Python による HTML ファイル作成の省力化について

——モジュール仕様書——

0448011

目　次

——Python による HTML ファイル作成の省力化について——

1　はじめに

このプログラムは Python2.5 で使用することを前提としたモジュールです。他のバージョンでも使用できるかもしれませんが動作の保障は出来ません。

本プログラムは HTML を作成するさいの省力化を目的に作成し、作成出来る HTML は HTML4.01 に準拠します。非準拠のタグ及び属性は使用できません。また JavaScript は非対応です。Frame に関するタグも使用することは出来ません。

2　PY データの作成方法

初めにテキストエディターを開き

```
#!/usr/bin/env python
# -*- coding:Shift_JIS -*-


from html import *
title("TEST")
all = head()


all += end()
print all
```

上記のように入力

この状態で文書情報 (head タグ）部が自動で作られ、終了部も作られています。

title(””) 部分でタイトルを決めているので TEST と書いている部分

を自由に変えてください。

本文は　all = head()　と　all += end()　の間に記述することで作成できます。変数 all にすべて本文内容を加えていき出力します。

3　使用のためのルール

表示を設定するためのタグ（要素）は基本的に対応していません。使用したい場合は CSS を設定するか、STYLE の属性を使ってください。(STYLE タグには対応していません)

搭載したタグには汎用属性として CLASS、ID、STYLE が設定してあります。その他の属性についてはタグごとに使えるものが指定してあります。また、必須の属性以外はすべて大文字で属性を書いてください。属性を連続で追加したい場合は「,」記号で区切ってください。

例)

```
<a href='link address ' class="test" id="01">リンクさせたい文書</a>
```

属性 class と属性 id を指定する場合、最低入力引数を入力した後に、追加可能属性を「,」で区切り、大文字で「属性名=???」として書き加えることで使用可能です。

例)

```
a('リンクさせたい文書','link address',CLASS="test", ID="01")
```

4　階層構造の記述方法

同一階層に属するものを + 記号で連結、下の階層に属するものは基本的に一つ目の引数に入力することで表記出来ます。基本を理解してもらうサンプルとして HTML のテーブルタグを使って HTML と本モジュールを使った場合の違いを理解してください

```
  ---HTML サンプル---
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN" "http://www/w3.org/TR/html4/loose.dtd">
<html lang="ja">
<head>
<meta HTTP-EQUIV="Content-Type"
 CONTENT="text/html;CHARSET=Shift_JIS">
<meta HTTP-EQUIV="Content-Style-Type"
 CONTENT="textcss">
<title>sample</title>
</head>
<body
>
<TABLE border="1">
<TBODY>
<TR><TH>自己紹介</TH><TH>じこしょうかい</TH></TR>
<TR><TD>  名前</TD><TD>なまえ</TD></TR>
<TR><TD>  年齢</TD><TD>ねんれい</TD></TR>
<TR><TD>  趣味</TD><TD>しゅみ</TD></TR>
<TR><TD>  住所</TD><TD>じゅうしょ</TD></TR>
</TBODY>
</TABLE>
```

上記の HTML のテーブルは構造を示すと<table>の内部に<tr>を配置し、その中に<td>（または<th>）を同一階層で配置しています。また<tr>同士も同一階層です。

---モジュール使用した場合のソースコード---

```
#!/usr/bin/env python
# -*- coding:Shift_JIS -*-
from html import *
title("sample")
all = head()
all += table(
  tbody(
    tr(th("自己紹介") + th("じこしょうかい"))
    +tr(td("  名前") + td("なまえ"))
    +tr(td("  年齢") + td("ねんれい"))
    +tr(td("  趣味") + td("しゅみ"))
    +tr(td("  住所") + td("じゅうしょ"))
  )
,BORDER="1")
all += end()
print all
```

変数 all に記述内容を加えるため、まずは変数 all に足していく処理を初めに書き、その処理を＝の右側に書いていきます。

<table>タグの内部に設置するため、第一引数の内部に<tr>,<th>,<td>タグを入れる。第二引数にはテーブルの線の太さを決める要素 BORDER を入れる。<tr>タグの第一引数に隣接するセルである<td>タグをいれ、同じ階層なので + 記号で繋ぐ。また<td>タグを内包している<tr>同士も + 記号でつないでいく。

Python だとインデントを利用してブロックごとに記述できます。これ

を利用し、HTML 用タグを書く場合も改行とインデントを組み合わせて使うことで階層を表現できます。

5　文書情報（head タグ）に書き込むデータの追加方法

(1) スタイルシートを適用する場合

title() の下部分に変数 all には代入せず

stylesheet(' スタイルシートのリンクアドレス')　とすることで追加可能

● HTML ソース

```
<link rel="stylesheet" type="text/css" href="スタイルシートのリンクアドレス" title="default">
```

● Python コマンド

stylesheet("スタイルシートのリンクアドレス")

追加可能属性

DEFAULT=""

DEFAULT を指定することで属性 title を変更することが出来る

(2) meta 構文を追加する

title() の下部分に変数 all には代入せず

meta("書き込む内容")　とすることで追加可能

● HTML ソース

```
<meta　書き込む内容>
```

● Python コマンド

meta(' 書き込む内容')

書き込む内容はそのまま反映されるので

例)meta(’content="MSHTML 6.00.2900.3157" name=GENERATOR’)

のように書く

(3) 関連ファイルを head 部で指定する

title() の下部分に変数 all には代入せず

link(’URL’,’ リンクタイプ’)　とすることで追加可能

● HTML ソース

<link href="URL" rel="**リンクタイプ**" >

● Python コマンド

link(’URL’,’ リンクタイプ’)

追加可能属性

TUIKA=””

TUIKA を指定することで好きな項目を追加することが出来る

link(’URL’,’ リンクタイプ’,TUIKA=’ 追加したい項目’)

link(’URL’,’**リンクタイプ**’,TUIKA=’title=**草津市** type="application/rss+xml"’)

のように変数 TUIKA を指定することで任意の文字列を指定できる

(4) BASE の設定

title() の下部分に変数 all には代入せず

base(’ リンクアドレス’)　とすることで追加可能

● HTML ソース

<base href="**リンクアドレス**">

● Python コマンド

base(’ リンクアドレス’)

6　使用可能タグ一覧

(1) 基本使用タグ

(i)　a　　最低引数数 2

● HTML ソース

```
<a href='link address'> リンクさせたい文書 </a>
```

● Python コマンド

a(' リンクさせたい文書','link address')

追加可能属性

CHARSET="", HREFLANG="", TYPE="", NAME="", REL="", REV="", SHAPE="", COORDS="",TABINDEX="", ACCESSKEY="", TARGET="", CLASS="", ID="", STYLE=""

(ii) acronym　　最低引数数 1

● HTML ソース

```
<acronym>EU など頭文字略語につかう acronym</acronym>
```

● Python コマンド

acronym('EU など頭文字略語につかう acronym')

追加可能属性

CLASS="", ID="", STYLE=""

(iii) address　　最低引数数 1

● HTML ソース

```
<address>問い合わせ先</address>
```

● Python コマンド

address("問い合わせ先")

追加可能属性

CLASS=””, ID=””, STYLE=””

（iv）b　最低引数数 1

● HTML ソース

太字にしたい文字

● Python コマンド

b(”太字にしたい文字”)

追加可能属性

CLASS=””, ID=””, STYLE=””

（v）dir　最低引数数 2

● HTML ソース

<bdo dir="**文字方向**"> **指定したい文字**</bdo>

● Python コマンド

bdo(’指定したい文字’,’文字方向’)

追加可能属性　CLASS=””, ID=””, STYLE=””

（vi）big　最低引数数 1

● HTML ソース

<big>**大きくしたい文字**</big>

● Python コマンド

big(’大きくしたい文字’)

追加可能属性

CLASS=””, ID=””, STYLE=””

（vii） blockquote　**最低引数数** 1

● HTML ソース

<blockquote>**引用した文章**</blockquote>

● Python コマンド

blockquote(' 引用した文章')

追加可能属性

CITE="", CLASS="", ID="", STYLE=""

（viii） br　**最低引数数** 0

● HTML ソース

　改行を示す

● Python コマンド

br()

追加可能属性

属性無し

（ix） button　**最低引数数** 2

● HTML ソース

<button type=' **ボタン種別**'>**ボタンに書く文字**</button>

● Python コマンド

button(' ボタンに書く文字',' ボタン種別')

追加可能属性

NAME="", VALUE="", DISABLED="", TABINDEX="", ACCESSKEY="", CLASS="", ID="", STYLE=""

（x）caption　最低引数数 1

● HTML ソース

```
<caption>テーブルの見出し</caption>
```

● Python コマンド

caption(' テーブルの見出し')

追加可能属性

ALIGN="", CLASS="", ID="", STYLE=""

（xi）cite　最低引数数 1

● HTML ソース

```
<cite>引用・参照</cite>
```

● Python コマンド

cite(' 引用・参照')

追加可能属性

CLASS="", ID="", STYLE=""

（xii）code　最低引数数 1

● HTML ソース

```
<code>プログラムコード</code>
```

● Python コマンド

code(' プログラムコード')

追加可能属性

CLASS="", ID="", STYLE=""

（xiii）dd　最低引数数 1

● HTML ソース

<dd>**定義の説明**</dd>

● Python コマンド

dd(' 定義の説明')

追加可能属性

CLASS="", ID="", STYLE=""

（xiv）dfn　最低引数数 1

● HTML ソース

<dfn>**定義部**</dfn>

● Python コマンド

dfn(' 定義部')

追加可能属性

CLASS="", ID="", STYLE=""

（xv）div　最低引数数 1

● HTML ソース

 定義したいもの

● Python コマンド

div(' 定義したいもの','test01')

追加可能属性

CLASS="", ID="", STYLE=""

（xvi）dl　最低引数数 1

● HTML ソース

<dl> **定義リスト** </dil>

● Python コマンド

dl(’定義リスト’)

追加可能属性

CLASS=””, ID=””, STYLE=””

（xvii）dt　最低引数数 1

● HTML ソース

<dt>**定義語**</dt>

● Python コマンド

dt(’定義語’)

追加可能属性

CLASS=””, ID=””, STYLE=””

（xviii）em　最低引数数 1

● HTML ソース

強調

● Python コマンド

em(’強調’)

追加可能属性

CLASS=””, ID=””, STYLE=””

（xix）fieldset　最低引数数 1

● HTML ソース

<fieldset>**グループ化したい入力部品**</fieldset>

● Python コマンド

fieldset(’グループ化したい入力部品’)

追加可能属性

CLASS="", ID="", STYLE=""

（xx）form　最低引数数 2

● HTML ソース

```
<form action="action の動作">フォームの内部</form>
```

● Python コマンド

form(' フォームの内部','action の動作')

追加可能属性

METHOD="", ENCTYPE ="", ACCEPT_CHARSET="", ACCEPT="", NAME="", TARGET="", CLASS="", ID="", STYLE=""

（xxi）h1　最低引数数 1

● HTML ソース

```
<h1>見出し文</h1>
```

● Python コマンド

h1(' 見出し文')

追加可能属性

CLASS="", ID="", STYLE=""

（xxii）h2　最低引数数 1

● HTML ソース

```
<h2>見出し文</h2>
```

● Python コマンド

h2(' 見出し文')

追加可能属性

CLASS="", ID="", STYLE=""

（xxiii）h3　最低引数数 1

● HTML ソース

<h3>**見出し文**</h3>

● Python コマンド

h3(' 見出し文')

追加可能属性

CLASS="", ID="", STYLE=""

（xxiv）h4　最低引数数 1

● HTML ソース

<h4>**見出し文**</h4>

● Python コマンド

h4(' 見出し文')

追加可能属性

CLASS="", ID="", STYLE=""

（xxv）h5　最低引数数 1

● HTML ソース

<h5>**見出し文**</h5>

● Python コマンド

h5(' 見出し文')

追加可能属性

CLASS="", ID="", STYLE=""

（xxvi）h6　最低引数数 1

● HTML ソース

　　<h6>**見出し文**</h6>

● Python コマンド

　h6(' 見出し文')

追加可能属性

　CLASS="", ID="", STYLE=""

（xxvii）hr　最低引数数 0

● HTML ソース

　　<hr>

● Python コマンド

　hr()

追加可能属性

　CLASS="", ID="", STYLE=""

（xxviii）i　最低引数数 1

● HTML ソース

　　斜体文字にしたい文章

● Python コマンド

　i(' 斜体文字にしたい文章')

追加可能属性

　CLASS="", ID="", STYLE=""

（xxix）img　最低引数数 2

● HTML ソース

<img src="**画像のアドレス**" alt="**画像の表示名**">

● Python コマンド

img(' 画像のアドレス',' 画像の表示名')

追加可能属性

LONGDESC="", WIDTH="", HEIGHT="", USEMAP="", ISMAP="", NAME="", CLASS="", ID="", STYLE=""

(xxx) ins　最低引数数 1

● HTML ソース

<ins>**挿入された部分**</ins>

● Python コマンド

ins(' 挿入された部分')

追加可能属性

CITE="", DATETIME="", CLASS="", ID="", STYLE=""

(xxxi) kbd　最低引数数 1

● HTML ソース

<kbd>**キーボードから入力された文字**</kbd>

● Python コマンド

kbd(' キーボードから入力された文字')

追加可能属性

CLASS="", ID="", STYLE=""

(xxxii) label　最低引数数 1

● HTML ソース

<label>**関連付ける内容**</label>

● Python コマンド

label(' 関連付ける内容')

追加可能属性　FOR_ID="", ACCESSKEY="", CLASS="", ID="", STYLE=""

label につける属性で「for」があります。特殊なので「FOR_ID―」として属性を指定できるようにしました。

（xxxiii）legend　最低引数数 1

● HTML ソース

```
<legend>説明文</legend>
```

● Python コマンド

legend(' 説明文')

追加可能属性

ACCESSKEY="", CLASS="", ID="", STYLE=""

（xxxiv）li　最低引数数 1

● HTML ソース

```
<li>アイテムリスト</li>
```

● Python コマンド

li(' アイテムリスト')

追加可能属性

CLASS="", ID="", STYLE=""

（xxxv）map　最低引数数 2

● HTML ソース

```
<map  name="mapname">中身</map>
```

● Python コマンド

map(' 中身','mapname')

追加可能属性

CLASS="", ID="", STYLE=""

(xxxvi) noframes　最低引数数 1

● HTML ソース

`<noframes>`**フレーム未対応版ページへの内容**`</noframes>`

● Python コマンド

noframes(' フレーム未対応版ページへの内容')

追加可能属性

CLASS="", ID="", STYLE=""

(xxxvii) noscript　最低引数数 1

● HTML ソース

`<noscript>`**スクリプト未対応版ページへの内容**`</noscript>`

● Python コマンド

noscript(' スクリプト未対応版ページへの内容')

追加可能属性

CLASS="", ID="", STYLE=""

(xxxviii) object　最低引数数 1

● HTML ソース

`<object>object` **タグ内部の記述部**`</object>`

● Python コマンド

object('object タグ内部の記述部')

追加可能属性

DATA=””, TYPE=””, CLASSID=””, CODETYPE=””, ARCHIVE=””, CODEBASE=””, WIDTH=””,HEIGHT=””, USEMAP=””, DECLARE=””, STANDBY=””, NAME=””, TABINDEX=””, CLASS=””, ID=””, STYLE=””

（xxxix）ol　最低引数数 1

● HTML ソース

順序のあるリスト

● Python コマンド

ol(’順序のあるリスト’)

追加可能属性

CLASS=””, ID=””, STYLE=””

（xl）optgroup　最低引数数 2

● HTML ソース

<optgroup label="**属性で指定したグループ名**">**指定したい内容**</optgroup>

● Python コマンド

optgroup(’指定したい内容’,’属性で指定したグループ名’)

追加可能属性

DISABLED=””, CLASS=””, ID=””, STYLE=””

（xli）option　最低引数数 1

● HTML ソース

<option>**内容**</option>

● Python コマンド

option(' 内容')

追加可能属性

SELECTED= "", VALUE="", LABEL="",DISABLED="",
CLASS="", ID="", STYLE=""

(xlii) p　最低引数数 1

● HTML ソース

段落内部

● Python コマンド

p(' 段落内部')

追加可能属性

CLASS="", ID="", STYLE=""

(xliii) param　最低引数数 3

● HTML ソース

```
<param name='name' value='value' valuetype='valuetype'></param>
```

● Python コマンド

param('name','value','valuetype')

追加可能属性

CLASS="", ID=""

(xliv) pre　最低引数数 1

● HTML ソース

<pre>**整形済みテキスト**</pre>

● Python コマンド

pre(' 整形済みテキスト')

追加可能属性

CLASS="", ID="", STYLE=""

(xlv) q　最低引数数 1

● HTML ソース

<q>**引用部分**</q>

● Python コマンド

q(' 引用部分')

追加可能属性

CITE="", CLASS="", ID="", STYLE=""

(xlvi) rb　最低引数数 1

● HTML ソース

<rb>**ルビをふる対象**</rb>

● Python コマンド

rb(' ルビをふる対象')

追加可能属性

CLASS="", ID="", STYLE=""

(xlvii) rbc　最低引数数 1

● HTML ソース

<rbc>**グループ化する** rb **要素**</rbc>

● Python コマンド

rbc(' グループする rb 要素')

追加可能属性

CLASS="", ID="", STYLE=""

(xlviii) rp　最低引数数 1

● HTML ソース

<rp>ルビが表示できない場合のカッコ</rp>

● Python コマンド

rp('ルビが表示できない場合のカッコ')

追加可能属性

CLASS="", ID="", STYLE=""

(xlix) rt　最低引数数 1

● HTML ソース

<rt>ルビテキスト</rt>

● Python コマンド

rt('ルビテキスト')

追加可能属性

RBSPAN="", CLASS="", ID="", STYLE=""

(l) rtc　最低引数数 1

● HTML ソース

<rtc>グループ化する rt 要素</rtc>

● Python コマンド

rtc('グループ化する rt 要素')

追加可能属性

CLASS="", ID="", STYLE=""

（li）ruby　最低引数数 1

● HTML ソース

<ruby>**ルビふる対象**</ruby>

● Python コマンド

ruby(’ルビふる対象’)

追加可能属性

CLASS=””, ID=””, STYLE=””

（lii）samp　最低引数数 1

● HTML ソース

<samp>**プログラムからの出力サンプル**</samp>

● Python コマンド

samp(’プログラムからの出力サンプル’)

追加可能属性

CLASS=””, ID=””, STYLE=””

（liii）script　最低引数数 2

● HTML ソース

<script type=’**スクリプトの言語**’>**スクリプト内容**</script>

● Python コマンド

script(’スクリプト内容’,’スクリプトの言語’)

追加可能属性

SRC=””, CHARSET=””, DEFER=””

（liv）select　最低引数数 1

● HTML ソース

<select>**フォーム内の表示**</select>

● Python コマンド

select(' フォーム内の表示')

追加可能属性

NAME="", SIZE="", MULTIPLE="", DISABLED="", TABINDEX="", CLASS="", ID="", STYLE=""

（lv）small　最低引数数 1

● HTML ソース

<small>**小さくする文**</small>

● Python コマンド

small(' 小さくする文')

追加可能属性

CLASS="", ID="", STYLE=""

（lvi）span　最低引数数 1

● HTML ソース

適用範囲

● Python コマンド

span(' 適用範囲')

追加可能属性

CLASS="", ID="", STYLE=""

（lvii）strong　最低引数数 1

● HTML ソース

```
<strong>強調する文</strong>
```

● Python コマンド

strong(' 強調する文')

追加可能属性

CLASS="", ID="", STYLE=""

（lviii）sub　最低引数数 1

● HTML ソース

```
<sub>下付きにしたい文字</sub>
```

● Python コマンド

sub(' 下付きにしたい文字')

追加可能属性

CLASS="", ID="", STYLE=""

（lix）sup　最低引数数 1

● HTML ソース

```
<sup>上付きにしたい文字</sup>
```

● Python コマンド

sub(' 上付きにしたい文字')

追加可能属性

CLASS="", ID="", STYLE=""

（lx）table　最低引数数 1

● HTML ソース

<table>**テーブル要素**</table>

● Python コマンド

table(' テーブル要素')

追加可能属性

SUMMARY="", WIDTH="", BORDER="", FRAME="", RULES="", CELLSPACING="", CELLPADDING="", CLASS="", ID="", STYLE=""

(lxi) tbody　最低引数数 1

● HTML ソース

<tbody>**テーブルボディ要素**</tbody>

● Python コマンド

tbody(' テーブルボディの要素')

追加可能属性

ALIGN="", VALIGN="", CHAR="", CHAROFF="", CLASS="", ID="", STYLE=""

(lxii) td　最低引数数 1

● HTML ソース

<td>**テーブルのセルの内容**</td>

● Python コマンド

td(' テーブルのセルの内容')

追加可能属性

ROWSPAN="", COLSPAN="", ALIGN="", VALIGN="", CHAR="", CHAROFF="", ABBR="", AXIS="", HEADERS="",

SCOPE=””, CLASS=””, ID=””, STYLE=””

（lxiii） textarea　**最低引数数** 3

● HTML ソース

　<textarea cols=’**テキストエリアの横幅**’ rows=’**テキストエリアの縦幅**’>**入力フォーム**</textarea>

● Python コマンド

　textarea(’ 入力フォーム’,’ テキストエリアの横幅’,’ テキストエリアの縦幅’)

追加可能属性

　NAME=””, DISABLED=””, READONLY=””, TABINDEX=””, ACCESSKEY=””, CLASS=””, ID=””, STYLE=””

（lxiv） tfoot　**最低引数数** 1

● HTML ソース

　<tfoot>**テーブルフッター要素**</tfoot>

● Python コマンド

　tfoot(’ テーブルフッター要素’)

追加可能属性

　ALIGN=””, VALIGN=””, CHAR=””, CHAROFF=””, CLASS=””, ID=””, STYLE=””

（lxv） th　**最低引数数** 1

● HTML ソース

　<th>**テーブルヘッダ部**</th>

● Python コマンド

th(' テーブルヘッダ部')

追加可能属性

ROWSPAN="", COLSPAN="", ALIGN="", VALIGN="", CHAR="", CHAROFF="", ABBR="", AXIS="", HEADERS="", SCOPE="", CLASS="", ID="", STYLE=""

(lxvi) thead　最低引数数 1

● HTML ソース

<thead>**テーブルヘッダー要素**</thead>

● Python コマンド

thead(' テーブルヘッダー要素')

追加可能属性

ALIGN="", VALIGN="", CHAR="", CHAROFF="", CLASS="", ID="", STYLE=""

(lxvii) tr　最低引数数 1

● HTML ソース

<tr>**テーブルの行内容**</tr>

● Python コマンド

tr(' テーブルの行内容')

追加可能属性

ALIGN="", VALIGN="", CHAR="", CHAROFF="", CLASS="", ID="", STYLE=""

(lxviii) tt　最低引数数 1

● HTML ソース

<tt>**等幅フォントで表示**</tt>

● Python コマンド

tt(' 等幅フォントで表示')

追加可能属性

CLASS="", ID="", STYLE=""

(lxix) ul　最低引数数 1

● HTML ソース

順序の無いリスト

● Python コマンド

ul(' 順序の無いリスト')

追加可能属性

CLASS="", ID="", STYLE=""

(lxx) var　最低引数数 1

● HTML ソース

<var>**プログラムコードの引数**</var>

● Python コマンド

var(' プログラムコードの引数')

追加可能属性

CLASS="", ID="", STYLE=""

(2) 定義名が HTML と違うタグ

この章に紹介されているタグはそのままの綴りでは使用できず、独自のコマンドとして定義したので Python コマンドに注意してください。

(i) del　最低引数数 1

del のコマンドが python に使われ使用できないので delete として定義した。

● HTML ソース

```
<del>削除部</del>
```

● Python コマンド

delete(' 削除部')

追加可能属性

CLASS="", ID="", STYLE=""

input タグは type 属性により追加できる属性に変化があるのでパターン別に分けた。

(ii) input type="submit"　最低引数数 1

● HTML ソース

```
<input type="submit" value="送信">
```

● Python コマンド

inputsubmit(' 送信')

追加可能属性

NAME="", CLASS="", ID="", STYLE=""

(iii) input type="reset"　最低引数数 1

● HTML ソース

```
<input type="reset" value="リセット">
```

● Python コマンド

inputreset(' リセット')

追加可能属性

CLASS=””, ID=””, STYLE=””

（iv） input type=”image” **最低引数数** 3

● HTML ソース

<input type="image" src="**画像のアドレス**" alt="**画像の表示名**" name ="cgi **用の** name">

● Python コマンド

inputimage(’ 画像のアドレス’,’ 画像の表示名’,’cgi 用の name’)

追加可能属性

CLASS=””, ID=””, STYLE=””

（v） input type=”checkbox” **最低引数数** 2

● HTML ソース

<input type="checkbox" name="cgi **用の** name" value="**オンの時の値**">

● Python コマンド

checkbox(’cgi 用の name’,’ オンの時の値’)

追加可能属性

CHECKED=””, CLASS=””, ID=””, STYLE=””

（vi） input type=”hidden” **最低引数数** 2

● HTML ソース

<input type="hidden" value="**初 期 値**" name ="cgi **用 の** name">

● Python コマンド

hidden(' 初期値','cgi 用の name')

追加可能属性

CLASS="", ID="", STYLE=""

(vii) input type="password" **最低引数数** 1

● HTML ソース

```
<input type="password" name ="cgi 用の name">
```

● Python コマンド

password('cgi 用の name')

追加可能属性

SIZE="", VALUE="", MAXLENGTH="", CLASS="", ID="", STYLE=""

(viii) input type="text" **最低引数数** 1

● HTML ソース

```
<input type="text" name ="cgi 用の name">
```

● Python コマンド

text('cgi 用の name')

追加可能属性

SIZE="", VALUE="", MAXLENGTH="", CLASS="", ID="", STYLE=""

(ix) input type="radio" **最低引数数** 2

● HTML ソース

```
<input type="radio" value="オンの時の値" name ="cgi 用
の name">
```

● Python コマンド

radio(’オンの時の値’,’cgi 用の name’)

追加可能属性

CHECKED=””, CLASS=””, ID=””, STYLE=””

(3) 独自仕様命令

(i) コメント　最低引数数 1

特殊なコマンド

html 内にだけ表記するコメントを書き加えたい場合

● Python コマンド

commnet(’コメント’)

とすることで

<!--　　**コメント**　　-->

として書き添えることが出来る

(4)

HTML に出力したファイルに改行をいれたい場合、改行位置に `\n` を加える。これは改行記号なので HTML に変換した場合は改行として出力される。

7　PY データの具体的な変換方法

変換する PY データと使用する HTML モジュールを同一フォルダに設置

コマンドプロンプトを起動

「python　PY データ名.py　＞　使用したい HTML 名.html」

プロンプト画面に特に何も表示されなければ変換完了です。

8　注意

基本的にエラーは文法ミスが大半になると思います。一部のタグは必要とする引数がデフォルトで 2 つ必要だったりするので確認してください。

また、文字コードの関係で使用できない漢字が一部あります。その場合の対処方法は用意していませんので、お手数ですがその部分のみを書き換えるなどで回避してください。