CS100 HW7 样例实现参考

注 意

- 请为本次作业分配充足的时间（参考：至少 1.5 天），并且不要到临近截止日期时才开始！

- 不要被本说明的长度吓到。它之所以长达 7 页，是因为在每个游戏对象的细节说明里，我们有意做了很多不必要的文字重复。我们希望你能够从这些重复的内容里提取出共同点与区分点，通过自己的构思与设计来解决继承结构等问题，而不是被我们用我们设计的结构束缚住。

- 本文档只是样例实现的介绍，你不必全部实现，也不必完全遵循其中要求。你的目标是做出「你自己」的游戏，而不是「与样例一致」的游戏。本文档只提供作为参考的大量的实现流程与数值等细节。

- 如果对任何部分有疑问，在你提问之前，请先在本说明和 FAQ 中寻找答案，或试着游玩样例游戏并观察样例游戏的处理方式。

- 请尽量在 Piazza 上公开提问而非私下询问 TA。有相同疑问的其他同学也可以看到你的问题，你也能看到我们对更多问题的回答。这会更高效地解决你的疑问。

- 请一定经常保存备份（无论离线或在线）！如果你为了添加某一点功能使得整个程序崩溃或是在修复时越修越糟，简单地回退一版并比较不同，就可以快速解决问题。

◆ 好的，所以这么多东西我怎么写？

可想而知，从零开始写出一个完整的游戏绝不是能一蹴而就的小工程。本游戏的大量内容需要逐步完成。因此，我们将推荐先实现的部分以蓝色字体标出。我们建议你先阅读完整的题目说明，对这个游戏基本了解，之后再开工。写码时建议从蓝色字体部分开始逐步进行。附件"建议的开始流程.pdf"提供了一个清晰的前几步流程，相信你完成这几步之后就会驾轻就熟。

在在提供的文件之中，你不需要也不应当修改 src/Framework 路径下的文件。对于 utils.hpp，在图片素材加载错误时你可能需要修改其中 ASSET_DIR 字符串，并且你可以在其中定义新的项目来添加属于你自己的内容（详见 FAQ：怎么添加新内容）。

下面是对游戏中每个组成部分的介绍。其中以符号开头的列表项是无关顺序的，数字或字母开头的列表项表示需要按顺序执行。

◆ GameWorld

◆ GameWorld::Init()

你的 GameWorld::Init()函数应当：

- 初始化任何用于记录关卡数据的成员变量。例如，记录游戏得分，游戏从开始运行的总帧数。
- 为需要显示的关卡数据创建文本显示。需要使用 TextBase 类，使用方法与你创建的所有 GameObject 均类似，详细用法参见 FAQ。
    - 得分数推荐显示在(WINDOW_WIDTH - 160, 8)。
- 创建背景（已经帮你创建作为示例）。
- 创建玩家（x=200，y=120）

◆ GameWorld::Update()

你的 GameWorld::Update()函数应当：

1. 为游戏生成新的哥布林。每 240ticks（8s）在（WINDOW_WIDTH-1，120）生成一个哥布林
2. 遍历所有游戏对象(GameObject)，并依次调用它们的 Update()函数。
3. 检测碰撞。注意不要让同一碰撞被触发多次。可能发生的碰撞有：
    - 子弹击中哥布林。
    - 斧头打到玩家。

- 哥布林碰到玩家。

4. 再次遍历所有游戏对象，将需要删除的对象从你的存储容器中移除。
   - 需要删除的对象应被标记为“死亡”或“HP 为零”等状态，GameWorld 才能找到。
5. 判断是否失败。若玩家死亡，则返回 LevelStatus::LOSING。
   - 在失败的显示画面上，我们预留了一个显示你分数的空位。你可以在此时创建一个文本来显示出你的游戏结果。建议以白色(RGB = 1，1，1)创建在(360，50)位置。关于文本颜色的细节请参考 FAQ。
6. 更新你在游戏中创建的文本显示(TextBase)的内容，以正确显示需要的信息，例如分数、HP 等。
7. 返回 LevelStatus::ONGOING，表示当前关卡在正常运行。

- **GameWorld::CleanUp()**

  你的 GameWorld::CleanUp()函数必须清空你所使用的容器。若你保存了其他对象，也需要将它们正确清除。

◆ 背景(Background)

- **当被创建时**
  - 背景的贴图编号为 IMGID_BACKGROUND。
  - 背景的位置为(x = WINDOW_WIDTH，y = WINDOW_HEIGHT / 2)。
  - 背景的所在层级为 LAYER_BACKGROUND。
  - 背景的宽度为 2*WINDOW_WIDTH，高度为 WINDOW_HEIGHT。（特殊的宽度和位置设计也是背景能不断向后跑的奥秘）
  - 背景不具有动画，即，动画编号为 ANIMID_NO_ANIMATION。
- **当 Update()时**

  背景每帧向后移动 3 个像素，若背景的横坐标小于等于 0，则向右移动 WINDOW_WIDTH

◆ 玩家(Player)

- **当被创建时**
  - 玩家的贴图编号为 ImageID::PLAYER。
  - 玩家的起始位置固定，（x=200，y=120）

- 玩家的所在层级为 LayerID::PLAYER。
- 玩家的宽度为 20，高度为 48。
- 玩家初始具有 AnimID::IDLE 动画。

◆ **当 Update()时**

- 玩家需要首先检查自己是否已经死亡。若已经死亡，它将等待 GameWorld 清理。如果有显示玩家 HP 的文字则需要擦除，然后立刻返回，无视下述步骤。
- 若按键 J 或鼠标左键按下，玩家则会开枪，子弹会从玩家坐标右边 30 个像素射出（子弹是从枪里面射出来的，而不是像某个瓦开头的游戏一样是从脑袋射出来的）
  - 开枪存在冷却，冷却为 10 帧
- 若按键 K 或空格被按下，玩家则会跳起
  - 跳跃是一个竖直上抛运动，存在向上初速度和重力加速度
  - 初速度的大小是 23 像素/帧，重力加速度的大小是 2 像素/帧 ^{2}
  - 跳跃将持续 24 帧，在跳跃状态下需要切换动画为 AnimID::JUMP
  - 若玩家正在跳跃，不会再次跳起（你可以自定义二段跳....）
  - 不限制跳跃过程中是否能开枪
  - 落地后需要切换回 AnimID::IDLE
  - 思考一下：要做到开枪（创建对象），玩家需要知道自己所在的 GameWorld，也就意味着需要储存这样一个成员。这个成员的类型应当是什么？GameWorld 创建子弹时，怎样把“所在的 GameWorld”这一信息告诉它？除了玩家，其他的类也可能需要知道所在的世界，那么这个信息应当存在哪儿？

◆ 子弹(Bullet)

◆ **当被创建时**

- 子弹的贴图编号为 ImageID::BULLET。
- 子弹的位置由创建它的代码决定。
- 子弹的所在层级为 LAYER_PROJECTILES。
- 子弹的宽度为 10，高度为 10。
- 子弹不具有动画。

◆ **当 Update()时**

1. 子弹需要首先检查自己是否已经死亡。若已经死亡，它将等待 GameWorld 清理。它的 Update()应当立刻返回，无视下述步骤。
2. 子弹将向右移动 10 像素。
3. 如果子弹飞出了屏幕右边界(x >= WINDOW_WIDTH)，子弹需要死亡。

◆ **当与哥布林碰撞时**

子弹将会对与它碰撞的哥布林造成 1 点伤害，然后死亡。

◆ 斧子(Axe)

◆ **当被创建时**

- 斧子的贴图编号为 ImageID::AXE。
- 斧子的位置由创建它的代码决定。
- 斧子的所在层级为 LAYER_PROJECTILES。
- 斧子的宽度为 25，高度为 25。
- 斧子不具有动画 AnimID::NO_ANIMATION。

◆ **当 Update()时**

1. 斧子需要首先检查自己是否已经死亡。若已经死亡，它将等待 GameWorld 清理。它的 Update()应当立刻返回，无视下述步骤。
2. 斧子将向左移动 10 像素。
3. 如果斧子飞出了屏幕左边界(x <= 0)，斧子需要死亡。

◆ **当与玩家碰撞时**

斧子将会对与它碰撞的玩家造成 1 点伤害，然后死亡。

◆ 哥布林(Goblin)

◆ **当被创建时**

- 哥布林的贴图编号为 ImageID::GOBLIN。
- 哥布林的位置由创建它的代码决定。
- 哥布林的所在层级为 LAYER_ZOMBIES。

- 哥布林的宽度为 20，高度为 48，如图。这样设定是为了给玩家跳过它操作的空间
- 哥布林初始具有 AnimID::IDLE 动画。
- 哥布林具有 5 点 HP。

◆ **当 Update()时：**

1. 哥布林需要首先检查自己是否已经死亡。若已经死亡，它将等待 GameWorld 清理。它的 Update()应当立刻返回，无视下述步骤。
2. 它将向左移动 3 像素。
3. 每 100ticks 丢出一个斧子，特殊的是，丢出斧子需要在动画播放 20ticks 后实现：你需要先播放动画 AnimID::THROW，等 20ticks 后在哥布林的坐标处创建一个斧子

◆ **当发生碰撞时**

- 如果与子弹发生碰撞，哥布林将会受到来自子弹伤害数值的伤害，并让那颗子弹死亡。
- （可选）如果与子弹发生碰撞，哥布林会播放受伤动画 AnimID::HURT，时长为 5 帧，可以选择加入打断施法或减速等特性
- 如果与玩家发生碰撞，哥布林将会立马死亡，并对玩家造成 1 点伤害

◆ 附录 – 碰撞处理

每个对象的碰撞体均为中心在(x, y)，横向宽度为 width，纵向高度为 height 的长方形。

若两对象的碰撞体长方形发生重叠，则认为两对象发生碰撞。

提示：碰撞关系似乎看起来有些复杂：有可能左右或上下交叉，有可能一个包围另一个，但实际上有非常简单的判断方法：考虑两个对象的中心点，是否碰撞是否跟他们中心点之间的 x 或 y 距离有关？

◆ 彩蛋（可选） – 飞升

◆ 依次按顺序按下上上下下左右左右 KJKJ 后，即可获得怒瑞玛的力量完成飞升

◆ 实现方法则是使用状态机，而且对键盘按键进行消抖处理，防止键盘把一次按键读取成连续多次