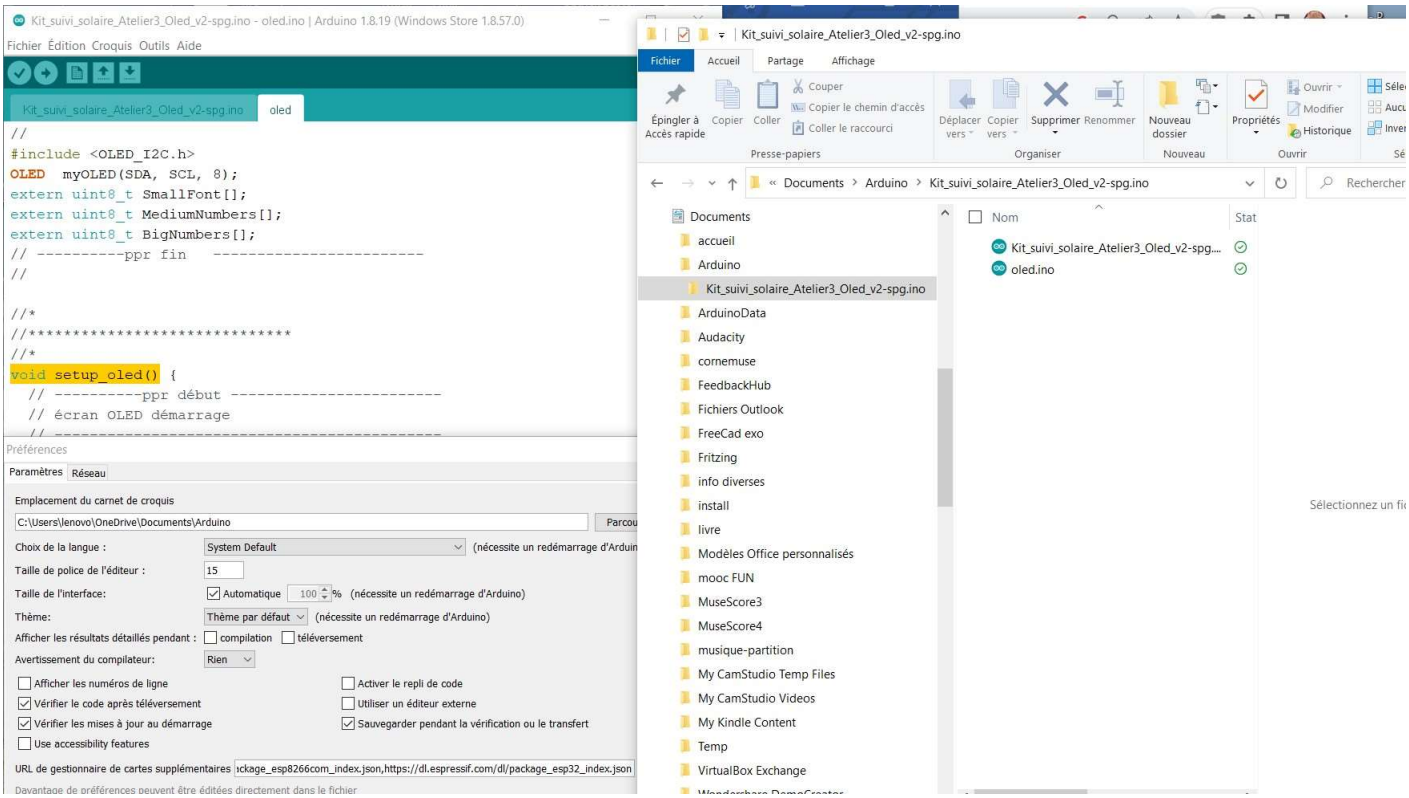
Utilisation de l'IDE Arduino : onglet/sous-programme

On améliore la lisibilité du code en découpant le programme Arduino en plusieurs onglets.

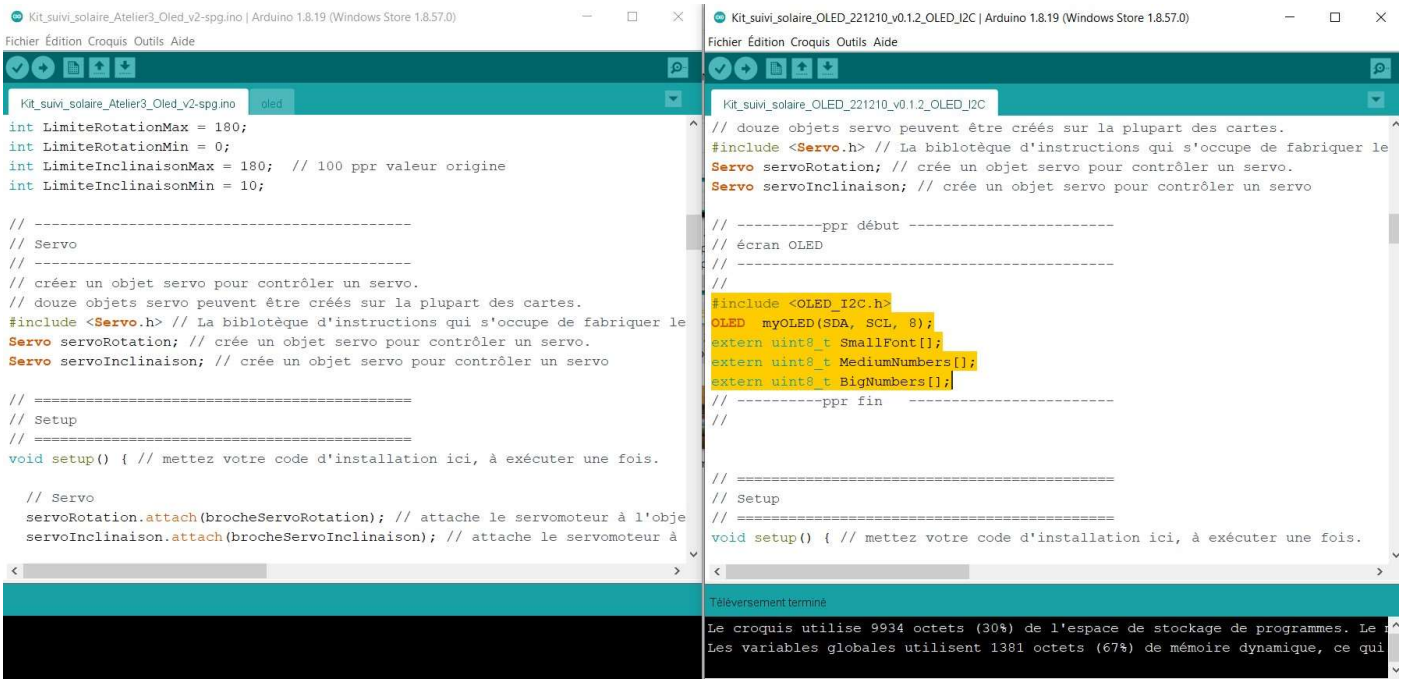
La méthode la plus simple et d'abord, d'écrire un sous-programme/sous-routine, en langage Arduino une fonctions pour structurer le programme. Ensuite, on place dans un onglet la fonction (sous-programme/sous-routine) : le nom de l'onglet sera sauvegardé en fichier .ino placé dans le même répertoire que le programme principal.

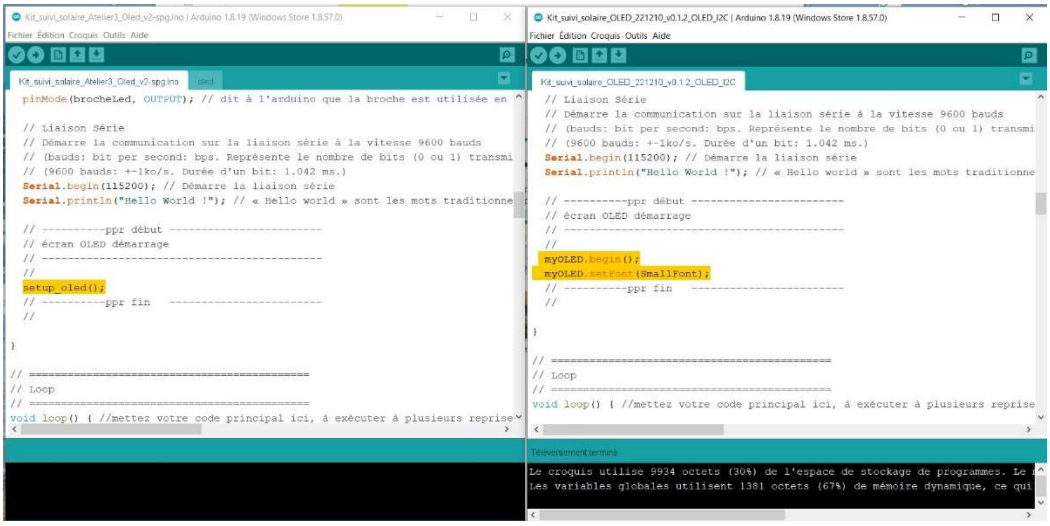
L'environnement de développement intégré/**I**ntegrated **D**éveloppement **E**nvironnement Arduino va modifier l'ensemble en un seul fichier type cpp qui sera envoyé dans le compilateur puis le linker (enfin vers la carte Arduino pour exécution).

Environnement :



Comparaison des 2 approches :

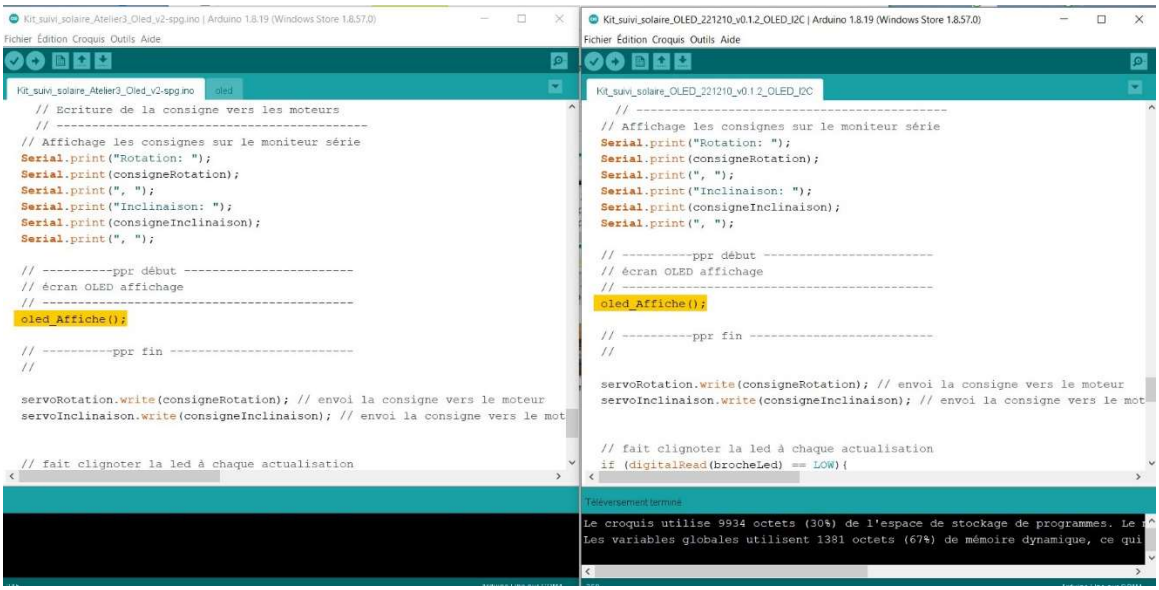




```
pinMode(brocheLed, OUTPUT); // dit à l'arduino que la broche est utilisée en
// Liaison Série
// Démarre la communication sur la liaison série à la vitesse 9600 bauds
// (bauds: bit per second: bps. Représente le nombre de bits (0 ou 1) transmi
// (9600 bauds: +-1ko/s. Durée d'un bit: 1.042 ms.)
Serial.begin(115200); // Démarre la liaison série
Serial.println("Hello World !"); // « Hello world » sont les mots traditionne
// ----------ppr début ------------------------
// écran OLED démarrage
// ---------------------------------------------
//
setup_oled();
// ----------ppr fin   ------------------------
//
}
// ============================================
// Loop
// ============================================
void loop() { //mettez votre code principal ici, à exécuter à plusieurs reprise
```

```
// Liaison Série
// Démarre la communication sur la liaison série à la vitesse 9600 bauds
// (bauds: bit per second: bps. Représente le nombre de bits (0 ou 1) transmi
// (9600 bauds: +-1ko/s. Durée d'un bit: 1.042 ms.)
Serial.begin(115200); // Démarre la liaison série
Serial.println("Hello World !"); // « Hello world » sont les mots traditionne
// ----------ppr début ------------------------
// écran OLED démarrage
// ---------------------------------------------
//
myOLED.begin();
myOLED.setFont(SmallFont);
// ----------ppr fin   ------------------------
//
}
// ============================================
// Loop
// ============================================
void loop() { //mettez votre code principal ici, à exécuter à plusieurs reprise
```

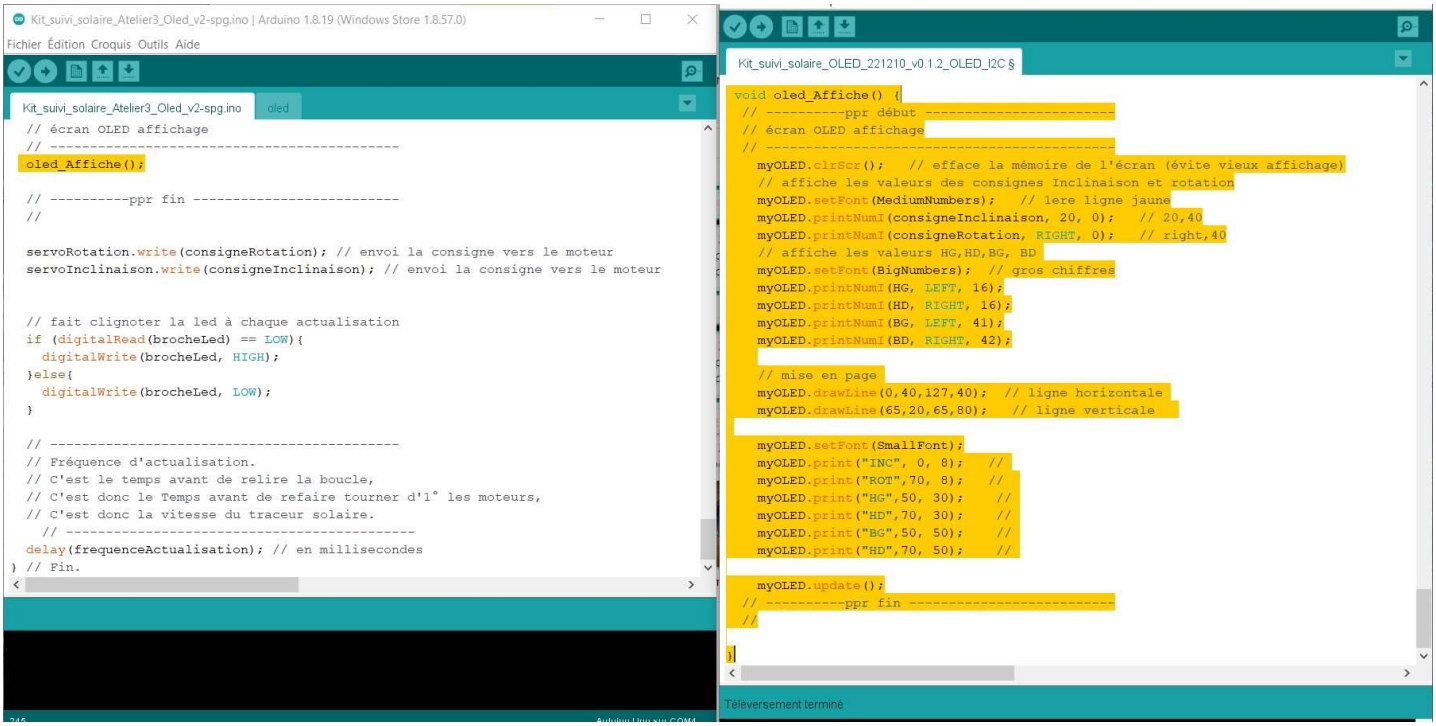
```
Le croquis utilise 9934 octets (30%) de l'espace de stockage de programmes. Le
Les variables globales utilisent 1381 octets (67%) de mémoire dynamique, ce qui
```



```
// Ecriture de la consigne vers les moteurs
// ---------------------------------------------
// Affichage les consignes sur le moniteur série
Serial.print("Rotation: ");
Serial.print(consigneRotation);
Serial.print(", ");
Serial.print("Inclinaison: ");
Serial.print(consigneInclinaison);
Serial.print(", ");

// ----------ppr début ------------------------
// écran OLED affichage
// ---------------------------------------------
oled_Affiche();

// ----------ppr fin --------------------------
//

servoRotation.write(consigneRotation); // envoi la consigne vers le moteur
servoInclinaison.write(consigneInclinaison); // envoi la consigne vers le mot

// fait clignoter la led à chaque actualisation
```

```
// ---------------------------------------------
// Affichage les consignes sur le moniteur série
Serial.print("Rotation: ");
Serial.print(consigneRotation);
Serial.print(", ");
Serial.print("Inclinaison: ");
Serial.print(consigneInclinaison);
Serial.print(", ");

// ----------ppr début ------------------------
// écran OLED affichage
// ---------------------------------------------
oled_Affiche();

// ----------ppr fin --------------------------
//

servoRotation.write(consigneRotation); // envoi la consigne vers le moteur
servoInclinaison.write(consigneInclinaison); // envoi la consigne vers le mot

// fait clignoter la led à chaque actualisation
if (digitalRead(brocheLed) == LOW){
```

```
Le croquis utilise 9934 octets (30%) de l'espace de stockage de programmes. Le
Les variables globales utilisent 1381 octets (67%) de mémoire dynamique, ce qui
```



```
// écran OLED affichage
// ---------------------------------------------
oled_Affiche();

// ----------ppr fin --------------------------
//

servoRotation.write(consigneRotation); // envoi la consigne vers le moteur
servoInclinaison.write(consigneInclinaison); // envoi la consigne vers le moteur


// fait clignoter la led à chaque actualisation
if (digitalRead(brocheLed) == LOW){
  digitalWrite(brocheLed, HIGH);
}else{
  digitalWrite(brocheLed, LOW);
}

// ---------------------------------------------
// Fréquence d'actualisation.
// C'est le temps avant de relire la boucle,
// C'est donc le Temps avant de refaire tourner d'1° les moteurs,
// C'est donc la vitesse du traceur solaire.
  // ---------------------------------------------
delay(frequenceActualisation); // en millisecondes
} // Fin.
```

```
void oled_Affiche() {
 // ----------ppr début ------------------------
 // écran OLED affichage
 // ---------------------------------------------
   myOLED.clrScr();   // efface la mémoire de l'écran (évite vieux affichage)
   // affiche les valeurs des consignes Inclinaison et rotation
   myOLED.setFont(MediumNumbers);   // 1ere ligne jaune
   myOLED.printNumI(consigneInclinaison, 20, 0);   // 20,40
   myOLED.printNumI(consigneRotation, RIGHT, 0);   // right,40
   // affiche les valeurs HG,HD,BG, BD
   myOLED.setFont(BigNumbers);  // gros chiffres
   myOLED.printNumI(HG, LEFT, 16);
   myOLED.printNumI(HD, RIGHT, 16);
   myOLED.printNumI(BG, LEFT, 41);
   myOLED.printNumI(BD, RIGHT, 42);

   // mise en page
   myOLED.drawLine(0,40,127,40);  // ligne horizontale
   myOLED.drawLine(65,20,65,80);   // ligne verticale

   myOLED.setFont(SmallFont);
   myOLED.print("INC", 0, 8);   //
   myOLED.print("ROT",70, 8);   //
   myOLED.print("HG",50, 30);    //
   myOLED.print("HD",70, 30);    //
   myOLED.print("BG",50, 50);    //
   myOLED.print("HD",70, 50);    //

   myOLED.update();
 // ----------ppr fin --------------------------
 //

}
```

```
Téléversement terminé
```

Code de l’onglet oled (devenu fichier oled.ino par l’IDE Arduino)

```
// ----------ppr début ----------------------
// écran OLED
// ----------------------------------------
//
#include <OLED_I2C.h>
OLED  myOLED(SDA, SCL, 8);
extern uint8_t SmallFont[];
extern uint8_t MediumNumbers[];
extern uint8_t BigNumbers[];
// ----------ppr fin   ----------------------
//

//*
//*****************************
//*
void setup_oled() {
  // ----------ppr début ----------------------
  // écran OLED démarrage
  // ----------------------------------------
  //
  myOLED.begin();
  myOLED.setFont(SmallFont);
  // ----------ppr fin   ----------------------
  //
}

 //*
 //************************
 //*

 void oled_Affiche() {
  // ----------ppr début ----------------------
  // écran OLED affichage
  // ----------------------------------------
    myOLED.clrScr();   // efface la mémoire de l'écran (évite vieux affichage)
```

```
  // affiche les valeurs des consignes Inclinaison et rotation
  myOLED.setFont(MediumNumbers);   // 1ere ligne jaune
  myOLED.printNumI(consigneInclinaison, 20, 0);   // 20,40
  myOLED.printNumI(consigneRotation, RIGHT, 0);   // right,40

  // affiche les valeurs HG,HD,BG, BD
  myOLED.setFont(BigNumbers);  // gros chiffres
  myOLED.printNumI(HG, LEFT, 16);
  myOLED.printNumI(HD, RIGHT, 16);
  myOLED.printNumI(BG, LEFT, 41);
  myOLED.printNumI(BD, RIGHT, 42);

  // mise en page
  myOLED.drawLine(0,40,127,40);  // ligne horizontale
  myOLED.drawLine(65,20,65,80);   // ligne verticale

  myOLED.setFont(SmallFont);
  myOLED.print("INC", 0, 8);   //
  myOLED.print("ROT",70, 8);   //

  myOLED.print("HG",50, 30);    //
  myOLED.print("HD",70, 30);    //
  myOLED.print("BG",50, 50);    //
  myOLED.print("HD",70, 50);    //

  myOLED.update();
 // ----------ppr fin -------------------------
 //

}
```