Table of Contents

1 Extension:IsAllowedHook.php

This code exhibits voodoo programming techniques. The most common of these is extending an instance's class at runtime after it has been instantiated, a technique that can be used to provide additional hooks into existing code without requiring modification of code-base files. For a list of all our scripts which exhibit voodoo, see Category:Code that uses voodoo.

```
<?php
# Extension:IsAllowedHook
# - Started: 2007-10-09
# - Licenced under LGPL (http://www.gnu.org/copyleft/lesser.html)

if ( !defined( 'MEDIAWIKI' ) ) die('Not an entry point.' );

define( 'ISALLOWED_VERSION', '0.0.0, 2007-10-09' );

$wgExtensionCredits['parserhook'][] = array(
        'name'        => "IsAllowedHook",
        'author'      => '[http://www.organicdesign.co.nz/nad User:Nad]',
        'description' => 'Adds a new hook called "IsAllowed" which is called from the User::isAllowed method.',
        'url'         => 'http://www.organicdesign.co.nz/Extension:IsAllowedHook.php',
        'version'     => ISALLOWED_VERSION
);

# This is the earliest hook I can find after which $wgUser is an instance of the User class
$wgHooks['userCan'][] = 'wfSetupIsAllowed';
function wfSetupIsAllowed() {

        # Return unless first call
        static $first = 0;
        if ( $first++ ) return true;
        global $wgUser;

        # Create a new User class ($User2) by extending the existing one with an overridden isAllowed method
        $User  = get_class( $wgUser );
        $User2 = $User.'2';
        eval( "class $User2 extends $User".' {
                function isAllowed( $action = "" ) {
                        $result = NULL;
                        wfRunHooks("IsAllowed",array( &$this, $action, &$result ) );
                        return $result === NULL ? $result = parent::isAllowed( $action ) : $result;
                }
        }' );

        # Replace the $wgUser object with an identical $User2 instance
        $oldUser = $wgUser;
        $wgUser  = new $User2();
        foreach ( array_keys( get_class_vars( $User ) ) as $k ) $wgUser->$k = $oldUser->$k;

        return true;
}
```

2 Extension:Livelets.php

This code is in our *Git* repository here.

Note: If there is no information in this page about this code and it's a MediaWiki extension, there may be something at mediawiki.org.

2.1 Notes

Add dev notes here, e.g. regarding the SWF live updates

3 Extension:NewUserMessage.php

This code is in our *Git* repository here.

Note: If there is no information in this page about this code and it's a MediaWiki extension, there may be something at mediawiki.org.

4 Extension:NoViewSource.php

Legacy: This article describes a concept that has been superseded in the course of ongoing development on the Organic Design wiki. Please do not develop this any further or base work on this concept, this is only useful for a historic record of work done. You may find a link to the currently used concept or function in this article, if not you can contact the author to find out what has taken the place of this legacy item.

This code exhibits voodoo programming techniques. The most common of these is extending an instance's class at runtime after it has been instantiated, a technique that can be used to provide additional hooks into existing code without requiring modification of code-base files. For a list of all our scripts which exhibit voodoo, see Category:Code that uses voodoo.

```
<?php
# Extension:NoViewSource
# - Licenced under LGPL (http://www.gnu.org/copyleft/lesser.html)
# - Author: [http://www.organicdesign.co.nz/nad User:Nad]{{Category:Extensions created with Template:Extension}}
# - Started: 2007-12-16

if (!defined('MEDIAWIKI')) die('Not an entry point.');

define('NOVIEWSOURCE_VERSION','1.0.2, 2007-12-18');

$wgExtensionFunctions[]        = 'wfSetupNoViewSource';
$wgExtensionCredits['other'][] = array(
        'name'        => 'NoViewSource',
        'author'      => '[http://www.organicdesign.co.nz/nad User:Nad]',
        'description' => 'Replaces the "view source" action with "edit" which links to login page.',
        'url'         => 'http://www.organicdesign.co.nz/Extension:NoViewSource.php',
        'version'     => NOVIEWSOURCE_VERSION
        );

function wfSetupNoViewSource() {
        global $wgHooks,$wgOut,$wgUser;
        if ($wgUser->isLoggedIn()) return;

        $wgHooks['SkinTemplateTabs'][] = 'wfNoViewSourceUpdateActions';

        # Create a new OutputPage class (Out2) which has its readOnlyPage (view source) method replaced with loginToUse
        class Out2 extends OutputPage {
                public function readOnlyPage( $source = null, $protected = false, $reasons = array() ) {
                        parent::loginToUse();
                        }
                }

        # Replace $wgOut with a sub-classed replica
        $oldOut = $wgOut;
        $wgOut  = new Out2();
        $vars   = get_class_vars('oldOut');
        if (is_array($vars)) foreach (array_keys($vars) as $k) $wgOut->$k = $oldOut->$k;
        }

function wfNoViewSourceUpdateActions(&$skin,&$actions) {
        if (isset($actions['viewsource'])) $actions['viewsource']['text'] = wfMsg('edit');
        return true;
        }
```

5 Extension:PublicCat.php

Legacy: This article describes a concept that has been superseded in the course of ongoing development on the Organic Design wiki. Please do not develop this any further or base work on this concept, this is only useful for a historic record of work done. You may find a link to the currently used concept or function in this article, if not you can contact the author to find out what has taken the place of this legacy item.

```
<?php
# - Licenced under LGPL (http://www.gnu.org/copyleft/lesser.html)
# - Author: [http://www.organicdesign.co.nz/aran Aran Dunkley]

if (!defined('MEDIAWIKI')) die('Not an entry point.');

define('PUBLICCAT_VERSION','1.0.7, 2008-10-28');

# Name of category in which to place articles to make them viewable from the public site
if (!isset($wgPublicCat)) $wgPublicCat = 'Public';

# Pattern to match with domain to determine if client accessing private wiki or public website
if (!isset($wgPublicCatPrivatePattern)) $wgPublicCatPrivatePattern = '/^wiki\\./';

$wgExtensionCredits['other'][] = array(
        'name'        => 'PublicCat',
        'author'      => '[http://www.organicdesign.co.nz/nad User:Nad]',
        'description' => 'Divides wiki into private and public by domain name, but allows unrestricted access to articles in a public category.
        'url'         => 'http://www.mediawiki.org/wiki/Extension:PublicCat',
        'version'     => PUBLICCAT_VERSION
);

# Hook in after article and title prepared
# - should be able to hook in earlier than this,
# - but note that it must be a hook that always executes, even for specialpages
$wgHooks['OutputPageBeforeHTML'][] = 'wfPublicCatValidate';

# Determine if the request is private or public from the domain and pattern
$wgPublicCatRequestIsPrivate = preg_match($wgPublicCatPrivatePattern,$_SERVER['HTTP_HOST']);
$wgPublicCatRequestIsPublic  = !$wgPublicCatRequestIsPrivate;

# Disable all actions except view for anonymous users
$wgGroupPermissions['*']['createaccount'] = false;
$wgGroupPermissions['*']['read']          = $wgPublicCatRequestIsPublic;
$wgGroupPermissions['*']['edit']          = false;
$wgGroupPermissions['*']['createpage']    = false;
$wgGroupPermissions['*']['createtalk']    = false;

# Allow unrestricted access to login and css's
$wgWhitelistRead = array('Special:Userlogin','-');
if (eregi('\\.css$',$_REQUEST['title'])) $wgWhitelistRead[] = $_REQUEST['title'];

# If request is public, restrict action to view, raw or render
if ($wgPublicCatRequestIsPublic) {
        if ($_REQUEST['action'] != 'view' && $_REQUEST['action'] != 'raw' && $_REQUEST['action'] != 'render')
                $_REQUEST['action'] = 'view';
}

# Function to check if passed a title is in a category
function inCat($title, $cat) {
        if (!is_object($title)) $title = Title::newFromText($title);
        $id   = $title->getArticleID();
        $dbr  = &wfGetDB(DB_SLAVE);
        $cat  = $dbr->addQuotes($cat);
        $cl   = $dbr->tableName('categorylinks');
        return $dbr->selectRow($cl, '0', "cl_from = $id AND cl_to = $cat", __METHOD__);
}

# Return a 404 not found error if request is public but requested title is not in the public cat
function wfPublicCatValidate() {
        global $wgPublicCat,$wgPublicCatRequestIsPublic,$wgTitle;
        if ($wgPublicCatRequestIsPublic && !wfPublicCatInCat($wgTitle,$wgPublicCat))
```

```
                wfHttpError(404,"404 Not Found","The requested URL was not found on this server!");
        return true;
}
```

6 Extension:TransformChanges.php

Legacy: This article describes a concept that has been superseded in the course of ongoing development on the Organic Design wiki. Please do not develop this any further or base work on this concept, this is only useful for a historic record of work done. You may find a link to the currently used concept or function in this article, if not you can contact the author to find out what has taken the place of this legacy item.

This code is in our *Git* repository here.

Note: If there is no information in this page about this code and it's a MediaWiki extension, there may be something at mediawiki.org.