

- 1、发送快连消息 FastConnectMessage
- 2、接收快连成功消息 FastConnectOkMessage

发送快速连接消息

与MPUSH服务端建立连接时，发送快连消息

```
1 //ConnClientChannelHandler.java
2
3 // 创建conn对象
4 private final Connection connection = new NettyConnection();
5 //建立连接事件方法
6 @Override
7 public void channelActive(ChannelHandlerContext ctx) throws Exception {
8     int clientNum = STATISTICS.clientNum.incrementAndGet();
9     LOGGER.info("client connect channel={}, clientNum={}", ctx.channel(), clientNum);
10    for (int i = 0; i < 3; i++) {
11        if (clientConfig != null) break;
12        clientConfig = ctx.channel().attr(CONFIG_KEY).getAndSet(null);
13        if (clientConfig == null) TimeUnit.SECONDS.sleep(1);
14    }
15    if (clientConfig == null) {
16        throw new NullPointerException("client config is null, channel=" + ctx.channel());
17    }
18    connection.init(ctx.channel(), true); //初始化sessionContext、设置RSA加密
19    if (perfTest) {
20        handshake();
21    } else {
22        tryFastConnect(); // 快连
23    }
24 }
25 private void tryFastConnect() {
26     Map<String, String> sessionTickets =
27         getFastConnectionInfo(clientConfig.getDeviceId());
28     //如果session为空，则先握手
29     if (sessionTickets == null) {
30         handshake();
31     }
32     return;
```

```
32 //如果sessionId不存在, 则握手
33 String sessionId = sessionTickets.get("sessionId");
34 if (sessionId == null) {
35     handshake();
36     return;
37 }
38 String expireTime = sessionTickets.get("expireTime");
39 if (expireTime != null) {
40     long exp = Long.parseLong(expireTime);
41     //session过期, 则握手
42     if (exp < System.currentTimeMillis()) {
43         handshake();
44         return;
45     }
46 }
47 final String cipher = sessionTickets.get("cipherStr");
48 FastConnectMessage message = new FastConnectMessage(connection);
49 message.deviceId = clientConfig.getDeviceId(); //设备ID
50 message.sessionId = sessionId; //会话ID
51 message.sendRaw(channelFuture -> {
52     if (channelFuture.isSuccess()) {
53         //发送成功, 继续设置当前的加密方式
54         clientConfig.setCipher(cipher);
55     } else {
56         //快连失败, 则握手
57         handshake();
58     }
59 });
60 LOGGER.debug("send fast connect message={}", message);
61 }
62 //获取session
63 private Map<String, String> getFastConnectionInfo(String deviceId) {
64     //key = "mp:fcd:<deviceId>"
65     String key = CacheKeys.getDeviceIdKey(deviceId);
66     return cacheManager.get(key, Map.class);
67 }
68 //保存session
69 private void saveToRedisForFastConnection(ClientConfig client, String sessionId, Long expireTime, byte[] sessionKey) {
70     Map<String, String> map = Maps.newHashMap();
```

```

71  map.put("sessionId", sessionId);
72  map.put("expireTime", expireTime + "");
73  map.put("cipherStr", connection.getSessionContext().cipher.toString());
74  String key = CacheKeys.getDeviceIdKey(client.getDeviceId());
75  cacheManager.set(key, map, 60 * 5); //5分钟
76  }
77  //发送握手消息
78  private void handshake() {
79      HandshakeMessage message = new HandshakeMessage(connection);
80      message.clientKey = clientConfig.getClientKey();
81      message.iv = clientConfig.getIv();
82      message.clientVersion = clientConfig.getClientVersion();
83      message.deviceId = clientConfig.getDeviceId();
84      message.osName = clientConfig.getOsName();
85      message.osVersion = clientConfig.getOsVersion();
86      message.timestamp = System.currentTimeMillis();
87      message.send();
88      LOGGER.debug("send handshake message={}", message);
89  }

```

接收快连成功消息

```

1  //ConnClientChannelHandler.java
2
3  @Override
4  public void channelRead(ChannelHandlerContext ctx, Object msg) throws Exception {
5      connection.updateLastReadTime();
6      if (msg instanceof Packet) {
7          Packet packet = (Packet) msg;
8          Command command = Command.toCMD(packet.cmd);
9          if (command == Command.HANDSHAKE) {
10             int connectedNum = STATISTICS.connectedNum.incrementAndGet();
11             connection.getSessionContext().changeCipher(new
AesCipher(clientConfig.getClientKey(), clientConfig.getIv()));
12             HandshakeOkMessage message = new HandshakeOkMessage(packet,
connection);
13             message.decodeBody();
14             byte[] sessionKey = CipherBox.I.mixKey(clientConfig.getClientKey(), message.serverKey);

```

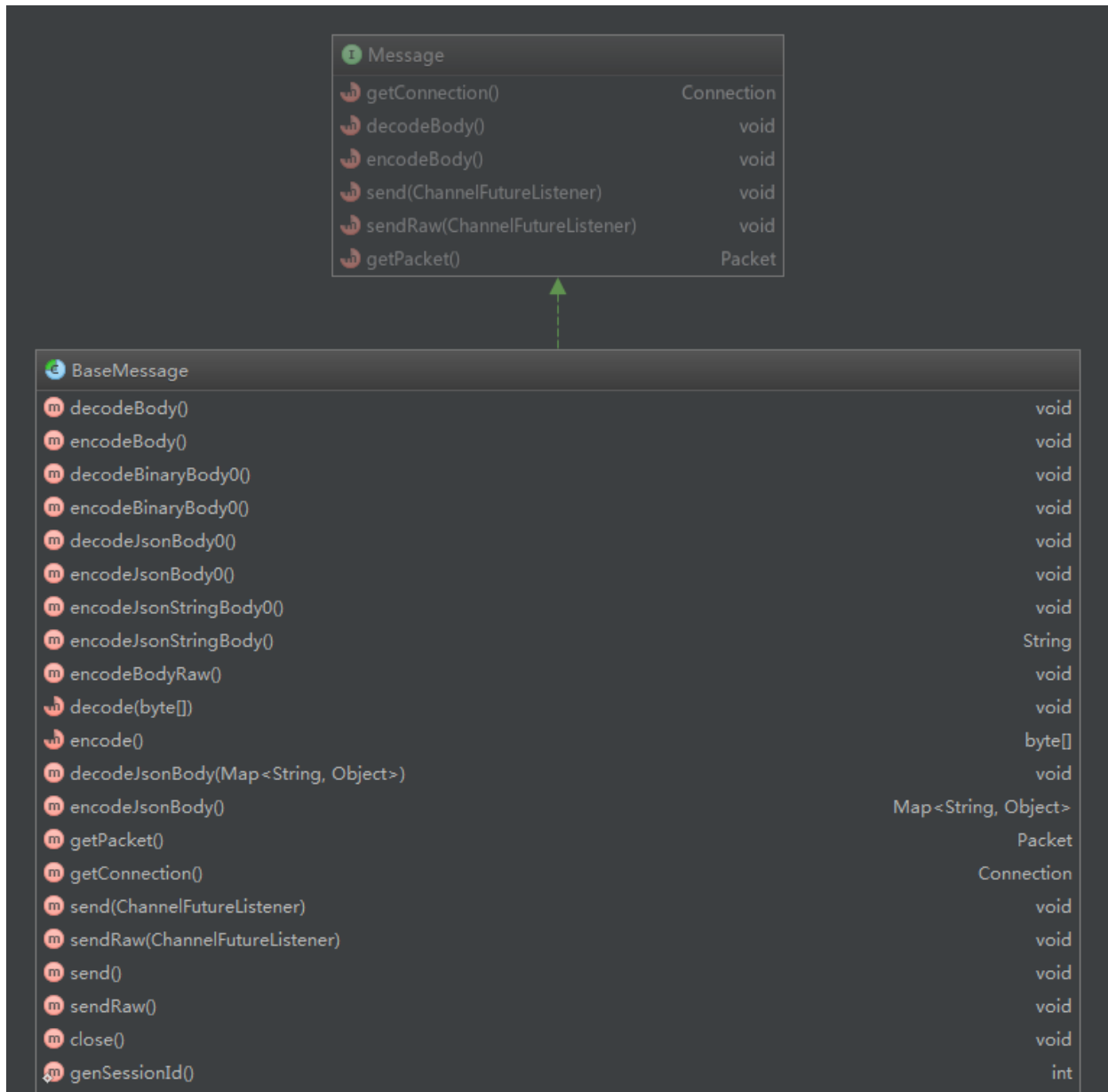
```
15 connection.getSessionContext().changeCipher(new AesCipher(sessionKey, clientConfig.getIv()));
16 connection.getSessionContext().setHeartbeat(message.heartbeat);
17 startHeartBeat(message.heartbeat - 1000);
18 LOGGER.info("handshake success, clientConfig={}, connectedNum={}", clientConfig, connectedNum);
19 bindUser(clientConfig);
20 if (!perfTest) {
21     saveToRedisForFastConnection(clientConfig, message.sessionId, message.expireTime, sessionKey);
22 }
23 } else if (command == Command.FAST_CONNECT) {
24     int connectedNum = STATISTICS.connectedNum.incrementAndGet();
25     String cipherStr = clientConfig.getCipher();
26     String[] cs = cipherStr.split(",");
27     byte[] key = AesCipher.toArray(cs[0]);
28     byte[] iv = AesCipher.toArray(cs[1]);
29     //设置AES解密
30     connection.getSessionContext().changeCipher(new AesCipher(key, iv));
31     FastConnectOkMessage message = new FastConnectOkMessage(packet, connection);
32     message.decodeBody(); //解码消息body
33     connection.getSessionContext().setHeartbeat(message.heartbeat);
34     //发送心跳消息
35     startHeartBeat(message.heartbeat - 1000);
36     //发送绑定用户消息
37     bindUser(clientConfig);
38     LOGGER.info("fast connect success, clientConfig={}, connectedNum={}", clientConfig, connectedNum);
39 } else if (command == Command.KICK) {
40     ...
41 } else if (command == Command.ERROR) {
42     ErrorMessage message = new ErrorMessage(packet, connection);
43     message.decodeBody();
44     LOGGER.error("receive an error packet=" + message);
45 } else if (command == Command.PUSH) {
46     ...
47 } else if (command == Command.HEARTBEAT) {
48     LOGGER.info("receive heartbeat pong...");
49 } else if (command == Command.OK) {
50     OkMessage message = new OkMessage(packet, connection);
```

```

51  message.decodeBody();
52  int bindUserNum = STATISTICS.bindUserNum.get();
53  if (message.cmd == Command.BIND.cmd) {
54      bindUserNum = STATISTICS.bindUserNum.incrementAndGet();
55  }
56  LOGGER.info("receive {}, bindUserNum={}", message, bindUserNum);
57
58  } else if (command == Command.HTTP_PROXY) {
59      ...
60  }
61  }
62  LOGGER.debug("receive package={}, chanel={}", msg, ctx.channel());
63  }

```

FastConnectMessage继承关系



m getSessionId()	int
m setRecipient(InetSocketAddress)	BaseMessage
m setPacket(Packet)	void
m setConnection(Connection)	void
m getExecutor()	ScheduledExecutorService
m runInRequestThread(Runnable)	void
m getCipher()	Cipher
m toString()	String

ByteBufMessage	
m decode(byte[])	void
m encode()	byte[]
m decode(ByteBuf)	void
m encode(ByteBuf)	void
m encodeString(ByteBuf, String)	void
m encodeByte(ByteBuf, byte)	void
m encodeInt(ByteBuf, int)	void
m encodeLong(ByteBuf, long)	void
m encodeBytes(ByteBuf, byte[])	void
m decodeString(ByteBuf)	String
m decodeBytes(ByteBuf)	byte[]
m decodeByte(ByteBuf)	byte
m decodeInt(ByteBuf)	int
m decodeLong(ByteBuf)	long

FastConnectMessage	
m decode(ByteBuf)	void
m encode(ByteBuf)	void
m toString()	String

Powered by yFiles

```

1 public final class FastConnectMessage extends ByteBufMessage {
2     public String sessionId;
3     public String deviceId;
4     public int minHeartbeat;
5     public int maxHeartbeat;
6     public FastConnectMessage(Connection connection) {
7         super(new Packet(FAST_CONNECT, genSessionId()), connection);
8     }
9     public FastConnectMessage(Packet message, Connection connection) {
10        super(message, connection);
11    }
12    @Override
13    public void decode(ByteBuf body) {
14        sessionId = decodeString(body);

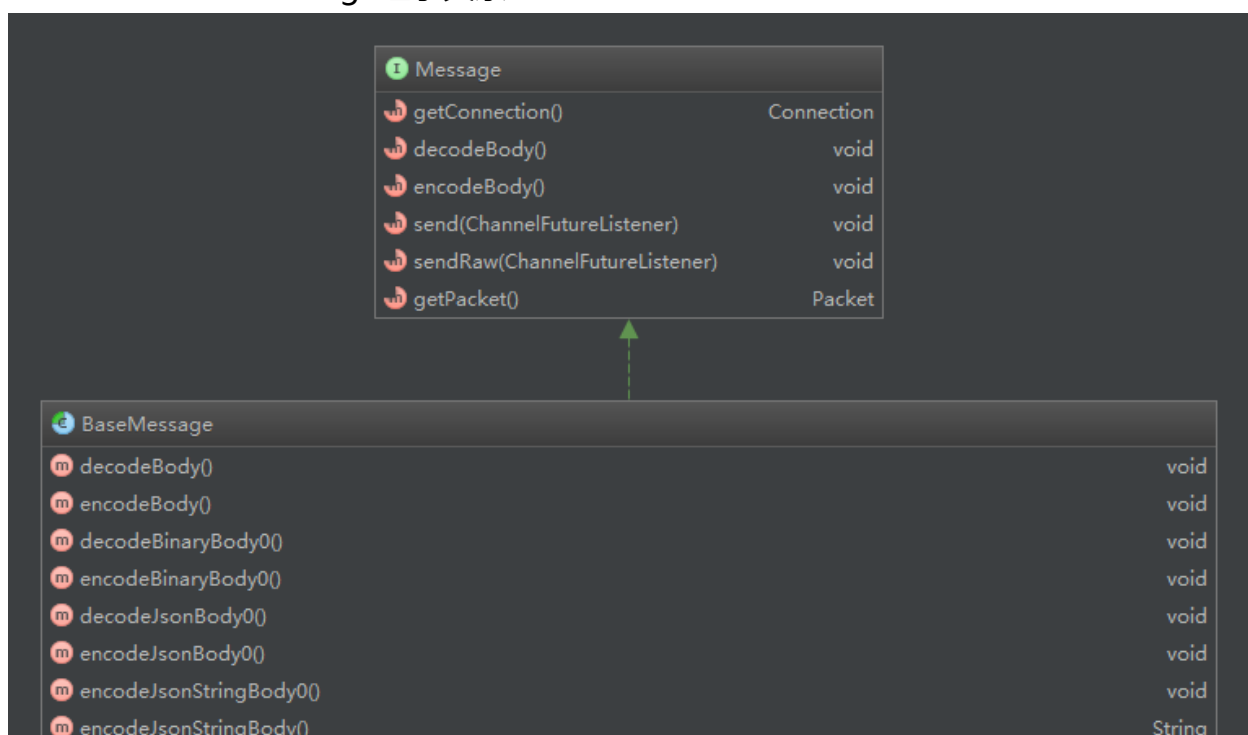
```





















```
















15  deviceId = decodeString(body);
16  minHeartbeat = decodeInt(body);
17  maxHeartbeat = decodeInt(body);
18  }
19  @Override
20  public void encode(ByteBuf body) {
21      encodeString(body, sessionId);
22      encodeString(body, deviceId);
23      encodeInt(body, minHeartbeat);
24      encodeInt(body, maxHeartbeat);
25  }
26  @Override
27  public String toString() {
28      return "FastConnectMessage{" +
29          "deviceId='" + deviceId + '\'' +
30          ", sessionId='" + sessionId + '\'' +
31          ", minHeartbeat=" + minHeartbeat +
32          ", maxHeartbeat=" + maxHeartbeat +
33          ", packet=" + packet +
34          '}';
35  }
36  }







```

FastConnectOkMessage继承关系



	encodeBodyRaw()	void
	decode(byte[])	void
	encode()	byte[]
	decodeJsonBody(Map<String, Object>)	void
	encodeJsonBody()	Map<String, Object>
	getPacket()	Packet
	getConnection()	Connection
	send(ChannelFutureListener)	void
	sendRaw(ChannelFutureListener)	void
	send()	void
	sendRaw()	void
	close()	void
	genSessionId()	int
	getSessionId()	int
	setRecipient(InetSocketAddress)	BaseMessage
	setPacket(Packet)	void
	setConnection(Connection)	void
	getExecutor()	ScheduledExecutorService
	runInRequestThread(Runnable)	void
	getCipher()	Cipher
	toString()	String

	ByteBufMessage	
	decode(byte[])	void
	encode()	byte[]
	decode(ByteBuf)	void
	encode(ByteBuf)	void
	encodeString(ByteBuf, String)	void
	encodeByte(ByteBuf, byte)	void
	encodeInt(ByteBuf, int)	void
	encodeLong(ByteBuf, long)	void
	encodeBytes(ByteBuf, byte[])	void
	decodeString(ByteBuf)	String
	decodeBytes(ByteBuf)	byte[]
	decodeByte(ByteBuf)	byte
	decodeInt(ByteBuf)	int
	decodeLong(ByteBuf)	long

	FastConnectOkMessage	
	from(BaseMessage)	FastConnectOkMessage
	decode(ByteBuf)	void
	encode(ByteBuf)	void
	setHeartbeat(int)	FastConnectOkMessage
	toString()	String