Ubisoft-Gamagora

Décembre 2015

**TP Gameplay répliqués**

Vous arrivez sur un projet de jeu point & clic et reprenez le travail d’un GPP.

Votre objectif est de faire la réplication du joueur, des NPC et des ingrédients gameplay.

Le Lead GPP, débordé, vous explique que le moteur est basé sur la librairie SFML et que si vous voulez savoir comment répliquer des variables, vous pouvez regarder comment fonctionne le spawn du personnage principal, qui lui est répliqué. Il vous indique la méthode Game::OnNetworkData() et s’excuse d’avance de la rareté des commentaires dans le code.

Il vous demande de faire le développement réseau de plusieurs sujets. Voici vos objectifs :

1. **Faire la réplication des déplacements du personnage principal**

A ce stade du projet, cette partie-là n’est pas critique et vous pouvez faire la réplication sans vous soucier des optimisations et de la latence.

1. Au vu de l’existant, quelle est la technique de réplication que vous allez mettre en place ? Faites les schémas et diagramme de séquences nécessaires à votre explication
2. Quelles questions sur la partie Réseau de SFML vous posez vous ? Notez les réponses que vous avez pu avoir.
3. Quels sont les outils à votre disposition ?
4. Faites le développement et n’oubliez pas de commenter le code. Pensez également à minimiser le volume d’informations envoyé.
5. **Faire la réplication de la bombe**

Il s’agit maintenant de faire la réplication de la bombe, au centre du gameplay du jeu. Comme le jeu est destiné au public le plus large, vous devez vous assurer que cette feature clef supporte les latences extrêmes des utilisateurs les plus éloignés des DSLAM.

1. Configurez NEWT pour avoir 3000 ms de latence entre vos deux PC et vous aider à développer. Utiliser 150ms de latence (aller-simple) pour valider la feature.
2. Faite la réplication de la bombe et assurez-vous qu’elle explose au même endroit, et en même temps chez tous les peers. Au vu de l’existant, quelle est la technique de réplication que vous allez mettre en place ?
3. Assurez-vous que le JIP est fonctionnel, pour la bombe et pour les personnages

Les dev testeurs vous remontent un bug ; ils arrivent avec la latence à faire en sorte que le réplica d’un joueur, visuellement dans la zone d’explosion de la bombe, ne meurt pas. Le master étant lui, en dehors de la zone.

1. Expliquez-en la raison (diagrammes des masters et réplica bienvenus)
2. Proposez une solution, en expliquant pourquoi celle-ci plutôt qu’une autre.
3. Quel feedback pensez-vous nécessaire d’ajouter pour limiter la frustration des joueurs ?
4. Faites la dev
5. **Gérer le Scoring**

Le Game Designer en charge du scoring a eu l’idée géniale d’ajouter du gameplay sur la bombe. Il vous demande de prototyper son gameplay. Ce qu’il veut c’est qu’un kill de NPC rapporte 10 points. Son expérience lui permet d’anticiper les problèmes, et pour éviter de frustrer les joueurs, il vous demande de partager (en parts égales) les points de kill entre tous les joueurs qui feront des dégâts dans un certain laps de temps. Au vu de la vitesse de déflagration, vous vous mettez d’accord sur 450 ms.

1. Spawnez un ennemi dans la map, en utilisant le code existant. Complétez-le pour que l’ennemi soit répliqué. Ce NPC peut rester immobile pour votre prototype.

Votre lead (qui sait qu’il aura à gérer les dégâts directs entre joueurs), vous demande de créer un manager dédié à la gestion des dégâts. Ce manager devra centraliser les demandes et gérer le scoring.

1. Proposez une architecture pour ce manager. Faites les schémas que vous jugerez utile pour expliquer votre système.
2. Faites le développement du manager et n’oubliez pas de commenter le code.
3. Validez la feature scoring en plaçant les deux joueurs à côté du même NPC et en activant la bombe quasi simultanément. Capturez une vidéo (par exemple avec FRAPS). Servez-vous en pour valider le fonctionnement.
4. Pour une meilleure compréhension, vous pouvez afficher le scoring ou le sortir en log.
5. Activez NEWT, et testez avec plusieurs latences (allez-simple) : 80ms, 150ms, 250ms, 1000ms. Que constatez-vous dans chaque cas? Comment l’expliquez-vous? Est-ce problématique et pourquoi?
6. Est-ce que votre manager est fonctionnel en cas de paquets perdus ? Expliquez pourquoi ? S’il vous manque des informations allez au paragraphe 1b.

Le Game designer passe tester votre prototype et croit remarquer qu’avec 150ms de latence (aller simple) le NPC ne meurt pas instantanément chez les joueurs.

* 1. Est-ce le cas ? Et pourquoi ?
  2. Si oui, comment pouvez-vous corriger cette impression ?