Zend_Controller_Action_Helper_ModelLoader

<ac:macro ac:name="unmigrated-inline-wiki-markup"><ac:plain-text-body><![CDATA[

<ac:macro ac:name="unmigrated-inline-wiki-markup"><ac:plain-text-body><![CDATA[

Zend Framework: Zend_Controller_Action_Helper_ModelLoader Component Proposal

Proposed Component Name	Zend_Controller_Action_Helper_ModelLoader
Developer Notes	http://framework.zend.com/wiki/display/ZFDEV/Zend_Controller_Action_Helper_ModelLoader
Proposers	Ralph Schindler Geoffrey Tran
Revision	1.0 July 31 2007 (wiki revision: 5)

Table of Contents

1. Overview
2. References
3. Component Requirements, Constraints, and Acceptance Criteria
4. Dependencies on Other Framework Components
5. Theory of Operation
6. Milestones / Tasks
7. Class Index
8. Use Cases
9. Class Skeletons

1. Overview

Zend_Controller_Action_Helper_ModelLoader aims to allow module based systems load model files from within its own module as well as outside modules. This will promote best practices within the Zend Framework.

2. References

3. Component Requirements, Constraints, and Acceptance Criteria

4. Dependencies on Other Framework Components

- Zend_Loader
- Zend_Controller_Front
- Zend_Controller_Action

- Zend_Controller_Action_Helper
- Zend_Exception

5. Theory of Operation

6. Milestones / Tasks

- Milestone 1: design notes will be published here

7. Class Index

- Zend_Controller_Action_Helper_ModelLoader

8. Use Cases

UC-01

```
...
            // load the DATA models

Zend_Controller_Action_HelperBroker::getExistingHelper('ModelLoader')->load(
                array('Blog_Users', 'Blog_Posts', 'Blog_Comments', 'Blog_Linkbacks'),
                ':moduleDir/models/Data'
                );

            // load ACL models

Zend_Controller_Action_HelperBroker::getExistingHelper('ModelLoader')->load(
                array(
                    'Blog_AclRole_User', 'Blog_AclResource_Post',
                    'Blog_AclAssert_UserIsPostOwner',
'Blog_AclAssert_UserIsCommentOwner',
                    'Blog_AclAssert_UserCanViewPost'
                    ),
                ':moduleDir/models/AccessControl'
                );
...
```

9. Class Skeletons

```
class Xend_Controller_Action_Helper_ModelLoader extends
Zend_Controller_Action_Helper_Abstract
{
    /**
     * $_modelPathSpec
     *
     * @var string
     */
    protected $_modelPathSpec = ':moduleDir/models';

    /**
     * $_modelPrefixSpec
     *
     * @var string
     */
    protected $_modelPrefixSpec  = ':moduleName_';

    /**
     * $_modelPostfixSpec
     *
     * @todo Postfixing of names?
     * @var string
     */
    //protected $_modelPostfixSpec = '';

    /**
     * setModelPathSpec()
     *
     * @param string $pathSpec
     * @return Xend_Controller_Action_Helper_ModelLoader
     */
    public function setModelPathSpec($pathSpec)
    {
        $this->_modelPathSpec = $pathSpec;
        return $this;
    }

    /**
     * setModelPrefixSpec()
     *
     * @param string $prefixSpec
     * @return Xend_Controller_Action_Helper_ModelLoader
     */
    public function setModelPrefixSpec($prefixSpec)
    {
        $this->_modelPrefixSpec = $prefixSpec;
        return $this;
    }

    /**
     * load()
     *
     * @param string $model
     * @param unknown_type $pathSpec
     * @param unknown_type $prefixSpec
     */
```

```
    public function load($models, $pathSpec = null, $prefixSpec = null)
    {
        $models = (array) $models;
        $modelPathSpec   = ($pathSpec !== null) ? $pathSpec : $this->_modelPathSpec;
        $modelPrefixSpec = ($prefixSpec !== null) ? $prefixSpec :
$this->_modelPrefixSpec;

        foreach ($models as $model) {

            if (class_exists($model)) {
                continue;
            }

            if (!isset($front)) {
                $front = Zend_Controller_Front::getInstance();
                $modules = $front->getControllerDirectory();
            }

            // strip the controller directory name to get the real module path.
            foreach ($modules as $moduleName => $moduleDir) {
                $modules[$moduleName] =
preg_replace('/\/'.$front->getModuleControllerDirectoryName().'$/', '', $moduleDir);
            }

            // init
            $validModule   = null;
            $strippedModel = $model;

            // find module based of what was supplied from the user
            if ($modelPrefixSpec != '') {
                foreach (array_keys($modules) as $checkModule) {
                    if (preg_match('/^'.str_replace(':moduleName', $checkModule,
$modelPrefixSpec).'/i', $model, $matches)) {
                        $validModule = $checkModule;
                        $strippedModel = preg_replace('/^'.$matches[0].'/i', '',
$model);
                        break;
                    }
                }
            }

            // no module based on name, find the module name we are currently in?
            if (!$validModule) {
                if ($this->_request instanceof Zend_Controller_Request_Abstract) {
                    $validModule = $this->_request->getModuleName();
                    $stripFrom = str_replace(':moduleName', $validModule,
$modelPrefixSpec);
                    $strippedModel = preg_replace('/^'.$stripFrom.'/i', '', $model);
                }
            }

            // if not a valid module, go default.
            if (!$validModule) {
                $validModule = 'default';
            }

            // translate the path to the place where models are stored
            $translatedPath = str_replace(':moduleDir', $modules[$validModule],
$modelPathSpec);
```

```
            // since we will use loadFile, we need to do our own Class->File (with
dir) mapping
            if (strstr($strippedModel, '_')) {
                $moreDir = substr($strippedModel, 0, strrpos($strippedModel, '_')+1);
                $strippedModel = str_replace($moreDir, '', $strippedModel);
                $translatedPath .= DIRECTORY_SEPARATOR . str_replace('_',
DIRECTORY_SEPARATOR, $moreDir);
            }

            // make sure to translate underscores to dir seperators
            $strippedModel = str_replace('_', DIRECTORY_SEPARATOR, $strippedModel);

            // load the file
            Zend_Loader::loadFile($strippedModel . '.php', $translatedPath);

            // if model doesnt exist now, we gots major problems
            if (!class_exists($model)) {
                throw new Zend_Controller_Action_Exception('Model by name ' . $model .
' not found in file ' . $strippedModel .'.php in path ' . $translatedPath);
            }

        }

    }

    public function direct($model, $pathSpec = null, $prefixSpec = null)
    {
        $this->load($model, $pathSpec, $prefixSpec);
        return $this;
    }
```

```
}
```

]]></ac:plain-text-body></ac:macro>
]]></ac:plain-text-body></ac:macro>