

ソフトウェア開発 第 13 回目授業

平野 照比古

2014/12/12

jQuery とは

jQuery の開発元¹には次のように書かれている。

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

¹jquery.com

jQuery の目的

- JavaScript のライブラリー
- これまでに解説してきた JavaScript の使い方を簡単にすることを目的
- jQuery には ver. 1 のものと ver. 2 の系列が提供

ver. 1 と ver. 2 の違い

jQuery 2.x

jQuery 2.x has the same API as jQuery 1.x, but does not support Internet Explorer 6, 7, or 8. All the notes in the jQuery 1.9 Upgrade Guide apply here as well. Since IE 8 is still relatively common, we recommend using the 1.x version unless you are certain no IE 6/7/8 users are visiting the site. Please read the 2.0 release notes carefully.

ver. 2 では古い Internet Explorer の対応を打ち切った

min ファイルの意味

配布されているライブラリーはコメントなどがそのまま入っているものと、コメントを取り除き、変数名などを短いものに置き換えて、ファイルサイズを小さくしたものの2種類がある。

jQuery オブジェクト

- jQuery ライブラリーは jQuery() というグローバル関数を一つだけ定義
- この関数の短縮形として、\$というグローバル関数も定義
- jQuery() 関数はコンストラクタではない。
- jQuery 関数は引数の与え方により動作が異なる
- 戻り値として jQuery オブジェクトと呼ばれる DOM の要素群を返す
- このオブジェクトには多数のメソッドが定義されていて、これらの要素群を操作可能

jQuery 関数の引数の型 (1)

jQuery() 関数の呼び出し方は引数の型により返される要素群が異なる。

- 引数が (拡張された)CSS セレクタ形式の場合(機能 1)
 - CSS セレクタにマッチした要素群を返す。
 - 省略可能な 2 番目の引数として要素や jQuery オブジェクトを指定した場合には、その要素の子要素からマッチしたものを返す。
 - この形はすでに解説した DOM のメソッド `querySelectorAll()` と似ている。
- 引数が要素や Document オブジェクトなどの場合(機能 2)
与えられた要素を jQuery オブジェクトに変換する。

jQuery 関数の引数の型 (2)

- 引数として HTML テキストを渡す場合(機能 3)

- テキストで表される要素を作成し、この要素を表す jQuery オブジェクトを返す。createElement() メソッドに相当する。
- 省略可能な 2 番目の引数は属性を定義するものであり、オブジェクトリテラルの形式で与える。

- 引数として関数を渡す場合(機能 4)

引数の関数はドキュメントがロードされ、DOM が操作で可能になった時に実行される。

jQuery オブジェクトのメソッドの基本

jQuery オブジェクトに対する多くの処理は HTML の属性や CSS スタイルの値を設定したり、読み出したりすること

- これらのメソッドに対してセッターとゲッターを同じメソッドを使う。メソッドに値を与えるとセッターになり、ないとゲッターになる。
- セッターとして使った場合は戻り値が jQuery オブジェクトとなるので、メソッドチェーンが使える。
- ゲッターとして使った場合は要素群の最初の要素だけ問い合わせる。

HTML 属性の取得と設定

`attr()` メソッドは HTML 属性用の jQuery のゲッター/セッターである。

- 属性名だけを引数に与えるとゲッターとなる。
- 属性名と値の 2 つを与えるとセッターになる。
- 引数にオブジェクトリテラルを与えると複数の属性を一時に設定できる。
- 属性を取り除く `removeAttr()` もある。

CSS 属性の取得と設定

`css()` メソッドは CSS のスタイルを設定する。

- CSS スタイル名は元来の CSS スタイル名（ハイフン付）でも JavaScript のキャメル形式でも問い合わせ、設定が可能である。
- 戻り値は単位を含めて文字列で返される。

HTML フォーム要素の値の取得と設定

- `val()` は HTML フォーム要素の `value` 属性の値の設定や取得が可能
- これにより、`<select>`要素の選択された値を得ることなどが可能

要素のコンテンツの取得と設定

- `text()` と `html()` メソッドはそれぞれ要素のコンテンツを通常のテキストまたは HTML 形式で返す。
- 引数がある場合には、既存のコンテンツを置き換える。

DOM 構造を変える

Table : ドキュメントの構造の変更するメソッド

<code>\$(T).method(C)</code>	<code>\$(C).method(T)</code>	機能
<code>append</code>	<code>appendTo</code>	要素 T の最後の子要素として C を付け加える
<code>prepend</code>	<code>prependTo</code>	要素 T の初めの子要素として C を付け加える
<code>before</code>	<code>insertBefore</code>	要素 T の直前の要素として C を付け加える
<code>after</code>	<code>insertAfter</code>	要素 T の直後の要素として C を付け加える
<code>replaceWith</code>	<code>replaceAll</code>	要素 T と C を置き換える

DOM の操作 (補足)

- 要素をコピーする `clone()`
- 、要素の子要素をすべて消す `empty()`
- 選択された要素 (とその子要素すべて) を削除する `remove()`

イベントハンドラーの登録

- イベントの種類ごとにメソッドが定義されている。
- 指定された jQuery オブジェクトが複数の場合にはそれぞれに対してイベントハンドラーが登録される。

イベントハンドラー登録のメソッド

<code>blur()</code>	<code>focusin()</code>	<code>mouseenter()</code>	<code>scroll()</code>
<code>change()</code>	<code>keydown()</code>	<code>mouseleave()</code>	<code>select()</code>
<code>click()</code>	<code>keypress()</code>	<code>mousemove()</code>	<code>submit()</code>
<code>dblclick()</code>	<code>keyup()</code>	<code>mouseover()</code>	<code>unload()</code>
<code>error()</code>	<code>load()</code>	<code>mouseup()</code>	
<code>focus()</code>	<code>mousedown()</code>	<code>resize()</code>	

イベントハンドラーの登録

- `hover()` : `mouseenter` イベントと `mouseleave` イベントに対するハンドラー を同時に登録できる。
- `toggle()` はクリックイベントに複数のイベントハンドラー を登録し、イベントが発生するごとに順番に呼び出す。

イベントハンドラー の登録解除

イベントハンドラー の登録解除は `unbind()` メソッドで行う。

- `removeEventListener()` と同様な形式として 1 番目の引数にイベントタイプ (文字列で与える)、2 番目の引数に登録した関数を与えるものがある
- 登録したイベントハンドラー には名前が必要

jQuery.ajax() 関数

- jQuery では Ajax の処理に関するいろいろな方法を提供
- jQuery.ajax() 関数を紹介
- この関数は引数にオブジェクトリテラルをとる。

jQuery.ajax() 関数の引数

Table : jQuery.ajax() 関数で利用できる属性 (一部))

属性名	説明
type	通信の種類。通常は"POST"または"GET"を指定
url	サーバーのアドレス
data	"GET"のときは URL の後に続けるデータ。 "POST"のときは null
dataType	戻り値のデータの型を指定する。"text"、"html"、 "script"、"json"、"xml"などがある
success	通信が正常終了したときに呼び出されるコールバック関数
error	通信が成功しなかったときに呼び出されるコールバック関数
timeout	タイムアウト時間をミリ秒単位で指定

今日は何の日—jQuery 版 (1)

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
5 <title>今日は何の日 (jQuery 版)</title>
6
7 <script type="text/ecmascript" src="jquery-1.11.2.min.js"></script>
```

6 行目はローカルに保存した jQuery ライブラリーを読み込んでいる。

今日は何の日—jQuery 版 (2)

```
8   <script type="text/ecmascript">
9   //
10  $(window).ready(function(){
11      function makeSelectNumber(from, to, prefix, suffix, id, parent){
12          var i, option;
13          var Select = $("&lt;select/&gt;", {"id":id});
14          if(parent) parent.append(Select);//$ (parent).append(Select);
15          for(i=from; i&lt;=to; i++) {
16              option = $("&lt;option/&gt;",{"value":i,"text":prefix+i+suffix});
17              Select.append(option);
18          //      $("&lt;option/&gt;",{"value":i,"text":prefix+i+suffix}).appendTo(Select);
19          }
20          return Select;
21      };</pre></div><div data-bbox="637 932 991 955" data-label="Page-Footer"><img alt="Navigation icons"/></div><div data-bbox="118 966 208 992" data-label="Page-Footer">平野 照比古</div><div data-bbox="376 966 618 992" data-label="Page-Footer">ソフトウェア開発第 13 回目授業</div><div data-bbox="793 966 890 992" data-label="Page-Footer">2014/12/12</div><div data-bbox="923 966 987 992" data-label="Page-Footer">23 / 35</div>
```

今日は何の日—jQuery 版 (2) 解説

- 10 行目の `$(window).ready()` はドキュメントが解釈されたときに引数の関数が呼び出されるイベントである。ここでの要素の参照は機能 2 を用いている。
- 11 行目から 21 行目は連続した番号を値に持つプルダウンメニューを作成する関数。仕様は以前と同じ
 - 13 行目では `<select>` 要素を作成し、同時に属性も設定している (機能 3)。
 - 15 行目から 19 行目で `<option>` 要素を定義し、`<select>` 要素の子要素にしている。なお、15 行目と 16 行目は `appendTo()` メソッドを用いると 18 行目のコメントアウトしてあるように記述することができる。

今日は何の日—jQuery 版 (3)

```
22 var i, today = new Date();
23 var y = today.getFullYear();
24 var m = today.getMonth();
25 var Form = $("#menu");
26 var Year = makeSelectNumber(2000,2020,"","年","year", Form);
27 var Month = makeSelectNumber(1,12,"","月","month", Form);
28 var Days = [];
29 for(i=28; i<=31; i++) {
30     Days[i] = makeSelectNumber(1,i,"","日","day");
31 }
32 Year.val(y); //$("#year").val(y);
33 Month.val(m+1); //$("#month").val(m+1);
34 var d = new Date(y, m+1,0).getDate();
35 Form.append(Days[d]);
36 $("#day").val(today.getDate());
```

今日は何の日—jQuery 版 (3) 解説

- 22 行目から 31 行目も依然とほとんど同じである。25 行目では機能 3 を用いて要素を参照している。
- 32 行目、33 行目と 36 行目はプルダウンメニューの初期値を本日に設定している。

今日は何の日—jQuery 版 (4)

```
37 var changePulldown = function(){  
38     var d2 = $("#day").val();  
39     d = new Date(Year.val(), Month.val(), 0).getDate();  
40     if( d != $("#day option").length) {  
41         $("#day").replaceWith(Days[d]);  
42         $("#day").val(Math.min($("#day option").length, d2));  
43     }  
44 }
```

今日は何の日—jQuery 版 (4) 解説

- jQuery のメソッドを用いていることをのぞけば 38 行目から 43 行目までは以前と同じである。
- 40 行目のセレクトは id が day(#day) の下にある<option>要素を選択し、その数を length で調べている。

今日は何の日—jQuery 版 (5)

```
45     jQuery.ajax({
46         type: "GET",
47         url:      "./aniversary.php",
48         data:      "month=" + Month.val() + "&day="+d2,
49         dataType: "text",
50         success : function(Data){
51             $("#details").text(Data);
52         },
53         error:function(){alert("error");}
54     });
55 };
56 Form.change(changePulldown);
57 changePulldown();
58 });
59 //]]>
60 </script>
```

今日は何の日—jQuery 版 (5) 解説

- 44 行目から 53 行目は Ajax の処理を定義
- ブラウザによる違いに対する対処や、XMLHttpRequest の処理部分が大幅に減っていて見やすいコードになっている

今日は何の日—jQuery 版 (6)

```
61 </head>
62 <body>
63   <form id="menu"></form>
64   <p id="details"> </p>
65 </body>
66 </html>
```

jQuery ファイルの違い

- jQuery のソースファイルには jQuery-<Version No>.js と jQuery-<Version No>.min.js の 2 種類が存在
- jQuery-<Version No>.min.js は jQuery-<Version No>.js からコメントや、空白を取り除き、変数名を短いものに置き換えるという操作を行って、ファイルサイズを小さくしたもの

jQuery 1.12.2.js のソース

- このリストでは詳細なコメントが付いていて比較的読みやすい。
- コメントを見るとクロスブラウザ対策が行われていることも見て取れる。
- ソースコードの最小化のテクニックとしてグローバルなオブジェクトを短い変数名で置き換える
- 冒頭の定義された関数の引数に、関数を渡している。その関数の仮引数は window となっている。これにより、開発時はわかりやすい変数名が使えるメリットがある。

短縮化のリスト

```
/*! jQuery v1.11.2 | (c) 2005, 2014 jQuery Foundation, Inc. | jquery.org/license */
!function(a,b){"object"==typeof module&&"object"==typeof module.exports?
module.exports=a.document?b(a,!0):
function(a){if(!a.document)
throw new Error("jQuery requires a window with a document");
return b(a)}:b(a)}("undefined"!=typeof window?window:this,function(a,b){
var c=[],d=c.slice,e=c.concat,f=c.push,g=c.indexOf,h={},
i=h.toString,j=h.hasOwnProperty,k={},l="1.11.2",
m=function(a,b){return new m.fn.init(a,b)},
n=/^\s\uFEFF\xA0+|^\s\uFEFF\xA0+$/g,
o=/^-ms-/,
p=/-([\da-z])/gi,
q=function(a,b){return b.toUpperCase()};
```

短縮化の方法

- 各関数内で変数名は 1 文字から始めている。
- 関数の仮引数も a から付け直している。
- JavaScript の固有の関数は当然のことながら変換されていない。
- このライブラリーは一つの関数を定義して、その場で実行している。
14 行目の `function()` の前に (がついている。
- 短縮化されたコードではこの部分が `!function()` となっている。関数オブジェクトを演算の対象とすることで実行する。
- そのほかにもキーワード `true` の代わりに `!0` としている。
- `if(){}else{}` 構文は `?:` に置き換えている。