服务器端基本操作流程

服务器Socket对象->设置服务器IP->服务器开启端口->绑定IP和端口号->开始监听->接收或发送消息（字节数组和字符串的相互转化-Enconding.UTF8）->结束接收->关闭连接

1.创建一个服务器端Socket对象：Socket serverSocket = new Socket(…)

2.设置一个本机（解读）IP：IPAddress ipAddress = IPAddress.Parse("127.0.0.1");

3.

服务器同步、异步接收客户端连接

1.同步方式 Accept Receive

2.异步方式 BeginAccept EndAccept / BeginReceive EndReceive

客户端基本操作流程

1.创建一个客户端连接的Socket对象：Socket clientSocket = new Socket(…)

2.调用clientSocket.Connect方法

**粘包和分包**

该问题是利用Socket在TCP协议下内部的优化机制导致的

粘包指当频繁发送大量短信息时，TCP将多条消息打包进行发送，而此时服务器端Receive方法只能接收到一条信息，无法自动拆解；（还有一种情况是由接收方导致的，即放在缓冲区内的数据没能被及时取走，下一条信息直接与上一条合并放入缓冲区）

分包指一次性发送大体量信息时，数据被分成多个数据流分批发送，此时服务器会将一条消息解读成多条消息

解决方案：

对于所有发送和接收（服务器端和客户端同理）的信息使用一个Message类来处理，对消息进行组拼和解读；

由客户端发往服务器端的消息由“数据长度(int32)-RequestCode(int32)-ActionCode(int32)-真实数据”四段组成，服务器接收到一条消息后首先检查数据长度即读取前四个字节，是否与整条数据长度符合，只在符合时才拆解读取后三段数据。

注：涉及一个反射调用问题（回调？）

**服务器处理消息的流程**

在服务器端，有如下对应关系：

RequestCode <-> Controller，ActionCode <-> ClientRequest

服务器首先根据RequestCode找到对应的Controller（存在一个Controller对应多个ClientRequest的情形），然后根据ActionCode找到Controller中对应的处理方法。

服务器抽象：

1.Client – 是服务器对接的每个客户端的抽象

|  |  |  |
| --- | --- | --- |
| Types | | |
| Socket | clientSocket | 客户端Socket |
| Server | server | 服务器引用 |
| Message | msg | 消息处理工具 |
| MySqlConnection | mysqlConn(=>get) | 数据库连接对象引用 |
| User | user | 客户端在数据库中的用户表抽象 |
| History | history | 在数据库中的战绩表抽象 |

|  |  |  |
| --- | --- | --- |
| Public Function | | |
| 构造方法 | Client(Socket clientSocket, Server server) | 对引用赋值，连接数据库 |
| string | GetUserData() | 返回客户端数据，以逗号分隔 |
| void | SetUserData(User user, History history) | 设置客户端数据 |
| void | Start() | 从连接Socket异步接收数据 |
| void | Send(ActionCode actionCode, string data) | 组拼响应消息并以Socket发送 |
|  | | |
| Private Function | | |
| void | ReceiveCallback(IAsyncResult ar) | 提供给Start()的回调方法 |
| void | OnProcessMessageCallback(RequestCode requestCode, ActionCode actionCode, string data) | 提供给Message类解读信息的回调方法，转发解读数据给server |
| void | Close() | 关闭数据库连接，关闭客户端Socket，从服务器客户端列表中移除自身 |

2.Room – 服务器上对战房间的抽象

MyServer.Client中创建并初始化MySqlConnection对象，

UserDAO中提供AuthenticatingUser(MySqlConnection conn, string username, string password)方法用于验证传递过来的用户真实性，

服务器对于请求是调用Controller进行的，故在UserController中提供Login方法来处理登录请求，而Login方法则通过调用UserDAO的方法来访问数据库，

**游戏主体部分**

Manager集

Request集

Panel集

GameFacede

当每个具体的Request类需要调用MessagePanel中用于显示提示信息的方法时，

BaseRequest持有GameFacade的单例引用

单例模式

GameFacade

BaseRequest

RegisterRequest

LoginRequest

(关于UI Canvas中Rect Transform属性Anchors和Pivot：

Anchors为锚点，表示当父物体发生大小形变时当前物体的运动参考点

Pivot为支点、中心点，其位置用于自身的形变参考)

服务器处理“请求房间列表”的时候，通过server取得房间列表，通过Room中的List取得client，再通过client取得用户的名称战绩数据，之后将数据返回给客户端

客户端处理服务器响应的流程：

1.当服务器给与游戏响应的时候（发回数据包ActionCode + data），ClientManager进行监听并拥有顶层接收方法private void ReceiveCallback(IAsyncResult ar)

（但需要注意的是，回调方法不属于Unity主线程，所有该回调方法涉及的方法调用中均无法直接访问Unity中的游戏资源。）

2.该方法使用Message类处理原始消息，然后将拆分出的ActionCode和data交给回调方法

private void OnProcessMessageCallback(ActionCode actionCode, string data)

{gameFacade.HandleResponse(actionCode, data); }

将数据通过GameFacade调用requestManager的引用来处理具体数据

3.客户端的RequestManager类提供<ActionCode,BaseRequest>字典和增减方法，使每个ActionCode对应一个具体的Request

4.RequestManager中的方法被提升到GameFacade中，当其他Request被创建时，他们通过基类中的方法将自己注册到RequestManager的字典里形成映射关系

5.服务器响应客户端使用Server类中的标准方法来向客户端发送消息

public void SendResponse(Client client, ActionCode actionCode, string data)

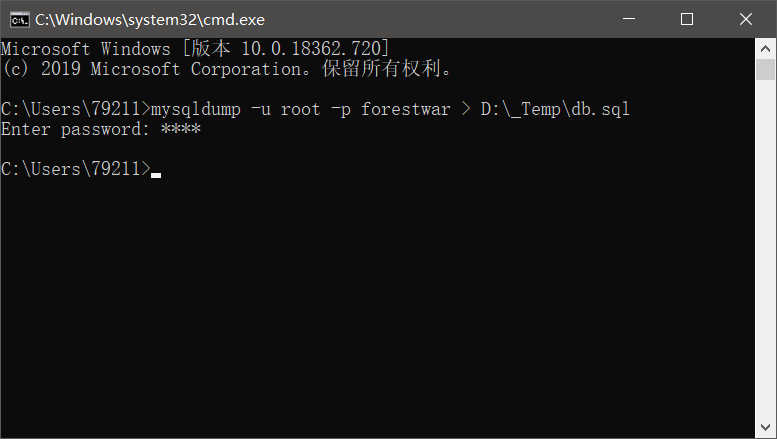
6.每个具体Request中重写OnResponse方法来处理信息

MySQL数据库备份还原

1.使用的MySQL5.7.25版本安装包所含的WorkBench8.0无法正常使用Data Export导出.sql文件备份

（备份失败并抛出错误：找不到information\_schema.column\_statistics这个元信息表。按照MySQL 8.0版本的文档提示，mysqldump加入了该参数 –column-statistics，该参数默认为打开，并且在WorkBench中不能关闭，在dump时[好像自带的mysqldump.exe也是8.0版本]，会在information\_schema.column\_statistics表中检查导出表的信息。）

经踩坑发现使用MySQL命令行调用mysqldump.exe可以导出数据库备份，但是需要在最前面手动加入创建数据库的语句（可能是导出语句的问题，没有创建语句不能在workbench中直接导入还原）。如下图所示：



2.在备份本地数据库之后，选用了远程连接数据库的方式对服务器上的数据库进行操作。

默认情况下数据库是不允许远程登录的，首先需要开启服务器上数据库（安装不同于本地）的远程访问权限。

——未知疑问：经过手写命令行的踩坑发现我们的安装包使用的其实就是新版（MySQL8.0）的语法规则。如果是导出低版本的数据库出现以上什么column报错则需要在CMD中使用 mysqldump -h172.16.0.100 -P3306 -B testdb --compact --column-statistics=0 > testdb.sql 语句来导出，但实际上只需要正常mysqldump就能导出不需要加别这个参数，说明我已经用了8.0了，并且5.x版本能在-p后直接写密码的方式在我的命令行也是无法使用的。但是为什么workbench仍然会报这个错并且没法解决呢。

由于MySQL 5和8版本的语法差异还造成了远程授权语法上的差异，8已经不能再写（网上很多老版本教程的坑）：

grant all privileges on \*.\* to 'root'@'%' identified by 'root' with grant option

语句来开启username-root password-root的远程访问，而要使用以下方法来设置远程访问用户：