3 5 TIPS — 上級編

遂に上級編のスタートです。ここから先のスクリプトは行数をみても100行以上あるスクリプトだらけで、読むだけでも大変です。そのかわりですが、今までと比べてウェブを便利にするアイディアで満載になっています。

勿論、今まで通り好みのデザインに変更したり、スクリプトが対応するサイトを追加したりと言った独自のカスタマイズを加える事はそう難しくありませんので、楽しみにしていてください！

3-5-1 特定のサイトへの最終訪問日時を表示する

ネットジャンキーなんていうマニアックな職業をやっていると、つい5分前に見たばかりのサイトの更新をいつの間にかまたチェックしているといった事が頻繁にあります。こんな事ばかりやっていると、全く仕事がはかどりません。ネットジャンキーは冗談ですが、インターネットを使う仕事をしていると、どうしても同じ所ばかりチェックしてしまう瞬間ができてしまいます。自分がどのくらい同じ所ばかり見ているのか知っておけば、少しはアクセスを抑えられるんじゃないかと妄想してしまいますが、このスクリプトはそんな希望を簡単に叶える事ができます。

次のスクリプトをShowVisitedTime.user.jsという名前をつけて保存してください。

ShowVisitedTime.user.js

ライセンス MITライセンス

```
// ==UserScript==
// @name        Show Visited Time
// @include     http://www.google.tld/
// @include     http://www.yahoo.co.jp/
// @version     1.00
// ==/UserScript==

const HISTORY = 3;
```



```
function main() {
  var key  = location.href;
  var date = eval(GM_getValue(key)) || new Array();

  showPanel(date);

  if(Dater.check(date[0])) {
    for (var i = HISTORY-1; i > 0; i--) {
      date[i] = date[i-1];
    }
    date[0] = Dater.create();

    GM_setValue(key, date.toSource());
  }
}

function showPanel(date) {
  var GM_infoPanel = document.createElement('div');

  with(GM_infoPanel.style) {
    bottom   = 0;
    right    = 0;
    padding  = '2px';
    opacity  = 0.7;
    fontSize = 'x-small';
    color    = '#000000';
    backgroundColor = '#EEEEEE';
    border   = '1px solid #C0C0C0';

    zIndex   = 100;
    position = 'fixed';
  }

  var msg = '';

  for (var i = 0, line; line = date[i]; i++) {
    msg += (i+1) + ':' + (Dater.get(line)) + '<br>';
  }

  GM_infoPanel.innerHTML = msg;
```

```
  document.body.appendChild(GM_infoPanel);
}

var Dater = {
  now: new Date(),
  cycle: 300 * 1000,

  create: function() {
    return (new Date()).getTime();
  },

  check: function(date) {
    return (!date || this.now > date + this.cycle)
  },

  get: function(date) {
    var base = new Date(date);
    var arr = new Array(
        base.getMonth()+1,
        base.getDate(),
        base.getHours(),
        base.getMinutes(),
        base.getSeconds()
        );

    for(var i in arr) {
      if(arr[i] < 10) arr[i] = '0' + arr[i];
    }

    return arr[0]+'-'+arr[1]+' '+
      arr[2]+':'+arr[3]+':'+arr[4];
  }
}

main();
```

解　説　このスクリプトをインストールすると、Included pagesに指定されたサイトへアクセスした際に、直近3件のアクセス日時が表示されるようになります（**図3-29**）。

図3-29

初期状態ではhttp://www.google.tld/とhttp://www.yaooh.co.jp/が指定されています。Included pagesの設定を、訪問日時を表示したいサイトに変更してください。

過去3回分のアクセスの日時が記録されます。ページをリロードしてもカウントされないように、一度記録すると次に記録されるのは300秒（5分）後となります。

データはFirefox本体へ保存しているので、このスクリプトをアンインストールする場合は「Also uninstall associated preferences」のチェックも忘れないようにしましょう（→**1**）。

Tips　初期状態では3回分のアクセスしか記録されませんが、ここは簡単に増やす事ができます。定数HISTORYの値がそのまま記録数になっているので、この数値を書き換えるだけです。好きな数値に書き換えればその分だけ記録される件数が増加します。

最低アクセス間隔になっている300秒と言うチェック秒数も簡単に変えられます。これは56行目にある次の部分で設定しています。

→**1**　1-2-7「ユーザスクリプトのアンインストール」（P.024）を参照のこと。

```
cycle: 300 * 1000,
```

JavaScript内部ではミリ秒単位で処理しているので、指定したい秒数に対して1,000倍の値を指定します。300秒にしたかったら、300×1,000が必要と言う事ですね。ここの値を増減させれば、アクセス間隔の範囲もその分だけ変化します。

今回はGM_setValue,GM_getValueが初めて登場しました。初めてなのに、いきなり複雑なデータを扱うためのeval技（→**2**）を使用しちゃっています。スクリプトをインストールしたばかりの時は保持されているデータも空なので、今回のように読み込むデータなければすぐ初期化をするようにしておくと良いでしょう。具体的には次の箇所です。

```
11: var key  = location.href;
12: var date = eval(GM_getValue(key)) || new Array();
```

今回は配列形式のデータを扱うのでこのようにしました。また、サイト毎に個別のデータを保存しておくため、キーの値にlocation.href（URL）を利用しています。

さて、過去数回のアクセス履歴はわかるようになりましたが、これだけだとまだまだ沢山アクセスしている感覚は出ません。せっかくなので、アクセス回数を記録するスクリプトもご用意しました。お好みに合わせて使い分けてください。

ShowVisitedCounter.user.js

ライセンス　MITライセンス

```
// ==UserScript==
// @name          Show Visited Counter
// @include       http://www.google.tld/
// @include       http://www.yahoo.co.jp/
// @version       1.00
// ==/UserScript==

const MSG_MAP = new Array('\u4ECA\u65E5: ', '\u6628\u65E5: ');
const HISTORY = 2;

function main() {
```

→**2**　1-4-2「GM_setValue, GM_getValue」の「複雑なデータ構造を扱う」（P.040）を参照のこと。


```
  var key  = location.href;
  var hash = eval(GM_getValue(key)) || false;

  if(!hash) {
    hash = {
      count: [0],
      time: Dater.create()
    };
  }

  if(Dater.check(hash.time)) {
    for(var i = HISTORY - 1; i > 0; i--) {
      hash.count[i] = (Dater.over(hash.time, i)) ? 0 : hash.count[i-1];
    }
    hash.count[0] = 0;
    hash.time = Dater.create();
  }

  hash.count[0] ++;
  showPanel(hash.count);

  GM_setValue(key, hash.toSource());
}

function showPanel(msg) {
  var GM_infoPanel = document.createElement('div');

  with(GM_infoPanel.style) {
    bottom   = 0;
    left     = 0;
    padding  = '2px';
    opacity  = 0.6;
    fontSize = 'x-small';
    color    = '#000000';
    backgroundColor = '#EEEEEE';
    border   = '1px solid #C0C0C0';

    zIndex   = 100;
    position = 'fixed';
  }
```

```
  var wt = new Array();
  for(var x in msg)
    wt.push((MSG_MAP[x]||'') + msg[x]);

  GM_infoPanel.innerHTML = wt.join('<br>');

  document.body.appendChild(GM_infoPanel);
}

var Dater = {
  now: new Date(),

  create: function() {
    var time = new Date();
    time.setHours(23, 59, 59);
    if(this.check(time))
      time.setDate(time.getDate()+1)
    return time;
  },

  check: function(time) {
    return (this.now > time);
  },

  over: function(time, day) {
    var nt = new Date(time);
    return (this.now > nt.setDate(nt.getDate()+(day||1)) );
  }
}

main();
```

解　説　このスクリプトをインストールすると、Included pagesに指定されたサイトへアクセスした回数が日別に記録されるようになります（**図3-30**）。

図3-30

初期状態ではhttp://www.google.tld/とhttp://www.yaooh.co.jp/が指定されています。Included pagesの設定を、訪問数を表示したいサイトに変更してください。

前日と当日の二日分のアクセス数が記録され、毎日00時になるとまた0からカウントをスタートします。

こちらのスクリプトもデータはFirefox本体へ保存しているので、アンインストールする場合は「Also uninstall associated preferences」のチェックも忘れないようにしましょう。

Tips　このスクリプトも、定数HISTORYの値を増やしてやるとその日数分アクセス数を記録していくようになります。「今日」や「昨日」の表記は配列MSG_MAPで指定しています。HISTORYをいじって記録する日数を増やしたら、このMSG_MAPにも対応する名称を追加しておいた方が見やすくなるでしょう。一応何も設定しないままでも動作はするので気にしなくても平気です。

00時になるとカウントがリセットされるようになっていますが、ここの時間も変更可能です。まず初めにリセットのタイミングの仕組みを説明します。初めてサイトにアクセスした時に、当日の23時59分59秒のデータを保存しま

す。二回目以降は最初に保存した日時と現在の日時を比較して現在の日時が上回っていたらカウントをリセット、その後に当日の23時59分59秒をセットします。これを延々繰り返せば、いつも00時にカウントがリセットされる理屈になります。さて、これでリセットするタイミングを変更する場合、どこを変えれば良いかわかりましたね。早速変更する所を探してみましょう。と言っても短いスクリプトなのですぐわかりますが、次の部分です。

```
var date = new Date();
date.setHours(23, 59, 59);
```

ここを好きな数値に変更すれば、その時間にリセットされるようになります。setHoursは引数が（時、分、秒、ミリ秒）なので、時間を設定するのが簡単です。多くの場合は、date.setHours(4, 0, 0);とかしておくと一日のサイクルがうまく表現できて良い感じになるのではないでしょうか。

3-5-2 特定のウェブサイトを画面内に呼び出す

機能が満載のウェブサービスを利用すると、最初のうちはヘルプ画面と行ったり来たり、なんて事がたまにあります。特に最近はJavaScriptがブーム気味で、キーボードショートカットが用意されているサービスも多いのですが、そんないちいち全部のショートカットは覚え切れません。こういった場合、操作説明の画面はサクサク見れた方が何かと便利なのですが、意外とそこまで考えて作られているサービスも少ないようです。そこでGreasemonkeyの登場です。サクサク見れるのが便利なら、サクサク見れるようにしちゃいましょうと言う事でご用意しました。

次のスクリプトをIncludeAnotherPage.user.jsという名前をつけて保存してください。

IncludeAnotherPage.user.js

ライセンス　パブリックドメイン

```
// ==UserScript==
// @name        Include Another Page
// @include     *
// @version     1.00
// ==/UserScript==
```