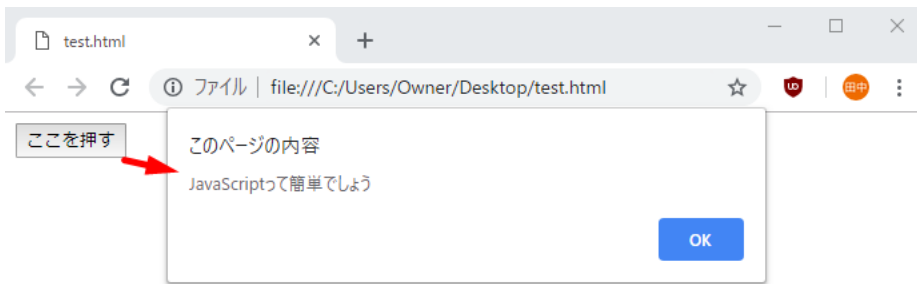


## ● JavaScript の基礎

JavaScript は HTML 同様、「タグ」言語ですので、コンパイルの必要がありません。簡単なエディタとブラウザがあれば、それで十分です。HTML と同様にエディタで作成したプログラムをブラウザで見るということだけです。用法としては、HTML ドキュメント内に JavaScript を埋め込んで使用します。基本的にはコンピュータとプラットフォームに依存しないので、どんな OS 上でも同じ様に動作します。

## ● JavaScript の簡単な例

JavaScript の技法を記述する前に、まず JavaScript で記述された簡単な例を見ることにしましょう。



```
<html>
<head>
<title>JavaScript の簡単な例</title>
</head>
<body>
  <form>
    <input type="button" value="ここを押す"
      onClick="alert('JavaScript って簡単でしょう')">
  </form>
</body>
</html>
```

このボタンをクリックしたら、「JavaScript って簡単でしょう」というダイアログが出ます。ここでの「onClick」が「ボタンがクリックされたら…」という意味で、「alert()」はダイアログを出す命令です。ダイアログに表したい文書を alert() の括弧「()」の中にダブルクォーテーション「"」でくくって書きます。

## ● JavaScript の基礎形式

JavaScript は HTML 文書の中に埋め込んで使います。そのために、ブラウザに、HTML 文書のどの部分に JavaScript のコードが書きこまれているかを知らせる必要があります。その役割は、`<script></script>` を用いて果たします。ブラウザは、`<script>` と `</script>` の間に書かれているものをスクリプトコードと判断します。

JavaScript は、HTML ドキュメントの任意の場所に埋め込むことができます。基本的な記述方式は次の 3 つです。

- 1) HTML の `<body>` と `</body>` の間に記述する
- 2) `<head>` と `</head>` の間に書く
- 3) HTML の外部に書く

各記述形式の例を見ましょう。

- 1) `<body>` と `</body>` の間に記述する

```
<html>
<head>
    <title>JavaScript の埋め込み方法</title>
</head>
<body>
    <script language="JavaScript">
        ここに JavaScript の文書を記述する
    </script>
</body>
</html>
```

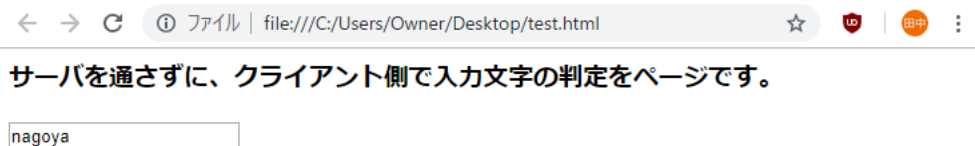
- 2) `<head>` と `</head>` の間に記述する形式

```
<html>
<head>
    <title>JavaScript の埋め込み方法</title>
    <script language="JavaScript">
        ここに JavaScript の文書を記述する
    </script>
</head>
<body>
    ここにページ関連のその他の内容と上記の JavaScript を起動させる内容を記述します。
</body>
</html>
```

### 3) HTML の外部に書く記述形式

```
<html>
<head>
  <title>JavaScript の埋め込み方法</title>
  <script language="JavaScript" src="外部のファイル名.js">
</head>
<body>
  ここにページ関連のその他の内容と上記の JavaScript を起動させる内容を記述します。
</body>
</html>
```

次に、少々難しいと思われるかもしれませんが、サーバに頼らず、クライアント側で、入力文字の判定を行う例を見てみましょう。この例では、nagoya 以外の文字を入力すると、再入力させるページです。



```
<html>
<head>
<title>入力文字のチェック</title>
  <script language="javascript">
    function key(){
      if(document.myform.mytext.value="nagoya"){
        alert("OK!")
      }else{
        alert("再入力")
      }
    }
  </script>
</head>
<body>
<h3>サーバを通さずに、クライアント側で入力文字の判定をページです。</h3>
<form name="myform">
  <input type="text" name="mytext" value="" onBlur="key()" >
</form>
</body>
</html>
```

## ● JavaScript の用語知識

ここでは、JavaScript を理解する上で、不可欠な概念と用語を解説します。

### オブジェクト

コンピュータの言語の世界では、「オブジェクト」指向言語というものがあります。オブジェクトとは、簡単に言えば、“もの”という概念を抽象化したものです。データとその性質およびその操作をひとまとまりの“もの”（オブジェクト）としてとらえる発想です。そしてそのオブジェクト単位でプロジェクトを作る技法を「オブジェクト指向」と言います。

JavaScript では、ほとんど全ての要素をオブジェクトとして扱えます。JavaScript のオブジェクトは「**組み込みオブジェクト**」と「**ナビゲータオブジェクト**」の2種類に分かれます。

#### 「組み込みオブジェクト」

文字列を管理する String オブジェクト、数値演算を行う Math オブジェクト、日付時刻管理の Date オブジェクトなど、言語仕様上あらかじめ組み込まれているオブジェクトを指します。

#### 「ナビゲーションオブジェクト」

ブラウザがあらかじめ持っている画像やフォームなどの部品を取り扱うものです。**ナビゲータオブジェクトの階層関係を明確にすることは大事です。**

=====

ナビゲータオブジェクトの最上層のオブジェクトは window オブジェクトと navigator オブジェクトです。

**window オブジェクト**：ブラウザを操作するためのオブジェクト

**navigator オブジェクト**：ブラウザに関する情報やプラグインに関する情報を管理するオブジェクト（説明省略）

window オブジェクトでは、ボタンイベントを起こしたり、入力したり、サブウィンドウを開いたりすることができます。

#### 【window オブジェクトの下層】

⇒ **document オブジェクト**、location オブジェクト、history オブジェクト、frame オブジェクトなど

#### 【document オブジェクトの下層】

⇒ **form オブジェクト**、link オブジェクト、anchor オブジェクトなど

#### 【form オブジェクトの下層】

⇒ button オブジェクト、text オブジェクト、textarea オブジェクト、radio オブジェクト、select オブジェクトなど

JavaScript の言語では、下層のオブジェクトは上層のオブジェクトのプロパティ（後述）と見なされます。つまり、JavaScript を使うときには、オブジェクトの上下関係さえはっきりすれば、オブジェクトやプロパティを意識する必要がなくなります。

⇒ そもそもオブジェクトの生成ってどうやってするのでしょうか？ → 確認していきましょう。

⇒ また、沢山例を見てオブジェクトとプロパティ（&メソッド）の関係を理解していきましょう。

## ●オブジェクトの生成

JavaScript のオブジェクトを生成するには、new キーワード（演算子）を使用します。例えば、JavaScript に組み込まれている String オブジェクトを用いて、String オブジェクトを生成するには

```
var str;           → 変数を宣言する文
str = new String(); → 「=」 は右辺の値を左辺に入れる「代入演算子」
```

あるいは、上記の 2 つを合わせて

```
var str = new String();
```

のように記述します。

## ●変数

変数とは、値を記憶しておく箱のようなものです。変数の宣言を行うには、キーワード var を使います。

書式：

```
var 変数名;    → 複数 var a1, a2, a3; と書くことも可能
```

変数型：「数値型」「文字列型」「ブーリアン型」「オブジェクト型」

JavaScript では、変数の型に関する要求が他の言語と違い、非常に柔軟です。キーワード var を用いて宣言しなくても、いきなり変数に数値を入れることもできます。また、入れる値として、変数に数値型の値、文字列型の値、ブーリアン型の値、オブジェクト型の値のどれを入れても良いのです。

代入式は次のようになります。

```
var 変数名 = 入れる値;
```

このようにして設定した後、値の代わりに変数名を用いて値を取り出すこともできますし、もちろん、値を入れ直すこともできます。

## ●プロパティ

オブジェクトの外観や性質などの属性を表す部分をプロパティといいます。例えば、document オブジェクトにはバックグラウンドの色を指定する bgColor などのプロパティがあります。

プロパティを実行するには、オブジェクト名をピリオド「.」で区切ってつなげて使います。また、プロパティには、ユーザが値を設定するものと値設定不要の読み出し専用の 2 種類があります。また、下層のオブジェクトは、上層のオブジェクトのプロパティとみなされます。

プロパティに値を設定する場合の書式：

オブジェクト名.プロパティ名=値;

例えば、「document.bgColor=White;」と書くことができます。

読み出し専用の場合の書式：

オブジェクト名.プロパティ名;

例えば、「document.lastModified;」（参考：ファイルの最終更新日時の制御）と書くことができます。

## ●メソッド

メソッドとは、オブジェクトを操作するための命令です。メソッドも、プロパティと同様、オブジェクト名とメソッド名をピリオド「.」でつなげて、次のように記述します。

オブジェクト名・メソッド名（値）;

右端の（）の中の「値」には、メソッドに引き渡すパラメータを書きます。パラメータが 1 個以上の場合は、各パラメータをカンマ「,」で区切って設定します。実行時の記述書式は次のようになります。

オブジェクト名メソッド名（値 1, 値 2, …）;

## ●関数

関数とは、一連の処理プロセスをまとめて、関数名でそれに名前を付け、その名前で、一連のプロセスを呼び出せるようにしたものです。

JavaScript に関する関数は、大きく分けると 2 種類あります。1 つはユーザ定義関数で、もう 1 つは JavaScript の仕様にあらかじめ用意されている「組み込み関数」です。

ユーザ定義関数

JavaScript における関数定義の書式は次のようになります。

```
function 関数名（引数 1, 引数 2, …） {  
    実行文;  
}
```

## >> 関数の続き

関数定義中、引数が必要でない場合には、引数の記述を省略することができますが、引数を省略した場合でも、関数名の後の括弧「()」を残す必要があります、省略できません。

関数の処理プロセス部分は { } で囲まれた「実行文」ブロックの中に記述します。実行文には、処理するセンテンスの数の制限はありません。センテンスを複数書いても大丈夫です。ただし、各文をセミコロン「;」で区切って書く必要があります。

JavaScript では、関数の中で使われている変数の有効範囲を明確にする必要があります。一般的には、var キーワードを使って関数内で定義した変数は、その関数の中で有効となります。このような、プログラムの一部分だけで有効な変数をローカル変数といいます。それに対して、関数の外側で定義された変数は、プログラム全体で有効になります。このような、プログラム全体で有効な変数をグローバル変数といいます。

関数を書く位置は、通常 HTML ドキュメントの<head></head>の間と決まっています。前述したように、js ファイル形式で外部から与えることもできます。

関数の呼び出しは、通常ページ上やイベントハンドラ（後述）内で行います。関数を呼び出したい部分に「関数名（引数 1，引数 2，…）」と記述して設定します。

## ● イベント処理

JavaScript では、ユーザがボタンをクリックしたりメニューを選択したりなどのユーザからのリクエストや HTML ドキュメントがロードされたりなどシステムからのリクエストによって発生する特定の動作・処理を起こすタイミングをイベントと言います。

イベントを捕らえて、そのタイミングで特定の処理を実行する命令をイベントハンドラと言います。

例： onClick="window.alert('JavaScript へようこそ')"

イベントハンドラ名="実行するステートメント";

### <代表的なイベントハンドラ>

onClick ... オブジェクトをクリックされた時にイベントを起こす。

onSubmit ... フォームの送信ボタンが押されたときにイベントを起こす

onChange ... フォームの内容が変更されたときにイベントを起こす。

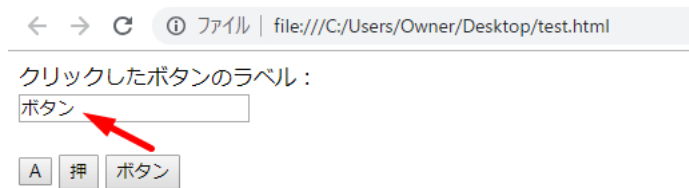
onBlur ... オブジェクトがフォーカスを失ったときにイベントを起こす。

onMouseOut ... マウスがオブジェクトから離れたときにイベントを起こす。

onMouseOver ... マウスがオブジェクトの中に入ったときイベントを起こす。

ここで、例を見てみましょう。

## ■ クリックしたボタンのラベルを得るには



### <文字データ扱い編>

テキストフィールドの内容に関する操作を行うには、Text オブジェクトの value プロパティを使います。書式は、「**document.テキストフィールド所在のフォーム名.テキストフィールド名.value**」です。

Text オブジェクトは Form オブジェクトのプロパティで、Form オブジェクトは、document オブジェクトのプロパティですので、書式のテキストフィールド名の前にある document.フォーム名は省略することができます。

```
<html>
<head>
<title>入力文字のチェック</title>
  <script language="javascript">
    function getlabel(obj){
      document.formname.txt1.value = obj;
    }
  </script>
</head>
<body>
<form name="formname">
  クリックしたボタンのラベル：<br>
  <input type="text" name="txt1"><br><br>

  <input type="button" value="A" onClick="getlabel('A')" >
  <input type="button" value="押" onClick="getlabel('押')" >
  <input type="button" value="ボ タ ン" onClick="getlabel('ボ タ ン')" >

</form>
</body>
</html>
```



■ ボタンクリックでテキストフィールド間の内容をコピーするには

文字を入力：

nagoya  copy

ボタン押下で左のテキストボックスの値が  
右のテキストボックスにコピーされる

```
<html>
<head>
<title>入力文字のチェック</title>
  <script language="javascript">
    function copyTextToText(){
      a = document.formname.text1.value;
      document.formname.text2.value = a;
    }
  </script>
</head>
<body>
<form name="formname">
  文字を入力：<br>
  <input type="text" name="text1" size="5" value="nagoya">
  <input type="text" name="text2" size="5">

  <input type="button" value="copy" onClick="copyTextToText();" >
</form>
</body>
</html>
```

## ● 条件制御文とループ制御文

プログラムでは、決められた数だけ操作を繰り返したり、ある条件を判断し、その判断に応じた処理を行ったりするケースをよく見かけます。ここでは、JavaScript によるループと条件制御について説明します。

### 条件判断文（条件分岐文）

条件を判断し、その判定結果によって、プログラムの実行部分を決めるには「if」を使います。if 文の最も基本的な書式は次の通りです。

```
if(条件式){
    条件式を満たしたとき実行される部分
} else {
    条件式を満たされないとき実行される部分
}
```

【要確認】 演算子の種類 （基本的に Java と同じ）

算術演算子／比較演算子／論理演算子／代入演算子

```
var test_listen=70;
var test_write=85;

if(test_listen >= 60 && test_write>=60){
    document.write("あなたは合格です");
} else {
    document.write("残念でした。不合格です！");
}
```

### ループ制御文

決められた回数、あるいは条件が満たされるまで、特定の処理を繰り返し行う形式をループといいます。

## ●for 文

for 文の書式は次のようになります。

```
for(初期値；終了条件；増減式){
    処理文
}
```

## >> ループ制御文の続き

for 文の実行手順は、ループに入る前に「初期値」が実行され、「終了条件」の値が真であれば、{ } に囲まれている処理文の実行に移り、「処理文」の実行の後に「増減式」が実行されます。例えば

```
for(i=0; i<10; i++){  
    処理文  
}
```

は、初期値として、i を 0 から始め、i が 10 未満の間、i の値に 1 を足しながら、処理文の処理を繰り返し実行します。

### ●while 文

while 文もループ制御文のひとつです。While 文の書式は次のようになります。

```
while(条件判定式){  
    処理文  
}
```

while 文では、条件を判定し、判定式が真である間は、{ } で囲まれた処理文を繰り返し実行し続けます。処理文を実行される前に、「条件判定式」の判定が行われるので、

while(true){ } と書くとき、無限ループになってしまいます。逆に、条件判定式の値が最初に false になると、ループは一度も実行されません。

### ●配列

配列とは、変数に添え字を用いて、複数のデータに順に格納しておくデータの列です。このデータの列を構成する要素を配列要素といいます。単純変数には 1 つしか格納できませんが、配列の場合、つまり変数に添え字を用いた場合、1 つの変数に対して連続した複数の領域が用意されます。

配列名は、変数名と同様、アルファベットで始まります。例えば、

```
var name = new Array(10);
```

と書くと、name は単純変数ではなく、name[0], name[1], name[2], … name[9] の 10 個の要素から構成された配列を表します。

配列の最初の要素の番号は 0 から始まります。したがって、上記の Array(10) は要素番号が 0 から 9 までの 10 個の要素から成り立っていることを表します。1 から始まるものと勘違いをしないように、ご注意ください。とはいうものの、JavaScript 言語は、他の言語と違って、配列の要素数に対して、非常に柔軟に対応しています。例えば、宣言文中の配列要素数を省略して、

```
var name = new Array();
```

と書くこともできます。

>>配列の続き

配列要素を直接宣言文に書き込む方法

```
var name = new Array(“鈴木”,“佐藤”,“伊藤”,“山崎”,“河野”,“山本”,“山田“,“後藤”,“近藤“,“田中”);
```

代入する方法

```
var name = new Array(10);  
name[0] = “鈴木”;  
name[1] = “佐藤”;  
name[2] = “伊藤”;  
name[3] = “山崎”;  
name[4] = “河野”;  
name[5] = “山本”;  
name[6] = “山田”;  
name[7] = “後藤”;  
name[8] = “近藤”;  
name[9] = “田中”;
```

と書くことができます。

次によく使う例として、

- ・チェックボックスの値取得
- ・ラジオボタンの値取得
- ・セレクトボックスの値取得

を考えてみましょう。

## ■ チェックボックスの値取得

☐ チェック項目1

☒ チェック項目2

ボタン押下後にチェックされているか確認

チェック項目1がチェックされていません。

チェック項目2がチェックされています。

```
<html>
<head>
  <title>チェックボックスの値取得 </title>
  <script language="javascript">
    function onClick() {
      var checkbox1 = document.getElementById("checkbox1");
      var checkbox2 = document.getElementById("checkbox2");

      var target = document.getElementById("output");
      if (checkbox1.checked == true) {
        target.innerHTML = "チェック項目 1 がチェックされています。<br/>";
      } else {
        target.innerHTML = "チェック項目 1 がチェックされていません。<br/>";
      }
      if (checkbox2.checked == true) {
        target.innerHTML += "チェック項目 2 がチェックされています。<br/>";
      } else {
        target.innerHTML += "チェック項目 2 がチェックされていません。<br/>";
      }
    }
  </script>
</head>
<body>
  <form name="formname">
    <input id="checkbox1" type="checkbox" />チェック項目 1<br/>
    <input id="checkbox2" type="checkbox" />チェック項目 2<br/>
    <input type="button" value="確認" onclick="onClick();" />

    <!-- 下記のエリアにメッセージを表示します -->
    <div id="output"></div>
  </form>
</body>
</html>
```

## ■ ラジオボタンの値取得

☐ 項目 1 ☐ 項目 2 ☒ 項目 3

このページの内容  
value値は3です。項目3が選択されています。

alertが表示される

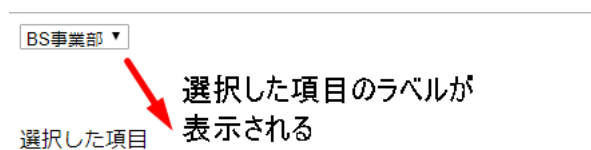
```
<html>
<head>
  <title>ラジオボタンの値取得</title>
<script language="javascript">
  function getRadioValue(name){
    //ラジオボタンオブジェクトを取得する
    var radios = document.getElementsByName(name);
    //取得したラジオボタンオブジェクトから選択されたものを探し出す
    var result;
    for(var i=0; i<radios.length; i++){
      if (radios[i].checked) {
        //選択されたラジオボタンの value 値を取得する
        result = radios[i].value;
        break;
      }
    }
    //value 値を元にアラートを出力を変更する
    switch(result){
      case "1":
        alert("value 値は" + result + "です。項目 1 が選択されています。");
        break;
      case "2":
        alert("value 値は" + result + "です。項目 2 が選択されています。");
        break;
      case "3":
        alert("value 値は" + result + "です。項目 3 が選択されています。");
        break;
    }
  }
</script>    <!-- 続きは次のページ -->
```

```
</head>
<body>
<form name="formname">

  <input type="radio" name="group1" value="1">項目 1
  <input type="radio" name="group1" value="2">項目 2
  <input type="radio" name="group1" value="3">項目 3
  <input type="button" value="ボタ ン" onclick="getRadioValue('group1');">

</form>
</body>
</html>
```

## ■ セレクトボックスの値取得



ここでは選択された項目のテキスト値、つまりラベル文字、を取得する方法についてみていきます。まず、`selectedIndex` プロパティを用いてセレクトメニューの番号を取得し、それからそのオプションに対応するテキストを取得します。

## >> セレクトボックスの値取得の続き

```
<html>
<head>
<title>セレクトボックスの値取得</title>
<script language="javascript">
    function getSelTerm(){

        k = document.formname.selectname.selectedIndex;
        selterm = document.formname.selectname.options[k].text;
        //alert("選択された項目は："+selterm);

        document.getElementById("span1").textContent = selterm;
    }

</script>
</head>
<body>
<form name="formname">
<select name="selectname" onChange="getSelTerm()">
    <option value="opt1">BS 事業部</option>
    <option value="opt2">SI 課</option>
    <option value="opt3">NI 課</option>
    <option value="opt4">〇〇課</option>
</select>
</form>
<br>
<p>選択した項目 <span id="span1"></span></p>
</body>
</html>
```

JavaScript について少しでも理解が深まったでしょうか。今後プログラマーとして知識を積み重ねていってください。「はじめに知っておきたい JavaScript 基礎編」の解説は以上となります。

参考文献：

■ ビラール イリヤス『CALL 教材開発テクニック－インターネットをベースにした教材作成の裏技』三恵社（2004 年）