PRéSENTATION

MISS NEVILLE

BY IMANE TARUF

SOMMAIRE

1. Pitch
2. Plus tu en as, pire c’est !
3. Démo
4. Principales features
5. Axes d’amélioration

01

PITCH

PITCH

Genre :

- Jeu d'horreur 2D en vue top down.
- Ambiance creepy.

Références :

- Mad Father et FAITH : Chapter 3.
- Inspiration des thèmes sur l'occulte et le paranormal.

Synopsis :

- Vous êtes un enquêteur venu enquêter sur la mystérieuse disparition d'une noble famille anglaise dans leur manoir maintenant maudit par la population. Résolvez des puzzles qui vont vous permettre de vous rapprocher un plus de la vérité…

Mad Father

FAITH : Chapter 3

02

"PLUS TU EN AS, PIRE C'EST !"

“PLUS TU EN AS, PIRE C’EST”

Plus le joueur avancera dans le scénario, plus le joueur résolvera de puzzles…

Et plus le scénario devient bizarre.

Et plus les événements paranormaux s’intensifient.

Et plus les puzzles deviennent durs.

03

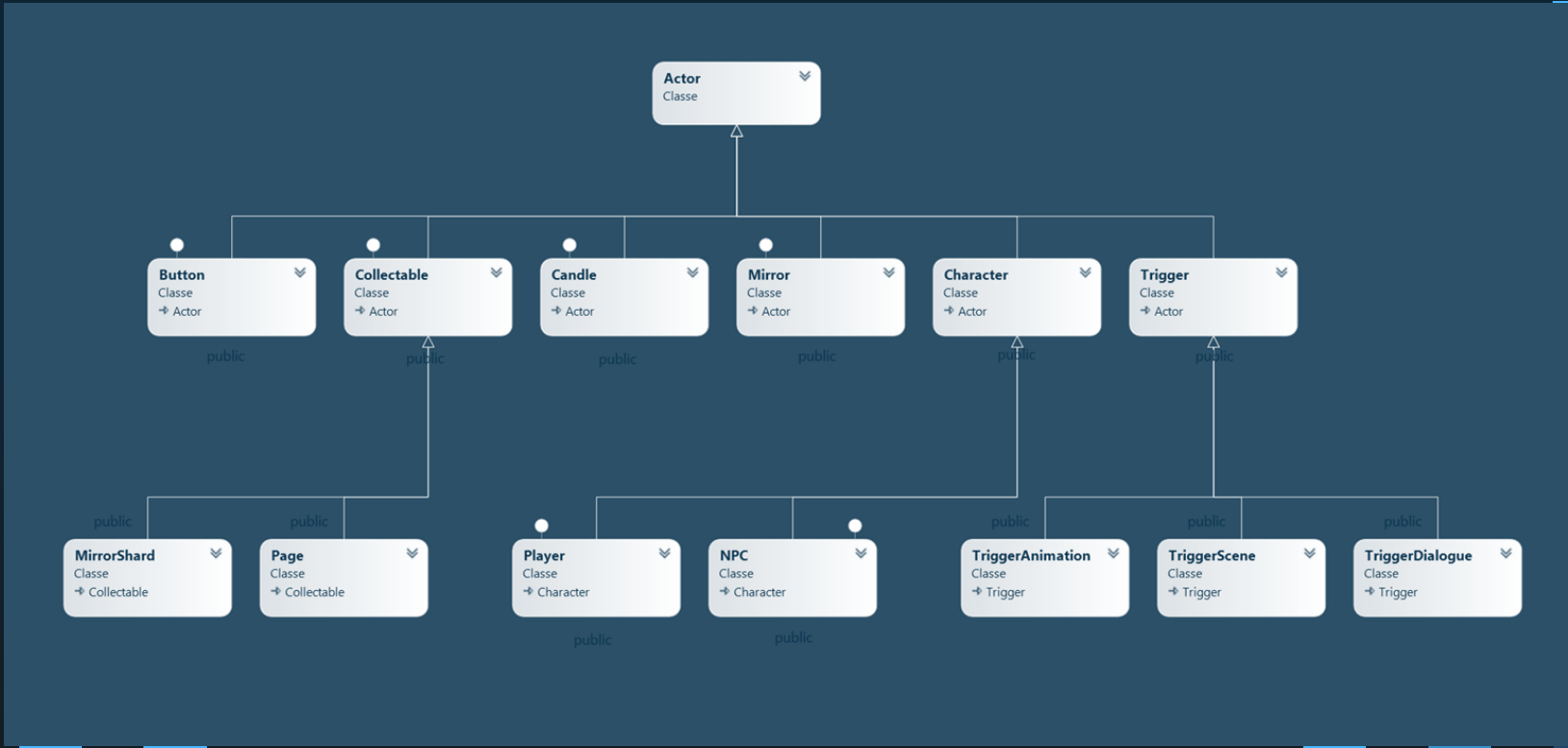
DÉMO

04

PRINCIPALES FEATURES

ACTEURS

Actor : élément contenant des informations classiques : *Sprite*, *Collision*, *Position, ...* et qui va venir peupler les différentes scènes du jeu et contribuer au gameplay.



CONTRÔLEURS

2 différents types de contrôleurs :

- Player Controller : Spécifique au joueur avec des capacités / restrictions uniques dans le jeu.
- Object Controller : Spécifique aux différents objets du jeu.

Objectifs :

- Le joueur peut contrôler un personnage et s'immerger dans le jeu.
- Donner du dynamisme au jeu et ne pas avoir que le joueur vivant dans ce manoir hanté.

INVENTAIRE

Collecter et conserver les objets récupérables tout au long du jeu !

- Tous les objets “Collectable” vont y être stockés.
- Les “Collectable” implémentent leur propre fonctionnalité d’utilisation de l’objet.

Objectifs :

- Garder une trace de l’avancée dans l’histoire du jeu.
- Transporter des objets utiles à la résolution de puzzles.

SCèNES

Le manoir est constitué de nombreuses pièces…

- Chaque pièce du manoir est reliée à au moins une autre pièce.
- Chaque pièce va être constituée de ses propres acteurs voir propres puzzles à résoudre.

Des fonctions bien utiles…

Chaque scène implémente 4 fonctions : *Enter*, *Update*, *Render et Exit.*

- Enter : Appelée pour instancier toutes les données initiales.
- Update : Appelée à chaque frame pour mettre à jour des données (Ex: Sprite des acteurs, mouvements des objets).
- Render : Appelée à chaque frame pour mettre à jour l'affichage en jeu (Ex: Afficher les sprites des acteurs).
- Exit : Appelée à la fermeture d'une scène.

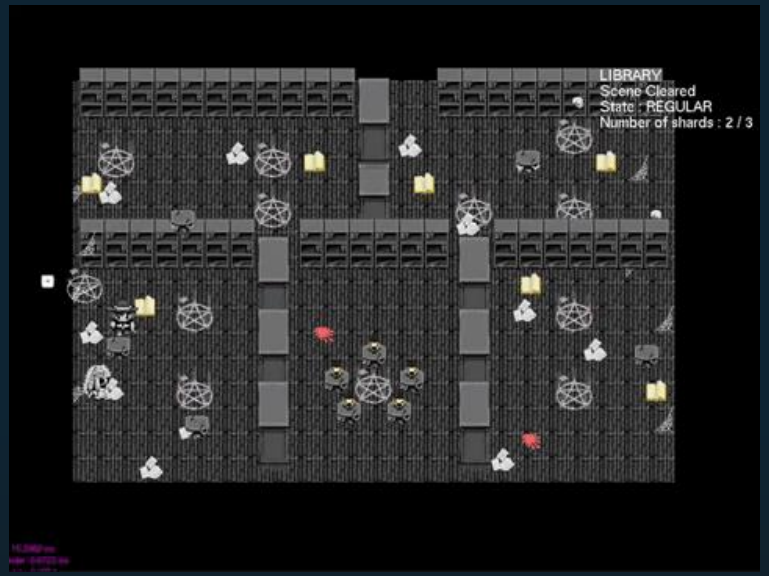
Objectifs :

- Setup des événements paranormaux uniques dans chaque pièce.
- L'idée d'un manoir / labyrinthe avec plusieurs pièces reliées non-linéairement.

PUZZLES

Objectifs :

- Challenger le joueur sur sa capacité de réflexion et sur son sang-froid en ces moments obscurs.
- Chaque puzzle tire parti des systèmes existants (interagir avec des éléments, utilisation de l'inventaire, parcourir les scènes, …).
- Setup des événements paranormaux uniques dans chaque pièce avec des puzzles de plus en plus durs.
- Faire transparaître le thème : Plus tu en as, pire c'est !



DELEGATES

Enregistrer des fonctions pour les utiliser à un instant T.

Objectifs :

- Déclencher des événements pour dynamiser le jeu et toujours aller dans le sens du manoir vivant bien creepy.

Ex : Un puzzle est résolu ? Une porte s'ouvre alors à un endroit du manoir.

GAME STATE

Problèmes :

- Comment gérer l'activation / la désactivation des différents systèmes (capacités, inventaire, dialogues, passage d'une scène à une autre, ...) tout au long du jeu ?
- Comment gérer ce qui doit être mis à jour (acteurs, scènes, puzzles, ...) ?

Solutions :

3 principaux Game State définis et uniques :

- "Regular" : Dans la phase d'exploration, contrôler / utiliser les compétences du personnage.
- "Inventory" : Dans l'inventaire, naviguer et sélectionner des objets.
- "Dialogue" : Dans une boîte de dialogue, lire des propos et suivre l'histoire.

GAME STATE

Chaque Game State implémente 4 fonctions : *Enter*, *Update*, *Render* et *Exit*.

- Enter : Appelée à l'entrée d'un State (Ex: le joueur ouvre l'inventaire).
- Update : Appelée à chaque frame pour mettre à jour des données (Ex: le joueur navigue dans l'inventaire).
- Render : Appelée à chaque frame pour mettre à jour l'affichage en jeu (Ex: Affichage de l'inventaire).
- Exit : Appelée à la sortie d'un State (Ex: le joueur ferme l'inventaire).

Définition d'un "Game State Controller" qui va s'occuper de gérer les différents "Game State" et la scène actuelle.

- Va appeler ces 4 fonctions notamment en fonction des actions du joueur.

Ex : Le joueur est en "Regular" puis lance un dialogue, alors on a *Exit* de "Regular" qui est appelée puis *Enter* de "Dialogue" qui est appelée. Et inversement quand le dialogue prend fin.

Objectifs

- Changer les contrôles du personnage d'un State à l'autre. N'autoriser que certaines actions et n'afficher certaines données que lorsqu'un State spécifique est actif.

Ex :
State "Regular" : Bouton X = Interagir avec des PNJ / objets
State "Dialogue" : Bouton X = Passer au dialogue suivant.

05

AXES D'AMÉLIORATION

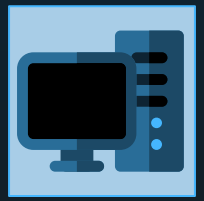
AXES D'AMÉLIORATION

Ennemis

Rajouter des ennemis pour embêter encore plus le joueur !

Scènes

Plus de pièces pour créer un manoir grandeur nature et y faire perdre le joueur !



Programmation

Faire encore et toujours plus généraliste. Mieux travailler les delegates !

VFX & SFX

Rajouter plus d'effets visuels et sonores pour intensifier l'ambiance creepy !

MERCI POUR VOTRE ATTENTION !