Rapport projet final Programmation

TECHNIQUES DE L'INFORMATIQUE.

PROGRAMMATION EN JEUX VIDÉO – 420.BX

Vincent Chagnon

Numéro de cours : 420-J13-AS

Date de remise 2 mai 2025

Insertion d’une liste

Pour la première partie, j’ai inséré une liste pour comptabiliser le score de chaque mini-jeu. Pour bien implémenter la liste, j’ai créé une nouvelle classe pour gérer la logique du score. J’ai créé une fonction pour ajouter le score que j’appelle à la fin de chaque mini-jeu. La fonction prend 2 Integer, l’un pour le score et l’autre pour l’index associé au mini-jeu. J’ai choisi une liste pour garder les scores, puisque que le jeu contient 3 min-jeu et dans chaque mini-jeu le score est calculé différemment. De plus, la liste me permet d’ajouter des scores dans le futur sans avoir besoin de modifier la logique de stockage du score. Avec un tableau, il serait beaucoup plus difficile d’ajouter des scores supplémentaires. En créant une classe pour gérer la logique de tout les scores, cela permet à chaque mini-jeu de gérer la gestion du score à sa propre façon et par la suite appeler la fonction pour ajouter le score à la fin du niveau. La plus grande difficulté rencontrée en créant la gestion des scores à été de l’ajouter aux jeux créés par mes collègue et d’être capable de garder le score en changeant de scène. Il était difficile de comprendre comment mes collègues avait créé leur mini-jeu dans le projet initial. Chaque jeu avait ses propres problèmes à corriger. J’ai fini par me concentrer sur ma partie que j’avait créé. J’ai manqué de temps pour corriger tous les problèmes avec nos mini-jeu. Par exemple, dans « SpaceShooter », le jeu ne se termine pas. Lorsque les astéroïdes sont détruits, le jeu continue, avec rien à faire. Dans « HideAndSeek », les objets sont seulement affichés dans la console et ils ne sont pas tous nommé. Il est difficile de deviner ce qu’est l’objet 2 par exemple. Dans « RunInTheForest » lorsque je retournais au menu, il était possible d’appuyer sur start, le jeu ne commençait pas, seulement le UI disparaissait. Je n’ai pas réussi à afficher le score après les mini-jeu. Lorsque j’appuie sur le bouton start, le UI du score se supprime de la scène et j’ai manqué de temps pour le corriger. Au début, le score restait à zéro puisque la liste se réinitialisait à chaque fois que je changeais de scène. J’ai dû modifier la fonction et ajouter une fonction qui crée une instance de l’objet et qui ne se détruit pas au chargement avec la fonction DontDestroyOnLoad() de Unity.

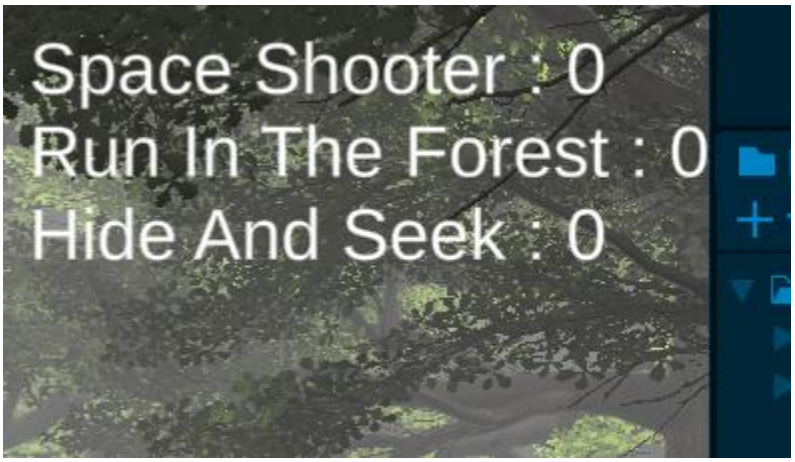
```
private void Awake()
{
    if (instance == null)
    {
        instance = this;
        DontDestroyOnLoad(gameObject);
    }
    else
    {
        Destroy(gameObject);
        return;
    }

    ShowScore();
}
```

Je n'ai pas réussi à le tester puisqu'en ajoutant cette logique, l'UI à commencé à se détruire en commençant le jeu. J'ai réussi à corriger le problème en partie. J'ai ajouté une coroutine pour patienter avant de modifier le texte du score.

```
Unity Message | 0 references
private void OnEnable()
{
    SceneManager.sceneLoaded += OnSceneLoaded;
}

Unity Message | 0 references
private void OnDisable()
{
    SceneManager.sceneLoaded -= OnSceneLoaded;
}

2 references
private void OnSceneLoaded(Scene scene, LoadSceneMode mode)
{
    StartCoroutine(ReassignResultatNextFrame());
}

1 reference
private System.Collections.IEnumerator ReassignResultatNextFrame()
{

    yield return null;

    Debug.Log("[ScoreManager] Rebinding TMP after scene load...");

    resultat = null;

    foreach (var tmp in FindObjectsOfType<TextMeshProUGUI>())
    {
        if (tmp.CompareTag("Resultat"))
        {
            resultat = tmp;
            break;
        }
    }

    if (resultat == null)
    {
        resultat = FindObjectOfType<TextMeshProUGUI>();
    }

    ShowScore();
}
```

Cela à fonctionner pour quand

l’on commence le jeu, mais à la fin d’un mini-jeu, le UI pour le score se supprimait toujours.



Utilisation de dictionnaires

Pour la deuxième partie, j'ai instauré un dictionnaire pour gérer l'histoire. Dans cette classe, deux dictionnaires ont été créé, l'un pour l'histoire au début du jeu et l'autre pour le mini-jeu « RunInTheForest ».

```
private Dictionary<int, string> storyScreens = new Dictionary<int, string>
{
    { 0, "Au fond de la forêt, un vieux cabanon t'attend..." },
    { 1, "Ici, loin de tout, tu es le dernier espoir des âmes égarées." },
    { 2, "Peu d'équipements. Beaucoup de volonté. Ton métier : guérir." },
    { 3, "Chaque patient cache une histoire... à toi de la découvrir." }
};

private Dictionary<int, string> miniGameIntroScreens = new Dictionary<int, string>
{
    { 0, "Un villageois est tombé gravement malade en pleine forêt.\nSeul toi, docteur du cabanon, peux lui venir en aide."
    { 1, "Le temps presse. S'il reste seul trop longtemps, il n'y survivra pas." },
    { 2, "Prends ton sac, traverse la forêt, et retrouve-le avant qu'il ne soit trop tard." }
};
```

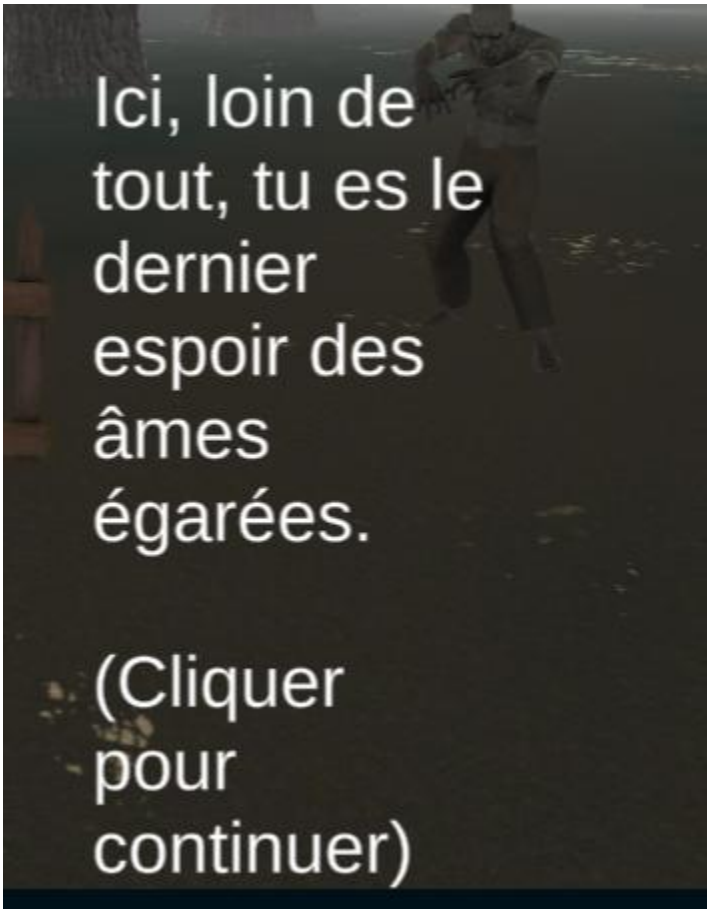
J'ai ajouté une fonction pour gérer la logique de texte et charger le mini-jeu à la fin de l'histoire.

```
public void NextStoryScreen()
{
    currentStoryIndex++;
    if (!isStartingMiniGame && storyScreens.ContainsKey(currentStoryIndex))
    {
        ShowCurrentStoryScreen();
    }
    else if (isStartingMiniGame && miniGameIntroScreens.ContainsKey(currentStoryIndex))
    {
        ShowMiniGameIntroScreen(currentStoryIndex);
    }
    else
    {
        if (isStartingMiniGame && !string.IsNullOrEmpty(nextSceneName))
        {
            try
            {
                SceneManager.LoadScene("RunInTheForest");
            }
            catch (System.Exception e)
            {
                Debug.LogError($"Failed to load scene '{nextSceneName}': {e.Message}");
            }
        }
        else
        {
            if (storyCanvas != null)
            {
                storyCanvas.SetActive(false);
            }
        }
        storyFinished = true;
    }
}
```

Cette fonction appel la fonction pour afficher le texte tant qu'il reste du texte à afficher

```
private void ShowCurrentStoryScreen()
{
    if (storyText != null && storyScreens.ContainsKey(currentStoryIndex))
    {
        storyText.text = storyScreens[currentStoryIndex] + "\n\n(Cliquer pour continuer)";
    }
}

2 references
private void ShowMiniGameIntroScreen(int index)
{
    if (storyText != null && miniGameIntroScreens.ContainsKey(index))
    {
        storyText.text = miniGameIntroScreens[index] + "\n\n(Cliquer pour continuer)";
    }
}
```

Cette classe ne m'a pas donnée beaucoup de problème. Au début, le jeu ne changeait pas de scène après l'histoire et c'est pourquoi j'ai ajouté le changement de scène dans cette classe. J'ai ajouté l'histoire à la suite des commentaires à la suite de ma présentation de projet. Il avait été mentionné que d'avoir une logique pour expliquer pourquoi un docteur fait des opérations dans son cabanon et avoir justification pour les mini-jeu serait intéressant.



Réflexion

J’ai appris à approfondir l’utilisation de dictionnaire. Je n’avais jamais utilisé de dictionnaire avant la réalisation de ce projet. J’ai remarqué que ça aide à visualiser les données enregistrées dans le dictionnaire. Cela rend le code plus facile à lire.

L’élément du cours qui à représenter le plus grand défi pour moi fut d’ajouter à un projet existant. Cela ma permet de réaliser à quel point il est important de bien organiser sont code à partir du début. En ajoutant au projet que j’avais réalisé à la dernière session, je me suis rendu compte de beaucoup d’erreur que j’avais fait dans l’organisation de mon code. Il était souvent difficile de me retrouver dans mon propre code. Je me suis rendu compte que mes classes aurait pu être organisé différemment. Plusieurs bugs qui avaient été laissé dans le projet original sont venu me hanter dans la réalisation de ce projet.

La prochaine étape dans mon parcours professionnel et académique est de travailler sur d’autre projet. Participer à des concours pour rencontrer d’autre personnes dans l’industrie du jeux vidéo. Peaufiner mes aptitudes à fabriquer des jeux et apprendre encore plus de nouvelle chose. Je me rends compte que je n’ai pas assez fait depuis le début du programme et je dois mettre les bouchés double pour rattraper le retard que je me suis imposé.

Tous les projets des autres étudiants me semblait assez bien fait. Il m’est très difficile d’en choisir un en particulier. Selon moi lui qui m’a le plus impressionné est celui de Kseniya Finchenko. Son projet me semblait assez impressionnant. Je n’ai pas vue beaucoup du jeu pendant la présentation. Cependant, de ce que j’ai vu, les sorts m’on rendu curieux. La ville me semblait assez bien fait, les assets semblait bien aller ensemble pour créer un style uni.

Je n’ai pas l’intention de poursuivre ce projet au-delà du cours. Je crois que de commencer d’autre projet va m’être plus utile pour ma carrière future. Ce projet était mon premier jeu et la meilleure manière de le continuer serait de le recommencer à zéro. J’ai aussi d’autre idée en tête pour créer des nouveaux jeux. Avec ce que j’ai appris dans ce cours, je me sens plus apte à créer un jeu plus complexe. Les jeux qui m’intéressent le plus sont généralement des jeux avec des mécaniques complexe. Et pour Shed Doctor moi et mes coéquipiers on n’était pas prêt à faire un projet trop compliqué.

Voici le lien pour la version jouable du jeu : https://ledogetier.itch.io/sheddoctor