[NdT : Ceci est une traduction française libre et non officielle du "Factorio Friday Facts" n°244]

Pluriels locaux et progrès sur la modernisation

Un nouveau développeur Python (par Klonan)

Les utilisateurs mobiles peuvent voir que le site Web est beaucoup plus facile à lire aujourd'hui, tout cela grâce à notre nouveau développeur Python : Sanqui. En plus de faire en sorte que le site Web soit plus convivial pour les appareils mobiles, nous avons beaucoup de tâches en attente à traiter sur lesquelles il commencera bientôt à travailler.

Sa première grande tâche est d'accélérer et d'optimiser le serveur de correspondance, pour lequel il progresse déjà. Une réécriture plus importante à long terme est en cours, mais au cours de la dernière semaine, il a déjà réduit de beaucoup la lenteur et les délais que les gens avaient pendant les périodes de pointe. Il reprendra en charge la gestion de la base de données et l'administration Web de HanziQ, ainsi que le nettoyage de notre base de code, la maintenance de nos services Web et le développement de fonctionnalités pour le portail des mods.

Pluriels locaux (par kovarex)

Des pluriels locaux sont parfois nécessaires dans Factorio. Par exemple, lorsqu'une heure est spécifiée, cela peut être soit 1 minute et 5 secondes, soit 5 minutes et 1 seconde, ainsi il doit donc y avoir une manière pour sélectionner le mot correct en fonction du nombre.

Pour la 0.16, nous avons ce système personnalisé très spécifique pour déterminer certaines traductions via 3 formes différentes, tandis qu'un code particulier sélectionne la bonne en fonction du nombre correspondant :

En anglais [NdT: et en français] :

```
minute1=minute
minute2-4=minutes
minute5+=minutes
```

Les traducteurs pourraient être très surpris, pourquoi avons-nous besoin de deux traductions différentes pour le pluriel de "minute" ? Les Tchèques qui liront comprendront, car la version locale tchèque du code ressemble à ceci :

```
minute1 = minuta
minute2-4 = minuty
minute5 + = minut
```

Oui, nous (les Tchèques) avons un pluriel différent pour le compte 2,3,4, que les autres pluriels.

Le problème est, que nous avons finalement découvert qu'il existe d'autres langues que l'anglais et le tchèque (!) [NdT : oui, le français par exemple] et que les possibilités de pluriels dans d'autres langues ne sont pas aussi simples que les anglophones ont tendance à le penser. Pour référence, voici le résumé des formes plurielles les plus communes.

C'est clair qu'avoir une clé de traduction différente pour chaque forme possible, pour chaque langue, serait trop fou, nous avons donc inventé quelque chose de plus générique. Avec la 0.17, la traduction de X minutes (le nombre est maintenant inclus dans la traduction pour pouvoir le mettre à une position différente, ou ajouter un autre mot quelque part) ressemblera à ceci:
Anglais:

```
minutes=__1__ __plural_for_parameter_1__[1]__minute__[rest]__minutes__
```

Tchèque:

```
minutes=__1__ __plural_for_parameter_1__[1]__minuta__[2-4]__minuty__[rest]__minut__
```

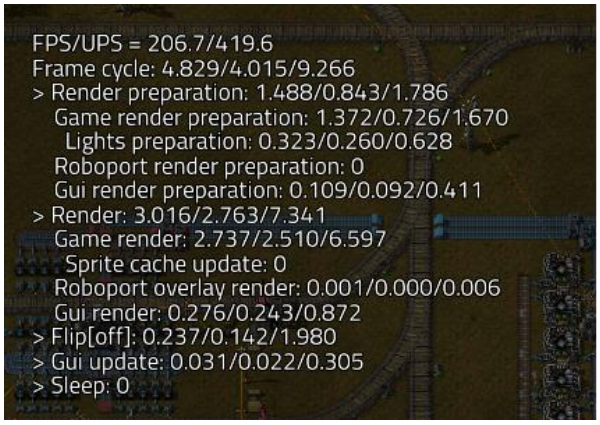
Roumain:

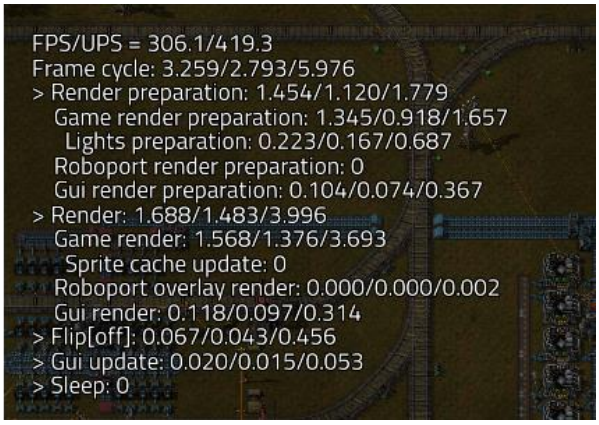
```
minutes=__1__ __plural_for_parameter_1__[1]__minut__[ends in 01-19]__minute__[rest]
__de minute__
```

etc.

Progrès sur la modernisation du moteur (par Posila)

La réécriture du rendu général se passe bien. Nous sommes au point où le jeu semble exactement identique (avec des corrections mineures sur le rendu de ligne) à notre principale branche 0.17, et la seule chose importante qui manque est la gestion correcte des erreurs - par exemple, nous ne voulons pas que le jeu se termine si une commande de capture d'écran échoue pour une raison quelconque. J'ai été surpris du nombre de choses que nous avons dû réécrire, mais nous avons pris notre temps, et je pense que cela en vaut déjà la peine. Le rendu du jeu est nettement meilleur (du moins du côté du CPU pour l'instant – plus à ce sujet dans de futurs "Friday Facts") dans OpenGL et DirectX.





En parlant de cela, certaines personnes se demandaient pourquoi nous nous acharnons avec le DirectX, et n'utilisons pas uniquement l'OpenGL partout. La raison en est que Windows est toujours le principal système d'exploitation pour les jeux sur PC et que, pour une raison inconnue, nous ne pouvons pas tirer autant de performance d'OpenGL que de DirectX (nous voyons environ 30% de temps en plus passé en appels OGL comparé à D3D). Il semble également que la qualité du pilote OpenGL soit la même partout. Nous nous sommes déjà heurtés au problème où notre première implémentation d'atlas de sprites de construction [NdT: recueil des modèles] fonctionnait bien sur Windows et macOS, mais était tellement inefficace sous Linux qu'il a fallu plus d'une demi-heure pour que le jeu se charge.

Nous essayons actuellement de concevoir le nouveau rendu autour de l'architecture des API actuelles - Vulkan / Metal / DirectX 12 - et à l'avenir nous aimerions créer des processus Vulkan et Metal, mais nous nous concentrons d'abord sur les anciennes API vu qu'une grande partie de nos joueurs actuels n'a pas de matériel compatible Vulkan. En utilisant OpenGL 3.3 Core et le DirectX 11 niveau 10.0, nous devrions couvrir 99% de nos joueurs actuels (selon les statistiques provenant de Steam). Ces API devraient être supportées par n'importe quel GPU dédié fabriqué au cours des 10 dernières années, et Intel HD Graphics intégré, depuis Sandy Bridge. Nous laisserons toujours 1% des joueurs sur le côté, donc ils seront bloqués à la 0.16, comme lorsque nous avons arrêté le support 32 bits, mais tout le monde devrait bénéficier de meilleures performances et, espérons-le, d'un jeu plus beau.

Comme toujours, faites-nous savoir ce que vous en pensez sur notre forum.