基礎演習Ⅰ

金曜日第４限

旭貴朗先生

自作ホームページと
Excelマクロプログラミング
及び
自作ゲーム

提出日１月19日

経営学部　経営学科　２年

学籍番号1310160232

ガーリックオニオン

目次

はじめに

今回、私は情報について学ぶために旭教授のゼミに入った。経営と情報は切っても切れないものであると考えており、経営学部の中で情報についても学ぶことができるゼミがこの旭ゼミだったからである。

２年次春学期の課題として自作ホームページを作成することとなった。作成にあたり、基本を HTML のファイルとしながら CSS、JavaScript の使用や簡易な Excel マクロプログラミングも行った。今まで、ホームページを作った経験はないが、授業内容や教材を参考にし、自分なりのホームページを作ることができたといえる。教材通りでないオリジナリティーを出したホームページの作成ということであったが、HTML ファイルにしても CSS にしても多くのタグや数値を使用しているので、この部分を少し変えるだけでも見た目は大きくかわり、他にはないホームページを作ることができた。このレポートは自作ホームページ作成に当たり、その手順と様々な技術、オリジナリティーを出した部分について説明したものである。これを１部とする。第１章では、Web ページの概要について書いている。第２章ではトップページ作成に関する説明をして、第３章では Excel マクロプログラミングについての説明をした。内容については主に 2 年次春学期の授業で扱われたもの及び情業で使用した教材(参考文献　参照)で主に構成されているが、一部インターネットの情報を参考にしている。

秋学期では授業で習った基礎的なプログラミングを元にゲームの作成を行った。もちろんゲームを作った経験はないが、授業内容やインターネット等で調べた内容をもとにオリジナルのゲームを作成することができた。その一連の工程をまとめた。これを２部とする。１章ではゲーム作成の概要を説明している。２章ではゲームを作るにあたって準備しておくべきものを紹介している。３章ではゲームの作成工程について記している。

１部及び２部のまとめは「おわりに」で記している。

1 部

I　Web ページ、HTML 概要

(1)Web ページについて

Web ページとはインターネット上の文書のことであり、Yahoo!などの Web ブラウザで閲覧されるものである。HTML ファイルとこのファイルに関連付けられた CSS ファイル、画像ファイルなど複数のファイルで構成されており、この HTML ファイルを Web ブラウザ上で開くと、複数のファイルの情報が 1 つのページに埋め込まれた形となって表れる。そして、トップページと複数のサブページをリンクすることによってホームページができる。

(2)準備

Web ページ作成に当たり必要なものをまとめる。

① テキストを入力するソフト

HTML ファイルや CSS ファイルはテキストファイルであるため、そのテキストを入力するためのソフトが必要となる。Windows の場合は「メモ帳」、Mac の場合は「CotEditor」を使用する。私が持つ PC の OS は Windows であるためメモ帳を使用する。メモ帳は Windows にあらかじめ入っているため、ダウンロードする必要はない。

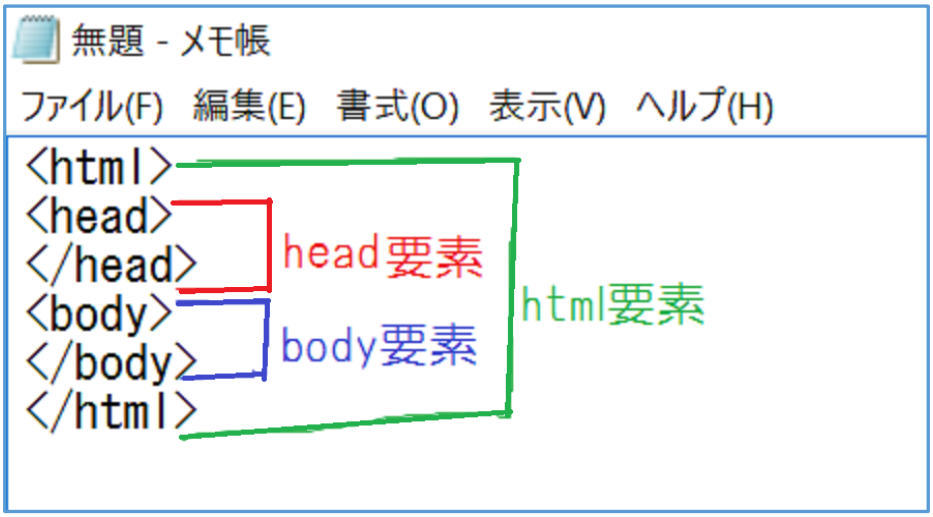
② アップロード場所の確保

たとえ HTML ファイルや CSS ファイルを作成し、自作のホームページを作ったとしても、アップロードしなければ自分のパソコンの中でしか見ることができなく、人に見てもらえない。情報を発信することが目的のホームページで人に見てもらえないようでは意味がない。そのため作成したファイルを Web サーバ上にアップロードするため、アップロードする場所を確保する必要がある。今回はホームページ作成サービスを無料で提供している Yahoo!Japan の「ジオシティーズ (https://geocities.yahoo.co.jp)」を利用する。ジオシティーズは Yahoo!Japan に登録することで利用することができる。

(3)　HTML について

HTML 文書は HTML の開始を示す<html>と終了を表す</html>の間に含まれる。そし

て、開始タグ、内容、終了タグによって構成されている複数の「要素」によって構成されている。開始タグに対し終了タグは「　/　(スラッシュ)」が付いており、スラッシュによりその要素が終了することを表している。



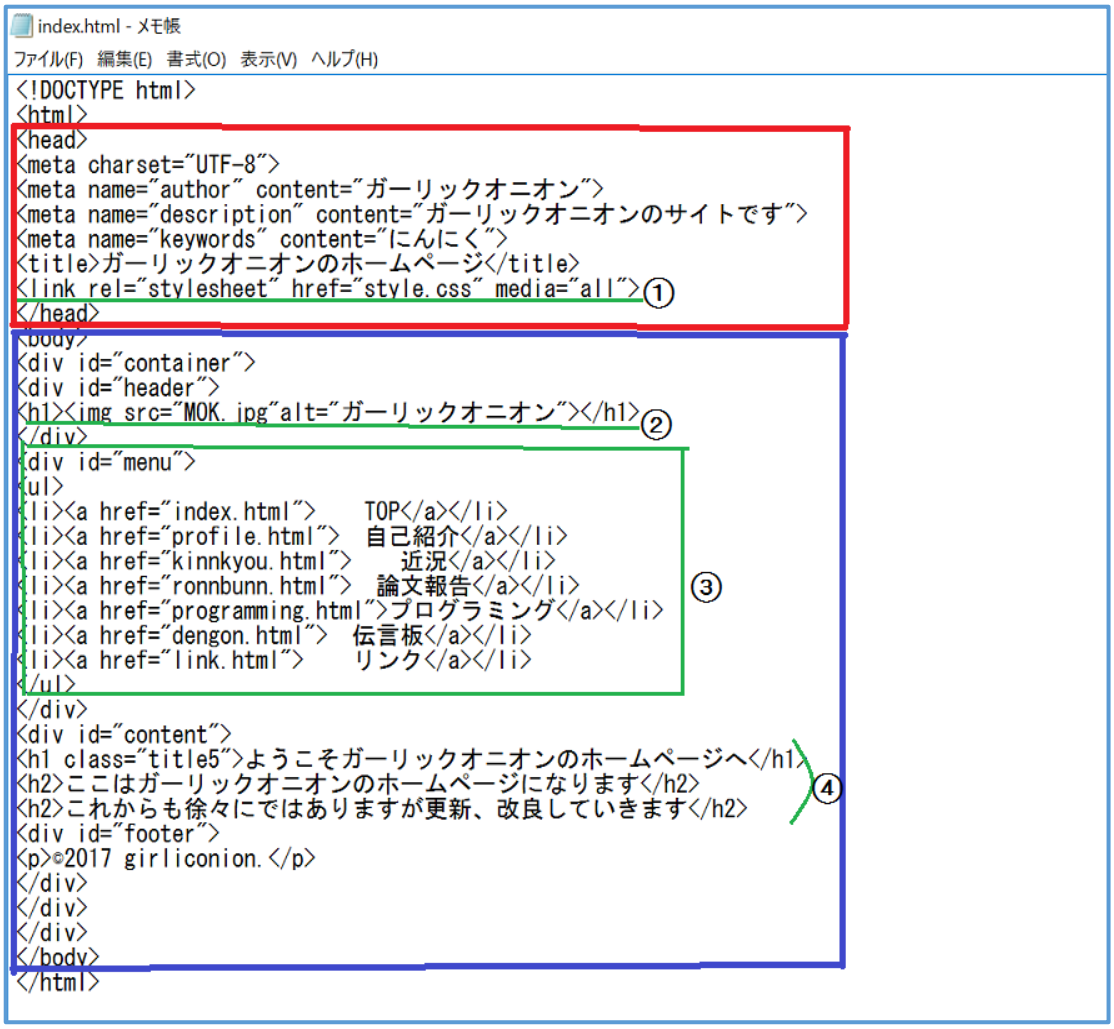
図Ⅰ－１　要素について

図Ⅱ－１が HTML の基本である。head 要素には、文書のタイトルや概要などの meta 要素といわれる表示と直接関係のない補足情報を書く。body 要素にはブラウザで表示される情報を記述する。各要素の開始タグと終了タグの間に内容を書くことによってホームページが出来上がっていく。

Ⅱ　自作ホームページの説明

(1)　トップページの作成

Ⅰ－(2)での準備が整ったら、いよいよページの作成である。メモ帳を開きタグを入力していく。まず作るのはトップページである。



図Ⅱ－１　HTML ソース

図Ⅱ－１はレポート作成時の私のホームページのトップページの HTML ファイル内容であるこのようなタグ打ちをすることによってブラウザ画面でホームページが表示される。

図Ⅱ－２　HTML ソースでブラウザに表示される画面

実際にブラウザ上で表示される画面が図Ⅱ－２である(CSS の反映もされている)。これから、どのようしてこのホームページが出来上がっていくかを説明していく。

◎head 要素

ここではまず DOCTYPE 宣言を行う。HTML にはいくつかバージョンがあるがこの DOCTYPE 宣言を見ることによって、どのバージョンで書かれたかわかるようになっている。今回は HTML5 の DOCTYPE 宣言を記述した。<!DOCTYPE HTML>がそれである。また、head 要素では文字コードやページの作者、関連キーワードなどの meta 要素と呼ばれる情報を入力する。ここに書く情報は基本的には表示と直接の関係はないものとなっている。だが、後にも書くが、CSS とのリンク付けはこの head 要素の中で行う。

①CSS ファイルとのリンク

緑下線①をご覧いただきたい。このタグを入力することによって CSS ファイルとリンクすることができる。CSS はページのデザインを決めるファイルであるため(詳しくはのちの項にて記述)、このリンク付けにより HTML ファイルに CSS ファイルで入力したデザイン情報が反映される。

◎body 要素

ここには直接表示に影響する内容について入力する

②画像挿入

図Ⅱ－３　画像挿入

緑下線②の img タグによって画像が挿入される。「src="MOK.jpg"」が画像の挿入を表す。「""」(ダブルクォート)の間にファイルの名前を入力し、画像を HTML と同じファイルに入れておくことで正常に表示される。。「alt="ガーリックオニオン"」は画像の代わりに表示される文字であり、画像が表示されない環境にある閲覧者のためのものである。結果は図Ⅱ－３である。青矢印のように画像が埋め込まれているのがわかる。今回、この画像はインターネットでダウンロードしたフリー素材を windows のアプリである「ペイント」を利用して、編集したものである。

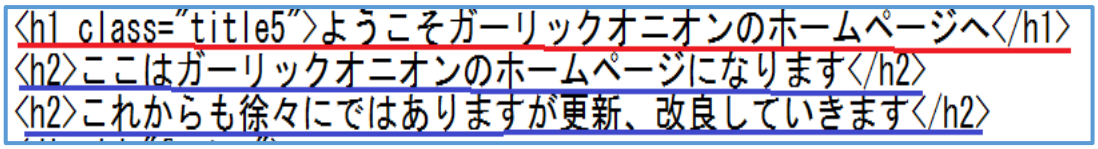
③メニューの入力

緑枠③はメニューについて入力されている。ul(li)要素を利用すると箇条書きに表示される。さらに　自己紹介のように入力すると「" "」の間にファイル名を入力することで別の HTML ファイルとリンクすることができる。そして、これがサブページとなる。<><>に囲まれた文字(この場合では自己紹介)にファイルがリンク付けされ、文字をクリックすることでそのファイルに移動する。ただ、これだけでは図Ⅱ－４のように箇条書きだけでメニューとしてのデザイン性に欠ける。そのため、見栄えをよくするために CSS によってデザインする必要がある。詳しくは CSS の項にて記述する。

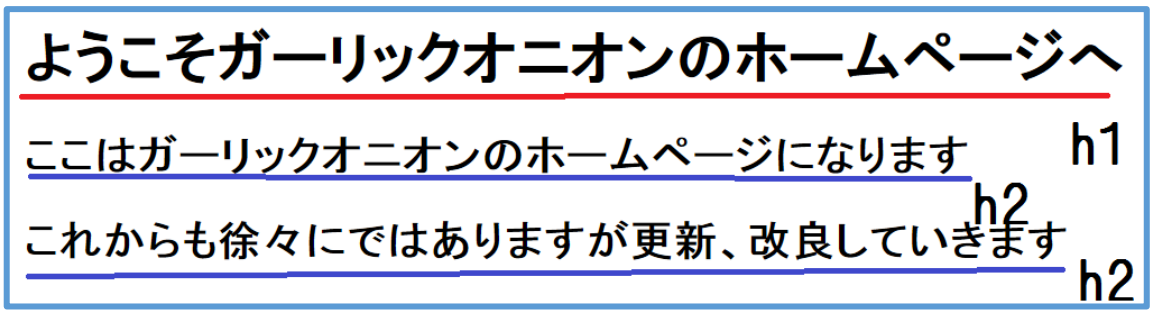
図Ⅱ－４　メニュー

④h 要素(見出し)

ここではｈ要素について説明する。ｈ要素は見出しを表す要素である。ｈ１からｈ６までの六段階あり、ｈ１が最も大きく、ｈ６が最も小さい見出しとなっている。それを<h1>文章</h1>のように囲う。



図Ⅱ－５　見出しのソース



図Ⅱ－６　見出しのソースの結果の画面

図Ⅱ－５はメモ帳上でのタグ打ちであり、図Ⅱ－６はその結果、ブラウザ上に表示される文字である。見ての通り赤下線のｈ１要素と青下線のｈ２要素では文字の大きさが異なり、違う大きさの見出しを作ることができる。なお、ｈ１開始タグ中にある class=”title5” は CSS との関連付けであるので後ほど説明する(図Ⅱ－６の表示には CSS の関連付けはされていない)。

(2)サブメニュー作成

ここではサブメニュー作成に使用した技術とその説明をする。

①Table タグ

table タグとは囲まれた部分を表のようにあらわすタグであり、tr タグ・th タグ・td タグで構成されている。

右の図Ⅱ－７はサブページで使用したものである。

```
<table>
<tr>
<th>名前</th>
<td>ガーリックオニオン</td>
</tr>
<tr>
<th>由来</th>
<td>おいしい</td>
</tr>
<tr>
<th>趣味</th>
<td>温泉、音楽</td>
</tr>
</table>
```

図Ⅱ－７　表のソース

名前 ガーリックオニオン
由来 おいしい
趣味 温泉、音楽

図Ⅱ－８　表のソースの結果

そして、図Ⅱ－８は図Ⅱ－７のタグをブラウザ上で表示したものである。見比べてみるとわかる通り、<th></th>で囲まれた名前・由来・趣味が同じ列。<td></td>で囲まれたガーリックオニオン・おいしい・温泉、音楽が同じ列になっている。そして、<tr></tr>で囲まれたそれぞれが同じ列となる。こうすることで、項目と回答がきれいにならべられ、ひとつの表になっている。

(3)CSS

CSS とは Cascading Style Sheets の略であり、Web ページの見栄えいわゆるデザインに関する役割を担う。CSS は HTML ファイルとは別に CSS ファイルを作り、これを HTML ファイルとリンクすることによって文書にデザインを反映させることができる。CSS によってテキストの装飾や配置、背景色などもう述べてあるようにこのファイルでデザインが決まるので、ここが個性の見せ所といえる。

```
body{
background-color:#fcc800;
front-size:14px;
color:#5c5c5c;
}
#container{
width:900px;
}
#header{
height:140px;
background-image:url(head.png);
}
#content{
width:900px;
float:left;
background-color:#eeeeee;
}
#menu ul{
margin: 0;
padding: 0;
list-style: none;
}
#menu li{
display: inline;
padding: 0;
margin: 0;
float: left;
}
#menu li a{
display: block;
border: 2px solid #666;
background-color: #ccc;
padding: 6px;
text-decolation:none;
color:#333;
width: 112px;
margin: 9px
text-align:center;
```

図Ⅱ－９　CSS ソース

この図Ⅱ－９が私のホームページの CSS ソース(レポート作成時)である。このソースによって、ホームページにデザインがつく。

①HTML への読み込み

HTML への読み込みを行わないと CSS は反映されない。前項でも述べているが、スタイルシートを HTML ファイルと同じファイルに入れたうえで
<link rel="stylesheet" href="style.css" media="all">を HTML ファイルに入力する必要がある(style.css の部分は CSS ファイル名を記す)。

②背景色

背景色は初期では白なので味気ない。CSS で色を変えると雰囲気も大きく変わる。背景色について説明する背景色を決めるのは body 部分の background-color である。

```
body{
background-color:#fcc800;
front-size:14px;
color:#5c5c5c;
}
```

図Ⅱ－１０　CSS 背景色ソース

この図Ⅱ－１０の赤下線が背景色を決めるものであり、#以降の英数字が色を表している。今回は６桁の RGB による色指定を行っている(３桁でもできる)。#以降の６桁のうち前半２桁が赤(Red)、中央２桁が緑(Green)、後ろ２桁が青(Blue)の色の数値表しており、１６進数(それぞれ最大値 FF)となっている。ちなみに図の数値(#fcc800)だとひまわり色(www.colordic.org)となる。明るいが明るすぎなく、温かみのある色となっている。また front-size の大きさは１４ｐｘにしたところ、丁度よい具合にすることができた。color は本文で表示される文字の色を決めるものである。ここではあえて黒ではなく#5c5c5c とグレーにすることでやわらかい雰囲気をだした。

図Ⅱ－１１　背景色(before)

図Ⅱ－１２　背景色(after)

図Ⅱ－１１・１２を見ての通り背景色が変わり雰囲気もよくなった。なお図Ⅱ－１０にある。front-size では文字の大きさ。color では文字の色を決める。この color でも RGB 色指定によりカラー指定を行った。文字のカラーも自由に変えられるので、初期設定の単調な黒にするのではなくグレーのような色にしている。

③ヘッダー、本文記入欄

CSS を利用してできることは背景色だけではない。#header をつかってヘッダーをデザインすることができる。height を利用し大きさを合わせ、ヘッダー用のフリー画像を使用する。ヘッダーはホームページの看板と呼ぶべき存在で印象に大きく影響するので、Windows にもともと入っている、「ペイント」というアプリケーションによって編集する。また、画像の大きさもここで調整することができる。もともと挿入してあった画像と組み合わせて違和感のないようにする。ヘッダーの右部分が寂しいので、ホームページ名をおしゃれなデザインで入力し、１目でガーリックオニオンのホームページとわかるようにした。#content を使って本文を記入する部分をデザインできる。大きさは９００px とする。色は初期状態の真白から変えて、グレー気味の白として、オリジナリティーを出した。

④メニュー表示

CSS を用いない場合メニューは図Ⅱ－４のように箇条書きの簡素なものになってしまうため、CSS を使い個性をつける。

```
#menu ul{
margin: 0;
padding: 0;
list-style: none;
}
#menu li{
display: inline;
padding: 0;
margin: 0;
float: left;
}
#menu li a{
display: block;
border: 2px solid #666;
background-color: #ccc;
padding: 6px;
text-decolation:none;
color:#333;
width: 112px;
margin: 9px
text-align:center;
front-size: 20px;
}
#menu li a:hover{
background-color: #86af22;
color: #fff;
}
a:link{
```

図Ⅱ－１３　CSS メニューソース

図Ⅱ－１４　CSS ソース結果

図Ⅱ－１３が CSS のソースである。なおこのソースは yume.hacca.jp を参考にし、自分なりの改良を加えている。このソースにより、縦並びであったメニュー表を図Ⅱ－１４のように横並びにすることができる。

```
#menu li{
display: inline;
padding: 0;
margin: 0;
```

```
float: left;
}
```

このタグを入力することでメニューを横並びにできる。

```
#menu li a{
display: block;
border: 2px solid #666;
background-color: #ccc;
padding: 6px;
text-decolation:none;
color:#333;
width: 112px;
margin: 9px
text-align:center;
front-size: 20px;
}
```

この#menu li a によってメニューのデザインを変えることができる。display: block によってメニューの枠が四角いものになる。border: 2px solid #666 では枠線の太さとその色が決まる。今回、色は濃い目の灰色にし、それぞれのメニューの枠がつながって見えるように枠線と太さを 2 px として、各枠の隙間を狭くした。background-color: #ccc によって背景色を決められるので、枠線より薄い灰色にした。ここでは 3 桁の RGB による色指定を行った。text-decolation:none を入力することで初期設定ではリンク付けした文字に表示される下線を消すことができる。各枠の大きさを調整することで、メニュー表自体の大きさも調整することができる。大きさを調整してヘッダーと大きさを合わせることで、見映えをよくすることができる。

図Ⅱ－１５　メニューにマウスを重ねる

```
#menu li a:hover{
```

background-color: #86af22;
color: #fff;
}
を入力することによってマウスをメニューの枠に重ねた時に図Ⅱ－１５のように色が変わる仕様になっている。色は#86af22 となりこれは RGB 色指定で図のように黄緑になっている。また、color:#fff によってマウスを重ねた際にメニューの文字が白になるようにした。この色指定は３桁の RGB を用いている。３桁の色指定は６桁のように細かい色の変更はできないが、白などのように単調な色では問題なく使うことができる。

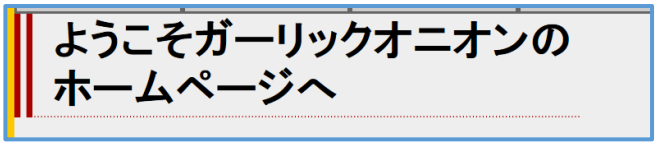
⑤見出しのデザイン

CSS によって見出しのデザインを変えることもできる。ここでもメニューと同じく yume.hacca.jp のサイトを参考にし、改良をしている。

```
.title5{
border-left: 14px double #aa0000;
border-bottom: 1px dotted #aa0000;
text-align: left;
background-color:#eee;
front-size:14px;
color:#000;
margin:0px;
padding: 2px 2px 2px 15px;
width:400px;
}
```

図Ⅱ－１６　CSS 見出しソース

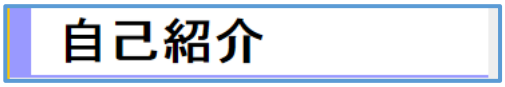
図Ⅱ－１６がそのソースである。border-left: 14px double #aa0000 によって見出しの左部分に赤の線(#aa0000はR部分のみの数値を増やしている)２本を作ることができ、border-bottom: 1px dotted #aa0000 によって見出しの底辺に赤の点線を作ることができる。また、本文記入欄の色がグレーのような色なので、見出しの背景色部分を白(#ｆｆｆ)にすることで、目立たせることに成功した。しかし見出し用なので、このソースを HTML ファイルの中の見出しの部分に反映させる必要がある。そこで見出しの項で書いたように見出しの開始タグ中に class=”title5”と入力する。そうすれば、その部分にのみ、この見出し用 CSS を反映させることができる。



図Ⅱ－１７　CSS 見出し

図Ⅱ－１７のように見出しにデザインがついた。

このソースの中身を少し変えれば、見出しも多種多様にすることができる。



図Ⅱ－１８　サブページ見出し

図Ⅱ－１８はサブページの見出しである。トップページの見出しとの差別化のために、見出しの色、背景色、見出しの大きさをトップページの見出しよりも小さく変えてある。これは class=”title2”とした。自己紹介以外のサブページについてもこの見出しを使用した。

(4)JavaScript

JavaScript とは Netscape 社が提案したもので、ブラウザによってスクリプトが解釈され Web ページに動きを与えることができる。JavaScript は HTML ファイルの中にプログラムを書くことで動きをつけることができる。このレポート作成時にはホームページ上にふたつ、プログラミング欄に加算器を JavaScript によるプログラミングを行っている。

①onMouseOver イベント処理

この onMouseOver のプログラミングを行うと文字通りマウスが入力した文字あるいは画像に重なるとプログラミングした動作を行うというものである。

```
</table>
</h3>
<span onMouseOver="javascript:window.alert('動作を終了しました')">**********</span>
<div id="footer">
<p>©2017 girliconion.</p>
```

図Ⅱ－１９　onMouseOver ソース

図Ⅱ－１９の赤下線の部分実際に行ったプログラミングのソースである。画面上に表示されている**********にマウスのカーソルが重なると動きを見せる。お語気の中身は window.alert である。このプログラミングはウインドウに警告文が現れるようになるというものである。警告文内容は(‘　’)で囲まれている部分に記入する。今回の場合は「動作を終了しました」と表示されるようになっている。

図Ⅱ－２０　ホームページの JavaScript(Before)

図Ⅱ－２１　マウスを＊に重ねた(After)

図Ⅱ－２０の赤丸部分の＊＊＊＊＊＊＊＊＊＊にマウスのカーソルが重なると図Ⅱ－２１のように「動作を終了しました」との警告文が出る。しかし、実際に動作が終了しているわけではなく、ただ警告文が出ているだけである。

②最終更新日

JavaScript によって最終更新日を自動的に表示させることができる。

```
<p>
  このページの最終更新日は
<SCRIPT TYPE="text/javascript">
<!--
  document.write(document.lastModified);
//-->
</SCRIPT>
です
</p>
```

図Ⅱ－２２　最終更新日ソース

図Ⅱ－２２がそのソースである。document.write(document.lastModified)によって最後に更新された日付・時間を自動的に表示させることができる。

図Ⅱ－２３　最終更新日結果

この図Ⅱ－２３の赤枠がその表示結果である。このように最新更新日を秒単位まで表示できる。これをすべてのページに載せることでどのページがいつ更新されたかが、閲覧者にも一目でわかるようになっている。

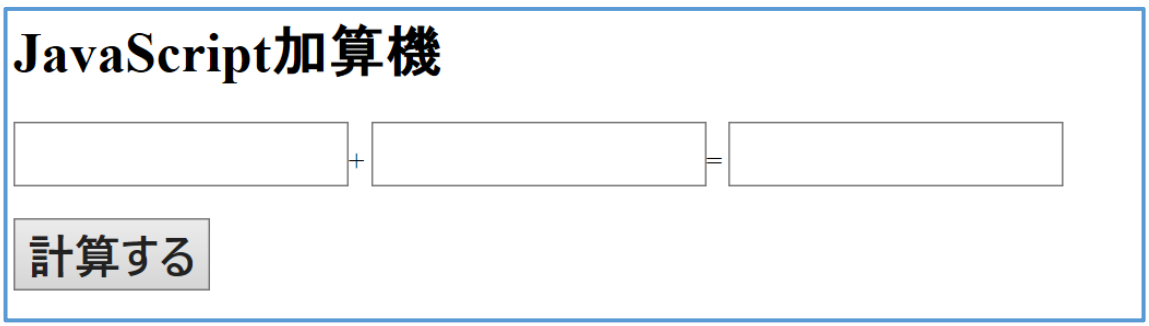
③加算機

JavaScript を利用して加算機を作ることができる。

```
<HTML><HEAD>
<TITLE>JavaSript計算機</TITLE>
<style>input{font-size:20pt;font-weight:bold}</style>

<script type="text/javascript">
<!--
 function calc(){
    form1.Z.value=eval(form1.X.value)+eval(form1.Y.value);
  }
//-->
</script>
</HEAD>

<BODY BGCOLOR="#FFFFFF">
<H1 align="left">JavaScript加算機</H1>

<FORM name="form1">
<input type="text" size="10"name="X">+
<input type="text" size="10"name="Y">=
<input type="text" size="10"name="Z"><BR><BR>
<input type="button"value="計算する"onClick="calc()">
</FORM>

</BODY>
</HTML>
```

図Ⅱ－２４　加算機ソース

図Ⅱ－２４がそのソースである。そして、結果が下の図Ⅱ－２５である。



図Ⅱ－２５　加算機結果

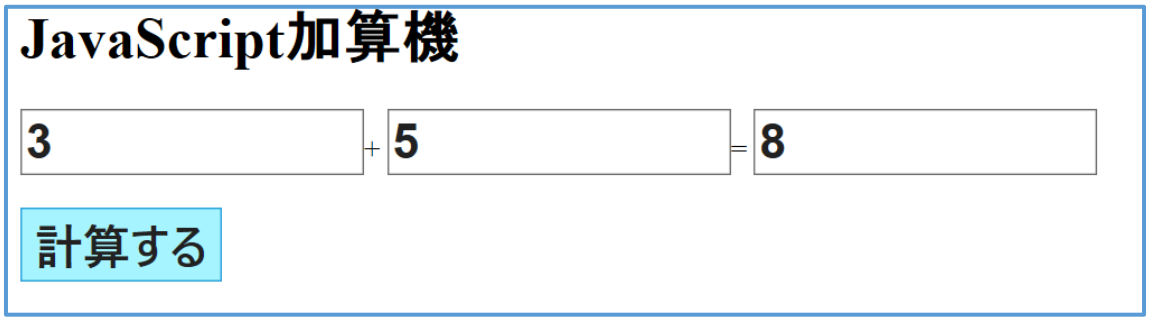
このようになる左の枠と中央の枠に数字を入力し、「計算する」のボタンを押すと計算して右の枠に表示するという仕組みである。

```
<FORM name="form1">
<input type="text" size="10"name="X">+
<input type="text" size="10"name="Y">=
<input type="text" size="10"name="Z"><BR><BR>
<input type="button"value="計算する"onClick="calc()">
</FORM>
```

このコードによって３つの枠と＋・＝の文字、計算するのボタンが表示される。

```
<style>input{font-size:20pt;font-weight:bold}</style>

<script type="text/javascript">
<!--
 function calc(){
     form1.Z.value=eval(form1.X.value)+eval(form1.Y.value);
  }
//-->
</script>
```

このコードで JavaScript の機能が働くようになる X が左枠、Y が中央枠、Z が右枠を表し、eval によって枠に打ち込んだ文字としての数字が“数”として扱われるようになり、ボタンも機能するようになる。



図Ⅱ－２６　加算機の動作

図Ⅱ－２６が実際に操作してみた結果である。３＋５＝８と表示され、正常に機能することが確認できた。

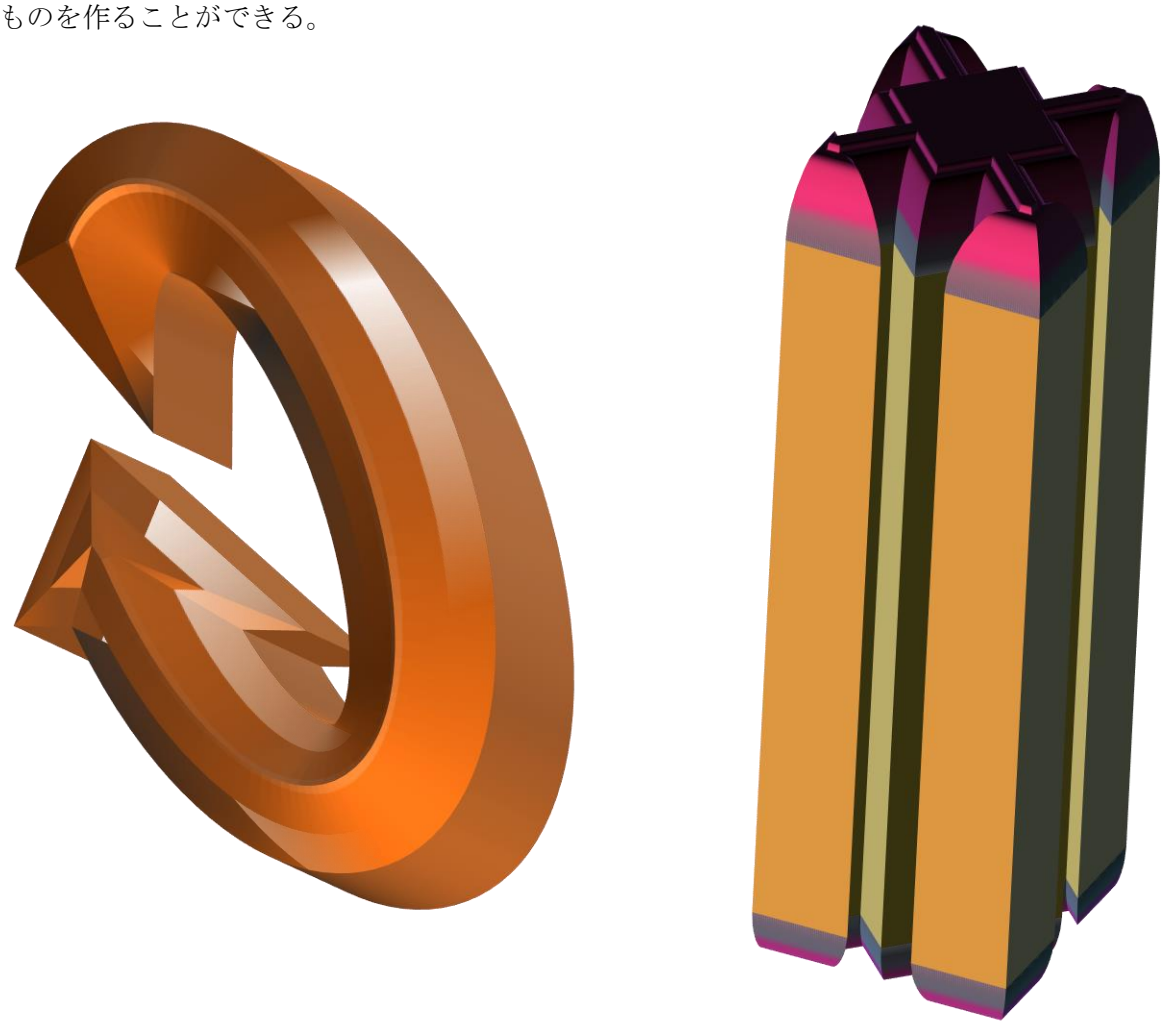
Ⅲ　EXCEL マクロプログラミング

(1) 概要

Excel などのアプリケーションを動かすプログラムをマクロという。Excel のマクロは VBA(Visual Basic for Application)という言語の構文にしたがって記述することが必要となる。Excel にボタンをつけボタンにプログラミングを加えることで、ボタンを押した際に動きをおこすということができる。

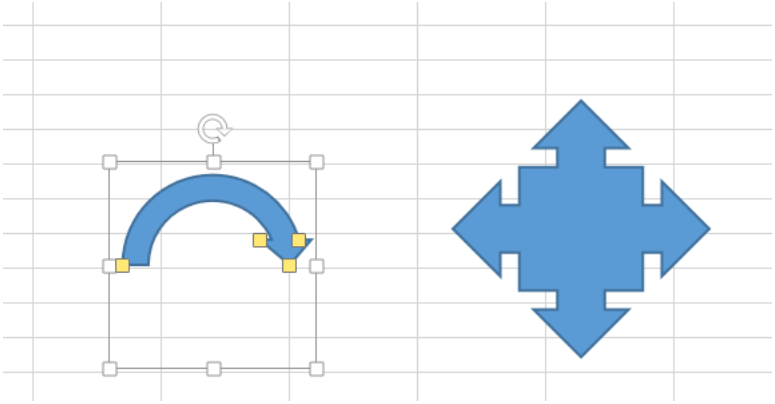
①回転する３D オブジェクト

Excel マクロプログラミングを利用して３D オブジェクトを回転させることができる。まず、Excel を開き、回転させる図形を選択する。まだ初心者なので、あらかじめ Excel 内に入っている図形を選択する。そして、図形の書式設定から奥行きをつけ、３D にしていく。この書式設定からは光や影、図形の質感などを調整することができるため、オリジナルのものを作ることができる。



図Ⅲ－１　回転させる３D オブジェクトモデル

図Ⅲ－１の３Ｄモデルを作ることができる。この図形も、もともとは下の図形であった。



図Ⅲ－２　　３Ｄモデルの元の図形

Excel 上で数値や書式を変えるだけで、図Ⅲ－２を図Ⅲ－３のように変えることができる。Excel 内だけでも影や光、質感を変えることができるのである程度は自由なデザインを作れるが、別のアプリケーションを利用することで Excel にもともと入っている図形以外にも多様なものを使用することができるようになる。

図形を設置した後は、ボタンを設置する。このボタンにプログラミングをする。

```
Sub ボタン2_Click()
|    For n = 1 To 800
         Cells(1, 2) = n
         '図形を3度回転させる'
         Sheet1.Shapes("環状矢印 1").IncrementRotation (3)
         If n < 500 Then
           Sheet1.Shapes("四方向矢印吹き出し 2").IncrementRotation (-3)
           End If
           DoEvents
    Next n

End Sub
```

図Ⅲ－３　　３Ｄオブジェクト回転用ソース

図Ⅲ－３がボタンに施したプログラミングである。IncrementRotation によって図形を回転させる。(　)の中に入れた数字が一回に回転する角度であり、今回は３度と－３度(逆回転で３度)ということである。それを For n=1 to 800 によって環状矢印は８００回繰り返すようにした。四方向矢印吹き出しは If n <500 とし、環状矢印が５００回繰り返すまで回転するとういうことした。この部分は数字を変えて、自分の好きなようにできる。このプログラミングを入力したあと、デバックで VBAProject のコンパイルをする。正常に処理されれば、ボタンを押した際に動きを見せることができる。結果、正常に動作を行うことができ、実際に回転させることができた。

２部　ゲームの作成

Ⅰ　ゲーム作成概要

今回は、秋学期の授業で習った基礎的な Flash プログラムや Flash 作成ソフトの操作を生かして Flash ゲームを作成した(授業で習った内容については『Flash プログラミング：Top』参照)。２章では、作成したゲームが完成するまでの一連の流れについて報告する。まずは作成にあたって必要なものを紹介、そして作成するゲームについて説明する。次にゲームが完成するまでの作成手順について説明する。

Ⅱ　ゲーム作成の準備

ここではゲームを作成するにあたって必要なものを紹介する。

①ゲームの計画書

ゲームを作るためには当然、どのようなゲームを作るのかを考える必要がある。頭の中で考えるだけでなく、ジャンルや動作、プレイ画面や流れを紙などにまとめておいた方が作成をスムーズに行うことができる。今回作成したゲームの内容については、後ほど説明する。

②ゲーム作成ソフト

ゲームの内容を考えまとめることができたとしても、ゲームを作成するソフトがなければ、作ることはできない。Flash ゲーム作成ソフトにも多くの種類があるが今回は KoolMoves を使用する。なお、使ったのは「体験版」であり、製品版ではない。もちろん、Flash 以外にもゲームを作成するソフトは多く存在する。それぞれ機能に違いがあるので、自分にあったソフトを見つけるのもいいだろう。

③　アップロード場所

ゲームが作れたとしても、インターネット上にアップロードできなければ自分のパソコンでしか遊ぶことしかできず、だれも遊べない。それではもったいない。そのため、アップロード場所を作った方がいいだろう。今回は１章で作成した自分のホームページ(http://www.geocities.jp/garlic_onion_gd/)にアップロードする。

④　音楽素材

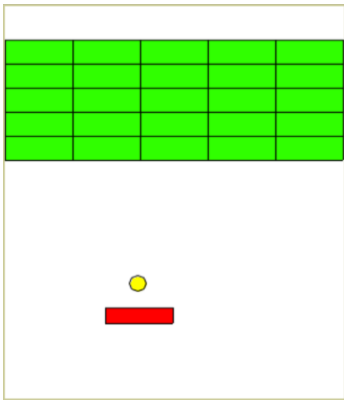
ゲームは音楽がないと盛り上がりに欠ける。そこで、音楽素材をあらかじめ用意しておく。ゲームの内容にあった音楽だとよい。注意すべきなのは著作権である。当然無断使用するのは著作権侵害にあたる。なので、フリー素材の使用をおすすめする。フリー素材とはいえ、

もちろん使用元を著作権情報覧に書くのを忘れないように。今回は『甘茶の音楽工房』(http://amachamusic.chagasi.com/)様から音楽をお借りした。

Ⅲ　ゲームの作成

⑴　ゲームの内容

今回計画したゲームの内容は、ブロック崩しである。バーを動かし、ボールを弾いてブロックに当てて、消していき、ブロックを全部消せたらクリア、ボールを落としたらゲームオーバーというものである。



図Ⅲ－１　メイン画面作成案(著者作成)

簡単に示すと図Ⅲ－１のようになる。緑のものがブロック、赤のものがバー、黄色のものがボールである。しかし、これだけでは面白くない。そこで、工夫点として、複数のステージを用意する・BGM を用意する・背景を変えるなどを施す。

ステージについて、ステージ１では図Ⅲ－１のようなシンプルなものにする。ステージ２では障害物となる、ボールをぶつけても消すことができないブロックを入れる。ステージ３ではブロックを星型に配置する。ステージ４では、バーとブロックとの間を狭めて、難易度を極端に上げる。

ゲームの流れとしては、まずタイトル画面があり、START を押すとステージセレクト画面へ移動する。ステージセレクト画面で STAGE１～４までの表示を押すことで、それぞれのステージがはじまる。ブロックをすべて消せたらクリア画面へ、ボールを落としたらゲームオーバー画面へ移動する。クリア・ゲームオーバー画面で NEXT を押すと著作権情報画面に移動する。

以上の点を計画としてまとめ、ゲーム制作にとりかかる。

⑵　ゲームの作成

KoolMoves の体験版ではフレーム一枚分しか途中保存することができない。そのため、

メインとなるステージ１を作り、そのあとにステージ２～４・タイトル・ステージセレクト・クリア・ゲームオーバー・著作権情報の画面を制作する。なお、制作は『Flash ゲーム講座&アクションスクリプトサンプル集(https://hakuhin.jp/as/block.html)』のブロック崩し作成を元にしているが、このホームページの著者とゲーム作成ソフト及びアクションスクリプトのバージョンが異なるため、その都度工夫をしている。

①下準備

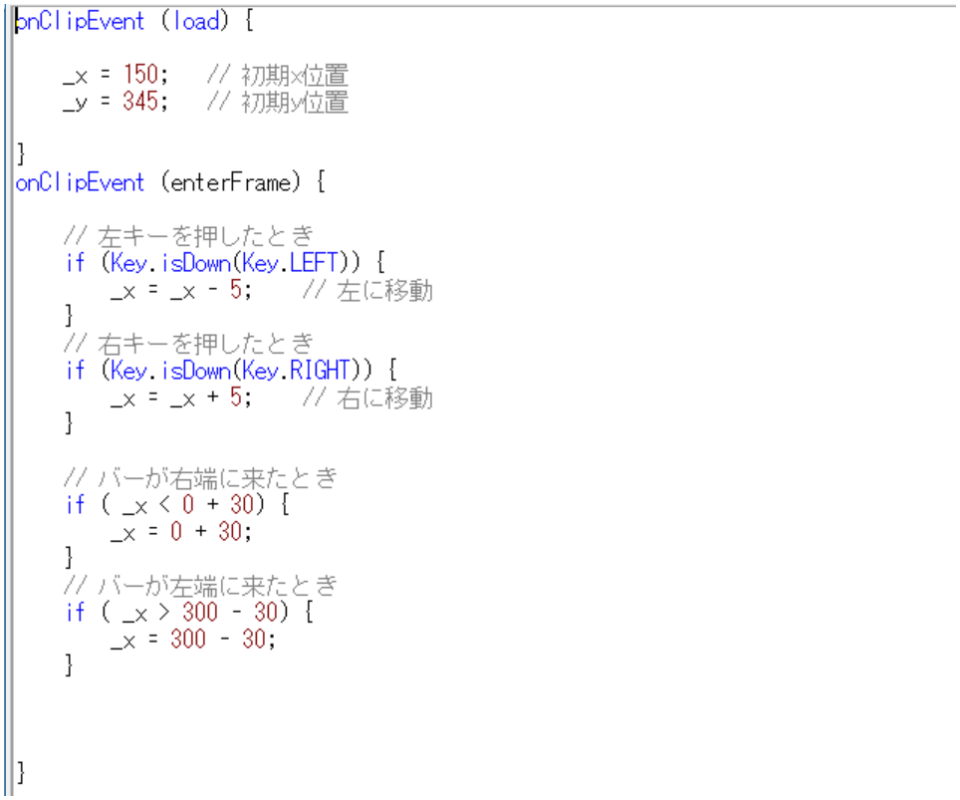
まずは、下準備をする。ファイルからエクスポート設定を開き、HTML ファイルから「ムービーのエクスポート」のチェックマークをはずす。これでループしなくなる。

次に画面の大きさを設定する。ムービーからムービーのサイズを選択し、「ムービーの幅(横の長さ)」と「ムービーの高さ(縦の長さ)」を変えて画面の大きさを決める。今回は、ムービーの幅を 300px、ムービーの高さを 375px とする。これは、ムービーのサイズを大きくしすぎると、動作が遅くなってしまうためである。

②バーを作る

下準備が終わったら、本格的な作成を始める。まずは、ブロック崩しにおける自機であるバーを作成する。長方形を描いた後、図形のプロパティから位置・サイズの編集を選び、幅を６０・高さを１５に設定する。そして、色を変更する(ここでは赤とする)。これらが終わったら、形状メニューから「ムービークリップへ変換」を選び、ムービークリップにする。名前を bar に変更していく。

ムービークリップに変換したら、アクションスクリプトにプログラミングしていく。



```
onClipEvent (load) {

    _x = 150;   // 初期x位置
    _y = 345;   // 初期y位置

}
onClipEvent (enterFrame) {

    // 左キーを押したとき
    if (Key.isDown(Key.LEFT)) {
        _x = _x - 5;    // 左に移動
    }
    // 右キーを押したとき
    if (Key.isDown(Key.RIGHT)) {
        _x = _x + 5;    // 右に移動
    }

    // バーが右端に来たとき
    if ( _x < 0 + 30) {
        _x = 0 + 30;
    }
    // バーが左端に来たとき
    if ( _x > 300 - 30) {
        _x = 300 - 30;
    }



}
```

図Ⅲ－２　バーのアクションスクリプト(著者作成)

図Ⅲ－２がバーのアクションスクリプトである。onClipEvent (load)を使って初期位置を設定する。_x = 150、_y=345 とする。ちなみに x の値は画面の左上を 0 として 1 p x 右にいくごとに 1 ずつ増え、 y の値は画面の左上を 0 として 1 p x 下にいくごとに 1 ずつ増えていく。

続いて、onClipEvent (enterFrame)を使って矢印キーによるバーの移動ができるようにする。ブロック崩しではバーの移動は右と左の 2 方向のみである。左の移動は、if (Key.isDown(Key.LEFT))によって可能となるこのプログラムの中に　_x = _x – 5　を打ち込むことで、←(左矢印)キーを押したときに x の値が 5 下がり、左に移動できるようになる。今回は 5 p x ずつ移動できるようにする。右への移動は if (Key.isDown(Key.RIGHT))を使う。このプログラムの中に　_x = _x + 5　を打つ。これにより、→(右矢印)キーを押したときに右に 5 p x ずつ移動できるようになる。

左右に移動できるようになったが、このままでは、いくらでも移動できるため、バーが画面の外に出て行ってしまう。そこでバーの移動できる範囲を制限する。

```
if ( _x < 0 + 30) {
    _x = 0 + 30;
```

```
}
```

このプログラムにより、図の中心のｘ座標の値が３０以下になろうとしたとき(バーの幅が６０なのでバーの左端のｘ座標が０以下になろうとしたとき)、それ以上ｘ座標が下がらないようになる。つまり、バーの左端が画面の左端より左に行かなくなるので、画面の左外に出ないようになる。続いて、

```
if ( _x > 300 - 30) {
        _x = 300 - 30;
        }
```

を入力する。このプログラムにより、図の中心のｘ座標の値が２７０以上になろうとしたとき(バーの幅が６０なのでバーの右端のｘ座標が３００以上になろうとしたとき)、それ以上ｘ座標が上がらないようになる。つまり、バーの右端が画面の右端より右に行かなくなるので、画面の右外に出ないようになる。これにより、バーが画面の外に出ることは完全になくなった。これでバーのプログラミングは終了である。

③　ボールを作る

バーを作り終えたら、ボールを作っていく。球体描くツールを利用してボールをデザインしたら、プロパティの位置・サイズの編集から幅１５・高さ１５に設定する。色を変えたら(今回は黄色とする)、ムービークリップに変換する。名前を ball に変更していく。

```
onClipEvent (load) {
    _x = 150.0; // 初期位置x
    _y = 330.0; // 初期位置y

    dx = 1;      // x移動量
    dy = -3;         // y移動量

}
onClipEvent (enterFrame) {

    // 移動量加算
    _x = _x + dx;
    _y = _y + dy;
    }
```

図Ⅲ－３　ボールの移動(著者作成)

図Ⅲ－３がボールの移動に関するプログラミングである。まず初期位置として、x の初期位置を１５０、y の初期位置を３３０とする。dx と dy はフレームごとの x と y の移動量である。縦軸の移動の速さの方が横軸の移動の速さよりも速くなるようにした。

続いて、画面の端に来た時に玉が反射するようにする。

```
// 左端に来たとき
    if (_x < 0 + 7.5) {
        _x = 0 + 7.5;
        dx = dx * -1;
    }

    // 右端に来たとき
    if (_x > 300 - 7.5) {
        _x = 300 - 7.5;
        dx = dx * -1;
    }

    // 上端に来たとき
    if (_y < 0 + 7.5) {
        _y = 0 + 7.5;
        dy = dy * -1;
    }

    // 下端に来たとき
    if (_y > 375 + 7.5) {
        _y = 375 + 7.5;
        dy = dy * -1;
    }
```

図Ⅲ－４　ボールの壁反射(著者作成)

図Ⅲ－４はボールが壁(画面の端)にあたった時に反射するようになるプログラミングである。まず左端にボールが当たった時に反射するようにする。

```
if (_x < 0 + 7.5) {
        _x = 0 + 7.5;
        dx = dx * -1;
}
```

このプログラミングにより、ボールの中心のｘ座標が 7.5 以下になろうとしたとき(ボールの幅は１５なのでボールの端のｘ座標が０以下になろうとしたとき)、ボールが反射するようになる。dx = dx * -1 によって反射移動がかのうになる。同じように右端・上端・下端にボールが当たった時のボールの動きを図Ⅲ－４のように入力する。

なお、下端の動きについてだが、最終的には下に落ちたらゲームオーバーになるので必要ないが、現段階では動きの確認も含めて入力し後ほど削除する。次に、バーとボールが当たった際に反射するようにする。

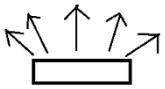
```
// ボールとバーが当たったとき
    if (_root.ball.hitTest(_root.bar)) {
        _root.ball._y = _root.bar._y - 15;
    }

if (_root.ball.hitTest(_root.bar)) {
        _root.ball._y = _root.bar._y - 15;
        dy *= -1;
        dx = ((_root.ball._x - _root.bar._x) / (20 + 5)) * 2;
    }
```

図Ⅲ－５　バーとの反射(著者作成)

図Ⅲ－５のプログラムによってボールがバーに当たった時に反射するようにできる。

```
if (_root.ball.hitTest(_root.bar)) {
        _root.ball._y = _root.bar._y - 15;
}
```

これにより、バーとブロックとの当たり判定をとることができる。次に反射するようにする。ボールとバーによる反射は、ボールと壁との反射とは少し異なる。



図Ⅲ－６　バーとの反射

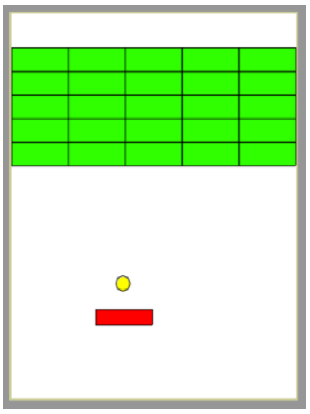
ボールはバーと当たったら図Ⅲ－６のように反射する。右端に当たったら右に、左端に当たったら左にボールは移動するといった具合である。そのためのプログラミングを入れる。それが

```
if (_root.ball.hitTest(_root.bar)) {
        _root.ball._y = _root.bar._y - 15;
        dy *= -1;
        dx = ((_root.ball._x - _root.bar._x) / (20 + 5)) * 2;
    }
```

である。これにより、ボールが当たるバーの座標ごとにボールが移動する方向が変わるのである。これでボールの基礎は終わりである。次はブロックを作っていく。

③　ブロックを作る

次はボールによって消されるブロックを作っていく。ペイントツールを使って長方形を描く。プロパティの位置・サイズの編集から幅を 60、高さを 22.5 に設定する。色を変えたら(今回は緑にする)、ムービークリップに変換する。ムービークリップの名前を bloc10 に変更する。ブロックは複数用意するので、block0 を複製する。ブロックは２５個用意する。名前は block0～24 とする。ブロックを用意したら配置をする。STAGE１ではきれいに並べるが、ボールが上部で反射できるように配置する(図Ⅲ－７参照)。



図Ⅲ－７　ブロック配置 STAGE1(著者作成)

ブロックにはプログラムは打ち込まない。

④　ボールへのプログラミング

次にボールプログラミングを行い、ボールとブロックが当たった時のアクションをプログラミングしていく。まずはボールと block0 との当たり判定をとる。
onClipEvent (load)の｛　｝の中に

flg0=0

を入力する。これによって flg0 という値が設定され、初期値は 0 となる。次に onClipEvent (enterFrame)の　｛　｝の中に

```
if (_root.ball.hitTest(_root.block0)) {
        if(flg0 == 0) {
        _root.block0._visible = false;
        flg0 = 1;
          }
```

を入力する。
このプログラミングによりボールと block0 がぶつかったときに、block0 が消えるようになる。内容としては、if (_root.ball.hitTest(_root.block0))がボールに block0 が当たった時のアクションの範囲を示し、if(flg0 == 0)は flg0 の値が 0 の時にアクションを起こすという意味である。_root.block0._visible = false;は block0 が見えなくなるというものである。flg0 = 1 は flg0 の値を 0 から 1 にするという意味である。つまりボールは block0 に初めて当たった時、block0 は見えなくなるということだ。ここでの注意は block0 は見えなくなるとい

うだけで、一見消えているように見えるが図形は消えておらず存在するということだ。
　次にボールと block0 があったった時の反射について考える。当たった時の反射のパターンは６パターンある。

図Ⅲ－８　ブロックの反射箇所(著者作成)

図Ⅲ－８がそのパターンである。色分けされている通り、左上角・左・左下角・右上角・右・右下角・上・下の６カ所ごとに反射のパターンを変える。
　まず左上角の反射について考える。onClipEvent (enterFrame)の　｛　｝の中のさらに if(flg0 == 0)｛　｝の中に

```
        if(
_root.ball._x+7.5 < (_root.block0)._x-30+6 &&
_root.ball._y+7.5 < (_root.block0)._y-11.25+6)
{
                dx = -3;
                dy = -3;
            }
else if
```

を入力するこうすることでボールが block0 の左上角に当たった時に左上方向に反射し、移動するようになる。if(flg0 == 0)｛　｝内に入れることで二回目以降の反射を防ぐことができる。(先ほども述べたが block0 は見えなくなるだけなので、if(flg0 == 0)｛　｝内にいれないと block0 が消えた(見えなくなった)後にも反射し続けてしまう。これではゲームにならない。)
　次に block0 の右上角に当たった時の反射のプログラミングをする。左上角の時と同じように、onClipEvent (enterFrame)の　｛　｝の中のさらに if(flg0 == 0)｛　｝の中に

```
else if (
(_root.block0)._x+30-6 < _root.ball._x-6.5 &&
```

```
_root.ball._y+6.5 < (_root.block0)._y-11.25+6
) {
                dx = 3;
                dy = -3;

        }
```
と入力する。これにより、ボールが block0 の右上角に当たった時に右上の方向に反射し、移動できるようになる。次に block0 の左下角に当たった時の反射のプログラミングをする。左上角の時と同じように、onClipEvent (enterFrame)の　｛　｝の中のさらに if(flg0 == 0)　｛　｝の中に、

```
else if (
_root.ball._x+7.5 < (_root.block0)._x-30+6 &&
(_root.block0)._y+11.25-6 < _root.ball._y-7.5
) {
                dx = -3;
                dy = 3;

        }
```
と入力する。これにより、ボールが block0 の左下角に当たった時に左下の方向に反射し、移動できるようになる。次に block0 の右下角に当たった時の反射のプログラミングをする。左上角の時と同じように、onClipEvent (enterFrame)の　｛　｝の中のさらに if(flg0 == 0)　｛　｝の中に、

```
else if (
(_root.block0)._x+30-6 < _root.ball._x-7.5 &&
(_root.block0)._y+11.25-6 < _root.ball._y-7.5
) {
                dx = 3;
                dy = 3;

        }
```

と入力する。これにより、ボールが block0 の右下角に当たった時に右下の方向に反射し、移動できるようになる。次に block0 の上側の面に当たった時の反射のプログラミングをする。左上角の時と同じように、onClipEvent (enterFrame)の　｛　｝の中のさらに if(flg0

== 0) { } の中に、

```
        else if (_root.ball._y+7.5 <
(_root.block0)._y-11.25+6) {
                dy = -3;
            }
```

と入力する。これにより、ボールが block0 の上側の面に当たった時に通常のように反射し、移動できるようになる。次に block0 の下側の面に当たった時の反射のプログラミングをする。左上角の時と同じように、onClipEvent (enterFrame)の { } の中のさらに if(flg0 == 0) { } の中に、

```
         else if ((_root.block0)._y+11.25-6 <
_root.ball._y-7.5) {
                dy = 3;

            }
```

と入力する。これにより、ボールが block0 の下側の面に当たった時に通常のように反射し、移動できるようになる。次に block0 の左側の面に当たった時の反射のプログラミングをする。左上角の時と同じように、onClipEvent (enterFrame)の { } の中のさらに if(flg0 == 0) { } の中に、

```
        else if (_root.ball._x+7.5 <
(_root.block0)._x-11.25+6) {
                dx = -3;

            }
```

と入力する。これにより、ボールが block0 の左側の面に当たった時に通常のように反射し、移動できるようになる。次に block0 の右側の面に当たった時の反射のプログラミングをする。左上角の時と同じように、onClipEvent (enterFrame)の { } の中のさらに if(flg0 == 0) { } の中に、

```
        else if ((_root.block0)._y+11.25-6 <
_root.ball._x-7.5) {
                dx = 3;

            }
```

と入力する。これにより、ボールが block0 の左側の面に当たった時に通常のように反射し、移動できるようになる。これらのプログラミングを間違いなく打ち込めることができれば、block0 に関するプログラミングは入力完了である。

block1~23 の分のプログラミングも入力する。block0 で打ち込んだプログラミングの flg と block の後ろにつく番号を変えるだけでよい。ブロック２５個分のプログラムの打ち込みが終わればブロックのプログラミングは終了である。

⑤　ゲームオーバー判定

次にゲームオーバー判定をつくる。ボールが下の画面の外にでたらゲームオーバーとする。そのためにまず、ボールに入力された画面の下端で反射するプログラミングを削除する。その代わりに、

```
if (_y > 375 + 7.5) {
        _parent.gotoAndPlay(18);
}
```

を入力する。このプログラミングにより、ボールが下の画面の外に出たときにトゥイーン１８に移動するようになる。あとはトゥイーン１８の位置にゲームオーバー画面を用意しておけばいい。できなければゲームオーバー画面のトゥイーンに合わせた数義を入力すればよい。ゲームオーバー画面は後ほど作る。

⑥クリア判定

続いてクリア判定をつくる。クリア条件はブロックがすべて消えることである。そこで各ブロックに関連付けられた flg の値を利用する。ブロックにボールが当たると flg の値が１になる。よって flg0～24 の値がすべて１になればクリアである。つまり、onClipEvent (enterFrame)の　｛　｝の中に、

```
if(flg0==1&&flg1==1&&flg2==1&&flg3==1&&flg4==1&&flg5==1&&flg6==1&&
flg7==1&&flg8==1&&flg9==1&&flg10==1&&flg11==1&&flg12==1&&flg13==1&&
```

```
flg14==1&&flg15==1&&flg16==1&&flg17==1&&flg18==1&&flg19==1&&flg20==1&&
        flg21==1&&flg22==1&&flg23==1&&flg24==1)
        {_parent.gotoAndPlay(40)}
```

を入力する。こうすれば、ブロックをすべて消したらトゥイーン４０へ移動する。あとはトゥイーン４０の部分にクリア画面を用意すればよい。できなければクリア画面のトゥイーンに合わせた数義を入力すればよい。クリア画面は後ほど作る。

⑦ゲームオーバー画面・クリア画面の作成

ゲームオーバー画面とクリア画面を作成する。今回はシンプルなものにした。

図Ⅲ－９　ゲームオーバー画面(著者作成)

図Ⅲ－１０　クリア画面(著者作成)

ゲームオーバー・クリア画面が完成したらゲームのゲームオーバー・クリア判定の際のトゥイーン移動とうまく対応できるようにする。それぞれの画面に NEXT の文字のボタンを作り、著作権画面に移動できるようにする。ムービーの停止をかけて、ここで動きを止め、ボタンによる移動のみを可能とさせる。

⑧　著作権情報画面の作成

著作権情報画面を作成する。ゲームオーバー・クリア画面の NEXT から移動できるようにする。

ここに自身がこのゲームを作ったことを含む著作権情報を書く。音楽は『甘茶の音楽工房』のフリー素材を利用した。詳しくは後ほど説明する。それぞれの文字にエフェクトをつけて工夫する。RESTAT の文字のボタンを作成し、これを押すとタイトル画面に移動できるようにする。ムービーの停止をかけて、ここで動きを止め、ボタンによる移動のみを可能とさせる。

図Ⅲ－１１　著作権情報(著者作成)

⑨　タイトル画面の作成

タイトル画面を作成する。

START を押すとステージセレクト画面へ移る。

操作方法の説明も書く。右矢印と左矢印による移動を簡潔にわかりやすく書く。また、エフェクトを使い躍動感を与えるなどをして工夫する。ムービーの停止をかけて、ここで動きを止め、ボタンによる移動のみを可能とさせる。

図Ⅲ－１２　タイトル(著者作成)

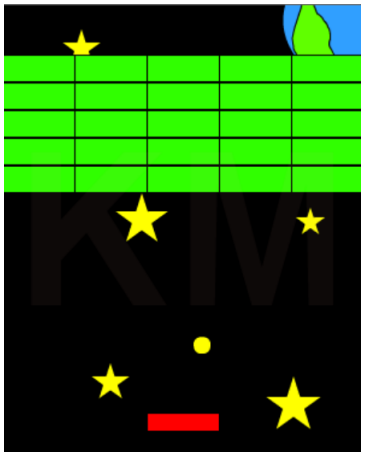
⑩　ステージセレクト画面

次にステージセレクト画面を作る。STAGE1～４までの文字にボタンの機能をつける。それぞれのボタンを押したらそれぞれのステージに移動できるようにする。STAGE1 はすでに完成している。これからは２～４を作っていく。

図Ⅲ－１３　ステージセレクト画面(著者作成)

⑪　メイン画面のデザイン

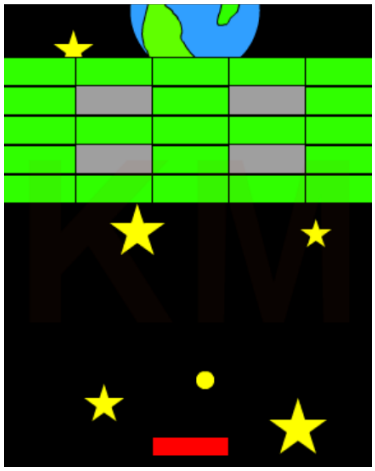
STAGE２を作る前に、STAGE1 のデザインを作る。地球をイメージしたものをつくり、ムービークリップに変換し、回転させる。また複数の星が左右に移動するようにし、宇宙をイメージした背景にする。そのため、背景色は黒である。この画面以外でも背景は一貫して黒である。また他のステージもこの背景を使い、地球の位置を変える程度にする。プログラミングになれたら、様々な背景を作ってみるといいかみしれない。



図Ⅲ―１４　STAGE１(著者作成)

⑫　STAGE２作成

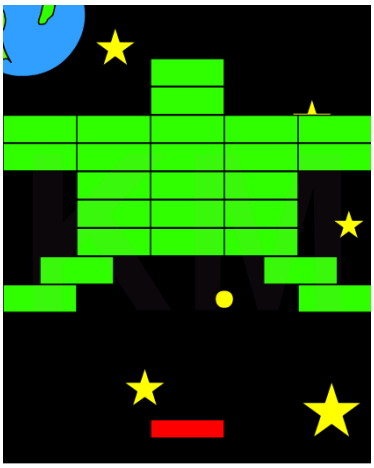
STAGE１を複製し、変化をつけて STAGE２とする。同じ名前の図形は使えないのでブロックの番号を２５～としてプログラムも編集する。そのうち４つを障害物ブロックとする。プログラムから、「ブロックが見えなくなるプログラム」と flg の値を削除する。こうすれば反射だけして消えない障害物の完成である。色を変えて普通のブロックと差別化する。クリア条件からも flg の値の該当部分を削除する。背景の地球の位置などを変えれば STAGE２の完成である。



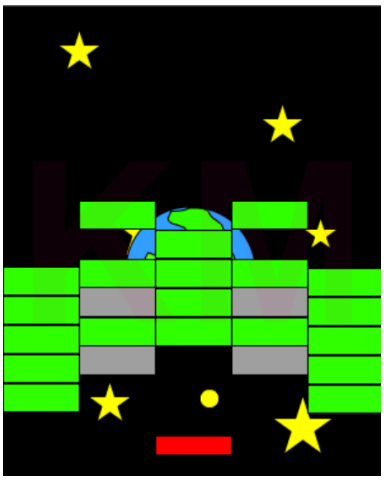
図Ⅲ―１５　STAGE２(著者作成)

⑬　STAGE３、４作成

同じく、STAGE１を複製して STAGE３を作る。ブロックの名前の番号を５０～にする。ブロックの形を星型にする。背景にも変化をつける。STAGE４は障害物ブロックを使おうと考えたため、STAGE２を複製する。ブロックの名前の番号を７４～にする。難易度を挙げようと考えたところ、ブロックとバーの間を狭めようと考えた。結果として、極端に難易度が上がってしまった。背景を変えて、ほかのステージとの差異を作る。



図Ⅲ－１６　STAGR３(著者作成)



図Ⅲ－１７　STAGE４(著者作成)

⑭音楽の挿入

⑬までですべてのステージの作成が終わったので、この時点でもうゲームとして遊べる。しかし、これだけでは盛り上がりに欠けるので音楽をつける。スコアタイムラインを表示し、フレームに音楽の再生のアクションをかける。音楽はあらかじめ用意しておく。今回は『甘茶の音楽工房』(http://amachamusic.chagasi.com/)様からおかりした。音楽をかける際、二重に音楽がかかるのを防ぐために、フレームが変わるときに音楽の停止のアクションをか

けるようにする。これでゲームは完成である。

⑮　保存・アップロード

⑭まででゲーム作成の工程は終了した。あとは保存し、アップロードするだけである。保存しないとすべての苦労が水の泡である。

「ファイル」から「ムービーのエクスポート」を選び、さらにそこから「SWF(Flash ムービー)および HTML ページとして」を選ぶ。そして、SWF ファイルと HTML ファイルを作成する(ファイル名は同じだとわかりやすい)。作成が終わったら。自分のホームページなどにアップロードしてすべての工程は終了だ。

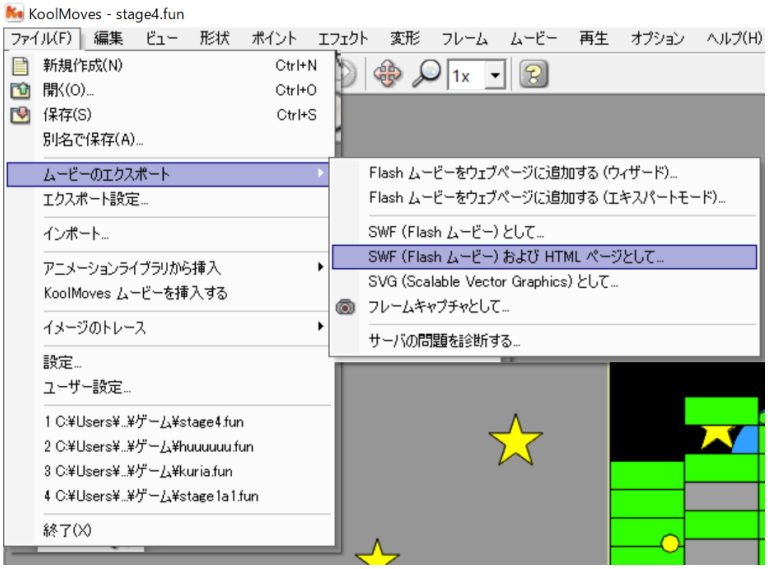


図　Ⅲ－１８　保存(著者作成)

Ⅳ　改善点

今回のプログラミングで、すべてのブロックの反射の動きを一括できるようなプログラミングをいくつか試したが、どれもうまくいかなかった。また、ブロックの反射のプログラミングをブロック内に書こうと試みたが、反射することはなかった。そのため、ボール内に大量のプログラミングを書くことになった。結果、おそらくプログラミングの多さの影響でブロックの量に制限が出てしまった(ブロック２７個目以降のプログラミングが保存後に消えるようになった)。この失敗を改善できるようにしたい。

おわりに

このレポートでは、HTML を中心に CSS と JavaScript を利用したホームページ作りの解説と Excel マクロプログラミングの解説についてまとめ、報告したものであるである。第１章(Ⅰ)では Web ページについてまとめた。第２章(Ⅱ)では自作ホームページの作成、主にオリジナリティーを出したところについての解説を行い、それに伴い HTML・CSS・JavaScript についての解説もした。第３章(Ⅲ)では、Excel マクロプログラミングについて概要と回転する３D オブジェクトの解説をした。

全体として、自作ホームページと Excel マクロプログラミングの解説となった。自分のホームページを作り、Excel マクロプログラミングを行ったことにより、PC を扱う上での基礎の情報・技術を学ぶことができた。また、自作のものを作るということは他者とは違うオリジナルのものを作成しなくてはならなく、そのためにはソース中のタグや数値の模索、試行錯誤する必要となる。この難しさを理解することができた。これからも新たな技術を学んでいきたい。

ゲーム作成について書いてきた。１章ではゲーム作成の概要を説明した。２章では準備するものを紹介した。３章では作成の工程を記した。

ゲームを作成してみて、プログラミングの難しさをしった。というのも、プログラミングは一文字のミスも許されず大文字小文字が異なるだけで正常に作動しないからだ。また、自分が作ったような単調なゲームでも細かくかつ多くのプログラミングが必要であった。なので、現在世に出ているゲームを作るのにどれほど労力がかかるのか少しわかった気がした。また、そのようなゲームがどのようなプログラムを持ち、それによってどのような動きを見せるのかが知りたくなった。

来年度以降は本格的な研究が始まる。最近はゲーム技術もしかり、様々な技術が発達し、我々の生活を豊かにしている。これらの技術を使えば衰退しつつある産業も復興できるのではないかと考える。例えば現在減少している銭湯の中には、プロジェクションマッピングを使って集客しているところもある。このように情報や技術は様々な産業で使え、衰退産業にも何か使えるのではないかと思う。私は情報や技術の産業利用について研究したい。どの産業に絞るかあるいはテーマを変えるかは春季休暇中に考え、来年度に向けて資料を集めていきたい。

参考文献

１部
千貫りこ(２０１２)『これからはじめる HTML&CSS の本』　技術評論社
色見本　「日本の伝統色　和色大辞典」www.colordic.org　2017.7.27 確認
koiki　「リストメニュー横並び」yume.hacca.jp/koiki/css/listoyoko.htm　2017.7.27 確認
koiki　「見出し」yume.hacca.jp/koiki/css/midashi/midashi.htm　2017.7.27 確認

２部
旭ゼミ　「Flash プログラミング：Top」
http://www2.toyo.ac.jp/~asahi/education/soron/shiryo/application/picture/flash/index.html　2018.1.13 確認

hakuhin　「Flash ゲーム講座&アクションスクリプトサンプル集」
http://hakuhin.jp/as/block.html　2018.1.13 確認

甘茶　「甘茶の音楽工房」http://amachamusic.chagasi.com/index.html
2018.1.13 確認