

## Part3. カウンタを作る

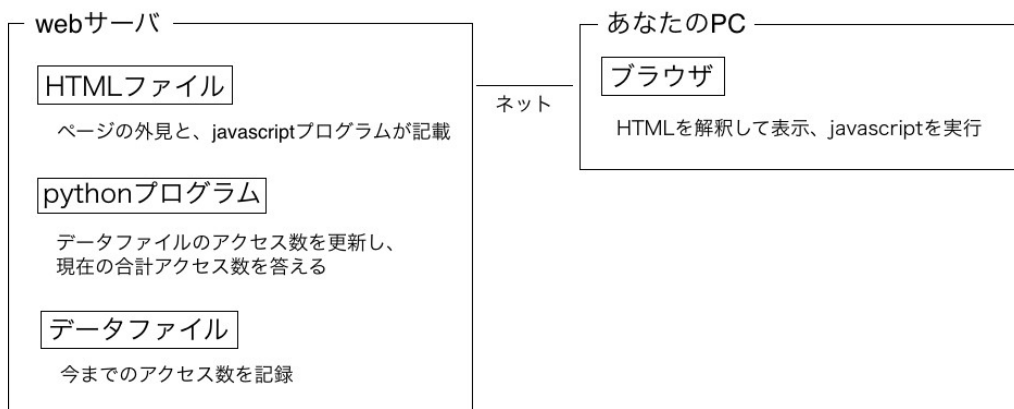
Python を使った web アプリの最も簡単なものとしてカウンタを作る。

1. 仕組み
2. ベースとなる python プログラム
3. カウンタのページとなる HTML
4. カウント数を表示する javascript
5. python からの javascript 呼び出し
6. javascript からの python 呼び出し
7. サーバへのアップロード

### 3-1. 仕組み

ここからはwebプログラミングを扱う。googleやtwitterなどのwebサービスたちが、どのような仕組みで動いているのかがなんとなくつかめることを目標としよう。

まずは簡単なwebサービスとして、アクセスカウンタ（そのページに何人がアクセスしたかをカウントする。「あなたは12345番目のお客様です」ってやつ）を作っていこう。



インターネットを介して世界に公開されるwebページは、webサーバと呼ばれるコンピュータ上に置かれる。webサーバでは特別なプログラムが走っていて、ユーザーからのアクセスを処理して適切なデータを返したり、プログラムを実行したりする。自分のPCをwebサーバにすることもできるが、今回の講習では企業が運営するレンタルサーバを借りてそれを使う。

カウンタのプログラムでは、次のファイルをwebサーバ上に用意する。

- (1) HTML ファイル：カウンタページのの外見を指定する。また、ページが読み込まれた時に python プログラムを実行するような javascript が書かれている。
- (2) python プログラム：データファイルに記録されている合計アクセス数に1を足して、現在の合計アクセス数をブラウザに表示させるような javascript プログラムを実行する。
- (3) データファイル：合計アクセス数を表す整数一つだけがかかれている。

次の手順でカウンタは動作する。

- (1) あなたがブラウザにアドレスを打ち込むと、ブラウザはそのアドレスが表示している web サーバに接続して、カウンタのページの HTML ファイルを要求する。
- (2) Web サーバからあなたの PC に対して HTML ファイルが転送される。
- (3) ブラウザは HTML を解読して画面上に表示する。HTML ファイルに書かれた javascript は**あなたの PC 上で**実行される。カウンタでは、web サーバ上にある python プログラムを実行するように指示されており、web サーバに python プログラムを実行するよう要求が送られる。
- (4) python プログラムが**web サーバ上で**実行される。web サーバ上におかれたデータファイルが更新され、HTML 上のアクセス数の部分を、現在のアクセス数の値に更新するような javascript プログラムがあなたの PC へと送られる。
- (5) ブラウザはその javascript を実行し、画面上に現在のアクセス数を表示する。

javascript があなたの PC で実行されるのに対して、python プログラムは web サーバで実行される。別に言語は python でなくともよく、web サーバ上で実行されるプログラム全般を CGI をいう。Javascript と CGI には次のような役割分担がある。

javascript：マウス操作などのユーザーの動きの追跡や、画面表示の更新などを行う。

CGI：web サーバに置かれたデータを読み込んだり、編集したりする。カウンタでは総アクセス数が記録されたデータファイルを扱う。

### 3-2. ベースとなる python プログラム

#### (1) py ファイルの作成

notepad++で次のプログラムを作り、自分の名前のフォルダに「counter\_base.py」という名前で保存しよう（左端の数字は行番号なので入力しない）。

```
1 f = open('counter.txt', 'r')
2
3 a = 0
4 for ln in f:
5     a = int(ln)
6 f.close()
7
8 print a
9
```

これは counter.txt というファイルに書かれている数字を読んで a という変数に代入し、それを表示する。

#### (2) データファイルの作成

次に、「counter.txt」というファイルを作って「100」とだけ書いて保存しよう。

#### (3) コマンドプロンプトからの実行と終了

自分の名前のフォルダに移動して（移動の方法は指示します）、

python counter\_base.py

と入力しよう。これは昨日までの ipython notebook で shift+Enter を押したのと同じように python プログラムの実行をする命令である。100 と表示されたら成功である。

#### (4) 課題

counter\_base.py を改良して、counter.txt に a+1 を上書きするようにせよ。つまり、counter.txt が 100 だったら、conter\_base.py を実行したら 101 になればいい。

### 3-3. カウンタのページとなる HTML

#### (1) HTML とは

HTML は web ページを記述するために特化したプログラミング言語である。HTML では、何の装飾もない平文に「この文字はこの位置に」「この部分はこの色に」「ここに画像を挿入」といった指示をを与えて、美しく表示する。

このような指示の範囲は「タグ」で指定される。タグとは、

```
<div> ... </div>
```

のように、<>で始まって</>で終わる領域のことである。

今回の講習のテーマはあくまで python なので、あまり詳しくは解説しない。

昔の講習で扱ったことがあるのでそのテキストがある

<http://mattya1089.sakura.ne.jp/2011html/> : サンプル

<http://mattya1089.sakura.ne.jp/2011html/textbook.pdf> : テキスト

外部教材：とほほの WWW 入門

<http://www.tohoho-web.com/www.htm>

HTML リファレンスやスタイルシート一覧などがあってとても便利

#### (2) 次のファイルを作って、「counter.html」で保存しよう。

```
1 <html>
2   <body>
3     合計来場者数: <span id="cnt">0</span> 人
4   </body>
5 </html>
6
```

できたらブラウザで開いてみよう。

合計来場者数：0 人

と表示されるだろうか。

意味を説明しよう。<html>は、html ファイルの範囲を表すタグで、最初と最後をくくっている。<body>は、実際に表示される部分を表すタグである。3 行目の <span>タグは、このままでは特に意味はなく、ふつうに中身の 0 がそこに表示される。<span id=" cnt" >とすることで、この文章の「0」のところに cnt という名前を付けたことになる。カウンタではもちろんこの場所の値をアクセスする度に書きかえたい。そのために「cnt という名前の場所を書き換えろ」という命令を後ほど使うことになる。

### 3-4. カウント数を表示する javascript

#### (1) javascript とは

javascript は、web ページの内容を動的に書きかえるためのプログラミング言語である。たとえば、マウスをある領域にもっていったら色が変わったり、メニューのうえに乗せたら中身が開いたり、…といったよくある演出を行うことができる。これも今回はあまり詳しく解説しないので、前のページに載せたテキストを見てほしい。

#### (2) counter.html を次のように改造しよう。

```
1 <html>
2   <head>
3     <script type="text/javascript">
4       function update(cnt){
5         var span = document.getElementById("cnt")
6         span.innerHTML = cnt
7       }
8     </script>
9   </head>
10  <body onload="update(10)">
11    合計来場者数: <span id="cnt">0</span> 人
12  </body>
13 </html>
14
```

こんどは開くと、合計来場者数：10 人と表示されるはずである。解説しよう。まず、3 行目から 8 行目までの<script>タグの領域が、javascript である。このエリアでは、update という名前の命令を作っている。この命令は引数 cnt をとり、

変数 span を用意し、id が cnt となっている場所を指すようにする(5 行目)  
(var は変数を作るの意味で、document.getElementById が  
引数の id 名の場所を得るという命令)

span が指す場所の HTML 文を、変数 cnt の中身にかきかえる(6 行目)  
という処理を行う。id が cnt となっている場所の中身は最初 0 で、update が実行  
されたら変数 cnt に代入されている 10 に書き換わるというわけである。  
update 命令は、11 行目で呼び出しが指定されている。onload=" update(10)" と  
書くことで、この body タグの領域がブラウザに読まれたら update 命令を引数 10  
で実行することになる。

#### ・課題

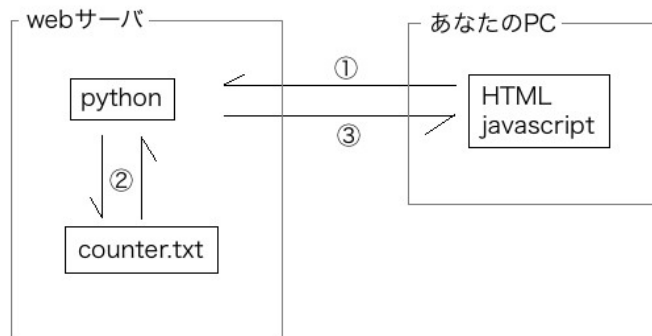
onload=" update(10)" を onclick=" update(10)" に書き換えてみよう。  
画面をクリックしたら数字が書き換わるようになる。

### 3-5. python からの javascript 呼び出し

さて、前のページで作った update 命令を使って、合計来場者数の数の部分を書きかえられるようになった。この update 命令の引数として、現在のアクセス数を渡してやればいいわけだ。

(1) どうやればいいの？

このことを実現するためには、さっき作ったような python プログラムから、javascript に対してアクセス数のデータを送ってやらないといけない。図にすると次のような手順である。



- ① javascript から python プログラムを呼び出す。
- ② python プログラムが実行され、counter.txt に書かれた総アクセス数を読み込み、counter.txt を、1 加算した値に書きかえる。
- ③ 総アクセス数を javascript に返し、さっき作った update 命令を実行する

(2) まずはこの②、③の部分を作る。

さっきの counter\_base.py をコピーして、「counter.cgi」とし、次のプログラムを書こう。4 行目から 13 行目まではさっきと同じ内容である。

```

1  #!/usr/local/bin/python
2  # -*- coding: utf-8 -*-
3
4  f = open('counter.txt', 'r')
5
6  a = 0
7  for ln in f:
8      a = int(ln)
9  f.close()
10
11 f = open('counter.txt', 'w')
12 f.write(str(a+1))
13 f.close()
14
15 print "Content-Type: text/javascript"
16 print ""
17 print "update(%d)" % a
18

```

解説しよう。

まず、.py でなく.cgi になったわけだが、このプログラムは web サーバによって実行されるので、web サーバに対してこれが実行するファイルであることを教えてあげないといけないのだが、今回使うサーバでは.cgi をつけることが決められているのである。

1 行目もサーバに対する指示で、「python で実行して下さい」程度の意味である。

2 行目は「このプログラムは utf-8 という文字コードで書かれています」という意味である。だから保存するときは utf-8 にして保存してほしい（やりかた分からなかったら聞いて）。

4 から 13 行目は、さっき作ってもらった、counter.txt を読み込んで 1 足した値を書き込むプログラムである。

15 から 17 行目が一番大切で、変数 a が 100 だったら

```
Content-Type: text/javascript  
update(100)
```

という内容を出力することになる。Content-Type... はこの出力が javascript として書かれていることを表す。次の update(100) は javascript として、この cgi を呼び出した javascript に送り返されて実行される。つまり、これをさっきの html から呼び出せば、総アクセス数を表示させることが可能になるのである！

このプログラムは web サーバ上でしか実行できない。だからうまく動くかどうかは web サーバに転送するまでわからない。次の次の章まで待ってほしい。



## 3-6. javascript からの python 呼び出し

前の前のページの図の①の部分完成させる。counter.html を次のように編集しよう。

```

1  <html>
2    <head>
3      <script type="text/javascript">
4        function load(){
5          var s = document.createElement('script')
6          s.src = 'counter.cgi'
7          document.body.appendChild(s)
8        }
9        function update(cnt){
10         var span = document.getElementById("cnt")
11         span.innerHTML = cnt
12       }
13     </script>
14   </head>
15   <body onload="load()">
16     合計来場者数: <span id="cnt">0</span> 人
17   </body>
18 </html>
19

```

4 から 7 行目の load 命令が追加され、15 行目の命令の呼び出しが onload=" load()"（body がブラウザにロードされたところで load 命令を呼び出す）となった。

load 命令は、

script 要素を生成して変数 s に代入 (5 行目)

s の src プロパティを " counter.cgi" に (6 行目)

body 領域の最後に s をくっつける (7 行目)

これは具体的には、</body> タグの手前に

<script src=' counter.cgi'></script>

というタグを挿入することに相当する。これをくっつけるとそのタイミングで counter.cgi が読み込まれ、web サーバ上で実行される。counter.cgi はさっき作ったファイルのことで、実行されると counter.txt の中身を読んで、1 を足したもので上書きして、update(カウント数) 命令を実行するような javascript 文を返してくる。これを受け取ったブラウザは、その javascript を実行し、合計来場者数の 0 が総カウント数に書き換えられる。

### 3-7. サーバへのアップロード

さくらのレンタルサーバ

<https://secure.sakura.ad.jp/rscontrol/>

にアクセス

ドメイン名 : mattya1089.sakura.ne.jp

パスワード : 6tshm5sah6

でログイン

左側のメニューの、「運用に便利なツール」という欄の、「ファイルマネージャ」にアクセス

まず、表示アドレスへの操作ボタン→フォルダ作成、で、自分の名前のついたフォルダを作る。ただし空白や日本語は入らないように英字のみで書こう

そのフォルダに入り、アップロードボタンを押して、

counter.html

counter.cgi

counter.txt

をアップロードしよう。

Counter.cgi を右クリックしてプロパティを選択し、3つある「実行」にチェックを入れる

counter.txt を右クリックしてプロパティを選択し、3つある「書込」にチェックを入れる

これはパーミッションの設定といって、cgi ファイルはサーバで実行しないといけないので実行可能にし、counter.txt はアクセスの度に書きかえないといけないので書き込み可能にする必要があるのである。

ブラウザから、

<http://mattya1089.sakura.ne.jp/>（今作った自分の名前のフォルダ名）/counter.html

にアクセスしてみよう！

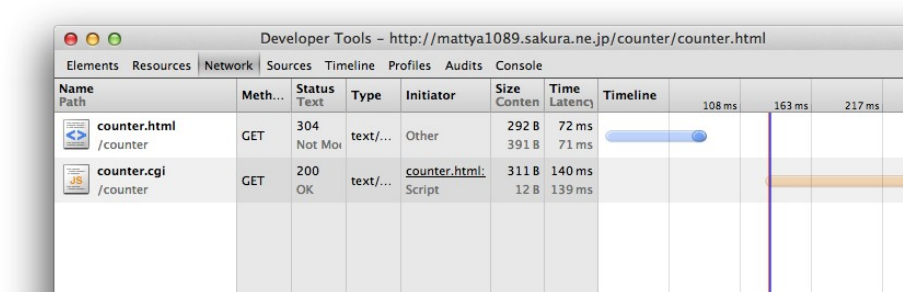
### ・ 課題

#### (1) 要素の検証を使ってみる

ブラウザ上で右クリックし、要素の検証を選択して、カウンタのページに再アクセスしよう。

次のような画面が開くだろうか？

合計来場者数: 109 人



開いたら Network のタブを選択して、`counter.cgi` を選択して、response を見てみよう。ここに `counter.cgi` が何を返したかが表示される。ここでは `update` (アクセス数) である。

この要素の検証機能はデバッグ（プログラムの間違いを直すこと）にとっても有効である。

#### (2) このカウンタは攻撃に弱い。更新ボタン (F5 キー) を押しっぱなしにするとカウンタがどんどんまわる。これを防ぐ方法を考えよ。

`javascript` を使えば現在時刻を取得したり、あるいはアクセス者の IP アドレスを知ったりできる。



## Part4. 検索システムを作る

Python を使った web アプリの応用として、検索システムを作る。

1. 超速習 HTML
2. 超速習 javascript
3. フォームのデータ取得
4. CGI に変数を渡す方法
5. フォームのデータを CGI に投げる
6. 検索システムを作る

### 4-1. 超速習 HTML

#### (1) タグの使い方

`<h1> ... </h1>`

見出しを表す部分で、囲われたところが強調される。

h2, h3, ... h6 まである。数字が大きいほど弱い。

`<br>`

改行する。閉じるタグはいらない。

`<div> ... </div>`

長方形の領域を表す。囲われた部分をその長方形内に収める。

`<span> ... </span>`

囲われた部分に何らかの装飾を施したりする。以下の例では style プロパティ（後述）を指定して太字にしたり下線を引いたり赤くしたりしている。

以下の例は全文書き出す必要はないよ。日本国憲法とかで Google 検索してその文章をコピーしてもいいかも。

```
1 <html>
2   <head>
3     <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
4   </head>
5   <body>
6     <h2>日本国憲法前文</h2>
7     <div style='width:600px; border:solid 1px;'>
8       <span style="font-weight: bold;">
        日本国民は、正当に選挙された国会における代表者を通じて行動し、われらとわれ
        らの子孫のために、諸国民との協和による成果と、わが国全土にわたつて自由の
        もたらす恵沢を確保し、政府の行為によつて再び戦争の惨禍が起こることの
        ないやうにすることを決意し、ここに<span style='text-decoration:
        underline;'>主権が国民に存することを宣言し、この憲法を確定する。</span><
        br>そもそも国政は、国民の厳粛な信託によるものであつて、その権威は国民に由来
        し、その権力は国民の代表者がこれを行使し、その福利は国民がこれを享受する。
        これは人類普遍の原理であり、この憲法はかかる原理に基くものである。<br><
        span style='color:red;'>
        われらは、これに反する一切の憲法、法令及び詔勅を排除する。</span>
9     </div>
10  </body>
11 </html>
12
```

### 日本国憲法前文

日本国民は、正当に選挙された国会における代表者を通じて行動し、われらとわれらの子孫のために、諸国民との協和による成果と、わが国全土にわたつて自由のもたらす恵沢を確保し、政府の行為によつて再び戦争の惨禍が起こることのないやうにすることを決意し、ここに主権が国民に存することを宣言し、この憲法を確定する。

そもそも国政は、国民の厳粛な信託によるものであつて、その権威は国民に由来し、その権力は国民の代表者がこれを行使し、その福利は国民がこれを享受する。これは人類普遍の原理であり、この憲法はかかる原理に基くものである。

われらは、これに反する一切の憲法、法令及び詔勅を排除する。

`<input type="..." value="..." >`

ボタンや入力欄を作る。type で何を作るかを指定し、value で中に書く文字を指定する。閉じるタグはいらない。

type=" text"            一行の入力欄

type=" password"      パスワード入力欄（文字が\*\*\*\*\*になるだけ）

type=" button"        ボタン

```
1 <html>
2   <head>
3     <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
4   </head>
5   <body>
6     <div>
7       ユーザー名:<input type="text" value="aaa"><br>
8       パスワード:<input type="password" value="pass"><br>
9       <input type="button" value="送信">
10    </div>
11  </body>
12 </html>
13
```

ユーザー名：	<input type="text" value="aaa"/>
パスワード：	<input type="password" value="...."/>
<input type="button" value="送信"/>	

### (2) プロパティ

プロパティというのは、

`<span style="..." >`

`<input type="..." value="..." >`

の style とか type とか value とかのことである。この例のタグ名に続けてスペース区切りで複数のプロパティを列挙できる。各プロパティは「プロパティ名=" プロパティ内容" 」という形式で記述する。コーテーションはダブル（" "）でもシングル（' '）でもいい。

重要なプロパティ

id=" aaa"

領域に名前 aaa をつける。

Javascript から名前呼び出せる。

onload=" func(引数)"

ブラウザに読み込まれた時に、

javascript の func 命令を呼び出す。

onclick=" func(引数)"

クリックされた時

style=" ..."

スタイル（後述）を指定する。

### (3) スタイル

スタイルというのは style プロパティで指定される文字列のことであり、そのタグに囲われた領域に何らかの装飾を施す。

```
<div style='width:200; height:400; border:solid 1px'>
```

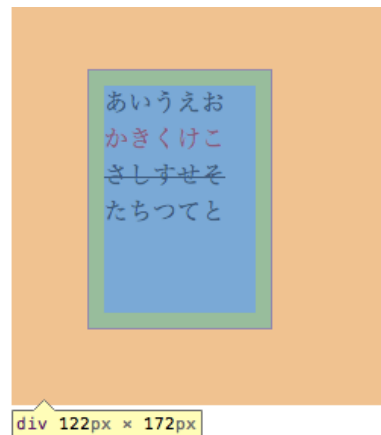
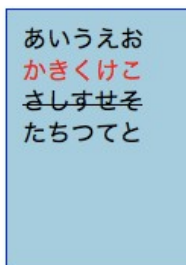
のように、複数のスタイルを ; で区切って列挙する。各スタイルは「スタイル名 : 内容」という形式で記述する。

#### 重要なスタイル

color:red	文字色。 色名は英語 (blue, green, gray など) で指定
background-color:blue	背景色
border:solid 1px blue	領域の枠線。 solid は実線で囲う 1px は太さ blue は色
width:100	領域の横幅。指定しなかったら内部にある要素に勝手に合わせる。
height:500	領域の縦幅
margin:5px	他の領域との間に空白を 5px つける
padding:5px	内部の要素との間に空白を 5px つける
font-weight:bold	太字にする
text-decoration:underline	下線。overline や line-through もあり。



```
1 <html>
2   <head>
3     <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
4   </head>
5   <body>
6     <div style="width:100; height:150; background-color:lightblue;
7       border:solid 1px blue; margin:50px; padding:10px">
8       あいうえお<br>
9       <span style="color:red">かきくけこ</span><br>
10      <span style="text-decoration:line-through">さしすせそ</span><br>
11      たちつてと
12    </div>
13  </body>
14 </html>
15
```



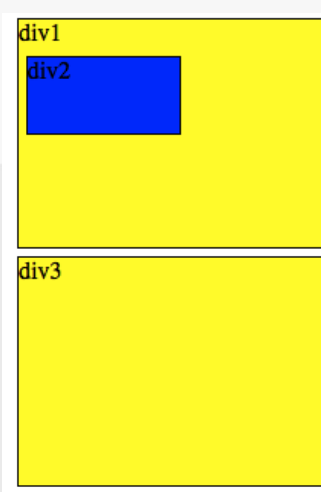
右はブラウザの要素を検証機能で見たもの。緑の領域がpaddingでオレンジの領域がmargin。

## (3) スタイルシート

```

1  <html>
2    <head>
3      <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
4      <style>
5        div {
6          border: solid 1px black;
7          margin: 5px;
8          width: 200;
9          height: 150;
10         background-color: yellow;
11       }
12       #aaa {
13         background-color: blue;
14         width: 100;
15         height: 50;
16       }
17     </style>
18   </head>
19   <body>
20     <div>
21       div1
22       <div id="aaa">
23         div2
24       </div>
25     </div>
26     <div>
27       div3
28     </div>
29   </body>
30 </html>
31

```



<style>タグを使うと、「全てのdivタグに同じスタイルを適用する」「あるidのタグのスタイルを決める」といったことができる。スタイルタグはheadタグの中に書くと良い。

上はその例で、5から11行目ですべてのdivタグに対するスタイルを決めている。

```

タグ名{
    スタイル
}

```

スタイルの部分は、style="..."の部分に書くのと同じ形式で書く。複数のスタイル間は改行で区切れるので見やすくなる。

12から16行目ではid名がaaaの部分のスタイルを定めている。

```

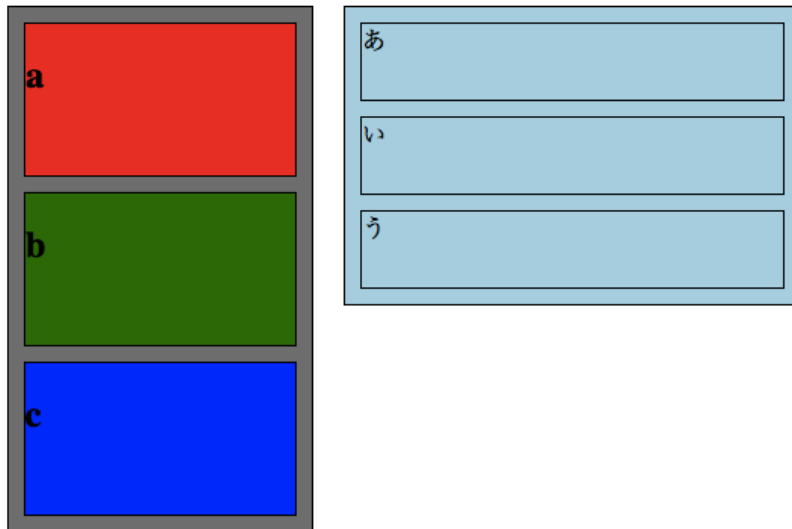
#id名{
    スタイル
}

```

id名がaaaなのは22行目のdivタグである。タグ名による指定よりも優先されるので、divタグだけれど5行目で決めた他のdivとは違うスタイルが適用される。

## (4) 練習

div タグを入れ子にして次のように表示される HTML ファイルを作れ。



ただし、div タグは並べて書くと縦方向に並んでいくので、普通に書こうとするとグレーの div と水色の div が縦に並んでしまうと思う。ここで、グレーの div タグのスタイルに

```
float:left;
```

を追加すると横に並ぶようになる。このスタイルは、次に来る領域を右に並べろという意味である（通常は下に並ぶ）。

たいていの web ページは div を使ってレイアウトを決めている。右は Yahoo のページに使われている div 領域の一部を赤枠で囲ったものである（実際はもっと細かく div で区切られている）。ブラウザの要素を検証機能を使って、どこが div かを調べることができる。



### 4-2. 超速習 javascript

javascript はひとつの立派なプログラミング言語なのだが、この講習では難しい仕事は python にやってもらうので、必要最低限のことだけを教えます。

(0) javascript は、HTML ファイルの中の<head>タグの内側に<script>タグをつくって、その間に書く。

```
1 <html>
2   <head>
3     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
4     <script type="text/javascript">
5
6     </script>
7   </head>
8   <body>
9
10  </body>
11 </html>
12
```

(1) HTML からの命令呼び出し

javascript で命令を作るには、

```
function 命令名(引数たち){
    ...
}
```

のように書く。HTML 側で、

<body onload=" 命令名(引数)" >と書くと、ページを開いた時にその命令が実行される。

<div onclick=" 命令名(引数)" >と書くと、その div の内側がクリックされた時に命令が実行される。下の例で確かめてみよう。alert は警告を表示する命令。

```
1 <html>
2   <head>
3     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
4     <script type="text/javascript">
5       function loaded(){
6         alert("loaded!!")
7       }
8       function clicked(){
9         alert("clicked!!")
10      }
11    </script>
12  </head>
13  <body onload="loaded()">
14    <div onclick="clicked()">
15      あああああ
16    </div>
17  </body>
18 </html>
19
```

(2) document.getElementById と innerHTML

var a = document.getElementById(“id名”)と書くことで、

<div id=“ id名” >

...

</div>

の領域が変数 a に代入される。var は変数を作る構文である。A に対しては次のような操作ができる。

・ 中身の書き換え

a.innerHTML = “HTML 文”

<div id=“ id名” >...</div>の...の部分を” HTML 文” に書きかえる。

```
1 <html>
2   <head>
3     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
4     <script type="text/javascript">
5       function clicked(){
6         var a = document.getElementById("div1")
7         a.innerHTML = "iiiiiiii"
8       }
9     </script>
10  </head>
11  <body>
12    <div id="div1" onclick="clicked()">
13      ああああ
14    </div>
15  </body>
16 </html>
17
```

“あああああ” をクリックしてみよ。

・ プロパティの取得

var b = a. プロパティ名

変数 b にプロパティの内容を代入する。たとえば a.value なら、

<input id=“ id名” value=“ 内容” >

の内容の部分が b に代入される。これは主に入力欄に入力された内容を取得するのに用いられる。

a. プロパティ名 = “プロパティ内容”

こうすると逆にプロパティの書き換えができる。

ただし、style プロパティをいじるときは、

a.style.(スタイル名) = 内容

とする（次の例参照）。

クリックされたところのスタイルを変更する。

```

1  <html>
2    <head>
3      <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
4      <script>
5        function click1(){
6          var d = document.getElementById("div1");
7          d.style.color = "red";
8        }
9        function click2(){
10         var d = document.getElementById("div2");
11         d.style.display = "none";
12       }
13     </script>
14   </head>
15   <body>
16     <div>
17       <div id="div1" onclick="click1()">div1</div>
18       <div id="div2" onclick="click2()">div2</div>
19     </div>
20   </body>
21 </html>
22

```

(4) createElement と appendChild

```

var s = document.createElement("script")
s.src = 
document.body.appendChild(s)

```

<body>タグの直下に、  
 <script src=" 空欄部分" ></script>  
 というタグを書き加える。

## 4-3. フォームからのデータ取得

次の HTML を書いて「tashi\_base.html」と名付けて保存せよ。

```

1  <html>
2    <head>
3      <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
4      <script type="text/javascript">
5        function pressed(){
6          var a = document.getElementById("form1")
7          var b = document.getElementById("form2")
8          var c = document.getElementById("form3")
9          c.value = Number(a.value) + Number(b.value)
10         }
11      </script>
12    </head>
13    <body>
14      <input id="form1" type="text">
15      +
16      <input id="form2" type="text">
17      <input id="btn" onclick="pressed()" type="button" value="足す">
18      <input id="form3" type="text">
19    </body>
20  </html>
21

```

これは「足す」ボタンが押された時、入力欄 form1, form2 に書かれた値を取得し、それを足し算して form3 に表示する javascript である。

6, 7, 8 行目で変数 a, b, c に form1, form2, form3 を表すオブジェクトを代入している。ここでのオブジェクトというのは、その id 名を持ったタグで囲われている領域を表す。このとき

a. プロパティ名

で a の領域のプロパティを取得できる。さらに、そのプロパティに対して代入を行うことで、HTML を変更できる。

9 行目では form3 の value プロパティに対し、足し算の結果を代入することで、form3 の入力欄に答えが表示されるようになる。

ちなみに Number 命令は javascript での文字列を数に変換する命令である。

右のように、上の3つのフォームに数を入力して解くボタンを押すと、下にその二次方程式の解が表示されるようなプログラムを作れ。

ちなみに javascript でルート x は、

Math.sqrt(x)

と書く。

x<sup>2</sup> +  x +  = 0  
  
 x = ,

## 4-4. CGI に変数を渡す方法

つぎのプログラムを作成し「query.cgi」と名付けてサーバーにアップロードせよ。

```

1  #!/usr/local/bin/python
2  # -*- coding: utf-8 -*-
3
4  import os
5  import cgi
6
7  if 'QUERY_STRING' in os.environ:
8      query = cgi.parse_qs(os.environ['QUERY_STRING'])
9  else:
10     query = {}
11
12     print "Content-Type: text/plain"
13     print ""
14     print "%s" % query.items()
15

```

ブラウザから

.....query.cgi?name=aaa&a=3&b=100

にアクセスすべし。このように、.cgi に続いて?マークを書き、引数 name, a, b に代入するものを&記号で区切って指定する。結果は次のようになるはず。

```
[('a', ['3']), ('b', ['100']), ('name', ['aaa'])]
```

まず、このプログラムでは7から10行目で query という辞書（Part2 参照）に、アドレスバーで与えられた引数たちを {引数名:内容, ...} の形で記憶する。これは4, 5行目で読み込んでいる os と cgi ライブラリの機能である。

14行目では、query 辞書に含まれている引数名とその内容との組をタプルのリストの形で出力している。そのリストを得るのが query.items() 命令なのである。

今回のケースでは、

| 引数名  | 内容      |
|------|---------|
| a    | ['3']   |
| b    | ['100'] |
| name | ['aaa'] |

となっている。内容が全て要素数1の文字列のリストになっていることに注意。なんでリストなのかというと、

```
query.cgi?a=3, 1, 4, 1
```

のようにしてリストを引数に渡すことができるからである。また、文字列のリストなので、数字を数として使うには int 命令で変換する必要がある。

引数を違うものにして、表示がどう変わるか確かめてみよ。



### 4-5. フォームのデータを CGI に投げる

さきほどの足し算 HTML は、足し算の計算を javascript で行なっていた。これを CGI (python) にやらせてみよう。

#### 方針

数の入力と結果を表示するための tashi.html と、足し算を行う tashi.cgi の二つを用意する。

tashi.html には、javascript で二つの命令が書かれていて、pressed() 命令ではボタンが押された時にフォームのデータを cgi に投げ、update(result) 命令では、引数 result を受け取って、それを答えのフォームに表示する処理をする。

足すボタンが押されたら pressed 命令が実行されて、フォームのデータを javascript で取り出し、たとえばそのデータが 15 と 20 だったら、

tashi.cgi?a=15&b=20

で tashi.cgi を呼び出す。すると tashi は引数 {a:15, b:20} を受け取るので、a と b を足し算して print 文で、” update(足した結果)” を出力する。この出力をブラウザが受け取り、update 命令が実行されて、答えが三番目のフォームに表示される。

#### (1) tashi.html

```
function pressed(){
    var a = document.getElementById("form1")
    var b = document.getElementById("form2")
    var s = document.createElement("script")
    s.src = "tashi.cgi?" + "a=" + a.value + "&b=" + b.value
    document.body.appendChild(s)
}
function update(result){
    document.getElementById("form3").value = result
}
```

javascript の部分だけ抜粋している。残りは 4-3 のと同じである。pressed 命令が実行されると、a.value が 15 で b.value が 20 なら、

<script src=" tashi.cgi?a=15&b=20" ></script>

というタグが、body 要素の最後にくっつけられる。くっつけられた瞬間にブラウザが tashi.cgi?a=15&b=20 にアクセスし、cgi に引数が渡される。

(2) tashi.cgi

2 ページ前のプログラムを思い出そう。このプログラムの 10 行目までで、query という名前の辞書に引数たちが、引数名：文字列のリストの形で渡されるのだった。辞書ということは、

`q['a']`

と書くと、その引数 a の中身にアクセスできるということである。さらに、この内容は文字列のリストなので、

`int(q['a'][0])`

と書くことで、そのリストの一つ目 `q['a'][0]` を整数に変換して得ることができる。

```
1  #!/usr/local/bin/python
2  # -*- coding: utf-8 -*-
3
4  import os
5  import cgi
6
7  if 'QUERY_STRING' in os.environ:
8      query = cgi.parse_qs(os.environ['QUERY_STRING'])
9  else:
10     query = {}
11
12  a = int(query['a'][0])
13  b = int(query['b'][0])
14
15  print "Content-Type: text/javascript"
16  print ""
17  print "update(%d)" % (a+b)
18
```

最後の 3 行で文字列

`Content-Type: text/javascript`

`update(a+b の計算結果)`

が出力され、これを解釈したブラウザは、javascript の `update(a+b の計算結果)` 命令を実行する。この命令は `tashi.html` に書かれていたもので、三番目のフォームの中身が計算結果へと更新されるのである。

この二つのファイルを作成してサーバーにアップロードし、うまくいくか確認せよ。

以上をまとめる。

- ・ javascript から cgi を呼び出してデータを送る方法

```
var s = document.createElement("script")
s.src = 
document.body.appendChild(s)
```

空欄の部分には (cgi 名).cgi?引数 1=\*\*\*&引数 2=&&& ... を書く

- ・ cgi でデータを受け取る方法

```
import os
import cgi

if 'QUERY_STRING' in os.environ:
    query = cgi.parse_qs(os.environ['QUERY_STRING'])
else:
    query = {}
```

query は辞書になっていて、  
query[' 引数名 '][0] でそれぞれの引数を得る

- ・ cgi から javascript にデータを送る (javascript の命令を実行する) 方法

```
print "Content-Type: text/javascript"
print ""
print 
```

空欄の部分には実行したい javascript 命令を書く。  
送りたいデータはその命令の引数として渡す。

- ・ 注意点

- (1) cgi をサーバにアップロードしたらプロパティの設定 (パーミッション設定) の実行にチェックを入れるのを忘れないこと。
- (2) テキストファイルなど、書き込みを行うファイルをアップロードしたら、プロパティの書き込みにチェックを入れるのを忘れないこと。
- (3) cgi の文字コードは utf-8 で保存すること。notepad++では上のメニューの Encoding から convert to utf-8 とかいうのをやる。
- (4) うまく動いていなかったらブラウザの要素の検証機能を使って見ること。右下にエラーマークが出ていたら押して中身を見ること。Cgi から返されたデータを調べたりも出来る。
- (5) ファイルマネージャからファイルを編集することもできる。このとき手元の PC のデータはもちろん変更されないので気をつけるべし。

### 4-6. 検索システムを作る

Google などの検索システムがどうやって動いているのかを考える。

ここでは、フォームに英単語を入力して検索ボタンを押したら、不思議の国のアリス（英文）の中に、その単語が何回出てくるかカウントする web プログラムを作ってみよう。

不思議の国のアリスに Alice が出てくる回数は 221 回でした。

次の順番で作っていくとやりやすいかも。

#### (1) python

まずは cgi ではなく、ipython notebook 上の python プログラムとして検索システムを作ろう。

<http://matty1089.sakura.ne.jp/cgi/form/alice.txt>

に不思議の国のアリスの全文（Project Gutenberg という著作権切れの文学を公開しているサイトから持ってきた）があるので、これをダウンロードして、alice.txt を読み込んで、ある単語が何回含まれているかをカウントする python プログラムを作る。英文から英単語を取り出すには、スペース記号で split をかければ良い。余裕があったらカンマやピリオドなどを取り除くと良い。

#### (2) html

上の画像のように、入力欄、検索ボタン、結果表示欄を HTML で作ろう。

#### (3) javascript

その HTML に命令 search と update を追加しよう。

search() は検索ボタンが押された時に実行される命令で、入力欄に書かれた単語を読み込んで、

search.cgi?s=単語

のような感じで cgi を呼び出し命令である。

update(result) は cgi から呼び出される命令で、引数 result をとって、それを結果表示欄に表示する命令である。

#### (4) cgi

最初の (1) で作った python ファイルを cgi にする。引数として英単語を受け取って、検索して、結果を update(result) を呼び出す形で出力する。

## Part5. チャットを作る

twitter みたいなをつくる。

1. しくみ
2. チャットの原型の原型
3. チャットの原型
4. 改良1
5. 改良2
6. 改良3

## K 会春季講習 2013 情報講座

### 5-1. しくみ

前で解説します

## 5-2. チャットの原型の原型

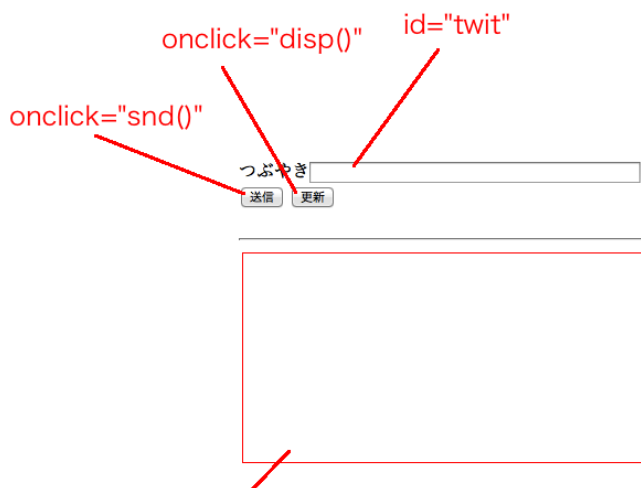
### (1) HTML ファイル(twtr.html)

Part4 のHTML のところを参考にしながら、空欄を埋めて次の画面を生成する HTML ファイルを作ろう。赤英字で書かれているのはプロパティで、その部分を表示するタグに、そのプロパティを指定すべし。たとえば送信ボタンだったら

```
<input type="button" value="送信" onclick="snd()" >
```

のように。

水平線は<hr>と書けばいい。



ここは書き込みを表示する予定の所で、空白のdiv領域を作っておく  
id="timeline"

```
1 <html>
2 <head>
3   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
4 </head>
5
6 <body onload="disp()">
7
8
9
10
11
12
13
14
15
16 </body>
17 </html>
18
```

### (2) 書き込み側 python プログラム

ipython notebook 上などで、次の python プログラムを作ろう。

1 行目に

```
query = { 'twit': ['適当な文字列'] }
```

と書く(これは辞書である)。

query['twit'][0]が、'適当な文字列'になるので、それを変数に代入する。

ファイル「twtr.txt」を追記モードで開いて、その変数+改行を書き込む。

それだけ。

### (3) 表示側 python プログラム

ipython notebook 上などで、次の python プログラムを作ろう。

ファイル「twtr.txt」を読み込みモードで開いて、各行を<div>...</div>でくくって連結したものを出力せよ。

そこに書かれているのが例えば

あいうえお

aiueo

abcde

123

だったら、次の文字列を表示する。

「<div>あいうえお</div><div>aiueo</div><div>abcde</div><div>123</div>」



## 5-3. チャットの原型

(1) twtr.html

さっきのHTMLの&lt;head&gt;タグの内側に、次のスクリプトを書く。

```

<script type="text/javascript">
  function disp(){
    var s = document.createElement('script');
    s.src = 'twtr_disp.cgi?';
    document.body.appendChild(s);
  }
  function snd(){
    var s = document.createElement('script');
    s.src = 'twtr_recv.cgi?twit=' + document.getElementById("twit").value;
    document.body.appendChild(s);
  }
</script>

```

ここではdispとsndという二つの命令を作っている。

disp命令は、twtr\_disp.cgiを呼び出すだけである。

snd命令は、twtr\_recv.cgi?twit=(入力欄に書いたもの)を呼び出すだけである。

cgiのあとに?をつけて、変数名=なにか と書くと、cgiにその変数が渡される。

ここでは書き込みを渡すのである。

(2) twtr\_recv.cgi (書き込み側)

さっき作った書き込み側pythonプログラムの1行目をつぎのものに置き換え、

「twtr\_recv.cgi」と名付けて保存せよ。このプログラムが行うのは、

twtr\_recv.cgi?twit=なにか

で呼び出されたとき、queryという辞書に引数の内容を格納することである。これは実質的に

```
query = { 'twit': ['なにか'] }
```

を行うのと同じことになる。

```

1  #!/usr/local/bin/python
2  # -*- coding: utf-8 -*-
3  import os
4  import cgi
5
6  if 'QUERY_STRING' in os.environ:
7      query = cgi.parse_qs(os.environ['QUERY_STRING'])
8  else:
9      query = {}
10

```

### (3) twtr\_disp.cgi (表示側)

さっきの表示用 python プログラムを次の空欄のところに貼り付ける。さっきの出力を消して、代わりに次の 2 番目の空欄のところに、出力していた文字列が入っている変数を書く。これを「twtr\_disp.cgi」として保存しよう。

これは書き込みの表示用プログラムで、呼び出したブラウザに対し、timeline という id が付いているタグの内側を 2 番目の空欄の内容にせよという意味になる。timeline という id は twtr.html の表示欄に対応している。

```
1  #!/usr/local/bin/python
2  # -*- coding: utf-8 -*-
3
4  
5
6
7
8
9
10
11
12  print "Content-Type: text/javascript"
13  print ""
14  print "document.getElementById('timeline').innerHTML=\"%s\";" % 
15
```

### (4) twtr.txt

何も書かれていない twtr.txt というファイルを作る。これが書き込みを記録するファイルとなる。

### (5) サーバへのアップロード

さくらサーバの自分のフォルダの中に、4 つをアップロードする。

twtr.html  
twtr\_recv.cgi  
twtr\_disp.cgi  
twtr.txt

cgi と txt のパーミッションを設定する。右クリック編集で開いてみて文字コードが SJIS ではなく EUC か UTF に、改行コードが CR+LF ではなく LF になっていることを確認する。

### (6) ブラウザでアクセス！

## 5-4. 改良 1

(1) twtr.html : 名前欄の追加、スタイルの改良  
名前欄を追加しよう。

名前  つぶやき

---

snd 命令を書き換えて、今までつぶやきが” aiueo” だったとき  
twtr\_recv.cgi?twit=aiueo  
だったのを、さらに名前が” hoge” だったら  
twtr\_recv.cgi?twit=aiueo&name=hoge  
で呼び出すようにする。

さらに、<head>タグの内側に右のタグ領域を書き加える。これは、span.twitの欄ならば  
<div class=” twit” >という、class プロパティ  
が” twit” となっているdivタグの領域に対して適応するスタイルを指定する。この中括弧の中に、適応するスタイルを書いていく。どんなふうなスタイルにすると見栄えがいいかを考えて、プログラミングしてみよ。Twitareaは名前とつぶやきを含むdivで、nameは名前、twitはつぶやきの文字列と考えて良い。

```
<style>
div.twitarea {
}
span.name {
}
span.twit {
}
</style>
```

下は、右のようにスタイルを指定した時の表示例である。もっといいスタイルを作ってみるべし。

名前  つぶやき

a
b
aaa
bbb
mattya
/(^o^)\

```
<style>
div.twitarea {
border: solid 1px gray;
padding: 5px;
width: 500;
}
span.name {
font-weight: bold;
}
span.twit {
}
</style>
```

(2) twtr\_recv.cgi の名前欄への対応

受け取る引数が

```
q['twit'][0]
```

```
q['name'][0]
```

の二つになる。これらをファイル twtr.txt に書き込むのだが、名前が hoge でつぶやきが aiueo なら、書き込まれる一行は

```
いままで：aiueo¥n
```

```
これから：hoge###aiueo¥n
```

となるようにしよう。つまり、名前とつぶやきの間に###を挟むのだ。

(3) twtr\_disp.cgi の名前欄への対応

こちらでは、twtr.txt から###で区切られたデータを読み取らないといけない。split 命令で###で区切り、名前 hoge, つぶやき aiueo なら次のような文字列が出力されるようにせよ。

```
<div class='twitarea'><span class='name'>hoge</span><br><span  
class='twit'>aiueo</span></div>
```

## 5-5. 改良 2

いま、新しいつぶやきは一番下に書かれてしまうので、逆順に書かれるようにしたい。また、現在の時刻を表示したい。

名前  つぶやき

<b>mattya</b> 新しいつぶやきが上に来る	2013-03-29 13:43:28.595418
<b>mattya</b> piyopiyo	2013-03-29 13:41:08.273362
<b>asdf</b> aaaa	2013-03-29 13:40:29.928062

## (1) twtr.html

スタイルに次のものを追加しよう。time は現在時刻の文字列に対する class プロパティである。このように指定すると、とりあえず右側に張り付くようになる。

```
span.time {
    position: absolute;
    right: 15px;
}
```

## (2) twtr\_recv.cgi

```
import time
import datetime
t = str(datetime.datetime.today())
```

これで変数 `t` に現在時刻を表す文字列が代入される。これも `###` で区切ってファイルに書き込む内容に追加する。

## (3) disp

新しいもののほど上に出力させるには、

```
str = 'あたらしい文字列' + str
```

とやれば、`str` に新しい文字列ほど前に連結されることを利用する。

あと時刻を `class='time'` の `span` タグで表示させる。

5-6. 改良 3

書き込みに通し番号を付けて記録すると、いろいろできる。  
たとえば、書き込みに対して返信をしたり、いいねボタンをつけたりできる。