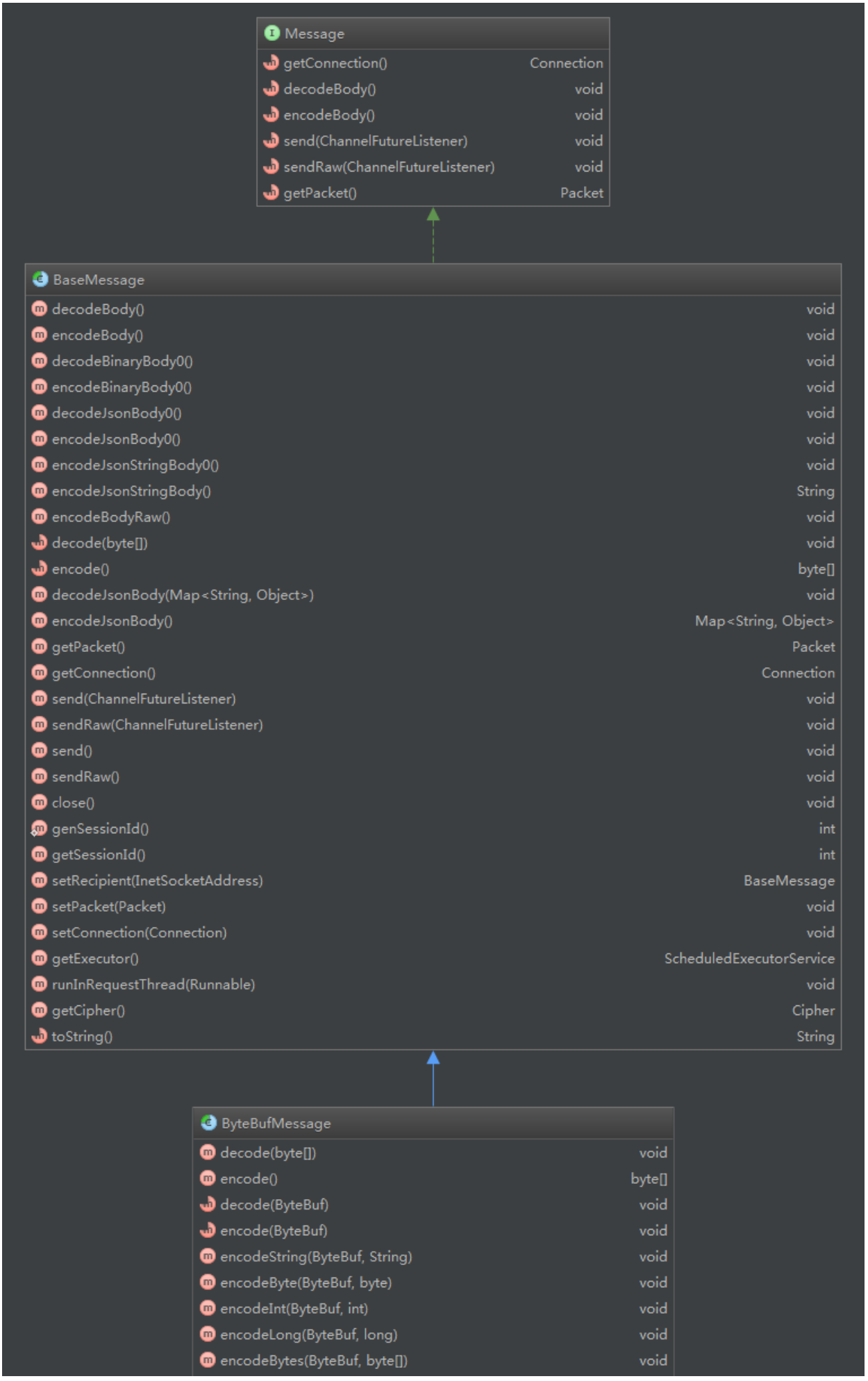
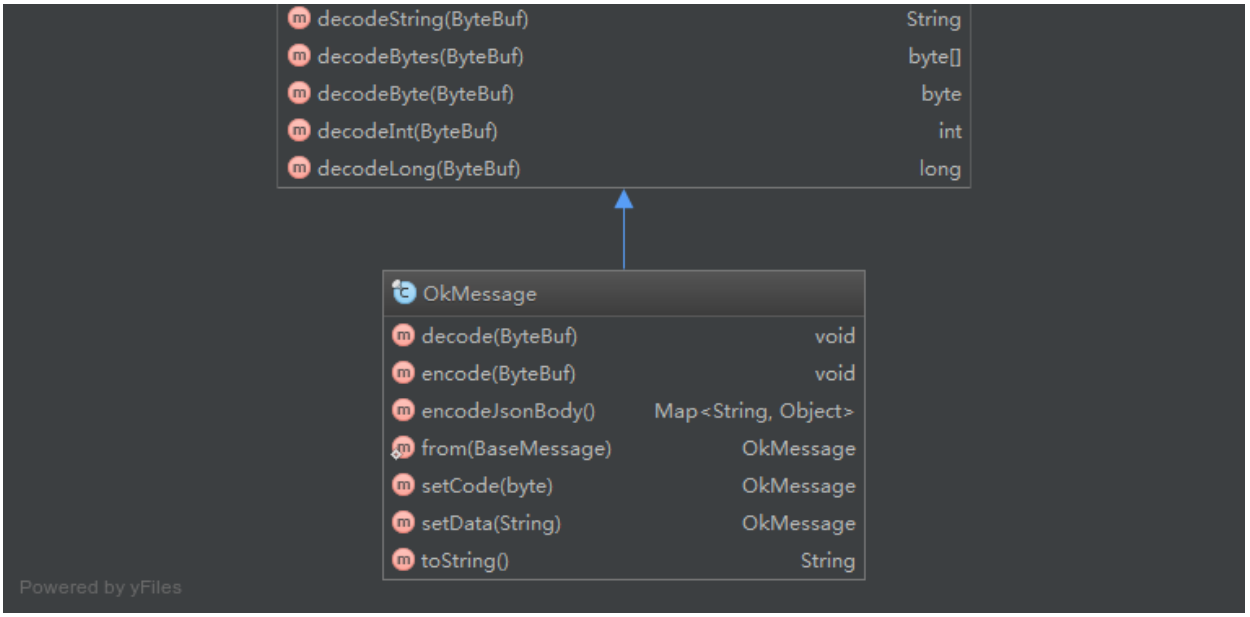
1、接收成功消息

```
//ConnClientChannelHandler.java

@Override
public void channelRead(ChannelHandlerContext ctx, Object msg) throws Exc
eption {
 connection.updateLastReadTime();
 if (msg instanceof Packet) {
 Packet packet = (Packet) msg;
 Command command = Command.toCMD(packet.cmd);
 if (command == Command.HANDSHAKE) {
  ...
  } else if (command == Command.FAST_CONNECT) {
  ...
  } else if (command == Command.KICK) {
  ...
  } else if (command == Command.ERROR) {
  ...
  } else if (command == Command.PUSH) {
  ...
  } else if (command == Command.HEARTBEAT) {
  LOGGER.info("receive heartbeat pong...");
  } else if (command == Command.OK) {
  OkMessage message = new OkMessage(packet, connection);
  message.decodeBody();
  //TODO 根据消息类型，做业务处理
  //int bindUserNum = STATISTICS.bindUserNum.get();
  //if (message.cmd == Command.BIND.cmd) {
  // //TODO 根据code，做业务处理
  // bindUserNum = STATISTICS.bindUserNum.incrementAndGet();
  //}
  } else if (command == Command.HTTP_PROXY) {
  ...
  }
  }
  LOGGER.debug("receive package={}, chanel={}", msg, ctx.channel());
}


```


Message
getConnection() Connection
decodeBody() void
encodeBody() void
send(ChannelFutureListener) void
sendRaw(ChannelFutureListener) void
getPacket() Packet
BaseMessage
decodeBody() void
encodeBody() void
decodeBinaryBody0() void
encodeBinaryBody0() void
decodeJsonBody0() void
encodeJsonBody0() void
encodeJsonStringBody0() void
encodeJsonStringBody() String
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




```java
public final class OkMessage extends ByteBufMessage {
 public byte cmd;
 public byte code;
 public String data;

 public OkMessage(byte cmd, Packet message, Connection connection) {
 super(message, connection);
 this.cmd = cmd;
 }

  public OkMessage(Packet message, Connection connection) {
  super(message, connection);
  }

  @Override
  public void decode(ByteBuf body) {
  cmd = decodeByte(body);
  code = decodeByte(body);
  data = decodeString(body);
  }

  @Override
  public void encode(ByteBuf body) {
  encodeByte(body, cmd);
  encodeByte(body, code);
  encodeString(body, data);
  }
```

```
28
29  @Override
30  public Map<String, Object> encodeJsonBody() {
31  Map<String, Object> body = new HashMap<>(3);
32  if (cmd > 0) body.put("cmd", cmd);
33  if (code > 0) body.put("code", code);
34  if (data != null) body.put("data", data);
35  return body;
36  }
37
38  public static OkMessage from(BaseMessage src) {
39  return new OkMessage(src.packet.cmd, src.packet.response(OK), src.conne
ction);
40  }
41
42  public OkMessage setCode(byte code) {
43  this.code = code;
44  return this;
45  }
46
47  public OkMessage setData(String data) {
48  this.data = data;
49  return this;
50  }
51
52  @Override
53  public String toString() {
54  return "OkMessage{" +
55  "data='" + data + '\'' +
56  "packet='" + packet + '\'' +
57  '}';
58  }
59 }
```