

HTML MANIAX

最終回

クッキーを活用したページデザイン

今回で連載は最終回。先月紹介したクッキーを処理する汎用スクリプトcookie.jsの応用例で締めくくりにしよう。クッキーの処理は少々ややこしいが、外部jsファイルにスクリプトをまとめてしまえば手軽に扱える。クッキーの詳細については先月号を参照してもらうことにして、今月は具体的にどう使えば役に立つのかを考えてみよう。

文:佐藤和人

HTML MANIAXで紹介したテクニックは <http://internet.impress.co.jp/maniax/> でソースを公開!



cookie.jsでらくらくクッキー管理

cookie.jsの使い方

- [1] script要素でcookie.jsを埋め込む。

```
<script type="text/javascript" src="cookie.js">
</script>
```

- [2] クッキーを書き込むときは、関数setCookieを使う。
setCookie(データ名, 値の配列, 保存する日数, パス);
例)

```
var name = "kazuto";
var mail = "kazuto@mvi.biglobe.ne.jp";
setCookie("data", new Array(name, mail),
  30, "/maniacx/");
```

- [3] クッキーを読み出すときは、関数getCookieを使う。
値の配列 = getCookie(データ名);
例)

```
var values = getCookie("data");
var name = values[0];
var mail = values[1];
```

先月作成したcookie.jsは、クッキーの処理に必要な2つの関数をまとめたシンプルなスクリプトだ。インターネットマガジンのホームページから入手して、自分のHTMLに埋め込めば、すぐに使えるようになっている。

クッキーを保存するときは関数setCookieにデータ名、保存するデータの配列、クッキーを保存する日数、クッキーを読み取れるパスを指定する。今月のサンプルソースでは、パスに「/maniacx/」を指定しているが、自分のサイトのURLに合わせて書き換えてほしい。たとえばURLがhttp://www.provider.co.jp/taro/なら、「/taro/」を指定するという具合だ。クッキーを読み込むときは、関数getCookieにデータ名を渡すと保存された配列が返される。

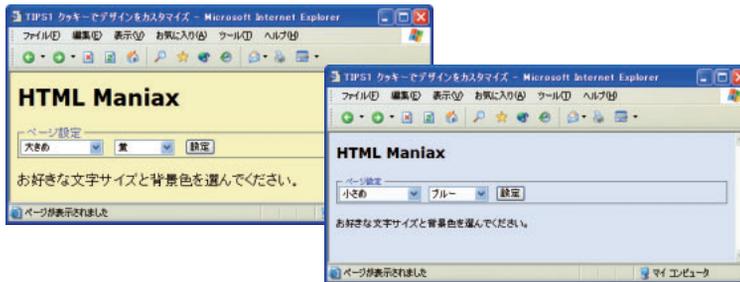
なお、今月のcookie.jsの中身は、先月のものをちょっと修正したものだということにご注意。使い方は同じだ。

[クッキーでデザインをカスタマイズ]

最初に cookie.js を使った簡単なサンプルとして、訪問者にページのフォントサイズと背景色を選んでもらう機能を作ってみよう。大きめの字で文章を読みたい人や、色にこだわりのある人に向けたサービスだ。



フォントサイズと背景色を選ばせる



tips1-1.html(スクリプト部分)

```
function setStyle(form) {
    if(!document.getElementById) return;
    var size = form.elements["size"].value;
    var color = form.elements["color"].value;
    setCookie("setting", new Array(size, color),
        30, "/maniaux/");
    location.reload();
}
function init() {
    if(!document.getElementById) return;
    var values = getCookie("setting");
    var form = document.forms["f1"];
    if(values[0]) {
        document.body.style.fontSize = values[0];
        form.elements["size"].value = values[0];
    }
    if(values[1]) {
        document.body.style.backgroundColor = values[1];
        form.elements["color"].value = values[1];
    }
}
```

tips1-1.html(フォーム部分)

```
<form name="f1" action="tips1-1.html"
    onsubmit="setStyle(this); return false;">
<fieldset><legend>ページ設定</legend>
<p><select name="size">
<option value=""> 文字の大きさ</option>
<option value="0.8em">小さめ</option>
<option value="1em">ふつう</option>
<option value="1.2em">大きめ</option>
</select>
(後略)
```

まずはソース のように、フォントサイズと背景色を選んでもらうためのフォームを作る。ここではselect要素でメニューを作り、option要素のvalue属性にCSS用のフォントサイズを入れておいた。ソース では省略したが、同様にselect要素とoption要素で背景色のメニューを作り、value属性に「#FFC」や「#EEF」などの色を設定してある。フォームが送信されると(onsubmit イベント)、関数 setStyle が呼び出される。

ソース の関数 setStyle では、フォームのメニューから選ばれたフォントサイズと背景色を取り出し、cookie.js の関数 setCookie を使ってクッキーに保存する。クッキーの名前は「 setting 」とし、サイズと背景色のデータを配列にして渡している。location.reload() は、ページを再読み込みして、訪問者の選択を反映させるものだ。

ソース の2つ目の関数 init は、body 要素の onload イベントで呼び出されるもので、ページが読み込まれたときにクッキーのデータをもとにフォントサイズと背景色を設定する。cookie.js の関数 getCookie でデータを配列 values に取り出すと、配列要素の1目がフォントサイズに、2目が背景色になっている。document.body.style.fontSize(または backgroundColor) に代入して、body 要素のフォントサイズ(または 背景色) を切り替える仕組みだ。

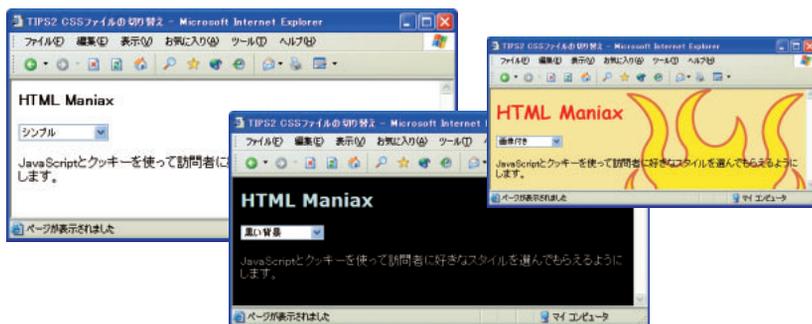
このスクリプトが対応しているのは、ウインドウズ版 IE 5以降、マッキントッシュ版 IE 5、ネットスケープ6、7、Opera 7だ。ソース の関数 init の中ではクッキーの設定をフォームに戻しているが、この部分はマッキントッシュ版 IE 5では働かない(常に最初のoption要素が選択されてしまう)。

[CSS ファイルの切り替え]

ネットスケープやOpera 7には、ページ作者が用意した複数の外部CSSファイルをメニューから切り替える機能がある(連載第7回参照)。クッキーとJavaScriptを利用して、この楽しい機能をIEでも実現させてみよう。



CSSの究極ワザはこれだ!



tips2-1.htm(link 要素の記述)

```
<link rel="stylesheet" type="text/css"
      href="maniax1.css" title="シンプル">
<link rel="alternate stylesheet" type="text/css"
      href="maniax2.css" title="黒い背景">
<link rel="alternate stylesheet" type="text/css"
      href="maniax3.css" title="画像付き">
```

tips2-1.htm(スクリプト部分)

```
function saveStyle(select) {
    if(!document.getElementById) return;
    var cssfile = select.value;
    if(cssfile) {
        setCookie("css", new Array(cssfile),
            30, "/maniax/");
        location.reload();
    }
}
function init() {
    if(!document.getElementById) return;
    var values = getCookie("css");
    if(!values[0]) return;
    var links = document.getElementsByTagName("link");
    for(i = 0; i < links.length; i++) {
        if(links[i].href.indexOf(values[0]) >= 0) { A
            links[i].rel = "stylesheet";
            links[i].disabled = false;
        }
        else if(links[i].rel == "stylesheet") { B
            links[i].disabled = true;
        }
    }
}
(後略)
```

外部CSSファイルの切り替え機能を使うには、ソースのように、複数のlink要素に別々のCSSファイルを指定する。第1候補のCSSはrel属性を「stylesheet」とし、2番目以降の候補は「alternate stylesheet」とする。どのlink要素にも必ずtitle属性で名前を付けておくこと。これだけでもネットスケープやOpera 7では、ブラウザのメニューからCSSファイルを選んでデザインを切り替えられるようになる。しかし、IEでは役に立たないし、ネットスケープやOpera 7でも再びページを訪れたときには第1候補のCSSファイルに戻ってしまう。そこで、選んだCSSファイルをクッキーに保存させてみよう。

まず、前ページと同様にフォームを作り、select要素とoption要素でデザインを選ぶメニューを作る(ソースは省略)。option要素のvalue属性にはCSSファイル名を入れておく。select要素のonchangeイベントからは、ソースの関数saveStyleを呼び出して、CSSファイル名をクッキーに保存する。

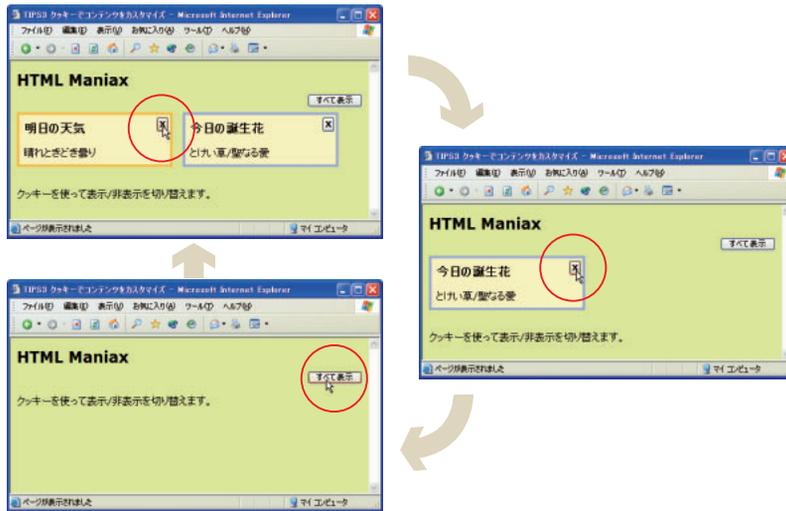
ソースの関数initは、ページが読み込まれたときにクッキーをもとにして外部CSSを選択するものだ。getElementsByNameでlink要素をリストアップして1つずつ調べ、href属性にクッキーのCSSファイルが含まれていたらソースのAでdisabledプロパティで有効にする。デフォルトで選ばれたCSSだったら(ソースのB)無効にする。「後略」の部分では、フォームのメニューをクッキーのデータに合わせる処理をしている。

ソースのスクリプトを外部jsファイルにして共有すれば、選ばれたCSSファイルのデザインをほかのページに反映させることも可能になる。

[クッキーでコンテンツをカスタマイズ]

クッキーの応用例としてよく知られているのは、My Yahoo!のようにユーザーがページをカスタマイズできるサービスだ。最後にcookie.jsを応用して、好きなコンテンツだけを表示させるスクリプトを紹介しよう。

 表示 / 非表示を自由に選べる



tips3-1.html(スクリプト部分)

```
function closeBlock(id) {
    if(!document.getElementById) return;
    var values = getCookie("blocks");
    values[values.length] = id;
    setCookie("blocks", values, 30, "/maniax/");
    location.reload();
}
function init() {
    if(!document.getElementById) return;
    var values = getCookie("blocks");
    for(i = 0; i < values.length; i++) {
        var div = document.getElementById(values[i]);
        if(div) div.style.display = "none";
    }
}
function showAll() {
    if(!document.getElementById) return;
    setCookie("blocks", new Array(), 30, "/maniax/");
    location.reload();
}
```

tips3-1.html(ブロックを閉じるボタン)

```
<div class="block" id="block1">
<p class="close"><input type="button" value="X"
onclick="closeBlock('block1');"></p>
```

このサンプルは、四角い枠で囲まれたブロックの中の「×」ボタンをクリックすると、そのブロックの表示が消えるというものだ。初めて訪れた人向けに注意書きを表示させ、常連の人はその表示を消して使ってもらおう、といった応用が考えられるだろう。消したブロックを表示させたいければ、「すべて表示」ボタンを押せばいい。

まず、ソース のように表示を切り替えたいブロックをdiv要素で囲んでID名(「block1」、「block2」など)を付ける。「×」ボタンはinput要素で作成し、onclickイベントで関数closeBlockを呼び出す。引数にはdiv要素のID名を渡す。このサンプルでは表示を切り替えられるブロックは2つだけだが、ID名を持つdiv要素と「×」ボタンを追加すれば、ブロックはいくらでも増やせるようになっている。

ソース の関数closeBlockでは、getCookieでクッキーに保存してある配列を取り出し、渡されたID名を配列に追加してクッキーに保存し直す。関数initはページが読み込まれたときに呼び出されるものだ。クッキーからID名の配列を取り出して、ID名に対応するdiv要素を1つずつgetElementByIdで探し、非表示にする(style.displayに「none」を代入する)。

ソース の3つ目の関数showAllは「すべて表示」ボタンから呼び出されるもので、「blocks」という名前のクッキーの中身を空にする。すると、どのブロックの表示も消されないの、すべてが表示されて見えるというわけだ。

今回はcookie.jsを使ったサンプルの紹介はこれだけだが、いくらでも応用は考えられる。自分のサイトでクッキーの使い方をあれこれ工夫してみよう。



[インターネットマガジン バックナンバーアーカイブ] ご利用上の注意

このPDFファイルは、株式会社インプレスR&D(株式会社インプレスから分割)が1994年～2006年まで発行した月刊誌『インターネットマガジン』の誌面をPDF化し、「インターネットマガジン バックナンバーアーカイブ」として以下のウェブサイト「All-in-One INTERNET magazine 2.0」で公開しているものです。

<http://i.impressRD.jp/bn>

このファイルをご利用いただくにあたり、下記の注意事項を必ずお読みください。

- 記載されている内容(技術解説、URL、団体・企業名、商品名、価格、プレゼント募集、アンケートなど)は発行当時のものです。
- 収録されている内容は著作権法上の保護を受けています。著作権はそれぞれの記事の著作者(執筆者、写真の撮影者、イラストの作成者、編集部など)が保持しています。
- 著作者から許諾が得られなかった著作物は収録されていない場合があります。
- このファイルやその内容を改変したり、商用を目的として再利用することはできません。あくまで個人や企業の非商用利用での閲覧、複製、送信に限られます。
- 収録されている内容を何らかの媒体に引用としてご利用する際は、出典として媒体名および月号、該当ページ番号、発行元(株式会社インプレス R&D)、コピーライトなどの情報をご明記ください。
- オリジナルの雑誌の発行時点では、株式会社インプレス R&D(当時は株式会社インプレス)と著作権者は内容が正確なものであるように最大限に努めましたが、すべての情報が完全に正確であることは保証できません。このファイルの内容に起因する直接のおよび間接的な損害に対して、一切の責任を負いません。お客様個人の責任においてご利用ください。

このファイルに関するお問い合わせ先

株式会社インプレスR&D

All-in-One INTERNET magazine 編集部

im-info@impress.co.jp