gui_unifiée/class_TabBienvenue.py

```
import os
import sys

#from PyQt5.QtCore import *
#from PyQt5.QtGui import *
#from PyQt5.QtWidgets import *
from PyQt5.QtCore import *
from PyQt5.QtWidgets import QApplication, QWidget, QGridLayout, QTabWidget,
QHBoxLayout, QScrollArea, QLabel, QSizePolicy, QGroupBox

from class_Parametres import Parametres # Ne sert que si est exécuté
séparemment
from numpy import load


"""
PARAMETRES
"""
FICHIER_CONTENU_AIDE = './contenu_bienvenue/'


"""
Classe TabBienvenue, hérite de la classe QGridLayout, c'est donc une grille
Cette classe représente le contenu d'une fenêtre PyQt
Elle peut donc aussi être utilisée comme un onglet dans une fenêtre
@author Maylis et Alexandre
"""
class TabBienvenue(QGridLayout) :
    """
    Constructeur, crée le contenu de l'onglet
    """
    def __init__(self, objParams, parent=None) :
        super(TabBienvenue, self).__init__(parent) # Appel du constructeur
de QGridLayout

        self.objParams = objParams

        # Layout de la fenêtre
        gridLayout = QGridLayout()

        # Création d'onglets dans la page d'aide
        onglets = QTabWidget()
        onglets.setTabPosition(2)
        self.bienvenue_ongl = QWidget()
        self.sujet_original = QWidget()

        # Ajout dynamique des onglets à la page d'aide
        onglets.addTab(self.bienvenue_ongl,"Bienvenue")
        onglets.addTab(self.sujet_original,"Sujet Original")
        self.contenuOngletBienvenue("bienvenue_ongl.html",
self.bienvenue_ongl, margeHaut = False )
        self.contenuOngletBienvenue("sujet_original.html",
self.sujet_original)

        # Récupération des valeurs d'accélération et vitesse moyenne
        try :
            moyenne = load( self.objParams.genererURLInfos() )
        except FileNotFoundError :
```

```
            print( "[Erreur Bienvenue] Fichier introuvable : " +
self.objParams.genererURLInfos() )
            moyenne = [0, 0, 0, 0]

        # Récupérations des informations générales
        label_nombre = QLabel("Nombre de grains : " + str(int(moyenne[2]))
)
        label_volume = QLabel("Volume moyen des grains : " +
str(round(moyenne[3],2)) + " px³")
        label_vitesse = QLabel("Vitesse moyenne des grains : " +
str(round(moyenne[0],2)) + " px/t")
        label_acceleration = QLabel("Accélération moyenne des grains : " +
str(round(moyenne[1],2)) + " px/t²")


        # Groupe d'informations générales
        group_box = QGroupBox("Informations générales des grains")
        group_layout = QHBoxLayout()
        group_layout.addWidget(label_nombre)
        group_layout.addWidget(label_volume)
        group_layout.addWidget(label_vitesse)
        group_layout.addWidget(label_acceleration)
        group_box.setLayout(group_layout)

        # Insersion dans la fenêtre des éléments
        gridLayout.addWidget(onglets, 1,1)
        gridLayout.addWidget(group_box, 2,1)
        self.addLayout(gridLayout, 1, 1)

        # Rendre le layout avec les images plus gros que les autres
        gridLayout.setColumnStretch(1,1)

    """
    Affiche dans l'onglet indiqué le contenu du fichier .html associé
    """
    def contenuOngletBienvenue(self, nom_fichier, objet, margeHaut = True)
:
        # Contenant avec barre de scroll
        zone_de_texte = QHBoxLayout()
        scroll_area = QScrollArea()
        scroll_area.setHorizontalScrollBarPolicy(Qt.ScrollBarAlwaysOff)
        scroll_area.setWidgetResizable(True)

        # Ouverture du fichier contenant l'aide écrite au format HTML
        lien_fichier = FICHIER_CONTENU_AIDE + nom_fichier
        if os.path.isfile( lien_fichier ) : # Si le chemin d'accès existe
            fichier = open(lien_fichier, 'r', encoding='utf-8')
            texte = QLabel(fichier.read())
            if margeHaut : texte.setStyleSheet("QLabel { padding: 20px;
margin-top: -40px; }"); # CSS
            else : texte.setStyleSheet("QLabel { padding: 20px; margin-top:
-20px; }"); # CSS
        else :
            texte = QLabel("Le fichier suivant est manquant : " +
lien_fichier)
            texte.setStyleSheet("QLabel { padding: 20px; }"); # CSS
            print( "[Erreur TabAide] " + lien_fichier + " n'existe pas !" )

        # Ajustement de la forme du texte à la taille de la fenêtre
        texte.adjustSize()
        texte.setSizePolicy(QSizePolicy.Expanding, QSizePolicy.Expanding)
```

```
        texte.setWordWrap(True)
        texte.setAlignment(Qt.AlignJustify)
        scroll_area.setWidget(texte)

        # Ajout dans le contenant et dans l'onglet correspondant
        zone_de_texte.addWidget(scroll_area)
        objet.setLayout(zone_de_texte)


"""
Code principal pour démonstration
"""
# Si on est le script principal
# Cela permet de ne pas exécuter ce bloc de codes lorsque ce script est
importé par un autre
# Source : https://stackoverflow.com/questions/419163/what-does-if-name-
main-do
if __name__ == '__main__' :
    application = QApplication(sys.argv) # Crée un objet de type
QApplication (Doit être fait avant la fenêtre)
    fenetre = QWidget() # Crée un objet de type QWidget
    fenetre.setWindowTitle("MODE DÉMONSTRATION") # Définit le nom de la
fenêtre
    fenetre.setLayout( TabBienvenue( Parametres() ) )
    fenetre.show() # Affiche la fenêtre
    application.exec_() # Attendre que tout ce qui est en cours soit
exécuté
```