Visual Studio による

将棋ゲームの制作

小見山　仁幾　佐伯　拓海

1．まえがき

私たちは Visualstdio 2013 で将棋ゲームを制作した。

2．原　　理

(1)VisualC を使い、Button オブジェクトに駒の画像ファイルを割り当てて将棋の駒に見立てる。

(2)Button の場所を交換するものではなく、Button の中にある情報を、移動するマスの Button に渡すようなプログラムである。

(3)王を取った時に、実行画面のウィンドウを強制的に閉じる。

3．研究内容

去年の先輩のプログラムを参考に作った。

(1)　盤の作成

Label と button を使い 9×9 の 81 マスを作り、少し間を空け 5×4 の 20 マスの持ち駒を表示する場所を 2 つ作り、1 マスごとに番号を振り、マスの間に線を引いた。

(2)　駒の配置

ゲームを開始したときに、歩のある場所の Button には歩の画像を表示すると言った様に、各駒の画像を表示した。

(3)　盤と将棋の背景の色付け

白い背景では将棋を行っている感じが薄れてしまうので番と駒の背景に色を付けた。(図 1)

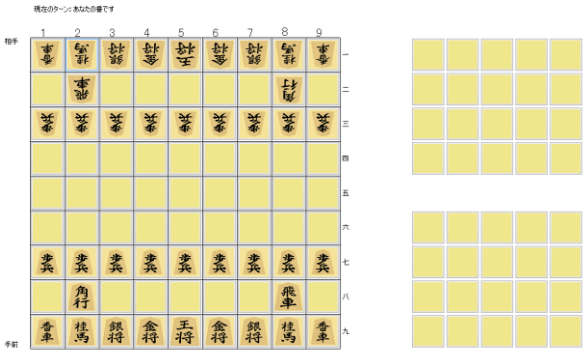

図 1　実行画面

(4)　手番

ゲームを開始したときに、手前側から動かせるようにするために、count という変数を作り、count が 0 なら手前側、1 ならば奥側になるようにした。

手前の行動が終わったら count を 1 にし、奥側の行動が終わったら count を 0 にし、交互に番を交代していくようにし、また Label を使い、どちらが行っているか分かり易くした。

(5)　駒の移動

① 駒の役割

各々の駒に移動できる範囲を設定した。

その際に kyori という変数を用い、今いる駒の番号からいくつ番号をプラス、もしくはマイナスするという様にした。

kyori が 9 であれば、現在いるマスの番号にプラス 9 をする。その数は真正面のマスの番号に当たるので前に進めたいのであれば、kyori !=の後に 9 を入れれば、1 マス分前に進む様な物にした。(図 2)

```
//歩
if (FileNum >= 21 && FileNum <= 29)
{
    if (kyori != -9) return;
    if (Ban[BtnNo - 9] >= 21 && Ban[BtnNo - 9] <= 29) return;
}
if (FileNum >= 1 && FileNum <= 9)
{
    if (kyori != 9) return;
    if (Ban[BtnNo + 9] >= 21 && Ban[BtnNo - 9] <= 29) return;
}
```

図 2　駒の動き

② 味方の上に置けない制限

自陣の駒が、味方の駒を誤って取れない様にした。FileNum という変数を作り、それに、選択した駒が何番か分かるようにした。

更に、FileNum2 という変数を作り、選択した駒を置く場所にいる番号が何番か分かるようにした。そして、選択した駒の番号が自陣の駒の番号であり、その駒を置く場所の番号が自陣の駒の番号であったときに、return でもう一度どこに置くかを選択できるようにした。（図 3）

```
//味方の上に置けない
if (FileNum <= 20 || FileNum >= 41 && FileNum <= 46)
{
    if (FileNum2 <= 20 && FileNum2 != 0) return;
    if (FileNum2 >= 41 && FileNum2 <= 46) return;
}

if (FileNum >= 21 && FileNum <= 40 || FileNum >= 51)
{
    if (FileNum2 != 0 && FileNum2 >= 21) return;
    if (FileNum2 >= 51) return;
}
```

図 3　味方の上に置けない制限

(6)　駒の禁止事項

① 相手の駒を動かせない

自分の番に相手の駒を選択できない様にした。

先程書いた、count という変数を使い、もし count が 0 の時（自分の番）であったのならば、もし選択した駒の番号が相手の駒の番号であれば、メッセージボックスで、『相手の駒です』と表示させ、もう一度、選択できるようにした。（図 4）

```
if ((count == 0 && (Ban[BtnNo] >= 21 && Ban[BtnNo] <= 40 ||
    Ban[BtnNo] > 50 && Ban[BtnNo] < 60)) ||

    (count == 1 && (Ban[BtnNo] < 21 && Ban[BtnNo] > 0 ||
    Ban[BtnNo] > 40 && Ban[BtnNo] < 50))) //相手の駒を動かせない
{
    MessageBox.Show("相手の駒です。");
    return;
```

図 4　相手の駒を動かせない

(7)持ち駒の制限

参考にしたプログラムでは、持ち駒で王を取れる事も可能であったため、制限を加えようと試みた。

① 移動全体の制限

まずは移動自体のプログラムに制限をかけた。（図 5）

Ban < 82、つまりは将棋盤の中でのみの行動であると制限付けた。（図 6）

```
// 移動
Ban[BtnNo] = FileNum;
btn2.BackgroundImage = Image.FromFile("D:¥¥将棋素材¥¥k" + FileNum + ".png");
```

図 5　変化前

```
// 移動  //変えたところ
if (ban < 82)
{
    Ban[BtnNo] = FileNum;
    btn2.BackgroundImage = Image.FromFile("D:¥¥将棋素材¥¥k" + FileNum + ".png");
}
```

図 6　変化後

② 持ち駒に限定した制限

次に、if 文で FileNum>=82 以上、つまりは持ち駒を選択した時、選択した場所の番号が手前側、奥側の駒以外の番号であれば、そのまま駒を置く。

もし手前側か、奥側の駒の番号であれば、先程書いた count を使い、持ち駒を置く場所に選択した所の番号が相手側の駒の番号であれば return して、選択するところまでに戻り、メッセージボックスで『置けません』と表示する。（図 7）

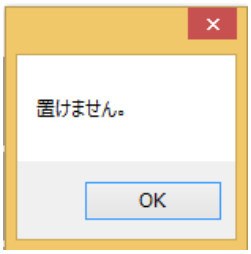

図 7　メッセージ

この際に味方側の駒を選択しても、(5) の②の味方の上に置けない処理があるので問題はない。

② ゲームの終わり

参考にしたプログラムでは王を取った後、メッセージボックスで『あなたの勝利』もしくは『相手の勝利』と出るだけで、その後、ゲームを続けることが出来、更には取った王を使うことが出来るという見過ごしがたい点があったので、王を取ったらゲームが終わるようにした。(図 8)

```
//勝敗//変えたところ
if (FileNum2 == 16)
    MessageBox.Show("先手の勝利");
if (FileNum2 == 16)
    this.Close();


if (FileNum2 == 36)
    MessageBox.Show("後手の勝利");
if (FileNum2 == 36)
    this.Close();
```

図 8　ゲームの終わり

メッセージボックスで表示をした後、更に実行画面を閉じるようにした。(図 9)

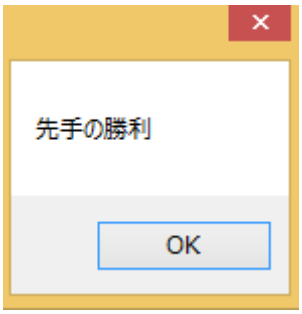

図 9　勝利のメッセージ

4．まとめ

・今回のゲーム作成では、ルールを追加していく事が大半を占めていた。
・2 人いたので相談したりしながら進めることが出来た。
・不具合があることが分かっていたが、すべてを直すことは出来なかった。
・二歩を追加したかった。

5．あとがき

最初、先輩の物を参考にしていた時、全く理解が出来ずに戸惑った。しかし、作っていく内に理解できていくのがとても楽しく、更に自分で考えたプログラムが実行できたときはとても嬉しかった。また機会があれば、将棋のゲームを作ることが出来れば良いと思った。(小見山)

初めてゲームを作ることが出来、とても楽しかったが、それと同時にゲームを作ることの難しさを知ることができいい経験になった。そして 2 人で協力して作ることの大切さを知ることができた。(佐伯)

6．参考文献

素材のプチッチ
http://putiya.com/index.html