|  |  |
| --- | --- |
| MiCO Documentation Working Group (MDWG) | Chaz Shi |
| Track Number: 0008CN | MXCHIP Co., Ltd |
| Version: 1.0 | 2015.10.10 |
| Category: Technical Manual |  |

APP控制智能工位牌开发实例

摘要（Abstract）

本文档仅介绍如何使用**MiCOKit开发套件（MiCOKit-3288或MiCOKit-3165）**开发一个简单的，使用APP控制智能工位牌的应用实例过程。

适用对象（Suitable Readers）

本实例适用于MiCOKit开发套件+智能工位牌硬件的开发者。

并适合所有MiCO-物联网（IoT）设备开发者参考。

版权声明（Copyright Notice）

Copyright (c) 2015 MDWG Trust and the persons identified as the document authors. All rights reserved.

**目 录**

[APP控制智能工位牌开发实例 1](#_Toc432259648)

[1. 概述 2](#_Toc432259649)

[2. 准备工作 2](#_Toc432259650)

[2.1. 注册并登录 2](#_Toc432259651)

[2.2. 登录IDE 2](#_Toc432259652)

[2.3. 同步项目 3](#_Toc432259653)

[2.4. 应用包结构 4](#_Toc432259654)

[3. 项目移植 4](#_Toc432259655)

[3.1. 代码移植 4](#_Toc432259656)

[3.2. 应用设置 6](#_Toc432259657)

[3.3. 添加模块 8](#_Toc432259658)

[3.4. 云编译 10](#_Toc432259659)

[3.5. 生成APP 10](#_Toc432259660)

[3.6. 效果图展示 11](#_Toc432259661)

[4. 项目解析 11](#_Toc432259662)

[4.1. 设备列表 12](#_Toc432259663)

[4.2. 控制设备 13](#_Toc432259664)

[5. 版本更新说明 14](#_Toc432259665)

# 概述

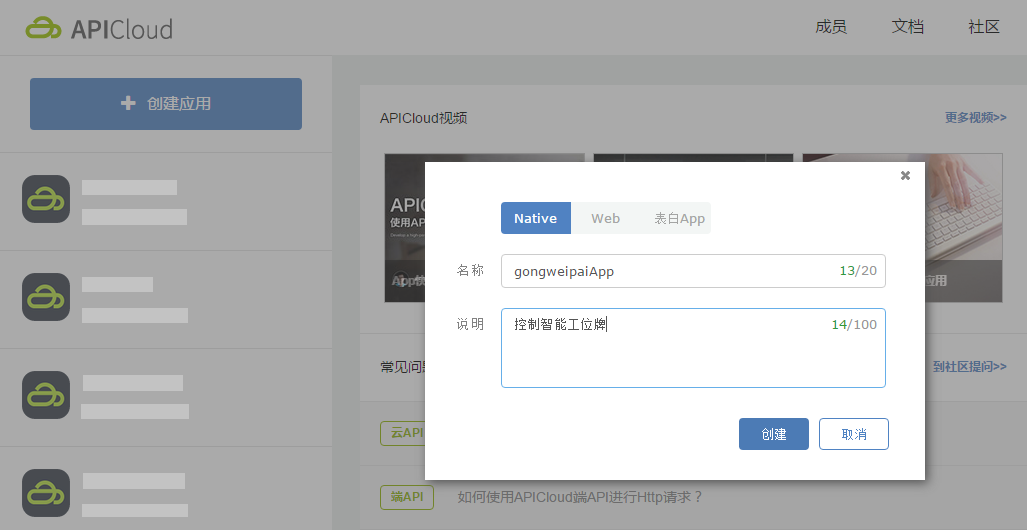
本文档会逐步引导您，快速开发一个简单的APP应用，控制MiCOKit开发套件（MiCOKit-3288或MiCOKit-3165）智能工位牌。

# 准备工作

下载(”http://www.apicloud.com/dev”)并安装APICloud IDE开发环境，IDE当前支持Windows系统。

## 注册并登录

APICloud系统：http://www.apicloud.com/console 点击左上角“创建应用”， 如图：选择“Native”，填写“名称”及“说明”，应用创建完成。



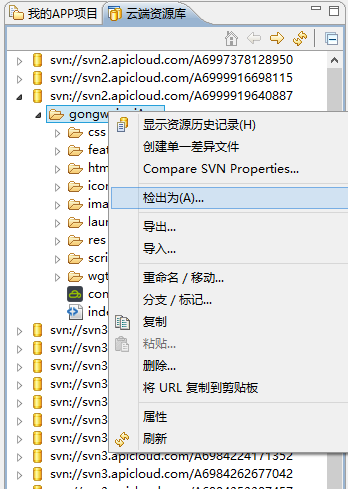
## 登录IDE

用以上注册的APICloud账号登录IDE

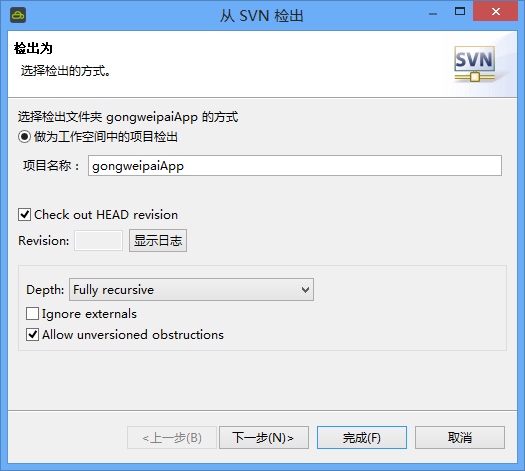


## 同步项目

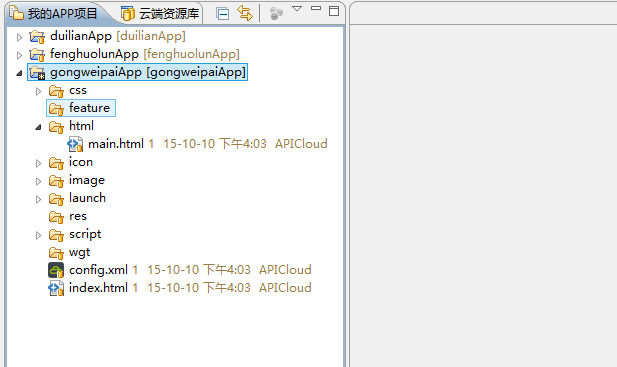
1. 登录后，左侧选择“云端资源库”，根据APICloud 创建的应用ID 选择SVN 项目。
2. 选择项目，右键“检出为”



1. 点击“完成”，应用创建完成。



## 应用包结构



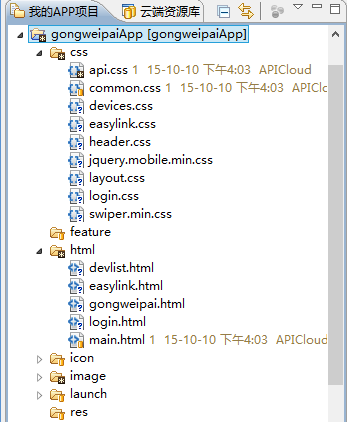
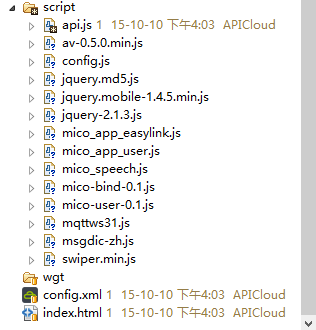
“config.xml”和 “index.html”必须包含，其它均为可选。“config.xml”是配置文件，“index.html”是启动页面，“icon”为图标文件目录，“launch”为启动图片目录（更多介绍详见Widget包结构说明文档）。

# 项目移植

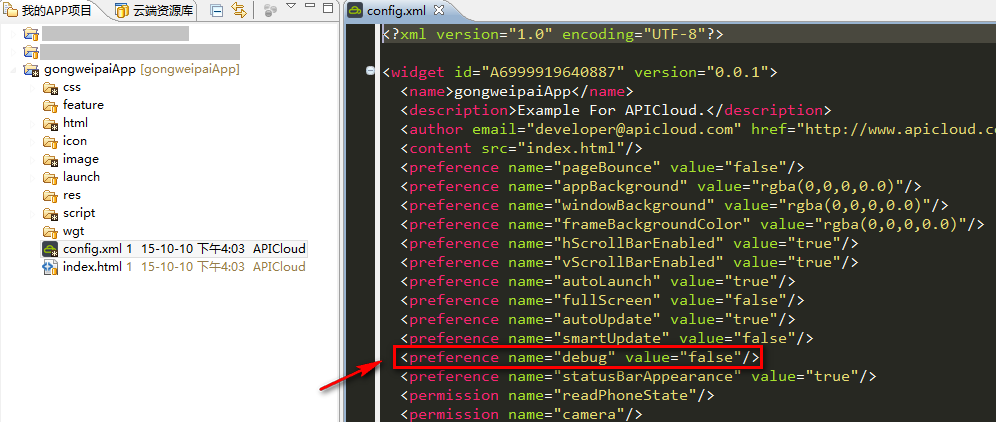
## 代码移植

下载源码“MiCO\_General\_gongweipai”， 解压后将源码中的文件拷贝到新建的gongweipaiApp工程下，**注意：不要拷贝文件夹，只单个文件地拷贝**，因为如果将svn文件也拷贝过来的话，就会导致gongweipaiApp无法正常提交或者更新。

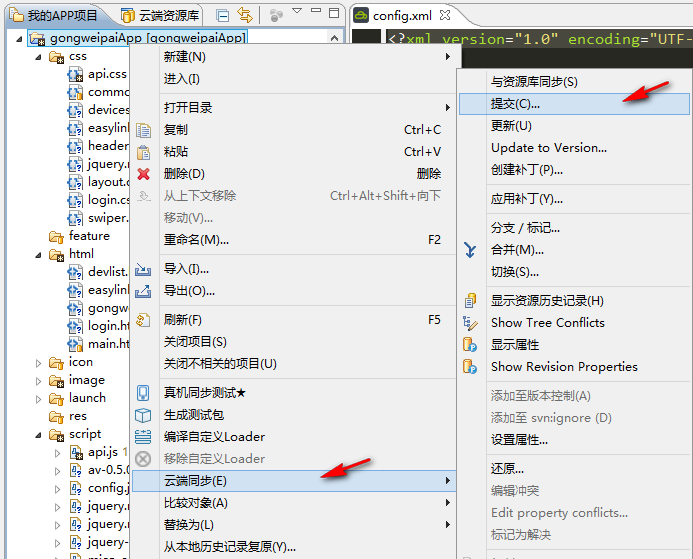
需要拷贝*script, image, html, css*文件夹里的全部文件和*index.html*文件，可以将原来的文件替换或者删除。替换后如下图

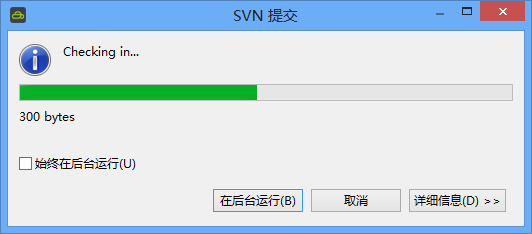
 

修改配置参数，将debug的value设置为false。



提交到云端。



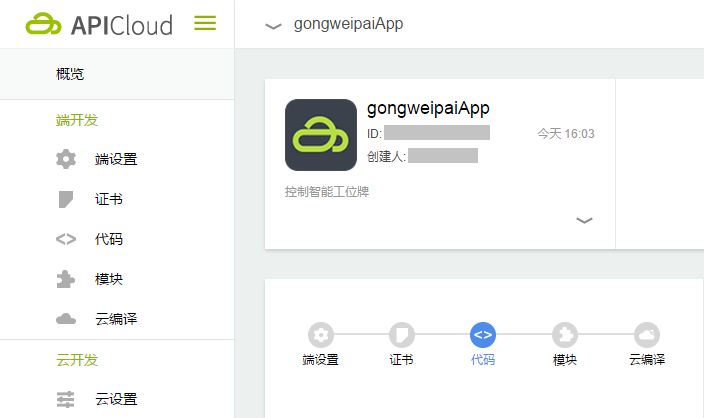


## 应用设置

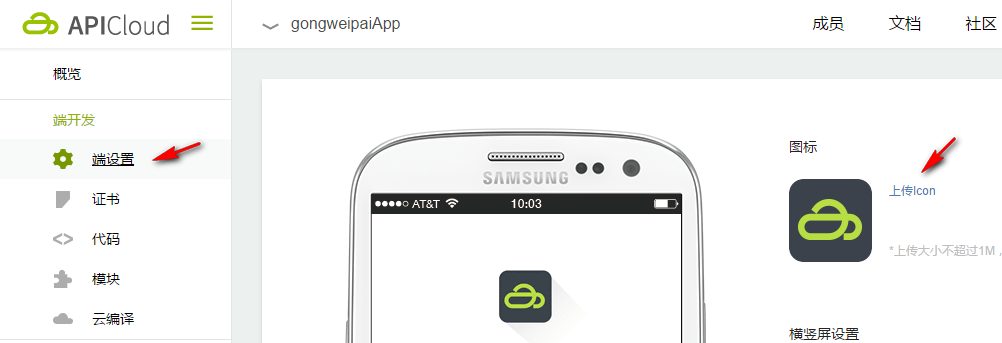
打开<http://www.apicloud.com/console>，找到之前创建的应用。



点击应用



**1）修改图标和启动页**



图标



启动页

## 添加模块

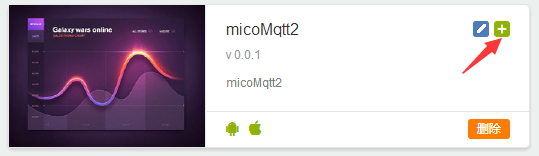
打开<https://git.oschina.net/bringmehome/MiCOSDK.git>，下载以下文件，



打开 <http://www.apicloud.com/module-custom>此页面，上传自定义模块，包含android和iOS版本的模块，一起上传，模块名和zip包的文件名一致。版本为0.0.1。



上传成功后点击加号，添加此模块

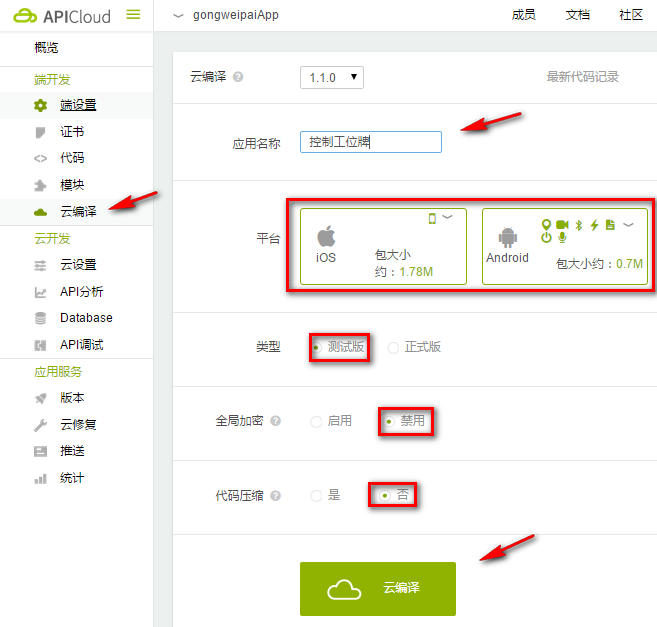


模块添加成功后界面如下，以下三个模块都需要添加：



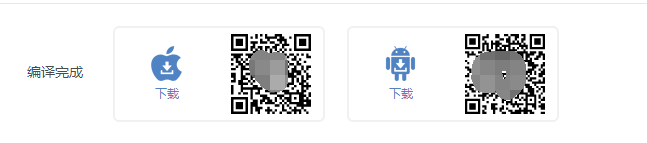
## 云编译

打开<http://www.apicloud.com/package>，参数设置，可以修改APP名称，选择APP安装的平台，可以iOS和Android一起选择。因为没有证书，所以只能选择测试版本，其他默认，点击云编译。

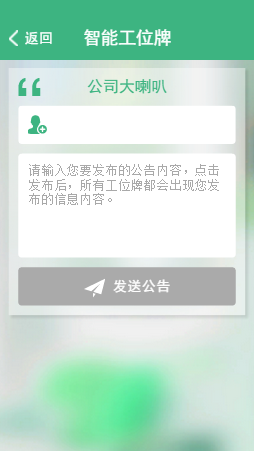


## 生成APP

等40-50秒后会生成两个二维码，如下图：



## 效果图展示

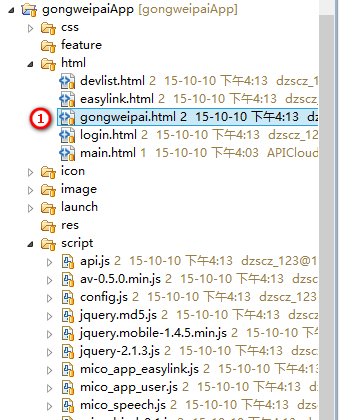


至此你完成了第一步，项目被你成功移植过来了。

# 项目解析

此项目中文件的作用：

1. css，控制页面的样式，配合html文件中的HTML代码来显示
2. html，页面的布局
3. image，页面需要显示的图片
4. script，业务逻辑



其中css和image文件不做赘述，重点介绍下HTML和script。

HTML中4个文件：

## 设备列表

1）devlist.html为设备列表界面，getdevlist()方法是获取设备列表以及将列表转成页面显示的元素，具体的接口是getDevList，返回的是json，如果需要自己设计列表界面，请参考showDev方法来解析json。

$mico.getDevList(userToken, *function*(*ret*, *err*, *devinfocb*))

这里的传人的userToken可以通过以下方法获取，前提是用户已经登录成功，关于用户注册部分可以使用默认的，或者参考mico.io。

*var* userToken = getUserInfo().get("userToken");

2）点击设备列表里的某一个设备后，会调用torgbctrl()跳转到新的页面（rgbctrl.html），跳转时候将deviceid和devicename一起传递过去，

pageParam : {

"deviceid" : id,

"devicename" : name

}

## 控制设备

1. 获取之前传递过来的参数：
   1. 获取deviceid：api.pageParam.deviceid;
   2. 获取devicename：api.pageParam.devicename;
2. 打开页面时候判断deviceid不为空，则自动去连接mqtt：ez\_connect(*device\_id*)
3. 连接mqtt
   1. api.require("micoMqtt2");先引用此模块，
   2. 传递host、clientID和topic三个参数用于打开mqtt。

Host固定，clientID（*需要按照此标准来定义：v1-app-[MAC] 版本号-app-手机MAC(12位)*），topic默认只监听device\_id + '/out/#'的通道。

1. 控制智能工位牌
   1. *function* publishcmd(*topic*, *command*)方法用于发送控制设备的指令。

代码示例：

// 发送按钮事件

$(".sub\_btn").click(function() {

if($(this).hasClass("active")){

var senddvidarray = sendidstring.split(" "); // 获取要发送的id数组

var senddvnamearray = sendnamestring.split(" "); 获取要发送到的name数组

for (var i = 0; i< senddvidarray.length; i++) {

if(senddvidarray[i] != "0" && senddvidarray[i] != ""){

//有的ID就发送 //dvidarray

var topic = senddvidarray[i] + '/in';

var command;

command = '{"sendmsg":"'+$("textarea").val()+'"}'; //控制设备的json指令

publishcmd(topic, command); //发送控制命令

//alert(senddvidarray[i]+ senddvnamearray[i] +command);

}else{

//为"0"的ID是没有选中的

}

}

}

});

注：只要包含api.require()的代码，基本都是引用的native的模块。

如果您完成到此步骤，那么恭喜您通关啦！

最简单的物联网设备已经被您开发出来了！

没有完成也不要灰心，仔细参照本文检查之前的步骤，如果还有问题，请移步至MiCO社区<http://mico.io>

Good Luck！

# 版本更新说明

|  |  |  |
| --- | --- | --- |
| 日期 | 版本 | 更新内容 |
|  |  |  |
|  |  |  |