# 游泳设计

**概况**：游泳系统包括游泳和跳水两部分。游泳部分里，玩家可以购买泳装和躺椅增加时间，在游泳时间内，会以间隔固定的时间给角色和精灵加经验。跳水部分里，玩家可以申请跳水，然后在指定时间跳水，并获得分数，当日分数的前5名会上游泳排行榜。

**游泳部分**：游泳部分在数据库中没有独立的数据表，内存中也没有独立的数据结构存储，所有信息均保存在属性中，所以下边就按协议来介绍。

cs\_0x0821\_start\_swim && sc\_0x0821\_start\_swim

当玩家进入游泳地图时（包括当日第一次进入和中途离开后再次进入），前端会发此协议，

首先判断是否有泳衣，如果没有泳衣就报错。然后获取玩家的剩余游泳时间。

剩余游泳时间包括两部分，常规时间和躺椅增加时间。玩家常规游泳时间根据是否买高级泳装和是否为VIP而不同，但是是一个固定值，不可延长。躺椅增加时间（extended\_left）是玩家购买延时躺椅后增加的时间，玩家一天可以多次购买延时躺椅。

如果玩家当日首次游泳。则初始化常规游泳时间，并加上躺椅时间。

如果玩家非当日首次游泳。则读属性直接获取剩余时间。

**其中，躺椅增加的时间，在购买配表时已经配上，所以代码中无需再次初始化。**

获取了剩余时间后，若剩余时间为0就报错。

若不为0，就把当前状态设置正在游泳的状态，置游泳时间戳，拉跳水排行榜信息，然后回包。

cs\_0x0822\_get\_exp && sc\_0x0822\_get\_exp

先做用户的合法性验证，例如没有剩余时间，游泳状态是未开始，两次获得经验时间过短，都会报错。

验证完合法性之后，首先扣除一定的游泳时间，然后分别给人物和精灵加经验。

**注意：在给客户端回包时，既需要角色和人物本次获得的经验，也需要他们累计获得的经验。当前角色和精灵累计获得的经验分别存在两个属性中，角色累计经验仅在出地图时清0，精灵经验除了在出地图清0外，如果更换了精灵，也会清0.**

如果时间用完，就把游泳状态设置为不在游泳，并用**onlineproto::sc\_0x0826\_inform\_end\_swim inform\_end\_swim** 协议通知玩家

cs\_0x0823\_swim\_pause && sc\_0x0823\_swim\_pause

玩家离开游泳地图时，前端会发这个协议这时会把精灵和角色的累计经验清0，并且把玩家状态变为不在游泳的状态。如果玩家此时还在跳水队列中，就移出队列，并且把跳水状态改为没有开始。

cs\_0x0825\_inform\_start\_swim && sc\_0x0825\_inform\_start\_swim、

暂未用上

cs\_0x0826\_inform\_end\_swim && sc\_0x0826\_inform\_end\_swim

用来通知玩家游泳结束，0x0822协议中用到

cs\_0x0827\_use\_chair && sc\_0x0827\_use\_chair

cs\_0x0828\_withdraw\_chair && sc\_0x0828\_withdraw\_chair

**使用和收回躺椅，主要用于界面中展示的动作，已写完，但是由于前端没有资源，还没有联调**

cs\_0x0829\_inform\_pause\_swim && sc\_0x0829\_inform\_pause\_swim

通知游泳暂停。 在0x0823协议中用到

**跳水部分**：

本部分有两个结构体和两个类。

struct diver\_t 是存储跳水玩家的跳水时间信息的结构， class Dive是管理跳水玩家的类，包含一个std::list<diver\_t> diver\_list;的成员变量和对这个成员变量操作的成员函数。

struct diver\_score\_info\_t 是存储玩家跳水玩家跳水得分信息的结构，class DiveRank是管理跳水玩家分数的类，维持了一个仅有5个名次的排行榜。包含一个std::vector<diver\_score\_info\_t> dive\_rank;成员变量和对这个成员变量操作的成员函数。

跳水流程比较简单，就不按协议写了。

首先玩家申请跳水，就将玩家加入到跳水队列中。**跳水列列的规则是，如果没有人在队列里，那么加入队列的玩家的跳水时间就是当前时间。如果有人在队列中，那么加入队列的玩家的跳水时间就是最后一个跳水玩家的时间再加10秒。玩家跳完水或者离开地图，就移出队列。**

玩家跳完水后，会有一个分数，一旦当日累计分数可以进入排行榜，就会将该玩家加入到排行榜中，并剔除排行榜的最后一名。