```
/**
 * Cookie plugin
 *
 * Copyright (c) 2006 Klaus Hartl (stilbuero.de)
 * Dual licensed under the MIT and GPL licenses:
 * http://www.opensource.org/licenses/mit-license.php
 * http://www.gnu.org/licenses/gpl.html
 *
 */

/**
 * Create a cookie with the given name and value and other optional parameters.
 *
 * @example $.cookie('the_cookie', 'the_value');
 * @desc Set the value of a cookie.
 * @example $.cookie('the_cookie', 'the_value', { expires: 7, path: '/', domain: 'jquery.com', secure: true });
 * @desc Create a cookie with all available options.
 * @example $.cookie('the_cookie', 'the_value');
 * @desc Create a session cookie.
 * @example $.cookie('the_cookie', null);
 * @desc Delete a cookie by passing null as value. Keep in mind that you have to use the same path and domain
 *       used when the cookie was set.
 *
 * @param String name The name of the cookie.
 * @param String value The value of the cookie.
 * @param Object options An object literal containing key/value pairs to provide optional cookie attributes.
 * @option Number|Date expires Either an integer specifying the expiration date from now on in days or a Date object.
 *                             If a negative value is specified (e.g. a date in the past), the cookie will be deleted.
 *                             If set to null or omitted, the cookie will be a session cookie and will not be retained
 *                             when the the browser exits.
 * @option String path The value of the path atribute of the cookie (default: path of page that created the cookie).
 * @option String domain The value of the domain attribute of the cookie (default: domain of page that created the cookie).
 * @option Boolean secure If true, the secure attribute of the cookie will be set and the cookie transmission will
 *                        require a secure protocol (like HTTPS).
 * @type undefined
 *
 * @name $.cookie
 * @cat Plugins/Cookie
 * @author Klaus Hartl/klaus.hartl@stilbuero.de
 */

/**
 * Get the value of a cookie with the given name.
 *
 * @example $.cookie('the_cookie');
 * @desc Get the value of a cookie.
 *
 * @param String name The name of the cookie.
 * @return The value of the cookie.
 * @type String
 *
 * @name $.cookie
 * @cat Plugins/Cookie
 * @author Klaus Hartl/klaus.hartl@stilbuero.de
 */
jQuery.cookie = function (name, value, options) {
  if (!name && !value) {
    /*
    Nothing sent in so return object containing all cookies
    Added by Don Myers 2010
    */
    var obj = {};
    var cookies = document.cookie.split(";");
    var cookie = '';
    var cname = '';
    var cvalue = '';

    for (i = 0; i < cookies.length; i++) {
      cookie = cookies[i].split("=");
      cname = jQuery.trim(cookie[0]);
      cvalue = (cookie.length > 1) ? decodeURIComponent(jQuery.trim(cookie[1])) : null;
      obj[cname] = cvalue;
    }

    return obj;
  }

  if (typeof value != 'undefined') { // name and value given, set cookie
    options = options || {};
    if (value === null) {
      value = '';
      options.expires = -1;
    }
    var expires = '';
    if (options.expires && (typeof options.expires == 'number' || options.expires.toUTCString)) {
      var date;
      if (typeof options.expires == 'number') {
        date = new Date();
        date.setTime(date.getTime() + (options.expires * 24 * 60 * 60 * 1000));
      }
      else {
        date = options.expires;
      }
      expires = '; expires=' + date.toUTCString(); // use expires attribute, max-age is not supported by IE
    }
    // CAUTION: Needed to parenthesize options.path and options.domain
    // in the following expressions, otherwise they evaluate to undefined
    // in the packed version for some reason...
    var path = options.path ? '; path=' + (options.path) : '';
    var domain = options.domain ? '; domain=' + (options.domain) : '';
    var secure = options.secure ? '; secure' : '';
    document.cookie = [name, '=', encodeURIComponent(value), expires, path, domain, secure].join('');
  } else { // only name given, get cookie
    var cookieValue = null;
    if (document.cookie && document.cookie !== '') {
      var cookies = document.cookie.split(';');
      for (var i = 0; i < cookies.length; i++) {
        var cookie = jQuery.trim(cookies[i]);
        // Does this cookie string begin with the name we want?
        if (cookie.substring(0, name.length + 1) == (name + '=')) {
          cookieValue = decodeURIComponent(cookie.substring(name.length + 1));
          break;
        }
      }
    }
    return cookieValue;
  }
};
```

```
/*
 * jQuery JSON Plugin
 * version: 2.1 (2009-08-14)
 *
 * This document is licensed as free software under the terms of the
 * MIT License: http://www.opensource.org/licenses/mit-license.php
 *
 * Brantley Harris wrote this plugin. It is based somewhat on the JSON.org
 * website's http://www.json.org/json2.js, which proclaims:
 * "NO WARRANTY EXPRESSED OR IMPLIED. USE AT YOUR OWN RISK.", a sentiment that
 * I uphold.
 *
 * It is also influenced heavily by MochiKit's serializeJSON, which is
 * copyrighted 2005 by Bob Ippolito.
 */

(function($) {
    /** jQuery.toJSON( json-serializble )
        Converts the given argument into a JSON respresentation.

        If an object has a "toJSON" function, that will be used to get the representation.
        Non-integer/string keys are skipped in the object, as are keys that point to a function.

        json-serializble:
            The *thing* to be converted.
     **/
    $.toJSON = function(o)
    {
        if (typeof(JSON) == 'object' && JSON.stringify)
            return JSON.stringify(o);

        var type = typeof(o);

        if (o === null)
            return "null";

        if (type == "undefined")
            return undefined;

        if (type == "number" || type == "boolean")
            return o + "";

        if (type == "string")
            return $.quoteString(o);

        if (type == 'object')
        {
            if (typeof o.toJSON == "function")
                return $.toJSON( o.toJSON() );

            if (o.constructor === Date)
            {
                var month = o.getUTCMonth() + 1;
                if (month < 10) month = '0' + month;

                var day = o.getUTCDate();
                if (day < 10) day = '0' + day;

                var year = o.getUTCFullYear();

                var hours = o.getUTCHours();
                if (hours < 10) hours = '0' + hours;

                var minutes = o.getUTCMinutes();
                if (minutes < 10) minutes = '0' + minutes;

                var seconds = o.getUTCSeconds();
                if (seconds < 10) seconds = '0' + seconds;

                var milli = o.getUTCMilliseconds();
                if (milli < 100) milli = '0' + milli;
                if (milli < 10) milli = '0' + milli;

                return '"' + year + '-' + month + '-' + day + 'T' +
                             hours + ':' + minutes + ':' + seconds +
                             '.' + milli + 'Z"';
            }

            if (o.constructor === Array)
            {
                var ret = [];
                for (var i = 0; i < o.length; i++)
```

```
                    ret.push( $.toJSON(o[i]) || "null" );

                return "[" + ret.join(",") + "]";
            }

            var pairs = [];
            for (var k in o) {
                var name;
                var type = typeof k;

                if (type == "number")
                    name = '"' + k + '"';
                else if (type == "string")
                    name = $.quoteString(k);
                else
                    continue;  //skip non-string or number keys

                if (typeof o[k] == "function")
                    continue;  //skip pairs where the value is a function.

                var val = $.toJSON(o[k]);

                pairs.push(name + ":" + val);
            }

            return "{" + pairs.join(", ") + "}";
        }
    };

    /** jQuery.evalJSON(src)
        Evaluates a given piece of json source.
     **/
    $.evalJSON = function(src)
    {
        if (typeof(JSON) == 'object' && JSON.parse)
            return JSON.parse(src);
        return eval("(" + src + ")");
    };

    /** jQuery.secureEvalJSON(src)
        Evals JSON in a way that is *more* secure.
    **/
    $.secureEvalJSON = function(src)
    {
        if (typeof(JSON) == 'object' && JSON.parse)
            return JSON.parse(src);

        var filtered = src;
        filtered = filtered.replace(/\\["\\\/bfnrtu]/g, '@');
        filtered = filtered.replace(/"[^"\\\n\r]*"|true|false|null|-?\d+(?:\.\d*)?(?:[eE][+\-]?\d+)?/g, ']');
        filtered = filtered.replace(/(?:^|:|,)(?:\s*\[)+/g, '');

        if (/^[\],:{}\s]*$/.test(filtered))
            return eval("(" + src + ")");
        else
            throw new SyntaxError("Error parsing JSON, source is not valid.");
    };

    /** jQuery.quoteString(string)
        Returns a string-repr of a string, escaping quotes intelligently.
        Mostly a support function for toJSON.

        Examples:
            >>> jQuery.quoteString("apple")
            "apple"

            >>> jQuery.quoteString('"Where are we going?", she asked.')
            "\"Where are we going?\", she asked."
     **/
    $.quoteString = function(string)
    {
        if (string.match(_escapeable))
        {
            return '"' + string.replace(_escapeable, function (a)
            {
                var c = _meta[a];
                if (typeof c === 'string') return c;
                c = a.charCodeAt();
                return '\\u00' + Math.floor(c / 16).toString(16) + (c % 16).toString(16);
            }) + '"';
        }
        return '"' + string + '"';
    };

    var _escapeable = /["\\\x00-\x1f\x7f-\x9f]/g;

    var _meta = {
        '\b': '\\b',
        '\t': '\\t',
        '\n': '\\n',
        '\f': '\\f',
        '\r': '\\r',
        '"' : '\\"',
        '\\': '\\\\'
    };
})(jQuery);
```

```
/*
Requires jQuery 1.4.3+

$.mvcController(name);
  load controller based on controller/method

$.mvcController(name,func);
  load controller based on controller/method
  run function or string when finished

*/
jQuery.mvcController = function (name,func) {
  var segs = name.split('/');
  var clas = segs[0];
  var meth = segs[1];
  var complete_name = mvc.controller_named + clas + mvc.method_named + meth; /* controller_index_method_index */
  jQuery.log(name,complete_name + '.' + mvc.constructor_named + '()',mvc.mvcpath + 'controllers/' + clas + '/' + meth + '.js');
  jQuery.getScript(mvc.mvcpath + 'controllers/' + clas + '/' + meth + '.js', function () {
    /* fire off construct */
    jQuery.exec(complete_name + '.' + mvc.constructor_named + '()');
    var ctrlr = window[complete_name];
    for (var elementid in ctrlr) {
      if (typeof(ctrlr[elementid]) === 'object') {
        for (var eventname in ctrlr[elementid]) {
          if (typeof(ctrlr[elementid][eventname]) === 'function') {
            /* data-mvc is now automagically attached via jquery 1.4.3+ */
            /* attach any events to matching classes and ids */
            jQuery('#' + elementid).mvcEvent(eventname,complete_name + '.' + elementid + '.' + eventname + '();');
            jQuery('.' + elementid).mvcEvent(eventname,complete_name + '.' + elementid + '.' + eventname + '();');
          }
        }
      }
    }
    /* fire off any when complete code sent in */
    jQuery.exec(func);
  });
};

/*
event = click,mouseover,change,keyup
func = indexController.action1.click() or func = function() { alert('welcome'); };
optional
data = json object
*/
jQuery.fn.mvcAction = function(event,func,data) {
  if (data) {
    jQuery(this).mvcData(data);
  }
  jQuery(this).mvcEvent(event,func);
};

/*
get view and send it to mvc.update to populate the html
required
name: name of the view file either absolute or relative /folder/file or folder/folder/file or file
optional
json: extra variables sent to view ajax call
update: true/false{default} update the DOM when id's or classes match
this will also try to load a controller for your new view

-- I like what jQuery is adding when it comes to a templating class not exactly sure how that fits thou...
*/
jQuery.mvcView = function (name, json, update) {
  var rtnjson = jQuery.mvcAjax('views/' + name, json, 'json', update);
  jQuery.mvcController(name);
  return rtnjson;
};

/*
load json properties into html based on matching selectors
matches on id,class,form element name
*/
jQuery.mvcUpdate = function (json) {
  if (json) {
    jQuery.exec(json.mvc_pre_view);
    for (var property in json) { /* we are only using strings or numbers */
      if (typeof(json[property]) === 'string' || typeof(json[property]) === 'number' || typeof(json[property]) === 'boolean') {
        var value = json[property];

        /* match classes & ids */
        jQuery('.' + property + ',#' + property).html(value);

        /* match any form element names */
        /* hidden field */
        if (jQuery('[name=' + property + ']').is('input:hidden')) {
          jQuery('input[name=' + property + ']').val(value);
        } /* input text */
        if (jQuery('[name=' + property + ']').is('input:text')) {
          jQuery('input[name=' + property + ']').val(value);
        } /* input textarea */
        if (jQuery('[name=' + property + ']').is('textarea')) {
          jQuery('textarea[name=' + property + ']').val(value);
        } /* input radio button */
        if (jQuery('[name=' + property + ']').is('input:radio')) {
          jQuery('input[name=' + property + '][value="' + value + '"]').attr('checked', true);
        } /* input checkbox */
        if (jQuery('[name=' + property + ']').is('input:checkbox')) {
          jQuery('input:checkbox[name=' + property + ']').attr('checked', (value === 1 || value === true));
        } /* input select */
        if (jQuery('[name=' + property + ']').is('select')) {
          jQuery('select[name=' + property + ']').val(value);
        }
      }

    }
    jQuery.exec(json.mvc_post_view);
  }
};

/*
Getters
return complete mvc data object
var value = $("#selector").mvcData(); (returns object)

return specific value
var value = $("#selector").mvcData("age"); (return value or undefinded)

Setters
$("#selector").mvcData({}); (clears it out)

$("#selector").mvcData("name","value");
*/
jQuery.fn.mvcData = function (name, value) {
    /* GET return Object if both empty */
    if (!name && !value) {
      return jQuery(this).data('mvc');
    }
    /* SET if name is a object */
    if (typeof(name) == 'object') {
      jQuery(this).data('mvc',name);
      return true;
    }
    /* GET if value is empty then they are asking for a property by name */
    if (!value) {
      var rtn;
      var temp = jQuery(this).data('mvc');
      if (temp) {
        rtn = temp[name];
      }
      return rtn;
    }
    if (name && value) {
      /* SET if name & value set */
      var temp = jQuery(this).data('mvc');
      if (temp) {
        temp[name] = value;
        jQuery(this).data('mvc',temp);
        return true;
      }
      return false;
    }
};

/*
Generic Event Set/Get
*/
jQuery.fn.mvcEvent = function (event, func) {
  if (typeof(event) == 'object' && !func) {
    /* SET clear all */
    jQuery(this).die().css('cursor', '');
    return true;
  }

  if (!event && !func) {
    /* GET return all events */
    var id = this.selector;
    var events = Array();

    jQuery.each(jQuery(document).data('events').live, function (name,value) {
      if (value.selector == id) {
        if (event !== '' && value.origType == event) {
          events.push(value.origType);
        } else if (!event)  {
          events.push(value.origType);
        }
      }
    });

    return events;
  }

  if (event && !func) {
    /* GET does event exist */
    var id = this.selector;
    var events = Array();
    jQuery.each(jQuery(document).data('events').live, function (name,value) {
      if (value.selector == id) {
        if (event !== '' && value.origType == event) {
          events.push(value.origType);
        } else if (!event)  {
          events.push(value.origType);
        }
      }
    });
    return (events.length !== 0);
  }

  if (event && typeof(func) == 'object') {
    /* SET clear function */
    jQuery(this).die(event);
    return true;
  }

  if (event && func) {
    /* SET event and function */
    jQuery(this).live(event,function () {
      mvc.event = jQuery(this);
      var dd = jQuery(this).data('mvcdata');
      mvc.data = (!dd) ? {} : dd;
      jQuery.exec(func);
    }).css('cursor', mvc.default_cursor);
    return true;
  }

}

/*
Used in model, view, form to get json with blocking
$.mvcAjax();
required
name = url of the file either /absolute/file.js or folder/file (based off of mvc folder)
optional
json addition json to send
type json{default} or any valid jQuery post dataType
update true/false{default} weither to send to the update function
*/
jQuery.mvcAjax = function (posturl, json, type, update) {
  /* NOTE: this is blocking ajax */
  json = (!json) ? {} : json;
  type = (!type) ? 'json' : type;

  json.mvc_posturl = posturl;
  json.mvc_type = type;
  json.mvc_update = update;
  json.mvc_timestamp = Number(new Date());

  if (jQuery.session_uid) {
    json.mvc_uuid = jQuery.session_uid();
    json.mvc_session_id = jQuery.session_id();
  }
  if (jQuery.cookie) {
    json.cookie = jQuery.cookie();
  }

  var rtnjson = {};
  jQuery.ajax({
    cache: false,
    type: 'POST',
    async: false,
    timeout: mvc.blocking_wait,
    url: mvc.ajax_url + posturl,
    dataType: type,
    data: json,
    success: function (ajaxjson) {
      rtnjson = ajaxjson;
    },
    error: function(jqXHR, textStatus, errorThrown) {
      jQuery.log(jqXHR,textStatus,errorThrown);
    }
  });

  /* blocking - continue */
  /* if update true then update the screen with returned json */
  update = (!update) ? mvc.auto_update_view : update;
  if (update) {
    jQuery.mvcUpdate(rtnjson);
  }
  return rtnjson;
};

/*
execute code
function or string
*/
jQuery.exec = function (code) {
  if (code !== '' || code !== undefined) {
    var func = (typeof(code) === 'function') ? code : new Function(code);
    try {
      func();
    } catch (err) {
      jQuery.log('jQuery.exec ERROR: ',err);
    }
  }
};

/*
client based redirect
*/
jQuery.redirect = function (url) {
  window.location.replace(url);
};

/* simple wrapper to make the syntax "look" cleaner */
/* var mine = Json; "looks" cleaner then var mine = {}; */
var data = Json = {};

/* does a element exist in the DOM?
another simple wrapper function
if ($("#selector).exists) {
  do something
}
*/
jQuery.fn.exists = function() {
  return jQuery(this).length > 0;
};

/* create a wrapper for $.postJSON(); - uses post instead of get as in $.getJSON(); */
jQuery.extend({
  postJSON: function( url, data, callback) {
    return jQuery.post(url, data, callback, 'json');
  }
});
```

```
/*
basic - add hidden
$("#form_id").mvcFormHidden('primary',23);
*/
jQuery.fn.mvcFormHidden = function (name, value) {
  return this.each(function () {
    var unique = mvc.auto_gen + name;
    if (jQuery('#' + unique).length > 0) {
      jQuery('#' + unique).attr('value', value);
    } else {
      jQuery('<input />').attr('type', 'hidden').attr('id', unique).attr('name', name).val(value).appendTo(this);
    }
  });
};

/*
basic
$("#form_id").mvcForm2Json();
advanced - add additional payload
$("#form_id").mvcForm2Json({'extra':'abc123'});
*/
jQuery.fn.mvcForm2Json = function(json) {
  json = (!json) ? {} : json;

  /* convert form to json object */
  jQuery.each(jQuery(this).serializeArray(), function () {
    if (json[this.name]) {
      if (!json[this.name].push) {
        json[this.name] = [json[this.name]];
      }
      json[this.name].push(this.value || '');
    } else {
      json[this.name] = this.value || '';
    }
  });
  json.mvc_post_selector = this.selector;
  json.mvc_timestamp = Number(new Date());
  json.mvc_url = mvc.self;
  json.mvc_application_folder = mvc.application_folder;

  return json;
};

/*
!needs work
Simplify Object to Record
*/
/*
jQuery.simplify = function(obj) {
  var data = {};
  for (var attr in obj) {
    if (typeof(obj[attr]) === 'boolean' || typeof(obj[attr]) === 'number' || typeof(obj[attr]) === 'string') {
      data[attr] = obj[attr];
    }
  }
  return data;
};
*/

/*
simple - send form to url from form's action attrubute + validation_url setting ie action="submit" url = submit_validate
$("#form_id").mvcFormValidate();
basic - send the form id as json to given url
$("#form_id").mvcFormValidate('url');
advanced - send the form id as json to given url submit the form on true (if mvc_model_valid = true)
$("#form_id").mvcFormValidate('url',true);
advanced - send the form id as json to given url submit the form on true (if mvc_model_valid = true) with extra payload
$("#form_id").mvcFormValidate('url',true,{'extra':'abc123'});
*/
jQuery.fn.mvcFormValidate = function (url, submit, update_view, json) {
/*
validate this against some back end php via ajax
pass back a json object with and array for the view [key] = value
and variable mvc_model_valid = true/false
if the json is not returned or invalid the form is considered invalid (valid = false)
other options include:
mvc_pre_view with valid javascript code to run
mvc_post_view with valid javascript code to run
*/
  submit = (!submit) ? mvc.validation_submit : submit;
  url = (!url) ? jQuery(this).attr('action') + mvc.validation_url : url;
  update_view = (!update_view) ? mvc.auto_update_view : update_view;

  mvc.ajax_responds = jQuery.mvcAjax(url,jQuery(this).mvcForm2Json(json),'json',true);

  if (mvc.ajax_responds !== null) {
    if (mvc.ajax_responds.mvc_model_valid === true && submit === true) {
      jQuery(this).unbind('submit').submit(); /* if returned false (no errors) then submit the form */
    }
    return (mvc.ajax_responds.mvc_model_valid === true) ? true : false;
  } else {
    return false;
  }
};

/*
basic - change the url of the form action
$("#form_id").mvcFormAction('new url');
*/
jQuery.fn.mvcFormAction = function (url) {
  return this.each(function () {
    jQuery(this).attr('action', url);
  });
};

/*
!needs work
wrapper to unbind a forms submit action
$("#formid").mvcUnbindSubmit();
*/
/*
jQuery.fn.mvcUnbindSubmit = function () {
  jQuery(this).submit(function() {
    return false;
  });
}
*/
```

```
models = {}; /* global model storage */

mvc.modelExists = function(name) {
  return (!models[name]) ? false : true;
}

/* jquery.mvcform.js needed to use this funciton */
jQuery.fn.mvcForm2Model = function(name,json) {
  models[name] = new model(name);
  jQuery.extend(models[name],$(this).mvcForm2Json(json));
};

function model(name,file) {
  this._model_name = name;
  this._model_filename = (!file) ? name : file;
};

model.prototype._load = function(extra) {
  this._action('load',extra);
};

model.prototype._save = function(extra) {
  this._action('save',extra);
}

model.prototype._remove = function(extra) {
  this._action('remove',extra);
}

model.prototype._action = function(action,extra) {
  extra = (!extra) ? {} : extra;
  /* is extra a where clause string */
  if (typeof(extra) != 'object') {
    var hold = extra;
    extra = {};
    extra._string = hold;
  }
  extra.mvc_model_action = action;
  extra.record = {};
  /* make a copy without the methods */
  for (var attr in this) {
    if (this.hasOwnProperty(attr)) {
      extra.record[attr] = this[attr];
    }
  }
  /* send it out via blocking ajax */
  var serverjson = jQuery.mvcAjax(mvc.model_url + this._model_filename + '.js',extra);

  /* merge it back with this object */
  jQuery.extend(this,serverjson);
}
```

```
/*
requires jquery 1.4.2+ and jquery.cookie.js
Written by Don Myers 2009
*/
jQuery.session_start = function() {
  /* setup if not alreay setup or push forward 1 year if it is */
  if (!jQuery.cookie("mvc_uuid")) {
    jQuery.cookie("mvc_uuid", (new Date().getTime()) + '-' + jQuery.uid(), { expires: 365, path: '/' });
  } else {
    jQuery.cookie("mvc_uuid", jQuery.cookie("mvc_uuid"), { expires: 365, path: '/' });
  }

  /* set the browser session */
  if (!jQuery.cookie("mvc_sessionid")) {
    jQuery.cookie("mvc_sessionid", jQuery.uid(), { path: '/' });
  }

  /* let's read and cache our session data */
  var jsontxt = jQuery.cookie("mvc_session");
  if (!jsontxt) {
    window.mvc_session = {};
  } else {
    window.mvc_session = jQuery.secureEvalJSON(jsontxt);
  }

  return jQuery.cookie("mvc_sessionid");
};

jQuery.session_uuid = function() {
  return jQuery.cookie("mvc_uuid");
};

jQuery.session_id = function() {
  return jQuery.cookie("mvc_sessionid");
};

jQuery.session_regenerate_id = function(delete_old_session) {
  if (delete_old_session) {
    jQuery.cookie("mvc_session", null, { path: '/' });
  }
  jQuery.cookie("mvc_sessionid", jQuery.uid(),{ path: '/' });
};

jQuery.session_destroy = function() {
  jQuery.cookie("mvc_sessionid", null,{ path: '/' });
  jQuery.cookie("mvc_session", null,{ path: '/' });
  return true;
};

jQuery.session = function(name, value) {
  /* cached in window object */
  if (!name && !value) {
    return window.mvc_session;
  } else if (!value) {
    return window.mvc_session[name];
  } else {
    window.mvc_session[name] = value;
    jQuery.cookie("mvc_session",jQuery.toJSON(window.mvc_session),{ path: '/' });
    return true;
  }
};

/* generate uuid and return it - RFC4122 v4 UUID */
jQuery.uid = function () {
  return "xxxxxxxx-xxxx-4xxx-yxxx-xxxxxxxxxxxx".replace(/[xy]/g, function (c) {
    var r = Math.random() * 16 | 0, v = c === "x" ? r : (r & 0x3 | 0x8);
    return v.toString(16);
  }).toUpperCase();
};
```

```
/*
http://www.jstorage.info/
 * ---------------------------- JSTORAGE ------------------------------------
 * Simple local storage wrapper to save data on the browser side, supporting
 * all major browsers - IE6+, Firefox2+, Safari4+, Chrome4+ and Opera 10.5+
 *
 * Copyright (c) 2010 Andris Reinman, andris.reinman@gmail.com
 * Project homepage: www.jstorage.info
 *
 * Licensed under MIT-style license:
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy
 * of this software and associated documentation files (the "Software"), to deal
 * in the Software without restriction, including without limitation the rights
 * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
 * copies of the Software, and to permit persons to whom the Software is
 * furnished to do so, subject to the following conditions:
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
 * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
 * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
 * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
 * SOFTWARE.
 */

/**
 * $.jStorage
 *
 * USAGE:
 *
 * jStorage requires Prototype, MooTools or jQuery! If jQuery is used, then
 * jQuery-JSON (http://code.google.com/p/jquery-json/) is also needed.
 * (jQuery-JSON needs to be loaded BEFORE jStorage!)
 *
 * Methods:
 *
 * -set(key, value)
 * $.jStorage.set(key, value) -> saves a value
 *
 * -get(key[, default])
 * value = $.jStorage.get(key [, default]) ->
 *    retrieves value if key exists, or default if it doesn't
 *
 * -deleteKey(key)
 * $.jStorage.deleteKey(key) -> removes a key from the storage
 *
 * -flush()
 * $.jStorage.flush() -> clears the cache
 *
 * -storageObj()
 * $.jStorage.storageObj() -> returns a read-ony copy of the actual storage
 *
 * -storageSize()
 * $.jStorage.storageSize() -> returns the size of the storage in bytes
 *
 * -index()
 * $.jStorage.index() -> returns the used keys as an array
 *
 * -storageAvailable()
 * $.jStorage.storageAvailable() -> returns true if storage is available
 *
 * -reInit()
 * $.jStorage.reInit() -> reloads the data from browser storage
 *
 * <value> can be any JSON-able value, including objects and arrays.
 *
 **/

(function($){
    if(!$ || !($.toJSON || Object.toJSON || window.JSON)){
        throw new Error("jQuery, MooTools or Prototype needs to be loaded before jStorage!");
    }

    var
        /* This is the object, that holds the cached values */
        _storage = {},

        /* Actual browser storage (localStorage or globalStorage['domain']) */
        _storage_service = {jStorage:"{}"},

        /* DOM element for older IE versions, holds userData behavior */
        _storage_elm = null,

        /* How much space does the storage take */
        _storage_size = 0,

        /* function to encode objects to JSON strings */
        json_encode = $.toJSON || Object.toJSON || (window.JSON && (JSON.encode || JSON.stringify)),

        /* function to decode objects from JSON strings */
        json_decode = $.evalJSON || (window.JSON && (JSON.decode || JSON.parse)) || function(str){
            return String(str).evalJSON();
        },

        /* which backend is currently used */
        _backend = false;

        /**
         * XML encoding and decoding as XML nodes can't be JSON'ized
         * XML nodes are encoded and decoded if the node is the value to be saved
         * but not if it's as a property of another object
         * Eg. -
         *   $.jStorage.set("key", xmlNode);        // IS OK
         *   $.jStorage.set("key", {xml: xmlNode}); // NOT OK
         */
        _XMLService = {

            /**
             * Validates a XML node to be XML
             * based on jQuery.isXML function
             */
            isXML: function(elm){
                var documentElement = (elm ? elm.ownerDocument || elm : 0).documentElement;
                return documentElement ? documentElement.nodeName !== "HTML" : false;
            },

            /**
             * Encodes a XML node to string
             * based on http://www.mercurytide.co.uk/news/article/issues-when-working-ajax/
             */
            encode: function(xmlNode) {
                if(!this.isXML(xmlNode)){
                    return false;
                }
                try{ // Mozilla, Webkit, Opera
                    return new XMLSerializer().serializeToString(xmlNode);
                }catch(E1) {
                    try {  // IE
                        return xmlNode.xml;
                    }catch(E2){}
                }
                return false;
            },

            /**
             * Decodes a XML node from string
             * loosely based on http://outwestmedia.com/jquery-plugins/xmldom/
             */
            decode: function(xmlString){
                var dom_parser = ("DOMParser" in window && (new DOMParser()).parseFromString) ||
                        (window.ActiveXObject && function(_xmlString) {
                    var xml_doc = new ActiveXObject('Microsoft.XMLDOM');
                    xml_doc.async = 'false';
                    xml_doc.loadXML(_xmlString);
                    return xml_doc;
                }),
                resultXML;
                if(!dom_parser){
                    return false;
                }
                resultXML = dom_parser.call("DOMParser" in window && (new DOMParser()) || window, xmlString, 'text/xml');
                return this.isXML(resultXML)?resultXML:false;
            }
        };

    ////////////////////////// PRIVATE METHODS ////////////////////////

    /**
     * Initialization function. Detects if the browser supports DOM Storage
     * or userData behavior and behaves accordingly.
     * @returns undefined
     */
    function _init(){
        /* Check if browser supports localStorage */
        if("localStorage" in window){
            try {
                if(window.localStorage) {
                    _storage_service = window.localStorage;
                    _backend = "localStorage";
                }
            } catch(E3) {/* Firefox fails when touching localStorage and cookies are disabled */}
        }
        /* Check if browser supports globalStorage */
        else if("globalStorage" in window){
            try {
                if(window.globalStorage) {
                    _storage_service = window.globalStorage[window.location.hostname];
                    _backend = "globalStorage";
                }
            } catch(E4) {/* Firefox fails when touching localStorage and cookies are disabled */}
        }
        /* Check if browser supports userData behavior */
        else {
            _storage_elm = document.createElement('link');
            if(_storage_elm.addBehavior){

                /* Use a DOM element to act as userData storage */
                _storage_elm.style.behavior = 'url(#default#userData)';

                /* userData element needs to be inserted into the DOM! */
                document.getElementsByTagName('head')[0].appendChild(_storage_elm);

                _storage_elm.load("jStorage");
                var data = "{}";
                try{
                    data = _storage_elm.getAttribute("jStorage");
                }catch(E5){}
                _storage_service.jStorage = data;
                _backend = "userDataBehavior";
            }else{
                _storage_elm = null;
                return;
            }
        }

        _load_storage();
    }

    /**
     * Loads the data from the storage based on the supported mechanism
     * @returns undefined
     */
    function _load_storage(){
        /* if jStorage string is retrieved, then decode it */
        if(_storage_service.jStorage){
            try{
                _storage = json_decode(String(_storage_service.jStorage));
            }catch(E6){_storage_service.jStorage = "{}";}
        }else{
            _storage_service.jStorage = "{}";
        }
        _storage_size = _storage_service.jStorage?String(_storage_service.jStorage).length:0;
    }

    /**
     * This functions provides the "save" mechanism to store the jStorage object
     * @returns undefined
     */
    function _save(){
        try{
            _storage_service.jStorage = json_encode(_storage);
            // If userData is used as the storage engine, additional
            if(_storage_elm) {
                _storage_elm.setAttribute("jStorage",_storage_service.jStorage);
                _storage_elm.save("jStorage");
            }
            _storage_size = _storage_service.jStorage?String(_storage_service.jStorage).length:0;
        }catch(E7){/* probably cache is full, nothing is saved this way*/}
    }

    /**
     * Function checks if a key is set and is string or numberic
     */
    function _checkKey(key){
        if(!key || (typeof key != "string" && typeof key != "number")){
            throw new TypeError('Key name must be string or numeric');
        }
        return true;
    }

    ////////////////////////// PUBLIC INTERFACE /////////////////////////

    $.jStorage = {
        /* Version number */
        version: "0.1.5.0",

        /**
         * Sets a key's value.
         *
         * @param {String} key - Key to set. If this value is not set or not
         *              a string an exception is raised.
         * @param value - Value to set. This can be any value that is JSON
         *              compatible (Numbers, Strings, Objects etc.).
         * @returns the used value
         */
        set: function(key, value){
            _checkKey(key);
            if(_XMLService.isXML(value)){
                value = {_is_xml:true,xml:_XMLService.encode(value)};
            }
            _storage[key] = value;
            _save();
            return value;
        },

        /**
         * Looks up a key in cache
         *
         * @param {String} key - Key to look up.
         * @param {mixed} def - Default value to return, if key didn't exist.
         * @returns the key value, default value or <null>
         */
        get: function(key, def){
            _checkKey(key);
            if(key in _storage){
                if(typeof _storage[key] == "object" &&
                        _storage[key]._is_xml &&
                            _storage[key]._is_xml){
                    return _XMLService.decode(_storage[key].xml);
                }else{
                    return _storage[key];
                }
            }
            return typeof(def) == 'undefined' ? null : def;
        },

        /**
         * Deletes a key from cache.
         *
         * @param {String} key - Key to delete.
         * @returns true if key existed or false if it didn't
         */
        deleteKey: function(key){
            _checkKey(key);
            if(key in _storage){
                delete _storage[key];
                _save();
                return true;
            }
            return false;
        },

        /**
         * Deletes everything in cache.
         *
         * @returns true
         */
        flush: function(){
            _storage = {};
            _save();
            /*
             * Just to be sure - andris9/jStorage#3
             */
            try{
                window.localStorage.clear();
            }catch(E8){}
            return true;
        },

        /**
         * Returns a read-only copy of _storage
         *
         * @returns Object
        */
        storageObj: function(){
            function F() {}
            F.prototype = _storage;
            return new F();
        },

        /**
         * Returns an index of all used keys as an array
         * ['key1', 'key2',..'keyN']
         *
         * @returns Array
        */
        index: function(){
            var index = [], i;
            for(i in _storage){
                if(_storage.hasOwnProperty(i)){
                    index.push(i);
                }
            }
            return index;
        },

        /**
         * How much space in bytes does the storage take?
         *
         * @returns Number
         */
        storageSize: function(){
            return _storage_size;
        },

        /**
         * Which backend is currently in use?
         *
         * @returns String
         */
        currentBackend: function(){
            return _backend;
        },

        /**
         * Test if storage is available
         *
         * @returns Boolean
         */
        storageAvailable: function(){
            return !!_backend;
        },

        /**
         * Reloads the data from browser storage
         *
         * @returns undefined
         */
        reInit: function(){
            var new_storage_elm, data;
            if(_storage_elm && _storage_elm.addBehavior){
                new_storage_elm = document.createElement('link');

                _storage_elm.parentNode.replaceChild(new_storage_elm, _storage_elm);
                _storage_elm = new_storage_elm;

                /* Use a DOM element to act as userData storage */
                _storage_elm.style.behavior = 'url(#default#userData)';

                /* userData element needs to be inserted into the DOM! */
                document.getElementsByTagName('head')[0].appendChild(_storage_elm);

                _storage_elm.load("jStorage");
                data = "{}";
                try{
                    data = _storage_elm.getAttribute("jStorage");
                }catch(E5){}
                _storage_service.jStorage = data;
                _backend = "userDataBehavior";
            }

            _load_storage();
        }
    };

    // Initialize jStorage
    _init();

})(window.jQuery || window.$);
```