DÉPARTEMENT
Sciences Informatiques

L3 I

Python
Projet 2025-2026

1 Objectifs

Le projet de la mineure « Python » permet de mettre en œuvre les principaux éléments du contenu du module dans le cadre de la conception d'une petite application en programmation objet. Le travail est à réaliser en trinôme (les attendus sont prévus pour un travail de trois personnes...).

Rappel des modalités d'évaluation (voir le document des MCC de la L3 I)
La mineure « Python » est en Contrôle Continu Intégral (CCI) : le projet compte pour la moitié de l'évaluation de ce module, l'autre moitié est constituée par une moyenne pondérée des différents TP notés. Dans tous les cas, il n'y a pas de seconde session et les notes obtenues sont reportées en seconde session.

2 Contexte du projet

2.1 Cadre général

On s'intéresse dans ce projet aux fichiers musicaux avec ou sans perte :
— avec perte de type `MP3` (**Moving Picture Experts Group Audio Layer 3**) ;
— sans perte de type `FLAC` (**Free Lossless Audio Codec**).
Ces fichiers musicaux possèdent des métadonnées, c'est-à-dire des informations complémentaires stockées dans le fichier `MP3` ou `FLAC` mais qui ne font pas partie du flux audio. Concernant ces métadonnées :
— pour les `MP3`, elles sont organisées selon le format « **ID3** » (cf. `https://id3.org/`) ;
— et pour les `FLAC` selon le format « **Vorbis comment** » (cf. `https://wiki.xiph.org/index.php/VorbisComment`).
Elles rassemblent notamment les informations de type durée du morceau, nom de l'artiste, nom de l'album, nom du morceau, etc. Les formats `MP3` et `FLAC` sont donc des conteneurs avec une partie pour les méta-données et une autre partie pour le flux audio proprement dit.

Par ailleurs, ces fichiers `MP3` et `FLAC` peuvent être organisés en playlist indépendamment de leur emplacement sur le disque dur de l'utilisateur. Ces playlists sont structurées et respectent plusieurs formats : dans le cadre de ce projet, nous retiendrons le format `XSPF` (**XML Shareable Playlist Format** : `https://xspf.org/`) : il s'agit d'un format XML de playlist.

2.2 Fonctionnalités attendues

Dans un premier temps, les principales actions de votre logiciel seront :
— l'extraction et l'affichage des méta-données d'un fichier `MP3` ou `FLAC` ;
— l'exploration complète d'un dossier et de l'ensemble de ses sous-dossiers contenant des fichiers musicaux : il faudra donc prévoir un filtre pour ne retenir que les fichiers `MP3` et/ou `FLAC` de cette arborescence, avec vérification de chaque extension et type MIME des fichiers trouvés pour ne retenir que ces fichiers `MP3` et `FLAC` ;
— la constitution et la sauvegarde d'une playlist par défaut contenant l'ensemble des fichiers `MP3` et `FLAC` du répertoire exploré récursivement.

Des fonctionnalités complémentaires sont attendues :
— en mode graphique :
  — extraire l'image (thumbnail) de couverture (cover) de l'album correspondant au morceau, l'afficher et l'enregistrer dans le dossier de l'album. Si cette image n'est pas disponible, la télécharger sur Internet, puis l'afficher, la valider par l'utilisateur et enfin l'enregistrer dans le dossier de l'album ;
  — ajouter un fichier en mode « drag & drop » pour composer une nouvelle playlist ;
— en mode console et graphique :
  — la possibilité d'écouter un morceau de musique ou une playlist enregistrée ;
  — la possibilité de modifier et sauvegarder les TAGS d'une chanson ;
  — la possibilité d'aller chercher les informations complètes d'un album, y compris les paroles de tous les morceaux, via une API-Web et de les afficher.

Pour les fichiers de playlist générés, vous utiliserez le validateur en ligne « `https://validator.xspf.org/` » pour vérifier que le fichier généré est valide.

2.2.1 Le mode CLI

En mode console (terminal) « CLI » : les paramètres attendus sur la ligne de commande sont : le nom du fichier `MP3` ou `FLAC` à analyser pour en extraire les métadonnées ou le nom du dossier à explorer pour générer une playlist par défaut : vous utiliserez un paramètre supplémentaire permettant de spécifier le type d'entrée (« `-f` » (**file**) pour un fichier, « `-d` » (**directory**) dans le cas d'un répertoire : ces 2 options étant exclusives l'une de l'autre). Le programme doit afficher directement dans la console le résultat de son analyse. Si aucun paramètre n'est indiqué le programme affiche un message d'erreur fonctionnelle ; avec l'option « `-h` » (**help**), le programme affiche l'aide et les options possibles. Une option supplémentaire « `-o` » (**output**) permet d'indiquer le fichier de sortie pour sauvegarder le résultat d'une extraction de la playlist dans un fichier (format `XSPF`) à spécifier par l'utilisateur.

En mode CLI, quelques scénarii d'exécution (exemples fictifs de lancement de vos 2 programmes) :

```
$ python3 cli.py
$ python3 cli.py -h
$ python3 cli.py -d .
$ python3 cli.py -f music.mp3
$ python3 cli.py -d ./music/ -o playlist.xspf
$ python3 cli.py -p music.mp3
$ python3 gui.py
```

Explications : les six premières commandes concernent le mode console (aussi appelé mode terminal ou fenêtre de commande) et la dernière commande permet de lancer l'interface graphique :
— la première ligne doit indiquer qu'il manque des paramètres et doit proposer de taper « `-h` » (ou « `--help` ») pour obtenir de l'aide ;
— la deuxième ligne affiche les modes d'utilisation de votre logiciel en mode console (i.e. les options possibles et leur rôle) ;
— la troisième ligne liste et analyse tous les fichiers à partir du dossier spécifié (« `-d` » ou directory) [ici à partir du dossier courant (« `.` »)] en parcourant l'ensemble de l'arborescence des sous-dossiers, en mode console ;
— la quatrième ligne prend en entrée le ficher « `music.mp3` » (« `-f` » ou file) et affiche à l'écran les métadonnées de ce fichier, en mode console (par défaut, on considère le fichier dans le répertoire courant mais le chemin peut être spécifié)
— la cinquième ligne prend en entrée le sous-dossier « `./music/` » du répertoire courant et sauvegarde la playlist générée dans le fichier xspf spécifié, en mode console ;

— la sixième ligne prend en entrée le ficher « `music.mp3` » (« `-p` » ou play) et joue le morceau, en mode console (par défaut, on considère le fichier dans le répertoire courant mais le chemin peut être spécifié) ;
— la dernière ligne correspond au lancement de l'interface graphique.

2.2.2 Le mode GUI

En mode graphique « GUI » : l'exploration d'une arborescence quelconque de fichiers permettra de lister tous les fichiers et leurs emplacements afin de générer la playlist par défaut ou une playlist personnalisée (au format XSPF). Pour chaque fichier MP3 ou FLAC, on pourra donc visualiser les principales métadonnées et sélectionner les morceaux qu'on souhaite ajouter dans une playlist.

Il faudra aussi intégrer la possibilité d'une recherche des métadonnées via une API.

Les métadonnées ou les playlists pourront ainsi être sauvegardées par l'utilisateur (qui pourra également réouvrir une playlist enregistrée précédemment).

3 Déroulement et attendus

3.1 Planning

— votre groupe doit être constitué en **semaine 40** (au plus tard le **dimanche 5 octobre**, **18h00**) et vous validerez sa composition via la section prévue dans la page du cours sur moodle ;
— vous identifierez les principales sous-tâches du projet à réaliser, leur niveau de priorité, la répartition des rôles au sein du groupe ainsi que le planning correspondant pour chaque tâche (période et durée). Vous créerez un diagramme de GANTT [1] de votre projet en utilisant pour cela un outil spécialisé. Dans tous les cas, votre diagramme de Gantt est à déposer sur la plateforme au format `png` (capture ou export) avant le **vendredi 17 octobre** (**semaine 42**), **18h00** : **1 point**. Ce fichier respectera les règles de nommage (§ 3.4) ;
— un premier point d'avancement est programmé le **7 novembre** (présence du trinôme obligatoire) : **2 points** (à titre indicatif, un niveau de réalisation d'environ 50% est attendu lors de ce point d'avancement en **semaine 45**), puis un second point d'avancement (à confirmer) le **28 novembre** (**semaine 48**) : **1 point** ;
— une vidéo faisant office de démonstration (au plus 5 min) : **5 points** (dépôt en **semaine 50**, **jeudi 11 décembre, 22h00** au plus tard). Cette démonstration permettra de visualiser toutes les fonctionnalités attendues des deux fonctionnements en mode console et graphique (il est donc très utile de prévoir un scenario...). Ce fichier respectera les règles de nommage (§ 3.4) ;
— tous les autres attendus (voir § 3.3) seront à déposer en **semaine 50** le **vendredi 12 décembre, 22h00** au plus tard ;
— une soutenance de 15 min par groupe (8 minutes de présentation et 7 minutes de question) est prévue le **vendredi 19 décembre** (**semaine 51**). Un ordre de passage avec la salle vous sera communiqué plus tard.

1. `https://www.ganttproject.biz/download/free`

3.2 Soutenance (4 points)

— la diapositive de titre présentera le groupe, le contexte, le sujet (i.e. : la page de garde sera compacte) ;
— les autres diapositives (8 au plus) devront présenter les spécificités de réalisation de l'équipe projet, la répartition des tâches et les principaux éléments de conception. Elles ne contiendront donc aucune information "évidente" (ex. détail du sujet, progression personnelle, ...) mais plutôt les choix faits et leur justification ;
— la diapositive de conclusion mettra en évidence le niveau d'achèvement du projet (points traités et non traités du cahier des charges et extensions s'il y en a) ;
— vous devrez prévoir une version pdf sur clé usb de votre diaporama au cas où.

NB : à éviter ABSOLUMENT : les diagrammes de classes UML illisibles (trop chargés, ...), les programmes (code Python), les captures d'écran (puisqu'il y a aussi une vidéo de démonstration), la liste des outils utilisés, ...

Important : vous devrez avoir votre machine portable allumée, prête avec l'ensemble des logiciels nécessaires déjà lancés AVANT d'entrer dans la salle. Votre portable devra disposer d'un port VGA/HDMI/USB C ou vous devrez prévoir un adaptateur correspondant à votre situation.

Vous veillerez à une répartition équitable de votre temps de parole au sein du groupe aussi bien pour la présentation que pour le temps des questions.

3.3 Attendus (7 points)

— complétude et qualité du projet : **3 points** (fichier « readme.md », code Python commenté ET documenté, ...). Les livrables sont **à déposer sur la plate-forme pédagogique** avant le **vendredi 12 décembre ET à partager via un lien** smash...
— fichier « `readme.md` » contenant les noms / prénoms / groupe projet / des membres du projet ainsi que les informations spécifiques indispensables (**1 point**) ;
— rapport de projet (minimum 5 pages, maximum 10 pages) : **3 points** (le fond et la forme seront évalués). Les 2 fichiers à rendre : le document de traitement de texte (format `odt`) et sa version `pdf` ;
— l'ensemble des fichiers sources du projet (.py) ;
— et la documentation détaillée (si possible au format `Doxygen`, à défaut au format `Pydoc`).

3.4 Règles de nommage et d'organisation des fichiers

Les archives doivent respecter le nommage suivant `L3_I-NOM1_NOM2_NOM3-xx.zip`, où `xx`, par exemple, sera remplacé par `Gantt` en réponse à la question 1 du §3.1 !

Lors du rendu final, **tous les fichiers et sous-dossiers** à remettre doivent être placés dans **un répertoire unique portant les noms du groupe** (sous la forme `NOM1_NOM2` ou `NOM1_NOM2_NOM3`) à compresser en un seul fichier au **format zip** qui sera déposé sur la plate-forme pédagogique moodle et doit respecter la structure ci-dessous :

```
NOM1_NOM2_NOM_3
    |__ doc
    |      |_ diaporama
    |      |_ documentation
    |      |_ rapport
    |__ src
    |      |_ library
    |      |_ cli
    |      |_ gui
    |_ README.md
```

Le non respect de ces règles entraînera des points de pénalité !