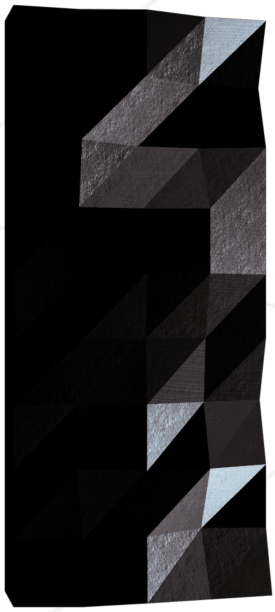

Piscine Unity - d02

Audio, Animation et Communication entre les scripts

Staff staff@staff.42.fr

Résumé: Ce document contient le sujet du jour 02 de la piscine Unity de 42.

Table des matières

Chapitre I

Préambule

Aujourd'hui vous allez vous transporter dans l'univers féerique et violent de Warcraft. Des humains, des orcs et surtout du sang. Vous apprendrez les rudiments et mécaniques du RTS.

Au programme donc, du combat, des unités devouées et prêtes à tout pour servir le grand maître que sera votre souris. Des villages pillés, réduits en ruines et en cendres. Pour vous accompagner dans votre quête rien de tel que le menestrel Youtube qui fera rugir les tambours de guerre pendant votre recherche :

- Une bande son qu'elle est bien
- Bon et parce que tout ca c'est quand même un peu violent, voila de quoi compenser un peu.

Bon courage a vous tous et que votre journée soit victorieuse.

Chapitre II

Consignes generales

- La piscine Unity est à faire entièrement et obligatoirement en `C#` uniquement. Pas de `Javascript/Unityscript`, de `Boo` ou autres horreurs.
- L'utilisation de fonctions ou de namespaces non autorises explicitement dans le header des exercices ou dans les regles de la journee sera considéré comme de la triche.
- Pour une utilisation optimale de Unity, vous devez travailler sur le `~/goinfre`, qui est en local sur le mac que vous utilisez. Pensez à bien récupérer vos projets avant de vous delog car le goinfre local est vidé régulièremment.
- Contrairement aux autres piscines, chaque journée ne demande pas un dossier `ex00/`, `ex01/`, ..., `exn/`. A la place pour la piscine `Unity`, vous devrez rendre votre dossier projet qui aura pour nom le nom de la journee : `d00/`, `d01/`, ... Toutefois, un dossier de projet contient par defaut un sous-dossiers inutile : le sous-dossier `"projet/Temp/"`. Assurez-vous de ne **JAMAIS** pusher ce dossier dans votre rendu.
- Au cas ou vous vous poseriez la question, il n'y a pas de norme imposée à `42` pour le `C#` pendant cette piscine Unity. Vous pouvez utiliser le style qui vous plaît sans restriction. Mais rappelez-vous qu'un code que votre peer-evaluateur ne peut pas lire est un code qu'elle ou il ne peut noter.
- Vous devez trier les assets de votre projet par dossier. Chaque dossier correspond à un et un seul type d'asset. Par exemple : `"Scripts/"`, `"Scenes/"`, `"Sprites/"`, `"Prefabs/"`, `"Sounds/"`, `"Models/"`, ...
- Assurez-vous de tester attentivement les prototypes fournis chaque jour. Ils vous aideront beaucoup dans la compréhension du sujet et du travail attendu.
- L'utilisation de l'Asset Store d'Unity est interdite hormis pour l'utilisation des Standard Assets. Vous êtes encouragés à utiliser les assets fournis chaque jour (quand nécessaire) ou à en chercher d'autres sur le net s'ils ne vous plaisent pas, sauf bien entendu pour les scripts car vous devez avoir écrit tout ce que vous rendez (hors scripts fournis par le staff, obviously). L'utilisation de l'Asset Store est partiellement interdite car quasiment tout le travail que vous avez à faire s'y trouve déjà sous une forme ou sous une autre.

- Pour les corrections à partir du d03 il vous sera demandé de builder les jeux pour les tester. `C'est le correcteur` qui doit build le jeu vous devez donc évidemment toujours push vos projets/sources. De ce fait votre projet doit correctement configuré pour le build. `Aucun` réglage de dernière minute ne doit être toléré.
- Important : Vous ne serez pas évalués par un programme, sauf si le contraire est explicite dans le sujet. Cela implique donc un certain degré de liberté dans la façon que vous choisissez de faire les exercices. Toutefois, gardez en tête les consignes de chaque exercice, et ne soyez pas FAINÉANTS, vous passeriez à coté de beaucoup de choses intéressantes.
- Ce n'est pas grave d'avoir des fichiers supplémentaires ou inutiles dans votre dossier de rendu. Vous pouvez choisir de séparer votre code en différents fichiers au lieu d'un seul, sauf si le header d'un exercice mentionne explicitement les fichiers à rendre. Un fichier ne doit définir qu'un et un seul comportement, pas de namespaces donc. Toute cette consigne ne s'applique bien evidement pas au sous-dossier `"projet/Temp/"` qui n'a pas le droit d'exister dans vos rendus.
- Lisez le sujet en entier avant de commencer. Vraiment, faîtes-le.
- Le sujet pourra être modifié jusqu'à 4h avant le rendu.
- Meme si le sujet d'un exercice est relativement court, ca vaut le coup de passer un peu de temps à comprendre parfaitement le travail attendu pour le faire au mieux.
- Parfois il vous sera demandé un soin particulier sur la qualité artistique de votre rendu. Dans ce cas, cela sera mentionné explicitement dans le sujet correspondant. N'hésitez alors pas à tester plein de choses différentes pour vous donner une idée des possibilités offertes par Unity.
- Par Odin, par Thor ! Refléchissez ! ! !

Chapitre III

Exercice 00 : Point and click

	Exercice : 00
Exercice 00 : Point and click	
Dossier de rendu : $ex00/$	
Fichiers à rendre : `Le dossier de votre projet.`	
Fonctions interdites : `Aucune`	

Dans ce premier exercice vous devez mettre en place un personnage sur la map fournie dans les assets. On doit pouvoir le déplacer en cliquant sur un endroit sur la map qui fera office de destination. Il ne doit pas se téleporter mais avancer jusqu'à sa destination. Le sprite doit également être orienté en direction de sa destination.

Rendez votre personnage un peu plus vivant en lui ajoutant un son lorsqu'il commence à se déplacer. Ajoutez lui également une animation de marche qui ne se déclanchera que lorsqu'il marche donc.

Chapitre IV

Exercice 01 : Plus on est de fous

	Exercice : 01
Exercice 01 : Plus on est de fous	
Dossier de rendu : $ex01/$	
Fichiers à rendre : `Le dossier de votre projet.`	
Fonctions interdites : `Aucune`	

Un personnage c'est bien, plusieurs c'est mieux. Vous allez donc mettre en place plusieurs personnages (avec les memes caractéristiques) et faire en sorte qu'on puisse les selectionner avant de leur donner un ordre.

On doit donc pouvoir cliquer gauche sur un personnage pour le selectionner. Ceci l'ajoute à la selection active. Si un ou des personnages sont déjà présent dans la selection active, alors elle est vidée. On doit pouvoir néanmoins rajouter un personnage sans vider la selection active en cliquant gauche avec la touche `control` enfoncée.

Si on clique gauche ailleurs que sur un personnage alors toute la selection va en direction du point visé comme lors de l'exercice précedent. Si on clique droit la selection est vidée.

Chapitre V

Exercice 02 : Batiments et villages

	Exercice : 02
Exercice 02 : Batiments et villages	
Dossier de rendu : $ex02/$	
Fichiers à rendre : `Le dossier de votre projet.`	
Fonctions interdites : `Aucune`	

Le but de cet exercice va être de mettre en place deux villages ennemis : l'un Orc et l'autre Humain. Les deux villages sont identiques, toutefois les sprites les représentant seront bien entendus différents.

Un village est composé d'un hôtel de ville et de 4 bâtiments de votre choix. Le nombre de points de vie des bâtiments est à votre discretion, toutefois l'hôtel de ville doit en avoir significativement plus que les 4 autres.

L'hôtel de ville de chaque village va spawner un nouveau guerrier humain ou orc, selon sa race, toutes les 10 secondes devant sa porte. Ce n'est pas grave si les personnage se superposent.

Les guerriers du joueur doivent pouvoir être déplacés conformement à l'exercice précédent. Les guerriers ennemis restent statiques pour l'instant puisque nous aborderons les combats dans le prochain exercice.

Chapitre VI

Exercice 03 : De l'action

	Exercice : 03
Exercice 03 : De l'action	
Dossier de rendu : $ex03/$	
Fichiers à rendre : `Le dossier de votre projet.`	
Fonctions interdites : `Aucune`	

Tout est en place il ne manque plus que d'un peu d'action. Vos personnages vont donc enfin pouvoir taper tout ce qui bouge, voire ce qui ne bouge pas.

Ajoutez donc la possibilité de pouvoir attaquer une cible. On doit pouvoir cliquer sur une cible pour faire attaquer toute la selection active. Si les personnages sont trop loins pour attaquer ils s'approcheront jusqu'à une distance correcte pour le faire. Si la cible s'echappe ils devront la suivre à moins que vous ne lui donniez un ordre contraire.

Vos personnages peuvent attaquer les unités et les bâtiments ennemis. N'oubliez pas d'ajouter un son et une animation pour le combat. Lorsqu'une unité ou un bâtiment n'a plus de vie, celui-ci disparait accompagné d'un son correspondant. La partie s'arrête lorsqu'un des deux hôtels de ville est détruit.

Attention : à chaque fois qu'un bâtiment qui n'est pas un hotel de ville est détruit, le temps de spawn d'unité de l'hôtel de ville du joueur concerné augmente de 2,5 secondes. Cela signifie donc que s'il ne reste que son hôtel de ville à un joueur, les unités spawneront toutes les 20 secondes.

Ajoutez également des informations dans la console pour savoir qui est attaqué et combien de vie il lui reste.

```
Orc Unit [50/100]HP has been attacked.
```

Faites également en sorte d'afficher un message lorsque la partie est terminée :

```
The Human Team wins.
```

Pour cet exercice, le camp ennemi est toujours inerte, ne perdez donc pas de temps à vouloir scripter une petite IA. Cela sera pour l'exercice suivant.

Chapitre VII

Exercice 04 : Joueur vs IA

	Exercice : 04
Exercice 04 : Joueur vs IA	
Dossier de rendu : $ex04/$	
Fichiers à rendre : `Le dossier de votre projet.`	
Fonctions interdites : `Aucune`	

Maintenant que votre jeu est prêt, il est temps de scripter un adversaire. Votre opposant va donc vouloir détruire tout ce qui est sur son passage, car il vous déteste personnellement. Sa première priorité sera donc de vous raser, et en particulier votre hôtel de ville. Cependant si ses unités se trouvent à proximité des votres, il ne pourra s'empêcher de laisser tomber son objectif un instant pour les défoncer.

De plus, si l'hôtel de ville de l'IA est attaqué par le joueur humain, l'IA rappelle des unités pour le défendre.

Les consignes précédentes forment le minimum acceptable pour votre IA. Une fois que ces comportements de base sont implementés correctement dans votre rendu, vous devez ameliorer votre IA pour la rendre moins stupide et plus intéressante à affronter.