

rolisz  
0.0.3

Generated by Doxygen 1.7.3

Wed May 4 2011 12:18:36



# Contents

<b>1</b>	<b>Todo List</b>	<b>1</b>
<b>2</b>	<b>Namespace Index</b>	<b>3</b>
2.1	Namespace List . . . . .	3
<b>3</b>	<b>Data Structure Index</b>	<b>5</b>
3.1	Class Hierarchy . . . . .	5
<b>4</b>	<b>Data Structure Index</b>	<b>7</b>
4.1	Data Structures . . . . .	7
<b>5</b>	<b>File Index</b>	<b>9</b>
5.1	File List . . . . .	9
<b>6</b>	<b>Namespace Documentation</b>	<b>11</b>
6.1	Roland Namespace Reference . . . . .	11
6.1.1	Detailed Description . . . . .	11
6.2	rolisz Namespace Reference . . . . .	11
6.2.1	Detailed Description . . . . .	11
<b>7</b>	<b>Data Structure Documentation</b>	<b>13</b>
7.1	acl Class Reference . . . . .	13
7.1.1	Detailed Description . . . . .	14
7.1.2	Constructor & Destructor Documentation . . . . .	14
7.1.2.1	__construct . . . . .	14
7.1.3	Member Function Documentation . . . . .	15
7.1.3.1	addNew . . . . .	15
7.1.3.2	checkConsistency . . . . .	15
7.1.3.3	deleteACL . . . . .	15
7.1.3.4	editACL . . . . .	16
7.1.3.5	getACL . . . . .	16
7.1.3.6	getName . . . . .	16
7.1.3.7	getPath . . . . .	17
7.1.3.8	init . . . . .	17
7.1.3.9	isAllowed . . . . .	17
7.1.3.10	run . . . . .	17
7.1.4	Field Documentation . . . . .	18
7.1.4.1	\$acl . . . . .	18
7.1.4.2	\$denyFunction . . . . .	18
7.1.4.3	\$prefix . . . . .	18

	7.1.4.4	\$requester . . . . .	18
	7.1.4.5	\$source . . . . .	18
	7.1.4.6	\$sourceType . . . . .	18
7.2		base Class Reference . . . . .	18
	7.2.1	Detailed Description . . . . .	19
	7.2.2	Constructor & Destructor Documentation . . . . .	19
	7.2.2.1	__construct . . . . .	19
	7.2.3	Member Function Documentation . . . . .	19
	7.2.3.1	check . . . . .	19
	7.2.3.2	get . . . . .	19
	7.2.3.3	set . . . . .	20
	7.2.4	Field Documentation . . . . .	20
	7.2.4.1	\$executionPoints . . . . .	20
	7.2.4.2	\$global . . . . .	20
	7.2.4.3	\$plugins . . . . .	20
	7.2.4.4	AppName . . . . .	20
	7.2.4.5	Module . . . . .	20
	7.2.4.6	Version . . . . .	20
7.3		databaseAdapter Class Reference . . . . .	20
	7.3.1	Detailed Description . . . . .	21
	7.3.2	Constructor & Destructor Documentation . . . . .	21
	7.3.2.1	__construct . . . . .	21
	7.3.3	Member Function Documentation . . . . .	21
	7.3.3.1	alterTable . . . . .	21
	7.3.3.2	commit . . . . .	21
	7.3.3.3	createTable . . . . .	21
	7.3.3.4	disconnect . . . . .	21
	7.3.3.5	dropTable . . . . .	22
	7.3.3.6	escapeValue . . . . .	22
	7.3.3.7	fetchAll . . . . .	22
	7.3.3.8	fetchRow . . . . .	22
	7.3.3.9	getError . . . . .	22
	7.3.3.10	getInsertID . . . . .	22
	7.3.3.11	numAffected . . . . .	22
	7.3.3.12	numRows . . . . .	22
	7.3.3.13	query . . . . .	22
	7.3.3.14	rollback . . . . .	22
	7.3.3.15	selectDatabase . . . . .	22
	7.3.3.16	startTransaction . . . . .	23
	7.3.3.17	tableExists . . . . .	23
7.4		form Class Reference . . . . .	23
	7.4.1	Detailed Description . . . . .	24
	7.4.2	Constructor & Destructor Documentation . . . . .	24
	7.4.2.1	__construct . . . . .	24
	7.4.3	Member Function Documentation . . . . .	24
	7.4.3.1	__call . . . . .	24
	7.4.3.2	buildString . . . . .	24
	7.4.3.3	getDefaultMethod . . . . .	24
	7.4.3.4	getName . . . . .	25
	7.4.3.5	getString . . . . .	25

7.4.3.6	input . . . . .	25
7.4.3.7	removeInput . . . . .	25
7.4.3.8	show . . . . .	25
7.4.3.9	validate . . . . .	25
7.4.4	Field Documentation . . . . .	26
7.4.4.1	\$form . . . . .	26
7.4.4.2	\$inputs . . . . .	26
7.4.4.3	\$model . . . . .	26
7.4.4.4	\$string . . . . .	26
7.5	MySQLiteDatabase Class Reference . . . . .	26
7.5.1	Detailed Description . . . . .	27
7.5.2	Constructor & Destructor Documentation . . . . .	27
7.5.2.1	__construct . . . . .	27
7.5.2.2	__destruct . . . . .	27
7.5.3	Member Function Documentation . . . . .	28
7.5.3.1	alterTable . . . . .	28
7.5.3.2	commit . . . . .	28
7.5.3.3	connect . . . . .	28
7.5.3.4	createTable . . . . .	28
7.5.3.5	disconnect . . . . .	29
7.5.3.6	dropTable . . . . .	29
7.5.3.7	escapeValue . . . . .	29
7.5.3.8	fetchAll . . . . .	29
7.5.3.9	fetchFirst . . . . .	30
7.5.3.10	fetchRow . . . . .	30
7.5.3.11	getError . . . . .	30
7.5.3.12	getInsertID . . . . .	30
7.5.3.13	numAffected . . . . .	31
7.5.3.14	numRows . . . . .	31
7.5.3.15	Query . . . . .	31
7.5.3.16	rollback . . . . .	31
7.5.3.17	selectDatabase . . . . .	31
7.5.3.18	startTransaction . . . . .	32
7.5.3.19	tableExists . . . . .	32
7.5.4	Field Documentation . . . . .	32
7.5.4.1	\$connection . . . . .	32
7.5.4.2	\$database . . . . .	32
7.5.4.3	\$queries . . . . .	32
7.5.4.4	\$result . . . . .	33
7.6	plugin Class Reference . . . . .	33
7.6.1	Detailed Description . . . . .	33
7.6.2	Constructor & Destructor Documentation . . . . .	33
7.6.2.1	__construct . . . . .	33
7.6.3	Member Function Documentation . . . . .	34
7.6.3.1	getDefaultExecutionPoints . . . . .	34
7.6.3.2	getDefaultMethod . . . . .	34
7.6.3.3	getName . . . . .	34
7.6.3.4	run . . . . .	35
7.7	pluginStructure Class Reference . . . . .	35
7.7.1	Detailed Description . . . . .	35

7.7.2	Constructor & Destructor Documentation . . . . .	35
7.7.2.1	__construct . . . . .	35
7.7.3	Member Function Documentation . . . . .	36
7.7.3.1	checkExecutionPoint . . . . .	36
7.7.3.2	checkPlugin . . . . .	36
7.7.3.3	registerExecutionPoint . . . . .	36
7.7.3.4	registerPlugin . . . . .	36
7.7.3.5	runPlugins . . . . .	37
7.7.3.6	unregisterExecutionPoint . . . . .	37
7.7.3.7	unregisterPlugin . . . . .	37
7.8	Query Class Reference . . . . .	37
7.8.1	Detailed Description . . . . .	38
7.8.2	Constructor & Destructor Documentation . . . . .	38
7.8.2.1	__construct . . . . .	38
7.8.3	Member Function Documentation . . . . .	38
7.8.3.1	buildFilters . . . . .	38
7.8.3.2	buildJoins . . . . .	39
7.8.3.3	buildOrderBy . . . . .	39
7.8.3.4	buildQuery . . . . .	39
7.8.3.5	getCount . . . . .	39
7.9	rolisz Class Reference . . . . .	39
7.9.1	Detailed Description . . . . .	40
7.9.2	Member Function Documentation . . . . .	40
7.9.2.1	autoload . . . . .	40
7.9.2.2	call . . . . .	40
7.9.2.3	connect . . . . .	41
7.9.2.4	fixBraces . . . . .	41
7.9.2.5	fixSlashes . . . . .	41
7.9.2.6	http404 . . . . .	42
7.9.2.7	plugin . . . . .	42
7.9.2.8	route . . . . .	42
7.9.2.9	run . . . . .	42
7.9.2.10	runPlugins . . . . .	42
7.9.2.11	start . . . . .	42
7.9.2.12	table . . . . .	42
7.9.2.13	urlFor . . . . .	43
7.10	router Class Reference . . . . .	43
7.10.1	Detailed Description . . . . .	43
7.10.2	Member Function Documentation . . . . .	44
7.10.2.1	matches . . . . .	44
7.10.2.2	route . . . . .	44
7.10.2.3	run . . . . .	44
7.10.2.4	urlFor . . . . .	44
7.11	table Class Reference . . . . .	45
7.11.1	Detailed Description . . . . .	46
7.11.2	Constructor & Destructor Documentation . . . . .	47
7.11.2.1	__construct . . . . .	47
7.11.3	Member Function Documentation . . . . .	47
7.11.3.1	__call . . . . .	47
7.11.3.2	__get . . . . .	47

7.11.3.3	__isset . . . . .	47
7.11.3.4	__set . . . . .	48
7.11.3.5	__sleep . . . . .	48
7.11.3.6	addRelation . . . . .	48
7.11.3.7	addRelationM2M . . . . .	48
7.11.3.8	addRelationM2MS . . . . .	49
7.11.3.9	addRelationS . . . . .	49
7.11.3.10	analyzeRelations . . . . .	49
7.11.3.11	countRows . . . . .	50
7.11.3.12	countRowsS . . . . .	50
7.11.3.13	delete . . . . .	50
7.11.3.14	find . . . . .	50
7.11.3.15	findS . . . . .	51
7.11.3.16	getColumns . . . . .	51
7.11.3.17	getData . . . . .	51
7.11.3.18	getPrimaryKey . . . . .	52
7.11.3.19	hydrate . . . . .	52
7.11.3.20	importRows . . . . .	52
7.11.3.21	isHydrated . . . . .	52
7.11.3.22	save . . . . .	53
7.11.3.23	set . . . . .	53
7.11.3.24	setAll . . . . .	53
7.11.4	Field Documentation . . . . .	53
7.11.4.1	\$connection . . . . .	53
7.11.4.2	\$modifiedData . . . . .	53
7.11.4.3	\$originalData . . . . .	53
7.11.4.4	\$primaryKey . . . . .	53
7.11.4.5	\$relations . . . . .	53
7.11.4.6	\$table . . . . .	54
7.11.4.7	\$tables . . . . .	54
7.12	template Class Reference . . . . .	54
7.12.1	Detailed Description . . . . .	55
7.12.2	Constructor & Destructor Documentation . . . . .	55
7.12.2.1	__construct . . . . .	55
7.12.3	Member Function Documentation . . . . .	55
7.12.3.1	__get . . . . .	55
7.12.3.2	__set . . . . .	55
7.12.3.3	__toString . . . . .	55
7.12.3.4	assign . . . . .	56
7.12.3.5	checkCompile . . . . .	56
7.12.3.6	compile . . . . .	56
7.12.3.7	getOutput . . . . .	56
7.12.3.8	includeSub . . . . .	57
7.12.3.9	view . . . . .	57
7.12.4	Field Documentation . . . . .	57
7.12.4.1	\$values . . . . .	57
<b>8</b>	<b>File Documentation</b>	<b>59</b>
8.1	C:/wamp/www/framework/base.php File Reference . . . . .	59
8.2	C:/wamp/www/framework/databaseAdapter.php File Reference . . . . .	59

8.3	C:/wamp/www/framework/misc_functions.php File Reference . . . . .	59
8.3.1	Function Documentation . . . . .	60
8.3.1.1	array_filter_key . . . . .	60
8.3.1.2	recursify . . . . .	60
8.4	C:/wamp/www/framework/plugin.php File Reference . . . . .	60
8.5	C:/wamp/www/framework/plugins/acl.php File Reference . . . . .	60
8.6	C:/wamp/www/framework/plugins/form.php File Reference . . . . .	61
8.7	C:/wamp/www/framework/pluginStructure.php File Reference . . . . .	61
8.8	C:/wamp/www/framework/rolisz.php File Reference . . . . .	61
8.9	C:/wamp/www/framework/router.php File Reference . . . . .	61
8.10	C:/wamp/www/framework/table.php File Reference . . . . .	62
8.10.1	Enumeration Type Documentation . . . . .	62
8.10.1.1	MANY . . . . .	62
8.10.1.2	NOT_ANALYZED . . . . .	62
8.10.1.3	NOT_RECOGNIZED . . . . .	62
8.10.1.4	ONE_TO_MANY . . . . .	62
8.10.1.5	SINGLE . . . . .	62
8.11	C:/wamp/www/framework/template.php File Reference . . . . .	62



# Chapter 1

## Todo List

**Global MySQLiDatabase::alterTable (p. 28)(\$name, \$params)** some checks for valid column names and parameters

**Global MySQLiDatabase::Query (p. 31)(\$query)** change return values for insert, update, delete, select

**Namespace rolisz (p. 11)** Error handling function

- Language Detection

- Internationalization

- Set conditions for dynamic part of url

- Make the interface implement countable, iterator interfaces

**Global table::\$tables (p. 54)** allow objects representing same table, but different columns

**Global table::\_\_call (p. 47)(\$name, \$args)** \$args could be a table of \$name

**Global table::addRelationM2M (p. 48)(\$connectortable, \$thisid, \$mappedid, \$connectedtable, \$thatid, \$cmappedid)**  
find a way to reduce arguments



## Chapter 2

# Namespace Index

### 2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<b>Roland</b>	.....	11
<b>rolisz</b>	.....	11



## Chapter 3

# Data Structure Index

### 3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

base . . . . .	18
plugin . . . . .	33
acl . . . . .	13
form . . . . .	23
pluginStructure . . . . .	35
Query . . . . .	37
rolisz . . . . .	39
router . . . . .	43
table . . . . .	45
template . . . . .	54
databaseAdapter . . . . .	20
MySQLiDatabase . . . . .	26



## Chapter 4

# Data Structure Index

### 4.1 Data Structures

Here are the data structures with brief descriptions:

<b>acl</b> . . . . .	13
<b>base</b> . . . . .	18
<b>databaseAdapter</b> . . . . .	20
<b>form</b> . . . . .	23
<b>MySQLiDatabase</b> . . . . .	26
<b>plugin</b> . . . . .	33
<b>pluginStructure</b> . . . . .	35
<b>Query</b> . . . . .	37
<b>rolisz</b> . . . . .	39
<b>router</b> . . . . .	43
<b>table</b> . . . . .	45
<b>template</b> . . . . .	54





## Chapter 5

# File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

C:/wamp/www/framework/ <b>base.php</b>	59
C:/wamp/www/framework/ <b>databaseAdapter.php</b>	59
C:/wamp/www/framework/ <b>misc_functions.php</b>	59
C:/wamp/www/framework/ <b>plugin.php</b>	60
C:/wamp/www/framework/ <b>pluginStructure.php</b>	61
C:/wamp/www/framework/ <b>rolisz.php</b>	61
C:/wamp/www/framework/ <b>router.php</b>	61
C:/wamp/www/framework/ <b>table.php</b>	62
C:/wamp/www/framework/ <b>template.php</b>	62
C:/wamp/www/framework/plugins/ <b>acl.php</b>	60
C:/wamp/www/framework/plugins/ <b>form.php</b>	61



## Chapter 6

# Namespace Documentation

### 6.1 Roland Namespace Reference

#### 6.1.1 Detailed Description

Szabo

### 6.2 rolisz Namespace Reference

#### 6.2.1 Detailed Description

##### Author

**Roland** (p. 11) Szabo

##### Todo

Error handling function  
Language Detection  
Internationalization  
Set conditions for dynamic part of url

##### Author

**Roland** (p. 11) Szabo

##### Todo

Make the interface implement countable, iterator interfaces

##### Author

**Roland** (p. 11) Szabo

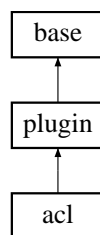


## Chapter 7

# Data Structure Documentation

### 7.1 acl Class Reference

Inheritance diagram for acl:



#### Public Member Functions

- **\_\_construct** (\$requester, \$prefix= "", \$deny= 'default', \$sourceType= 'db')
- **getName** ()
- **run** ()
- **isAllowed** (\$function)
- **init** (\$prefix= "", \$dbCon=FALSE)
- **addNew** (\$type, \$what, \$where=FALSE)
- **editACL** (\$type, \$what, \$towhat, \$where=FALSE)
- **deleteACL** (\$type, \$what)
- **getACL** (\$type)

#### Private Member Functions

- **checkConsistency** ()
- **getPath** (\$array)

## Private Attributes

- **\$acl**
- **\$source**
- **\$sourceType**
- **\$requester**
- **\$prefix = NULL**
- **\$denyFunction**

### 7.1.1 Detailed Description

This plugin allows you to easily work with Access Control Lists. It integrates with the router class, automatically checking if the current requester has permission to run the functions associated with the URL.

#### Author

**Roland** (p. 11) Szabo

### 7.1.2 Constructor & Destructor Documentation

**7.1.2.1** `__construct ( $ requester, $ prefix = ' ', $ deny = ' default ', $ sourceType = ' db' )`

Constructor for ACL lists. Initializes internal list. The third parameter Fourth parameter is

#### Parameters

string	<i>\$requester</i>	the requester that will be used for ACL.
function	<i>\$deny</i>	indicates a function that will be called instead of those that the requester can't access.
array   string	<i>\$prefix</i>	is either the prefix for the tables, if database based ACL's are used or it can be an array, passed like this: <code>array( 'requesters' =&gt; array( 'users' =&gt; array( 'test' , 'rolisz' ), 'mods' =&gt; array( 'bad_mod' ), 'admins' ), 'resources' =&gt; array( 'posts' , 'stats' , 'comments' , 'users' ), 'actions' =&gt; array( 'view' , 'edit' , 'delete' , 'ban' ), 'permissions' =&gt; array( array( 'users' , 'posts' , 'view' ), array( 'users' , 'comments' , 'view' ), array( 'users' , 'comments' , 'add' ) ) )</code> . It can be a database table, in which case the plugin will use the default database connection through the table class. Or it can be an XML file.
	<i>\$sourceType</i>	optional, defaults to 'db'. It must be 'xml' if you use an XML file, or a <b>databaseAdapter</b> (p.20) object if you want to get the ACL table from a different database.

### 7.1.3 Member Function Documentation

#### 7.1.3.1 `addNew ( $ type, $ what, $ where = FALSE )`

Adds a new type of resource, action, requester or permission. Works only with database and XML based ACLs.

##### Parameters

string	<i>\$type</i>	It can be 'resource', 'action', 'requester' or 'permission'
mixed	<i>\$what</i>	For 'resource', 'action' and 'requester' it should be a string containing the name of the new element. For 'permission' it should be an associative array containing key-value pairs for each of the types. If no action has to be defined then it action should be NULL.
string	<i>\$where</i>	Used only for 'requester', it should the name of the node below which to insert. If it is not found, the new value is inserted as a new group.

##### Return values

<i>false</i>	It returns false if you are trying to use it without a database or XML file.
<i>true</i>	

#### 7.1.3.2 `checkConsistency ( )` [private]

This checks for the consistency of the ACL list.

##### Return values

<i>true</i>	
<i>false</i>	

#### 7.1.3.3 `deleteACL ( $ type, $ what )`

Delete something from the ACL. Parameters the same as for `addNew()` (p. 15), except the last parameter is not used again

##### Parameters

string	<i>\$type</i>	It can be 'resource', 'action', 'requester' or 'permission'
mixed	<i>\$what</i>	For 'resource', 'action' and 'requester' it should be a string containing the name of the new element. For 'permission' it should be an associative array containing key-value pairs for each of the types. If no action has to be defined then it action should be NULL.

**Return values**

<i>false</i>	It returns false if you are trying to use it without a database or XML file or if an element can't be found.
<i>true</i>	

**7.1.3.4 editACL ( \$ type, \$ what, \$ towhat, \$ where = FALSE )**

Edit the ACL. Parameters the same as for **addNew()** (p. 15), except an extra parameter.

**Parameters**

string	<i>\$type</i>	It can be 'resource', 'action', 'requester' or 'permission'
mixed	<i>\$what</i>	For 'resource', 'action' and 'requester' it should be a string containing the name of the new element. For 'permission' it should be an associative array containing key-value pairs for each of the types. If no action has to be defined then it action should be NULL.
string	<i>\$towhat</i>	What to change this to
string	<i>\$where</i>	Used only for 'requester', it should the name of the node after which to insert. If it is not found, the new value is inserted as a new group.

**Return values**

<i>false</i>	It returns false if you are trying to use it without a database or XML file or if an element can't be found.
<i>true</i>	

**7.1.3.5 getACL ( \$ type )**

Retrieves all requesters, resources, actions or permissions from the database or XML file.

**Parameters**

string	<i>\$type</i>	
--------	---------------	--

**Returns**

array

**7.1.3.6 getName ( )**

Get name.

**Returns**

string



Reimplemented from **plugin** (p. 34).

#### 7.1.3.7 `getPath ( $ array )` [`private`]

Recursively search for requester in acl list

##### Parameters

array	<i>\$array</i>	
-------	----------------	--

#### 7.1.3.8 `init ( $ prefix = ' ' , $ dbCon = FALSE )`

This creates the necessary tables for ACL. It creates a few default elements for everything. You may specify a prefix for the tables. You may optionally pass it a database connection. Else it will use the framework connection to the database.

##### Parameters

string	<i>\$prefix</i>	defaults to empty string "
<b>database</b> (p. 20)	<i>\$dbCon</i>	

#### 7.1.3.9 `isAllowed ( $ function )`

Verifies if requester has clearance for this.

##### Parameters

function	<i>\$function</i>	
----------	-------------------	--

##### Return values

<i>true</i>	
<i>false</i>	

#### 7.1.3.10 `run ( )`

Executes the plugin at an execution point. It takes the current Route from the router class.

### 7.1.4 Field Documentation

7.1.4.1 **\$acl** [private]

7.1.4.2 **\$denyFunction** [private]

7.1.4.3 **\$prefix = NULL** [private]

7.1.4.4 **\$requester** [private]

7.1.4.5 **\$source** [private]

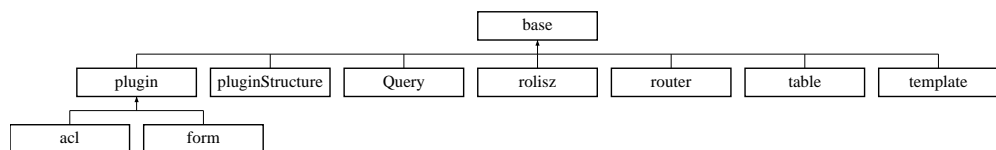
7.1.4.6 **\$sourceType** [private]

The documentation for this class was generated from the following file:

- C:/wamp/www/framework/plugins/acl.php

## 7.2 base Class Reference

Inheritance diagram for base:



### Static Public Member Functions

- static **get** (\$var)
- static **check** (\$var)
- static **set** (\$var, \$value=FALSE)

### Data Fields

- const **AppName** = 'rolisz PHP framework'
- const **Version** = '0.0.3'
- const **Module** = 'Base class'
- static **\$plugins**
- static **\$executionPoints**

### Static Protected Attributes

- static **\$global**

## Private Member Functions

- `__construct()`

### 7.2.1 Detailed Description

Base class for all the others in the framework. Defines some framework details, global variables and setters and getters for framework variables.

### 7.2.2 Constructor & Destructor Documentation

#### 7.2.2.1 `__construct()` [private]

Base constructor is private to disable creation of objects and to reinforce the usage of the singleton pattern.

Reimplemented in **plugin** (p. 33), **pluginStructure** (p. 35), and **template** (p. 55).

### 7.2.3 Member Function Documentation

#### 7.2.3.1 `static check($var)` [static]

Checks if \$var variable has been set in the framework.

##### Parameters

string	\$var	
--------	-------	--

##### Return values

<i>true</i>	
<i>false</i>	

#### 7.2.3.2 `static get($var)` [static]

Return the value of a framework variable or false if it was not found.

##### Parameters

string	\$var	
--------	-------	--

##### Return values

<i>true</i>	
<i>false</i>	

### 7.2.3.3 static set ( \$ var, \$ value = FALSE ) [static]

Set the value of a framework variable. If \$var param is a string, then a variable called \$var will have the value of \$value. If \$var is an array, it should be a key-pair value like this array ('var\_name'=>'132', '2ndvar'=>123)

#### Parameters

string   array	<i>\$var</i>	
mixed	<i>\$value</i>	

## 7.2.4 Field Documentation

### 7.2.4.1 \$executionPoints

### 7.2.4.2 \$global [static, protected]

### 7.2.4.3 \$plugins

### 7.2.4.4 const AppName = 'rolisz PHP framework'

### 7.2.4.5 const Module = 'Base class'

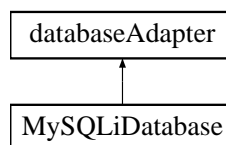
### 7.2.4.6 const Version = '0.0.3'

The documentation for this class was generated from the following file:

- C:/wamp/www/framework/base.php

## 7.3 databaseAdapter Class Reference

Inheritance diagram for databaseAdapter:



### Public Member Functions

- **\_\_construct** (\$host, \$username, \$password, \$db)
- **disconnect** ()
- **escapeValue** (\$value)
- **fetchRow** (\$query=false, \$type='assoc')

- **fetchAll** (\$query=false, \$type='assoc')
- **getError** ()
- **getInsertID** ()
- **numRows** ()
- **numAffected** ()
- **query** (\$query)
- **selectDatabase** (\$db)
- **startTransaction** ()
- **commit** ()
- **rollback** ()
- **tableExists** (\$table)
- **createTable** (\$name, \$params)
- **dropTable** (\$name)
- **alterTable** (\$name, \$params)

### 7.3.1 Detailed Description

Defines all the functions a database adapter class should have to be working with rolisz

### 7.3.2 Constructor & Destructor Documentation

#### 7.3.2.1 `__construct ( $ host, $ username, $ password, $ db )`

Reimplemented in **MySQLiDatabase** (p. 27).

### 7.3.3 Member Function Documentation

#### 7.3.3.1 `alterTable ( $ name, $ params )`

Reimplemented in **MySQLiDatabase** (p. 28).

#### 7.3.3.2 `commit ( )`

Reimplemented in **MySQLiDatabase** (p. 28).

#### 7.3.3.3 `createTable ( $ name, $ params )`

Reimplemented in **MySQLiDatabase** (p. 28).

#### 7.3.3.4 `disconnect ( )`

Reimplemented in **MySQLiDatabase** (p. 29).

**7.3.3.5 dropTable ( \$ name )**

Reimplemented in **MySQLiDatabase** (p. 29).

**7.3.3.6 escapeValue ( \$ value )**

Reimplemented in **MySQLiDatabase** (p. 29).

**7.3.3.7 fetchAll ( \$ query = false, \$ type = 'assoc' )**

Reimplemented in **MySQLiDatabase** (p. 29).

**7.3.3.8 fetchRow ( \$ query = false, \$ type = 'assoc' )**

Reimplemented in **MySQLiDatabase** (p. 30).

**7.3.3.9 getError ( )**

Reimplemented in **MySQLiDatabase** (p. 30).

**7.3.3.10 getInsertID ( )**

Reimplemented in **MySQLiDatabase** (p. 30).

**7.3.3.11 numAffected ( )**

Reimplemented in **MySQLiDatabase** (p. 31).

**7.3.3.12 numRows ( )**

Reimplemented in **MySQLiDatabase** (p. 31).

**7.3.3.13 query ( \$ query )****7.3.3.14 rollback ( )**

Reimplemented in **MySQLiDatabase** (p. 31).

**7.3.3.15 selectDatabase ( \$ db )**

Reimplemented in **MySQLiDatabase** (p. 31).

### 7.3.3.16 startTransaction ( )

Reimplemented in **MySQLiDatabase** (p. 32).

### 7.3.3.17 tableExists ( \$table )

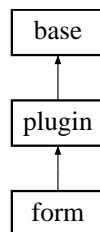
Reimplemented in **MySQLiDatabase** (p. 32).

The documentation for this class was generated from the following file:

- C:/wamp/www/framework/databaseAdapter.php

## 7.4 form Class Reference

Inheritance diagram for form:



### Public Member Functions

- **\_\_construct** (\$form=array(), \$model=FALSE)
- **input** (\$type, \$options)
- **\_\_call** (\$name, \$options)
- **removeInput** (\$name)
- **show** ()
- **getString** ()
- **validate** ()
- **getName** ()

### Static Public Member Functions

- static **getDefaultMethod** ()

### Private Member Functions

- **buildString** ()

## Private Attributes

- **\$inputs**
- **\$form**
- **\$string**
- **\$model = FALSE**

### 7.4.1 Detailed Description

Helper plugin to generate and validate forms.

### 7.4.2 Constructor & Destructor Documentation

#### 7.4.2.1 `__construct ( $ form = array() , $ model = FALSE )`

Initializes a new form.

#### Parameters

array	<i>\$form</i>	details about the form
array	<i>\$model</i>	optional, a table class initialized with a database table

### 7.4.3 Member Function Documentation

#### 7.4.3.1 `__call ( $ name, $ options )`

Magic method for adding new inputs by calling their type directly. Throws an errors if trying to call an inexistent input type.

#### Parameters

string	<i>\$name</i>	
array	<i>\$options</i>	

#### Returns

`$this`

#### 7.4.3.2 `buildString ( )` [private]

Builds the HTML string with the inputs wrapped in unordered list items.

#### 7.4.3.3 `static getDefaultMethod ( )` [static]

Default method is `getString()` (p. 25)

Reimplemented from `plugin` (p. 34).



#### 7.4.3.4 getName ( )

Return the plugin name

Reimplemented from **plugin** (p. 34).

#### 7.4.3.5 getString ( )

Returns a string containing the form.

##### Returns

string

#### 7.4.3.6 input ( \$ type, \$ options )

Adds an input element to the form. For checkboxes that have multiple values, the name is automatically suffixed with []

##### Parameters

string	<i>\$type</i>	
array	<i>\$options</i>	

##### Returns

\$this

#### 7.4.3.7 removeInput ( \$ name )

Remove an input from the form. Useful for the forms generated automatically from databases

##### Parameters

string	<i>\$name</i>	
--------	---------------	--

#### 7.4.3.8 show ( )

Outputs the form

#### 7.4.3.9 validate ( )

Validates this form

##### Return values

<i>true</i>	
-------------	--

<i>false</i>
--------------

#### 7.4.4 Field Documentation

7.4.4.1 **\$form** [private]

7.4.4.2 **\$inputs** [private]

7.4.4.3 **\$model = FALSE** [private]

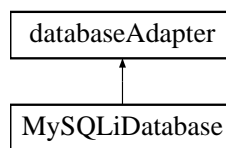
7.4.4.4 **\$string** [private]

The documentation for this class was generated from the following file:

- C:/wamp/www/framework/plugins/**form.php**

### 7.5 MySQLiDatabase Class Reference

Inheritance diagram for MySQLiDatabase:



#### Public Member Functions

- **\_\_construct** (\$host, \$username, \$password, \$db)
- **disconnect** ()
- **\_\_destruct** ()
- **selectDatabase** (\$db)
- **startTransaction** ()
- **commit** ()
- **rollback** ()
- **escapeValue** (\$value)
- **Query** (\$query)
- **fetchFirst** (\$query=false)
- **fetchRow** (\$query=false, \$type=1)
- **fetchAll** (\$query=false, \$type=1)
- **getError** ()
- **getInsertID** ()
- **numRows** ()

- **numAffected** ()
- **tableExists** (\$table)
- **createTable** (\$name, \$params)
- **dropTable** (\$name)
- **alterTable** (\$name, \$params)

### Data Fields

- **\$connection**
- **\$result**
- **\$database**
- **\$queries**

### Private Member Functions

- **connect** (\$host, \$username, \$password)

#### 7.5.1 Detailed Description

MySQLi specific implementation of **databaseAdapter** (p. 20)

#### 7.5.2 Constructor & Destructor Documentation

##### 7.5.2.1 `__construct ( $ host, $ username, $ password, $ db )`

Class constructor, initializes connection to MySQL database. Not implemented as singleton pattern because you can have multiple objects, each with a connection to a different database.

#### Parameters

string	<i>\$host</i>	
string	<i>\$username</i>	
string	<i>\$password</i>	
string	<i>\$db</i>	

Reimplemented from **databaseAdapter** (p. 21).

##### 7.5.2.2 `__destruct ( )`

Class destructor. Disconnects from the database.

### 7.5.3 Member Function Documentation

#### 7.5.3.1 alterTable ( \$ name, \$ params )

Alters \$name table. To change it's name by \$params must contain an element with key 'name' and value the new name of the table in params. To add a column, \$params must contain an element with key 'add' and value an array of columns you wish to add (specified as at createTable). To modify a column, \$params must contain an element with key 'edit' and value an array of columns you wish to edit (specified as at createTable). To drop a column, \$params must contain an element with key 'delete' and value the name of the column you want to delete.

##### Parameters

string	<i>\$name</i>	
array	<i>\$params</i>	

##### Return values

<i>true</i>	
<i>false</i>	

##### Todo

some checks for valid column names and parameters

Reimplemented from **databaseAdapter** (p. 21).

#### 7.5.3.2 commit ( )

Commits the last transaction

Reimplemented from **databaseAdapter** (p. 21).

#### 7.5.3.3 connect ( \$ host, \$ username, \$ password ) [private]

#### 7.5.3.4 createTable ( \$ name, \$ params )

Creates a table.

##### Parameters

string	<i>\$name</i>	Has to match this regex [A-Za-z0-9_.-]*
array	<i>\$params</i>	Example: <code>array('id'=&gt;array('INT','NOT NULL','AUTO_INCREMENT','PRIMARY KEY'),'text'=&gt;array('TEXT','NOT NULL'))</code>

##### Return values

<i>true</i>	
<i>false</i>	

Reimplemented from **databaseAdapter** (p. 21).

#### 7.5.3.5 disconnect ( )

Disconnects from the database

Reimplemented from **databaseAdapter** (p. 21).

#### 7.5.3.6 dropTable ( \$ name )

It drops \$name table if it exists. If it doesn't exist it returns true (you wanted it gone, right?).

##### Parameters

string	<i>\$name</i>	
--------	---------------	--

##### Return values

<i>true</i>	
<i>false</i>	

Reimplemented from **databaseAdapter** (p. 22).

#### 7.5.3.7 escapeValue ( \$ value )

Escapes a string for safe MySQL insertion

##### Parameters

string	<i>\$value</i>	
--------	----------------	--

##### Returns

string

Reimplemented from **databaseAdapter** (p. 22).

#### 7.5.3.8 fetchAll ( \$ query = false, \$ type = 1 )

Fetches all the rows from the results of a query. Returns numeric array, associative array or both depending on second parameter:1,2,3

##### Parameters

string	<i>\$query</i>	
int	<i>\$type</i>	1 - Associative array, 2 - Numeric array, 3 - Both

**Returns**

mixed

Reimplemented from **databaseAdapter** (p. 22).

**7.5.3.9 fetchFirst ( \$ query = false )**

Fetches the first value from the first row of the result of a query.

**Parameters**

string	<i>\$query</i>	
--------	----------------	--

**Returns**

mixed

**7.5.3.10 fetchRow ( \$ query = false, \$ type = 1 )**

Fetches first row from the results of a query. Returns numeric array, associative array or both depending on second parameter.

**Parameters**

string	<i>\$query</i>	
int	<i>\$type</i>	1 - Associative array, 2 - Numeric array, 3 - Both

**Returns**

mixed

Reimplemented from **databaseAdapter** (p. 22).

**7.5.3.11 getError ( )**

Return the last MySQLi error

**Returns**

string

Reimplemented from **databaseAdapter** (p. 22).

**7.5.3.12 getInsertID ( )**

Returns the id of the last insertion

**Returns**

int

Reimplemented from **databaseAdapter** (p. 22).

#### 7.5.3.13 numAffected ( )

Returns the number of rows affected by an UPDATE query

##### Returns

int

Reimplemented from **databaseAdapter** (p. 22).

#### 7.5.3.14 numRows ( )

Returns the number of rows a query returned

##### Returns

int

Reimplemented from **databaseAdapter** (p. 22).

#### 7.5.3.15 Query ( \$ query )

Executes a query

##### Parameters

string	<i>\$query</i>	
--------	----------------	--

##### Returns

mixed

##### Todo

change return values for insert, update, delete, select

#### 7.5.3.16 rollback ( )

Rolls back last transaction

Reimplemented from **databaseAdapter** (p. 22).

#### 7.5.3.17 selectDatabase ( \$ db )

Connects to a different database

**Parameters**

string	<i>\$db</i>	
--------	-------------	--

**Return values**

<i>true</i>	
<i>false</i>	

Reimplemented from **databaseAdapter** (p. 22).

**7.5.3.18 startTransaction ( )**

Starts a new transaction

Reimplemented from **databaseAdapter** (p. 23).

**7.5.3.19 tableExists ( \$ table )**

Checks if a table exists in the current database

**Parameters**

string	<i>\$table</i>	
--------	----------------	--

**Return values**

<i>true</i>	
<i>false</i>	

Reimplemented from **databaseAdapter** (p. 23).

**7.5.4 Field Documentation****7.5.4.1 MySQLi object \$connection**

Stores the MySQLi connection

**7.5.4.2 string \$database**

Stores the working database

**7.5.4.3 array \$queries**

Stores all the queries that have been executed



#### 7.5.4.4 array \$result

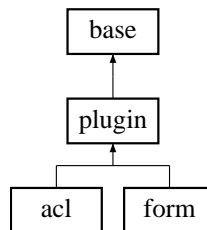
Stores the latest result from a query

The documentation for this class was generated from the following file:

- C:/wamp/www/framework/databaseAdapter.php

## 7.6 plugin Class Reference

Inheritance diagram for plugin:



### Public Member Functions

- `__construct ()`
- `run ($arg)`
- `getName ()`

### Static Public Member Functions

- static `getDefaultExecutionPoints ()`
- static `getDefaultMethod ()`

#### 7.6.1 Detailed Description

Plugin class. This is inherited for all the other plugins. Not an interface in case you don't want to redeclare some of the functions in other plugins

#### 7.6.2 Constructor & Destructor Documentation

##### 7.6.2.1 \_\_construct ( )

Constructor

Reimplemented from **base** (p. 19).

### 7.6.3 Member Function Documentation

#### 7.6.3.1 static getDefaultExecutionPoints ( ) [static]

Returns the default excution points. By default it doesn't have any. The following execution points are available inside the rolisz framework:

- beforeMatch fires in the router module before trying to match the current URL.
- afterMatch fires in the router module after it has succesfully matched a URL. If no URL has been matched, it won't fire.
- hydrateTable fires in the table class if an object has been hydrated from the database.
- saveTable fires in the table class if an object has been successfully saved to the database.
- deleteTable fires in the table class if an object has been successfully deleted from the database.
- compileTemplate fires in the template engine after compiling a template.

#### Return values

<i>false</i>	
<i>string</i>	
<i>array</i>	

#### 7.6.3.2 static getDefaultMethod ( ) [static]

Returns name of the the default function to run at an execution point. Default is 'run'

#### Returns

string

Reimplemented in **form** (p. 24).

#### 7.6.3.3 getName ( )

Returns the name of the plugin

#### Return values

<i>string</i>	
---------------	--

Reimplemented in **acl** (p. 16), and **form** (p. 25).

#### 7.6.3.4 run ( \$arg )

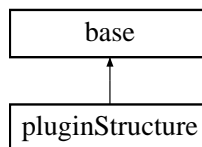
Default function run at an execution point

The documentation for this class was generated from the following file:

- C:/wamp/www/framework/plugin.php

## 7.7 pluginStructure Class Reference

Inheritance diagram for pluginStructure:



### Public Member Functions

- **\_\_construct** ()
- **registerExecutionPoint** (\$name)
- **unregisterExecutionPoint** (\$name)

### Static Public Member Functions

- static **registerPlugin** (\$name, \$execution=FALSE)
- static **unregisterPlugin** (\$name)
- static **checkPlugin** (\$name)
- static **checkExecutionPoint** (\$name)
- static **runPlugins** (\$name)

#### 7.7.1 Detailed Description

Class that handles registering plugins and execution points.

#### 7.7.2 Constructor & Destructor Documentation

##### 7.7.2.1 \_\_construct ( )

Base constructor is private to disable creation of objects and to reinforce the usage of the singleton pattern.

Reimplemented from **base** (p. 19).

### 7.7.3 Member Function Documentation

#### 7.7.3.1 static checkExecutionPoint ( \$ name ) [static]

Checks if an execution point exists.

##### Parameters

string	<i>\$name</i>	
--------	---------------	--

##### Return values

<i>true</i>	
<i>false</i>	

#### 7.7.3.2 static checkPlugin ( \$ name ) [static]

Checks if a plugin called \$name exists.

##### Parameters

string	<i>\$name</i>	
--------	---------------	--

##### Return values

<i>true</i>	
<i>false</i>	

#### 7.7.3.3 registerExecutionPoint ( \$ name )

Adds a new execution point to the internal execution point list.

##### Parameters

string	<i>\$name</i>	
--------	---------------	--

#### 7.7.3.4 static registerPlugin ( \$ name, \$ execution = FALSE ) [static]

Registers a plugin. If the plugin constructor requires parameters, you have to instantiate an object and pass that as a parameter to this function.

##### Parameters

<i>\$name</i>	- string <ul style="list-style-type: none"> <li>• class</li> <li>• file</li> </ul>
---------------	------------------------------------------------------------------------------------

- plugin object
- array with a class and the function to call from it

**Parameters**

<i>\$execution</i>	optional and specifies when to execute the plugin if it has to be executed sometime during the execution of rolisz classes. For a list of events, see plugin() class.
--------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------

**7.7.3.5 static runPlugins ( \$ name ) [static]**

Runs plugins and functions that are associated with the \$name execution point. Any additional parameters will be passed to the function in the same order as they are received.

**Parameters**

string	<i>\$name</i>	
--------	---------------	--

**7.7.3.6 unregisterExecutionPoint ( \$ name )**

Removes an execution point from the internal execution point list. Attention, you can remove rolisz's own execution points, after which it may no longer work as expected.

**Parameters**

string	<i>\$name</i>	
--------	---------------	--

**7.7.3.7 static unregisterPlugin ( \$ name ) [static]**

Unregisters a plugin.

**Parameters**

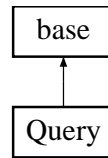
string	<i>\$name</i>	name of the plugin, as returned by the <b>plugin::getName()</b> (p. 34) method.
--------	---------------	---------------------------------------------------------------------------------

The documentation for this class was generated from the following file:

- C:/wamp/www/framework/pluginStructure.php

**7.8 Query Class Reference**

Inheritance diagram for Query:



### Public Member Functions

- **\_\_construct** (\$class, \$filters, \$ordergroup, \$columns)
- **buildQuery** ()
- **getCount** ()

### Private Member Functions

- **buildFilters** (\$property, \$value, \$class)
- **buildOrderBy** ()
- **buildJoins** (\$child, \$parent)

### 7.8.1 Detailed Description

/class **Query** (p. 37) Helper class for making the more complicated join queries. Should not be called directly, only through the find() function

### 7.8.2 Constructor & Destructor Documentation

#### 7.8.2.1 \_\_construct ( \$ class, \$ filters, \$ ordergroup, \$ columns )

Constructor.

#### Parameters

string	<i>\$class</i>	
array	<i>\$filters</i>	
array	<i>\$ordergroup</i>	
array	<i>\$columns</i>	

### 7.8.3 Member Function Documentation

#### 7.8.3.1 buildFilters ( \$ property, \$ value, \$ class ) [private]

Processes the filters passed in the constructor and returns appropriate SQL queries

#### Parameters

mixed	<i>\$property</i>	
mixed	<i>\$value</i>	

table	<i>\$class</i>	
-------	----------------	--

**Returns**

string

**7.8.3.2 buildJoins ( \$ child, \$ parent ) [private]**

Makes the SQL strings for the JOINS necessary in the filtering

**Parameters**

table	<i>\$child</i>	
table	<i>\$parent</i>	

**7.8.3.3 buildOrderBy ( ) [private]**

Build the order, group, limit SQL strings for the query. They are put in variables inside the class.

**7.8.3.4 buildQuery ( )**

Simply joins and returns all the stuff done in the previous functions

**Returns**

string with SQL statement

**7.8.3.5 getCount ( )**

Get's a count for how many rows would a query return

**Returns**

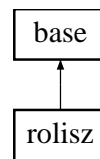
int;

The documentation for this class was generated from the following file:

- C:/wamp/www/framework/**table.php**

## 7.9 rolisz Class Reference

Inheritance diagram for rolisz:



## Static Public Member Functions

- static **call** (\$funcs, \$args)
- static **connect** (\$dbtype, \$host, \$username, \$password, \$db)
- static **autoload** (\$class)
- static **http404** ()
- static **fixSlashes** (\$str)
- static **fixBraces** (\$str)
- static **plugin** ()
- static **runPlugins** (\$name)
- static **route** (\$pattern, \$funcs, \$http= 'GET', \$name=FALSE)
- static **run** ()
- static **urlFor** (\$name, \$params=array())
- static **table** (\$table, \$id=FALSE, \$columns=FALSE, \$connection=FALSE)
- static **start** ()

### 7.9.1 Detailed Description

Central class, contains only the most common functions or wrappers for the other classes

### 7.9.2 Member Function Documentation

#### 7.9.2.1 static autoload ( \$ class ) [static]

Autoloader function. Lazy-loads classes from files first from the framework directory, then the framework plugin directory, then the current one and finally a plugins folder situated in the current folder. Files must be named the same the classes

#### Parameters

string	<i>\$class</i>	
--------	----------------	--

#### 7.9.2.2 static call ( \$ funcs, \$ args ) [static]

Calls functions and includes files and passes them \$args as argument

#### Parameters

mixed	<i>\$funcs</i>	
-------	----------------	--



array	<i>\$args</i>	
-------	---------------	--

#### 7.9.2.3 static connect ( *\$dbtype*, *\$host*, *\$username*, *\$password*, *\$db* ) [static]

Connects to a database adapter that implements the interface defined in **databaseAdapter.php** (p. 59) Checks for the existence of the class in **databaseAdapter.php** (p. 59) and then in a file called *\$dbtype.DatabaseAdapter.php*

##### Parameters

string	<i>\$dbtype</i>	Database type. For now there is support for MySQL
string	<i>\$host</i>	
string	<i>\$username</i>	
string	<i>\$password</i>	
string	<i>\$db</i>	

##### Returns

**databaseAdapter** (p. 20)

#### 7.9.2.4 static fixBraces ( *\$str* ) [static]

Fix mangled braces

##### Parameters

string	<i>\$str</i>	
--------	--------------	--

##### Returns

string

#### 7.9.2.5 static fixSlashes ( *\$str* ) [static]

Convert Windows double-backslashes to slashes

##### Parameters

string	<i>\$str</i>	
--------	--------------	--

##### Returns

string

**7.9.2.6 static http404 ( )** [static]

Trigger an HTTP 404 error. If the 404 framework variable has been set to a function, it will get called. Else, it just triggers an error saying the page was not found.

**7.9.2.7 static plugin ( )** [static]

Plugin functions Return an instance of **pluginStructure** (p. 35)

**Returns**

**pluginStructure** (p. 35)

**7.9.2.8 static route ( \$ pattern, \$ funcs, \$ http = 'GET', \$ name = FALSE )** [static]

Wrappers for the router class

**See also**

**router::route** (p. 44)

**7.9.2.9 static run ( )** [static]**See also**

**router::run** (p. 44)

**7.9.2.10 static runPlugins ( \$ name )** [static]

Runs plugins associated with \$name execution point. Additional parameters will be passed to plugin as parameters

**Parameters**

string	<i>\$name</i>	
--------	---------------	--

**7.9.2.11 static start ( )** [static]

Sets up autoload, initializes some constants and starts session.

**7.9.2.12 static table ( \$ table, \$ id = FALSE, \$ columns = FALSE, \$ connection = FALSE )** [static]

Wrappers for tables Returns a new instance of table class. For parameters

**See also**

**table** (p. 45)

**Returns**

table

**7.9.2.13** `static urlFor ( $ name, $ params = array() ) [static]`

**See also**

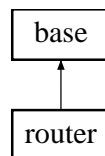
**router::urlFor** (p. 44)

The documentation for this class was generated from the following file:

- C:/wamp/www/framework/**rolisz.php**

## 7.10 router Class Reference

Inheritance diagram for router:

**Static Public Member Functions**

- static **route** (\$pattern, \$funcs, \$http= 'GET', \$name=FALSE)
- static **run** ()
- static **urlFor** (\$name, \$params=array())

**Static Private Member Functions**

- static **matches** (\$route)

### 7.10.1 Detailed Description

This class provides URL routing, pretty URLs and named URLs functionality.

## 7.10.2 Member Function Documentation

### 7.10.2.1 static matches ( \$ route ) [static, private]

Checks if \$route matches the current URL

#### Parameters

string	<i>\$route</i>	
--------	----------------	--

#### Return values

	<i>true</i>	
	<i>false</i>	

### 7.10.2.2 static route ( \$ pattern, \$ funcs, \$ http = ' GET ' , \$ name = FALSE ) [static]

Set a new route pattern.

#### Parameters

string   array	<i>\$pattern</i>	defines the route. It can contain variables like <code>/:var/</code> and catch-alls at the end: <code>url/*</code> . If it's an array of routes (that all lead to the same thing), the \$name parameter will be ignored.
mixed	<i>\$funcs</i>	defines the functions that can be passed as arguments. The functions can given as a list names separated by <code> </code> , an array consisting of things that return true to <code>is_callable()</code> . If the array contains things that return false, it will trigger an error and the element will be unset. Also, you can just pass an anonymous function. If you give it as a string, you can also pass files.
string	<i>\$http</i>	optional, defaults to GET, and is the HTTP method for which this route is valid. It can be GET or POST. If you want the route to be valid for AJAX calls only, use AGET or APOST.
string	<i>\$name</i>	optional, defaults to FALSE and gives the name of the route, to be used with <code>urlFor()</code> (p. 44).

### 7.10.2.3 static run ( ) [static]

Process routes based on the incoming URI. If no match is found, a 404 error is triggered. `beforeMatch` and `afterMatch` plugins are called here. Throttles outputting the script until `rolisz::get` (p. 19)('THROTTLE') time has passed.

### 7.10.2.4 static urlFor ( \$ name, \$ params = array() ) [static]

Returns a URL with values filled in for a given routing pattern.

#### Parameters

string	<i>\$name</i>	name of the URL pattern, as given in <b>route()</b> (p. 44) function
array	<i>\$params</i>	has to be key-value pairs for each parameter in the URL pattern.

**Returns**

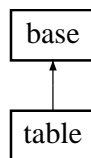
string

The documentation for this class was generated from the following file:

- C:/wamp/www/framework/router.php

## 7.11 table Class Reference

Inheritance diagram for table:

**Public Member Functions**

- **\_\_construct** (\$table, \$id=FALSE, \$columns=FALSE, \$connection=FALSE)
- **\_\_set** (\$name, \$value)
- **\_\_get** (\$name)
- **\_\_isset** (\$name)
- **\_\_sleep** ()
- **\_\_call** (\$name, \$args)
- **getPrimaryKey** (\$value=FALSE)
- **getData** (\$object=FALSE)
- **isHydrated** ()
- **save** ()
- **delete** ()
- **find** (\$table, \$filters=array(), \$ordergroup=array(), \$columns=array(), \$import=TRUE)
- **countRows** (\$table, \$filters=array(), \$ordergroup=array())
- **addRelation** (\$tablename, \$arg1, \$arg2=FALSE)
- **addRelationM2M** (\$connectortable, \$thisid, \$mappedid, \$connectedtable, \$thatid, \$cmappedid)
- **setAll** (\$values)

### Static Public Member Functions

- static **importRows** (\$table, \$values)
- static **set** (\$table, \$id=FALSE, \$columns=FALSE, \$connection=FALSE)
- static **addRelationM2MS** (\$table, \$connectortable, \$thisid, \$mappedid, \$connectedtable, \$thatid, \$cmappedid)
- static **addRelationS** (\$table, \$tablename, \$arg1, \$arg2=FALSE)
- static **findS** (\$table, \$filters=array(), \$ordergroup=array(), \$columns=array())
- static **countRowsS** (\$table, \$filters=array(), \$ordergroup=array())

### Data Fields

- **\$table**

### Static Public Attributes

- static **\$tables** = array()
- static **\$relations** = array()

### Private Member Functions

- **hydrate** (\$id)
- **getColumns** ()
- **analyzeRelations** ()

### Private Attributes

- **\$connection**
- **\$originalData** = array()
- **\$modifiedData** = array()

### Static Private Attributes

- static **\$primaryKey** = array()

#### 7.11.1 Detailed Description

Provides CRUD and ORM functionality. Detects automatically all the columns of a table. If you have a table that gets called a lot you should extend this table and set the tables columns in your code, to save a database query that gets the columns of this table.

### 7.11.2 Constructor & Destructor Documentation

#### 7.11.2.1 `__construct ( $ table, $ id = FALSE, $ columns = FALSE, $ connection = FALSE )`

Initializes the table.

##### Parameters

string	<i>\$table</i>	
int	<i>\$id</i>	optional
array	<i>\$columns</i>	- primary key is defined like this: 'PRI'=>array('id'=>'int'). Optional. Defaults to all columns
<b>database</b> (p. 20)	<i>\$connection,</i>	Optional. Defaults to global connection in rolisz.

### 7.11.3 Member Function Documentation

#### 7.11.3.1 `__call ( $ name, $ args )`

Magic method for setting or getting related tables. If function is called without argument, it will find the related tables. It will automatically save the connection!

##### Parameters

string	<i>\$name</i>	
array	<i>\$args</i>	

##### Todo

\$args could be a table of \$name

#### 7.11.3.2 `__get ( $ name )`

Magic method for accesing dynamic properties

##### Parameters

string	<i>\$name</i>	
--------	---------------	--

##### Returns

mixed

#### 7.11.3.3 `__isset ( $ name )`

Magic method that checks if a column exists

##### Parameters

string	<i>\$name</i>	
--------	---------------	--

**Return values**

<i>true</i>	
<i>false</i>	

**7.11.3.4 \_\_set ( \$ name, \$ value )**

Magic method for setting dynamic properties

**Parameters**

string	<i>\$name</i>	
string	<i>\$value</i>	

**7.11.3.5 \_\_sleep ( )**

Magic method for serialization

**7.11.3.6 addRelation ( \$ tablename, \$ arg1, \$ arg2 = FALSE )**

Adds a relation to another table. This is for many-to-one or one-to-one relations. If third argument isn't passed, it defaults to the table primary key. Returns the table we are working on, to allow chaining of methods.

**Parameters**

string	<i>\$tablename</i>	
string	<i>\$arg1</i>	
string	<i>\$arg2</i>	

**Returns**

*\$this*

**7.11.3.7 addRelationM2M ( \$ connectortable, \$ thisid, \$ mappedid, \$ connectedtable, \$ thatid, \$ cmappedid )**

Adds a many-to-many relation between tables. Returns the table we are working on, to allow chaining of methods.

**Parameters**

string	<i>\$connectortable</i>	
string	<i>\$thisid</i>	
string	<i>\$mappedid</i>	
string	<i>\$connectedtable</i>	



string	<i>\$thatid</i>	
string	<i>\$mappedid</i>	

**Todo**

find a way to reduce arguments

**Returns**

\$this

**7.11.3.8** `static addRelationM2MS ( $ table, $ connectortable, $ thisid, $ mappedid, $ connectedtable, $ thatid, $ cmappedid )` [static]

Adds a Many-To-Many relationship statically. See **addRelationM2M()** (p. 48).

**Parameters**

string	<i>\$table</i>	The table from which to start the relation
string	<i>\$connectortable</i>	
string	<i>\$thisid</i>	
string	<i>\$mappedid</i>	
string	<i>\$connectedtable</i>	
string	<i>\$thatid</i>	
string	<i>\$cmappedid</i>	

**7.11.3.9** `static addRelationS ( $ table, $ tablename, $ arg1, $ arg2 = FALSE )` [static]

Adds a relation to another table statically. See **addRelation()** (p. 48).

**Parameters**

string	<i>\$table</i>	The table from which to start the relation
string	<i>\$tablename</i>	
string	<i>\$arg1</i>	
string	<i>\$arg2</i>	

**7.11.3.10** `analyzeRelations ( )` [private]

Analyzes the relation between tables and marks them accordingly. So far only ONE TO MANY and MANY TO MANY ones.

### 7.11.3.11 `countRows ( $ table, $ filters = array(), $ ordergroup = array() )`

Returns the number of rows that result from a SELECT query. Parameters are the same as for **find()** (p. 50), except that you can't return only some columns. Also, ordering is not included in the query.

#### Parameters

string	<i>\$table</i>	Table in which to search
array	<i>\$filters</i>	Filters to apply. Can be name=>value pair, SQL statement, or recursive array relating to conditions on other tables
array	<i>\$ordergroup</i>	How to order, group and limit the search

#### Returns

int

### 7.11.3.12 `static countRowsS ( $ table, $ filters = array(), $ ordergroup = array() )` [static]

Static wrapper for **countRows()** (p. 50). Arguments the same.

#### Parameters

string	<i>\$table</i>	Table in which to search
array	<i>\$filters</i>	Filters to apply. Can be name=>value pair, SQL statement, or recursive array relating to conditions on other tables
array	<i>\$ordergroup</i>	How to order, group and limit the search

#### Returns

int

### 7.11.3.13 `delete ( )`

Delete the current record from the database

### 7.11.3.14 `find ( $ table, $ filters = array(), $ ordergroup = array(), $ columns = array(), $ import = TRUE )`

Makes and executes a SQL query, based on various filters, orders, groups, and columns to return.

#### Parameters

string	<i>\$table</i>	Table in which to search
array	<i>\$filters</i>	Filters to apply. Can be name=>value pair, SQL statement, or recursive array relating to conditions on other tables
array	<i>\$ordergroup</i>	How to order, group and limit the search

array	<i>\$columns</i>	What columns to return in search. You can pass key value pairs to retrieve values from other tables such as 'relatedTable'=>'column'
bool	<i>\$import</i>	- whether to import the resulting stuff into tables or not. Default is true. Set to false if you use the \$columns parameter to retrieve values from other tables.

**Returns**

array

**7.11.3.15** `static findS ( $ table, $ filters = array() , $ ordergroup = array() , $ columns = array() )` [static]

Static wrapper for **find()** (p. 50). Arguments the same.

**Parameters**

string	<i>\$table</i>	Table in which to search
array	<i>\$filters</i>	Filters to apply. Can be name=>value pair, SQL statement, or recursive array relating to conditions on other tables
array	<i>\$ordergroup</i>	How to order, group and limit the search
array	<i>\$columns</i>	What columns to return in search

**Returns**

array

**7.11.3.16** `getColumns ( )` [private]

Populates the \$data property of the object with the columns of the table and gets the primary key of the table, if it exists

**7.11.3.17** `getData ( $ object = FALSE )`

Returns the values of the current row as a key=>value array. If the \$object parameter is set to true, it returns an object with it's properties having the role of key-value pairs. If it hasn't been hydrated, it returns false.

**Parameters**

boolean	<i>\$object</i>	
---------	-----------------	--

**Returns**

mixed

**7.11.3.18** `getPrimaryKey ( $ value = FALSE )`

Returns the primary key of this table. If \$value is TRUE, then it returns the value of the primary key for this row

**Parameters**

bool	<i>\$value</i>	
------	----------------	--

**Return values**

<i>int</i>	Value of the primary key
<i>string</i>	The name of the primary key

**7.11.3.19** `hydrate ( $ id )` [private]

Populate the object with data from the table, selected according to primary key value

**Parameters**

int	<i>\$id</i>	
-----	-------------	--

**Returns**

array

**7.11.3.20** `static importRows ( $ table, $ values )` [static]

Imports an array of rows and returns an array of table objects filled with values. If there is only one row in \$values, it returns it as an object instead of as an array.

**Parameters**

string	<i>\$table</i>	
array	<i>\$values</i>	

**Returns**

array|object

**7.11.3.21** `isHydrated ( )`

Returns whether or not this object has been hydrated (did it load data from a table). It also returns true if it has already been saved to the database.

**Return values**

<i>true</i>	
<i>false</i>	

7.11.3.22 `save ( )`

Save changes made to object. If the object was hydrated, it will be updated. If it was created from scratch, it will be inserted into the database

**Return values**

<i>true</i>	If modifying was succesfull
<i>int</i>	If a row was inserted, it returns the insert ID
<i>false</i>	If there was an error

7.11.3.23 `static set ( $ table, $ id = FALSE, $ columns = FALSE, $ connection = FALSE )`  
[static]

Static convenience wrapper functions Initializes a certain table

**Parameters**

string	<i>\$table</i>	
int	<i>\$id</i>	
array	<i>\$columns</i>	
<b>database</b> (p. 20)	<i>\$connection</i>	

7.11.3.24 `setAll ( $ values )`

Set all the values of row at once. It doesn't save the row to the database.

**Parameters**

array	<i>\$values</i>	
-------	-----------------	--

## 7.11.4 Field Documentation

7.11.4.1 `$connection` [private]7.11.4.2 `$modifiedData = array()` [private]7.11.4.3 `$originalData = array()` [private]7.11.4.4 `$primaryKey = array()` [static, private]7.11.4.5 `$relations = array()` [static]

Static variable containing the relations between tables

#### 7.11.4.6 `$table`

Contains the table name of this object

#### 7.11.4.7 `$tables = array()` [`static`]

Static variable containing the columns of all tables that have been instantiated

#### Todo

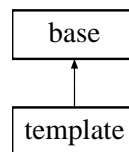
allow objects representing same table, but different columns

The documentation for this class was generated from the following file:

- `C:/wamp/www/framework/table.php`

## 7.12 `template` Class Reference

Inheritance diagram for `template`:



### Public Member Functions

- `__construct()`
- `__set($var, $value=FALSE)`
- `assign($var, $value=FALSE)`
- `__get($name)`
- `__toString()`
- `view($tpl)`
- `getOutput($template)`
- `includeSub($template)`

### Private Member Functions

- `checkCompile($template)`
- `compile($template)`

### Static Private Attributes

- static `$values`

### 7.12.1 Detailed Description

Provides basic templating capabilities

### 7.12.2 Constructor & Destructor Documentation

#### 7.12.2.1 `__construct ( )`

Empty constructor. It's created separately to prevent inheriting the singleton pattern from base.

Reimplemented from **base** (p. 19).

### 7.12.3 Member Function Documentation

#### 7.12.3.1 `__get ( $ name )`

Return value of template variable, false if not found. This is separate from the rolisz global variables.

##### Parameters

string	<i>\$name</i>	
--------	---------------	--

##### Return values

<i>true</i>	
<i>false</i>	

#### 7.12.3.2 `__set ( $ var, $ value = FALSE )`

Set the value of a template variable magically. If \$var param is string, then a variable called \$var will have the value of \$value. If \$var is array, it should be a key-pair value like this array('var\_name'=>'132','2ndvar'=>123). This is separate from the rolisz global variables.

##### Parameters

string   array	<i>\$var</i>	
mixed	<i>\$value</i>	

#### 7.12.3.3 `__toString ( )`

Magic method for the serialization of the template object.

**Returns**

string containing the template

**7.12.3.4 assign ( \$ var, \$ value = FALSE )**

Set the value of a template variable. If \$var param is string, then a variable called \$var will have the value of \$value. If \$var is array, it should be a key-pair value like this array('var\_name'=>'132','2ndvar'=>123). This is separate from the rolisz global variables.

**Parameters**

string   array	<i>\$var</i>	
mixed	<i>\$value</i>	

**7.12.3.5 checkCompile ( \$ template ) [private]**

Checks if the file has been compiled to the temp folder and if it is more recent than the last change to the template

**Parameters**

string	<i>\$template</i>	
--------	-------------------	--

**Return values**

<i>true</i>	
<i>false</i>	

**7.12.3.6 compile ( \$ template ) [private]**

Performs the replacing of the shorthand tags to full PHP tags in the \$template file. Places the results in a temp folder, in a file called the md5 value of \$template, to prevent collisions.

**Parameters**

string	<i>\$template</i>	
--------	-------------------	--

**7.12.3.7 getOutput ( \$ template )**

Compiles, executes, and filters a template source.

**Parameters**

string	<i>\$template</i>	The template to process;
--------	-------------------	--------------------------



**Returns**

mixed The template output string

**7.12.3.8 includeSub ( \$ template )**

Includes another template

**Parameters**

string	<i>\$template</i>	
--------	-------------------	--

**7.12.3.9 view ( \$ tpl )**

Prints the \$tpl template after compiling

**Parameters**

string	<i>\$tpl</i>	
--------	--------------	--

**7.12.4 Field Documentation****7.12.4.1 \$values** [static, private]

The documentation for this class was generated from the following file:

- C:/wamp/www/framework/**template.php**



## Chapter 8

# File Documentation

### 8.1 C:/wamp/www/framework/base.php File Reference

#### Data Structures

- class **base**

#### Namespaces

- namespace **rolisz**

### 8.2 C:/wamp/www/framework/databaseAdapter.php File Reference

#### Data Structures

- class **databaseAdapter**
- class **MySQLiDatabase**

#### Namespaces

- namespace **rolisz**

### 8.3 C:/wamp/www/framework/misc\_functions.php File Reference

#### Functions

- static **recursify** (\$array, \$end= "")
- static **array\_filter\_key** (\$input, \$callback)

### 8.3.1 Function Documentation

#### 8.3.1.1 static array\_filter\_key ( \$ input, \$ callback ) [static]

Filter an array by keys. Arguments similar to array\_filter.

##### Parameters

array	<i>\$input</i>	
function	<i>\$callback</i>	

##### Returns

array

#### 8.3.1.2 static recursify ( \$ array, \$ end = ' ' ) [static]

Turn linear array into a recursively nested array, with optional argument to be the the value at the end

ex: array('1','2','3') turns into array('1'=>array('2'=>array('3'=>"))

##### Parameters

array	<i>\$array</i>	
mixed	<i>\$end</i>	

##### Returns

array

## 8.4 C:/wamp/www/framework/plugin.php File Reference

### Data Structures

- class **plugin**

### Namespaces

- namespace **rolisz**

## 8.5 C:/wamp/www/framework/plugins/acl.php File Reference

### Data Structures

- class **acl**

**Namespaces**

- namespace **rolisz**

**8.6 C:/wamp/www/framework/plugins/form.php File Reference****Data Structures**

- class **form**

**Namespaces**

- namespace **rolisz**

**8.7 C:/wamp/www/framework/pluginStructure.php File Reference****Data Structures**

- class **pluginStructure**

**Namespaces**

- namespace **rolisz**
- namespace **Roland**

**8.8 C:/wamp/www/framework/rolisz.php File Reference****Data Structures**

- class **rolisz**

**Namespaces**

- namespace **rolisz**

**8.9 C:/wamp/www/framework/router.php File Reference****Data Structures**

- class **router**

## Namespaces

- namespace **rolisz**

## 8.10 C:/wamp/www/framework/table.php File Reference

### Data Structures

- class **table**
- class **Query**

## Namespaces

- namespace **rolisz**

## Enumerations

- enum **SINGLE**
- enum **ONE\_TO\_MANY**
- enum **MANY**
- enum **NOT\_RECOGNIZED**
- enum **NOT\_ANALYZED**

### 8.10.1 Enumeration Type Documentation

#### 8.10.1.1 enum **MANY**

#### 8.10.1.2 enum **NOT\_ANALYZED**

#### 8.10.1.3 enum **NOT\_RECOGNIZED**

#### 8.10.1.4 enum **ONE\_TO\_MANY**

#### 8.10.1.5 enum **SINGLE**

Constants defined for describing the different kinds of relations between tables

## 8.11 C:/wamp/www/framework/template.php File Reference

### Data Structures

- class **template**

### Namespaces

- namespace **rolisz**

# Index

- \$acl
  - acl, 18
- \$connection
  - MySQLiDatabase, 32
  - table, 53
- \$database
  - MySQLiDatabase, 32
- \$denyFunction
  - acl, 18
- \$executionPoints
  - base, 20
- \$form
  - form, 26
- \$global
  - base, 20
- \$inputs
  - form, 26
- \$model
  - form, 26
- \$modifiedData
  - table, 53
- \$originalData
  - table, 53
- \$plugins
  - base, 20
- \$prefix
  - acl, 18
- \$primaryKey
  - table, 53
- \$queries
  - MySQLiDatabase, 32
- \$relations
  - table, 53
- \$requester
  - acl, 18
- \$result
  - MySQLiDatabase, 32
- \$source
  - acl, 18
- \$sourceType
  - acl, 18
- \$string
  - form, 26
- \$table
  - table, 53
- \$tables
  - table, 54
- \$values
  - template, 57
- \_\_call
  - form, 24
  - table, 47
- \_\_construct
  - acl, 14
  - base, 19
  - databaseAdapter, 21
  - form, 24
  - MySQLiDatabase, 27
  - plugin, 33
  - pluginStructure, 35
  - Query, 38
  - table, 47
  - template, 55
- \_\_destruct
  - MySQLiDatabase, 27
- \_\_get
  - table, 47
  - template, 55
- \_\_isset
  - table, 47
- \_\_set
  - table, 48
  - template, 55
- \_\_sleep
  - table, 48
- \_\_toString
  - template, 55
- acl, 13
  - \$acl, 18
  - \$denyFunction, 18
  - \$prefix, 18



- \$requester, 18
  - \$source, 18
  - \$sourceType, 18
  - \_\_construct, 14
  - addNew, 15
  - checkConsistency, 15
  - deleteACL, 15
  - editACL, 16
  - getACL, 16
  - getName, 16
  - getPath, 17
  - init, 17
  - isAllowed, 17
  - run, 17
- addNew
  - acl, 15
- addRelation
  - table, 48
- addRelationM2M
  - table, 48
- addRelationM2MS
  - table, 49
- addRelationS
  - table, 49
- alterTable
  - databaseAdapter, 21
  - MySQLiDatabase, 28
- analyzeRelations
  - table, 49
- AppName
  - base, 20
- array\_filter\_key
  - misc\_functions.php, 60
- assign
  - template, 56
- autoload
  - rolisz, 40
- base, 18
  - \$executionPoints, 20
  - \$global, 20
  - \$plugins, 20
  - \_\_construct, 19
  - AppName, 20
  - check, 19
  - get, 19
  - Module, 20
  - set, 19
  - Version, 20
- buildFilters
  - Query, 38
- buildJoins
  - Query, 39
- buildOrderBy
  - Query, 39
- buildQuery
  - Query, 39
- buildString
  - form, 24
- C:/wamp/www/framework/base.php, 59
- C:/wamp/www/framework/databaseAdapter.php, 59
- C:/wamp/www/framework/misc\_functions.php, 59
- C:/wamp/www/framework/plugin.php, 60
- C:/wamp/www/framework/plugins/acl.php, 60
- C:/wamp/www/framework/plugins/form.php, 61
- C:/wamp/www/framework/pluginStructure.php, 61
- C:/wamp/www/framework/rolisz.php, 61
- C:/wamp/www/framework/router.php, 61
- C:/wamp/www/framework/table.php, 62
- C:/wamp/www/framework/template.php, 62
- call
  - rolisz, 40
- check
  - base, 19
- checkCompile
  - template, 56
- checkConsistency
  - acl, 15
- checkExecutionPoint
  - pluginStructure, 36
- checkPlugin
  - pluginStructure, 36
- commit
  - databaseAdapter, 21
  - MySQLiDatabase, 28
- compile
  - template, 56
- connect
  - MySQLiDatabase, 28
  - rolisz, 41
- countRows
  - table, 49
- countRowsS
  - table, 50

- createTable
  - databaseAdapter, 21
  - MySQLiDatabase, 28
- databaseAdapter, 20
  - \_\_construct, 21
  - alterTable, 21
  - commit, 21
  - createTable, 21
  - disconnect, 21
  - dropTable, 21
  - escapeValue, 22
  - fetchAll, 22
  - fetchRow, 22
  - getError, 22
  - getInsertID, 22
  - numAffected, 22
  - numRows, 22
  - query, 22
  - rollback, 22
  - selectDatabase, 22
  - startTransaction, 22
  - tableExists, 23
- delete
  - table, 50
- deleteACL
  - acl, 15
- disconnect
  - databaseAdapter, 21
  - MySQLiDatabase, 29
- dropTable
  - databaseAdapter, 21
  - MySQLiDatabase, 29
- editACL
  - acl, 16
- escapeValue
  - databaseAdapter, 22
  - MySQLiDatabase, 29
- fetchAll
  - databaseAdapter, 22
  - MySQLiDatabase, 29
- fetchFirst
  - MySQLiDatabase, 30
- fetchRow
  - databaseAdapter, 22
  - MySQLiDatabase, 30
- find
  - table, 50
- findS
  - table, 51
- fixBraces
  - rolisz, 41
- fixSlashes
  - rolisz, 41
- form, 23
  - \$form, 26
  - \$inputs, 26
  - \$model, 26
  - \$string, 26
  - \_\_call, 24
  - \_\_construct, 24
  - buildString, 24
  - getDefaultMethod, 24
  - getName, 24
  - getString, 25
  - input, 25
  - removeInput, 25
  - show, 25
  - validate, 25
- get
  - base, 19
- getACL
  - acl, 16
- getColumns
  - table, 51
- getCount
  - Query, 39
- getData
  - table, 51
- getDefaultExecutionPoints
  - plugin, 34
- getDefaultMethod
  - form, 24
  - plugin, 34
- getError
  - databaseAdapter, 22
  - MySQLiDatabase, 30
- getInsertID
  - databaseAdapter, 22
  - MySQLiDatabase, 30
- getName
  - acl, 16
  - form, 24
  - plugin, 34
- getOutput
  - template, 56
- getPath

- acl, 17
- getPrimaryKey
  - table, 51
- getString
  - form, 25
- http404
  - rolisz, 41
- hydrate
  - table, 52
- importRows
  - table, 52
- includeSub
  - template, 57
- init
  - acl, 17
- input
  - form, 25
- isAllowed
  - acl, 17
- isHydrated
  - table, 52
- MANY
  - table.php, 62
- matches
  - router, 44
- misc\_functions.php
  - array\_filter\_key, 60
  - recursify, 60
- Module
  - base, 20
- MySQLiDatabase, 26
  - \$connection, 32
  - \$database, 32
  - \$queries, 32
  - \$result, 32
  - \_\_construct, 27
  - \_\_destruct, 27
  - alterTable, 28
  - commit, 28
  - connect, 28
  - createTable, 28
  - disconnect, 29
  - dropTable, 29
  - escapeValue, 29
  - fetchAll, 29
  - fetchFirst, 30
  - fetchRow, 30
  - getError, 30
  - getInsertID, 30
  - numAffected, 31
  - numRows, 31
  - Query, 31
  - rollback, 31
  - selectDatabase, 31
  - startTransaction, 32
  - tableExists, 32
- NOT\_ANALYZED
  - table.php, 62
- NOT\_RECOGNIZED
  - table.php, 62
- numAffected
  - databaseAdapter, 22
  - MySQLiDatabase, 31
- numRows
  - databaseAdapter, 22
  - MySQLiDatabase, 31
- ONE\_TO\_MANY
  - table.php, 62
- plugin, 33
  - \_\_construct, 33
  - getDefaultExecutionPoints, 34
  - getDefaultMethod, 34
  - getName, 34
  - rolisz, 42
  - run, 34
- pluginStructure, 35
  - \_\_construct, 35
  - checkExecutionPoint, 36
  - checkPlugin, 36
  - registerExecutionPoint, 36
  - registerPlugin, 36
  - runPlugins, 37
  - unregisterExecutionPoint, 37
  - unregisterPlugin, 37
- Query, 37
  - \_\_construct, 38
  - buildFilters, 38
  - buildJoins, 39
  - buildOrderBy, 39
  - buildQuery, 39
  - getCount, 39
  - MySQLiDatabase, 31
- query

- databaseAdapter, 22
- recursify
  - misc\_functions.php, 60
- registerExecutionPoint
  - pluginStructure, 36
- registerPlugin
  - pluginStructure, 36
- removeInput
  - form, 25
- Roland, 11
- rolisz, 11, 39
  - autoload, 40
  - call, 40
  - connect, 41
  - fixBraces, 41
  - fixSlashes, 41
  - http404, 41
  - plugin, 42
  - route, 42
  - run, 42
  - runPlugins, 42
  - start, 42
  - table, 42
  - urlFor, 43
- rollback
  - databaseAdapter, 22
  - MySQLiDatabase, 31
- route
  - rolisz, 42
  - router, 44
- router, 43
  - matches, 44
  - route, 44
  - run, 44
  - urlFor, 44
- run
  - acl, 17
  - plugin, 34
  - rolisz, 42
  - router, 44
- runPlugins
  - pluginStructure, 37
  - rolisz, 42
- save
  - table, 53
- selectDatabase
  - databaseAdapter, 22
  - MySQLiDatabase, 31
- set
  - base, 19
  - table, 53
- setAll
  - table, 53
- show
  - form, 25
- SINGLE
  - table.php, 62
- start
  - rolisz, 42
- startTransaction
  - databaseAdapter, 22
  - MySQLiDatabase, 32
- table, 45
  - \$connection, 53
  - \$modifiedData, 53
  - \$originalData, 53
  - \$primaryKey, 53
  - \$relations, 53
  - \$table, 53
  - \$tables, 54
  - \_\_call, 47
  - \_\_construct, 47
  - \_\_get, 47
  - \_\_isset, 47
  - \_\_set, 48
  - \_\_sleep, 48
  - addRelation, 48
  - addRelationM2M, 48
  - addRelationM2MS, 49
  - addRelationS, 49
  - analyzeRelations, 49
  - countRows, 49
  - countRowsS, 50
  - delete, 50
  - find, 50
  - findS, 51
  - getColumns, 51
  - getData, 51
  - getPrimaryKey, 51
  - hydrate, 52
  - importRows, 52
  - isHydrated, 52
  - rolisz, 42
  - save, 53
  - set, 53
  - setAll, 53
- table.php

- MANY, 62
- NOT\_ANALYZED, 62
- NOT\_RECOGNIZED, 62
- ONE\_TO\_MANY, 62
- SINGLE, 62
- tableExists
  - databaseAdapter, 23
  - MySQLiDatabase, 32
- template, 54
  - \$values, 57
  - \_\_construct, 55
  - \_\_get, 55
  - \_\_set, 55
  - \_\_toString, 55
  - assign, 56
  - checkCompile, 56
  - compile, 56
  - getOutput, 56
  - includeSub, 57
  - view, 57
- unregisterExecutionPoint
  - pluginStructure, 37
- unregisterPlugin
  - pluginStructure, 37
- urlFor
  - rolisz, 43
  - router, 44
- validate
  - form, 25
- Version
  - base, 20
- view
  - template, 57