Tutorial PLONK

1. Classes Plonk

Plonk

- ➔ cookie
  -> cookie.php
- ➔ database
  -> database.php
- ➔ filesystem
  -> csv.php
  -> directory.php
  -> file.php
- ➔ filter
  -> filter.php
- ➔ session
  -> session.php
- ➔ template
  -> template.php
- ➔ website
  -> controller.php
  -> website.php
- ➔ plonk.php

2. Structure Website

Structure

www

- core
  - css
    - images
    - *.css (all css files)
  - img
  - includes
    - config.php
    - functions.php
  - js
  - layout
    - layout.tpl
- library
  - plonk
  - <other libraries>
- modules
  - <name module>
    - <name module>.db.php
    - <name module>php
    - layout
      - <name view>.tpl
- index.php

Explanation

1. Index.php
   This php file does nothing more than include all the necessary files and hold an array of every module needed in the website.

2. Core->layout->layout.tpl
   This file contains the general html, for example the header, footer, … so each part of the site that's the same on each page.

3. Modules (modulename.php + viewname.tpl)

Each module has the same structure.

1. In the folder “module” you create a folder with the name of the module (WITHOUT capitals!!). (ex. home)
2. In this folder, you create a php file with the name of the module. (ex. home.php)
3. You also create a file called <nameModule>.db.php. (ex. home.db.php). This file will hold all the queries and other interaction with the database.
4. Next you create a folder named layout in this folder.
5. In this folder, all the views (page specific HTML) will be placed (ex. home.tpl).

4. Database-connection (modulename.db.php)

Each modules has its own db.php file, in which each interaction with the database can be placed.
In the example, we have written home.db.php and a static function to get the Users from the database.
In the home.php you then just have to write HomeDB.<nameOfTheMethod> to use the function of this class.

The name of the database, username and password is stored in the config.php, so you only have to fill this in once.

5. Basic implementation of the php.file

Here you can use the home.php as an example.

As you can see, there are 2 arrays that need to be initialized.
-> The first one is $views: here, you have to put all the views you have placed in the layout folder.
-> The second one is $actions: all the actions that need to take place after pressing a button on the page have to be placed here.

For each view, you will also have to write a “show” method.
In our example, we only have one view (home.tpl), so our method has to be showHome.

For each action, you will have to write a “do” method
In our example, this will be doLogin and doLogout.
The only thing extra you need to put in the “.tpl” file is this line, in front of the submit button

```
<input type="hidden" name="formAction" id="formLogin" value="doLogin" />
<input class="button" name="btnLogin" id="btnLogin" type="submit" value="Login"/>
```

6. Assign variables

{$name} : this is how you put the variable in your .tpl file

If you want to assign a variable in php which is located in the main layout tpl, you can do this by writing the next piece of code.

```
$this->mainTpl->assign('name', <value>);
```

If you want to assign a variable which is located in a view

```
$this->pageTpl->assign('name',<value>);
```

7. Iterations

Iterations can be just to fill lists (ex. Dropdownlists)
Iterations always start with 'i'!!
This code is used to fill an dropdownlist with data from the database.
($levels has all the ErasmusLineLevels from the database)

Code in tpl

```
<select name="userLevel" value="">
    {iteration:iUserLevel}
        {$userLevel}
    {/iteration:iUserLevel}
</select>
```

Code in php

```
$this->pageTpl->setIteration('iUserLevel');

foreach ($levels as $key => $value) {

    $this->pageTpl->assignIteration
        ('userLevel', '<option value=' . $value['Name'] . '> ' . $value['Name'] . '</option>');
    $this->pageTpl->refillIteration('iUserLevel');
}
$this->pageTpl->parseIteration('iUserLevel');
```

8. Options

You can use options to show or hide specific parts of the page, for example, if an admin is logged in, you can show differents parts then when a student is logged in

Code in tpl

```
{option:showAdmin}
        <text>
{/option:showAdmin}
```

Code in php

```
$this->pageTpl->assignOption('showAdmin');
```

9. Plonk Session

->Set:

PlonkSession::set('<name>',<value>)

ex. PlonkSession::set('LoggedIn',true);

->Check:

PlonkSession::exists('<name>');

ex. PlonkSession::exists('LoggedIn');

->Destroy:

PlonkSession::destroy();

Be careful if you use PlonkSession during implementation, because if you don't destroy the session (ex. fault in the website) you can get some weird output the next time you run your site.

10. Plonk::dump($variable)

This is a very useful function when it comes to debugging and tracking errors in your website. You can give any variable to Plonk::dump(..), so you can see what is stored in this variable.

11. Suggestions and questions

If you have any questions about the use of this library, you can contact us on nathan.vanassche@kahosl.be and stephane.polet@kahosl.be

We will also put our example website online so you all can have an example about how a website is built using the PLONK-library. This is a site with a single module, just so you can see the structure.