PlusG SMART Solution をお試し頂きまして、ありがとうございます。本書は簡単なアプリケーションの製作を通して、PlusG SMART Solution を利用したアプリケーションの開発工程を理解して頂く事を目的としています。PlusG SMART Solution の評価や導入に、お役立てください。

1. 準備

1.1. システム要件

本書では、次の機器を使用します。要件を満たす機器及び環境をご用意ください。

Android タブレットまたはスマートフォン

専用クライアント「PGSMonitor」を動かすために使います。

[システム要件]

- OS: Android 3.2 以降
- NET: リソースエディタを実行する Window PC との TCP/IP 通信(例: 無線 LAN)



Windows PC

リソースエディタ「SGResourceEditor」、統合開発環境「SWEET」を動かすた めに使います。

[システム要件]

- OS : Windows XP SP3 32bit, Windows Vista 32/64bit, Windows 7 32/64bit
- NET: PGSMonitor を実行する Android 機器との TCP/IP 通信 (例:無線 LAN または、アクセスポイントのある有線 LAN)、 ビルドを実行する Linux PC との Telnet & Samba 通信



Linux PC

アプリケーションをビルドするために使います。

Linux PC をお持ちでない方は、仮想 PC ソフトウェアをご利用ください。弊社 では「VirtualBox」を使用しております。VirtualBox 向けのアプライアンス(構 築済イメージ)を用意しております。



VitualBox の導入手順は、別紙の「VirtualBox インストールガイド」

(<u>http://www.si-linux.co.jp/pub/SmartSolution/</u>)をお読みください。VirtualBox のインストーラーは、 公式サイト(<u>https://www.virtualbox.org</u>/)より、アプライアンスは弊社サイト

(<u>http://www.si-linux.co.jp/pub/SmartSolution/</u>) より、ダウンロードできます。

既存の Linux PC を利用する方は、システム要件をご確認ください。

[システム要件]

- OS: Debian GNU/Linux 6.0(推奨)
- PKG: gcc, make, telnetd, samba
- NET: PGSMonitor を実行する Android 機器との TCP/IP 通信、 SWEET を実行する Windows PC との Telnet & Samba 通信

1.2. PGSMonitor のセットアップ

Android にアプリケーションの画面を表示するソフトウェア「PGSMonitor」をインストールします。

下の2次元コードが示す URL から PGSMonitor.apk をダウンロード&インストールしてください。

2次元コードを利用できない方は、Android のブラウザから

http://www.si-linux.co.jp/pub/SmartSolution/PGSMonitor/CurrentVersion/ にアクセスしてください。

Index of /pub/ SmartSolution/ PGSMonitor/ CurrentVersion

Name Last modified Size Description

🗞 🛜 📶 🔲 19:11

Parent Directory
 PGSMonitor.apk 15-Jan-2013 12:42 329K

Apache/2.2.16 (Debian) Server at www.silinux.co.jp Port 80





1.3. リソースエディタのセットアップ

Windows PC にアプリケーションの画面を作成するソフトウェア「SGResourceEditor」(以下、リソース エディタ)をインストールします。

Windows のブラウザから

http://www.si-linux.co.jp/pub/SmartSolution/ResourceEditor/CurrentVersion/ にアクセスして、

ResourceEditor*.*.*.zip(*はバージョン番号)をダウンロードしてください。ファイルが複数ある場合は、 数字の大きい方をダウンロードしてください。



ダウンロード後、任意のフォルダ(インストール先)に展開してください。

ResourceEditor1.8.2.5.zip 圧縮 (zip 形式) フォルター		開く(0) 新しいウィンドウで開く(E)		
	(すべて展開(T)		
	۲	ESET NOD32 Antivirus で検査 詳細設定オプション		
		Git Init Here Git Bash		展開先の選択とファイルの展開
	•	ATOKで学習する(A) EmEditor プログラムから聞く(H) WinMerge		ファイルを下のフォルダーに展開する(F):
		送る(N)	•	
		切り取り(T) コピー(C)		
		ショートカットの作成(S) 削除(D) 名前の変更(M)		
		プロパティ(R)		2 展開(E) キャンセル

1.4. SWEET のセットアップ

Windows PC に統合開発環境「SWEET」をインストールします。

Windows のブラウザから http://www.si-linux.co.jp/product/sweet/download/ にアクセスして、

SweetInstaller*_*_*_*.exe(*はバージョン番号)をダウンロードしてください。ファイルが複数ある場合は、数字の大きい方をダウンロードしてください。



ダウンロード後はファイルを実行して、ウィザードに従ってインストールしてください。

SweetInstaller1_3_8_1.EXE	
アプリケーション	
 8.97 MB	

1.5. サンプルプロジェクトのダウンロード

Windows PC にサンプルのプロジェクトファイルをダウンロードします。

Windows のブラウザから <u>http://www.si-linux.co.jp/pub/SmartSolution/Tutorial/</u> にアクセスして、 「OKOnly.pgsspro」を任意のフォルダにダウンロードしてください。



2. サンプルによる動作確認

2.1. サンプルプロジェクトの読み込み

サンプルプロジェクトを利用して、お使いの環境でアプリケーションが動作するかを確認します。 リソースエディタを起動します。展開先フォルダの SGResourceEditor.exe を実行してください。



サンプルプロジェクトファイルを開きます。メニューから[ファイル]→[プロジェクト/リソースの読み込み]

を実行してください。

Reg SGResourceEditor	
ファイル(F) 表示(V) ヘルプ(H)	
新規作成(N) Ctrl+N 参照	│ 🗖 🛎 🖬 │ Re │ 扮羚│ 🖪 🔚 👄 │
プロジェクト/リソースの読込み(0) Ctrl+0	
プロジェクトの保存(全てのリソース)(S) Ctrl+S	
スケルトンの作成(M)	
最近使ったファイル	
全てのリソースを破棄	
アプリケーションの終了(X)	
Key Value	FRAME LINEAR PAGE RADIO RELATIVE SCROLL TABLE TABLEROW
	BUTTON CHECKBOX EDIT GRID METER PICTURE PROGRESS PULLDOWN
● 接続 ● プロパティー	」 📗 🔮 挿入オブジェクト 🕒 クリップボード 👋 標準画像(BUILTIN) 🚽 ユーザ登録画像(STATION) 🔴 通信ログ
既存のリソースを開く	

ダウンロードした"OKOnly.pgsspro"を開きます。

ファイルの場所(I):	🔒 Tutorial		- + I	G 👂 🖻	•		
P	名前	^	更新日日	ŧ	種類		t
- <u>(</u>			2013/01	/07 18:04	ファイル	ノフォル	
血表示した場所	OKOnly.pgssp	pro	2013/01	/09 14:47	PGSSPRC)ファイル	
デスクトップ デスクトップ ライブラリ	_ rop.pgsspro		2015/01	20:36	POSSERC	5.57476	
	•	ш		(2		•
ットワーク	ファイル名(N):	OKOnlypgsspro			-	開((0)	
	ファイルの種類(T):	SGResourceEditor Files (* i	son *pessor	n)	-	キャンセル	

2.2. リソースエディタ-PGSMonitor 接続

プロジェクトを開いたら、ウィンドウの左下にある[接続]タブを選択して、タブ内の[待受]ボタンをクリッ

修 SGResourceEditor - dialog1 ファイル(F) 編集(E) 表示(V) ウィンドウ(W) ヘルプ(H)	
¥ 参照	🗖 🗃 🖬 Re 💯 🖽 🖶 🔿
E Root = okonly È Dislog:dislog1	BTN button 1.0K
待受ポート 5000 2 待受 停止 IPアドレス 0 , 0 , 0 , 0 接続 切断	¥I (
送信	FRAME LINEAR PAGE RADIO RELATIVE SCROLL TABLE TABLEROW
	BUTTON CHECKBOX EDIT GRID METER PICTURE PROGRESS PULLDOWN
D	• Text
 接続 プロパティー 	 ● 挿入オブジェクト ● クリップボード ● 標準画像(BUILTIN) - ユーザ登録画像(STATION) ● 通信ログ
スケルトンソースコードのテンプレートの種類を決定します。	

クします。リソースエディタが PGSMonitor からの接続待ち受けを開始します。

OS によっては初回実行時に[Windows セキュリティの重要な警告]が表示される事があります。その場合は [アクセスを許可する]ボタンをクリックしてください。

【注意】

[Windows セキュリティの重要な警告]は、デフォルトでは「プライベートネットワーク」のアクセスのみ許可するようになっています。後の工程で PGSMonitor と通信ができない場合は、この設定とネットワーク種類の不一致が原因の可能性がありますので、ネットワーク管理者やパワーユーザーに問い合わせて、設定を確認してください。

ウィンドウの右下にある[通信ログ]タブを選択します。待ち受けが開始されると、ログに次のメッセージが 表示されます。

::**#待ち受け開始しました。PORT:5000 [TCP] (*は数字)

😽 SGResourceEditor - dialog1	
ファイル(F) 編集(E) 表示(V) ウィンドウ(W) ヘルプ(H)	
¥ 参照	
⊡Root = okonly ⊕ Dialog dialog 1	B ^C dialog1 BTN button1:0K
待受ボート 5000 待受 停止 IPアドレス 0 , 0 , 0 , 0 接続 切断 送信	▲ 17:34:17指待ち受け開始しました。PORT:5000 [TCP] ②
● 接続 ● プロパティー	● 挿入オブジェクト ● クリップボード ● 標準画像(BUILTIN) ● ユーザ登録画像(STATION) ● 通信ログ
スケルトンソースコードのテンプレートの種類を決定します。	

リソースエディタに PGSMonitor を接続します。Android で PGSMonitor を起動してください。



アプリメニューを開いて [IP アドレス登録]をタップします。IP アドレス登録画面が開きます。 各フィールドに値を入力して[登録]ボタンをタップしてください。自動で再接続が開始されます。



ユーザー名 : guest パスワード : guest 接続方法 : TCP 接続 TCP IP アドレス : Windows PC (リソースエディタ)の IP アドレス TCP ポート : 5000 接続に成功すると PGSMonitor 上にポップアップで「TCP 接続:接続開始!」と表示されます。

リソースエディタの通信ログに次のメッセージが表示されます。

::**<受信:HELLO OS:Android Resolution *** x *** (*は数字)

2.3. 画面送信-イベント受信

リソースエディタから PGSMonitor にサンプルダイアログを送信します。リソースエディタの[接続]タブ内 にある[送信]ボタンをクリックしてください。

【注意】

[送信]ボタンが無効になっている場合、Android 側のスリープ等により接続が切れている事が考えられます。 その時は、Android をスリープから復帰して、アプリメニューの[再接続]を実行してください。

්ල SGResourceEditor - dialog1	
ファイル(F) 編集(E) 表示(V) ウィンドウ(W) ヘルフ(H) ¥	□ 🛩 🖬 Re 💯 🖪 🖬 🔿
E-Root = okonly ⊕ Dialog dialog 1	PG dialog1
待愛ボート 5000 IPアドレス 172、16、1、13 接続 切断 送信	▲ 18:58:00#接続要求を許可します。 18:58:00#接続要求許可 172.16.1.3:5000 172.16.1.13:38655 通信ポート
	18:58:01<受信:HELLO 0S:Android Resolution 540 x 888 18:58:19<受信:KEEPALIVE終了ステータス 18:58:19H通信を切断しました。 18:58:27#接続要求を許可しました。 18:58:27#接続要求許可 172.16.1.3:5000 172.16.1.13:42894 通信ポート 18:58:27*受信:HELLO 0S:Android Resolution 540 x 888
🧶 接続 🕚 ブロパティー	● 挿入オブジェクト 🔵 クリップボード 🥌 標準画像(BUILTIN) – ユーザ登録画像(STATION) 🔮 通信ログ
スケルトンソースコードのテンプレートの種類を決定します。	

ダイアログの送信 – 受信に成功すると、PGSMonitorに次の様なサンプルダイアログが表示されます。



サンプルダイアログが表示されたら、ダイアログ上の[OK]ボタンをタップしてください。リソースエディ タの通信ログに次のメッセージが表示されます。このメッセージは、PGSMonitorで"button1"ボタンがタッ プされた事を通知しています。

::**<受信: CONDITION {'Command':'CONDITION','ActionName':'button1',...(*は数字)

このように PlusG SMART Solution では、ユーザーが Android の画面を操作すると、予め設定されたコマ ンドがサーバー(アプリケーション)に通知(送信)されるようになっています。サーバー(アプリケーショ ン)は受け取ったコマンドに応じて機器の制御などの振る舞いを行い、必要であればコマンドを使ってクライ アント(Android)にフィードバックを指示(送信)します。これらのやり取りが PlusG SMART Solution に おける基本的な処理の流れとなります。

3. アプリケーションの製作

3.1. 概要

ここから、アプリケーションの製作手順を説明します。「画面にトグルを表示して、トグルの ON/OFF が切り替えられると、ファイルに値を書き込む。」というアプリケーションを製作していきます。



3.2. リソースの破棄

新しいプロジェクトを始める前に、既存のプロジェクト(サンプル)を破棄します。リソースエディタのメ ニューから[ファイル]→[全てのリソースを破棄]を実行してください。ツリーから全てのダイアログが削除さ れます。

ResourceEditor - dialog1	
ファイル(F) 編集(E) 表示(V) ウィンドウ(W) ヘルプ(H)	
新規作成(N) Ctrl+N 参照] 🗖 🗃 🖬 Re 555 🖪 🖪 🔿
新規リソースの作成	
プロジェクト/リソースの読込み(0) Ctrl+0	
データ送信	
プロジェクトの保存(全てのリソース)(S) Ctrl+S	button 1:0K
リソースの保存(このリソースのみ)	
このリソースを閉じる(C)	
スケルトンの作成(M)	
最近使ったファイル	
全てのリソースを破棄	
アプリケーションの終了(X)	
	 ★ 18:58:19<5受信: KEEPALIVE終了ステータス 18:58:19 18:58:19 18:58:19 18:58:27 18:58:28 18:58:58 18:58 19:00:58 18:58 19:58 19:58 19:58 19:58 19:58<
● 接続 ● プロパティー	」 / ● 挿入オブジェクト ● クリップボード ● 標準画像(BUILTIN) - ユーザ登録画像(STATION) ● 通信ログ

3.3. 画面の作成

新しい画面を作成します。作成する画面のイメージは次の通りです。



ツールバーの[新規ダイアログ]ボタンをクリックしてください。リソースツリーにダイアログが追加されま



アプリケーションのプロパティ値を変更します。プロパティ値を変更するには、[プロパティ]タブに切り替 えます。

SGResourceEditor - SGResourceEditor1	
ファイル(F) 編集(E) 表示(V) ウィンドウ(W) ヘルプ(H)	
¥ 参照	🗖 📽 🖬 Re 👭 🖪 🖬 🗰
-Name(Tutorial) L. Dialog:Dialog	SGResourceEdito1
Key Value Name Tutorial DeviceHeight	FRAME LINEAR PAGE RADIO RELATIVE SCROLL TABLE TABLEROW
	BUTTON CHECKBOX EDIT GRID METER PICTURE PROGRESS PULLDOWN
 	● 挿入オブジェクト ● クリップボード ● 標準画像(BUILTIN) ● ユーザ登録画像(STATION) ● 通信ログ

ッリーから対象のノードを選択します。リストにプロパティと値の一覧が表示されるので、変更するプロパ ティ行の Value 列をクリックします。値が反転(選択中)表示になり、変更できるようになります。



アプリケーションのプロパティを変更する場合は、「Root = ...」を選択します。次のプロパティ値を変更してください。

Name:Tutorial

ダイアログのプロパティを変更します。ツリーの「Dialog:Dialog」を選択して、次のプロパティ値を変更 してください。

Name: Top

TemplateType: Shared

TargetLayoutWidth: 100

TargetLayoutHeight: 100

ダイアログに FRAME グループを追加します。 [挿入オブジェクト]タブ内の[FRAME]をダブルクリックして ください。 ダイアログに FRAME グループが追加されます。

* SGResourceEditor										
ファイル(F) 表示(V) ヘルプ(H)										
¥ 参照			Re	₩ <mark>□</mark>						
Key Value	SUT		LINEAR CHECKBOX	PAGE Edite EDIT Text	RADIO GRID	RELATIVE	SCROLL SCROLL PICTURE	TABLE PROGRESS	TABLEROW PULLDOWN	
● 接続 ● ブロパティー	挿	入オブジ	יבי אליבי	クリップボード	🧑 標準通	回像(BUILTIN)	📒 ユーザ3	登録画像(STA	TION) 🧶 通f	言ログ

次のプロパティ値を変更します。

Name: frame1

Width: PARENT

Height: PARENT

FRAME グループに PICTURE オブジェクトを追加してください。

SGResourceEditor	
ファイル(F) 表示(V) ヘルプ(H)	An Training (1931)
¥ 参照] □ 🗳 🖬 Re 999 🖸 🗔 👄
[2] (2) (2) (2) (2) (2) (2) (2) (2) (2) (2)	
	FRAME LINEAR PAGE RADIO RELATIVE SCOLL TABLE TABLEROW FRAME LINEAR PAGE RADIO RELATIVE PROGRESS PULLDOWN BUTTON CHECKBOX EDIT GRID METER PICTURE PROGRESS PULLDOWN
 接続 ブロパティー 	→ 挿入オブジェクト ・ クリップボード ● 標準画像(BUILTIN) ・ ユーザ登録画像(STATION) ● 通信ログ
新規にダイアログを作成します	

プロパティ値を変更してください。

Name: picture1

Width: PARENT

Height: PARENT

File: frame.9.png

FRAME グループに LINEAR グループを追加してください。

SGResourceEditor		
ファイル(F) 表示(V) ヘルプ(H)		
¥ 参照	〕	
[a] [a] [a] [b] [b] [b] [b] [b] [b] [b] [b] [b] [b		
key value	Image: Second	
 接続 ブロパティー 	▲ 挿入オブジェクト ● クリップボード ● 標準画像(BUILTIN) ● ユーザ登録画像(STATION) ● :	通信ログ
新規にダイアログを作成します		

プロパティ値を変更してください。

Name: linear1

Width: PARENT

Height: PARENT

Padding: 10,20,10,20

Align: CENTER | MIDDLE

LINEAR グループに TOGGLE オブジェクト(1つ目)を追加してください。

SGResourceEditor - SGResourceEditor1	
ファイル(F) 編集(E) 表示(V) ウィンドウ(W) ヘルプ(H)	
¥ 参照	🗖 📽 🖬 Re 奶% 🖪 🖪 🔶
□ Name(Tutorial) L_Dialog:Dialog	SGResourceEditor1
「読み 送	
Name Tutorial DeviceWidth DeviceHeight	BUTTON CHECKBOX EDIT RADIOBTN SLIDEBAR TEXT TOGGLE
 ● 接続 ● ブロパティー 	 ● 挿入オブジェクト ● クリップボード ● 標準画像(BUILTIN) ● ユーザ登録画像(STATION) ● 通信ログ

プロパティ値を変更してください。

Name: sw1

Height: 20

Weight: 1

FileOff: switch_off.9.png

FileOn: switch_on.9.png

LINEAR グループに TOGGLE オブジェクト(2つ目)を追加して、プロパティ値を変更してください。

Name: sw2

Height: 20

Weight: 1

FileOff: switch_off.9.png

FileOn: switch_on.9.png

3.4. 動作の追加

2つのトグルにタップ時のアクションを設定します。リソースツリーの Dialog/frame1/linear1/sw1 を選択して、OnClick プロパティ行の[▼]をクリックしてください。コマンドのリストが表示されます。

G SGResourceEditor - Top	
ファイル(F) 編集(E) 表示(V) ウィンドウ(W) ヘルプ(H)	
¥ 参照	□ 😂 🖬 📭 555 🖬 🖬 🖬 🖬 🖬
Boot = Tutorial Dialog Top Group:FRAME(frame 1) OpictPIOTURE(picture 1) OpictPIOTURE(picture 1) OpictPIOTOGELE(see 1) OpicetTOGGELE(see 1) OpicetTOGGELE(see 2) Command NOTIFY Command NOTIFY	#9 Top TOOL sw1 picture 1 frame 9 png
Key Value	
Margin V	
Disable	
BGColor V	
Clickable 🥂 👗	
OnClick 🕂 🕫 🖊 🚺 💟	14:39:28 #持ち受け開始しました。PORT:5000 [TCP]
Attr	14:39:33#接続要求を許可します。
	14:39:33#接続要求許可 172.16.1.3:5000 172.16.1.13:43195 通信ポート
TSize	14:39:33<受信:HELLO OS:Android Resolution 540 x 888
TColor CLOSE	14:39:55%受信: KEEPALIVE
TAlign SOUND	14:40.55公支信: RELFALIVE
FileOff TIMER	14:41:00\SZTE: NEEFALIVE
FileOn DELETE	
FilePath	
🧶 接続 🥚 プロパティー	● 挿入オフジェクト ● クリッフホード ● 標準画像(BUILTIN) - ユーザ登録画像(STATION) ● 通信ログ
属性を指定します	

リストから[CONDITION]をクリックしてください。ツリーに CONDITION コマンドが追加されます。



CONDITION コマンドの Name プロパティの値を"sw1"に変更してください。

同様に Dialog/frame1/linear1/sw2 にも CONDITION コマンドを追加してください。Name プロパティ

の値は"sw2"にしてください。

3.5. プロジェクトの保存

作成したリソース(ダイアログ)を保存します。リソースエディタのメニューから[ファイル]→ [プロジェ クトの保存(全てのリソース)]を実行してください。ファイル名は"Top"にしてください。

SGResourceE	ditor - Top		
「ファイル(F)」 編	集(E) 表示(V) ウィンドウ(W) ヘルプ	H)	
新規作成(ト	l) Ctrl	+N 参昭	
新規リソー	スの作成		
プロジェク	ト/リソースの読込み(0) Ctrl	+0	
データ送信			sw1 sw2
プロジェク	トの保存(全てのリソース)(S) Ctr	+ S	
リソースの保存(このリソースのみ)			picture 1:frame.9png
このリソースを閉じる(C)			
スケルトンの作成(M)			
最近使った	ファイル		
全てのリン	ースを破棄	8	
アプリケー	ションの終了(X)		
Name	sw2		
Alias	700015	=	🖌 📕 🖌 Ediel 🔛 🖓 📶 💭 💳 🔤 🗍
lype Wodth	TUGGLE		
Height	20	÷.	BUTTON CHECKBOX EDIT GKID METER PICTURE PROGRESS PULLDOWN
Weight	1		• Text ON
Padding		Y	
Margin		T	RADIOBTN SLIDEBAR TEXT TOGGLE
Hide		Y	
Disable		Y	
BGColor		T -	e
🔴 接続 🥚 :	プロパティー	1	🗁 📗 🧉 挿入オブジェクト 🧧 クリップボード 🔮 標準画像(BUILTIN) 📒 ユーザ登録画像(STATION) 🌘 通信ログ
ノロジェクトを使	任します。(全てのダイアログの保存)		

3.6. 画面/動作の確認

リソースエディタと PGSMonitor を接続して、作成したダイアログを送信してください。 ダイアログが表示されたら、いずれかのトグルをタップしてください。リソースエディタの通信ログに 「'Command':'CONDITION'」を含む行が表示されれば、動作も正しく設定されています。

3.7. スケルトンの出力

実行ファイルを作るために、ソースコードを出力します。リソースエディタのメニューから[ファイル]→[ス ケルトンの作成]を実行してください。[スケルトン作成]ダイアログが開きます。

G SGResourceEditor - Top			- 🗆 <mark>- X -</mark>
[ファイル(F)] 編集(E) 表示(V) ウィンドウ(W)	ヘルプ(H)		
新規作成(N) 新規リソースの作成	Ctrl+N		
プロジェクト/リソースの読込み(0)	Ctrl+O		
データ送信		sw1 sw2	
プロジェクトの保存(全てのリソース)(S) リソースの保存(このリソースのみ)	Ctrl+S		
このリソースを閉じる(C)			
スケルトンの作成(M)		picture 1:frame.9png	
最近使ったファイル			
全てのリソースを破棄			
アプリケーションの終了(X)			
「「「「「」」」、「「」」、「「」」、「「」」、「「」」、「「」」、「」、「」、	8 8 8		
Key Value	^		
Hide			
Disable	▼		
BGColor	T		
Clickable	▼ 5		
OnClick	 [14:53:55(受信: KEFPALTVE	*
Attr	• E	14:54:55<受信:KEPALIVE	
TextOn	=	14:55:55<受信:KEPALIVE	
TSize		14:56:55<受信:KEEPALIVE	
TColor	▼	14:57:55<受信:KEEPALIVE	
TAlign	•	14:58:55<受信:KEEPALIVE	
FileOff switch_off.9png	>>	14:59:55<受信:KEEPALIVE	
FileOn switch_on.9png	>>	15:00:56<受信:KEEPALIVE	H
FileSource		15:01:55<受信: KEEPALIVE	
Filerath	*		•
🥚 接続 🤮 プロパティー		 ・ 挿入オブジェクト ・ ・ ・	
			0

[スケルトン作成]が開いたら、次の操作を実施してください。

- 1. [出力フォルダ]に任意のフォルダ(ソースフォルダ)を指定する。
- 2. チェックボックスの右側全てにチェックを点ける。
- 3. [すべてのダイアログ]ラジオボタンを選択する。

操作が終わったら[スケルトン作成]ボタンをクリックしてください。出力フォルダにソースファイル (.cpp, .h) が作成されます。

上書き 🗐 ·	ーここにチェックを入れる マンメイン	ると全て上書きになります。個別指定は下のチェックで。 Tutorial.cpp
	 ▼プリケーション ▼イアログ生成 ▼イアログ生成 	TutorialApplication.cpp TutorialApplication.h TutorialDialogMaker.cpp TutorialDialogMaker.h
	♥ 31 /000 ♥ 350N ♥ 100 50 10ト	Top このダイアログのみ ・ でのダイアログ
	Makefile (4)	スケルトン生成

3.8. コーディング

出力されたソースは基本構造のみを備えている(スケルトン)ため、これにトグルがタップされた時の動作 を付加します。

TutorialDialogTop.hの次の箇所を変更してください。

24 行目付近:

void ResourceToDevice();

 \downarrow

void ResourceToDevice(PGJsonObject*);

TutorialDialogTop.cppの次の箇所を変更してください。

3 行目付近:

(なし)

 \downarrow

#include <fcntl.h>

40 行目付近、OnInit 関数内

ResourceToDevice();

 \downarrow

//ResourceToDevice();

87 行目付近、OnCondition 関数内

ResourceToDevice();

 \downarrow

//ResourceToDevice();

97 行目付近、OnUpdate(PGJsonObject*,PGJsonObject*) 関数内

PGDialog::OnUpdate(to,from);

 \downarrow

PGDialog∷OnUpdate(to,from);

ResourceToDevice(to);

155 行目付近、ResourceToDevice() 関数定義

ResourceToDevice()

 \downarrow

ResourceToDevice(PGJsonObject* to)

157 行目付近、ResourceToDevice() 関数内

(なし)

 \downarrow

```
}
              else{
                      perror("write");
              }
              close(fd);
      }
}
else if(name == "sw2") {
      char c = '?';
      if(attr == "ON") {
            c = '1';
      }
      else{
             c = '0';
      Ł
      int fd = open("./sw2", 0_WRONLY | 0_CREAT, 0644);
      if(fd >= 0) {
              int l = write(fd, &c, sizeof(c));
              if(1 > 0)
              }
              else{
                     perror("write");
              }
              close(fd);
      }
}
```

3.9. ビルド

アプリケーションのビルドは Linux で行います。ソースフォルダを Linux PC の任意のディレクトリ(以下、 ビルドディレクトリ) に移動してください。

移動後、Windows PC で SWEET を起動して、スケルトンフォルダにある Tutorial.slpro を開いてください。

The second secon	fop.cpp] 未編集 - [\/mary\kaihatsu\Skeleton\TutorialDialogTop.cpp]	
🚟 ファイル(F) 編集(E) 表示(V) [プロ3	ミュクト」デバッグ CVS ウィンドウ(W) ヘルプ(H)	_ 6 ×
D 📽 🖬 🕼 😫 🙀 🗐 📑 🔅	プロジェクトの設定 💦 👻 セルフ 🐨 デバッグ 🔹 🖉 👹 🖉 🏈 🌦 🗛 🕮 🛴 ! 💷 🧃 🖉	4 7 7
TutorialDialogTop.cpp	プロジェクトを新規作成	
	プロジェクトを開く 1 [20	90
¥¥mary¥kaihatsu¥Skeleton¥	プロジェクトを保存 :Shared	E
Tutorial	プロジェクトを閉じる ialDialogTop.h"	1
Source		
Tutorial.cpp	7 <mark>#define</mark> debugs { cout << FILE << ':' << LINE << ':' << FUNCTION << end ; }	
h] TutorialApplication	8///	
TutorialDialogMaker.cpp	10//コンストラクタ 11 TutorialDialogTop::TutorialDialogTop()	
el TutorialDialog Maker h		
h TutorialDialogToph	13 debugs; 14 }	
Document	17// デストラクタ	
	20 debugs; 21 }	
		+
🗁 ファイル 📋 プロジェウト <u>ត</u> CVS	<	•
×		
		-
ターケット機コンソール 🧱 開発機コンソール	📰 結果出力 🕎 GDB12/2-ル 🔁 バックトレース 🔶 フレークホイクト 🕢 変数ウォッチ 🕜 タゲ一覧 🔲 マークー覧 🕎 CVS12/2-ル	
プロジェクトを開きます	🚰 開発機 切断 🖤 GDB 切断 🌮 5-5*9ト 切断 ፵ CVS 切断 UTF-8 LF 1行:1列 挿入	

プロジェクトの設定を、お使いの環境に合わせて変更してください。

號 統合開発環境 SWEET 製品版 - Tutoria	il - [\\mary\kaihatsu\Skeleton\TutorialDialogTop.cpp]	
このでは、「「「「「「「「「「「「」」」」 「「「」」 「「」 「「」 「」 「」 「」	ロジェクト」デバック CVS ウィンドウ(W) ヘルプ(H)	- 8 ×
🛛 🗅 🖨 🖬 🗿 🗋 🖨 🖬 🚺	プロジェクトの設定	X 7 0
TutorialDialogTop.cpp	プロジェクトを新規作成	
	プロジェクトを開く	19.0.
¥¥mary¥kaihatsu¥Skeleton¥	プロジェクトを保存 :Shared	III
🕞 🛅 Tutorial	プロジェクトを閉じる ialDialogTop.h"	
c) TutorialApplication.cpp	7 #define debugs { cout << _FILE_ << ':' << _LINE_ << ':' << _FUNCTION_ << endl; }	
h TutorialApplication h	8// 10// コンストラクタ	
h] TutorialDialogMaker.cpp	11 TutorialDialogTop::TutorialDialogTop()	
TutorialDialogTop.cpp	12 t 13 debugs;	
h lutorialDialog loph	14 }	
Document	16 // 17 // デストラクタ	
	18 TutorialDialogTop::~TutorialDialogTop()	
	20 debugs;	
🗁 ファイル 📋 プロジュウト 📷 CVS		•
x		
		*
		•
🥼 🌗 🐓 ターケット機コンソール 🔤 開発機コンソ	ノール 📰 結果出力 200 GDB12ソソール 🔁 バックトレース ● ブレークポイント 🐼 変数ウォッチ ሰ タン゙ー覧 🔜 マーウーー覧 200 GVS12ソソール	
プロジェクトの設定	🚰 開発機 切断 🖤 GDB 切断 🎺 ターガット 切断 ፵ CVS 切断 UTF-8 LF 1行:1列 挿入	

[開発機]

ホスト名/IP: (Linux PC の IP アドレス)

ユーザー名: (Linux PC のログインユーザー名)

```
パスワード: (ログインユーザーのパスワード)
ルートパスワード: (Linux PC の管理者パスワード)
エンコード: (Linux PC の標準文字コード)
```



プロジェクトの設定変更が済んだら、開発機にログインして、Makefile 作成を実行してください。





Makefile が作成されたら、ビルドタイプを[セルフ]&[デバッグ]に変更して、コンパイルを実行してくださ



ジが表示された場合は、メッセージの内容から適宜トラブルシュートを行ってください。

🚼 [\\mary\kaihatsu\Skeleton\TutorialDialogTop.c	pp] 未編集 - [\\many\kaihatsu\Skeleton\TutorialDialogTop.cpp]	
🞇 ファイル(F) 編集(E) 表示(V) プロジェク	7ト デバッグ CVS ウィンドウ(W) ヘルプ(H)	_ 6 ×
0 😅 🖬 🕼 達 🖬 🗊 👗 🛍 🛍	」 🛍 🎨 💦 🔹 🛃 セルフ 🗸 デボッグ 🗸 🐺 👹 💭 加 🏍 🔛 🔙 📜 🖬	$\widehat{\mathbb{Z}}_{i}^{t} \mid \widehat{\{^{k}\}} \; \widehat{\{\!\}}$
TutorialDialogTop.cpp	logTop.h	
	0	. 19,0 🔺
¥¥mary¥kaihatsu¥Skeleton¥	2 // TemplateType:Shared	I
🖃 🦳 Tutorial	4 <mark>#include</mark> "TutorialDialogTop.h"	
TutorialApplication.cpp	7 #define debugs { cout << _FILE << ':' << _LINE << ':' << _FUNCTION << end ; }	
	9///	
h] TutorialDialogMakerh	11 TutorialDialogTop::TutorialDialogTop()	
TutorialDialogTop.cpp	debugs;	
	14/2	
Document	16///	
	18 TutorialDialogTop::~TutorialDialogTop()	
	20 debugs;	
	22	
🗁 ファイル 📋 ጋግድንኳንኑ 📶 CVS	23	•
× 10 . 10 . 150 .	,]
1 make DEBUG=Y	, . 1901190119011901170118011901110011120.	· · · · · · ·
3 cp obj/Tutorial		E
4kaihatsu@mary 7/Skeleton 5ビルドが正常に終了しました。		÷
< <u>m</u>		F
🚺 ターケット機コンソール 🕎 開発機コンソール 🚛	結果出力 🕎 GDBユソソール 🔁 バックトレース 🔶 フレークポイント 🐼 変数ウォッチ 🕜 ダゲー覧 🔲 マークー覧 🧱 CVSユソノール	
<i>ل</i> 7*1	■ 開発機 192.168.56.101 ■ GDB 192.168.56.101	2 0

3.10.動作確認

アプリケーションを実行します。[開発機コンソール]タブ内のシェルで"./sample"を実行してください。



アプリケーション実行後、Android で PGSMonitor を起動して、IP アドレス登録に Linux PC の IP アドレ スを設定してください。接続に成功した場合、先程作成したダイアログが表示されます。 左側のトグルをタップしてください。トグルの表示が変わり、開発機コンソールにコマンド受信メッセージ が表示されます。



受信メッセージが表示されたら、開発機コンソールで Ctrl+Z を入力してアプリケーションを停止します。



Is コマンドを実行してください。sw1 というファイルが作成されています。

"cat sw1"を実行して中身を見てください。'0'または'1'が表示されます。このファイルはアプリケーションの ResourceToDevice 関数に記述したコードにより sw1 トグルの状態を反映するようになっています。

実際のアプリケーション開発では、このファイルに代わってデバイスファイルを読み書きする事で、周辺機 器の制御ができます。



以上