イベント処理の指定
このページでは、イベントに処理を指定する方法を見ていきます。

HTML タグの属性に指定する
今まで、HTML の属性(onclick や onload 等) に JavaScript の関数を指定していました。以下のような感じです。最も簡単な指定法です。

```
<input type="button" value="押して下さい" onclick="fnc()">
```

DOM 要素に指定する
イベント処理は HTML タグの属性に記述するより、 JavaScript から指定した方が保守管理が楽になります。 それで今度は、JavaScript を使って id 属性を付けた要素にイベントを指定する方法を見てみましょう。
以下のスクリプトでは、id 属性を付けた div タグに、onmouseover イベントと onmouseout イベントを指定しました。 文字上にカーソルを乗せると文字列が書き換えられ、 カーソルを外すとさらに文字列が変化します。

```
<div id="dom">ここにイベントを指定します</div>

<script>
var obj = document.getElementById("dom");

//マウスカーソルが乗った時の処理
function fnc1()
{
  obj.innerHTML = "マウスカーソルが乗りました";
}

//マウスカーソルが外れた時の処理
function fnc2()
{
  obj.innerHTML = "マウスカーソルが外れました";
}

//イベントに関数を指定する
obj.onmouseover = fnc1;
obj.onmouseout  = fnc2;
```

</script>
＜サンプル＞文字列にマウスカーソルを合わせると、文字が変化します
ここにイベントを指定します

スクリプトの説明
では上記スクリプトを詳しく見ていきましょう。
div タグに id 属性を付ける
ここにイベントを指定します
最初に div タグに id 属性を付け、属性値を「dom」にしました。 こうすることで JavaScript を使って操作することができます。 確認したいのは、onmouseover や onmouseout といったイベント系の属性が記入されていない点です。 これらは JavaScript で指定します。
属性値を持つ要素の取得
var obj = document.getElementById("dom");
続いて script タグ内を見ていきます。最初に getElementById を使って、id 属性値 dom を持つ要素を取得し、変数 obj に格納します。
関数 fnc1 と関数 fnc2
function fnc1{ obj.innerHTML= "" }
function fnc2{ obj.innerHTML= "" }
次に、二つの関数が出てきます。関数 fnc1 はマウスカーソルが乗った時用で、 関数 func2 はカーソルが外れた時にためのものです。 innerHTML を使ってタグ内の文字列を書き換えます。
イベントに関数を指定する
obj.onmouseover = fnc1;
obj.onmouseout  = fnc2;
最後に、イベントに関数を指定しました。ここで右辺に注目してください。 指定するのは関数そのもので、「fnc1()」のように括弧付で書いてはいけません。 ここが上記の HTML 属性での指定と違うので、気を付けて下さい。

次のページでは、addEventListener()を利用したイベントの登録方法を見ていきます。

addEventListener()
前のページで見た方法では、同じイベントに対して処理を追加するということができません。 一番最後に指定したものが適用されます。処理を追加登録する場合は、addEventListener()を利用します。

DOM 要素にイベントを指定すると上書きになる

下のサンプルをご覧下さい。HTML のタグの onclick 属性で関数 fnc1 を呼び出すようにしています。 そこに JavaScript を使って onclick イベントに関数 fnc2 を呼び出すよう記述しました。 果たして結果はどうなるでしょうか？

```
<form>
<input type="button" id="ev" value="イベント発動" onclick="fnc1()">
</form>

<script>
var obj = document.getElementById("ev");

function fnc1(){ alert("関数１が呼び出されました"); }
function fnc2(){ alert("関数２が呼び出されました"); }

obj.onclick = fnc2;
</script>
```

サンプルのボタンを押すと分かるように、関数 fnc2 だけ呼び出されました。 つまり同じ要素の同じイベントに対しては、処理は追加ではなく上書きされることになります。
では JavaScript でイベント処理を追加するにはどうすればいいでしょうか。
addEventListener()を使えば追加が可能です。
イベント処理には、イベントハンドラとイベントリスナの２種類があります。 イベントハンドラは１つのイベントについて１つだけしか指定できませんが、 イベントリスナは複数設定できます。

addEventListener()によるイベントリスナの指定
最初に addEventListener()は Internet Explorer 8 以前の IE では動かないことを記述しておきます。 これらに対応させる必要がある方は attachEvent()を使いますが、当サイトでは解説は割愛致します。
addEventListener()を使ってイベントリスナを追加するには、以下のようにします。
DOM 要素.addEventListener( イベント , 処理 , false )
１番目の引数には、追加するイベントを指定します。
２つ目の引数は、主に関数を指定、若しくは直接関数を記述します。
３つ目はイベントの伝播形式を指定しますが、取り敢えず false にします。
実際にサンプルスクリプトを見てみましょう。 以下のサンプルでは、最初にボタンの onclick 属性で関数 fnc1 を呼び出すようにしています。 次に addEventListener()で関数 fnc2 を追加しました。 さらにもう一つ関数を追加しています。

```
<form>
<input type="button" id="ev2" value="イベント発動" onclick="fnc1()">
</form>

<script>
var obj2 = document.getElementById("ev2");

//既存の関数をイベントリスナに登録する
obj2.addEventListener( "click" , fnc2 , false );

//直接関数を記述して登録する
obj2.addEventListener( "click" , function () {
  alert("イベント３の追加")
} , false );
</script>
```

上のボタンを押すと、３つのアラートが表示されると思います。 つまり click イベントにイベントリスナを追加することができたのです。

関数をイベントリスナとして登録する

```
obj2.addEventListener( "click" , fnc2 , false );
```

既出の関数を追加するのは、第２引数に関数を指定するだけでＯＫです。 第一引数のイベントは「onclick」ではなく「click」であることに気をつけてください。

関数を直接記述する

```
obj2.addEventListener( "click" , function () {
  alert("イベント３の追加")
} , false );
```

上のように第２引数に直接関数を記述することができます。 関数名は必要ないので記述していません。 上記のサンプルでは alert()でダイアログを出すようにしています。

もし上のサンプルで、関数形式にせず直接 alert()を書くとどうなるでしょうか？ ボタンを押す前、ページが読み込まれた時点でアラートダイアログが表示されてしまいます。