Excution de jeu réseau sous Unreal Engine

À propos

Unreal Engine impose l'architecture client / serveur sous laquelle le serveur exécute la simulation autoritaire et la réplique à tous les clients.

3 modes réseau sont offerts:

1. Client dédié: Instance de jeu représentant un joueur devant être connecté à un serveur
2. Serveur dédié: Instance de jeu sans joueur local ni rendu servant la partie aux clients
3. Listen server: Instance client et serveur capable d'être à la fois joueur et serveur pour d'autres clients

Unreal Engine offre plusieurs méthodes pour exécuter un jeu réseau. Aucune méthode n'est parfaite. Le choix de la méthode dépend du mode de jeu désiré, besoin de fidélité, temps de mise en place et utilisation visée.

Play In Editor (PIE)

Exécution des instances de jeu à même l'editeur actif

Support	Effort de mise en place	Fidélité	Utilisation principale
Listen Server Dedicated Server	Minimal	Basse	Développement

Étapes

- Ouvrir le projet dans Unreal Editor
- Sous le menu contextuel de Play
    - Choisir Mode "New Editor Window (PIE)"
    - Choisir le nombre de joueurs sous "Number of Players"
    - Choisir NetMode "Play as Listen Server" pour listen-server ou "Play as Client" pour dedicated server
- Appuyer sur le bouton Play
- Si "Play as Listen Server" est choisi, la première instance est le listen-server avec les autres instances sont les clients connectés
- Si "Play as Client" est choisi, toutes les instances seront des clients connectés à une autre instance dedicated server caché

Notes:

- Rapide: Ne nécessite pas de packager ni de cooker les assets
- Debugage aisé: Éditeur connecté à instance (World Outliner, Blueprint Debugger et autres outils)
- Lent: Exécution plus lente puisque chaque instances partagent le même processus et cookent "on the fly" les assets

- Basse fidélité: Plusieurs UGameInstance et UWorld sont disponnibles puisque plusieurs instances s'éxécutent au seins du même processus (peut être la source de bugs)

Editor Standalone

Exécution des instances de jeu sous-tendue par des editeurs dédiés

Support	Effort de mise en place	Fidélité	Utilisation principale
Listen Server Dedicated Server	Minimal	Moyenne	Développement

Méthode#1 - Étapes à partir de l'Editor

- Ouvrir le projet dans Unreal Editor
- Sous le menu contextuel de Play
  - Choisir Mode "Standalone Game"
  - Choisir le nombre de joueurs sous "Number of Players"
  - Choisir NetMode "Play as Listen Server" pour listen-server ou "Play as Client" pour dedicated server
- Appuyer sur le bouton Play
- Si "Play as Listen Server" est choisi, la première instance est le listen-server avec les autres instances sont les clients connectés
- Si "Play as Client" est choisi, toutes les instances seront des clients connectés à une autre instance dedicated server

Méthode#2 - Étapes à partir de la ligne de commande

- Localiser l'exécutable UnrealEditor.exe
  - Si moteur précompilé, voir `[LauncherInstall]/[VersionNumber]/Engine/Binaries/Win64/UE4Editor.exe`
  - Si moteur bâti des sources, voir `[SourceRepoPath]/Engine/Binaries/Win64/UnrealEditor.exe`
- Localiser le projet a exécuter
  - Trouver le fichier uprojet
- Lancer un listen-server client
  - Invoquer `[PathToEditor]/UnrealEditor.exe [PathToProject].uproject 127.0.0.1 -game -log -windowed -ResX=800 -ResY=450`
  - Voir exemple launch_editor_client.bat
- Lancer un listen-server server
  - Invoquer `[PathToEditor]/UnrealEditor.exe [PathToProject].uproject ?listen -game -log -windowed -ResX=800 -ResY=450`
  - Voir exemple launch_editor_server.bat
- Lancer un dedicated server

    - Invoquer `[PathToEditor]/UnrealEditor.exe [PathToProject].uproject ?listen -server -log`
    - Voir exemple launch_editor_dedicated_server.bat

Notes:

- Rapide: Ne nécessite pas de packager ni de cooker les assets
- Lent: Exécution plus lente puisque les assets sont cooked "on the fly"
- Fidélité moyenne: Exécutable unique ne permettant pas d'exclure le code client du dedicated serveur et d'exclure le code serveur du dedicated client (risque d'exécution accidentelle)

Packaged

Exécution des instances de jeu dans des excutables indépendants avec Unreal Engine précompilé

Support	Effort de mise en place	Fidélité	Utilisation principale
Listen-server	Moyen	Élevé	Test Démonstration Livraison

Étapes

- Ouvrir le projet dans Unreal Editor
- Packager le listen-server à partir de Platforms -> Windows
    - Choisir Binary Configuration
        - Debug Game (code pas optimisé, aisé à débugger)
        - Development (code un peu optimisé, débuggable)
        - Test (code optimisé, difficile a débugger, accès aux outils)
        - Shipping (code optimisé, très difficile à débugger, outils retirés)
    - Choisir Build Target: [ProjectName]
    - Cliquer sur Package Project
    - Sélectionner la destination pour le package (Suggéré: [workspace]/Packaged)
    - Application va être compilé, les assets cookés et le tout sera packagé (quelques dizaines de minutes à plusieurs heures)
    - Localiser l'exécutable à [PackageDestination]/Windows/[ProjectName].exe
- Lancer un listen-server packaged client
    - Invoquer `[PathToPackage]/[ProjectName].exe 127.0.0.1 -log -windowed -ResX=800 -ResY=450`
    - Voir exemple launch_packaged_listenserver_client.bat
- Lancer un listen-server packaged server
    - Invoquer `[PathToPackage]/[ProjectName].exe ?listen -log -windowed -ResX=800 -ResY=450`
    - Voir exemple launch_packaged_listenserver_client.bat

Notes:

- Mise en place moyenne: Nécessite de compiler le code, cooker les assets et packager le tout
- Exécution rapide: Code compilé et assets cookés s'exécutant nativement hors moteur
- Fidélité élevé: Excécute un vrai client sans support ni artifice du moteur

Packaged (Dedicated Server)

Exécution des instances de jeu dans des excutables indépendants avec Unreal Engine bâti des sources

Support	Effort de mise en place	Fidélité	Utilisation principale
Listen-server			Test
Dedicated Client	Élevé	Exacte	Démonstration
Dedicated Server			Livraison

Étapes

- Compiler Unreal Engine
  - Obtenir les sources
  - Installer les prérequis
  - Compilez les sources
- Définir une target Client et une une target Server
- Générer la solution
- Ouvrir le projet dans Unreal Editor bâti
- Packager les instances à partir de Platforms -> Windows
  - Choisir Binary Configuration
    - Debug Game (code pas optimisé, aisé à débugger)
    - Development (code un peu optimisé, débuggable)
    - Test (code optimisé, difficile a débugger, accès aux outils)
    - Shipping (code optimisé, très difficile à débugger, outils retirés)
- Packager le dedicated client
  - Choisir Build Target: Client
  - Cliquer sur Package Project
  - Sélectionner la destination pour le package (Suggéré: [workspace]/Packaged)
  - Application va être compilé, les assets cookés et le tout sera packagé (une à plusieurs heures)
  - Localiser l'exécutable à [PackageDestination]/WindowsClient/[ProjectName]Client.exe
- Packager le dedicated server
  - Choisir Build Target: Server
  - Cliquer sur Package Project
  - Sélectionner la destination pour le package (Suggéré: [workspace]/Packaged)
  - Application va être compilé, les assets cookés et le tout sera packagé (une à plusieurs heures)
  - Localiser l'exécutable à [PackageDestination]/WindowsServer/[ProjectName]Server.exe
- Lancer un dedicated client packaged

    - Invoquer `[PathToPackage]/[ProjectName].exe 127.0.0.1 -log -windowed -ResX=800 -ResY=450`
    - Voir exemple launch_packaged_dedicated_client.bat
- Lancer un listen-server packaged server
    - Invoquer `[PathToPackage]/[ProjectName].exe ?listen -log`
    - Voir exemple launch_packaged_dedicated_server.bat

Notes:

- Mise en place longue: Nécessite de compiler le code du jeu et du moteur, cooker les assets du jeu et du moteur et packager le tout
- Exécution rapide: Code compilé et assets cookés s'exécutant nativement hors moteur
- Fidélité exacte: Permet de retirer les composantes serveur (code) du client et les composantes client (code, textures, matériaux, etc.) du serveur