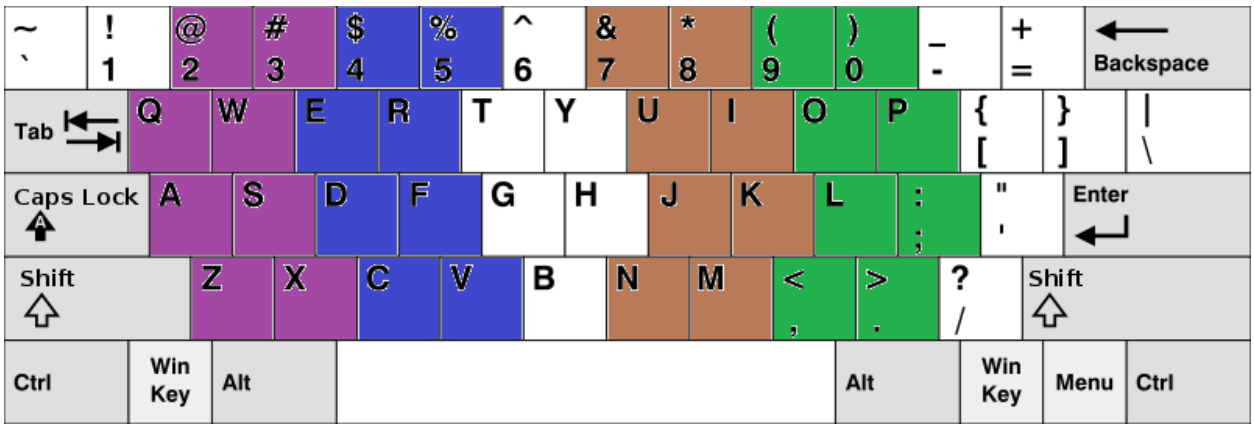
2018 台大椰林資訊營 AI Challenge
Quantum Vortex API 使用說明

遊戲方式

- 開啟方式
  - Windows
    - 開啟 Anaconda Prompt，執行 `pip install pygame`
    - 移動到 `challenge2018` 資料夾下，並輸入 `python main.py`
  - Ubuntu
    - 開啟終端機，執行 `pip3 install pygame`
    - 移動到 `challenge2018` 資料夾下，並輸入 `python3 main.py`
- 真人對戰



  - 角色操作(最多可手動控制四隻角色)
    - 改變移動模式：`z` / `c` / `n` / `,`
    - 重力塌陷　　：`2` / `4` / `7` / `9`
    - 粒子繞射　　：`3` / `5` / `8` / `0`
    - 高能電磁脈衝：`q` / `e` / `u` / `o`
    - 核分裂　　：`w` / `r` / `i` / `p`
    - 超次元狙擊　：`a` / `d` / `j` / `l`
    - 激發光譜調和：`s` / `f` / `k` / `;`
    - 重力波共振　：`x` / `v` / `m` / `.`
- 使用 AI 進行對戰
  - cmd 下輸入 `python main.py "111" "222"`，即可開啟 `team_player111.py` (當成 player 0) 與 `team_player222.py` (當成 player 1) 的 AI 進行遊戲，不足四個的玩家將會以鍵盤控制補齊。
  - 所有的 AI 程式碼都必須在 `AI` 這個資料夾中，檔名為 `team_playerXXX.py`，其中 XXX 為 AI 名稱 (可使用英文字母或數字，長度 10 以內為佳)，AI 名稱會出現在計分板上。
  - 可以使用 `~` 表示鍵盤控制的玩家，`_` 表示預設的 AI (`team_default.py`)
    e.g. `python main.py "_" "_" "~" "~"`
- 跳過動畫特效

- 在打滿四位玩家的情況下，在結尾加上 `--no-cutin` 即可。
  e.g. `python main.py "_" "_" "~" "~" --no-cutin`

Helper 取得場上資訊

- Helper 裡面有大量函式幫助你取得場上的資訊！
- 為方便 API 理解及記憶，一些專有名詞將以較簡略的名詞代替，並附有 API 中這些專有名詞的英文翻譯。
  - 重力塌陷　　：爆炸 / Explosion
  - 量子貨幣　　：白球 / White Ball, wb
  - 脈衝波　　：子彈 / Bullet
  - 暗物質　　：子彈 / Bullet (註1)
  - 粒子繞射　　：多重子彈 / Multibullet
  - 高能電磁脈衝：大子彈 / Bigbullet
  - 太空艙控制室：頭 / Head
  - 附屬量子貨幣：身體 / Body
  - 量子穿隧　　：衝刺 / Dash
  - (註1) 暗物質在遊戲中歸類於無主人的子彈，意即所有玩家碰到暗物質皆會死亡。
- 若對 API 進行無意義的詢問，可能會得到 `None` 的回傳值，可用 `if var is None` 判斷 `var` 變數是否為 `None`，否則程式碼將無法運行。

地圖相關

- `helper.getNearestGravOnRoute()`
  - 若行進方向上沒有重力場，回傳 None。
  - 否則回傳 (center, radius)，代表目前行進方向的直線路徑上，離玩家最近的重力場。
    - cecnter 為 tuple (x, y) ，代表該重力場的圓心位置。
    - radius 代表該重力場的半徑。
- `helper.getAllGravs()`
  - 回傳 list of (center, radius)，代表場地上所有的重力場。
    - center 為 tuple (x, y) ，代表重力場的圓心位置。
    - radius 代表重力場的半徑。
- `helper.getNearestPosToCenter()`
  - 若玩家不在重力場內，回傳 None。
  - 否則回傳 tuple，代表目前行進方向的直線路徑上，最靠近玩家身處的重力場中心的點。
    - tuple 為 (x, y) ，代表該點的位置。
- `helper.getBallNumInRange(center, radius)`
  - center 是一個 tuple (x, y)，代表某點的位置。
  - radius 是一個 float，代表半徑大小。
  - 回傳距離 center 小於 radius 的範圍中，場地上白球的數量。
- `helper.getOtherBulletNumInRange(center, radius)`
  - center 是一個 tuple (x, y)，代表某點的位置。
  - radius 是一個 float，代表半徑大小。

    - 回傳距離 center 小於 radius 的範圍中，場地上所有子彈的數量。
- `helper.getAllBallsPos()`
    - 回傳 list of tuples，代表場地上所有白球的位置。
        - tuple 為 (x, y) ，代表該點的位置。
- `helper.getExplosivePos()`
    - 回傳 list of tuples，代表場地上所有爆炸道具的位置。
- `helper.getMultibulletPos()`
    - 回傳 list of tuples，代表場地上所有多重子彈道具的位置。
- `helper.getBigbulletPos()`
    - 回傳 list of tuples，代表場地上所有大子彈道具的位置。
- `helper.canGetByExplosion(pos)`
    - pos 為 tuple (x, y)，代表爆炸道具的位置。
    - 回傳 int，代表在 pos 發動爆炸時，能吃到的白球數。
- `helper.canGetBySpin()`
    - 若玩家不在重力場內，回傳 None。
    - 否則回傳 int，代表在目前位置進行旋轉時能吃到的白球數。
- `helper.canGetOnRoute()`
    - 回傳 int，代表在目前位置行進方向的直線路徑上能吃到的白球數。
- `helper.getNearestballOnRoute()`
    - 若玩家目前位置行進方向的直線路徑上沒有白球，回傳 None。
    - 回傳 tuple (x, y)，代表在目前位置行進方向的直線路徑上能吃到的最近的白球的位置。

自身相關

- `helper.getMyIndex()`
    - 回傳 int，代表玩家編號。
- `helper.getMyHeadPos()`
    - 回傳 tuple (x, y)，代表玩家頭的位置。
- `helper.getMyBodyPos()`
    - 回傳 list of tuples，代表玩家所有身體的位置。
- `helper.getMyDir()`
    - 回傳 tuple (x, y)，代表玩家的移動單位向量。
- `helper.getMyGrav()`
    - 若玩家不在重力場內，回傳 None。
    - 否則回傳 (center, radius)，代表該重力場。
        - center 為 tuple (x, y) ，代表該點的位置。
- `helper.getDashPos()`
    - 若玩家不是正在衝刺，回傳 None。
    - 否則回傳 tuple，代表衝刺完後玩家會到達的位置。
        - tuple 為 (x, y) ，代表該點的位置。
- `helper.getMyDashRemainTime()`

    - 回傳 int，代表離衝刺完成還有幾個 tick 的時間。
- `helper.getMyDashCoolRemainTime()`
    - 回傳 int，代表離衝刺冷卻完成還有幾個 tick 的時間。
- `helper.checkMeInGrav()`
    - 回傳 boolean，代表玩家是否處在重力場內。
- `helper.checkMeCircling()`
    - 回傳 boolean，代表玩家是否正在繞重力場旋轉。
- `heler.checkInvisible()`
    - 回傳 boolean，代表玩家是否正在衝刺。
- `helper.getMyCirclingRadius()`
    - 回傳 float，代表玩家正在旋轉的半徑大小。
- `helper.getMyBullet()`
    - 回傳 list of tuples，代表玩家所有子彈。
    - tuple 為 (pos, direction, radius, speed)
        - pos 為 tuple (x, y)，代表子彈位置。
        - direction 為 tuple (x, y)，代表子彈前進方向。
        - radius 為 int，代表子彈大小。
        - speed 為 float，代表子彈速度。
- `helper.getMyScore()`
    - 回傳 int，代表玩家目前的分數。

其他玩家相關

- `helper.headOnRoute()`
    - 回傳 list of tuples，代表在玩家目前行進方向的直線路徑上所有除了自身以外活著的玩家的頭的位置。
- `helper.bodyOnRoute()`
    - 回傳 list of tuples，代表在玩家目前行進方向的直線路徑上所有除了自身以外活著的玩家的身體的位置。
- `helper.collisionOnRoute(pos1, radius1, _dir, pos2, radius2)`
    - pos1 為 tuple (x, y)，代表移動的物體的位置。
    - radius1 為 int，代表移動的物體的半徑。
    - _dir 為 tuple (x, y)，代表移動的物體的移動單位向量。
    - pos2 為 tuple (x, y)，代表不動的物體的位置。
    - radius2 為 int，代表不動的物體的半徑。
    - 回傳 boolean，代表這兩樣物體會不會相撞。
- `helper.getPlayerHeadPos(player_id)`
    - player_id 是 int，代表想詢問的玩家編號。
    - 若該玩家已經死亡，回傳 None。
    - 否則回傳 tuple (x, y)，代表玩家頭的位置。
- `helper.getPlayerBodyPos(player_id)`
    - player_id 是 int，代表想詢問的玩家編號。
    - 若該玩家已經死亡，回傳 None。
    - 否則回傳 list of tuples，代表玩家所有身體的位置。

- `helper.getPlayerDir(player_id)`
  - player_id 是 int，代表想詢問的玩家編號。
  - 若該玩家已經死亡，回傳 None。
  - 否則回傳 tuple (x, y)，代表玩家的移動單位向量。
- `helper.getPlayerDashRemainTime(player_id)`
  - player_id 是 int，代表想詢問的玩家編號。
  - 若該玩家已經死亡，回傳 None。
  - 否則回傳 int，代表離衝刺完成還有幾個 tick 的時間。
- `helper.getPlayerDashCoolRemainTime(player_id)`
  - player_id 是 int，代表想詢問的玩家編號。
  - 若該玩家已經死亡，回傳 None。
  - 否則回傳 int，代表離衝刺冷卻完成還有幾個 tick 的時間。
- `helper.checkPlayerInGrav(player_id)`
  - player_id 是 int，代表想詢問的玩家編號。
  - 若該玩家已經死亡，回傳 None。
  - 否則回傳 boolean，代表玩家是否處在重力場內。
- `helper.checkPlayerInvisible(player_id)`
  - player_id 是 int，代表想詢問的玩家編號。
  - 若該玩家已經死亡，回傳 None。
  - 否則回傳 boolean，代表玩家是否正在衝刺。
- `helper.checkPlayerCircling(player_id)`
  - player_id 是 int，代表想詢問的玩家編號。
  - 若該玩家已經死亡，回傳 None。
  - 否則回傳 boolean，代表玩家是否正在繞重力場旋轉。
- `helper.getPlayerCirclingRadius(player_id)`
  - player_id 是 int，代表想詢問的玩家編號。
  - 若該玩家已經死亡，回傳 None。
  - 否則回傳 float，代表玩家正在旋轉的半徑大小。
- `helper.getAllPlayerBullet()`
  - 回傳 list of tuples，代表所有自身以外的子彈。
  - tuple 為 (index, pos, direction, radius, speed)
    - index 為 int，代表發射該子彈的玩家編號（若該子彈屬於暗物質，則玩家編號為-1）。
    - pos 為 tuple (x, y)，代表子彈位置。
    - direction 為 tuple (x, y)，代表子彈前進方向。
    - radius 為 int，代表子彈大小。
    - speed 為 float，代表子彈速度。
- `helper.getPlayerScore(player_id)`
  - player_id 是 int，代表想詢問的玩家編號。
  - 回傳 int，代表玩家目前的分數。
- `helper.getAllBodyPos()`
  - 回傳 list of tuples，代表除了自身以外所有活著的玩家的身體的位置。

常數相關

- `helper.explosive_radius` : 重力塌陷的爆炸範圍半徑
- `helper.head_radius` : 太空艙控制室(頭)的半徑
- `helper.body_radius` : 太空艙附屬量子貨幣(身體)的半徑
- `helper.wb_radius` : 場地上量子貨幣(白球)的半徑
- `helper.normal_speed` : 正常狀態下太空艙的移動速度
- `helper.dash_speed` : 量子穿隧(衝刺)狀態下太空艙的移動速度
- `helper.dash_cool` : 量子穿隧(衝刺)的冷卻時間
- `helper.bullet_acc` : 脈衝波(子彈)的加速度
- `helper.dash_time` : 量子穿隧(衝刺)狀態下太空艙的總耗時
- `helper.bullet_speed` : 脈衝波(子彈)的初速度
- `helper.bullet_radius` : 脈衝波(子彈)的半徑

技能卡使用

在 AI 程式碼中的 `__init__` 增加 `self.skill = [1, 1, 1, 1, 1, 1, 1]`，數字代表有幾張技能卡，`self.skill` 中的第 0 ~ 6 項分別對應到

- 重力塌陷(爆炸)
- 粒子繞射(多重子彈)
- 高能電磁脈衝(大子彈)
- 核分裂
- 超次元狙擊
- 激發光譜調和
- 重力波共振

AI 回傳值

- `AI_NothingToDo` : 不做任何動作
- `AI_MoveWayChange` : 改變移動方式
- `AI_Explosion` : 使用技能卡-重力塌陷(爆炸)
- `AI_MultiBullet` : 使用技能卡-粒子繞射(多重子彈)
- `AI_BigBullet` : 使用技能卡-高能電磁脈衝(大子彈)
- `AI_NuclFission` : 使用技能卡-核分裂
- `AI_HypSniping` : 使用技能卡-超次元狙擊
- `AI_ExSpecHarmony` : 使用技能卡-激發光譜調和
- `AI_GravResonance` : 使用技能卡-重力波共振