Projet
V3.1.0

Document en ligne : www.mickael-martin-nevot.com

Date de rendu : **25 janvier 2023 à minuit**
Date de présentation : **26 janvier 2023**
Travail : **groupe de trois**

1 Assistance

Vous pouvez contacter l'enseignant en cas de besoin en formalisant et en ciblant précisément votre demande. Pour ce faire, vous devez respecter les règles de communication et d'envoi (ci-dessous).

2 Communication et envoi

2.1 Généralités

En joignant vos coordonnées (*e-mail* et téléphone portable notamment) à un message ou à votre livraison, vous pourrez être joint en cas de problème.

2.2 Communication

Chaque communication devra être faite :

- à l'adresse électronique : mmartin.nevot@gmail.com ;
- en faisant figurer `[AMU][LP]` en début de sujet.

2.3 AMeTICE

Votre livrable devra être :

- nommé de la manière suivante (`Nom1`, `Nom2`, `Nom3` étant vos noms, par ordre lexicographique de vos noms de famille, et `Prénom1`, `Prénom2`, `Prénom3` vos prénoms) : `Nom1 Prénom1 - Nom2 Prénom2 - Nom3 Prénom3` ;
- compressé dans une seule archive au format ZIP **n'excédant pas 10 Mo** ;
- remis, avant la date de rendu, sur AMeTICE (http://ametice.univ-amu.fr), à la section rendu du cours correspondant.

3 Présentation orale

Vous effectuerez une présentation orale sous forme de démonstration de votre projet respectant les

consignes suivantes :

- une durée maximale de **15 minutes** devant un jury ;
- à la **date de présentation** (selon l'ordre de passage que vous sera communiqué par votre enseignant responsable) ;
- sur la réalisation de votre **projet** à laquelle des questions de cours viendront s'ajouter.

4 Sujet

Vous devez écrire la **spécification fonctionnelle technique** et coder l'**implémentation** d'une application en **Java** qui doit utiliser les concepts vus lors de cet enseignement.

Le sujet porte sur les réalisations des TD `TD4-2 : Java, cas pratique`, `TD5 : Algorithmique « avancée »` et `TD6 : UML`.

La partie « graphique » de votre application peut être minimaliste (affichage en ligne de commande par exemple).

Vous devez utiliser le plus possible d'**algorithmique**, de **structures de données**, de particularités du langage **Java** et de **modèles de conception**, notamment ceux vus dans cet enseignement. En particulier, votre application doit comporter des **classes abstraites**, des **interfaces**, des **collections**, avec des **itérateurs**, des **exceptions**, des ***threads***, de la **généricité**, et au moins un algorithme de **tri**. De plus, vous devez réaliser l'ensemble des diagrammes de **cas d'utilisation**, de **séquences** et de **classes** de l'application.

Vous devez apporter un soin tout particulier à la présentation du **code source** (**indentation**, respect d'une **convention de nommage**, **commentaires**, etc.) et à l'**architecture** des répertoires.

Votre application doit être **évolutive**, **modulaire** et professionnelle (**robuste**, **fiable** et intégralement **fonctionnelle**).

5 Livrables

Vous devez envoyer les éléments suivants avant la date de rendu :

- la **spécification** minimale (sous forme d'un ou plusieurs documents) :
  - l'**étude de conception** ;
  - les **diagrammes UML ;**
  - la clôture du travail (**synthèse en anglais**, **bilan technique**, **problèmes rencontrés**, **écarts avec les prévisions**, **mesures d'amélioration**) ;
- l'**application** :
  - le **code source** ;
  - l'application au format **JAR** ;
  - le **projet au format Eclipse** ;
  - le fichier « **lisez-moi** » ;
  - le **manuel d'utilisation** (présentant plusieurs scénarios d'utilisation pas à pas), vidéo ou non ;
  - la documentation complète générée par la **Javadoc** de l'application.

6 Conseils

Voici quelques conseils :

- pensez à utiliser des algorithmes en **pseudo-code** pour formaliser vos idées avant de coder des algorithmes complexes ;
- gérez la répartition de réalisation des fonctionnalités entre les membres du groupe (en se répartissant les différents modules par exemple) ;
- pour les redoublants et ceux qui sont à l'aise, vous pouvez mettre en place quelques fonctionnalités intéressantes supplémentaires, comme **l'intégration continue** (avec des solutions comme **Travis CI**, **GitLab** ou **Jenkins**) ou une véritable **interface graphique** (avec des solutions comme **JavaFX** ou **Swing**). À faire **uniquement** après avoir terminé tous les points essentiels notés ci-dessus et après avoir optimisé et approfondi les fonctionnalités demandées ;
- utilisez des **modèles de conception** adaptés à vos problématiques et optimisez votre code source en tentant de diminuer la complexité de vos algorithmes ;
- testez rigoureusement votre application.

Voici quelques conseils sur la présentation orale :

- testez toujours le matériel avant une présentation orale ;
- habillez-vous d'une tenue correcte en respect avec l'exercice ;
- ne vous cachez pas derrière quelqu'un d'autre, et ne renvoyez pas la faute à autrui ;
- ayez une voix qui porte (ce qui ne signifie pas forcément forte) ;
- regardez l'assistance ;
- employez une approche positive mais réelle (valorisante mais sans mensonge) ;
- assurez-vous que ce qui est montré est bien vu.

Voici les compétences relationnelles communément recherchées lors d'une présentation orale :

- supprimer les parasites ;
- avoir une bonne gestuelle ;
- se mouvoir correctement ;
- gérer l'utilisation de son regard.