```php
<?



function getDaySelect() {


  $data[0][value] = "";
  $data[0][text] = "";
  for($i=1;$i<=31;$i++) {

    if ($i < 10) {
      $dummy = "0$i";
    } else {
      $dummy = "$i";
    }

    $data[$i][value] = $dummy;
    $data[$i][text] = $dummy;
  }

  return $data;
}

function getMonthSelect() {

  $data[0][value] = "";
  $data[0][text] = "";

  for($i=1;$i<=12;$i++) {

    if ($i < 10) {
      $dummy = "0$i";
    } else {
      $dummy = "$i";
    }

    $data[$i][value] = $dummy;
    $data[$i][text] = $dummy;
  }

  return $data;
}



function query($query,$link = "") {
  global
    $SCRIPT_FILENAME;

  $oid = mysql_query($query);
```

```
52
53    if (!$oid) {
54      $code = returnCode(error,generic,__FILE__,__LINE__);
55    }
56
57    do {
58      $data = mysql_fetch_assoc($oid);
59      if ($data) {
60        $content[] = $data;
61      }
62    } while ($data);
63
64    $script = basename($SCRIPT_FILENAME);
65    if (($link != "") and (count($content) == 0)) {
66
67      if (getType($link) == "integer") {
68        Header("Location: $script?page=$link&code=$code");
69      } else {
70        Header("Location: $link&code=$code");
71      }
72    } else {
73      return $content;
74    }
75  }
76
77
78  Class Render {
79
80    function ShowData($name,$data,$parameters,$value = "",$event = "") {
81
82      $content .= "<table bgcolor=#eeeeee style='border: 2px solid
    salmon;'>";
83      $content .= "<tr><td>Widget Name</td><td><b>$name</b></td></tr>\n";
84
85      if ($parameters != "") {
86            $content .=
    "<tr><td>Parameters</td><td>$parameters</td></tr>\n";
87      }
88
89      $content .= "<tr><td>Data
    count</td><td>".count($data)."</td></tr>\n";
90
91
92      if (count($data)>0) {
93
94        $content .= "<tr><td colspan=2><hr style='height: 1px;
    background: salmon; width: 98%;' ></td></tr>\n";
95
96
97        foreach($data as $k => $v) {
98
```

```
99      $content .= "<tr><td align=center>$k</td><td>";
100
101     foreach($v as $k1 => $v1) {
102
103       $content .= "$k1: $v1<br>";
104     }
105     $content .="<hr style='height: 1px; background: salmon; width:
    98%;'></td></tr>\n";
106
107       }
108
109     }
110
111
112
113     $content .= "</table>";
114
115     return $content;
116   }
117
118   function explodeParameters($parameters) {
119
120     $buffer = $parameters;
121
122     do {
123
124       $result = ereg("^([[:alnum:]]+)",$buffer,$token);
125
126       if ($result) {
127
128     $buffer = ereg_replace("^$token[1]","",$buffer);
129
130     $result2 = ereg("^=\"([[:alnum:] ]*)\"",$buffer,$token2);
131
132     if ($result2) {
133
134       $buffer = ereg_replace("^=\"$token2[1]\"[[:space:]
    ]*","",$buffer);
135
136       $par[$token[1]] = $token2[1];
137
138     }
139       }
140
141     } while ($result);
142
143     return $par;
144   }
145
146
147   function Select($name,$data,$parameters = "",$value = "",$event = "")
```

```
    {
148
149     /*
150
151        $data is formatted like this
152
153        $data[0][value]
154        $data[0][text]
155
156     */
157
158     $attributes = Render::explodeParameters($parameters);
159     $event = $attrubutes[event];
160
161     $content .= "<select name=\"$name\" $event class=dataentry>\n";
162
163     if (count($data) == 0) {
164       $content .= "<option
    value=\"\">          
      \n";
165     } else {
166
167       if (isset($attributes[first])) {
168     $content .= "<option value=\"\">$attributes[first]\n";
169       }
170     }
171
172     if (count($data) > 0) {
173       foreach($data as $k => $v) {
174
175     if (($v[value] == $value) or ($v[value] == $attributes[value]) or
    ($v[checked]) or ($v[selected])) {
176       $content .= "<option value=\"$v[value]\" selected> $v[text]\n";
177     } else {
178       $content .= "<option value=\"$v[value]\"> $v[text]\n";
179     }
180       }
181     }
182
183     $content .= "</select>\n";
184
185     return $content;
186   }
187
188   function CheckBoxList($name,$data) {
189
190     /*
191
192        $data is formatted like this
193
194        $data[0][value]
```

```
195         $data[0][text]
196         $data[0][checked]
197
198     */
199
200     $content .= "<table cellspacing=0 cellpadding=0 width=100%>\n";
201
202     foreach($data as $k => $v) {
203
204       if (($k % 2) == 0) {
205     $content .= my_comma(checkboxlist,"</td></tr>")."<tr><td
    width=1%>\n";
206       } else {
207     $content .= "</td><td width=1%>\n";
208       }
209
210       if ($v[checked]) {
211     $content .= "<input type=checkbox name=\"".$name."_$v[value]\"
    value=\"*\" checked></td><td width=49%>$v[text]\n";
212       } else {
213     $content .= "<input type=checkbox name=\"".$name."_$v[value]\"
    value=\"*\"></td><td width=49%>$v[text]\n";
214       }
215     }
216
217 #    echo count($data);
218
219     $contemt .= "</td></tr>\n";
220     $content .= "</table>\n";
221
222     return $content;
223   }
224
225   function RadioBox($name,$data) {
226
227     /*
228
229         $data is formatted like this
230
231         $data[0][value]
232         $data[0][text]
233         $data[0][checked]
234
235     */
236
237     $content .= "<table cellspacing=0 cellpadding=0 width=100%>\n";
238
239     foreach($data as $k => $v) {
240
241       if (($k % 2) == 0) {
242     $content .= my_comma(checkboxlist,"</td></tr>")."<tr><td
```

```
    width=1%>\n";
243       } else {
244     $content .= "</td><td width=1%>\n";
245       }
246
247       if ($v[checked]) {
248     $content .= "<input type=radio name=\"$name\" value=\"*\"
    checked></td><td width=49%>$v[text]\n";
249       } else {
250     $content .= "<input type=radio name=\"$name\" value=\"*\"></td><td
    width=49%>$v[text]\n";
251       }
252     }
253
254 #    echo count($data);
255
256     $contemt .= "</td></tr>\n";
257     $content .= "</table>\n";
258
259     return $content;
260   }
261
262
263   function HiddenList($name,$data) {
264
265     /*
266
267        $data is formatted like this
268
269        $data[0][value]
270        $data[0][text]
271
272     */
273
274     $content = "<input type=hidden name=$name
    value=\"".addslashes(serialize($data))."\">";
275
276     return $content;
277   }
278
279   function Date($name,$mode = "",$today = "") {
280
281     switch($mode) {
282     case forward:
283       $year_min = date(Y);
284       $year_max = date(Y)+2;
285       break;
286     case backward:
287       $year_min = date(Y)-20;
288       $year_max = date(Y)+2;
289       break;
```

```
290     default:
291       $year_min = date(Y)-20;
292       $year_max = date(Y)+20;
293       break;
294     }
295
296     if ($today == "") {
297       $today = date(Ymd);
298     }
299
300     $content .= "<select name=$name"."_d class=dataentry>\n";
301     $content .= "<option>\n";
302
303     for ($i=1;$i<=31;$i++) {
304       if ($i < 10) {
305     $dummy = "0$i";
306       } else {
307     $dummy = "$i";
308       }
309       if (substr($today,6,2) == $dummy) {
310     $content .= "<option value=$dummy selected>$dummy\n";
311       } else {
312     $content .= "<option value=$dummy>$dummy\n";
313       }
314     }
315     $content .= "</select>\n";
316
317     $content .= "<select name=$name"."_m class=dataentry>\n";
318     $content .= "<option>\n";
319     for ($i=1;$i<=12;$i++) {
320       if ($i < 10) {
321     $dummy = "0$i";
322       } else {
323     $dummy = "$i";
324       }
325       if (substr($today,4,2) == $dummy) {
326     $content .= "<option value=$dummy selected>$dummy\n";
327       } else {
328     $content .= "<option value=$dummy>$dummy\n";
329       }
330     }
331     $content .= "</select>\n";
332
333     $content .= "<select name=$name"."_y class=dataentry>\n";
334     $content .= "<option>\n";
335
336     for ($i=$year_min;$i<=$year_max;$i++) {
337       if (substr($today,0,4) == $i) {
338     $content .= "<option value=$i selected>$i\n";
339       } else {
340     $content .= "<option value=$i>$i\n";
```

```
341         }
342       }
343       $content .= "</select>\n";
344 
345       return $content;
346    }
347 
348     function getDate($name) {
349 
350       return
    $GLOBALS[$name."_y"].$GLOBALS[$name."_m"].$GLOBALS[$name."_d"];
351     }
352 
353 }
354 
355 
356 
357 
358 
359 #ob_start();
360 #ob_implicit_flush(0);
361 
362 function CheckCanGzip(){
363   global $HTTP_ACCEPT_ENCODING;
364 
365   if (headers_sent() ||
366       connection_aborted()){
367     return 0;
368   }
369 
370   if (ereg("x-gzip",$HTTP_ACCEPT_ENCODING) !== false) {
371     return "x-gzip";
372   }
373 
374   if (ereg("gzip",$HTTP_ACCEPT_ENCODING) !== false) {
375     return "gzip";
376   }
377 
378 
379   return 0;
380 }
381 
382 /* $level = compression level 0-9, 0=none, 9=max */
383 
384 function GzDocOut($level=9,$debug = 0){
385 
386 
387   $ENCODING = CheckCanGzip();
388 
389 
390   if ($ENCODING){
```

```
391
392     $Contents = ob_get_contents();
393     ob_end_clean();
394       $s = "\n<!-- Koriandol compress module : $ENCODING
    (".strlen($Contents)."/".strlen(gzcompress($Contents,$level)).")
    -->\n";
395       $Contents .= $s;
396
397     header("Content-Encoding: $ENCODING");
398     print "\x1f\x8b\x08\x00\x00\x00\x00\x00";
399     $Size = strlen($Contents);
400     $Crc = crc32($Contents);
401     $Contents = gzcompress($Contents,$level);
402     $Contents = substr($Contents, 0, strlen($Contents) - 4);
403     print $Contents;
404     print pack('V',$Crc);
405     print pack('V',$Size);
406     exit;
407   }else{
408     ob_end_flush();
409     exit;
410   }
411 }
412
413 class Template {
414   var
415     $template_file,
416     $buffer,
417     $foreach,
418     $content,
419     $pars,
420     $debug,
421     $parsed;
422
423
424   function Template($filename,$debug = "") {
425     $this->template_file = $filename;
426     $this->debug = $debug;
427     $this->parsed = false;
428     $fp = fopen ($filename, "r");
429     $this->buffer = fread($fp, filesize($filename));
430     fclose($fp);
431     $i=0;
432     do {
433       $result =
    ereg("<\[foreach\]>(.+)<\[\/foreach\]>",$this->buffer,$token);
434       if ($result) {
435     $this->foreach[] = $token[1];
436     $this->buffer =
    ereg_replace("<\[foreach\]>.+<\[\/foreach\]>","<[foreach$i]>",$this->bu
    ffer);
```

```
437         }
438         $i++;
439       } while ($result);
440     }
441
442     function setContent($name, $value, $pars = "") {
443       $this->content[$name][] = $value;
444       $this->pars[$name][] = $pars;
445
446
447     }
448
449
450
451     function pre($var) {
452       if ($this->debug == "DEBUG") {
453         return "<!- begin:$var -->";
454       }
455     }
456
457     function post($var) {
458       if ($this->debug == "DEBUG") {
459         return "<!- end:$var -->";
460       }
461     }
462
463     function parse() {
464
465       $this->parsed = true;
466
467       if ($this->debug == "DEBUG") {
468         $pre = "<!- \$token[1] -->";
469         $post = "<!- \$token[1] -->";
470       }
471
472
473
474
475       for($i=0;$i<count($this->foreach);$i++) {
476
477         $result =
    ereg("<\[([a-zA-Z0-9_]+)\]>",$this->foreach[$i],$token);
478         if ($result) {
479             $iterations = count($this->content[$token[1]]);
480         }
481
482         for ($j=0;$j<$iterations;$j++) {
483             $buffer = $this->foreach[$i];
484             do {
485
486                 $result = ereg("<\[([a-zA-Z0-9_]+)\]>",$buffer,$token);
```

```
487                 if ($result) {
488
489                     $buffer =
    ereg_replace("<\[$token[1]\]>",$this->pre($token[1]).$this->content[$to
    ken[1]][$j].$this->post($token[1]),$buffer);
490
491
492
493                 }
494
495                 /* nuovo */
496                 $result_2 =
    ereg("<\[([a-zA-Z0-9_]+)::([a-zA-Z0-9_]+)[[:space:] ]*([[:alnum:]
    =\.\%\#\'\"\_-]*)\]>",$buffer,$token);
497                 if ($result_2) {
498
499
500
501                     $buffer =
    ereg_replace("<\[$token[1]::$token[2][[:space:] ]*$token[3]\]>",
502                         $this->pre($token[1]).
503                         $this->render($token[1],
504                               $this->content[$token[1]][$j],
505                               $token[2],
506                               $token[3]).
507                         $this->post($token[1]),
508                         $buffer);
509                 }
510                 /* nuovo */
511
512             } while ($result or $result_2);
513             $this->content["foreach$i"][0] .= $buffer;
514         }
515     }
516
517     do {
518         $result = ereg("<\[([a-zA-Z0-9_]+)\]>",$this->buffer,$token);
519         if ($result) {
520             $this->buffer =
    ereg_replace("<\[$token[1]\]>",$this->pre($token[1]).$this->content[$to
    ken[1]][0].$this->post($token[1]),$this->buffer);
521         }
522     } while ($result);
523
524
525     do {
526       $result = ereg("<\[([a-zA-Z0-9_]+)::([a-zA-Z0-9_]+)[[:space:]
    ]*([[:alnum:] =\.\%\#\'\"\_\-]*)\]>",$this->buffer,$token);
527       if ($result) {
528           $this->buffer =
    ereg_replace("<\[$token[1]::$token[2][[:space:] ]*$token[3]\]>",
```

```
529                       $this->pre($token[1]).
530                       $this->render($token[1],
531                             $this->content["$token[1]"][0],
532                             $token[2],
533                             $token[3]).
534                       $this->post($token[1]),
535                       $this->buffer);
536 
537       }
538     } while ($result);
539 
540   }
541 
542   function close() {
543 
544     if (!$this->parsed) {
545       $this->parse();
546     }
547 
548     echo $this->buffer;
549 #    gzdocout();
550   }
551 
552   function get() {
553 
554     if (!$this->parsed) {
555       $this->parse();
556     }
557 
558     return $this->buffer;
559   }
560 
561   function render($name,$data,$widget,$parameters,$value = "",$event =
    "") {
562 
563     $parameters = $parameters." ".$this->pars[$name][0];
564 
565     switch($widget) {
566     case "select2":
567     case "Select2":
568 
569     /*
570 
571        $data is formatted like this
572 
573        $data[0][value]
574        $data[0][text]
575 
576     */
577 
578       $content = Render::Select($name,$data,$parameters,$value,$event);
```

```
579       break;
580     case "checkbox":
581     case "CheckBox":
582     case "checkBox":
583       $content = Render::CheckBoxList($name,$data,$value,$event);
584       break;
585     case "radiobox":
586     case "RadioBox":
587     case "radioBox":
588       $content = Render::RadioBox($name,$data,$value,$event);
589       break;
590
591     case "show_data":
592     case "showdata":
593     case "showData":
594     case "Showdata":
595     case "ShowData":
596     case "inspect":
597     case "Inspect":
598
599       $content =
    Render::ShowData($name,$data,$parameters,$value,$event);
600       break;
601
602     default:
603
604       $pars = TagAux::parsePars($parameters);
605
606       if ($pars[library]) {
607     $library = $pars[library];
608     require_once "include/tags/$library.inc.php";
609
610       } else {
611     $library = "TagLibrary";
612       }
613
614       echo $this->parameters[$widget][0];
615
616       eval("\$content =
    ".$library."::".$widget."(\$name,\$data,TagAux::parsePars(\$parameters)
    );");
617
618       break;
619
620     }
621     return $content;
622   }
623 }
624
625 Class TagAux {
626
```

```
627  function parsePars($parameters) {
628
629     $buffer = $parameters;
630     do {
631       $result = ereg("^([[:alnum:] \_]+)",$buffer,$token);
632       if ($result) {
633     $buffer = ereg_replace("^$token[1]","",$buffer);
634     $result2 = ereg("^=\"([[:alnum:]\.\-\_\%\#
    \_\-]*)\"",$buffer,$token2);
635     if ($result2) {
636       $buffer = ereg_replace("^=\"$token2[1]\"[[:space:]
    ]*","",$buffer);
637       $par[$token[1]] = $token2[1];
638        }
639       }
640
641     } while ($result);
642
643     return $par;
644   }
645
646
647 }
648
649 Class TagLibrary {
650
651   function htmlInclude($nome,$data,$pars) {
652
653     $url = $data;
654
655     $content .= "<!-- begin: $url -->\n";
656
657     $fp = fopen("$url","r");
658
659     if (!$fp) {
660       $content .=" Error while opening $url ";
661     } else {
662       $content .= fread($fp,filesize("$url"));
663     }
664
665
666     fclose($fp);
667
668     $content .= "<!-- end:$url -->\n";
669
670     return $content;
671   }
672
673   function help($message) {
674     global $dateLibraryDeploy;
675
```

```
676     if (!$dateLibraryDeploy) {
677       $content .= "\n\n<script language=\"javascript\"
    src=\"js/help.js\"></script>\n";
678       $content .= "<link href=\"css/help.css\" rel=\"stylesheet\"
    type=\"text/css\">\n";
679       $content .= "<div id=help></div>\n";
680     }
681 
682     $content .= " <a href=#><img border=0 src=img/help.png
    onClick=\"showHelp(this,'$message')\"></a>";
683 
684     return $content;
685 
686   }
687 
688 
689 }
690 
691 ?>
```