Collaboration au développement d'HABBY

Participer au développement Python d'HABBY

Préambule

Vous pouvez soumettre des demandes de modifications du code Python sur le projet HABBY https://github.com/YannIrstea/habby:

- Si vous souhaitez de nouvelles fonctionnalités.
- Si vous avez trouvé un bug et l'avez corrigé vous-même.

2021/03/12 08:48 · qroyer

Création d'un environnement Python

Dépendances

Le projet Python HABBY est hébergé sur Github accessible au lien suivant : https://github.com/YannIrstea/habby.

- Python >= 3
- Git

Aide à la création de l'environnement Python pour Windows

- Installer les dépendances.
- Télécharger la wheele GDAL : https://www.lfd.uci.edu/~gohlke/pythonlibs/#gdal
  - https://www.gisinternals.com/release.php
- Installer Microsoft Visual C++ 14.0 : https://visualstudio.microsoft.com/fr/visual-cpp-build-tools/
- Ouvrez le fichier 'creation_env_habby.bat' et spécifiez :
  - le chemin d'accès à votre Python système en remplaçant le chemin de la variable : 'python_source_path=' ;
  - le chemin d'accès à votre la wheel GDAL que vous avez préalablement téléchargée en remplaçant le chemin de la variable : 'gdal_wheel_path=' ;
  - sauvegardez le fichier.
- Lancer le fichier 'creation_env_habby.bat'.
- Si toutes les étapes se sont bien déroulées, vous devriez voir apparaitre la fenêtre principale d'HABBY.
- Votre environnement virtuel Python pour HABBY est prêt.

Aide à la création de l'environnement Python pour Linux

```
pip3 install pip --upgrade
pip3 install virtualenv
```

```
cd ../habby_dev
virtualenv --python /usr/bin/python3.6 env_virtuels/env_habby_dev_pip
source env_virtuels/env_habby_dev_pip/bin/activate
pip3 install gdal==2.2.3 --global-option=build_ext --global-option="-
I/usr/include/gdal/"
pip3 install -r habby/requirements.txt
python habby/habby.py
```

2021/03/11 21:55 · qroyer

Structure du projet Python

Préambule

Le projet Python HABBY contient les fichiers suivants :

2021/04/22 09:51 · qroyer

Astuces

Réactivation des 'print'

Dans certaines fonctions lancées en multiprocessing la fonction 'print' est modifiée pour renvoyer les warnings et erreurs au processus principale.

Pour la remettre par défaut cette fonction 'print' lors d'un débogage par exemple, utilisez le code suivant :

```
sys.stdout = sys.__stdout__  # import sys
```

Réactivation du débogage pour les sous-process en QThread

```
RecursionError: maximum recursion depth exceeded while calling a Python
object
```

Si vous apercevez cette erreur, c'est que vous souhaitez débogager dans un sous-process en QThread.

Pour palier à cette erreur il faut renommer temporairement le nom de la méthode “run” de la classe “MyProcessManager” du fichier “src/process_manager_mod.py” par le nom “start”.

2021/04/22 09:14 · qroyer

Utilisation de GIT

- Configuration nom ou ID Github :

```
git config --global user.name "John Doe"
```

- Configuration adresse mail ou adresse mail du compte Github :

```
git config --global user.email johndoe@example.com
```

- Clonage du projet Github en local :

```
git clone https://github.com/YannIrstea/habby
```

- Clonage du projet Github en local avec autoritsation de modification sur Github :

```
git clone https://<TOKEN KEY>@github.com/YannIrstea/habby.git
```

- Unordered List ItemQuelques exemples:

```
git status : connaitre l'état actuel du projet
git checkout dev1 : changer de branche vers la branche 'dev1' du projet
git checkout master : changer de branche pour revenir à la branche master du
projet
git pull : mettre à jour le projet local avec les dernières modifications
disponible sur Github
git gui : ouvrir l'interface graphique de git pour réaliser les 'commits'
git push : mettre à jour le projet sur Github avec les dernières
modifications du projet en local (nécessite une autorisation avec une clef
TOKEN)
```

2021/11/17 11:34 · qroyer

Traduction du logiciel

Prés-requis

- Linguist.exe : (https://github.com/thurask/Qt-Linguist/releases)
- environnement Python HABBY avec PyQt5

Utilisation dans le code

Traduire des champs dans des classes Qt

```
self.tr(‘string to translate’)
```

Traduire des champs en dehors des classes Qt

Si pas de classe ou heritage de classe qui pose problème :

```
from PyQt5.QtCore import QCoreApplication
text = QCoreApplication.translate(‘Input’, ‘string to translate’) # ‘Input’ sera le nom de la ‘fausse’ classe dans QLinguist et ‘Neglect’ le string à traduire.
```

Ou

```
app = QApplication(sys.argv)
languageTranslator = QTranslator(app)
if language == 0:
    input_file_translation = 'Zen_EN'
    languageTranslator.load(input_file_translation,
os.path.join(os.getcwd(), 'translation'))
if language == 1:
    input_file_translation = 'Zen_FR'
    languageTranslator.load(input_file_translation,
os.path.join(os.getcwd(), 'translation'))
elif language == 2:
    input_file_translation = 'Zen_ES'
    languageTranslator.load(input_file_translation,
os.path.join(os.getcwd(), 'translation'))
app.installTranslator(languageTranslator)
app.translate(‘Input’, ‘string to translate’)
```

Mise à jour des fichiers .ts à traduire dans HABBY

- Vérifier que le fichier ‘habby_trans.pro’ contient bien les fichier.py contenant les champs à traduire
- Pour mettre à jour les fichiers, lancer dans l’environnement virtuel et dans le repertoire ‘habby’, lancer la commande :

```
python -m PyQt5.pylupdate_main habby_trans.pro
```

- Ouvrir le fichier de langue souhaité (ex :‘Zen_FR.ts’) dans le répertoire ‘translation’ avec Linguist.exe
- Dans le logiciel Linguist, renseigner les champs ‘French translation’ souhaités ;
- Sauvegarder le fichier ;
- Lancer l’invite de commande dans ‘habby’
- Mettre à jour le fichier de langage choisi en lançant la commande (ici pour le fichier français) :

```
C:\habby_dev\dependence\linguist_5.13.2\lrelease.exe translation/Zen_FR.ts
```

- Relancer HABBY

2021/03/11 21:55 · qroyer

Création d'un executable HABBY

Windows

Pre-requis

- Librairie pyinstaller installée dans l'environnement Python
- Installer Inno Script Studio à partir de ce lien https://www.kymoto.org/products/inno-script-studio/downloads
- Installer Inno Setup à partir de ce lien https://jrsoftware.org/isdl.php

Step-by-step

```
pyinstaller tips/executables/habby.spec --distpath=build/pyinstaller --
workpath=build/pyinstaller/temp
robocopy biology build/pyinstaller/habby/biology /E > nul
robocopy doc build/pyinstaller/habby/doc /E > nul
robocopy model_hydro build/pyinstaller/habby/model_hydro /E > nul
robocopy translation build/pyinstaller/habby/translation /E > nul
robocopy file_dep build/pyinstaller/habby/file_dep /E > nul
:: setup file
start "" /w "C:\Program Files (x86)\Inno Script Studio\isstudio.exe" -
compile tips\executables\setup_from_pyinstaller.iss
```

Linux

```
pyinstaller tips/executables/habby.spec --distpath=build/pyinstaller --
workpath=build/pyinstaller/temp
cp -r biology build/pyinstaller/habby/biology
cp -r doc build/pyinstaller/habby/doc
cp -r model_hydro build/pyinstaller/habby/model_hydro
cp -r translation build/pyinstaller/habby/translation
cp -r file_dep build/pyinstaller/habby/file_dep
:: zip file
zip -r build/pyinstaller/habby.zip build/pyinstaller/habby
```

Mac

Fix Me!

2021/03/30 17:01 · qroyer

Changer et publier la nouvelle version d'HABBY

Numérotation et fonctionnement

La numérotation de la version du logiciel est de type versionnage sémantique X.Y.Z (ex: HABBY v1.1.1)

- X : Changements non rétrocompatibles. Obligatoire de re-création des anciens projets HABBY.
- Y : Ajouts de fonctionnalités rétrocompatibles. Proposition de re-création des anciens projets HABBY.
- Z : Corrections d'anomalies rétrocompatibles.

Étape par étape

Par exemple HABBY v1.1.1 ⇒ HABBY v1.2.0 :

1. fichier 'habby.py', ligne 27, changer la valeur X.Y.Z de la variable 'HABBY_VERSION_STR' ('1.1.1' ⇒ '1.2.0')
2. fichier 'tips\executables\setup_from_pyinstaller.iss', ligne 5, changer la valeur X.Y.Z de la variable 'MyAppVersion' ('1.1.1' ⇒ '1.2.0')
3. Commiter et pusher les modifications de code
4. Compiler les exécutables et installeurs des différents systèmes d'exploitation
5. Uploader les exécutables et changer le numéro de version X.Y.Z sur la page de téléchargement du wiki ('1.1.1' ⇒ '1.2.0')
6. Ajouter un tag de release sur 'https://github.com/YannIrstea/habby/tags' ('1.1.1' ⇒ '1.2.0')

2021/09/29 11:00 · qroyer
2021/03/01 16:44 · qroyer

Participer à la documentation d'HABBY

Technologie DokuWiki

Ce site internet est réalisé avec **DokuWiki** : https://www.dokuwiki.org

La syntaxe DokuWiki est la suivante : https://www.dokuwiki.org/fr:wiki:syntax

Syntaxe Dokuwiki HABBY

Pour se référer aux instructions concernant l'interface d'HABBY, la syntaxe est la suivante :

1. **Menu - Sous-menu** <hi #47B5E6>**Menu - Sous-menu**</hi>
2. **Affichage interface** <hi #9BFFFF>**Affichage interface**</hi>
3. '*Valeur renseignée*' //'Valeur renseignée'//
4. **[Bouton]** <hi #9BFFFF>**[Bouton]**</hi>
5. **[Bouton principal]** <hi #47B5E6>**[Bouton principal]**</hi>
6. Fix Me! En travaux .. Fix Me! FIXME En travaux .. FIXME

Création de page

- Créer un nouveau lien dans la table des matière de la page https://habby.wiki.inrae.fr/fr:start (ou https://habby.wiki.inrae.fr/en:start pour la version anglaise)
- Créer cette nouvelle page en cliquant sur le nouveau lien de la table des matières (surligné en rouge quand la page n'existe pas)
- Si la page créée se trouve dans un répertoire (de plusieurs pages) et pour conserver un ordre d'affichage, ajouter à la fin, avec N le numéro de la page (N), par ex '0' :

```
{{indexmenu_n>N}}
```

- Pour créer une page (représentant un répertoire) qui affiche le contenu de toute les pages du répertoire dans le bon ordre, créer la puis ajouter avec path, le chemin de la page, par ex 'fr:guide_utilisateur' :

```
{{namespace>path&order=indexmenu}}
```

Afficher une page existant sur la page actuelle

```
{{page>fr:manuel_reference:modeles_2d:sub_description:sub_carto&noheader}}
```

2021/03/01 16:50 · qroyer