

Mamur Framework



Contents

Package mamur Classes	1
Class mailclass	1
Var \$bcc	1
Var \$cc	1
Var \$colour	1
Var \$error	1
Var \$from	1
Var \$headers	1
Var \$mailbody	1
Var \$message	1
Var \$replyto	2
Var \$sign	2
Var \$subject	2
Var \$to	2
Constructor mailclass	2
Method copyto	2
Method message	2
Method postmessage	2
Method send	3
Class pageMeta	3
Constructor __construct	3
Method updateFromDataSet	3
Package mamur Procedural Elements	5
mainController	5
Class mamurController	5
Var \$controller	6
Var \$model	6
Var \$view	6
Method getController	6
Method getView	7
Method processUri	7
Method response	7
static configuration class mamurConfig	9
mamurModel	10
Class mamurConfig	10
Method createPlugIns	11
Method getInstance	11
Method getPluginDir	11
Method getPlugInsToLoad	11
Method persistSetting	12
Method processConfig	12
Method relativeDir	13
Method runPlugIns	13

Method setGlobal	13
Method setPlugIn	13
Method upDateConfig	14
Method get	14
Class mamurConfigData	15
Constructor construct	15
Method getAll	16
Method get	16
Method isset	16
Method set	16
Method unset	17
Class mamurDataObject	17
Constructor construct	18
Method appendRecord	18
Method back	19
Method deleteRecord	19
Method getAll	19
Method getAttribute	19
Method getCurrentRecordNumber	20
Method getRecord	20
Method getStatus	20
Method last	20
Method modified	20
Method next	21
Method persist	21
Method read	21
Method setAttribute	21
Method setCurrentRecordNumber	21
Method setStatus	22
Method get	22
Method isset	22
Method set	23
Method unset	23
Class mamurModel	24
Var \$countSerial	24
Var \$dataObjects	25
Var \$defaultPageExt	25
Var \$defaultPageName	25
Var \$error404PageName	25
Var \$global	25
Var \$hostdomain	26
Var \$hostScheme	26
Var \$inSession	26
Var \$lastErrorPage	26
Var \$locid	26
Var \$locidAccepted	27
Var \$logOutFlag	27
Var \$mamurBaseDir	27
Var \$mamurlogDir	27

Var \$mamurPluginDir	27
Var \$mamurSystemDir	28
Var \$mamurURL	28
Var \$oldSession	28
Var \$options	28
Var \$pageDir	28
Var \$pageDirList	29
Var \$pageExt	29
Var \$pageName	29
Var \$pagePagePrintCallBack	29
Var \$pageProcessCallBack	29
Var \$pageQuery	30
Var \$pageURL	30
Var \$phpParameters	30
Var \$serverBaseRequestCallBack	30
Var \$session	30
Var \$sessionClearCallBack	31
Var \$subdomain	31
Var \$tags	31
Var \$templateTags	31
Var \$themesDir	31
Var \$topdomain	32
Var \$urlCallBack	32
Var \$webBaseDir	32
Var \$xmlPageType	32
Constructor construct	32
Method checkLogOut	33
Method convertToUserZone	33
Method decrypt	33
Method deleteDataObject	34
Method dirListToDataObject	34
Method doPagePagePrintCallBacks	34
Method encrypt	35
Method getApiSalt	35
Method getContentBase	35
Method getContentList	36
Method getCookieSerialNo	36
Method getDataBaseDir	36
Method getDataObject	36
Method getEditStatus	37
Method getEncryptedSession	37
Method geterror404PageName	37
Method getErrorPage	38
Method getGlobal	38
Method getHomeUri	38
Method getHostDomain	38
Method getHostScheme	38
Method getMamurBaseDir	38
Method getMamurlogDir	39

Method <u>getMamurSystemDir</u>	39
Method <u>getMamurUrl</u>	39
Method <u>getMamurUserDir</u>	39
Method <u>getNonce</u>	40
Method <u>getNonMamurPage</u>	40
Method <u>getOption</u>	40
Method <u>getPageContent</u>	41
Method <u>getPageDir</u>	41
Method <u>getPageDirList</u>	41
Method <u>getPageExt</u>	41
Method <u>getPageName</u>	41
Method <u>getParameters</u>	42
Method <u>getPhpBase</u>	42
Method <u>getPluginDir</u>	42
Method <u>getRandomString</u>	42
Method <u>getSalt</u>	43
Method <u>getServerBase</u>	43
Method <u>getSharedContent</u>	44
Method <u>getSharedContentBase</u>	44
Method <u>getSubDomain</u>	45
Method <u>getTag</u>	45
Method <u>getTagData</u>	45
Method <u>getTemplateFile</u>	45
Method <u>getTopDomain</u>	46
Method <u>getTopPageDir</u>	46
Method <u>getUrl</u>	46
Method <u>getUser</u>	46
Method <u>GetUserIP</u>	46
Method <u>getUserZoneOffset</u>	46
Method <u>getWebRoot</u>	47
Method <u>getXmlPageType</u>	47
Method <u>isError404Page</u>	47
Method <u>isLoggedIn</u>	47
Method <u>locidCookie</u>	48
Method <u>openPageMeta</u>	48
Method <u>pageProcessHookContinue</u>	48
Method <u>pageTime</u>	48
Method <u>passParameters</u>	49
Method <u>pbkdf2</u>	49
Method <u>processPageMetaData</u>	50
Method <u>processXML</u>	50
Method <u>readDataObjects</u>	50
Method <u>readPageXML</u>	50
Method <u>registerPagePrintFunction</u>	51
Method <u>registerPageProcessFunction</u>	51
Method <u>registerServerBaseFunction</u>	51
Method <u>registerSessionClearFunction</u>	51
Method <u>registerSessionLogoutFunction</u>	52
Method <u>registerUrlFunction</u>	52

Method relativeDir	52
Method removePage	53
Method rmdir	53
Method saveDataObjects	53
Method setEditStatus	54
Method setErrorPageTemplate	54
Method setGlobal	54
Method setHostData	54
Method setLocidCookie	55
Method setNonce	55
Method setOption	56
Method setPageUri	56
Method setSessionCookie	56
Method setTag	57
Method setTagList	57
Method setUpSession	57
Method setUser	58
Method timeStampToUserData	58
Method unique serial	58
Method userLogin	59
Method userLogout	59
mamurView	60
Function mamurViewReplacePlaceholder	60
Class mamurForm	60
Constructor construct	61
Method doEndform	61
Method doForm	61
Method doInput	62
Method doOption	62
Method doSelect	62
Method doTextarea	62
Method endSelect	62
Method fldAttrib	62
Method generalPlaceholder	63
Method optionList	63
Method validate	63
Class mamurPlaceholders	63
Constructor construct	64
Method all	64
Method build random	65
Method date	65
Method endform	65
Method endselect	65
Method form	66
Method generalPlaceholder	66
Method globalTag	66
Method http header	66
Method input	67
Method mamur	67

Method nonce	67
Method option	67
Method optionlist	68
Method page_content	68
Method page_selected	69
Method page_timer	69
Method page_timerms	69
Method php	69
Method random	69
Method select	70
Method shared	70
Method tag	70
Method textarea	71
Method title	71
Method unique_serial	71
Class mamurView	71
Var \$mamur	72
Var \$model	72
Var \$oddeven	72
Var \$pageBuild	73
Var \$pageBuildId	73
Var \$pageOutput	73
Var \$pagePhp	73
Var \$placeholderClasses	73
Var \$templateTags	74
Var \$urlDir	74
Constructor construct	74
Method defaultErrorPage	74
Method directPhpView	74
Method doDatePlaceholder	74
Method doGlobalPlaceholder	75
Method doNoncePlaceholder	75
Method dopage_pageTimerPlaceholder	75
Method dopage_timermsPlaceholder	75
Method doPhpAndPrint	76
Method doPhpPlaceholder	76
Method doRandomPlaceholder	76
Method doTagPlaceholder	76
Method doUniqueSerialPlaceholder	77
Method getPlaceholder	77
Method includePageFile	77
Method insertPlaceholder	77
Method metaContentStr	78
Method printTemplatePage	78
Method processPlaceholders	78
Method redirect	78
Method setModel	79
Method showBuiltPage	79
Method templatedView	79

start up	80
Function mamurAutoClassLoad	80
Function mamurErrorHandler	80
Function mamurExceptionHandler	81
Function mamurShutDown	81
Class mamurStart	81
Method setup	82
configtests.php	83
core mamur model tests.php	84
dataobject tests.php	85
modeldataobject tests.php	86
Class coreConfigTests	86
Method setUpBeforeClass	86
Method testConfigClasses	86
Method testConfigGlobals	87
Method testConfigPersist	87
Method testConfigPlaceholders	87
Method testConfigSettings	87
Method testConfigSetup	88
Method testSetConfigData	88
Class coreMamurModeltest	88
Var \$model	88
Method setUpBeforeClass	89
Method testGetPageContent	89
Method testGetSharedContent	89
Method testMappedContentbyFirstSection	89
Method testModel	89
Method testSetUri	89
Class dataObjectTests	89
Method setUpBeforeClass	89
Method testDataObjectAttributes	90
Method testDataObjectConstruct	90
Method testDataObjectDelete1stRecord	90
Method testDataObjectRecords	90
Method testDataObjectSelectRecord	90
Method testDataObjectSetting	91
Class modelDataObjectTests	91
Var \$model	91
Method setUpBeforeClass	92
Method testGetDataObject	92
Method testSaveDataObjects	92
Method testSaveDataObjects2fish	92
Method testSaveDataObjects3Des	92
Method testSaveDataObjectsNone	93
Appendices	94
Appendix A - Class Trees	95
mamur	95

Package mamur Classes

Class mailclass *[line 20]*

- **Package** mamur

mailclass::\$bcc

mixed = [line 31]

mailclass::\$cc

mixed = [line 30]

mailclass::\$colour

mixed = [line 28]

mailclass::\$error

mixed = [line 29]

mailclass::\$from

mixed = [line 25]

mailclass::\$headers

mixed = [line 24]

mailclass::\$mailbody

mixed = [line 26]

mailclass::\$message

mixed = [line 23]

mailclass::\$replyto

mixed = [line 32]

mailclass::\$sign

mixed = [line 27]

mailclass::\$subject

mixed = [line 22]

mailclass::\$to

mixed = [line 33]

Constructor *void* function mailclass::mailclass([\$cr_override = "\r\n"]) *[line 35]*

Function Parameters:

- **\$cr_override**

void function mailclass::copyto(\$copy) *[line 98]*

Function Parameters:

- **\$copy**

void function mailclass::message(\$colour, \$subject, \$body, \$from, \$sign) *[line 61]*

Function Parameters:

- **\$colour**
- **\$subject**
- **\$body**
- **\$from**
- **\$sign**

void function mailclass::postmessage(\$colour, \$subject, \$intro, \$postarray, \$from, \$sign) *[line 70]*

Function Parameters:

- **\$colour**
- **\$subject**
- **\$intro**
- **\$postarray**
- **\$from**
- **\$sign**

void function mailclass::send(\$name, \$to) [*line 104*]

Function Parameters:

- **\$name**
- **\$to**

Class pageMeta

[*line 2*]

- **Package** mamur

Constructor *void* function pageMeta::__construct() [*line 3*]

- **Access** public

void function pageMeta::updateFromDataSet(\$xmlPagefile, \$dataset) [*line 7*]

Function Parameters:

- **\$xmlPagefile**
- **\$dataset**

- **Static**
- **Access** public

Package mamur Procedural Elements

mainController

This file contains the static main Controller Class - mamurController Control may be dispatched to other controllers and mvc patterns according to configuration xml.

Licence: This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, version 3 of the License. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

- **Package** mamur
- **Sub-Package** core
- **Author** Martin Marsh < martinmarsh@sygenius.com>
- **Version** 110
- **Copyright** Copyright (c) 2011, Sygenius Ltd
- **Release** Mamur 1.10
- **Mvc** controller
- **Name** mainController
- **License** [GNU Public License, version 3](#)

Class mamurController

[line 43]

mamurController is a static class which is always the top level

controller. For simple pages it just processes the url according to configuration xml which defines the tags which can be used to build a page using templates. For more complex

dynamic pages configuration xml may define another controller which optionally may run a different mvc pattern to respond to certain page requests

- **Package** mamur
- **Sub-Package** coreController
- **Abstract Element**

mamurController::\$controller

mixed = [line 45]

- **Static**
- **Access** protected

mamurController::\$model

mixed = [line 45]

- **Static**
- **Access** protected

mamurController::\$view

mixed = [line 45]

- **Static**
- **Access** protected

void function mamurController::getController() [line 119]

Get controller gets the current controller dispatched by processUri

- **Static**
- **Access** public

void function `mamurController::getView()` [*line 127*]

Get view gets the current view instance

- **Static**
- **Access** public

void function `mamurController::processUri()` [*line 54*]

This method processes all page requests either passing control to another controller or responds directly to the request

- **Static**
- **Access** public

unknown_type function `mamurController::response($uri)` [*line 138*]

Function Parameters:

- `$uri` **\$uri** - address of a page

response method has the control steps to respond to a url and update print buffer.

- **Static**
- **Access** protected

static configuration class mamurConfig

This file contains the configuration class

- **Package** mamur
- **Sub-Package** config
- **Author** Martin Marsh < martinmarsh@sygenius.com>
- **Version** 105
- **Copyright** Copyright (c) 2011, Sygenius Ltd
- **Release** Mamur 1.10
- **Mvc** model
- **Name** static configuration class mamurConfig
- **License** [GNU Public License, version 3](#)

mamurModel

This file contains the main model Class - mamurModel

Licence: This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, version 3 of the License. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

- **Package** mamur
- **Sub-Package** core
- **Author** Martin Marsh < martinmarsh@sygenius.com>
- **Version** 110
- **Copyright** Copyright (c) 2011, Sygenius Ltd
- **Release** Mamur 1.10
- **Mvc** model
- **Name** mamurModel
- **License** [GNU Public License, version 3](#)

Class mamurConfig

[line 35]

The class provides a single point of reference to get all configuration details

Configuration is set up in index.php, the main configuration file configuration.xml. The bootstrap is responsible for setting up this config class by calling the method processConfig. Configuration details can then be read directly by getting the appropriate data class and using the magic __get methods to get the appropriate variables; for further details see mamurConfig::\$globalSet. Also Magic set method allows configuration data to be added at run time but will be lost when the request terminates. To update the xml configuration an explicit config update and flush must be called. class methods can write and persist configuration details to the configuration.xml file or perform configuration related tasks. Do not modify - core system class overwritten by updates

- **Package** mamur
- **Sub-Package** coreModel

void function mamurConfig::createPlugins() [line 328]

Instantiates each plugin and saves the object in a config

- **Static**
- **Access** public

void function mamurConfig::getInstance() [line 50]

- **Static**
- **Access** public

void function mamurConfig::getPluginDir([\$subdir = ""]) [line 352]

Function Parameters:

- **\$subdir**

- **Static**
- **Access** public

string function mamurConfig::getPluginsToLoad() [line 309]

Gets a list of plugins to load and associated data

- **Static**

- **Access** public

void function mamurConfig::persistSetting([\$name = "], [\$value = "]) [line 421]

Function Parameters:

- *\$name* **\$name** - name of setting
- *\$value* **\$value** - value to persist

Persists to the XML config data a setting value

The data is not saved unless upDateConfig is called Name must be defined and if apild or salt is set then for security the XML file must have a value set to new ie must not have been set before

- **Access** public

void function mamurConfig::processConfig(\$mamurPageConfig) [line 72]

Function Parameters:

- *\$mamurPageConfig* **\$mamurPageConfig** - settings made during start up and added to the settings data class

processConfig saves configuration setup in index.php and bootstrap

then reads configuration.xml file for the main setup variables It is normally called only once by the controller bootstrap mamur.php Each data group is handled by a mamurConfigData class which makes it easy to use settings when required since the class instance can be obtained by a call to this class eg \$set =mamurConfig::getInstance->settings ; //gets settings data print \$set->server;

- **Access** public

void function mamurConfig::relativeDir(\$dir, \$subdir) [line 356]

Function Parameters:

- **\$dir**
- **\$subdir**

- **Static**
- **Access** public

void function mamurConfig::runPlugIns(&\$model, &\$view) [line 343]

Function Parameters:

- **&\$model**
- **&\$view**

- **Static**
- **Access** public

void function mamurConfig::setGlobal(\$name, \$value) [line 462]

Function Parameters:

- **\$name**
- **\$value**

- **Static**
- **Access** public

void function mamurConfig::setPlugIn([\$name = "], \$status, \$file, [\$version = "]) [line 381]

Function Parameters:

- *\$name* **\$name** - plugin name
- *\$status* **\$status** - status eg active
- *\$file* **\$file** - file location
- *\$version* **\$version** - plugin version

Defines a plugin and places the plugin details in configuration.xml

- **Access** public

void function `mamurConfig::upDateConfig()` [*line 454*]

This is normally called by controller at end of page

it automatically saves and configurations which have been set using `persitSettings()` method

- **Access** public

void function `mamurConfig::__get($variable)` [*line 247*]

Function Parameters:

- *\$variable* **\$variable** an item (value or array) in the data array

__get magic method allows

`$config=mamurConfig::getInstance(); $config->variable` and `$config->$mayvariable` constructs

- **Access** public

Class mamurConfigData

[line 24]

mamurConfigData encapsulates a simple data array used to hold configuration data so that name=variable configurations can be accessed by \$dataClass->\$myVariable constructs.

However, some configurations have multiple attributes per variable in which case the data encapsulated is an array of mamurConfigData objects so that \$dataClass->\$myVariable->\$myAttribute can be used.

- **Package** mamur
- **Sub-Package** coreModel
- **Author** Martin Marsh < martinmarsh@sygenius.com>
- **Version** 105
- **Copyright** Copyright (c) 2011, Sygenius Ltd
- **Release** Mamur 1.10
- **Mvc** model
- **Name** mamurConfigData
- **License** [GNU Public License, version 3](#)

Constructor *void* function mamurConfigData::__construct(0) [line 34]

Function Parameters:

- **\$settings 0** - array of values or mamurConfigData objects

Constructor initialises data in a simple associated data array of values or for multi-attribute configuration an array of mamurConfigData objects.

- **Access** public

void function mamurConfigData::getAll() [line 42]
getAll method returns entire data array

- **Access** public

void function mamurConfigData::__get(\$variable) [line 52]
Function Parameters:

- *\$variable* **\$variable** an item (value or array) in the data array

__get magic method allows
\$dataClass->variable and \$dataClass->\$mayvariable constructs

- **Access** public

void function mamurConfigData::__isset(\$variable) [line 68]
Function Parameters:

- *\$variable* **\$variable**

__isset magic method allows isset(\$dataClass->variable) construct

- **Access** public

void function mamurConfigData::__set(\$variable, \$value) [line 82]
Function Parameters:

- *\$variable* **\$variable**
- **\$value**

__set magic method allows \$dataClass->variable=value construct

if multi-level data use `$dataClass->variable=new mamurConfigData($array)` so that `$dataClass->variable->variable` works

- **Access** public

void function mamurConfigData::__unset(\$variable) [line 92]

Function Parameters:

- *unknown_type* **\$variable**

allows unset(\$dataclass->variable) construct

- **Access** public

Class mamurDataObject

[line 31]

mamurDataObject encapsulates data in an array structure.

The concept is that instead of using session arrays similar data items can be grouped into meaningful data objects which can be persisted to session data or to an ORM or database. `mamurDataObject` can be a single data item or a list of items accessed using `$mydataObjectClass->$myVariableName` object magic methods. Also dataobjects can support mutiple rows to form a table There are methods to switch rows and to index through

row items. Note unlike using an ORM everything is stored in memory so it is designed for use of small datasets or subsets of data from a query.

- **Package** mamur
- **Sub-Package** coreModel
- **Author** Martin Marsh < martinmarsh@sygenius.com>
- **Version** 105
- **Copyright** Copyright (c) 2011, Sygenius Ltd
- **Release** Mamur 1.10
- **Mvc** model
- **Name** mamurDataObject
- **License** [GNU Public License, version 3](#)

Constructor *void* function mamurDataObject::__construct() [*line 40*]

Constructor initialises data in a simple associated data array

```
$dataSet['table'][$tableName][$record][$fieldName]=$value;
```

- **Access** public

void function mamurDataObject::appendRecord([\$recordData = array()]) [*line 104*]

Function Parameters:

- *array* **\$recordData** an associated array

Appends a new record defined by an associated array

or a blank record if not given and sets the record pointer to point to the new record allowing access to variables

- **Access** public

void function mamurDataObject::back() [*line 156*]

- **Access** public

integer function mamurDataObject::deleteRecord([\$record = -1]) [*line 120*]

Function Parameters:

- *integer* **\$record** record number

Deletes specified record or if not given the current record

If the record deleted is the current record the current record will be set to the next record or if not possible the last record

- **Access** public

array function mamurDataObject::getAll() [*line 55*]

getAll method returns entire data table array

- **Access** public

void function mamurDataObject::getAttribute(\$name) [*line 183*]

Function Parameters:

- **\$name**

- **Access** public

array function mamurDataObject::getCurrentRecordNumber() [*line 78*]
returns current record number

- **Access** public

array function mamurDataObject::getRecord() [*line 64*]
getRecord method returns associated array of a record

- **Access** public

void function mamurDataObject::getStatus(\$name) [*line 196*]
Function Parameters:

- **\$name**

- **Access** public

void function mamurDataObject::last() [*line 166*]

- **Access** public

void function mamurDataObject::modified() [*line 214*]

- **Access** public

integer function mamurDataObject::next() [*line 146*]

If possible Moves record pointer forward by one

- **Access** public

void function mamurDataObject::persist() [*line 204*]

- **Access** public

void function mamurDataObject::read() [*line 209*]

- **Access** public

void function mamurDataObject::setAttribute(\$name, \$value) [*line 177*]

Function Parameters:

- **\$name**
- **\$value**

- **Access** public

record function mamurDataObject::setCurrentRecordNumber(\$record) [*line 88*]

Function Parameters:

- *integer* **\$record** valid record number

Sets current record returning the new record number set

- **Access** public

void function mamurDataObject::setStatus(\$name, \$value) [*line 191*]

Function Parameters:

- **\$name**
- **\$value**

- **Access** public

void function mamurDataObject::__get(\$variable) [*line 226*]

Function Parameters:

- *\$variable* **\$variable** an item

__get magic method allows
\$dataClass->variable constructs

- **Access** public

void function mamurDataObject::__isset(\$variable) [*line 243*]

Function Parameters:

- *\$variable* **\$variable**

__isset magic method allows isset(\$dataClass->variable) construct

- **Access** public

void function mamurDataObject::__set(\$variable, \$value) [*line 255*]

Function Parameters:

- *\$variable* **\$variable**
- **\$value**

__set magic method allows \$dataClass->variable=value construct

- **Access** public

void function mamurDataObject::__unset(\$variable) [*line 265*]

Function Parameters:

- *unknown_type* **\$variable**

allows unset(\$dataclass->variable) construct

- **Access** public

Class mamurModel

[line 64]

mamurModel is the main model class to access data and files

and provides methods to change the model state. In our MVC model we place all business logic, object model entities and interfaces in the model and have avoided creating a library of general methods and functions which can be used directly by views and controllers.

Everything which is not a controller or view related is considered to be part of the model. The model has 2 main classes this one and mamurConfig which contains only configuration and dynamic settings. In the model there are two data object classes:

1) mamurConfigData - a class used by mmaurConfig to create configuration entities for each type of configuration information eg settings, globals, classes, placeholders etc.

2) mamurDataObject - a container class which allows data and class instances to be encapsulated in a primitive object which can be persisted between page requests. Similar items can be saved in a named mamurDataObject so that it is easier to persist and delete them as a group. Alternatively, mamurDataObject can contain one or more data records from a database query. Status and Attributes can be set for mamurDataObjects to help process them say through a form.

The main model handles the creation, storage, retrieval, garbage collection of mamurDataObjects and acts as an abstraction layer from the session storage mechanism. mamurDataObjects are created or retrieved from session store on demand by using the main model's getDataObject('objectname') method.

- **Package** mamur
- **Sub-Package** coreModel

mamurModel::\$countSerial

mixed = [line 72]

- **Access** protected

mamurModel::\$dataObjects

mixed = [line 79]

- **Access** protected

mamurModel::\$defaultPageExt

mixed = [line 67]

- **Access** protected

mamurModel::\$defaultPageName

mixed = [line 67]

- **Access** protected

mamurModel::\$error404PageName

mixed = [line 70]

- **Access** protected

mamurModel::\$global

mixed = [line 75]

- **Access** protected

mamurModel::\$hostdomain

mixed = [line 73]

- **Access** protected

mamurModel::\$hostScheme

mixed = [line 73]

- **Access** protected

mamurModel::\$inSession

mixed = [line 79]

- **Access** protected

mamurModel::\$lastErrorPage

mixed = [line 71]

- **Access** protected

mamurModel::\$locid

mixed = [line 74]

- **Access** protected

mamurModel::\$locidAccepted

mixed = [line 74]

- **Access** protected

mamurModel::\$logOutFlag

mixed = [line 79]

- **Access** protected

mamurModel::\$mamurBaseDir

mixed = [line 76]

- **Access** protected

mamurModel::\$mamurlogDir

mixed = [line 76]

- **Access** protected

mamurModel::\$mamurPluginDir

mixed = [line 76]

- **Access** protected

mamurModel::\$mamurSystemDir

mixed = [line 76]

- **Access** protected

mamurModel::\$mamurURL

mixed = [line 65]

- **Access** protected

mamurModel::\$oldSession

mixed = [line 79]

- **Access** protected

mamurModel::\$options

mixed = [line 79]

- **Access** protected

mamurModel::\$pageDir

mixed = [line 67]

- **Access** protected

mamurModel::\$pageDirList

mixed = [line 67]

- **Access** protected

mamurModel::\$pageExt

mixed = [line 67]

- **Access** protected

mamurModel::\$pageName

mixed = [line 67]

- **Access** protected

mamurModel::\$pagePagePrintCallBack

mixed = [line 77]

- **Access** protected

mamurModel::\$pageProcessCallBack

mixed = [line 77]

- **Access** protected

mamurModel::\$pageQuery

mixed = [line 67]

- **Access** protected

mamurModel::\$pageURL

mixed = [line 65]

- **Access** protected

mamurModel::\$phpParameters

mixed = [line 66]

- **Access** protected

mamurModel::\$serverBaseRequestCallBack

mixed = [line 78]

- **Access** protected

mamurModel::\$session

mixed = [line 79]

- **Access** protected

mamurModel::\$sessionClearCallBack

mixed = [line 77]

- **Access** protected

mamurModel::\$subdomain

mixed = [line 73]

- **Access** protected

mamurModel::\$tags

mixed = [line 79]

- **Access** protected

mamurModel::\$templateTags

mixed = [line 69]

- **Access** protected

mamurModel::\$themesDir

mixed = [line 68]

- **Access** protected

mamurModel::\$topdomain

mixed = [line 73]

- **Access** protected

mamurModel::\$urlCallback

mixed = [line 77]

- **Access** protected

mamurModel::\$webBaseDir

mixed = [line 65]

- **Access** protected

mamurModel::\$xmlPageType

mixed = [line 66]

- **Access** protected

Constructor *void* function **mamurModel::__construct()** *[line 93]*

Constructor sets up properties according to configuration

On first install Sets apiID and Salt Manages cookies (if enabled), decrypt stored session cookies, logouts and set up user session data

- **Access** public

void function mamurModel::checkLogOut() [line 243]

Checks to see if logout required.

A hook function can modify this method and extend login period

- **Access** public

void function mamurModel::convertToUserZone(\$date, [\$format = DATE_ATOM]) [line 1300]

Function Parameters:

- **\$date**
- **\$format**

- **Access** public

Decrypted function mamurModel::decrypt(\$value, [\$a = 44], [\$b = 16]) [line 644]

Function Parameters:

- *string* **\$value** - base64 encoded 256bit encrypted data
- *integer* **\$a**
- *integer* **\$b**

Decypt a String using set cipher and optionally

2 security integers to identify the salt bases to use

- **Access** public

void function mamurModel::deleteDataObject(\$name) [*line 1629*]

Function Parameters:

- **\$name**

- **Access** public

void function mamurModel::dirListToDataObject(\$listArray, \$dataObjectName, \$fieldName) [*line 342*]

Function Parameters:

- *unknown_type* **\$listArray**
- *unknown_type* **\$dataObjectName**
- *unknown_type* **\$fieldName** to place the list in

Places a directory list into a data object at the current record
and in a named field or dataObject item each field is labelled

- **Access** public

void function mamurModel::doPagePagePrintCallBacks() [*line 1348*]

- **Access** public

encrypted function mamurModel::encrypt(\$dataArray, [\$a = 44], [\$b = 8], \$value) [*line 705*]

Function Parameters:

- *string* **\$value**
- *integer* **\$a** (must be less than 70)
- *integer* **\$b** (must be less than 70)
- **\$dataArray**

encrypt a String using set cipher and optionally

2 security integers to identify the salt bases to use

- **Access** public

16byte function mamurModel::getApiSalt() [*line 630*]

Gets an Api salt in a similar way to getSalt

- **Access** public

void function mamurModel::getContentBase([\$subdir = ""]) [*line 1095*]

Function Parameters:

- **\$subdir**

- **Access** public

void function mamurModel::getContentList(\$contType, \$match) [line 1231]

Function Parameters:

- *unknown_type* **\$contType**
- *unknown_type* **\$match**

Gets a list of content names give a search pattern for a name

use '*' to match a sub string in the name part eg * returns all files and n*x returns files starting with n and ending in x file extension .html is optional although generally one should not be given

- **Access** public

void function mamurModel::getCookieSerialNo() [line 837]

Generates a cookie unique serial number

- **Access** public

void function mamurModel::getDataBaseDir([\$subdir = ""]) [line 1108]

Function Parameters:

- **\$subdir**

- **Access** public

void function mamurModel::getDataObject(\$name) [line 1542]

Function Parameters:

- **\$name**

- **Access** public

void function mamurModel::getEditStatus([\$status = false]) [*line 548*]

Function Parameters:

- **\$status \$status**

get page edit status

- **Access** public

encrypted function mamurModel::getEncryptedSession() [*line 947*]

Gets session data in an encrypted string identical to that

persisted (via session cookie). Use this call in unit tests to set \$cookie['session'] to allow persistence between models

- **Access** public

void function mamurModel::getError404PageName() [*line 1086*]

- **Access** public

void function mamurModel::getErrorPage() [line 1363]

- **Access** public

void function mamurModel::getGlobal(\$name) [line 1314]

Function Parameters:

- **\$name**

- **Access** public

void function mamurModel::getHomeUri() [line 1492]

- **Access** public

void function mamurModel::getHostDomain() [line 1484]

- **Access** public

void function mamurModel::getHostScheme() [line 1480]

- **Access** public

void function mamurModel::getMamurBaseDir([\$subdir = ""]) [line 1119]

Function Parameters:

- **\$subdir**

- **Access** public

void function mamurModel::getMamurlogDir([\$subdir = "]) [line 1123]

Function Parameters:

- **\$subdir**

- **Access** public

void function mamurModel::getMamurSystemDir([\$subdir = "]) [line 1127]

Function Parameters:

- **\$subdir**

- **Access** public

void function mamurModel::getMamurUrl() [line 1062]

- **Access** public

void function mamurModel::getMamurUserDir([\$subdir = "]) [line 1113]

Function Parameters:

- **\$subdir**

- **Access** public

nonce function mamurModel::getNonce([\$name = 'mamurData']) [*line 1528*]

Function Parameters:

- *string* **\$name** - name of nonce

A named Nonce value will be returned if it has been set.

For security nonces are saved to server side session dataObject as well as in the encrypted session cookie. If security fails the nonce will be set to false which may indicate a stolen session cookie and a security action such as logging is required. Use a nonce when it makes sense to protect access or use of a form which is related to an action in a browser tab window. The nonce value must still be compared to the value used in the form but a mismatch indicates that the form may have been previously submitted

- **Access** public

void function mamurModel::getNonMamurPage() [*line 1070*]

- **Access** public

option function mamurModel::getOption([\$name = 0]) [*line 291*]

Function Parameters:

- *\$name* **\$name**

Gets a named option

- **Access** public

string function mamurModel::getPageContent(\$name) [*line 1140*]

Function Parameters:

- *string* **\$name** - of content eg main

Gets Page Specific Content by name

- **Access** public

void function mamurModel::getPageDir() [*line 1082*]

- **Access** public

void function mamurModel::getPageDirList() [*line 1467*]

- **Access** public

void function mamurModel::getPageExt() [*line 1079*]

- **Access** public

void function mamurModel::getPageName() [*line 1075*]

- **Access** public

phpParamters function mamurModel::getParameters() [*line 271*]

Gets php pass parameters defined by a php tag

- **Access** public

void function mamurModel::getPhpBase([\$subdir = ""]) [*line 1099*]

Function Parameters:

- **\$subdir**

- **Access** public

void function mamurModel::getPluginDir([\$subdir = ""]) [*line 1131*]

Function Parameters:

- **\$subdir**

- **Access** public

void function mamurModel::getRandomString(\$length, [\$supperonly = false]) [*line 808*]

Function Parameters:

- *integer* **\$length**
- *bool* **\$supperonly**

Gets a random string of Letters and numbers of
specified length with option to return uppercase only

- **Access** public

32byte function mamurModel::getSalt([\$a = 84], [\$b = 11], [\$len = 32]) [line 619]

Function Parameters:

- *integer* **\$a**
 - value from 0 to 99 (optional defaults to 84)
- *integer* **\$b**
 - value from 0 to 99 (optional defaults to 11)
- *integer* **\$len** - length of salt (optional defaults to 32)

Gets a salt by concatenating two equal length strings from

the configuration salt string. The default is 32 bytes but any even number may be used. Each installation has a unique salt and unless re-installed the salt will always remain the same. Allowing it to be used for encryption of long lived resources. To allow different salts to be defined there is an option to give two integers to identify the salt required. This method also makes it harder to see the salts by looking at the configuration file

- **Access** protected

void function mamurModel::getServerBase() [line 1040]

the server base is usually the user directory but a call back function allows it to be switched programatically to another area. This would allow another file area to be used for the page and content area.

Note the database and logs are not affected and are not remapped

- **Access** public

string function mamurModel::getSharedContent(\$name, \$type, [\$group = ""], [\$mapped = ""]) [*line 1175*]

Function Parameters:

- *string* **\$name** - name of content eg main
- **\$type**
- **\$group**
- **\$mapped**

Gets Shared Content by name, type and by optional group and mapping

group use directory notation to add additional classification to a type of content eg articles could be divided by year and month eg the group would be year/month mapping allows the name to be expanded according to a uri or post value. the name is appended by an additional map value separated by an underscore mapping can be byFirstSection, byLastSection, byPage, byGet or byPost section refers to the portion between the / in the uri eg /firstsection/middlesection/lastsection/anypagename if pagename is omitted index is assumed. The first and lastsections can be the same or null if empty in which case just the name of the content without @author sygenius underscore separator is used. Post or Get variables use the same name as the content type eg ?article=1

- **Access** public

void function mamurModel::getSharedContentBase([\$subdir = ""]) [*line 1104*]

Function Parameters:

- **\$subdir**

- **Access** public

void function mamurModel::getSubDomain() [*line 1476*]

- **Access** public

unknown_type function mamurModel::getTag([\$name = 'tag'], [\$index = 0]) [*line 326*]

Function Parameters:

- *\$name* **\$name**
- *\$index* **\$index**

Gets a tag value

- **Access** public

void function mamurModel::getTagData() [*line 1310*]

- **Access** public

void function mamurModel::getTemplateFile([\$subdir = ""]) [*line 1090*]

Function Parameters:

- **\$subdir**

- **Access** public

void function `mamurModel::getTopDomain()` [*line 1488*]

- **Access** public

void function `mamurModel::getTopPageDir()` [*line 1459*]

- **Access** public

void function `mamurModel::getUrl()` [*line 1058*]

- **Access** public

unknown_type function `mamurModel::getUser()` [*line 478*]

Gets current user session data

- **Access** public

void function `mamurModel::GetUserIP()` [*line 854*]

Gets best value for user IP address

returns "127.0.0.1" if no server variables are set which is probably the case
ie if not running under a server (during testing)

- **Access** public

void function `mamurModel::getUserZoneOffset($date)` [*line 1285*]

Function Parameters:

- **\$date**

- **Access** public

void function mamurModel::getWebRoot() [*line 1065*]

- **Access** public

void function mamurModel::getXmlPageType() [*line 1343*]

- **Access** public

void function mamurModel::isError404Page() [*line 1050*]

- **Access** public

unknown_type function mamurModel::isLoggedIn([\$id = "], [\$name = "], [\$status = "], [\$group = "], [\$statusName = "]) [*line 511*]

Function Parameters:

- **\$id \$id**
- **\$name \$name**
- **\$status \$status**
- **\$group \$group**
- **\$statusName \$statusName**

Checks if user logged in

- **Access** public

void function mamurModel::locidCookie() [line 156]

Reads and saves status of user location cookie a value

value set for each permanent cookie which indicates a different user account or a physcal different machine If a user is logged in but uses different locid this indicates the probable use at different locations eg home/office but could be in the same building or a mobile device

- **Access** public

void function mamurModel::openPageMeta(\$pageRelUrl) [line 1372]

Function Parameters:

- **\$pageRelUrl**

- **Access** public

void function mamurModel::pageProcessHookContinue() [line 1335]

- **Access** public

page function mamurModel::pageTime([\$check = false]) [line 385]

Function Parameters:

- **\$check \$check**

Gets page processing lapsed time

- **Access** public

void function mamurModel::passParameters(\$var) [*line 263*]

Function Parameters:

- *\$var* **\$var**

Saves php parameters defined by a php tag

- **Access** public

string function mamurModel::pbkdf2(\$p, \$s, \$c, \$kl, [\$a = 'sha256']) [*line 773*]

Function Parameters:

- *string* **\$p** p password
- *string* **\$s** s salt
- *int* **\$c** c iteration count (use 1000 or higher)
- *int* **\$kl** kl derived key length
- *string* **\$a** a hash algorithm

PBKDF2 Implementation (as described in RFC 2898) This is recommended for password hashing but takes some computing power due to min 1000 loops, so only use on infrequent activities such as for passwords.

Overhead is probably less than 10ms Reference: <http://www.itnewb.com/v/Encrypting-Passwords-with-PHP-for-Storage-Using-the-RSA-PBKDF2-Standard>
<http://en.wikipedia.org/wiki/PBKDF2>

- **Access** public

void function mamurModel::processPageMetaData() [line 956]

Process the current Page meta data including the template to use

- **Access** public

void function mamurModel::processXML(\$xmlPagefile, &\$doc, &\$pageTags) [line 1384]

Function Parameters:

- **\$xmlPagefile**
- **&\$doc**
- **&\$pageTags**

- **Access** public

void function mamurModel::readDataObjects() [line 1553]

- **Access** public

void function mamurModel::readPageXML() [line 966]

Reads page XML file which defines template and page meta data

- **Access** public

void function mamurModel::registerPagePrintFunction(&\$ref, \$function) [*line 1447*]

Function Parameters:

- **&\$ref**
- **\$function**

- **Access public**

void function mamurModel::registerPageProcessFunction(&\$ref, \$function) [*line 1435*]

Function Parameters:

- **&\$ref**
- **\$function**

- **Access public**

void function mamurModel::registerServerBaseFunction(&\$ref, \$function) [*line 1453*]

Function Parameters:

- **&\$ref**
- **\$function**

- **Access public**

void function mamurModel::registerSessionClearFunction(&\$ref, \$function) [*line 1423*]

Function Parameters:

- **&\$ref**
- **\$function**

- **Access** public

void function mamurModel::registerSessionLogoutFunction(&\$ref, \$function) [line 1441]

Function Parameters:

- **&\$ref**
- **\$function**

- **Access** public

void function mamurModel::registerUrlFunction(&\$ref, \$function) [line 1429]

Function Parameters:

- **&\$ref**
- **\$function**

- **Access** public

void function mamurModel::relativeDir(\$dir, \$subdir) [line 1257]

Function Parameters:

- *string* **\$dir** directory reference
- *string* **\$subdir** additional directory or page name to append to directory reference

relativeDir extends a directory reference by adding and additional string the additional string can be a subdirectory or page name.

The directory may end with an optional / character. The additional string may start with an optional / character. relative directory ensures that a / is added between and that // cannot occur

- **Access** public

unknown_type function `mamurModel::removePage($page)` [*line 375*]

Function Parameters:

- *\$page* **\$page**

Removes a page

- **Access** public

unknown_type function `mamurModel::rmdir($dir)` [*line 357*]

Function Parameters:

- *\$dir* **\$dir**

recursive remove directory - protected as it is dangerous.

- **Access** protected

void function `mamurModel::saveDataObjects()` [*line 1563*]

- **Access** public

void function mamurModel::setEditStatus([\$status = false]) [*line 557*]

Function Parameters:

- *\$status* **\$status**

Sets page edit status

- **Access** public

void function mamurModel::setErrorPageTemplate() [*line 1357*]

- **Access** public

void function mamurModel::setGlobal(\$name, \$value) [*line 1330*]

Function Parameters:

- *unknown_type* **\$name**
- *unknown_type* **\$value**

Sets a global for use on current page the value is not persisted.

- **Access** public

void function mamurModel::setHostData(\$host) [*line 571*]

Function Parameters:

- *\$host* **\$host** - domain name

Looks at the a domain name (normally the host for current server)
and sets model attributes realating to various parts of the domain name

- **Access** public

void function mamurModel::setLocidCookie() [line 900]

Sets a cookie to help identify a users at a location

Cookies will be set according to machine and account This is approxiamte as accounts may be shared and users may use more than 1 machine at a location. Each mobile device is assumed to be a different "location"

- **Access** public

void function mamurModel::setNonce([\$name = 'mamurData'], [\$length = 16]) [line 1504]

Function Parameters:

- *string* **\$name** - name of nonce
- *integer* **\$length** - number of characters to use 16 if not given

A nonce is a number used once and can be used to prevent forms
from being resent. They also provide a security check see `getNonce`

- **Access** public

void function mamurModel::setOption(\$value, [\$name = 0]) [*line 282*]

Function Parameters:

- *\$value* **\$value** - of option
- *\$name* **\$name** - of option

Saves an option

- **Access** public

unknown_type function mamurModel::setPageUri(\$uri) [*line 406*]

Function Parameters:

- *\$uri* **\$uri**

Sets page uri and associated internal variables using the value of the url parameter passed.

the URI would normally be from the web server ie the current page.

- **Access** public

void function mamurModel::setSessionCookie() [*line 922*]

Sets an encrypted session based cookie which contains the

session data stored in \$this->session array Any pointers to session files are therefore encrypted The session cookie can be changed without affecting the data

- **Access** public

void function mamurModel::setTag(\$value, [\$name = 'tag'], [\$index = 0]) [*line 304*]

Function Parameters:

- *\$value* **\$value**
- *\$name* **\$name**
- *\$index* **\$index**

Sets a Tag

- **Access** public

unknown_type function mamurModel::setTagList(\$listName, [\$name = 'tag']) [*line 314*]

Function Parameters:

- *\$listName* **\$listName**
- *\$name* **\$name**

Sets an array (list) of tags

- **Access** public

void function mamurModel::setUpSession() [*line 172*]

Sets up session and processes cookies as per configuration

- **Access** public

unknown_type function mamurModel::setUser([\$name = 'unknown'], [\$id = ''], [\$loggedin = false], [\$status = 0], [\$group = 'unknown'], [\$statusName = 'unknown']) [*line 464*]

Function Parameters:

- **\$name \$name**
- **\$id \$id**
- **\$loggedin \$loggedin**
- **\$status \$status**
- **\$group \$group**
- **\$statusName \$statusName**

Sets Current User deatils

- **Access public**

void function mamurModel::timeStampToUserData(\$stamp, [\$format = DATE_ATOM]) [*line 1273*]

Function Parameters:

- **\$stamp**
- **\$format**

- **Access public**

void function mamurModel::unique_serial() [*line 876*]

generates a unique serial number based on date,microtime, call count and a random number so there is very low risk of duplication in moderately busy systems.

The number may be used externally and vowels are mamped out to prevent spelling of words which would look unprofessional

- **Access** public

unknown_type function mamurModel::userLogin([\$status = 0], [\$statusName = 'unknown']) [*line 488*]

Function Parameters:

- *\$status* **\$status**
- *\$statusName* **\$statusName**

Logs in current user

- **Access** public

unknown_type function mamurModel::userLogout([\$status = ""]) [*line 536*]

Function Parameters:

- *\$status* **\$status**

Logs out a user

- **Access** public

mamurView

This file contains the static main view Class - mamurView

Licence: This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, version 3 of the License. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

- **Package** mamur
- **Sub-Package** coreView
- **Author** Martin Marsh < martinmarsh@sygenius.com>
- **Version** 110
- **Copyright** Copyright (c) 2011, Sygenius Ltd
- **Release** Mamur 1.10
- **Mvc** view
- **Name** mamurView
- **License** [GNU Public License, version 3](#)

unknown_type function mamurViewReplacePlaceholder(\$matches) [line 505]

Function Parameters:

- *\$matches* **\$matches**

Warning this is a call back function and NOT a class method.

This is not neat but is a v5.2+ work around for not having Anonymous functions which would be used in the above class

Class mamurForm

[line 30]

This file contains the core view Class - mamurForm

Licence: This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, version 3 of the License. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

- **Package** mamur
- **Sub-Package** coreView
- **Author** Martin Marsh < martinmarsh@sygenius.com >
- **Version** 110
- **Copyright** Copyright (c) 2011, Sygenius Ltd
- **Release** Mamur 1.10
- **Mvc** view
- **Name** mamurForm
- **License** [GNU Public License, version 3](#)

Constructor *void* function mamurForm::__construct(\$setModel, \$setView) [*line 36*]

Function Parameters:

- **\$setModel**
- **\$setView**

void function mamurForm::doEndform() [*line 100*]

- **Access** public

void function mamurForm::doForm(\$parms) [*line 70*]

Function Parameters:

- **\$parms**

- **Access** public

void function `mamurForm::doInput()` [*line 76*]

- **Access** public

void function `mamurForm::doOption()` [*line 96*]

- **Access** public

void function `mamurForm::doSelect()` [*line 84*]

- **Access** public

void function `mamurForm::doTextarea()` [*line 80*]

- **Access** public

void function `mamurForm::endSelect()` [*line 88*]

- **Access** public

void function `mamurForm::fldAttrib($attr, $ITag)` [*line 508*]

Function Parameters:

- **\$attr**

- **\$ITag**

- **Access** protected

void function `mamurForm::generalPlaceholder($attr, $tag)` *[line 112]*

Function Parameters:

- **\$attr**
- **\$tag**

- **Access** public

void function `mamurForm::optionList()` *[line 92]*

- **Access** public

void function `mamurForm::validate($invalue, $name)` *[line 633]*

Function Parameters:

- **\$invalue**
- **\$name**

Class `mamurPlaceholders`

[line 29]

This file contains the core view Class - `mamurPlaceholders`

Licence: This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, version 3 of the License. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

- **Package** mamur
- **Sub-Package** coreView
- **Author** Martin Marsh < martinmarsh@sygenius.com >
- **Version** 110
- **Copyright** Copyright (c) 2011,Sygenius Ltd
- **Release** Mamur 1.10
- **Mvc** view
- **Name** mamurPlaceholders
- **License** [GNU Public License, version 3](#)

Constructor *void* function mamurPlaceholders::__construct(\$model, \$view) [*line 34*]

Function Parameters:

- **\$model**
- **\$view**

- **Access** public

void function mamurPlaceholders::all(\$param) [*line 89*]

Function Parameters:

- **\$param**

- **Access public**

void function mamurPlaceholders::build_random(\$param) [*line 165*]

Function Parameters:

- **\$param**

- **Access public**

void function mamurPlaceholders::date(\$param) [*line 187*]

Function Parameters:

- **\$param**

- **Access public**

void function mamurPlaceholders::endform(\$param) [*line 313*]

Function Parameters:

- **\$param**

- **Access public**

void function mamurPlaceholders::endselect(\$param) [*line 298*]

Function Parameters:

- **\$param**

- **Access** public

void function mamurPlaceholders::form(\$param) [line 277]

Function Parameters:

- **\$param**

- **Access** public

void function mamurPlaceholders::generalPlaceholder(\$param, \$tag) [line 321]

Function Parameters:

- **\$param**
- **\$tag**

void function mamurPlaceholders::globalTag(\$param) [line 145]

Function Parameters:

- **\$param**

- **Access** public

void function mamurPlaceholders::http_header(\$param, \$tag) [line 135]

Function Parameters:

- **\$param**
- **\$tag**

- **Access public**

void function mamurPlaceholders::input(\$param) [line 283]

Function Parameters:

- **\$param**

- **Access public**

void function mamurPlaceholders::mamur() [line 123]

- **Access public**

void function mamurPlaceholders::nonce(\$param) [line 203]

Function Parameters:

- **\$param**

- **Access public**

void function mamurPlaceholders::option(\$param) [line 308]

Function Parameters:

- **\$param**

- **Access** public

void function mamurPlaceholders::optionlist(\$param) [*line 303*]

Function Parameters:

- **\$param**

- **Access** public

string function mamurPlaceholders::page_content(\$param, \$tag) [*line 48*]

Function Parameters:

- *array* **\$param**
 - array of tag parameters
- *string* **\$tag**
 - tag name

page_content tag inserts content associated with a page and stored in the page specific

folder or table This content can contain tags so view->processPlaceholders is called on the content

- **Access** public

void function mamurPlaceholders::page_selected(\$param) [line 218]

Function Parameters:

- **\$param**

- **Access** public

void function mamurPlaceholders::page_timer() [line 179]

- **Access** public

void function mamurPlaceholders::page_timerms() [line 183]

- **Access** public

void function mamurPlaceholders::php(\$param) [line 193]

Function Parameters:

- **\$param**

- **Access** public

void function mamurPlaceholders::random(\$param) [line 155]

Function Parameters:

- **\$param**

- **Access** public

void function mamurPlaceholders::select(\$param) [*line 293*]

Function Parameters:

- **\$param**

- **Access** public

string function mamurPlaceholders::shared(\$param, \$tag) [*line 66*]

Function Parameters:

- *array* **\$param**
 - array of tag parameters
- *string* **\$tag**
 - tag name

shared tags inserts shared content of a particular type

as defined by the tag name ie 'template', 'structure', 'section' 'blog' 'menu' 'news' 'article'
This content can contain tags so view->processPlaceholders is called on the content

- **Access** public

void function mamurPlaceholders::tag(\$param) [*line 209*]

Function Parameters:

- **\$param**

- **Access** public

void function mamurPlaceholders::textarea(\$param) [*line 288*]

Function Parameters:

- **\$param**

- **Access** public

void function mamurPlaceholders::title() [*line 127*]

- **Access** public

void function mamurPlaceholders::unique_serial() [*line 160*]

- **Access** public

Class mamurView [*line 48*]

mamurView is a basic class to print out pages.

Pages are prepared in two stages.

First content is aggregated into a single page replacing placeholder tags with either static content or php function calls to the required dynamic view method. If set in the configuration the results of the first stage can be saved to a page build area so that it is not necessary to repeat the first stage on every page request. This is a sort of cache in that pages have to be rebuilt if their static content or templates are edited. The second stage renders the page using PHP to replace the php tags with dynamic output. All output is buffered so page headers and redirection can be requested at any stage. The controller instructs the page buffer built by view to be flushed which sends the response to the page request.

- **Package** mamur
- **Sub-Package** coreView

mamurView::\$mamur

mixed = [line 54]

- **Access** protected

mamurView::\$model

mixed = [line 55]

- **Access** protected

mamurView::\$oddeven

mixed = [line 60]

- **Access** protected

mamurView::\$pageBuild

mixed = [line 62]

- **Access** protected

mamurView::\$pageBuildId

mixed = [line 64]

- **Access** protected

mamurView::\$pageOutput

mixed = [line 58]

- **Access** protected

mamurView::\$pagePhp

mixed = [line 63]

- **Access** protected

mamurView::\$placeholderClasses

mixed = [line 61]

- **Access** protected

mamurView::\$templateTags

mixed = [line 56]

- **Access** public

mamurView::\$urlDir

mixed = [line 57]

- **Access** protected

Constructor *void* function **mamurView::__construct()** *[line 69]*

- **Access** public

void function **mamurView::defaultErrorPage()** *[line 396]*

- **Access** public

void function **mamurView::directPhpView()** *[line 218]*

- **Access** public

void function **mamurView::doDatePlaceholder(\$serparams)** *[line 360]*

Function Parameters:

- **\$serparams**

- **Access** public

void function mamurView::doGlobalPlaceholder(\$serparams) [*line 322*]

Function Parameters:

- **\$serparams**

- **Access** protected

void function mamurView::doNoncePlaceholder(\$serparams) [*line 347*]

Function Parameters:

- **\$serparams**

- **Access** public

void function mamurView::dopage_pageTimerPlaceholder() [*line 378*]

- **Access** public

void function mamurView::dopage_timermsPlaceholder() [*line 374*]

- **Access** public

void function mamurView::doPhpAndPrint(\$input) [*line 463*]

Function Parameters:

- **\$input**

- **Access** public

void function mamurView::doPhpPlaceholder(\$serparams) [*line 280*]

Function Parameters:

- **\$serparams**

- **Access** public

void function mamurView::doRandomPlaceholder(\$serparams) [*line 329*]

Function Parameters:

- **\$serparams**

- **Access** public

void function mamurView::doTagPlaceholder(\$serparams) [*line 307*]

Function Parameters:

- *unknown_type* \$serparams

This method dynamically replaces a 'tag' type

palceholder with content which hasd previously saved in the tag. The content can have other tags inside which even will be inserted dynamically. This could give rise to inconsistency if same content is inserted both dynamically and statically (eg a shared content tag is referred to in body and in a 'tag' content and the configuration pageBuild is set to yes.

- **Access** protected

void function mamurView::doUniqueSerialPlaceholder() [*line 343*]

- **Access** public

void function mamurView::getPlaceholder(\$class) [*line 89*]

Function Parameters:

- **\$class**

- **Access** public

void function mamurView::includePageFile() [*line 384*]

- **Access** public

void function mamurView::insertPlaceholder(\$tag, \$pairs) [*line 97*]

Function Parameters:

- **\$tag**
- **\$pairs**

- **Access public**

void function mamurView::metaContentStr(\$metaName, \$metaFields) [line 164]

Function Parameters:

- **\$metaName**
- **\$metaFields**

void function mamurView::printTemplatePage() [line 234]

- **Access public**

void function mamurView::processPlaceholders(\$text) [line 471]

Function Parameters:

- **\$text**

- **Access public**

void function mamurView::redirect([\$redirect = ""]) [line 421]

Function Parameters:

- **\$redirect**

- **Access** public

void function mamurView::setModel(&\$model) [line 81]

Function Parameters:

- **&\$model**

- **Access** public

void function mamurView::showBuiltPage(\$builtFile) [line 276]

Function Parameters:

- **\$builtFile**

- **Access** public

void function mamurView::templatedView() [line 196]

- **Access** public

start up

This file contains the start up class and custom error functions.

You may wish to adapt these for your own use; if so please change the next line to "NO". ::UPDATE_ALLOWED:YES If set to yes any changes the start directories defined in the line: `mamurStart::setup('./mamur/private','/mamur/public');` will be copied to the update any other changes will be lost. If set to no this file must be modified manually, however, it is not expected that this file will be updated very frequently. This page should not Output anything ie ensure that there are no spaces before php tag and do not use an end tag. Note: Mamur is >5.24 compatible and to avoid naming issues the mamur uses only classes and functions starting with "mamur". Also mamur uses no global variables or sessions. Post 5.3 mamur will run in global namespace and new user modules should use namespaces.

- **Package** mamur
- **Sub-Package** startup
- **Author** Martin Marsh < martinmarsh@sygenius.com>
- **Version** 1.12
- **Copyright** Copyright (c) 2011, Sygenius Ltd
- **Release** Mamur 1.10
- **Mvc** setup
- **Name** start up
- **License** [GNU Public License, version 3](#)

void function mamurAutoClassLoad(\$name) [line 177]

Function Parameters:

- **\$name**

Mamur Autoload Class function registered at start up

Loads a class configured in configuration.xml

void function mamurErrorHandler(\$errno, \$errmsg, \$filename, \$linenum, \$vars) [line 246]

Function Parameters:

- **\$errno \$errno** - the level of the error raised, as an integer.
- **\$errmsg \$errmsg** - the error message, as a string.
- **\$filename \$filename** - the filename that the error was raised in, as a string.
- **\$linenumthe \$linenum** - line number the error was raised at, as an integer.

- **\$vars \$vars** - read only array of every variable that existed in the scope when error was triggered in.

PHP Default Error handler

This is the mamur default Error handler. Its output is controlled by settings in Configuration XML. User error messages can use:
`trigger_error($message,E_USER_ERROR)` - allows variable trace if a setting or
`trigger_error($message,E_USER_NOTICE)`- for general debug messages. firePhp will also be logged errors if the word "TRACE" appears in error message. eg
`trigger_error('TRACE message key words in message')` - will issue firePHP trace and log a `USER_NOTICE` error.

Note: Policy to continue on `USER_ERROR` - add call `exit` in your script after `trigger USER_ERROR` needs or include the word "FATAL" in capitals in message.

unknown_type function `mamurExceptionHandler($exception)` [line 212]

Function Parameters:

- *\$exception* **\$exception**

PHP Exception handler for uncaught exceptions

void function `mamurShutDown()` [line 191]

Fatal Errors must also be trapped and logged

Class `mamurStart`

[line 91]

The class and the access commands above are placed in `mamur.php` file normally located in the home page directory. If there is a `.htaccess` or other uri rewrite system then all `.html` files and `.htm` files should be rooted to this file.

Alternatively every url using the mamur framework must (after any domain name definition) start with `mamur.php` followed by the rest of the page request.

Files with `.php` extension can bypass mamur or be integrated into mamur and handled by special controllers.

This file defines file locations set up configuration class to read the configuration.xml in the mamur user directory and runs the main mamurController invoking the appropriate mvc modules

- **Package** mamur
- **Sub-Package** startup
- **Abstract Element**

void function mamurStart::setup(\$privateDir, \$publicDir) [line 100]

Function Parameters:

- *\$privateDir* **\$privateDir** - directory relative to this file mamur where private files are stored with php r/w access
- *\$publicDir* **\$publicDir** - web home directory for mamur to access resources etc

See class description for details

- **Static**
- **Access** public

configtests.php

- **Package** mamur

require **\$system.'/modules/core/models/mamurConfigData.php'** *[line 11]*

require **\$system.'/modules/core/models/mamurConfig.php'** *[line 12]*

core_mamur_model_tests.php

- **Package** mamur

require **\$system.'/modules/core/models/mamurConfigData.php'** *[line 12]*

require **\$system.'/modules/core/models/mamurConfig.php'** *[line 13]*

require **\$system.'/modules/core/models/mamurModel.php'** *[line 14]*

dataobject_tests.php

- **Package** mamur

require **\$system.'**modules/core/models/mamurDataObject.php' *[line 11]*

modeldataobject_tests.php

- **Package** mamur

require **\$system.'**modules/core/models/mamurConfigData.php' [line 13]

require **\$system.'**modules/core/models/mamurConfig.php' [line 14]

require **\$system.'**modules/core/models/mamurDataObject.php' [line 15]

require **\$system.'**modules/core/models/mamurModel.php' [line 16]

Class coreConfigTests

[line 23]

Unit tests to validate configuration classes

Assumes phpunit has been installed and autoloads

- **Package** mamur
- **Sub-Package** test
- **Author** martinmarsh@sygenius.com

void function coreConfigTests::setUpBeforeClass() [line 25]

- **Static**
- **Access** public

void function coreConfigTests::testConfigClasses() [line 105]

Test settings XML values in Classes group

- **Access** public

void function coreConfigTests::testConfigGlobals() [*line 93*]

Test settings XML values in Globals group

- **Access** public

void function coreConfigTests::testConfigPersist() [*line 180*]

Test config settings persistence and XML config updating

- **Access** public

void function coreConfigTests::testConfigPlaceholders() [*line 151*]

Test settings XML values in Placeholders group

- **Access** public

void function coreConfigTests::testConfigSettings() [*line 79*]

Test settings XML values in Settings group

- **Access** public

void function coreConfigTests::testConfigSetup() [line 56]

- **Access** public

void function coreConfigTests::testSetConfigData() [line 35]

Test mamurConfigData class methods work

- **Access** public

Class coreMamurModeltest

[line 25]

Unit tests to validate core mamurModel class

Assumes phunit has been installed and autoloads

- **Package** mamur
- **Sub-Package** test
- **Author** martinmarsh@sygenius.com

coreMamurModeltest::\$model

mixed = [line 27]

- **Static**
- **Access** public

void function coreMamurModeltest::setUpBeforeClass() [line 29]

- **Static**
- **Access** public

void function coreMamurModeltest::testGetPageContent() [line 132]

void function coreMamurModeltest::testGetSharedContent() [line 143]

void function coreMamurModeltest::testMappedContentbyFirstSection() [line 183]

void function coreMamurModeltest::testModel() [line 87]

- **Access** public

void function coreMamurModeltest::testSetUri() [line 112]

Class dataObjectTests

[line 23]

Unit tests to validate configuration classes

Assumes phpunit has been installed and autoloader

- **Package** mamur
- **Sub-Package** test
- **Author** martinmarsh@sygenius.com

void function dataObjectTests::setUpBeforeClass() [line 27]

- **Static**
- **Access** public

void function dataObjectTests::testDataObjectAttributes() [line 93]

Test Attribute settings

- **Access** public

void function dataObjectTests::testDataObjectConstruct() [line 36]

Test dataobject construct

- **Access** public

void function dataObjectTests::testDataObjectDelete1stRecord() [line 144]

Deleting 1st Record when 1 or less record exists

- **Access** public

void function dataObjectTests::testDataObjectRecords() [line 107]

Test record functions

- **Access** public

void function dataObjectTests::testDataObjectSelectRecord() [line 178]

Record selection tests

- **Access** public

void function `dataObjectTests::testDataObjectSetting()` [*line 60*]

Test setting items and reading them

- **Access** public

Class `modelDataObjectTests` [*line 28*]

Unit tests to validate configuration classes

Assumes `phpunit` has been installed and autoloads

- **Package** `mamur`
- **Sub-Package** `test`
- **Author** `martinmarsh@sygenius.com`

`modelDataObjectTests::$model`

mixed = [*line 30*]

- **Static**
- **Access** public

void function `modelDataObjectTests::setUpBeforeClass()` [*line 32*]

- **Static**
- **Access** public

void function `modelDataObjectTests::testGetDataObject()` [*line 73*]

Test `getDataObject`

- **Access** public

void function `modelDataObjectTests::testSaveDataObjects()` [*line 96*]

Test `saveDataObjects`

- **Access** public

void function `modelDataObjectTests::testSaveDataObjects2fish()` [*line 228*]

Test `saveDataObjects - 2fish`

- **Access** public

void function `modelDataObjectTests::testSaveDataObjects3Des()` [*line 192*]

Test `saveDataObjects - tripleDes`

- **Access** public

void function modelDataObjectTests::testSaveDataObjectsNone() [*line 153*]

Test saveDataObjects - no cipher

- **Access** public

Appendices

Appendix A - Class Trees

Package mamur

coreConfigTests

- PHPUnit_Framework_TestCase
 - [coreConfigTests](#)

coreMamurModeltest

- PHPUnit_Framework_TestCase
 - [coreMamurModeltest](#)

dataObjectTests

- PHPUnit_Framework_TestCase
 - [dataObjectTests](#)

mailclass

- [mailclass](#)

mamurConfig

- [mamurConfig](#)

mamurConfigData

- [mamurConfigData](#)

mamurController

- [mamurController](#)

mamurDataObject

- [mamurDataObject](#)

mamurForm

- [mamurForm](#)

mamurModel

- [mamurModel](#)

mamurPlaceholders

- [mamurPlaceholders](#)

mamurStart

- [mamurStart](#)

mamurView

- [mamurView](#)

modelDataObjectTests

- PHPUnit_Framework_TestCase
 - [modelDataObjectTests](#)

pageMeta

- [pageMeta](#)

Index

C

coreConfigTests::testSetConfigData()	88
<i>Test mamurConfigData class methods work</i>	
coreMamurModeltest	88
<i>Unit tests to validate core mamurModel class</i>	
coreConfigTests::testConfigSetup()	88
coreConfigTests::testConfigSettings()	87
<i>Test settings XML values in Settings group</i>	
coreConfigTests::testConfigPlaceholders()	87
<i>Test settings XML values in Placeholders group</i>	
coreMamurModeltest::\$model	88
coreMamurModeltest::setUpBeforeClass()	89
coreMamurModeltest::testModel()	89
coreMamurModeltest::testSetUri()	89
coreMamurModeltest::testMappedContentbyFirstSection()	89
coreMamurModeltest::testGetSharedContent()	89
coreMamurModeltest::testGetPageContent()	89
coreConfigTests::testConfigPersist()	87
<i>Test config settings persistence</i>	
coreConfigTests::testConfigGlobals()	87
<i>Test settings XML values in Globals group</i>	
constructor mamurModel::__construct()	32
<i>Constructor sets up properties according to configuration</i>	
constructor mamurForm::__construct()	61
constructor mamurDataObject::__construct()	18
<i>Constructor initialises data in a simple associated data array</i>	
constructor mamurConfigData::__construct()	15
<i>Constructor initialises data in a simple associated data array of values or for multi-attribute configuration an array of mamurConfigData objects.</i>	
constructor pageMeta::__construct()	3
constructor mamurPlaceholders::__construct()	64
constructor mamurView::__construct()	74
coreConfigTests::setUpBeforeClass()	86
coreConfigTests::testConfigClasses()	86
<i>Test settings XML values in Classes group</i>	
coreConfigTests	86
<i>Unit tests to validate configuration classes</i>	
core_mamur_model_tests.php	84
configtests.php	83
constructor mailclass::mailclass()	2

D

dataObjectTests::testDataObjectRecords()	90
<i>Test record functions</i>	
dataObjectTests::testDataObjectSelectRecord()	90
<i>Record selection tests</i>	
dataObjectTests::testDataObjectSetting()	91
<i>Test setting items and reading them</i>	
dataObjectTests::testDataObjectDelete1stRecord()	90
<i>Deleting 1st Record when 1 or less record exists</i>	
dataObjectTests::testDataObjectConstruct()	90
<i>Test dataobject construct</i>	
dataObjectTests	89
<i>Unit tests to validate configuration classes</i>	
dataObjectTests::setUpBeforeClass()	89
dataObjectTests::testDataObjectAttributes()	90
<i>Test Attribute settings</i>	
dataobject_tests.php	85

M

mamurModel::setTag()	57
<i>Sets a Tag</i>	
mamurModel::setSessionCookie()	56
<i>Sets an encrypted session based cookie which contains the</i>	
mamurModel::setPageUri()	56
<i>Sets page uri and associated internal variables using the value of the url parameter passed.</i>	
mamurModel::setTagList()	57
<i>Sets an array (list) of tags</i>	
mamurModel::setUpSession()	57
<i>Sets up session and processes cookies as per configuration</i>	
mamurModel::unique_serial()	58
<i>generates a unique serial number based on date,microtime, call count and a random number so there is very low risk of duplication in moderately busy systems.</i>	
mamurModel::timeStampToDate()	58
mamurModel::setUser()	58
<i>Sets Current User details</i>	
mamurModel::setOption()	56
<i>Saves an option</i>	
mamurModel::setNonce()	55
<i>A nonce is a number used once and can be used to prevent forms</i>	
mamurModel::setEditStatus()	54
<i>Sets page edit status</i>	
mamurModel::saveDataObjects()	53
mamurModel::rmdir()	53
<i>recursive remove directory - protected as it is dangerous.</i>	
mamurModel::setErrorPageTemplate()	54
mamurModel::setGlobal()	54
<i>Sets a global for use on current page the value is not persisted.</i>	
mamurModel::setLocidCookie()	55
<i>Sets a cookie to help identify a users at a location</i>	
mamurModel::setHostData()	54

<i>Looks at the a domain name (normally the host for current server)</i>	
mamurModel::userLogin()	59
<i>Logs in current user</i>	
mamurModel::userLogOut()	59
<i>Logs out a user</i>	
mamurForm::generalPlaceholder()	63
mamurForm::fldAttrib()	62
mamurForm::endSelect()	62
mamurForm::optionList()	63
mamurForm::validate()	63
mamurPlaceholders::all()	64
mamurPlaceholders	63
<i>This file contains the core view Class - mamurPlaceholders</i>	
mamurForm::doTextarea()	62
mamurForm::doSelect()	62
mamurForm	60
<i>This file contains the core view Class - mamurForm</i>	
mamurViewReplacePlaceholder()	60
<i>Warning this is a call back function and NOT a class method.</i>	
mamurView	60
<i>This file contains the static main view Class - mamurView</i>	
mamurForm::doEndform()	61
mamurForm::doForm()	61
mamurForm::doOption()	62
mamurForm::doInput()	62
mamurModel::removePage()	53
<i>Removes a page</i>	
mamurModel::relativeDir()	52
<i>relativeDir extends a directory reference by adding and additional string the additional string can be a subdirectory or page name.</i>	
mamurModel::getUser()	46
<i>Gets current user session data</i>	
mamurModel::getUrl()	46
mamurModel::getTopPageDir()	46
mamurModel::GetUserIP()	46
<i>Gets best value for user IP address</i>	
mamurModel::getUserZoneOffset()	46
mamurModel::getXmlPageType()	47
mamurModel::getWebRoot()	47
mamurModel::getTopDomain()	46
mamurModel::getTemplateFile()	45
mamurModel::getSharedContent()	44
<i>Gets Shared Content by name, type and by optional group and mapping</i>	
mamurModel::getServerBase()	43
<i>the server base is usually the user directory but a call back function allows it to be switched programatically to another area. This would allow another file area to be used for the page and content area.</i>	
mamurModel::getSalt()	43
<i>Gets a salt by concatinating two equal length strings from</i>	
mamurModel::getSharedContentBase()	44
mamurModel::getSubDomain()	45
mamurModel::getTagData()	45

mamurModel::getTag()	45
<i>Gets a tag value</i>	
mamurModel::isError404Page()	47
mamurModel::isLoggedIn()	47
<i>Checks if user logged in</i>	
mamurModel::registerPageProcessFunction()	51
mamurModel::registerPagePrintFunction()	51
mamurModel::readPageXML()	50
<i>Reads page XML file which defines template and page meta data</i>	
mamurModel::registerServerBaseFunction()	51
mamurModel::registerSessionClearFunction()	51
mamurModel::registerUrlFunction()	52
mamurModel::registerSessionLogoutFunction()	52
mamurModel::readDataObjects()	50
mamurModel::processXML()	50
mamurModel::pageProcessHookContinue()	48
mamurModel::openPageMeta()	48
mamurModel::locidCookie()	48
<i>Reads and saves status of user location cookie a value</i>	
mamurModel::pageTime()	48
<i>Gets page processing lapsed time</i>	
mamurModel::passParameters()	49
<i>Saves php parameters defined by a php tag</i>	
mamurModel::processPageMetaData()	50
<i>Process the current Page meta data including the template to use</i>	
mamurModel::pbkdf2()	49
<i>PBKDF2 Implementation (as described in RFC 2898)</i>	
<i>This is recommended for password hashing but takes some computing power due to min 1000 loops, so only use on infrequent activities such as for passwords.</i>	
mamurPlaceholders::build_random()	65
mamurPlaceholders::date()	65
mamurView::includePageFile()	77
mamurView::getPlaceholder()	77
mamurView::doUniqueSerialPlaceholder()	77
mamurView::insertPlaceholder()	77
mamurView::metaContentStr()	78
mamurView::redirect()	78
mamurView::processPlaceholders()	78
mamurView::printTemplatePage()	78
mamurView::doTagPlaceholder()	76
<i>This method dynamically replaces a 'tag' type</i>	
mamurView::doRandomPlaceholder()	76
mamurView::doNoncePlaceholder()	75
mamurView::doGlobalPlaceholder()	75
mamurView::doDatePlaceholder()	74
mamurView::dopage_pageTimerPlaceholder()	75
mamurView::dopage_timermsPlaceholder()	75
mamurView::doPhpPlaceholder()	76
mamurView::doPhpAndPrint()	76
mamurView::setModel()	79
mamurView::showBuiltPage()	79
modelDataObjectTests::testGetDataObject()	92

<i>Test getDataObject</i>	
modelDataObjectTests::setUpBeforeClass()	92
modelDataObjectTests::\$model	91
modelDataObjectTests::testSaveDataObjects()	92
<i>Test saveDataObjects</i>	
modelDataObjectTests::testSaveDataObjects2fish()	92
<i>Test saveDataObjects - 2fish</i>	
modelDataObjectTests::testSaveDataObjectsNone()	93
<i>Test saveDataObjects - no cipher</i>	
modelDataObjectTests::testSaveDataObjects3Des()	92
<i>Test saveDataObjects - tripleDes</i>	
modelDataObjectTests	91
<i>Unit tests to validate configuration classes</i>	
modeldataobject_tests.php	86
mamurErrorHandler()	80
<i>PHP Default Error handler</i>	
mamurAutoClassLoad()	80
<i>Mamu Autoload Class function registered at start up</i>	
mamurView::templatedView()	79
mamurExceptionHandler()	81
<i>PHP Exception handler for uncaught exceptions</i>	
mamurShutDown()	81
<i>Fatal Errors must also be trapped and logged</i>	
mamurStart::setup()	82
<i>See class description for details</i>	
mamurStart	81
<i>The class and the access commands above are placed in mamur.php file normally located in the home page directory. If there is a .htaccess or other uri rewrite system then all .html files and .htm files should be rooted to this file.</i>	
mamurView::directPhpView()	74
mamurView::defaultErrorPage()	74
mamurPlaceholders::page_content()	68
<i>page_content tag inserts content associated with a page and stored in the page specific</i>	
mamurPlaceholders::optionlist()	68
mamurPlaceholders::option()	67
mamurPlaceholders::page_selected()	69
mamurPlaceholders::page_timer()	69
mamurPlaceholders::php()	69
mamurPlaceholders::page_timerms()	69
mamurPlaceholders::nonce()	67
mamurPlaceholders::mamur()	67
mamurPlaceholders::form()	66
mamurPlaceholders::endselect()	65
mamurPlaceholders::endform()	65
mamurPlaceholders::generalPlaceholder()	66
mamurPlaceholders::globalTag()	66
mamurPlaceholders::input()	67
mamurPlaceholders::http_header()	66
mamurPlaceholders::random()	69
mamurPlaceholders::select()	70
mamurView::\$pageOutput	73
mamurView::\$pageBuildId	73
mamurView::\$pageBuild	73

mamurView::\$pagePhp	73
mamurView::\$placeholderClasses	73
mamurView::\$urlDir	74
mamurView::\$templateTags	74
mamurView::\$oddeven	72
mamurView::\$model	72
mamurPlaceholders::textarea()	71
mamurPlaceholders::tag()	70
mamurPlaceholders::shared()	70
<i>shared tags inserts shared content of a particular type</i>	
mamurPlaceholders::title()	71
mamurPlaceholders::unique_serial()	71
mamurView::\$mamur	72
mamurView	71
<i>mamurView is a basic class to print out pages.</i>	
mamurModel::getRandomString()	42
<i>Gets a random string of Letters and numbers of</i>	
mamurModel::getPluginDir()	42
mamurDataObject::back()	19
mamurDataObject::appendRecord()	18
<i>Appends a new record defined by an associated array</i>	
mamurDataObject	17
<i>mamurDataObject encapsulates data in an array structure.</i>	
mamurDataObject::deleteRecord()	19
<i>Deletes specified record or if not given the current record</i>	
mamurDataObject::getAll()	19
<i>getAll method returns entire data table array</i>	
mamurDataObject::getRecord()	20
<i>getRecord method returns associated array of a record</i>	
mamurDataObject::getCurrentRecordNumber()	20
<i>returns current record number</i>	
mamurDataObject::getAttribute()	19
mamurConfigData::__unset()	17
<i>allows unset(\$dataclass->variable) construct</i>	
mamurConfigData::__set()	16
<i>__set magic method allows \$dataClass->variable=value construct</i>	
mamurConfig::__get()	14
<i>__get magic method allows</i>	
mamurConfig::updateConfig()	14
<i>This is normally called by controller at end of page</i>	
mamurConfig::setPlugin()	13
<i>Defines a plugin and places the plugin details in configuration.xml</i>	
mamurConfigData	15
<i>mamurConfigData encapsulates a simple data array used to hold configuration data so that name=variable configurations can be accessed by \$dataClass->\$myVariable constructs.</i>	
mamurConfigData::getAll()	16
<i>getAll method returns entire data array</i>	
mamurConfigData::__isset()	16
<i>__isset magic method allows isset(\$dataClass->variable) construct</i>	
mamurConfigData::__get()	16
<i>__get magic method allows</i>	
mamurDataObject::getStatus()	20

mamurDataObject::last()	20
mamurModel	24
<i>mamurModel is the main model class to access data and files</i>	
mamurDataObject::__unset()	23
<i>allows unset(\$dataclass->variable) construct</i>	
mamurDataObject::__set()	23
<i>__set magic method allows \$dataClass->variable=value construct</i>	
mamurModel::\$countSerial	24
mamurModel::\$dataObjects	25
mamurModel::\$defaultPageName	25
mamurModel::\$defaultPageExt	25
mamurDataObject::__isset()	22
<i>__isset magic method allows isset(\$dataClass->variable) construct</i>	
mamurDataObject::__get()	22
<i>__get magic method allows</i>	
mamurDataObject::persist()	21
mamurDataObject::next()	21
<i>If possible Moves record pointer forward by one</i>	
mamurDataObject::modified()	20
mamurDataObject::read()	21
mamurDataObject::setAttribute()	21
mamurDataObject::setStatus()	22
mamurDataObject::setCurrentRecordNumber()	21
<i>Sets current record returning the new record number set</i>	
mamurConfig::setGlobal()	13
mamurConfig::runPlugIns()	13
mailclass::\$to	2
mailclass::\$subject	2
mailclass::\$sign	2
mailclass::copyto()	2
mailclass::message()	2
mailclass::send()	3
mailclass::postmessage()	2
mailclass::\$replyto	2
mailclass::\$message	1
mailclass::\$colour	1
mailclass::\$cc	1
mailclass::\$bcc	1
mailclass::\$error	1
mailclass::\$from	1
mailclass::\$mailbody	1
mailclass::\$headers	1
mainController	5
<i>This file contains the static main Controller Class - mamurController</i>	
<i>Control may be dispatched to other controllers and mvc patterns according to configuration xml.</i>	
mamurController	5
<i>mamurController is a static class which is always the top level</i>	
mamurConfig::getPluginDir()	11
mamurConfig::getInstance()	11
mamurConfig::createPlugIns()	11
<i>Instantiates each plugin and saves the object in a config</i>	
mamurConfig::getPlugInsToLoad()	11

<i>Gets a list of plugins to load and associated data</i>	
mamurConfig::persistSetting()	12
<i>Persists to the XML config data a setting value</i>	
mamurConfig::relativeDir()	13
mamurConfig::processConfig()	12
<i>processConfig saves configuration setup in index.php and bootstrap</i>	
mamurConfig	10
<i>The class provides a single point of reference to get all configuration details</i>	
mamurModel	10
<i>This file contains the main model Class - mamurModel</i>	
mamurController::\$view	6
mamurController::\$model	6
mamurController::\$controller	6
mamurController::getController()	6
<i>Get controller gets the current controller dispatched by processUri</i>	
mamurController::getView()	7
<i>Get view gets the current view instance</i>	
mamurController::response()	7
<i>response method has the control steps to respond to a url and update print buffer.</i>	
mamurController::processUri()	7
<i>This method processes all page requests either passing</i>	
mamurModel::\$error404PageName	25
mamurModel::\$global	25
mamurModel::getEditStatus()	37
<i>get page edit status</i>	
mamurModel::getDataObject()	36
mamurModel::getDataBaseDir()	36
mamurModel::getEncryptedSession()	37
<i>Gets session data in an encripted string identical to that</i>	
mamurModel::getError404PageName()	37
mamurModel::getGlobal()	38
mamurModel::getErrorPage()	38
mamurModel::getCookieSerialNo()	36
<i>Generates a cookie unique serial number</i>	
mamurModel::getContentList()	36
<i>Gets a list of content names give a search pattern for a name</i>	
mamurModel::dirListToDataObject()	34
<i>Places a directory list into a data object at the current record</i>	
mamurModel::deleteDataObject()	34
mamurModel::decrypt()	33
<i>Decrypt a String using set cipher and optionally</i>	
mamurModel::doPagePagePrintCallBacks()	34
mamurModel::encrypt()	35
<i>encrypt a String using set cipher and optionally</i>	
mamurModel::getContentBase()	35
mamurModel::getApiSalt()	35
<i>Gets an Api salt in a similar way to getSalt</i>	
mamurModel::getHomeUri()	38
mamurModel::getHostDomain()	38
mamurModel::getPageDirList()	41
mamurModel::getPageDir()	41
mamurModel::getPageContent()	41

<i>Gets Page Specific Content by name</i>	
mamurModel::getPageExt()	41
mamurModel::getPageName()	41
mamurModel::getPhpBase()	42
mamurModel::getParameters()	42
<i>Gets php pass parameters defined by a php tag</i>	
mamurModel::getOption()	40
<i>Gets a named option</i>	
mamurModel::getNonMamurPage()	40
mamurModel::getMamurlogDir()	39
mamurModel::getMamurBaseDir()	38
mamurModel::getHostScheme()	38
mamurModel::getMamurSystemDir()	39
mamurModel::getMamurUrl()	39
mamurModel::getNonce()	40
<i>A named Nonce value will be returned if it has been set.</i>	
mamurModel::getMamurUserDir()	39
mamurModel::convertToUserZone()	33
mamurModel::checkLogOut()	33
<i>Checks to see if logout required.</i>	
mamurModel::\$mamurURL	28
mamurModel::\$mamurSystemDir	28
mamurModel::\$mamurPluginDir	27
mamurModel::\$oldSession	28
mamurModel::\$options	28
mamurModel::\$pageDirList	29
mamurModel::\$pageDir	28
mamurModel::\$mamurlogDir	27
mamurModel::\$mamurBaseDir	27
mamurModel::\$inSession	26
mamurModel::\$hostScheme	26
mamurModel::\$hostdomain	26
mamurModel::\$lastErrorPage	26
mamurModel::\$locid	26
mamurModel::\$logOutFlag	27
mamurModel::\$locidAccepted	27
mamurModel::\$pageExt	29
mamurModel::\$pageName	29
mamurModel::\$themesDir	31
mamurModel::\$templateTags	31
mamurModel::\$tags	31
mamurModel::\$topdomain	32
mamurModel::\$urlCallBack	32
mamurModel::\$xmlPageType	32
mamurModel::\$webBaseDir	32
mamurModel::\$subdomain	31
mamurModel::\$sessionClearCallBack	31
mamurModel::\$pageQuery	30
mamurModel::\$pageProcessCallBack	29
mamurModel::\$pagePagePrintCallBack	29
mamurModel::\$pageURL	30
mamurModel::\$phpParameters	30
mamurModel::\$session	30

mamurModel::\$serverBaseRequestCallBack	30
mailclass	1

P

pageMeta::updateFromDataSet()	3
pageMeta	3

S

start up	80
<i>This file contains the start up class and custom error functions.</i>	
static configuration class mamurConfig	9
<i>This file contains the configuration class</i>	