

[杰理AC692x系列OTA开发说明（iOS平台）](#header-n229)  
 [声明](#header-n343)  
 [版本](#header-n239)  
 [概述](#header-n251)  
 [1、导入JL\_BLEKit.framework](#header-n257)  
 [2、实例化SDK](#header-n259)  
 [3、执行OTA升级](#header-n263)

# 杰理AC692x系列OTA开发说明（iOS平台）

* 对应的芯片类型：AC692x，BD29
* 对应的APP开发环境：iOS平台，iOS 10.0以上，Xcode 11.0以上

## 声明

1. 本项⽬所参考、使⽤技术必须全部来源于公知技术信息，或⾃主创新设计。
2. 本项⽬不得使⽤任何未经授权的第三⽅知识产权的技术信息。
3. 如个⼈使⽤未经授权的第三⽅知识产权的技术信息，造成的经济损失和法律后果由个⼈承担。

## 版本

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 日期 | 编辑 | 修改内容 |
| v1.0 | 2019年09月09日 | 冯 洪鹏 | OTA升级功能 |
| v1.1 | 2020年04月20日 | 冯 洪鹏 | 增加升级的错误回调 |

## 概述

本文档是为了后续开发者更加便捷移植杰理OTA升级功能而创建。

## 1、导入JL\_BLEKit.framework

将*JL\_BLEKit.framework*导入Xcode工程项目里，添加*Privacy - Bluetooth Peripheral Usage Description*和*Privacy - Bluetooth Always Usage Description*两个权限。

### 2、实例化SDK

1.导入JL\_RunSDK.h文件；  
  
键入以下代码：  
 - (BOOL)application:(UIApplication \*)application didFinishLaunchingWithOptions:(NSDictionary \*)launchOptions {  
  
 runSdk = [[JL\_RunSDK alloc] init];  
  
 return YES;  
}  
   
//升级界面，写入以下代码  
- (void)viewDidLoad {  
 [super viewDidLoad];  
   
 //监听强制升级通知  
 [JL\_Tools add:kUI\_JL\_UPDATE\_STATUS Action:@selector(noteOtaUpdate:) Own:self];  
}

### 3、执行OTA升级

-(void)noteOtaUpdate:(NSNotification\*)note{  
 if (\_selectPath.length == 0) return;  
 \_otaData = [NSData dataWithContentsOfFile:\_selectPath];  
   
 [JL\_Manager cmdOTAData:self.otaData Result:^(JL\_OTAResult result, float progress) {  
 if (result == JL\_OTAResultUpgrading ||  
 result == JL\_OTAResultPreparing)  
 {  
 [self isUpdatingUI:YES];  
 //NSLog(@"%.1f%%",progress\*100.0f);  
 NSString \*txt = [NSString stringWithFormat:@"%.1f%%",progress\*100.0f];  
 self.updateSeek.text = txt;  
 self.updateProgress.progress = progress;  
   
 if (result == JL\_OTAResultPreparing) self.updateTxt.text = kJL\_TXT("校验文件中");  
 if (result == JL\_OTAResultUpgrading) self.updateTxt.text = kJL\_TXT("正在升级");  
  
 [self otaTimeCheck];//增加超时检测  
 }else{  
 [self otaTimeClose];//关闭超时检测  
 }  
  
   
 if (result == JL\_OTAResultSuccess) {  
 NSLog(@"OTA 升级完成.");  
 self.updateTxt.text = kJL\_TXT("升级完成");  
 self.updateProgress.progress = 1.0;  
 }  
   
 if (result == JL\_OTAResultReboot) {  
 NSLog(@"OTA 设备准备重启.");  
 //self.updateTxt.text = kJL\_TXT("设备准备重启");  
 self.updateTxt.text = kJL\_TXT("升级完成");  
 [DFUITools showText:kJL\_TXT("升级完成") onView:self.view delay:1.0];  
  
 [DFAction delay:2.5 Task:^{  
 [self isUpdatingUI:NO];  
 [JL\_Tools post:@"UI\_CHANEG\_VC" Object:@(1)];  
 }];  
 }  
   
 if (result == JL\_OTAResultFailCompletely) {  
 self.updateTxt.text = kJL\_TXT("升级失败");  
 [DFUITools showText:kJL\_TXT("升级失败") onView:self.view delay:1.0];  
  
 [DFAction delay:2.5 Task:^{  
 [self isUpdatingUI:NO];  
 }];  
 }  
   
 if (result == JL\_OTAResultFailKey) {  
 self.updateTxt.text = kJL\_TXT("升级文件KEY错误");  
 [DFUITools showText:kJL\_TXT("升级文件KEY错误") onView:self.view delay:1.0];  
  
 [DFAction delay:2.5 Task:^{  
 [self isUpdatingUI:NO];  
 }];  
 }  
 }];  
}