DOCUMENTATION - Projet Automatisation Microdissecteur

Arborescence du dossier du programme

```
Projet_microdissecteur_serveurDemo
|
└──Dev
|   |   form1.png
|   |   form2.png
|   |   form3.png
|   |   README_micropicell.md
|   |   README_micropicell.pdf
|   |   README_user.md
|   |   README_user.pdf
|   |
|   └──CTRemoteDeployment-5.1.3
|   |   |   CTRemote.ridl
|   |   |   CTRemote.tlb
|   |   |   README.txt
|   |   |
|   |   └──CTRemoteTest
|   |   |   |   *.dll [...]
|   |   |   |   CTRemoteTest.exe
|   |   |   |   Test.jpg
|   |   |   |   [...]
|   |   |   |
|   |   |   └──Projet_vbnet_MACHINE
|   |   |   |   CTRemoteClient.sln
|   |   |   |   Form1.Designer.vb
|   |   |   |   Form1.resx
|   |   |   |   Form1.vb
|   |   |   |   Form2.Designer.vb
|   |   |   |   Form2.resx
|   |   |   |   Form2.vb
|   |   |   |
|   |   |   └──[...]
|   |   |
|   └──example
|           example_ij.csv
|           example_param.txt
|           example_qp.txt
|
|
└──Projet_microdissecteur
    |   README_machine.md
    |   SCRIPT_PRINCIPAL.py
    |
    └──Client
```

```
    |   |   CTRemoteClient.application
    |   |   setup.exe
    |   |
    |   └───Appplication files
    |
    └───Coord
    |
    └───Dependencies
    |   |
    |   └───Fiji.app
    |
    └───Images
    |       test.tiff
    |
    └───Macro
        |   get_ROI_ImageJ.ijm
        |   HE_segmentation.ijm
        |
        └───Classifiers
            |
            └───classifier.model
```

Serveur de démonstration

Fonctionnement

1. Exécuter *CTRemoteTest.exe* en tant qu'administrateur et cliquer sur `register CTRemote`
2. Exécuter une nouvelle instance de *CTRremote.exe* en normal user, et cliquer sur `Start Server As Demo`
3. Exécuter le client à partir du *CTRemoteClient.application* : si le serveur est allumé (étape 2) et que CTRemote a bien été référencé dans le projet VB (voir section suivante), le client se connecte automatiquement à CTRemote.

Informations

La description de l'interface de CTRemote (API) est décrite dans *CTRemote.ridl*. La description de l'interface côté machine est décrite dans *CTRemote.tlb*.

Projet VBNET

Utilisation du projet sur VS2019

- Le code du projet en visual basic est présent dans le dossier Projet_vbnet_MACHINE. Il peut être visualisé et modifié en lançant le fichier *CTRemoteClient.sln* avec Visual Studio 2019.
- La référence à CTRemote à été ajoutée au projet avec *CTRemote.tlb*, présent dans le dossier *CTRemoteDeployment-5.1.3* : `Project` -> `Add Reference` -> `Browse` -> `CTRemote.tlb`

- CTRemote ne supportant que le 32bits, le client doit être compilé en 32bits. Ceci peut être modifié dans : `Project` -> `CTRemoteClient Properties` -> `Compile` -> `Target CPU` -> `x86`
- Pour tester le client en mode debugging, cliquer sur `Start` (**F5**) ou : `Debug` -> `Start Debugging`
- Pour publier le client, choisir le dossier cible dans : `Project` -> `CTRemoteClient Properties` -> `Publish` et cliquer sur `Publish Now`

Coder avec VS2019

- L'interface utilisateur des formulaires peut être visualisée en double cliquant sur le *nom_du_formulaire.vb*.
- Des boutons peuvent être ajoutés à partir de la toolbox à gauche (`View` -> `ToolBoox` ou **Ctrl + Alt + X**).
- En simple cliquant sur un bouton, on peut accéder à ses propriétés à droite (notamment le nom de la variable correspondante, dans `Design` -> `Name`).
- En double cliquant sur un bouton, on accède au code correspondant.

Contenu du code

Le formulaire 1 est le formulaire principal, qui appelle les différentes fonctions de CTRemote. Cliquer sur le bouton `Predict` renvoie vers le deuxième formulaire.

Le deuxième formulaire permet de récupérer les informations d'acquisition (microscope, type de support utilisé) ainsi que les objectifs d'analyse (type de marquage, outils d'analyse) de l'utilisateur. Ces informations sont implémentées dans le fichier des paramètres, dont un exemple est fourni dans *example_param.txt* (voir section **Python-Tests** pour plus d'explications). Cliquer sur le bouton `Start` renvoie vers le script python.

Code Python

Fonction

Le code python permet de lancer un processus d'analyse d'image en fonction des objectifs de l'utilisateur, récupérés à partir du deuxième formulaire VBNET. Il va lancer un script QuPath / ImageJ / Ilastik qui analysera l'image et enregistrera les coordonées des ROI d'intérêt.

Tests

Pour tester le script python en dehors du programme, il faut le lancer avec en paramètres le chemin de l'image à analyser, le chemin de la lame entière (elle n'a pas besoin d'exister, pour le moment le script n'en fait rien) et le fichier de paramètres dont un exemple est fourni dans *example_param.txt* :

```
C:/Users/Marianne/AppData/Local/Programs/Python/Python38/python.exe
C:/Users/Marianne/Projet_microdissecteur/SCRIPT_PRINCIPAL.py
C:/Users/Marianne/Projet_microdissecteur/Images/acquired_image.tiff image
C:/Users/Marianne/Projet_microdissecteur/exemple_param.txt
```

Pour le moment, la seule ligne importante dans ce fichier de paramètres est la ligne 5 : elle indique quel logiciel et donc quelle macro va être executée. Les lignes dans ce fichier, dans l'ordre, indiquent :

1. Le microscope d'acquisition (microdissecteur/scanner/confocal/...) ;
2. Le type de support (slide/box/...) ;
3. La position des supports (support1:position;support2:position;support3:position) ;
4. Le type de marquage (fluo/H&E/...) ;
5. Le logiciel d'analyse d'image à utiliser ;
6. Les positions X et Y de l'image acquise (X;Y). La ligne est vide si aucune image n'a été acquise au microdissecteur avec le programme ;
7. La liste des channels de fluorescence (channel1,channel2). La ligne est remplie seulement si le marquage (ligne 4) est de type fluorescence ;
8. La liste des channels de fluorescence qui doivent être présents sur les cellules d'intérêt (channel1,channel2). La ligne est remplie seulement si le marquage (ligne 4) est de type fluorescence.

Logiciels d'analyse d'image

Le formulaire implémenté en VBNET ne permet de lire qu'un seul type de fichiers .csv (un fichier d'exemple *example_ij.csv* est présent dans le dossier **Dev\example**).

ImageJ

Pour générer ce type de fichier avec une macro ImageJ, on peut utiliser ce code :

```
//Function to get results of each ROI selection
function get_coordinates(name){
        nr = nResults;
        Roi.getCoordinates(x, y);
        for (i=0; i<x.length; i++){
                setResult("Coord", i+nr, name);
                setResult("X", i+nr, x[i]);
                setResult("Y", i+nr, y[i]);
        }
}

//Get & save results in .csv file named after the image
n = roiManager("count");
for (i=0; i<n; i++){ //For each ROI, list each coordinates with get_coordinates
        name = i+1;
        roiManager("Select", i);
        get_coordinates(name);
}
fileName = getDirectory("home")+"Projet_microdissecteur\\Coord\\"+s[0]+"_ROI.csv";
saveAs("Results", fileName);
}
```

Le nom du fichier en sortie est important pour pouvoir être lu ensute. Le script python s'occupe ensuite de transformer les ROI en fonction du recalage nécessaire par rapport au microdissecteur.

La macro doit ensuite être ajoutée dans le dossier *Projet_microdissecteur/Coord*, et cette macro doit ensuite être executée par le script Python.

QuPath

Pour générer ce type de fichier avec un script Qupath, il faut utiliser ce code :

```
//Create file
def path = buildFilePath("C:\\Users\\Marianne\\Projet_Microdissecteur\\Coord",
'results_ROI.txt');
def file = new File(path);
file.text = "";

//Add ROI to file
for (annotation in getAnnotationObjects()){
        if (annotation.getPathClass() == getPathClass("Positive")){
                def roi = annotation.getROI();
                file << roi.getAllPoints() << System.lineSeparator();
        }else{
                continue;
        }
}
```

Ce code génère un fichier présent dans le dossier **Dev\example** (*example_qp.csv*). La fonction `getPathClass` doit être lancée sur la classe dont les ROI doivent être récupérées (ici, la classe "Positive"). Le script python se charge ensuite de créer le bon type de fichier .csv et de transformer le ROI en fonction du recalage nécessaire.

Les chemins d'accès pour la partie QuPath (dans le script Python) doivent être changés en fonction de la localisation de QuPath, des dossiers de projet et des dossiers de scripts.

Ilastik

Pour le moment, le script Ilastik n'est pas fonctionnel, et les chemins doivent être changés (dans le script Python) en fonction de la localisation du classifier et de Ilastik.