

Participer au développement Python d'HABBY

Préambule

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

- Python ≥ 3
- Git

Aide à la création de l'environnement Python pour Windows

- Installer les dépendances.
- Télécharger la wheel GDAL : <https://github.com/cgohlke/geospatial-wheels>
 - <https://www.gisinternals.com/release.php>
- Installer Microsoft Visual C++ 14.0 : <https://visualstudio.microsoft.com/fr/visual-cpp-build-tools/> si besoin
- 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 apparaître la fenêtre principale d'HABBY.
- Votre environnement virtuel Python pour HABBY est prêt.

Aide à la création de l'environnement Python pour Linux

- installation de GDAL for Linux and GDAL for Python

(<https://mothergeo-py.readthedocs.io/en/latest/development/how-to/gdal-ubuntu-pkg.html>)

```
sudo apt-get install python3.6-dev
sudo add-apt-repository ppa:ubuntugis/ppa
sudo apt-get update
sudo apt-get install gdal-bin
ogrinfo --version
sudo apt-get install libgdal-dev
```

- Lancer le fichier 'tips\virtual_envs\creation_env_habby_pip.sh'.

```
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==X.X.X --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 déboguer 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 autorisation de modification sur Github :

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

- Unordered List Item Quelques 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 QApplication
text = QApplication.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\dependance\linguist_5.13.2\lrelease.exe translation/Zen_FR.ts
```

- Relancer HABBY

2021/03/11 21:55 · qroyer

Création d'un exécutable 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



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-cr ation des anciens projets HABBY.
- Y : Ajouts de fonctionnalit s r trocompatibles. Proposition de re-cr ation des anciens projets HABBY.
- Z : Corrections d'anomalies r trocompatibles.

 tape par  tape

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 ex cutables et installeurs des diff rents syst mes d'exploitation
5. Uploader les ex cutables et changer le num ro de version X.Y.Z sur la page de t l chargement du wiki ('1.1.1' ⇒ '1.2.0')
6. Ajouter un tag de release sur '<https://github.com/Yannlrstea/habby/tags>' ('1.1.1' ⇒ '1.2.0')

2021/09/29 11:00 · qroyer

From:

<https://habby.wiki.inrae.fr/> - **HABBY**

Permanent link:

<https://habby.wiki.inrae.fr/doku.php?id=fr:develop:collaboration:dev>

Last update: **2021/04/27 14:49**

