Capture the flag

# RAPPEL DES REGLES

2 équippes, Rouge et Bleu (nommés B et R dans la suite de ce document) s’affrontent dans une arène.

Une équipe marque un point si elle arrive à amener le drapeau adverse dans sa base ET que son drapeau est présent.

Une fois le point marqué, le drapeau ennemi retourne au camp adverse.

Si une équipe possède le drapeau adverse et que son propre drapeau a aussi été volé, elle doit le récuppérer pour pouvoir marquer un point.

Si un porteur de drapeau est éliminé, le drapeau reste à la position à laquelle se trouvait le porteur.

Si un agent de la même couleur que le drapeau le ramasse, il retourne à la base instantanément.

Si c’est un agent de l’équipe adverse, il devient le nouveau porteur de drapeau.

Quand un agent est éliminé, son game object est désactivé et il sera réactivé au point de respawn de son équippe au bout de 8 secondes.

Quand un agent tire une roquette, il a un cooldown de 2 secondes avant de pouvoir tirer à nouveau. Les munitions sont en quantité inifinie.

# CODE

## Setup

### Comportement des bots

Vous avez la possibilité d’ajouter du code pour les bots et pour l’équippe.

Le setup se fait de la manière suivante :

Créez un game object vide et faites-en un prefab.

Dupliquez le script BotBehaviourBaseExample et donnez lui le nom que vous désirez avec la contrainte d’y mettre votre nom de famille (afin d’éviter d’avoir des duplicatas de noms de fichiers entre les projets de chacun). Ce fichier contient certaines informations et donne l’accès aux components du game object commun utilisé par les bots. Vous avez toute latitude de modifier ce fichier.

Ajoutez ce component au prefab préalablement créé.

Pour que votre code soit utilisé, sélectionnez l’objet « system » de la scène. Sur son component « GameMaster » trouvez le paramètre « Bot\_controller\_prefab » et ajoutez votre prefab sur une des deux cases (ou les deux si vous voulez le même comportement pour les deux équippes).

### Comportement des équippes

Suivez la même logique que pour le comportement des bots, en dupliquant cette fois-ci le script « TeamBehaviourBase » et en ajoutant votre prefab dans le paramètre « Team\_controller\_prefab » du GameMaster.

### Tester son IA contre celle de quelqu’un d’autre

Vous allez devoir exporter vos prefabs (celui de l’IA des bots et celui de l’IA de l’équippe).

Sélectionnez vos deux prefabs dans l’onglet projet puis cliquez dans la barre de menu sur Assets 🡪 Export Package. Vérifiez que la case « export dependencies est cochée et décochez les scripts qui ne sont pas spécifiques à votre projet. Cliquez sur export.

Une fois le package créé, il suffit de l’importer dans le projet de quelqu’un d’autre et de paramétrer le GameMaster pour utiliser vos prefabs et ceux de l’adversaire.

## Scripts

#### Bot

Le script Bot donne accès à l’ID, team\_ID et \_ennemy\_team\_ID pour ce bot.

Vous y trouverez également la fonction CanSeeObject qui permet de savoir si un objet peut être vu par ce bot. Vous êtes libre d’utiliser une autre méthode mais vous devez respecter les contraintes suivantes :

Un bot a une distance de vision infinie mais un angle de vision de 140° autour de son vecteur forward.

Est également présente la fonction ShootInDirection qui permet à votre bot de tirer. Il est obligatoire d’utiliser cette fonction pour tirer. Une fois un tir effectué, un cooldown est lancé pour empêcher de pouvoir tirer en rafales. Vous pouvez passer la variable « can\_shoot » en public pour pouvoir savoir si un bot est en cooldown.

Hormis cette information « can\_shoot », vous ne pouvez pas modifier ce fichier, seulement y accéder depuis votre BotBehaviour.

#### Team

Contient les infos de base de l’équippe (ID, position base, drapeau, position base ennemie).

Contient une fonction SendMessageToTeam qui permet de faire exécuter une fonction dans BotBehaviour.

Exemple :

Dans votre BotBehaviour, vous avez déclaré une fonction :

void SpotedEnemyFlag()

{

}

Vous pouvez, toujours dans ce même script, appeler :

team.SendMessageToTeam(« SpottedEnemyFlag ») ;

Cela executera la fonction précédente pour chaque bot de votre équippe.

#### GameMaster

C’est le script qui gère la partie. Il est responsable du spawn des différentes entités, de la gestion de la capture des drapeaux et du score.

Il vous donne accès aux informations suivantes :

GetFlagCarrierID : ID du porteur de drapeau d’une équippe donnée. Exemple : Si vous voulez savoir si un de vos bots possède le drapeau ennemi, il suffit de regarde si son ID correspond à celle du porteur du drapeau ennemi.

IsTeamFlagHome : permet de savoir si le drapeau d’une équippe est à sa base

GetScore : récuppération du score d’une équippe.

Le script « Bot » possède une variable permettant d’accéder au GameMaster.

## Contraintes

Un agent doit voir un objet pour pouvoir le prendre en compte (que ce soit un agent ennemi, un drapeau ou une roquette). Les agents d’une même équippe peuvent par contre s’échanger des informations.

Les objets ont des tags, « Flag » pour les drapeaux, « Bot » pour les bots, « Rocket » pour les roquettes.

Les paramètres des agents sont fixés dans le navmesh agent. Si vous devez faire tourner votre agent sur lui-même, pensez à désactiver le déplacement du NavMeshAgent d’abord, pour le réactiver ensuite. La vitesse de rotation est fixée à 360° par seconde.

Vous pouvez enregistrer la position et l’orientation des éléments qu’un bot voit (par exemple pour essayer de prédire sa position future).