Openfire 模块分析

# PresenceManager

**ofPresense table**

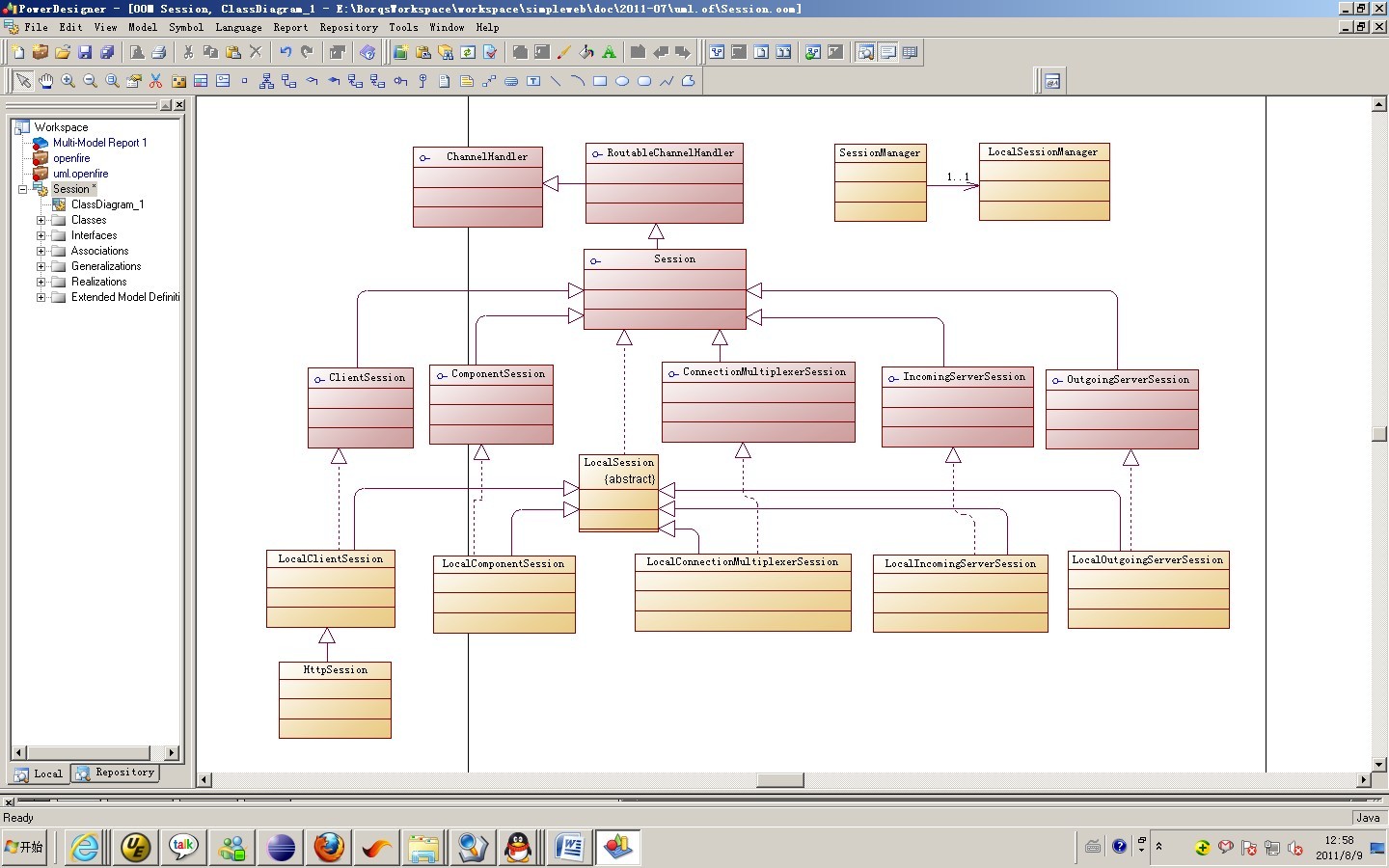
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 主键 | 类型 | 约束 |  |
| username | 是 | varchar(64) | NOT NULL |  |
| offlinePresence |  | text |  |  |
| offlineDate |  | char(15) | NOT NULL |  |

提供了Presence 信息的数据库操作。

例如: isAvailable

getPresence

# SessionManager



LocalSessionManager本地会话管理职责是保持那些已经连接到java虚拟机并且不在路由表中的session 。

1、在client或者http client create Session时, Session存放在PreAuthenticatedSessions Map.

2、当授权成功时调用LocalClientSession. setAuthToken

或者当匿名授权时LocalClientSession.setAnonymousAuth

调用SessionManager.addSession

将PreAuthenticatedSessions 移除，

将Session Address 存入routingTable

添加Session改变事件

所以openfire中的session 包含这几个层次：

Mina IoSession

PreAuthenticatedSession

routingTable Session

3、getSession

如果JID 合法，先在PreAuthenticatedSessions 中查找。如果为空，

再在routingTable 中查找4、Session 类型概述

1. LocalClientSession表示Client-xmpp Server Session ,
2. HttpSession表示Http Client-xmpp Server Session
3. LocalConnectionMultiplexerSession表示连接管理器-xmpp Server Session
4. LocalOutgoingServerSession和LocalIncomingServerSession 表示xmpp Server之间的Session

E、 RemoteSessionLocator 用于集群

登录过程：IQAuthHandler，SASLAuthentication

# Offline Message

1、数据库表

**ofOffline table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 主键 | 类型 | 约束 |  |
| username | 是 | varchar(64) | NOT NULL |  |
| messageID | 是 | bigint(20) | NOT NULL | 自动升序生成 |
| creationDate |  | char(15) | NOT NULL |  |
| messageSize |  | int(11) | NOT NULL |  |
| stanza |  | text | NOT NULL |  |

2、离线存储策略类型：

bounce

All messages are bounced to the sender.

drop

All messages are silently dropped.

store

All messages are stored.

store\_and\_bounce

Messages are stored up to the storage limit, and then bounced.(离线消息被有限存储，超限消息返还发送者)

store\_and\_drop

Messages are stored up to the storage limit, and then silently dropped.

默认每人的离线消息： 100KB，可以由xmpp.offline.quota配置

默认离线存储策略：store\_and\_bounce，可以由xmpp.offline.type配置

要对离线消息进行自定义处理，写一个插件，实现OfflineMessageListener 接口，

OfflineMessageStrategy.addListener并添加到listener队列，即可。

# VCardManager

1. 数据库表

ofVCard table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段 | 主键 | 类型 | 约束 |  |
| username | 是 | text | NOT NULL |  |
| vcard |  | mediumtext | NOT NULL |  |

DefaultVCardProvider 提供了对Vcard 的数据库操作。

VCardEventDispatcher，VCardListener 提供了Vcard事件处理的操作。

可以在自己插件中实现VCardListener接口对Vcard进行操作，调用

VCardEventDispatcher 将自己的监听器加到监听器列表中去。

# PacketRouter 和ChannelHandler



在PacketRouterImpl 中引用IQRouter，MessageRouter，PresenceRouter

MulticastRouter 分别被IQRouter，MessageRouter，PresenceRouter引用

RemotePacketRouter 在openfire 中并未实现。

......

ChannelHandler 是各个handler 的父接口。各个handler处理不同的命名空间。

RoutingTable中存放的数据

usersCache： Key: full JID, Value: {nodeID, available/unavailable}

anonymousUsersCache：Key: full JID, Value: {nodeID, available/unavailable}

usersSessions：Key: bare JID, Value: list of full JIDs of the user

# Connection、ConnectionHandler、StanzaHandler

ClientStanzaHandler 用于Client直接连到Server时XML stanzas的处理。

ServerStanzaHandler用于Server-to-server communication XML stanzas的处理。

MultiplexerStanzaHandler 用于Connection Managers XML stanzas的处理。

ServerDialback 用于防止域名欺骗，如果用SASL做S-to-S的认证,dialback就没有必要使用了。

对客户端连接的响应都采用NIOConnction.

ConnectionHandler. messageReceived调用StanzaHandler. process 调用PacketRouterImpl. route

客户服务器间传输的包IQRouter ，PresenseRouter，MessageRouter会在router方法中调用LocalSession.process 进行处理

LocalClientSession.canProcess 用于判断包要不要被阻塞掉。

# IQHandler

## 纵述IQHandler

ChannelHander 的直接子类和IQHandler的子类是用来实现xmpp的协议。

IQHandler 的主要职责是处理IQ Packet，它的子类只需要指定IQHandlerInfo (也就是要处理包的名字，命名空间)和处理方法。所有的命名空间见<http://xmpp.org/resources/schemas/>

命名空间对应的xmpp协议见<http://xmpp.org/resources/schemas/>

IQ具体命名空间对应到相应的handler 见IQRouter.route(IQ packet)

## AdHandlerCommandHandler

对应的xmpp协议是[XEP-0050: Ad-Hoc Commands](http://xmpp.org/extensions/xep-0050.html) 通常用来和[XEP-0004](http://wiki.jabbercn.org/XEP-0004)(定义了向服务器提交的数据表单) 交换信息，响应表单的请求。



采用命令模式，若干命令到继承AdHocCommand，

如：查询一些统计信息的命令，如：一些获取当前在线用户数，获得当前活跃用户数。

执行用户增删改查的操作的命令

组的增删改查的操作的命令

执行VCard的相关操作的命令

…….

在AdHocCommandManager中提供了命令的相关操作，如：添加一个命令，删除一个命令，执行一个命令等操作。

已定义的命令见：AdHocCommandHandler . addDefaultCommands。

## IQAuthHandler

对应的协议是[XEP-0078](http://www.xmpp.org/extensions/xep-0078.html)，即jabber:iq:auth protocol。非SASL的认证，用于用户的登录，客户端使用本议定书和服务器认证。现在已经过时。

## IQBindHandler

实现RFC3920中资源绑定的一节，用于sasl协商后，为了满足RFC3920定义的寻址格式(第三章)和节传输规则(第十章)，客户端<node@domain>必须拥有一个相关的资源ID（由服务器生成或由客户端程序提供）；以确保在流上使用的地址是一个“全JID”（<node@domain/resource>）。比如：

demo@slieer-computer/spark、

abc@slieer-computer/Spark 2.6.3

## IQDiscoInfoHandler

负责处理disco#info的请求，对应的协议是[XEP-0030](http://xmpp.org/extensions/xep-0030.html)

1、client获取注册的Server feature 和cluster node 信息。

2、查询某个JID，JEP的例子说应答服务器在确认请求方是可以信任的前提下可以返回这个用户是否注册。

3、查询 conference room (muc)

## IQDiscoItemsHandler

负责处理disco#items的请求，对应的协议是[XEP-0030](http://xmpp.org/extensions/xep-0030.html)

~~从XMPPServer . getServerItemsProviders() 分析，~~

## IQLastActivityHandler

对应的协议是[XEP-0012](http://xmpp.org/extensions/xep-0012.html) ,即jabber:iq:last 协议，允许用户找到另一个用户已经离线的秒数，只提供给那些已经订阅用户。

## IQOfflineMessagesHandler

对应的协议是JEP-0013，按照client请求的规则，将离线发送给client。

## IQPEPHandler

对应的协议是[XEP-0163](http://xmpp.org/extensions/xep-0163.html) Personal Eventing via Pubsub

## IQPEPOwnerHandler

对应的协议是XEP-0163

## IQPingHandler

对应的协议是实现XEP-0199

## IQPrivacyHandler

处理隐私的列表。

## IQPrivateHandler

jabber:iq:private protocol

## IQRegisterHandler

jabber:iq:register protocol

## IQRosterHandler

jabber:iq:roster protocol.

## IQSessionEstablishmentHandler

对应的协议是[RFC6121](http://xmpp.org/rfcs/rfc6121.html) / RFC3921 session建立一节。.

## IQSharedGroupHandler

Openfire 自已定义的Handler

## IQTimeHandler

jabber:iq:time protocol (time info) as as defined by JEP-0090. 已不赞使用。

## IQvCardHandler

vcard-temp protocol

## IQVersionHandler

xmpp server目前的版本信息

## STUNIQHandler

它提供了对p2p session地址发现，用于媒体传输和接收UDP数据包。

## 自定义IQ

1. 确定IQ命名空间命名，xml Schemas
2. 继承IQHandler
3. IQRouter. addHandler(IQHandler) 添加到Handler列表
4. 如果iq 要通过未认证的xmpp，则需要修改IQRouter . route(IQ packet)的源码

### 自定义Form 处理器

1. 继承AdHocCommand
2. 调用AdHocCommandManager. addCommand添加到命令列表

# MulticastDNSService

# MUCUser



# Presence Handler

PresenceSubscribeHandler

PresenceUpdateHandler