

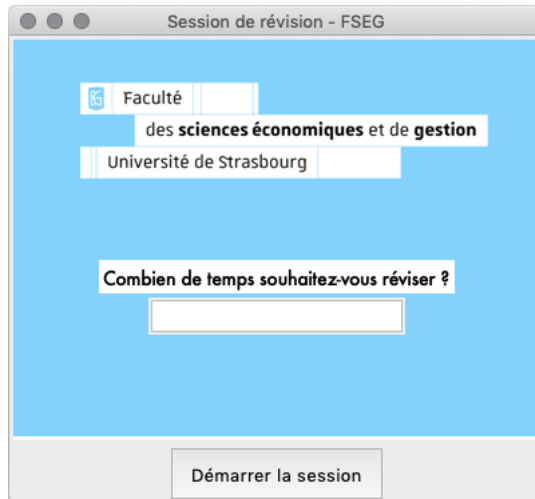
README

AUTOMATISATION D'UNE SESSION DE TRAVAIL

Filippo D'AGOSTINO

1 Utilisation de l'outil

Après paramétrage, ce programme permettra de lancer une session de révision qui consiste à ouvrir un ensemble de fichiers (logiciels, documents ou encore des liens internet) programmés à être fermés selon un temps prédéfini pour effectuer une pause et garder un bon rythme de travail. Voici l'interface de lancement :



Session de révision - FSEG

Faculté

Université de Strasbourg

Combien de temps souhaitez-vous réviser ?

Démarrer la session

2 Fonctionnement

Afin de montrer un exemple, paramétrons une session de révision comprenant trois sites internet : `ernest.unistra.fr`, `bu.unistra.fr` et `ecosia.org`, mais vous pouvez créer autant de combinaisons que vous souhaitez !

2.1 Bibliothèque

Tout d'abord voici la bibliothèque à importer :

```
import tkinter
import tkinter.messagebox
import threading
from selenium import webdriver
```

`tkinter` permettra de paramétrer l'interface de lancement du programme, *selenium* d'interagir avec les pages internet et *threading* d'arrêter la session.

2.2 Interface

Créez la première couche de l'interface :

```
root = tkinter.Tk()
root.title('Session de révision — FSEG')
root.geometry("400x350")
```

Importez le fond d'écran de la FSEG par le nom de chemin du fichier FE-FSEG :

```
logo = tkinter.PhotoImage(file = "...")
```

Formez la deuxième couche de l'interface permettant d'ajouter des éléments :

```
canvas = tkinter.Canvas(root, width = 410, height = 300, relief = 'raised',
canvas.pack(fill = "both", expand = True)
```

Ajoutez le titre :

```
label = tkinter.Label(root, text='Combien de temps souhaitez-vous réviser ?')
label.config(font=('futura', 14))
canvas.create_window(200, 180, window=label)
```

Ajoutez la boîte de saisie :

```
entry = tkinter.Entry(root)
canvas.create_window(200, 210, window=entry)
```

Enfin, ajoutez le fond d'écran :

```
canvas.create_image(0, 0, image=logo, anchor = "nw")
```

2.3 Programme d'exécution

Déclarez une variable vide afin de stocker par la suite le driver :

```
driver = []
```

Déclarez une variable vide afin de stocker par la suite le driver :

```
driver = []
```

Définissez une première fonction pour le lancement de la session :

```
def session():
    time = int(entry.get()) * 60
    start_time = threading.Timer(time, end)
    start_time.start()
```

Afin de convertir la saisie en minutes et lancer un timer correspondant à la saisie où lorsque le timer sera à la quantité de la saisie le deuxième programme de fermeture se lancera.

```
time2 = float(time) - 30
start_time2 = threading.Timer(time2, endmsg)
start_time2.start()
```

Afin de prévenir l'utilisateur que la session prendra fin 30s avant sa fermeture, le fonctionnement est le même que celui d'avant.

```
global driver
```

```
if not driver:
    driver = webdriver.Chrome(executable_path = '...')
    driver.get('https://ernest.unistra.fr')
    driver.execute_script("window.open('https://bu.unistra.fr/')")
    driver.execute_script("window.open('https://www.ecosia.org')")
    driver.switch_to.window(driver.window_handles[0])
```

Afin de lancer les pages internet il faut installer le driver chrome (lien : <https://chromedriver.storage.googleapis.com/index.html?path=100.0.4896.20/>) puis en faire en sorte à ce qu'elles restent ouvertes et la dernière ligne permet de revenir au premier onglet.

Définissez une deuxième fonction pour le lancement de la notification de fermeture :

```
def endmsg():
    tkinter.messagebox.showinfo("La session se termine",
    "Il reste 30s avant la fin de la session, courage!")
```

Définissez une troisième fonction pour la fermeture de la session :

```
def end():
    global driver

    if driver:
        driver.close()
        driver = []

    choice = tkinter.messagebox.askquestion(title = "Fin_de_la_session", message = "Voulez-vous fermer la session ?", type="yesno")
    if choice == "yes" : session()
    if choice == "no" : tkinter.messagebox.showinfo("Fin_de_la_session", "Arrêt de la session")
```

La procédure est identique pour l'ouverture avec désormais des instructions pour la fermeture des onglets suivi d'une proposition de continuer en "yes" (et répéter la même durée de session) ou arrêter en "no" en affichant un message de fin.

Définissez le bouton de lancement de la session :

```
Button = tkinter.Button(root, text='Démarrer la session', command=session,
```

Affichez le bouton de lancement de la session :

```
Button.pack()
```

Donnez la priorité à l'interface de lancement par rapport aux autres fenêtres ouvertes (optionnel) :

```
root.lift()
root.attributes('-topmost', True)
root.after_idle(root.attributes, '-topmost', False)
```

Terminez par la boucle principale :

```
root.mainloop()
```

3 Extensions

Pour interagir avec les fichiers et les logiciels de l'ordinateur il suffit d'ajouter les fonctions de *subprocess*, par exemple (sous MacOS) :

```
def TP():

    time = int(entry.get()) * 60
    start_time = threading.Timer(time, endTP)
    start_time.start()

    time2 = float(time) - 30
    start_time2 = threading.Timer(time2, endmsg)
    start_time2.start()

    subprocess.Popen(['open', '/Applications/RStudio.app'])
    subprocess.Popen(['open', '/Applications/Anaconda-Navigator.app'])

    global driverTP

    if not driverTP:

        driverTP = webdriver.Chrome(executable_path =
            '/Users/Filippo/Desktop/FSEG/TP/chromedriver.exe')
        driverTP.get('https://master-ds2e.github.io/ML-Programming/')
        driverTP.execute_script("window.open('https://stackoverflow.com')")
        driverTP.execute_script("window.open('https://www.ecosia.org')")
        driverTP.switch_to.window(driverTP.window_handles[0])

    def endTP():

        subprocess.call(['osascript', '-e', 'tell application "RStudio" to quit'])
        subprocess.call(['osascript', '-e', 'tell application "Anaconda-Navigator" to quit'])

        global driverTP

        if driverTP:
            driverTP.close()
            driverTP = []

        choice = tkinter.messagebox.askquestion(title = "Fin de la session",
            message = "C'est l'heure de la pause! Souhaitez-vous continuer?")
        if choice == "yes" : TP()
        if choice == "no" : tkinter.messagebox.showinfo("Fin de la session", "Aller à la pause")
```