API description

h20191030502

December 2019

0.1 MyStrategy

In language pack for your programming language you can find file named `MyStrategy/my_strategy.<ext>`. This file contains class `MyStrategy` with `get_action` method, where your strategy's logic should be implemented.

This method will be called each tick, separately for each of your units.

The method takes following arguments:

- Current unit for which to compute next action (*Unit* object)
- Current game state (*Game* object)
- Debug helper object (*Debug* object). This object allows you to do custom rendering from inside your strategy code. Note that using this has no effect when testing your strategy on the server, so use it for local debugging only, otherwise it will cause additional time overhead.

The method should return *UnitAction* object, defining the desired action for specified unit.

0.2 Objects description

In this section, some fields may be absent (denoted as $Optional\langle type\rangle$). The way this is implemented depends on the language used. If possible, a dedicated optional (nullable) type would be used, otherwise other methods may be used (like a nullable pointer type).

Some objects may take one of several forms. The way it is implemented depends on the language. If possible, a dedicated sum (algebraic) data type is used, otherwise other methods may be used (like variants being classes inherited from abstract base class).

$float32$ — 32-bit floating point number, is called $float$, and $float64$ is called *double* in some languages.

0.2.1 Bullet

Defines a bullet. Fields:

- $weapon\_type$: $WeaponType$ — type of the weapon the bullet was shot from
- $unit\_id : int$ — id of unit who shot the bullet
- $player\_id : int$ — id of player of the shooter
- $position : Vec2\langle float64\rangle$ — bullet's position (center of the bullet)
- $velocity : Vec2\langle float64\rangle$ — bullet's velocity (in units per second)
- $damage : int$ — damage that will be dealt to a unit when hit

- *size* : $float64$ — bullet's size (side length of square that defines the bullet)
- *explosion_params* : *Optional⟨ExplosionParams⟩* — explosion parameters, if applicable

0.2.2 BulletParams

Parameters used to create a bullet when shooting a weapon. Fields:

- *speed* : $float64$ — bullet's speed (length of velocity vector)
- *size* : $float64$ — bullet's size
- *damage* : *int* — bullet's damage

0.2.3 Color

Defines color (used for debug rendering). Fields:

- r : $float32$ — red component
- g : $float32$ — green component
- b : $float32$ — blue component
- a : $float32$ — alpha component (opacity)

0.2.4 ColoredVertex

Defines a vertex (used for debug rendering). Fields:

- *position* : $Vec2\langle float32\rangle$ — vertex position
- *color* : *Color* — vertex color

0.2.5 CustomData

Defines custom data sent to *Debug* object (used for debug rendering). May take one of the following forms:

- *Log* — used for logging text. Fields:
  - *text* : *string* — text to display
- *Rect* — draw a rectangle. Fields:
  - *pos* : $Vec2\langle float32\rangle$ — rectangle position (bottom left corner)
  - *size* : $Vec2\langle float32\rangle$ — rectangle size
  - *color* : *Color* — filling color
- *Line* — draw a line segment. Fields:

  - $p1 : Vec2\langle float32\rangle$ — first end point
  - $p2 : Vec2\langle float32\rangle$ — second end point
  - $width : float32$ — line width
  - $color : Color$ — line color

- *Polygon* — draw a convex polygon. Each vertex may be colored separately — color will be interpolated between vertices. Fields:

  - $vertices : List\langle ColoredVertex\rangle$ — list of vertices

- *PlacedText* — text. Fields:

  - $text : string$ — text to display
  - $pos : Vec2\langle float32\rangle$ — position (in game coordinates)
  - $alignment : TextAlignment$ — text alignment
  - $size : float32$ — text font size in pixels
  - $color : Color$ — text color

0.2.6 ExplosionParams

Parameters of explosion from a mine or an exploding bullet. Fields:

- $radius : float64$ — radius of explosion (half of side length of explosion's square area)

- $damage : int$ — damage dealt by explosion

0.2.7 Game

Defines current game state. Fields:

- *current_tick* : *int* — current tick index

- *properties* : *Properties* — game's properties (constants)

- *level* : *Level* — level (map)

- *players* : $List\langle Player\rangle$ — list of players (strategies) participating in the game

- *units* : $List\langle Unit\rangle$ — list of alive units

- *bullets* : $List\langle Bullet\rangle$ — list of flying bullets

- *mines* : $List\langle Mine\rangle$ — list of **planted** mines

- *loot_boxes* : $List\langle LootBox\rangle$ — list of loot boxes

0.2.8 Item

Defines an item contained in a loot box. May take one of the following forms:

- *HealthPack* — health pack. Fields:
  - *health* : *int* — health restored
- *Weapon* — a weapon. Fields:
  - *weapon_type* : *WeaponType* — type of the weapon
- *Mine* — a mine. No fields.

0.2.9 JumpState

Defines unit's jump state. Fields:

- *can_jump* : *boolean* — whether unit can start/continue jumping
- *speed* : *float*64 — jump speed (in units per second)
- *max_time* : *float*64 — max jump time (in seconds)
- *can_cancel* : *boolean* — whether current jump can be canceled

0.2.10 Level

Defines the level. Fields:

- *tiles* : *List*⟨*List*⟨*Tile*⟩⟩ — 2d list of level tiles

0.2.11 LootBox

Defines a loot box. Fields:

- *position* : *Vec*2⟨*float*64⟩ — loot box's position (bottom middle point)
- *size* : *Vec*2⟨*float*64⟩ — loot box's size
- *item* : *Item* — item contained in this loot box

0.2.12 Mine

Defines a **planted** mine. Fields:

- *player_id* : *int* — id of player whose unit planted the mine
- *position* : *Vec*2⟨*float*64⟩ — mine's position (bottom middle point)
- *size* : *Vec*2⟨*float*64⟩ — mine's size
- *state* : *MineState* — current mine state

- $timer$: $Optional\langle float64\rangle$ — time left until state change. May be absent
- $trigger\_radius$: $float64$ — mine's triggering radius
- $explosion\_params$: $ExplosionParams$ — explosion params

0.2.13 MineState

Defines mine's state. Variants:

- Preparing
- Idle
- Triggered

0.2.14 Player

Defines player (strategy) participating in the game. Fields:

- id : int — player's id
- $score$: int — current player's score

0.2.15 Properties

Defines game's properties (constants). Fields:

- max_tick_count : int — max game duration in ticks
- $ticks\_per\_second$: $float64$ — number of ticks per "second"
- $updates\_per\_tick$: int — number of updates per tick. Each update advances game time by $\frac{1}{ticks\_per\_second \times updates\_per\_tick}$
- $loot\_box\_size$: $Vec2\langle float64\rangle$ — size of all loot boxes
- $unit\_size$: $Vec2\langle float64\rangle$ — size of all units
- $unit\_max\_horizontal\_speed$: $float64$ — max horizontal speed of a unit
- $unit\_fall\_speed$: $float64$ — unit's fall speed
- $unit\_jump\_time$: $float64$ — unit's regular jump time
- $unit\_jump\_speed$: $float64$ — unit's regular jump speed
- $jump\_pad\_jump\_time$: $float64$ — unit's jump time off a jump pad
- $jump\_pad\_jump\_speed$: $float64$ — unit's jump speed off a jump pad
- $unit\_max\_health$: int — max (starting) unit's health

- $weapon\_params$: $Map\langle WeaponType \to WeaponParams\rangle$ — weapon parameters by weapon type
- $mine\_size$: $Vec2\langle float64\rangle$ — size of **planted** mines
- $mine\_explosion\_params$: $ExplosionParams$ — explosion params for mines
- $mine\_prepare\_time$: $float64$ — mine preparing time
- $mine\_trigger\_time$: $float64$ — mine triggering time
- $mine\_trigger\_radius$: $float64$ — mine triggering radius
- $kill\_score$: int — score given for each unit killed

0.2.16 TextAlignment

Defines text alignment (used for debug rendering). Variants:

- Left
- Center
- Right

0.2.17 Tile

Defines level's tile. Variants:

- Empty
- Wall
- Platform
- Ladder
- JumpPad

0.2.18 Unit

Defines a unit. Fields:

- $player\_id$: int — owner player's id
- id : int — unit's id
- $health$: int — unit's health
- $position$: $Vec2\langle float64\rangle$ — unit's position (bottom middle point)
- $size$: $Vec2\langle float64\rangle$ — unit's size
- $jump\_state$: $JumpState$ — current unit's jump state
- $mines$: int — number of mines in unit's inventory
- $weapon$: $Optional\langle Weapon\rangle$ — current unit's weapon, if any

0.2.19 UnitAction

Defines unit's action. Fields:

- $velocity$: $float64$ — target horizontal velocity
- $jump$: $boolean$ — whether to start/continue jumping or go up ladders
- $jump\_down$: $boolean$ — whether to jump down through platforms/go down ladders
- aim : $Vec2\langle float64\rangle$ — aiming direction. Ignored if length is less than 0.5
- $shoot$: $boolean$ — controls shooting/reloading
- $reload$: $boolean$ — whether to reload your weapon
- $swap\_weapon$: $boolean$ — whether to swap weapon with the one in a loot box
- $plant\_mine$: $boolean$ — controls planting mines

0.2.20 Vec2

Defines a 2d vector. Fields:

- x : $float$
- y : $float$

0.2.21 Weapon

Defines unit's weapon. Fields:

- $type$: $WeaponType$ — weapon's type
- $params$: $WeaponParams$ — weapon parameters
- $magazine$: int — current magazine size
- $spread$: $float64$ — current spread value
- $fire\_timer$: $Optional\langle float64\rangle$ — time until next possible shot (absent if shooting is possible already)
- $last\_angle$: $Optional\langle float64\rangle$ — last aiming angle (absent upon pickup)

0.2.22 WeaponParams

Defines weapon parameters (constants). Fields:

- $magazine\_size : int$ — max magazine size
- $fire\_rate : float64$ — time between consequent shots
- $reload\_time : float64$ — time required for reloading
- $min\_spread : float64$ — min spread value
- $max\_spread : float64$ — max spread value
- $recoil : float64$ — recoil value
- $aim\_speed : float64$ — aiming speed
- $bullet : BulletParams$ — bullet parameters
- $explosion : Optional\langle ExplosionParams\rangle$ — bullet's explosion parameters (if applicable)

0.2.23 WeaponType

Variants:

- Pistol
- AssaultRifle
- RocketLauncher