ECx00U&EGx00U 系列

QuecOpen HTTP API

参考手册

LTE Standard 模块系列

版本：1.0

日期：2021-09-18

状态：受控文件

上海移远通信技术股份有限公司（以下简称“移远通信”）始终以为客户提供最及时、最全面的服务为宗旨。如需任何帮助，请随时联系我司上海总部，联系方式如下：

上海移远通信技术股份有限公司
上海市闵行区田林路 1016 号科技绿洲 3 期（B 区）5 号楼　　邮编：200233
电话：+86 21 5108 6236　　邮箱：info@quectel.com

或联系我司当地办事处，详情请登录：http://www.quectel.com/cn/support/sales.htm。

如需技术支持或反馈我司技术文档中的问题，请随时登陆网址：
http://www.quectel.com/cn/support/technical.htm 或发送邮件至：support@quectel.com。

前言

移远通信提供该文档内容以支持客户的产品设计。客户须按照文档中提供的规范、参数来设计产品。同时，您理解并同意，移远通信提供的参考设计仅作为示例。您同意在设计您目标产品时使用您独立的分析、评估和判断。在使用本文档所指导的任何硬软件或服务之前，请仔细阅读本声明。您在此承认并同意，尽管移远通信采取了商业范围内的合理努力来提供尽可能好的体验，但本文档和其所涉及服务是在“可用”基础上提供给您的。移远通信可在未事先通知的情况下，自行决定随时增加、修改或重述本文档。

使用和披露限制

许可协议

除非移远通信特别授权，否则我司所提供硬软件、材料和文档的接收方须对接收的内容保密，不得将其用于除本项目的实施与开展以外的任何其他目的。

版权声明

移远通信产品和本协议项下的第三方产品可能包含受移远通信或第三方材料、硬软件和文档版权保护的相关资料。除非事先得到书面同意，否则您不得获取、使用、向第三方披露我司所提供的文档和信息，或对此类受版权保护的资料进行复制、转载、抄袭、出版、展示、翻译、分发、合并、修改，或创造其衍生作品。移远通信或第三方对受版权保护的资料拥有专有权，不授予或转让任何专利、版权、商标或服务商标权的许可。为避免歧义，除了正常的非独家、免版税的产品使用许可，任何形式的购买都不可被视为授予许可。对于任何违反保密义务、未经授权使用或以其他非法形式恶意使用所述文档和信息的违法侵权行为，移远通信有权追究法律责任。

商标

除另行规定，本文档中的任何内容均不授予在广告、宣传或其他方面使用移远通信或第三方的任何商标、商号及名称，或其缩略语，或其仿冒品的权利。

第三方权利

您理解本文档可能涉及一个或多个属于第三方的硬软件和文档（“第三方材料”）。您对此类第三方材料的使用应受本文档的所有限制和义务约束。

移远通信针对第三方材料不做任何明示或暗示的保证或陈述，包括但不限于任何暗示或法定的适销性或特定用途的适用性、平静受益权、系统集成、信息准确性以及与许可技术或被许可人使用许可技术相关的不侵犯任何第三方知识产权的保证。本协议中的任何内容都不构成移远通信对任何移远通信产品或任何其他硬软件、设备、工具、信息或产品的开发、增强、修改、分销、营销、销售、提供销售或以其他方式维持生产的陈述或保证。此外，移远通信免除因交易过程、使用或贸易而产生的任何和所有保证。

免责声明

1) 移远通信不承担任何因未能遵守有关操作或设计规范而造成损害的责任。

2) 移远通信不承担因本文档中的任何因不准确、遗漏、或使用本文档中的信息而产生的任何责任。

3) 移远通信尽力确保开发中功能的完整性、准确性、及时性，但不排除上述功能错误或遗漏的可能。除非另有协议规定，否则移远通信对开发中功能的使用不做任何暗示或法定的保证。在适用法律允许的最大范围内，移远通信不对任何因使用开发中功能而遭受的损害承担责任，无论此类损害是否可以预见。

4) 移远通信对第三方网站及第三方资源的信息、内容、广告、商业报价、产品、服务和材料的可访问性、安全性、准确性、可用性、合法性和完整性不承担任何法律责任。

文档历史

修订记录

版本	日期	作者	变更表述
-	2020-11-11	Herry GENG	文档创建
1.0	2021-09-18	Herry GENG	受控版本

目录

表格索引

1 引言

移远通信 LTE Standard ECx00U 系列和 EGx00U 模块支持 QuecOpen®方案；QuecOpen®是基于 RTOS 的嵌入式开发平台，可简化 IoT 应用的软件设计和开发过程。有关 QuecOpen®的详细信息，请参考 ***文档 [1]***。

本文档主要介绍在 QuecOpen®方案下，ECx00U 系列和 EGx00U 模块 HTTP 相关 API、调用流程以及示例。

1.1. 适用模块

表 1：适用模块

模块系列	模块
ECx00U	EC200U 系列
	EC600U 系列
EGx00U	EG500U-CN
	EG700U-CN

2 HTTP 相关接口调用流程

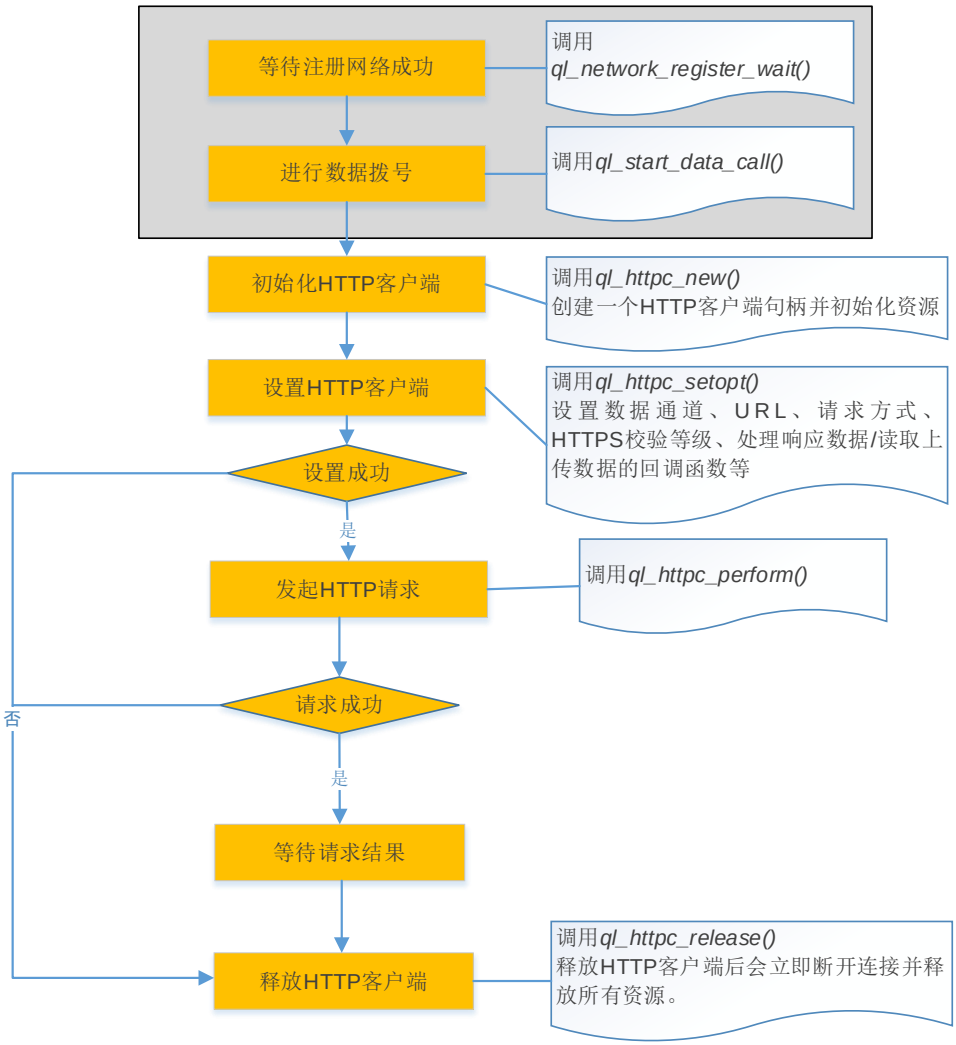

图 1：HTTP 接口调用流程

HTTP 服务基于指定的数据通道进行数据收发，因此在发送 HTTP 请求之前，首先需等待网络注册成功并进行数据拨号以建立数据通道（如上图中灰底部分）。若在其他任务中已经建立了数据通道，可跳过注网拨号步骤，直接调用 *ql_httpc_new()*创建 HTTP 客户端句柄，并调用 *ql_httpc_setopt()*配置已建立的数据通道（数据通道默认为 1）、URL、HTTP 请求方式、HTTPS 校验等级以及自定义 HTTP 消息头等参数。

调用 *ql_httpc_perform()*发送 HTTP 请求时，该函数会立即返回，相关事件发生时会调用相应的回调函

数。必须等待此 HTTP 连接断开后，方可执行下一步 HTTP 相关操作。

备注

1. ECx00U 系列和 EGx00U QuecOpen 模块仅支持 HTTP/1.1，同时支持 GET/POST/PUT/HEAD 请求方式。
2. 同一个 HTTP 客户端在同一时间只能执行一次 HTTP 请求，当前请求处理完成后才能进行下一次请求。若需同时执行多次 HTTP 请求，可通过创建多个 HTTP 客户端实现。
3. 数据拨号函数 *ql_network_register_wait()*和 *ql_start_data_call()*的详细信息，请参考***文档 [2]***。

3 HTTP API

3.1. 头文件

HTTP API 的头文件为 *ql_http_client.h*，位于 SDK 包的 *components\ql-kernel\inc* 目录下。若无特别说明，本文档所述头文件均在该目录下。

3.2. 函数概览

表 2：函数概览

函数	说明
ql_httpc_new()	创建一个新的 HTTP 客户端句柄并初始化 HTTP 客户端资源。
ql_httpc_setopt()	配置 HTTP 客户端属性。
ql_httpc_formadd()	配置 HTTP 表单属性。仅在设置 *HTTP_CLIENT_OPT_METHOD* 为 *HTTP_METHOD_POST* 时有效。
ql_httpc_perform()	发送 HTTP 请求。
ql_httpc_getinfo()	获取 HTTP 消息头中指定键值。
ql_httpc_release()	释放 HTTP 客户端资源。

3.3. API 详解

3.3.1. ql_httpc_new

该函数用于创建一个新的 HTTP 客户端句柄并初始化 HTTP 客户端资源。

- **函数原型**

```
int ql_httpc_new(http_client_t *client, http_client_event_cb_t cb, void *arg)
```

- **参数**

client：
[Out] HTTP 客户端句柄。

cb：
[In] HTTP 请求事件的回调函数。详见***第 3.3.1.1 章***。

arg：
[In] HTTP 请求事件的回调参数。

- **返回值**

详见***第 3.3.1.3 章***。

3.3.1.1. http_client_event_cb_t

该函数为 HTTP 请求事件回调函数。发送 HTTP 请求后，相关事件发生时自动调用该回调函数。

- **函数原型**

```
typedef void(*http_client_event_cb_t)(http_client_t *client, int evt, int evt_code, void *arg)
```

- **参数**

client：
[In] HTTP 客户端句柄。由 *ql_httpc_new()*获取。

evt：
[In] HTTP 事件类型。详见***第 3.3.1.2 章***。

evt_code：
[In] HTTP 事件结果码。详见***第 3.3.1.3 章***。

arg：
[In] HTTP 请求事件的回调参数。由 *ql_httpc_new()*设置。

- **返回值**

无

3.3.1.2. http_event_id_e

HTTP 事件类型定义如下：

```
typedef enum{
    HTTP_EVENT_SESSION_ESTABLISH          = 0,
    HTTP_EVENT_RESPONE_STATE_LINE          = 1,
    HTTP_EVENT_SESSION_DISCONNECT          = 2,
}http_event_id_e
```

- **参数**

参数	描述
HTTP_EVENT_SESSION_ESTABLISH	HTTP 会话连接建立。
HTTP_EVENT_RESPONE_STATE_LINE	接收 HTTP 响应头。接收响应头后，可通过 *ql_httpc_getinfo()*获取响应码、内容长度等参数值，详见***第 3.3.5 章***。
HTTP_EVENT_SESSION_DISCONNECT	HTTP 会话连接断开。

3.3.1.3. http_error_code_e

HTTP 事件结果码定义如下：

```
typedef enum{
    HTTP_SUCCESS                  = 0,
    HTTP_ERR_UNKOWN               = -1,
    HTTP_ERR_WOUNDBLOCK           = -2,
    HTTP_ERR_INVALID_PARAM        = -3,
    HTTP_ERR_OUT_OF_MEM           = -4,
    HTTP_ERR_BUSY                 = -5,
    HTTP_ERR_BAD_REQUEST          = -6,
    HTTP_ERR_SOCKE_CONNECT_FAIL   = -7,
    HTTP_ERR_DNS_FAIL             = -8,
    HTTP_ERR_SOCKET_EXCEPTION     = -9,
```

```
    HTTP_ERR_TIMEOUT                    = -10,
}http_error_code_e
```

- **参数**

参数	描述
HTTP_SUCCESS	函数执行成功
HTTP_ERR_UNKOWN	未知错误
HTTP_ERR_WOUNDBLOCK	当前操作未完成
HTTP_ERR_INVALID_PARAM	无效参数
HTTP_ERR_OUT_OF_MEM	内存不足
HTTP_ERR_BUSY	系统忙碌
HTTP_ERR_BAD_REQUEST	错误的请求
HTTP_ERR_SOCKE_CONNECT_FAIL	socket 连接失败
HTTP_ERR_DNS_FAIL	域名解析失败
HTTP_ERR_SOCKET_EXCEPTION	socket 出现异常
HTTP_ERR_TIMEOUT	请求超时

3.3.2. ql_httpc_setopt

该函数用于配置 HTTP 客户端属性。

- **函数原型**

```
int ql_httpc_setopt(http_client_t *client, int opt_tag, ...)
```

- **参数**

client：
[In] HTTP 客户端句柄。由 *ql_httpc_new()*获取。

opt_tag：
[In] 属性标签。详见***第 3.3.2.1 章***。

…：
[In] 属性值。详见***第 3.3.2.1 章***。

- **返回值**

详见*第 3.3.1.3 章*。

3.3.2.1. http_option_e

HTTP 配置属性标签枚举定义如下：

```
typedef enum{
    HTTP_CLIENT_OPT_PDPCID = 1,
    HTTP_CLIENT_OPT_SSLCTXID,
    HTTP_CLIENT_OPT_BASIC_AUTH,
    HTTP_CLIENT_OPT_REQUEST_HEADER,
    HTTP_CLIENT_OPT_WRITE_HEADER,
    HTTP_CLIENT_OPT_INTERVAL_TIME,
    HTTP_CLIENT_OPT_METHOD,
    HTTP_CLIENT_OPT_WRITE_FUNC,
    HTTP_CLIENT_OPT_WRITE_DATA,
    HTTP_CLIENT_OPT_READ_FUNC,
    HTTP_CLIENT_OPT_READ_DATA,
    HTTP_CLIENT_OPT_UPLOAD_LEN,
    HTTP_CLIENT_OPT_URL,
    HTTP_CLIENT_OPT_SSL_VERIFY_LEVEL,
    HTTP_CLIENT_OPT_SSL_CACERT_PATH,
    HTTP_CLIENT_OPT_SSL_OWNCERT_PATH,
    HTTP_CLIENT_OPT_URI,
    HTTP_CLIENT_OPT_SIM_ID
}http_option_e
```

- **参数**

属性标签	描述
HTTP_CLIENT_OPT_PDPCID	设置 HTTP 客户端使用的数据通道号，即执行数据拨号操作时使用的 PDP 上下文 ID。详情请参考*文档 [2]*。 参数类型：int。
HTTP_CLIENT_OPT_SSLCTXID	设置 HTTPS 客户端使用的 SSL 上下文 ID。 参数类型：int；范围：0~5。
HTTP_CLIENT_OPT_BASIC_AUTH	设置 HTTP 基本认证的用户名和密码。 格式：Username:Password。最大长度为 255 字节。 参数类型：char *。
HTTP_CLIENT_OPT_REQUEST_HEADER	设置自定义 HTTP 请求头。可以多次配置此选

	项，设置多个自定义 HTTP 请求。 参数类型：char *。
HTTP_CLIENT_OPT_WRITE_HEADER	设置是否保存 HTTP 响应头。 参数类型：int。 *0* 不保存 *1* 保存
HTTP_CLIENT_OPT_INTERVAL_TIME	设置等待 HTTP 响应数据的最大间隔时间。 单位：秒；范围：1~65536 秒。 参数类型：int。
HTTP_CLIENT_OPT_METHOD	设置 HTTP 请求方式。 参数类型：*http_method_e*（详见***第 3.3.2.2 章***）。
HTTP_CLIENT_OPT_WRITE_FUNC	设置处理 HTTP 响应数据的回调函数。 参数类型：*http_client_write_data_cb_t*（详见***第 3.3.2.4 章***）。
HTTP_CLIENT_OPT_WRITE_DATA	设置处理 HTTP 响应数据的回调函数的参数。 参数类型：void *。
HTTP_CLIENT_OPT_READ_FUNC	设置读取 HTTP 请求上传数据的回调函数。此选项仅在设置 *HTTP_CLIENT_OPT_METHOD* 为 *HTTP_METHOD_POST/HTTP_METHOD_PUT* 后有效。 参数类型：*http_client_read_data_cb_t*（详见***第 3.3.2.5 章***）。
HTTP_CLIENT_OPT_READ_DATA	设置读取 HTTP 请求上传数据的回调函数的参数。 参数类型：void *。
HTTP_CLIENT_OPT_UPLOAD_LEN	设置 HTTP 上传的数据长度。此选项仅在设置 *HTTP_CLIENT_OPT_METHOD* 为 *HTTP_METHOD_POST/HTTP_METHOD_PUT* 后有效。 参数类型：int。
HTTP_CLIENT_OPT_URL	设置 HTTP 客户端访问的 URL。URL 必须以 https://或 http://开头。 参数类型：char *。
HTTP_CLIENT_OPT_SSL_VERIFY_LEVEL	设置 HTTPS 的校验等级。 参数类型：int。范围详见***第 3.3.2.3 章***。
HTTP_CLIENT_OPT_SSL_CACERT_PATH	设置 HTTPS 的 CA 证书的路径。 参数类型：char *。
HTTP_CLIENT_OPT_SSL_OWNCERT_PATH	设置 HTTPS 的本地证书的路径。 参数类型：char*。
HTTP_CLIENT_OPT_URI	设置 HTTP 客户端访问的 URI。 参数类型：char*。
HTTP_CLIENT_OPT_SIM_ID	设置 HTTPS 的本地证书的路径。即执行数据拨号操作时使用的(U)SIM 卡。 参数类型：int。

3.3.2.2. http_method_e

HTTP 请求方式枚举定义如下：

```
typedef enum {
    HTTP_METHOD_NONE,
    HTTP_METHOD_GET,
    HTTP_METHOD_POST,
    HTTP_METHOD_PUT,
    HTTP_METHOD_HEAD,
    HTTP_METHOD_LAST
} http_method_e
```

- **参数**

参数	描述
HTTP_METHOD_NONE	无请求。
HTTP_METHOD_GET	HTTP GET 请求。请求指定的页面信息，并返回实体主体。
HTTP_METHOD_POST	HTTP POST 请求。从客户端向服务器提交数据进行处理请求（例如提交表单或者上传文件）。
HTTP_METHOD_PUT	HTTP PUT 请求。从客户端向服务器传送的数据取代指定的文档内容。
HTTP_METHOD_HEAD	HTTP HEAD 请求。只请求页面的首部。
HTTP_METHOD_LAST	上一次请求。

3.3.2.3. https_verify_level_e

HTTPS 校验等级定义如下：

```
typedef enum{
    HTTPS_VERIFY_NONE          = 0,
    HTTPS_VERIFY_SERVER        = 1,
    /*HTTPS_VERIFY_SERVER*/
    HTTPS_VERIFY_SERVER_CLIENT = 2,
}https_verify_level_e
```

- **参数**

参数	描述
HTTPS_VERIFY_NONE	不校验。
HTTPS_VERIFY_SERVER	校验服务器。
HTTPS_VERIFY_SERVER_CLIENT	客户端与服务器双向校验。设置为该等级时，需上传本地证书；此外，若服务器不要求校验客户端，则该参数等同于 *HTTPS_VERIFY_SERVER*。

3.3.2.4. http_client_write_data_cb_t

收到 HTTP 响应数据后，会自动调用该回调函数进行数据处理。

- **函数原型**

```
typedef int(*http_client_write_data_cb_t)(http_client_t *client, void *arg, char *data, int size, unsigned char end)
```

- **参数**

client：
[In] HTTP 客户端句柄。由 *ql_httpc_new()*获取。

arg：
[In] 回调参数。通过 *ql_httpc_setopt()*的属性标签 *HTTP_CLIENT_OPT_WRITE_DATA* 进行配置，详见***第 3.3.2.1 章***。

data：
[In] 待处理数据。

size：
[In] 待处理数据的长度。单位：字节。

end：
[In] 数据包结束标识。
1 当前包为最后一包数据
0 当前包不是最后一包数据

- **返回值**

实际处理数据的长度。

3.3.2.5. http_client_read_data_cb_t

当发生 HTTP POST/PUT 请求时，会自动调用该回调函数读取待上传的数据。

- **函数原型**

```
typedef int(*http_client_read_data_cb_t)(http_client_t *client, void *arg, char *data, int size)
```

- **参数**

client：
[In] HTTP 客户端句柄。由 *ql_httpc_new()*获取。

arg：
[In] 回调参数。由 *ql_httpc_setopt()*的属性标签 *HTTP_CLIENT_OPT_READ_DATA* 进行配置，详见***第 3.3.2 章***。

data：
[In] 用于存储待上传数据的空间。

size：
[In] 用于存储待上传数据的空间大小。单位：字节。

- **返回值**

实际读取的数据长度。

3.3.3. ql_httpc_formadd

该函数用于配置 HTTP 表单属性，仅在设置 *HTTP_CLIENT_OPT_METHOD* 为 *HTTP_METHOD_POST* 时有效，详见***第 3.3.2.2 章***。

- **函数原型**

```
int    ql_httpc_formadd(http_client_t *client, int opt_tag, ...)
```

- **参数**

client：
[In] HTTP 客户端句柄。由 *ql_httpc_new()*获取。

opt_tag：
[In] 属性标签。详见***第 3.3.3.1 章***。

…:
[In] 属性值。详见*第 3.3.3.1 章*。

- **返回值**

详见*第 3.3.1.3 章*。

3.3.3.1. http_formtopt_e

HTTP 表单头部属性定义如下：

```
typedef enum{
    HTTP_FORM_NAME = 1,
    HTTP_FORM_FILENAME = 2,
    HTTP_FORM_CONTENT_TYPE = 3,
}http_formopt_e
```

- **参数**

参数	描述
HTTP_FORM_NAME	设置上传数据在服务器上的存储格式为 file 或 image 等。 参数类型：char*。
HTTP_FORM_FILENAME	设置上传数据在服务器保存的名字。仅在 *HTTP_FORM_NAME* 配置为 file 时有效。 参数类型：char*。
HTTP_FORM_CONTENT_TYPE	设置上传数据对应的内容类型。仅在 *HTTP_FORM_NAME* 配置为 file 时有效。 参数类型：char*。

3.3.4. ql_httpc_perform

该函数用于发送 HTTP 请求。

- **函数原型**

```
int   ql_httpc_perform(http_client_t *client)
```

- **参数**

client：
[In] HTTP 客户端句柄。由 *ql_httpc_new()*获取。

- **返回值**

详见*第 3.3.1.3 章*。

备注

当返回值不为 *HTTP_SUCCESS* 时，表示函数调用失败；当返回值为 *HTTP_SUCCESS* 时，因该函数为异步函数，需通过事件回调函数 *http_client_event_cb_t()*判断请求的结果是否成功。

3.3.5. ql_httpc_getinfo

该函数用于获取 HTTP 消息头中指定键值。

- **函数原型**

```
int   ql_httpc_getinfo(http_client_t   *client, int info,...)
```

- **参数**

client：
[In] HTTP 客户端句柄。由 *ql_httpc_new()*获取。

info：
[In] 属性标签。详见*第 3.3.5.1 章*。

…：
[Out] 属性值。详见*第 3.3.5.1 章*。

- **返回值**

详见*第 3.3.1.3 章*。

3.3.5.1. http_info_e

HTTP 响应头属性定义如下：

```
typedef enum{
    /*Response  code*/
    HTTP_INFO_RESPONSE_CODE = 0,
    HTTP_INFO_LOCATION,
    HTTP_INFO_CONTENT_LEN,
    HTTP_INFO_CHUNK_ENCODE,
    HTTP_INFO_ACCEPT_RANGES,
```

```
}http_info_e
```

- **参数**

参数	描述
HTTP_INFO_RESPONSE_CODE	HTTP 响应码。 参数类型：int*。
HTTP_INFO_LOCATION	获取重定向的 URL。该值仅当响应码为 3xx 时方可获取。 参数类型：char*。
HTTP_INFO_CONTENT_LEN	获取内容长度。该值仅当 HTTP 响应头中带有内容长度时方可获取。 参数类型：int *。
HTTP_INFO_CHUNK_ENCODE	获取响应体数据是否为 chunk 编码格式。 参数类型：int *。 *0* 否 *1* 是
HTTP_INFO_ACCEPT_RANGES	获取 Server 是否接收 Range 请求。 参数类型：int *。 *0* 不接收 *1* 接收

3.3.6. ql_httpc_release

该函数用于释放 HTTP 客户端资源。

- **函数原型**

```
int   ql_httpc_release(http_client_t *client)
```

- **参数**

client：
[In] HTTP 客户端句柄。由 *ql_httpc_new()*获取。

- **返回值**

详见***第 3.3.1.3 章***。

4 示例

有关 HTTP 相关功能示例，请参考 QuecOpen SDK 中应用层 HTTP 示例文件 *http_demo.c*、*http_post_demo.c*、*http_post_demo.c* 和 *https_get_demo.c*，均位于 *components\ql-application\http* 目录下。

5 附录 参考文档及术语缩写

表 3：参考文档

文档名称
[1] Quectel_ECx00U&EGx00U 系列_QuecOpen_CSDK_快速开发指导
[2] Quectel_ECx00U&EGx00U 系列_QuecOpen_数据拨号 API_参考手册

表 4：术语缩写

缩写	英文全称	中文全称
API	Application Programming Interface	应用程序编程接口
CA	Certificate Authority	证书颁发机构
DNS	Domain Name Server	域名系统（服务）协议
HTTP	Hypertext Transfer Protocol	超文本传输协议
IoT	Internet of Things	物联网
RTOS	Real-Time Operating System	实时操作系统
SDK	Software Development Kit	软件开发工具包
URI	Uniform Resource Identifier	统一资源标识符
URL	Uniform Resource Locator	统一资源定位符