# 面向对象程序设计

### ——————————宠物小精灵对战系统

## 一、设计任务的描述

**设计任务一：**

设计宠物小精灵类，并实现宠物小精灵的加入，宠物小精灵的攻击、升级、战斗等等。

**设计任务二：**

用户注册与平台登录，并用socket通信---tcp来进行用户信息的传输，实现注册、登录等功能，并实现界面交互以及精灵分配、查看其他用户情况等功能

**设计任务三：**

游戏对战的设计，并设计升级赛和决斗赛，并设计战斗的具体细节（暴击、闪避等），查看用户的徽章、胜率等功能

## 功能需求说明及分析

1. **设计宠物小精灵的类**

·包含不同的精灵类型 ----力量型，防御型、肉盾型、敏捷型

·包含9种不同的精灵属性 ----互相有克制关系，如下图

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 属性 | 火属性 | 水属性 | 草属性 | 暗属性 | 光属性 | 风属性 | 地属性 | 电属性 | 机械属性 |
| 火属性 | 1 | 0.5 | 2 | 2 | 0.5 | 1 | 0.5 | 1 | 2 |
| 水属性 | 2 | 1 | 0.5 | 1 | 1 | 1 | 1 | 1 | 2 |
| 草属性 | 0.5 | 2 | 1 | 1 | 1 | 1 | 2 | 1 | 0.5 |
| 暗属性 | 0.5 | 1 | 1 | 1 | 2 | 1 | 1 | 0.5 | 0.5 |
| 光属性 | 0.5 | 1 | 0.5 | 2 | 1 | 1 | 1 | 0.5 | 1 |
| 风属性 | 0.5 | 1 | 2 | 1 | 1 | 1 | 0.5 | 1 | 1 |
| 地属性 | 2 | 1 | 0.5 | 1 | 1 | 1 | 1 | 1 | 1 |
| 电属性 | 1 | 0.5 | 1 | 2 | 0.5 | 1 | 0.5 | 1 | 2 |
| 机械属性 | 0.5 | 0.5 | 2 | 1 | 1 | 1 | 2 | 0.5 | 1 |

**根据以上共组合出截然不同的31种小精灵**

·包含名字、等级、经验值、攻击力、防御力、生命值等基本属性

**----**并设计了相关的成长属性，对于不同的属性、类型的精灵有不同的属性增幅

·包含1-15级的升级系统，每级升级经验为level\*100 + 2^(level-1)

**----上述升级经验值的设计可以保证升级的难度越来越大，经过测试为非线性的递增方式，符合实际情况**

·每个精灵有自己独特的攻击方式

----根据其继承的基类（类型，属性）衍生出不同的属性成长或是攻击方式，例如火属性精灵的**火焰攻击经过重写后会调整为1.2倍的攻击伤害**，水属性精灵的**水属性攻击会无视对方一定的防御属性**等等，**该模块通过虚函数的重载实现**。

**2.实现c/s模式的用户交互**

·用户注册与登录，并实现登录、登出等功能

·将用户的数据保存在服务器

·为新注册的用户分配初始精灵

·查询精灵、用户情况

----上述功能由socket通信实现，通过设计一系列的报文，来传递请求类型

----**用户未登录之前，本地无任何数据，需要读取服务器内的数据来为用户加载自身信息**

|  |  |
| --- | --- |
| 请求类型（首位） | |
| 登录 | 0 |
| 注册 | 1 |
| 登出 | 2 |
| 请求背包 | 3 |
| 加载背包 | 4 |
| 请求用户列表 | 5 |
| 请求加载用户列表 | 6 |
| 升级赛 | 7 |
| 决斗赛 | 8 |
| 加载用户 | 9 |
| 加载精灵 | 10 |
| 获得新精灵 | 11 |
| 失去精灵 | 12 |

----**通过分析接收到的不同的类型与数据库进行交互，操作数据库，来实现数据的存储、更新等功能**

**3.战斗功能的加入**

·升级赛、决斗赛、精灵失去与获得

----战斗在本地进行，战斗结束后，根据结果向服务器发送报文对服务器内的数据进行更新，并在本地进行相应的更新

·模拟战斗

----在精灵的战斗函数中加入了**暴击与闪避的判断**，实现了随机性的增加

·查询用户信息、自身徽章信息

----查询其他用户的功能主要通过向服务器发送请求报文，读取服务器内的数据来获取其他用户的信息；自身的徽章信息则是直接查询本地的信息（本地的信息实际上也是来自于之前的服务器的）来读取。

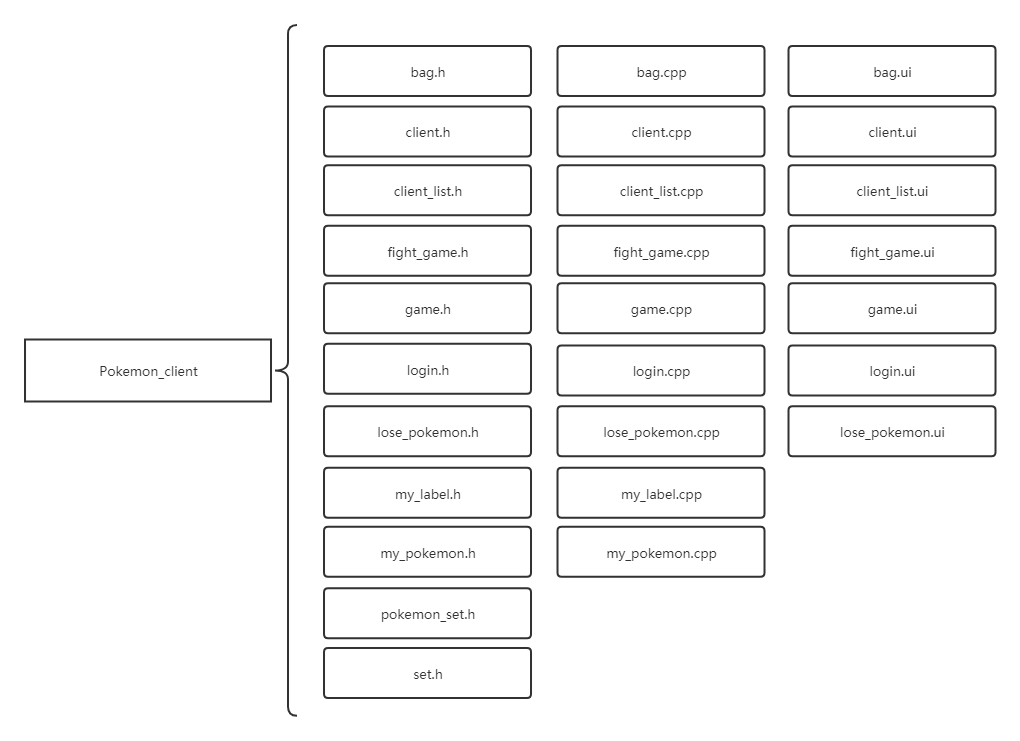
## 总体设计方案及说明

1. **软件开发环境**

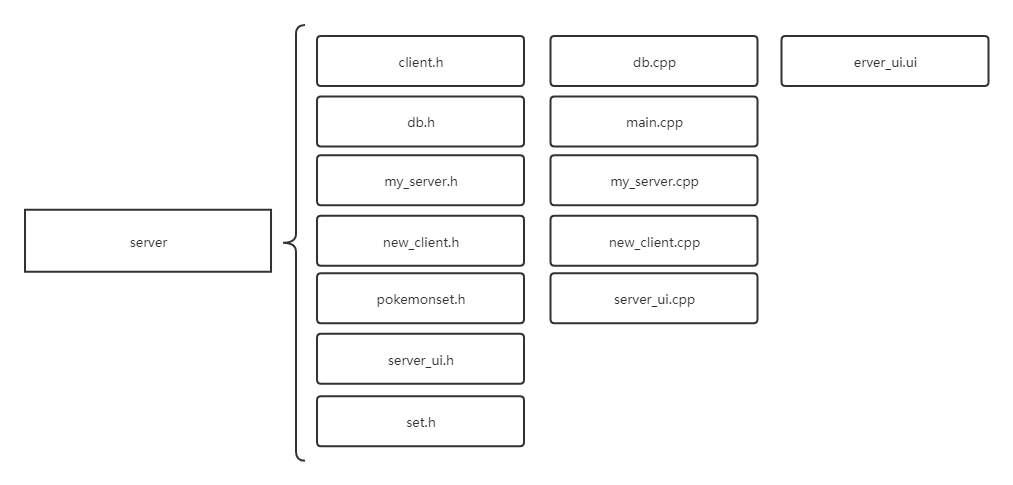
Qt5.12.9、windows 10、vs studio 2019

1. **总体结构**

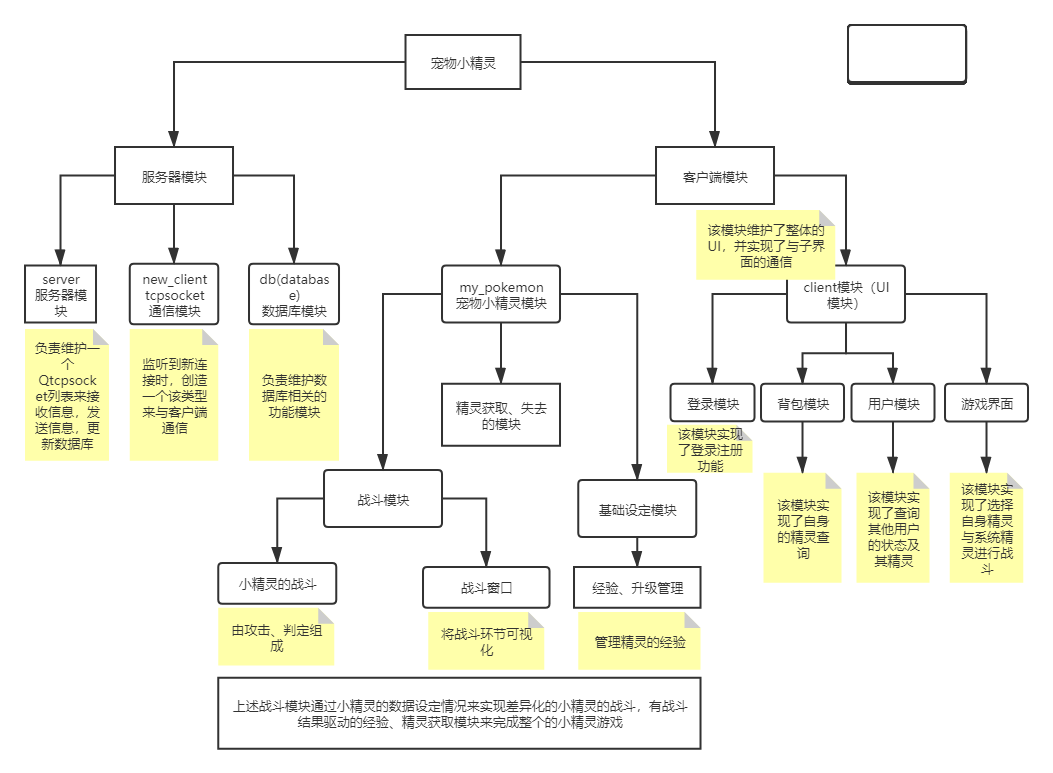
客户端部分：pokemon\_client.pro



服务器部分：server.pro



1. **模块划分**

****

## 数据结构说明

1. **精灵类 pokemon**

char m\_name[20]; //名字长为20

int m\_type; //0代表力量型高攻击，1代表肉盾型高生命值，2代表防御性高防御，3代表敏捷型低攻击间隔

int m\_attack; //攻击力 力量 > 防御 = 肉盾 > 敏捷

int m\_defense; //防御力 防御 > 肉盾 > 力量 > 敏捷

int m\_HP; //生命值 肉盾 > 防御 = 力量 > 敏捷

int m\_speed; //速度 敏捷型 > 攻击型 = 防御型 > 肉盾型

int m\_property; //属性

int m\_growing\_att; //成长对应不同精灵 45 50 55

int m\_growing\_def; //成长对应不同精灵 20 25 30 35

int m\_growing\_HP; //成长对应不同精灵 80 100 120

int m\_growing\_speed; //成长对应不同精灵 15 12 8

double m\_dodge; //闪避几率 0.06 0.03 0.01

double m\_crit; //暴击几率 0.2 0.1 0.05

int type\_id; //精灵id

int pokemon\_id; //全局唯一精灵id

QPixmap pokemon\_icon; //精灵图标

int m\_level; //等级 最大上限15级

int m\_exp; //经验值 每级上限是level\*100 + 2^(level-1) 经验获取是level^2 + 2^等级差（奖励越级挑战） \* （16 - 等级）\*10

1. **用户类client**

login \*my\_login; //登录窗口

bag \*my\_bag; //背包界面

client\_list \*my\_client\_list; //用户列表界面

game\* my\_game; //游戏界面

int win = 0; //胜场

int lose = 0; //负场

int my\_pokemon\_number = 0; //我当前的精灵数

int my\_max\_number = 0； //我满级的精灵数

bool online = false; //判断是否在线

int port; //端口号

QHostAddress \*serverIP; //ip

QString username; //用户名

QString password; //密码

QTcpSocket \*tcpsocket; //socket通信

QList<pokemon\*> pokemon\_list; //精灵列表

QStringList pokemon\_Slist; //名称列表

int user\_id; //用户id

1. **游戏类 game**

int fight\_type; //战斗类型

QList<pokemon\*> temp; //新生成的精灵列表

QList<pokemon\*> my\_pokemon; //我的精灵的列表

1. **服务器类 my\_server**

SqliteDBAOperator\* my\_db; //指向数据库的指针

QList<new\_client\*> client\_list; //用户列表

int pokemon\_number; //服务器内精灵总数

1. **服务端client类**

int user\_id; //用户id

QString Pokemon\_id; //所拥有的精灵id

QString USER\_NAME; //用户名

QString PASSWORD; //密码

## 各模块设计说明

**一、服务器部分**

**①server模块：该模块作为整个服务器的核心模块有以下功能**

1. 与database模块进行通信，来实现数据库的增删改查功能
2. 维护一个client列表，并从client（服务端）子模块接收消息，并进行回复

**②database模块：该模块负责维护数据库**

1. 与server模块进行通信，来对数据库进行增删改查、

**③client（服务端）：该模块使用Qtcpsocket，与客户端通信，与server传递信息**

1. 从客户端接收信息，并将信息传递给server端进行判断
2. 从server端接收回复，并将信息传递给客户端
3. **客户端部分**

**①宠物小精灵模块：该模块实现了宠物小精灵类的设计与功能实现**

1. 实现了小精灵类，并根据属性、类型细化出31种不同的小精灵
2. 实现了小精灵的战斗功能，战斗功能下辖攻击、闪避暴击判断等功能，来实现自动化的但又有随机性的战斗功能
3. 实现了部分精灵的攻击函数重写，来进一步达到差异化处理小精灵的目的
4. 实现了战斗场景的ui，将战斗中的细节可视化

**②ui模块：该模块实现了客户端的整体ui**

1. **背包模块：**该模块通过ui设计使得用户查询自己的精灵属性部分可视化，能够直观的了解自身的精灵情况。该模块通过发送查询背包的报文，向服务器发送加载请求来达到数据读取的目的
2. **用户列表模块：**该模块通过ui设计使得用户可以可视化的查询其他用户的在线情况、拥有的精灵情况、胜率等功能。该模块通过发送查询用户列表类型的报文，从服务器加载信息
3. **游戏模块：**该模块通过ui设计使得用户可以以图形化的界面来选择自身的精灵与系统随机生成的精灵进行战斗。在升级赛中，可以自由选择对手来增加经验值，根据结果向服务器发送报文更新服务器内用户的信息，同时也在本地更新自身的信息；在决斗赛中，通过返回结果、对手、自身id等信息向服务器发送更新报文，也会在本地更新自己的信息。
4. **徽章读取模块：**该模块通过ui设计使得用户可以在自身客户端界面选择是否显示自身的徽章情况

**③socket通信模块：**该模块实现了报文的写作和发送，以及接收服务器发送的回复来操作本地的数据的功能

1. 实现了根据ui界面的操作信息向服务器发送信息
2. 实现了根据从服务端接收的信息向ui界面，以及用户本地的数据进行操作的功能
3. 实现了socket中tcp的可靠信息传输

**④登录模块：**该模块通过ui模块和socket模块实现了登录、注册功能

1. 实现了登录功能，并且通过server端的数据库修改，实现了已登录的用户不能重复登陆的功能，用户名全局唯一。
2. 实现了注册功能，通过验证服务端是否存在用户名，来确定是否可用。可用则会发放用户id，并通过宠物小精灵模块随机发放三个初始小精灵。

## 其他说明

1. 本项目中不包含任何友元函数
2. 所有函数均为类内函数