応用記述

AVR1614 : ウィジェット ツールキット – 始める前に

Atmel マイクロ コントローラ

前提条件

- 必要な知識
  - マイクロ コントローラとCプログラミング言語の基礎知識
- ソフトウェア必要条件
  - Atmel® Studio 6
  - Atmelソフトウェア枠組み3.3.0またはそれ以降
- ハードウェア前提条件
  - mXT143E Xplain評価基板
  - Xplain系統MCU評価基板
  - 書き込み器/デバッガ
    - Atmel AVR® JTAGICE3
    - Atmel AVR Dragon™
    - Atmel AVR JTAGICEmk II
    - Atmel AVR ONE!
- 予想完了時間
  - 2時間

序説

この資料の狙いはAtmelソフトウェア枠組みと共に配布されるウィンドウ システムとウィジェット ツールキット(WTK)を紹介することです。

この応用記述は以下を通して行う練習として構成されます。

- 画像使用者インターフェース(GUI:Graphical User Interface)を作るために画面上で画像ウィジェットを構成設定する基礎
- 使用者がウィジェットで相互に影響した時に還元する方法
- 画面上に独自の画像要素を描画する方法

Rev. 8300B-07/2012, 8300BJ1-03/2014

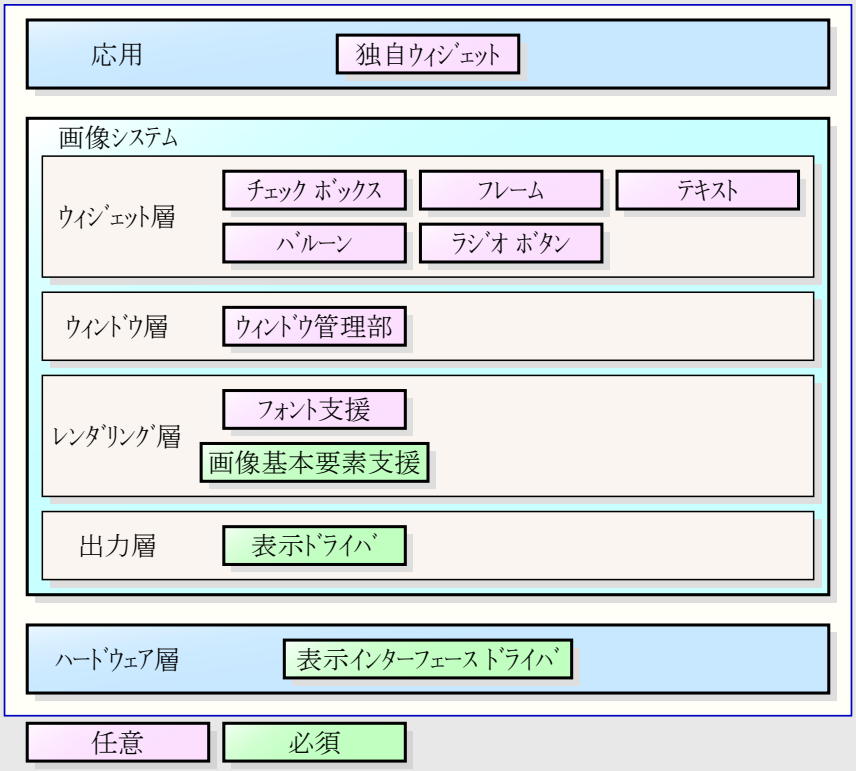
目次

1. ウィンドウ システムとウィジェット ツールキットの紹介

1.1. 概要

画像表示システムは画像応用で使用することができる多数のドライバ層から成ります。後続する図は利用可能な層が図1-1.で図解されます。

図1-1. 画像表示システムの階層



1.2. ウィンドウ システム

ウィンドウ システムは2次元ウィンドウを構成するための部署です。ウィンドウは各ウィンドウが1つの親と多数の子を持つことができる樹状構造で構成されます。ウィンドウ システムは影響を及ぼされるウィンドウの事象処理部によって待ち行列にされて処理されることも事象に許します。

表1-1.はウィンドウを定義する特性を示します。

表1-1. ウィンドウ特性

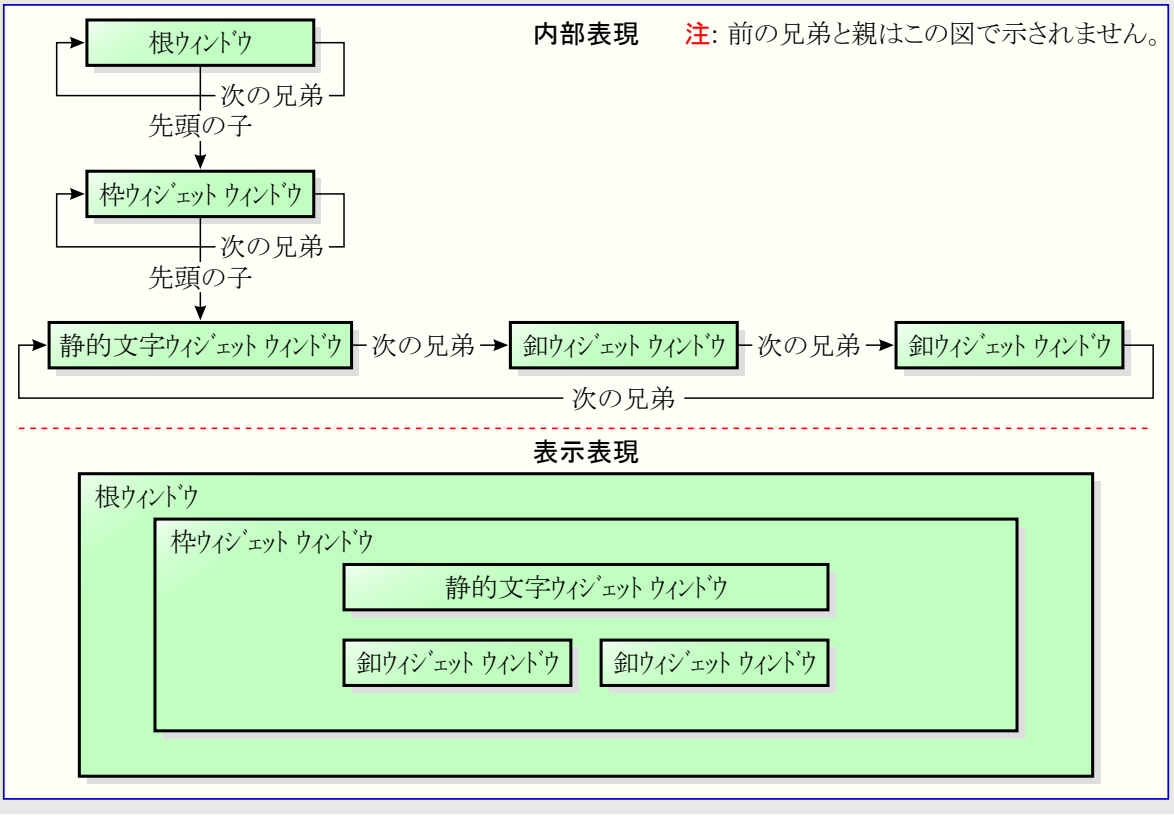
特性	説明
位置と領域/大きさ	画面上の位置とウィンドウの大きさ
親と子の関連	ウィンドウはそれらが1つの親と多数の兄弟、多数の子を持つことができる樹状構造で構成されます。
可視性	ウィンドウは画面上で可視/割り当て、または不可視/未割り当てにすることができます。
背景	背景ビットマップはデータまたはプログラムのどちらかのメモリに格納されたビットマップまたは無地単色のどちらにもできます。
ウィンドウ動きフラグ	現在実装されている動きは次の通りです。 ・ 押下での立ち上げ：ウィンドウ内でポインタ事象が起きた時に先頭の子としてウィンドウを立ち上げます。 ・ 親を再描画：ウィンドウを描画する時に親を再描画します(透過ウィンドウに有用)。
事象処理部	ウィンドウに関する事象を処理する関数へのポインタ

ｳｨﾝﾄﾞｳ ｼｽﾃﾑはそれ自身またはｳｨｼﾞｪｯﾄ ﾂｰﾙｷｯﾄと共に使用することができます。

図1-2.はいくつかのｳｨｼﾞｪｯﾄ例を表現するのに使用されるｳｨﾝﾄﾞｳ間の関連を示します。この図は先頭の子(top_child)と次の兄弟(next_sibling)のﾎﾟｲﾝﾀを示します。親と、単に逆の方向で位置を指示する、前の兄弟(prev_sibling)のﾎﾟｲﾝﾀは図を簡単にするために省略されています。

この図は位置と大きさがどう影響を及ぼして表示器上でｳｨﾝﾄﾞｳがどう表現され得るかも図解します。

図1-2. ｳｨﾝﾄﾞｳ表現



1.3. 事象処理

ｳｨﾝﾄﾞｳ ｼｽﾃﾑは使用者またはｼｽﾃﾑ相互作用が起きる時に利用可能なｳｨﾝﾄﾞｳを更新するために事象を使用します。ｳｨﾝﾄﾞｳ ｼｽﾃﾑに関する全体事象処理部は影響を及ぼされるｳｨﾝﾄﾞｳの各々に対する形態設定された事象処理部を呼びます。表1-2.はｳｨﾝﾄﾞｳ ｼｽﾃﾑによって発行され得る事象の形式を記述します。

表1-2. ｳｨﾝﾄﾞｳ用事象形式

事象名	説明
ﾎﾟｲﾝﾀ事象	使用者がﾏｳｽをｸﾘｯｸ、移動、開放、またはｳｨﾝﾄﾞｳ内側の画面を触った。
ｷｰﾎﾞｰﾄﾞ事象	使用者がｷｰﾎﾞｰﾄﾞまたはｷｰﾊﾟｯﾄﾞでｷｰを押した。
立ち上げ事象	ｳｨﾝﾄﾞｳがそれの親ｳｨﾝﾄﾞｳの内側で先頭に立ち上げられた。
立ち下げ事象	別のｳｨﾝﾄﾞｳが先頭に立ち上げられたので、このｳｨﾝﾄﾞｳはもはや先頭の子ではありません。
ﾌｫｰｶｽ取得事象	ｳｨﾝﾄﾞｳがｷｰﾎﾞｰﾄﾞのﾌｫｰｶｽ(割り当て)を得ました。
ﾌｫｰｶｽ喪失事象	ｳｨﾝﾄﾞｳがｷｰﾎﾞｰﾄﾞのﾌｫｰｶｽ(割り当て)を失いました。
描画事象	(ｳｨﾝﾄﾞｳ背景が描画された後の)ｳｨﾝﾄﾞｳ描画要求
属性事象	(背景、位置などのような)ｳｨﾝﾄﾞｳの属性が変更されました。
消失事象	ｳｨﾝﾄﾞｳ ｼｽﾃﾑそれ自身によって処理されないｳｨﾝﾄﾞｳに割り当てられた全ﾒﾓﾘを開放する要求。

1.3.1. 描画事象

描画事象はｳｨﾝﾄﾞｳを描画する時に使用されます。ｳｨﾝﾄﾞｳ描画時、以下が起こります。

1. 親はWIN_BEHAVIOR_REDRAW_PARENT属性が設定されている場合に描画されます。
2. ｳｨﾝﾄﾞｳの背景が描画されます。
3. ｳｨﾝﾄﾞｳ/ｳｨｼﾞｪｯﾄを描画するためにｳｨﾝﾄﾞｳに関する事象処理部が呼ばれます。
4. 子は底の子で始まり先頭の子で終わるように描画されます。

1.4. ウィジェット ツールキット

ウィジェット ツールキットは画像表示で使用するための多くの共通ウィジェットを実装します。ツールキットはウィジェットを実装するためにウィンドウ システムと表示ドライバからの機能を使用します。ウィジェットは使用者に還元を示すことができる、画面上に配置される画像表示要素として定義されます。多くのウィジェットはマウスまたは接触画面によって操作することもできます。

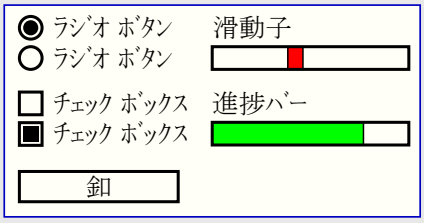
例のウィジェットは釦ウィジェットです。このウィジェットは画像使用者インターフェース(GUI)で一般的です。このウィジェットは"Initiale Launch Sequence"のようないくつかの文字を表示することができ、使用者がウィジェット上でクリックする時にそれが押されたことを示すために色を変えます。使用者が押下を開放すると、釦上の文字によって示されるような活動を始めます。

表1-3.はウィジェット ツールキットで配給される、予め実装されるウィジェット形式の一覧を一覧にし、一方図1-3.はそれらが表示器上に描画される時にウィジェットのいくつかがどう見え得るかを示します。

表1-3. 実装されるウィジェット形式

ウィジェット名	代表的な使用法/使用例
基礎枠	他のウィジェット用の主ウィンドウ
釦	使用者からの接触/マウス押下の記録
チェック ボックス	スイッチのON(チェック)またはOFF(非チェック)の機能を使用者に許します。
進捗バー	作業の進捗または(車の燃料または速度の水準のような)変数の現在の状態を画像的に表現するのに使用
ラジオ ボタン	利用可能な多数の機能の内から1つを選ぶのを使用者に許します。
滑動子	マウスまたは接触入力で特別な入力値を設定することを使用者に許します。
ラベル	画面に文字を表示

図1-2. ウィジェット描画例



1.5. ウィンドウ システム事象待ち行列

前で記述されたように、ウィンドウ システムは内部的な事象待ち行列を持ちます。処理されるべき使用者入力と起こるウィンドウ システムの更新のために、この待ち行列は規則的に処理されなければなりません。本来、ウィジェット ツールキットがウィンドウ システムに基づくため、ウィジェットは規則的に処理されるべきウィンドウ システムの事象待ち行列も必要です。

加えて、ウィンドウ システムは(接触感知器や押し釦のような)使用者入力ドライバを自動的にインターフェースせず、故に使用者入力の低位処理とウィンドウ システムの事象待ち行列で対応する事象を生成するのは応用の責任です。

このような使用者入力処理と事象待ち行列処理を行う方法の例については練習プロジェクトのmain関数内の作業繰り返しを参照してください。

2. 練習課題

この実践作業はウィンドウ システムとウィジェット ツールキットで始める方法を網羅します。この練習の目的はコードに精通して直ぐにあなた自身のGUI応用を作成することを学ぶことです。

練習は以下の4つの課題から成ります。

課題1. 始める前に

この課題はAtmel Studio内の新しいウィジェット ツールキット プロジェクトを開始する方法と、あなたの基板で走行する応用を得る方法を示します。

課題2. 釦追加

この課題は釦ウィジェットを追加する方法を示します。

課題3. 基礎枠追加

この課題は画面での描画に使用することができる基礎枠ウィジェットを追加する方法を示します。

課題4. 全てを一緒に接続

この課題は操作されて画面で表示されるべき計数器の値を許すために釦と枠を共に接続する方法を示します。

ご幸運を!

2.1. 課題1: 始める前に

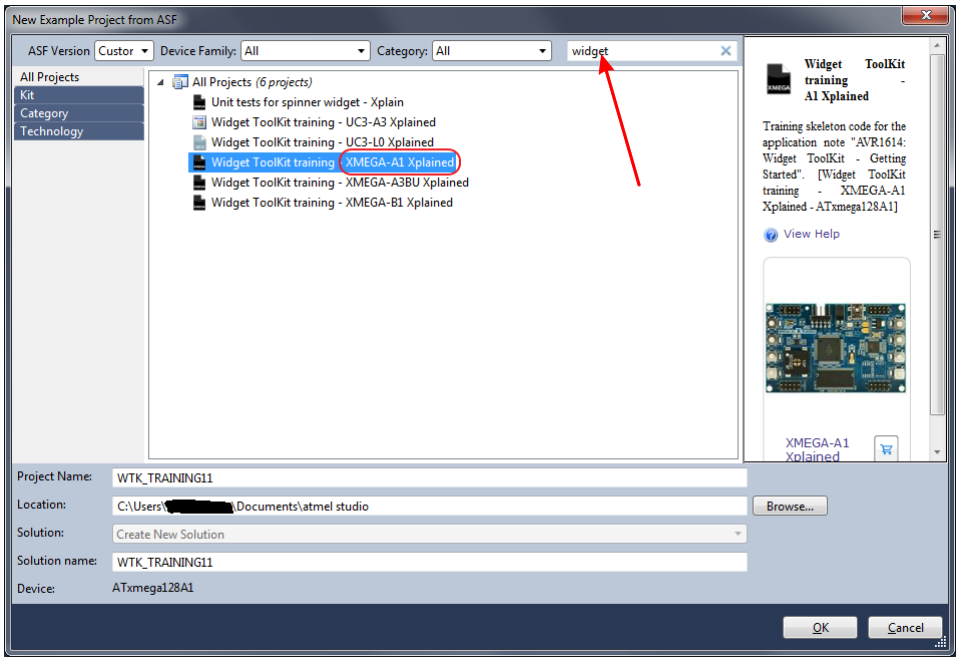
画像使用者ｲﾝﾀｰﾌｪｰｽを始めましょう。

この課題の目的はあなたが以下の方法を学ぶことです。

- 評価基板とﾃﾞﾊﾞｯｶﾞの接続
- ｺｰﾄﾞのｺﾝﾊﾟｲﾙ
- Xplain基板へのｺｰﾄﾞの格納

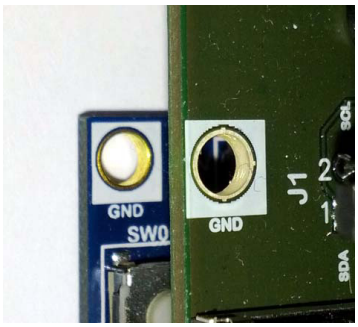
1. ｺﾝﾋﾟｭｰﾀにAtmel StudioとAtmelｿﾌﾄｳｪｱ枠組みの正しい版がｲﾝｽﾄｰﾙされていることを確実にしてください。
2. Atmel Studioを開始して図2-1.で示されるように新しいｳｨｼﾞｪｯﾄ ﾂｰﾙｷｯﾄ ﾌﾟﾛｼﾞｪｸﾄを作成してください。
   1. ″File″⇒″New″⇒″Example project from ASF ...″をｸﾘｯｸしてください。
   2. 検索領域で″widget″を入力してください。
   3. Xplain基板用の練習ﾌﾟﾛｼﾞｪｸﾄを選択して″OK″をｸﾘｯｸしてください。

図2-1. ASFﾌﾟﾛｼﾞｪｸﾄ ｳｨｻﾞｰﾄﾞでのｳｨｼﾞｪｯﾄ ﾂｰﾙｷｯﾄ練習ﾌﾟﾛｼﾞｪｸﾄの作成



3. mXT143E XplainをXplain MCU基板に接続してください。図2-2.で示されるように、それらの周囲に白い正方形を持つﾄﾞﾘﾙ穴の整列によって正しい向きを保証してください。

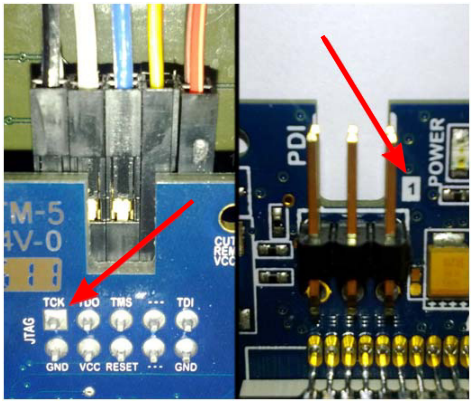
図2-2. Xplain基板の正しい整列



4. mXT143E XplainのSPIｽｲｯﾁを正しい位置(XMEGA以外″Native SPI″を使用)に設定してください。
5. 書き込み器/ﾃﾞﾊﾞｯｶﾞをXplain MCU基板上のJTAG(10ﾋﾟﾝ)またはPDI(6ﾋﾟﾝ)のﾍｯﾀﾞに接続してください。

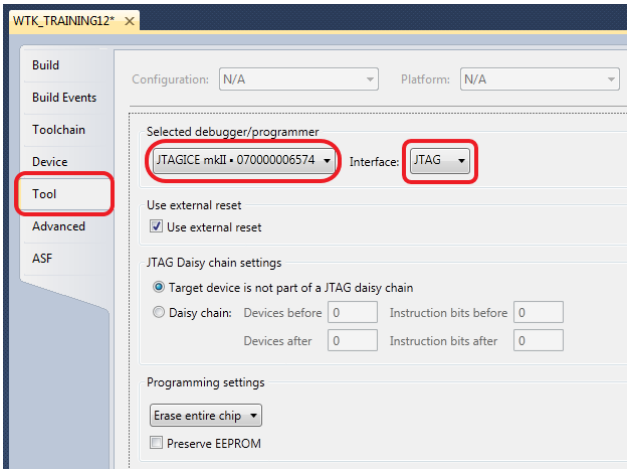
- Atmel JTAICE3は1つの方法だけが合致します。ﾍｯﾀﾞと同じﾋﾟﾝ数を持つ探針を使用することを確実にしてください。
- Atmel AVR Dragonは基板へ接続するためのﾘﾎﾞﾝ ｹｰﾌﾞﾙが必要です。
- Atmel AVR JTAGICEmk ⅡとAVR ONE!はﾊﾞﾗ線探針または拡張器が必要です。
- 図2-3.はﾍｯﾀﾞのどの位置が1番ﾋﾟﾝかを示します。JTAGまたはPDIのﾍｯﾀﾞへの特定のﾂｰﾙ接続の詳細についてはAtmel Studioのﾍﾙﾌﾟを参照してください。

図2-3. JTAGとPDIのﾍｯﾀﾞでの1番ﾋﾟﾝ位置



6. コンピュータから書き込み器/デバッガへUSBケーブルを接続してそれに通電してください。未だインストールされていなければUSBドライバをインストールしてください。
7. コンピュータからXplain基板にUSBケーブルを接続してください。この練習に関してはXplain用のドライバをインストールする必要はありません。この時点でWindows®がドライバのインストールを問う場合は"キャンセル"を押してください。
8. ALT+F7押下または"Project"⇒"Properties"をクリックすることによってプロジェクト特性を開き、そして**図2-4.**で示される"Tool"タブであなたの書き込み器/デバッガとインターフェースを選んでください。

図2-4. Atmel Studioプロジェクト用ツール設定



9. 応用を構築してデバッグ作業を開始するために、F5押下または"Debug"⇒"Continue"をクリックしてください。
10. 応用を使用してそれを動かす方法を理解するためにソースを調べてください。実行を一時停止するのにCTRL+F5または"Debug"⇒"Break All"を押し、その後に一時停止解除のために9.段階を繰り返すことができます。

応用は何をしますか?

さて、背景色をあなたが好きな色に変更しましょう。背景色はapp_widget.cソース ファイルに於いてGFX_COLORマクロを用いてAPP_BACKGROUND_COLORマクロで定義されます。

11. プロジェクトのオンライン資料をへ行くために"Help"⇒"View ASF Example Help"をクリックしてください。
12. このマクロに関するパラメータが何かを知るために"app_widget.c"の資料へ行って"GFX_COLOR"をクリックしてください。
13. あなたの好きな色に背景色を変更し、再コンパイルして走らせてください。

2.2. 課題2: 釦追加

さあ、コードにいくつかの変更をし、クリックすることができる釦を追加しましょう。

この課題の目的はあなたが以下の方法を知ることです。

- 釦作成
- クリック事象処理

1. app_widget.cでapp_widget_launch関数を見つけ出してください。別のウィジェットが追加される前に違うウィジェットがどう形態設定されるかを理解してみてください。
2. "//! ¥todo Add code to set up button here."と言う行の下に以下のコードを複写または入力してください。

```
area.pos.x = 10;
area.pos.y = 150;
area.size.x = 90;
area.size.y = 40;

btn = wtk_button_create(parent, &area, "Click", (win_command_t)BUTTON_ID);
if (!btn) {
  goto error_widget;
}
win_show(wtk_button_as_child(btn));
```

3. コードをコンパイルして走らせてください。

上のコードは何をしますか?

4. 各釦押下に対して釦押下を処理してcounter変数を増すために"widget_frame_commad_handler"へコードを追加してください。
5. コードをコンパイルしてそれが動くことを確かめてください。

2.3. 課題3: 基礎枠追加

さあ、描画できるウィジェットを作りましょう。基礎枠ウィジェットは画面を描画するのに使用することができる呼び戻し関数を持ちます。この呼び戻し関数の手助けとで、簡単な方法で使用者に視覚的な還元を実装することが可能です。最初にウィジェットを作成しましょう。

この課題の目的はあなたが以下の方法を知ることです。

- 基礎枠作成
- 基礎枠描画処理部構成設定

1. app_widget.cでapp_widget_launch関数を見つけ出してください。更にｺｰﾄﾞへ基礎枠を追加しましょう。
2. "//! ¥todo Add code to set up basic frame here."と言う行の下に以下のｺｰﾄﾞを複写または入力してください。

```
area.pos.x += area.size.x + 40;

sub_frame_background.type = GFX_BITMAP_SOLID;
sub_frame_background.data.color = GFX_COLOR(127, 0, 0);

sub_frame = wtk_basic_frame_create(parent, &area, &sub_frame_background,
                                   sub_frame_draw_handler, NULL, NULL);
if (!sub_frame) {
    goto error_widget;
}
win_show(wtk_basic_frame_as_child(sub_frame));
```

3. ｺｰﾄﾞをｺﾝﾊﾟｲﾙして走らせてください。

上のｺｰﾄﾞは何をしますか?

4. ｳｨｼﾞｪｯﾄの背景の色を違う色に変えてください。
5. 再ｺﾝﾊﾟｲﾙして走らせて画面上で色が変更されたことを確かめてください。

2.4. 課題4: 全てを一緒に接続

さて、私たちは使用者に視覚的な還元を与えるために描画することができる基礎枠と入力を獲得する釦を持ちます。前の課題で作成した枠上の釦のｸﾘｯｸ回数を表示しましょう。

この課題の目的はあなたが以下の方法を知ることです。

- 描画と画面上の文字形式
- あなた自身の基礎枠描画処理部作成

1. app_widget.cでsub_frame_draw_handler関数を見つけ出してください。
2. "¥todo Add code here to draw text on screen."を含む行を見つけてください。

画面上で文字を描画するにはgfx_draw_string関数の使用が必要です。これに関し、それがどう使用されるかを知る必要があります。この関数に関する資料を探しましょう。

3. 課題1でと同じ方法でｵﾝﾗｲﾝ ﾌﾟﾛｼﾞｪｸﾄ資料の主頁を開いてください。
4. これがﾌｫﾝﾄに関連する画像ｼｽﾃﾑ内の関数であるため、"Graphical display system"⇒"Font suport"下でそれの資料を見つけ出してください。
5. それのﾊﾟﾗﾒｰﾀの内容を読むために"gfx_draw_string"をｸﾘｯｸしてください。
6. &sysfontでのﾌｫﾝﾄを使用して位置X:clip⇒origin.xとY:clip⇒origin.yにbuffer内に格納された文と、GFX_COLORﾏｸﾛ経由での好みの色を表示するのにgfx_draw_string関数を用いるｺｰﾄﾞを追加してください。
7. 再ｺﾝﾊﾟｲﾙしてｺｰﾄﾞを走らせてください。

釦をｸﾘｯｸした時に何故文字が更新されないのでしょうか?。

さて、各釦の押下で表示される値を増加(+1)することによって使用者相互作用を追加します。そのような押下の各々は主ｳｨｼﾞｪｯﾄ枠によって扱われ得る命令事象を生成します。

8. widget_frame_command_handler関数を見つけ出してください。
9. "¥todo Add code here to handle button press(釦押下を処理するためのｺｰﾄﾞをここに追加)"を含む行を見つけ出してください。
10. 最初に、計数を増加するためのｺｰﾄﾞを追加してください。

新しい値を表示するために、副枠ｳｨｼﾞｪｯﾄ ｳｨﾝﾄﾞｳの再描画を起動しなければなりません。

11. sub_frame(副枠)のｳｨﾝﾄﾞｳへのﾎﾟｲﾝﾀを得るのにwtk_basic_frame_as_child関数を、それの再描画を発行するのにwin_redraw関数を使用してください。これらの関数の資料は"Window System"単位部内と、ｵﾝﾗｲﾝ ﾌﾟﾛｼﾞｪｸﾄ資料内の"Widget toolkit"⇒"Basic frame widget"で得られます。
12. ｺﾝﾊﾟｲﾙして走らせ、そして釦をｸﾘｯｸした時に文字が更新されることを確かめてください。

おめでとう御座います!。今やあなたは作成したあなた自身のGUI要素と、GUIでの使用者相互作用時にそれを更新する追加論理を持ちます。

時間が有れば、以下によってより立派な計数器出力を作ることができます。

- ｳｨｼﾞｪｯﾄの中央に文字を移動
- gfx_draw_rectangle関数で文字の周囲に矩形を描画

(課題のどれかに対して助言が必要なら、完全な解決策に関するapp_widget_solution.cﾌｧｲﾙをご覧ください。

3. 改訂履歴

資料改訂	日付	注釈
8300A	2010年9月	初版資料公開
8300B	2012年7月	Atmelソフトウェア枠組みとXplain系統の基板用の練習を更新

Atmel | Enabling Unlimited Possibilities®

Atmel Corporation
1600 Technology Drive
San Jose, CA 95110
USA
TEL (+1)(408) 441-0311
FAX (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
TEL (+852) 2245-6100
FAX (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parking 4
D-85748 Garching b. Munich
GERMANY
TEL (+49) 89-31970-0
FAX (+49) 89-3194621

Atmel Japan G.K.
141-0032 東京都品川区
大崎1-6-4
新大崎勧業ﾋﾞﾙ 16F
ｱﾄﾒﾙ ｼﾞｬﾊﾟﾝ合同会社
TEL (+81)(3)-6417-0300
FAX (+81)(3)-6417-0370

本応用記述はAtmelのAVR1614応用記述(doc8300.pdf Rev.8300B-07/12)の翻訳日本語版です。日本語では不自然となる重複する形容表現は省略されている場合があります。日本語では難解となる表現は大幅に意訳されている部分もあります。必要に応じて一部加筆されています。頁割の変更により、原本より頁数が少なくなっています。

必要と思われる部分には( )内に英語表記や略称などを残す形で表記しています。

青字の部分はﾘﾝｸとなっています。一般的に赤字の0,1は論理0,1を表します。その他の赤字は重要な部分を表します。