

iOS Pebbleプラグイン Buildマニュアル Device Connect 1.0対応版

目次

1. Buildに必要なパッケージ	3
2. Xcode側の準備	3
3. プロジェクトのImport	4
4. 必要なパッケージのビルド	5
5. ビルドされたパッケージの確認	8
4. PebbleAppのビルドと、iOS Pebble への登録	9
4.1 iOSの設定	9
4.2 PebbleAppのビルド	10
4.3 dConnectDevicePebbleプロジェクトへの登録	10
5. 通信プログラム作成時の注意点	11
5.1 Pebble Cプログラム作成上の注意	11
5.3 通信プログラム作成上の注意 (Pebble側)	12
付録A: 更新履歴	13

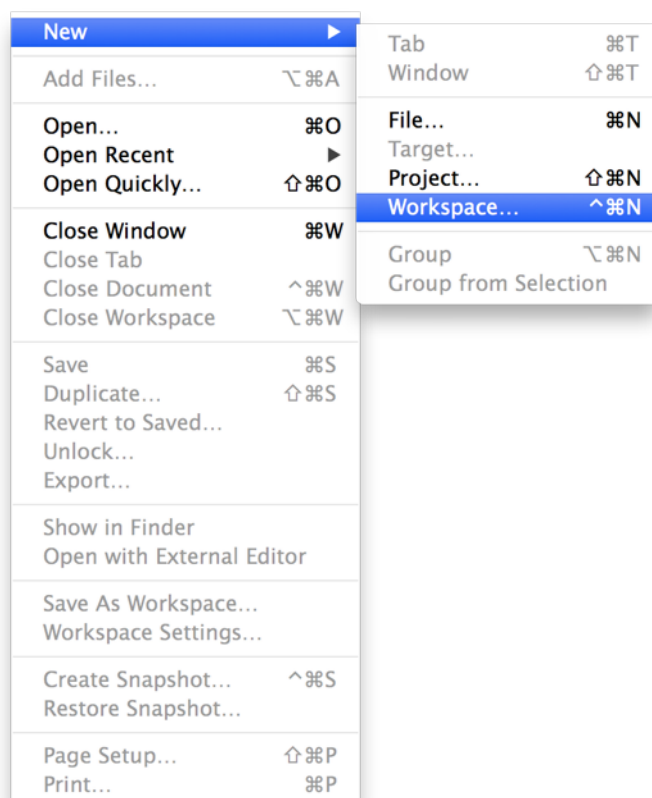
1. Buildに必要なパッケージ

iOS PebbleプラグインのBuildに必要なパッケージは以下の通りである。

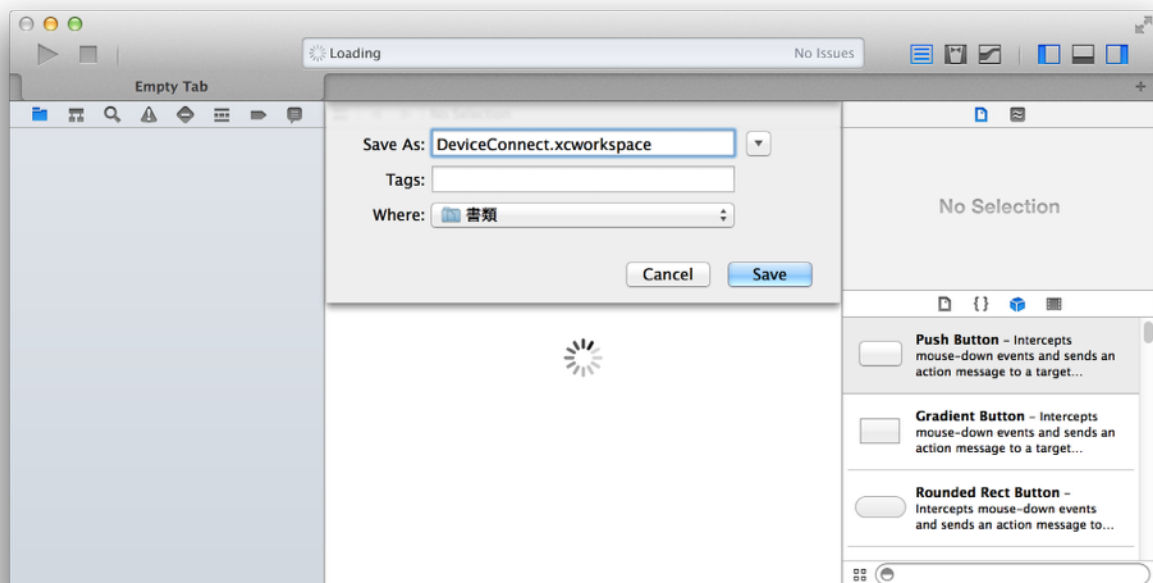
DConnectSDK	dConnectSDKライブラリやリソース
dConnectDevicePebble	iOS Pebbleデバイスプラグイン
pebbleKit.framework	Pebbleのライブラリ
PebbleVendor.framework	

2. Xcode側の準備

iOS DeviceConnectでは複数のXcodeプロジェクトが関わってくるので、それらプロジェクトを一元管理する為の共同作業スペースとしてXcodeのWorkspaceを用意する。



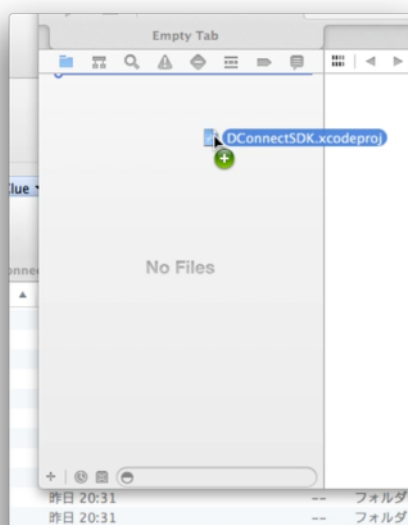
Workspaceの名称と保存場所は特に問わない。



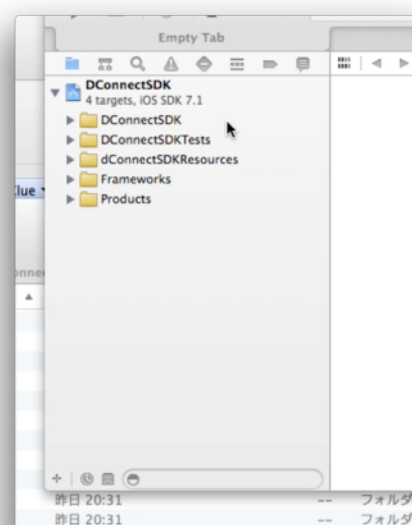
3. プロジェクトのImport

節「iOS PebbleのBuildに必要なパッケージ」で挙げた物のビルドを行う為にまずWorkspaceに「必要なパッケージ」のXcodeプロジェクトをImportする必要がある。

まず、Finder上のdConnectSDKのプロジェクトファイル「DConnectSDK.xcodeproj」を、先ほど用意したWorkspaceのプロジェクトナビゲーターにドラッグ&ドロップしてプロジェクトをImportする。正しくImportされた場合、ナビゲーターにdConnectSDKプロジェクトのファイル一式が表示される。



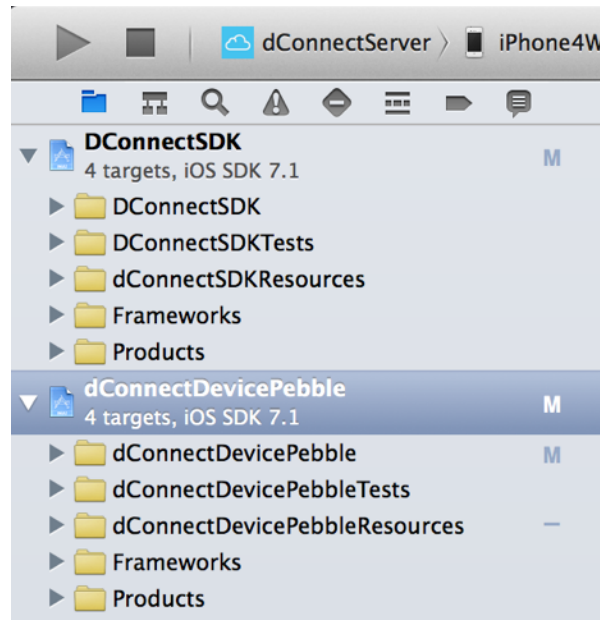
プロジェクトファイルをドラッグ&ドロップ。



ナビゲーターにプロジェクトが追加される。

そしてdConnectSDKと同じ手順でiOS Pebbleデバイスプラグインのプロジェクトファイル「dConnectDevicePebble.xcodeproj」をWorkspaceにImportする。最終的にdConnectSDKとiOS Pebbleデバイスプラグインのファイル一式がナビゲーターに表示されれば完了である。

4. 必要なパッケージのビルド



節「プロジェクトのImport」で必要なパッケージをビルドする為のプロジェクトをImportしたことで、プロジェクトのファイル一式に加え、Workspaceにビルドターゲットという物が追加される。必要なパッケージの内、どれをビルドするかを選択するのに今回使うのがビルドターゲットとなる。

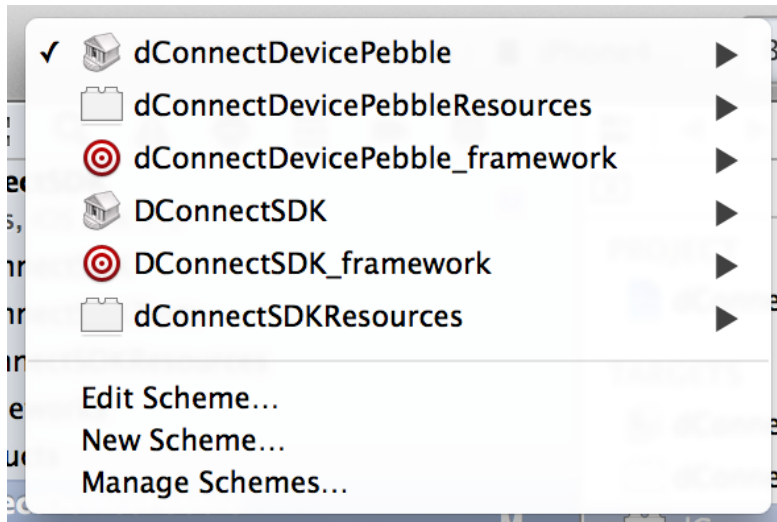
必要なパッケージの間にはビルドの依存関係があり、特にDConnectSDKはデバイスプラグインに先立ってビルドしなければならない。ビルドの依存関係を解決する形でのビルドターゲットのビルド順序は以下の様になる。

1. DConnectSDK_framework
2. DConnectSDK_resources
3. dConnectDevicePebble_framework
4. dConnectDevicePebble_resources

そして作業しているXcodeのWorkspace内でのビルドターゲットの切り替え方法は、Xcodeのウィンドウの左上、RunやStopボタンの右にあるのが現在選択されているターゲットであり、そのターゲットをクリックする事で表示されるWorkspace内のビルドターゲット一覧から対象のビルドターゲットを選択する事でビルドターゲットの切り替えを行う事ができる。



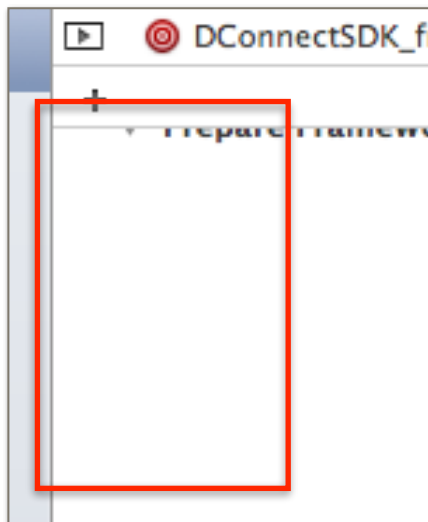
Xcodeのウィンドウの左上に現在選択されているターゲットが表示される。



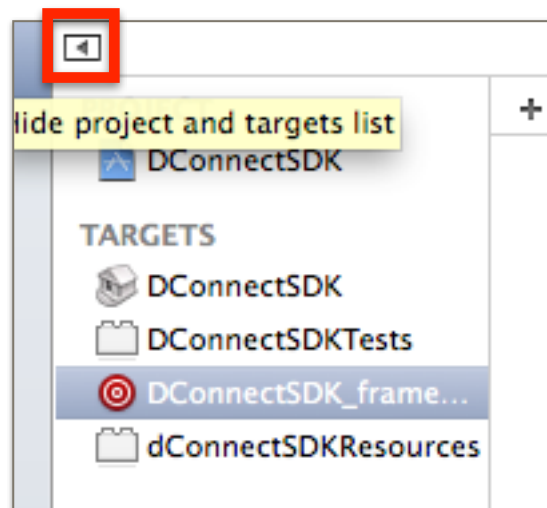
ターゲットをクリックすると一覧が表示されるので、ビルドしたいターゲットを選ぶ。

あとは先ほど示したビルドターゲットのビルド順序でビルドを行う。ビルドを行うにはRunボタンを押すか、ショートカット⌘+Bを使う等して実行する事ができる。

また、各プロジェクトにはDebugとReleaseというビルド設定が有る。Debugビルド設定は開発段階のデバッグに適したパッケージのビルドを行い、Releaseビルド設定はデバッグ情報を除いて処理速度や成果物のデータサイズが最適化される形式でのパッケージのビルドを行う。このDebugやReleaseのビルド設定を切り替える方法は、DConnectSDKを例に挙げれば、DConnectSDKのプロジェクトファイルを選択し各種プロジェクト情報を表示させる。更にビルドターゲットの各種ビルド情報を表示させるが、ビルドターゲットの一覧が非表示の際はその一覧を表示させた上で、フレームワーク（.framework拡張子ファイル）のビルドターゲットであるDConnectSDK_frameworkを選択し各種ビルド設定を表示する。

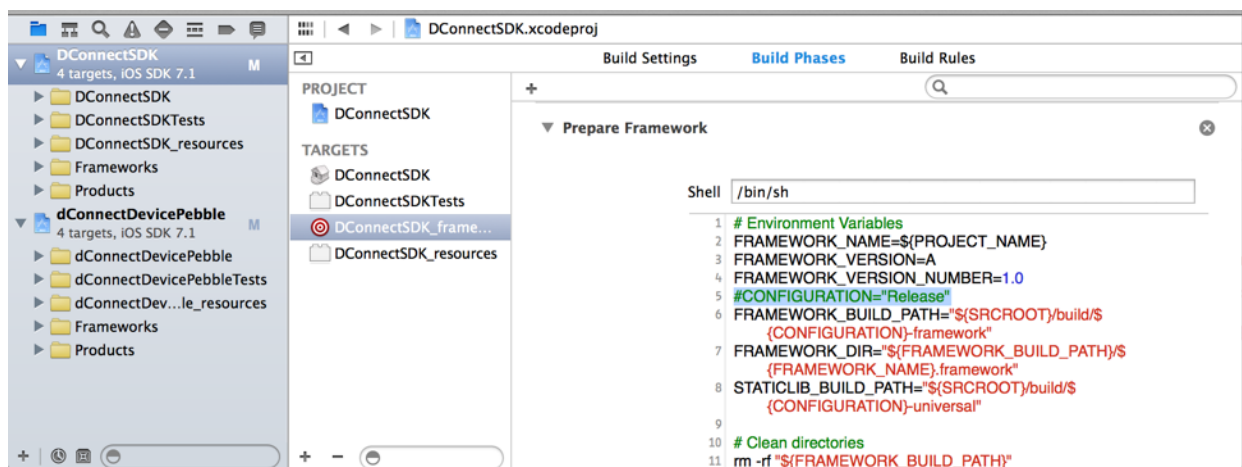


枠部分にプロジェクトのビルドターゲット一覧が表示できる。



左上のトグルボタンを押す事で、非表示になっているビルドターゲット一覧を表示させる事ができる。

ビルドターゲットDConnectSDK_frameworkを選択したらBuild Phaseタブを選択し、表示されたビルドの各フェーズの内、スクリプトを走らせるフェーズを表示する。そのスクリプトの内、CONFIGURATIONビルド環境変数を"Debug"なり"Release"なりに設定する事でDebug・Releaseビルドを切り替える事ができる（何も設定されていない場合はDebugビルドになる）。



5. ビルドされたパッケージの確認

DConnectSDKとiOS Pebbleデバイスプラグインのプロジェクトが入ったディレクトリの下に、必要に応じて以下のような名称のディレクトリが生成される。

1. build

そして、DConnectSDKやiOS Pebbleデバイスプラグイン毎のプロジェクトディレクトリ以下、フレームワーク（.framework拡張子ファイル）やバンドル（.bundle拡張子ファイル）がビルドされている。ビルドに失敗する場合は設定ミスの可能性があるので確認すること。

DConnectSDK

- ・（Release時）build/Release-framework/DConnectSDK.framework
- ・（Debug時）build/Debug-framework/DConnectSDK.framework
- ・（Release時）build/Release-resources/dConnectSDK_resources.bundle
- ・（Debug時）build/Debug-resources/dConnectSDK_resources.bundle

iOS Pebbleデバイスプラグイン

- ・（Release時）build/Release-framework/dConnectDevicePebble.framework
- ・（Debug時）build/Debug-framework/dConnectDevicePebble.framework
- ・（Release時）build/Release-framework/dConnectDevicePebble_resources.bundle
- ・（Debug時）build/Debug-framework/dConnectDevicePebble_resources.bundle

アプリを開発する際は、上記フレームワークとリンクさせ、バンドルをコピーさせるようにアプリのビルドターゲットで設定する。

また、上記の物に加え、Pebbleデバイスプラグイン直下にある、下記のフレームワークもビルドターゲットで設定する。

iOS PebbleSDK

- ・ pebbleKit.framework
- ・ pebbleVendor.framework

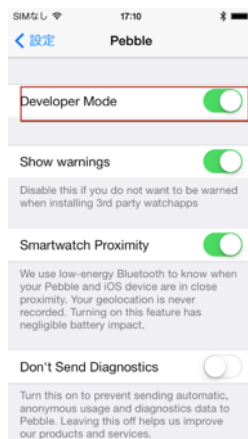
4. PebbleAppのビルドと、iOS Pebble への登録

4.1 iOSの設定

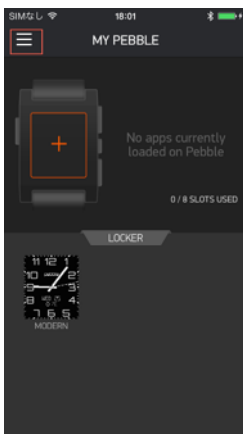
AppleStore からダウンロードした Pebble アプリにて、開発用の設定を行う必要がある。
Pebble にプログラムをインストールする前に、Pebble アプリで必要な設定を行う。
PebbleAppは、ターミナル上でビルドする。



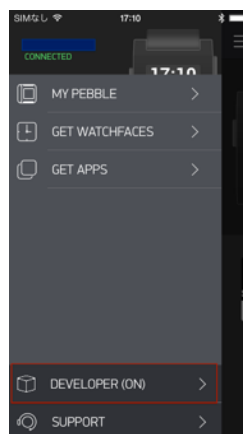
iOSの設定画面
にあるPebble
をタップ



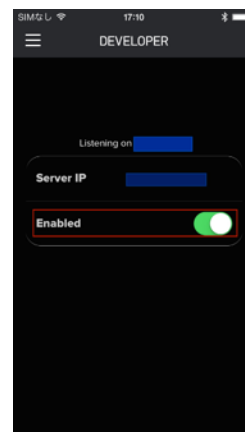
DeveloperMode
を有効化



MY PEBBLE
をタップ



DEVELOPER
をタップ



Enabledを
有効化

4.2 PebbleAppのビルド

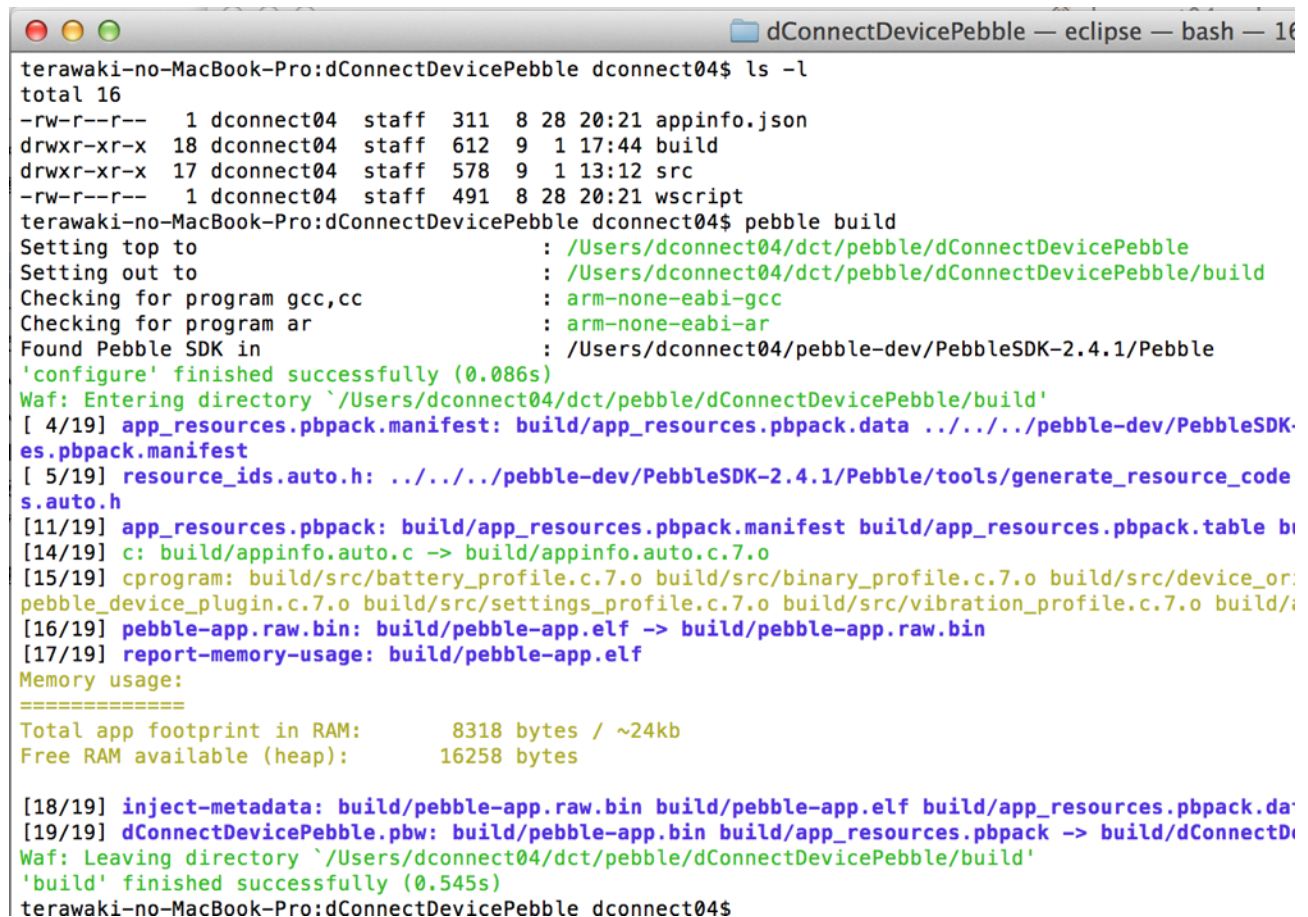
pebbleのライブラリ。以下のサイトの従いpebbleのビルド環境を構築する。

<https://developer.getpebble.com/download-sdk/macosex/>

PebbleAppは、ターミナル上でビルドする。

src ディレクトリが存在するディレクトリにて、以下を実行する。

pebble build



```
terawaki-no-MacBook-Pro:dConnectDevicePebble dconnect04$ ls -l
total 16
-rw-r--r--  1 dconnect04  staff  311  8 28 20:21 appinfo.json
drwxr-xr-x  18 dconnect04  staff  612  9  1 17:44 build
drwxr-xr-x  17 dconnect04  staff  578  9  1 13:12 src
-rw-r--r--  1 dconnect04  staff  491  8 28 20:21 wscript
terawaki-no-MacBook-Pro:dConnectDevicePebble dconnect04$ pebble build
Setting top to           : /Users/dconnect04/dct/pebble/dConnectDevicePebble
Setting out to           : /Users/dconnect04/dct/pebble/dConnectDevicePebble/build
Checking for program gcc,cc : arm-none-eabi-gcc
Checking for program ar    : arm-none-eabi-ar
Found Pebble SDK in       : /Users/dconnect04/pebble-dev/PebbleSDK-2.4.1/Pebble
'configure' finished successfully (0.086s)
Waf: Entering directory `/Users/dconnect04/dct/pebble/dConnectDevicePebble/build'
[ 4/19] app_resources.pbpack.manifest: build/app_resources.pbpack.data ../../pebble-dev/PebbleSDK-2.4.1/app_resources.pbpack.manifest
[ 5/19] resource_ids.auto.h: ../../pebble-dev/PebbleSDK-2.4.1/Pebble/tools/generate_resource_code_s.auto.h
[11/19] app_resources.pbpack: build/app_resources.pbpack.manifest build/app_resources.pbpack.table build/app_resources.pbpack.data
[14/19] c: build/appinfo.auto.c -> build/appinfo.auto.c.7.o
[15/19] cprogram: build/src/battery_profile.c.7.o build/src/binary_profile.c.7.o build/src/device_or_pebble_device_plugin.c.7.o build/src/settings_profile.c.7.o build/src/vibration_profile.c.7.o build/src/watermark.c.7.o
[16/19] pebble-app.raw.bin: build/pebble-app.elf -> build/pebble-app.raw.bin
[17/19] report-memory-usage: build/pebble-app.elf
Memory usage:
=====
Total app footprint in RAM:      8318 bytes / ~24kb
Free RAM available (heap):      16258 bytes

[18/19] inject-metadata: build/pebble-app.raw.bin build/pebble-app.elf build/app_resources.pbpack.data
[19/19] dConnectDevicePebble.pbw: build/pebble-app.bin build/app_resources.pbpack -> build/dConnectDevicePebble.pbw
Waf: Leaving directory `/Users/dconnect04/dct/pebble/dConnectDevicePebble/build'
'build' finished successfully (0.545s)
terawaki-no-MacBook-Pro:dConnectDevicePebble dconnect04$
```

全てを build しなおしたい場合には、以下を実行したのち、再度ビルドを行う。

pebble clean

4.3 dConnectDevicePebbleプロジェクトへの登録

PebbleApp/build/dConnectDevicePebble.pbwを

dConnectDevicePlugin/dConnectDevicePebble/dConnectDevicePebbleResources

にコピーし、更新する。

コピー後には、プロジェクトをクリーンして、再ビルドを行う。

5. 通信プログラム作成時の注意点

5.1 Pebble Cプログラム作成上の注意

(1)使用できない標準関数

itoa() strtok_r() 等は用意されていない。

strtok() 等はコンパイルは通るが、実行時エラーが発生する。

strtok()等、内部で static 変数を使っている標準関数は使えない可能性が高い。

itoa()のかわりに、snprintf(buf, sizeof(buf), "%d", data); を使用する。

(2)Pebble のプログラムに ctype.h で定義されている isdigit() isalpha() 等

これらの関数のどれか1つでも使うと、約300byte プログラムサイズが増える。

現プログラムでは、IsDigit()等の自作関数を使用している。

(3)構造体を定義の、__packed__ 指定

構造体のサイズが最小になるという効果がある。

```
typedef struct __attribute__((__packed__)) {
```

```
    .
```

```
    .
```

```
} TheStruct ;
```

(4)bool は 1byte なので、積極的に使用する。

(5)可変引数マクロ

デバッグ用に以下のようなものを使うと良い。

```
#define DEBUG_MODE //リリース時には、コメントにして build
```

```
#ifdef DEBUG_MODE
```

```
    #define DEBUG_MSG(args...) APP_LOG(APP_LOG_LEVEL_DEBUG, args)
```

```
#else
```

```
    #define DEBUG_MSG(args...)
```

```
#endif
```

5.2 通信プログラム作成上の注意 (iOS側)

sendAppMessagesLaunch にて Pebble 側のアプリケーションを立ち上げた直後1秒程度は安定した通信ができない。

5.3 通信プログラム作成上の注意 (Pebble側)

(1)送信側の問題(iOS,pebble 共に持っている問題)

受信側は、受信終了後 ACK を送信側に返しても、送信側でその ACK を受け取れないことがある。

この場合には送信側は送信エラーと判定するが、受信側は受信が正常に終了したと判断することに注意する。

(2)送受信可能なバイト数は以下の数値に左右される。今回作成したプログラムは、以下の数値以上を設定することが必要である。

```
const int inbound_size = 128;
const int outbound_size = 128;
app_message_open(inbound_size, outbound_size);
```

現行のPebbleOSでは、PebbleDictionary 自体の最大サイズは、124byteとなっている。PebbleDictionary に、2つの bytes を登録した場合の最大サイズは、それぞれは 64byte・44byteである。

文字列として送信可能なのは40文字程度である。

以下の設定でも、この制限は変わらない。

```
int inbound_size = app_message_inbox_size_maximum();
int outbound_size = app_message_outbox_size_maximum();
app_message_open(inbound_size, outbound_size);
```

(3)連続送信

ハンドラー内では、2回以上の送信は不可能である。

タイマーハンドラー等を使用して、送信を時間的に分割する。

(4) Bluetooth の通信速度

消費電力の関係で、デフォルトの状態では Bluetooth の動作速度は遅くなっている。

この状態だと、Pebble が iOS からの ACK を受信するのに2秒以上かかったり、ACK を取りこぼしたりする。

動作速度を上げる為には、app_message_outbox_begin() の直前に以下の2行を追加する。

(常にこの2行が必要であることに注意する。下の1行だけでは動作しない。)

```
app_comm_set_sniff_interval(SNIFF_INTERVAL_REDUCED);
app_comm_set_sniff_interval(SNIFF_INTERVAL_NORMAL);
```

逆に動作速度を下げて消費電力を提言する為には、app_message_outbox_begin()の直前に以下の2行を追加する。

```
app_comm_set_sniff_interval(SNIFF_INTERVAL_NORMAL);
app_comm_set_sniff_interval(SNIFF_INTERVAL_REDUCED);
```

付録A: 更新履歴

変更日時	変更内容
2014/09/10	初版作成。
2014/09/24	バンドルのビルド先変更による記述の修正。
2014/09/24	必要なパッケージのビルドにおける画像の変更。