WICHTIG: Vor Verwendung der Umfragen, bitte Deki AJAX Extension Erweiterung freischalten!
https://web.archive.org/web/20120926021836/http://developer.mindtouch.com/App_Catalog/Simple_Poll

```
// MultiPoll
//    A mult-question extension of SimplePoll (neilw and carles.coll)
//
//    Version history:
//        0.60    29-August-2010        buzz_burrowes   fixed problems
caused by 10.0.0 release
//        0.50    21-December-2009      neilw           Functional but
not beautiful
//
// Parameters have been defined as follows; code has not been written to
match this documentation yet!
//
// Usage:  MultiPoll(questions:list, name:str?, path:str?, closed:str? or
bool?, color_bar:str?, fixed_behavior: str?, language: str?)
//    questions: A list of poll questions and answers.  Each entry should
be a map with two elements:
//               "q": str or xml       Question
//               "a": list of str      Possible answers to the question
//    name:      (optional) Unique name for the poll, in case you want to
store multiple poll results
//                on the same page properties
//    path:      (optional) Path to page where properties will be
stored.  Must be readable/writable
//                by voters!
//    closed:    (optional) Either "true" to close the poll, or a valid
date (GMT!!!!) string indicating
//                when the poll should close
//    color_bar:       (optional) color of results bars (default:
"#B00000")
//    fixed_behavior:            (optional) default -> Automatic.
//                                edit -> Edit Poll
//                                view -> View Poll Statistics
//                                view_details -> View who answers what
//    language:                 (optional) default ->
//                                                    1st  -> Page
language
//                                                    2nd  -> Site
language
//                                                    3rd  -> 'en-us'
//
// get initialized
//
var questions = $0 ?? $questions ?? [];
var fixed_behavior = $fixed_behavior ?? "";
var language = __request.args.language ?? args.language ??
```

```
    ( page.language..'' != '' ? page.language : (site.language..'' != ''
? site.language : 'en-us') );

// -- START LANGUAGE STRINGS
var LANGUAGE_ES = {
    warning_question: "CUIDADO: la pregunta tiene que ser una cadena de
carecteres o un XML",
    warning_no_answers: "CUIDADO: No has pasado respuestas. Así no
podremos hacer un encuesta!",

    error_no_data_page: "ERROR: no puedo encontrar la pagina con los
datos de la encuesta",
    error_invalid_close_date: "ERROR: 'cerrada' no es una fecha valida;
asumo que la encuesta esta abierta",
    error_invalid_data: "ERROR: los datos de la encuesta existen pero
tienen un formato erroneo (tiene que ser de tipo 'map')",
    error_updating_vote: "ERROR: actualizando el voto ",
    error_creating_vote: "ERROR: creando el voto ",
    error_reading_vote: "ERROR: leyendo el voto ",

    txt_your_answer: "Tu respuesta: ",
    txt_change_my_vote: "cambiar mi voto",
    txt_vote: "votar en esta encuesta",
    txt_view_results: "ver resultados",
    txt_poll_closes_in: "La encuesta se cierra en ",
    txt_votes: " votos contados",
    txt_view_details: "ver detalles",
    txt_poll_closed: "esta encuesta esta cerrada",

    msg_poll_closed: "Ahora la encuesta esta cerrada, lo siento!",

    button_submit: "guardar"
};

var LANGUAGE_DE = {
    warning_question: "WARNING: question must be string or xml",
    warning_no_answers: "WARNING: You have provided no answers.  Not much
of a poll!",

    error_no_data_page: "ERROR: can't find page with data store",
    error_invalid_close_date: "ERROR: 'closed' is not a valid date;
assuming poll is open",
    error_invalid_data: "ERROR: poll data store exists but has the wrong
type (instead of 'map')",
    error_updating_vote: "ERROR: updating poll ",
    error_creating_vote: "ERROR: creating poll ",
    error_reading_vote: "ERROR: reding poll ",

    txt_your_answer: "Ihre Antwort: ",
    txt_change_my_vote: "Antworten anpassen",
```

```
    txt_vote: "Antworten in dieser Umfrage",
    txt_view_results: "Zeige Antworten",
    txt_poll_closes_in: "Umfrageschluss ist ",
    txt_votes: " Benutzer haben teilgenommen",
    txt_view_details: "zeige Details",
    txt_poll_closed: "diese Umfrage ist beendet",

    msg_poll_closed: "Die Umfrage ist beendet!",

    button_submit: "Absenden"
};

var LANGUAGE_IT = {
    warning_question: "ATTENZIONE: la domanda deve essere una stringa o
xml",
    warning_no_answers: "ATTENZIONE: non hai risposto.  Così non
riusciamo a fare un sondaggio!!",

    error_no_data_page: "ERRORE: non trovo la pagina con i dati del
sondaggio",
    error_invalid_close_date: "ERRORE: 'chiuso' non è una data valida; si
presume che il sondaggio sia aperto",
    error_invalid_data: "ERRORE: lo store per i dati del sondaggio esiste
ma ha un formato errato (deve essere di tipo 'map')",
    error_updating_vote: "ERRORE: aggiornamento sondaggio ",
    error_creating_vote: "ERRORE: creazione sondaggio ",
    error_reading_vote: "ERRORE: lettura sondaggio ",

    txt_your_answer: "La tua risposta: ",
    txt_change_my_vote: "Cambia il mio voto",
    txt_vote: "Vota il sondaggio",
    txt_view_results: "Risultati",
    txt_poll_closes_in: "Il sondaggio chiude il ",
    txt_votes: " voti",
    txt_view_details: "Dettaglio dei voti",
    txt_poll_closed: "Questo sondaggio è finito",

    msg_poll_closed: "Il sondaggio adesso è chiuso, spiacenti!",

    button_submit: "submit"
};

var TXTS = {
    en: LANGUAGE_DE, 'de-ch': LANGUAGE_DE,
    es: LANGUAGE_ES, 'es-es': LANGUAGE_ES,
    it: LANGUAGE_IT, 'it-it': LANGUAGE_IT
};
var lg = language;
// -- END LANGUAGE STRINGS
```

```
// validate questions and answers XXX needs work
if (questions is not list) <p>TXTS[lg].warning_question;</p>;
else {
    foreach (var q in questions) {
        if (q.q is not str && q.q is not xml)
            <p> "Question ".. __index .." must be string or xml" </p>; //
XXX polyglot?
        if (! #q.a)
            <p> "Question ".. __index .." has no answers" </p>; // XXX
polyglot?
    }
}

var CONST_POLLDATA_NAME = "poll_default_name";

var poll_name = $2 ?? $name?? CONST_POLLDATA_NAME;
if (poll_name is not str) {
   /* <p>"ERROR: el nombre tiene que ser una cadena de caracteres. Le
assignamos automáticamente el nombre '"..CONST_POLLDATA_NAME.."'"</p>; */
    let poll_name = CONST_POLLDATA_NAME;
}
var poll_arg = "poll_"..poll_name;
var path = $3 ?? $path;
var p = (path == nil ? page : wiki.getpage(path));
var p_api = null;
if (p == nil) <p> TXTS[lg].error_no_data_page </p>;
else let p_api = p.api;

var closed = $4 ?? $closed ?? false;
var closing_time = false;

if (closed is not bool) {
    if (!date.isvalid(closed)) {
        <p> TXTS[lg].error_invalid_close_date </p>;
        let closed = false;
    }
    else {      // convert to local time
        let closing_time = date.format(closed,"r");
        let closed = date.compare(date.now, closing_time) > 0;
    }
}
var bar = $5 ?? $color_bar ?? "#B00000";
var viewURI = page.uri & { (poll_arg):"view" };
var viewDetailsURI = page.uri & { (poll_arg):"view_details" };
var editURI = page.uri & { (poll_arg):"edit" };
// Fetch the data store
var data = json.parse(p.properties[poll_name].text ?? '{}');
if (data is not map) <p> TXTS[lg].error_invalid_data </p>;
// Now figure out what to show
var my_vote = (data[user.name] ?? {}).poll;
```

```
var can_vote = !closed && !user.anonymous &&
wiki.pagepermissions(path).update;
var has_voted = (my_vote != nil);
var showform = (fixed_behavior=='edit') ||
            (
             (fixed_behavior=='') &&
                (can_vote && (!has_voted || __request.args[poll_arg] ==
"edit")
                      && __request.args[poll_arg] != "view"
                      && __request.args[poll_arg] != "view_details"
                 )
            );

<form id=(@form)>
<table align="left" cellpadding="5" style="padding:10px;
background-color:#fff; border:2px solid #808080">
    if (!showform)
    {
        <tr><td align="left" style="font-weight:bold">;
            "Aktuelle Umfrageresultate (Total von " .. #data .. "
Teilnehmer)";    //XXX POLY
        </td></tr>;
    }
    foreach (var q in questions) {
        var q_num = __index;
        var first_q = (q_num == 0);
        var last_q  = (q_num == #questions-1);
        var question = q.q;
        var answers  = q.a;
        var vote = my_vote[q_num];
        <tr><td style="font-weight:bold; padding-bottom:10px"> (q_num+1)
.. ") "; question </td></tr>;
        if (showform) {     // Allow user to enter or edit poll
response(s)
            <tr><td align="left">
                <ul style="text-align:left">
                    foreach (var opt in answers) {
                        if ((fixed_behavior == 'edit') &&
(__request.args[poll_arg] != 'edit') && (has_voted)) {
                            if(has_voted && vote==__index) {
<B>TXTS[lg].txt_your_answer;</B>opt; }
                        }
                        else {
                            <li style="list-style:none">
                                <input type="radio" name=("poll"..q_num)
                                    value=(__index) checked=(has_voted &&
vote==__index ? 'checked' : nil)> " "..opt.." " </input>;
                            </li>;
                        }
                    }
```

```
                </ul>;
                if (last_q) {
                    <span style="text-align:left; padding-top:10px">
                        if ((fixed_behavior == 'edit') &&
(__request.args[poll_arg] != 'edit') && (has_voted)) {
                            if (can_vote) <a href=(editURI)> has_voted ?
TXTS[lg].txt_change_my_vote : TXTS[lg].txt_vote </a>;
                        }
                        else { // XXX fix this to create array of
responses (use for loop)
                            <input type="button"
value=(TXTS[lg].button_submit) ctor="when($this.click) {
                                var m = { 'poll':{} };
                                Deki.$('form#' + {{@form}} + '
input').each(function() {
                                    var q;
                                    for (q = 0; q < {{ #questions }}; q++)
                                        if ($(this).attr('name') ==
('poll'+q) && $(this).attr('checked')) m['poll'][q] = Deki.$(this).val();
                                });
                                Deki.publish({{@channel}}, {
{{user.name}}:m }); }"/>; " ";
                        }
                        if (fixed_behavior == '')
                            <a href=(viewURI)> <span
style="font-size:smaller">TXTS[lg].txt_view_results</span> </a>;
                    </span>
                    if (closing_time) {
                        <br />;
                        var ct = date.changetimezone(closing_time,
user.timezone);
                        let ct =
date.format(string.substr(ct,0,string.lastindexof(ct," ")),"MMM d, yyyy,
h:mm tt");
                        <span style="font-size:smaller">
TXTS[lg].txt_poll_closes_in .. ct </span>;
                    }
                }
            </td></tr>;
        }
        else {              // Display poll results
            var results = { (k):data[k].poll[q_num]
                 foreach var k in map.keys(data) where (data[k].poll !=
nil && data[k].poll[q_num] != nil) };
            var total = #results;
            if ( (fixed_behavior=='view') || ( (fixed_behavior=='') &&
(__request.args[poll_arg] != "view_details") ) ) {
                <tr><td align="left">
                    <span style="font-size:smaller"> total ..
TXTS[lg].txt_votes </span>; // XXX POLY
```

```
                    <table>
                        foreach (var i in num.series(0,#answers-1)) {
                            var num_votes = #[ k foreach var k in
map.keys(results) where (results[k] == i) ];
                            var pct = num.round(100*num_votes/
num.max(total,1),1);
                            var width = num.round(2 * pct, 0);
                            <tr>
                                <td> answers[i] </td>
                                <td>
                                    <img src="/skins/common/icons/
icon-trans.gif" style=("background-color:"..bar) height="10"
width=(width) />
                                    <img src="/skins/common/icons/
icon-trans.gif" style="background-color:#C0C0C0" height="10"
width=(200-width) />
                                    " " .. pct .. "% (" .. num_votes ..
")"
                                </td>
                            </tr>;
                        }
                    </table>
                    if (fixed_behavior=='' && last_q) {
                       <span style="text-align:left; padding-top:10px;
font-size:smaller">
                          <a
href=(viewDetailsURI)>TXTS[lg].txt_view_details</a>;
                           ; ;
                          if (can_vote) <a href=(editURI)> has_voted ?
TXTS[lg].txt_change_my_vote : TXTS[lg].txt_vote </a>;
                          else if (closed) TXTS[lg].txt_poll_closed;
                       </span>;
                     }
                </td></tr>;
            }
            else {
                // Show details
                <tr><td align="left">
                    <span style="font-size:smaller"> total ..
TXTS[lg].txt_votes </span>; // XXX POLY
                    <table>
                        <tr>
                            foreach (var i in num.series(0,#answers-1))
                                <th style="text-align:left; border:1px
solid #606060"> answers[i] </th>;
                        </tr>;
                        <tr>
                            foreach (var i in num.series(0,#answers-1)) {
                                var result = [ k foreach var k in
map.keys(results) where results[k] == i ];
```

```
                                    <td valign="top" style="text-align:left">
                                        if (#result) {
                                            <span
style="font-size:.8em">"("..#result.." votes)" </span>;
                                            <br />;
                                        }
                                        foreach (var who in result) {
                                            if (__index) <br />;
                                            who;
                                        }
                                    </td>;
                                }
                            </tr>;
                        </table>;
                        if(fixed_behavior=='' && last_q) {
                            <span style="text-align:left; padding-top:10px;
font-size:smaller">
                                <a href=(viewURI)> <span
style="font-size:smaller">TXTS[lg].txt_view_results</span> </a>;
                                 ; ;
                                if (can_vote) <a href=(editURI)> has_voted ?
TXTS[lg].txt_change_my_vote : TXTS[lg].txt_vote </a>;
                                else if (closed) TXTS[lg].txt_poll_closed;
                            </span>;
                        }
                    </td></tr>;
                }
            }
        }
</table>;
</form>;

// Code to update the page properties, largely cribbed from SteveB's
"UpdateStore" template
dekiapi();
var store = poll_name;
<script type="text/javascript"> "
   Deki.subscribe('"..@channel.."', null, function(c, m, d) {
        var closed = " .. json.emit(closed) .. ";
        var closing_time = " .. json.emit(closing_time) .. ";
        var d = new Date();
        if (closed || (closing_time && d.getTime() >
Date.parse(closing_time))) {
            alert('"..TXTS[lg].msg_poll_closed.."');
            window.location.href = '" .. page.uri .. "';
            return;
        }
        var prop = 'urn:custom.mindtouch.com#'  + '"..store.."';
        Deki.Api.ReadPageProperty('"..p_api.."', prop, function(result) {
            var data = eval('(' + (result.value || '{}') + ')');
```

```
            for (var k in m) data[k] = m[k];
            if(result.etag)
                Deki.Api.UpdatePageProperty(result.href,
YAHOO.lang.JSON.stringify(data), result.etag,
                    function() { window.location.href = '" .. page.uri ..
"'; },
                    function(result) {
alert('"..TXTS[lg].error_updating_vote.." (esatdo: '+result.status+' -
'+result.text+')'); }
                );
            else
                Deki.Api.CreatePageProperty('"..p_api.."', prop,
YAHOO.lang.JSON.stringify(data),
                    function() { window.location.href = '" .. page.uri ..
"'; },
                    function(result) {
alert('"..TXTS[lg].error_creating_vote.." (esatdo: '+result.status+' -
'+result.text+')'); }
                );
          },
          function(result) { alert('"..TXTS[lg].error_reading_vote.."
(esatdo: '+result.status+' - '+result.text+')'); }
        );
    }, null);
" </script>
```