

2. Form Objects フォーム・オブジェクト 【PR267】

Web ページ内からのメールやアンケートなどで使われる、「ボタン」や「フィールド」は、「フォーム・オブジェクト」と呼ばれる。form オブジェクトは、CGI¹に要素を渡すための GUI である。

こうした要素のグループを「フォーム (form)」と呼び、その中の各々の要素を「エレメント (elements)」²と呼ぶ。これら form や elements などの要素は、通常、すべて配列として扱われる。

サンプル：

html での表現

```
<form>
<input type="text"><BR>
<input type="text"><BR>
<input type="button"><BR>
</form>
```

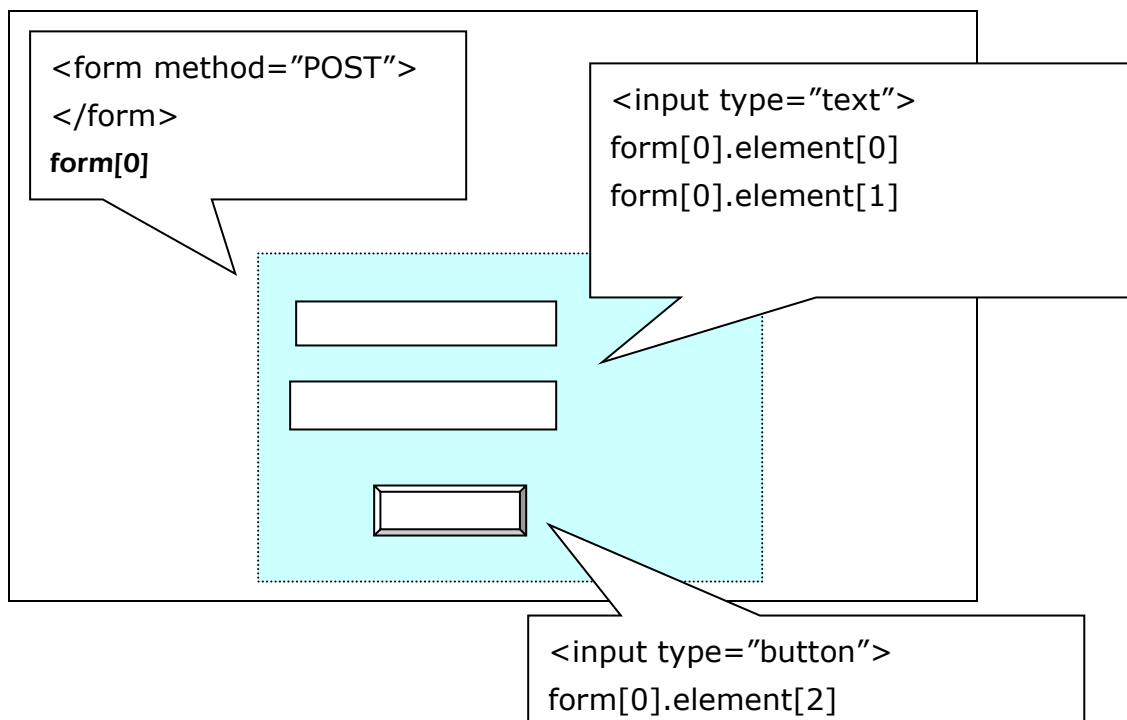
JavaScript での表現：要素の順番で表わす

form[0]

form[0].elements[0]

form[0].elements[1]

form[0].elements[2]



¹ Common Gateway Interface の略。

² elements : PR270

2-1. フォームの定義

フォームの定義は、`<form>` タグで定義される。

```
<form action="URL" method="GET|POST" name="フォーム名">
```

`action` は、実行する CGI の URL。

`method` は、CGI へのデータの送信方法。標準入力方式に従う「POST」と、環境変数 `QUERY_STRING` に送る「GET」がある。

2-2. フォームの各エレメント

1) テキストフィールド（テキスト・オブジェクト）

1 行だけのテキストフィールド。

```
<input type="TEXT"  
  name="テキスト・フィールド名"  
  value="デフォルトで表示される文字"  
  maxlength="入力できる最大文字数"  
  size="画面に表示されるフィールドの長さ">
```

サンプル：

```
<form>  
<input type="text" name="address" size="20">  
</form>
```

`size` などを入力する数値は半角の数なので、全角を使う日本語の場合、2 倍の値が必要。

2) テキストエリア

複数行のテキストフィールド。また、スクロールも可能。

```
<textarea  
name="テキストエリア名"  
row="行数"  
cols="1行あたりの文字数">  
  // この間に書かれた文字は、テキストエリア内に初期値として  
  // 表示される  
</textarea>
```

サンプル： <textarea name="msg" rows=5 cols=20>



3) ボタン (button オブジェクト)

```
<input type="button"  
name="ボタン名"  
value="ボタン表面に表示される文字">
```

ボタンの幅は、表面に表示される文字列により、自動的に調整される。

サンプル； <input type="button" name="myBtn" value="Click me!">

4) ラジオボタン (radio オブジェクト)

ユーザーが、複数の項目の中から「ひとつだけ選択」するためのオブジェクト。

```
<input type="radio"
      name="ラジオボタン名"
      value="送られる文字列">
```

ラジオボタンでは、選択する項目の数だけ<input type= “radio” >の文が必要になる。

サンプル :

```
<form name="myForm">
```

性別は？


```
<input type="radio" name="gender" value="male" CHECKED>男<BR>
```

```
<input type="radio" name="gender" value="female">女<BR>
```

```
</form>
```

※ CHECKED があると、デフォルトで選択されている。

5) チェックボックス (checkbox オブジェクト)

ユーザーが、「複数項目を選択」するためのオブジェクト。

```
<input type="checkbox"
      name="チェックボックス名"
      value="送られる文字列">
```

「チェックボックス」も「ラジオボタン」と同様に、選択する項目の数だけ

<input type= “check” >が必要になる。

6) 選択ボックス (select オブジェクト)

あらかじめ決められている項目をメニューの中から選択する。

```
<SELECT
  name="選択メニュー名"
  size="表示される行数"
  <option value="項目名 1">選択ボックスに表示される文字列 1
  <option value="項目名 2">選択ボックスに表示される文字列 2
  <option value="項目名 3">選択ボックスに表示される文字列 3
</SELECT>
```

サンプル : Select を使った Form による OS の選択メニュー

```
<form name="myForm">
<SELECT name="OS" size="1">
<option value="WIN">Windows
<option value="MAC">Mac OSX
<option value="LINUX">Linux
<option value="OTHER">Other
</SELECT>
</form>
```

2-3. Form のアクセス

2-3-1. 名前による Form へのアクセス

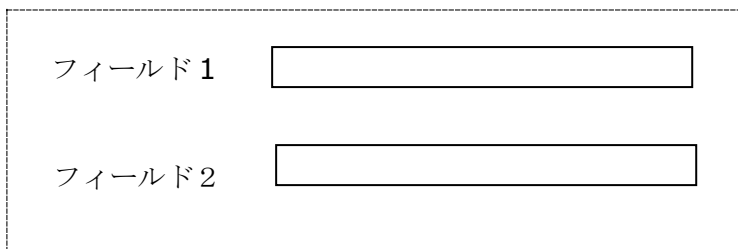
参考にする、テキストボックスを使った、サンプルのフォーム。
まずは、`<form>`を html で記述する。

```
<form name="myForm">
```

```
  フィールド 1<input type="text" name="text_1" size="20">
```

```
  フィールド 2<input type="text" name="text_2" size="20">
```

```
</form>
```



JavaScript で上段のテキストボックス「フィールド 1」にアクセスするには、

```
document.myForm.text_1.value
```

同じく、下段のテキストボックス「フィールド 2」にアクセスするには、

```
document.myForm.text_2.value
```

とそれぞれのテキストボックスを指し示す。

`value` は、テキストオブジェクトのデータを指している。

2-3-2. 配列による Form へのアクセス

`document` オブジェクトは、フォームへのアクセスを `forms` 配列として管理している。
同様に、各エレメントへのアクセスも `elements` 配列として管理している。

「配列」を使い、上のサンプルにアクセスするには、

```
document.forms[0].text_1.value
```

または、エレメントにも配列を使いアクセスすることができる。

```
document.forms[0].elements[0].value
```

サンプルプログラム: テキストボックスに入れた文字を、もうひとつのテキストボックスに表示する。

1) 名前をそのまま使って、コピーをする例

> テキストボックスとボタンの<form>

```
<form name="myForm">
  フィールド 1 : <input type="text" name="text_1" size="20"><BR>
  フィールド 2 : <input type="text" name="text_2" size="20"><BR>
  <input type="button" name="copy" value="Copy" onClick="copyField()">
</form>
```

> コピーを実行する JavaScript 「copyField()」を定義する。

```
<script language="JavaScript">
  function copyField(){
    // text_1 の内容を text_2 に代入する。
    document.myForm.text_2.value=document.myForm.text_1.value;
  }
</script>
```

応用：上の例では、コピーするフォーム名が固定されているが、同じようなフォームがいくつもあると copyField2() のような関数を複数作らなくてはならない。

これをさけるためには、form オブジェクトへの参照を、copyField() 関数のパラメータとして渡す。

```
<input type="button" name="copy" value="Copy" onClick="copyField(this.form)">
this.form (このフォーム) で、オブジェクト自身への参照になる。
```

次に、CopyFiled() 関数を、form オブジェクトへの参照を引数とするように書きなおすと、

```
function copyField(theForm){
  theForm.text_2.value=theForm.text_1.value
}
```

■this を使って、書きなおしサンプル。

this³はオブジェクト、それ自身を指し示します。使い方によっては非常に便利です。

```
<html>
<head>
<script>
function copyField(theForm){
  theForm.text_2.value=theForm.text_1.value
}
</script>
</head>
<form name="myForm">
フィールド 1 : <input type="text" name="text_1" size="20"><BR>
フィールド 2 : <input type="text" name="text_2" size="20"><BR>
<input type="button" name="copy" value="Copy" onClick="copyField(this.form)">
</form>
```

³ this ; PR37

2) サンプル・プログラム：配列を使った例。

次に、「フォームブロック」に、3つの「テキストボックス」を JavaScript の配列を使って用意し、テキストを表示する例を考えよう。

forms[] と elements[] は、配列なので '0' から始まる。

forms[0].elements[0] の値を得るには、配列を使い

forms[0].elements[0].value

とする。

```
<html>
<head></head>
<body>
<script LANGUAGE="JavaScript">
  document.write("<form method='POST'>");
  for (i=0; i<3; i++){
    document.write ("<input type='text' size=20><BR>");
    document.forms[0].elements[i].value="テキストボックス番号"+i;
  }
  document.write("</form>");
</script>
</body>
</html>
```

この例では、html の記述も JavaScript で記述している。



2-4. イベント処理

フォームの要素である「テキストボックス」「ボタン」などは、マウスがクリックしたり、その上を通過したときなどに、何かの動作が起こる。こうしたことを、「イベント」と呼び「イベントオブジェクト」で処理される。

イベント関数の代表的なものは、

onClick, onChange, onFocus, onMouseOver

などがある。*

「イベント処理」の後、「呼び出しもと」のフォームの要素を参照するためには、呼び出しもとの側から、**this**（部品オブジェクトの参照）または **this.form**（部品のあるフォームオブジェクトへの参照）などが使われる。

サンプル：

ボタンクリックで、現在時刻を表示するサンプルで、**this** の使われ方を見る。

```
<script LANGUAGE="JavaScript">
```

```
function dispd(f){
  f.today.value=new Date();
}
```

```
/script>
```

テキストボックス
today を参照してい

this が置かれているフォームへの参照

```
<form method="POST">
```

押して、計算
をする。

```
<input type="button" value="Push" onClick="dispd(this.form)"> <BR>
```

数字をテキ
ストボックス
に入力して

```
<input type="text" size=20 name="today"> <BR>
```

```
</form>
```

部品名

も、それはただの「文字列」である。

この文字列を「数値」に置き換える関数に **eval** がある。⁴

* イベントは「イベントオブジェクト」としてまとめられています。【PR318】

⁴ eval ; 【PR70】

◆サンプル 足し算が得意な電卓

```
<html>
<head>
<script LANGUAGE="JavaScript">
    // 足し算の関数 calc の定義
    function calc(f){
        f.z.value=eval(f.x.value)+eval(f.y.value);
    }
</script>
</head>
<form method="POST">
    ボタン
<input type="button" value="計算" onClick="calc(this.form)"><BR>
    入力用ボックス
<input type="text" size=10 name="x">+
<input type="text" size=10 name="y">=
<input type="text" size=10 name="z"><BR>
</form>
</html>
```

10-5.ラジオボタンの値の取得

イベント処理関数で、ラジオボタンの値を取得するには、呼び出しもとから **this** 引数をもらわなくてはならない。

サンプル :

```
function backdisp(parts){
    f=parts.form; //ラジオボタンのあるフォームへの参照を f に代入
    f.box.value=parts.value;
        // ↑クリックされたラジオボタンの値
    .....
}

<input type="text" .... name="box" ...>
        // テキストボックスの名前
<input type="radio" ... name="back" ....onClick="backdisp(this)">
        //   ↑ラジオボタングループの名前
```

プログラム 1 : ラジオボタンで勝手に信号機

```
<html>
<head>
<TITLE>ラジオボタン</TITLE>
<script LANGUAGE="JavaScript">
  function backdisp(parts){
    f=parts.form;  //フォームへの参照
    f.box.value=parts.value;
    if (parts.value=="緑")
      document.bgColor=0x00ff00;
    if (parts.value=="黄")
      document.bgColor=0xffff88;
    if (parts.value=="赤")
      document.bgColor=0xff0000;
  }
</script>
</head>
<body>
<form method="POST">
<input type="radio" name="back" value="緑" onClick="backdisp(this)">進め
<input type="radio" name="back" value="黄" onClick="backdisp(this)">注意
<input type="radio" name="back" value="赤" onClick="backdisp(this)">止まれ<BR>
信号は、<INPUT name="box" size=3>です。
<BR>
</body>
</html>
```

このプログラムの場合、参照で渡すのは漢字で書かれた色です。
このとき、クリックされたラジオボタンを参照（this）引数として渡す。