53iq WiFi 屏开发包

DC 版

目录

适用范围 5
框架结构 5
API 接口函数 6
系统变量 6
uart_free_protocol 6
version_code 6
version_name 6
page_mapping 6
page_name_mapping 6
configuration 6
device_type 7
product_key 7
device_function 7
protocol_config 7
工具方法类 7
eval(equation, variables) 7
StripFileName(path) 7
StripPath(path) 8
GetFileName(name) 8
GetExtension(name) 8
os.exists(path) 8
os.rmdir(path) 8
CreateDirectory(path) 8
ReadFile(path) 8
WriteFile(path, content, append) 8
serialize(obj) 9
deserialize(lua) 9
SaveConfig() 9
LoadConfig() 9
CRC16(data, start, len) 9
CheckSum(packet, start, ending) 9
CheckSumAll(packet) 9
BytesToHexString(arr, split) 10
FormatTime(time) 10
事件总线 10
BindEvent(category, id, fn) 10
GetEvent(category, id) 11
StartTimer(id, timeout, count, onComplete) 11
SetInterval(id, timeout, onComplete) 11
SetTimeOut(id, timeout, onComplete) 11
HttpGet(id, url, onComplete) 11

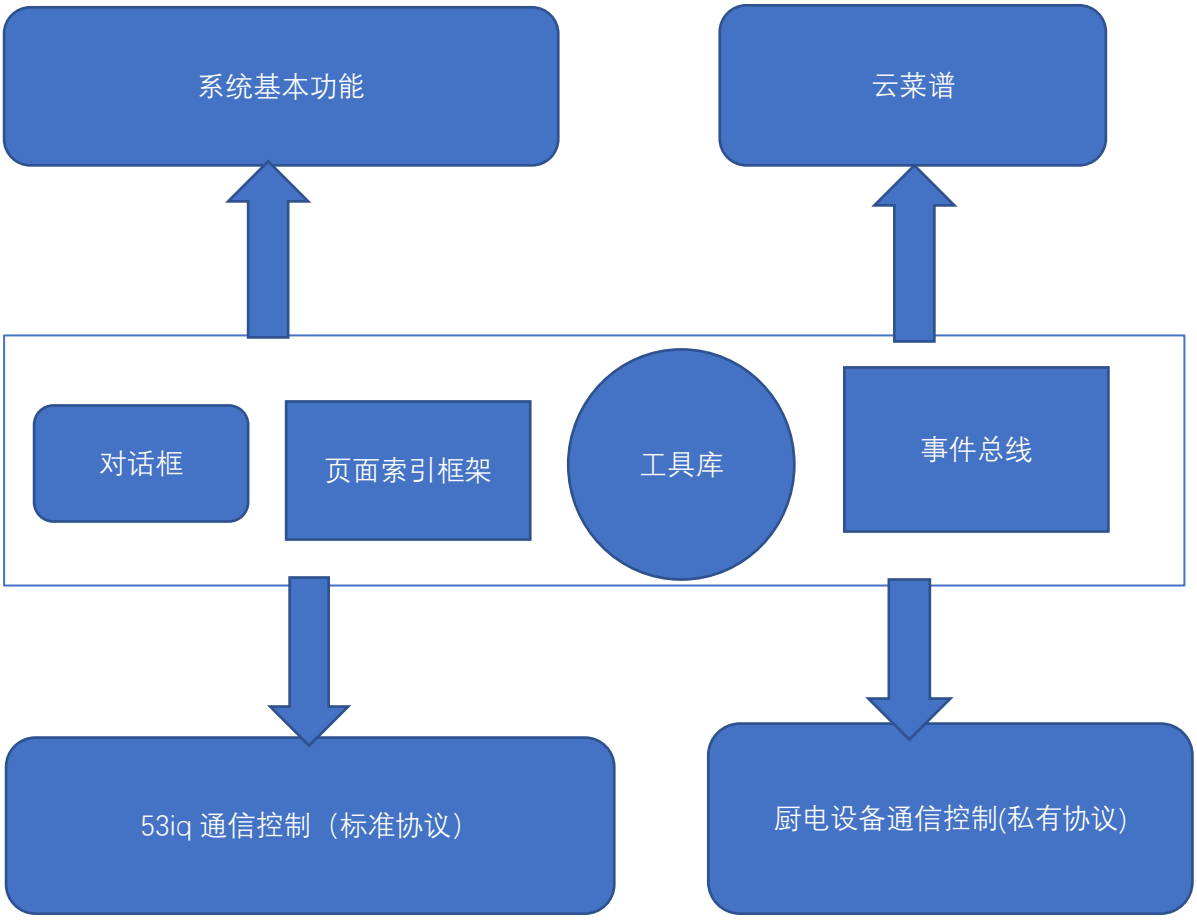
HttpPost(id, url, content, data, onComplete) 12
HttpDownload(id, url, path, onComplete) 12
页面索引 12
GetScreenID(name) 12
GetScreenName(id) 12
ChangeScreen(name) 12
XxxPageOpen(screen) 12
XxxPageProcess(screen, control, value) 13
对话框 13
ShowDialog(title, content, fn) 13
CloseDialog(screen) 13
ShowLoading(modal, fn, timeout) 13
CloseLoading() 13
基础功能 14
ShowLock() 14
CloseLock() 14
SyncServerTime() 14
SetDateTime(datetime) 14
CheckOTA() 14
StartUpgrade(url) 14
GetWiFiMac() 14
EnablePowerManage() 14
DisablePowerManage() 15
ScanAP() 15
RegisterServer() 15
GetApiToken() 15
云菜谱 15
GetMenuInfoPath(mid) 15
GetMenuImagePath(path, mid) 15
GetMenuImageUrl(url) 15
GetMenuList(page_index, page_size) 16
GetMenuDetail(id) 16
ViewMenuInfo(data, index) 16
on_draw(screen) 16
53iq 标准协议通信控制 16
AnalyzeCloudAllFunctionState(data, start) 16
AnalyzeCloudCombFunctionState(data, start, len) 16
GetFunctionData() 17
GetCloudCommand(command_type) 17
SendCloudCommand(command_type) 17
ResponseCloudCommand(serialnum, command_type) 17
ClientSendData(data) 17
ProcessNetCommand(packet) 17
UI 操作控制 17

on_function_state_change(name, value, last_value) 17
GetDeviceFunctionItem(name) 18
GetFunctionValue(name) 18
SetFunctionValues(items) 18
SetFunctionValue(name, val) 18
SendFunctionValues(items) 18
SendFunctionValue(name, val) 18
ChangeFunctionSwitch(name) 19
设备私有协议通信控制 19
ProcessUartCommand(packet) 19
AnalyzeDeviceFunctionState(data, start) 19
GetDeviceCommand(command_type) 19
SendDeviceCommand(command_type) 19
on_clound_command_processed(data) 19
on_device_state_query() 20
UartSendData(data) 20
蒸烤箱产品功能部分 20
调试方法 20
界面输出调试 20
串口日志调试 20
开发实例：标准油烟机开发 20

适用范围

本开发包用于 WiFi 屏对接 53iq 厨电开发平台，快速实现厨电的智能化，包括各种工具库、页面索引框架、扩展对话框功能、事件总线框架、系统基本功能库（OTA、持久化设置、时间同步等）、云菜谱、53iq 厨电平台通信控制框架和厨电设备通信控制框架

框架结构



API 接口函数

系统变量

uart_free_protocol

功能说明：指定系统是否使用自由
值说明：1 为使用自由协议，0 为使用 WiFi 屏内部协议
使用说明：对接 53iq 厨电平台需要使用自由协议来实现，因此该项需要设置为 1

version_code

功能说明：版本号值
使用说明: 该值为 OTA 升级时的检查判断依据, 建议以“日期+发布次数”的格式来定义, 如: 2018070101，即 2018 年 7 月 1 日第 1 次发布的版本号

version_name

功能说明：版本号名称
使用说明：用于界面上显示的可读性较为友好的版本号，根据项目实际情况自行定义

page_mapping

功能说明：页面索引配置对象
使用说明：根据工程中的页面名称顺序，依次顺序填入该对象，索引从 0 开始：page_mapping={[0]=…}

page_name_mapping

功能说明：页面索引配置对象一一对应的页面标题索引对象，主要用于标题栏的自动生成，以及智能帮助功能上报上下文环境使用

configuration

功能说明：系统配置对象，该对象中有一系列关于系统配置的值，且该对象记录的配置值会持久化到磁盘中
配置值说明：
backlight：背光亮度

remote_control_enable：是否允许 APP 远程控制
power_manage_enable：是否启用自动休眠
auto_sleep_delay：自动休眠时间（毫秒）
sync_server_time：是否自动同步网络时间
unlock_timeout：锁屏长按解锁时间（毫秒）
loading_timeout：默认等待对话框超时时间（毫秒）

device_type

功能说明：云端定义的基本产品类型 ID，主要用于云菜谱的分类获取，请根据当前设备类型填写

product_key

功能说明：当前设备在厨电开发平台注册创建时分配的 key，请根据实际值填写

device_function

功能说明：表示当前设备在厨电开发平台定义的功能列表，标准协议的解析依据，填写的描述值必须和定义的保持一致

protocol_config

功能说明：私有协议配置对象，免去私有协议对接开发（功能开发中…）

工具方法类

eval(equation, variables)

功能说明：执行指定的 lua 脚本字符串
参数说明：
equation: lua 脚本字符串
variables: 附加参数，由于 lua5.1 之后废弃了 setfenv 方法，因此该参数不再使用，请将需要传入的方法参数拼接在 equation 脚本字符串中

StripFileName(path)

功能说明：去掉完整路径中的文件名，即获取文件路径所在的目录路径
参数说明：path 为完整的文件路径

StripPath(path)

功能说明：去掉完整路径中的目录路径部分，即获取有效的文件名
参数说明：path 为完整的文件路径

GetFileName(name)

功能说明：获取不含扩展名的文件名
参数说明：完整的文件名称

GetExtension(name)

功能说明：获取文件扩展名
参数说明：完整的文件名称

os.exists(path)

功能说明：判断指定目录或文件路径是否存在
参数说明：目录路径或文件路径

os.rmdir(path)

功能说明：删除指定目录或文件路径
参数说明：目录路径或文件路径

CreateDirectory(path)

功能说明：创建指定目录路径或文件路径所在的目录路径
参数说明：目录路径或文件路径

ReadFile(path)

功能说明：读取指定路径的文件内容
参数说明：文件路径
返回说明：文件内容字符串

WriteFile(path, content, append)

功能说明：将指定内容以追加或覆盖的方式写入指定路径的文件中，若文件不存在则创建

参数说明：

path：文件路径

content：写入内容字符串

append：是否追加

serialize(obj)

功能说明：序列化对象
参数说明：table 对象
返回说明：序列化后的字符串

deserialize(lua)

功能说明：反序列化对象
参数说明：serialize 方法序列化后返回的字符串
返回说明：序列化字符串表示的 table 对象

SaveConfig()

功能说明：持久化保存系统的 configuration 配置对象

LoadConfig()

功能说明：读取保存的 configuration 配置对象

CRC16(data, start, len)

使用说明：CRC16 校验算法
参数说明：

CheckSum(packet, start, ending)

使用说明：加和校验算法
参数说明

CheckSumAll(packet)

使用说明：全帧加和校验算法

BytesToHexString(arr, split)

使用说明：数组转 16 进制字符串，多用于通信数据的日志打印

FormatTime(time)

使用说明：格式化时间，格式为：“xx h: xx m: xx s”

事件总线

系统原本程序的结构多为设定方法 ID，并在对应的触发事件方法中取自行处理的方式，该方式下代码的跳跃性比较大，且判断逻辑分支会比较多，触发事件方法会显得比较臃肿。为改善该情况，设计了事件总线的机制，将回调处理方法和调用方法直接定义在一起，改善代码的可读性，如延迟执行某个方法，使用事件总线机制改善前和改善后的示例效果如下：
改善前：

```
...
start_timer(1, 1000, 0, 0)
...
function on_timer(timer)
    if timer ==1 then
        do()
    elseif timer==2 then
        ...
    end
end
```

改善后：

```
SetTimeOut(1, 1000, function()
    do()
end)
function on_timer(timer)
    local timer_complete=GetEvent("on_timer", timer)
    if timer_complete~=nil then
        timer_complete()
    end
end
```

BindEvent(category, id, fn)

使用方法：该方法为事件总线机制的基础方法，用于将某个处理方法绑定到指定类别下的指定事件序号上
参数说明：

category: 事件类别
id: 事件序号
fn: 回调方法

GetEvent(category, id)

使用方法：该方法为事件总线机制的基础方法，用于获取指定类别下的指定事件序号的处理方法，目前框架已定义了"on_timer","on_http_request","on_http_download"这几种事件类别，自定义事件类别时应避免重复，自定义事件类别后，还需要在统一的事件触发入口加入如下类似代码进行事件的统一回调处理：

```
function on_xxx(sn, other)    --系统统一的回调入口
    local callback=GetEvent("my event", sn)
    if callback ~=nil then
        callback (other)    --自定义事件的统一回调入口
    end
end
```

参数说明：
category：事件类别
id: 事件序号

StartTimer(id, timeout, count, onComplete)

使用说明：启动定时器，可指定定时类型
参数说明：
id：事件序号
timeout：超时时间
count：是否倒计时
onComplete：触发回调方法

SetInterval(id, timeout, onComplete)

使用说明：启动重复定时执行器

SetTimeOut(id, timeout, onComplete)

使用说明：启动一次性定时器

HttpGet(id, url, onComplete)

使用说明：Get 方式请求接口方法

HttpPost(id, url, content, data, onComplete)

使用说明：Post 方式请求接口方法

HttpDownload(id, url, path, onComplete)

使用说明：Http 下载文件

页面索引

机制说明：由于目前 UI 交互机制必须明确指定“页面 ID“，”控件 ID“，且 UI 在编辑时经常会设计页面顺序的调整，导致部分脚本失效甚至错乱，因此引入页面索引机制，在页面顺序调整后，只需要在 page_mapping 和 page_name_mapping 对象中对应的调整顺序即可保证其他脚本不失效或错乱

GetScreenID(name)

功能说明：获取指定名称的页面 ID
使用说明：在涉及调用控件的交互方法时，如 set_value、set_text、get_vale、get_text 等，用该方法来代替直接传入的页面 ID
参数说明：页面名称

GetScreenName(id)

功能说明：获取指定页面 ID 所对应的页面名称，多用于 on_screen_change 或 on_control_notify 回调方法中的判断，用名称的判断来代替 ID 的判断，避免页面顺序调整后，部分代码失效或错乱

ChangeScreen(name)

功能说明：切换到指定名称的页面，说明同上

XxxPageOpen(screen)

使用说明：页面打开时的处理方法，Xxx 为页面名称
参数说明：当前页面的 ID

XxxPageProcess(screen, control, value)

使用说明：页面控件事件处理方法，Xxx 为页面名称
参数说明：
screen：页面 ID
control：控件 ID
value：控件当前值

对话框

ShowDialog(title, content, fn)

使用说明：打开对话框，由于系统没有内置的对话框机制，因此特意封装了对话框功能
参数说明：
title：标题
content：说明
fn：回调方法

CloseDialog(screen)

使用说明：关闭对话框方法
参数说明：关闭后需要返回的页面，默认为打开对话框之前的页面

ShowLoading(modal, fn, timeout)

使用方法：等待对话框
参数说明：
modal：是否模态，否的话点击对话框四周将自动关闭对话框
fn：回调方法
timeout：超时时间

CloseLoading()

使用方法：关闭等待对话框

基础功能

ShowLock()

使用说明：打开锁屏界面

CloseLock()

使用说明：关闭锁屏界面

SyncServerTime()

使用说明：同步网络时间

SetDateTime(datetime)

使用说明：设置系统时间
参数说明：要设置的时间值，格式 yyyy/MM/dd HH:mm:ss

CheckOTA()

使用说明：检查 OTA 更新

StartUpgrade(url)

使用说明：开始更新指定地址的程序包

GetWiFiMac()

使用说明：获取本地 WiFi 的 MAC 地址，由于系统原因需要做延迟获取处理，特此封装

EnablePowerManage()

使用说明：开启自动休眠功能

DisablePowerManage()

使用说明：关闭自动休眠功能

ScanAP()

使用说明：扫码热点

RegisterServer()

使用说明：注册到服务器，设置 socket 连接信息

GetApiToken()

使用说明：获取有效 Token

云菜谱

GetMenuInfoPath(mid)

使用说明：获取当前菜谱的缓存路径
参数说明：云菜谱 ID

GetMenuImagePath(path, mid)

使用说明：获取菜谱图片的存储路径
参数说明：

path：云菜谱路径
mid：菜谱 ID

GetMenuImageUrl(url)

使用说明：获取云菜谱图片的远程路径，该路径访问的是动态图片接口，根据当前设备屏幕的分辨率设定合适的图片尺寸
参数说明：云菜谱接口返回的图片 url 地址

GetMenuList(page_index, page_size)

使用说明：获取指定分页的云菜谱列表
参数说明：
page_index：页码，从 1 开始
page_size：每页数量

GetMenuDetail(id)

使用说明：获取指定菜谱的详细信息
参数说明：菜谱 ID

ViewMenuInfo(data, index)

使用说明：分步查看指定菜谱的详细信息
参数说明：
data：菜谱内容
index: 分步数

on_draw(screen)

53iq 标准协议通信控制

该部分为对接 53iq 厨电开发平台标准协议实现部分，通常不需要修改

AnalyzeCloudAllFunctionState(data, start)

使用说明：解析云端全指令的功能状态，方法会将 data 中从指定的 start 位置开始，根据 device_function 中定义的设备功能，依次取解析赋值到 device_function 中，同时会触发 on_function_state_change 方法，请重写该方法来实现 UI 的联动处理
参数说明：
data: 协议帧数据
start: 功能状态数据在 data 中的开始序号

AnalyzeCloudCombFunctionState(data, start, len)

使用说明：解析云端组合指令的功能状态，说明同上
参数说明：
data: 协议帧数据

start：功能状态数据在 data 中的开始序号
len：功能状态数据长度

GetFunctionData()

使用说明：该方法会将当前 device_function 中记录的状态值按照标准协议，生成全功能状态的数据，然后可用于具体协议帧的组装

GetCloudCommand(command_type)

使用说明：该方法用于生成指定类型的标准协议指令数据
参数说明：指令类型

SendCloudCommand(command_type)

使用说明：直接向云端发送指定指令类型的指令数据
参数说明：指令类型

ResponseCloudCommand(serialnum, command_type)

使用说明：应答指定流水号、指令类型的请求帧
参数说明：
serialnum：流水号
command_type：指令类型

ClientSendData(data)

使用说明：向云端发送数据，增加了联网状态的判断和日志的打印的统一云端发送方法

ProcessNetCommand(packet)

使用说明：处理云端指令的入口
参数说明：系统收到的从云端发送的数据

UI 操作控制

on_function_state_change(name, value, last_value)

使用说明：云端指令和电控上报数据会更新虚拟设备状态（device_function），每个状态值变

化时会触发该方法，因此 UI 需要同步设备状态应在此方法内实现
参数说明：
name：状态名称
vaue：状态值
last_value：变化前的状态值

GetDeviceFunctionItem(name)

使用说明：获取当前设备指定状态项
参数说明：状态名称
返回说明：device_function 中的对应名称的表数据

GetFunctionValue(name)

使用说明：获取当前设备指定状态值
参数说明：状态名称
返回说明：状态值

SetFunctionValues(items)

使用说明：将一组状态项设置到虚拟设备状态中，界面操作需要更新状态时调用
参数说明：一组{{name1: value1},…}形式的状态项

SetFunctionValue(name, val)

使用说明：更新指定名称的设备状态为指定的值，界面操作需要更新状态时调用
参数说明：
name：状态名称
val：状态值

SendFunctionValues(items)

使用说明：更新一组状态项并生成发送对应的控制指令，界面操作需要更新状态并发送指令时调用
参数说明：一组{{name1: value1},…}形式的状态项

SendFunctionValue(name, val)

使用说明：更新指定名称的设备状态为指定的值，界面操作需要更新状态时调用
参数说明：

name: 状态名称
val: 状态值

ChangeFunctionSwitch(name)

使用说明：反转某个开关状态的值并发送指令，多用于界面上的开关按钮点击处理
参数说明：状态名称

设备私有协议通信控制

ProcessUartCommand(packet)

使用说明：设备通信控制处理统一入口，负责解析帧格式是否有效，然后将有效的数据帧传给解析方法解析状态
参数说明：串口上报的数据

AnalyzeDeviceFunctionState(data, start)

使用说明：设备端协议设备状态解析方法
参数说明：
data: 串口数据
start: 状态数据起始地址
返回说明：本地数据解析出来的状态相比之前的设备状态是否有任何功能状态的变化

GetDeviceCommand(command_type)

使用说明：生成设备端协议控制指令的方法
参数说明：指令类型

SendDeviceCommand(command_type)

使用说明：生成并发送设备端协议指令
参数说明：指令类型

on_clound_command_processed(data)

使用说明：当云端指令导致设备状态变更时触发，需要在此处发送电控协议指令

on_device_state_query()

使用说明：当收到云端查询指令时触发的事件方法，按需求酌情处理

UartSendData(data)

蒸烤箱产品功能部分

调试方法

界面输出调试

可以在每个界面合适的位置放置一个或多个文本控件，保持控件 ID 不变，并使用 set_text 方法将信息输出到文本控件上

串口日志调试

将板卡上的日志输出口接线引出，使用串口调试工具以 115200 波特率接收日志信息，配合 print 方法将需要调试的日志打印出来查看分析

电控通信调试

利用串口调试工具的 Rx 脚和 Gnd 脚，可以截获正在运行的板卡上的收发数据，再结合协议文档分析控制数据是否正确

开发实例：标准油烟机开发

1. 填写云端创建定义的设备功能表: device_function
2. 电控协议解析和指令生成
3. UI 编辑或整合美工编辑的 UI
4. 编写 UI 联动逻辑