釧路工業高等専門学校　プログラミングサポートチーム

Ultimate プログラミング講義資料

1: Arduino IDE

プログラムは Arduino IDE というメモ帳のようなソフトウェアを使って書きます.

Arduino IDE の全体像は以下の画像の通りです.

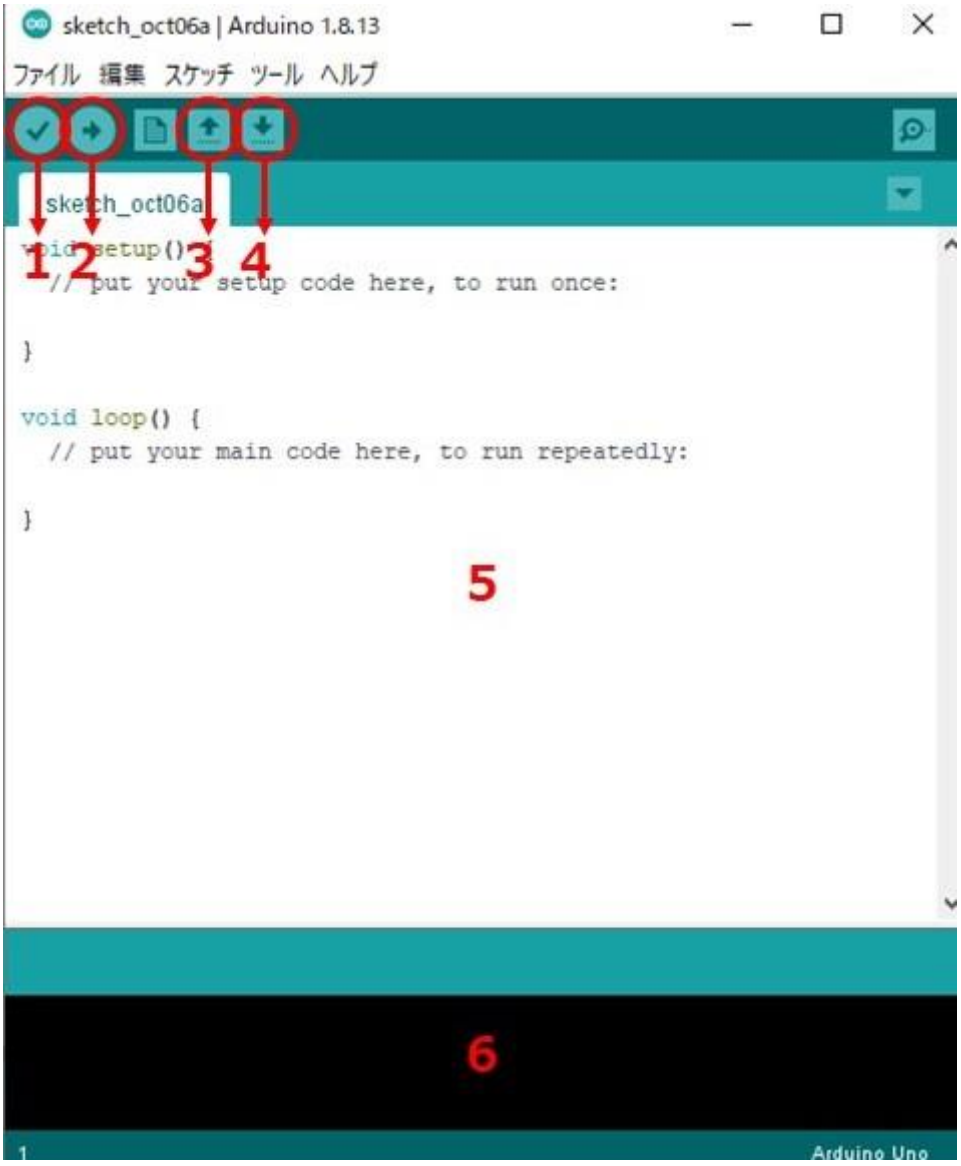

図 1.Arduino IDE の全体像

1. 検証ボタン　　：書いたプログラムが文法的に正しいかどうか検証します.

2. 書き込みボタン：Ultimate にプログラムを書き込みます.

3. 開くボタン　　：パソコンに保存してあるプログラムを開きます.

4. 保存ボタン　　：書いたプログラムをパソコンに保存します.

5. エディタ　　：ここにプログラムを書きます.

6. コンソール　　：プログラムの検証時等にエラーなどが表示されます.

2 : Arduino IDE の設定

プログラムを書いて Altimate に書き込むには Arduino IDE の設定をしなければなりません.

ツール→ポート を Arduino Mega or Mega 2560 に設定する.

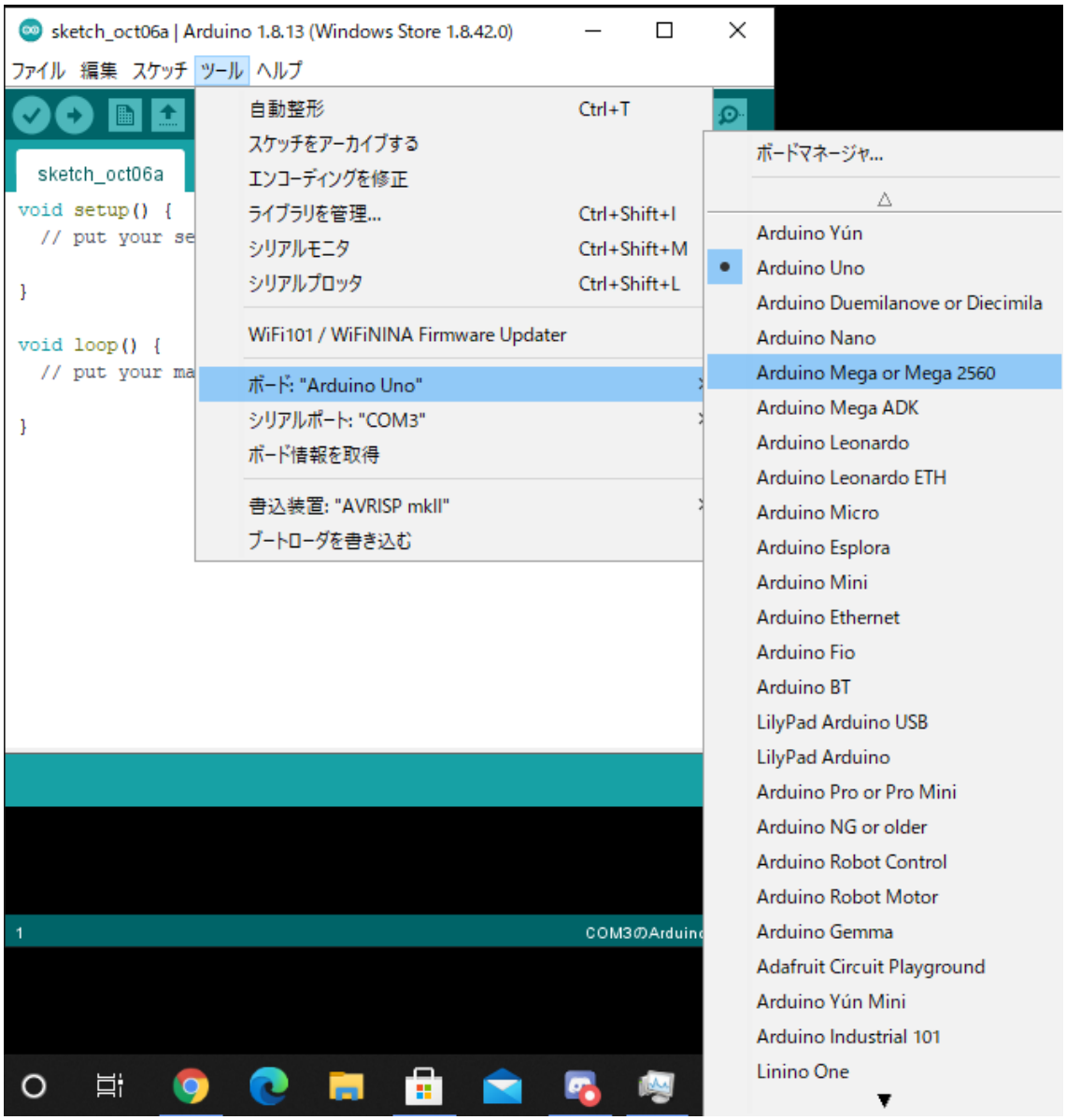

図 2　Arduino IDE の設定

同様にスケッチ→ライブラリをインクルード→,zip 形式のライブラリをインストール

から配布したライブラリをインストールしておく.

3：プログラミング用語集

定数 ： 定まった数．数学における一時関数 y=ax+b の a や b の部分

変数 ： 変わる数．数学における一時関数 y=ax+b の y や x の部分

関数 ： 変数に値を入れると何か一つの解がでるもの．一時関数では x に値を入れると y が定まる．

整数 ： 0 を起点として 1 ずつ足していたっり引いていったりして出てくる数

(例: …-3, -2, -1, 0 , 1 , 2, 3…)

浮動小数点数 ： 実数(小数や整数)，小数点がある数

(例: …-3.0, -2.2, -1.0, 0 , 1.2 , 2.3, 3,4…)

符号なし ： 符号のない数，正の数，非負数

Bit：ビット，単位．2 進数における 1 桁

2 進数 ：0 と 1 で表現される数.2 で繰り上がる．私たちが普段使用してる 10 で繰り上がる数の表現は 10 進数．例(10 進数で 3 は 2 進数で 11)

ライブラリ ： 既に用意されているプログラム(関数や変数)集．呼び出して使う．

インデント ： tab キーやスペースキーを使って行う字下げ

データ型 ： 主に型と呼ぶ. 扱うデータの種類,意味を指定する.

型	意味, 扱えるデータ
int	-32768 から 32767 の整数
unsigned int	0 から 65535 までの符号なし整数
long	2147483648 から 2147483647 までの整数. Int 型の拡張.
unsigned long	0 から 4,294,967,295 の符号なし long 型
float, double	浮動小数点(実数)
char	文字
byte	0 から 255 までの 8bit 数値
word	0 から 65535 までの 16bit 数値
boolean	true(真)か false(偽)
void	関数の定義において何も返さないとき

サイズを指定した整数型

int8_t	8bit の整数

uint8_t	**8bit の符号なし整数**
int16_t	**16bit の整数**
uint16_t	**16bit の符号なし整数**
int32_t	**32bit の整数**
uint32_t	**32bit の符号なし整数**
int64_t	**64bit の整数**
Uint64_t	**64bit の符号なし整数**

4：プログラミング基礎構文集

ライブラリを読み込む

```
#include <ライブラリ.h>
```

変数を定義する

```
型  変数名;

型  変数名 1, 変数名 2;
```

,(カンマ)で区切ることで複数個同時に定義ができる

型の前に const を付けるとリードオンリー(書き換え不可, 定数)になる.

変数に値を入れる

変数名 = 値;

ここでいう値とは変数や定数のこと.

変数の定義と同時に行うことも可能

算術演算子

演算子	意味	使用例
+	和(足し算)	答 = 値 1 + 値 2;
-	差(引き算)	答 = 値 1 - 値 2;
*	積(掛け算)	答 = 値 1 * 値 2;
/	商(割り算)	答 = 値 1 / 値 2;
%	余剰(余り)	答 = 値 1 % 値 2;

ここでいう値とは変数や定数のこと.

++ で +1 , --で-1 を表現することができる

x = x+ y を x += y と表現することができる.+の部分は他の四則演算子(+, -, * , /)に

置き換え可能

関数を定義する

```
型名  関数名(型名  関数内で使う関数外で定義された変数たち) {

 処理

 return  返す数(答え)

}

例

int to, tobe;

int add(int to, int tobe) {

int ans = to + tobe;

return ans;

}
```

型が void の場合に限り return が存在しない

※ 関数内で使う関数外で定義された変数たちを引数と言う

関数を呼び出す

```
関数名(関数内で使う関数外で定義された変数たち)

例

add (to, tobe)
```

コメント

プログラムの動きに関係ないコメントを書くことができる

```
// 一行の時

/*複数行の時

これで囲む*/

例

int to, tobe; // 変数を定義 to は足す数 tobe は足される数

int add(int to, int tobe) { // 関数を定義

int ans = to + tobe;

return ans; // 答えを返す
```

```
}
```

条件分岐

```
if(  条件 1 ) { //  もし条件だったら処理 1 をする

処理 1

} else if (  条件 2 ) {    //  条件 1 でなく条件 2 でなかったら処理 2 をする

処理 2

} else { //  どれでもなかったら処理 3 をする

処理 3

}
```

条件に使用する演算子

演算子	意味	使用例
==	等しい	a == 1
!=	等しくない	a != 1
<=	以下(左辺は右辺以下)	a <= 1
>=	以上(左辺は右辺以上)	a >= 1

<	未満(左辺は右辺未満)	a < 1
>	超過(左辺は右辺超過)	a > 1
&&	論理積(かつ)	a > 1 && b >1
||	論理和(または)	a == 1 || b == 1
!	否定(でなかったとき)	!1

もうひとつの条件分岐

```
switch(変数) {

case 値 // 変数が値だったら処理１ を実行する

処理１

break;

default // どの case にも一致しなかったら処理２ が実行される

処理２

}
```