1. Development environment

a. Development tool: Xcode

b. This SDK is applicable to systems above macOS 10.11

2. Integrated SDK

a. New xcode project

b. Add Selpic.framework to the new project

c. Import the <SelpicM/SelpicM.h> framework in the viewController that needs to reference the SDK interface, and follow the SelPicDelegate protocol, as shown below

```
#import "mainViewController.h"
#import <SelpicM/SelpicM.h>

@interface mainViewController ()<SelPicDelegate>
```

d.Initialize the SDK in the viewController initialization method, as shown below

```
[SelpicManager sharedManager].delegate = self;
```

3. Introduction to SDK interface

a. Obtain device information

```
[[SelpicManager sharedManager]getDeviceInfo:^(DeviceInfoModel *
_Nonnull info, NSError * _Nonnull error) {
        if (!error) {
```

```
        if (!error) {
            NSLog(@"info-
BatteryPercentage:%d",info.BatteryPercentage);
            NSLog(@"info-deviceType:%@",info.deviceType);
            NSLog(@"info-isCharging:%d",info.isCharging);
            NSLog(@"info-versionIndex:%d",info.versionIndex);
            NSString *ischargeStr;
            if (info.isCharging) {
                ischargeStr = @"charging...";
            }else{
                ischargeStr = @"Non-charged";
            }
            [self->BatteryPercentageLab setTitle:[NSString
stringWithFormat:@"BatteryPercent:%d%%",info.BatteryPercentage]];
            [self->deviceTypeLab setTitle:[NSString
stringWithFormat:@"deviceType:%@",info.deviceType]];
            [self->isChargingLab setTitle:ischargeStr];
            [self->versionIndexLab setTitle:[NSString
stringWithFormat:@"version:%d",info.versionIndex]];
        }else{
            NSLog(@"connect failure");
        }
    }];
```

b. Set printing parameters

First, initialize a PrintParmModel object and assign a value

```
    PrintParmModel *parModel = [PrintParmModel new];
    parModel.PrtGrayScale = 1;
    parModel.PrtPlusWidth = 6;
    parModel.prtVoltage = 1;
```

Then call the interface

```
[[SelpicManager sharedManager]setPrintParam:parModel
Complite:^(NSError * _Nonnull error) {
        if (!error) {
```

```
        if (!error) {
            NSLog(@"set PrintParam success!");
        }else{
            NSLog(@"failure!");
        }
    }];
```

c.Send print data

```
 NSImage *img = disPlayImgview.image;
    [[SelpicManager sharedManager]sendPrintData:img
Complite:^(NSError * _Nonnull error) {
        if (!error) {
            NSLog(@"send data sucess....");
        }else{
            NSLog(@"failure");
        }
    }];
```

SelPicDelegate protocol method

```
// Device connection, disconnection, and protocol method for reading
and writing data
-(void)RecvDeviceStatus:(RecvDataModel*)recvModel{
    if (recvModel.deviceStaus == DidWriteData) {
        NSLog(@"deviceStaus:--DidWriteData");
    }else if (recvModel.deviceStaus == DidConnectDevice){
         NSLog(@"deviceStaus:--DidConnectDevice");
    }else if (recvModel.deviceStaus == DidReadData){
         NSLog(@"deviceStaus:--DidReadData");
    }else if (recvModel.deviceStaus == DidDisConnectDevice){
        NSLog(@"deviceStaus:--DidDisConnectDevice");
    }
}
```

```
//Protocol method for device plug-in power supply
-(void)RecvData:(RecvDataModel*)recvModel{
     NSLog(@"BatteryStatus:%d",recvModel.BatteryStatus);
```

```
    NSLog(@"BatteryStatus:%d",recvModel.BatteryStatus);
    if (recvModel.BatteryStatus) {
        NSLog(@"charging...");
    }else{
        NSLog(@"Non-charged state");
    }
}
```