|  |
| --- |
| Module Experimentation  Pour Unity      Auteur : Pergandi  Date : 20/01/2020  Version : 1 |

HISTORIQUE DES VERSIONS

|  |  |  |
| --- | --- | --- |
| Modifications | Auteurs | Date |
| Edition initiale | Pergandi | 20/01/2020 |

SOMMAIRE

[1. Description 4](#_Toc30515529)

[2. Prérequis 4](#_Toc30515530)

[3. Développer/Tester sur le matériel 4](#_Toc30515531)

[4. Rappel propriétés et règles de diffusion 4](#_Toc30515532)

[5. Fonctionnalité du module experimentation 4](#_Toc30515533)

[6. Contenu du livrable 5](#_Toc30515534)

[7. Les données d’entrée 5](#_Toc30515535)

[7.1. Structures des fichiers d’entrée 5](#_Toc30515536)

[7.2. Format des fichiers txt 6](#_Toc30515537)

[7.3. Le fichier « experimentation » 6](#_Toc30515538)

[7.4. Le fichier Subject.txt 7](#_Toc30515539)

[7.4.1. Le fichier Parameters.txt 7](#_Toc30515540)

[7.4.1. Le fichier Protocol.txt 7](#_Toc30515541)

[8. L’expérience exemple 9](#_Toc30515542)

[9. Les données de sorties 10](#_Toc30515543)

[10. Contenu Unity 12](#_Toc30515544)

[10.1. Structure projet 12](#_Toc30515545)

[10.2. Composition de la scène ExperimentationExample 13](#_Toc30515546)

# Description

Nom : Experimentation

Pour qui : pour expérimentateur et programmeur

Quelle fonction : module de gestion d’une expérimentation sous Unity

Pour quel bénéfice : faciliter le développement mais également l’utilisation d’une application Unity pour de l’expérimentation.

# Prérequis

Aucun pour les utilisateurs/expérimentateurs

Débutant pour les développeurs

# Développer/Tester sur le matériel

OS : windows 10

Unity : 2019.1.3f1

Dispositifs testés : PC

# Rappel propriétés et règles de diffusion

Ce module est la propriété de l’Université Aix-Marseille et du CNRS. Il n’est pas destiné à être diffusé en dehors de l’ISM et des enseignements/projets pédagogiques de la FSS.

# Fonctionnalité du module experimentation

Un expérimentateur qui utilise une application Unity réalisé avec le module « Experimentation » lui permet de configurer une expérimentation via 4 fichiers txt. L’expérimentateur peut modifier :

* les informations concernant le sujet (définies avec le développeur)
* les paramètres de l’expérimentation (définis avec le développeur)
* le protocole (l’ordre de essai, les valeurs des modalités, comment rejouer les essais échoués, …)

Ce module sécurise les données (il vérifie que les données d’entrée sont bien lues) et évite tout écrasement de données sauvegardées (par la datage des fichiers et dossier).

Ce module permet au développeur :

* D’accéder à toutes les données des fichiers txt qui configure une expérimentation
* De faciliter le déroulement d’une passation (il ne définit que le déroulement d’un essai, le déroulement global des essais est effectué par le module)
* De faciliter l’enregistrement de données (le développeur doit juste déclarer des colonnes et choisir les données dans Unity)

# Contenu du livrable

* Le document présent
* Un package Unity « Experimentation – n° version.unitypackage »
* Un dossier « ExperimentationExample » contenant un exemple simple d’un exécutable Unity intégrant le module Experimentation. Cette application réalise une expérimentation de pointage avec souris. Toutes les ressources de cette application se trouvent dans le package Unity fourni.

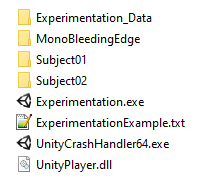
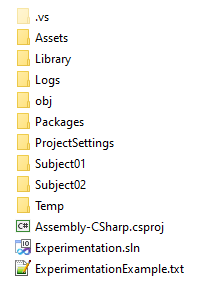
# Les données d’entrée

## Structures des fichiers d’entrée

Pour faire tourner une application intégrant le module Experimentation une structure de fichiers doit être réalisée.

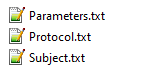
Dans le dossier racine du projet Unity ou de l’exécutable il faut qu’il y ait :

* un fichier txt « experimentation » (dont le nom est à définir avec le développeur)
* au minimum un dossier « sujet » dont le nom est libre
* dans le ou les dossiers « sujet » :
  + le fichier Subject.txt qui contient des informations liées au sujet
  + le fichier Paramters.txt qui contient des paramètres de l’expérimentation que l’on peut éditer
  + le fichier Protocol.txt qui contient le protocole à jouer

Comme exemple, dans le dossier « ExperimentationExample » vous avez le fichier « experimentation » « ExperimentationExample.txt » et deux dossiers sujets « Sujet01 » et « Sujet02 » chacun contenant les 3 fichiers « Subject.txt », « Parameters.txt » et « Protocol.txt »

*Structure d’un dossier racine d’un exécutable (à gauche) et d’un projet unity (à droite)*

Le fichier « ExperimentationExample.txt » a été défini tel quel dans l’application il ne faut pas modifier son nom. Cependant vous pouvez créer autant de dossier sujet que désiré avec un nom libre, mais ces dossiers doivent absolument contenir les 3 fichiers de configuration « Subject.txt », « Parameters.txt » et « Protocol.txt ».



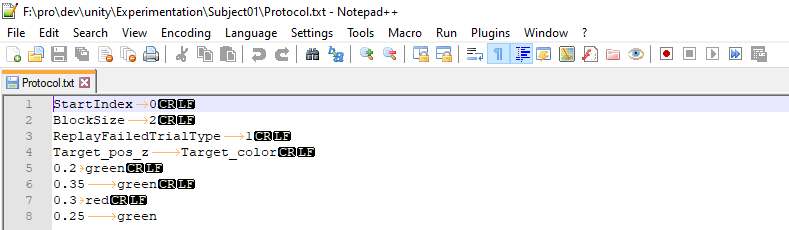
*Structure d’un dossier « Sujet »*

## Format des fichiers txt

Les fichiers de configuration utilisent uniquement deux caractères blancs à savoir la tabulation et le saut de ligne pour séparer les données. N’utilisez pas d’accents et d’espaces pour les valeurs. Ne créez pas non plus des lignes sans valeurs en fin de document. Enfin pas de virgule pour le nombre décimaux c’est le point qui est utilisé.

Ces règles simplistes et stricts permettent d’avoir un code simple sans erreur de lecture des données et d’assurer une fiabilité de fonctionnement.

Il est conseillé d’utiliser un éditeur tel que Notepad++ afin de visualiser les caractères blancs (bouton  ).



*Visualisation des caractères « blancs » sous Notepad++ (CRLF = saut de ligne, flèche orange = tabulation)*

## Le fichier « experimentation »

Le fichier « experimentation » contient seulement deux paramètres :

* WorkingDirectory qui est le nom du dossier sujet à jouer
* SaveData qui attend une valeur « true » ou « false » afin de sauvegarder ou non les données

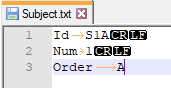


*Exemple d’un fichier « experimentation »*

Dans notre exemple, vous pouvez modifier le fichier ExperimentationExample.txt pour choisir le deuxième sujet. Il vous suffit de lancer l’exécutable Experimentation.exe, vous verrez en haut à gauche que le numéro du sujet change.

## Le fichier Subject.txt

Ce fichier doit contenir les caractéristiques du sujet. Une caractéristique est composée d’un nom et de sa valeur (séparés par une tabulation). Il doit y avoir une ligne par caractéristique. Dans notre exemple, le fichier Subject.txt contient les attributs suivants : Id, Num, Order avec les valeurs respectives S1A, 1 et A



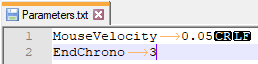
*Exemple d’un fichier « Subject.txt »*

Le choix des attributs du sujet est complétement libre. Ce fichier sert à garder trace de certaines informations concernant le sujet par exemple comme son identifiant ou l’ordre du protocole utilisé. Les informations fournis peuvent être également utilisés dans le programme pour réaliser certaines fonctions (en accord avec le développeur), comme par exemple modifier le contenu de l’expérimentation selon le groupe d’âge du sujet.

### Le fichier Parameters.txt

Ce fichier contient des paramètres de l’expérimentation que l’on peut donc modifier. Ces paramètres permettent à l’expérimentateur de choisir des valeurs encore non déterminées. Par exemple, s’il y a un doute sur la durée de présentation d’un stimulus, il est alors judicieux de mettre cette durée comme paramètre dans ce fichier. Le nom des paramètres et leurs valeurs doivent être choisis en accord avec le développeur. Le format est identique à celui du fichier Subject.txt.

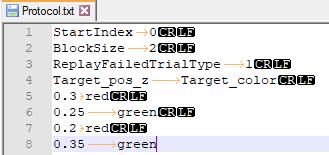
Dans notre exemple d’expérience nous avons les paramètres MouseVelocity (pour choisir la vitesse de la souris utilisée pour déplacer un curseur) et la durée d’un chrono (chrono qui va donner un signal pour l’atteinte d’une cible)



*Exemple d’un fichier Parameters.txt*

### Le fichier Protocol.txt

Ce fichier d’écrit l’ensemble des essais dans un tableau de données.



*Exemple d’un fichier « Protocol.txt »*

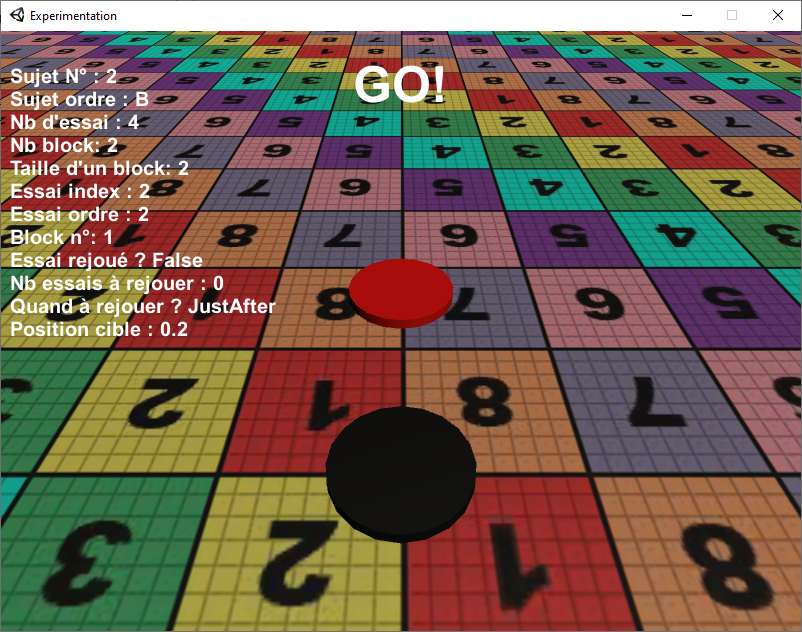
En début du fichier nous avons trois paramètres dont le nom et l’ordre ne doivent pas être modifiés. Les 3 paramètres sont :

* StartIndex : ce paramètre permet de jouer le protocole à partir d’un numéro d’essai. Par défaut cette valeur est 0 afin de jouer le protocole depuis le début. Mais il se peut que lors d’une passation que le programme plante, qu’un problème survienne avec le sujet, qu’une coupure électrique intervienne ou que tout autre imprévu apparaisse. Il serait dommage de perdre le sujet après 2h de passation alors qu’il ne restait que 3 essais à jouer. Pas de panique il suffit de mettre le n° du dernier essai réalisé et de relancer l’application.
* BlockSize : ce paramètre définit le nombre d’essai par bloque. Dans notre exemple il y a deux essais par bloque. Ce paramètre n’a d’utilité qu’avec le paramètre suivant. S’il n’y a pas de bloque mettez la valeur -1.
* ReplayFailedTrialType : ce paramètre permet de choisir si l’on souhaite rejouer des essais qui échoueraient (ex : perte de la capture de mouvement, comportement du sujet ne respectant pas la consigne, …). Les conditions pour déclarer qu’un essai a échoué doivent être déterminer avec le développeur. 4 valeurs sont possibles 0, 1, 2, 3 correspondants aux cas suivants :
  + 0 : pas d’essai à rejouer
  + 1 : rejoue l’essai échoué de suite après
  + 2 : rejoue l’essai échoué à la fin du bloque
  + 3 : rejoue l’essai échoué à la fin du protocole

Après ces 3 paramètres, un tableau décrit le protocole à jouer. Ce dernier doit être défini avec le développeur. Le tableau contient un entête avec le nom de chaque variable par colonne. Chaque ligne correspond à un essai avec les modalités des variables. Dans notre exemple d’expérience, il y a deux variables, la distance d’une cible « Target\_pos\_z » (en mètre) et la couleur de la cible « Target\_color » (rouge ou verte). Nous avons dans l’exemple 4 essais.

# L’expérience exemple

L’expérience proposé en exemple consiste à un sujet de contrôler via la souris un curseur noir à la forme cylindrique afin d’atteindre une cible blanche aussi cylindrique, le tout se trouvant sur un plan horizontal quadrillé. Après la fin d’un chrono d’une durée paramétrable (via le fichier « Parameters.txt »), un « GO » apparait et la cible change de couleur. Si la cible devient verte il faut l’atteindre, si elle est rouge il ne faut pas bouger. Le sujet a pour consigne d’atteindre la cible verte le plus rapidement possible. Si le sujet n’’atteint pas la cible quand elle est verte, nous considérons que le sujet n’a pas respecté la consigne, c’est un essai échoué/raté, il sera rejoué (Cette expérience est purement fictive et à visée pédagogique, les hypothèses testées dans cette expérience seront celles de votre imagination !). Sur la gauche un ensemble d’information est affiché afin de prendre connaissances de la configuration de l’expérimentation et des évolutions du protocole.



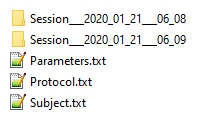
*Visuel de l’expérience exemple*

Lancer l’application « Expérience.exe » afin de découvrir l’expérimentation. Fermer l’application, maintenant il est conseillé de modifier les différents fichiers d’entrées afin d’apprendre à les utiliser, par exemple :

* Modifier l’identifiant d’un sujet
* Modifier la durée du chrono
* Modifier la vitesse de la souris
* Modifier ou ajouter des essais
* Modifier les options de « replay » des essais échoués (cible verte non atteinte)
* Créer un troisième dossier « Sujet » au nom de votre choix, attribuer lui l’identifiant 3 dans le fichier « Subject.txt ». Affecter le nom du dossier choisi au paramètre «WorkingDirectory » du fichier « ExperienceExample.txt ». Vous pouvez lancer l’expérimentation pour ce nouveau sujet, vous devriez voir l’identifiant 3 affiché en haut à gauche.

# Les données de sorties

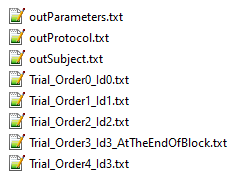
Dans le dossier du sujet choisi, à chaque lancement de l’application un nouveau dossier « Session » daté est créé.



*Exemple de deux dossiers « Session » l’un créé le 21 01 2020 à 06h08, et l’autre à la même date une minute après.*

Chaque dossier « Session » comprend les fichiers suivants :

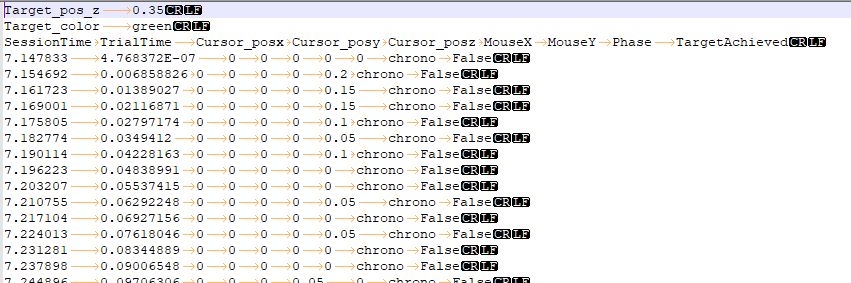
* outParameters.txt : c’est une copie du fichier Parameters.txt situé dans le dossier « Sujet ». Cela permet au programme de vérifier lui-même qu’il a bien lu les données et à l’expérimentateur de rappeler les paramètres qui ont été utilisés lors de cette session.
* outProtocol.txt : idem que précédemment sauf que cela concerne la copie du fichier Protocol.txt
* outSubject.txt : idem pour le fichier Subject.txt
* Trial\_OrderXXX\_IdYYY.txt : il y a un fichier d’enregistrement par essai. Le numéro suivant « Order » correspond à l’ordre dans lequel l’essai a été joué. Le numéro suivant « Id » est le numéro d’apparition dans le fichier Protocol.txt. S’il y a aucun essai rejoué le numéro « Order » et l’identifiant « Id » sont identiques. Si à la fin d’un fichier « Trial », il y a une des options de « replay » d’essai raté (« JustAfrer », « AtTheEndOfBlock », « AtTheEnd ») cela signifie qu’il s’agit d’un essai raté et que celui-ci a été rejoué selon l’option choisi. Dans l’exemple illustré ci-dessous l’essai Id3 a été rejoué à la fin du bloque.



*Exemple de contenu d’un fichier « Session »*

Ouvrez un des fichiers « Trial ». En début de fichier vous trouverez toutes les modalités de l’essai (dans notre exemple « Target\_pos\_z » et « Target\_color »). Ensuite il y a un tableau avec en entête le nom des données enregistrées. « SessionTime » et « TrialTime » sont automatiquement gérés par le module, les autres données sont créées par le développeur (en accord avec l’expérimentateur). « SessionTime » correspond au chrono depuis le lancement de l’application. « TrialTime » est le chrono depuis le début de l’essai (commence donc à 0). Ensuite chaque ligne correspond à une frame/image (la fréquence d’enregistrement dépend de la configuration de Unity et de sa synchronisation ou non avec le rafraichissement vertical de l’écran qui est souvent 60Hz ; voir Edit>Project Settings>Quality section Other).

Les autres données correspondent à la position du curseur, aux mouvements de la souris, à la phase courante de l’essai (« chrono » ou « attente de cible ») et à l’atteinte ou non de la cible.



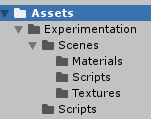
*Exemple d’un fichier d’enregistrement « Trial »*

# Contenu Unity

Créez un projet Unity vierge et importez le package « Experimentation – n° version.unitypackage ». **Pensez à copier-coller** dans la racine du projet Unity toute la structure des fichiers d’entrée (fichier ExperimentationExample.txt + les dossiers sujets)

## Structure projet

Le package Experimentation se trouve dans le dossier « Experimentation » contenant un sous dossier « Scripts » et un sous dossier « Scenes ».



*Contenu du package « Experimentation »*

Le sous dossier « Scripts » contient 7 fichiers C# dans lequel se trouve tout le code pour faire fonctionner le module :

* **MExp** : c’est le seul fichier à utiliser et doit être affecté à un des game object de votre scène Unity pour utiliser le module Experimentation. C’est le script « maître » du module et représente votre expérimentation.
* LData : gère et stock les données à enregistrer
* LParameters : gère et donne accès aux paramètres déclarés dans le fichier Parameters.txt
* LProtocol : gère l’ordre et stocke les essais tel que défini dans le fichier Protocol.txt
* LSubject : gère et donne accès aux caractéristiques du sujet déclaré dans Subject.txt
* LTrial : gère et donne accès aux valeurs d’un essai
* USaveMatrixData : script Unity a affecté à un game object dont on souhaite enregistrer sa matrice de transformation



*Fichiers C# du module Experimentation*

Dans le sous dossier scène se trouve la scène « ExperimentationExample » et toutes ses ressources (materials, textures et scripts). C’est cette scène qui correspond à l’application exemple fourni. Son sous dossier « Scripts » comprend :

* **MyExperimentation** : script qui gère tout le déroulement de l’expérience et qui illustre comment utiliser le module Experimentation pour une expérimentation particulière.
* DetectCollision : script utilisé pour détecter si le curseur touche la cible

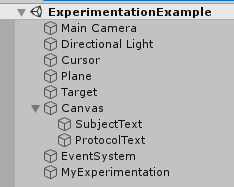


*Scripts de l’experimentation exemple*

## Composition de la scène ExperimentationExample

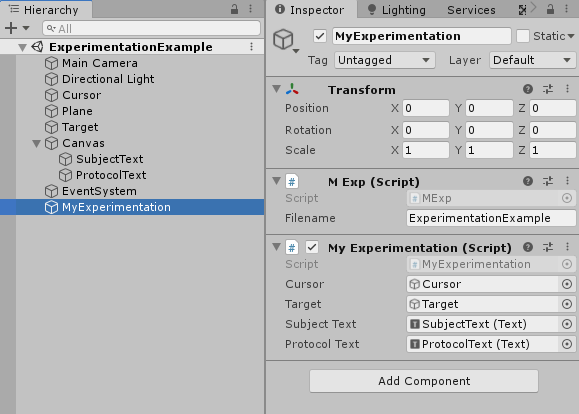
Ouvrez cette scène (si ce n’est déjà fait). La scène est composée de quelques éléments :

* Main Camera : caméra de la scène
* Direction Light : lumière de la scène
* Cursor : curseur contrôler par le sujet via la souris
* Plane : le plan quadrillé très coloré
* Target : la cible à atteindre
* Canvas : permet d’afficher du texte 2D
* SubjectText : texte destiné au sujet (chrono + le GO)
* ProtocolText : texte avec toutes les informations concernant le déroulement de l’expérience
* EventSystem : accompagne Canvas pour l’affichage de texte
* MyExperimentation : game object qui contient les scripts faisant fonctionner l’expérimentation



*Contenu de la scène ExperimentationExample*

Sélectionnez le game object « MyExperimentation ». Vous pouvez voir que le script MExp du module Experimentation y est affecté ainsi que le script « MyExperimentation ». Ce dernier possède comme paramètres d’entrée le curseur, la cible et les deux textes qui seront manipulés par le script.



*Structure du game object MyExperimentation*

Ouvrez le script « MyExperimentation » vous trouverez toutes les explications concrètes sur l’utilisation du module dans les commentaires. Si vous avez besoin d’information complémentaires tous les scripts du module sont commentés expliquant son utilisation détaillée. Bonne poursuite.