XEngine消息服务文档

目录

[XEngine消息服务文档 1](#_Toc207357363)

[前言 4](#_Toc207357364)

[阅读者 4](#_Toc207357365)

[概述 4](#_Toc207357366)

[相关模块 4](#_Toc207357367)

[一 技术结构 4](#_Toc207357368)

[1.1 TCP 4](#_Toc207357369)

[1.2 HTTP 4](#_Toc207357370)

[1.3 WEBSOCKET 4](#_Toc207357371)

[1.4 MQTT 5](#_Toc207357372)

[1.5 心跳 5](#_Toc207357373)

[1.6 协议解释 5](#_Toc207357374)

[1.7 网络错误码 5](#_Toc207357375)

[二 配置环境 5](#_Toc207357376)

[2.1 WINDOWS 5](#_Toc207357377)

[2.1.1 配置环境 5](#_Toc207357378)

[2.1.2 编译运行 5](#_Toc207357379)

[2.2 LINUX 6](#_Toc207357380)

[2.3 MacOS 6](#_Toc207357381)

[三 接口协议 6](#_Toc207357382)

[3.1 TCP 6](#_Toc207357383)

[3.1.1 投递包协议 6](#_Toc207357384)

[3.1.2 获取包协议 7](#_Toc207357385)

[3.1.3 删除包协议 8](#_Toc207357386)

[3.1.4 创建主题 9](#_Toc207357387)

[3.1.5 删除主题 9](#_Toc207357388)

[3.1.6 通知消息 10](#_Toc207357389)

[3.1.7 主题绑定 11](#_Toc207357390)

[3.1.9 编号获取 11](#_Toc207357391)

[3.1.10 用户验证 12](#_Toc207357392)

[3.1.11 消息修改 13](#_Toc207357393)

[3.1.12 主题修改 14](#_Toc207357394)

[3.1.13 未读消息 15](#_Toc207357395)

[3.1.14 删除消息 16](#_Toc207357396)

[3.2 WEBSOCKET 16](#_Toc207357397)

[3.2.1 请求 17](#_Toc207357398)

[3.2.2 回复 18](#_Toc207357399)

[3.3 MQTT 19](#_Toc207357400)

[3.4 HTTP 19](#_Toc207357401)

[3.2.1 查询用户列表 19](#_Toc207357402)

[3.2.2 查询主题列表 20](#_Toc207357403)

[3.2.3 请求在线用户 21](#_Toc207357404)

[3.2.4 删除主题 21](#_Toc207357405)

[3.2.5 删除用户 22](#_Toc207357406)

[3.2.6 创建主题 22](#_Toc207357407)

[3.2.7 获取主题 23](#_Toc207357408)

[四 配置说明 23](#_Toc207357409)

[4.1 服务器配置 23](#_Toc207357410)

[4.1.1 基本配置 24](#_Toc207357411)

[4.1.2 最大配置 24](#_Toc207357412)

[4.1.3 日志配置 24](#_Toc207357413)

[4.1.4 数据库配置 24](#_Toc207357414)

[4.1.5 HTTP代理配置 24](#_Toc207357415)

[4.1.6 高速缓存配置 25](#_Toc207357416)

[4.1.7 报告配置 25](#_Toc207357417)

[4.1.8 验证配置 25](#_Toc207357418)

[4.1.9 消息通知 25](#_Toc207357419)

[4.2 数据库配置 25](#_Toc207357420)

[4.2.1 用户数据库 26](#_Toc207357421)

[4.2.2 消息数据库 26](#_Toc207357422)

[五 高级配置 26](#_Toc207357423)

[5.1 消息服务协议 26](#_Toc207357424)

[5.1.1 消息属性 26](#_Toc207357425)

[六 其他功能 27](#_Toc207357426)

[6.1 HTTP代理 27](#_Toc207357427)

[附录 27](#_Toc207357428)

[附录1 类型定义 27](#_Toc207357429)

[附录2 协议定义 27](#_Toc207357430)

[附录3 转换定义 27](#_Toc207357431)

[附录4 更新历史 27](#_Toc207357432)

|  |  |  |  |
| --- | --- | --- | --- |
| 文件状态：  [ ] 草稿  [√] 正式发布 | 文件名称： | XEngin消息服务文档 | |
| 适用版本： | V3.17.0.1001 | |
| 发布日期： | 2025-09-01 | |
| 拟 制： qyt | | |

# 前言

## 阅读者

开发人员,测试人员

## 概述

此文档包含了存储服务的相关技术说明和接口定义!

## 相关模块

此服务使用了XEngine作为开发包.要使用此服务代码必须配置安装好XEngine开发环境.

此服务还使用了第三方模块jsoncpp

# 一 技术结构

服务器启动会绑定三个端口,一个TCP消息端口,一个HTTP消息端口和WEBSOCKET端口

## 1.1 TCP

TCP协议使用了XEngine的标准协议头来实现消息通信.TCP协议速度更快,功能更多.

## 1.2 HTTP

提供HTTP接口用于服务器信息查询和管理.

## 1.3 WEBSOCKET

Websocket类型使用TEXT.负载JSON即可实现,WEBSOCKET消息支持订阅模式.

## 1.4 MQTT

MQTT协议基于TCP实现,使用MQTT5.0版本协议.负载JSON数据,JSON数据为WEBSOCKET相同,因为需要转换为我们自己的通信标准,回复也是

## 1.5 心跳

为了保持会话的持续,心跳必须按照固定时间发送.

## 1.6 协议解释

TCP私有协议采用的是XEngine定义的协议,你可以参考XEngine的协议文档详细了解协议的定义与解释.

## 1.7 网络错误码

现在网络错误码都存在公用的协议头文件中了.返回的服务器错误将在这里面定义,关于错误内容可以在这里面找到.

# 二 配置环境

## 2.1 WINDOWS

需要下载XEngine.

通过VS来编写代码,编译,调试.

### 2.1.1 配置环境

按照XEngine Readme文件的说明配置环境

### 2.1.2 编译运行

在完成环境配置后.你可以进入代码目录.使用VS打开XEngine\_MQServiceApp.sln,然后编译.

成功后,你需要拷贝XEngine\_Release下的文件到你编译的目录下.运行XEngine\_MQServiceApp.exe即可

注意：XEngine环境可以通过VSCopy脚本拷贝到你的编译目录下,前提是你配置好你的XEngine环境..

## 2.2 LINUX

支持大多数发型版本系统,你可以在github的发布页面查看,或者查看github的actions脚本.

## 2.3 MacOS

MacOS需要13以及以上版本,编译和运行方式可以参考Linux.

# 三 接口协议

## 3.1 TCP

请注意:协议头的wReserver字段将作为服务器回复处理结果使用.为0表示处理成功,其他值表示失败!  
 协议负载无论是通用协议还是HTTP协议,他们负载的内容都可以负载JSON.具体参考协议说明.

### 3.1.1 投递包协议

要使用消息队列,你需要先投递一个包给消息队列服务器,这样其他程序才能从消息队列服务中取得一个包.

wReserve:如果是定时发布,协议头的此字段可以表示自身是否接受

#### 3.1.1.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQPOST

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ) + MSGLEN

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

末尾跟上DATA数据

nPubTime表示定时消息,只有订阅了这个消息并且在线的用户才会在这个时间段收到

byVersion 表示负载的内容格式,具体可参考ENUM\_XENGINE\_PROTOCOLHDR\_PAYLOAD\_TYPE

#### 3.1.1.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPPOST

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

### 3.1.2 获取包协议

#### 3.1.2.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQGET

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

#### 3.1.2.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPGET

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ) + MSGLEN

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

协议体:如果成功,协议后面跟上获取到的数据

byVersion返回的内容表示负载的类型,参考:ENUM\_XENGINE\_PROTOCOLHDR\_PAYLOAD\_TYPE

### 3.1.3 删除包协议

#### 3.1.3.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQDEL

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

#### 3.1.3.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPDEL

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

### 3.1.4 创建主题

要投递一个包到消息队列,你必须先创建一个主题,必须指明主题,数据才能进入消息队列.

#### 3.1.4.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQTOPICCREATE

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

#### 3.1.4.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPTOPICCREATE

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

### 3.1.5 删除主题

#### 3.1.5.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQTOPICDELETE

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

#### 3.1.5.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPTOPICDELETE

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

### 3.1.6 通知消息

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_MSGNOTIFY

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ) + message

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

...body

### 3.1.7 主题绑定

绑定后可以接受通知和不设置序列号取消息,支持跳转到指定序列号开始获取

#### 3.1.7.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQTOPICBIND

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

#### 3.1.7.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPTOPICBIND

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

### 3.1.9 编号获取

可以获取指定主题总消息个数,起始消息编号,结束消息编号

#### 3.1.9.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQNUMBER

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_XMQ

#### 3.1.9.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPNUMBER

unPacketSize = sizeof(XENGINE\_MQNUMBER)

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:sizeof(XENGINE\_MQNUMBER

### 3.1.10 用户验证

#### 3.1.10.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQUSERLOG

unPacketSize = sizeof(XENGINE\_PROTOCOL\_USERAUTH)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XENGINE\_PROTOCOL\_USERAUTH

#### 3.1.10.2 回复

协议头:wReserve 0表示成功,其他值表示失败

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_AUTH

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPUSERLOG

unPacketSize = 0

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

### 3.1.11 消息修改

#### 3.1.11.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = *ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ*

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQMODIFY

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:根据消息Key和序列号来确定修改的消息

XENGINE\_PROTOCOL\_XMQ

#### 3.1.11.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = *ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ*

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPMODIFY

unPacketSize = 0

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

### 3.1.12 主题修改

#### 3.1.12.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = *ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ*

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQTOPICMODIFY

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ) + topicname

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:XMQ协议只需要填写主题名,末尾跟上要修改的主题名字符串即可

XENGINE\_PROTOCOL\_XMQ + tocpiname

#### 3.1.12.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = *ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ*

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPTOPICMODIFY

unPacketSize = 0

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

### 3.1.13 未读消息

#### 3.1.13.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = *ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ*

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQUNREAD

unPacketSize = 0

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:无

#### 3.1.13.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = *ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ*

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPUNREAD

unPacketSize = JSON

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:

{

"Array" :

[

{

"Count" : 11,

"Name" : "XEngine\_NotifyKey"

}

],

"Count" : 1

}

### 3.1.14 删除消息

#### 3.1.14.1 请求

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = *ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ*

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REQDELETE

unPacketSize = sizeof(XENGINE\_PROTOCOL\_XMQ)

byVersion = 1

byIsReply = TRUE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

协议体:根据消息Key和序列号来确定删除的消息

XENGINE\_PROTOCOL\_XMQ

#### 3.1.14.2 回复

协议头:

wHeader = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_HEADER

xhToken = 0

unOperatorType = *ENUM\_XENGINE\_COMMUNICATION\_PROTOCOL\_TYPE\_XMQ*

unOperatorCode = XENGINE\_COMMUNICATION\_PROTOCOL\_OPERATOR\_CODE\_MQ\_REPDELETE

unPacketSize = 0

byVersion = 1

byIsReply = FALSE

wReserve = 0

wPacketSerial = 0

wTail = XENGIEN\_COMMUNICATION\_PACKET\_PROTOCOL\_TAIL

## 3.2 WEBSOCKET

WebSocket协议负载JSON,返回也是,WebSocket请求需要心跳协议来包活,在没有数据交换的时候.

WebSocket的负载数据类型只能是TEXT.WEBSOCKET可以使用订阅模式

WEBSOCKET作为长连接,其操作方式与TCP相同

头字段的内容都与TCP的请求相对应.

st\_MQProtocol 在消息负载的时候才存在

st\_Auth 在登录的时候才存在

st\_User 在注册和删除用户的时候存在

st\_Payload表示负载的数据,byVersion表示负载类型,参考协议头的定义,如果是二进制,需要先base64编码后传输.

如果没有负载内容,st\_Payload可以不存在.

wReserve字段在订阅消息协议的时候必须存在

### 3.2.1 请求

{

"unOperatorType":7,

"unOperatorCode":7001,

"wReserve":1,

"byVersion":2,

"byIsReply":1,

"xhToken":2123123123123,

"st\_MQProtocol":{

"tszMQKey":"主题名",

"tszMQUsr":"所属用户,可为NULL",

"nSerial":0,

"nKeepTime":0,

"nPubTime":0,

"nMSGAttr":0

},

"st\_Auth":{

"tszUserName":"123123aa",

"tszUserPass":"123123",

"enClientType":10,

"enDeviceType":20

},

"st\_User":{

"nUserLevel":0,

"nUserState":0,

"nPhoneNumber":1333333,

"nIDNumber":511111,

"tszUserName":"123123aa",

"tszUserPass":"123123",

"tszEMailAddr":"41adawd@1111.com"

},

"st\_Payload":{

"nPayLen":6,

"tszPayData":"123456"

}

}

### 3.2.2 回复

{

"unOperatorType":7,

"unOperatorCode":7001,

"wReserve":0,

"byVersion":2,

"st\_MQProtocol":{

"tszMQKey":"主题名",

"tszMQUsr":"所属用户,可为NULL",

"nSerial":0,

"nKeepTime":0,

"nPubTime":0,

"nMSGAttr":0

},

"st\_Payload":{

"nPayType":5,

"nPayLen":6,

"tszPayData":"123456"

}

}

示例:消息发布

{

"unOperatorType": 7,

"unOperatorCode": 28672,

"wReserve": 0,

"byVersion": 2,

"byIsReply": 0,

"xhToken": 0,

"st\_MQProtocol": {

"tszMQKey": "XEngine\_CommKey",

"tszMQUsr": null,

"nSerial": 0,

"nKeepTime": 0,

"nPubTime": -1,

"nMSGAttr": 0

},

"st\_Payload": {

"nPayLen": 6,

"tszPayData": "123456"

}

}

## 3.3 MQTT

MQTT协议现在支持用户登录和用户消息发布以及订阅和取消订阅.其他协议将在后续版本支持.

MQTT不支持用户注册消息,用户注册消息需要通过TCP实现

MQTT使用5.0版本协议通信.必须填充用户和密码.

MQTT暂时不支持SSL

MQTT的消息发布协议需要按照WEBSOCKET协议填充JSON

MQTT支持QOS1和QOS0发布消息

## 3.4 HTTP

HTTP用作管理接口,可以通过HTTP协议来管理服务器.

http自带TOKEN验证,请求的用户必须是级别0的管理员权限,需要用户自己在数据库中修改级别.默认级别是普通用户20.请求方法如下

方法:GET

地址:http://127.0.0.1:5202/api?function=login&user=123123aa&pass=123123

得到的返回值msg里面会有token,这个token是数值,保存下来后每次请求都需要带这个token才可以.

所有请求,都应该带上token.下面的请求默认表示代上token

{

“xhToken”:123456789

}

### 3.2.1 查询用户列表

#### 3.2.1.1 请求

方法:POST

地址:http://127.0.0.1:5202/api?function=getuser

负载:

{

"xhToken": 123456789

}

#### 3.2.1.2 回复

{

"Array":[

{

"nIDNumber":511000000000101000,

"nPhoneNumber":13699999999,

"nUserLevel":0,

"nUserState":0,

"tszCreateTime":"2023-08-11 13:36:54",

"tszEMailAddr":"486179@qq.com",

"tszLoginTime":"2023-08-11 13:37:22",

"tszUserName":"123123aa",

"tszUserPass":"123123"

}

],

"Count":1,

"code":0

}

### 3.2.2 查询主题列表

#### 3.2.2.1 请求

方法:POST

地址:http://127.0.0.1:5202/api?function=getlist

负载:

{

"xhToken": 123456789

}

#### 3.2.2.2 回复

{

"Array":[

{

"tszTopicName":"XEngine\_CommKey"

},

{

"tszTopicName":"XEngine\_NotifyKey"

}

],

"Count":2,

"code":0

}

### 3.2.3 请求在线用户

#### 3.2.3.1 请求

方法:POST

地址:http://127.0.0.1:5202/api?function=online

负载:

{

"xhToken": 123456789

}

#### 3.2.3.2 回复

{

"Array":[

{

"tszUserAddr":"123123aa"

}

],

"Count":1,

"code":0

}

### 3.2.4 删除主题

#### 3.2.4.1 请求

方法:POST

地址:http://127.0.0.1:5202/api?function=deletetopic

负载:

{

"xhToken": 123456789,

"Object": {

"name": "operator name"

}

}

#### 3.2.4.2 回复

{

"msg": "success",

"code": 0

}

### 3.2.5 删除用户

#### 3.2.5.1 请求

方法:POST

地址:http://127.0.0.1:5202/api?function=deleteuser

负载:

{

"xhToken": 123456789,

"Object": {

"name": "operator name"

}

}

#### 3.2.5.2 回复

{

"msg": "success",

"code": 0

}

### 3.2.6 创建主题

#### 3.2.6.1 请求

方法:POST

地址:http://127.0.0.1:5202/api?function=createtopic

负载:

{

"xhToken": 123456789,

"Object": {

"name": "operator name"

}

}

#### 3.2.6.2 回复

{

"msg": "success",

"code": 0

}

### 3.2.7 获取主题

#### 3.2.7.1 请求

方法:POST

地址:http://127.0.0.1:5202/api?function=gettopic

负载:

{

"xhToken": 123456789,

"Object": {

"name": "operator name"

}

}

#### 3.2.7.2 回复

{

"msg": "success",

"lpszTopicName": “operator name”,

“Count”:1

}

# 四 配置说明

## 4.1 服务器配置

基础配置文件:XEngine\_Config.json

### 4.1.1 基本配置

* bDeamon: 1为守护进程后台运行0为前台运行
* tszTopic:默认公有主题名称
* nTCPPort:TCP通信端口
* nHttpPort:HTTP通信端口
* nWSPort:WEBSOCKET 通信端口
* nMQTTPort:mqtt 通信端口

### 4.1.2 最大配置

XMax 配置

* MaxClient 最大允许客户端
* MaxQueue 最大允许队列
* IOThread:网络IO处理线程个数
* nTCPThread:TCP处理线程个数
* nHttpThread:HTTP处理线程个数
* nWSThread:WEBSOCKET处理线程个数
* nMQTTThread:mqtt处理线程个数

### 4.1.3 日志配置

XLog 配置

* MaxSize:日志文件大小
* MaxCount:日志文件个数
* LogLeave:允许保存的级别
* LogType:日志类型
* tszLOGFile:日志文件路径

### 4.1.4 数据库配置

XSql 配置.需要MYSQL的支持

* SQLAddr:数据库地址
* SQLPort:数据库端口
* SQLUser:用户名
* SQLPass:密码

### 4.1.5 HTTP代理配置

XPass配置

* nTimeout:超时时间,单位秒
* tszPassRegister:注册代理地址,比如:http://127.0.0.1:5600/api/v1/register
* tszPassUNReg:注销代理地址,比如:http://127.0.0.1:5600/api/v1/unregister
* tszPassLogin:登录代理地址,比如:http://127.0.0.1:5600/api/v1/login
* tszPassLogout:登出代理地址,比如:http://127.0.0.1:5600/api/v1/logout

### 4.1.6 高速缓存配置

XMemory配置

* bDataQueryEnable:是否启用消息查询高速缓存
* bDataInsertEnable:是否启用消息插入缓存
* bUserQueryEnable:是否启用用户查询缓存
* nTimeLast:消息最后更新保存时间
* nTimeCount:消息最大保存时间

### 4.1.7 报告配置

XReport 配置

* bEnable:是否启用
* tszServiceName:服务名称
* tszAPIUrl:API地址

### 4.1.8 验证配置

XVerification配置,需要配合接口进行验证

* bEnable:是否启用HTTP接口验证
* nVType:验证类型,1 basic,2 digest
* tszAuthPass:三方接口验证地址,如果

### 4.1.9 消息通知

XNotify 配置

* EmailNotify,电子邮件通知
* bEnable:是否启用
* tszEMailSubject:消息通知标题
* tszServiceAddr:smtp邮箱地址
* tszUser:用户名
* tszPass:密码

## 4.2 数据库配置

配置文件:XEngine\_DBConfig.json 用于数据库操作的一些配置

### 4.2.1 用户数据库

#### 4.2.1.1 用户时间表

* bPubClear:定时发布完成是否清理数据库

### 4.2.2 消息数据库

MQData 配置

* nDBMonth:数据库保存时间,默认月
* bCommSub:用户注册默认订阅公用消息

# 五 高级配置

## 5.1 消息服务协议

typedef struct

{

*XCHAR* tszMQKey[*MAX\_PATH*];

*XCHAR* tszMQUsr[*MAX\_PATH*];

\_\_int64x nSerial;

\_\_int64x nPubTime;

int nKeepTime;

XENGINE\_PROTOCOL\_MSGATTR st\_MSGAttr;

}XENGINE\_PROTOCOL\_XMQ, \* LPXENGINE\_PROTOCOL\_XMQ;

意义可以参考相关头文件说明.需要注意nPubTime的说明和意思.

### 5.1.1 消息属性

目前消息属性已经支持基础功能.

只需要配置消息的时候设置其标志位为1就可以启用

目前支持以下两种消息属性

* byAttrAll:通知所有人.此消息类型的nPubTime属性必须为0,tszMQUsr将被忽略
* byAttrSelf:通知自己,自己也能收到此消息
* byAttrReply:对方必须回复,此消息暂时不支持
* byAttrActive:主动获取不受限制,比如超时的,不是自己发给自己的
* byAttrEMail:此消息还需要电子邮件通知,根据用户注册信息的电子邮件来通知
* byAttrPhone:此消息还需要短信通知,根据用户注册信息的电话号码来通知

# 六 其他功能

## 6.1 HTTP代理

支持HTTP代理的协议有:

* 登录
* 登出
* 注册
* 注销

# 附录

## 附录1 类型定义

参考文件 XEngine\_CommHdr.h

## 附录2 协议定义

参考文件XEngine\_ProtocolHdr.h

## 附录3 转换定义

参考文件 XEngine\_Types.h

## 附录4 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| 更新日期 | 更新说明 | 修改者 | 适用版本 |