Friday Facts #277 - Des nouvelles sur l'Interface Graphique d'Utilisateur

par Kovarex et Klonan le 2019-01-11.

Des nouvelles sur l'Interface Graphique d'Utilisateur

Ceci est la suite du dernier rapport de situation de FFF-269. Comme ce n'est pas une surprise, le plus gros des travaux de la version 0.17 est l'interface graphique. J'aime croire que nous avons beaucoup appris des pièges du processus de création collaborative de l'interface graphique.

- Deux ou trois personnes ont commencé à discuter de ce qui pourrait être cool de changer dans l'interface graphique particulière. Certaines personnes se sont jointes au hasard et ont quitté la discussion en cours. Les arguments pour écarter certaines idées doivent être répétés encore et encore. Ensuite, la discussion est terminée à cause de quelque chose.
- Une semaine plus tard, les gens ont recommencé à en reparler, la plupart d'entre eux ont oublié la plupart des informations ou en ont discuté avec différentes personnes. Ils supposent donc que certains détails des modifications sont compris par tout le monde, alors qu'ils ne le sont pas.
- Ils parviennent à un accord sur la manière dont cela devrait être fait.
- Ils en discutent au hasard une semaine plus tard et se rendent compte qu'ils avaient des idées complètement différentes sur la façon dont cela devrait être fait. Ils ne les ont tout simplement pas articulés de manière précise. Les deux sont un peu fâchés de devoir rouvrir et renégocier le sujet.
- Quelqu'un commence à implémenter l'interface graphique, mais à mi-chemin, il est découvert qu'il y avait un autre niveau de malentendu lors de la spécification de la façon dont le travail devait être effectué.

Etant donné que de nombreuses interfaces graphiques sont pensées et travaillées en parallèle, ces situations se chevauchent et amplifient les problèmes de confusion dans nos têtes à propos de ce sur quoi nous nous sommes mis d'accord.
Heureusement, nous avons fini par comprendre que cela ne pouvait pas être fait de la sorte, et comme il y a beaucoup de travail dans l'interface graphique, nous devons créer un processus. Ça marche ainsi :

- Premièrement, il y a une discussion générale sur l'interface graphique, tous les membres de l'équipe peuvent partager leurs idées.
- Kovarex + Twinsen sont assis seuls dans le bureau et discutent pendant un certain temps (peut-être des heures) de tous les avantages et inconvénients de la façon dont les choses doivent être faites et concluent un accord.
- Twinsen écrit de manière détaillée un document UX détaillé sur l'interface graphique contenant la structure et, plus important encore, le comportement.
- Twinsen + Kovarex discute du document UX et propose des modifications jusqu'à ce qu'ils s'accordent sur la version finale.
- Albert + Aleš prennent le document UX et crée une maquette d'interface utilisateur basée sur celui-ci.

- Kovarex + Twinsen + Albert s'accordent sur la maquette de l'interface utilisateur ou proposent des modifications.
- Quelqu'un est affecté à l'implémentation de l'interface graphique basée sur le document et la maquette d'interface utilisateur.
- Kovarex vérifie que la mise en œuvre est correcte et signale certaines incohérences qu'il peut constater. Une partie de cette étape consiste à nous assurer que nous partageons autant de styles et de codes d'interface graphique que possible entre différentes interfaces graphiques.
- Kovarex + Albert jette un dernier coup d'œil à la mise en œuvre et corrige les derniers détails jusqu'à ce qu'ils s'accordent à dire que l'écran est complètement terminé.

Avoir les maquettes de documents UX / UI toujours disponibles s'est avéré un énorme gain de temps. Non seulement cela nous aide à résoudre les problèmes de communication, mais nous n'avons pas non plus à mémoriser et reformuler les décisions d'il y a quelque temps, car nous pouvons simplement ouvrir le document et voir ce sur quoi nous nous sommes mis d'accord et continuer instantanément là où nous l'avions laissé.
Une bonne partie de cette stricte démarche est que nous avons maintenant une meilleure connaissance de l'état d'avancement des travaux.
Voici les écrans d'interface graphique que nous espérons livrer pour 0.17:

	UX général	UX brouillon	UX revu	UI maquette	UI maquette	implémentation brouillon	implémentation revue	Revue finale
Chargement	✓	✓	✓	✓	✓	✓	✓	✓
Sauvegarde	✓	✓	✓	✓	✓	✓	✓	✓
Options graphiques	✓	✓	✓	✓	✓	✓	✓	✓
Options de contrôle	✓	✓	✓	✓	✓	✓	✓	✓
Options du son	✓	✓	✓	✓	✓	✓	✓	✓
Options d'interface	✓	✓	✓	✓	✓	✓	✓	✓
Autres options	✓	✓	✓	✓	✓	✓	✓	✓
Générateur de carte	✓	✓	✓	✓	✓	✓	✓	✓
IGU : Technologies	✓	✓	✓	✓	✓	✓	✗	✗
Infobulle : Technologies	✓	✓	✓	✗	✗	✗	✗	✗
Infobulle : recette/item/entité	✓	✓	✓	✓	✗	✗	✗	✗
Bar d'action	✓	✓	✓	✓	✓	✓	✗	✗
IGU : Train	✓	✓	✓	✗	✗	✗	✗	✗
Gérer/Installer : Mods	✓	✓	✓	✓	✓	✗	✗	✗
Écran principal	✓	✓	✓	✗	✗	✗	✗	✗
Navigateur de recettes	✓	✓	✓	✗	✗	✗	✗	✗
Écran personnage	✓	✗	✗	✗	✗	✗	✗	✗
Structure du menu	✓	✓	✓	✗	✗	✗	✗	✗
Nouvel partie	✓	✓	✗	✗	✗	✗	✗	✗
Panneau d'aide	✓	✓	✓	✗	✗	✗	✗	✗
Séléctionneur d'icônes de chat	✓	✗	✗	✗	✗	✗	✗	✗
Bibliothèque des blueprints	✓	✗	✗	✗	✗	✗	✗	✗

Vous pouvez voir, mais il reste encore beaucoup à faire, mais le travail est susceptible d'accélérer. La conclusion est que 0,17 expérimental en janvier est possible, mais cela pourrait aussi être en février :).

Les petits détails

Par kovarex
De plus, une des raisons pour lesquelles le travail avance plus lentement est que, puisque nous élaborons des versions finales de tout, nous voulons le peaufiner avant de le considérer comme terminé, car nous n'aurons pas le temps de revenir à la tâche.
Par exemple, dans chaque écran de paramètres, nous avons maintenant le bouton "réinitialiser par défaut". La première étape logique consistait à n'activer le bouton que lorsque certaines des options étaient différentes des options par défaut. Mais la prochaine étape logique consistait à informer l'utilisateur, quels paramètres seraient modifiés lorsqu'il utiliserait le bouton de réinitialisation. Nous soulignons donc simplement tous les paramètres autres que ceux par défaut lorsque vous passez la souris

sur le bouton de réinitialisation. Cela fait du bien, car vous avez tout à coup un retour instantané sur ce que vous pouvez attendre de l'action.

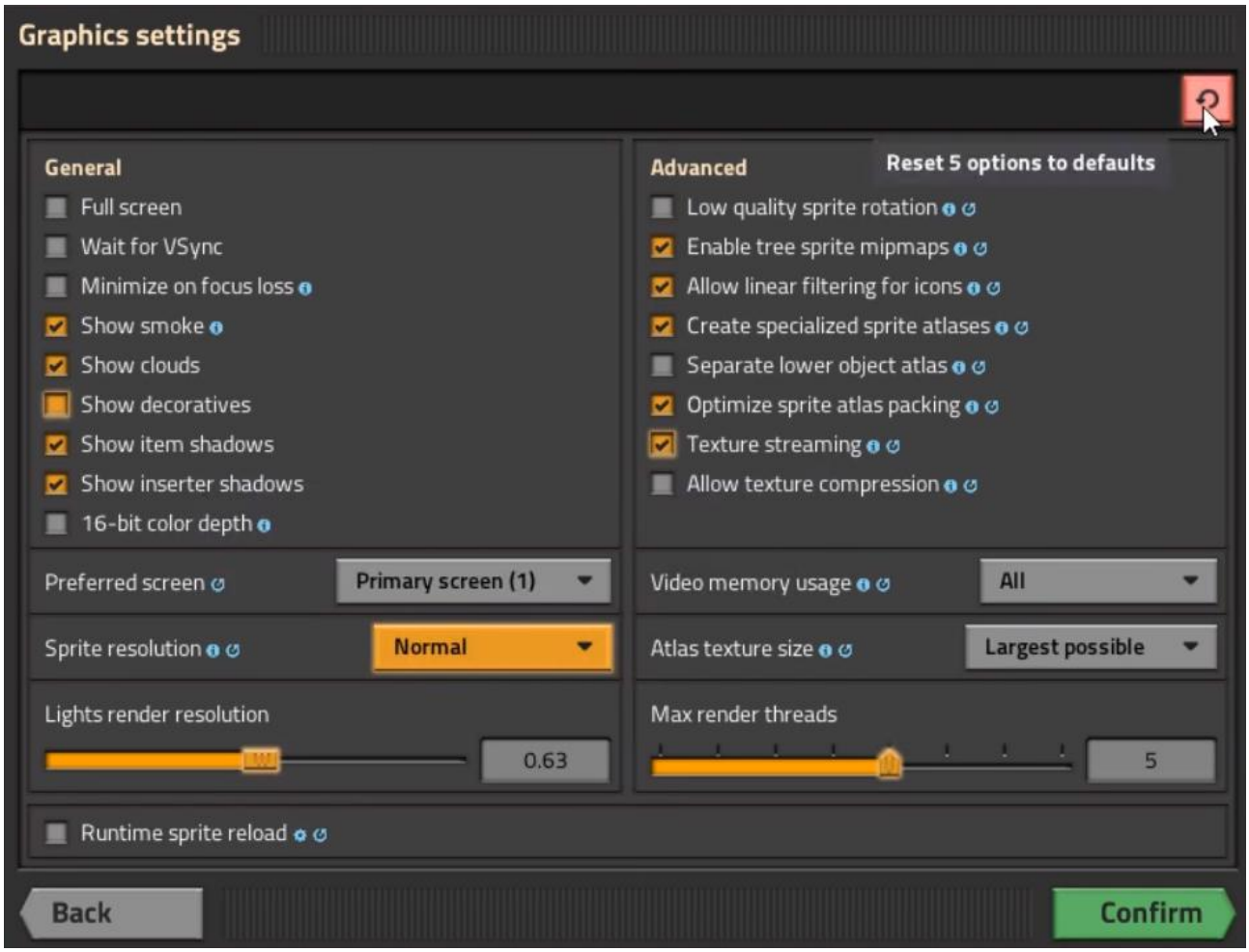


Figure 1: Cliquez pour visionner la vidéo

Je crois comprendre que passer trop de temps sur l'interface graphique de configuration n'a peut-être pas l'air d'être une bonne idée, car elle n'est pas très utilisé et les écrans de jeu sont plus importants, mais bon nombre de ces principes que nous réalisons en cours de route seront utiles lorsqu'il faudra concevoir des modifications pour l'interface graphique du jeu.

Le problème d'échelle et la solution

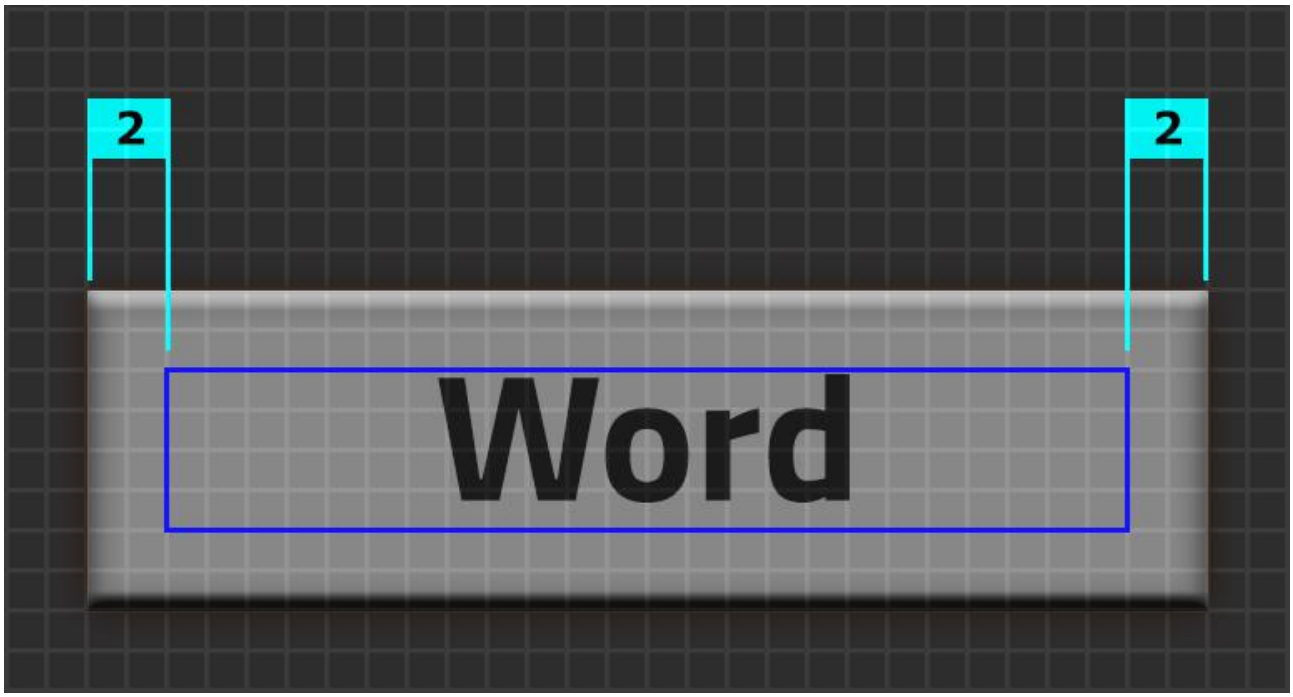
par Kovarex

L'un des nombreux objectifs de la réécriture de l'interface graphique était de s'assurer que celle-ci soit correcte dans toutes les valeurs d'échelle de l'interface utilisateur possibles. Par correct, nous voulons surtout dire que les proportions de tout restent les mêmes, donc elles sont juste plus petites, mais pas déformées de manière étrange. Cela peut sembler une chose simple à faire, mais ce n'est pas parce que toutes les valeurs de l'interface graphique (tailles, positions des widgets, marges, etc.) sont des valeurs entières, car à la fin, chaque élément doit être sur un pixel spécifique aussi longtemps comme nous voulons qu'il soit clair et net.

Imaginez que vous ayez un écart de 1 pixel entre deux boutons de 20 pixels et que vous vouliez qu'il s'affiche à 120%. Les boutons sont agrandis à 24 pixels, mais l'écart reste soit de 1 pixel, soit de 2, mais les proportions de la présentation changent.

Pour résoudre ce problème (et d'autres problèmes), nous avons décidé d'utiliser ce que nous appelons

des "modules". Un module mesure 4 pixels en échelle standard (100%), et presque tout est une multiplication de modules (tailles, positions, rembourrages, etc.). De plus, nous avons limité les valeurs d'échelle possibles à des multiples de 25% (de 75% à 200%). Cela signifie que la taille d'un module peut être différente (de 3 pixels à 75% à 8 pixels à 200%), mais les proportions de chaque élément restent identiques, de sorte que le résultat est correct.



Cela fonctionne assez bien jusqu'à présent, mais cela pose un autre problème pour 0.17. Je ne sais pas si quelqu'un l'a remarqué, mais les emplacements d'éléments (filtres d'inventaire / logistique, d'artisanat, etc.) ont été intentionnellement réduits davantage que d'autres éléments de l'interface utilisateur. La raison en est que les icônes 32x32 que nous avons pour tous les articles / fluides / recettes ne semblent pas bonnes si elles sont trop étirées.

Figure 2: 32x32 icônes étendues à 200% ne fonctionnent pas bien.

Mais comme nous avons dû abolir cette règle spéciale pour 0,17, nous devons nous assurer que toutes les icônes 32x32 (285 éléments, 27 signaux, fluides et plus) devront être fournies en résolution 64x64, de sorte que toutes les interfaces utilisateur prises en charge auront fière allure. Cela nécessitera beaucoup de travail, mais puisque nous rendons les icônes à partir de modèles 3D de toute façon, cela devrait être gérable. Nous allons probablement retarder les icônes haute résolution à 0,17 pendant la phase expérimentale.

Défense procédurale

Par Klonan
Nous avons le scénario de Wave Defense dans le jeu depuis quelques années maintenant et au fil des ans, j'ai recueilli de nombreuses réactions. Un problème que j'ai déterminé est que le scénario manque vraiment de rejouabilité, en raison de la carte corrigée. Puisque la carte ne change pas du tout, une fois que vous avez un ensemble de blueprints et une tactique qui fonctionne, les répétitions sont pour la plupart ennuyeuses.
Avec les modifications récentes et l'excellent travail de TOGoS, la génération de carte procédurale fonctionne vraiment bien et est très fiable pour un ensemble donné de paramètres. Cela m'a donné l'idée qu'il serait peut-être possible de faire en sorte que le Wave Defense utilise une nouvelle carte à chaque fois. J'ai expérimenté certaines valeurs prédéfinies et je pense que cela fonctionnera très bien. Cependant, j'ai une certaine indécision en moi, et il y a plusieurs avantages et inconvénients entre des cartes personnalisées faites à la main et des mondes générés aléatoirement.

0.1 Avantages d'une carte sur mesure

- La carte peut être spécifiquement conçue et modifiée pour une meilleure expérience. Je peux tester et analyser la façon dont les biters se déplacent sur la carte, ajuster le placement des arbres, des ressources, régler la difficulté, etc.
- Nous n'avons pas à nous soucier des modifications du moteur de génération de cartes qui brisent le scénario.
- Il s'agit d'une expérience fiable pour tous les joueurs. Les utilisateurs peuvent partager des astuces, des conceptions et des tactiques spécifiques, plus spécifiques à la carte.
-

0.2 Avantages des cartes procédurales

- Le scénario a une rejouabilité beaucoup plus grande.
- Nous n'avons pas à nous soucier de la migration des données cartographiques, des tuiles, des entités, etc. vers des versions plus récentes.
- Toute amélioration de la génération de carte sera reflétée dans le scénario.
- Les gens ne peuvent pas trafiquer le scénario à l'aide des Blueprints d'autres gens.
- Nous pouvons ajouter des options de configuration pour les personnes qui souhaitent personnaliser l'expérience.
- Il est facile d'ajouter un support pour que les serveurs continuent à fonctionner après la victoire / défaite.

• Je suis donc en train de faire les changements pour vérifier si les procédures peuvent fonctionner, et cela ne devrait pas prendre trop de temps, car la plupart des scripts de scénario resteront les mêmes. Je voulais demander quelques commentaires et réflexions de la part de la communauté: Est-ce que beaucoup d'entre vous ont joué dans la Wave Defense? Pensez-vous qu'une carte aléatoire serait plus amusante? Pensez-vous que vous y joueriez davantage si la carte était différente à chaque fois?

Comme toujours, nous vous invitons à nous dire ce que vous pensez sur notre forum.