MipaProject　程序设计

一、模块划分与模块功能说明

根据程序的需求，我们可以把程序划分为以下多个功能完全不同的模块：

UI:负责整体调配各个模块，综合用户操作，绘制工作以及游戏逻辑计算等功能。

Paint:负责 OpenGL 相关绘制工作，从外部导入三维模型，在窗口中绘制它们，并且根据需要进行其它相关控制。

Sprite:负责动画计算。这里的动画指的是静态模型可能进行的一系列平移与旋转操作，在 Sprite 类里统一计算位移的量。

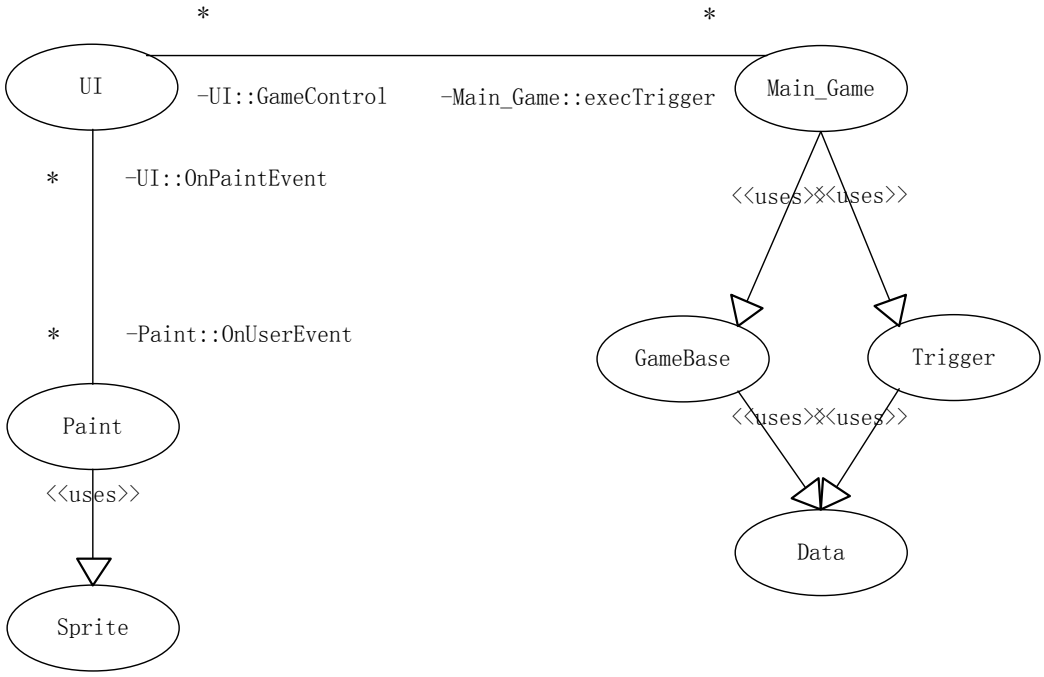
Trigger:负责游戏逻辑计算。所有的游戏操作被描述为触发器式的方式进行计算，并且在这里计算结果。

Main_Game:游戏数据保存和计算的核心。为 UI 与 Trigger 系列的接口。

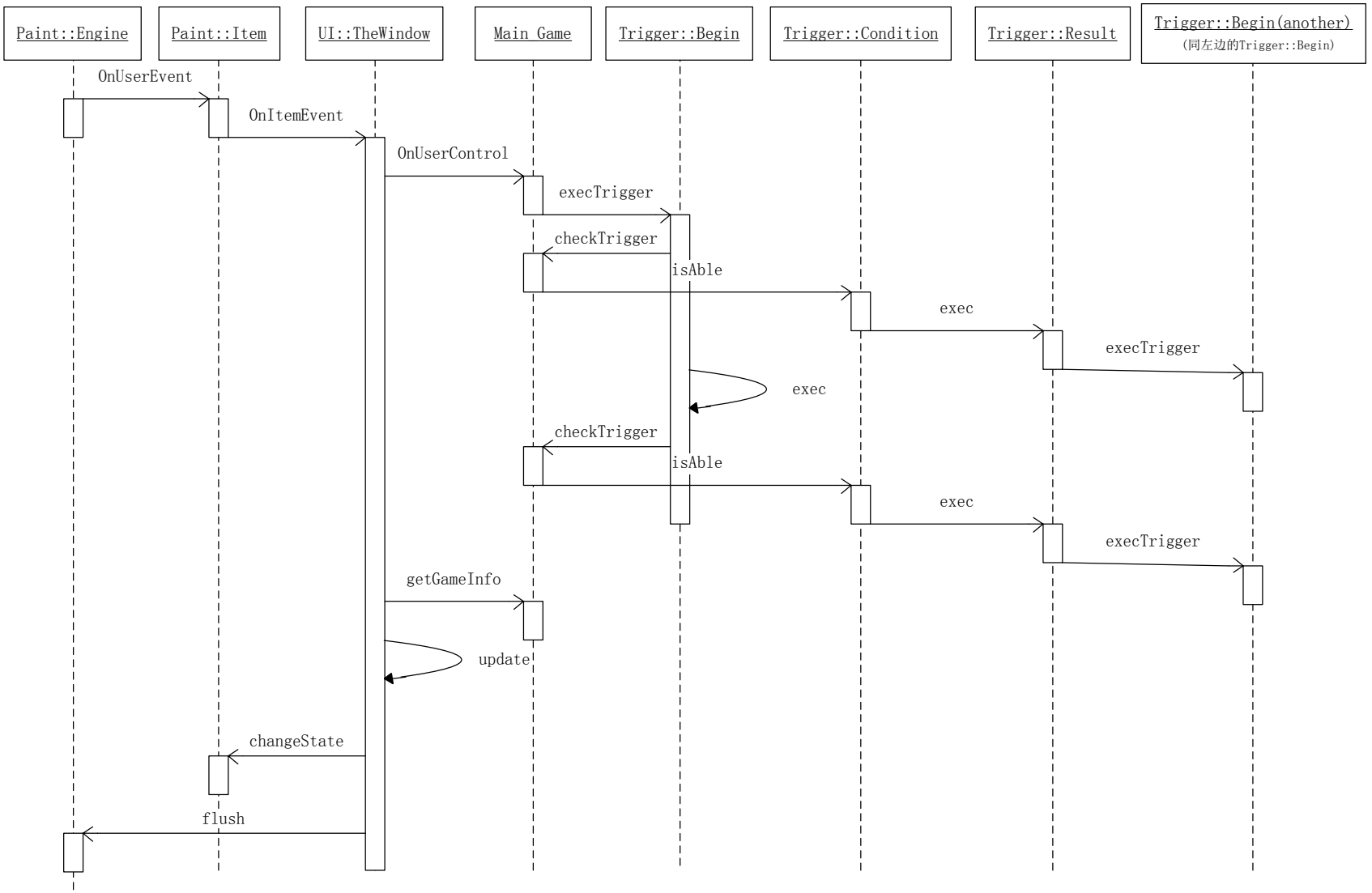
GameBase:游戏数据的保存时所使用的类型。

Data:保存数据的游戏内数据库。因为无法使用脚本化的语言保存为文本文件，所以选择将游戏数据内建在程序内。

各个模块之间的关系如图所示



# 整体时序图


Paint::Engine
Paint::Item
UI::TheWindow
Main Game
Trigger::Begin
Trigger::Condition
Trigger::Result
Trigger::Begin(another)
(同左边的Trigger::Begin)
OnUserEvent
OnItemEvent
OnUserControl
execTrigger
checkTrigger
isAble
exec
execTrigger
exec
checkTrigger
isAble
exec
execTrigger
getGameInfo
update
changeState
flush

二、分模块设计说明

## 1 Main_Game 设计说明

### 1.1 程序描述

此类为用户界面与游戏数据之间的接口，隔离游戏数据与外部程序，在内部执行所有的游戏逻辑计算。

### 1.2 功能

完成两种主要功能：

Trigger 逻辑计算：任意触发器触发->完成触发器连锁计算并且更新游戏数据。（无输出）

得到游戏数据：其它模块调用时，返回 Main_Game 保存的游戏数据。

### 1.3 性能

所有函数应当快速返回，不能发生阻塞或者死循环。

### 1.4 输人项

获取游戏数据时：输入所需的玩家编号。

Trigger 逻辑计算时：给予一个以 Trigger_Begin_Base 为基的子类，指明需要什么数据；给予一个 Trigger_Set_InputInfo 类，包含全部输入的数据。

### 1.5 输出项

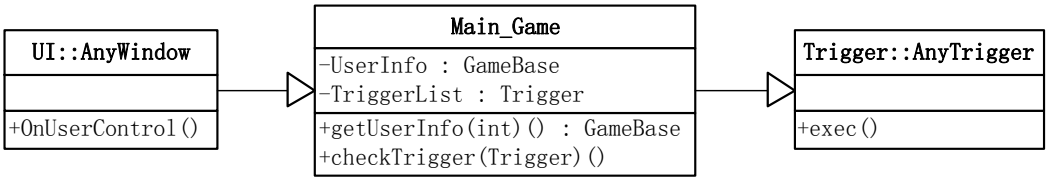
获取游戏数据时：直接获得以 GameBase_UserInfo 类保存的某个玩家的全部游戏数据。

Trigger 逻辑计算时：没有输出，但是内部数据会自动更新。

### 1.6 流程逻辑

见整体流程图。

### 1.7 接口




### 1.8 限制条件

Main_Game 采用单件模式。

### 1.9 尚未解决的问题

Main_Game 的接口都是根据上下层模块的需求而设计的。所以暂时无法说明有什么缺失，只能等待其它类的完善。

## 2 UI 模块设计说明

UI 模块主要是为了设计的游戏进行界面设计与绘制。包含游戏登录界面，游戏准备界面，卡组编辑界面，游戏主界面的设计。游戏登录界面需要实现玩家登录功能，游戏准备界面可以选择对战玩家与选择卡组，卡组编辑界面实现卡组的选取与组合功能，游戏主界面实现游戏对战功能。

### 2.1 程序描述

程序的目的是实现游戏界面，使玩家可以通过游戏界面进入游戏，然后与 UI 进行交互以进行游戏。

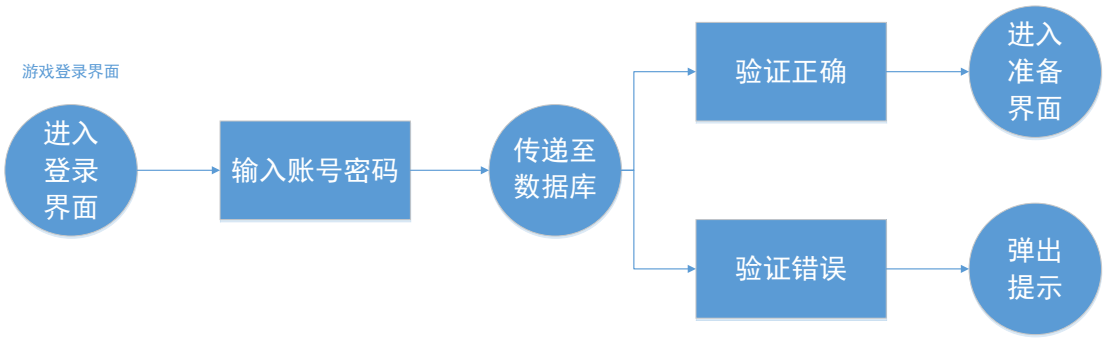
### 2.2 功能

第一部分：游戏登录界面。

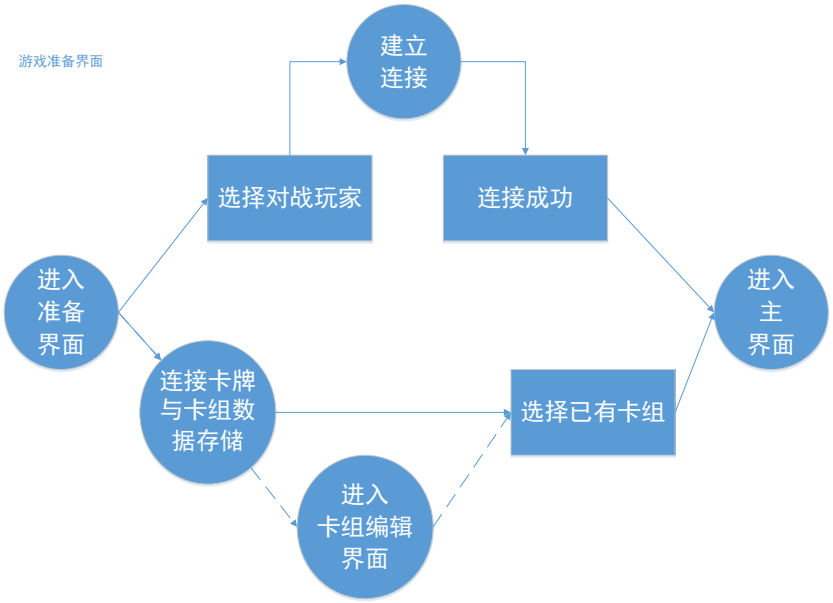
输入：账号密码、验证状态

输出：账号密码、当前程序运行阶段

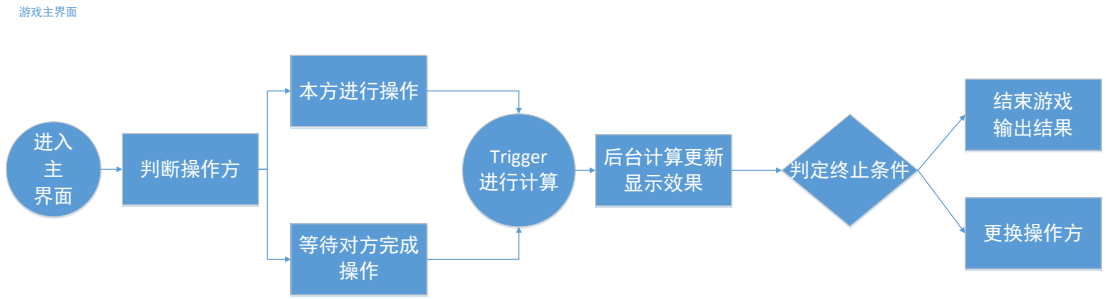
接收账号密码输入，传递给数据库，接收判断结果，决定是否进入下一界面。




第二部分：准备界面。
输入：对手 IP、连接状态、选择卡组编号
输出：对手 IP、选择卡组编号
　　接收对手 IP 输入，更新连接状态是否成功。确定选择卡组编号。传递给后台后进入主界面。




第三部分：主界面。
输入：场上卡牌和随从编号、场上卡牌和随从状态、操作方、游戏终止状态
输出：场上卡牌和随从编号
判断操作方，允许一方进行操作。
接收后台计算的结果，实时进行更新。
响应游戏结束的信号，结束游戏。



## 2.3 性能

UI 的性能要求有一定的鲁棒性，不会因为误操作而导致程序崩溃。时间上响应应在 1 秒以内即可。

## 2.4 输入项

输入项：
登录界面：账号密码为字符串，接收数据库验证结果为整型数
准备界面：对手 IP 为字符串、接收连接状态结果为整型数、卡组编号为整型数
主界面：输入为选择的当前场上的卡牌或者随从的编号为整型数、卡牌随从的状态为整型数和字符串，操作方、终止状态为整型数。

输入：账号密码、验证状态
输出：当前程序运行阶段
输入：对手 IP、连接状态、选择卡组编号
输出：对手 IP、选择卡组编号
输入：场上卡牌和随从编号、场上卡牌和随从状态、操作方、游戏终止状态
输出：场上卡牌和随从编号、

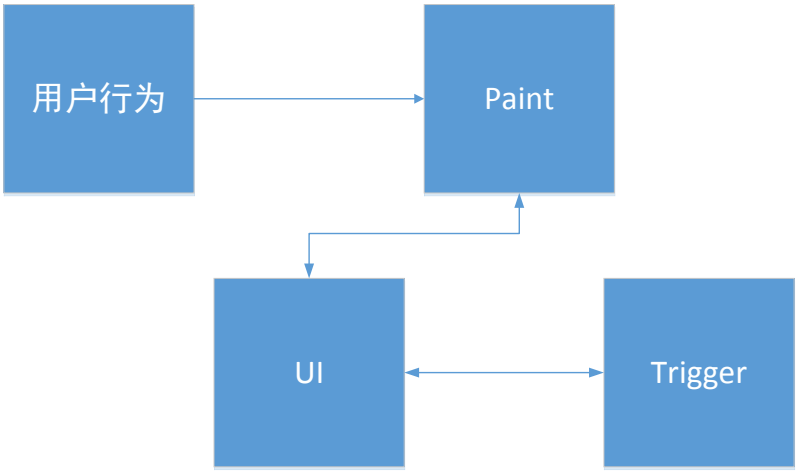
## 2.5 输出项

登录界面：账号密码为字符串，当前程序运行阶段为整型数
准备界面：对手 IP 为字符串、卡组编号为整型数
卡牌编辑界面：输出为选择的一系列卡牌编号
主界面：输入为选择的当前场上的卡牌或者随从的编号为整型数

## 2.6 流程逻辑

本程序在整体软件中主要跟逻辑判断程序 Trigger 等模块进行交互响应。

用户行为通过 Paint 模块进行采集，Paint 模块将信息反馈给 UI。UI 再将行为传递给 Trigger 进行事件的逻辑判定以及信息计算更新等等，从 Trigger 得到反馈。UI 将更新结果反馈给 Paint，Paint 使用更新后的信息重新绘制。




## 2.7 接口

UI 主要进行接口的调用，不另外提供接口给其他部分的程序。

## 2.8 限制条件

程序运行时，需要调用 paint 模块实现的元件。同时受 trigger 模块的控制，trigger 模块由用户的动作触发，从而控制 UI 模块发生变化。

## 2.9 尚未解决的问题

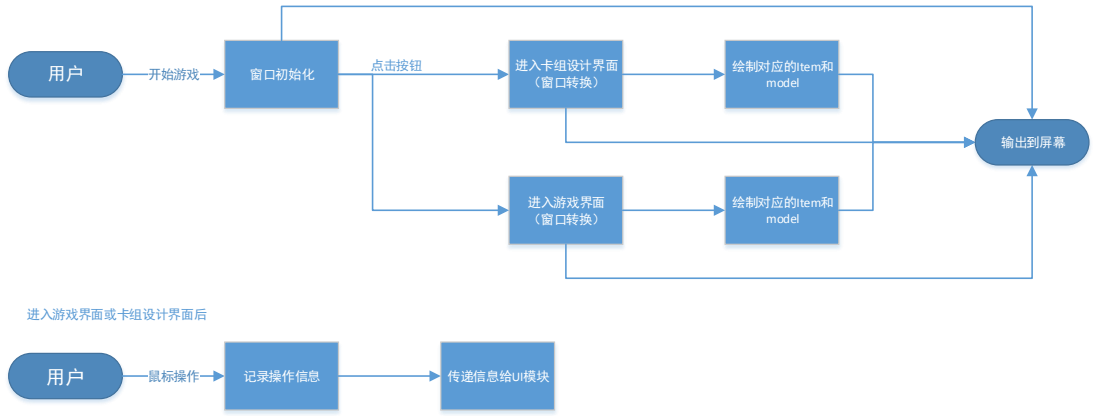
目前我们实现了原型界面的绘制。原型界面直接调用了 QT 里面现有的原件，实现了游戏界面的简化版。未来需要实现接入 paint 元件和实现与 trigger 模块的接口等。

# 3 Paint 模块设计说明

## 3.1 程序描述

负责使用 OpenGL 进行游戏界面的绘制，模型的导入，基本单元的绘制以及界面的刷新，还有用户控制信息的读取。

## 3.2 功能




## 3.3 性能

由于 Paint 模块需要针对用户的操作（鼠标拖动卡片等）及时的更新界面，因此它需要精确的知道鼠标的点击位置，位移向量等，同时及时性要求该模块会以某一固定时间间隔更新界面。

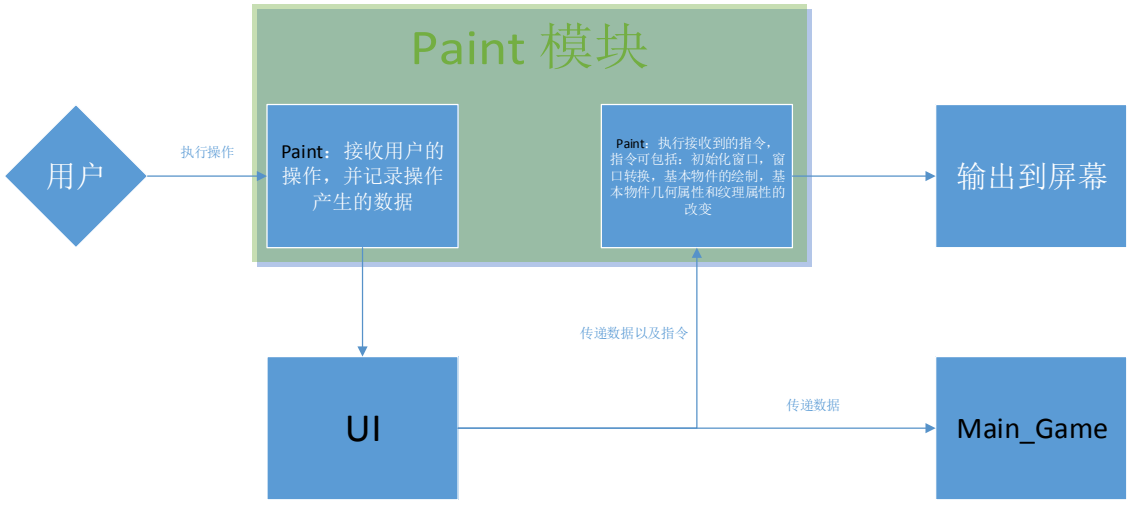
## 3.4 输入项

绘制的物品的位置和纹理信息，鼠标的坐标和位移。数据的类型皆为 float 型向量。

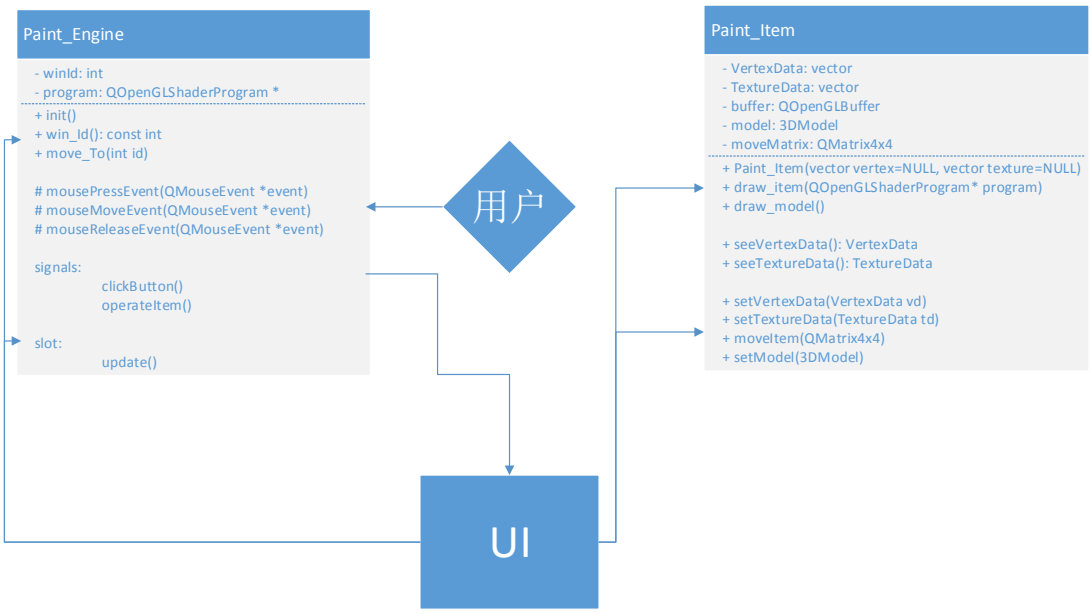
## 3.5 输出项

无其他输出，在界面中绘制输入的物体，或者更新已绘制物体的位置、形状等。

## 3.6 流程逻辑




## 3.7 接口




**3.8 限制条件**

直接受 UI 模块的控制，在没有收到 UI 模块的信息前不能擅自行动

**3.9 尚未解决的问题**

3D 模型的导入，判断鼠标点击的是哪个 3 维物体，摄像机变换。

另外，与 paint 模型动画相关的 Sprite 类，尚未具体设定。

## 4 Trigger 模块设计说明

**4.1 程序描述**

程序设计的主要目的是为了解决整个游戏的逻辑计算，所有的游戏功能被描述成触发器的方式进行计算，并得到相应结果后传递给 main_game。

**4.2 功能**

程序实现的功能主要是通过不同触发器之间的相互调用，完成游戏的基本计算部分。程序是通过前触发判定——触发器执行内容——后触发判定这样的一个逻辑设计的。（可能需要画成 IPO 图）

**4.3 性能**

性能要求暂无。

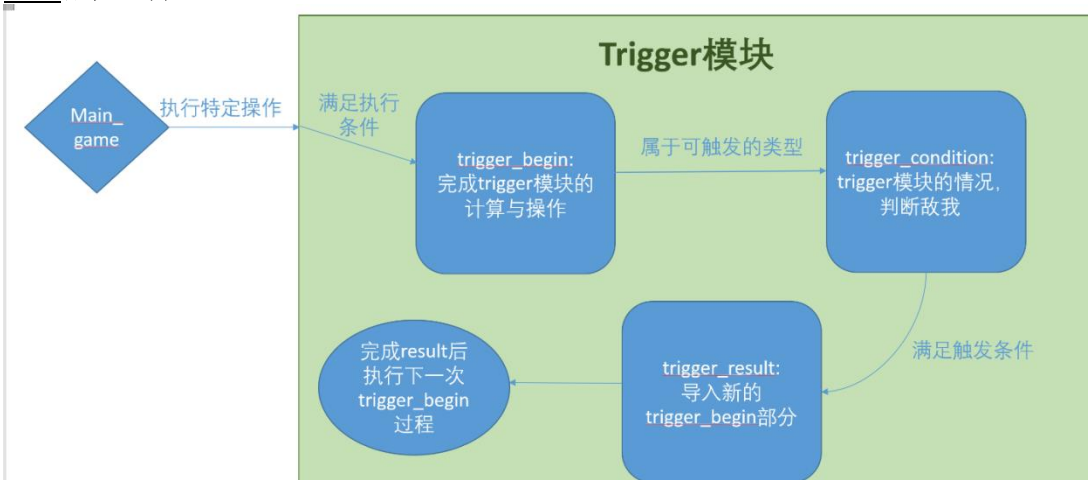
**4.4 输入项**

所有的输入项都是 Trigger_Set_inputinfo。

**4.5 输出项**

Trigger 部分完成逻辑计算后会将计算结果广播给 main_game。

**4.6 流程逻辑**

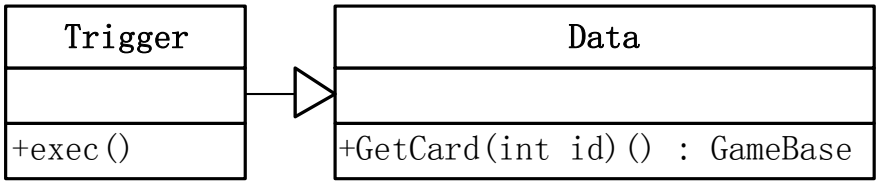


### 4.7 接口
见整体流程图。
### 4.8 限制条件
Trigger 部分的运行受 main_game 限制，必须由 main_game 启动。
### 4.9 尚未解决的问题
程序暂时没有完成卡牌的 trigger 在调用后需要移除的过程，以及将卡牌本身的 trigger 添加至 main_game 的过程，这些会在未来的设计中完成的。此外，暂时没有编写法术卡的部分，这会随着程序开发的逐渐推进而加入相应的功能。

## 5 GameBase 设计说明
### 5.1 程序描述
GameBase 相关的类负责以结构体的形式，存储游戏内保存的全部数据。
### 5.2 功能
纯数据存储类，没有功能。
存储的数据相见各个子类。
### 5.3 尚未解决的问题
有的数据的存储方式没有确定，主要是各种卡片所受到的各种不同持续性的效果的保存，例如生命力永久增加，攻击力永久降低，被沉默从而失去所有效果等。

## 6 Data 设计说明
### 6.1 程序描述
因为本程序的特点：不需要灵活地修改数据，希望所有用户拥有一致的数据；解析脚本性的语言比较复杂，所以对于每个卡片的触发器也只能编入程序中来描述。所以在程序中使用 Data 类来保存所有的游戏数据。
### 6.2 功能
说明该程序应具有的功能，可采用 IPO 图（即输入一处理一输出图）的形式。
### 6.3 性能
立刻返回。因为是 O(1)级的分支结构，所以一般也不会出现阻塞。
### 6.4 输人项
只需要输入需要获得的卡片的 id 编号。
### 6.5 输出项
返回一个含有这个卡片的全部数据的 GameBase_Card 类。
### 6.6 流程逻辑


**6.7 接口**




**6.8 限制条件**

输入的数值应当在已有的卡片的范围内。

**6.9 尚未解决的问题**

暂无