PT・OT・アスリートのためのプログラミングとマイコン電子工作入門 (3) パソコン・プログラミング

増田 正¹, 大 路 駿 介^{2†}

¹ 福島大学共生システム理工学類, ² 東京医科歯科大学医学部附属病院

要旨: パソコン上で実行するプログラムの作成方法について解説した。対象とする OS は Windows で、プログラミング言語としては C# を取り上げた。プログラム作成のための統合開発環境として Visual Studio を用い、ボタンやテキストボックスなどのグラフィカルなユーザインタフェース (GUI) をもったフォームアプリケーションの作成方法を紹介した。さらに、パソコンとマイコンを USB ケーブル経由で連携させ、マイコンに接続した LED をパソコンからの指令で点灯させるとともに、マイコンによって計測した反応時間をパソコンの画面上に表示させる方法についても触れた。パソコンとの連携により、マイコンの利用分野を大きく広げることができる。

キーワード: Windows, Visual Studio, C#, Arduino

1. パソコンでのプログラミング

プログラムを自作できれば、パソコンの活用分野を大きく広げることができる。既存のソフトウェアでは実現できなかった処理を実行できるようになれば、研究のオリジナリティを高めることができる。あるいは、個々の処理は既知の単純なものであっても、大量のデータを間違いなく処理するために専用のプログラムが必要になるかも知れない。さらに、パソコンとマイコンを連携させることができれば、独自の計測装置や刺激提示装置を開発することもできるようになる。

とりあえずプログラミングの勉強をしてみようという時には、どのプログラミング言語を選択するかが問題になる。現在、広く用いられている言語は多数あり、それぞれ得意な応用分野があるので、一概にどれが良いと決めることはできないが、理工学系の大学では現在でも C/C++ 言語を最初に学習する例が多いようである。オブジェクト指向型言語として Java は幅広い分野で利用されており、近年では Python も注目されている。

どの言語を用いるにしても、言語を習得すること自体を目的とするのではなく、自ら実行したいことをまず決めて、それをどのように実現できるのかを考えることが理解への近道となる。いずれの言語であっても、基本的な計算はす

べて実現可能なはずであるので、いずれかのプログラミング言語を習得すれば、別の言語の理解も促進される。

2. 統合開発環境

パソコン上でのプログラムの作成にあたっては、統合開発環境 (Integrated Development Environment: IDE) と呼ばれるソフトウェアを利用するのが便利である。IDE は、プログラムの生成時、入力時、ビルド時、実行時などのそれぞれの段階において適切な選択肢を示すとともに、エラーを指摘し、ヒントを提示するなどしてプログラムの作成を支援してくれる。

プログラムを新規作成する際に、作成するアプリケーションの種類を選択すると、IDE がプログラムの土台となる基本的な部分を自動的に生成してくれる。ユーザは、このプログラムの土台となるコードの中の適切な位置に独自のコードを追加してプログラムを完成させる。

コード入力時には、変数や関数の候補の提示、関数に与える引数の提示、関数の定義場所や参照場所への移動などの機能を IDE は提供する。また、文法上のエラーも指摘してくれる。

コード入力完了したら、ユーザが入力したプログラムの文字列を、コンピュータで実行可能な形式に変換する。この過程では、IDE がユーザの入力したコードに加えてライブラリなどの関連プログラムを結合して実行可能なプログラムを作成する。

実行可能になったプログラムを IDE から起動すると、実行時のプログラムの流れや、変数値の変化を確認できる。プログラム中の調べたい場所にブレークポイントを設定することにより、プログラムの実行がその位置に到達した時

2020 年 3 月 31 日受付

[†] 〒113-8519 東京都文京区湯島 1-5-45

東京医科歯科大学医学部附属病院スポーツ医学診療センター・アスレティックリハビリテーション部門

大路 駿介

Tel&Fax: 03-5803-4721

E-mail: ohji.spt@tmd.ac.jp

に一時停止し、変数値を確認できる。また、それ以降のプログラムを1行ずつ実行させることもできる。

3. Visual Studio

統合開発環境のソフトウェアは多数あるが、この解説では Microsoft 社が提供している Visual Studio を取り上げる。Visual Studio には無償版と有償版があるが、個人でプログラムを作成する程度であれば無償の Community 版で十分である。現時点での最新版は Community 2019 である。

Visual Studio を利用するためには、Microsoft 社のサイトからソフトウェアをダウンロードし、パソコンにインストールする必要がある。インストール時に、どのような種類のアプリケーションを作成可能にするかの選択があるので、「.NET デスクトップ開発」にチェックを入れて選択する。インストールが完了した後、ユーザのアカウントを Microsoft 社に登録する必要がある。メールアドレスとパスワードを入力すると登録が完了し、利用可能になる。

Visual Studio は非常に多機能なソフトウェアなので、最初は、どこにどのような機能があるのかが分からないと思う。そのため、まずは入門的な課題を行って簡単なプログラムを作成し、基本的な使い方を習得するのが良い。多機能ではあるが、頻繁に利用する機能はそれほど多くはない。

4. C# 言語

Visual Studio では様々なプログラミング言語を利用可能であるが、ここでは C# 言語を利用する¹⁾。C# 言語は、古くから用いられている C 言語、および C 言語をオブジェクト指向型に発展させた C++ 言語をさらに発展させたもので、C/C++ 言語の問題点を解消し、より安全で誤りの少ないプログラムを作成できるようにしたものである。基本的なプログラムの書き方は C/C++ と同じであるが、C 言語におけるポインタを明示的には使えなくして、安全性を高めている。

同様な位置づけの言語として、多くの分野で幅広く用いられている Java 言語がある。Java については、次のタブレットプログラミングの解説で取り上げる。

5. フォームアプリケーション

Visual Studio と C# 言語を利用したアプリケーションのうち、クリックできるボタンなどの GUI 機能をもったフォームアプリケーションを作成する¹⁾。Visual Studio のファイルメニューから新規プロジェクトを選択し、C# 言語の「Windows フォームアプリケーション (.NET Framework)」を選択する。プロジェクト名を入力すると白地のフォームを持ったプログラムの土台が出来上がる。この時点で、問題が生じていないかを確認するために緑三角形アイコンの実行ボタンをクリックする。ビルド時や実行時にエラーが発生せ

ず、画面上に白地のフォームが表示されれば正常に実行されている。

Visual Studio の場合には、プロジェクトの上の階層にソリューションがある。ソリューションは、関連する複数のプロジェクトを束ねたものであるが、1つのプロジェクトだけを持っていても良い。

図1に示す Visual Studio の画面配置では、左上がフォームの作成やコードの入力、右上がソリューションやプロジェクトに含まれるファイルの一覧、左下がビルド時のエラーなどのメッセージの表示、右下がフォームやフォーム上の部品などのプロパティ（属性）表示となっている。これらの配置はユーザが自由に変更することができる。図1中央のウィンドウはフォームの設計画面を切り離して独立した表示に切り替えたものである。

フォームアプリケーション作成のためには、フォーム上にボタンやテキストボックスなどの部品を配置し、これらに対応するコードを入力する。ボタンなどの部品は画面左端のツールボックス内にあるので、必要な部品をマウスでドラッグしてフォーム上の適当な位置に配置する。

それぞれの部品のプログラム内での名前やテキストボックス内に表示されるテキスト、配置される位置や色などは部品のプロパティとして設定されている。部品のプロパティを開くと、これらの設定値の確認や変更を行うことができる。

6. ボタンのクリックに反応してテキストを表示

簡単なプログラムとして、ボタンとテキストボックスを配置して、ボタンを押すとテキストボックス内のテキストが変更されるプログラムを作成する。テキストを切り替えるためにボタンを2つと、テキストボックス1つをフォーム上に配置する。特に指定しなければ、ボタンには button1

図1 フォームアプリケーションを作成するときの Visual Studio の画面

と button2, テキストボックスには textBox1 の名前がつく。分かり易いように, button1 は buttonStart, button2 は buttonStop と名前を変更し, ボタン上に表示されるテキストも「START」と「STOP」に修正する。

それぞれのボタンをダブルクリックすると, ボタンをクリックした時に反応するコードがプログラムの本体である Form1.cs ファイル内に追加される。このコードは関数であり, 関数の本体を表す { } の中に太字のコードを追加する(図2)。

図2は, テキストボックスのテキストプロパティを指定の文字列に変更するというコードである。プログラムを実行するとフォームが表示される。フォーム上のボタンを押すと, 押したボタンに応じてテキストボックス内に表示される文字列が「STARTED」あるいは「STOPPED」に変化する(図3)。

プログラムの作成時や実行時にエラーが生じた場合には, プログラムを1段階ずつ実行し, どの段階でエラーが発生しているかを確認し, 必要な修正を行う。また, Visual Studio がエラーメッセージを表示している場合には, そのメッセージをインターネットで検索するとエラーの原因や対処法が見つかる場合が多い。

7. Arduino との連携

Arduino とパソコンのやりとりは COM ポート (シリアル通信ポート) を通して行う。前回の解説で LED が点灯した後にスイッチを押すまでの反応時間を出力する Arduino プログラムを作成したので, これを修正して利用する。まず, LED の点灯を実験者側のスイッチ 0 を押した時ではなく, パソコン側から文字「N」を受信した時に変更する(図4)。

Arduino のシリアルモニタの入力欄に「N」を入力し「送

```
public partial class Form1 : Form {
    public Form1() {
        InitializeComponent();
    }
    private void ButtonStart_Click(object sender,
        EventArgs e) {
        textBox1.Text = "STARTED";
    }
    private void ButtonStop_Click(object sender,
        EventArgs e) {
        textBox1.Text = "STOPPED";
    }
}
```

図2 テキストボックス書換え用のコード

図3 画面上に表示されたフォーム

```
int ledPin = 9;
int switchPin1 = A1; // 反応時間計測のスイッチ
bool measuring = false; // 反応時間計測時に true に設定
unsigned long startTime, stopTime; // 開始時刻と終了時刻
void setup() {
    pinMode(ledPin, OUTPUT); // 9 番ピンを出力に設定
    pinMode(switchPin1, INPUT); // A1 番ピンを入力に設定
    Serial.begin(115200); // 通信速度を設定
}
void loop() {
    if(measuring) { // 計測モードの時
        bool switchValue1 = digitalRead(switchPin1);
        // スイッチの状態を調べる
        if(!switchValue1) {
            // 反応時間計測スイッチが押されたとき
            digitalWrite(ledPin, LOW); // LED を消灯
            measuring = false; // 計測モードを解除
            stopTime = millis(); // 終了時刻を記録
            Serial.println(stopTime - startTime); // 経過時間を出力
        }
    }
    else { // 計測モードではないとき
        if(Serial.available() > 0) {
            // 受信した文字があるかを調べる
            char a = Serial.read(); // 受信文字を読み込む
            if(a == 'N') { // 受信した文字が N だった場合
                digitalWrite(ledPin, HIGH); // LED を点灯
                measuring = true; // 計測モードに設定する
                startTime = millis(); // 開始時刻を記録
            }
        }
    }
}
```

図4 反応時間計測の Arduino 側プログラム

信」ボタンをクリックして LED が点灯し, これを見た被験者がスイッチ 1 を押す。LED の点灯からスイッチ 1 を押すまでの反応時間が Arduino からパソコンに送信されるので, この値がシリアルモニタ上に表示されていれば Arduino 側は正常に動作している。シリアルモニタを開いたままにしていると, C# のプログラムから COM ポートにアクセスした時にエラーになるので, シリアルモニタは閉じる。

C# のプログラムで LED 点灯の指示である文字「N」を送信したり, 反応時間の値を受信するためには, SerialPort を利用する。ツールボックスにある SerialPort をフォーム内にドラッグすると, フォーム上ではなくフォームの下に配置される。名前は serialPort1 になる。serialPort1 をクリックして, プロパティウィンドウのイベントアイコンをクリックすると, SerialPort に関連した関数を入力できるようになる。DataReceived のところに「serialPort1_DataReceived」と入力すると, シリアルポートにデータが到達した時に反応するコードが Form1.cs 内に追加される。

シリアルポートを利用するためにはシリアルポートの何

番目に Arduino が接続されているかを指定する必要がある。また、通信速度を Arduino 側の設定と一致させる必要がある。これらの設定はフォームの起動時に実行されるコンストラクタ関数 Form() の中に記述する。Form() {} の中に太字のコードを追加する。ここで COMx の x は Arduino が接続されているポート番号で、Arduino ソフトウェアのツールメニューから「シリアルポート」を見ると確認できる。図 5 では、5 番を指定した。

文字を送信するためには、スタートボタンのクリックに反応する部分に「serialPort1.Write("N");」と記入する。

データを受信するためには、serialPort1 にデータが到達した時に反応する関数 serialPort1_DataReceived 内にコードを追加する。これにより、変数 receivedText に受信した文字列が代入される。この次の行にブレークポイントを設定すると、Arduino 側でスイッチを押したときにこの部分が実行され一時停止する。一時停止した状態で、その時の変数の内容を確認できる。

この受信した文字列をテキストボックスに表示したいのであるが、前節で行ったようなテキストボックスへの代入をこの位置で直接行くとエラーになる。そこで、少し複雑になるが ShowText 関数を Invoke で呼び出して表示を行う。これにより、スイッチのオン／オフに応じてテキストボックス

に表示されるテキストが変化する。

パソコン側で表示された反応時間は文字列であるので、反応時間を記録する時には、数値に変換してファイルに保存する必要がある。保存したファイルを Excel などのソフトで開けば、さらにその先の処理を行うことができる。

ここで紹介した例は、1 種類の刺激と 1 つのスイッチという単純な反応時間の計測であるが、2 個あるいは 4 個といった複数の刺激を提示し、それらに対応するスイッチを押すような選択反応への拡張も容易である。次にどの刺激を提示するのかや、どれだけの刺激間隔を設けるのかなどは、パソコン側で乱数を発生して決定することができる。この際には、実験者がフォーム上のボタンをクリックして刺激指令を発生させるのではなく、プログラム内で自動的に指令を出すことになる。そして、あらかじめ設定した回数の刺激を提示した後、プログラムを終了するような構成にする。

8. ライブラリの活用

パソコンでプログラムを作成し、様々な作業を行わせる時に、すべての手順を自分で記述することは現実的ではない。実際には、いろいろな機能をまとめたライブラリを自分のプログラムに組み込んで利用する。

1 つの例として、OpenCV (Open Source Computer Vision Library) という、無料の画像処理用のライブラリを紹介する。このライブラリを用いると、画像ファイルの読み込みや表示などの基本的な機能から、人物の顔認識などの高度な画像処理までを行うことができる。元は C/C++ 言語用に開発されたが、様々なプログラミング言語から利用可能なように拡張されており、C# 言語からも利用可能である。一般的に、ライブラリを提供している WEB サイトから、自分のパソコンに合ったバージョンのソフトウェアをダウンロードし、インストールを行うと利用可能になる。プログラムの作成時には、インストールしたライブラリのファイルシステム内での場所を指定する必要がある。

この他に、3 次元グラフィック表示のための OpenGL (Open Graphics Library) や、行列演算などを行う数値計算ライブラリなどもある。また、パソコンに接続するデータ収録装置などには、対応するライブラリがメーカーから提供されているので、これをインストールすることにより、自作のプログラムから装置を制御できるようになる。

次回（最終回）では、タブレット端末でのプログラム作成方法と Arduino との連携を解説する予定である。

参考文献

- 1) 藤本邦昭：ゼロからはじめる Visual C# 入門、森北出版、(2018)。

```
using System;
using System.Windows.Forms;
namespace WindowsFormsFirst {
    public partial class Form1 : Form {
        public Form1() {
            InitializeComponent();
            serialPort1.PortName = "COM5";
            // シリアルポートのポート番号を設定する
            serialPort1.BaudRate = 115200;
            // 転送速度を設定する
            serialPort1.Open(); }
        // シリアルポートを利用可能にする
        private void ButtonStart_Click(object sender,
            EventArgs e) {
            serialPort1.Write("N"); }
            // シリアルポートに"N"を送信する
        private void ShowText(string msg) {
            textBox1.Text = msg; }
        private void serialPort1_DataReceived(object sender,
            System.IO.Ports.SerialDataReceivedEventArgs e) {
            string receivedText =
                serialPort1.ReadLine();
            Invoke(new Action<string>(ShowText),
                receivedText); } } }
```

図 5 反応時間計測のパソコン側プログラム