Documentation programmeur pour AirplaneDesign

# Installation

Via Outils >> Gestionnaire d’Addon installer l’atelier AirplaneDesign ainsi que les dépendances : Plot et CurvedShapes

# Commandes

|  |  |  |  |
| --- | --- | --- | --- |
| Commandes | module | ui | icone |
| Create a Plane | apdPlane.py | apdPlane.ui | apdPlane.xpm |
|  |  |  |  |
| Create a wing | apdWing.py | apdWing.ui | apdWing.xpm |
| Create a nacelle | apdNacelle.py | apdNacelle.ui | apdNacelle.xpm |
|  |  |  |  |
| Create a rib | apdRib.py | apdRib.ui | apdRib.xpm |

# Sous-programme

|  |  |
| --- | --- |
| Commandes | module |
| Wing : charge un example | apdWingUI.py |
| Rib : gérer l’UI | apdRibUI.py |

# Librairie

|  |  |
| --- | --- |
| Commandes | module |
|  |  |
| Création profils aile / nacelle | apdLibShapes.py |

# Structure du dossier AirplaneDesign

Racine : source py des modules

Dossiers :

* Resources : interfaces utilisateur
  + icons : icones
* App :
  + xfoil, usage futur
* Doc : documentation (+ expérimental générée par DOxygen)
* Examples : modélisation d’avion(s)
* Translations : traductions
* Wingribprofil : librairie de profils

# Création d’une nouvelle commande dans l’atelier

Example de l’intégration de la commande nacelle :

## Outils

Outils utilisés pour la création du module nacelle

* Edition de code : Visual Studio Code
* Création d’interface utilisateur : Qt Creator
* Création d’icônes : Gimp

## Intégration de la commande

Dans InitGUI.py importer le module qui gérera la création de la nacelle : import apdNacelle

Ajouter la commande dans la liste : self.comList= [… , apdNacelle]

## Module de création de la nacelle

* Définir la classe Nacelle avec les propriétés de la nacelle (longueur, diamètre ...) avec la fonction execute qui réalise la construction de la nacelle
* Définir la classe NacelleTaskPanel qui va lancer l’interface graphique
* Définir la classe ViewProviderNacelle qui définit les objets graphiques (interface, icones) associés
* Définir la classe CommandeNacelle qui fait l’interface avec FreeCAD

## Ressources pour la nacelle

* créer une interface utilisateur (ui) dans resources avec Qt Creator
* créer une icône (xpm) dans resources/icons

# Traduction des modules

Le langage d’origine est l’anglais pour une plus grande diffusion. Est décrit ici une méthode pour traduire les modules de l’atelier.

Nota : la structure pour l’utilisation de la traduction est implémentée. Pour les modules supplémentaires il suffit

Le processus de traduction se fait en 4 étapes :

1. génération d’un fichier XML \*.ts contenant les textes à traduire
2. concaténation de tous les fichiers \*.ts
3. utilisation d’un traducteur automatique pour générer des fichier \*.ts avec les traduction (un par langue)
4. compilation des fichiers \*.ts en fichier \*.qm utilisables par FreeCAD

## Outils

On aura besoin des exécutables pylupdate (ou pylupdate5), lupdate, lconvert et lrelease. Eventuellement QtLinguist pour corriger des traductions automatiques. Pour Windows je l’ai téléchargé ici : https://fr.freedownloadmanager.org/Windows-PC/Qt-Linguist-GRATUIT.html

et mis les outils dans le sous-dossier WinTools

## Génération \*.ts des ui

Pour chaque interface graphique il faut générer le fichier ts avec l’utilitaire lupdate depuis le sous-dossier translations :

lupdate MonModule.ui -ts MonModule\_ui.ts

## Génération \*.ts des modules python

Pour chaque module python comportant du texte à traduire il faut générer le fichier ts avec l’utilitaire pylupdate5 depuis le sous-dossier translations :

pylupdate5 MonModule.py -ts MonModule\_py.ts

## Concatenation des fichiers \*.ts

Avec l’utilitaire lconvert :

lconvert -i <liste des fichier \*.ts> -o apdWB.ts

Dans Wintools le batch apdTranslate.bat réalise la génération et la concaténation des fichiers \*.ts. Si besoin mettre à jour ce fichier.

## Traduction automatique via le site Crowdin

Ouvrir un compte sur le site Crowdin et créer un projet ainsi que les langues de tranduction.

Dans le volet Content charger le fichier apdWB.ts puis dans le volet Home lancer la pre-translation (via MT) et enfin lancez Buid and download qui déclenche le téléchargement d’un fichier. Décompressez le zip : vous avez autant de dossiers que de langues de traduction, chaque dossier contenant le fichier ts enrichi des traductions.

Il peut être nécessaire de reprendra les traductions manuellement via QtLinguist.

Pour chaque langue il faut publier la traduction dans le format compréhensible par python via la commande lrelease (ou apdRelease.bat) :

WinTools\lrelease fr\apdWB.ts -qm apdWB\_fr.qm

# Roadmap v 0.5

# - [ ] Generate Rib from Wing

# - [ ] Xfoil integration

# - [ ] Increase the width of wizard dialog box so the rotation columns are visible

# - [ ] Add option to Automatically generate a mirror wing for the other side

# - [ ] Wing wizard remembers settings from previous wing

# - [ ] Copy and paste the wing wizard table (eg to and from a spreadsheet)

# - [ ] More "complete plane" example files (eg like the GordLoft\_4 example)

# - [ ] Fuselage design module

# - [ ] Add an "open folder" or "install profile" button in the rib design dialog to make it easier to find/add extras

# - [ ] Wizard de création de l’avion : création de la structure hiérarchique (fuselage, aile) et gestion des paramètres principaux (tableur ou objet DynamicData)

# - [ ] ----------