# Scale for project Nibbler

#### Git repository

### Introduction

Nous vous demandons, pour le bon déroulement de cette notation :

- De rester courtois, poli, respectueux, constructif lors de cet échange. Le lien de confiance entre la communauté 42 et vous en dépend.
- De bien mettre en évidence auprès de la personne notée (ou du groupe) les dysfonctionnements éventuels.
- D'accepter qu'il puisse y avoir parfois des différences d'interprétation sur les demandes du sujet ou l'étendue des fonctionnalités. Restez ouvert d'esprit face a la vision de l'autre (a-t-il raison ou tort ?), et notez le plus honnêtement possible.

Bonne soutenance à tous!

# Guidelines

RAPPELEZ VOUS QUE VOUS NE DEVEZ CORRIGER QUE CE QUI SE TROUVE SUR LE DÉPÔT DE RENDU DE L'ÉTUDIANT.

Il s'agit de faire un "git clone" du dépôt de rendu, et de corriger ce qui s'y trouve.

Si le correcteur n'a pas encore fait ce projet, il est obligatoire de lire le sujet en entier avant de commencer cette soutenance.

# Ratings

Define the type of error (if there is an error), which ended the correction.

Ok Empty wolkcomplete Workuthor filealid compiletowne Cheat Crash

# Attachments

Subject Intra - Elearning - Subnotions

## Sections

## **Préliminaires**

### Consignes préliminaires

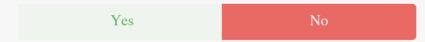
Vérifiez d'abord les éléments suivants :

- Il y a bien un rendu (dans le dépôt git)
- Pas de triche, les deux étudiants doivent pouvoir expliquer leur code.
- Vous ne devez pas trouver sur le depot de (sources de) bibliotheques que le groupe n'a pas ecrites lui meme. Cela signifie que n'importe quelle bibliotheque exterieure utilisee dans le projet ne doit pas etre presente sur le depot. Cela inclu bien entendu les bibliotheques graphiques, mais aussi des bibliotheques utilitaires comme Boost, etc. Voir le sujet.
- Le groupe corrigé doit fournir un script ou un ensemble de scripts permettant une installation simple et efficasse des dependances conformement au sujet. Le groupe corrigé est donc seul responsable de la mise en place de son environnement d'evaluation sur le systeme utilise pour la correction quelque soit l'etat de celui-ci.
- Le correcteur est seul juge pour determiner si cette phase d'installation est trop longue ou inadaptee et peut decider unilateralement de mettre un terme a la soutenance prematurement en flagant le rendu en "incomplet work". Un grand pouvoir implique de grandes responsabilites. Don't be a dick, mais soyez pas naifs non plus.
- Dans l'executable principal, le groupe corrigé NE DOIT PAS

avoir fait la MOINDRE référence à une bibliothèque d'affichage précise (comme OpenGL, SDL, etc). Cela serait un hors sujet pur et simple. Cela inclut non seulement les primitives d'affichage, mais aussi les primitives de gestion d'évènements. Par exemple la présence de types Qt ou SDL, ou carrément des appels à des fonctions de ces bibliotheques.

- La boucle de jeu doit IMPERATIVEMENT se trouver dans le programme principal et surtout pas dans les bibliothèques. Toutefois, certaines bibliothèques peuvent imposer leur propre boucle, dans ce cas, cette boucle DOIT être manipulée pour être synchronisée avec la boucle du corps et en dépendre entièrement. Le contraire est hors sujet, flagguer "incomplet work". Voir le sujet et plus bas dans ce bareme en cas de doute.
- Un makefile comportant les regles habituelles doit permettre de compiler l'executable principal ainsi que les 3 bibliotheques dynamiques.
- Si le rendu comporte moins de 3 bibliotheques dynamiques utilisees a l'execution, flagguer "incomplet work".

Si un élément de cette liste n'est pas respecte, la notation s'arrête la. Utilisez le flag approprie. Vous êtes encouragés à continuer de débattre du projet, mais le barème n'est pas appliqué.



# Tests de fonctionnalité

#### Lancement

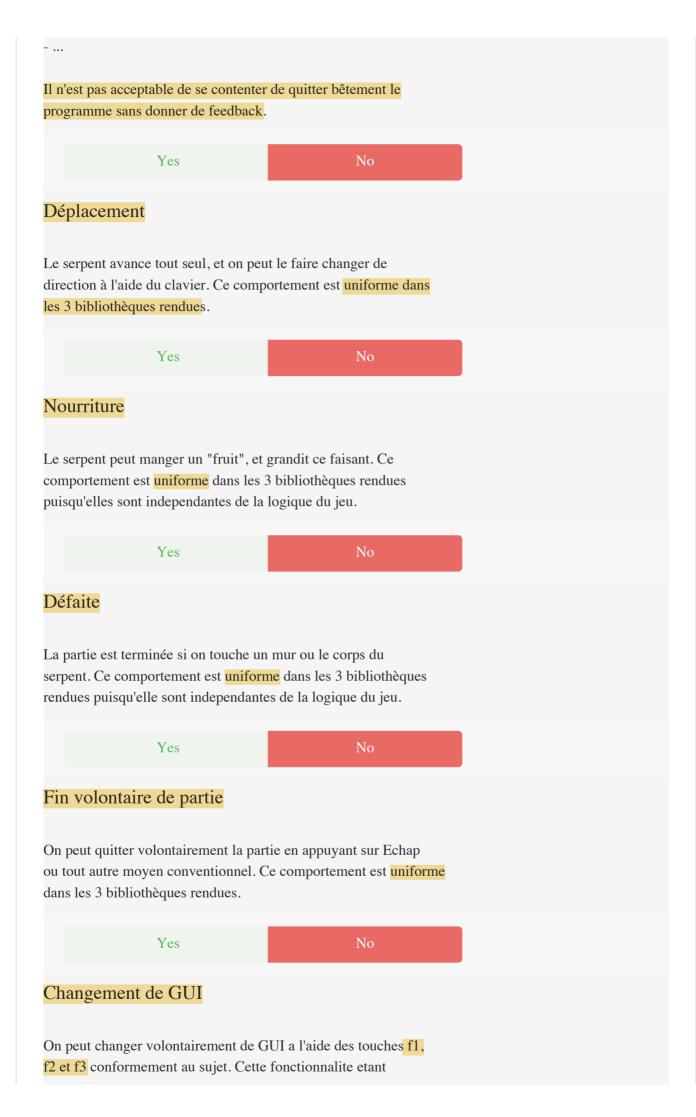
L'executable principal permet au moins de choisir la taille de la surface de jeu, et choisi une bibliotheque dynamique pour afficher le jeu. Le jeu s'execute alors normalement.



#### Erreurs de lancement

Les erreurs au lancement doivent être gérées. Le minimum à gérer est :

- Nombre d'arguments insuffisant.
- Mauvaise largeur / hauteur (nombres négatifs, trop petits, trop grands, pas des nombres ...)



centrale dans le projet, assurez-vous bien qu'elle soit stable. Verifiez a differents moments de l'execution, dans un ordre quelconque, en switchant rapidement de l'une a l'autre, etc...



# Implementation

On rappelle que la boucle de jeu et donc la logique du jeu doit être située dans l'executable principal tel que demande dans le sujet. AUCUNE exception n'est acceptable. Si une bibliothèque impose sa propre boucle, alors cette boucle doit être synchronisée avec la boucle de l'executable principal. Oui, c'est délicat et oui cette partie du code doit être réfléchie. Mais cela implique surtout que cette bibliotheque graphique n'etait probablement pas adaptee a ce projet. La liberte impose de faire des choix...

### Boucle de jeu cohérente

La boucle de jeu devrait ressembler à ceci :

```
while (...)
{
traitementEvenements();
miseAJourDonneesDeJeu();
miseAJourAffichage();
}
```

C'est le cas?



#### Gestion d évènements

La gestion des évènements clavier doit être fonctionnelle et propre. Une forêt d'ifs n'est PAS acceptable ici.



# Interface core/bibliothèques

Le but principal du projet est double. D'une part, il s'agit de desolidariser les aspects graphiques/inputs du programme en les relocalisant dans une bibliotheques dynamique. Le rendu doit proposer l'utilisation de 3 bibliotheques graphiques differentes. D'autre part, il s'agit d'interagir de maniere uniforme avec l'une ou l'autre des bibliotheques dynamiques creees par le groupe.

Les étudiants doivent avoir prévu une interface intelligemment conçue pour interagir avec leurs modules d'affichage. Un minimum serait une interface ressemblant grossierement à la suivante :

```
class INibblerDisplay
{
public:
virtual void init(...) = 0;
virtual void getEvents(...) = 0;
virtual void updateGameData(...) = 0;
virtual void refreshDisplay(...) = 0;
virtual void stop(...) = 0;
};
```

On devrait y trouver de quoi initialiser la bibliotheque graphique contenue dans la bibliotheque dynamique, de quoi notifier la bibliotheque de l'etat du jeu pour affichage, de quoi recuperer les inputs utilisateur, etc. Conformement aux consignes de la partie preliminaire, je vous rappelle qu'aucune reference d'aucune sorte a l'une ou l'autres des bibliotheques graphiques ne doit exister dans le programme principal, et par consequence, dans cette interface.

Est-ce qu'on a bien une interface adaptée dans le programme principal qui permet de communiquer uniformement avec au moins les trois bibliothèques dynamiques encapsulant les trois bibliotheques graphiques choisies ?



### Plug-ins d affichage objets / Design coherent

Les bibliotheques dynamiques de Nibbler peuvent donc etre vue comme des plug-ins implementant une interface particuliere (c'est a dire un "contrat" exposant leurs fonctionnalites disponibles pour le programme utilisateur). Cela sous entend par definition que le code de l'executable principal et des bibliotheques dynamiques soit en C++ conformement au sujet. Toutefois, la communication entre un programme et des bibliotheques dynamiques est en C. Cela implique donc d'utiliser une construction syntaxique speciale "extern C" afin de renseigner le compilateur C++ qu'une petite partie du code est en C et non en C++. Ces petites fonctions C sont appellees "points d'entree" d'une bibliotheque.

Les points d'entree peuvent renvoyer des infos generiques sur la

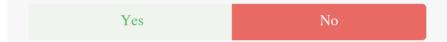
bibliotheque afin d'assurer sa compatibilite avec le programme principal, mais il doit OBLIGATOIREMENT exister un point d'entree permettant de recuperer une instance de la classe d'affichage contenue dans la bibliotheque dynamique sous forme d'un pointeur sur l'interface dont nous avons parle dans la section precedente. Dans le cas contraire, le rendu est a cote de la plaque.

Exemple naïf mais parfaitement recevable:

extern "C"
{
INibblerDisplay\* getDisplayModule() { return new NibblerDisplayOpenGL(); }
}

Les points d'entree doivent etre des fonctions tres courtes se contentant generalement de renvoyer une donnee, telle qu'une nouvelle instance. Si vous constatez qu'un traitement plus long est fait dans un point d'entree, cette question n'est pas validée.

Verifier que les trois bibliothèques fonctionnent de la manière décrite ci-dessus.



## Gestion d erreur des bibliotheques dynamiques

Verifier que le groupe a prévu une gestion des erreurs possibles au chargement, a l'utilisation et au dechargement des bibliotheques dynamiques à base d'une exception. Vous devez refuser les gestions d'erreurs qui ne sont pas a base d'exceptions, ainsi que les exceptions scalaires.



# Le groupe

# Approche du projet par le groupe

Nibbler est un projet a realiser en binomes. Sa complexite relative en fait un candidat ideal pour entammer une approche reflechie de la repartition du travail.

- 5 etoiles -> Les deux etudiants ont participe de maniere active au developpement et a la soutenance. Tous deux sont incollables aussi bien sur le fonctionnement general du projet

que sur les detail d'implementation de la partie de leur binome (par exemple concernant les bibliotheques graphiques).

- 4 etoiles -> Les deux etudiants ont participe de maniere active au developpement et connaissent bien leurs parties respectives, mais ne connaissent pas la partie de leur binome en details.
- 3 etoiles -> Pas utilise sur ce bareme.
- 2 etoiles -> Un etudiant est tres en retrait par rapport a l'autre, aussi bien que sur le travail realise que sur la soutenance, mais il semble visiblement avoir participe a hauteur de ses moyens.
- 1 etoile -> Pas utilise sur ce bareme.
- 0 etoiles -> Vous pensez qu'un seul etudiant sur les deux a effectue tout le travail ou presque (quelqu'en soit la raison), quelque soit la qualite du dit travail.

### Rate it from 0 (failed) through 5 (excellent)



#### Doriuc

#### Sons

Il y a une (ou plusieurs) bibliothèque de sons, et elle est gérée de la même manière que les bibliotheques graphiques. c'est-à-dire via une bibliotheque dynamique que l'étudiant a créée, qui est loadée et utilisée au runtime.

Yes No

# Multiplayer

On a deux ou plus snakes qui se battent pour la nourriture ?! (un mode single player doit rester possible)



#### Jeu en réseau

L'étudiant a fait un jeu bad ass - multiplayer et en réseau (le bonus precedent est valide aussi)?

Yes No

#### D autres bonus

Les étudiants ont ajouté à leur projet d'autres fonctionnalités améliorant l'expérience de jeu ? Comptez-les ici, vous pouvez en noter jusqu'à 5 disctinctes.

Exemple de fonctionnalité bonus valide :

- Un système de score
- Des bonus / malus
- Une tête se démarquant graphiquement du reste du corps
- Une vitesse de deplacement de plus en plus rapide
- Des obstacles sur le plateau de jeu
- etc ...

Rate it from 0 (failed) through 5 (excellent)

# Conclusion

Leave a comment on this correction

\* Comment