

# AnyChat SDK Filter Plus

服务器扩展编程接口

## 开发指南

(版本: V4.9)



广州佰锐网络科技有限公司

**GuangZhou BaiRui Network Technology Co.,Ltd.**

<http://www.bairuitech.com> <http://www.anychat.cn>

2014 年 01 月

# 目 录

|                             |    |
|-----------------------------|----|
| 一、概述.....                   | 3  |
| 二、接口规范.....                 | 4  |
| 1 获取版本信息.....               | 4  |
| 2 初始化SDK FILTER PLUS.....   | 5  |
| 3 验证用户身份.....               | 5  |
| 4 用户进入房间请求.....             | 6  |
| 5 用户登录动作.....               | 7  |
| 6 用户注销动作.....               | 8  |
| 7 用户进入房间动作.....             | 8  |
| 8 用户离开房间动作.....             | 8  |
| 9 收到客户端通信数据包.....           | 9  |
| 10 收到用户文字聊天通信数据.....        | 9  |
| 11 释放SDK FILTER PLUS资源..... | 10 |
| 三、工作流程图.....                | 11 |
| 四、关于SDK FILTER BASE.....    | 12 |
| 五、SDK FILTER PLUS使用.....    | 12 |
| 六、技术支持.....                 | 13 |

## 一、概述

随着“AnyChat 语音视频互动平台”的不断完善，以及用户的不断增加，用户对于服务器端的 SDK 一直比较的期待，在之前的版本中，服务器并没有开放 SDK 编程接口，所以必须使用独立的数据库来进行身份验证，这样往往会造成视频系统一套用户数据库，上层的应用系统（如游戏系统、IM 平台等）也有一套用户数据库，而这两个数据库的存在，以及他们之间数据的同步一直为用户带来不便。

“SDK Filter Plus”是一套服务器扩展编程接口，通过该接口，可以编写符合服务器调用规范的插件，在插件里面完成一些业务层面逻辑处理的任务，如连接上层应用的数据库验证用户身份等功能，从而实现 AnyChat SDK 服务器满足各种应用需求的需要。

SDK Filter 的形式为动态连接库（DLL），该动态库需要按 SDK Filter 的编程接口提供相关的方法，供服务器调用。在服务器的配置文件（AnyChatCoreServer.ini）中，可以配置动态连接库的名称，服务器根据配置信息来加载 SDK Filter，如果加载成功，会在服务器的日志输出信息文件中输出相应的提示信息。

我们在 SDK 包中提供了“SDK Filter Plus”插件的示例代码供参考，位于 SDK 包的 src\server\c++文件夹下面。

自 V2.9 版本开始，服务器端支持新的开发接口：“AnyChat Server SDK”，与“SDK Filter Plus”类似可以实现相同的功能，通常来说，“SDK Filter Plus”适合业务逻辑较简单的应用，而“AnyChat Server SDK”则适合业务逻辑较复杂的应用，实现独立的业务层服务器，有对应的界面显示。

有关“AnyChat Server SDK”的详细介绍可参考相关的开发文档（《AnyChat Server SDK 开发指南》）及 SDK 包中所附带的相关源代码。

## 二、接口规范

本节主要讲述 AnyChat 服务器 SDK Filter Plus 的接口，所有 SDK Filter Plus 都必须按下面的要求提供所有的接口，如果所提供的接口没有按接口规范规定的要求来提供，可能导致服务器不稳定，或是加载 SDK Filter Plus 失败。

SDK Filter Plus 的文件名可以为任意，没有限制，但是接口必须是动态连接库（DLL）的形式。

SDK Filter Plus 采用“C 调用约定”（即用 `__cdecl` 关键字说明），`__cdecl` 是 C 和 C++ 程序的缺省调用方式，如果是采用其它语言编写 SDK Filter Plus 插件需要特别注意是否满足该调用方式。接口定义采用“C 编译方式”（即用 `extern "C"` 修饰符）并需要去掉输出函数名的下划线前缀（可使用 MFC 提供的 `__declspec(dllexport)` 修饰符），详细信息可参考 SDK 包中的 SDK Filter Plus 示例代码。

### 1 获取版本信息

接口定义：BRFP\_API void **BRFP\_GetVersion**(OUT LPTSTR lpVersion, IN DWORD dwVSLen)

参 数：

|        |           |              |
|--------|-----------|--------------|
| LPTSTR | lpVersion | 保存版本信息的缓冲区   |
| DWORD  | dwVSLen   | 输入的版本信息缓冲区长度 |

返回值：

void

备 注：

该方法用于获取 SDK Filter Plus 的版本信息，传入一个缓冲区，在函数体里面将版本信息拷贝到该缓冲区，版本信息可以包含版本值，公司信息等，如果该 SDK Filter Plus 被服务器加载成功，版本信息将会被输出到服务器的日志信息文件中，便于用户调试。dwVSLen 用于指示传入的缓冲区大小，防

止函数体里面拷贝数据时内存溢出。

## 2 初始化SDK Filter Plus

**接口定义：** BRFP\_API BOOL **BRFP\_InitFilter**(BRFP\_FilterDataCallBack lpFunc, LPVOID lpUserValue)

### 参 数：

BRFP\_FilterDataCallBack lpFunc 发送数据回调函数句柄，当 SDK Filter Plus 有数据需要发送时，则可通过该回调函数句柄传递给服务器，由服务器将数据发送到客户端（单次最大发送缓冲区大小为 1200 字节，超过该大小，将只会发送前 1200 字节的缓冲区）。

LPVOID lpUserValue 回调函数附带参数，在发送数据时，必须使用该参数。

### 返回值：

BOOL 初始化成功返回 TRUE，否则返回 FALSE

### 备 注：

可以在该方法中读取相关的配置参数（SDK Filter Plus 可以有自己的参数配置文件，如保存连接数据库的相关参数，而不用把这些参数固定的程序里面），建立数据库连接等初始化工作。无论是初始化成功，还是初始化失败，在服务器的日志输出信息文件中都将输出相关的提示信息，便于用户调试。

## 3 验证用户身份

**接口定义：** BRFP\_API DWORD **BRFP\_VerifyUser**(IN LPCTSTR lpUserName, IN LPCTSTR lpPassword, OUT LPDWORD lpUserID, OUT LPDWORD lpUserLevel, OUT LPTSTR lpNickName, IN DWORD dwNCLen)

### 参 数：

|                    |                |
|--------------------|----------------|
| LPCTSTR lpUserName | 用于身份验证的用户名     |
| LPCTSTR lpPassword | 用于身份验证的用户密码    |
| LPDWORD lpUserID   | 输出用户的 ID 号（唯一） |

|         |             |                   |
|---------|-------------|-------------------|
| LPDWORD | lpUserLevel | 输出用户的级别           |
| LPTSTR  | lpNickName  | 输出用户的昵称（可以与用户名相同） |
| DWORD   | dwNCLen     | 保存用户昵称的缓冲区大小      |

返回值:

DWORD 返回出错代码, 参考 SDK\GVErrorCodeDefine.h 文件中有关出错代码的定义, 通常会返回如下的值:

GV\_ERR\_SUCCESS (0) 表示验证通过, 允许用户登录系统

GV\_ERR\_CERTIFY\_FAIL(200) 用户名或密码有误, 认证失败

GV\_ERR\_ACCOUNT\_LOCK(202) 帐户已被暂时锁定, 认证失败

备 注:

该方法用于验证用户的身份信息。用户名与用户密码为输入参数, 后面其它的参数为输出参数, 用于反馈结果给服务器程序。身份验证是否成功通过该方法的返回值来判定。dwVSLen 用于指示传入的缓冲区大小, 防止函数体里面拷贝数据时内存溢出。如果该方法不输出用户的昵称, 即当 lpNickName 的长度为 0 时, 服务器会自动将用户名默认为用户的昵称。

该方法仅当客户端调用 **BRAC\_Login** 方法, 而且传入密码参数后方能被服务器所调用, 如果客户端登录时不传入密码参数, 服务器将会以游客的方式完成客户端的登录请求, ID 由服务器自动分配。

## 4 用户进入房间请求

接口定义: BRFP\_API DWORD **BRFP\_PrepareEnterRoom**(DWORD dwUserId, DWORD dwRoomId, LPCTSTR lpRoomName, LPCTSTR lpPassword)

参 数:

DWORD dwUserId 请求者用户 ID

DWORD dwRoomId 请求进入的房间编号

LPCTSTR lpRoomName 请求进入的房间名

LPCTSTR lpPassword 房间密码

返回值:

DWORD 返回出错代码, 参考 SDK\GVErrorCodeDefine.h 文件中有关

出错代码的定义。

#### 备 注:

该方法将在客户端 SDK 调用进入房间的相关 API 接口后将被触发, SDK Filter Plus 需要判断该请求的合法性, 服务器根据返回值决定是否让用户进入房间。

当客户端调用 BRAC\_EnterRoom (根据房间 ID 进入房间) 进入房间时, dwRoomId 与 lpPassword 参数有效。

当客户端调用 BRAC\_EnterRoomEx (根据房间名进入房间) 进入房间时, lpRoomName 与 lpPassword 参数有效, 此时的 dwRoomId 是由服务器自动分配的。

## 5 用户登录动作

接口定义: BRFP\_API void **BRFP\_OnUserLoginAction**(DWORD dwUserId, LPCTSTR szUserName, DWORD dwLevel, LPCTSTR szIpAddr, DWORD dwHandle)

#### 参 数:

DWORD dwUserId 登录用户 ID

LPCTSTR szUserName 登录用户名

DWORD dwLevel 登录用户级别

LPCTSTR szIpAddr 登录用户 IP 地址

DWORD dwHandle 用户句柄, 在 SDK Filter Plus 中向用户发送数据时需要使用该句柄

返回值: void

#### 备 注:

当用户登录系统成功之后, 服务器会调用该方法, 通知 SDK Filter Plus 有用户登录。

这儿需要保存 dwUserId 与 dwHandle 的对应关系, 因为向用户发送数据 (使用回调函数句柄) 时需要传入该参数

## 6 用户注销动作

接口定义：BRFP\_API void **BRFP\_OnUserLogoutAction**(DWORD dwUserId)

参 数：

DWORD dwUserId 注销用户的 ID

返回值： void

备 注：

当用户从系统注销，或是网络断开之时，服务器会调用该方法，通知 SDK Filter Plus 有用户离开系统。

## 7 用户进入房间动作

接口定义：BRFP\_API void **BRFP\_OnUserEnterRoomAction**(DWORD dwUserId, DWORD dwRoomId)

参 数：

DWORD dwUserId 进入房间用户的 ID

DWORD dwRoomId 进入的房间 ID

返回值： void

备 注：

当用户进入房间成功之后，服务器会调用该方法，通知 SDK Filter Plus 有用户进入房间。

## 8 用户离开房间动作

接口定义：BRFP\_API void **BRFP\_OnUserLeaveRoomAction**(DWORD dwUserId, DWORD dwRoomId)

参 数：

DWORD dwUserId 离开房间用户的 ID

DWORD dwRoomId 离开的房间 ID

返回值： void

备 注：



当用户从房间离开之时，服务器会调用该方法，通知 SDK Filter Plus 有用户离开房间。

## 9 收到客户端通信数据包

接口定义: BRFP\_API void **BRFP\_OnRecvUserFilterData**(DWORD dwUserId, LPCSTR lpBuf, DWORD dwLen)

参 数:

|        |          |            |
|--------|----------|------------|
| DWORD  | dwUserId | 发送数据的用户 ID |
| LPCSTR | lpBuf    | 数据缓冲区      |
| DWORD  | dwLen    | 缓冲区大小      |

返回值: void

备 注:

当客户端调用“BRAC\_SendSDKFilterData”方法向服务器发送数据时，服务器会调用该方法，把客户端发送的数据交给 SDK Filter Plus 来处理，由于数据缓冲区可以由上层应用自定义，从而能达到实现更复杂应用的目的，加强了 AnyChat SDK 的可扩展性。

如 SDK Filter Plus 需要向客户端发送自定义数据包，则可采用“BRFP\_InitFilter”方法所由服务器所传入的通信回调句柄进行数据发送。

## 10 收到用户文字聊天通信数据

接口定义: BRFP\_API void **BRFP\_OnRecvUserTextMsg**(DWORD dwRoomId, DWORD dwSrcUserId, DWORD dwTarUserId, BOOL bSecret, LPCTSTR lpTextMessage, DWORD dwLen)

参 数:

|         |               |                         |
|---------|---------------|-------------------------|
| DWORD   | dwRoomId      | 文字消息所对应的房间编号            |
| DWORD   | dwSrcUserId   | 源用户 ID，文字消息发送者          |
| DWORD   | dwTarUserId   | 目标用户 ID，消息接收者，-1 表示所有人  |
| BOOL    | bSecret       | 是否为悄悄话，目标用户 ID 不为-1 时有效 |
| LPCTSTR | lpTextMessage | 文字消息内容                  |

DWORD      dwLen      文字消息长度

返回值:      void

备注:

当客户端调用“BRAC\_SendTextMessage”向房间内用户发送文字消息时，服务器将会调用该方法。

在某些应用场合中，需要在服务器上保存用户的所有文字聊天记录，则可通过该方法来完成，如可将相关内容写入数据库中。

## 11 释放SDK Filter Plus资源

接口定义: BRFP\_API void **BRFP\_ReleaseFilter**(void)

参 数:

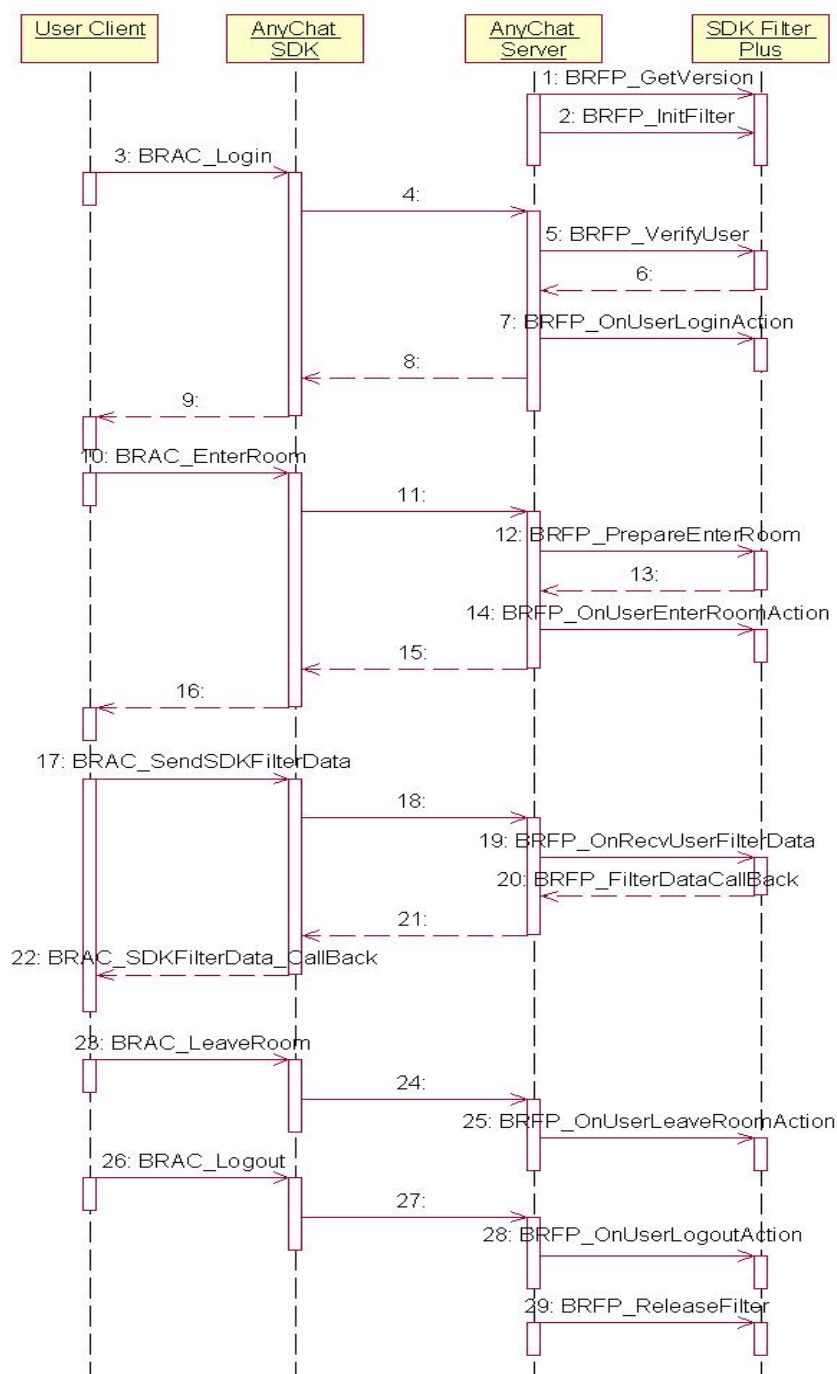
void

返回值:      void

备 注:

该方法将在服务器退出时被调用，用于释放 SDK Filter Plus 打开的相关资源，如关闭数据库等。

### 三、工作流程图



## 四、关于SDK Filter Base

目前 SDK Filter 分两个版本，分别是：SDK Filter Base 与 SDK Filter Plus（最低需求：V2.7 版本 SDK）。早期的 SDK Filter Base 主要实现了将用户的身份验证过程提交给 SDK Filter 来完成，功能相对简单。目前的 SDK Filter Plus 实现了更多的接口，可以让客户端与 Filter 插件之间通信，同时在服务器端也开放了更多的接口，可扩展性更强。

早期的 SDK Filter Base 提供的接口定义如下：

### 1、获取 SDK 插件版本信息

BRGS\_API void **BRGS\_GetVersion**(OUT LPTSTR lpVersion,IN DWORD dwVSLen)

### 2、初始化 SDK Filter

BRGS\_API BOOL **BRGS\_InitFilter**(void)

### 3、验证用户身份

BRGS\_API DWORD **BRGS\_VerifyUser**(IN LPCTSTR lpUserName,IN LPCTSTR lpPassword, OUT LPDWORD lpUserID, OUT LPDWORD lpUserLevel, OUT LPTSTR lpNickName,IN DWORD dwNCLen)

### 4、释放 SDK Filter 资源

BRGS\_API BOOL **BRGS\_ReleaseFilter**(void)

有关 SDK Filter Base 相关接口的详细信息，请参考 SDK 中提供的相关源代码（VC++版本），由于该接口已经被 SDK Filter Plus 替代，故不再另行提供独立的开发指南文档。

## 五、SDK Filter Plus使用

当按本《开发指南》所要求的接口准备好 SDK Filter Plus 插件（DLL 动态库）后，需要配置给服务器，然后服务器才能加载该插件。

在服务器的配置文件 AnyChatCoreServer.ini 中，默认有如下两项参数配置项：

[Function Settings]

CloseFrontLink=1

SDKFilter=

SDKFilterPlus=

AutoAddRoom=1

其中“SDKFilter”配置项用于配置早期的“SDK Filter Base”插件，“SDKFilterPlus”配置项用于配置本《开发指南》所描述的“SDK Filter Plus”插件。该两配置项默认为空，表示不加载任何 SDK Filter 插件。

当需要使用 SDK Filter Plus 插件时，首先请将插件的 DLL 文件（如 SDK 包中的 SDKFilterPlus.dll 文件）拷贝到服务器程序所在的目录，然后将“SDKFilterPlus”配置项改为“SDKFilterPlus=SDKFilterPlus.dll”，保存配置文件后，最后重启服务器程序，通过服务器的日志文件“AnyChatCoreServer.log”便可判断 SDK Filter Plus 插件是否被服务器加载成功。

## 六、技术支持

在使用“AnyCha SDK Filter Plus”和“AnyChat Platform Core SDK”的过程中遇到技术问题，需要帮助时，请通过下面的联系方式与我们联系，我们将热忱为您服务：

AnyChat官方网站：<http://www.anychat.cn/>（了解AnyChat最新动态）

AnyChat技术论坛：<http://bbs.anychat.cn/>（技术问题交流）

AnyChat知识中心：<http://www.anychat.cn/faq/>（常见技术问题均有详细解答）

佰锐科技官方网站：<http://www.bairuitech.com/>

TEL: +86-020-85276986

QQ: 75513809

Email: [service@bairuitech.com](mailto:service@bairuitech.com)