

Projet Informatique – Sections Electricité et Microtechnique

Printemps 2024 : *Microrécif* © R. Boulic & collaborators

Rendu3 (20 mai 23h59)

Objectif de ce document : Ce document utilise l'approche introduite avec la série théorique sur les [méthodes de développement de projet](#) qu'il est important d'avoir faite avant d'aller plus loin.

En plus de préciser ce qui doit être fait, ce document identifie des ACTIONS à considérer pour réaliser le rendu de manière rigoureuse. Ces ACTIONS sont équivalentes à celles indiquées pour le projet d'automne ; elles ne sont pas notées, elles servent à vous organiser. Vous pouvez adopter une approche différente du moment que vous respectez l'architecture minimale du projet (donnée Fig 9b).

1. Buts du rendu3 : dialogue avec l'interface graphique et simulation

Votre approche peut évoluer entre ce que vous avez décrit pour le rendu2 en matière de structuration des données et ce que vous mettez en œuvre finalement du moment que vous respectez les responsabilités des différents modules (Fig ci-contre).

Lancement et comportement attendu:

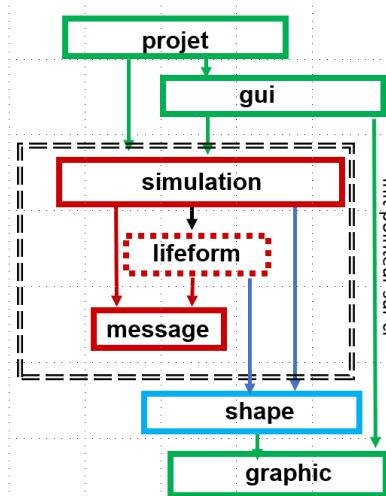
Le programme sera lancé soit en indiquant un nom de fichier à ouvrir (rendu2) :

```
./projet test1.txt
```

Soit sans aucun argument sur la ligne de commande :

```
./projet
```

Dans ce second cas, l'interface est créée et le programme attend qu'on lui demande d'ouvrir un fichier avec le bouton Open.



Donnée Fig 9b

. 1.1 Rapport final (MAX 2 pages):

Le Rapport ne contient PAS de page de titre, ni de table des matières. Il est écrit avec une police 11 au minimum et 14 au maximum, interligne simple. Le rapport est écrit en français ou en anglais ; une orthographe ou une grammaire défaillante peut induire les correcteurs en erreur. Le Rapport ne duplique pas le précédent. Il contient :

- **la description de l'exécution de votre programme pour les fichiers de test suivants :**
 - **txx.txt** : pour l'état initial puis les mises à jour 1, 2 et 3 (sans génération d'algue), capturer l'image de la simulation (montrez seulement la partie de l'image du Monde qui change) ; brièvement décrire les modifications observées sur les entités corail, algue et scavenger.
 - **tzz.txt** : montrer 4 captures de l'ensemble de la simulation (initiale puis les mises à jour 10, 20 et 30) avec génération d'algues ; expliquer les modifications observées. Combien d'entités obtenez-vous à l'itération 30 ?
- **Méthodologie et conclusion** : comment avez-vous organisé votre travail à plusieurs, avec quels outils ? (git?, VSCode ? etc...) ; indiquer la répartition des contributions par module et comment vous avez organisé le travail au sein du groupe (par quels modules avez-vous commencé, comment les avez-vous testés). Indiquer la proportion de travail simultané en groupe (c'est à dire côté à côté ou en-ligne sur le même code) par rapport au travail indépendant (chacun de son côté). Avec le recul, est-ce que vous modifieriez cette proportion?
 - Quel était le bug le plus fréquent, pourquoi ? Quel est celui qui vous a posé le plus de problème et comment a-t-il été résolu, ...).
 - Pour conclure fournissez une brève auto-évaluation de votre travail et de l'environnement mis à votre disposition (points forts, points faibles, améliorations possibles).

Le rapport final doit être inclus dans le fichier archive du rendu final (en format pdf).

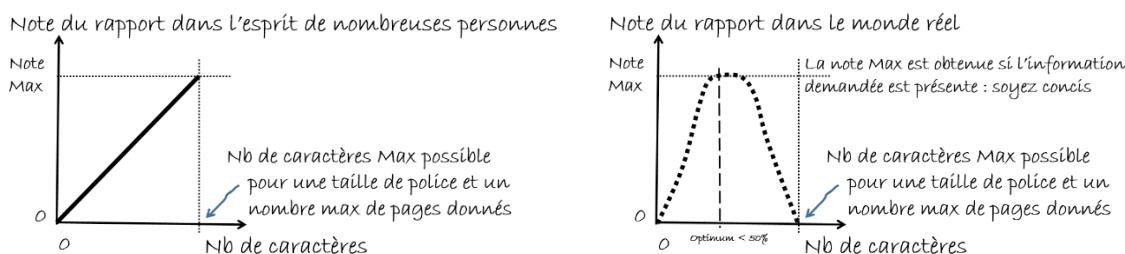


Figure 1 : un mauvais rapport dilue l'information utile jusqu'à atteindre le nombre maximum de caractères possibles sur la page (fig gauche) : en fait ce type de rapport sera pénalisé parce qu'il est peu lisible ; un bon rapport est celui qui fournit les informations demandées avec concision avec une mise en page aérée et lisible (fig droite)

1.2 Evaluation du rendu3 : en plus d'évaluer les critères sur la qualité du code (ne pas oublier de corriger tout warning obtenu aux rendus précédents), nous effectuerons une évaluation manuelle de votre programme comme suit :

- lancement avec les 2 syntaxes : avec et sans nom de fichier sur la ligne de commande.

- évaluation de la mise à jour de la simulation dans différents scénarios pour faire apparaître les comportements demandés (détails dans la donnée générale). La majorité des scénarios sera constitué des cas très simples discutés dans la section suivante. Seuls les derniers scénarios auront une complexité plus grande pour l'évaluation des performances.

2. Organisation du travail

Nous recommandons de tester votre simulation en mode Step avec l'affichage graphique de chaque état successif de la simulation car c'est un puissant outil de mise au point. Utilisez les fichiers de test fournis.

N'hésitez pas à compléter avec du code de scaffolding qui affiche la valeurs des variables importantes dans le terminal pour compléter vos outils de mise au point. D'ailleurs, en cas de bug, l'affichage dans le terminal devrait être redirigé vers un fichier de texte (cf cours redirection de la sortie) pour pouvoir l'ouvrir avec un éditeur de texte après le crash du programme et analyser l'évolution des valeurs de vos variables.

fichier	Generation d'algue	But du test: vérifier...
t28.txt	désactivée	•
t29.txt	désactivée	•
t30.txt	désactivée	•
t31.txt	désactivée	•
t32.txt	désactivée	•
t33.txt	désactivée	•
t34.txt	désactivée	•
t35.txt	désactivée	•
t36.txt	désactivée	•
t37.txt	désactivée	•
t38.txt	désactivée	•
t39.txt	désactivée	•
t40.txt	désactivée	•
t41.txt	désactivée	•
t42.txt	désactivée	•
t43.txt	désactivée	•
t44.txt	activée	•
t45.txt	activée	• autre contexte general pour évaluer la robustesse et les performances du programme

3. Forme du rendu

Convention de style : il est demandé de respecter les [conventions de programmation du cours](#).

- Seulement deux méthodes/fonctions au-dessus de 40 lignes (max 80 lignes) sont autorisées pour l'ensemble du code du rendu3.

Documentation : l'entête de vos fichiers source doit indiquer le nom du fichier et les noms des membres du groupe avec, **pour les fichiers .cc**, une estimation du pourcentage de contribution de chaque membre du groupe au code de ce fichier.

Rendu : pour chaque rendu **UN SEUL membre d'un groupe** (noté **SCIPER1** ci-dessous) doit téléverser un fichier **zip**¹ sur moodle (pas d'email). Le non-respect de cette consigne sera pénalisé de plusieurs points. Le nom de ce fichier **zip** a la forme :

¹ Nous exigeons le format zip pour le fichier archive

Compléter le fichier fourni **mysciper.txt** en remplaçant 111111 par le numéro SCIPER de la personne qui télécharge le fichier archive et 222222 par le numéro SCIPER du second membre du groupe.

Le fichier archive du rendu2 doit contenir (**aucun répertoire**) :

- Rapport (fichier pdf)
- Fichier texte édité **mysciper.txt**
- Votre fichier **Makefile** produisant un executable **projet**
- Tout le code source (.cc et .h) nécessaire pour produire l'exécutable.

*On doit obtenir l'exécutable **projet** après décompression du fichier **zip** en lançant la commande **make** dans un terminal de la VM (sans utiliser VSCode).*

Auto-vérification : Après avoir téléchargé le fichier **zip** de votre rendu sur moodle (upload), récupérez-le (download), décompressez-le et assurez-vous que la commande **make** produit bien l'exécutable lorsqu'elle est lancée dans un *terminal de la VM* (sans utiliser VSCode) et que celui-ci fonctionne correctement.

Exécution sur la VM: votre projet sera évalué sur la VM à distance (compilation avec l'option -std=c++17).

Backup : Il y a un backup automatique sur votre compte myNAS.

Debugging : Visual Studio Code offre un outil intéressant pour la recherche de bug (VSCode tuto).

5. Travail en groupe et oral final individuel

Même si le travail est réparti entre les deux membres du groupe, l'oral final portera sur l'ensemble du code que les deux membres du groupe doivent connaître et être capables de répondre à des questions sur son fonctionnement.

La constatation d'une incapacité individuelle à répondre à des questions sur le code de votre groupe peut conduire à une note différente pour les deux membres du groupe.