# Participer au développement Python d'HABBY

# **Github**

https://github.com/YannIrstea/habby

# **Environnement Python**

Si besoin, vous trouverez ci-dessous, les étapes pour faciliter la création de l'environnement virtuel Python d'HABBY.

#### Windows

## **Dépendances**

- Python >= 3
- GIT
- Microsoft Visual C++ 14.0: https://visualstudio.microsoft.com/fr/visual-cpp-build-tools/
- GDAL : https://www.gisinternals.com/release.php

# Étape par étape

- Avoir Cloné le projet git https://github.com/YannIrstea/habby
- Télécharger les dépendances décrite ci-dessus.
- 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.

## Linux

#### Mac

# **Traduction du logiciel**

## Prés-requis

- Linguist.exe: (https://github.com/thurask/Qt-Linguist/releases)
- environnement virtuel d'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')
```

Attention : ne pas traduire 'Warning :' dans les champs de log car utilisé pour le code couleur.

https://habby.wiki.inrae.fr/ Printed on 2025/10/16 20:41

# 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
- Lancer dans l'environnement virtuel et dans 'habby' et pour mettre à jour les fichiers, 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

From:

https://habby.wiki.inrae.fr/ - HABBY

Permanent link:

https://habby.wiki.inrae.fr/doku.php?id=fr:develop:collaboration:dev&rev=1615485481

Last update: 2021/03/11 18:58

