DT-930 と DT-970 では CPU 性能が異なるため、DT-930 のソースをそのまま DT-970 で利用すると、ウェイトされる時間は異なります。このため、期待する時間ウェイトさせるためにはソース修正が必要となります。

3) 互換表示モード

DT-930 互換表示モード 2 を使用した場合、グラフィック描画には互換性はないため表示位置の修正が必要となります。

互換表示モードについては、「デバイス制御ライブラリファレンスマニュアル」を参照してください。

4) sprintf 関数のゼロパディング

printf 系書式のフィールド長先頭に 0 を記載して文字列を整形した場合(例:"%05s")、DT-930 では、空白部分は 0 埋めされていましたが、DT-970 では、空白部分はそのままとなります。0 埋めをするには別途アプリケーションの修正が必要となります。

(数値に対する書式 "%05d" の場合は、DT-930, DT-970 ともに 0 埋めされます。)

5) stddef.h のインクルード

DT-930 では、stdio.h または stdlib.h の中で stddef.h をインクルードしていたため、これらをインクルードしていれば、明示的に stddef.h をインクルードする必要がありませんでした。

しかし、DT-970 では stddef.h の定義内容を利用する場合は、明示的に stddef.h をインクルードする必要があります。

6) 関数のプロトタイプ宣言

プロトタイプ宣言せずに関数呼び出すと、ビルド時にエラーとなります。関数を使用する際は必ずプロトタイプ宣言をしてください。

※ DT-970 基本開発キット Ver.1.02 以前は、プロトタイプ宣言しなくてもビルドエラーにならず、第5引数以降の値が正しく参照されませんでした。

<NG 例>

```
#include <stdlib.h>
#include <itron.h>
#include <CMNDEF.H>
#include <bios1mac.h>
#include <string.h>
#include <ctype.h>

void ap_start( void )
{
    ER ret = 0;
    ret = MyFunc1(1, 2, 3, 4, 5);
}
```

上記 NG 例では、ビルド時に下記のエラーが表示されます。

出力

```
コンパイル中 ...
RX Family C/C++ Compiler V2.00.01 [10 May 2013]
RX Family Assembler V2.00.00 [15 Feb 2013]
Renesas Optimizing Linker W1.00.00 [12 Dec 2012]
Copyright (C) 2011, 2013 Renesas Electronics Corporation
Copyright (C) 2003-2012 University of Illinois at Urbana-Champaign.
All rights reserved.
SAMPLE.C:
C:#Users#casio#Sample#SAMPLE.C(27):E0520223:Function "MyFunc1" declared implicitly
```

<OK 例>

```
#include <stdlib.h>
#include <itron.h>
#include <CMNDEF.H>
#include <bios1mac.h>
#include <string.h>
#include <ctype.h>

extern ER MyFunc1(H a1, H a2, H a3, H a4, H a5); // MyFunc 関数のプロトタイプ宣言

void ap_start( void )
{
    ER ret = 0;
    ret = MyFunc1(1, 2, 3, 4, 5);
}
```

※ 上記は extern によるプロトタイプ宣言の例ですが、ヘッダファイルのインクルードでも構いません。

7) pwr_vibrator 関数による振動の継続

DT-930 では pwr_vibrator 関数は停止するまで振動し続けていましたが、DT-970 では振動開始後約 4 秒で自動的に停止されます。

振動を継続するためには、再度 pwr_vibrator 関数で振動を開始してください。

pwr_vibrator 関数については、「デバイス制御ライブラリ リファレンスマニュアル」を参照してください。

8) カシオ IR インターフェイスの廃止

DT-970 では、カシオ Ir インターフェイスはサポートしていません。DT-970 で IrDA を使用するアプリケーションを作成する場合は、IrCOMM プロトコルの IrDA 制御関数 (Ir_xxx) を使用してください。この場合、相手先は IrCOMM に対応している機器に変更する必要があります。

IrDA 制御関数については、「デバイス制御ライブラリ リファレンスマニュアル」を参照してください。

9) トリガーキーの検出処理

DT-970 では、OBR 使用時のトリガーキー、L/R キーの動作が DT-930 と異なります。DT-930 では、トリガーキーを押下したままの状態でも OBR_open 関数を実行すると、レーザが発光し読取動作を開始します。DT-970 では、トリガーキーを押下したままの状態でも OBR_open 関数を実行しても、レーザを発光しません。読取動作を開始するためには、OBR_open 関数実行後にトリガーキーを押下してください。

また、DT-930 では、L/R キーが OBR キーに登録されていると、OBR_close 関数実行時にキーロールオーバーが無効になります。DT-970 では、L/R キーは常にキーロールオーバーが有効になります。

7.2. パッチ

PATCH001.LOD および PATCH970.LOD は「DT-970 Export SDK」に含まれていて、既定のインストールフォルダにインストールした場合は、下記フォルダに存在します。

32 ビット OS: C:\Program Files\Windows CE Tools\wce500\DT-970\Patch

64 ビット OS: C:\Program Files (x86)\Windows CE Tools\wce500\DT-970\Patch

7.3. サンプルアプリケーション

サンプルアプリケーションは、「DT-970 Export SDK」に含まれていて、既定のインストールフォルダにインストールした場合は、下記フォルダに存在します。

32 ビット OS: C:\Program Files\Windows CE Tools\wce500\DT-970\Sample

64 ビット OS: C:\Program Files (x86)\Windows CE Tools\wce500\DT-970\Sample

もしくは、「DT-970 基本開発キット」の CD の下記フォルダに存在します。

¥Sample

サンプルアプリケーションを使用する場合は、お使いの PC の任意の場所へコピーしてください。

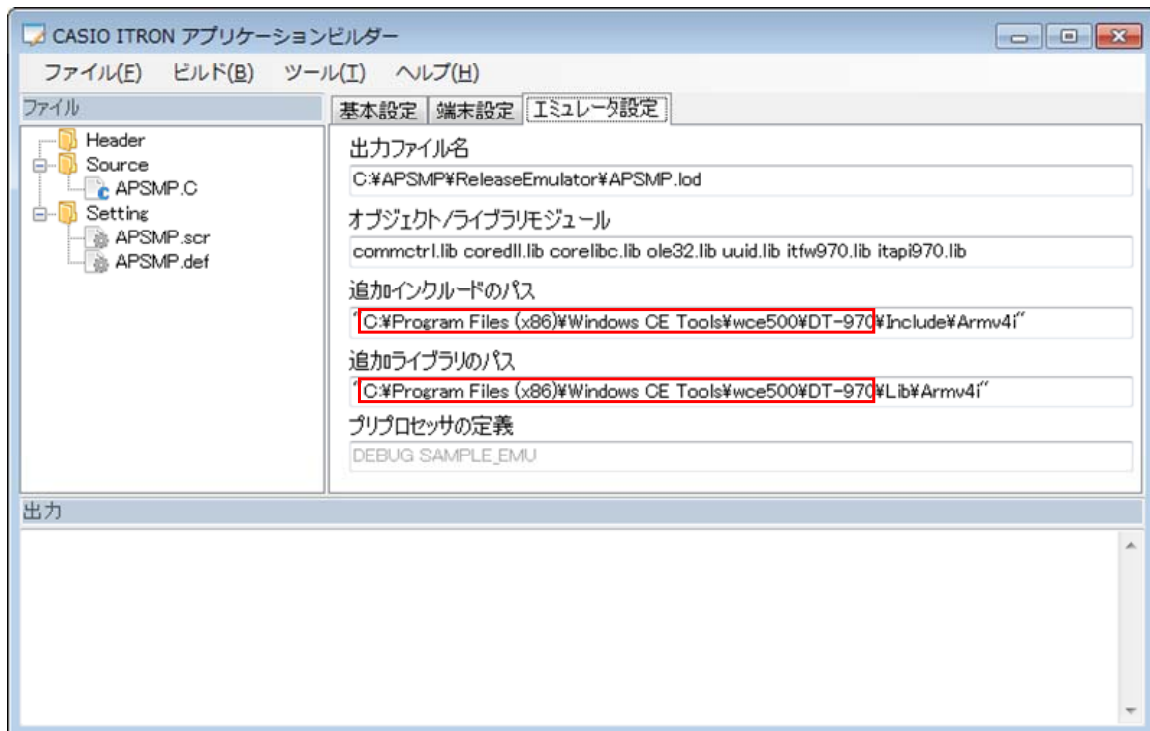
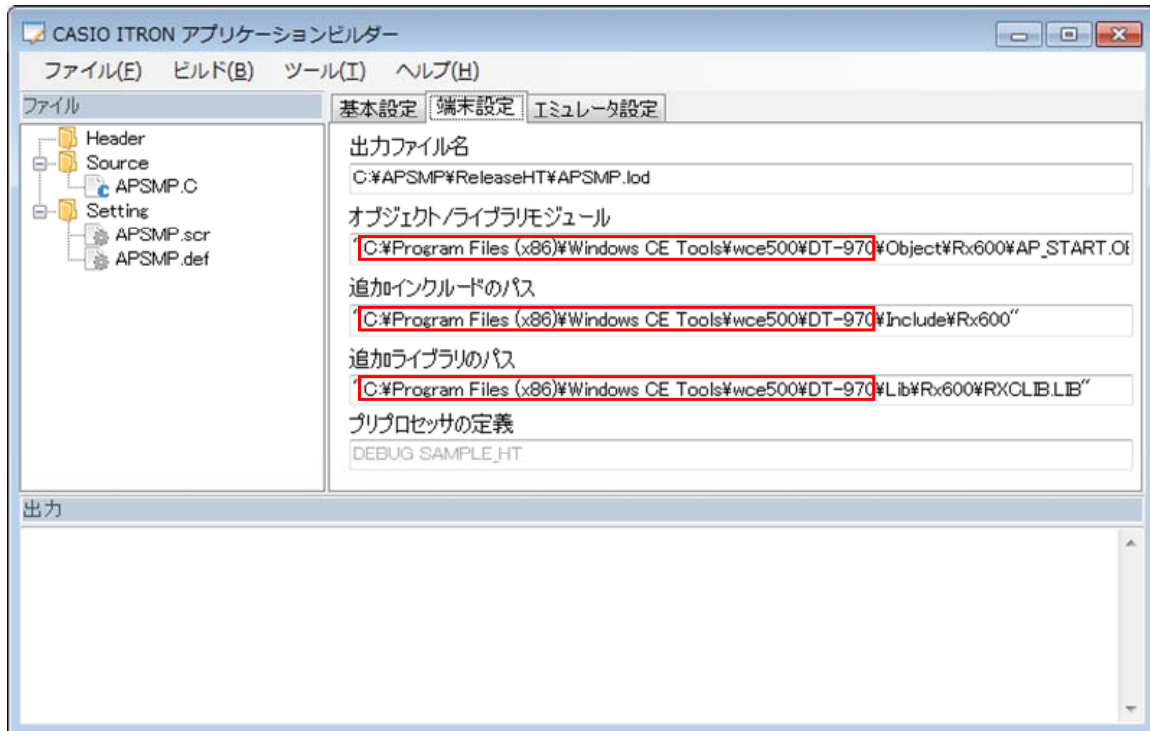
使用する OS に応じてプロジェクトファイルの名称が異なります。

このとき、書き込み権限のないフォルダでビルドを行うと、エラーが発生します。

32 ビット OS: (サンプルアプリケーション名).ciprj

64 ビット OS: (サンプルアプリケーション名)64.ciprj

また、サンプルアプリケーションをビルドするとき、「DT-970 Export SDK」を既定のインストールフォルダから変更した場合は、下記の赤枠内を変更したフォルダに書きなおしてから、ビルドを実行してください。



カシオ計算機お問い合わせ窓口

製品に関する最新情報

製品サポートサイト（カシオペア・ハンディターミナル）

<http://casio.jp/support/ht/>

製品の取扱い方法のお問い合わせ

情報機器コールセンター



0570-022066

市内通話料金でご利用いただけます。

携帯電話・PHS 等をご利用の場合、**042-503-7241**

カシオ計算機株式会社

〒151-8543 東京都渋谷区本町 1-6-2

TEL 03-5334-4638(代)