

```
/*
// Copyright (C) 2010 - www.intertesto.com - www.fitzcarraldo.it
// Author: gianluca.sabena@intertesto.com
//
// This program is free software: you can redistribute it and/or modify
// it under the terms of the GNU General Public License http://www.gnu.org/licenses/gpl-2.0.html
//
// ----- Docs -----
//
// Fields:[{type:"",label:"",field:"",value:"",help:"", ,...},{...},...] - Define a series of input types element
// type: name of input type - see below
// label: label of input elements
// field: name of the field used when store data in properties
// elements: list or map of values (apply to radio and select)
// value: default value when create new form
// help: optional text to help the insertion
// viewshow:bool (optional, default: true) show field in view mode - TO DO!!!
// isreadonly: bool (optional, default:false) make field readonly
// isrequired:bool (optional, default:false) field is required, disabled save button if no text is inserted
// hideoninsert:bool
// hideonselect:bool
// jqautocomplete : map - {idautocomplete:"css_id_of_input_form",
//       fieldlabel :{mysql_column:"label text", ...},
//       fieldupdate :{mysql_column:"css_id_input_to_update"},
//       fieldsearch:"name_of_input_form_submint", (syntax: mm_main-queryname-0-field )
//       divcontainer:"css_id_div" (look for id and replace _insert- with _update- if a user select a
record)
//
//
//
// TYPE:
// -- html_text = html input type text - size:'num' (default 16) html max input -
//       maxlength:'num' (optional,default:255) maximum chars length allowed
//       jqmaxchars:num - bind a jq function that check a max length
//       alwaysupdate: bool - if true update data to input.value parameters
// -- html_mail = html input type mail - size:'num' (default 16) html max input -
//       maxlength:'num' (optional,default:255) maximum chars length allowed
//       jqmaxchars:num - bind a jq function that check a max length
// -- html_link = html input type web link - size:'num' (default 16) html max input -
//       maxlength:'num' (optional,default:255) maximum chars length allowed
//       jqmaxchars:num - bind a jq function that check a max length
// -- html_hidden = html input type hidden
// -- html_textarea = html input type textarea - row: number of rows - cols: num of colons -
//       newline: bool (default false) replace '\n' with <br />
// -- html_select = html input type select - elements: list or map used to create drop down select,
//       if map map.key is stored and map.value is displayed as label
//       ismultiple:bool (def:false) - allow multiple selection
// -- html_checkbox = html input type checkbox - elements: single value, set elements = value
//       to start a new form checked
```

```

// -- html_radio = html input type radio - elements:list of radio value-label
// -- jq_date = jQuery datepicker - display date in format dd-mm-yyyy, but save in format yyyy-mm-dd
//           so it is test sortable
// -- mysql_datetime = display a mysql rendered date on insert/update - format in select state -
alwaysupdate:bool - always update date to current date
//
// NOTE:
//
//
// TODO:
// -
*/
var inputs = $inputs ?? $0;
var data = $data ?? $1 ?? [{}];
var state = $state ?? $2; /* renderform | renderdiv */
var idprefix = $idprefix ?? $3 ?? ""; /* a prefix to use in html id name generation */
var forcereadonly = $4 ?? false; /* force any field to be readonly */

/*"debug: data:";data;*/
var loc = "ita";
var labels = {ita:{
  msgFieldRequired:' - Campo obbligatorio',
  msgFieldCharsCount:'Caratteri rimanenti: ',
  msgFieldCharsFull:'Attenzione! Massimo numero di caratteri raggiunto: ',
  msgFieldSaveRequired:'Prima di salvare Completa i campi obbligatori: ',
},
  eng:{MsgNoData:"Store contains no data"}};

var firstSelectLabel = "--- Seleziona ---"; /* text added as first option in a html drop down menu */
var firstSelectValue = "---null-select---"; /* value saved as first option in a html drop down menu */
var requiredfields = []; /* list of fields required */
var fieldvalues = list.collect(inputs,'field');

var editmode = true; /* true=use data in filed, false=use default value */
if ( state=='update' ) {
  let editomode = true;
} else if (state=='delete') {
  let forcereadonly = true;
}

if (state=='insert' || state=='update' || state=='delete') {
  foreach (var input in inputs where ((!(state == "update" && input.hideonupdate==true)) && (!(state ==
"insert" && input.hideoninsert == true) )))
  {
    var jmessage = "";
    var jsform = ""; /* javascript jem function to attach to any form elements */
    if (input.isrequired == true) let requiredfields..=[idprefix..input.field];
    var type = input.type ? (string.toLowerCase(input.type)) : "html_text"; // default to 'text'
    if (input.type == "html_hr") {

```

```

    <div class="fHr" >web.html("<hr />");</div>;
  }
  else if (input.type=="html_div") { var dummy = true}
  else if (input.type == "html_label") {
    <div class="fTitle" >web.html(input.value);</div>;
  } else {
    var inputclass = "";
    let jmessage = (input.jqmaxchars+0 > 0 ?
labels[loc].msgFieldCharsCount..input.jqmaxchars:"")..(input.isrequired ? labels[loc].msgFieldRequired :
"");
    let jsform = (input.isrequired > 0 ? "when($this.keyup || $this.change) CheckField(); ":"");
    if (input.isrequired) let inputclass="frequired";
    var r = ((input.isreadonly || forcereadonly) ? "readonly":"nul");
    <div class=(input.field..' fRow')>;
    <span class="fLabel" > (input.label && input.type!='html_hidden' ?
string.tocamelcase(input.label):!) </span>;
    <span class="fField" >;
      if (type == "html_text" || type == "html_link" || type == "html_mail" || type=="mm_sql") {
        var v = (editmode ? (data[0][input.field].." ?? "")) : (input.value.." ?? ""));
        let v = (input.alwaysupdate ? input.value : v);
        let jsform ..= (input.jqmaxchars+0 > 0 ? "when($this.keyup
limitChars($this,"..input.jqmaxchars..",'fInfo"..input.field..",""..jmessage..");":"");
        <input type="text" id=(idprefix..input.field) name=(idprefix..input.field) class=(inputclass)
value=(v) size=(input.size ?? 16) maxlength=(input.maxlength ?? 255) (r)=(r) ctor=(jsform) />;
      }
      else if (type == "html_textarea") {
        if (type=='mt_body') let mtbodyid = input.field;
        var v = (editmode ? data[0][input.field] : input.value);
        let jsform ..= (input.jqmaxchars+0 > 0 ? "when($this.keyup
limitChars($this,"..input.jqmaxchars..",'fInfo"..input.field..",""..jmessage..");":"");
        <textarea id=(idprefix..input.field) name=(idprefix..input.field) class=(inputclass)
cols=(input.cols ?? 30) rows=(input.rows ?? 4) ctor=(jsform)>v</textarea>;
      }
      else if (type == "html_hidden") {
        var v = (editmode ? data[0][input.field] : input.value);
        let v = (input.alwaysupdate ? input.value : v);
        <input type="hidden" id=(idprefix..input.field) name=(idprefix..input.field) class=(inputclass)
value=(v) />;
      }
      else if (type == "html_select") {
        var m = "nul";
        if (input.ismultiple==true) let m = "multiple";
        <select id=(idprefix..input.field) name=(idprefix..input.field) class=(inputclass) (m)=(m) >;
          /*<option value=(firstSelectValue) > firstSelectLabel </option>;*/
          var val = "";
          var key = "";
          if (typeof input.elements == 'list') {
            let val = input.elements;
            let key = input.elements;
          }

```

```

else if (typeof input.elements == 'map') {
  let val = map.values(input.elements);
  let key = map.keys(input.elements);
} key;
foreach (var opt in val) {
  var v = "nul";
  if (editmode && input.ismultiple==true && list.contains(data[0][input.field].."["",
key[__index]]))
    let v = "selected";
  else if (editmode && data[0][input.field] == key[__index]) let v = "selected";
  else if (!editmode && opt == input.value) let v = "selected";
  if (__index == 0 && key[__index] == '----') {
    <option value=(key[__index]) (v)=(v) > 'Seleziona'; </option>;
  } else {
    <option value=(key[__index]) (v)=(v) > opt; </option>;
  }
}
</select>;
}
else if (type == "html_radio") {
  foreach (var opt in input.elements) {
    var v = "nul";
    if (editmode && data[0][input.field] == opt) let v = "checked";
    else if (!editmode && opt == input.value) let v = "checked";
    <input style="width:auto;" id=(idprefix..input.field) type="radio"
name=(idprefix..input.field) class=(inputclass) value=(opt) (v)=(v) > " ..opt.." </input>;
  }
}
else if (type == "html_checkbox") {
  var v = "nul";
  if (editmode && data[0][input.field] == input.elements) let v = "checked";
  else if (!editmode && input.elements == input.value) let v = "checked";
  <input id=(idprefix..input.field) type="checkbox" name=(idprefix..input.field)
class=(inputclass) value=(input.elements) (v)=(v) />;
}
else if (type == "jq_date") {
  var v = (editmode ? data[0][input.field] : input.value);
  <input id=(idprefix.."jqdateview".."__count) type="text" readonly="readonly"
value=(date.isvalid(v) ? date.format(v,'dd-MM-yyyy'):"") ctor=("$this.datepicker({
altField:'#"..idprefix..input.field.."',dateFormat:'dd-mm-yy',altFormat: 'yy-mm-dd' }))/>
  <input id=(idprefix..input.field) type="hidden" name=(idprefix..input.field) class=(inputclass)
value=(date.isvalid(v) ? date.format(v,'yyyy-MM-dd HH:mm:ss'):"") />;
}
else if (type == "mysql_datetime") {
  var v = (editmode ? data[0][input.field] : date.Format(date.now, "yyyy-MM-dd HH:mm:ss"));
  let v = (input.alwaysupdate ? date.Format(date.now, "yyyy-MM-dd HH:mm:ss") : v);
  var r = (input.isreadonly ? "readonly":"nul");
  <input type="text" id=(idprefix..input.field) name=(idprefix..input.field) class=(inputclass)
value=(v) size=(input.size ?? 16) maxLength=(input.maxLength ?? 255) (r)=(r) />;
}

```

```

    else if (type == "int_locator") {
        var v = (editmode ? data[0][input.field] : input.value);
        <div id="int_locator" >template("Xsys/Locator",[input.gmapOpt, input.idlat, input.idlng,
input.address ]);</div>
    }
    else {
        <span style="color:red"> "ERROR: unknown input type '"..type.."'" </span>;
    }
</span> /*class="fField"*/
<div class="fHelp" ><span id=("fInfo"..input.field)>jmessage;</span>"
";<span>input.help</span></div>;
</div>
}
if (#input.jqautocomplete ) {
    var wfid = string.split(idprefix,"-")[1].."_"..input.field;
    var wfid_update = "";
    var wfid_insert = "";
    if (string.contains(wfid,"_update")) {
        let wfid_update = wfid;
        let wfid_insert = string.replace(wfid,"_update","_insert");
    } else {
        let wfid_insert = wfid;
        let wfid_update = string.replace(wfid,"_insert","_update");
    }
    var t = input.jqautocomplete;
    let t ..= {idautocomplete: input.field};
    if (string.startswith(idprefix,"mm_model")) {
        <script>"
            var jqacd_"..wfid_insert.."="..json.emit(t)..";
            var jqacd_"..wfid_update.."="..json.emit(t)..";
        "</script>;
    } else {
        <script>"
            var jqacd_"..wfid_insert.."="..json.emit(t)..";
            var jqacd_"..wfid_update.."="..json.emit(t)..";
            $(document).ready(function(){
crb_autocomplete("'"..string.remove(idprefix,#idprefix-1,1).."'',jqacd_"..wfid..");});"</script>
            }
        }
    } /* foreach input in inputs*/
} else if (state=='select') {
    let data = data[0];
    foreach ( var input in inputs where (!(state == "select" && input.hideonselect))) {
        var t = "";
        if (input.type == "html_text") {
            let t = data[input.field].."";
        }
        else if (input.type == "html_select") {
            var d = [];
            if (typeof (data[input.field]) !='list') let d = [data[input.field]];

```

```

else let d = data[input.field];
foreach (var e in d) {
  if (typeof (input.elements) == 'list') {
    let t ..= e.." ";
  }
  else if (typeof (input.elements) == 'map') {
    var j = input.elements;
    let t ..= j[e].." ";
  }
}
}
else if(input.type == "html_link"){
  var v = string.trim(string.toLowerCase(data[input.field]));
  if(string.startswith(v,"http://") || string.startswith(v,"https://")){
    let t = web.link(v,v);
  }else{
    let t = web.link("http://"..v,v);
  }
}
else if(input.type == "html_mail"){
  var v = string.trim(string.toLowerCase(data[input.field]));
  let t = web.link("mailto:"..v,v);
}
else if (input.type == "html_textarea") {
  if (input.newline == true) let t = web.html(string.replace(data[input.field],'\n','<br />'));
  else let t = data[input.field];
}
else if (input.type == "mtsid_user" && data[input.field] != firstSelectValue) {
  var v = wiki.getuser(data[input.field]+0);
  if (input.link=='page') let t = web.link(site.uri..'User:'..v['name'],v['name']);
  else if (input.link=='changes') let t =
web.link(site.uri..'Special:Contributions?target='..v['name'],v['name']);
  else let t = v['name'];
}
else if (input.type == "mtmid_users") {
  var s = "";
  var t = "";
  foreach ( var i in data[input.field] where i != firstSelectValue) {
    var v = wiki.getuser (i+0);
    if (input.link=='page') let t ..= s..web.link(site.uri..'User:'..v['name'],v['name']);
    else if (input.link=='changes') let t ..=
s..web.link(site.uri..'Special:Contributions?target='..v['name'],v['name']);
    else let t ..= v['name'];
    let s = ", ";
  }
}
else if (input.type == "mtsid_page" && data[input.field] != firstSelectValue ) {
  var v = wiki.getpage(data[input.field]+0);
  if(#v && input.link!=false) let t = web.link(v['uri'],v['title']);
  else if(#v) let t = v['name'];
}

```

```

}
else if (input.type == "mtmid_pages") {
  var s = "";
  foreach ( var i in data[input.field] where != firstSelectValue ) {
    var v = wiki.getpage(i+0);
    if(#v && input.link!=false) let t .= s..web.link(v['uri'],v['title']);
    else if(#v) let t.= v['name'];
    let s = ", ";
  }
} else if (input.type == "mysql_datetime") {
  if (date.isvalid(data[input.field]))
    let t = (#input.dateformat >0 ? date.format(data[input.field],input.dateformat) : data[input.field]);
  else let t="----";
}
else if (input.type == "jq_date") {
  if (date.isvalid(data[input.field]))
    let t = (#input.dateformat >0 ? date.format(data[input.field],input.dateformat) : data[input.field]);
  else let t="----";
}
if (input.type == "int_locator") {
  if(#data['latitude']>0 && #data['longitude']>0) {
    <div class="geo fRow">;
    <span class="fLabel" >input.label;</span>;
    <br />;
    <span class="fField">web.image("http://maps.google.com/maps/api/
staticmap?center="..data['latitude']..".."data['longitude'].."&zoom=11&size=280x230&mapttype=terrain&|"..data['la
</div>;
  }
}
if (input.type == "html_hr") {
  <div class="fHr" >;web.html("<hr />");</div>;
}
if (input.type=="html_div") { var dummy = true}
if (input.type == "html_label") {
  <div class="fTitle" >;web.html(input.value);</div>;
}
if (#t > 0) {
  <div class=(input.field..' fRow')>;
  <span class="fLabel" > (input.label && input.type!='html_hidden' ?
string.tocamelcase(input.label):" ) </span>;
  <span class="fField" >;web.html(t);</span>
  </div>
}
} /* foreach */
} /* if state=='select' */

// ----- javascript -----

<script type="text/jem">"
if (typeof(requiredfields)=='undefined') var requiredfields = [];

```

```
requiredfields = requiredfields.concat(eval("..json.emit(requiredfields).."));  
"</script>
```