eBeam SmartMarker ,Smartpen Developer Guide for iOS

Luidia R&D S/W

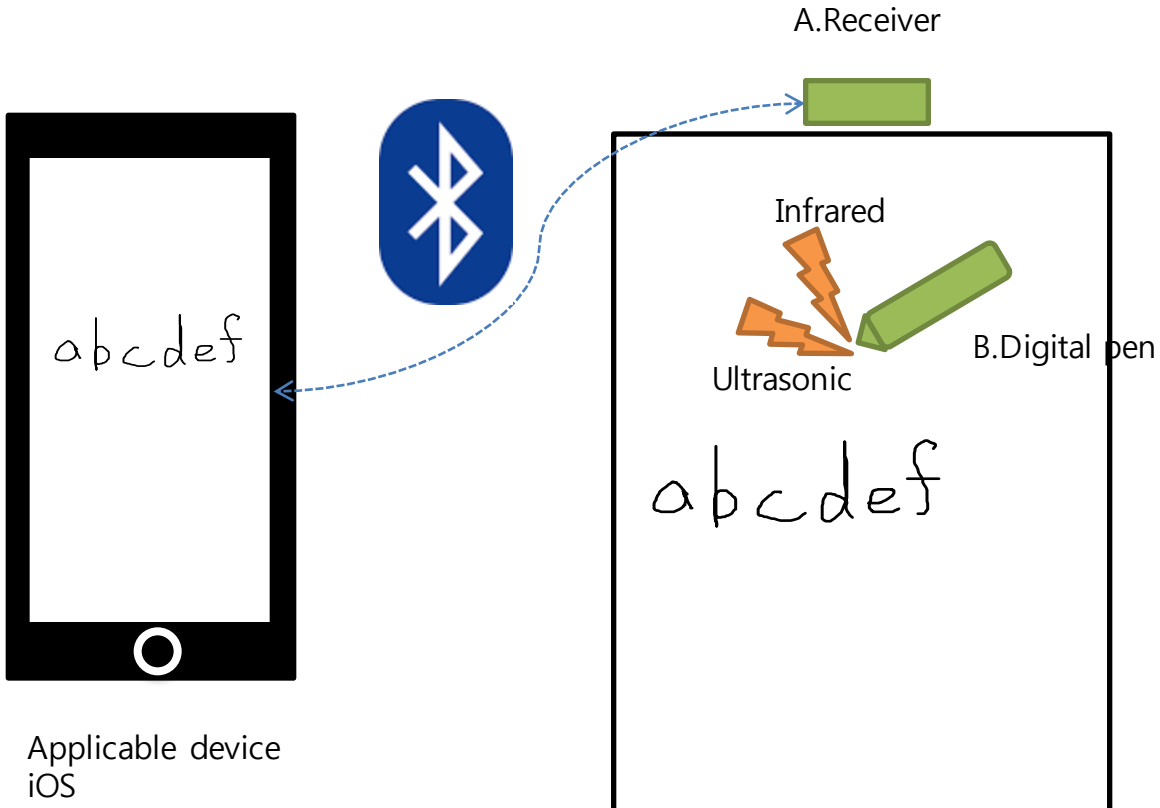
2018. 05

Contents

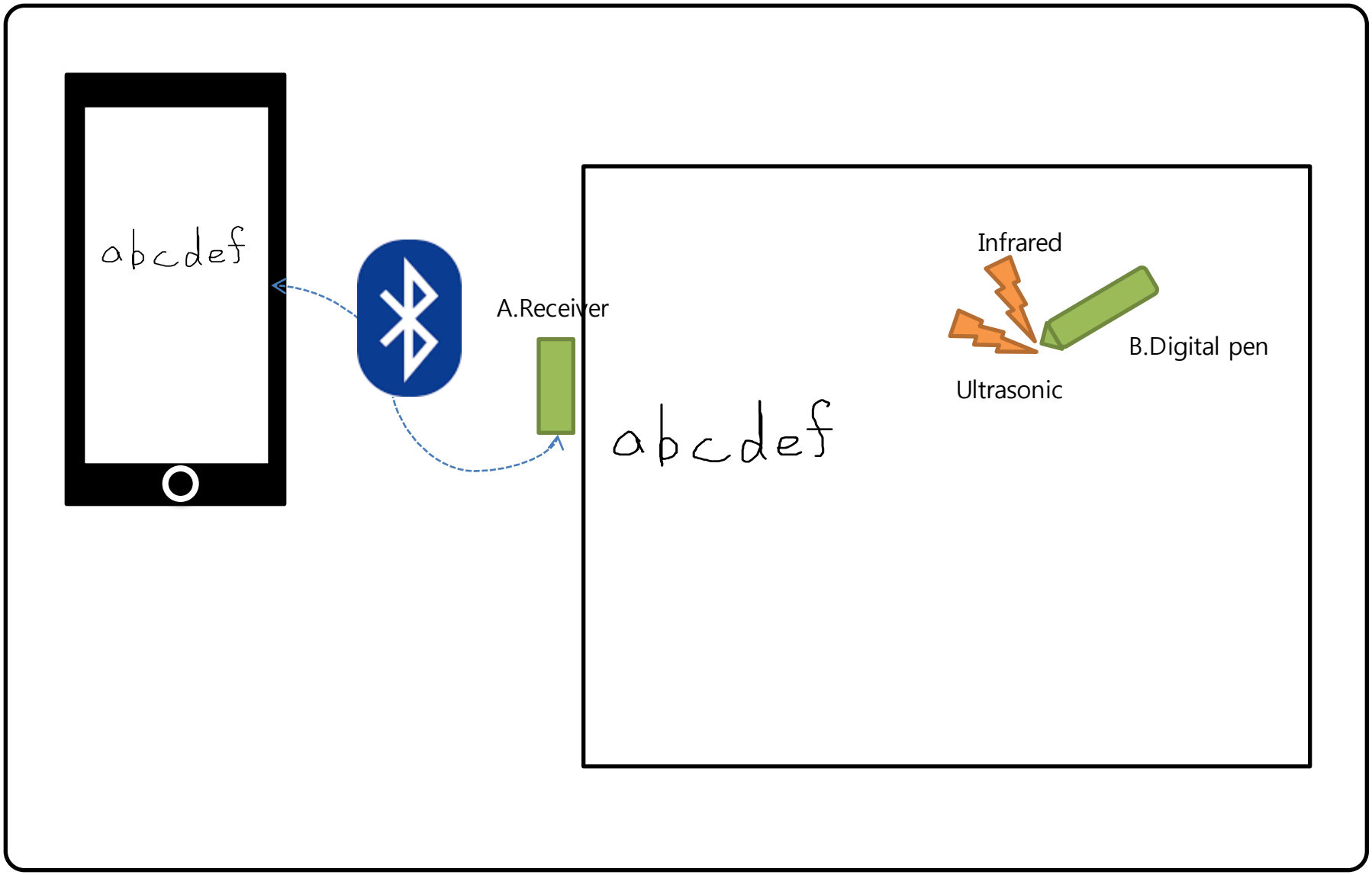
Concept > Luidia Hardwares

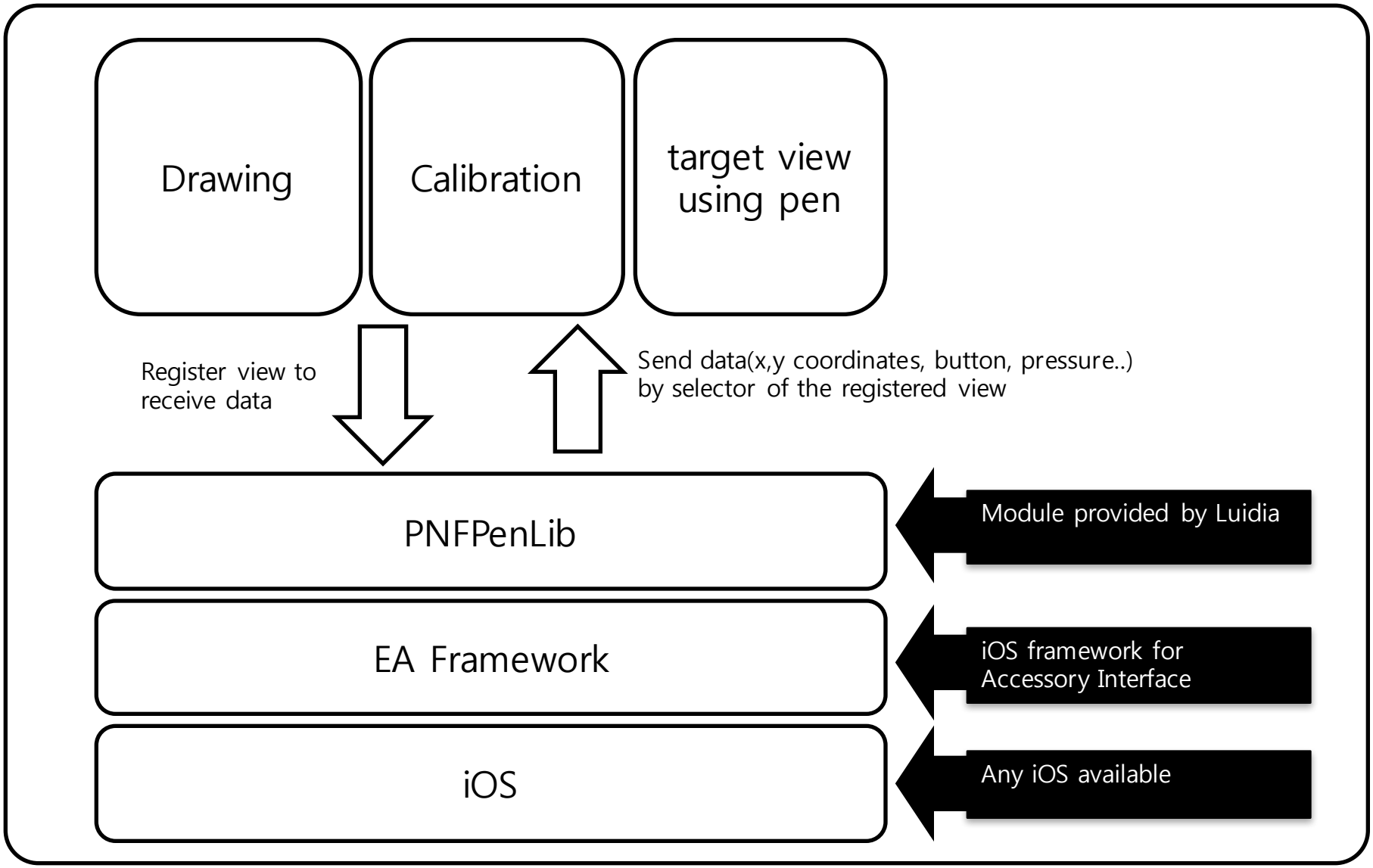
Model	Devices	Connection	Writing	Image
eBeam Smart Pen	iPhone,iPod,iPad, Windows,Android	Wireless(BLE)	On the paper Or desk	
eBeam Smart Marker	iPhone,iPod,iPad, Windows,Android	Wireless(BLE)	On the whiteboard	

Concept > Hardware Structure (eBeam SmartMarker ,Smartpen)



Concept > Hardware Structure (SmartMarker ,Smartpen)




Drawing
Calibration
target view using pen
Register view to receive data
Send data(x,y coordinates, button, pressure..) by selector of the registered view
PNFPenLib
EA Framework
iOS
Module provided by Luidia
iOS framework for Accessory Interface
Any iOS available

- Add PNFModule folder of the sample soures into your project

Development > Components of Test Sample (PenTest)

※ $(SrcHome) : [unZipped folder]/

Folder		File	Description
$(SrcHome)/PenTest/	./	main.m	
		PenTest-Info.plist	
		PenTest-Prefix.pch	
		AppDelegate.h .m	
		ViewController.h .m .xib	Main controller
		BTNameChangeViewController.m	SmartMarker ,Smartpen name change.
	DrawView/	DrawView.h .m	Drawing lines according to the coordinate from pen.
		DrawViewController.h .m .xib	

Development > Components of Test Sample (PenTest)

※ $(SrcHome) : [unZipped folder]/

Folder		File	Description
$(SrcHome)/Common/	Cache/	CacheMgr.m	pen data cache
	Calibration/	PenCalibrationViewController.m .xib MarkerCalibrationViewController.m .xib	2 points calibration view(eBeam SmartPen) 2 points calibration view(eBeam SmartMarker)
	Common/	Toast+UIView.h .m	Shows error information about Pen.
		UIImage+ImageNamed.m	Load image data
		Common.h	Default Calibration value
	PNFModule/	libPNFPenLib.a	Standard library
		PNFDefine.h	Constants
		PNFPenLib.h	Interfaces
		PNFPenLibExtension.h	Interfaces
	PNFStrokePoint/	PNFStrokePoint.h .m	Objects for drawings
	Resource/		

● PNFPenLibExtension Class

Inherits from	NSObject
Declared in	PNFPenLibExtension.h

➢ Overview

PNFPenLibExtension is the class of PNFPenLib Library to manage the information of device , make calibrated coordinates and tranfer it to the other classes.

➢ Members

ptRaw			
Type	CGPoint	Property	readonly
Description	Coordinates before calibrating		
Range	0 ~ 6500		
Device	eBeam SmartMarker ,Smartpen		
Usage			

ptConv			
Type	CGPoint	Property	readonly
Description	Calibrated coordinates		
Range	According to the target view size		
Device	eBeam SmartMarker ,Smartpen		
Usage			

PenStatus			
Type	int	Property	readonly
Description	Where pentip is pressed or not		
Range	PEN_DOWN : Pentip down PEN_MOVE : Move with Pentip down PEN_UP : Pentip up PEN_HOVER : pen hover		
Device	eBeam SmartMarker ,Smartpen		
Usage			

<table>
<tr><td colspan="4">StationPosition</td></tr>
<tr><td>Type</td><td>int</td><td>Property</td><td>readonly</td></tr>
<tr><td>Description</td><td colspan="3">Current position of eBeam SmartMarker station.</td></tr>
<tr><td>Range</td><td colspan="3">DIRECTION_LEFT
DIRECTION_RIGHT
DIRECTION_TOP
DIRECTION_BOTTOM
DIRECTION_BOTH
(defined in PNFDefine.h)</td></tr>
<tr><td>Device</td><td colspan="3">eBeam SmartMarker</td></tr>
<tr><td>Usage</td><td colspan="3">[[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(PenCallBackFunc:) name:@"PNF_MSG" object:nil];

....
....
-(void) PenCallBackFunc:(NSNotification *)call {
if ([szS isEqualToString:@"CHANGE_DEVECE_POSITION"] || [szS isEqualToString:@"CHANGE_DEVECE_POSITION_FIRST"]) {
if (penController.StationPosition == DIRECTION_LEFT)
position = @"Left";
else if (penController.StationPosition == DIRECTION_RIGHT)
position = @"Right";
else if (penController.StationPosition == DIRECTION_TOP)
position = @"Top";
else if (penController.StationPosition == DIRECTION_BOTTOM)
position = @"Bottom";
else
position = @"Both";
}</td></tr>
</table>

bStopped			
Type	BOOL	Property	readonly
Description	Whether Pause is set or not If it is set, Pen data is not transferred to target view.		
Range	Yes / No		
Device	eBeam SmartMarker ,Smartpen		

Usage	[penController stopPen]; // set pause
	NSLog(@"%@", penController.bStopped ? @"YES",@"NO");
	/// display YES
	
	[penController restartPen]; // release pause
	NSLog(@"%@", penController.bStopped ? @"YES",@"NO");
	/// display NO		

AudioMode			
Type	Int	Property	readonly
Description	Audio Mode of Smart Marker		
Range	YES = beep only NO = beep + voice		
Device	eBeam SmartMarker		
Usage			

Volume			
Type	Int	Property	readonly
Description	Audio volume of Smart Marker		
Range	0 ~ 255 0 = loud 255 = slient		
Device	eBeam SmartMarker		
Usage			

battery_station			
Type	Int	Property	readonly
Description	Battery status of sensor		
Range	0 ~ 100		
Device	eBeam SmartMarker ,Smartpen		
Usage			

battery_pen			
Type	Int	Property	readonly
Description	Battery status of pen		
Range	• Smart Marker 0 = High Else = Low • Smart Pen 0 ~ 100		
Device	eBeam SmartMarker ,Smartpen		
Usage			

➢ Methods

BLEInit		
Description	Start to communicate with device	
input	N/A	
Device	eBeam SmartMarker ,Smartpen	

Usage	-(void) viewDidLoad
	{
	.....
	penController = [[[PNFPenLibExtension alloc] init] autorelease];
	#if TARGET_LUIDIA_EBEAMMARKER
	//[eBeam Smart marker device]
	[penController setDefaultModelCode:eBeamSmartMarker];
	[penController setProjectiveLevel:4];
	[penController fixStationPosition:DIRECTION_LEFT];
	#else
	//[eBeam Smart pen device]
	[penController setDefaultModelCode:eBeamSmartPen];
	[penController setProjectiveLevel:4];
	[penController fixStationPosition:DIRECTION_TOP];
	#endif
	[penController BLEInit];
	
	#if TARGET_LUIDIA_CACHEMODE
	self.cacheMgr = [[[CacheMgr alloc] init] autorelease];
	self.cacheMgr.maxQueueCount = 1000;
	self.cacheMgr.delegate = self;
	#endif
	.....
	}	

BLEConnect		
Description	Connect to communicate with device	
out	int	CONNECTED : success FIRST_DATA_RECV : first data read SESSION_CLOSED: receiving error (should reconnect the device) (Define in PNFDefine.h)
input	N/A	
Device	eBeam SmartMarker ,Smartpen	

Usage	-(void) selectBLEDevice:(CBPeripheral *)peripheral
	{
	[penController BLEConnect:peripheral];
	.....
	}	

<table>
<tr><th colspan="3">getBLEPeripheral</th></tr>
<tr><td>Description</td><td colspan="2">Direct connect to communicate with device</td></tr>
<tr><td>out</td><td>int</td><td>CONNECTED : success
FIRST_DATA_RECV : first data read
SESSION_CLOSED: receiving error (should reconnect the device)
(Define in PNFDefine.h)</td></tr>
<tr><td rowspan="2">input</td><td>NSUUID</td><td>Saved CBPeripheral identifier.UUIDString</td></tr>
<tr><td>NSString</td><td>Saved CBPeripheral name</td></tr>
<tr><td>Device</td><td colspan="2">eBeam SmartMarker ,Smartpen</td></tr>
<tr><td>Usage</td><td colspan="2">-(IBAction)bleDirectConnectClicked:(id)sender
{
if (self.penController) {
if (!self.penController.bConnected) {
if(self.saveBLEUUID) {
NSUUID *nsUUID = [[NSUUID alloc] initWithUUIDString:self.saveBLEUUID];

CBPeripheral *savePeripheral = [penController getBLEPeripheral:nsUUID
deviceName:self.saveBLEName];
[self selectBLEDevice:savePeripheral];
}
}
}
}</td></tr>
</table>

BLEDisconnectClicked		
Description	Disconnect device	
out	Void	
input	N/A	
Device	eBeam SmartMarker ,Smartpen	

Usage	- (IBAction)BLEDisconnectClicked:(id)sender {
	[penController BLEDisconnect];
	}	

startCalibrationMode		
Description	Calbration Mode Start.	
out	Void	
input	N/A	
Device	eBeam SmartMarker ,Smartpen	

Usage	- (void)viewDidLoad {
	[m_PenController startCalibrationMode];
	}	

startCalibrationMode		
Description	Calbration Mode End.	
out	Void	
input	N/A	
Device	eBeam SmartMarker ,Smartpen	

Usage	- (void) dealloc {
	[m_PenController endCalibrationMode];
	}	

<table>
<tr><td colspan="3">sendCalibrationDataToDevice</td></tr>
<tr><td>Description</td><td colspan="2">Send data for calibration with position</td></tr>
<tr><td>out</td><td>Void</td><td>[[NSNotificationCenter defaultCenter] addObserver:self
selector:@selector(PenCallBackFunc:) name:@"PNF_MSG" object:nil];</td></tr>
<tr><td rowspan="2">input</td><td>DEVICE_DIRECTION</td><td>position of eBeam device

DIRECTION_LEFT
DIRECTION_RIGHT
DIRECTION_TOP
DIRECTION_BOTTOM
DIRECTION_BOTH
(defined in PNFDefine.h)</td></tr>
<tr><td>CGPoint[]</td><td>Original points</td></tr>
<tr><td>Device</td><td colspan="2">eBeam SmartMarker ,Smartpen</td></tr>
<tr><td>Usage</td><td colspan="2">-(void) runApplyProcess {
// CGPoint m_CaResultPoint[4];
.....
[PenController sendCalibrationDataToDevice:(enum DEVICE_DIRECTION)type
CalibPoint:m_CalResultPoint]];
}

-(void) PenCallBackFunc:(NSNotification *) call {
NSString * szS = (NSString *) [call object];
if ([szS isEqualToString:@"CALIBRATION_SAVE_OK"]) {
.....
}else if([szS isEqualToString:@"CALIBRATION_SAVE_FAIL"] ||
[szS isEqualToString:@"DI_SEND_ERR"]){
.....
}
}</td></tr>
</table>

setCalibration		
Description	Set data for calibration	
out	Void	
input	CGRect	square which consists of calibrated coordinates
	Float	Margin between displayed point and edge of screen
Device	eBeam SmartMarker ,Smartpen	

Usage	-(void) successMarkerCalibrationViewController {
	CGSize drawingSize = [self GetDrawingSizeByCalibration];
	[penController setCalibration:
	scaleRect(CGRectMake(0, 0, drawingSize.width, drawingSize.height))
	GuideMargin:0];
	.....
	}
	
	-(void) successPenCalibrationViewController {
	CGSize drawingSize = [self GetDrawingSizeByCalibration];
	[penController setCalibration:
	scaleRect(CGRectMake(0, 0, drawingSize.width, drawingSize.height))
	GuideMargin:0];
	}	

setProjectiveLevel		
Description	Set calibration points	
out	Void	
input	Int	
Device	eBeam SmartMarker ,Smartpen	

Usage	-(void) viewDidLoad
	{
	.....
	[penController setProjectiveLevel:4];
	.....
	
	}	

<table>
<tr><td colspan="3">changeDeviceName</td></tr>
<tr><td>Description</td><td colspan="2">Send change name data for SmartMarker ,Smartpen</td></tr>
<tr><td>out</td><td>Void</td><td>[[NSNotificationCenter defaultCenter] addObserver:self
selector:@selector(PenCallBackFunc:) name:@"PNF_MSG" object:nil];</td></tr>
<tr><td rowspan="2">input</td><td>NSString</td><td>deviceName</td></tr>
<tr><td></td><td></td></tr>
<tr><td>Device</td><td colspan="2">eBeam SmartMarker ,Smartpen</td></tr>
<tr><td>Usage</td><td colspan="2">-(IBAction)changeClicked:(id)sender {
.....
[m_PenController changeDeviceName:changeName];
}

-(void) PenCallBackFunc:(NSNotification *) call {
NSString * szS = (NSString *) [call object];
if ([szS isEqualToString:@"ChangeDeviceName_OK"]) {
.....
}else if([szS isEqualToString:@"ChangeDeviceName_FAIL"] ||
[szS isEqualToString:@"DI_SEND_ERR"]) {
.....
}
}</td></tr>
</table>

changeAudioMode		
Description	Change Audio mode of Smart Marker	
Out	Void	
Input	BOOL	Yes:/No
Device	eBeam SmartMarker	
Usage	[penController changeAudioMode:YES]; -> Change to beep only [penController changeAudioMode:NO]; -> change to beep and voice	

changeVolume		
Description	Change audio volume	
Out	Void	
Input	int	0 ~ 255
Device	eBeam SmartMarker	
Usage	[penController changeVolume:0]; -> max [penController changeVolume:255]; -> min	

➢ Overview

Set the PNFPenLibExtension.

➢ Example

1. Create PNFPenLibExtension object

```
-(void) viewDidLoad {
    ....
    [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(FreeLogMsg:) name:@"PNF_LOG_MSG" object:nil];
    [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(PenCallBackFunc:) name:@"PNF_MSG" object:nil];
    [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(BLESearchDeviceName:) name:@"BLE_SEARCH_DEVICE_NAME"
    object:nil];
    [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(BLEState:) name:@"PNF_BLE_STATE_MSG" object:nil];
    [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(PenHandlerWithMsg:) name:@"PNF_PEN_READ_DATA" object:nil];

    self.penController = [[[PNFPenLibExtension alloc] init] autorelease];

#if TARGET_LUIDIA_EBEAMMARKER
    [self.penController setDefaultModelCode:eBeamSmartMarker];
    [self.penController setProjectiveLevel:4];
    [self.penController fixStationPosition:DIRECTION_LEFT];
#else
    [self.penController setDefaultModelCode:eBeamSmartPen];
    [self.penController setProjectiveLevel:4];
    [self.penController fixStationPosition:DIRECTION_TOP];
#endif

    [self.penController BLEInit];
    ....
}
```

➢ Overview

Search for and connect to Luidia eBeam Device.

➢ Example

1. Create NSNotificationCenter "BLE_SEARCH_DEVICE_NAME"

```
- (void) viewDidLoad {
    [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(BLESearchDeviceName:)
    name:@"BLE_SEARCH_DEVICE_NAME" object:nil];
}
```

2. Scan button click.

```
-(IBAction)BLEConnectClicked:(id)sender {
    ....
    [penController BLEScan];
    ....
}
```

3. Callback BLESearchDeviceName

```
-(void) BLESearchDeviceName:(NSNotification *) obj {
    NSMutableDictionary * bleObj = (NSMutableDictionary *) [obj object];
    if (self.bleSearchController) {
//        CBPeripheral* peripheral = [bleObj objectForKey:@"peripheral"];
//        if([peripheral.name containsString:@"eSM"] || [peripheral.name containsString:@"eBP"])
          {
              [self.bleSearchController.deviceList addObject:bleObj];
              [self.bleSearchController refresh];
          }
     }
}
```

4. Connect to eBeam Device

```
-(void) selectBLEDevice:(CBPeripheral *)peripheral {
    [self.penController BLEConnect:peripheral];
    ....
}
```

➢ Overview

Direct connect to Luidia eBeam Device.

➢ Example

1. Save CBPeripheral UUIDString and name

```
if ([szS isEqualToString:@"FIRST_DATA_RECV"]) {
    [[NSUserDefaults standardUserDefaults] setObject:self.saveBLEUUID forKey:@"eBeamDeviceUUID"];
    [[NSUserDefaults standardUserDefaults] setObject:self.saveBLEName forKey:@"eBeamDeviceName"];
    [[NSUserDefaults standardUserDefaults] synchronize];
}
```

2. Direct connect button click.

```
-(IBAction)bleDirectConnectClicked:(id)sender {
    if(self.saveBLEUUID)
    {
        NSUUID *nsUUID = [[NSUUID alloc] initWithUUIDString:self.saveBLEUUID];
        CBPeripheral *savePeripheral = [self.penController getBLEPeripheral:nsUUID deviceName:self.saveBLEName];
        [self selectBLEDevice:savePeripheral];
    }
}
```

3. Connect to eBeam Device

```
-(void) selectBLEDevice:(CBPeripheral *)peripheral {
    [self.penController BLEConnect:peripheral];
    ....
}
```

➢ Overview

Convert data received from the Luidia eBeam Device

➢ Example

```
#define TARGET_LUIDIA_CACHEMODE 0

1. Create NSNotificationCenter "PNF_PEN_READ_DATA"
-(void) viewDidLoad {
    ....
    [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(PenHandlerWithMsg:)
     name:@"PNF_PEN_READ_DATA" object:nil];
    ....
}

2. Convert received data
-(void) PenHandlerWithMsg:(NSNotification*) note {
    NSDictionary* dic = [note object];
    [self PenHandlerWithDictionary:dic];
}
-(void) PenHandlerWithDictionary:(NSDictionary*) dic {
    int PenStatus = [[dic objectForKey:@"PenStatus"] intValue];
    CGPoint ptRaw = [[dic objectForKey:@"ptRaw"] CGPointValue];
    CGPoint ptConv = [[dic objectForKey:@"ptConv"] CGPointValue];
    int Temperature = [[dic objectForKey:@"Temperature"] intValue];
    int modelCode = [[dic objectForKey:@"modelCode"] intValue];
    int SMPenFlag = [[dic objectForKey:@"SMPenFlag"] intValue];
    int SMPenState = [[dic objectForKey:@"SMPenState"] intValue];
    int press = [[dic objectForKey:@"pressure"] intValue];
    int packetIndex = 0;
    [self PenHandlerWithArgs:ptRaw
                      ptConv:ptConv
                   PenStatus:PenStatus
                 Temperature:Temperature
                   ModelCode:modelCode
                   SMPenFlag:SMPenFlag
                  SMPenState:SMPenState
                    Pressure:press PacketIndex:packetIndex];
}
```

➢ Example

```
-(void) PenHandlerWithArgs:(CGPoint) Arg_ptRaw ptConv:(CGPoint) Arg_ptConv PenStatus:(int) Arg_PenStatus
            Temperature:(int) Arg_Temperature ModelCode:(int) Arg_modelCode
             SMPenFlag :(int) Arg_SMPenFlag SMPenState:(int) Arg_SMPenState
               Pressure:(int) Arg_pressure {
        CGPoint ptDrawing;
        switch (Arg_PenStatus) {
            case PEN_DOWN:
                break;
            case PEN_MOVE:
                break;
            case PEN_UP:
                break;
        }
    }
    ptDrawing = PenController.ptConv ;
}
```

➢ Example

```
#define TARGET_LUIDIA_CACHEMODE 1

1. Create NSNotificationCenter "PNF_PEN_READ_DATA"
-(void) viewDidLoad {
    [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(PenHandlerWithMsg:)
     name:@"PNF_PEN_READ_DATA" object:nil];

    self.cacheMgr = [[[CacheMgr alloc] init] autorelease];
    self.cacheMgr.maxQueueCount = 1000;
    self.cacheMgr.delegate = self;
}
2. Convert received data
-(void) PenHandlerWithMsg:(NSNotification*) note {
    NSDictionary* dic = [note object];
    [self PenHandlerWithDictionary:dic];
}
-(void) PenHandlerWithDictionary:(NSDictionary*) dic {
    int PenStatus = [[dic objectForKey:@"PenStatus"] intValue];
    CGPoint ptRaw = [[dic objectForKey:@"ptRaw"] CGPointValue];
    CGPoint ptConv = [[dic objectForKey:@"ptConv"] CGPointValue];
    int Temperature = [[dic objectForKey:@"Temperature"] intValue];
    int modelCode = [[dic objectForKey:@"modelCode"] intValue];
    int SMPenFlag = [[dic objectForKey:@"SMPenFlag"] intValue];
    int SMPenState = [[dic objectForKey:@"SMPenState"] intValue];
    int press = [[dic objectForKey:@"pressure"] intValue];
    int packetIndex = 0;
    if (PenStatus == PEN_HOVER || PenStatus == PEN_HOVER_DOWN || PenStatus == PEN_HOVER_MOVE) {
        [self PenHandlerWithArgs:ptRaw
                                ptConv:ptConv
                             PenStatus:PenStatus
                           Temperature:Temperature
                             ModelCode:modelCode
                             SMPenFlag:SMPenFlag
                            SMPenState:SMPenState
                      Pressure:press PacketIndex:packetIndex];
     }else{
        [self.cacheMgr addObject:dic];
     }
}
```

➢ Example

```
-(void) CacheMgrPullData:(id)obj  {
    if (obj) {
      if ([obj isKindOfClass:[NSDictionary  class]]) {
          NSDictionary* dic = (NSDictionary*)obj;

          int PenStatus   = [[dic objectForKey:@"PenStatus"]  intValue];
          CGPoint ptRaw = [[dic objectForKey:@"ptRaw"]  CGPointValue];
          CGPoint ptConv = [[dic objectForKey:@"ptConv"] CGPointValue];
          int Temperature = [[dic objectForKey:@"Temperature"]  intValue];
          int modelCode = [[dic objectForKey:@"modelCode"]  intValue];
          int SMPenFlag = [[dic objectForKey:@"SMPenFlag"]  intValue];
          int SMPenState = [[dic objectForKey:@"SMPenState"]  intValue];
          int press = [[dic objectForKey:@"pressure"] intValue];
           int packetIndex = 0;
          [self PenHandlerWithArgs:ptRaw
                                  ptConv:ptConv
                               PenStatus:PenStatus
                             Temperature:Temperature
                               ModelCode:modelCode
                               SMPenFlag:SMPenFlag
                              SMPenState:SMPenState
                        Pressure:press PacketIndex:packetIndex];
        }
    }

   [self.cacheMgr  playWithPull];
}
```

➢ Example

```
-(void) PenHandlerWithArgs:(CGPoint) Arg_ptRaw ptConv:(CGPoint) Arg_ptConv PenStatus:(int) Arg_PenStatus
            Temperature:(int) Arg_Temperature ModelCode:(int) Arg_modelCode
             SMPenFlag :(int) Arg_SMPenFlag SMPenState:(int) Arg_SMPenState
               Pressure:(int) Arg_pressure {
        CGPoint ptDrawing;
        switch (Arg_PenStatus) {
            case PEN_DOWN:
                break;
            case PEN_MOVE:
                break;
            case PEN_UP:
                break;
        }
    }
    ptDrawing = PenController.ptConv ;
}
```

➢ Overview

Information of device status is sent by notification named as "PNF_LOG_MSG".

➢ Example

1. Add Notification

```
[[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(FreeLogMsg:)
name:@"PNF_LOG_MSG" object:nil];
```

2. Handler for Message

```
-(void) FreeLogMsg:(NSNotification *) note
{
    NSString * szS = (NSString *) [note object];
    if ([szS isEqualToString :@"FAIL_LISTENING"] ) {
    }
    else if ([szS isEqualToString:@"CONNECTED"]) {
    }
    else if ([szS isEqualToString:@"INVALID_PROTOCOL"]) {
    }
    else if ([szS isEqualToString:@"SESSION_CLOSED"]) {
    }
    else if ([szS isEqualToString:@"PEN_RMD_ERROR"]) {

    }
    else if ([szS isEqualToString:@"FIRST_DATA_RECV"]) {
    }
}
```

Log String Message	Description
CONNECTED	Device is connected
NOT_CONNECTED	Device is disconnected
FAIL_LISTENING	Fail to receive. Need to reconnect.
INVALID_PROTOCOL	Invalid hardware
SESSION_CLOSED	Session is disconnected
FIRST_DATA_RECV	First data is received after connecting
PEN_RMD_ERROR	Abnormal drawing data

example source : ViewController.h ViewController.m

➢ Overview

Information of device status is sent by notification named as "PNF_MSG".

➢ Example

1. Add Notification

```
[[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(PenCallBackFunc:)
name:@"PNF_MSG" object:nil];
```

2. Handler for Message

```
-(void) PenCallBackFunc:(NSNotification *)call {
    NSString * szS = (NSString *) [call object];
    if([szS isEqualToString:@"BATTERY_INFO"]) {
    }
    else if([szS isEqualToString:@"NEW_PAGE"]) {
    }
    else if([szS isEqualToString:@"DUPLICATE_PAGE"]) {
    }
    else if ([szS isEqualToString:@"CHANGE_DEVECE_POSITION"]) {
    }
    else if([szS isEqualToString:@"CHANGE_DEVECE_POSITION_FIRST"]) {
    }
    else if([szS isEqualToString:@"DI_SEND_ERR"]) {
    }
    else if([szS isEqualToString:@"ChangeDeviceName_OK"]) {
    }
    else if([szS isEqualToString:@"ChangeDeviceName_FAIL"]) {
    }
    else if([szS isEqualToString:@"CALIBRATION_SAVE_OK"]) {
    }
    else if([szS isEqualToString:@"CALIBRATION_SAVE_FAIL"]) {
    }
}
```

Log String Message	Description
BATTERY_INFO	Battery information
NEW_PAGE	Button smartMarker ,Smartpen
DUPLICATE_PAGE	Long press button smartMarker ,Smartpen
CHANGE_DEVECE_POSITION	Change device position
CHANGE_DEVECE_POSITION_FIRST	Change device position first
DI_SEND_ERR	Send data fail
ChangeDeviceName_OK	Device name change success
ChangeDeviceName_FAIL	Device name change fail
CALIBRATION_SAVE_OK	Calibration change success
CALIBRATION_SAVE_FAIL	Calibration change fail

example source : ViewController.h ViewController.m

➢ Overview

Pen coordinates is converted to screen coordinates by projective matrix which is set in the calibration view.

➢ Example

1. create calibration controller

```
if (penController.modelCode == eBeamSmartMarker) {
    MarkerCalibrationViewController* cVController = [[MarkerCalibrationViewController alloc]
    initWithNibName:@"MarkerCalibrationViewController" bundle:nil];
} else {
    PenCalibrationViewController* cVController = [[PenCalibrationViewController alloc]
    initWithNibName:@"PenCalibrationViewController" bundle:nil];
}
```

2. connect Pen controller and calibration controller

```
[cVController SetPenController:penController];
```

3. show calibration view

```
[self presentViewController:cVController animated:NO completion:^{ }];
```

➢ Overview

Transfers coordinate data to the SmartMarker,Smartpen.

➢ Example

4. Create NSNotificationCenter "PNF_PEN_READ_DATA" ," PNF_LOG_MSG" ," PNF_MSG" and Calibration Mode Start.

```
- (void)viewDidLoad
{
    [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(PenHandlerWithMsg:)
    name:@"PNF_PEN_READ_DATA" object:nil];
    [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(FreeLogMsg:)
    name:@"PNF_LOG_MSG" object:nil];
    [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(PenCallBackFunc:)
    name:@"PNF_MSG" object:nil];

    [m_PenController startCalibrationMode];
}
```

5. send calibration data

```
-(void) runApplyProcess {
  CGPoint  m_CaResultPoint[4];
  ……
  [PenController sendCalibrationDataToDevice:(enum  DEVICE_DIRECTION)type CalibPoint:m_CalResultPoint]];
}
```

➢ Overview

Calibration data is saved automatically by this library.
App need not save the data.

➢ Example

6. Receive callback calibration data

```
-(void) PenCallBackFunc:(NSNotification *) call {
        NSString * szS = (NSString *) [call object];
        if ([szS isEqualToString:@"CALIBRATION_SAVE_OK"]) {
            ……
        }else if ([szS isEqualToString:@"CALIBRATION_SAVE_FAIL"] || [szS isEqualToString:@"DI_SEND_ERR"]) {
            ……
        }
 }
```

7. Save calibration data

```
-(void) successPenCalibrationViewController {
    /// after click the last calibration point
    CGSize drawingSize = [self GetDrawingSizeByCalibration];
    [penController setCalibration:
                        scaleRect(CGRectMake(0, 0, drawingSize.width, drawingSize.height))
                        GuideMargin:0];
```

8. End Calibration Mode

```
- (void)dealloc
   [m_PenController endCalibrationMode];
}
```