Templates and Template Engine

A template is any text or the contents of a file containing executable constructions of the template language and therefore can be modified in the work process of your application. ClickBlocks uses PHP as a template language. To simplify working with templates there is a special class - template engine **ClickBlocks\Core\Template**.

Consider working with a class for example of a simple HTML-template:

```
<html>
<head>
<title><?=$title;?></title>
</head>
<body>
<?=$text;?>
</body>
</html>
```

Create an instance of the class:

```
$tpl = new Template('/path/to/my/template');
```

The first parameter is the class constructor or the path to the template file or template itself.

Once an instance is created, we can get the template using the method **getTemplate()**:

```
$template = $tpl->getTemplate();
```

Or override template using method **setTemplate()**:

```
$tpl->setTemplate('/path/to/new/template');
```

As in the case of the constructor, the method parameter is the path to the template file or template itself.

Now we define the values of two template variables. You can do this by referring to the template variables as class properties (properties overload):

```
// Sets the value of variable "title".
$tpl->title = 'Some Title';
// Sets the value of variable "text".
$tpl->text = 'Hello World!';
```

The variables defined in this way (through properties overload) are local template variables. There are also global template variables. The difference between them is that the global variables are defined for all templates without exception. And local variables exist only for a particular template. If the name of a local variable has the same name of a global variable, the local variable value will override the global variable value.

You can create a global template variable as follows:

```
$tpl['foo'] = 'some vlaue';
```

Variable **foo** is defined as a global variable and will be available in all templates.

You can get an array of all global variables as follows:

```
print_r(Template::getGlobals());
```

You can also set all global variables at once:

```
Template::setGlobals(['var1' => 'val1', 'var2' => 'val2', ...], true);
```

The first method parameter -is an associative array of global variables. The second parameter of the method is responsible for merging the new

variables with the old global variables. The default value is FALSE.

Similarly, you can set (or get) the values of all local variables:

```
// Sets values of local variables.
$tpl->setVars(['var1' => 'val1', 'var2' => 'val2', ...], false);
// Gets values of local variables.
print_r($tpl->getVars());
```

Parameters of method **setVars()** are completely equivalent to parameters of method **setGlobals()**.

Once all variables in a template are defined, we can process the template and get the result. You can do this by calling method **render()**:

```
$result = $tpl->render();
```

You can also work with the object of template engine as a string (implementation of magic method **__toString()**).

```
echo 'Result is ' . $tpl;
```