Blade Pack Development Notes

Blade Packs Interface

The core system as i'm sure your aware, comes as a stripped to the max CMS solution, this approach was taken to ensure that you only run the functionality you need, by installing functionality you need. Not only does this approach give you a leaner quicker system, only having what you need, it means that you will not be prone to any vulnerabilities that may be yet un-found in functionality you don't use. Blade packs are add-on extra's to the core system that can add anything from extra functionality, through to change the theme of your site. It is basically a name for a general purpose plugin. Each blade pack several things, title, description, author, socket allocation, and blades.

XML settings

Title – The actual blade pack name.
Version – The version of the blade pack.
Class – The class type, either system, theme, language or upgrade (for system upgrades only).
Author – The name of who wrote the blade pack
Description – A brief summary of what the blade pack achieves.

Blade pack file

Socket allocation – This is how we interface the blades into the system, by allocating a blade to a socket, we can tell the system where to run the blade in the system code.
Blades – These are simply functions re-branded, they correspond to the allocated sockets above. When a socket is reached in the system code, if a blade is allocated to a socket and is set to active, then the code in the blade is run.

Installation

Blade packs are installed via the blade pack installer, located in the blade manager area of the administration console. Blade packs are distributed as a parcel (zip file), this parcel is self extracted and installed automatically. When the blade manager is run in the admin area the blade packs are searched, listed and broken down into class type, you then have the option of which blade packs to activate or deactivate by searching for it in it's correct class type and clicking the activate, deactivate links.

Alternatively, you can also remove blade packs completely from the system by clicking the delete link.

Important Notes When Writing Blade Packs

When writing blade packs there are several things you need to know.

First, more than one blade may be allocated to a single socket. They will be loaded as they are

referenced.

Second, if your blade pack contains sensitive areas that should only be run from within admin or that restrict access based on the user level of the person logged in, please ensure you protect these blades from unauthorized access. Failure to do this may result in your blade pack having sensitive areas open to the public front end or to users without the correct level of access, which could be a security issue. Use the session variable $_SESSION['adminType'] to determine the type of user logged in, either ‘user’, ‘admin’ or ‘sadmin’ (super administrator). Alter your code accordingly.

Third, there are two types of socket in which to apply a blade, ones that only output data, like messages to the screen, and ones that send a variable to the blade for manipulation.

Forth, there are two types of variable that can be sent from a socket, ones that send by reference and ones that send the actual variable. Any socket that sends a variable by reference, basically means you can alter the variable to affect the core system code. Those that send the actual variable have no means to return changes made to the variable.

e.g. variables sent by reference (&$name) support read/write access meaning any changes to the variable in the blade will result in the variable being altered in the core system code.

variables sent ($name) support only read access from variable, meaning that any changes to the variable in the blade will not affect the variable in the core system code and will be lost on completion of blade code.

Full Socket List

Below is a non exhaustive list of available sockets and spare variables within the razorCMS code, this list is not final, but details the most common and widely used sockets. Before attempting to write any blade pack, please spend some time familiarizing yourself with the code around each socket. Also try looking at an already existent blade pack to see how it integrates into the system. If you require a socket to be placed into core code to help integrate a feature for a blade pack, please contact the project lead via the support forum.

Sockets for output to admin front end

These are general purpose sockets, for adding extra ouput to the administration area of the site. They could be used to add links in for configuring new functionality, or maybe outputting extra html in certain parts of the administration area, like adding an extra footer, maybe even add to the default header.

admin-xhtml-head

Adds blades to the head section of the administration page, inside the head tags <head></head>. Blade

contents are displayed as the last declaration in the head section, just before the closing head tag.

admin-xhtml-header

Adds blades to the header of the administration page, blade contents are displayed directly under the site slogan.

admin-xhtml-topnav

Adds blades to the topnav section of the administration area, can be used to add extra links into the administration area, any content is displayed after the normal topnav links.

admin-xhtml-content

Adds blades to the content section of the admin page, blade contents will appear after all other content on the admin page.

admin-xhtml-footer

Adds blades to the footer section of the admin page, blade contents appear after all other footer content.

admin-xhtml-endofdoc

Adds blades to the end of the admin page, just before the closing body tag. This is a special purpose socket that was mainly added for google analytics integration, which is best placed as the last thing in a html document.

Sockets for output to public front end

These are general purpose sockets, for adding extra ouput to the public area of the site. They can be used for outputting any kind of information that needs to be displayed on the public front end. Use these sockets to display anything from extra header info, links, sidebars, anything that would be common across all public facing pages.

public-xhtml-head1

Adds blades to the head section of the public facing pages, inside the head tags <head></head>. Blade contents are displayed as the last declaration in the head section but before public-xhtml-head2 blade content, just before the closing head tag.

public-xhtml-head2

Adds blades to the head section of the public facing pages, inside the head tags <head></head>. Blade contents are displayed as the last declaration in the head section after public-xhtml-head1 blade content, just before the closing head tag.

public-xhtml-header

Adds blades to the header of the public facing pages, blade contents are displayed directly under the site slogan.

public-xhtml-topnav

Adds blades to the topnav section of the public facing pages, can be used to add extra links into the administration area, any content is displayed after the normal topnav links.

public-xhtml-content

Adds blades to the content section of the public facing pages, blade contents will appear after all other content on the page that is being displayed.

public-xhtml-leftnav

Adds blades to the leftnav section of the public facing pages, blade contents are displayed after all left navigation links are displayed and before infobar contents are displayed.

public-xhtml-leftbar

Adds blades to the leftbar section of the public facing pages, blade contents are displayed after all lnfobar contents are displayed.

public-xhtml-footer

Adds blades to the footer section of the public facing pages, blade contents appear after all other footer content.

public-xhtml-endofdoc

Adds blades to the end of the public facing pages, just before the closing body tag. This is a special purpose socket that was mainly added for google analytics integration, which is best placed as the last thing in a html document.

admin login/logout sockets

These are special purpose sockets for use with manipulating the login, logout process of the admin area

of the system.

admin-pre-login-logout

Adds blades before the login process is started when accessing the admin section of the system. Blade contents appear first before any login information is displayed.

admin-pre-login-form

Adds blades before the login form is displayed to the screen. Blade contents appear before the login form.

admin-login-form >> &$form

Adds blades in after the login form has been generated, but before it is displayed on the screen. The generated login form is sent by reference to blades allocated to this socket. Use mainly to change the actual login form using a blade by changing the generated form that is sent by reference.

razorArray settings sockets

These sockets are mainly used for altering the actual contents of the razorArray. The razorArray is the major array that contains non sensitive settings, which is retrieved from the razor_data.txt file.

admin-edit-razorarray >> &$razorArray

Adds blades that modify the razorArray major settings array from within the admin area of the system. razorArray is sent by reference to the blade allocated to this socket.

public-edit-razorarray >> &$razorArray

Adds blades that modify the razorArray major settings array from within the public facing pages. razorArray is sent by reference to the blade allocated to this socket.

Theme changing sockets

This group of sockets is used for altering the themes of both the public and admin facing front ends.

admin-change-theme >> &$theme

Adds blades that change the admin theme, the variable $theme is set to the default theme path which is part of the system and is always present. The variable is then sent by reference to any allocated blades where the theme path is changed for the new one.

public-change-theme >> &$theme

Adds blades that change the public theme, the variable $theme is set to the default theme path which is part of the system and is always present. The variable is then sent by reference to any allocated blades where the theme path is changed for the new one.

public-css-address >> &$css

Adds blades that can modify the public facing CSS path. Public theme requires this due to SEF URL Support for public facing pages. Use when modifying the public theme, sends variable $css by reference to any allocated blades so the path to CSS file can be changed.

editor-css-path >> &$editorCSS

Used to set the active editor css, this will enable the active editor in the content creation areas of the admin side, to display the content using a separate editor css file modeled on the website theme. Use this socket in blade pack themes in conjunction with a css file created for editors.

url manipulation sockets

These are special purpose sockets that are used to manipulate any url's that are generated and displayed onto the users screen, and any urls that are clicked on. Primarily used for SEF URL support, for translating urls to a different form, and back again.

url-in >> &$slab

Adds blades that modify incoming url's, used in SEF URL blade pack to strip the data from a SEF URL to obtain the slab name.

url-out >> &$urlFormat

Adds blades that modify outgoing url's, used in SEF URL blade pack to convert a php style url, into a SEF URL for output to the users screen.

Content editor socket

A special purpose socket, designed to integrate any kind of html editor into all areas where html content is created or edited.

editor >> &$te

Adds blades to content creation pages within the admin area, is used primarily to offer html editor support when creating or editing content. In default mode, content is added by text box, the content is

then stored in a variable which is sent by reference to allocated blades. This info can then be changed using the particular html editor you wish altering the variable sent to the blade pack.

Add extra page to admin

This is a special purpose socket used solely for adding extra pages to the administration area.

admin-select-page >> &$defaultPage

Adds blades that create an extra page within the administration area, under default conditions, there is a set number of pages within the admin area, if none are selected, the default page (admin home) is loaded. When using this socket, if the default set pages do not match, the default is stored in a variable and then sent by reference to any allocated blades so further checks can be performed for any extra pages added by the blade. Think of this as a way to extend the amount of pages in the admin area by adding on to the page select function.

Media directory contents

This is a special purpose socket, designed to interact with the media dir and files within the media manager.

admin-media-data >> &$mediaDir, &$mediaFiles

Adds blades that to interact with the media data, this socket sends the current media dir and all files present in variables by reference. Use this socket to affect media data that is read in, like stripping out certain file types, or allowing only jpg files.

Add extra info to page creation pages in admin area

These sockets are used when you want to add any kind of extra info to any pages in the admin area that change or edit content, like the addition of extra text box's, text, selection boxes and so on.

admin-xpage-info-output >> &$extraInfo

Used to output any extra info you may want to display in the add new page part of the admin area. The variable $extraInfo is sent by reference which is typically set to nothing by default. Any allocated blades can place content into the variable that will be outputted to the screen before any of the default form info.

admin-xpage-info-output-ed >> &$extraInfo, $slab

Used to output any extra info you may want to display in the edit page part of the admin area. The variable $extraInfo is sent by reference which is typically set to nothing by default. $slab variable

which is the name of the page being edited is also sent to help the blade know which page the extra content is for. Any allocated blades can place content into the variable $extraInfo that will be outputted to the screen before any of the default form info.

admin-xpage-info-input >> $slab

Used to collect any additional info that may have been outputted using the sockets above. $slab is sent to any allocated blades so the blade knows which page to collect the data for. Any data can then be collected using post methods.

admin-xinfo-info-output >> &$extraInfo

Used to output any extra info you may want to display in the add infobar content part of the admin area. The variable $extraInfo is sent by reference which is typically set to nothing by default. Any allocated blades can place content into the variable that will be outputted to the screen before any of the default form info.

admin-xinfo-info-output-ed >> &$extraInfo, $slab

Used to output any extra info you may want to display in the edit infobar content part of the admin area. The variable $extraInfo is sent by reference which is typically set to nothing by default. $slab variable which is the name of the page being edited is also sent to help the blade know which page the extra content is for. Any allocated blades can place content into the variable $extraInfo that will be outputted to the screen before any of the default form info.

admin-xlink-info-output >> &$extraInfo

Used to output any extra info you may want to display in the add external link part of the admin area. The variable $extraInfo is sent by reference which is typically set to nothing by default. Any allocated blades can place content into the variable that will be outputted to the screen before any of the default form info.

When deleting a page through admin

This is a special purpose socket, used to offer extra functionality when deleting a page.

admin-on-page-delete >> $slab

Used to add extra functionality when deleting pages, sends the variable $slab to any allocated blades, so the blade knows which page is being deleted.

Language support socket

This is a special purpose socket designed to interface external language utilities with the core system.

language-select >> &$text

Used to add language support to the core system outputted text, sending the variable $text to any allocated blades. This should offer support for on the fly language translators or a language file that references text to another language from a table.

Add functionality pages, in page blades

This is a special purpose socket with one aim, to add blades into content in a special way. In page blades are a way to add blades to the page when creating or editing them, placing the blades anywhere you want to in the page content in a special way so they may be stripped out in real time running the appropriate blade when a user browses to the page.

admin-add-page-function >> &$addFunction,&$content

Used as a way of placing blades onto content creation pages, the variables $addFunction and $content are sent to any allocated blades so they may return the special blade call in the page content.

admin-add-info-function >> &$addFunction, &$content

Used as a way of placing blades onto infobar creation pages, the variables $addFunction and $content are sent to any allocated blades so they may return the special blade call in the page content.

admin-add-link-function >> &$addFunction, &$content

Used as a way of placing blades onto external link creation pages, the variables $addFunction and $content are sent to any allocated blades so they may return the special blade call in the page content.

Scan content before outputting

These are special purpose blades primarily wrote to scan content on the fly, for use with in page blades. However they may be used for other purposes where scanning of outputted content may be required i.e. content filters.

scan-content-slug >> &$contentSlug

Used to scan content on the fly before outputting to the screen. The variable $contentSlug is sent to any allocated blade which is content to be outputted to the screen. Any allocated blades may then choose to alter the content in any way. Was primarily placed to help strip out special in page blade calls, so the in

page blade may be run midway through content on a page. However may be used for other purposes.

scan-content-info >> &$contentInfo

Used to scan info bar content on the fly before outputting to the screen. The variable $contentInfo is sent to any allocated blade which is content to be outputted to the screen. Any allocated blades may then choose to alter the content in any way. Was primarily placed to help strip out special in page blade calls, so the in page blade may be run midway through info bar content in the info bar. However may be used for other purposes.

Link manipulation sockets

These sockets are special purpose sockets that are used to manipulate the way links are generated or listed.

create-link-from-cat-before

Used to add extra information before a link, use this socket to add information in between links when they are generated.

create-link-from-cat-after

Used to add extra information after a link, use this socket to add information in between links when they are generated.

edit-link-creation >> &$returnLink, $slab, $title

Used to edit the actual link when it is made, sends the slab name and title for informational purposes, to determine what the link is for, and also sends the actual link by reference so it may be altered by the blade.

General purpose spare sockets

These are all general purpose spare sockets placed in the index pages of both the public front end and the admin front end. They where placed into the index pages to offer any kind of large scale functionality add on that may require a good handle in the index part of the system (the place where code is run from).

public-index-socket1
public-index-socket2
public-index-socket3
admin-index-socket1
admin-index-socket2
admin-index-socket3

Spare sockets to add settings to settings manager

These are general purpose sockets placed inside the settings manager code. They are used to display extra settings that may be needed in the settings manager by blades that require non sensitive settings to be saved. In addition to the display sockets, there are also spare sockets for reading the data back in that any blades may have displayed.

admin-settings-display1
admin-settings-display2
admin-settings-display3
admin-settings-add1
admin-settings-add2
admin-settings-add3

Spare variables for passing between blade packs

These a general purpose spare variables that may be passed from one blade to another, they may be used by blades storing variables using the global method so they can be retrieved by other blades using the same method.

PLEASE NOTE : When using the spare variables, it is important to treat them as temporary storage. When using spare variables it is important to follow some simple rules.

When first setting a spare variable, check to see if it is already set by using the isset() function. If it is already in use, try another. Once you have found an empty spare variable, store your data with the name of your blade followed by : then the data you want to store (bladename:data). Use string functions to strip the blade name and colon to obtain your data.

When accepting data from a spare variable, please be aware that you need to check it's origin, use string functions to check the blade name before the colon to ensure it is coming from the correct place before using the data.

When you have finished using the spare variable in your blade, set the contents to nothing (sparevar1 = '';), this way it will be available for others to use. Spare variables are temporary storage to pass amongst blade packs, so once you have retrieved your variable data save it to a local variable in the blade and set the spare variable to nothing.

Public index variables
$spareVar1
$spareVar2
$spareVar3
$spareArray1
$spareArray2
$spareArray3

Admin index variables
$spareVar1;
$spareVar2;
$spareVar3;
$spareArray1
$spareArray2
$spareArray3