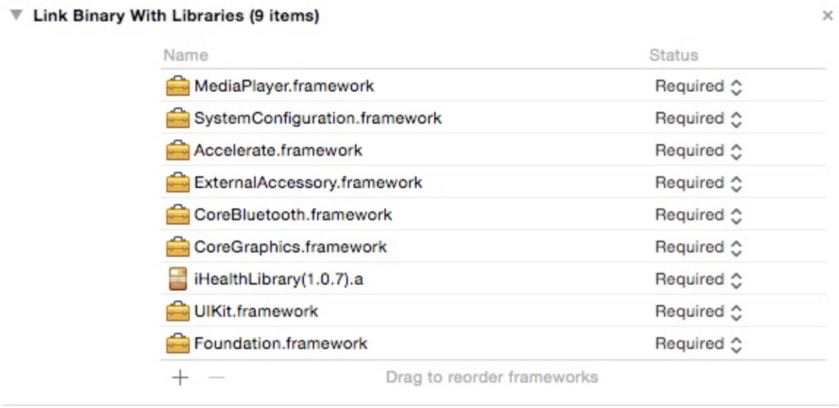
Instruction Manual of BP SDK

1. Relevant file and Frameworks configuration

A. **BPSDK, including**: BPHeader.h, BPMacroFile.h, BP3.h, BP3Controller.h, BP5.h, BP5Controller.h, BP7.h, BP7Controller.h, ABI.h, ABIController.h, BP3L.h, BP3LController.h, BP7S.h, BP7SController.h, KN550BT.h, KN550BTController.h, KD926.h, KD926Controller.h, iHealthLibrary(x.x.x).a, support iOS6.0+

B. Frameworks



Link Binary With Libraries (9 items)

Name	Status
MediaPlayer.framework	Required
SystemConfiguration.framework	Required
Accelerate.framework	Required
ExternalAccessory.framework	Required
CoreBluetooth.framework	Required
CoreGraphics.framework	Required
iHealthLibrary(1.0.7).a	Required
UIKit.framework	Required
Foundation.framework	Required

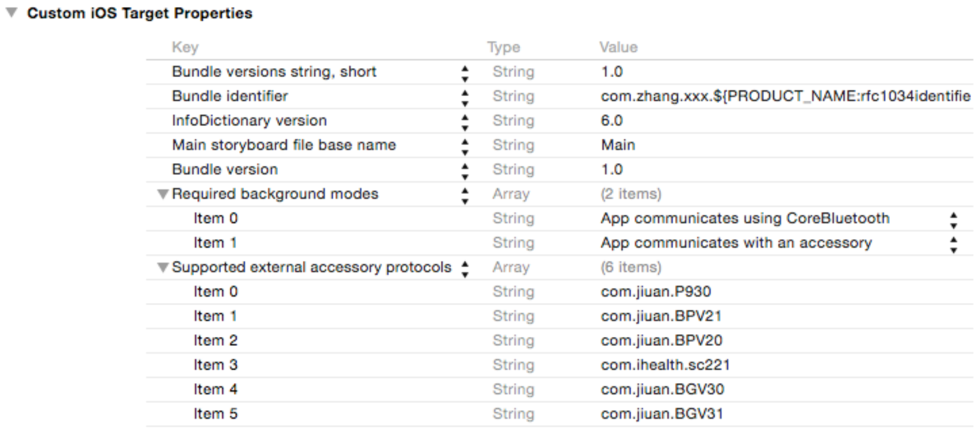
Drag to reorder frameworks

C. Configuration

Add new 'Item' in 'Info':

Add 2 new Item in '**Supported external accessory protocols**': com.jiuan.BPV20, com.jiuan.P930, com.jiuan.BPV21, com.jiuan.BPV24, com.jiuan.BPAV10, com.jiuan.BPV22,

Add 1 new Item in '**Required background modes**': App communicates with an accessory、App communicates using CoreBluetooth



Custom iOS Target Properties

Key	Type	Value
Bundle versions string, short	String	1.0
Bundle identifier	String	com.zhang.xxx.${PRODUCT_NAME:rfc1034identifie
InfoDictionary version	String	6.0
Main storyboard file base name	String	Main
Bundle version	String	1.0
▼Required background modes	Array	(2 items)
Item 0	String	App communicates using CoreBluetooth
Item 1	String	App communicates with an accessory
▼Supported external accessory protocols	Array	(6 items)
Item 0	String	com.jiuan.P930
Item 1	String	com.jiuan.BPV21
Item 2	String	com.jiuan.BPV20
Item 3	String	com.ihealth.sc221
Item 4	String	com.jiuan.BGV30
Item 5	String	com.jiuan.BGV31

2. Operation Procedure

(1) Operation procedure for BP3

a) register plug-in device info：BP3ConnectNoti;

b) initialize controller classes：

```
BP3Controller *controller = [BP3Controller shareBP3Controller];
```

c) Access control class instance after receive BP3ConnectNoti:

```
NSArray *bpDeviceArray = [controller getAllCurrentBP3Instace];
BP3 *bpInstance = [bpDeviceArray objectAtIndex: 0];
```

d) Using ‘bpInstance’ call communication module of the device

(2) Operation procedure for BP5

a) register plug-in device info： BP5ConnectNoti:

b) initialize controller classes：

```
BP5Controller *controller = [BP5Controller shareBP5Controller];
```

c) Access control class instance after receive BP5ConnectNoti:

```
NSArray *bpDeviceArray = [controller getAllCurrentBP5Instace];
BP5 *bpInstance = [bpDeviceArray objectAtIndex: i];
```

d) Using ‘bpInstance’ call communication module of the device

(3) Operation procedure for BP7

a) register plug-in device info： BP7ConnectNoti；

b) initialize controller classes：

```
BP7Controller *controller = [BP7Controller shareBP7Controller];
```

c) Access control class instance after receive BP7ConnectNoti:

```
NSArray *bpDeviceArray = [controller getAllCurrentBP7Instace];
BP7 *bpInstance = [bpDeviceArray objectAtIndex: i];
```

d) Using ‘bpInstance’ call communication module of the device

(4) Operation procedure for ABI

For ABI Mesure（both arm and leg）

a) Register plug-in device info: ABIConnectNoti;

b) Initialized controller class:

```
ABIController *controller = [ABIController
shareABIController];
```

c) Access controller class instance after receive ABIConnectNoti:

```
ABI *bpInstance = [controller getCurrentABIInstace];
```

d) Using ‘bpInstance’ call communication module of the device.

For Arm Mesure（arm only）

a) Register plug-in device info: ArmConnectNoti;

b) Initialized controller class:

```
ABIController *controller = [ABIController
shareABIController];
```

c) Access controller class instance after receive ArmConnectNoti:

```
ABI *bpInstance = [controller getCurrentArmInstance];
```

d) Using ‘bpInstance’ call communication module of the device.

(5) Operation procedure for BP3L

a）register plug-in device info：BP3LConnectNoti;

b）initialize controller classes：

```
        BP3LController *controller = [BP3LController
shareBP3LController];
```

c）Access control class instance after receive BP3LConnectNoti：

```
        NSArray *bpDeviceArray = [controller
getAllCurrentBP3LInstace];
        BP3L *bpInstance = [bpDeviceArray objectAtIndex: 0];
```

Using ‘bpInstance’ call communication module of the device

(6) Operation procedure for BP7S

a）register plug-in device info：BP7SConnectNoti;

b）initialize controller classes：

```
        BP7SController *controller = [BP7SController
shareBP7SController];
```

c）Access control class instance after receive BP7SConnectNoti：

```
        NSArray *bpDeviceArray = [controller
getAllCurrentBP7SInstace];
        BP7S *bpInstance = [bpDeviceArray objectAtIndex: 0];
```

Using ‘bpInstance’ call communication module of the device

(7) Operation procedure for KN550BT

a) register plug-in device info:KN550BTConnectNoti;

b) initialize controller classes:

```
          KN550BTController *controller = [KN550BTController
shareKN550BTController];
```

c) Access control class instance after receive KN550BTConnectNoti:

```
          NSArray *bpDeviceArray = [controller
getAllCurrentKN550BTInstace];
          KN550BT *bpInstance = [bpDeviceArray objectAtIndex: 0];
```

Using 'bpInstance' call communication module of the device

(8) Operation procedure for KD926

a) register plug-in device info:KD926ConnectNoti;

b) initialize controller classes:

```
          KD926Controller *controller = [KD926Controller
shareKD926Controller];
```

c) Access control class instance after receive KD926ConnectNoti:

```
          NSArray *bpDeviceArray = [controller
getAllCurrentKD926Instace];
          KD926 *bpInstance = [bpDeviceArray objectAtIndex: 0];
```

Using 'bp Instance' call communication module of the device

3. Interface Method:

BP3:

a) Establish measurement connection

```
-(void)commandStartMeasureWithUser:(NSString *)userID
clientID:(NSString *)clientID clientSecret:(NSString
*)clientSecret
Authentication:(BlockUserAuthentication)disposeAuthentication
Block pressure:(BlockPressure)pressure
xiaoboWithHeart:(BlockXioaboWithHeart)xiaobo
xiaoboNoHeart:(BlockXioaboNoHeart)xiaoboNoHeart
result:(BlockMesureResult)result errorBlock:(BlockError)error;
```

Import parameters:

'userID', the only identification for the user, by the form of email or cell phone #(cell-phone-# form is not supported temperately)

‘clientID’ and ‘clientsecret’ are the only identification for user of SDK, are required registration from iHealth administrator, please email: lvjincan@ihealthlabs.com.cn for more information.

Return parameter:

disposeAuthenticationBlock is the return parameter of ‘userid’, ’clientID’, ’clientSecret’ after the verification. The interpretation for the verification:

1. UserAuthen_RegisterSuccess, New-user registration succeeded.
2. UserAuthen_LoginSuccess, User login succeeded.
3. UserAuthen_CombinedSuccess, The user is iHealth user as well, measurement via SDK has been activated, and the data from the measurement belongs to the user.
4. UserAuthen_TrySuccess, Testing without Internet connection succeeded.
5. UserAuthen_InvalidateUserInfo, Userid/clientID/clientSecret verification failed.
6. UserAuthen_SDKInvalidateRight, SDK has not been authorized.
7. UserAuthen_UserInvalidateRight, User has not been authorized.
8. UserAuthen_InternetError, Internet error, verification failed.

The measurement via SDK will be operated in the case of 1-4, and will be terminated if any of 5-8 occurs. The interface needs to be re-called after analyzing the return parameters.

Notice: by the first time of new user register via SDK, ‘iHealth disclaimer’ will pop up automatically, and require the user agrees to continue. SDK application requires Internet connection; there is 10-day tryout if SDK cannot connect Internet, SDK is fully functional during tryout period, but will be terminated without verification through Internet after 10 days.

‘Pressure’: Pressure value in the process of measurement, the unit is ‘mmHg’.

- **‘Xiaobo’**: Wavelet data set including pulse rate
- **‘XiaoboNoHeart’**: Wavelet data set without pulse rate
- **‘Result’**: result of the measurement, including systolic pressure, diastolic pressure, pulse rate and irregular judgment. **Relevant key:** time, sys, dia, heartRate, irregular

·'**Error**': refer 'error' in in part 5

b) Measurement termination

```
-(void)stopBPMeassure:(BlockStopResult)result
ErrorBlock:(BlockError)error;
```

Return parameter:

·'**Result**': '**YES**' means measurement has been terminated, '**NO**' means termination failed.
·'**error**': refer 'error' in 'Establish measurement connection'

c) Battery life query

```
-(void)commandEnergy:(BlockEnergyValue)energyValue
errorBlock:(BlockError)error;
```

Return parameter:

·**energyValue**: battery percentage, '**80**' stands for 80%.
·'**error**': refer 'error' in 'Establish measurement connection'

BP5:

a) Establish measurement connection

```
-(void)commandStartMeasureWithUser:(NSString *)userID
clientID:(NSString *)clientID clientSecret:(NSString
*)clientSecret
Authentication:(BlockUserAuthentication)disposeAuthentication
Block pressure:(BlockPressure)pressure
xiaoboWithHeart:(BlockXioaboWithHeart)xiaobo
xiaoboNoHeart:(BlockXioaboNoHeart)xiaoboNoHeart
result:(BlockMesureResult)result errorBlock:(BlockError)error;
```

Import parameters:

'**userID**', '**clientID**', '**clientSecret**',: refer in 'BP3'

Return parameter:

·'**disposeAuthenticationBlock**': refer in 'BP3'
· '**Pressure**': Pressure value in the process of measurement, the unit is 'mmHg'.
·'**Xiaobo**': Wavelet data set including pulse rate
·'**XiaoboNoHeart**': Wavelet data set without pulse rate
·'**Result**': result of the measurement, including systolic pressure, diastolic pressure, pulse rate and irregular judgment. Relevant key: time, sys, dia, heartRate, irregular

·'**error**'： refer 'error' in in part 5

BP7:

a) Establish measurement connection

```
-(void)commandStartGetAngleWithUser:(NSString *)userID
clientID:(NSString *)clientID clientSecret:(NSString
*)clientSecret
Authentication:(BlockUserAuthentication)disposeAuthentication
Block angle:(BlockAngle)angleInfo
errorBlock:(BlockError)error;
```

Import parameters:

'userID', 'clientID', 'clientSecret',: refer in 'BP3'

Return parameter:

'disposeAuthenticationBlock': refer in 'BP3'
'**AngleInfo**': angle info, including **angle**、**isLeftHand.**
'**angle**' stands for the inclination angle of the device, from 0 to 180; '**isLeftHand**' stands for the judgment if the device is on the left hand, '**true**' means left hand, otherwise is right hand.
error'： refer 'error' in in part 5

b) Measurement start:

When 'angle' is between 10-30, measurement starts:

Return parameter:

```
-(void)commandStartMeasure:(BlockPressure)pressure
xiaoboWithHeart:(BlockXioaboWithHeart)xiaobo
xiaoboNoHeart:(BlockXioaboNoHeart)xiaoboNoHeart
result:(BlockMesureResult)result errorBlock:(BlockError)error;
```

- '**Pressure**': Pressure value in the process of measurement, the unit is 'mmHg'.
- '**Xiaobo**': Wavelet data set including pulse rate
- '**XiaoboNoHeart**': Wavelet data set without pulse rate
- '**Result**': result of the measurement, including systolic pressure, diastolic pressure, pulse rate and irregular judgment. Relevant key: time, sys, dia, heartRate, irregular
- '**error**'： refer 'error' in in part 5

BP5 & BP7(self-adaptive)

a) Hypogenous query

```
-(void)commandFounction:(BlockDeviceFounction)founction
errorBlock:(BlockError)error;
```

Return parameter:

Function: judge if the device supports BT auto–connection, offline detection, and if the function on or off, corresponding KEY as haveBlue, haveOffline, blueOpen, offlineOpen. 'True' means yes or on, 'False' means no or off.
'error': refer 'error' in in part 5

b) set up BT auto-connection

```
-(void)commandSetBlueConnect:(BOOL)open
respond:(BlockBlueSet)blockBuleSet
errorBlock:(BlockError)error
```

Import parameter:

Open: True means on; False means off.

Return parameter:

blockBuleSet: device current statues, True means on, False means off.

'error': refer 'error' in in part 5

c) set up offline detection

```
-(void)commandSetOffline:(BOOL)open
respond:(BlockOfflineSet)blockOfflineSet
errorBlock:(BlockError)error
```

Import parameter:

Open: True means on; False means off.

Return parameter:

blockOfflineSet: device current statues, True means on, False means off.

'error': refer 'error' in in part 5

d) Battery life query

```
-(void)commandEnergy:(BlockEnergyValue)energyValue
errorBlock:(BlockError)error
```

Return parameter:

energyValue: battery percentage, '80' stands for 80%.

'error': refer 'error' in in part 5

e) Measurement termination

```
-(void)stopBPMeassureErrorBlock:(BlockStopSuccess)success
errorBlock:(BlockError)error
```

Return parameter:

'Result': **'YES'** means measurement has been terminated, **'NO'** means termination failed.

'error': refer 'error' in in part 5

f) upload offline data

```
-(void)commandBatchUpload:(BlockBachCount)totalCount
pregress:(BlockBachProgress)progress
dataArray:(BlockBachArray)uploadDataArray
errorBlock:(BlockError)error;
```

Return parameter:

TotalCount: item quantity of total data
Progress: upload completion ratio , from 0.0 to 1.0 or **0%~100%, 100%** means upload completed
UploadDataArray:offline data set, including measurement time, systolic pressure, diastolic pressure, pulse rate, irregular judgment. corresponding KEY as time, sys, dia, heartRate, irregular
'error': refer 'error' in in part 5

ABI:

a) Establish measurement connection

```
-(void)commandStartMeasureWithUser:(NSString *)userID
clientID:(NSString *)clientID clientSecret:(NSString
*)clientSecret
Authentication:(BlockUserAuthentication)disposeAuthentication
Block armPressure:(BlockPressure)armPressure
legPressure:(BlockPressure)legPressure
armXiaoboWithHeart:(BlockXioaboWithHeart)armXiaobo
legXiaoboWithHeart:(BlockXioaboWithHeart)legXiaobo
armXiaoboNoHeart:(BlockXioaboNoHeart)armXiaoboNoHeart
legXiaoboNoHeart:(BlockXioaboNoHeart)legXiaoboNoHeart
armResult:(BlockMesureResult)armResult
```

```
legResult:(BlockMesureResult)legResult
errorBlock:(BlockError)error;
```

Import parameters:

‘userID’, ‘clientID’, ‘clientSecret’,: refer in ‘BP3’

Return parameter:

‘disposeAuthenticationBlock’: refer in ‘BP3’
‘armPressure’:Upper-arm blood pressure value during measurement, unit as mmHg
‘legPressure’: Ankle blood pressure value during measurement, unit as mmHg.
‘armXiaobo’：Wavelet value of upper-arm BPM, with heartbeats.
‘legXiaobo’：Wavelet value of ankle BPM, with heartbeats.
‘armXiaoboNoHeart’: Wavelet value of upper-arm BPM, without heartbeats.
‘legXiaoboNoHeart’: Wavelet value of ankle BPM, without heartbeats.
‘armResult’： BP value of upper-arm BPM, including time, sys, dia, heartRate, irregular heartbeat.
‘legResult’：BP value of ankle BPM, including time, sys, dia, heartRate, irregular heartbeat.

‘error’：error codes details in part 5.

b) Stop measuring

```
-(void)stopABIMeassureErrorBlock:(BlockStopSuccess)success
errorBlock:(BlockError)error;
```

Return parameter:

‘Result’: ‘YES’ means measurement has been terminated, ‘NO’ means termination failed.
‘error’: refer ‘error’ in in part 5.

c) Check battery life

```
-(void)commandQueryEnergy:(BlockEnergyValue)armEnergy
leg:(BlockEnergyValue)legEnergy errorBlock:(BlockError)error;
```

Return parameters:

‘armEnergy’：Returns battery ratio of upper-arm BPM, 80 means

80%.

‘legEnergy’： Returns batter ratio of ankle BPM, 80 means 80%

‘error’: refer ‘error’ in in part 5.

d) Establish Arm measurement connection

```
-(void)commandStartMeasureWithUser:(NSString *)userID
clientID:(NSString *)clientID clientSecret:(NSString
*)clientSecret
Authentication:(BlockUserAuthentication)disposeAuthentication
Block armPressure:(BlockPressure)armPressure
armXiaoboWithHeart:(BlockXioaboWithHeart)armXiaobo
armXiaoboNoHeart:(BlockXioaboNoHeart)armXiaoboNoHeart
armResult:(BlockMesureResult)armResult
errorBlock:(BlockError)error;
```

Import parameters:

See Api(1)

Return parameter:

See Api(a)

e) Stop Arm measuring

```
-(void)stopABIArmMeassureBlock:(BlockStopResult)result
errorBlock:(BlockError)error;
```

Return parameter:

See Api(b)

f) Check Arm battery life

```
-(void)commandQueryEnergy:(BlockEnergyValue)armEnergy
errorBlock:(BlockError)error;
```

Return parameters:

See Api(c)

BP3L:

a) Establish measurement connection

```
-(void)commandStartMeasureWithUser:(NSString *)userID
clientID:(NSString *)clientID clientSecret:(NSString
*)clientSecret
Authentication:(BlockUserAuthentication)disposeAuthentication
Block pressure:(BlockPressure)pressure
xiaoboWithHeart:(BlockXioaboWithHeart)xiaobo
xiaoboNoHeart:(BlockXioaboNoHeart)xiaoboNoHeart
result:(BlockMesureResult)result errorBlock:(BlockError)error;
```

Import parameters:

‘userID’, the only identification for the user，by the form of email or cell phone #(cell-phone-# form is not supported temperately).

‘clientID’ and ‘clientsecret’ are the only identification for user of SDK, are required registration from iHealth administrator, please email: lvjincan@ihealthlabs.com.cn for more information.

Return parameter:

‘disposeAuthenticationBlock’ is the return parameter of ‘userid’, ’clientID’, ’clientSecret’ after the verification. The interpretation for the verification:

1. UserAuthen_RegisterSuccess,
New-user registration succeeded.
2. UserAuthen_LoginSuccess, User login succeeded.
3. UserAuthen_CombinedSuccess, The user is iHealth user as well, measurement via SDK has been activated, and the data from the measurement belongs to the user.
4. UserAuthen_TrySuccess, Testing without Internet connection succeeded.
5. UserAuthen_InvalidateUserInfo, Userid/clientID/clientSecret verification failed.
6. UserAuthen_SDKInvalidateRight, SDK has not been authorized.
7. UserAuthen_UserInvalidateRight, User has not been authorized.
8. UserAuthen_InternetError, Internet error, verification failed.

The measurement via SDK will be operated in the case of 1-4, and will be terminated if any of 5-8 occurs. The interface needs to be re-called after analyzing the return parameters.

Notice: by the first time of new user register via SDK, 'iHealth disclaimer' will pop up automatically, and require the user agrees to continue. SDK application requires Internet connection; there is 10-day tryout if SDK cannot connect Internet, SDK is fully functional during tryout period, but will be terminated without verification through Internet after 10 days.

'Pressure': Pressure value in the process of measurement, the unit is 'mmHg'.

- **'Xiaobo'**: Wavelet data set including pulse rate
- **'XiaoboNoHeart'**: Wavelet data set without pulse rate
- **'Result'**: result of the measurement, including systolic pressure, diastolic pressure, pulse rate and irregular judgment. **Relevant key:** sys, dia, heartRate, irregular
- **'Error'**: refer 'error' in in part 5

b) Measurement termination

```
-(void)stopBPMeassure:(BlockStopResult)result
ErrorBlock:(BlockError)error;
```

Return parameter:

- **'Result'**: 'YES' means measurement has been terminated, 'NO' means termination failed.
- **'error'**: refer 'error' in 'Establish measurement connection'

c) Hypogenous query

```
-(void)commandFounction:(BlockDeviceFounction)founction
errorBlock:(BlockError)error;
```

Return parameter:

- **'Function'**: judge if the device supports BT auto-connection, offline detection, and if the function on or off, corresponding KEY as haveBlue, haveOffline, blueOpen, offlineOpen. 'True' means yes or on, 'False' means no or off.
- **'error'**: refer 'error' in in part 5.

d) Battery life query

```
-(void)commandEnergy:(BlockEnergyValue)energyValue
errorBlock:(BlockError)error;
```

Return parameter:

'energyValue'：battery percentage, '80' stands for 80%.

'error': refer 'error' in in part 5.

BP7S:

a) Hypogenous query

```
-(void)commandFounction:(BlockDeviceFounction)founction
errorBlock:(BlockError)error;
```

Return parameter:

- **'Function':** judge if the device supports BT auto-connection, offline detection, and if the function on or off, corresponding KEY as haveAngleSensor、haveAngleSet、haveHSD、haveOffline、mutableUpload、selfUpdate. 'True' means yes or on, 'False' means no or off.
- **'error':** refer 'error' in in part 5.

b) Battery life query

```
-(void)commandEnergy:(BlockEnergyValue)energyValue
errorBlock:(BlockError)error;
```

Return parameter:

'energyValue'：battery percentage, '80' stands for 80%.

'error': refer 'error' in in part 5.

c) upload offline data

```
-(void)commandTransferMemoryDataWithUser:(NSString
*)userID clientID:(NSString *)clientID
clientSecret:(NSString *)clientSecret
Authentication:(BlockUserAuthentication)disposeAuthenticat
ionBlock totalCount:(BlockBachCount)totalCount
pregress:(BlockBachProgress)progress
dataArray:(BlockBachArray)uploadDataArray
errorBlock:(BlockError)error;
```

Import parameters:

'userID', 'clientID', 'clientSecret',: refer in 'BP3'.

Return parameter:

'totalCount': item quantity of total data.
'progress':upload completion ratio , from 0.0 to 1.0 or 0%~100%, 100% means upload completed .
'uploadDataArray': offline data set, including measurement time, systolic pressure, diastolic pressure, pulse rate, irregular judgment. corresponding KEY as time、sys、dia、heartRate、irregular、hsdValue、chooseHand、startAngle、measureAngle. **'error':** refer 'error' in in part 5.

KN550BT:

a) Hypogenous query

```
-(void)commandFounction:(BlockDeviceFounction)founction
errorBlock:(BlockError)error;
```

Return parameter:

- **'Function':** judge if the device supports BT auto-connection, offline detection, and if the function on or off, corresponding KEY as haveBlue, haveOffline, blueOpen, offlineOpen. 'True' means yes or on, 'False' means no or off.
- **'error':** refer 'error' in in part 5.

b) Battery life query

```
-(void)commandEnergy:(BlockEnergyValue)energyValue
errorBlock:(BlockError)error;
```

Return parameter:

'energyValue'：battery percentage, '80' stands for 80%.

'error': refer 'error' in in part 5.

c) upload offline data

```
-(void)commandTransferMemoryDataWithUser:(NSString
*)userID clientID:(NSString *)clientID
clientSecret:(NSString *)clientSecret
Authentication:(BlockUserAuthentication)disposeAuthenticat
ionBlock totalCount:(BlockBachCount)totalCount
pregress:(BlockBachProgress)progress
dataArray:(BlockBachArray)uploadDataArray
```

```
errorBlock:(BlockError)error;
```

Import parameters:

‘userID’, ‘clientID’, ‘clientSecret’,: refer in ‘BP3’.

Return parameter:

‘totalCount’: item quantity of total data.
‘progress’ : upload completion ratio , from 0.0 to 1.0 or 0%~100%, 100% means upload completed .
‘uploadDataArray’: offline data set, including measurement time, systolic pressure, diastolic pressure, pulse rate, irregular judgment. corresponding KEY as time、sys、dia、heartRate、irregular.
‘error’: refer ‘error’ in in part 5.

KD926:

a) Hypogenous query

```
-(void)commandFounction:(BlockDeviceFounction)founction
errorBlock:(BlockError)error;
```

Return parameter:

- ‘Function’: judge if the device supports BT auto-connection, offline detection, and if the function on or off, corresponding KEY as haveBlue, haveOffline, blueOpen, offlineOpen. ‘True’ means yes or on, ‘False’ means no or off.
- ‘error’: refer ‘error’ in in part 5.

b) Battery life query

```
-(void)commandEnergy:(BlockEnergyValue)energyValue
errorBlock:(BlockError)error;
```

Return parameter:

‘energyValue’：battery percentage, ‘80’ stands for 80%.

‘error’: refer ‘error’ in in part 5.

c) upload offline data

```
-(void)commandTransferMemoryDataWithUser:(NSString
*)userID clientID:(NSString *)clientID
clientSecret:(NSString *)clientSecret
Authentication:(BlockUserAuthentication)disposeAuthenticat
ionBlock totalCount:(BlockBachCount)totalCount
pregress:(BlockBachProgress)progress
dataArray:(BlockBachArray)uploadDataArray
errorBlock:(BlockError)error;
```

Import parameters:

'userID', 'clientID', 'clientSecret',: refer in 'BP3'.

Return parameter:

'totalCount': item quantity of total data.
'progress' : upload completion ratio , from **0.0** to **1.0** or **0%~100%, 100%** means upload completed .
'uploadDataArray': offline data set, including measurement time, systolic pressure, diastolic pressure, pulse rate, irregular judgment. corresponding KEY as time、sys、dia、heartRate、irregular.
'error': refer 'error' in in part 5.

4. Parameter supplementary instruction

Device connection status: BP3ConnectNoti, BP5ConnectNoti, BP7ConnectNoti, ABIConnectNoti, BP3LConnectNoti, BP7SConnectNoti, KN550BTConnectNoti, KD926ConnectNoti
Device disconnection status: BP3DisConnectNoti, BP5DisConnectNoti, BP7DisConnectNoti, ABIDisConnectNoti, ArmConnectNoti,ArmDisConnectNoti, BP3LDisConnectNoti, BP7SDisConnectNoti, KN550BTDisConnectNoti, KD926DisConnectNoti
'serialNumber' is for separating different device when multiple device have been connected.

5. Error

```
typedef enum {
                        BPError0 = 0,//Unable to take measurements due to arm/wrist
movements.
                        BPError1,//Failed to detect systolic pressure
                        BPError2,//Failed to detect diastolic pressure
BPError3,//Pneumatic system blocked or cuff is too tight during inflation
BPError4,//Pneumatic system leakage or cuff is too loose during inflation
BPError5,//Cuff pressure reached over 300mmHg
BPError6,//Cuff pressure reached over 15 mmHg for more than 160 seconds
BPError7,//Data retrieving error
BPError8,//Data retrieving error
BPError9,//Data retrieving error
BPError10,//Data retrieving error
BPError11,//Communication Error
BPError12,//Communication Error
BPError13,//Low battery
BPError14,//
BPError15,//Systolic exceeds 260mmHg or diastolic exceeds 199mmHg
BPError16,//Systolic below 60mmHg or diastolic below 40mmHg
BPError17,//Arm/wrist movement beyond range
BPNormalError = 30,//device error, error message displayed automatically
BPOverTimeError,//Abnormal communication
BPNoRespondError,//Abnormal communication
BPBeyondRangeError,//device is out of communication range.
BPDidDisconnect,//device is disconnected.
BPAskToStopMeasure//measurement has been stopped.

}BPDeviceError;
```

6. Demo