**SDK C使用说明**

|  |  |  |  |
| --- | --- | --- | --- |
| **版本号** | **修订日期** | **修订内容** | **修订人** |
| V1.0 | 2016.02.22 | 新增 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 主要流程



## Step1 打开设备（SDK）

int regina\_device\_open(const char \*conf,

size\_t size,

const char \*register\_code,

const char \*profile,

size\_t prof\_len,

const regina\_module\_info\_t \*modules,

size\_t module\_count,

const regina\_uplink\_data\_t \*data,

regina\_uplink\_data\_ack\_t \*ack)

**参数**：

* conf SDK配置信息，调用conf.h中的regina\_conf\_get()获取；
* size SDK配置信息长度， 定义在conf.h中的REGINA\_CONF\_SIZE；
* register\_code 注册码，在portal上进行设置，并将该值传入sdk；
* profile 设备档案信息，设备第一次上电后，该参数将被忽略
* prof\_len 设备档案信息字节数
* modules 包含所有模块的版本信息数组
* module\_count 模块数量
* data所有要上传到服务器的数据，可为NULL；
* ack所有数据的服务器确认信息，如果data中没有某种类型的数据，

其在ack中对应的确认信息，默认为 REGINA\_ACK\_SUCCESS，可为NULL。

**返回值**

打开设备成功返回REGINA\_ERR\_NONE，否则返回相应错误码。

## Step2 检查连接并保活

设备打开后，将与服务器建立TCP长连接，因此需要定期发送心跳，检查连接状态。

int regina\_device\_check\_and\_keepalive()

该函数会自动维护当前的连接状态，如果检测到连接断开、或超时将以错误码的形式返回，如果连接正常，则返回REGINA\_ERR\_NONE。

## Step3上传数据

接口为：

int regina\_device\_send(const regina\_uplink\_data\_t \*data,

regina\_uplink\_data\_ack\_t \*ack);

**参数**

* data 将要上传的数据，如果不含任何有效数据，sdk不会发送任何数据到服务器；
* ack 所有数据的服务器确认信息。

该接口发送所有的数据到服务器，如果ack不为NULL，则等到接收到所有的服务器确认信息后返回，否则，数据发送完毕后返回。

**返回值**

数据成功送达服务器，返回REGINA\_ERR\_NONE，否则返回相应错误码。

## Step4处理服务器下发数据

服务器下发的数据分为通知、配置和软件更新三类，SDK以回调的形式通知用户：

int regina\_device\_handle\_downlink\_data(void \*context,

notification\_callback nt\_callback,

configuration\_callback cf\_callback,

software\_update\_callback su\_callback);

**返回值**

如果处理服务器下发数据未发生错误，返回REGINA\_ERR\_NONE，否则返回相应错误码。

## Step5 关闭设备

当不再使用RGMP SDK时，可调用如下接口，关闭设备（SDK）。

void regina\_device\_close();

# 实例

## 主动升级

以每天主动获取一次软件更新信息，并处理软件更新信息为例，具体代码如下：

* sample.h

|  |
| --- |
| #include <regina/device.h>  void sample\_handle\_software\_update(void \*context,  const regina\_module\_update\_info\_t \*modules,  int count); |

* sample.c

|  |
| --- |
| #include “sample.h”  #include “conf.h”  void sample\_handle\_software\_update(void \*context,  const regina\_module\_update\_info\_t \*modules,  int count)  {  /\* 下载软件，并更新 \*/  }  int main(int argc, char \*\*argv)  {  int err;  regina\_uplink\_data\_t data;  regina\_uplink\_data\_ack\_t ack;  time\_t start\_time;  char \*profile;  size\_t prof\_len;  regina\_module\_info\_t modules[] = {  {  “module\_name”,  “1.0.0”  }  };  ……  /\* 填写“设备档案”等数据 \*/  ……  err = regina\_device\_open(regina\_conf\_get(), REGINA\_CONF\_SIZE,  “注册码”,  profile, prof\_len,  modules, sizeof(modules) / sizeof(modules[0]),  &data, &ack);  /\* 错误处理 \*/  ……  start\_time = time(NULL);  while (1) {  err = regina\_device\_check\_and\_keepalive();  /\* 错误处理 \*/  …  sleep(60);    if (time(NULL) >= start\_time + 60 \*24) {  data.check\_software\_update = 1;  start\_time = time(NULL);  }    err = regina\_device\_send(&data, &ack);  /\* 错误处理 \*/  err = regina\_device\_handle\_downlink\_data(NULL, NULL, NULL,  sample\_handle\_software\_update);  /\* 清空data中保存的数据 \*/  ……  }  regina\_device\_close();  return 0;  } |

# 附录 A 最低配置

* RAM：10KB
* ROM: 256KB