Lycée Paul ELUARD

Maintenant que vous maitrisez l'utilisation de Qt Creator, les Widgets, les connecteurs, etc., nous allons nous intéresser à la partir graphique Qt Designer de Qt creator. Cet outil de saisie graphique va vous permettre :

- De placer graphiquement des Layouts.
- De placer graphiquement des Widgets.
- De configurer les connecteurs les signaux et les slots.

Dans chaque partie, il vous sera demandé de réaliser une certaine tâche professionnelle. A chaque fois, en conclusion de votre travail, vous devrez :

- Montrer au professeur que le **cahier des charges** a bien été rempli et répondre à ses questions ;
- Répondre aux questions du **document-réponse**.

Seront pris en compte dans l'évaluation de votre travail :

- La bonne réalisation des **installations** ;
- Le **soin** accordé au matériel ;
- Le bon **rangement** du matériel en fin de TP ;
- La **clarté** des explications données au professeur ;
- La **qualité rédactionnelle** du document-réponse.

En fin du TP, vous veillerez à :

- **Ranger** soigneusement le matériel ;
- Modifier le nom du document-réponse, en remplaçant les « **YYY** » par vos noms ;
- **L'envoyer** par mail ou le glisser dans la BAL de votre professeur.

Matériel disponible :

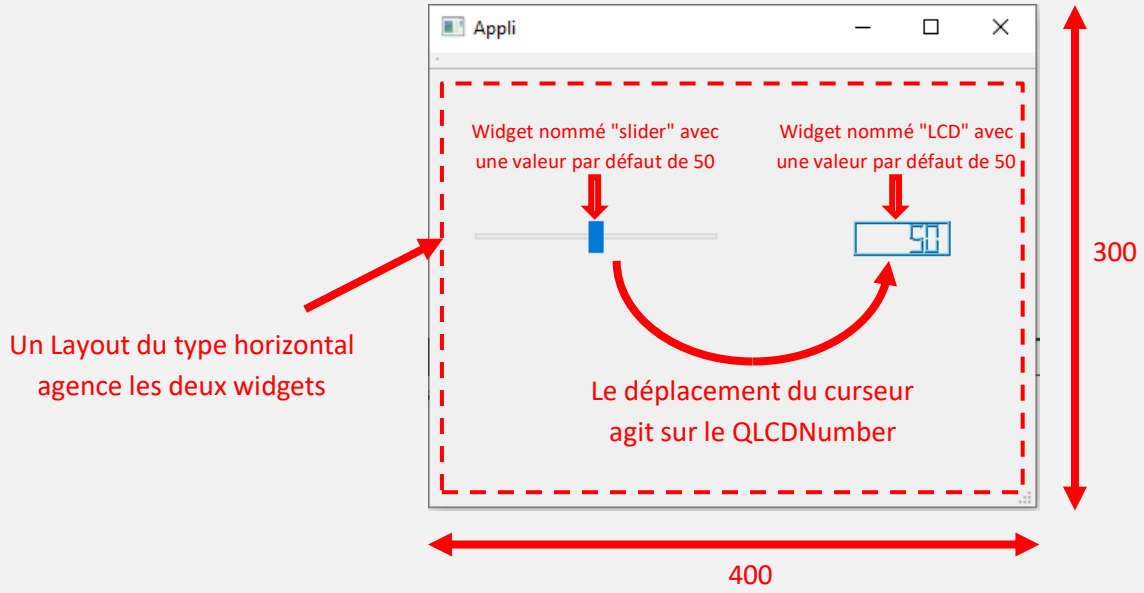
- Ordinateur équipé de *Windows 10*, et des logiciels *Qt Creator* ;
- Baies informatiques câblées et équipées de switchs ;
- Câbles RJ45.

Ressources disponibles :

- Cours sur langage C++;
- Cours sur le FramWork Qt.

1. LIAISON INTERFACE CODE CPP

En vous aidant de l'annexe N°1 créez un projet de type "Application Qt avec widgets" **avec interface utilisateur**, puis sous designer créez l'interface suivante :



Pour ce faire vous devez suivre la démarche suivante :

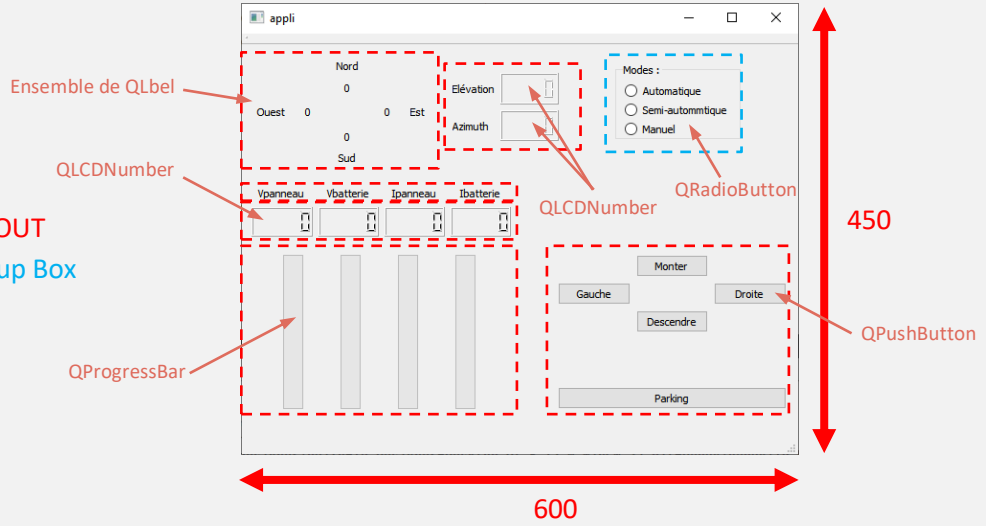
Etape N°1 : Utiliser l'onglet Signaux et slot au centre de designer pour créer un connecteur entre le slide et le LCD.

Etape N°2 : Supprimez le précédent connecteur puis repassez en mode édition et modifiez le fichier Appli.h de façon à ajouter à votre classe un slot nommé changer_LCD et un signal nommé Envoyer2LCD. Chacune de ces méthodes traite un entier. Dans le fichier Appli.cpp ajoutez la description de ces deux méthodes. Ajoutez également une méthode nommée Afficher qui reçoie un entier et change la valeur du LCD en conséquence. Attention, pour accéder à un pointeur type widget créé dans Designer il faut utiliser la syntaxe ui->Nom_du_widget (par exemple ui->slider->setValue(15)). Ajoutez au sein du constructeur de votre classe appli les deux connecteurs : slider vers Afficher() puis Envoyer2LCD vers Changer_LCD().

- Complétez le compte-rendu de mesures.
- Expliquez le rôle et le fonctionnement de la méthode connect() de l'étape N°2.

2. INTERFACE PANNEAU SOLAIRE

Créez un nouveau projet avec une interface utilisateur de façon à obtenir l'application suivante :

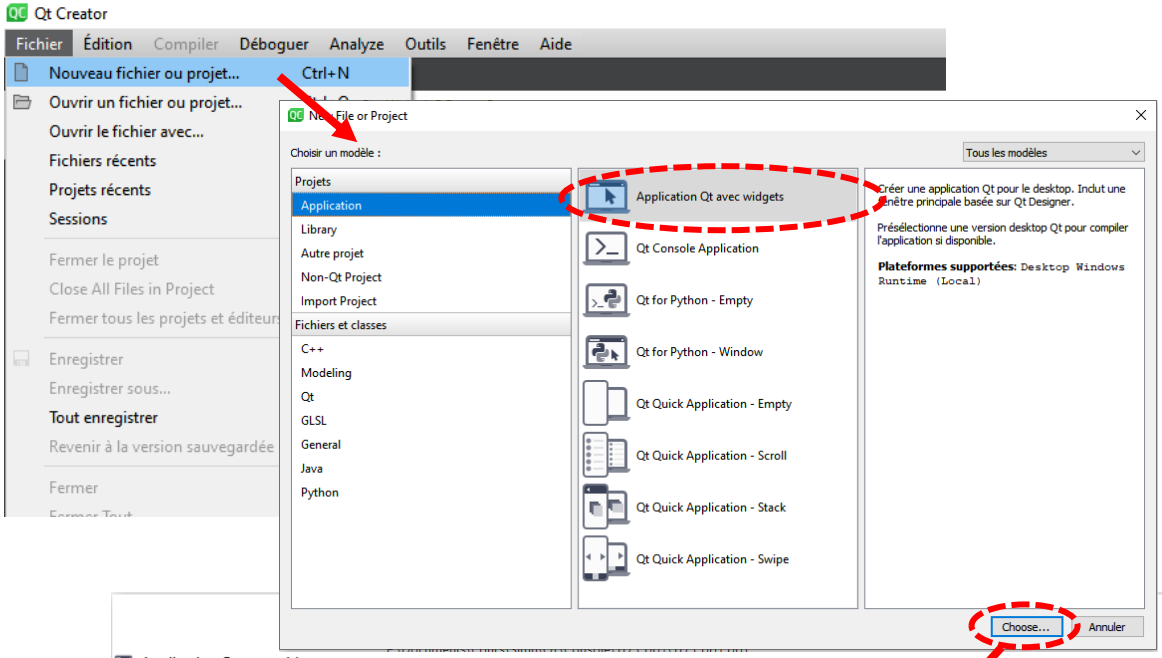


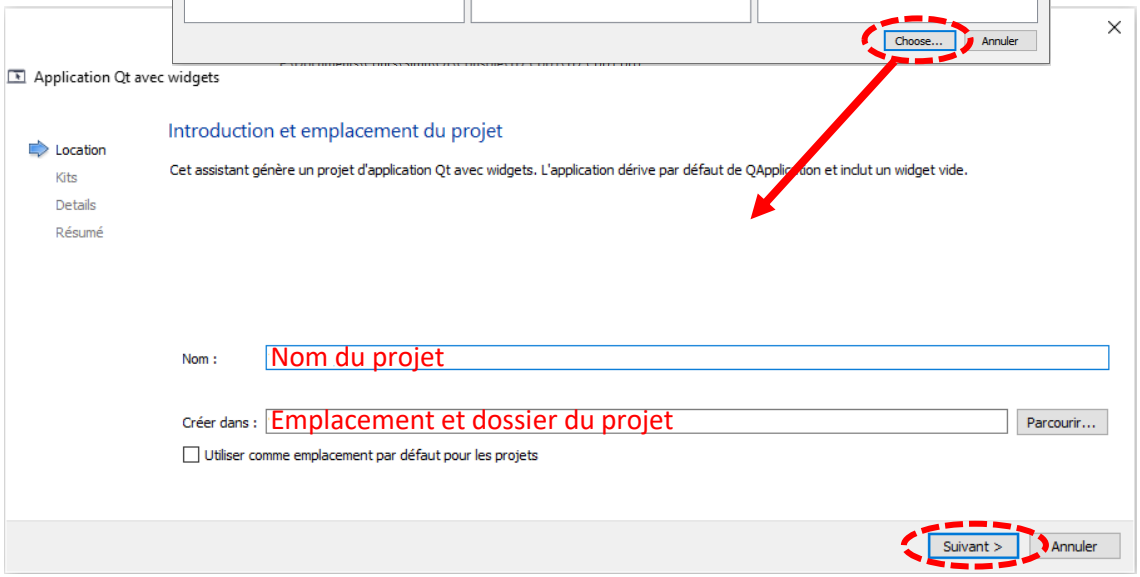
- Nommez bien tous les widgets de façon cohérentes de façon à vous en souvenir.
- Utilisez le lien suivant https://doc.qt.io/qt-5/qtserialport-index.html pour ajouter à votre projet la librairie QSerialPort à votre projet.
- A l'aide du cours (dans le lecteur réseau Classes) Utilisez les méthodes setBaudRate, setDataBits, setParity, setFlowControl et setStopBits pour configurer la liaison série COM1 à 9600 Bauds, sans parité, 1 bit de stop, 8 bits et sans contrôle de flux. Utilisez maintenant la méthode open en mode ReadWrite. Cette méthode revoie un bool pour indiquer si l'ouverture à bien eu lieu. Testez cette variable pour indiquer dans la console (avec la méthode qDebug("texte…")) un message d'erreur ou de bonne ouverture de COM1.
- Utilisez l'annexe N°3 pour mettre en œuvre la communication entre le panneau solaire et l'interface que vous venez de réaliser.

- Complétez le compte-rendu de mesures.
- Ajoutez quelques chronogrammes et copie d'écran montrant le bon fonctionnement de la liaison série.

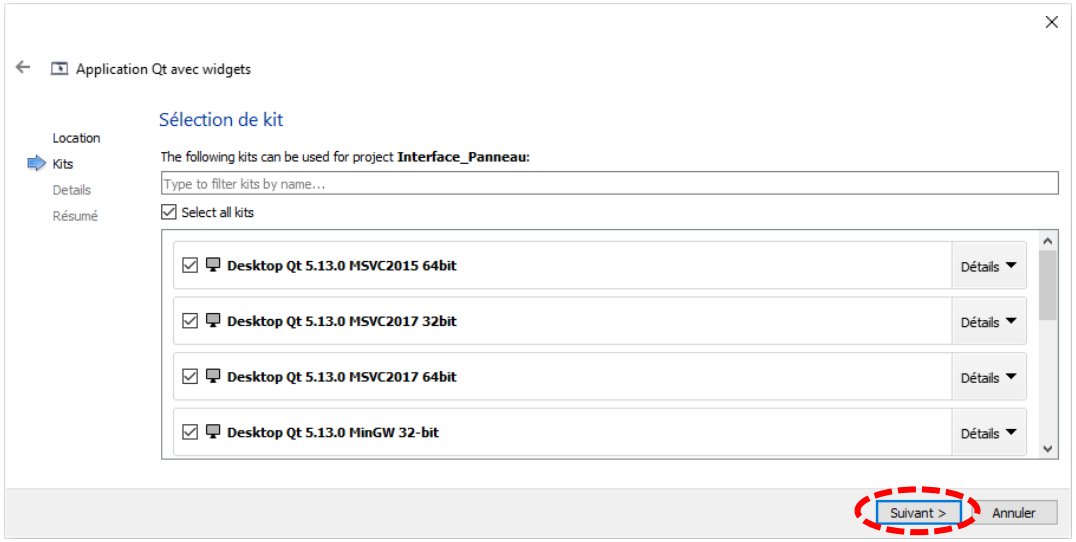
3. ANNEXE 1 : Création d'un projet avec interface utilisateur

Lancez l'application Qt Creator et cliquez sur Fichier -> Nouveau fichier ou projet…

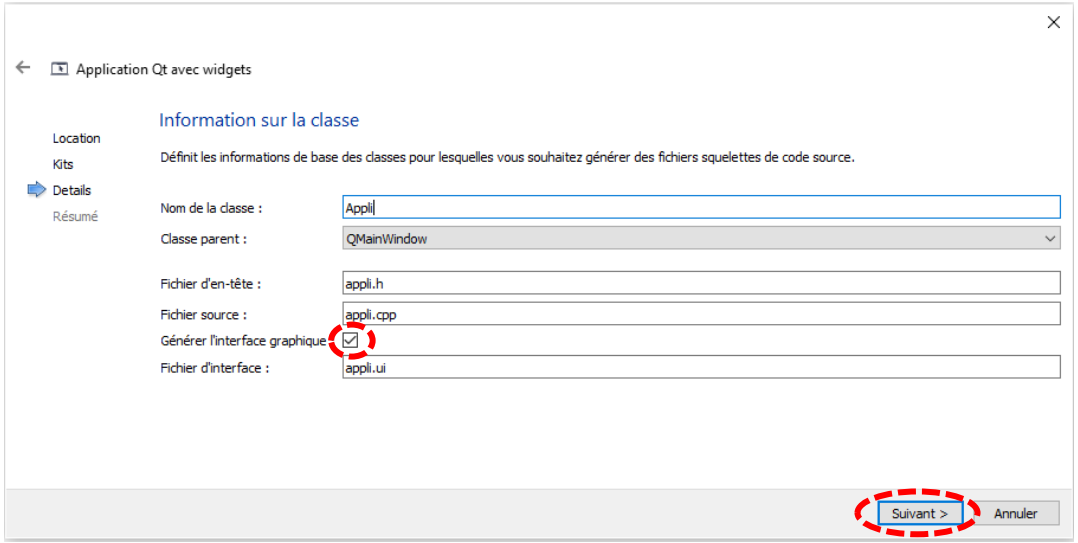


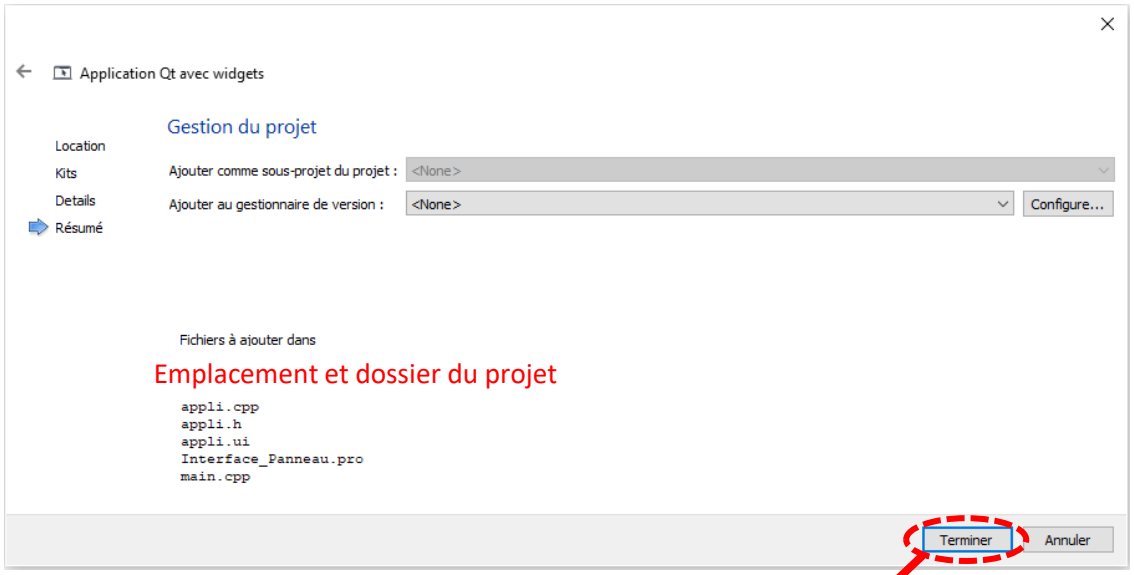


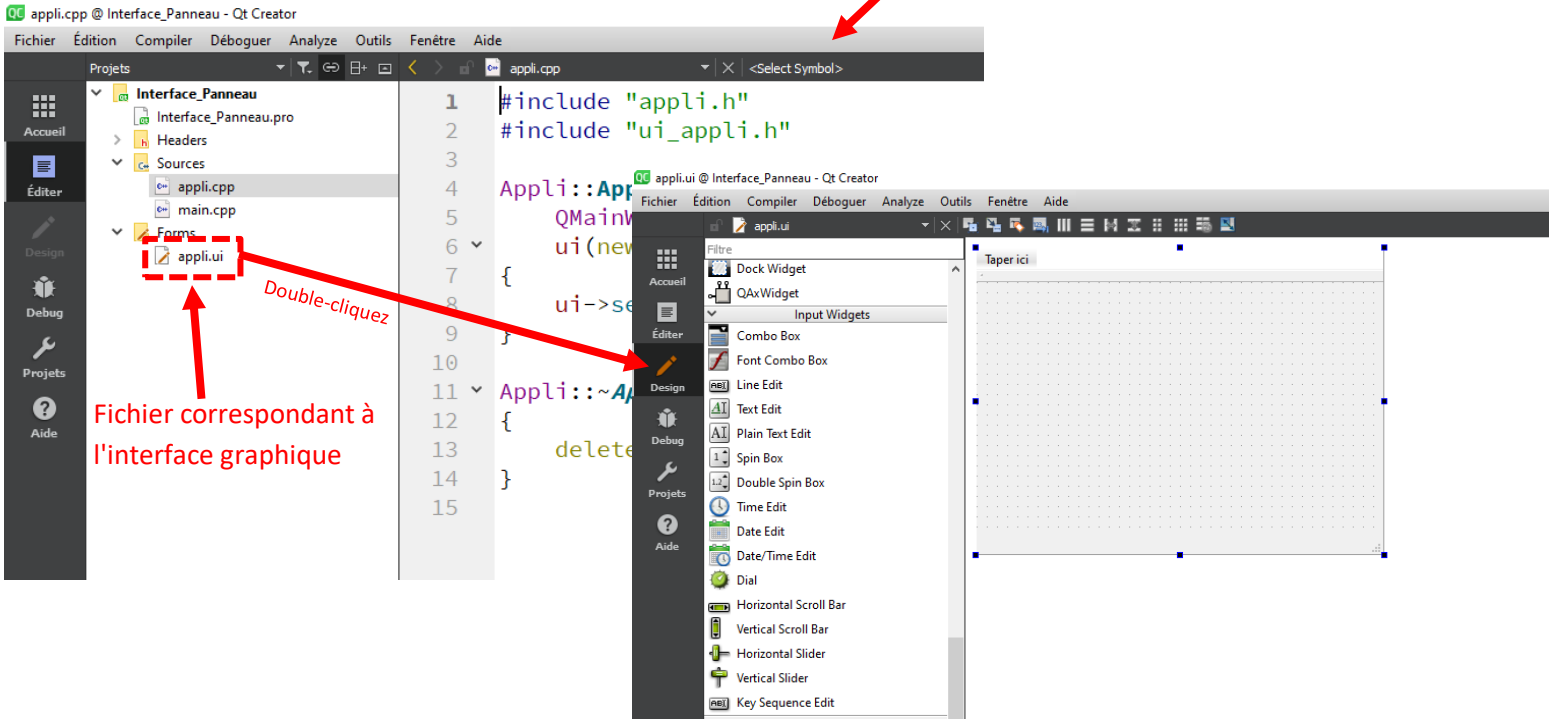
Sélectionnez le compilateur que vous souhaitez utiliser pour ce projet puis cliquez sur Suivant :



Donnez un nom à la classe qui va être créée pour vous par Qt creator (Appli dans notre exemple) et sélectionnez Générer une interface graphique (c'est là que vous allez utiliser Qt Designer) :

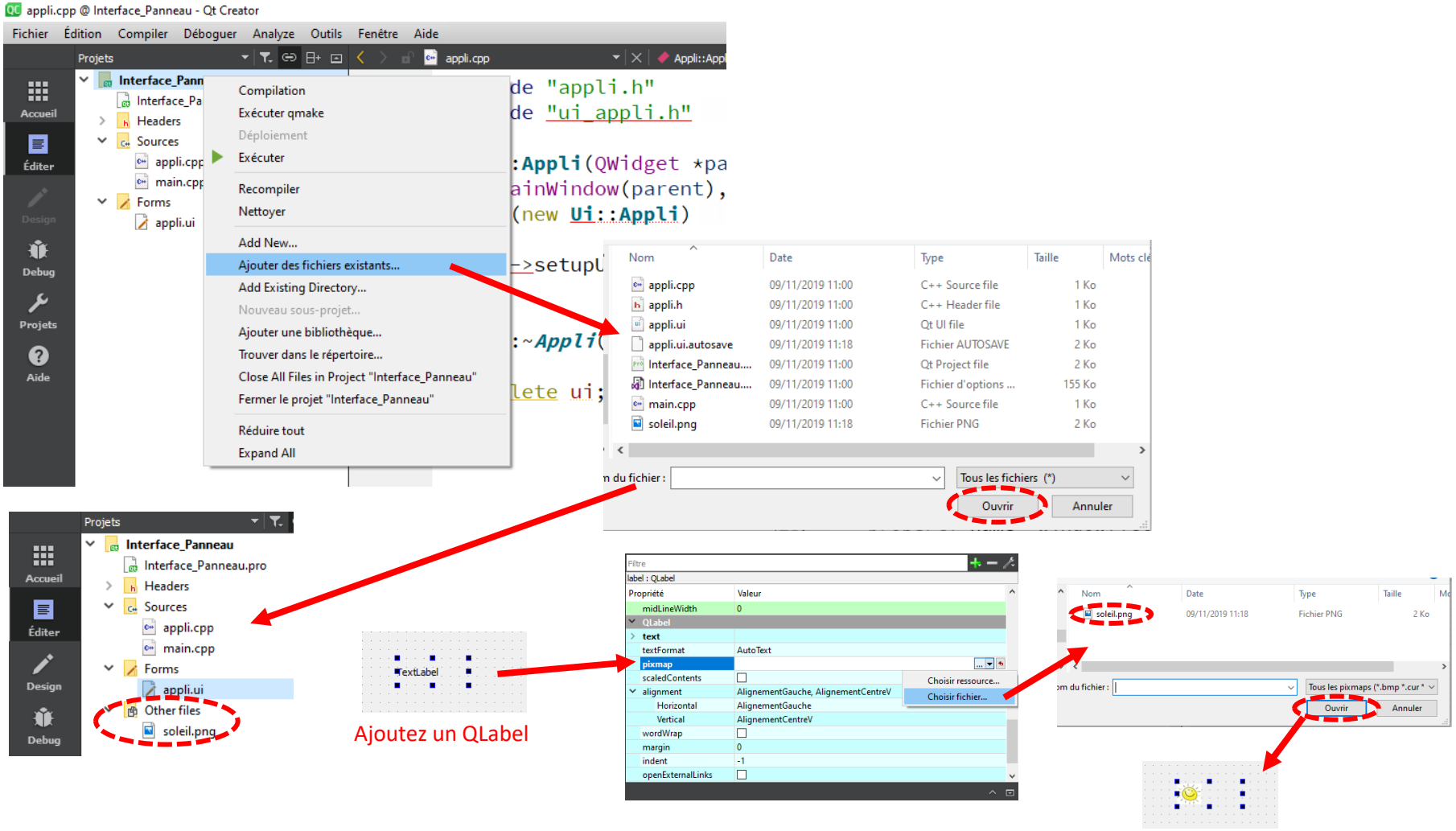






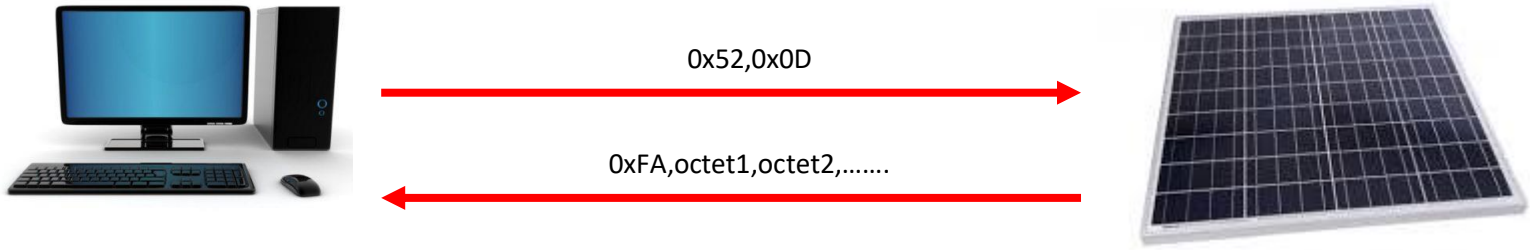
3. ANNEXE 2 : Ajout d'un fichier ressource image dans notre projet et l'insérer dans un QLabel

Sélectionner la racine du projet et cliquez droit puis cliquez sur "Ajouter des fichiers existants…" :



3. ANNEXE 3 : Liaison série entre l'interface et le panneau solaire

Dans la communication qui a lieu entre l'interface et le panneau solaire c'est l'interface qui débute les échanges. Elle débute la communication en envoyant les deux caractères 0x52 et 0x0D (soit les caractères 'R' et '\n'). De son côté le panneau solaire répond en envoyant une trame débutant par le caractère 0xFA et comportant toutes les informations :



En éditant le programme de gestion de la liaison série du panneau solaire on obtient toutes les informations sur les format de la trame envoyée par le panneau :

```
fprintf(port1,"%X",0xfa);
fprintf(port1,"%3U",lum_est);          //data luminosite
fprintf(port1,"%3U",lum_ouest);
fprintf(port1,"%3U",lum_nord);
fprintf(port1,"%3U",lum_sud);
fprintf(port1,"%3U",lum_moy);
fprintf(port1,"%3U",eca);
fprintf(port1,"%3U",ece);
fprintf(port1,"%X",0);                 //reserve
fprintf(port1,"%3U",ten_p);            //data energie
fprintf(port1,"%3U",ten_b);
fprintf(port1,"%3U",Ip);
fprintf(port1,"%3U",Ib);
fprintf(port1,"%X",charge);
fprintf(port1,"%X",full);
fprintf(port1,"%X",empty);
fprintf(port1,"%3U",ec_com);           //data eclairage
fprintf(port1,"%3U",ec_int);
mo_ie=mo_ie-10;
mo_ia=mo_ia-12;
fprintf(port1,"%3U",mo_ie);            //data moteur
fprintf(port1,"%3U",mo_ia);
fprintf(port1,"%X",mo_sure);
fprintf(port1,"%X",mo_sura);
fprintf(port1,"%X",mo_on);
fprintf(port1,"%3LU",ang_azim);
fprintf(port1,"%3U",ang_elev);
fprintf(port1,"%X",butee);
fprintf(port1,"%X",cor_mode);          //data correction
fprintf(port1,"%3U",cor_per);
fprintf(port1,"%3U",cor_seuil);
fprintf(port1,"%X",13);
```