Configuration Format Documentation

COS 301: REDIRECTION

2018

Contents

1 Introduction

The purpose of this document is to provide insight into the functions and code utilised in the lua scripts generated by the DotA 2 bot configuration system.

2 Terminology and concepts

Dota/Dota 2 Defense of the Ancients 2, the game for which we are designing the bot configuration system.

Ancient The core building of each team which must be protected at all costs. Losing this building loses the game.

Bot A computer controlled hero in a team in Dota.

Radiant The team on the bottom left hand side of the Dota map.

Dire The team on the top right hand side of the Dota map.

HP Hit Points, the amount of sustained damage a hero can take.

MP Mana Points, used for casting spells.

Lane One of three paths on the Dota map. The lanes are located on the Left/Top side of the map, Middle and Bottom/Right side of the map and are referred to as top, mid and bottom or bot lanes, respectively. Lanes can also be classified into 3 types: Safe Lane, Mid Lane and Off lane. The Safe Lane has its Tower placed furthest from the base to provide additional support for the heroes in that lane. For Radiant, Safe Lane is the bot lane. For Dire, it is top lane. Radiant and Dire have their mid towers placed mostly equally apart on each side of the map, providing a very equal match-up between the heroes in the lane. Offlane is the hardest lane, because the tower is closest to the base and the activity in the lane is likely futher away from this tower in this lane than the other team. For Radiant, Offlane is top lane and for Dire it is bottom lane. Note that a safe lane for one team faces the offlane of the opposing team.

River The river that divides the map into the Radiant and Dire sides.

Jungle The areas on the map that are not a lane and not in the river.

Objective An entity in the Dota game that is one of many steps in achieving victory. An objective can be a Tower, a Barracks or Roshan.

Roshan A massive ancient creep that grants bonuses to the heroes that kill him, including Aegis of Immortals, Cheese and a Refresher Shard. The Aegis will grant any hero who wields it a second life; the cheese will instantly restore 2000 HP and MP, and the Refresher Shard is a one-time use item that instantly refreshes all cooldowns of a hero.

Tower A defensive turret that can be destroyed by the opposing team. There are 3 turrets per lane, each placed more or less the same distance apart in the lane, but distances do vary. These towers are referred to as Tier x towers, where x is either 1, 2 or 3 with a Tier 1 tower being furthest from base and Tier 3 being closest to base. There are an additional 2 towers in front of the team's Ancient, called Tier 4 towers.

Teamfight A big fight involving nearly all team members of both teams in any arbitrary location on the map. The victor of the teamfight is determined by the number of losses, and if it is even, it is then decided by the number of core heroes still alive, and if it is still even, then it is either decided by the net gold change of a team, or it is dubbed a draw. Victory in a teamfight will likely result in an objective being taken, whereas defeat will result in a very defensive stance being taken by the team.

Creeps Little minions that spawn and run down a lane to meet the opposing team's creeps, also called a creep wave. There are also neutral creeps that spawn in the jungle areas of the map that give gold and experience when killed. If a team destroys a barracks of the enemy team, then the creeps of the team that destroyed the barracks will start spawning stronger, upgraded creeps in that lane, called super creeps. Once all barracks are destroyed, the creeps are further upgraded to mega creeps. An upgraded creep has more HP, and gives less gold and experience.

Push A collective effort from a team to attempt to advance down a lane to attempt to take an objective, a teamfight, or both.

Farm The simple activity of a hero collecting gold and experience by killing creeps.

Defend The act of staying close to or even behind your own tower and creep wave when the enemy team is pushing down your lane.

Roaming The act of not remaining in a lane for too long, but rather moving between lanes to attempt to kill enemy heroes with the other heroes in that lane or in the jungle.

Ganking The act of a group of team players grouping together to attempt to kill enemy heroes on the map.

3 Scripts and their Functions

The following points are subdivided by file and then further by the important functions that are being implemented to be used by the bot team.

3.1 Configuration Format

The ConfigurationFormat.ts contains the ConfigurationFormat class. The file looks as follows:

```
export interface ConfigurationFormat {
  push: {
    top: Configuration;
    mid: Configuration;
    bot: Configuration;
  };
  farm: {
    top: Configuration;
    mid: Configuration;
    bot: Configuration;
  };
  defend: {
    top: Configuration;
    mid: Configuration;
    bot: Configuration;
  };

  roam: Configuration;

  roshan: Configuration;
}

export interface Configuration {
  compoundConditions: CoumpoundCondition[];
  initalValue: any;
}

export interface CoumpoundCondition {
  conditions: Condition[];
  logicalOperator: LogicalOperator[];
}

export interface Condition {
  trigger: Trigger;
  operator: Operator;
  conditional: any;
  action: Action;
  value: any;
}

export enum Action {
  Modify = 1,
  Return,
}

export enum Trigger {
  Time = 1,
  EnemyHeroesAlive,
  AlliedHeroesAlive,
  NumEnemyHeroesVisible,
  RadiusAlliedHeroes,
}

export enum LogicalOperator {
  AND = 1,
  OR,
}

export enum Operator {
  LessThan = 1,
  LessThanEqualTo,
  EqualTo,
  GreaterThanEqualTo,
  GreaterThan,
  NotEqual,
}
```

Each of push, farm, defend, roam and roshan correspond to a function in team_desires.lua, which are explained in more detail below. Push, defend and farm are objects that have properties for each of the 3 lanes on the map. Each lane as well as roshan and roam attributes is an array of Configuration objects that will be converted into lua conditions.

3.1.1 Explanation of each of the interfaces

A Condition instance will have 5 properties:

- trigger of enumerated type Trigger
- an operator of enumerated type Operator
- a conditional of type any as it may be a number or function name in Lua
- an action of enumerated type Action, and finally
- a value of type any, which is the value that the common variable in the Lua function will be modified by and can be numerical or a function name that returns a value

Thus, the general format of a Condition as represented in code will look as follows:

```
  if (${trigger} ${operator} ${conditional}) {
    ${action} ${value}
  }
```

A CompoundCondition object has 2 properties:

- an array of n Conditions, and

- an array of $n - 1$ logicalOperator enumerated types

The array of conditions will be used to build up a compound if statement joined together by the logical operators. In the event of a simple if statement with only one condition, then the logicalOperators array will be empty and the condition array will consist of one instance of Condition and use the format of a single Condition instance as shown above.

The general format of a CompoundCondition instance is shown below:

```
if (${conditions[0]} ${logicalOperator[0]} ${conditions[1]} ... $
    {logicalOperator[n - 1]} ${conditions[n]}) {
  ${conditions[0].action} MEAN(${conditions[0].value}, ..., ${
      conditions[n].value})
}
```

A Configuration instance has 2 properties:

- an array of CompoundConditions, and
- and initial value

An instance of Configuration represents the total configurations found within a function in team_desires.lua. The array of CompoundConditions represent all the if statement conditions that are to be evaluated, and the intial value represents the starting number that the variable to be manipulated within the function will be.

The general format of a Configuration instance expanded into the if conditions in a function is as follows:

```
local common = ${initialValue}

if (${CompoundCondition[0]}) {...}

if (${CompoundCondition[1]}) {...}

...

if (${CompoundCondition[n]}) {...}

return common
```

3.1.2 Explanation of the Enumerated Types

Action determines the type of action to be taken inside an if statement. The 2 possible options are to return a value or to modify the variable common inside the Lua function.

Trigger determines what is being tested in an if statement. The possible values are

- time (the time in the Dota Game),
- EnemyHeroesAlive which corresponds to the number of enemy heroes that are alive
- AlliedHeroesAlive which corresponds to the number of allied heroes that are alive
- NumEnemyHeroesVisible that corresponds to the number of enemy heroes that are currently visible on the minimap
- RadiusAlliedHeroes which corresponds to the radius of the circle that covers all the allied heroes on the map

Operator determines the type of operator to be used in an if statement. They are as follows

- LessThan
- LessThanEqualTo
- EqualTo
- GreaterThanEqualTo
- GreaterThan
- NotEqual

The explanations for these operators are self-explanatory.

LogicalOperator determines the type of logical operator to be used to join multiple Condition instances. The possible values are

- OR which corresponds to the logical operator OR, that returns true if either or both conditions are true
- AND which corresponds to the logical operator OR, that returns true only if both conditions are true

3.2 Team Desires (team_desires.lua)

This file contains functions that determine the desires of the team as a whole.

3.2.1 UpdatePushLaneDesires()

This functions returns a vector of 3 floating point values between 0 and 1 that represents the collective desire of the bot team to push each of the 3 lanes on the map. A value of 1 is an absolute desire and 0 is no desire at all. The first value corresponds to the top lane, the second for mid lane and the last for bottom lane.

There are several factors that may influence the desire of a team to push a specfic lane:

- Whether the enemy team is alive or not, or rather, how many of the heroes on the enemy team are alive. Fewer heroes alive will correlate with a greater desire to push any lane.
- Whether the entire allied team is alive or not. There will be little to no desire to push if the team is mostly dead.
- Where the creep waves have met in the lane. A creep wave closer to the team's base/tower will result in a higher desire to push the lane to the opposite side of the map.
- Whether the team has any illusions/controllable minions with them, or can spawn any. If the team does, they will have a greater desire to push.
- Whether the enemy team is close by or not. If they are close by, there will be little desire to push the lane.

3.2.2 UpdateDefendLaneDesires()

This function also returns a vector of 3 floating point values between 0 and 1 representing the team's desire to push top, mid and bottom lanes respectively. A value of 1 is an absolute desire and 0 is no desire at all.

Factors that may influence the defence of a lane:

- If the enemy team is pushing down a lane, the team will have a higher desire to defend that lane.
- If enemy creeps are mega creeps, then the team will have a much stronger desire to defend
- If any creeps are attacking buildings, the team will have a strong desire to defend.

3.2.3 UpdateFarmLaneDesires()

This function also returns a vector of 3 floating point values between 0 and 1 representing the team's desire to push top, mid and bottom lanes respectively. A value of 1 is an absolute desire and 0 is no desire at all.

Factors influencing the desire to farm the lane:

- It is early in the game and the heroes cannot sustain a full push
- The heroes are spread out accross the map and cannot group together for a push
- The heroes desire to Push and Defend are very low

3.2.4 UpdateRoamDesire()

This function returns a floating point value between 0 and 1, as well as a unit handle of a hero on the enemy team. The number represents the team's desire to go gank the specified target (the unit handle) on the enemy team.

Factors that may influence the desire for a player to go gank:

- An enemy hero is alone in a lane
- It is early in the game. (Note: Ganking is important throughout the game, but it is more so in the early game since a team wishes to **get** an advantage as soon as possible, later in the game you wish to **maintain** the advantage, if you have it. Recovering from a deficit is explained next)
- If the team feels they are behind, in terms of kills or farm, the team will have a strong desire to go gank a target.

Generally, the team will attempt to gank a/the enemy core, but any hero is still better than none. Priority therefore falls first to cores and then to supports.

3.2.5 UpdateRoshanDesire()

This function simply returns a single floating point value between 0 and 1, and it represents the team's desire to go and attempt to defeat Roshan.

Factors influencing a team to go and attempt a Roshan kill:

- The enemy team has dead heroes. More dead heroes will result in a higher desire to go attempt to kill Roshan.
- The team wishes to push the Tier 3 tower in any given lane. This should result in a higher desire to kill Roshan.