Author: J.F. Benckhuijsen (jfbenck@users.sourceforge.net)

Version: 0.1

Kiss Generator Middleware

1. Problem description

Each application generaly exists out of three parts, with each their own responsibilities: User Interface, Middleware and Storage. This document describes how the middleware of the generated webapplication is created. In this webapplication there are three parts: Authentification, Sequence Rules and “the rest”. Authentification and the Sequence Rules are handled in a seperate document. This document describes “the rest”.

2. Sollution

In KISS an application generally out of actions and objects they apply on. Objects store information, but don't have any real methods themselves. Actions are the methods excecuted on these objects. They transform the values stored by the objects according to the status of the system and the parameters which are being passed to them.

For each action and each object a equivalent PHP class is created. The implementation of the object and action have some similarities. I'll discuss these first. After that i'll discuss the action and object specific parts of these classes.

2.1 Similarities

The name of these classes is generally the name of the object or action. Each of these classes have some attributes and methods. Generally they have three types of attributes:

1. An attribute named “id”. This attribute stores the id of the object in the database.
2. An attribute named “obj”. This attribute stores and object returned by the mysql_fetch_object() call, and stores the current values in the database.
3. For each attribute of the KissAction or the KissObject and equivalent attribute.

<FIXME>Storage of Reference Objects </FIXME>

<FIXME>Role Objects </FIXME>

Also there is are a number of standard methods. First there is a find_by_id($id) method, which retrieves an action or an object from the database by it's (unique) id. The object thus holds a reference to the object or action after that. Also there is a set and a get method for each attribute of the KissAction or the KissObject.

2.2 KissAction attributes and methods

A KissAction class has one additional method called the excecute method. This method contains the code used to excecute the action. This method is generally empty, except maybe for some checks. It's the responsibility of the application programmer to fill this method with the real code.

<FIXME> A KissAction class also has some additional attributes. These attributes store the id's of the KissObjects with which this action interacts.</FIXME>

2.3 KissObject attributes and methods

<FIXME>The store() method stores the changed values of the object into the database.</FIXME>

<FIXME>The delete() method deletes the selected method from the database. This method is generally called by an action if that action is the terminating action of that object type. This is not a fysical delete from the database, but only a logical delete </FIXME>

2.4 Error handling

2.5 Template KissAction and KissObject classes in PHP

```
<?php

class KissObject {
		var		$id;
		var		$obj;
		var		$attr;

		function find_by_id ($id) {

		}

		function store () {

		}

		function delete () {

		}

		function setAttr ($attr) {
				$this->attr = $attr;
		}

		function getAttr () {
				if ($obj) {
						$retval = $this->obj->attr;
				}
```

```
                        return $retval;
        }
}
?>

<?php
class KissAction {
        var             $id;
        var             $obj;

        function find_by_id($id) {

        }

        function excecute () {

        }

        function setAttr ($attr) {
                $this->attr = $attr;
        }

        function getAttr () {
                if ($obj) {
                        $retval = $this->obj->attr;
                }

                return $retval;
        }
}
?>
```