

Table des matières

Installation	1
Environnement	1
Bibliothèques Python	2
Réglages	4
Les données à afficher	4
Contenu du mail	4
Paramétrages serveur mail	5
Destinataires	5
Fonctionnement	5
Interactions script/utilisateur	6
Traitement des données	7
Passage en html	8
Construction du mail	9
Partage	10
Enregistrement	10

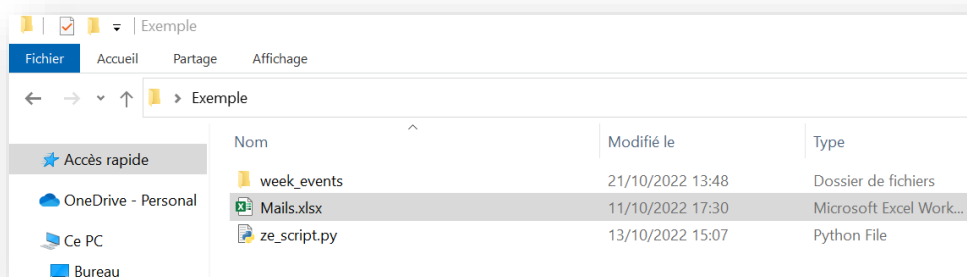
Installation

Le bon fonctionnement du programme requiert un environnement spécifique. Ci-dessous veuillez trouver des explications sur comment mettre en place cet environnement et sur ce qu'il faut installer.

Environnement

Le script python doit être placé dans un fichier avec d'autres éléments :

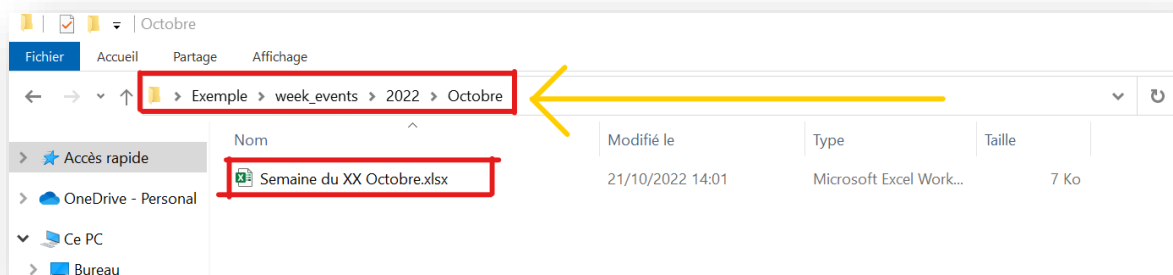
- La liste des destinataires enregistrée dans un Excel
 - Ce fichier doit s'appeler "Mails.xlsx", sans quoi le programme ne marchera pas
- Un dossier qui contient les données brutes et celles enregistrées
 - Ce dossier doit s'appeler "week_events", sans quoi le programme ne marchera pas



Cette installation permet au programme d'accéder aux fichiers et de les manipuler facilement.

Le dossier "week_events" remplit deux fonctions :

- Avant le lancement du programme, il contient le fichier xlsx avec les données brutes extraites de Bloomberg
- À la fin du programme, le fichier initial est supprimé, et, les données du tableau qui est envoyé par mail sont enregistrées dans une base de données hiérarchisée de la sorte:
 - Année (dossier)
 - Mois (dossier)
 - Semaine (fichier xlsx)



Bibliothèques Python

```
#ces premières lignes nous permettent d'importer les bibliothèques python dont on se sert au cours du programme
import os
import glob
import os.path
import pandas as pd
#pd est un shortcut d'appel de la fonction
from xbbg import blp
from easygui import *
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
#from = import d'une fonction particulière de la bibliothèque
import smtplib, ssl
import datetime
from calendar import month_name
pd.options.mode.chained_assignment = None
import webbrowser
```

En installant Python sur votre machine, certaines bibliothèques sont incluses, d'autres non, le cas échéant il faut les télécharger soi-même. Ci-dessus, vous pouvez voir celles dont nous avons besoin. Pour voir si celles-ci sont bien installées vous avez plusieurs méthodes :

- Dans l'invite de commandes, entrez :
 - Python
 - help('modules')

Ceci vous indiquera les modules installés dans votre environnement actuel.

```

U:\>python
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> help('modules')

Please wait a moment while I gather a list of all available modules...

C:\Program Files\WindowsApps\PythonSoftwareFoundation.Python.3.10_3.10.2288.0_x64__qbz5n2kfra8p0\
rimental. See NEP 47.
__import__(info.name)
C:\Program Files\WindowsApps\PythonSoftwareFoundation.Python.3.10_3.10.2288.0_x64__qbz5n2kfra8p0\
s replacing distutils.
warnings.warn("Setuptools is replacing distutils.")
PIL                binascii            mmapfile            sunau
PyQt5              binhex              mmsystem            symtable
__future__         bisect              modulefinder        sys
__abc              blpapi              msilib              sysconfig
__aix_support      builtins            msvcrt              tabnanny
__ast              bz2                 multiprocessing     tarfile
__asyncio           cProfile            names                telnetlib
__bisect           cachetools          netbios              tempfile
__blake2            calendar            netrc                 tenacity
__bootsubprocess   certifi             nntplib              textwrap

```

Vérifiez si les bibliothèques requises sont bien installées

- Autrement, utilisez la commande `import` suivi du nom d'une bibliothèque, l'interpréteur Python indiquera une erreur si la bibliothèque n'est pas installée.
 - Dans le cas où une bibliothèque n'est pas installée, ouvrez l'invite de commandes et entrez `pip install nom_de_la_bibliothèque`

```
import la_bibliothèque
```

```

-----
ModuleNotFoundError                                Traceback (most recent call last)
Input In [4], in <cell line: 1>()
----> 1 import la_bibliothèque

ModuleNotFoundError: No module named 'la_bibliothèque'

```

Invite de commandes

```

Microsoft Windows [version 10.0.19042.1586]
(c) Microsoft Corporation. All rights reserved.

U:\>pip install la_bibliothèque

```

Particulièrement, la bibliothèque xbbg requiert l'API Bloomberg, il faut donc utiliser deux fois pip install :

- `pip install blpapi --index-url=https://bcms.bloomberg.com/pip/simple/`
- `pip install xbbg`

Réglages

Vous pourriez avoir besoin de modifier le fonctionnement du programme avant son utilisation, veuillez trouver ci-dessous ce qui peut être intéressant à modifier dans le code pour adapter son fonctionnement à vos besoins.

Les données à afficher

L'extraction des données a lieu à partir d'un fichier Excel, dont le programme n'exploite que certaines colonnes. Si vous souhaitez modifier cette liste de colonnes et exploiter plus de données, vous devez l'indiquer dans le programme dans la variable *bonne_liste*.

```
bonne_liste = ['Name', 'Ticker', 'Date', 'MKTTime (CET Time)', 'Description']
```

Contenu du mail

Vous pouvez modifier le contenu du mail avec plusieurs éléments :

- `msg['From']` : destinataire, indiquez une adresse mail
- `msg['To']` : destinataire(s), indiquez une ou des adresses mail
- `msg['Subject']` : objet du mail
- `msg['Cc']` : les adresses mail à mettre en copie

Pour modifier le tableau HTML :

```
def dataframe_to_html(dataframe, table_name):
    #ici on converti le dataframe en html
    zeheader = list(dataframe)
    #enregistrement des noms des colonnes du tableau
    zerangées = dataframe.values.tolist()
    #ici chaque rangée est transformée en liste qui sont mis dans une grande liste
    header = ""
    tablesplit = ""
    for i in range(len(zeheader)):
        for i in range(len(zeheader)):
            #definit le nombre de colonnes
            header = header + "
            <th style='text-align:center;background:#B10C24;color:white;padding: 2px;padding-left: 2 px;padding-center: 2 px;'>"" + str(zeheader[i]) + ""</th>
            ""
    for item in zerangées:
        tablesplit = tablesplit + ""
        <tr>
        ""
        for i in range(len(item)):
            tablesplit = tablesplit + ""
            <td style='text-align:center;border: 1px solid #CCC;padding : 2px;padding-left: 2 px;padding-center: 2 px;'>"" + str(item[i]) + ""</td>""
            tablesplit = tablesplit + ""
        </tr>
        ""
    html_table = ""
    <table id={} width=50% padding=5px cellspacing=0 cellpadding=1.5 color: #333;style='font-family: Calibri Light, Arial, sans-serif;font-size: 11pt;border-coll
    <tr>
    {}
    </tr>
    {}
    </table>
    ""
    return html_table
```

La fonction ci-dessus indique comment le data frame pandas est converti en un tableau html, dans le cas où vous souhaiteriez adapter son format (police, couleurs etc...), indiquez le code html selon vos préférences, cependant, veuillez à indiquer ces instructions ligne par ligne si vous souhaitez envoyer ce contenu par email.

Paramétrages serveur mail

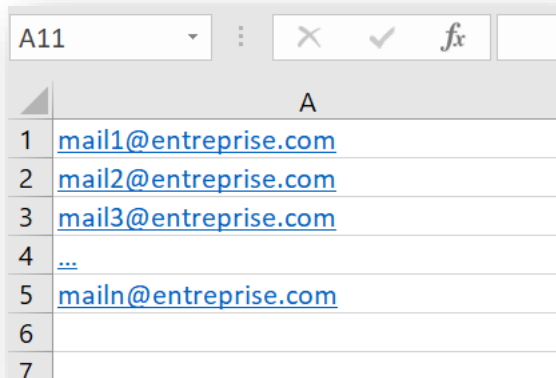
Le script utilise des fonctions pour utiliser le protocole smtp grâce à la bibliothèque Python correspondante. Deux éléments pourraient avoir besoin d’être modifiés :

- Courriel émetteur, indiquez le votre
- Code du serveur smtp, indiquez celui de votre entreprise

```
sender_email = "Exec Desk Python [redacted]"  
smtp_server = '[redacted]'
```

Destinataires

Comme vu dans l’installation du programme, vous avez mis en place un dossier qui contient le script et un fichier “Mails.xlsx”. Ce fichier est lu par le script et contient la liste des personnes à qui vous voulez envoyer votre tableau des évènements de la semaine.

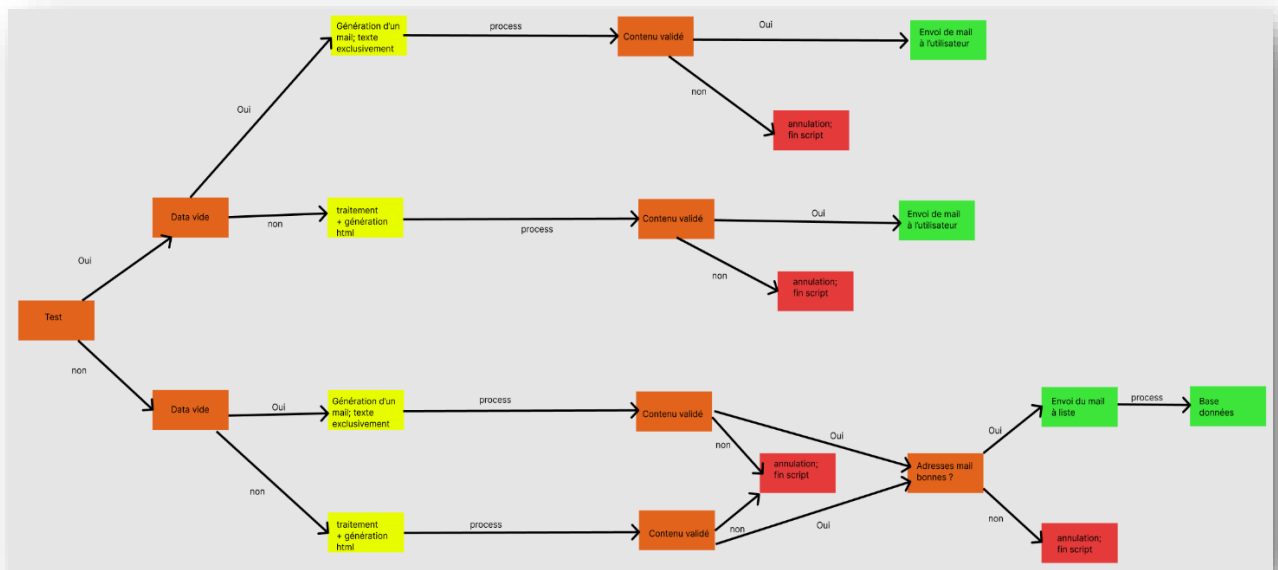


	A
1	mail1@entreprise.com
2	mail2@entreprise.com
3	mail3@entreprise.com
4	...
5	mailn@entreprise.com
6	
7	

Il suffit de tenir ce fichier de destinataires à jour, le script fera parvenir le mail à toutes les personnes qui y sont indiquées, n’indiquez pas de header. En cas de modification, veuillez à bien le refermer avant tout lancement du programme.

Fonctionnement

Le script repose en partie sur une interaction du script avec son utilisateur. Plutôt qu’imprimer des messages d’erreur qui peuvent être difficiles à comprendre, le script fait vérifier le contenu qu’il produit à la personne qui l’a lancé. Le script prévoit aussi plusieurs cas particuliers qui seront détaillés plus bas.

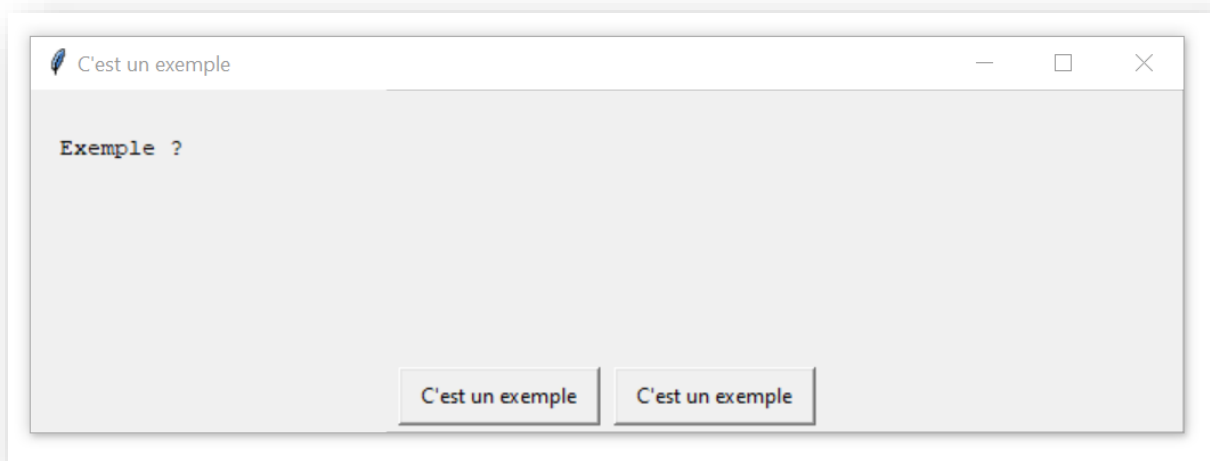


Code couleur :

- **Orange** : interaction script/utilisateur
- **Jaune** : Travail du script
- **Rouge** : annulation
- **Vert** : processus complété

Interactions script/utilisateur

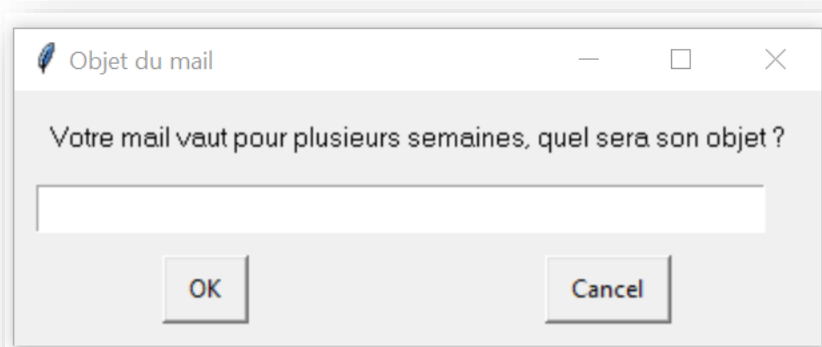
La bibliothèque *easygui* de Python permet de demander au script de faire apparaître différents types de boîtes de texte. Comme celle ci-dessous qui permet d'indiquer un choix entre deux options.



Sur le schéma plus haut, sont indiquées les principales interactions, d'autres interviennent dans des cas spécifiques et n'y apparaissent pas, mais elles ne diffèrent aucunement dans leur fonctionnement.

Interactions :

- **Test** : indique si le programme lancé est un test pour s'assurer de son bon fonctionnement, la différence fondamentale est, que dans le cas test=Oui, le programme ne va pas interagir avec la liste de destinataires et ne va envoyer de mail qu'à la personne effectuant le test.
- **Data** : indique s'il y a des données à traiter. Il peut y avoir des semaines au cours desquelles aucun évènement n'aura lieu. Le cas échéant, le script saute ses étapes de traitement de données et génère un court message informant les destinataires.
- **Contenu validé** : le script au cours de son exécution fait apparaître dans le navigateur web une fenêtre qui présente le contenu du tableau html qui sera envoyé par email, si le contenu n'est pas satisfaisant, le script s'arrête.
- **Adresses mail** : de façon similaire à la validité du contenu, l'utilisateur vérifie la validité des adresses email des destinataires, le script s'arrête si l'utilisateur trouve une erreur.
- **Objet du mail** : demande à l'utilisateur si le fichier traité concerne les événements de la semaine à venir, plusieurs semaines ou autres, dans le cas où le fichier initial concerne autre chose qu'une seule semaine, il sera demandé à l'utilisateur d'indiquer l'objet du mail.



- En fin de script plusieurs options d'enregistrement du fichier final sont proposées.

Traitement des données

Un bloc du script est dédié à retraiter le fichier Excel des événements de la semaine, obtenu sur le terminal Bloomberg. Au début, le fichier est assez fourni, mais seules quelques informations nous intéressent : le nom de l'entreprise, son ticker, la date de l'évènement, l'heure relative au marché (ouverture/fermeture) et la nature de l'évènement.

À ces colonnes présentes dans le fichier initial, le script ajoute une colonne « montant », qui concerne spécifiquement les événements qui sont des sorties de dividendes. Le montant indiqué correspond à la somme reversée en devise locale par action détenue.

Name	Ticker	Date	MKTTime (CET Time)	Description	Amount
Akelius Residential Property A	AKELD SS	24/10/2022	07:30	Q3 2022 Earnings Release	
SEB SA	SK FP	24/10/2022	Aft-mkt	Q3 2022 Sales and Revenue Release	
Koninklijke Philips NV	PHIA NA	24/10/2022		Q3 2022 Earnings Release	
Engie SA	ENGI FP	24/10/2022		Q3 2022 Sales and Revenue Release	
Pearson PLC	PSON LN	24/10/2022	08:00	Q3 2022 Sales and Revenue Release	
Nokia Oyj	NOKIA FH	24/10/2022		Dividend Ex-Date	0.02
Akzo Nobel NV	AKZA NA	24/10/2022		Dividend Ex-Date	0.44

Ce montant est obtenu grâce à une fonction *bdp* utilisée dans python, ce qui est rendu possible par l'utilisation de la bibliothèque *xbbg*.

```
if df['Description'][ind] == "Dividend Ex-Date":
    try:
        df1 = blp.bdp(tickers = "{} Equity".format(df['Ticker'][ind]), flds = ["DVD_SH_LAST"])
        a = df1['dvd_sh_last'][0]
        df['Amount'][ind] = a
    except:
        print("BDP request failed, please check your script")
```

Passage en html

Pour s'afficher correctement dans une boîte mail, le tableau qui a été généré par le script doit être converti en html, selon une rédaction qui place le style html souhaité ligne par ligne.

```
def dataframe_to_html(dataframe, table_name):
    #ici on converti le dataframe en html
    zeheader = list(dataframe)
    #enregistrement des noms des colonnes du tableau
    zerangées = dataframe.values.tolist()
    #ici chaque rangée est transformée en liste qui sont mis dans une grande liste
    header = ""
    tablesplit = ""
    for i in range(len(zeheader)):
        #definit le nombre de colonnes
        header = header + "<th style='text-align:center;background:#B10C24;color:white;padding: 2px;padding-left: 2 px;padding-center: 2 px;'>"" + str(zeheader[i]) + "</th>"
    for item in zerangées:
        tablesplit = tablesplit + "<tr>"
        for i in range(len(item)):
            tablesplit = tablesplit + "<td style='text-align:center;border: 1px solid #CCC;padding: 2px;padding-left: 2 px;padding-center: 2 px;'>"" + str(item[i]) + "</td>"
        tablesplit = tablesplit + "</tr>"
    html_table = "<table id={} width=50% padding=5px cellspacing=0 cellpadding=1.5 color: #333;style='font-family: Calibri Light, Arial, sans-serif;font-size: 11pt;border-collapse: collapse;border-spacing: 0;'>"
    tablesplit = "<tr>"
    tablesplit = tablesplit + "</tr>"
    tablesplit = "</table>"
    return html_table.format(table_name, header, tablesplit)
```


Cette fonction prend le data frame du script et le décompose ligne par ligne, cellule par cellule pour formater correctement les données.

Unnamed: 0	Name	Ticker	Date	Time	Description	Amount
0	14 Cie Financiere Richemont SA	CFR SW	2022-09-21 10:00:00	NaN	Dividend Ex-Date	0
1	19 Eni SpA	ENI IM	2022-09-19 10:00:00	NaN	Dividend Ex-Date	0
2	15 TotalEnergies SE	TTE FP	2022-09-21 10:00:00	NaN	Dividend Ex-Date	0
3	10 IG Group Holdings PLC	IGG LN	2022-09-22 10:00:00	NaN	Dividend Ex-Date	0
4	13 Norsk Hydro ASA	NHY NO	2022-09-21 10:00:00	NaN	Dividend Ex-Date	0
5	12 SPIE SA	SPIE FP	2022-09-22 10:00:00	NaN	Dividend Ex-Date	0
6	11 Hargreaves Lansdown PLC	HL/ LN	2022-09-22 10:00:00	NaN	Dividend Ex-Date	0
7	9 Axfood AB	AXFO SS	2022-09-22 10:00:00	NaN	Dividend Ex-Date	0
8	4 TietoEVRY Oyj	TIETO FH	2022-09-23 10:00:00	NaN	Dividend Ex-Date	0
9	20 STMicroelectronics NV	STM FP	2022-09-19 10:00:00	NaN	Dividend Ex-Date	0

Name	Ticker	Date	MKTTime (CET Time)	Description	Amount
Akelius Residential Property A	AKELD SS	24/10/2022	07:30	Q3 2022 Earnings Release	
SEB SA	SK FP	24/10/2022	Aft-mkt	Q3 2022 Sales and Revenue Release	
Koninklijke Philips NV	PHIA NA	24/10/2022		Q3 2022 Earnings Release	
Engie SA	ENGI FP	24/10/2022		Q3 2022 Sales and Revenue Release	
Pearson PLC	PSON LN	24/10/2022	08:00	Q3 2022 Sales and Revenue Release	
Nokia Oyj	NOKIA FH	24/10/2022		Dividend Ex-Date	0.02
Akzo Nobel NV	AKZA NA	24/10/2022		Dividend Ex-Date	0.44

Le premier tableau présente un exemple d'un data frame *pandas* affiché dans *Jupyter*, le second est un tableau issu d'un data frame auquel la fonction présentée plus haut a été appliquée.

Construction du mail

Grâce au module *MIMEMultipart* de la bibliothèque python *email*, on peut définir l'objet msg qui dispose de plusieurs attributs :

- msg['From'] : destinataire
- msg['To'] : destinataire(s)
- msg['Subject'] : objet du mail
- msg['Cc'] : les adresses en copie

C'est aussi à cet objet msg qu'on attache le data frame converti en html en indiquant au préalable de traiter ce qu'on lui attache comme du html.

```
part = MIMEText(corps, 'html')
msg.attach(part)
```

Partage

Une fois que tous les attributs de l'objet `msg` ont été indiqués et que son corps formaté en html lui a été attaché, il reste à l'envoyer.

```
try:
    smtp = smtplib.SMTP(smtp_server)
    smtp.sendmail(sender_email, receiver_email.split(","), msg.as_string())
    smtp.close()
except(
    smtplib.SMTPConnectError,
    smtplib.SMTPDataError,
    smtplib.SMTPAuthenticationError,
    smtplib.SMTPException,
    smtplib.SMTPHeloError,
    smtplib.SMTPRecipientsRefused,
    smtplib.SMTPResponseException,
    smtplib.SMTPSenderRefused,
    smtplib.SMTPServerDisconnected) as lerreur:
    print( "Erreur = " + str(lerreur))
```

Les instructions ci-dessus, issues du module `smtp`, prennent comme argument des variables (`sender_email`, `receiver_email` et `msg`) dans lesquelles sont enregistrés le destinataire, les destinataires et l'objet `msg`, qui spécifiquement doit être considéré comme une chaîne de caractères.

Enregistrement

L'enregistrement a lieu en toute fin du programme pour permettre, si tout a marché, de conserver les données dans un ensemble de dossier hiérarchisés selon le modèle suivant : année > mois > semaine.

```
if save_ou_pas == True:
    month_path = year_path + '\\{}'.format(current_month)
    zefile = '\\Semaine_{}_{}'.format(current_day, current_month)
    zefile = zefile + '.xlsx'
    if os.path.exists(r'{}'.format(month_path)) == True:
        df.to_excel(month_path + zefile)
    elif os.path.exists(r'{}'.format(year_path)) == True:
        os.mkdir(month_path)
        df.to_excel(month_path + zefile)
    else:
        os.mkdir(year_path)
        os.mkdir(month_path)
        df.to_excel(month_path + zefile)
    os.remove(r'{}'.format(latest_file))
```

Le programme s'assure de la construction de cette base de données tout seul, l'utilisateur n'a pas besoin de mettre en place les dossiers.