ＨＳＰによるゲーム製作　～文法・基礎編～

電子制御工学科　坂口　彰浩

１．はじめに

情報処理センター広報誌Ｎｏ６で「HSP によるゲーム製作」という記事を寄稿しました。この中で、HSP という言語を使うと、比較的簡単にゲームを作ることができる（記事の中では３０行程度のシューティングゲームのスクリプト）ことを紹介しました。この記事では、ゲーム作成を通してプログラミングに興味を持ってもらうような導入用としての内容でした。しかし、HSP という言語の能力はもっと高く、HSP のプログラミングコンテストも開催されるくらいです。そこで、今回は“文法編”として、「興味を持ったので自分でもっと色んなゲームを作りたい。もっと詳しく知りたい。」といったような声(？)に応えて、少し詳しく説明します。

２．ＨＳＰの文法

２．１　プログラミングを始める前に

前回の記事でも書きましたが、プログラミングを行ううえで重要なことについて、もう一度簡単に復習しておきます。コンピュータは天才であると同時にバカでもあるということです。大げさかもしれませんが、コンピュータを使えばできないことはないと言って良いくらい優秀です。しかし、人間に命令されなければ、全く何もできません。人工知能はコンピュータ自身が考えて行動するといっても、人工知能を作るのは人間です。結局は、非常に高性能なコンピュータがあっても使えなければ唯のゴミと一緒です。つまり、「どのように命令したらコンピュータが望み通りに動いてくれるか？」（これをアルゴリズムと呼ぶ）を考えることが重要なのです。

前回の記事を読んでいない方のために、簡単に HSP について書いておきます。HSP は、Hot Soup Processor の頭文字をとったもので、フリーソフトでありながら、非常に簡単にゲームを作ることができるソフトウェアです。また、HSP というキーワードで検索を掛けると非常に多くの HP を見つけることができます。

２．２　画像の表示

まず、最初は画面に画像を表示してみましょう。図１にサンプルスクリプトを示します。

```
01	buffer 2,500,500	;隠れウィンドウの作成. 縦 500 横 500. ID=2.
02	picload "start.bmp"	;画像を読み込む命令
03	screen 0,500,500	;ウィンドウの作成. 縦 500 横 500. ID=0.
04	gcopy 2,0,0,500,500	;隠れウィンドウの画像を表示
05	stop			;命令実行の一時停止
```

図　1　画像の表示スクリプト

２行目の命令が画像を読み込むための命令です。読み込んだ画像は、１行目で作成した隠れウィンドウに表示します。この命令の数字の意味は、前から“ウィンドウのＩＤ番号”、“横のサイズ”“縦のサイズ”です。３行目が画面に表示するウィンドウを作成する命令です。数字の意味は１行目と同じです。４行目が隠れウィンドウに読み込んだ画像を ID=0 のウィンドウにコピーする命令です。４行目の場合、ID が 2 のウィンドウの画像の横 0 縦 0 の場所から横 500 縦 500 の場所までの画像をコピーします。５行目は、命令を一時的に停止する命令です。この命令がないとプログラムが終了してしまい、一瞬でウィンドウが消えてしまいます。HSP のスクリプトを作成するときに注意しなければならないことがあります。それは、例外を除いて、命令の前に Tab キーを入力することです。分かりにくいですが、図１のスクリプトで行番号と命令の間には Tab キーが入力されています。

２．３　文字の表示

画面に文字を表示してみましょう。図２にサンプルスクリプトを示します。

```
01      screen 0,500,500               ;ウィンドウの作成. 縦 500 横 500. ID=0.
02      pos 200, 200                   ;表示位置の設定. 横 200 縦 200
03      color 0,255,0                  ;表示色（緑）の設定.
04      font “MS ゴシック”, 20         ;表示文字の設定.
05      mes “あいう”                   ;文字の表示.
06      stop                           ;命令実行の一時停止
```

図 2　文字の表示スクリプト

２行目は表示する位置を変更する命令です。最初の数字が横の位置、後ろの数字が縦の位置を表しています。３行目は文字の色を変更する命令です。３つの数字は、赤・緑・青の強さを 0 から 255 で表しています。例えば、”0, 0, 0”なら黒、”255, 255, 255”なら白となります。４行目は文字の形や大きさを変更する命令です。表示させたい書体と文字の大きさを変更できます。５行目の命令が文字を表示するための命令です。ダブルクウォーテーションの間に表示したい文字を入力します。

２．４　ボタンの表示

ゲーム中に行動を選択するときなどに使うボタンを表示させてみましょう。図３にサンプルスクリプトを示します。

```
01      screen 0,500,500               ;ウィンドウの作成. 縦 500 横 500. ID=0.
02      pos 200, 200                   ;表示位置の変更. 横 200 縦 200
03      button “クリック”, *click      ;表示色（緑）の設定.
04      stop                           ;命令実行の一時停止
05                                     ;文字の表示.
06 *click                              ;ラベル click
07      pos 180, 300                   ;表示位置の変更. 横 180 縦 300
08      mes “クリックしたね”           ;文字の表示
09      stop                           ;命令実行の一時停止
```

図 3　ボタンの表示スクリプト

３行目がボタンを表示するための命令です。ダブルウォーテーションの間には、ボタンに表示させたい文字を入力します。その後にある“ ***click** ”はラベルと呼ばれ、ボタンがクリックされたときに移動する先のラベルを指定します。この場合、ボタンがクリックされると“ ***click** ”がある６行目に移動します。サンプルスクリプトにあるようにラベルは Tab を入力しないで、行の先頭から*の後に入力します。ボタンが押されない場合は、ずっと４行目の命令で実行を待っていることに注意しましょう。

２．５　変数と乱数

例えば、シューティングゲームで、敵機の弾がいつも同じところから撃たれていては、つまらないゲームになるでしょう。色んなところから弾が飛んできてこそ、やりがいがあるというものです。そんなときに用いるのが乱数です。乱数とは、ある範囲の中のランダムな値のことです。図４に、サンプルスクリプトを示します。

２行目は、乱数の初期化を行う命令です。初期化を行わないと毎回同じ乱数を作ってしまいます。３行目が実際に乱数を発生させる命令です。この場合、1 から 400 の間のランダムな値を決定します。その値は、変数 **tate** に代入されます。ここで、「変数」という言葉が出てきました。変数とは、値を入れておく箱と考えてください。この箱に値を入れると（代入と言う）、いつでも参照できるので、非常

```
01    screen 0,500,500              ;ウィンドウの作成. 縦 500 横 500. ID=0.
02    randomize                     ;乱数の初期化.
03    rnd tate, 400                 ;乱数の決定.
04    yoko=300                      ;値の代入.
05    moji=”あいうえお”              ;文字の代入.
06    pos yoko, tate                ;表示位置の変更. 横 yoko 縦 tate.
07    mes moji                      ;文字の表示.
08    stop                          ;命令実行の一時停止.
```

図 4　変数と乱数のスクリプト

に便利なものです。４行目にあるように、記号“ = “を使って値を箱に入れることもできます。このスクリプトの場合、変数 tate には 1 から 400 までの値が、変数 yoko には 300 が入ることになります。また、５行目にあるように、数字ではなく文字も入れることができます。文字のときは、入れたい文字をダブルクウォーテーションで囲んでください。６・７行目にあるように、箱の中身を参照したいときは、変数名（箱の名前）をそのまま入力してください。特に、７行目の文字の表示で変数を使う場合、ダブルクウォーテーションがいらないことに注意しましょう。

２．６　場合分け

ゲームでは、条件によって実行する命令が違うときがあります。そのような時は、図５のサンプルスクリプトを使います。

```
01    screen 0,500,500              ;ウィンドウの作成. 縦 500 横 500. ID=0.
02    randomize                     ;乱数の初期化.
03    rnd value, 10                 ;乱数の決定.
04    pos 200, 150                  ;表示位置の変更. 横 yoko 縦 tate.
05    if value<=5{                  ;条件５以下.
06           mes “５以下”            ;文字の表示.
07    }
08    if value>=6{                  ;条件６以上.
09           mes “６以上”            ;文字の表示.
10    }
11    stop                          ;命令実行の一時停止.
```

図 5　場合分けのスクリプト

５・８行目の命令が、条件によって実行する命令を変更するための命令です。“ if ”の後に条件を書きます。条件“ value<=5 ”は、「value という箱の中身の数が５以下」を表しています。この条件が正しいときだけ６行目を実行します。条件“ value>=6 ”は、「value という箱の中身の数が６以上」を表していて、この条件が正しいときだけ９行目を実行します。

２．７　繰り返し

例えば、シューティングゲームの場合、数回自機が弾に当たるまでゲームをプレイし続けます。このように、繰り返し続けるためには図６のようなサンプルスクリプトを使います。繰り返すための方法は他にもいろいろありますが、ここではその一例を紹介します。

このスクリプトで繰り返しに関連するのは９行目の命令です。goto 命令は、指定したラベルに強制的に移動させる命令です。このスクリプトでは、ラベル*top に移動します。つまり、このスクリプトは、最後の行まで実行すると再び最初に戻ってくるので、永遠に実行し続けることになります。それ以

```
01      screen 0,500,500      ;ウィンドウの作成. 縦 500 横 500. ID=0.
02      count=0               ;変数 count に 0 を代入.
03 *top                       ;ラベル top.
04      cls 0                 ;画面クリア.
05      pos 230, 220          ;表示位置の変更.横 230 縦 220.
06      mes count             ;文字の表示.
07      count=count+1         ;count の値を１増やして count に代入
08      wait 100              ;１秒待つ.
09      goto *top             ;ラベル top に移動.
```

図 6　繰り返しのスクリプト

外で新しく出てきた命令が、４行目と８行目にあります。cls 命令は,画面を消去する命令です。後に続く数字で消去後の色を選択できます。0 は白、1 は明るい灰色、2 は灰色、3 は暗い灰色、4 は黒色です。８行目の wait 命令は、その名の通り時間を待つ命令です。その後の値で時間を決定します。10ms 単位で変更できるので、サンプル用のように 100 であれば１秒待つことになります。

３．終わりに

HSP の文法について簡単に紹介してきましたが、これらはそのホンの一部です。これ以外にも、色々便利な命令がたくさんあります。しかし、最低限ここで説明したことを使うだけでも、それなりのゲームを作ることができます。例えば、じゃんけんゲーム、もぐらたたきゲーム、ロールプレイングゲームの戦闘シーンといったゲームを作ることができます。さらに、複雑なゲームを作りたいのであれば、参考文献に記載している本や HP を参照してください。

この様に、６種類程度の約束事で簡単なゲームが作れるため、プログラミングを始めてみたい人にとっては非常によい教材だといえるでしょう。ただし、これを読んだからといって、すぐにゲームが作れる訳ではありません。サンプルを入力して、１行１行確認していきましょう。ことわざで言う「急がば回れ」です。スクリプトを打ち込めば打ち込むほど、よいスクリプトが書けるようになります。実は、平成１６年度佐世保高専電子制御工学科公開講座で、これと同様の講座を開きました。その内容とこの内容は全く同じです。この講座の中学生１０数名の受講者は、上記３種類のゲームの内最低１個を２日間（１日目は説明と演習だったので、実際は半日程度）で完成させて帰りました。中には１年生が数名いましたが、彼らは３つの中で一番難しい戦闘シーンのゲームを作り、そして、さらにそれを改良したり、別のゲームを完成させたりと、非常に意欲的に取り組んでいました。中学１年生でもこれ位できるんです。それ以上の学年の人はもっとできるはずなんです。「プログラムは難しい！！」という思い込みを捨てて、とりあえず HSP に触れてみてください。きっと考えが変わると思います。

参考文献

１．うすあじ：「はじめてのＨＳＰ」，工学社（2002）
２．おにたま，うすあじ：「ＨＳＰプログラミング入門」，秀和システム（2002）
３．大槻有一郎：「１２歳からはじめるＨＳＰわくわくゲームプログラミング教室」，ラトルズ（2004）

参考ウェブサイト

１．「Hot Soup Processor のページ」　http://www.onionsoft.net/hsp/
２．「HSP BEGINER'S CLUB」　http://www.rinku.zaq.ne.jp/ultimate/
３．「HSP センター」　http://hspcenter.com/