Documentation

Outil d'importation

1. Introduction

Le rôle de cette outil est de simplifier l'import et la création de librairie pour OpenPLC, ces tache implique de modifier le code source du logiciel et copier des ficher au bonne emplacement.

Ce document décrit le fonctionnement de cette outil. Il est décomposé deux parties, la premier décrit ses fonction et option. La seconde décrit pas-à-pas sont utilisation pour les cas les plus courant.

Ce document ne décrit pas le contenue des fichier de la libraire, cette information peut être trouve dans la *documentation* d'OpenPLC.

2. Description

2.1 Glossaire

<lib> : Nom de la librairie, sans caractères spéciaux (ex : accents), sans espace

2.2 Architecture des librairies

Les librairies sont des dossier composé de tout les informations nécessaire à OpenPLC pour un bloque. Les fichiers nécessaire dans la librairie peuvent varier en fonction du bloque. En voici un exemple nécessitants un code spécifique Arduino :

- XML pour le bloque visuel
- Description ST dans le fichier txt
- Code c pour le compiler matiec
- Code c Arduino
- Module c++ spécifique Arduino

L'architecture exacte est très flexible et est décris dans le ficher « config.json »

2.3 Fichier de configuration

name : str, nom de la librairie (<lib>)

files : dict, informations lier au fichier (emplacement, …)

arduino : dict, information lier au fichier Arduino

path : str, chemin vers le dossier des fichiers pour Arduino (peut être identique celui de matiec), contiens :

- codes des bloques (<lib>.h)
- description des bloques en ST (<lib>.txt)

module : dict, information lier au fichier des module supplémentaire Arduino

path : str, chemin vers le dossier des modules. Si ils ont le même nom que ce d'une autre librairie, ils serons pas importé en double. Ce qui peut êtres utile si deux librairies partage un module, contiens :

- modules Arduino (<module>.c)

modules : list[str], liste des module nécessaire pour la librairie. Ceux dans le dossier des modules qui ne sont pas dans la liste y sont automatiquement rajouté. Ceux qui sont dans la liste mais n'y dans le dossier n'y dans OpenPLC remonterons un message d'erreur. L'auto installation depuis internet n'est pas encore supporte.

matiec : dict, information lier au fichier pour le compiler matiec

path : str, chemin vers le dossier des fichiers pour matiec (peut être identique celui d'Arduino), contiens :

- codes des bloques (<lib>.h)
- description des bloques en ST (<lib>.txt)

xml : dict, information lier au fichier pour l'éditeur

path : str, chemin vers le dossier des fichiers pour l'éditeur, contiens :

- fichier de description des bloques (<lib>.xml)

generated : bool, indique si tout les fichiers nécessaire existe, si *false* l'utilisateur pour décider de les faire générer, basé sur le ficher xml (cf. Section 3.2)

blocks : list[str], list des bloques, automatiquement mis-à-jour par le programme.

Exemple de configuration :

```
{
  "name": "AW9523B",
  "files": {
    "arduino": {
      "path": "./arduino",
      "module": {
        "path": "./arduino/module",
        "modules": ["I2C"]
      }
    },
    "matiec": { "path": "./matiec" },
    "xml": { "path": "./plcopen" }
  },
  "generated": true,
  "blocks": ["DI_1", " DQ_1", " DI_8", " DQ_8"]
}
```

3. Utilisation

3.1 Import d'une librairie

Ces étapes nécessite que la librairie et l'outil sont à porté de main et python3 est installé.

- Ouvrir un terminal (win-r, "cmd")
- Glisser déposer le fichier « import-lib.py » dans le terminal
- Glisser déposer aussi la librairie (zip ou dossier)
- Fermer la fenêtre (ou ctr+c)
- Ouvrir un projet OpenPLC, et vérifier que la librairie apparais à droite

3.2 Création d'une nouvelle librairie

```
Microsoft Windows [version 10.0.22631.4751]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\mathi>"C:\Import Tool INSA\import-lib.py"
Detecting OpenPLC Editor installation...
library path (ctr-c to exit): C:\newLib
The config file is missing, do you want created it (y, n): y
The library has no xml file, now? (y, n): y
The library is not generated, do you want to do it (y, n): y
The library has been generated, do you want to import it (y, n): n
library path (ctr-c to exit):
```

Figure 1 : Prompte de l'outil – Nouvelle librairie

- Ouvrir un terminal (win-r, "cmd")
- Glisser déposer le fichier « import-lib.py » dans le terminal
- Créer un nouveau dossier du nom de la librairie (<lib>)
- Glisser déposer le dossier vierge dans le terminal
- Le système demandera si vous voulez crée un fichier config, dite oui ("y")
- Rajouter le ficher « <lib>.xml » dans le dossier« <lib>/xml/ » (Pour plus d'information cf. ***documentation*** d'OpenPLC.)
- Le système demandera si vous voulez générer la librairie, dite oui ("y")
- Il demandera aussi si vous voulez importer la librairie, dite non ("n")
- Fermer le terminal
- Dans les dossier « arduino » et « matiec » ce trouve un fichier « <lib>.h ». Dans ces fichiers ce trouve le code c des bloques, les commentaire commençant par « /*TF : » sont a compléter. (Pour plus d'information cf. ***documentation*** d'OpenPLC.)

```
// Initialization part
static void RLY_1_init__(RLY_1 *data__, BOOL retain)
{
    __INIT_VAR(data__->EN, __BOOL_LITERAL(TRUE), retain)
    __INIT_VAR(data__->ENO, __BOOL_LITERAL(TRUE), retain)
    __INIT_VAR(data__->B_AD, /*TF: initial value*/, retain)
    __INIT_VAR(data__->P_AD, /*TF: initial value*/, retain)
    __INIT_VAR(data__->RLY, /*TF: initial value*/, retain)
}
```

```
// Actual Code
#define GetFbVar(var, .
#define SetFbVar(var, v
/*TF: Code*/
#undef GetFbVar
#undef SetFbVar
    return;
} // RLY_1_body__()
```

Figure 2 : <lib>.h - à compléter