

Documentation

Gestionnaire de contacts en Python 3

Villena Guillaume, Saldo Yann, Omar Mdala

Novembre - Décembre 2016

Table des matières

I.	Manuel d'utilisation	2
A.	Installation	2
1.	Installation automatique	2
2.	Installation manuelle	2
B.	Fonctionnalités	3
1.	Gestion centralisée	3
2.	Ajout de contacts	3
3.	Edition de contact	3
4.	Importation de contacts.....	3
5.	Exportation de contacts	3
6.	Suppression de contacts.....	3
7.	Validation des entrées	4
8.	Recherche de contacts	4
9.	Conversion de contacts	4
10.	Modularité	4
11.	Ergonomie.....	4
C.	Formats et types de champs disponible	5
1.	Formats	5
2.	Types de champs	5
D.	Prise en main des interfaces.....	5
1.	Ligne de commande simple	6
2.	Ligne de commande interactive	8
3.	Interface Web	11
II.	Documentation des fonctions	12
A.	Construction générale du projet	12
B.	Ecrire un nouveau format pour le gestionnaire.....	12
C.	Ecrire un nouveau type de champs pour le gestionnaire	13

I. Manuel d'utilisation

A. Installation

1. Installation automatique

Dans ce cas l'installation des modules complémentaire est nécessaire. Pour se faire, lancer dans un terminal en administrateur, la commande suivante : `pip -r requirement.txt` tout en étant dans le dossier ou le projet a été extrait. Lancer cette commande, elle aura pour effet l'installation des dépendances du projet. Quand l'installation est terminée, vous pouvez lancez le fichier `main.py` avec la commande `python3 main.py`.

2. Installation manuelle

Dans le cas où vous souhaiteriez utiliser le projet sans toucher à l'installation de votre python, il est possible d'utiliser le fichier zip qui accompagne le projet, et qui contient toutes les modules nécessaires au bon fonctionnement du projet.

Le principe est simple :

1. Ouvrez le fichier `modules.zip`
2. Extraire l'archive au niveau du projet. (Si le projet se trouve dans `/home/pro/` extraire tous les fichiers de l'archive dans ce dossier)
3. Lancer le fichier principal : `python3 main.py`

B. Fonctionnalités

Dans cette partie vous trouverez un résumé rapide des différentes fonctionnalités apporté par ce logiciel.

1. Gestion centralisée

Ce gestionnaire de contacts fonctionne de manière centralisée, c'est à dire que le logiciel met en mémoire dans une base de données l'ensemble des contacts, et y permet l'ajout, modification, suppression de ceux-ci. Depuis l'interface face il est possible à tout moment d'exporter depuis la base des informations, tout comme importer des données dans la base.

2. Ajout de contacts

Grâce à ce programme il est possible d'ajouter des contacts dans la base, qui pourront ensuite être exporté vers différents formats connu tel que le Idif, vcard, csv, ...

Chaque contact possède des propriétés de base, plus un certain nombre de propriétés qui sont personnalisable, mais pas forcément exportable. Car les formats de destinations sont parfois limités à un nombre spécifique de champs.

3. Edition de contact

Ajouter un contact c'est bien, mais il peut arriver que les informations concernant un contact évoluent dans le temps. Dans ce cas, il est possible d'éditer un contact, afin de modifier certaines valeurs, mais encore, d'en ajouter si le besoin est. De cette façon il n'est pas nécessaire de supprimer un contact pour une petite modification sur l'un d'eux

4. Importation de contacts

Le logiciel qui vous est actuellement présenté, supporte un certain nombre de format d'importation. Cela permet de constituer facilement une base de données avec des donnée provenant de différent format. De plus une fois dans le format de la base l'exportation de ces données sont alors bien plus facile.

5. Exportation de contacts

Ce logiciel offre la possibilité d'exporter un/des contact vers des formats divers. Ces formats correspondent généralement à ceux les plus utilisés sur certain équipement comme des téléphone portable ou encore téléphone IP, bien logiciel de messagerie. La fonction d'exportation permet de créer un fichier compatible avec le logiciel souhaité. Dans le cas où le format n'est pas disponible, il est possible d'écrire un module supplémentaire pour la gestion de ce format.

6. Suppression de contacts

Les contacts ne sont pas éternels et garder des contacts qui ne sont pas utile n'est pas forcément une bonne idée. C'est pourquoi il est possible de retirer un/des contact de la base de données.

7. Validation des entrées

Afin d'être certain que les valeurs renseignées par l'utilisateur sont bien validées, un système de validation par type de champ est en place. Ainsi pour une adresse email, il n'est pas possible de mettre autre chose, qu'une adresse email. Dans ce cas la validation est effectuée sur la forme, et non pas la réelle validité de l'adresse. Il en va de même pour les numéros de téléphone.

8. Recherche de contacts

Lorsque la base de données devient grande il peut être intéressant de chercher un contact dans celle-ci. Cette fonctionnalité est principalement utilisée pour rechercher un contact avant l'édition.

9. Conversion de contacts

Cette fonctionnalité est la combinaison entre de l'importation et l'exportation. En effet, une fois importé sur un format commun (celui de la base de données) il devient alors très facile d'exporter ces données sur un autre format. La conversion est donc la succession d'une importation et d'une exportation. Cela permet une grande modularité sur le format d'importation et d'exportation.

10. Modularité

Dans ce logiciel la modularité est quelque chose qui est important. Sa conception fait qu'il est facile d'ajouter un nouveau format d'exportation ou un nouveau type de champ. De cette façon, le logiciel est capable d'évoluer dans le temps. De nouveau format d'exportation ont donc été développés pour ce projet afin de démontrer cette capacité. Aussi, les contacts ont un nombre de propriété qui n'est pas fixé, en plus de fonctionnalité de base il était possible d'ajouter un nombre infini de champ mais à cause d'une erreur de dernière minute, il n'a pas été possible de corriger et donc la fonctionnalité a été désactivée.

11. Ergonomie

Ce programme est conçu pour permettre une utilisation des plus simple qu'il soit. Il dispose donc de 3 interfaces qui sont : l'interface interactive, l'interface web et enfin une interface en ligne de commande.

L'interface la plus simple d'utilisation reste l'interface web puisqu'elle est très visuelle. L'interface interactive est destinée aux personnes qui ne veulent pas d'interface graphique lourde mais être tout de même guidé dans les étapes pour ajouter supprimer éditer un élément de l'annuaire.

Les interfaces sont présentées dans la suite de cette documentation.

C. Formats et types de champs disponible

1. Formats

Ce programme supporte un certain nombre de formats d'exportation/importation tel que

- Zip
- Csv
- Vcard
- Pdf (uniquement en exportation)
- Ldif
- Json

Afin de normaliser les valeurs et les vérifications, des types de champs ont été ajouté, ils permettent une gestion plus simplifier des différentes données qui peuvent être renseigné.

2. Types de champs

Voici les types par default ainsi qu'une courte explication sur la vérification effectuée, et le rôle du champ.

- Phone : champ approprié pour stocker des numéro de téléphone, la vérification effectuer permet aussi de formater le numéro sur le format international (+xx xxx xxx xxx)
- Basic : Ce champs est le champ il plus primitif, il permet de stocker une simple chaine de caractère sans vérification particulière
- Date : Ce champ permet la saisie de date mais aussi de valider leur bonne forme. Toute les date doivent être au format jj/mm/aaaa.
- Number : Ce type permet de valider le fait que l'utilisateur entre bien un nombre (potentiellement flottant)
- Image : Ce champ est un des plus complexe, en entrée de ce champ il est possible de fournir un fichier local, ou une url vers un image de type JPEG uniquement. La valeur qui est mise en mémoire n'est pas le chemin mais la version base64 de l'image. On voit tout m'intérêt d'avoir des types de champs particulier, la logique de traitement peut être différente d'un type a un autre.
- Name : Ce type-là permet simplement d'appliquer une mise en forme pour un nom propre (première lettre en majuscule)
- Email : Ce type permet de valider essentiellement le format de l'adresse entré par l'utilisateur. Il doit respecter le standard d'une adresse email à savoir chose@domain.com

D. Prise en main des interfaces

Dans cette partie seront expliqué les différentes façons de rentrer en interaction avec le logiciel. Elles sont au nombre de trois et offre à l'utilisateur la possibilité de choisir celle qui convient le mieux à son besoin. Ex : pour une gestion ergonomique on utilisera interface web, dans le cas d'une intégration dans des script de type bash on utilisera une CLI simple. Et dans le cas où l'on ne possède pas d'interface graphique, une CLI interactive est disponible. Ci-dessous vous trouverez des explications quant à l'utilisation de ces dernières

1. Ligne de commande simple

Cette interface permet une intégration dans divers scripts tel que le BASH. En effet toutes les fonctionnalités du programme sont disponibles via un ensemble d'argument qui peuvent s'ajouter. Voici les commandes disponibles (à noter la commande '-h', l'aide est disponible à tout moment et permet de visualiser les commandes disponibles) :

La visualisation des formats et champs disponibles, peut se faire en utilisant les arguments -F pour les formats, et -f pour les champs.

Il en résulte une liste :

```
Available formats
- zip
- pdf
- csv
- ldif
- vcard
- json
```

Figure 2 - Résultat de la commande "main.py -F"

```
Available fields
- ImageField
- BasicField
- DateField
- PhoneField
- NumberField
- EmailField
- NameField
```

Figure 1 - Résultat de la commande "main.py -f"

Les autres fonctionnalités du programme sont disponibles au travers un certain nombre de sous commande qui sont présentées ci-dessous :

- Search : cette option permet d'effectuer des recherches dans la base de données. Cette fonctionnalité demande 2 arguments supplémentaires, le fichier de base de données et un mot clé pour effectuer la recherche (Attention la recherche n'est pas sensible à la casse). Cette commande retourne un code d'erreur 10 si elle ne trouve pas de contacts à afficher.
Voici un exemple de commande : `main.py search data\base.csv Vil`
Ici `data\base.csv` correspond au fichier de base de données, et `Vil` correspond au mot clé à chercher. Dans le cas de cet exemple, « 0 » est l'identifiant numérique du contact. Il pourra être utilisé pour effectuer des modifications par la suite.

```
0 - [Nom]: Villena
    [Prenom]: Guillaume
    [Telephone 1]: +33 06789996
    [Telephone 2]: +33 38665986
    [Email]:
    [Date de naissance]:
    [Adresse 1]:
    [Adresse 2]:
    [Code postal]: 06410
    [Ville]: Biot
    [Pays]: France
    [Photo]:
```

Figure 3 - résultat de la recherche

- Cli : Cette commande permet de lancer l'interface ligne de commande interactive. Cette interface sera détaillée un peu plus loin dans ce document, mais il est important de savoir qu'il est possible de donner un argument pour charger directement une base. Voici un exemple : `main.py cli -b data\base.csv`
Dans le cas où la base n'est pas spécifiée, l'interface interactive vous proposera de créer ou charger une base de données.
- Import : Cette commande permet d'importer dans la base de données un contact venant d'un format dont la lecture est possible. En conséquence il est nécessaire de fournir en paramètre une base de données, ainsi que le fichier à importer.
Dans le cas où le format n'est pas supporté le programme retourne le code d'erreur 11.

Voici un exemple de commande pour importer un fichier de type vcard :
`main.py import data\base.csv VG.vcf`

- Export : Cette commande permet l'exportation de tout ou une portion des contacts présent dans la base vers un format en particulier.

La commande se présente de cette façon :

`main.py export [-s SELECT_CONTACT] [-z] base TO`

L'utilisation des identifiants unique prend ici son sens, l'argument « -s » peut prendre une suite de valeur numérique séparée par des virgules par exemple « 1,5,9,13 » va sélectionner les contacts 1, 5, 9 et 13 pour l'exportation, tout en supposant qu'ils existent. Si tout de même vous souhaitez exporter toute la base de données, il suffit de ne pas renseigner cette option.

L'option « -z » permet dans le cas de la création de plusieurs fichiers (format vcard par exemple) de les regrouper dans une archive zip.

« Base » correspond au chemin vers le fichier de la base de données, et « TO » correspond au chemin vers un fichier/dossier pour l'exportation.
Si le format d'exportation est inconnu, le code d'erreur 10 sera utilisé.

Voici un exemple de commande : `main.py export -s 1,9 data\base.csv ./export.csv`

- Display : Cette commande permet d'afficher tous les contacts, ou de la même façon que la commande export il est possible d'effectuer une sélection plus précise des contacts, pour afficher certains contacts.

Il est nécessaire de renseigner la base de données pour que le programme sache à quel endroit se trouvent les contacts.

Voici un exemple de commande : `main.py display data\base.csv -s 0,1`

- Web :

Cette commande permet de lancer le serveur web intégré au projet. Il fait partie des interfaces utilisateur disponibles, et c'est sans doute l'interface la plus ergonomique. Une partie entière est dédiée à cette interface. Il est à noter que cette commande accepte deux options : « -p » et « -ip ». -p permet de choisir le port d'écoute du serveur web, et -ip permet de choisir sur quelle interface le serveur va écouter. Par défaut, le port est 8089 et l'IP d'écoute est 127.0.0.1

- Convert: Cette commande permet le passage d'un format de contact à un autre. Elle prend deux paramètres, le chemin vers le fichier de source (option « -f »), et le chemin vers le fichier de sortie sur un autre format (option « -t »). Il est possible d'ajouter l'option « -z » afin de réunir les éventuels multiples fichiers au sein d'une archive de type zip.

Voici un exemple de commande :

`main.py convert -f path/input.ldif -t path/export/vcards -z`

- Contact (gestion des contacts) : Cette commande permet la gestion des contacts dans la base. Il est possible d'ajouter, modifier, supprimer un contact depuis cette commande.
 - Pour ajouter un contact, il faut simplement utiliser la commande :
`main.py contact path/to/base.csv -a`

L'ajout de contact se déroule sous forme d'une série de question. Auxquelles il faut répondre pour remplir les différentes valeurs du contact.

A noter : les champs suivis par une étoile sont des champs obligatoires, cela signifie qu'il n'est pas possible de les laisser vide. Pour les autres champs la validation décrite plus tôt s'applique.

```
(Nom)* : Fedh
(Prenom)* : Thib
(Telephone 1) :
(Telephone 2) :
(Email) :
(Date de naissance) :
(Adresse 1) :
(Adresse 2) :
(Code postal) :
(Ville) : Biot
(Pays) : France
(Photo) :
Voulez vous ajouter de nouveaux champs ? (Ces champs peuvent ne pas être présent dans certain format d'exportation)(O/N) : n
```

Figure 4 - Remplissage des champs

- Pour éditer un contact, il suffit de se prémunir de l'identifiant du contact et de le donner dans la ligne de commande (il est aussi possible de demander l'édition de plusieurs contacts en même temps, pour cela il suffit de séparer les identifiants des contacts, par une virgule)

Voici un exemple de commande (1 est un identifiant de contact) :

```
main.py contact path/to/base.csv -e 1
```

L'édition d'un contact est très proche de l'ajout d'un contact, en réalité c'est le même « questionnaire » qui est utilisé mais les anciennes valeurs sont présentées.

Voici un exemple en mode édition :

```
(Nom)*[Delaporte] :
(Prenom)*[Dyulan] :
(Telephone 1) :
(Telephone 2) :
(Email) :
(Date de naissance) :
(Adresse 1) :
(Adresse 2) : 785
(Code postal)[06450] :
(Ville)[Valbonne] :
(Pays) :
(Photo) :
Voulez vous ajouter de nouveaux champs ? (Ces champs peuvent ne pas être présent dans certain format d'exportation)(O/N) : n
```

Figure 5 - Mode édition de contact

- Pour supprimer un contact, il faut se prémunir de l'identifiant unique du contact et utiliser l'option « -d » dans la ligne de commande.

Voici un exemple pour supprimer le contact avec l'identifiant 1 :

```
main.py contact path/to/base.csv -d 1
```

- Clear : Cette commande permet de vider entièrement la base de données. Elle a tout de même besoin du chemin vers la base de données.

Voici un exemple de commande : `main.py clear data/base.csv`

2. Ligne de commande interactive

Pour utiliser l'interface interactive en ligne de commande, utilisez l'option « cli » lors du lancement du programme. (`main.py cli`)

1) Menu principal avant chargement d'une base

Le premier menu qui s'affiche est un menu qui vous permet de charger / créer une base de données. De plus elle donne la possibilité de convertir des données.

Le menu se présente comme ceci :

```
-----Contact Manager-----
1 - Charger une base
2 - Nouvelle base
3 - Convertir des données
4 - Quitter

Entrez votre choix :
```

Figure 6 - interface principale CLI

Dans un premier temps, et pour le premier lancement, aucune base de données existe. C'est pourquoi il est intéressant d'utiliser l'action n°2 pour créer une nouvelle base de données. Le programme vous demande alors de renseigner un chemin valide (absolu ou relatif) vers la base de données. Les bases sont créées dans le format CSV et par la suite vous aurez la possibilité de passer à un autre format grâce l'option « convertir des données » ou encore en exportant les données.

Les options ici sont relativement explicite. Il faudra utiliser l'option 1 pour charger une base de données précédemment créée. L'option « conversion de données » permet d'outrepasser l'utilisation d'une base de données dans le cas où elle n'est pas nécessaire. Ici le programme propose en plus de convertir les donnée une possibilité de gestion de l'annuaire.

2) Menu de gestion de l'annuaire

Voici le deuxième menu :

```
-----Contact Manager-----
1 - Visualiser la base
2 - Exporter un/des contacts
3 - Importer un/des contacts
4 - Ajouter un contact
5 - Editer un contact
6 - Supprimer un contact
7 - Convertir des données
8 - Quitter

Entrez votre choix :
```

Figure 7 - Menu de gestion de l'annuaire

Ici détaillons les diverses options :

1. Visualiser la base :

Cette option permet de visualiser l'ensemble de la base donnée. Voici un exemple de ce que le programme produit en sortie :

```
Liste des contacts
----- Contact -----
[Nom]: Villena
[Prenom]: Guillaume
[Telephone 1]: +33 06789996
[Telephone 2]: +33 38665986
[Email]:
[Date de naissance]:
[Adresse 1]:
[Adresse 2]:
[Code postal]: 06410
[Ville]: Biot
[Pays]: France
[Photo]:

----- Contact -----
[Nom]: Delaporte
[Prenom]: Dyulan
[Telephone 1]:
[Telephone 2]:
[Email]:
[Date de naissance]:
[Adresse 1]:
[Adresse 2]: 785
[Code postal]: 06450
[Ville]: Valbone
[Pays]:
[Photo]:
```

Figure 8 - Visualisation de la base

2. Exporter un/des contacts

```
Selection du format d'exportation
1 - zip
2 - ldif
3 - json
4 - vcard
5 - pdf
6 - csv
Votre choix:
```

Figure 9 - Sélection du format d'exportation

en version simplifié (uniquement les champs important) afin de vous permettre de faire un choix parmi tous les contacts. Ici pour choisir plusieurs contacts, il suffit d'écrire les n° des contacts séparés par des virgules.

L'étape qui suit est le choix du chemin dans lequel le fichier doit être exporté. Il peut être absolu ou relatif, cela n'a pas d'importance pour le fonctionnement du programme.

Cette deuxième option vous permet d'exporter un ou plusieurs contacts vers divers formats. Ces formats ont été présentés plus haut. Pour exporter choisissez l'option n°2 dans le « Menu de gestion de l'annuaire », et choisissez dans le menu qui suit le format qui vous convient parmi ceux qui sont proposés. L'étape d'après vous permet de déterminer quel contact doit être exporté. Ici la base de données est affichée

3. Importer un/des contacts

L'importation d'un contact est relativement simple le programme demande simplement le chemin du fichier à importer. Dans le cas où le chemin spécifié est un dossier il est possible d'importer plusieurs fichiers. Le programme vous demandera alors de renseigner une liste de nombres (correspondant au menu) séparés par des virgules.

4. Ajouter un contact

Ajouter un contact consiste en le remplissage d'un formulaire dans lequel on demande chaque champ un par un. Les champs tels que le nom et le prénom et ceux qui présentent un astérisque sont des champs obligatoires. Il faut garder à l'esprit que les champs sont soumis à la validation décrite plus haut dans ce document.

A la fin du processus, on vous demande s'il faut ajouter des champs supplémentaires. Des champs qui ne sont pas présents dans les entêtes prédéfinies. Il est possible d'en ajouter une infinité. Tout en choisissant des types appropriés.

5. Edition d'un contact.

Voici une utilisation de la fonction de recherche. Cette fonction de recherche permet dans le cas de l'édition de contact de rechercher le contact à éditer dans la base afin d'éviter de devoir choisir dans une grande liste

6. Suppression de contact

La suppression de contact exploite la notion d'identifiant unique. Il suffit de renseigner le numéro affiché en face de chaque résumé de contact. Attention la suppression est définitive.

7. Conversion de données

Cette fonctionnalité permet de convertir des données provenant des formats supportés vers un autre format supporté. On rentre dans un premier le chemin du fichier à convertir puis on donne le format de destination.

3. Interface Web

L'interface web offre la possibilité d'utiliser ce logiciel à distance, mais aussi de permettre à un ensemble d'utilisateur de faire des recherches, et effectuer des actions sur la base de données. L'accès à la base par le web exige une authentification, et une légère configuration. Découvrez ici comment mettre en place cette interface.

L'interface web permet d'une manière très ergonomique de gérer l'annuaire. Elle offre globalement les mêmes possibilités que les autres interfaces.

Pour commencer lancez le programme avec les paramètre pour l'interface web : `main.py web -p 80 -ip 127.0.0.1 data\base.csv`

Lorsque vous vous connectez sur l'interface vous obtenez un page de connexion comme celle-ci

[Contact manager](#) [Liste](#) [Recherche](#) [Gestion](#) [Convertir des données](#)

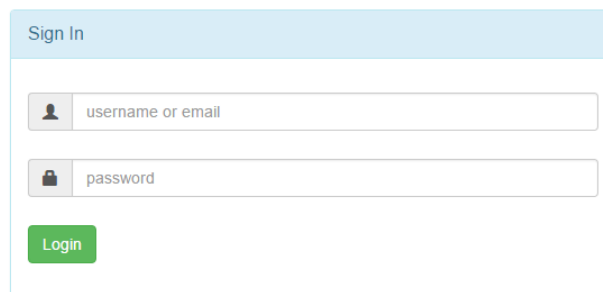


Figure 10 - interface de connexion pour l'annuaire

Connectez-vous avec l'identifiant Admin et mot de passe : password

-  [Accueil](#)
-  [Recherche](#)
-  [Importation](#)
-  [Exportation](#)
-  [Ajouter un contact](#)

Vous obtenez alors une page avec le listing des contacts dans la base.

Dans le menu à gauche vous trouverez les principales actions de la gestion de l'annuaire.

En revanche toutes les autres fonctionnalités sont présente. La fonctionnalité de champ illimité n'a pas encore été implémenté, mais si le temps l'avais permis elle aurait été présente.

Les fonctionnalités de recherche édition/exportation/ajout sont quand a-t-elle fonctionnelle.

L'interface web se veut intuitive il n'y a donc pas grand-chose à apporter comme information pertinente.

Figure 11 - menu de l'interface web

II. Documentation des fonctions

A. Construction générale du projet

Le projet est constitué, de 2 principaux dossiers :

- App
- appData

Dans ces dossiers se trouvent les fichiers principaux pour le fonctionnement de l'application. A cela s'ajoute un fichier principal "main.py" qui permet l'exécution des différentes interfaces de l'application.

Dans le cas du dossier "apps" on trouvera à l'intérieur les trois applications WebApp, CLiApp, CmdApp

Ces fichiers contiennent le code de cœur des applications, la logique de leur fonctionnement.

Dans le dossier appdata, on trouve toutes les classe qui permettent le fonctionnement du projet.

La classe Contact est une classe contient la définition d'un contact ainsi qu'un certain nombre d'éléments/ fonction pour la manipulation des données.

Le projet est conçu sur de l'orienté objets. Tout le code python utilise donc cette notation en objet. Cela simplifie dans certain cas la manipulation des données.

Les types de champs et les descriptions des formats d'entrée sortie, dossier « field_type » et « io_format » sont des conteneurs pour les formats et les type.

Toute les classe dans ces dossiers sont auto chargée. Cela a pour avantage la modularité. Ainsi si un nouveau type doit être écrit, il sera chargé automatique sans aucune modification du projet. Il en va de même pour les entrées sorties.

Dans le dossier « webdata » on trouve les données statiques tel que les document css html et javascript. Dans le dossier hadlers on trouvera la logique des pages, traitement des get/post, stockage des donnée ...

Afin de simplifier la gestion des menu une classe a été créée afin de pouvoir créer en quelques ligne la validation des menus ainsi que l'appel d'une fonction de callback. On trouve cette classe dans le dossier appData sous le nom de EasyMenu. Cette classe, permet aussi la sélection multiple dans un menu. Elle est exploitée presque partout dans la CLI interactive.

Il y a peu de commentaire dans le code, mais les noms de variable et de fonction/package ou encore classe sont explicité.

B. Ecrire un nouveau format pour le gestionnaire

Pour écrire un nouveau format pour le gestionnaire d'annuaire, il faut écrire une nouvelle classe dans le dossier webData/io_format. Voici un exemple de la structure obligatoire d'une t'elle classe.

```
class MonInpuOutput:
    EXT = ".abc" # remplacer avec l'extention du format

    @staticmethod
    def export_data(contacts, file_path: str):
```

```

    if not file_path.endswith(MonInpuOutput.EXT):
        file_path += MonInpuOutput.EXT

    folder = os.path.dirname(file_path)
    if not os.path.exists(folder):
        os.makedirs(folder)

    # ecrire ici la logique pour ecrire le fichier dans le format voulu

    return file_path

@staticmethod
def import_data(file_path):
    temp_contacts = []
    #Ecrire ici la logique pour importer un contact
    return temp_contacts

instance = {"format": "abc", "class": MonInpuOutput } # remplacer format par le
format tel qu'il sera afficher dans le projet ( ex :.jpg deviendra jpeg )

```

Comme on peut le voir la structure est simple. Le format est défini par deux fonction statique : import_data et export_data. Import_data contient un paramètre qui correspond au fichier qui est transmis pour importer.

Dans le cas d'export_data, la fonction contient deux paramètres que sont le fichier de sortie, et un tableau de contact.

C. Ecrire un nouveau type de champs pour le gestionnaire

Si vous souhaitez écrire un nouveau champ pour le gestionnaire, il suffis de placer une nouvelle classe dans le dossier appData/field_type. Cette classe aura cette forme :

```

class MonField(BasicField):
    DESCRIPTION = "Mon Champ simple sans validation particulière"
    NAME = "Texte"

    def __init__(self, write_name, name="", value="", canskip=False):
        super().__init__(write_name, name=name, value=value, canskip=canskip)

    def set_name(self, name: str):
        self.name = name

    def set_value(self, value: str):
        self.value = value

    def get_name(self):
        return self.name

    def get_WName(self):
        return self.writable_name

    def get_value(self):
        return self.value

    def is_valid(self, value=""): # fonction qui doit retourner un Boolean et
    qui permet de determiner si l'entrée est valide.
        return True

    def get_error_message(self):

```

```
return 'C\'est une erreur'
```

```
instance = {"type": "MonField", "class": MonField}
```

ici la classe est un peu plus complexe et complète. Il est impératif de réécrire les fonction `is_valid` et `set_value`. `is_valid` sera appelé a chaque entrée de l'utilisateur et détermine si l'on doit ou pas faire boucler l'entrée pour que celui-ci entre une valeur valide. `set_value` est un fonction qui permet de définir la valeur réelle du champ. Dans le cas d'une image la valeur en entrée est une url ou chemin vers un fichier et la valeur réellement écrite dans la mémoire est une valeur en base64 de l'image.