20231104

游戏核心玩法v1.0

单人游戏-键鼠操作。玩家可以控制一个玩家飞船。键盘负责平移，鼠标负责方向。

有血量系统。

两种攻击模式-左键单击可以发射机炮（有射速上限），伤害较低，无穿透效果；长按鼠标左键可以蓄力进行磁轨炮攻击，伤害较高，有穿透效果。（相当于要实现判定线）

可以考虑两种敌人：

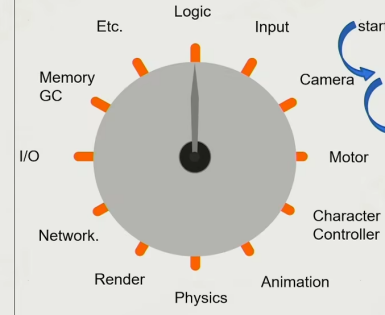
敌方普通飞船：数量多，只会普攻，速度较慢，比较迟钝。

敌方英雄-和玩家飞船属性完全一致。和玩家1v1（要考虑如何实现ai功能）

（是否可以加入子弹能在战场边界反弹n次的设定？是否在战场中加入一些障碍？是否考虑比屏幕大的地图？）

（结算系统（分数，kd，胜率。。。。。。一定要实现的）；成就系统）

20231105

组件-游戏对象-场景的游戏组织架构。

不同的组件（比如生命组件，位置组件，运动组件）构成一个游戏对象（或者精灵）。

许多游戏对象构成一个场景。

游戏通过帧的更新来完成动作。每一帧包含场景中所有游戏对象的不同状态截面，通过更新来处理状态的变化，实现状态的连贯。所以每个游戏组件都需要一个update（或者tick）方法来更新自己的状态。Update在本游戏中可以分为：逻辑处理和渲染两个部分。

游戏对象之间需要沟通互动来完成游戏内交互。这需要通过事件系统来完成。

如何实现一个事件系统，完成游戏对象之间的互相通信：

先使用pygame定义一串自定义的事件

然后需要在每个游戏对象中提前定义好对应事件的处理函数（回调函数）（？）

最后我们需要一个事件处理者，使用发布订阅机制来完成事件的接收和分发

**问题：事件处理者作为谁的成员？通过什么机制来完成发布-订阅机制？事件被分发后，在游戏update的什么部分完成？**

20231109

将游戏从单纯的pygame变成了pygame+pymunk。引入了物理系统。

主要变化在于，弃用了姿态组件，引入了物理组件，负责一个游戏对象的物理属性处理（暂不包括碰撞和事件处理）。在引入的时候注意到了两个包坐标系不相符的问题。通过封装在物理组件里的转换解决了这个问题：将所有坐标同步为pygame坐标。物理组件自身涉及到坐标的时候自己处理自己的pymunk坐标。输出仍然输出的是pygame坐标

**现在存在的问题：**

1. **事件系统仍然没有构造起来，事件接受仍然耦合在游戏对象里。**
2. 没有做碰撞。因为碰撞也涉及到事件处理。碰撞到底是作为一个单独的组件，还是继承物理组件作为一个单独的组件，还是耦合在游戏对象中进行处理呢？
3. **处于学习和探索的阶段。各个组件，游戏对象，场景都没有抽象出基类来。这不利于游戏规范化。**

**20231110**

实现了子弹的发射。但是发射参数（子弹生成的位置）还和飞船有轻微的耦合

Pymunk非常方便的实现了碰撞事件的检测。昨天的第二点似乎暂时解决了。但是其它事件系统还没什么头绪。不知道有没有必要去做

昨天的问题在今天的实践中得到了部分解决。决定把碰撞处理交给游戏对象（或者任何具有物理实体的对象）来做。**之后需要在每个游戏对象中定义其类型**，以便处理碰撞。

常数类型定义在了constants.py中，将其算作了组件的一部分。

疑问：

1. 是否需要将碰撞处理提取出来，变成一个组件，输入类型，执行监测就可以？

注意：

删除一个游戏对象需要执行以下步骤：

**游戏精灵自身kill**

**从空间中移除其body和shape**

对于碰撞检测，一个物理体可能会在一次更新中和另一个产生多次碰撞（穿模）。需要做特殊处理，比如检测其是否已经被移除，来确保只碰撞一次

根据您的项目结构，以下是可能需要修改的关键部分，以便实现和适应摄像头系统：

**1. 游戏场景 (scences 目录)**

* **bulletAndGuntest.py**, **shipMove.py**, **shipMoveAllPID.py**:
  + 这些文件看起来像是处理不同游戏场景或关卡的脚本。
  + 您需要检查这些文件中的渲染逻辑，确保所有对象（如玩家飞船、敌人、子弹等）都是根据摄像头位置来渲染的。
  + 特别注意背景渲染逻辑，以便它能随着摄像头的移动而适当地滚动或更新。

**2. 游戏对象 (gameObjects 目录)**

* **bullet.py**, **playerShip2\_0.py**, **playerShipAllPID.py**:
  + 这些文件可能包含玩家飞船、子弹和其他游戏对象的定义。
  + 需要确保这些对象的位置是相对于游戏世界而不是屏幕坐标设定的。
  + 在实例化这些对象时，可能需要添加或修改它们以接受摄像头对象作为参数，或者在渲染时考虑摄像头的位置。

**3. 组件 (components 目录)**

* 这些文件似乎包含了游戏的各种组件，如物理、健康组件等。
* 虽然这些组件可能主要处理游戏逻辑而非渲染，但如果它们以任何方式影响对象的位置或渲染，您可能需要进行调整。

**4. 主游戏循环 (main.py)**

* 这是游戏的主入口点。
* 您需要在这里初始化摄像头对象，并在游戏的主循环中更新摄像头位置以及调整所有渲染调用。

**5. 检查其它相关文件**

* 如果项目中还有其他处理游戏逻辑或渲染的文件，也需要检查和修改。

在实现摄像头系统时，主要的挑战在于确保所有游戏对象都是在摄像头视角下正确渲染的，而不是仅仅基于它们在游戏世界中的绝对位置。这可能需要对现有代码进行相当程度的重构或调整。 ​

20231204

似乎基本完成了重构。之前的问题基本解决

待办：

* 1. 抽象一下bullet，实现一个抛体类
  2. 在游戏对象中加入缩放参数，修改渲染管线加一步进行缩放。在摄像头中加入可以缩放地图的选项。
  3. 游戏对象的handel\_event定义了它关心的事情。如果能够根据这里，在创建游戏对象的时候自动订阅就好了

20231205

完成了帧率控制和物理引擎时间-现实时间的一定程度上的对齐

20231228：

完成了游戏的基本框架，添加了一堆内容。根据优先级，现在还欠缺：

1. 继续学习Games104，学而不思则罔，思而不学则殆
2. 游戏地形对象的特殊化设计：

如果还使用游戏对象作为基类，那么巨大的地形对象将会把内存吃光

1. 游戏对象的销毁逻辑。销毁前需要执行的一段动画目前没法实现。

同样的，销毁时播放的一段音效也未实现

1. 重新考虑游戏对象是否需要传入场景作为其成员。利弊何如？
2. 游戏中按钮，会话框等元素还未制作。
3. 场景管理器的逻辑耦合度比较高。需要将其抽象入游戏引擎
4. 在场景管理器的基础上实现场景切换逻辑设计和（游戏对象）管理
5. 游戏BGM系统
6. 粒子系统
7. 游戏中的图像缓存和渲染时的旋转放缩计算是否能够继续优化？
8. 游戏流监视系统，用于完成积分，成就等功能