Maps and links handler

Disclaimer

This SOFTWARE PRODUCT is provided by El Condor "as is" and "with all faults." El Condor makes no representations or warranties of any kind concerning the safety, suitability, lack of viruses, inaccuracies, typographical errors, or other harmful components of this SOFTWARE PRODUCT. There are inherent dangers in the use of any software, and you are solely responsible for determining whether this SOFTWARE PRODUCT is compatible with your equipment and other software installed on your equipment. You are also solely responsible for the protection of your equipment and backup of your data, and El Condor will not be liable for any damages you may suffer in connection with using, modifying, or distributing this SOFTWARE PRODUCT.

You can use this SOFTWARE PRODUCT freely, if you would you can credit me in program comment:

El Condor – CONDOR INFORMATIQUE – Turin

Comments, suggestions and criticisms are welcomed: mail to rossati@libero.it

Conventions

Commands syntax, instructions in programming language and examples are with font `COURIER NEW`. The optional parties of syntactic explanation are contained between `[square parentheses]`, alternatives are separated by `|` and the variable parties are in ***`italics`.***

Contents

1 Mapper

Mapper is a PHP script for handle maps with linkable symbols inside; the symbols can have great flexibility of shape, color, dimension and type of interaction.

1.1 Components

Mapper is built on three scripts:

- `mapper.php` shows map and linkable symbols
- `mapper.js` handle the point set
- `genimg.php` for create geometric symbols

1.2 Syntax

To show linkable map include a script `mapper.php` and call a function `createMap`:

```
include 'mapper/mapper.php';
createMap($parm,[$handle=false])
```

Where `$parm` is an array of parameters and `$handle` `true` allow to modify the points set (see par. 1.4).

1.3 Parameters

The essential parameters concerns the image and the points where to put the symbols; this one has almost a name and the point coordinates: mapper shows the images with symbols using default values.

Is possible to personalize the map setting some properties at general level (i.e. for all points) or for some categories of point or for every point. The setting at point level override the setting associated to a category which override the setting at general level which, finally, override the defaults setting.

The parameter name is case-insensitive (except for the location's name).

☞ Some parameters can appear both at general level both at point level.

The parameters below, which are in logical order, refers to a visualization map; for the map used to manage points i.e. add or change or remove points see the paragraph 1.4.

1.3.1 General level parameters

1.3.1.1 image

Is the map image name, possibly with path.

syntax	`"image"=>"image_directory"`
default	`world.png`
example	`array("image"=>"images/europe.png",...);`

1.3.1.2 points

syntax	`"points"=>array("location_name"=>array(X=>x_coord,Y=>y_coord[, level point parameters],...);`
default	`no points`

1.3.1.3 event

syntax	`"event"=>"function_name|complete handle"`
note	the `event` can have a function name or a complete event handler, this must have the form `on`*`EventName`*`='`*`something`*`'`. In the first case is generated an `onclick` event: `on`*`Click`*`='`*`function_name`*`(`*`location_name`*`)'`.
example	`"event"=>"handlePoint"` `"event"=>"onMouseOver='See()'"`

1.3.1.4 color

syntax	`"color"=>"red|green|blue|yellow|`*`six digit hexadecimal value`*`"`
default	`red`

example `"color"=>"00EOE0"`

1.3.1.5 width

syntax `"width"=>`*dimension*
default `10`
note is the dimension of the symbol, the point coordinates are at the center of the symbol.

1.3.1.6 alpha

syntax `"alpha"=>`*transparency*
default `0.7`
note the transparency (alpha channel) is a value within `0` and `1`.

1.3.1.7 shape

syntax *"shape"=>"*`circle|square|triangle|diamond|`*link|UNICODE"*
default `circle`
note *link* is an url which provide an image, in this case `color` parameter is meaningless. *UNICODE* is a numeric value of character, that, moreover, provide many symbols; *UNICODE* has the form `&#`*n*`;` or `&#x`*h*`;`, where *n* ad *h* are respectively decimal and hexadecimal values.

1.3.1.8 type

syntax *"type_name"*`=>array(...)`
default `no`
note *type_name* is a character string, *array* can contain `event`, `color`, `shape` and `alpha` parameters.

1.3.2 Level point parameters

In the points parameter there can be present `event, color, shape` and `alpha` which overrides the default or the analogous parameter appearing at general level.

1.3.2.1 anchor

syntax `"anchor"=>"`*href_link*`"`
default no anchor
example `"anchor"=>"'doc.pdf' target=_blank"`
note *href_link* is a reference (without `href=`), it can contain also other Anchor parameters (like `target`).

1.3.2.2 type

syntax `"type"=>"`*type_name*`"`
default no
note this permits to refer to a *type_name* set of parameters.

1.4 Handle points

Set points coordinate in map can be annoying, Mapper facilitate this task allowing to insert, update and delete points.

This is done calling `createMap` with second parameter set to `true`. Only `image` and `points` parameters are needed.

You must include `mapper.php` and call `createMap` function:

```
include 'mapper/mapper.php';
createMap($parm,true)
```

Besides it must be included a javascript code `mapper.js`:

```
<script type='text/javascript' src='mapper/js/mapper.js'></script>
```

`createMap` add two buttons, one for show the list of points and one for restore the initial situation; the user must insert a button for capture the points. Note if you capture the points, the actual situation becomes the

new initial situation.

1.4.1 Delete point

Clicking on point appears a form where we can delete point.

1.4.2 Change name point

Clicking on point appears a form where we can change the name point.

1.4.3 Change point coordinates

Choice new point and assign the name of the location.

1.4.4 Retrieve points

For retrieve the set Point you can access to the array `mapper.Points` or use a `mapper.returnPoints` function. In the array the key is the point name and the value is an array which contains X-position, Y-position and ID name; below is a sample of points array, the right size is what is returned by `mapper.returnPoints` function

```
mapper.Points = {"Tanzania":[350,220,"ID350220"],"Kenia":
[355,195,"ID355195"],"Birmania":[478,185,"ID478185"]}
```

Here a sample script for handle the changes made.

```
// $points contains the points in json format
$aPoints = json_decode($points,true);
$aDelPoints = array_diff(array_keys($aOldPoints),array_keys($aPoints));  // deleted
$aNewPoints = array_diff(array_keys($aPoints),array_keys($aOldPoints));  // new points
$aUpdPoints = array_intersect(array_keys($aPoints),array_keys($aOldPoints));
echo "\nUpdate Points DataBase";
foreach($aUpdPoints as $k) {
  if ($aPoints[$k][0] != $aOldPoints[$k][0] && $aPoints[$k][1] != $aOldPoints[$k][1]) {
     $sql = "UPDATE mappa_coord SET X={$aPoints[$k][0]},Y={$aPoints[$k][1]} WHERE Link
='$k'";
     mysql_query($sql);
     echo "\nUpdated $k";
  }
}
foreach($aDelPoints as $k) {
  $sql = "DELETE FROM mappa_coord WHERE Link ='$k'";
  mysql_query($sql);
  echo "\nDeleted $k";
}
foreach($aNewPoints as $k) {
  $sql = "INSERT INTO mappa_coord (Mappa,X,Y,Link) VALUES('Mondo',{$aPoints[$k][0]},
{$aPoints[$k][1]},'$k')";
  mysql_query($sql);
  echo "\nInserted $k";
}
```

1.5 Example

1.5.1 Map with different events and symbols

```
$aParams = array("image"=>"images/world.png",
                 "event"=>"gestClick",
                 "alpha"=>1,
                 "event"=>"alert",
                 "towns"=>array("color" =>"blue","width" =>14,"alpha" =>0.5),
                 "points"=>array(
                     "Tanzania"=>array("x"=>355,"Y"=>225, "type"=>"towns"),
                     "Kenia"=>array("x"=>360,"Y"=>200,"type"=>"towns"),
                     "Birmania"=>array("x"=>483,"Y"=>190,
```

```
                                "event"=>"onMouseOver='alert(\"Burma\")'"),
                        "U.S."=>array("x"=>93,"Y"=>118, "shape"=>"images/us.png",
                                 "anchor"=>"'docs/Mapper_us.pdf'target=_blank")));
```

Here we have a two points styled by `type = towns` (Tanzania and Kenia), a point with an image and a link to a document (U.S.), a general click event (`alert`) and a particular event (Birmania).

1.6 Use with JOOMLA!

Mapper needs the Sourcerer® plug-in which permits to insert JavaScript, PHP scripts, HTML tags, and CSS in an article; below is the sample for create a map (for handle the placements point).

```
{source}
<?php
include 'mapper/mapper.php';l
JHTML::script("mapper.js","mapper/js/",false);   // include javascript mapper functions
$aParams = array("image"=>"mapper/images/world.png",
                 "Shape"=>"Circle",
                 "alpha"=>1,
                 "points"=>array("Kenia"=>array("x"=>360,"Y"=>200),
                 "Tanzania"=>array("x"=>355,"Y"=>225),
                 "Birmania"=>array("x"=>483,"Y"=>190),
                 "U.S."=>array("x"=>93,"Y"=>118)));
echo createMap($aParams,true);
echo "<input type=button onclick='alert(mapper.returnPoints())' value='Save'>";
?>
{/source}
```

1.7 Styling map

The map has ID name `m_imgMap`; every point has the ID name equal to `ID`*`x`*`-`*`y`*, where *x* and *y* are the point coordinate in the map.

Here is a sample of styling.

```
<style>
   #m_imgMap {border: 1px solid black;padding:2px 3px}
</style>
```

1.8 Environment

PHP 5 GD support

1.9 Slider

This is a description of a widget used in demo.

- Include the Javascript script slider.js:

  ```
  <script type='text/javascript' src='js/slider.js'></script>
  ```

- Insert a `DIV` tag this must have an id name and a class name.

  The `DIV` can contain parameters for slide personalization in the form: `parm=value;...`

  The parameters, whit our defaults can be:

  `value=50` if the value is out of range is set a medium value of `from` and `to`.

  `from=0`

  `to=100`

  `color=gray`

  `colorcursor=silver`

  The *`idname`* prefix the id and name of the:

  - *`idname`*`Value` are ID and name of the field which collect the slide value
  - *`idname`*`CursorSlide` is the id of the cursor slide.

  The class name is used to locate the DIV to transform in slide widgets.

- Call the slider.build function at event `onLoad`, with the optional parameter `className`:

  ```
  <body bgcolor=teal onLoad='slider.build()'>
  ```

  The default class name is `slider`, if you would use another name:
  `slider.build(`*`myClassName`*`)`

Example:

```
<div id=alpha class=slider>
  value=0.5;from=0;to=1;color="lime";width=100;height=20,cursorcolor="olive"
</div>
```