《中间件》聊天系统

版本：2.6

# 系统概述

聊天系统包括综合信息，全服信息、联盟信息、私聊信息；方便玩家之间的交流与互动；聊天的信息按时间顺序显示，玩家的发言信息可以屏蔽敏感词，内容字数限制，发言的频率时间间隔限制；

# 系统设计

使用ChatService基类操作聊天信息，提供发送，接收，敏感词过滤功能

1. **发送信息**

Send方法：向全服发送信息，存储在全服缓存ChatCacheSet中；

SendWhisper方法：给玩家发送私聊信息

1. **接收信息**

Receive方法：根据玩家和已接收的版本ID从缓存ChatCacheSet中筛选出聊天信息；

# 使用示例

1. 新建DemoChatService类继承ChatService基类

|  |
| --- |
| public class DemoChatService : ChatService  {  private readonly GameUser \_user;  private const int MsgTimeOut = 0;//分钟  private const int IntervalSend = 10;  private SensitiveWordService \_wordServer;  public DemoChatService()  : this(new GameUser())  {  }  public DemoChatService(GameUser user)  : base(user.UserId.ToInt())  {  \_wordServer = new SensitiveWordService();  this.\_user = user;  }  public static bool IsAllow(GameUser user, ChatType chatType)  {  if (user != null)  {  if (chatType == ChatType.World && (DateTime.Now - user.Property.ChatDate).TotalSeconds >= IntervalSend)  {  return true;  }  else if (chatType != ChatType.World)  {  return true;  }  }  return false;  }  public void SystemSend(ChatType chatType, string content)  {  var chat = new ChatData();  chat .Version = NextVersion  Send(chat);  }  public void Send(ChatType chatType, string content)  {  var chat = new ChatData();  chat .Version = NextVersion  \_user.Property.ChatDate = DateTime.Now;  Send(chat);  }  public void SendWhisper(GameUser toUser, string content)  {  if (\_user == null || toUser == null)  {  throw new Exception("发送人或接收人为空值");  }  \_user.Property.ChatDate = DateTime.Now;    var chat = new ChatData();  chat .Version = NextVersion  SendWhisper(toUser.UserId.ToInt(), chat);  }  protected override List<ChatMessage> GetRange(List<ChatMessage> msgList)  {  if (msgList.Count > 50)  {  int pageCount;  return msgList.GetPaging(1, 50, out pageCount);  }  return msgList;  }  protected override bool HasReceive(ChatMessage message)  {  var m = message as ChatData;  return m != null && m.Version > \_user.Property.ChatVesion; //&& m.SendDate.AddMinutes(MsgTimeOut) > DateTime.Now;  }  protected override string FilterMessage(string message)  {  return \_wordServer.Filter(message);  }  protected override void WriteLog(ChatMessage message)  {    }  } |

1. 处理发送（Python脚本）

|  |
| --- |
| def takeAction(urlParam, parent):  user = parent.Current.User;  chatService =DemoChatService(user)  chatService.Send(ChatType.Whisper,chatinfo.Id, chatinfo.Content)  return |

1. 处理接收（Python脚本）

|  |
| --- |
| def takeAction(urlParam, parent):  user = parent.Current.User;  chatService = DemoChatService(user)  actionResult.ChatMaxNum = chatService.CurrVersion  chatList = chatService.Receive()  return |