

## Collaboratory

Generated by Doxygen 1.6.3

Mon Feb 6 18:30:34 2012



# Contents

<b>1</b>	<b>Class Index</b>	<b>1</b>
1.1	Class Hierarchy . . . . .	1
<b>2</b>	<b>Class Index</b>	<b>5</b>
2.1	Class List . . . . .	5
<b>3</b>	<b>Class Documentation</b>	<b>9</b>
3.1	AccessDeniedException Class Reference . . . . .	9
3.2	Annotation Class Reference . . . . .	11
3.2.1	Detailed Description . . . . .	13
3.2.2	Constructor & Destructor Documentation . . . . .	13
3.2.2.1	__construct . . . . .	13
3.2.3	Member Function Documentation . . . . .	14
3.2.3.1	fromScan . . . . .	14
3.2.3.2	getAnnotationId . . . . .	14
3.2.3.3	getColumns . . . . .	14
3.2.3.4	getColumnTypes . . . . .	14
3.2.3.5	getPrimaryKeys . . . . .	14
3.2.3.6	getTableName . . . . .	14
3.2.4	Member Data Documentation . . . . .	15
3.2.4.1	\$annotationId . . . . .	15
3.2.4.2	\$changedUserId . . . . .	15
3.2.4.3	\$createdUserId . . . . .	15
3.2.4.4	\$order . . . . .	15
3.2.4.5	\$polygon . . . . .	15
3.2.4.6	\$scanId . . . . .	15

3.2.4.7	\$timeChanged	15
3.2.4.8	\$timeCreated	15
3.2.4.9	\$transcriptionEng	15
3.2.4.10	\$transcriptionOrig	15
3.3	AnnotationController Class Reference	17
3.3.1	Detailed Description	19
3.3.2	Member Function Documentation	19
3.3.2.1	actionLoad	19
3.3.2.2	actionSave	19
3.3.2.3	polygonEqual	19
3.3.2.4	textEqual	19
3.4	AnnotationList Class Reference	20
3.4.1	Detailed Description	21
3.5	APCCacheEntry Class Reference	22
3.5.1	Detailed Description	23
3.6	AssociativeEntity Class Reference	24
3.6.1	Detailed Description	26
3.6.2	Member Function Documentation	26
3.6.2.1	getInsertQuery	26
3.6.2.2	save	26
3.7	Authentication Class Reference	27
3.7.1	Detailed Description	29
3.7.2	Constructor & Destructor Documentation	29
3.7.2.1	__construct	29
3.7.3	Member Function Documentation	29
3.7.3.1	assertLoggedIn	29
3.7.3.2	assertPermissionTo	29
3.7.3.3	checkPassword	30
3.7.3.4	generateUniqueToken	30
3.7.3.5	getPermissionList	30
3.7.3.6	getRank	30
3.7.3.7	getUser	30
3.7.3.8	getUserId	31
3.7.3.9	hasPermissionTo	31

3.7.3.10	isLoggedIn	31
3.7.3.11	login	31
3.7.3.12	logout	32
3.7.4	Member Data Documentation	32
3.7.4.1	\$instance	32
3.8	AuthenticationController Class Reference	33
3.8.1	Detailed Description	34
3.8.2	Member Function Documentation	35
3.8.2.1	actionGetGuestPermissionList	35
3.8.2.2	actionGetPermissionList	35
3.8.2.3	actionHasPermissionTo	35
3.8.2.4	actionKeepAlive	35
3.8.2.5	actionLogin	35
3.8.2.6	actionLogout	35
3.9	Author Class Reference	36
3.9.1	Detailed Description	38
3.9.2	Constructor & Destructor Documentation	38
3.9.2.1	__construct	38
3.9.3	Member Function Documentation	38
3.9.3.1	fromBook	38
3.9.3.2	getBookId	38
3.9.3.3	getColumns	39
3.9.3.4	getColumnTypes	39
3.9.3.5	getPrimaryKeys	39
3.9.3.6	getTableName	39
3.9.4	Member Data Documentation	39
3.9.4.1	\$bookId	39
3.9.4.2	\$personId	40
3.10	AuthorController Class Reference	41
3.10.1	Detailed Description	43
3.10.2	Member Function Documentation	43
3.10.2.1	actionLoad	43
3.11	AuthorList Class Reference	44
3.11.1	Detailed Description	45

3.12 Binding Class Reference . . . . .	46
3.12.1 Detailed Description . . . . .	48
3.12.2 Constructor & Destructor Documentation . . . . .	49
3.12.2.1 __construct . . . . .	49
3.12.3 Member Function Documentation . . . . .	49
3.12.3.1 getBindingId . . . . .	49
3.12.3.2 getColumns . . . . .	49
3.12.3.3 getColumnTypes . . . . .	49
3.12.3.4 getPrimaryKeys . . . . .	49
3.12.3.5 getTableName . . . . .	50
3.12.3.6 loadDetails . . . . .	50
3.12.3.7 saveDetails . . . . .	50
3.12.4 Member Data Documentation . . . . .	50
3.12.4.1 \$bindingId . . . . .	50
3.12.4.2 \$bindingLanguageList . . . . .	50
3.12.4.3 \$bookList . . . . .	50
3.12.4.4 \$libraryId . . . . .	50
3.12.4.5 \$provenanceList . . . . .	50
3.12.4.6 \$scanList . . . . .	50
3.12.4.7 \$signature . . . . .	51
3.12.4.8 \$status . . . . .	51
3.12.4.9 \$summary . . . . .	51
3.12.4.10 \$userId . . . . .	51
3.12.4.11 STATUS_UPLOADED . . . . .	51
3.13 BindingController Class Reference . . . . .	52
3.13.1 Detailed Description . . . . .	53
3.13.2 Member Function Documentation . . . . .	54
3.13.2.1 actionDelete . . . . .	54
3.13.2.2 actionLoad . . . . .	54
3.14 BindingLanguage Class Reference . . . . .	55
3.14.1 Detailed Description . . . . .	57
3.14.2 Constructor & Destructor Documentation . . . . .	57
3.14.2.1 __construct . . . . .	57
3.14.3 Member Function Documentation . . . . .	57

3.14.3.1	<a href="#">createOrGet</a>	57
3.14.3.2	<a href="#">getBindingId</a>	57
3.14.3.3	<a href="#">getColumns</a>	58
3.14.3.4	<a href="#">getColumnTypes</a>	58
3.14.3.5	<a href="#">getPrimaryKeys</a>	58
3.14.3.6	<a href="#">getTableName</a>	58
3.14.4	<a href="#">Member Data Documentation</a>	58
3.14.4.1	<a href="#">\$bindingId</a>	58
3.14.4.2	<a href="#">\$languageId</a>	59
3.15	<a href="#">BindingLanguageController Class Reference</a>	60
3.15.1	<a href="#">Detailed Description</a>	61
3.15.2	<a href="#">Member Function Documentation</a>	62
3.15.2.1	<a href="#">actionLoad</a>	62
3.16	<a href="#">BindingLanguageList Class Reference</a>	63
3.16.1	<a href="#">Detailed Description</a>	64
3.17	<a href="#">BindingList Class Reference</a>	65
3.17.1	<a href="#">Detailed Description</a>	66
3.18	<a href="#">BindingStatusException Class Reference</a>	67
3.18.1	<a href="#">Detailed Description</a>	68
3.19	<a href="#">BindingUploadController Class Reference</a>	69
3.19.1	<a href="#">Detailed Description</a>	70
3.19.2	<a href="#">Member Function Documentation</a>	71
3.19.2.1	<a href="#">actionDeleteUpload</a>	71
3.19.2.2	<a href="#">actionGetBindingStatus</a>	71
3.19.2.3	<a href="#">actionUniqueLibrarySignature</a>	71
3.19.2.4	<a href="#">actionUpload</a>	71
3.19.2.5	<a href="#">uniqueLibrarySignature</a>	71
3.20	<a href="#">Book Class Reference</a>	72
3.20.1	<a href="#">Detailed Description</a>	75
3.20.2	<a href="#">Constructor &amp; Destructor Documentation</a>	75
3.20.2.1	<a href="#">__construct</a>	75
3.20.3	<a href="#">Member Function Documentation</a>	75
3.20.3.1	<a href="#">fromBinding</a>	75
3.20.3.2	<a href="#">fromBindingPage</a>	75

3.20.3.3	<a href="#">getBookId</a>	75
3.20.3.4	<a href="#">getColumns</a>	76
3.20.3.5	<a href="#">getColumnTypes</a>	76
3.20.3.6	<a href="#">getPrimaryKeys</a>	76
3.20.3.7	<a href="#">getTableName</a>	76
3.20.3.8	<a href="#">loadDetails</a>	76
3.20.3.9	<a href="#">saveDetails</a>	76
3.20.4	<a href="#">Member Data Documentation</a>	77
3.20.4.1	<a href="#">\$authorList</a>	77
3.20.4.2	<a href="#">\$bindingId</a>	77
3.20.4.3	<a href="#">\$bookId</a>	77
3.20.4.4	<a href="#">\$bookLanguageList</a>	77
3.20.4.5	<a href="#">\$firstPage</a>	77
3.20.4.6	<a href="#">\$lastPage</a>	77
3.20.4.7	<a href="#">\$maxYear</a>	77
3.20.4.8	<a href="#">\$minYear</a>	77
3.20.4.9	<a href="#">\$placePublished</a>	77
3.20.4.10	<a href="#">\$preciseDate</a>	77
3.20.4.11	<a href="#">\$printVersion</a>	78
3.20.4.12	<a href="#">\$publisher</a>	78
3.20.4.13	<a href="#">\$title</a>	78
3.21	<a href="#">BookController Class Reference</a>	79
3.21.1	<a href="#">Detailed Description</a>	81
3.21.2	<a href="#">Member Function Documentation</a>	81
3.21.2.1	<a href="#">actionFirstLastPages</a>	81
3.21.2.2	<a href="#">actionLoad</a>	81
3.21.2.3	<a href="#">actionSearch</a>	81
3.21.2.4	<a href="#">addFulltext</a>	81
3.21.2.5	<a href="#">splitFulltextQuery</a>	81
3.22	<a href="#">BookLanguage Class Reference</a>	83
3.22.1	<a href="#">Detailed Description</a>	85
3.22.2	<a href="#">Constructor &amp; Destructor Documentation</a>	85
3.22.2.1	<a href="#">__construct</a>	85
3.22.3	<a href="#">Member Function Documentation</a>	85



3.22.3.1	<a href="#">createOrGet</a>	85
3.22.3.2	<a href="#">getBookId</a>	85
3.22.3.3	<a href="#">getColumns</a>	86
3.22.3.4	<a href="#">getColumnTypes</a>	86
3.22.3.5	<a href="#">getPrimaryKeys</a>	86
3.22.3.6	<a href="#">getTableName</a>	86
3.22.4	<a href="#">Member Data Documentation</a>	86
3.22.4.1	<a href="#">\$bookId</a>	86
3.22.4.2	<a href="#">\$languageId</a>	87
3.23	<a href="#">BookLanguageController Class Reference</a>	88
3.23.1	<a href="#">Detailed Description</a>	90
3.23.2	<a href="#">Member Function Documentation</a>	90
3.23.2.1	<a href="#">actionLoad</a>	90
3.24	<a href="#">BookLanguageList Class Reference</a>	91
3.24.1	<a href="#">Detailed Description</a>	92
3.25	<a href="#">BookList Class Reference</a>	93
3.25.1	<a href="#">Detailed Description</a>	94
3.26	<a href="#">BookNotFoundException Class Reference</a>	95
3.26.1	<a href="#">Detailed Description</a>	96
3.27	<a href="#">Cache Class Reference</a>	97
3.27.1	<a href="#">Detailed Description</a>	97
3.27.2	<a href="#">Member Function Documentation</a>	97
3.27.2.1	<a href="#">getEntry</a>	97
3.27.2.2	<a href="#">getFileEntry</a>	97
3.28	<a href="#">CacheEntry Class Reference</a>	98
3.28.1	<a href="#">Detailed Description</a>	99
3.28.2	<a href="#">Constructor &amp; Destructor Documentation</a>	99
3.28.2.1	<a href="#">__construct</a>	99
3.29	<a href="#">Configuration Class Reference</a>	100
3.29.1	<a href="#">Detailed Description</a>	102
3.29.2	<a href="#">Constructor &amp; Destructor Documentation</a>	102
3.29.2.1	<a href="#">__construct</a>	102
3.29.3	<a href="#">Member Function Documentation</a>	102
3.29.3.1	<a href="#">getBaseUrl</a>	102

3.29.3.2	<a href="#">getBoolean</a>	102
3.29.3.3	<a href="#">getDouble</a>	103
3.29.3.4	<a href="#">getInteger</a>	103
3.29.3.5	<a href="#">getPath</a>	103
3.29.3.6	<a href="#">getString</a>	104
3.29.4	<a href="#">Member Data Documentation</a>	104
3.29.4.1	<a href="#">\$instance</a>	104
3.30	<a href="#">ConfigurationException Class Reference</a>	105
3.31	<a href="#">Controller Class Reference</a>	107
3.31.1	<a href="#">Detailed Description</a>	107
3.31.2	<a href="#">Member Function Documentation</a>	107
3.31.2.1	<a href="#">createInstance</a>	107
3.31.2.2	<a href="#">getArray</a>	108
3.31.2.3	<a href="#">getBoolean</a>	108
3.31.2.4	<a href="#">getDouble</a>	108
3.31.2.5	<a href="#">getInteger</a>	109
3.31.2.6	<a href="#">getString</a>	109
3.31.2.7	<a href="#">handleRequest</a>	109
3.32	<a href="#">ControllerBase Class Reference</a>	110
3.32.1	<a href="#">Detailed Description</a>	111
3.32.2	<a href="#">Member Function Documentation</a>	111
3.32.2.1	<a href="#">handleLoad</a>	111
3.33	<a href="#">ControllerException Class Reference</a>	112
3.34	<a href="#">Database Class Reference</a>	114
3.34.1	<a href="#">Member Function Documentation</a>	116
3.34.1.1	<a href="#">commit</a>	116
3.34.1.2	<a href="#">convertFromType</a>	116
3.34.1.3	<a href="#">convertFromTypes</a>	116
3.34.1.4	<a href="#">doTransaction</a>	117
3.34.1.5	<a href="#">execute</a>	117
3.34.1.6	<a href="#">inTransaction</a>	117
3.34.1.7	<a href="#">rollBack</a>	118
3.34.1.8	<a href="#">startTransaction</a>	118
3.34.2	<a href="#">Member Data Documentation</a>	118

3.34.2.1	\$instance	118
3.35	DatabaseException Class Reference	119
3.36	Entity Class Reference	121
3.36.1	Detailed Description	122
3.36.2	Member Function Documentation	122
3.36.2.1	arePrimaryKeysFilled	122
3.36.2.2	delete	122
3.36.2.3	getAllValues	122
3.36.2.4	getColumns	123
3.36.2.5	getColumnTypes	123
3.36.2.6	getDefaultColumns	123
3.36.2.7	getDefaultColumnTypes	123
3.36.2.8	getDeleteQuery	123
3.36.2.9	getInsertQuery	124
3.36.2.10	getMarkedAsDeleted	124
3.36.2.11	getPrimaryKeys	124
3.36.2.12	getPrimaryKeyValue	124
3.36.2.13	getSelectQuery	124
3.36.2.14	getTableName	125
3.36.2.15	getTypes	125
3.36.2.16	getUpdateQuery	125
3.36.2.17	getValues	125
3.36.2.18	load	125
3.36.2.19	save	125
3.36.2.20	saveDetails	126
3.36.2.21	saveWithDetails	126
3.36.2.22	setRawValues	126
3.36.2.23	setValues	126
3.37	EntityException Class Reference	127
3.38	EntityList Class Reference	129
3.38.1	Detailed Description	130
3.38.2	Constructor & Destructor Documentation	130
3.38.2.1	__construct	130
3.38.3	Member Function Documentation	130

3.38.3.1	add	130
3.38.3.2	buildSelectionQuery	130
3.38.3.3	delete	130
3.38.3.4	find	131
3.38.3.5	get	131
3.38.3.6	getByKeyValue	131
3.38.3.7	getColumns	132
3.38.3.8	getEntities	132
3.38.3.9	getIterator	132
3.38.3.10	getPrimaryKeys	132
3.38.3.11	getTableName	132
3.38.3.12	getTotal	132
3.38.3.13	getType	133
3.38.3.14	getTypes	133
3.38.3.15	getValue	133
3.38.3.16	getValues	133
3.38.3.17	load	133
3.38.3.18	markAllAsDeleted	133
3.38.3.19	markAllAsUpdated	134
3.38.3.20	remove	134
3.38.3.21	removeAt	134
3.38.3.22	save	134
3.38.3.23	setValue	134
3.38.3.24	setValues	134
3.38.3.25	tryGet	135
3.39	ExceptionBase Class Reference	136
3.39.1	Detailed Description	136
3.39.2	Member Function Documentation	136
3.39.2.1	getIdentifier	136
3.39.2.2	getTimestamp	136
3.40	FileCacheEntry Class Reference	137
3.40.1	Detailed Description	138
3.41	FormatException Class Reference	140
3.42	HelpController Class Reference	142

3.42.1 Detailed Description . . . . .	143
3.42.2 Member Function Documentation . . . . .	144
3.42.2.1 actionLoad . . . . .	144
3.42.2.2 setLeaves . . . . .	144
3.43 HelpPage Class Reference . . . . .	145
3.43.1 Detailed Description . . . . .	147
3.43.2 Constructor & Destructor Documentation . . . . .	147
3.43.2.1 __construct . . . . .	147
3.43.3 Member Function Documentation . . . . .	147
3.43.3.1 getChildren . . . . .	147
3.43.3.2 getColumns . . . . .	148
3.43.3.3 getColumnTypes . . . . .	148
3.43.3.4 getHelpPageId . . . . .	148
3.43.3.5 getPrimaryKeys . . . . .	148
3.43.3.6 getTableName . . . . .	148
3.43.3.7 getValues . . . . .	149
3.43.4 Member Data Documentation . . . . .	149
3.43.4.1 \$helpPageId . . . . .	149
3.43.4.2 \$helpType . . . . .	149
3.43.4.3 \$pageName . . . . .	149
3.44 HelpPageList Class Reference . . . . .	150
3.44.1 Detailed Description . . . . .	151
3.44.2 Member Function Documentation . . . . .	152
3.44.2.1 add . . . . .	152
3.45 HelpParagraph Class Reference . . . . .	153
3.45.1 Detailed Description . . . . .	155
3.45.2 Constructor & Destructor Documentation . . . . .	155
3.45.2.1 __construct . . . . .	155
3.45.3 Member Function Documentation . . . . .	156
3.45.3.1 getChildren . . . . .	156
3.45.3.2 getColumns . . . . .	156
3.45.3.3 getColumnTypes . . . . .	156
3.45.3.4 getContent . . . . .	156
3.45.3.5 getHelpParagraphId . . . . .	156

3.45.3.6	<a href="#">getPrimaryKeys</a>	157
3.45.3.7	<a href="#">getTableName</a>	157
3.45.3.8	<a href="#">getValues</a>	157
3.45.4	<a href="#">Member Data Documentation</a>	157
3.45.4.1	<a href="#">\$actionName</a>	157
3.45.4.2	<a href="#">\$helpPageId</a>	157
3.45.4.3	<a href="#">\$helpParagraphId</a>	157
3.45.4.4	<a href="#">\$paragraphParentId</a>	158
3.45.4.5	<a href="#">\$settingName</a>	158
3.45.4.6	<a href="#">\$title</a>	158
3.46	<a href="#">JavascriptMinifier Class Reference</a>	159
3.46.1	<a href="#">Detailed Description</a>	159
3.46.2	<a href="#">Constructor &amp; Destructor Documentation</a>	160
3.46.2.1	<a href="#">__construct</a>	160
3.46.3	<a href="#">Member Function Documentation</a>	160
3.46.3.1	<a href="#">action</a>	160
3.46.3.2	<a href="#">get</a>	160
3.46.3.3	<a href="#">isAlphaNum</a>	160
3.46.3.4	<a href="#">min</a>	161
3.46.3.5	<a href="#">minify</a>	161
3.46.3.6	<a href="#">next</a>	161
3.46.3.7	<a href="#">peek</a>	161
3.47	<a href="#">JavascriptMinifierException Class Reference</a>	162
3.47.1	<a href="#">Detailed Description</a>	163
3.48	<a href="#">Language Class Reference</a>	165
3.48.1	<a href="#">Detailed Description</a>	167
3.48.2	<a href="#">Constructor &amp; Destructor Documentation</a>	167
3.48.2.1	<a href="#">__construct</a>	167
3.48.3	<a href="#">Member Function Documentation</a>	167
3.48.3.1	<a href="#">getColumns</a>	167
3.48.3.2	<a href="#">getColumnTypes</a>	167
3.48.3.3	<a href="#">getLanguageId</a>	167
3.48.3.4	<a href="#">getPrimaryKeys</a>	168
3.48.3.5	<a href="#">getTableName</a>	168

3.48.4	Member Data Documentation	168
3.48.4.1	\$languageId	168
3.48.4.2	\$languageName	168
3.49	LanguageController Class Reference	169
3.49.1	Detailed Description	171
3.49.2	Member Function Documentation	171
3.49.2.1	actionLoad	171
3.50	LanguageList Class Reference	172
3.50.1	Detailed Description	173
3.51	Library Class Reference	174
3.51.1	Detailed Description	176
3.51.2	Constructor & Destructor Documentation	176
3.51.2.1	__construct	176
3.51.3	Member Function Documentation	176
3.51.3.1	getColumns	176
3.51.3.2	getColumnTypes	177
3.51.3.3	getLibraryId	177
3.51.3.4	getPrimaryKeys	177
3.51.3.5	getTableName	177
3.51.4	Member Data Documentation	177
3.51.4.1	\$info	177
3.51.4.2	\$libraryAddress	177
3.51.4.3	\$libraryId	177
3.51.4.4	\$libraryName	178
3.51.4.5	\$website	178
3.52	LibraryList Class Reference	179
3.52.1	Detailed Description	180
3.53	Log Class Reference	181
3.53.1	Detailed Description	183
3.53.2	Constructor & Destructor Documentation	183
3.53.2.1	__construct	183
3.53.2.2	__destruct	183
3.53.3	Member Function Documentation	183
3.53.3.1	debug	183

3.53.3.2	error	183
3.53.3.3	info	184
3.53.3.4	trace	184
3.53.3.5	warn	184
3.53.4	Member Data Documentation	184
3.53.4.1	\$instance	184
3.54	Mailer Class Reference	185
3.54.1	Detailed Description	185
3.54.2	Member Function Documentation	185
3.54.2.1	sendActivationMail	185
3.54.2.2	sendMail	185
3.54.2.3	sendPasswordRestorationMail	186
3.54.2.4	sendUserDeclinedMail	186
3.55	MailerException Class Reference	187
3.55.1	Detailed Description	188
3.56	MainController Class Reference	189
3.56.1	Detailed Description	190
3.56.2	Member Function Documentation	191
3.56.2.1	actionIndex	191
3.56.2.2	actionScript	191
3.56.2.3	actionStyle	191
3.57	Note Class Reference	192
3.57.1	Detailed Description	194
3.57.2	Constructor & Destructor Documentation	194
3.57.2.1	__construct	194
3.57.3	Member Function Documentation	194
3.57.3.1	getColumns	194
3.57.3.2	getColumnTypes	194
3.57.3.3	getPrimaryKeys	195
3.57.3.4	getTableName	195
3.57.3.5	getUserId	195
3.57.4	Member Data Documentation	195
3.57.4.1	\$text	195
3.57.4.2	\$userId	195



3.58	NoteController Class Reference	196
3.58.1	Detailed Description	197
3.58.2	Member Function Documentation	197
3.58.2.1	actionLoad	197
3.58.2.2	actionSave	198
3.59	NotLoggedOnException Class Reference	199
3.59.1	Detailed Description	200
3.60	PasswordIncorrectException Class Reference	201
3.61	Pdf Class Reference	203
3.61.1	Detailed Description	203
3.61.2	Constructor & Destructor Documentation	203
3.61.2.1	__construct	203
3.62	PdfController Class Reference	204
3.62.1	Detailed Description	205
3.62.2	Member Function Documentation	206
3.62.2.1	actionDownload	206
3.62.2.2	actionGenerate	206
3.63	PdfException Class Reference	207
3.64	PendingUser Class Reference	209
3.64.1	Detailed Description	211
3.64.2	Constructor & Destructor Documentation	211
3.64.2.1	__construct	211
3.64.3	Member Function Documentation	211
3.64.3.1	fromUser	211
3.64.3.2	getColumns	211
3.64.3.3	getColumnTypes	212
3.64.3.4	getPendingUserId	212
3.64.3.5	getPrimaryKeys	212
3.64.3.6	getTableName	212
3.64.4	Member Data Documentation	212
3.64.4.1	\$confirmationCode	212
3.64.4.2	\$expirationDate	212
3.64.4.3	\$pendingUserId	212
3.64.4.4	\$userId	212

3.65	Permission Class Reference . . . . .	213
3.65.1	Detailed Description . . . . .	215
3.65.2	Member Function Documentation . . . . .	215
3.65.2.1	getActionName . . . . .	215
3.65.2.2	getColumns . . . . .	215
3.65.2.3	getColumnTypes . . . . .	215
3.65.2.4	getPermissionListForRank . . . . .	215
3.65.2.5	getPrimaryKeys . . . . .	216
3.65.2.6	getTableName . . . . .	216
3.65.2.7	rankHasPermission . . . . .	216
3.65.3	Member Data Documentation . . . . .	216
3.65.3.1	\$actionName . . . . .	216
3.65.3.2	\$minRank . . . . .	216
3.66	Person Class Reference . . . . .	217
3.66.1	Detailed Description . . . . .	219
3.66.2	Constructor & Destructor Documentation . . . . .	219
3.66.2.1	__construct . . . . .	219
3.66.3	Member Function Documentation . . . . .	219
3.66.3.1	getColumns . . . . .	219
3.66.3.2	getColumnTypes . . . . .	219
3.66.3.3	getPersonId . . . . .	220
3.66.3.4	getPrimaryKeys . . . . .	220
3.66.3.5	getTableName . . . . .	220
3.66.4	Member Data Documentation . . . . .	220
3.66.4.1	\$name . . . . .	220
3.66.4.2	\$personId . . . . .	220
3.67	PersonList Class Reference . . . . .	221
3.67.1	Detailed Description . . . . .	222
3.68	Provenance Class Reference . . . . .	223
3.68.1	Detailed Description . . . . .	225
3.68.2	Constructor & Destructor Documentation . . . . .	225
3.68.2.1	__construct . . . . .	225
3.68.3	Member Function Documentation . . . . .	225
3.68.3.1	fromBinding . . . . .	225

3.68.3.2	<a href="#">getBindingId</a>	225
3.68.3.3	<a href="#">getColumns</a>	226
3.68.3.4	<a href="#">getColumnTypes</a>	226
3.68.3.5	<a href="#">getPrimaryKeys</a>	226
3.68.3.6	<a href="#">getTableName</a>	226
3.68.4	<a href="#">Member Data Documentation</a>	226
3.68.4.1	<a href="#">\$bindingId</a>	226
3.68.4.2	<a href="#">\$personId</a>	227
3.69	<a href="#">ProvenanceController Class Reference</a>	228
3.69.1	<a href="#">Detailed Description</a>	230
3.69.2	<a href="#">Member Function Documentation</a>	230
3.69.2.1	<a href="#">actionLoad</a>	230
3.70	<a href="#">ProvenanceList Class Reference</a>	231
3.70.1	<a href="#">Detailed Description</a>	232
3.71	<a href="#">Query Class Reference</a>	233
3.71.1	<a href="#">Detailed Description</a>	233
3.71.2	<a href="#">Member Function Documentation</a>	234
3.71.2.1	<a href="#">clear</a>	234
3.71.2.2	<a href="#">delete</a>	234
3.71.2.3	<a href="#">from</a>	234
3.71.2.4	<a href="#">insert</a>	234
3.71.2.5	<a href="#">returning</a>	234
3.71.2.6	<a href="#">select</a>	235
3.71.2.7	<a href="#">update</a>	235
3.71.2.8	<a href="#">where</a>	235
3.71.2.9	<a href="#">whereFulltext</a>	235
3.72	<a href="#">QueryFormatException Class Reference</a>	236
3.73	<a href="#">RegistrationFailedException Class Reference</a>	238
3.74	<a href="#">ResultSet Class Reference</a>	240
3.74.1	<a href="#">Detailed Description</a>	240
3.74.2	<a href="#">Constructor &amp; Destructor Documentation</a>	240
3.74.2.1	<a href="#">__construct</a>	240
3.74.2.2	<a href="#">__destruct</a>	240
3.74.3	<a href="#">Member Function Documentation</a>	241

3.74.3.1	<a href="#">asArrays</a>	241
3.74.3.2	<a href="#">getAmount</a>	241
3.74.3.3	<a href="#">getFirstRow</a>	241
3.74.3.4	<a href="#">getIterator</a>	241
3.74.3.5	<a href="#">tryGetFirstRow</a>	241
3.75	<a href="#">ResultSetException Class Reference</a>	242
3.76	<a href="#">ResultSetIterator Class Reference</a>	244
3.76.1	<a href="#">Detailed Description</a>	244
3.76.2	<a href="#">Member Function Documentation</a>	244
3.76.2.1	<a href="#">current</a>	244
3.76.2.2	<a href="#">key</a>	244
3.76.2.3	<a href="#">next</a>	244
3.76.2.4	<a href="#">rewind</a>	244
3.76.2.5	<a href="#">valid</a>	244
3.77	<a href="#">ResultSetRow Class Reference</a>	245
3.77.1	<a href="#">Detailed Description</a>	245
3.77.2	<a href="#">Constructor &amp; Destructor Documentation</a>	245
3.77.2.1	<a href="#">__construct</a>	245
3.77.3	<a href="#">Member Function Documentation</a>	245
3.77.3.1	<a href="#">getValue</a>	245
3.77.3.2	<a href="#">getValues</a>	245
3.78	<a href="#">Scan Class Reference</a>	247
3.78.1	<a href="#">Detailed Description</a>	250
3.78.2	<a href="#">Constructor &amp; Destructor Documentation</a>	250
3.78.2.1	<a href="#">__construct</a>	250
3.78.3	<a href="#">Member Function Documentation</a>	250
3.78.3.1	<a href="#">createEmptyScan</a>	250
3.78.3.2	<a href="#">fromBinding</a>	250
3.78.3.3	<a href="#">fromBindingPage</a>	250
3.78.3.4	<a href="#">getColumns</a>	251
3.78.3.5	<a href="#">getColumnTypes</a>	251
3.78.3.6	<a href="#">getPrimaryKeys</a>	251
3.78.3.7	<a href="#">getScanId</a>	251
3.78.3.8	<a href="#">getTableName</a>	251

3.78.4	Member Data Documentation	251
3.78.4.1	\$bindingId	251
3.78.4.2	\$bookTitle	251
3.78.4.3	\$height	252
3.78.4.4	\$page	252
3.78.4.5	\$scanId	252
3.78.4.6	\$scanName	252
3.78.4.7	\$scanType	252
3.78.4.8	\$status	252
3.78.4.9	\$uploadId	252
3.78.4.10	\$width	252
3.78.4.11	\$zoomLevel	252
3.78.4.12	STATUS_PENDING	252
3.78.4.13	TYPE_JPEG	253
3.79	ScanController Class Reference	254
3.79.1	Detailed Description	255
3.79.2	Member Function Documentation	256
3.79.2.1	actionLoad	256
3.79.2.2	actionReorder	256
3.80	ScanList Class Reference	257
3.80.1	Detailed Description	258
3.81	Session Class Reference	259
3.81.1	Detailed Description	261
3.81.2	Constructor & Destructor Documentation	261
3.81.2.1	__construct	261
3.81.3	Member Function Documentation	261
3.81.3.1	exists	261
3.81.3.2	getVar	261
3.81.3.3	setVar	261
3.81.3.4	unsetVar	262
3.81.4	Member Data Documentation	262
3.81.4.1	\$instance	262
3.82	Setting Class Reference	263
3.82.1	Detailed Description	265

3.82.2	Member Function Documentation	265
3.82.2.1	getColumns	265
3.82.2.2	getColumnTypes	265
3.82.2.3	getPrimaryKeys	265
3.82.2.4	getSetting	265
3.82.2.5	getSettingName	266
3.82.2.6	getTableName	266
3.82.2.7	setSetting	266
3.82.3	Member Data Documentation	266
3.82.3.1	\$settingName	266
3.82.3.2	\$settingValue	266
3.82.3.3	\$visible	267
3.83	SettingController Class Reference	268
3.83.1	Detailed Description	270
3.83.2	Member Function Documentation	270
3.83.2.1	actionGetSetting	270
3.84	SettingException Class Reference	271
3.85	SettingNotFoundException Class Reference	273
3.86	Singleton Class Reference	275
3.86.1	Detailed Description	275
3.86.2	Member Function Documentation	275
3.86.2.1	getInstance	275
3.87	SingletonException Class Reference	277
3.88	StylesheetMinifier Class Reference	279
3.88.1	Detailed Description	279
3.88.2	Constructor & Destructor Documentation	279
3.88.2.1	__construct	279
3.88.3	Member Function Documentation	279
3.88.3.1	min	279
3.88.3.2	minify	279
3.89	TilePyramidBuilder Class Reference	281
3.89.1	Detailed Description	283
3.89.2	Member Function Documentation	283
3.89.2.1	doIteration	283

3.89.2.2	<a href="#">resolveInconsistencies</a>	283
3.89.2.3	<a href="#">runBuilder</a>	283
3.89.3	<a href="#">Member Data Documentation</a>	283
3.89.3.1	<a href="#">\$instance</a>	283
3.90	<a href="#">TilePyramidBuilderException Class Reference</a>	284
3.90.1	<a href="#">Detailed Description</a>	285
3.91	<a href="#">Translator Class Reference</a>	286
3.91.1	<a href="#">Detailed Description</a>	287
3.91.2	<a href="#">Constructor &amp; Destructor Documentation</a>	288
3.91.2.1	<a href="#">__construct</a>	288
3.91.3	<a href="#">Member Function Documentation</a>	288
3.91.3.1	<a href="#">getEntry</a>	288
3.91.4	<a href="#">Member Data Documentation</a>	288
3.91.4.1	<a href="#">\$instance</a>	288
3.92	<a href="#">Upload Class Reference</a>	289
3.92.1	<a href="#">Detailed Description</a>	291
3.92.2	<a href="#">Constructor &amp; Destructor Documentation</a>	291
3.92.2.1	<a href="#">__construct</a>	291
3.92.3	<a href="#">Member Function Documentation</a>	292
3.92.3.1	<a href="#">createEmptyUpload</a>	292
3.92.3.2	<a href="#">delete</a>	292
3.92.3.3	<a href="#">fromToken</a>	292
3.92.3.4	<a href="#">generateNewToken</a>	292
3.92.3.5	<a href="#">getColumns</a>	292
3.92.3.6	<a href="#">getColumnTypes</a>	293
3.92.3.7	<a href="#">getContent</a>	293
3.92.3.8	<a href="#">getFileLocation</a>	293
3.92.3.9	<a href="#">getPrimaryKeys</a>	293
3.92.3.10	<a href="#">getTableName</a>	293
3.92.3.11	<a href="#">getUploadId</a>	293
3.92.3.12	<a href="#">setContent</a>	293
3.92.4	<a href="#">Member Data Documentation</a>	294
3.92.4.1	<a href="#">\$uploadId</a>	294
3.92.4.2	<a href="#">STATUS_AVAILABLE</a>	294

3.93 UploadController Class Reference . . . . .	295
3.93.1 Detailed Description . . . . .	296
3.93.2 Member Function Documentation . . . . .	297
3.93.2.1 actionDelete . . . . .	297
3.93.2.2 actionFetchToken . . . . .	297
3.93.2.3 actionFetchUploads . . . . .	297
3.93.2.4 actionUpload . . . . .	297
3.94 UploadException Class Reference . . . . .	298
3.95 User Class Reference . . . . .	300
3.95.1 Detailed Description . . . . .	303
3.95.2 Constructor & Destructor Documentation . . . . .	303
3.95.2.1 __construct . . . . .	303
3.95.3 Member Function Documentation . . . . .	303
3.95.3.1 checkPassword . . . . .	303
3.95.3.2 delete . . . . .	304
3.95.3.3 fromEmailAddress . . . . .	304
3.95.3.4 fromUsername . . . . .	304
3.95.3.5 fromUsernameAndPassword . . . . .	304
3.95.3.6 getColumns . . . . .	305
3.95.3.7 getColumnTypes . . . . .	305
3.95.3.8 getPrimaryKeys . . . . .	305
3.95.3.9 getTableName . . . . .	305
3.95.3.10 getUserId . . . . .	306
3.95.4 Member Data Documentation . . . . .	306
3.95.4.1 \$activationStage . . . . .	306
3.95.4.2 \$affiliation . . . . .	306
3.95.4.3 \$banned . . . . .	306
3.95.4.4 \$email . . . . .	306
3.95.4.5 \$firstName . . . . .	306
3.95.4.6 \$homeAddress . . . . .	306
3.95.4.7 \$lastActive . . . . .	306
3.95.4.8 \$lastName . . . . .	306
3.95.4.9 \$occupation . . . . .	306
3.95.4.10 \$passwordHash . . . . .	307



3.95.4.11 \$passwordRestoreToken . . . . .	307
3.95.4.12 \$rank . . . . .	307
3.95.4.13 \$registrationDate . . . . .	307
3.95.4.14 \$userId . . . . .	307
3.95.4.15 \$username . . . . .	307
3.95.4.16 \$website . . . . .	307
3.95.4.17 ACTIVE_STAGE_PENDING . . . . .	307
3.95.4.18 RANK_NONE . . . . .	307
3.96 UserActivationController Class Reference . . . . .	308
3.96.1 Detailed Description . . . . .	309
3.96.2 Member Function Documentation . . . . .	310
3.96.2.1 actionActivateUser . . . . .	310
3.96.2.2 actionResendActivationMail . . . . .	310
3.96.2.3 actionSetAutoAcceptance . . . . .	310
3.96.2.4 actionSetUserAccepted . . . . .	310
3.97 UserActivationException Class Reference . . . . .	312
3.98 UserBannedException Class Reference . . . . .	314
3.99 UserController Class Reference . . . . .	316
3.99.1 Detailed Description . . . . .	318
3.99.2 Member Function Documentation . . . . .	318
3.99.2.1 actionBanUser . . . . .	318
3.99.2.2 actionChangeForgottenPassword . . . . .	318
3.99.2.3 actionChangeRole . . . . .	318
3.99.2.4 actionCreate . . . . .	318
3.99.2.5 actionDeleteUser . . . . .	318
3.99.2.6 actionEmailExists . . . . .	318
3.99.2.7 actionLoad . . . . .	318
3.99.2.8 actionPasswordForgotten . . . . .	319
3.99.2.9 actionSave . . . . .	319
3.99.2.10 actionUnBanUser . . . . .	319
3.99.2.11 actionUsernameExists . . . . .	319
3.100 UserList Class Reference . . . . .	320
3.100.1 Detailed Description . . . . .	321
3.101 UserNotFoundException Class Reference . . . . .	322

3.101.1 Detailed Description . . . . .	323
3.102UserPendingException Class Reference . . . . .	324
3.103UserShouldActivateException Class Reference . . . . .	326

# Chapter 1

## Class Index

### 1.1 Class Hierarchy

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

Cache . . . . .	97
CacheEntry . . . . .	98
APCCacheEntry . . . . .	22
FileCacheEntry . . . . .	137
Controller . . . . .	107
AuthenticationController . . . . .	33
BindingUploadController . . . . .	69
ControllerBase . . . . .	110
AnnotationController . . . . .	17
AuthorController . . . . .	41
BindingController . . . . .	52
BindingLanguageController . . . . .	60
BookController . . . . .	79
BookLanguageController . . . . .	88
HelpController . . . . .	142
LanguageController . . . . .	169
ProvenanceController . . . . .	228
ScanController . . . . .	254
SettingController . . . . .	268
UserController . . . . .	316
MainController . . . . .	189
NoteController . . . . .	196
PdfController . . . . .	204
UploadController . . . . .	295
UserActivationController . . . . .	308
Entity . . . . .	121
Annotation . . . . .	11
AssociativeEntity . . . . .	24

Author . . . . .	36
BindingLanguage . . . . .	55
BookLanguage . . . . .	83
Note . . . . .	192
Provenance . . . . .	223
Binding . . . . .	46
Book . . . . .	72
HelpPage . . . . .	145
HelpParagraph . . . . .	153
Language . . . . .	165
Library . . . . .	174
PendingUser . . . . .	209
Permission . . . . .	213
Person . . . . .	217
Scan . . . . .	247
Setting . . . . .	263
Upload . . . . .	289
User . . . . .	300
EntityList . . . . .	129
AnnotationList . . . . .	20
AuthorList . . . . .	44
BindingLanguageList . . . . .	63
BindingList . . . . .	65
BookLanguageList . . . . .	91
BookList . . . . .	93
HelpPageList . . . . .	150
LanguageList . . . . .	172
LibraryList . . . . .	179
PersonList . . . . .	221
ProvenanceList . . . . .	231
ScanList . . . . .	257
UserList . . . . .	320
ExceptionBase . . . . .	136
AccessDeniedException . . . . .	9
BindingStatusException . . . . .	67
BookNotFoundException . . . . .	95
ConfigurationException . . . . .	105
SettingNotFoundException . . . . .	273
ControllerException . . . . .	112
DatabaseException . . . . .	119
EntityException . . . . .	127
FormatException . . . . .	140
JavascriptMinifierException . . . . .	162
MailerException . . . . .	187
NotLoggedInException . . . . .	199
PasswordIncorrectException . . . . .	201
PdfException . . . . .	207
QueryFormatException . . . . .	236
RegistrationFailedException . . . . .	238

ResultSetException . . . . .	242
SettingException . . . . .	271
SingletonException . . . . .	277
TilePyramidBuilderException . . . . .	284
UploadException . . . . .	298
UserActivationException . . . . .	312
UserBannedException . . . . .	314
UserNotFoundException . . . . .	322
UserPendingException . . . . .	324
UserShouldActivateException . . . . .	326
JavascriptMinifier . . . . .	159
Mailer . . . . .	185
Pdf . . . . .	203
Query . . . . .	233
ResultSet . . . . .	240
ResultSetIterator . . . . .	244
ResultSetRow . . . . .	245
Singleton . . . . .	275
Authentication . . . . .	27
Configuration . . . . .	100
Database . . . . .	114
Log . . . . .	181
Session . . . . .	259
TilePyramidBuilder . . . . .	281
Translator . . . . .	286
StylesheetMinifier . . . . .	279



## Chapter 2

# Class Index

### 2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">AccessDeniedException</a>	9
<a href="#">Annotation</a>	11
<a href="#">AnnotationController</a>	17
<a href="#">AnnotationList</a>	20
<a href="#">APCCacheEntry</a>	22
<a href="#">AssociativeEntity</a>	24
<a href="#">Authentication</a>	27
<a href="#">AuthenticationController</a>	33
<a href="#">Author</a>	36
<a href="#">AuthorController</a>	41
<a href="#">AuthorList</a>	44
<a href="#">Binding</a>	46
<a href="#">BindingController</a>	52
<a href="#">BindingLanguage</a>	55
<a href="#">BindingLanguageController</a>	60
<a href="#">BindingLanguageList</a>	63
<a href="#">BindingList</a>	65
<a href="#">BindingStatusException</a>	67
<a href="#">BindingUploadController</a>	69
<a href="#">Book</a>	72
<a href="#">BookController</a>	79
<a href="#">BookLanguage</a>	83
<a href="#">BookLanguageController</a>	88
<a href="#">BookLanguageList</a>	91
<a href="#">BookList</a>	93
<a href="#">BookNotFoundException</a>	95
<a href="#">Cache</a>	97
<a href="#">CacheEntry</a>	98
<a href="#">Configuration</a>	100

ConfigurationException	105
Controller	107
ControllerBase	110
ControllerException	112
Database	114
DatabaseException	119
Entity	121
EntityException	127
EntityList	129
ExceptionBase	136
FileCacheEntry	137
FormatException	140
HelpController	142
HelpPage	145
HelpPageList	150
HelpParagraph	153
JavascriptMinifier	159
JavascriptMinifierException	162
Language	165
LanguageController	169
LanguageList	172
Library	174
LibraryList	179
Log	181
Mailer	185
MailerException	187
MainController	189
Note	192
NoteController	196
NotLoggedInException	199
PasswordIncorrectException	201
Pdf	203
PdfController	204
PdfException	207
PendingUser	209
Permission	213
Person	217
PersonList	221
Provenance	223
ProvenanceController	228
ProvenanceList	231
Query	233
QueryFormatException	236
RegistrationFailedException	238
ResultSet	240
ResultSetException	242
ResultSetIterator	244
ResultSetRow	245
Scan	247
ScanController	254



ScanList	257
Session	259
Setting	263
SettingController	268
SettingException	271
SettingNotFoundException	273
Singleton	275
SingletonException	277
StylesheetMinifier	279
TilePyramidBuilder	281
TilePyramidBuilderException	284
Translator	286
Upload	289
UploadController	295
UploadException	298
User	300
UserActivationController	308
UserActivationException	312
UserBannedException	314
UserController	316
UserList	320
UserNotFoundException	322
UserPendingException	324
UserShouldActivateException	326

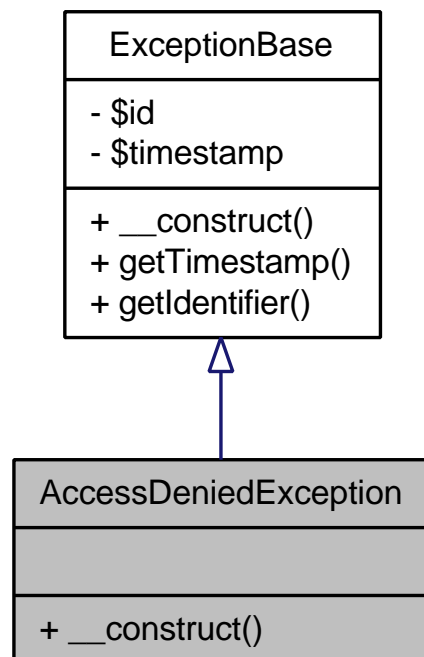


## Chapter 3

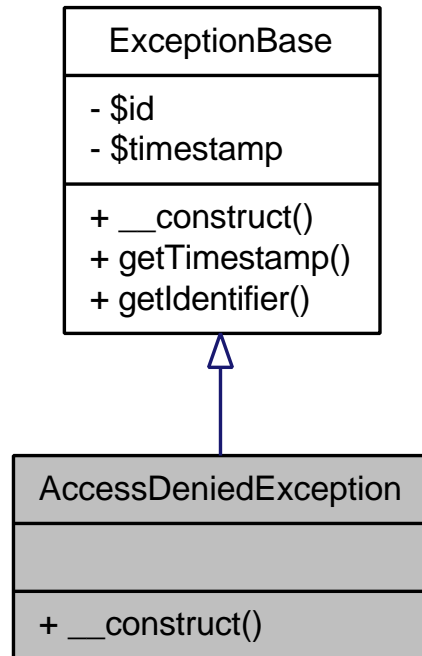
# Class Documentation

### 3.1 AccessDeniedException Class Reference

Inheritance diagram for AccessDeniedException:



Collaboration diagram for AccessDeniedException:



## Public Member Functions

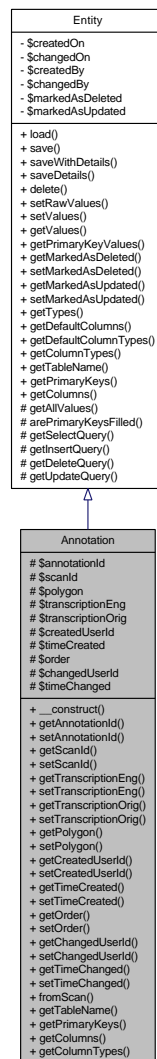
- **\_\_construct** (\$actionname)

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

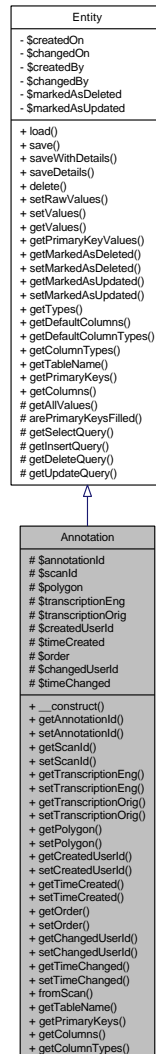
- [www/backend/util/authentication.php](http://www/backend/util/authentication.php)

## 3.2 Annotation Class Reference

Inheritance diagram for Annotation:



Collaboration diagram for Annotation:



## Public Member Functions

- [\\_\\_construct](#) (\$id=null)
- [getAnnotationId](#) ()
- [setAnnotationId](#) (\$id)
- [getScanId](#) ()
- [setScanId](#) (\$scanId)
- [getTranscriptionEng](#) ()
- [setTranscriptionEng](#) (\$text)
- [getTranscriptionOrig](#) ()
- [setTranscriptionOrig](#) (\$text)

- **getPolygon** ()
- **setPolygon** (\$vertices)
- **getCreatedUserId** ()
- **setCreatedUserId** (\$id)
- **getTimeCreated** ()
- **setTimeCreated** (\$time)
- **getOrder** ()
- **setOrder** (\$order)
- **getChangedUserId** ()
- **setChangedUserId** (\$id)
- **getTimeChanged** ()
- **setTimeChanged** (\$time)

### Static Public Member Functions

- static **fromScan** (\$scan)
- static **getTableName** ()
- static **getPrimaryKeys** ()
- static **getColumns** ()
- static **getColumnTypes** ()

### Protected Attributes

- **\$annotationId**
- **\$scanId**
- **\$polygon**
- **\$transcriptionEng**
- **\$transcriptionOrig**
- **\$createdUserId**
- **\$timeCreated**
- **\$order**
- **\$changedUserId**
- **\$timeChanged**

#### 3.2.1 Detailed Description

Class representing an annotation entity.

#### 3.2.2 Constructor & Destructor Documentation

##### 3.2.2.1 Annotation::\_\_construct (\$id = null)

Constructs an annotation entity.

##### Parameters

**\$id** Id of the annotation. Default (null) will create a new annotation.

### 3.2.3 Member Function Documentation

#### 3.2.3.1 static Annotation::fromScan (\$ *scan*) [static]

Returns all the annotations which belong to one scan

##### Parameters

*\$scan* The scan model

##### Returns

Array of annotation models

#### 3.2.3.2 Annotation::getAnnotationId ()

Getters and setters.

#### 3.2.3.3 static Annotation::getColumns () [static]

Gets all the columns that are not primary keys as an array.

Reimplemented from [Entity](#).

#### 3.2.3.4 static Annotation::getColumnTypes () [static]

Gets all the column types, per column, including primary keys.

##### Returns

Array of all column types.

Reimplemented from [Entity](#).

#### 3.2.3.5 static Annotation::getPrimaryKeys () [static]

Get an array with the primary keys.

Reimplemented from [Entity](#).

#### 3.2.3.6 static Annotation::getTableName () [static]

Get the name of the corresponding table.

Reimplemented from [Entity](#).



## 3.2.4 Member Data Documentation

### 3.2.4.1 Annotation::\$annotationId [protected]

Id of this annotation.

### 3.2.4.2 Annotation::\$changedUserId [protected]

The Id of the user who last modified this annotation.

### 3.2.4.3 Annotation::\$createdUserId [protected]

The Id of the user who created this annotation.

### 3.2.4.4 Annotation::\$order [protected]

The position of this annotation on the list of annotation which belong to one scan.

### 3.2.4.5 Annotation::\$polygon [protected]

The polygon of this annotation.

### 3.2.4.6 Annotation::\$scanId [protected]

Id of the scan to which this annotation belongs.

### 3.2.4.7 Annotation::\$timeChanged [protected]

the time and date on which this annotation was last modified.

### 3.2.4.8 Annotation::\$timeCreated [protected]

The time and date on which this annotation was created.

### 3.2.4.9 Annotation::\$transcriptionEng [protected]

The English translation of the transcribed text.

### 3.2.4.10 Annotation::\$transcriptionOrig [protected]

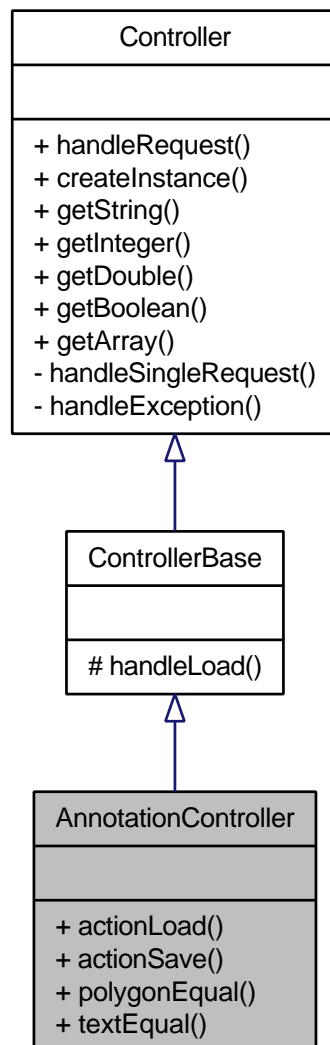
The transcribed text.

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

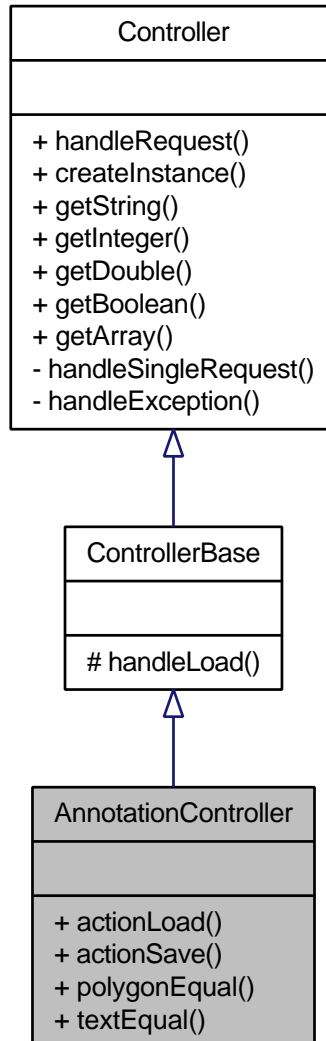
- [www/backend/models/annotation/annotation.php](http://www/backend/models/annotation/annotation.php)

### 3.3 AnnotationController Class Reference

Inheritance diagram for AnnotationController:



Collaboration diagram for AnnotationController:



## Public Member Functions

- [actionLoad](#) (\$data)
- [actionSave](#) (\$data)

## Static Public Member Functions

- static [polygonEqual](#) (\$a, \$b)
- static [textEqual](#) (\$a, \$b)

### 3.3.1 Detailed Description

[Annotation](#) controller class.

### 3.3.2 Member Function Documentation

#### 3.3.2.1 AnnotationController::actionLoad (\$ *data*)

Loads annotations.

#### 3.3.2.2 AnnotationController::actionSave (\$ *data*)

Saves annotations.

#### 3.3.2.3 static AnnotationController::polygonEqual (\$ *a*, \$ *b*) [**static**]

Compares two polygons for equality with delta.

#### 3.3.2.4 static AnnotationController::textEqual (\$ *a*, \$ *b*) [**static**]

Compares two texts for equality, ignoring line ending differences and trailing/leading whitespace.

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

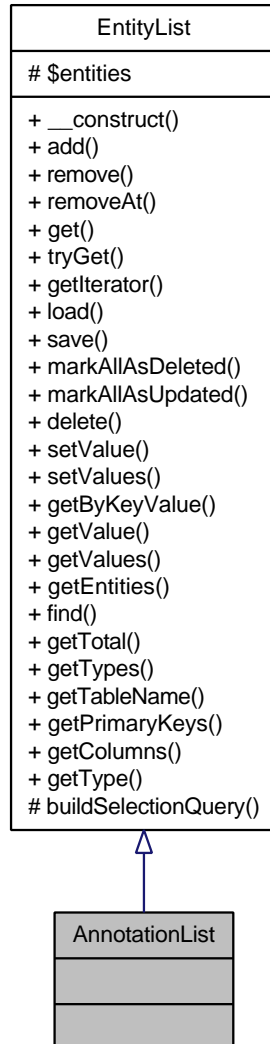
- [www/backend/controllers/annotationcontroller.php](#)

### 3.4 AnnotationList Class Reference

Inheritance diagram for AnnotationList:



Collaboration diagram for AnnotationList:



### 3.4.1 Detailed Description

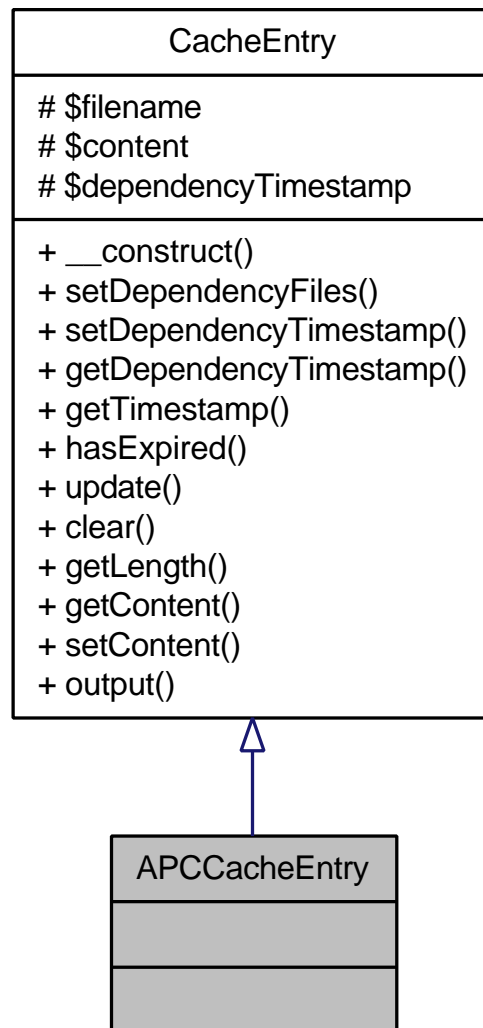
Class representing an annotation entity list.

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

- [www/backend/models/annotation/annotationlist.php](http://www/backend/models/annotation/annotationlist.php)

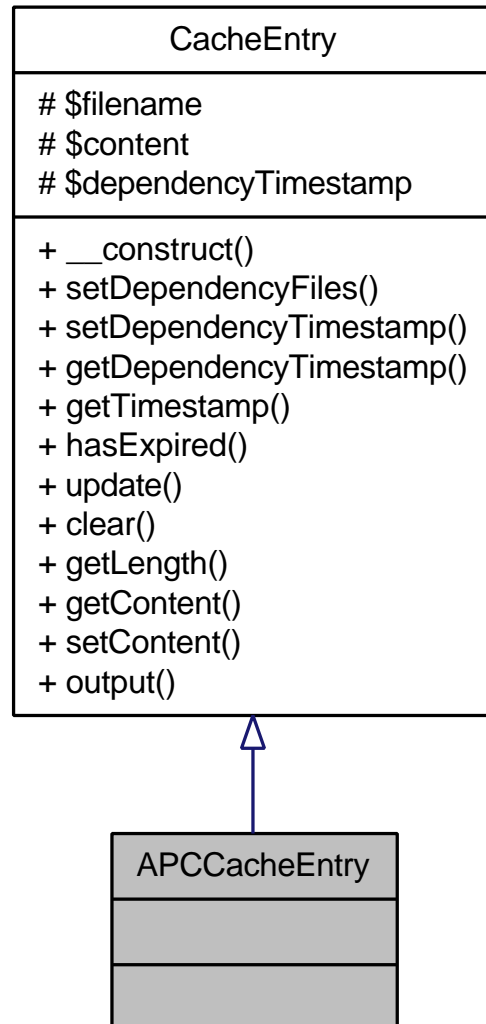
### 3.5 APCCacheEntry Class Reference

Inheritance diagram for APCCacheEntry:





Collaboration diagram for APCCacheEntry:



### 3.5.1 Detailed Description

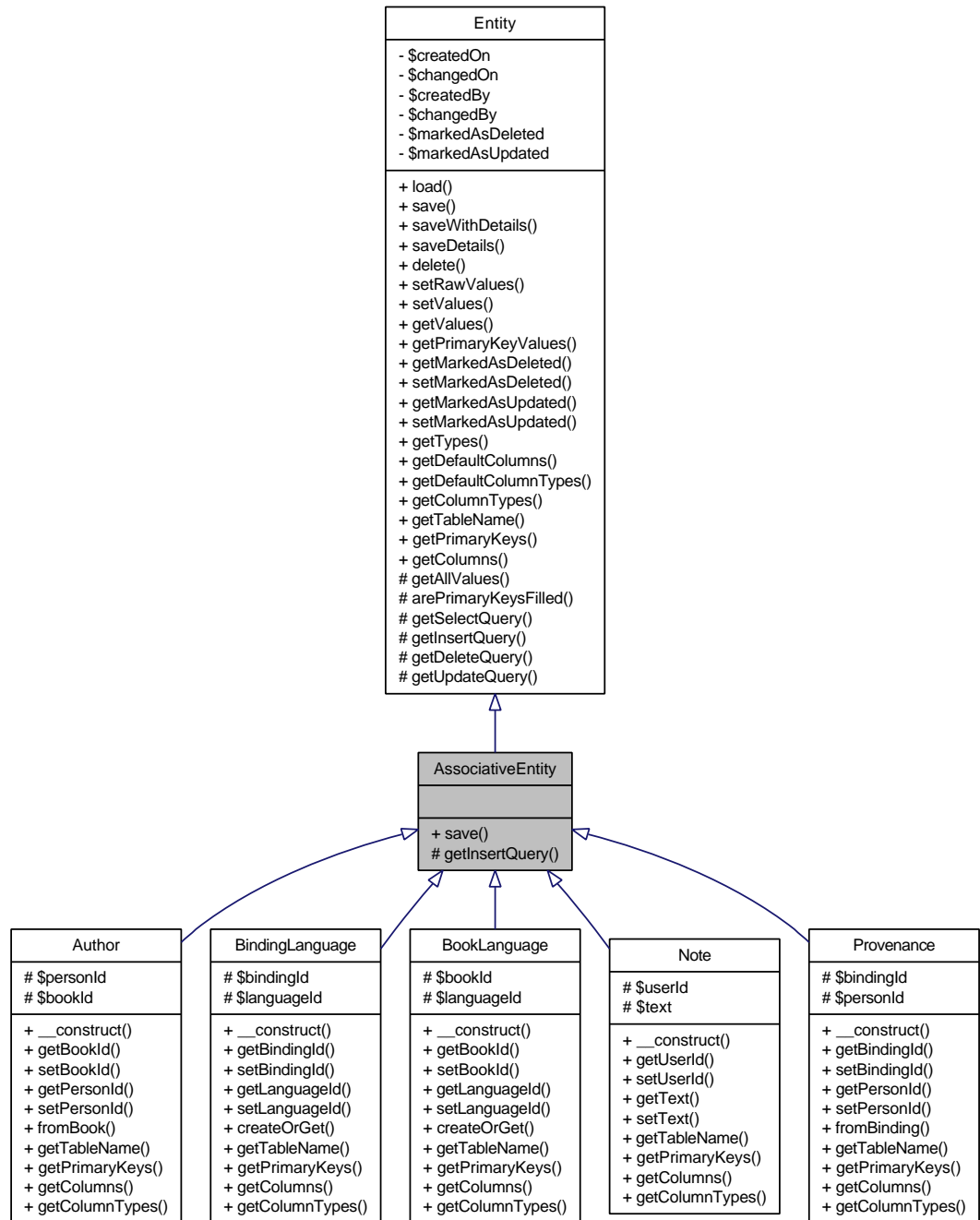
APC cache entry class.

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

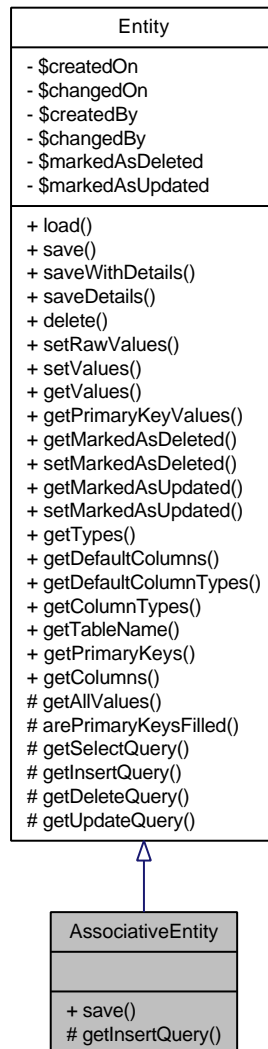
- [www/backend/framework/util/cache.php](http://www/backend/framework/util/cache.php)

### 3.6 AssociativeEntity Class Reference

Inheritance diagram for AssociativeEntity:



Collaboration diagram for AssociativeEntity:



## Public Member Functions

- [save\(\)](#)

## Protected Member Functions

- [getInsertQuery\(\)](#) (\$returning=false)

### 3.6.1 Detailed Description

A specific kind of entity representing 'associative' tables.

A table is considered associative if and only if all its primary keys are also foreign keys and are not automatically generated when doing an insertion.

### 3.6.2 Member Function Documentation

#### 3.6.2.1 `AssociativeEntity::getInsertQuery ($ returning = false)` `[protected]`

Returns the query needed to insert this entity into the database.

##### Parameters

*\$returning* If true, a returning clause is added to the query which returns all primary keys.

##### Returns

[Query](#) to insert this entity in the database.

Reimplemented from [Entity](#).

#### 3.6.2.2 `AssociativeEntity::save ()`

Overloads [save\(\)](#) in such a manner that primary keys should always be set and that a new entry will be created if it does not exist yet.

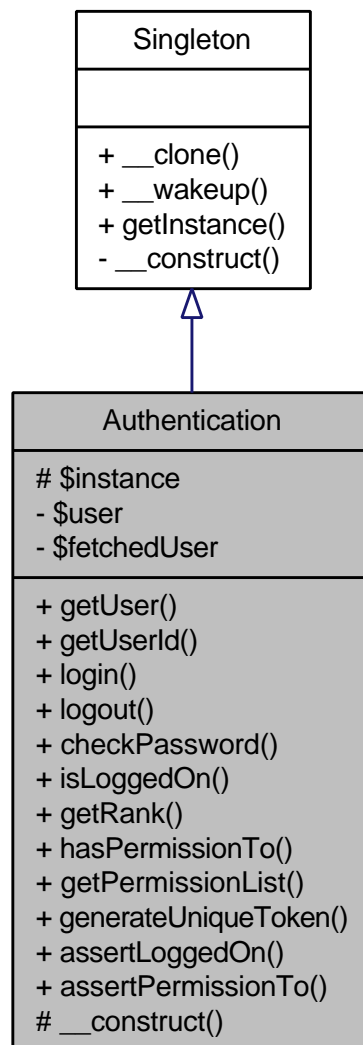
Reimplemented from [Entity](#).

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

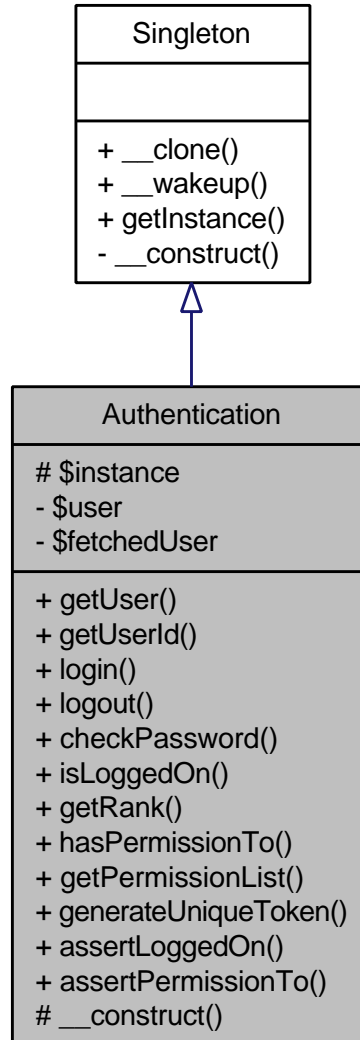
- `www/backend/framework/database/assocentity.php`

## 3.7 Authentication Class Reference

Inheritance diagram for Authentication:



Collaboration diagram for Authentication:



## Public Member Functions

- [getUser \(\)](#)
- [getUserId \(\)](#)
- [login \(\\$username, \\$password\)](#)
- [logout \(\)](#)
- [checkPassword \(\\$password\)](#)
- [isLoggedIn \(\)](#)
- [getRank \(\)](#)
- [hasPermissionTo \(\\$action\)](#)
- [getPermissionList \(\)](#)

## Static Public Member Functions

- static [generateUniqueToken](#) ()
- static [assertLoggedIn](#) ()
- static [assertPermissionTo](#) (\$action)

## Protected Member Functions

- [\\_\\_construct](#) ()

## Static Protected Attributes

- static [\\$instance](#)

### 3.7.1 Detailed Description

[Authentication](#) utility class.

### 3.7.2 Constructor & Destructor Documentation

#### 3.7.2.1 [Authentication::\\_\\_construct](#) () [protected]

Constructs an authentication class instance.

Reimplemented from [Singleton](#).

### 3.7.3 Member Function Documentation

#### 3.7.3.1 static [Authentication::assertLoggedIn](#) () [static]

Asserts that the user is logged on.

Throws an [NotLoggedInError](#) in all other cases.

#### 3.7.3.2 static [Authentication::assertPermissionTo](#) (\$ action) [static]

Asserts that the currently logged in user, if any, has permission to perform the specified action.

#### Parameters

*string* \$action The name of the action, there should be an entry with it in the Permissions table.

#### Exceptions

[AccessDeniedException](#) If the current user has no permission.

**3.7.3.3 Authentication::checkPassword (\$ *password*)**

Checks a password of a user.

**Parameters**

*\$password* The password of the user to check.

**Returns**

true or false.

**3.7.3.4 static Authentication::generateUniqueToken () [static]**

Generates a unique token.

**Returns**

string A string representing a unique 32-digit hexadecimal number.

**3.7.3.5 Authentication::getPermissionList ()**

Returns an array of all actions the currently logged in user is allowed to perform.

**Returns**

array An array of strings representing action names.

**3.7.3.6 Authentication::getRank ()**

Gets the rank of the current user.

**Returns**

The rank of the current user.

**3.7.3.7 Authentication::getUser ()**

Gets the currently logged on user.

**Returns**

The currently logged on user. Null if no user is logged on.



### 3.7.3.8 Authentication::getId ()

Gets the user id of the currently logged on user.

#### Returns

The user id.

### 3.7.3.9 Authentication::hasPermissionTo (\$ action)

Indicates whether the currently logged in user, if any has permission to permorm the action with the specified name. If no users are logged in, it is checked whether guests are allowed to perform the specified action.

#### Parameters

*string* \$action The name of the action, there should be an entry with it in the Permissions table.

#### Returns

bool Whether the current user has a rank high enough dor this action.

### 3.7.3.10 Authentication::isLoggedIn ()

Checks whether a user is logged on.

#### Returns

Whether a user is logged on.

### 3.7.3.11 Authentication::login (\$ username, \$ password)

Logs a user in.

#### Parameters

*\$username* The username of the user to log in.

*\$password* The password of the user to log in.

#### Returns

The new user.

#### Exceptions

*UserBannedException* If the user is banned.

*UserPendingException* If the user is not yet activated.

*UserNotFoundException* If no user with the provided username/password combination exists.

### 3.7.3.12 Authentication::logout ()

Logs the current user out.

## 3.7.4 Member Data Documentation

### 3.7.4.1 Authentication::\$instance [**static**, **protected**]

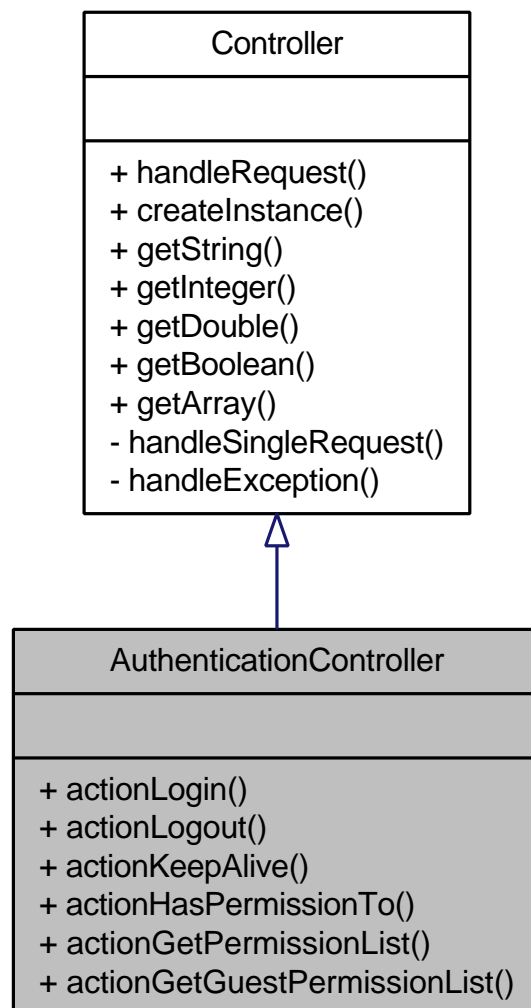
Unique instance.

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

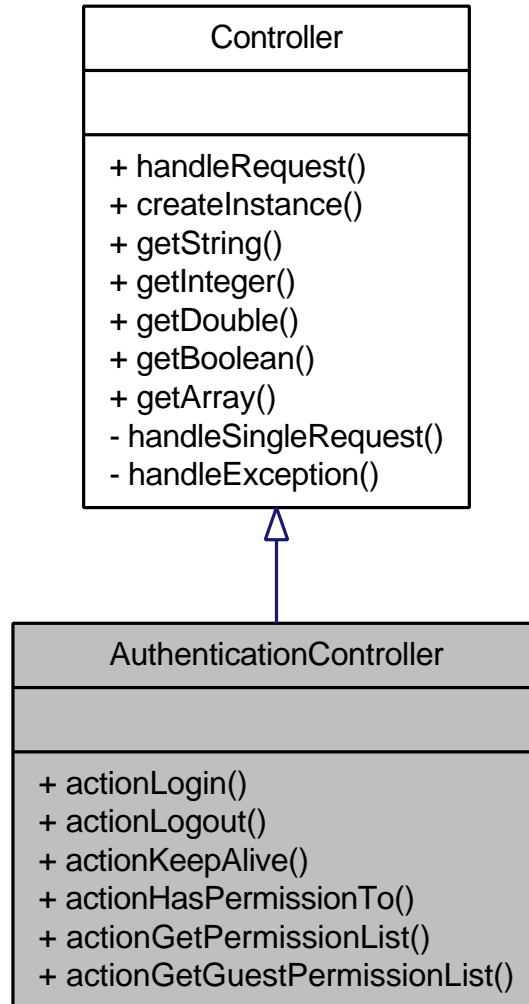
- [www/backend/util/authentication.php](http://www/backend/util/authentication.php)

## 3.8 AuthenticationController Class Reference

Inheritance diagram for AuthenticationController:



Collaboration diagram for AuthenticationController:



### Public Member Functions

- [actionLogin](#) (\$data)
- [actionLogout](#) (\$data)
- [actionKeepAlive](#) (\$data)
- [actionHasPermissionTo](#) (\$data)
- [actionGetPermissionList](#) ()
- [actionGetGuestPermissionList](#) ()

### 3.8.1 Detailed Description

[Authentication](#) controller class.

## 3.8.2 Member Function Documentation

### 3.8.2.1 AuthenticationController::actionGetGuestPermissionList ()

Returns a list of a guest's permissions

### 3.8.2.2 AuthenticationController::actionGetPermissionList ()

Returns a list of the user's permissions

### 3.8.2.3 AuthenticationController::actionHasPermissionTo (\$ *data*)

Checks whether a user has permission to do something

### 3.8.2.4 AuthenticationController::actionKeepAlive (\$ *data*)

Checks whether to keep alive auser and sends back the results

### 3.8.2.5 AuthenticationController::actionLogin (\$ *data*)

Login

### 3.8.2.6 AuthenticationController::actionLogout (\$ *data*)

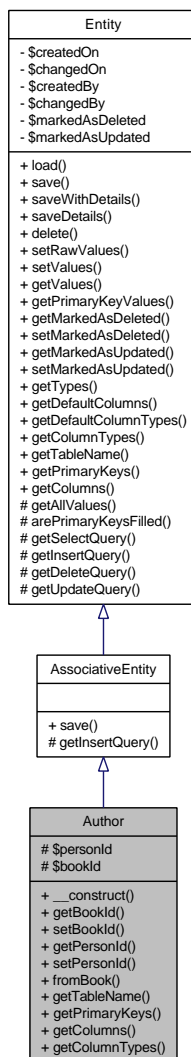
Logout

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

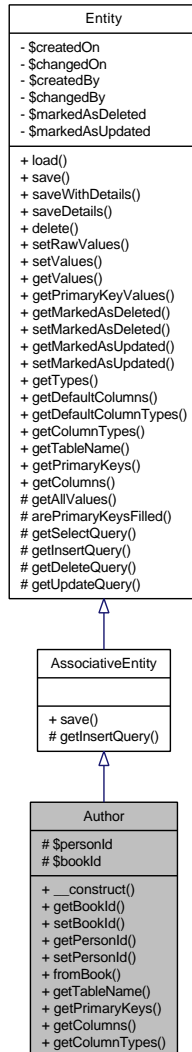
- [www/backend/controllers/authenticationcontroller.php](#)

### 3.9 Author Class Reference

Inheritance diagram for Author:



Collaboration diagram for Author:



## Public Member Functions

- `__construct` (\$personId=null, \$bookId=null, \$createNew=false)
- `getBookId` ()
- `setBookId` (\$bookId)
- `getPersonId` ()
- `setPersonId` (\$personId)

## Static Public Member Functions

- static `fromBook` (\$book)

- static [getTableName \(\)](#)
- static [getPrimaryKeys \(\)](#)
- static [getColumns \(\)](#)
- static [getColumnTypes \(\)](#)

## Protected Attributes

- [\\$personId](#)
- [\\$bookId](#)

### 3.9.1 Detailed Description

Class representing a author entity. Associatieve between book and person.

### 3.9.2 Constructor & Destructor Documentation

#### 3.9.2.1 `Author::__construct ($personId = null, $bookId = null, $createNew = false)`

Constructs an author entity by a person and book id.

#### Parameters

*\$personId*

*\$bookId*

*\$createNew* If true, a new author will be created if the specified one did not exist yet. If one already exists, it doesn't really matter whether this is true or false.

### 3.9.3 Member Function Documentation

#### 3.9.3.1 `static Author::fromBook ($book) [static]`

Returns all the authors which belong to one book

#### Parameters

*\$book* The book model

#### Returns

Array of author models

#### 3.9.3.2 `Author::getBookId ()`

Getters and setters.



### 3.9.3.3 static Author::getColumns () [static]

Gets all the columns.

#### Returns

Array of all columns, except primary keys.

Reimplemented from [Entity](#).

### 3.9.3.4 static Author::getColumnTypes () [static]

Gets all the column types, per column, including primary keys.

#### Returns

Array of all column types.

Reimplemented from [Entity](#).

### 3.9.3.5 static Author::getPrimaryKeys () [static]

Gets the primary keys.

#### Returns

Array of all primary keys.

Reimplemented from [Entity](#).

### 3.9.3.6 static Author::getTableName () [static]

Gets the table name.

#### Returns

The table name.

Reimplemented from [Entity](#).

## 3.9.4 Member Data Documentation

### 3.9.4.1 Author::\$bookId [protected]

[Book](#) id.

#### 3.9.4.2 Author::\$personId [protected]

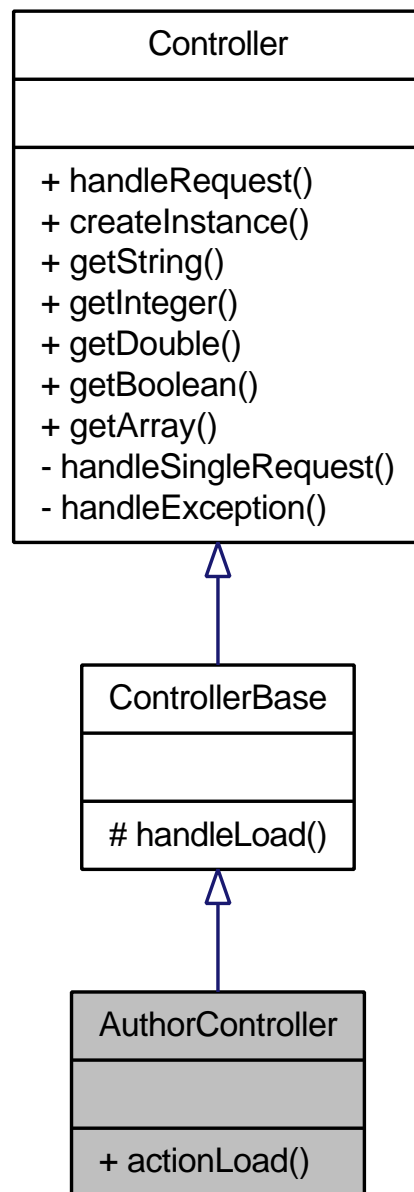
[Person](#) id.

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

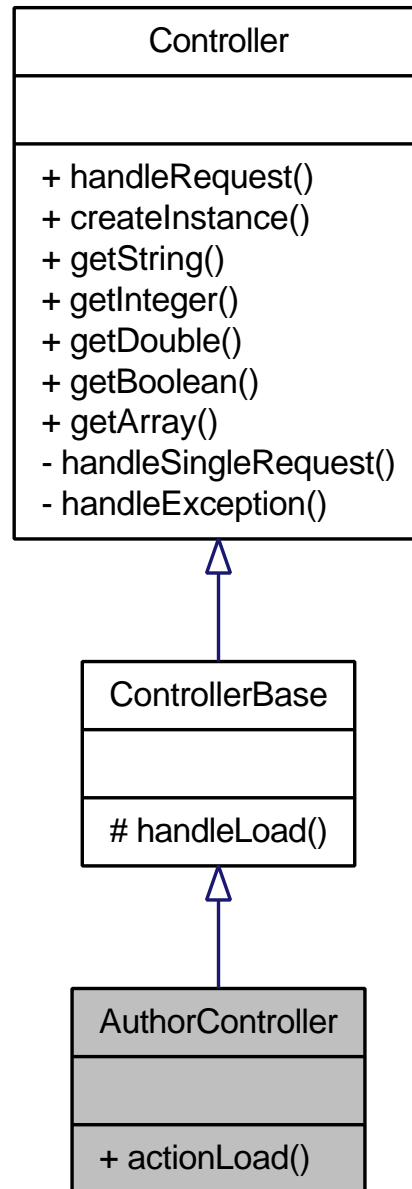
- [www/backend/models/author/author.php](#)

## 3.10 AuthorController Class Reference

Inheritance diagram for AuthorController:



Collaboration diagram for AuthorController:



## Public Member Functions

- [actionLoad](#) (\$data)

### 3.10.1 Detailed Description

[Author](#) controller class.

### 3.10.2 Member Function Documentation

#### 3.10.2.1 AuthorController::actionLoad (\$ *data*)

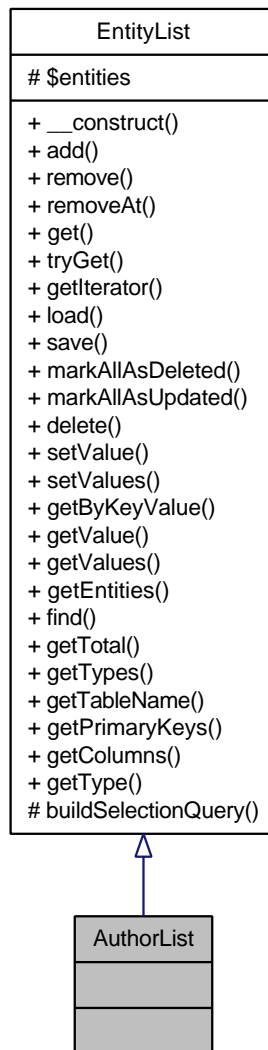
Loads authors.

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

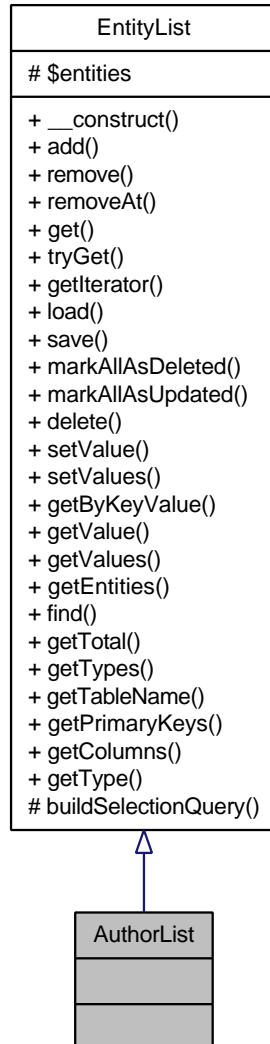
- `www/backend/controllers/authorcontroller.php`

### 3.11 AuthorList Class Reference

Inheritance diagram for AuthorList:



Collaboration diagram for AuthorList:



### 3.11.1 Detailed Description

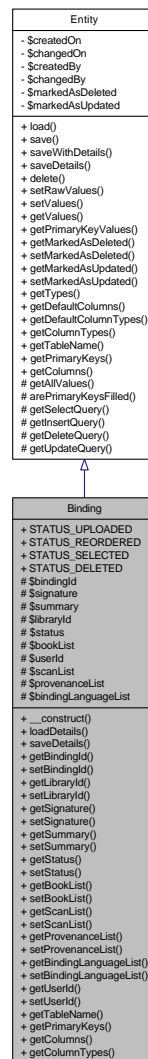
Class representing an author entity list.

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

- [www/backend/models/author/authorlist.php](http://www/backend/models/author/authorlist.php)

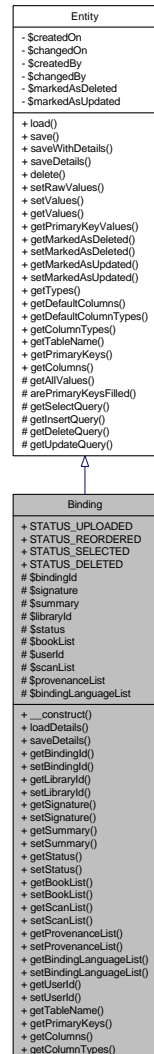
### 3.12 Binding Class Reference

Inheritance diagram for Binding:





Collaboration diagram for Binding:



## Public Member Functions

- [\\_\\_construct](#) (\$id=null)
- [loadDetails](#) ()
- [saveDetails](#) ()
- [getBindingId](#) ()
- [setBindingId](#) (\$bindingId)
- [getLibraryId](#) ()
- [setLibraryId](#) (\$libraryId)
- [getSignature](#) ()
- [setSignature](#) (\$signature)

- **getSummary** ()
- **setSummary** (\$summary)
- **getStatus** ()
- **setStatus** (\$status)
- **getBookList** ()
- **setBookList** (\$bookList)
- **getScanList** ()
- **setScanList** (\$scanList)
- **getProvenanceList** ()
- **setProvenanceList** (\$provenanceList)
- **getBindingLanguageList** ()
- **setBindingLanguageList** (\$bindingLanguageList)
- **getUserId** ()
- **setUserId** (\$userId)

### Static Public Member Functions

- static [getTableName](#) ()
- static [getPrimaryKeys](#) ()
- static [getColumns](#) ()
- static [getColumnTypes](#) ()

### Public Attributes

- const [STATUS\\_UPLOADED](#) = 0
- const [STATUS\\_REORDERED](#) = 1
- const [STATUS\\_SELECTED](#) = 2
- const [STATUS\\_DELETED](#) = 3

### Protected Attributes

- [\\$bindingId](#)
- [\\$signature](#)
- [\\$summary](#)
- [\\$libraryId](#)
- [\\$status](#)
- [\\$bookList](#)
- [\\$userId](#)
- [\\$scanList](#)
- [\\$provenanceList](#)
- [\\$bindingLanguageList](#)

#### 3.12.1 Detailed Description

Class representing a binding entity.

## 3.12.2 Constructor & Destructor Documentation

### 3.12.2.1 Binding::\_\_construct (\$ *id* = null)

Constructs a binding by id.

#### Parameters

*\$id* Id of the binding. Default (null) will create a new binding.

## 3.12.3 Member Function Documentation

### 3.12.3.1 Binding::getBindingId ()

Getters and setters.

### 3.12.3.2 static Binding::getColumns () [static]

Gets all the columns.

#### Returns

Array of all columns, except primary keys.

Reimplemented from [Entity](#).

### 3.12.3.3 static Binding::getColumnTypes () [static]

Gets all the column types, per column, including primary keys.

#### Returns

Array of all column types.

Reimplemented from [Entity](#).

### 3.12.3.4 static Binding::getPrimaryKeys () [static]

Gets the primary keys.

#### Returns

Array of all primary keys.

Reimplemented from [Entity](#).

**3.12.3.5 static Binding::getTableName () [static]**

Gets the table name.

**Returns**

The table name.

Reimplemented from [Entity](#).

**3.12.3.6 Binding::loadDetails ()**

Loads all associated entity lists.

**3.12.3.7 Binding::saveDetails ()**

Saves all associated entity lists.

Reimplemented from [Entity](#).

**3.12.4 Member Data Documentation****3.12.4.1 Binding::\$bindingId [protected]**

Id of this binding.

**3.12.4.2 Binding::\$bindingLanguageList [protected]**

List of all languages of the annotations in this book.

**3.12.4.3 Binding::\$bookList [protected]**

Books for this binding.

**3.12.4.4 Binding::\$libraryId [protected]**

[Library](#)

**3.12.4.5 Binding::\$provenanceList [protected]**

List of all provenances for this book.

**3.12.4.6 Binding::\$scanList [protected]**

List of all scans for this book.

**3.12.4.7 Binding::\$signature [protected]**

Signature of the binding

**3.12.4.8 Binding::\$status [protected]**

[Binding](#) status

**3.12.4.9 Binding::\$summary [protected]**

Summary of the contents of the binding.

**3.12.4.10 Binding::\$userId [protected]**

[User](#) this binding was created by.

**3.12.4.11 const Binding::STATUS\_UPLOADED = 0**

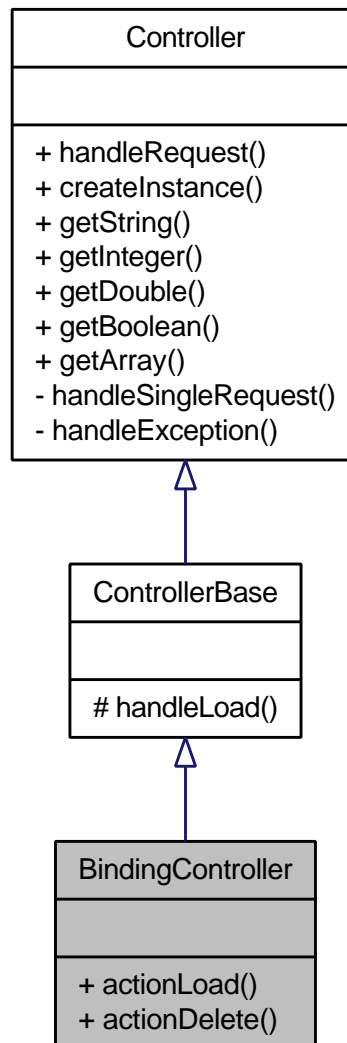
[Binding](#) status constants.

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

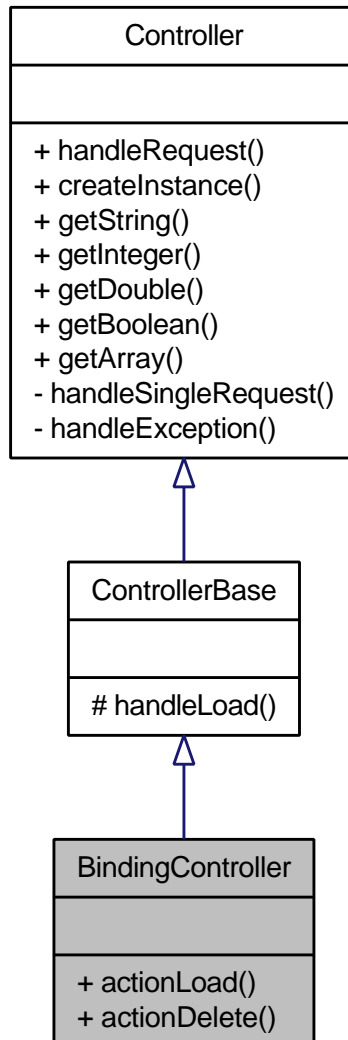
- [www/backend/models/binding/binding.php](#)

### 3.13 BindingController Class Reference

Inheritance diagram for BindingController:



Collaboration diagram for BindingController:



### Public Member Functions

- [actionLoad](#) (\$data)
- [actionDelete](#) (\$data)

#### 3.13.1 Detailed Description

[Binding](#) controller class.

### 3.13.2 Member Function Documentation

#### 3.13.2.1 BindingController::actionDelete (\$ *data*)

Deletes a binding

#### 3.13.2.2 BindingController::actionLoad (\$ *data*)

Loads bindings.

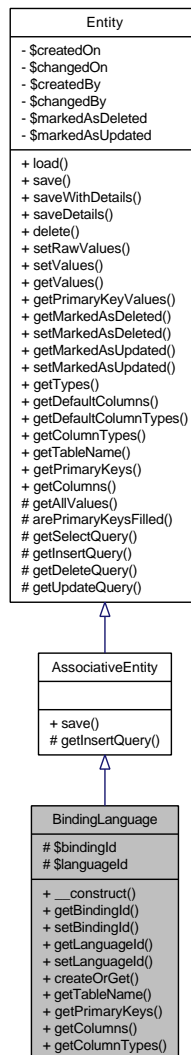
The documentation for this class was generated from the following file:

- [www/backend/controllers/bindingcontroller.php](http://www/backend/controllers/bindingcontroller.php)

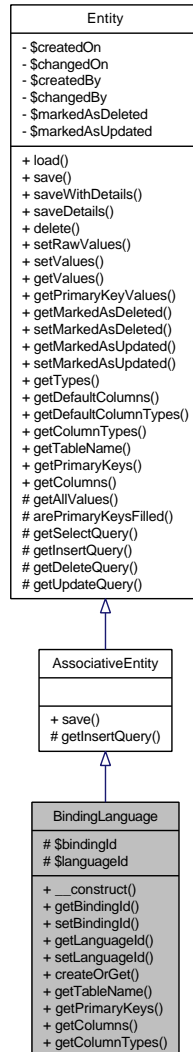


## 3.14 BindingLanguage Class Reference

Inheritance diagram for BindingLanguage:



Collaboration diagram for BindingLanguage:



## Public Member Functions

- `__construct` (\$bindingId=null, \$languageId=null, \$createNew=false)
- `getBindingId` ()
- `setBindingId` (\$bindingId)
- `getLanguageId` ()
- `setLanguageId` (\$id)

## Static Public Member Functions

- static `createOrGet` (\$binding, \$language)

- static [getTableName](#) ()
- static [getPrimaryKeys](#) ()
- static [getColumns](#) ()
- static [getColumnTypes](#) ()

## Protected Attributes

- [\\$bindingId](#)
- [\\$languageId](#)

### 3.14.1 Detailed Description

Associative entity representing the language of a binding.

### 3.14.2 Constructor & Destructor Documentation

#### 3.14.2.1 BindingLanguage::\_\_construct (\$ *bindingId* = null, \$ *languageId* = null, \$ *createNew* = false)

Constructs a binding-language relation.

#### Parameters

*\$bindingId*

*\$languageId*

*\$createNew* If true, a new relation will be created if the specified one did not exist yet. If one already exists, it doesn't really matter whether this is true or false.

### 3.14.3 Member Function Documentation

#### 3.14.3.1 static BindingLanguage::createOrGet (\$ *binding*, \$ *language*) [static]

Creates a new relation between a binding and a language or retrieves one if it already exists.

The id's of the binding and the language should be set.

#### Parameters

*Binding* \$binding

*Language* \$language

#### 3.14.3.2 BindingLanguage::getBindingId ()

Getters and setters.

**3.14.3.3 static BindingLanguage::getColumns () [static]**

Gets all the columns.

**Returns**

Array of all columns, except primary keys.

Reimplemented from [Entity](#).

**3.14.3.4 static BindingLanguage::getColumnTypes () [static]**

Gets all the column types, per column, including primary keys.

**Returns**

Array of all column types.

Reimplemented from [Entity](#).

**3.14.3.5 static BindingLanguage::getPrimaryKeys () [static]**

Gets the primary keys.

**Returns**

Array of all primary keys.

Reimplemented from [Entity](#).

**3.14.3.6 static BindingLanguage::getTableName () [static]**

Gets the table name.

**Returns**

The table name.

Reimplemented from [Entity](#).

**3.14.4 Member Data Documentation****3.14.4.1 BindingLanguage::\$bindingId [protected]**

[Binding](#).

**3.14.4.2 BindingLanguage::\$languageId [protected]**

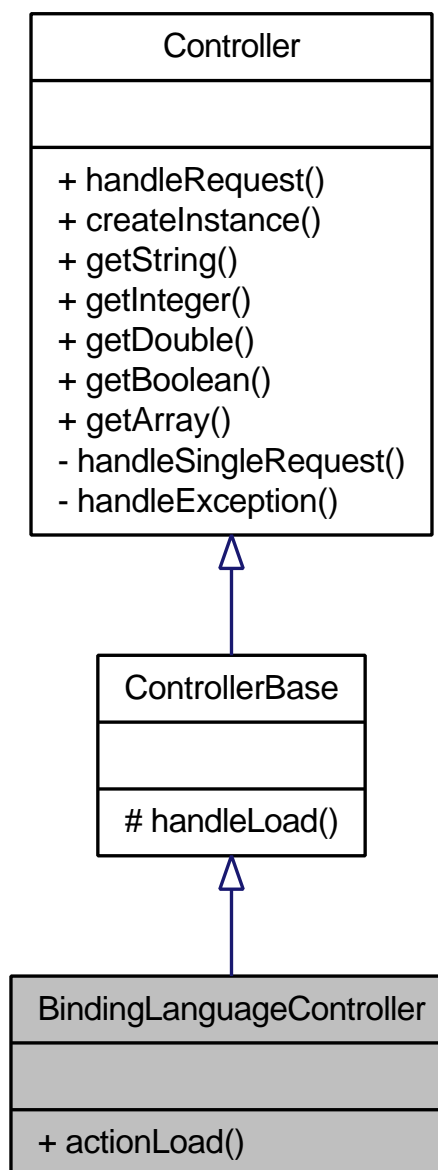
[Language](#).

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

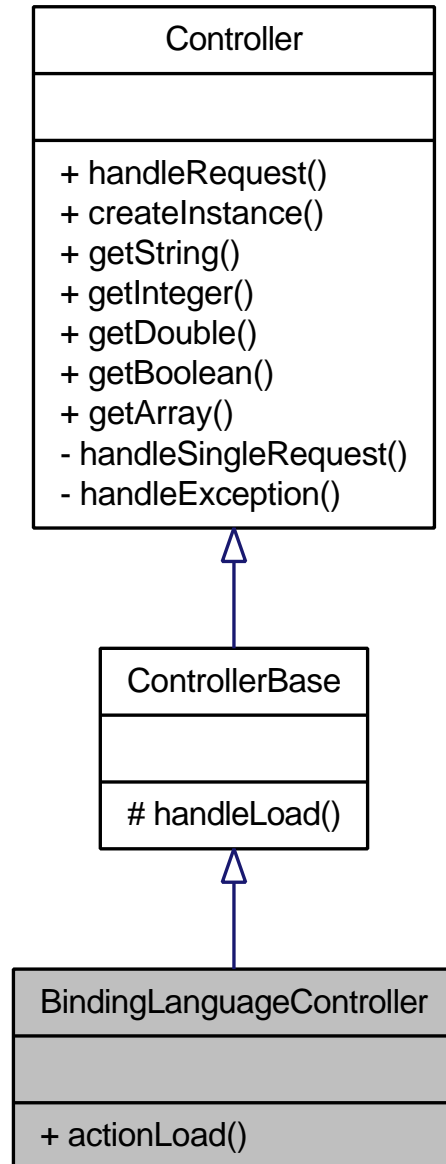
- [www/backend/models/language/bindinglanguage.php](http://www/backend/models/language/bindinglanguage.php)

### 3.15 BindingLanguageController Class Reference

Inheritance diagram for BindingLanguageController:



Collaboration diagram for BindingLanguageController:



## Public Member Functions

- [actionLoad](#) (\$data)

### 3.15.1 Detailed Description

[Binding](#) controller class.

### 3.15.2 Member Function Documentation

#### 3.15.2.1 BindingLanguageController::actionLoad (\$ *data*)

Loads provenances.

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

- [www/backend/controllers/bindinglanguagecontroller.php](#)



## 3.16 BindingLanguageList Class Reference

Inheritance diagram for BindingLanguageList:



Collaboration diagram for BindingLanguageList:



### 3.16.1 Detailed Description

Class representing a binding language entity list.

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

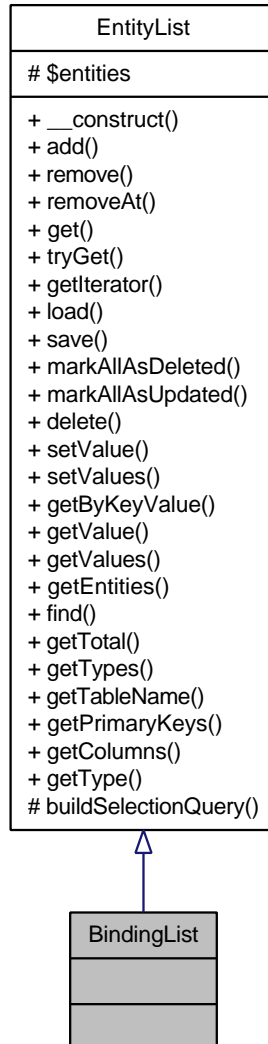
- [www/backend/models/language/bindinglanguage.php](http://www/backend/models/language/bindinglanguage.php)

## 3.17 BindingList Class Reference

Inheritance diagram for BindingList:



Collaboration diagram for BindingList:



### 3.17.1 Detailed Description

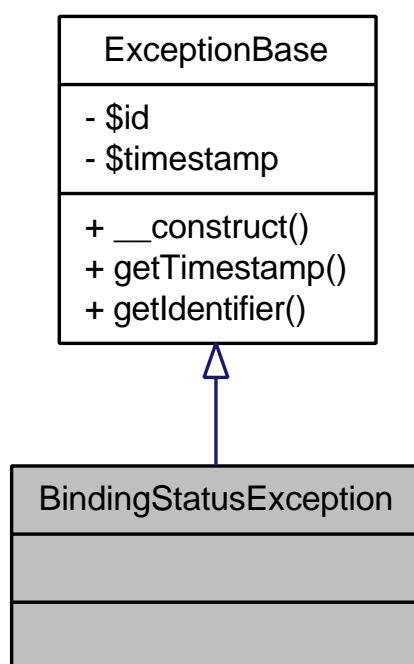
Class representing a binding entity list.

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

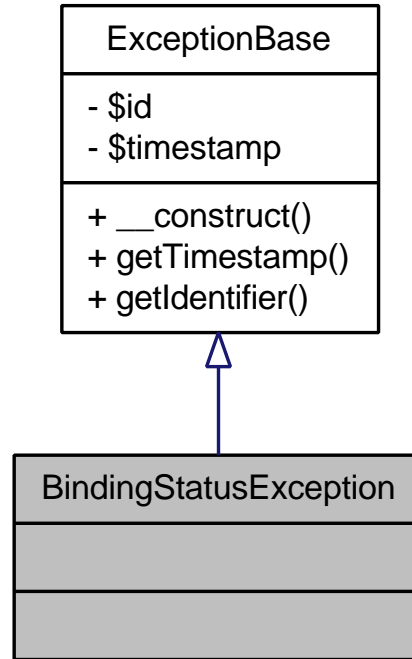
- [www/backend/models/binding/bindinglist.php](http://www/backend/models/binding/bindinglist.php)

## 3.18 BindingStatusException Class Reference

Inheritance diagram for BindingStatusException:



Collaboration diagram for BindingStatusException:



### 3.18.1 Detailed Description

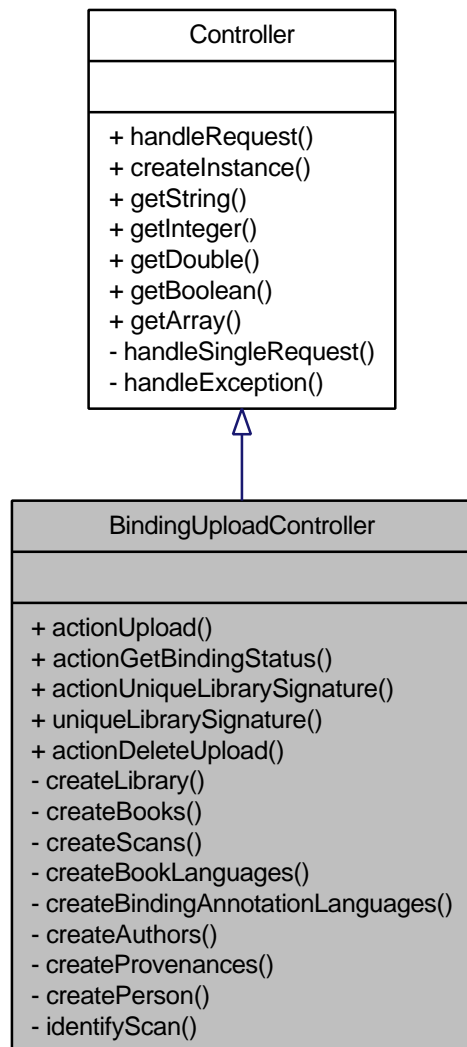
Exceptions.

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

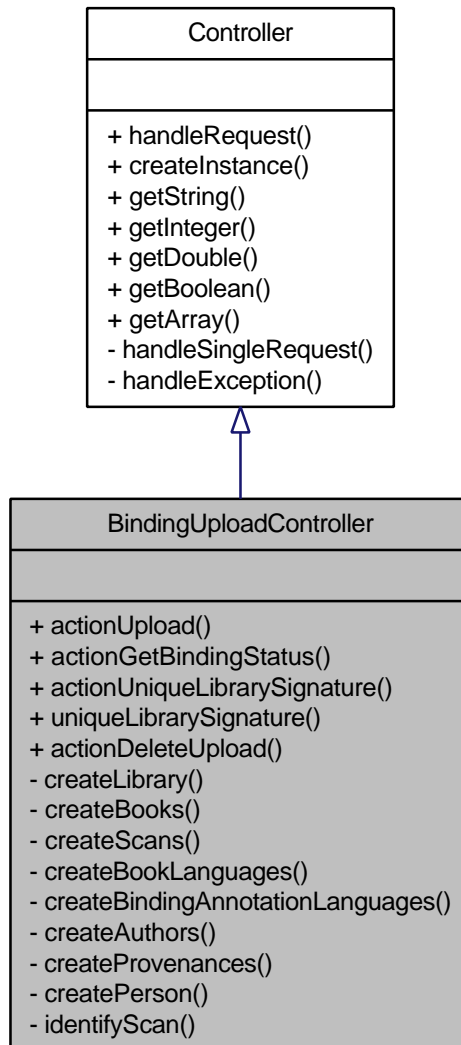
- `www/backend/controllers/bindinguploadcontroller.php`

## 3.19 BindingUploadController Class Reference

Inheritance diagram for BindingUploadController:



Collaboration diagram for BindingUploadController:



## Public Member Functions

- [actionUpload](#) (\$data)
- [actionGetBindingStatus](#) (\$data)
- [actionUniqueLibrarySignature](#) (\$data)
- [uniqueLibrarySignature](#) (\$libraryName, \$signature, \$bindingId)
- [actionDeleteUpload](#) (\$data)

### 3.19.1 Detailed Description

[Binding](#) upload controller class.



## 3.19.2 Member Function Documentation

### 3.19.2.1 BindingUploadController::actionDeleteUpload (\$ *data*)

Deletes a binding.

### 3.19.2.2 BindingUploadController::actionGetBindingStatus (\$ *data*)

Returns the binding and it's status which the user is currently uploading/modifying

### 3.19.2.3 BindingUploadController::actionUniqueLibrarySignature (\$ *data*)

Returns whether the combination of library and signature in the sent data is unique.

### 3.19.2.4 BindingUploadController::actionUpload (\$ *data*)

Uploads a binding.

### 3.19.2.5 BindingUploadController::uniqueLibrarySignature (\$ *libraryName*, \$ *signature*, \$ *bindingId*)

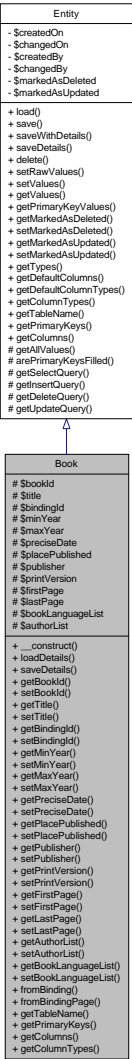
Returns whether the inputcombination of library and signature is unique.

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

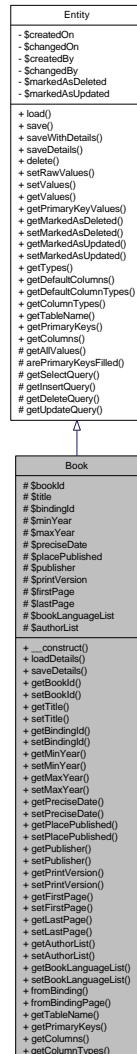
- [www/backend/controllers/bindinguploadcontroller.php](#)

### 3.20 Book Class Reference

Inheritance diagram for Book:



Collaboration diagram for Book:



## Public Member Functions

- `__construct ($id=null)`
- `loadDetails ()`
- `saveDetails ()`
- `getBookId ()`
- `setBookId ($bookId)`
- `getTitle ()`
- `setTitle ($title)`
- `getBindingId ()`
- `setBindingId ($bindingId)`

- **getMinYear** ()
- **setMinYear** (\$minYear)
- **getMaxYear** ()
- **setMaxYear** (\$maxYear)
- **getPreciseDate** ()
- **setPreciseDate** (\$preciseDate)
- **getPlacePublished** ()
- **setPlacePublished** (\$placePublished)
- **getPublisher** ()
- **setPublisher** (\$publisher)
- **getPrintVersion** ()
- **setPrintVersion** (\$printVersion)
- **getFirstPage** ()
- **setFirstPage** (\$page)
- **getLastPage** ()
- **setLastPage** (\$page)
- **getAuthorList** ()
- **setAuthorList** (\$authorList)
- **getBookLanguageList** ()
- **setBookLanguageList** (\$bookLanguageList)

### Static Public Member Functions

- static [fromBinding](#) (\$binding)
- static [fromBindingPage](#) (\$binding, \$range)
- static [getTableName](#) ()
- static [getPrimaryKeys](#) ()
- static [getColumns](#) ()
- static [getColumnTypes](#) ()

### Protected Attributes

- [\\$bookId](#)
- [\\$title](#)
- [\\$bindingId](#)
- [\\$minYear](#)
- [\\$maxYear](#)
- [\\$preciseDate](#)
- [\\$placePublished](#)
- [\\$publisher](#)
- [\\$printVersion](#)
- [\\$firstPage](#)
- [\\$lastPage](#)
- [\\$bookLanguageList](#)
- [\\$authorList](#)

### 3.20.1 Detailed Description

Class representing a book entity.

### 3.20.2 Constructor & Destructor Documentation

#### 3.20.2.1 Book::\_\_construct (\$ *id* = null)

Constructs a book by id.

##### Parameters

*\$id* Id of the book. Default (null) will create a new book.

### 3.20.3 Member Function Documentation

#### 3.20.3.1 static Book::fromBinding (\$ *binding*) [static]

Returns all the books from one binding

##### Parameters

*\$binding* The binding model

##### Returns

Array of book models

#### 3.20.3.2 static Book::fromBindingPage (\$ *binding*, \$ *range*) [static]

Returns all the books from one binding in a certain page range

##### Parameters

*\$book* The book model

*\$range* The page range

##### Returns

Array of book models

#### 3.20.3.3 Book::getBookId ()

Getters and setters.

**3.20.3.4 static Book::getColumns () [static]**

Gets all the columns.

**Returns**

Array of all columns, except primary keys.

Reimplemented from [Entity](#).

**3.20.3.5 static Book::getColumnTypes () [static]**

Gets all the column types, per column, including primary keys.

**Returns**

Array of all column types.

Reimplemented from [Entity](#).

**3.20.3.6 static Book::getPrimaryKeys () [static]**

Gets the primary keys.

**Returns**

Array of all primary keys.

Reimplemented from [Entity](#).

**3.20.3.7 static Book::getTableName () [static]**

Gets the table name.

**Returns**

The table name.

Reimplemented from [Entity](#).

**3.20.3.8 Book::loadDetails ()**

Loads all the associated lists.

**3.20.3.9 Book::saveDetails ()**

Saves all the associated lists.

Reimplemented from [Entity](#).

### 3.20.4 Member Data Documentation

#### 3.20.4.1 Book::\$authorList [protected]

list containing all the authors of this book.

#### 3.20.4.2 Book::\$bindingId [protected]

[Binding](#) identifier of the binding this book is part of.

#### 3.20.4.3 Book::\$bookId [protected]

Id of this book.

#### 3.20.4.4 Book::\$bookLanguageList [protected]

list containing all the languages linked to this book.

#### 3.20.4.5 Book::\$firstPage [protected]

First page.

#### 3.20.4.6 Book::\$lastPage [protected]

Last page.

#### 3.20.4.7 Book::\$maxYear [protected]

Maximum year of publication.

#### 3.20.4.8 Book::\$minYear [protected]

Minimum year of publication.

#### 3.20.4.9 Book::\$placePublished [protected]

Place of publication.

#### 3.20.4.10 Book::\$preciseDate [protected]

Precise date of publication.

**3.20.4.11 Book::\$printVersion [protected]**

Version of the book.

**3.20.4.12 Book::\$publisher [protected]**

Publisher of the book.

**3.20.4.13 Book::\$title [protected]**

Title of this book.

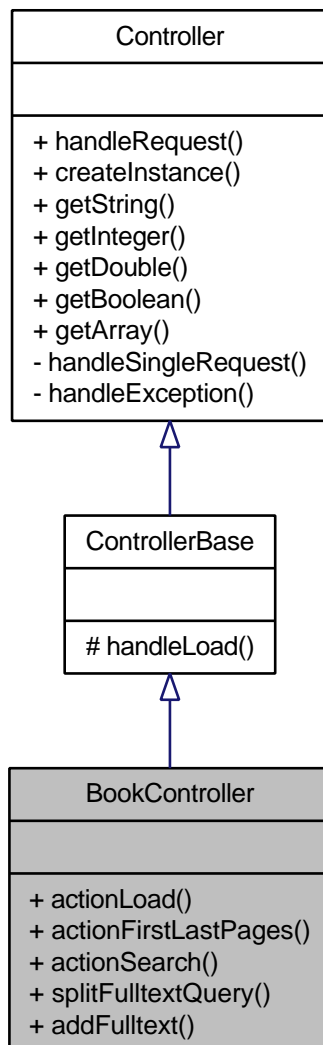
The documentation for this class was generated from the following file:

- [www/backend/models/book/book.php](http://www/backend/models/book/book.php)

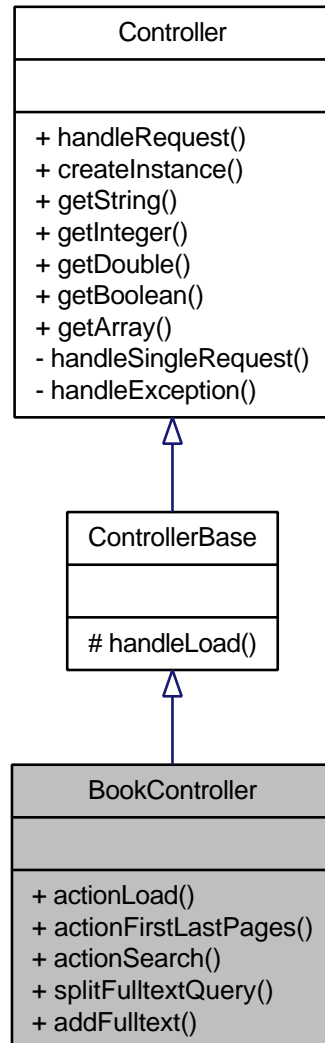


## 3.21 BookController Class Reference

Inheritance diagram for BookController:



Collaboration diagram for BookController:



## Public Member Functions

- [actionLoad](#) (\$data)
- [actionFirstLastPages](#) (\$data)
- [actionSearch](#) (\$data)

## Static Public Member Functions

- static [splitFulltextQuery](#) (\$query)
- static [addFulltext](#) (\$name, \$columns, \$value, \$addheadline, \$saltvector, [Query](#) &\$query, array &\$binds, &\$c, &\$headline)

### 3.21.1 Detailed Description

[Book](#) controller class.

### 3.21.2 Member Function Documentation

#### 3.21.2.1 BookController::actionFirstLastPages (\$ data)

Sets the first and last pages of the books

#### 3.21.2.2 BookController::actionLoad (\$ data)

Loads books.

#### 3.21.2.3 BookController::actionSearch (\$ data)

Searches for books.

#### 3.21.2.4 static BookController::addFulltext (\$ name, \$ columns, \$ value, \$ addheadline, \$ altvector, Query &\$ query, array &\$ binds, &\$ c, &\$ headline) [static]

Adds a fulltext query to the given query.

##### Parameters

**string** \$name The name of this fulltext query (for binding name).

**mixed** \$columns The columns to search on.

**string** \$value The user-supplied search string.

**boolean** \$addheadline Whether to add this fulltext query to the headlines.

**string** \$altvector A (indexed) vector for the given column, if available. Only valid for a single column.

**Query** \$query The query to operate on.

**array** \$binds The current array of bindings.

**int** \$c The current binding counter.

**string** \$headline The current headline query string.

#### 3.21.2.5 static BookController::splitFulltextQuery (\$ query) [static]

Splits a fulltext user-input query into distinct queries.

This method provides with:

- a list of exact (quoted) query parts in Postgres format,

- a fulltext Postgres query string for all other parts,
- a headline fulltext Postgres query string for this query.

**Parameters**

*string* \$query The user specified query, using - for NOT and double quotes for quoting.

**Returns**

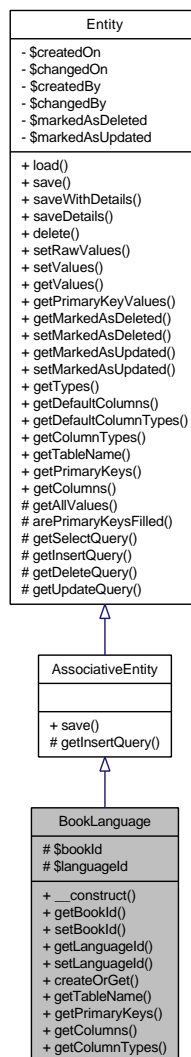
array The split fulltext query.

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

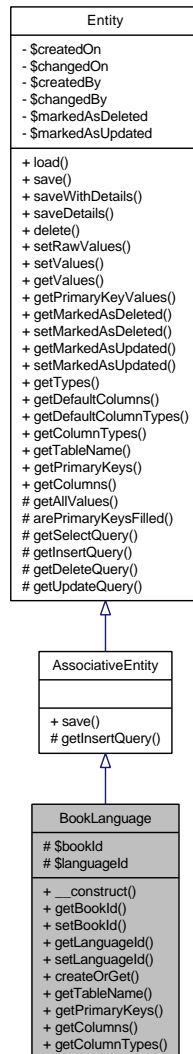
- [www/backend/controllers/bookcontroller.php](http://www/backend/controllers/bookcontroller.php)

## 3.22 BookLanguage Class Reference

Inheritance diagram for BookLanguage:



Collaboration diagram for BookLanguage:



## Public Member Functions

- `__construct` (\$bookId=null, \$languageId=null, \$createNew=false)
- `getBookId` ()
- `setBookId` (\$bookId)
- `getLanguageId` ()
- `setLanguageId` (\$id)

## Static Public Member Functions

- static `createOrGet` (\$book, \$language)

- static [getTableNames \(\)](#)
- static [getPrimaryKeys \(\)](#)
- static [getColumns \(\)](#)
- static [getColumnTypes \(\)](#)

## Protected Attributes

- [\\$bookId](#)
- [\\$languageId](#)

### 3.22.1 Detailed Description

Associative entity representing the language of a book.

### 3.22.2 Constructor & Destructor Documentation

#### 3.22.2.1 BookLanguage::\_\_construct (\$bookId = null, \$languageId = null, \$createNew = false)

Constructs a book-language relation.

#### Parameters

*\$bookId*

*\$languageId*

*\$createNew* If true, a new relation will be created if the specified one did not exist yet. If one already exists, it doesn't really matter whether this is true or false.

### 3.22.3 Member Function Documentation

#### 3.22.3.1 static BookLanguage::createOrGet (\$book, \$language) [static]

Creates a new relation between a book and a language or retrieves one if it already exists.

The id's of the book and the language should be set.

#### Parameters

*Book* \$book

*Language* \$language

#### 3.22.3.2 BookLanguage::getBookId ()

Getters and setters.

**3.22.3.3 static BookLanguage::getColumns () [static]**

Gets all the columns.

**Returns**

Array of all columns, except primary keys.

Reimplemented from [Entity](#).

**3.22.3.4 static BookLanguage::getColumnTypes () [static]**

Gets all the column types, per column, including primary keys.

**Returns**

Array of all column types.

Reimplemented from [Entity](#).

**3.22.3.5 static BookLanguage::getPrimaryKeys () [static]**

Gets the primary keys.

**Returns**

Array of all primary keys.

Reimplemented from [Entity](#).

**3.22.3.6 static BookLanguage::getTableName () [static]**

Gets the table name.

**Returns**

The table name.

Reimplemented from [Entity](#).

**3.22.4 Member Data Documentation****3.22.4.1 BookLanguage::\$bookId [protected]**

[Book](#).



**3.22.4.2 BookLanguage::\$languageId [protected]**

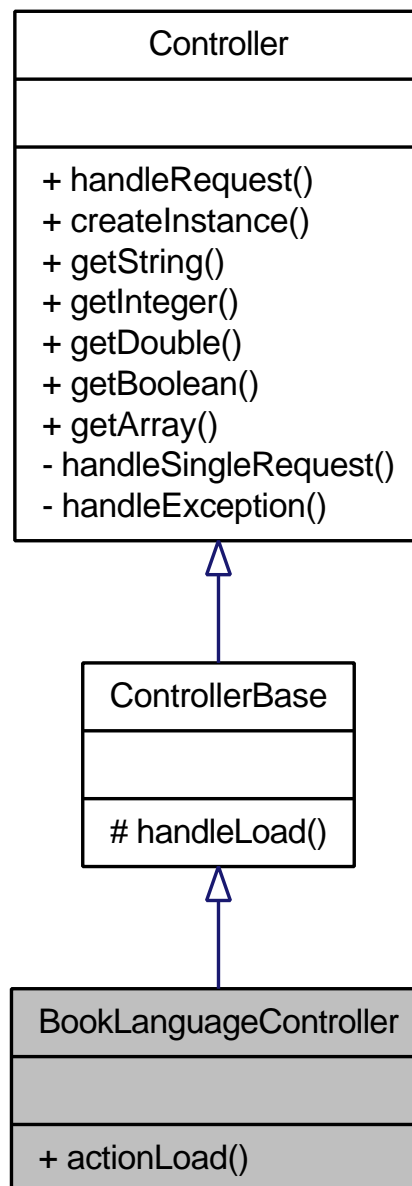
[Language](#).

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

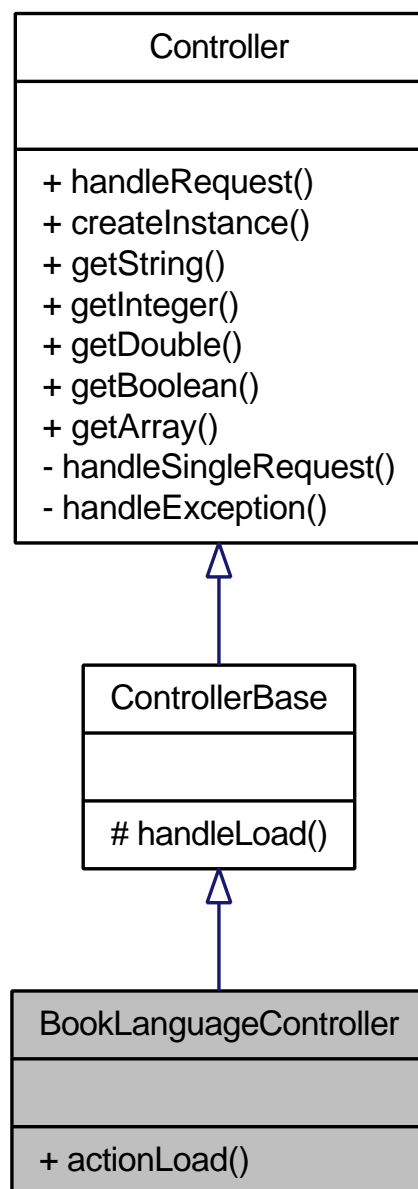
- [www/backend/models/language/booklanguage.php](#)

### 3.23 BookLanguageController Class Reference

Inheritance diagram for BookLanguageController:



Collaboration diagram for BookLanguageController:



## Public Member Functions

- [actionLoad](#) (\$data)

### 3.23.1 Detailed Description

[Book](#) language controller class.

### 3.23.2 Member Function Documentation

#### 3.23.2.1 BookLanguageController::actionLoad (\$ *data*)

Loads languages associated to books.

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

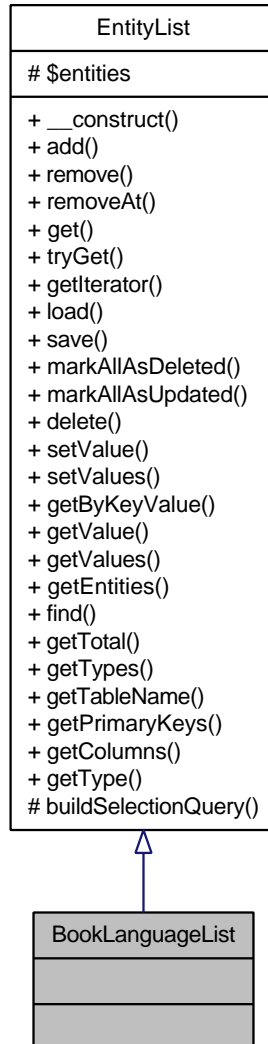
- [www/backend/controllers/booklanguagecontroller.php](#)

## 3.24 BookLanguageList Class Reference

Inheritance diagram for BookLanguageList:



Collaboration diagram for BookLanguageList:



### 3.24.1 Detailed Description

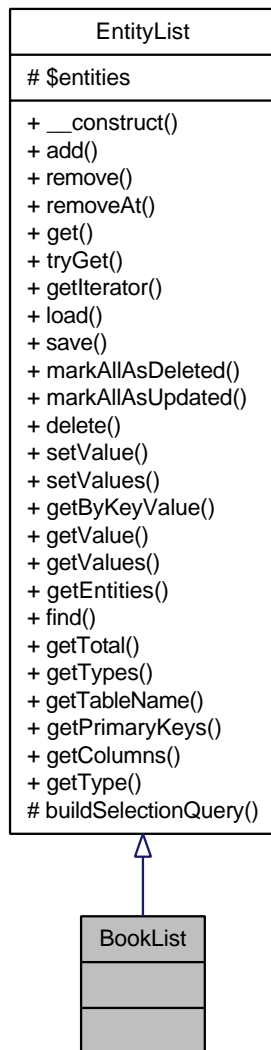
Class representing a book language entity list.

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

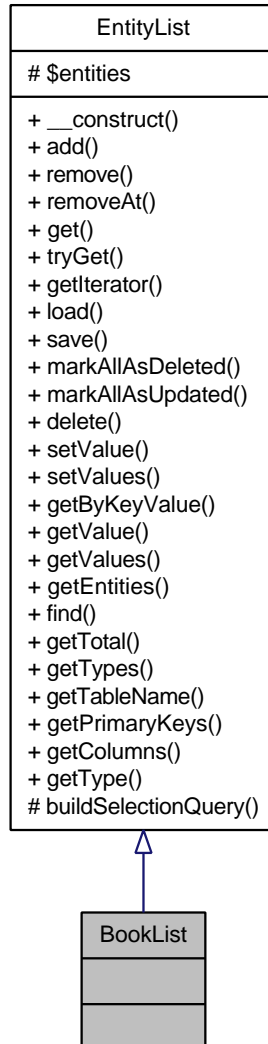
- [www/backend/models/language/booklanguagelist.php](http://www/backend/models/language/booklanguagelist.php)

## 3.25 BookList Class Reference

Inheritance diagram for BookList:



Collaboration diagram for BookList:



### 3.25.1 Detailed Description

Class representing a library entity list.

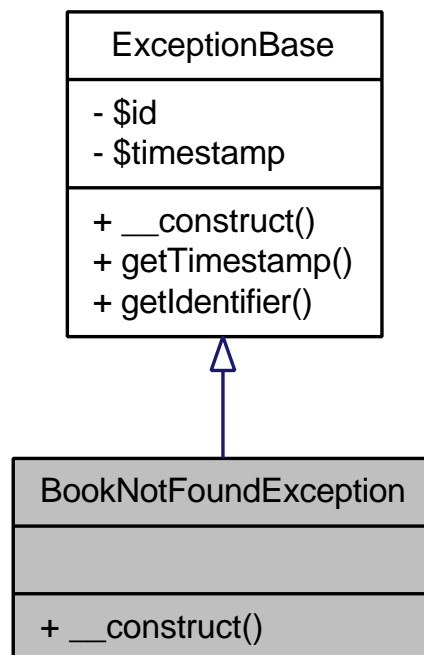
The documentation for this class was generated from the following file:

- [www/backend/models/book/booklist.php](http://www/backend/models/book/booklist.php)

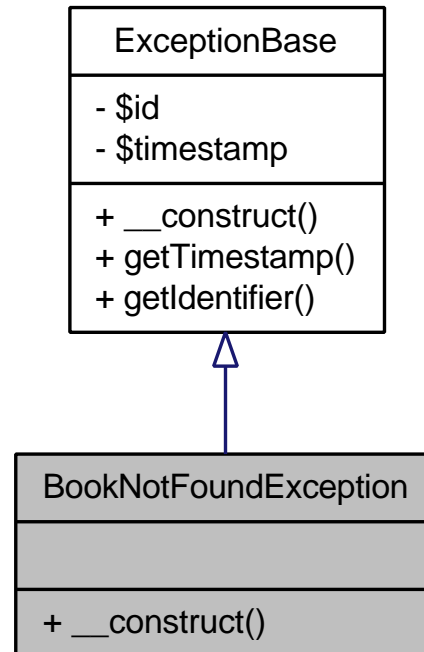


## 3.26 BookNotFoundException Class Reference

Inheritance diagram for BookNotFoundException:



Collaboration diagram for BookNotFoundException:



## Public Member Functions

- `__construct` (\$bookId)

### 3.26.1 Detailed Description

Exceptions.

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

- `www/backend/controllers/bookcontroller.php`

## 3.27 Cache Class Reference

### Static Public Member Functions

- static [getEntry \(\)](#)
- static [getFileEntry \(\)](#)

#### 3.27.1 Detailed Description

[Cache](#) class.

#### 3.27.2 Member Function Documentation

##### 3.27.2.1 static `Cache::getEntry ()` [`static`]

Gets a cache entry.

##### Parameters

... Values on which cache depends.

##### Returns

The cache entry.

##### 3.27.2.2 static `Cache::getFileEntry ()` [`static`]

Gets a file cache entry.

##### Parameters

... Values on which cache depends.

##### Returns

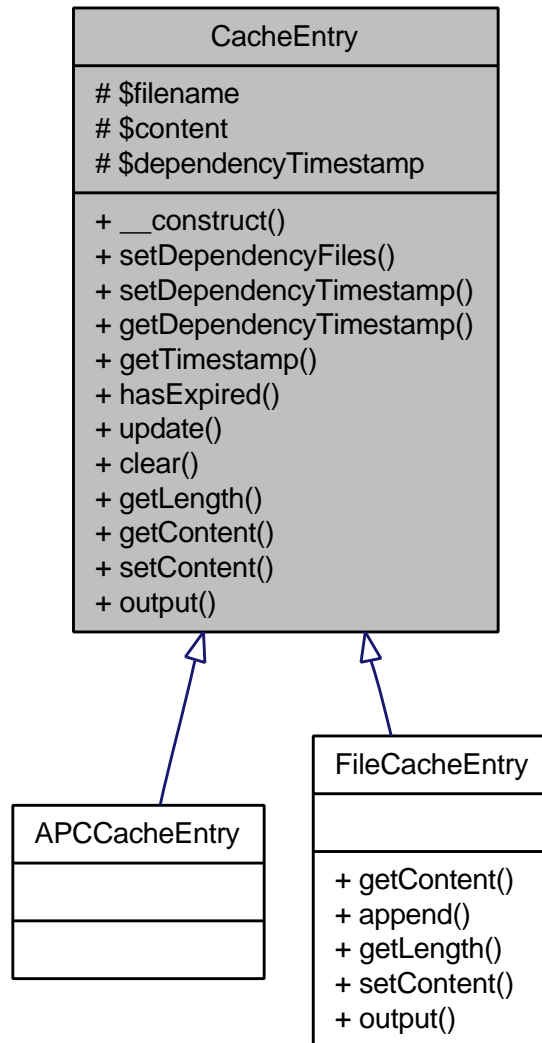
The cache entry.

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

- `www/backend/framework/util/cache.php`

### 3.28 CacheEntry Class Reference

Inheritance diagram for CacheEntry:



#### Public Member Functions

- `\_\_construct` (\$key)
- `setDependencyFiles` (\$files)
- `setDependencyTimestamp` (\$timestamp)
- `getDependencyTimestamp` ()
- `getTimestamp` ()
- `hasExpired` ()
- `update` ()

- **clear ()**
- **getLength ()**
- **getContent ()**
- **setContent (\$content)**
- **output ()**

### Protected Attributes

- **\$filename**
- **\$content**
- **\$dependencyTimestamp**

### 3.28.1 Detailed Description

[Cache](#) entry class.

### 3.28.2 Constructor & Destructor Documentation

#### 3.28.2.1 CacheEntry::\_\_construct (\$ *key*)

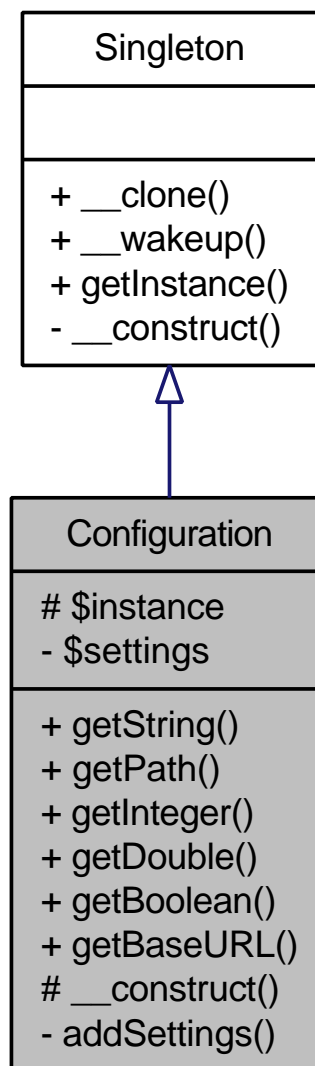
Constructs a cache entry.

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

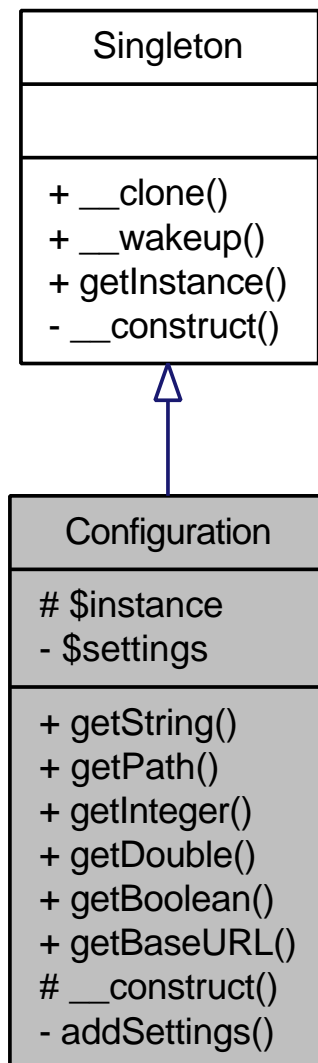
- `www/backend/framework/util/cache.php`

### 3.29 Configuration Class Reference

Inheritance diagram for Configuration:



Collaboration diagram for Configuration:



### Public Member Functions

- [getString](#) (\$name, \$default=null)
- [getPath](#) (\$name, \$default=null)
- [getInteger](#) (\$name, \$default=null)
- [getDouble](#) (\$name, \$default=null)
- [getBoolean](#) (\$name, \$default=null)

## Static Public Member Functions

- static [getBaseUrl \(\)](#)

## Protected Member Functions

- [\\_\\_construct \(\)](#)

## Static Protected Attributes

- static [\\$instance](#)

### 3.29.1 Detailed Description

[Configuration](#) class.

### 3.29.2 Constructor & Destructor Documentation

#### 3.29.2.1 Configuration::\_\_construct () [protected]

Constructs a configuration class instance.

Reimplemented from [Singleton](#).

### 3.29.3 Member Function Documentation

#### 3.29.3.1 static Configuration::getBaseUrl () [static]

Returns the base URL.

#### 3.29.3.2 Configuration::getBoolean (\$ name, \$ default = null)

Gets a boolean value.

#### Parameters

**\$name** The name of the setting.

**\$default** The default of this setting. Defaults to an exception if no setting with the given name is found.

#### Returns

The setting its value.

#### Exceptions

[SettingNotFoundException](#) If the setting could not be found and there is no default.



### 3.29.3.3 Configuration::getDouble (\$ name, \$ default = null)

Gets a floating point value.

#### Parameters

*\$name* The name of the setting.

*\$default* The default of this setting. Defaults to an exception if no setting with the given name is found.

#### Returns

The setting its value.

#### Exceptions

*SettingNotFoundException* If the setting could not be found and there is no default.

### 3.29.3.4 Configuration::getInteger (\$ name, \$ default = null)

Gets an integer value.

#### Parameters

*\$name* The name of the setting.

*\$default* The default of this setting. Defaults to an exception if no setting with the given name is found.

#### Returns

The setting its value.

#### Exceptions

*SettingNotFoundException* If the setting could not be found and there is no default.

### 3.29.3.5 Configuration::getPath (\$ name, \$ default = null)

Gets a string value denoting a path. If the path does not start with '/', the current directory is prepended.

*Note:* only works with Unix-style paths.

#### Parameters

*\$name* The name of the setting.

*\$default* The default of this setting. Defaults to an exception if no setting with the given name is found.

**Returns**

The setting its value.

**Exceptions**

*[SettingNotFoundException](#)* If the setting could not be found and there is no default.

**3.29.3.6 Configuration::getString (\$ name, \$ default = null)**

Gets a string value.

**Parameters**

*\$name* The name of the setting.

*\$default* The default of this setting. Defaults to an exception if no setting with the given name is found.

**Returns**

The setting its value.

**Exceptions**

*[SettingNotFoundException](#)* If the setting could not be found and there is no default.

**3.29.4 Member Data Documentation****3.29.4.1 Configuration::\$instance [static, protected]**

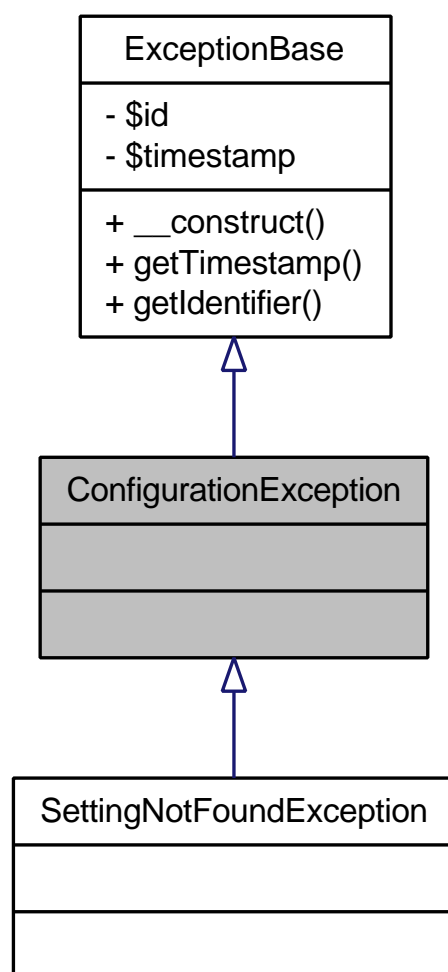
Unique instance.

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

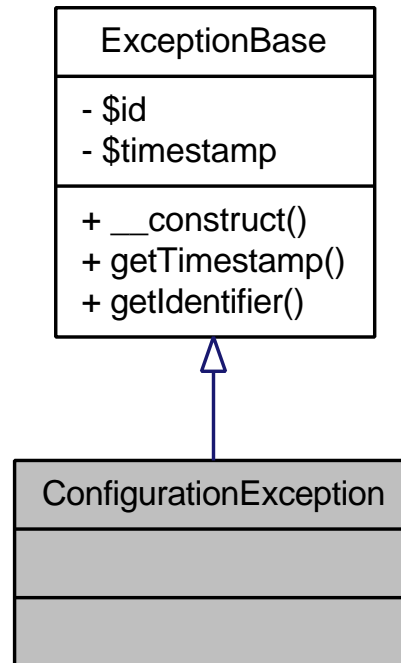
- [www/backend/framework/util/configuration.php](#)

### 3.30 ConfigurationException Class Reference

Inheritance diagram for ConfigurationException:



Collaboration diagram for ConfigurationException:

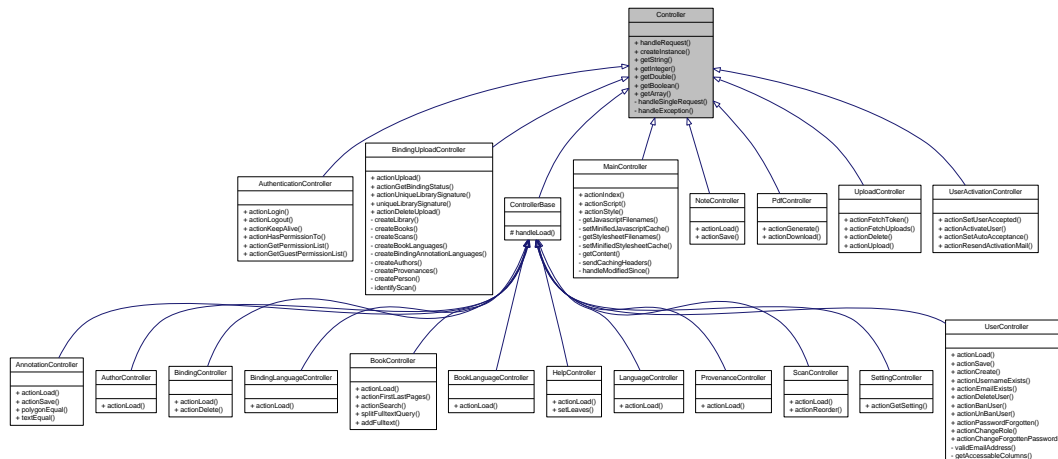


The documentation for this class was generated from the following file:

- [www/backend/framework/util/configuration.php](http://www/backend/framework/util/configuration.php)

## 3.31 Controller Class Reference

Inheritance diagram for Controller:



### Static Public Member Functions

- static [handleRequest](#) ()
- static [createInstance](#) (\$type)
- static [getString](#) (&\$data, \$key, \$default= "", \$trim=true, \$maxLength=-1)
- static [getInteger](#) (&\$data, \$key, \$default=0, \$positive=false, \$minValue=null, \$maxValue=null)
- static [getDouble](#) (&\$data, \$key, \$default=0, \$positive=false, \$minValue=null, \$maxValue=null)
- static [getBoolean](#) (&\$data, \$key, \$default=false)
- static [getArray](#) (&\$data, \$key, \$default=array(), \$maxLength=-1)

### 3.31.1 Detailed Description

[Controller](#) class.

### 3.31.2 Member Function Documentation

#### 3.31.2.1 static Controller::createInstance (\$ type) [static]

Creates a controller instance.

#### Parameters

*\$type* Type name of the controller instance.

**Returns**

An instance of the controller of the given type.

**3.31.2.2** `static Controller::getArray (&$ data, $ key, $ default = array (), $ maxLength = -1) [static]`

Gets a string from data.

**Parameters**

*\$data* Data to fetch value from.  
*\$key* Key of the value.  
*\$default* Default value.  
*\$maxLength* Maximum length of the array.

**Returns**

The sanitized array value.

**3.31.2.3** `static Controller::getBoolean (&$ data, $ key, $ default = false) [static]`

Gets a boolean from data.

**Parameters**

*\$data* Data to fetch value from.  
*\$key* Key of the value.  
*\$default* Default value.

**Returns**

The sanitized boolean value.

**3.31.2.4** `static Controller::getDouble (&$ data, $ key, $ default = 0, $ positive = false, $ minValue = null, $ maxValue = null) [static]`

Gets a double from data.

**Parameters**

*\$data* Data to fetch value from.  
*\$key* Key of the value.  
*\$default* Default value.  
*\$positive* Whether the value must be positive.

*\$minValue* Minimum value.

*\$maxValue* Maximum value.

#### Returns

The sanitized double value.

**3.31.2.5** `static Controller::getInteger (&$ data, $ key, $ default = 0, $ positive = false, $ minValue = null, $ maxValue = null) [static]`

Gets an integer from data.

#### Parameters

*\$data* Data to fetch value from.

*\$key* Key of the value.

*\$default* Default value.

*\$positive* Whether the value must be positive.

*\$minValue* Minimum value.

*\$maxValue* Maximum value.

#### Returns

The sanitized integer value.

**3.31.2.6** `static Controller::getString (&$ data, $ key, $ default = "", $ trim = true, $ maxLength = -1) [static]`

Gets a string from data.

#### Parameters

*\$data* Data to fetch value from.

*\$key* Key of the value.

*\$default* Default value.

*\$trim* Whether to trim spaces of value.

*\$maxLength* Maximum length of value.

#### Returns

The sanitized string value.

**3.31.2.7** `static Controller::handleRequest () [static]`

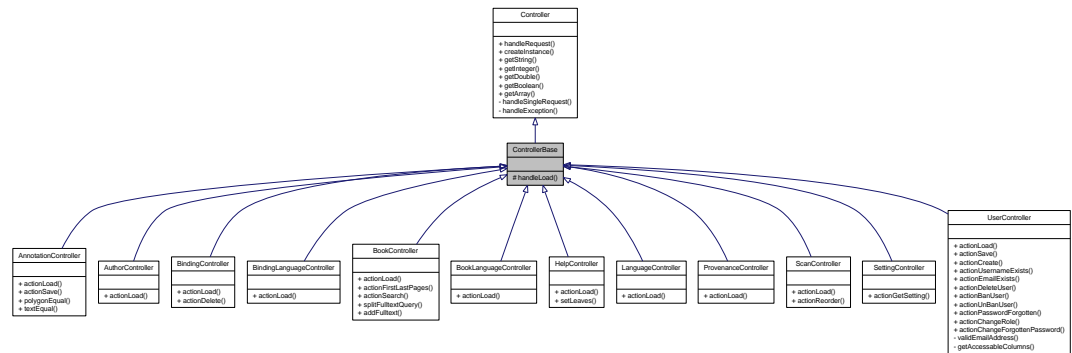
Handles a request.

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

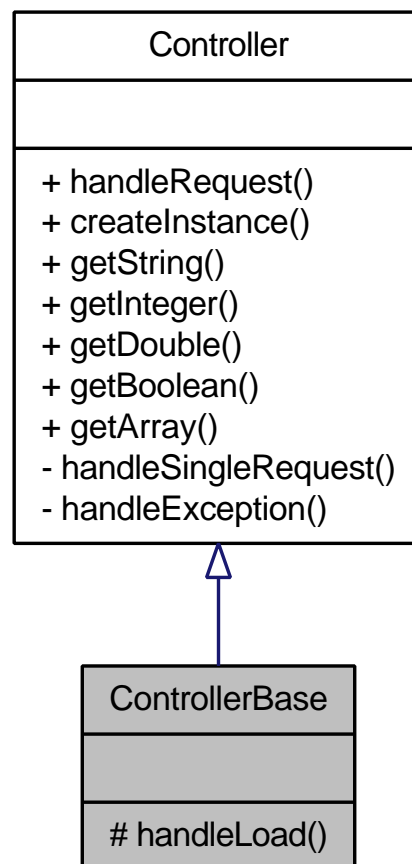
- [www/backend/framework/controller/controller.php](http://www/backend/framework/controller/controller.php)

### 3.32 ControllerBase Class Reference

Inheritance diagram for ControllerBase:



Collaboration diagram for ControllerBase:





## Protected Member Functions

- [handleLoad](#) (\$data, \$type, \$idColumn=null, \$columns=null, array \$defaultSorters=array())

### 3.32.1 Detailed Description

[Controller](#) base class.

### 3.32.2 Member Function Documentation

#### 3.32.2.1 ControllerBase::handleLoad (\$ data, \$ type, \$ idColumn = null, \$ columns = null, array \$ defaultSorters = array ()) [protected]

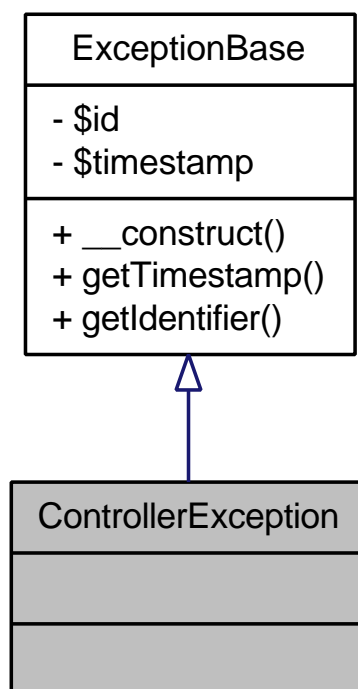
Generalizes a load query.

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

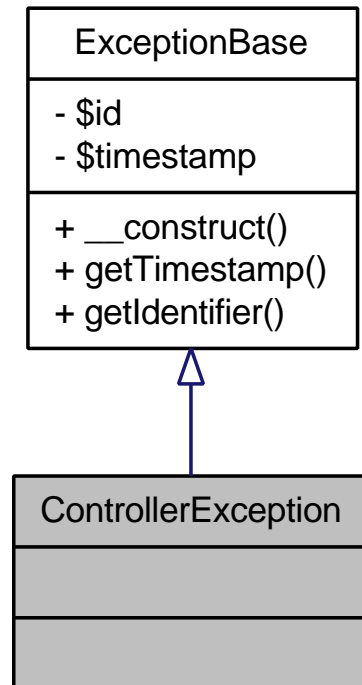
- [www/backend/controllers/controllerbase.php](#)

### 3.33 ControllerException Class Reference

Inheritance diagram for ControllerException:



Collaboration diagram for ControllerException:

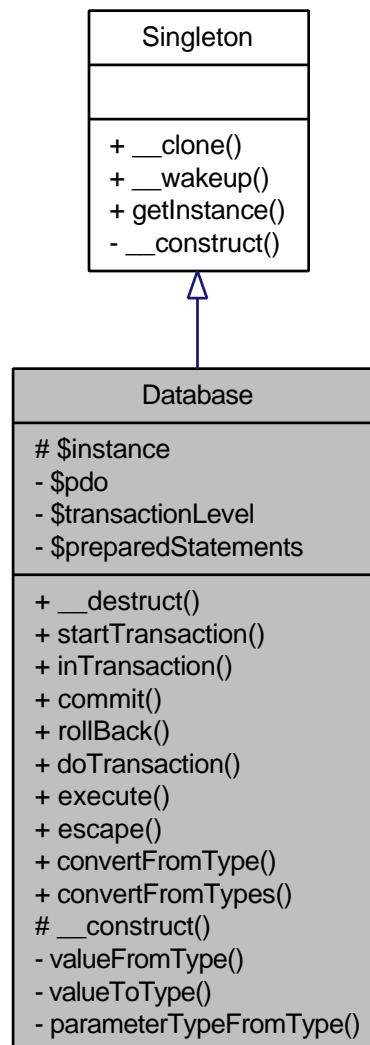


The documentation for this class was generated from the following file:

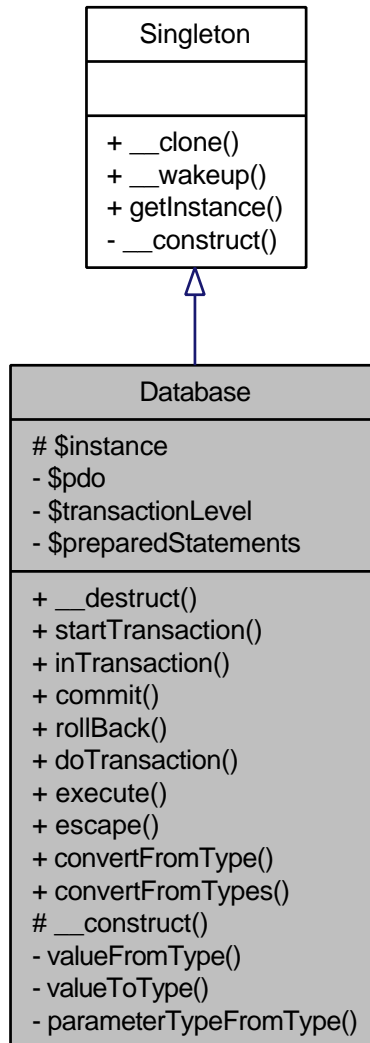
- [www/backend/framework/controller/controller.php](http://www/backend/framework/controller/controller.php)

### 3.34 Database Class Reference

Inheritance diagram for Database:



Collaboration diagram for Database:



## Public Member Functions

- [startTransaction \(\)](#)
- [inTransaction \(\)](#)
- [commit \(\)](#)
- [rollBack \(\)](#)
- [doTransaction \(\\$callback\)](#)
- [execute \(\\$query, \\$arguments=array\(\), \\$types=null\)](#)
- [escape \(\\$value\)](#)

## Static Public Member Functions

- static [convertFromType](#) (\$value, \$type)
- static [convertFromTypes](#) (\$values, \$types)

## Static Protected Attributes

- static [\\$instance](#)

### 3.34.1 Member Function Documentation

#### 3.34.1.1 Database::commit ()

Commits the current transaction.

Commits the current transaction, finalizing all changes made to the database. The transaction will now have ended and startTransaction can be called start a new one (or the query methods can be called directly).

#### Exceptions

[DatabaseException](#) If no transaction has started or the commit failed for some other reason.

#### 3.34.1.2 static Database::convertFromType (\$ value, \$ type) [static]

Converts database data to normal data.

#### Parameters

*\$value* Value to convert.  
*\$type* Type of the value.

#### Returns

Resulting value.

#### 3.34.1.3 static Database::convertFromTypes (\$ values, \$ types) [static]

Converts database data to normal data.

#### Parameters

*\$values* Values to convert.  
*\$types* Types of those values.

#### Returns

Resulting values.

#### 3.34.1.4 Database::doTransaction (\$ callback)

Exception-safe transaction handler.

Starts a transaction and executes the provided callback. If that callback terminates normally the transaction is committed, if it throws an exception however the transaction will be rolled back and it is rethrown.

##### Parameters

*callback* \$callback The function to execute as a transaction. May contain calls to this method for recursive transactions.

##### Returns

mixed Returns whatever value the callback returns, if any.

##### Exceptions

*Exception* Rethrows exceptions from the callback.

#### 3.34.1.5 Database::execute (\$ query, \$ arguments = array (), \$ types = null)

Executes query.

##### Parameters

*\$query* Query to execute.

*\$arguments* Arguments to bind to query.

*\$types* Types of those arguments.

##### Returns

Result set.

#### 3.34.1.6 Database::inTransaction ()

Checks whether currently in a transaction.

##### Returns

A boolean indicating whether currently in a transaction.

##### Exceptions

*DatabaseException*

#### 3.34.1.7 Database::rollBack ()

Rolls back a transaction.

Undoes all queries made in the current transaction. The transaction will have ended, so \ [startTransaction\(\)](#) should be called again when trying again.

##### Exceptions

*[DatabaseException](#)* If no transaction has started, rollback is not supported or it failed for some other reason.

#### 3.34.1.8 Database::startTransaction ()

Starts a database transaction. Nested transaction are allowed.

##### Exceptions

*[DatabaseException](#)* If the transaction could not be started.

### 3.34.2 Member Data Documentation

#### 3.34.2.1 Database::\$instance [static, protected]

Unique instance.

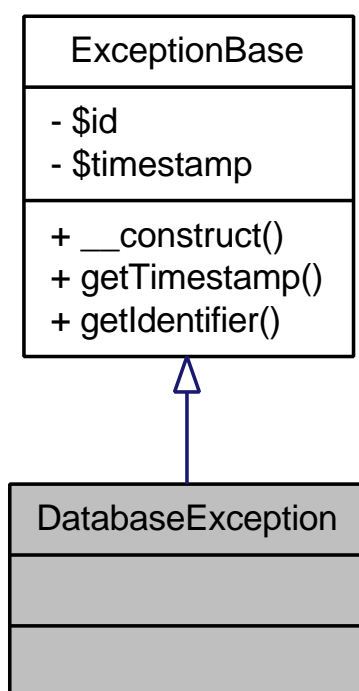
The documentation for this class was generated from the following file:

- [www/backend/framework/database/database.php](#)

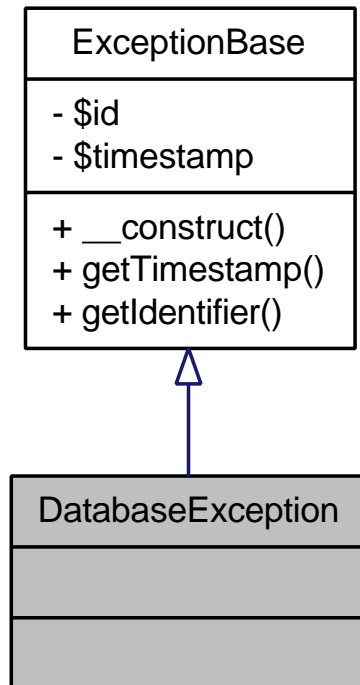


## 3.35 DatabaseException Class Reference

Inheritance diagram for DatabaseException:



Collaboration diagram for DatabaseException:

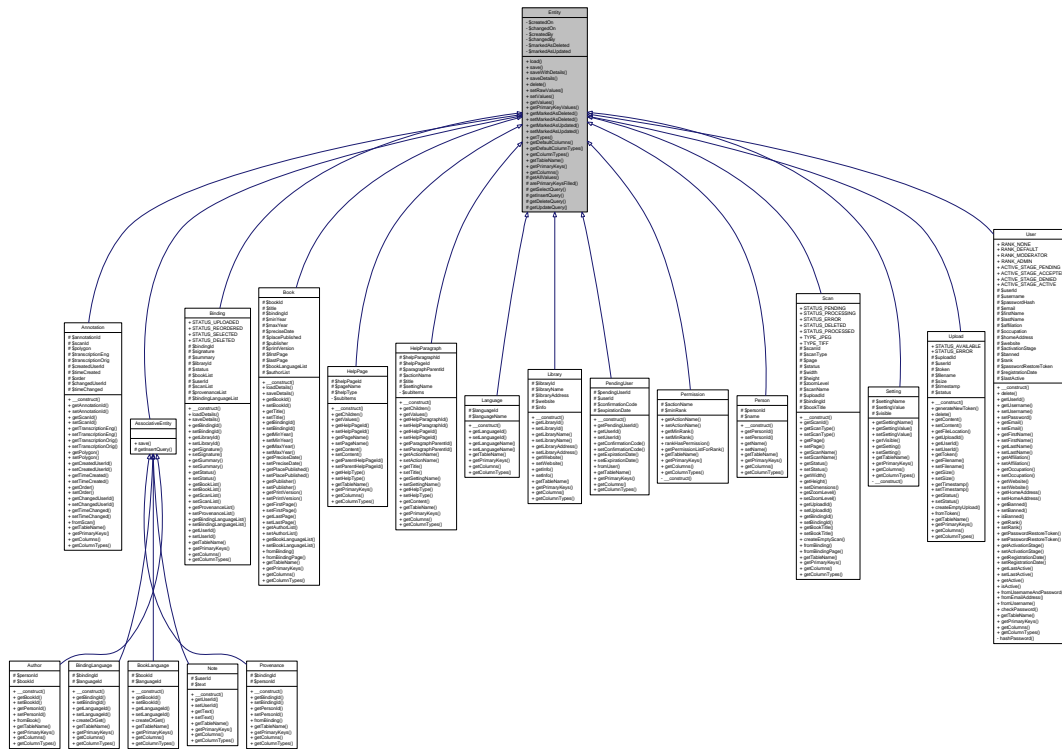


The documentation for this class was generated from the following file:

- [www/backend/framework/database/database.php](http://www/backend/framework/database/database.php)

### 3.36 Entity Class Reference

Inheritance diagram for Entity:



## Public Member Functions

- `load ()`
- `save ()`
- `saveWithDetails ()`
- `saveDetails ()`
- `delete ()`
- `setRawValues ($values)`
- `setValues ($values)`
- `getValues ($columns=null)`
- `getPrimaryKeyValues ()`
- `getMarkedAsDeleted ()`
- `setMarkedAsDeleted ($marked)`
- `getMarkedAsUpdated ()`
- `setMarkedAsUpdated ($marked)`

## Static Public Member Functions

- static [getTypes](#) (\$default=true)
- static [getDefaultColumns](#) ()
- static [getDefaultColumnTypes](#) ()
- static [getColumnTypes](#) ()
- static [getTableName](#) ()
- static [getPrimaryKeys](#) ()
- static [getColumns](#) ()

## Protected Member Functions

- [getAllValues](#) (\$keys=true, \$default=true)
- [arePrimaryKeysFilled](#) ()
- [getSelectQuery](#) ()
- [getInsertQuery](#) (\$returning=false)
- [getDeleteQuery](#) ()
- [getUpdateQuery](#) ()

### 3.36.1 Detailed Description

Abstract class for any database entity.

### 3.36.2 Member Function Documentation

#### 3.36.2.1 Entity::arePrimaryKeysFilled () [protected]

Checks if all primary keys are filled.

##### Returns

Whether the primary keys are filled.

#### 3.36.2.2 Entity::delete ()

Deletes the entity from the database.

Reimplemented in [Upload](#), and [User](#).

#### 3.36.2.3 Entity::getAllValues (\$ keys = true, \$ default = true) [protected]

Gets all values of this entity.

##### Returns

All values of this entity.

**3.36.2.4 static Entity::getColumns () [static]**

Gets all the columns.

**Returns**

Array of all columns, except primary keys.

Reimplemented in [Annotation](#), [Author](#), [Binding](#), [Book](#), [HelpPage](#), [HelpParagraph](#), [BindingLanguage](#), [BookLanguage](#), [Language](#), [Library](#), [Note](#), [Permission](#), [Person](#), [Provenance](#), [Scan](#), [Setting](#), [Upload](#), [PendingUser](#), and [User](#).

**3.36.2.5 static Entity::getColumnTypes () [static]**

Gets all the column types, per column, including primary keys.

The istring type denotes a string that should be compared in a case insensitive manner.

**Returns**

Array of all column types.

Reimplemented in [Annotation](#), [Author](#), [Binding](#), [Book](#), [HelpPage](#), [HelpParagraph](#), [BindingLanguage](#), [BookLanguage](#), [Language](#), [Library](#), [Note](#), [Permission](#), [Person](#), [Provenance](#), [Scan](#), [Setting](#), [Upload](#), [PendingUser](#), and [User](#).

**3.36.2.6 static Entity::getDefaultColumns () [static]**

Gets columns that are available in all entities.

**Returns**

[Query](#) to select an entity from the database.

**3.36.2.7 static Entity::getDefaultColumnTypes () [static]**

Gets all the default column types, per column.

**Returns**

Array of all default column types.

**3.36.2.8 Entity::getDeleteQuery () [protected]**

Returns the query needed to delete this entity from the database.

**Returns**

[Query](#) to delete this entity from the database.

### 3.36.2.9 Entity::getInsertQuery (\$returning = false) [protected]

Returns the query needed to insert this entity into the database.

#### Parameters

*\$returning* If true, a returning clause is added to the query which returns all primary keys.

#### Returns

[Query](#) to insert this entity in the database.

Reimplemented in [AssociativeEntity](#).

### 3.36.2.10 Entity::getMarkedAsDeleted ()

Getters and setters.

### 3.36.2.11 static Entity::getPrimaryKeys () [static]

Gets the primary keys.

#### Returns

Array of all primary keys.

Reimplemented in [Annotation](#), [Author](#), [Binding](#), [Book](#), [HelpPage](#), [HelpParagraph](#), [BindingLanguage](#), [BookLanguage](#), [Language](#), [Library](#), [Note](#), [Permission](#), [Person](#), [Provenance](#), [Scan](#), [Setting](#), [Upload](#), [PendingUser](#), and [User](#).

### 3.36.2.12 Entity::getPrimaryKeyValues ()

Gets all primary key values of this entity.

#### Returns

All primary key values of this entity.

### 3.36.2.13 Entity::getSelectQuery () [protected]

Returns the query needed to select this entity from the database.

#### Returns

[Query](#) to select an entity from the database.

**3.36.2.14 static Entity::getTableName () [static]**

Gets the table name.

**Returns**

The table name.

Reimplemented in [Annotation](#), [Author](#), [Binding](#), [Book](#), [HelpPage](#), [HelpParagraph](#), [BindingLanguage](#), [BookLanguage](#), [Language](#), [Library](#), [Note](#), [Permission](#), [Person](#), [Provenance](#), [Scan](#), [Setting](#), [Upload](#), [PendingUser](#), and [User](#).

**3.36.2.15 static Entity::getTypes (\$ default = true) [static]**

Gets all types of this entity.

**Returns**

All types of this entity, by column name.

**3.36.2.16 Entity::getUpdateQuery () [protected]**

Returns the query needed to update this entity in the database.

**Returns**

[Query](#) to update this entity in the database.

**3.36.2.17 Entity::getValues (\$ columns = null)**

Gets some values of this entity.

**Parameters**

*\$values* Array of names to fetch, or null if you want to fetch everything.

Reimplemented in [HelpPage](#), and [HelpParagraph](#).

**3.36.2.18 Entity::load ()**

Loads this entity.

**3.36.2.19 Entity::save ()**

Saves the entity to the database. A database row is inserted if the entity does not exist in the database yet. A database row is updated if the entity exists in the database.

Reimplemented in [AssociativeEntity](#).

**3.36.2.20 Entity::saveDetails ()**

Saves entity its relations.

Reimplemented in [Binding](#), and [Book](#).

**3.36.2.21 Entity::saveWithDetails ()**

Saves entity with its relations.

**3.36.2.22 Entity::setRawValues (\$ values)**

Sets raw values of this entity. Only for use by entity list!

**Parameters**

*\$values* Array of name-value pairs to set.

**3.36.2.23 Entity::setValues (\$ values)**

Sets some values of this entity.

**Parameters**

*\$values* Array of name-value pairs to set.

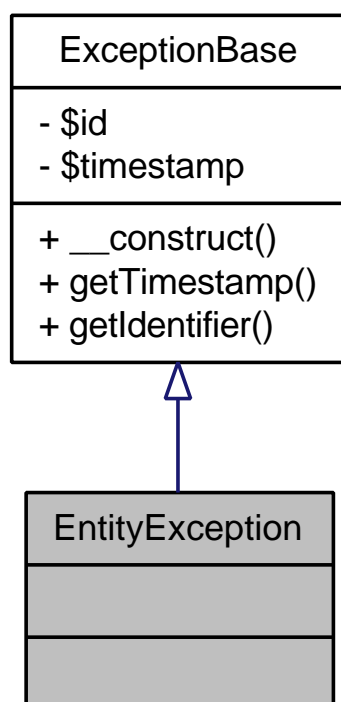
The documentation for this class was generated from the following file:

- [www/backend/framework/database/entity.php](http://www/backend/framework/database/entity.php)

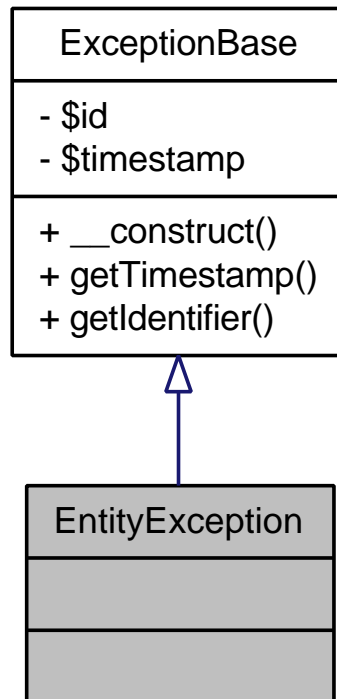


## 3.37 EntityException Class Reference

Inheritance diagram for EntityException:



Collaboration diagram for EntityException:

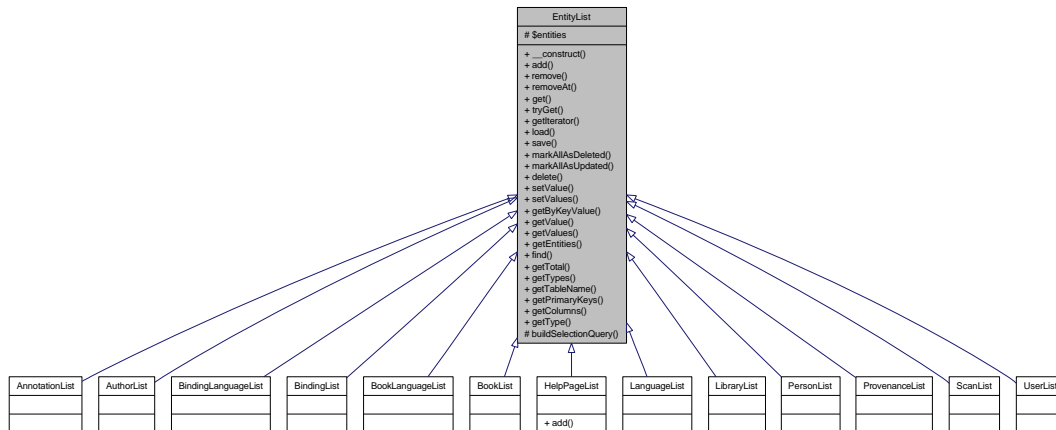


The documentation for this class was generated from the following file:

- [www/backend/framework/database/entity.php](http://www/backend/framework/database/entity.php)

## 3.38 EntityList Class Reference

Inheritance diagram for EntityList:



### Public Member Functions

- [\\_\\_construct](#) ()
- [add](#) (\$entity)
- [remove](#) (\$entity)
- [removeAt](#) (\$index)
- [get](#) (\$index)
- [tryGet](#) (\$index)
- [getIterator](#) ()
- [load](#) (\$keyValues)
- [save](#) ()
- [markAllAsDeleted](#) (\$marked)
- [markAllAsUpdated](#) (\$marked)
- [delete](#) ()
- [setValue](#) (\$name, \$value)
- [setValues](#) (\$values)
- [getByKeyValue](#) (\$key, \$value)
- [getValue](#) (\$name)
- [getValues](#) (\$names)
- [getEntities](#) ()

### Static Public Member Functions

- static [find](#) (\$conditions=array(), \$offset=0, \$limit=null, \$ordering=array(), &\$total=null)
- static [getTotal](#) ()
- static [getTypes](#) (\$default=true)

- static [getTableName](#) ()
- static [getPrimaryKeys](#) ()
- static [getColumns](#) ()
- static [getType](#) ()

### Static Protected Member Functions

- static [buildSelectionQuery](#) ()

### Protected Attributes

- `$entities`

#### 3.38.1 Detailed Description

Abstract class for a list of database entities.

#### 3.38.2 Constructor & Destructor Documentation

##### 3.38.2.1 `EntityList::__construct` ()

Constructs an entity list.

#### 3.38.3 Member Function Documentation

##### 3.38.3.1 `EntityList::add` (*\$entity*)

Adds an entity.

##### Parameters

*\$entity* [Entity](#) to add.

Reimplemented in [HelpPageList](#).

##### 3.38.3.2 `static EntityList::buildSelectionQuery` () [`static`, `protected`]

Returns the query to select fields of this entity type.

##### 3.38.3.3 `EntityList::delete` ()

Deletes all entities from the database.

### 3.38.3.4 static EntityList::find (\$ conditions = array () , \$ offset = 0, \$ limit = null, \$ ordering = array () , &\$ total = null) [static]

Finds entities.

#### Parameters

*\$conditions* Column-value pairs of equal conditions the list should satisfy. case insensitive strings will be compared Case insensitively.

*\$offset* Offset to start list at.

*\$limit* Amount of entities to return. Pass null for no limit.

*\$ordering* Column-direction pairs of ordering.

*\$total* Will contain the total number of records, without the limit or offset constraint.

#### Returns

[Entity](#) list with entities that pass the given criteria.

### 3.38.3.5 EntityList::get (\$ index)

Fetches an entity by index.

#### Parameters

*\$index*

#### Returns

[Entity](#) at index.

### 3.38.3.6 EntityList::getByKeyValue (\$ key, \$ value)

Gets an entity by a key value combination.

#### Parameters

*\$key* Name of the value.

*\$value* Value.

#### Returns

Array of values.

**3.38.3.7 static EntityList::getColumns () [static]**

Gets all the columns.

**Returns**

Array of all columns, except primary keys.

**3.38.3.8 EntityList::getEntities ()**

Returns the entities.

**Returns**

Array of models.

**3.38.3.9 EntityList::getIterator ()**

Gets an iterator of this list.

**Returns**

[Entity](#) list iterator.

**3.38.3.10 static EntityList::getPrimaryKeys () [static]**

Gets the primary keys.

**Returns**

Array of all primary keys.

**3.38.3.11 static EntityList::getTableName () [static]**

Gets the table name.

**Returns**

The table name.

**3.38.3.12 static EntityList::getTotal () [static]**

Gets the total amount of entities in database.

**3.38.3.13 static EntityList::getType () [static]**

Gets the type of the entity of this entity list.

**Returns**

Type of the entity.

**3.38.3.14 static EntityList::getTypes (\$ default = true) [static]**

Gets all types of this entity list.

**Returns**

All types of this entity list, by column name.

**3.38.3.15 EntityList::getValue (\$ name)**

Gets a value of all these entities.

**Parameters**

*\$name* Name of the value.

**Returns**

Array of values.

**3.38.3.16 EntityList::getValues (\$ names)**

Gets some values of all these entities.

**Parameters**

*\$names* Array of names to get.

**Returns**

Array of array with key value pairs.

**3.38.3.17 EntityList::load (\$ keyValues)**

Loads entities.

**3.38.3.18 EntityList::markAllAsDeleted (\$ marked)**

Marks all entities as deleted.

**3.38.3.19 EntityList::markAllAsUpdated (\$ *marked*)**

Marks all entities as updated.

**3.38.3.20 EntityList::remove (\$ *entity*)**

Removes an entity.

**Parameters**

*\$entity* [Entity](#) to remove.

**3.38.3.21 EntityList::removeAt (\$ *index*)**

Removes an entity at an index.

**Parameters**

*\$index* Index of entity to remove.

**3.38.3.22 EntityList::save ()**

Saves all entities to the database.

**3.38.3.23 EntityList::setValue (\$ *name*, \$ *value*)**

Sets a value to all these entities.

**Parameters**

*\$name* Name of the value.

*\$value* Its value.

**3.38.3.24 EntityList::setValues (\$ *values*)**

Sets some values to all these entities.

**Parameters**

*\$values* Array of name-value pairs to set.



### 3.38.3.25 EntityList::tryGet (\$ *index*)

Tries to fetch an entity by index.

#### Parameters

*\$index*

#### Returns

[Entity](#) at index or null if not found.

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

- [www/backend/framework/database/entitylist.php](http://www/backend/framework/database/entitylist.php)

## 3.39 ExceptionBase Class Reference

Inheritance diagram for ExceptionBase:



### Public Member Functions

- `__construct ($id)`
- `getTimestamp ()`
- `getIdentifier ()`

#### 3.39.1 Detailed Description

Exception base class.

#### 3.39.2 Member Function Documentation

##### 3.39.2.1 ExceptionBase::getIdentifier ()

Gets the exception unique identifier.

##### Returns

Identifier to the exception.

##### 3.39.2.2 ExceptionBase::getTimestamp ()

Gets the exception its formatted timestamp.

##### Returns

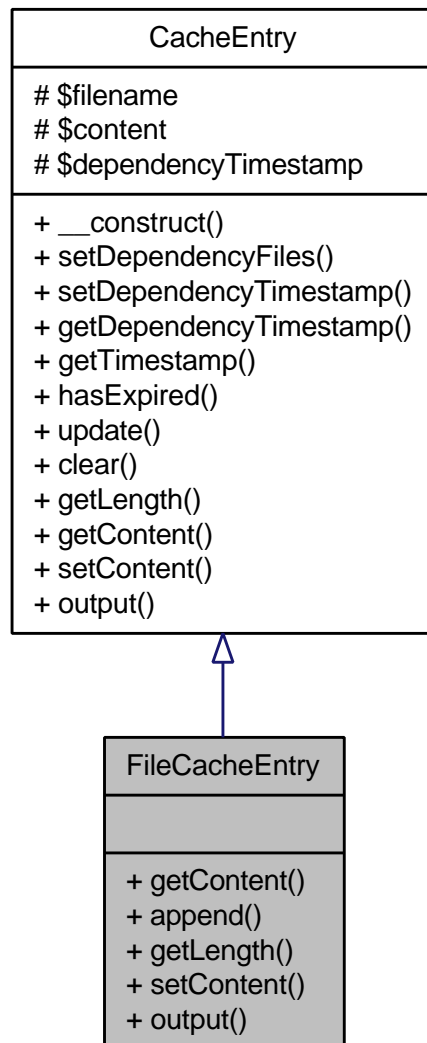
The exception its timestamp.

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

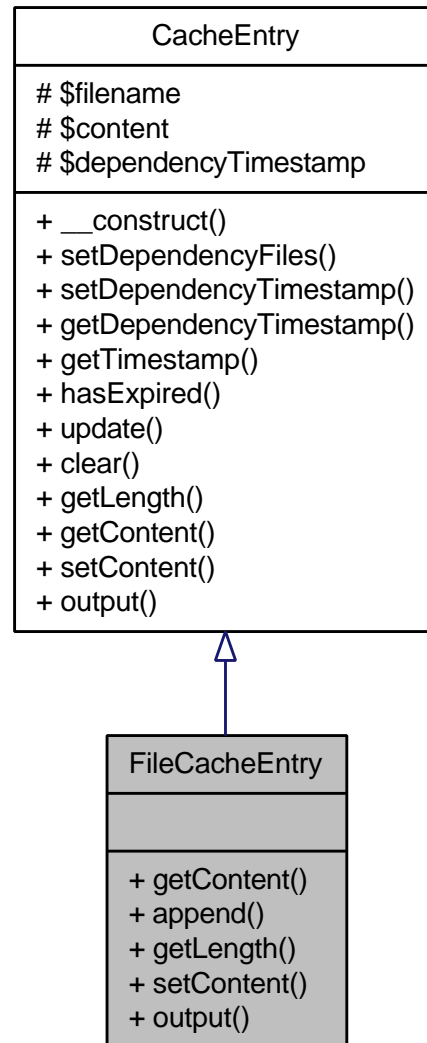
- `www/backend/framework/util/exceptionbase.php`

## 3.40 FileCacheEntry Class Reference

Inheritance diagram for FileCacheEntry:



Collaboration diagram for FileCacheEntry:



## Public Member Functions

- **getContent** ()
- **append** (\$content)
- **getLength** ()
- **setContent** (\$content)
- **output** ()

### 3.40.1 Detailed Description

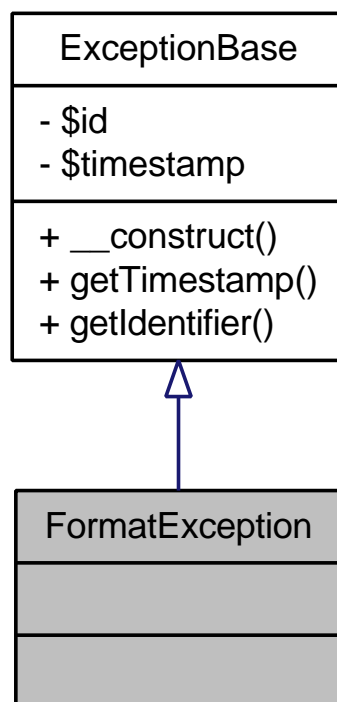
File cache entry class.

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

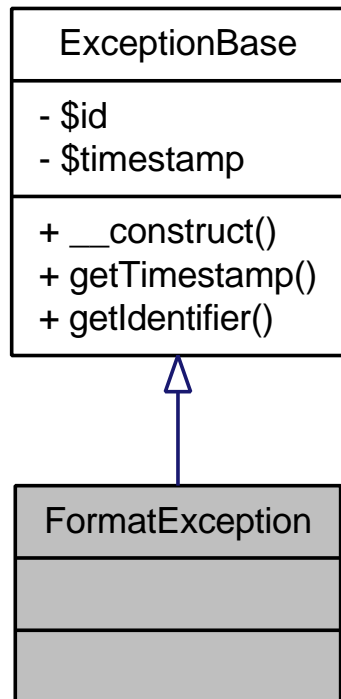
- [www/backend/framework/util/cache.php](#)

### 3.41 FormatException Class Reference

Inheritance diagram for FormatException:



Collaboration diagram for FormatException:

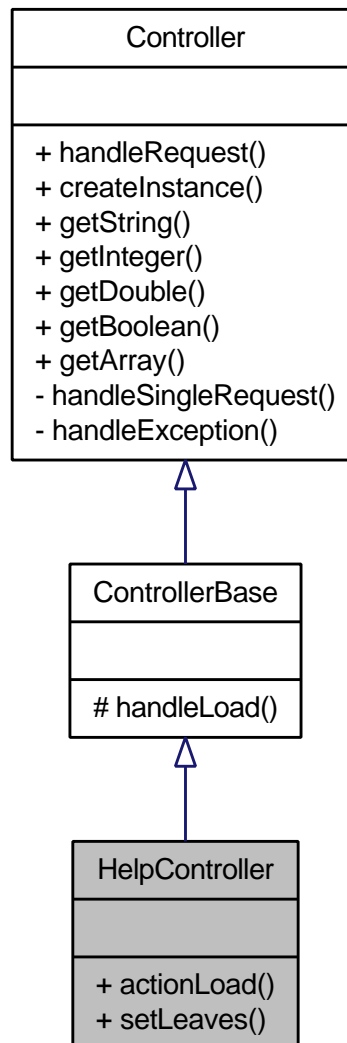


The documentation for this class was generated from the following file:

- [www/backend/framework/util/exceptionbase.php](http://www/backend/framework/util/exceptionbase.php)

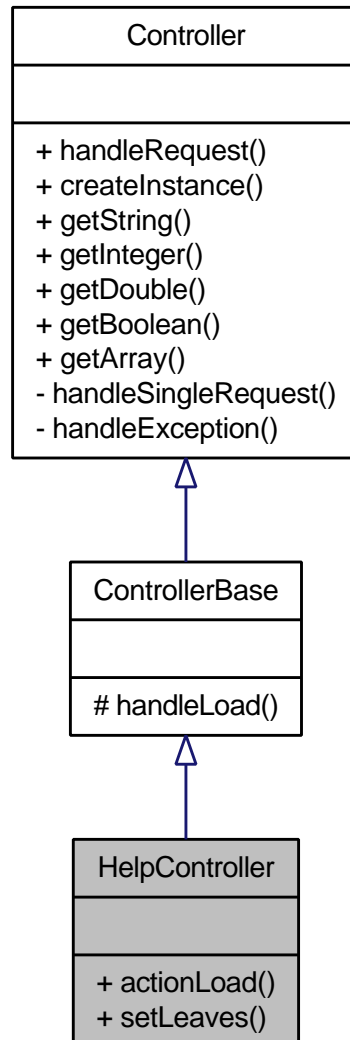
### 3.42 HelpController Class Reference

Inheritance diagram for HelpController:





Collaboration diagram for HelpController:



## Public Member Functions

- [actionLoad](#) (\$data)
- [setLeaves](#) (&\$helpItem)

### 3.42.1 Detailed Description

Help controller class.

## 3.42.2 Member Function Documentation

### 3.42.2.1 HelpController::actionLoad (*\$ data*)

Loads help pages.

### 3.42.2.2 HelpController::setLeaves (&*\$ helpItem*)

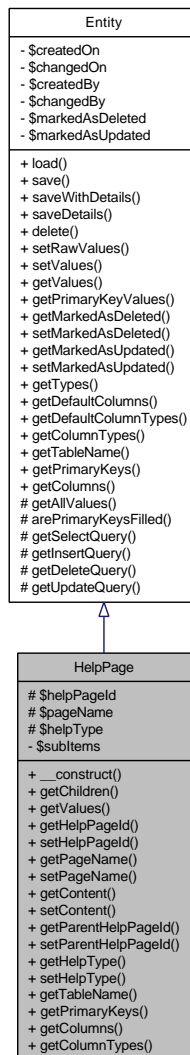
Loads the children of the requested node and checks whether they are leaves or not.

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

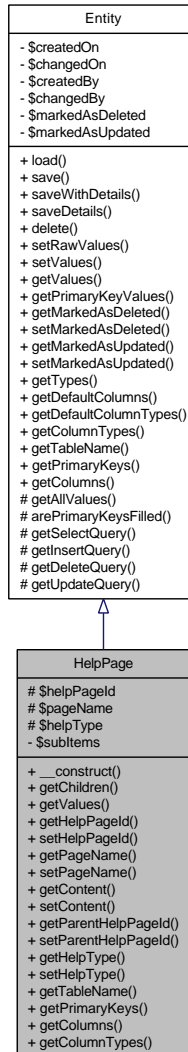
- [www/backend/controllers/helpcontroller.php](http://www/backend/controllers/helpcontroller.php)

## 3.43 HelpPage Class Reference

Inheritance diagram for HelpPage:



Collaboration diagram for HelpPage:



## Public Member Functions

- [\\_\\_construct](#) (\$id=null)
- [getChildren](#) ()
- [getValues](#) (\$columns=null)
- [getHelpPageId](#) ()
- [setHelpPageId](#) (\$id)
- [getPageName](#) ()
- [setPageName](#) (\$name)
- [getContent](#) ()
- [setContent](#) (\$con)

- `getParentHelpPageId ()`
- `setParentHelpPageId ($id)`
- `getHelpType ()`
- `setHelpType ($type)`

### Static Public Member Functions

- static `getTableName ()`
- static `getPrimaryKeys ()`
- static `getColumns ()`
- static `getColumnTypes ()`

### Protected Attributes

- `$helpPageId`
- `$pageName`
- `$helpType`

## 3.43.1 Detailed Description

[Entity](#) representing a help page.

## 3.43.2 Constructor & Destructor Documentation

### 3.43.2.1 HelpPage::\_\_construct (\$ id = null)

Constructs a [HelpPage](#) by id.

#### Parameters

*int* \$id Id of the page. Default (null) will create a new help page.

## 3.43.3 Member Function Documentation

### 3.43.3.1 HelpPage::getChildren ()

Returns all the children of a node based on the current permissions

#### Returns

Array of help paragraphs.

**3.43.3.2 static HelpPage::getColumns () [static]**

Gets all the columns.

**Returns**

Array of all columns, except primary keys.

Reimplemented from [Entity](#).

**3.43.3.3 static HelpPage::getColumnTypes () [static]**

Gets all the column types, per column, including primary keys.

**Returns**

Array of all column types.

Reimplemented from [Entity](#).

**3.43.3.4 HelpPage::getHelpPageId ()**

Getters and setters.

**3.43.3.5 static HelpPage::getPrimaryKeys () [static]**

Gets the primary keys.

**Returns**

Array of all primary keys.

Reimplemented from [Entity](#).

**3.43.3.6 static HelpPage::getTableName () [static]**

Gets the table name.

**Returns**

The table name.

Reimplemented from [Entity](#).

### 3.43.3.7 HelpPage::getValues (\$ columns = null)

Gets some values of this entity.

#### Parameters

*\$values* Array of names to fetch, or null if you want to fetch everything.

Reimplemented from [Entity](#).

## 3.43.4 Member Data Documentation

### 3.43.4.1 HelpPage::\$helpPageId [protected]

Identifier.

### 3.43.4.2 HelpPage::\$helpType [protected]

The type of the page.

### 3.43.4.3 HelpPage::\$pageName [protected]

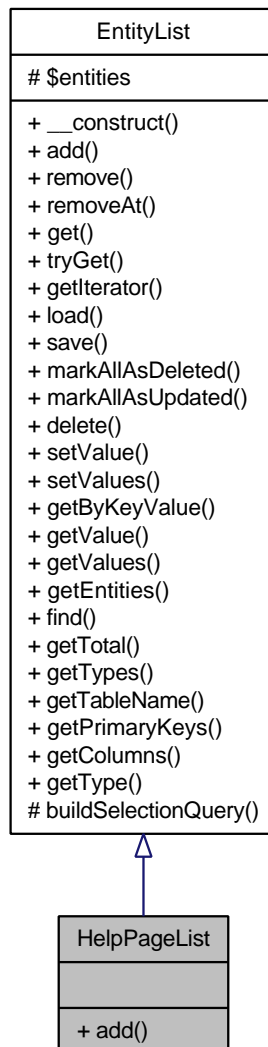
The name of the page.

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

- `www/backend/models/help/helppage.php`

### 3.44 HelpPageList Class Reference

Inheritance diagram for HelpPageList:





Collaboration diagram for HelpPageList:



## Public Member Functions

- [add](#) (\$helpPage)

### 3.44.1 Detailed Description

Class representing a help entity list.

### 3.44.2 Member Function Documentation

#### 3.44.2.1 HelpPageList::add (\$ *entity*)

Adds an entity.

##### Parameters

*\$entity* [Entity](#) to add.

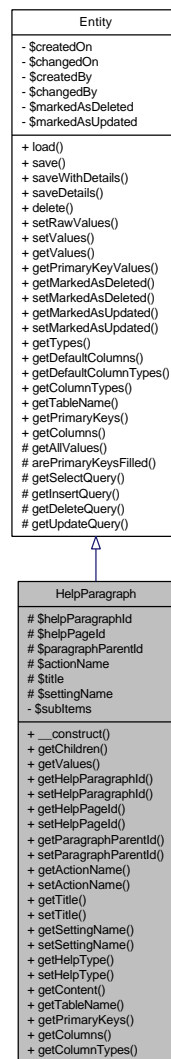
Reimplemented from [EntityList](#).

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

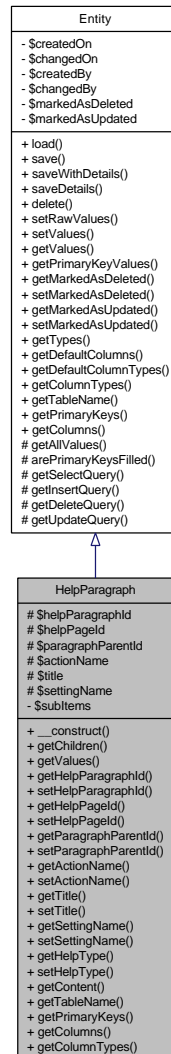
- [www/backend/models/help/helppagelist.php](http://www/backend/models/help/helppagelist.php)

## 3.45 HelpParagraph Class Reference

Inheritance diagram for HelpParagraph:



Collaboration diagram for HelpParagraph:



## Public Member Functions

- [\\_\\_construct](#) (\$id=null)
- [getChildren](#) ()
- [getValues](#) (\$columns=null)
- [getHelpParagraphId](#) ()
- [setHelpParagraphId](#) (\$id)
- [getHelpPageId](#) ()
- [setHelpPageId](#) (\$id)
- [getParagraphParentId](#) ()
- [setParagraphParentId](#) (\$id)

- **getActionName** ()
- **setActionName** (\$name)
- **getTitle** ()
- **setTitle** (\$title)
- **getSettingName** ()
- **setSettingName** (\$name)
- **getHelpType** ()
- **setHelpType** (\$type)
- **getContent** ()

### Static Public Member Functions

- static **getTableName** ()
- static **getPrimaryKeys** ()
- static **getColumns** ()
- static **getColumnTypes** ()

### Protected Attributes

- **\$helpParagraphId**
- **\$helpPageId**
- **\$paragraphParentId**
- **\$actionName**
- **\$title**
- **\$settingName**

### 3.45.1 Detailed Description

[Entity](#) representing a paragraph in a help page.

### 3.45.2 Constructor & Destructor Documentation

#### 3.45.2.1 HelpParagraph::\_\_construct (\$id = null)

Constructs a [HelpPage](#) by id.

#### Parameters

*int* \$id Id of the page. Default (null) will create a new help page.

### 3.45.3 Member Function Documentation

#### 3.45.3.1 HelpParagraph::getChildren ()

Returns all the children of a node based on the current permissions

##### Returns

Array of help paragraphs.

#### 3.45.3.2 static HelpParagraph::getColumns () [static]

Gets all the columns.

##### Returns

Array of all columns, except primary keys.

Reimplemented from [Entity](#).

#### 3.45.3.3 static HelpParagraph::getColumnTypes () [static]

Gets all the column types, per column, including primary keys.

##### Returns

Array of all column types.

Reimplemented from [Entity](#).

#### 3.45.3.4 HelpParagraph::getContent ()

Retrieves the HTML content of this paragraph using the HelpContents table.

The entity should be loaded before this is called.

##### Returns

string The content of this paragraph.

#### 3.45.3.5 HelpParagraph::getHelpParagraphId ()

Getters and setters.

**3.45.3.6 static HelpParagraph::getPrimaryKeys () [static]**

Gets the primary keys.

**Returns**

Array of all primary keys.

Reimplemented from [Entity](#).

**3.45.3.7 static HelpParagraph::getTableName () [static]**

Gets the table name.

**Returns**

The table name.

Reimplemented from [Entity](#).

**3.45.3.8 HelpParagraph::getValues (\$ columns = null)**

Gets some values of this entity.

**Parameters**

*\$values* Array of names to fetch, or null if you want to fetch everything.

Reimplemented from [Entity](#).

**3.45.4 Member Data Documentation****3.45.4.1 HelpParagraph::\$actionName [protected]**

The action described by this paragraph (possibly null). Not displayed to users who can't do this action.

**3.45.4.2 HelpParagraph::\$helpPageId [protected]**

The page on which this paragraph is displayed.

**3.45.4.3 HelpParagraph::\$helpParagraphId [protected]**

Identifier.

**3.45.4.4 HelpParagraph::\$paragraphParentId [protected]**

The paragraph containing this one, or null.

**3.45.4.5 HelpParagraph::\$settingName [protected]**

If not null, the content depends on this setting.

**3.45.4.6 HelpParagraph::\$title [protected]**

The title of the paragraph.

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

- [www/backend/models/help/helpparagraph.php](http://www/backend/models/help/helpparagraph.php)



## 3.46 JavascriptMinifier Class Reference

### Public Member Functions

- [\\_\\_construct](#) (\$input)

### Static Public Member Functions

- static [minify](#) (\$js)

### Public Attributes

- const **ORD\_LF** = 10
- const **ORD\_SPACE** = 32
- const **ACTION\_KEEP\_A** = 1
- const **ACTION\_DELETE\_A** = 2
- const **ACTION\_DELETE\_A\_B** = 3

### Protected Member Functions

- [action](#) (\$command)
- [get](#) ()
- [isAlphaNum](#) (\$c)
- [min](#) ()
- [next](#) ()
- [peek](#) ()

### Protected Attributes

- **\$a** = ""
- **\$b** = ""
- **\$input** = ""
- **\$inputIndex** = 0
- **\$inputLength** = 0
- **\$lookAhead** = null
- **\$output** = ""

#### 3.46.1 Detailed Description

Javascript minifier class.

## 3.46.2 Constructor & Destructor Documentation

### 3.46.2.1 JavascriptMinifier::\_\_construct (\$ *input*)

Constructor.

#### Parameters

*string* \$input Javascript to be minified.

## 3.46.3 Member Function Documentation

### 3.46.3.1 JavascriptMinifier::action (\$ *command*) [protected]

Action -- do something! What to do is determined by the \$command argument.

action treats a string as a single character. Wow! action recognizes a regular expression if it is preceded by ( or , or =.

[next\(\)](#) [get\(\)](#)

#### Exceptions

*JavaScriptMinifierException* If parser errors are found:

- Unterminated string literal
- Unterminated regular expression set in regex literal
- Unterminated regular expression literal

#### Parameters

*int* \$command One of class constants: ACTION\_KEEP\_A Output A. Copy B to A. Get the next B. ACTION\_DELETE\_A Copy B to A. Get the next B. (Delete A). ACTION\_DELETE\_A\_B Get the next B. (Delete B).

### 3.46.3.2 JavascriptMinifier::get () [protected]

Get next char. Convert ctrl char to space.

#### Returns

string|null

### 3.46.3.3 JavascriptMinifier::isAlphaNum (\$ *c*) [protected]

Is \$c a letter, digit, underscore, dollar sign, or non-ASCII character.

#### Returns

bool

### 3.46.3.4 JavascriptMinifier::min () [protected]

Perform minification, return result

[action\(\)](#) [isAlphaNum\(\)](#)

#### Returns

string

### 3.46.3.5 static JavascriptMinifier::minify (\$js) [static]

Minifies Javascript.

[\\_\\_construct\(\)](#) [min\(\)](#)

#### Parameters

*string* \$js Javascript to be minified

#### Returns

string

### 3.46.3.6 JavascriptMinifier::next () [protected]

Get the next character, skipping over comments. [peek\(\)](#) is used to see if a '/' is followed by a '/' or '\*'.

[get\(\)](#) [peek\(\)](#)

#### Exceptions

*JavascriptMinifierException* On unterminated comment.

#### Returns

string

### 3.46.3.7 JavascriptMinifier::peek () [protected]

Get next char. If is ctrl character, translate to a space or newline.

[get\(\)](#)

#### Returns

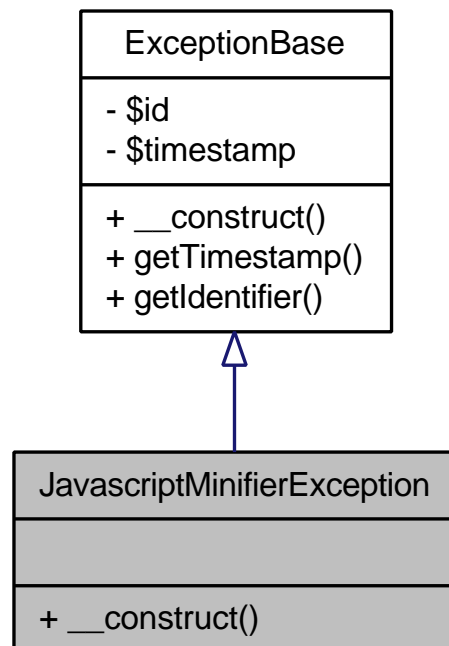
string|null

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

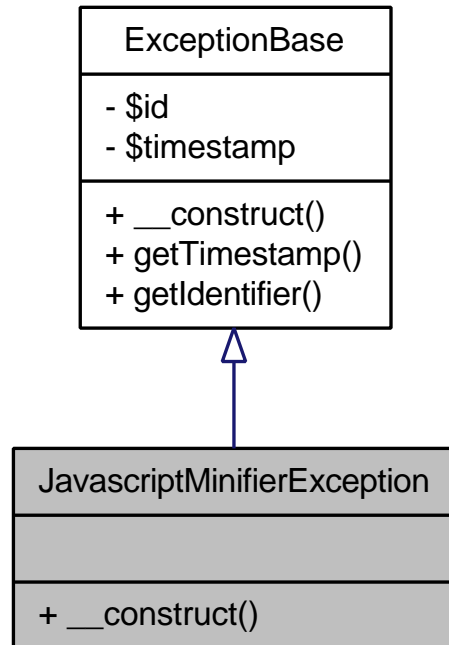
- [www/backend/util/jsmin.php](#)

### 3.47 JavascriptMinifierException Class Reference

Inheritance diagram for JavascriptMinifierException:



Collaboration diagram for JavascriptMinifierException:



## Public Member Functions

- `__construct` (\$msg)

### 3.47.1 Detailed Description

jsmin.php - PHP implementation of Douglas Crockford's JSMIn.

This is pretty much a direct port of jsmin.c to PHP with just a few PHP-specific performance tweaks. Also, whereas jsmin.c reads from stdin and outputs to stdout, this library accepts a string as input and returns another string as output.

PHP 5 or higher is required.

[Permission](#) is hereby granted to use this version of the library under the same terms as jsmin.c, which has the following license:

-- Copyright (c) 2002 Douglas Crockford (www.crockford.com)

[Permission](#) is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies

or substantial portions of the Software.

The Software shall be used for Good, not Evil.

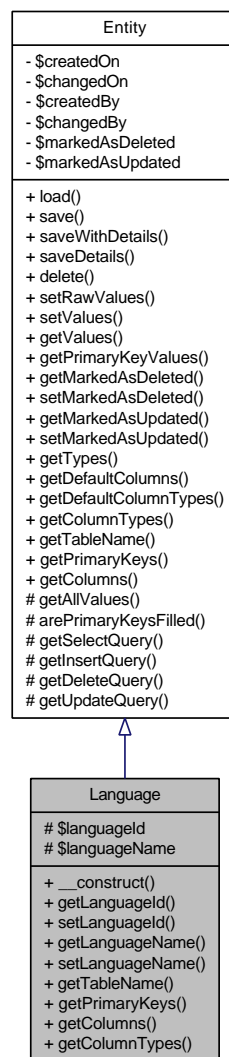
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE. Exceptions.

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

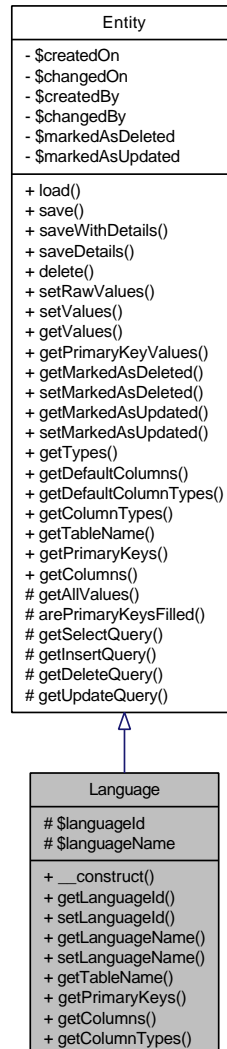
- [www/backend/util/jsmin.php](http://www/backend/util/jsmin.php)

## 3.48 Language Class Reference

Inheritance diagram for Language:



Collaboration diagram for Language:



## Public Member Functions

- [\\_\\_construct](#) (\$id=null)
- [getLanguageId](#) ()
- [setLanguageId](#) (\$id)
- [getLanguageName](#) ()
- [setLanguageName](#) (\$name)

## Static Public Member Functions

- static [getTableName](#) ()



- static [getPrimaryKeys \(\)](#)
- static [getColumns \(\)](#)
- static [getColumnTypes \(\)](#)

### Protected Attributes

- [\\$languageId](#)
- [\\$languageName](#)

### 3.48.1 Detailed Description

Class representing a language.

### 3.48.2 Constructor & Destructor Documentation

#### 3.48.2.1 `Language::__construct ($id = null)`

Constructs a language entity.

#### Parameters

*\$id* Id of the language. Default (null) will create a new one.

### 3.48.3 Member Function Documentation

#### 3.48.3.1 `static Language::getColumns () [static]`

Gets all the columns that are not primary keys as an array.

Reimplemented from [Entity](#).

#### 3.48.3.2 `static Language::getColumnTypes () [static]`

Gets all the column types, per column, including primary keys.

#### Returns

Array of all column types.

Reimplemented from [Entity](#).

#### 3.48.3.3 `Language::getLanguageId ()`

Getters and setters.

**3.48.3.4 static Language::getPrimaryKeys () [static]**

Get an array with the primary keys.

Reimplemented from [Entity](#).

**3.48.3.5 static Language::getTableName () [static]**

Get the name of the corresponding table.

Reimplemented from [Entity](#).

**3.48.4 Member Data Documentation****3.48.4.1 Language::\$languageId [protected]**

The identifier of a language.

**3.48.4.2 Language::\$languageName [protected]**

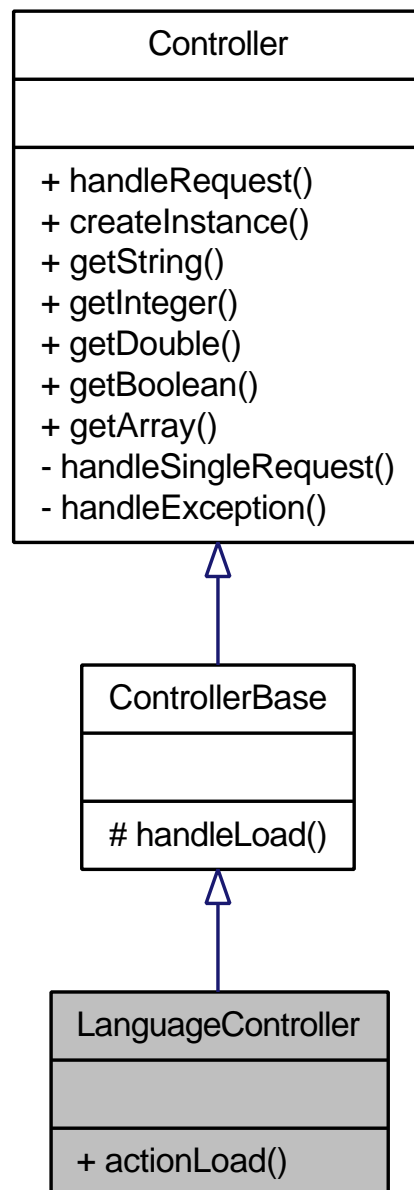
The name of the language.

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

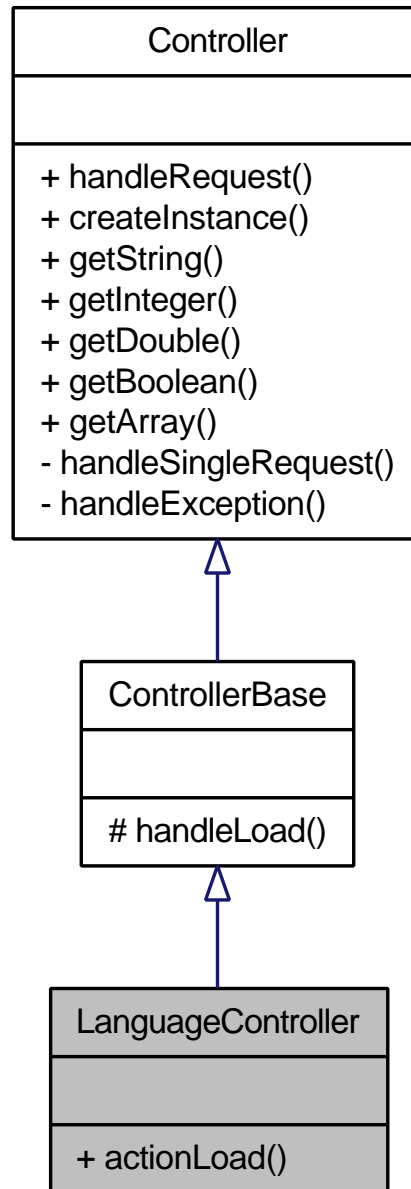
- [www/backend/models/language/language.php](#)

## 3.49 LanguageController Class Reference

Inheritance diagram for LanguageController:



Collaboration diagram for LanguageController:



## Public Member Functions

- [actionLoad](#) (\$data)

### 3.49.1 Detailed Description

[Language](#) controller class.

### 3.49.2 Member Function Documentation

#### 3.49.2.1 LanguageController::actionLoad (\$ *data*)

Loads languages.

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

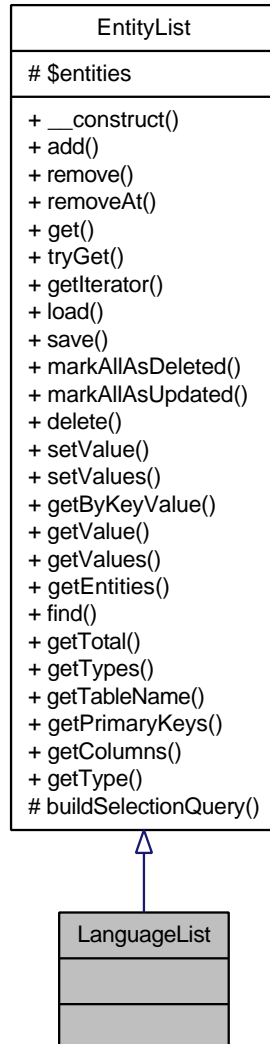
- [www/backend/controllers/languagecontroller.php](#)

### 3.50 LanguageList Class Reference

Inheritance diagram for LanguageList:



Collaboration diagram for LanguageList:



### 3.50.1 Detailed Description

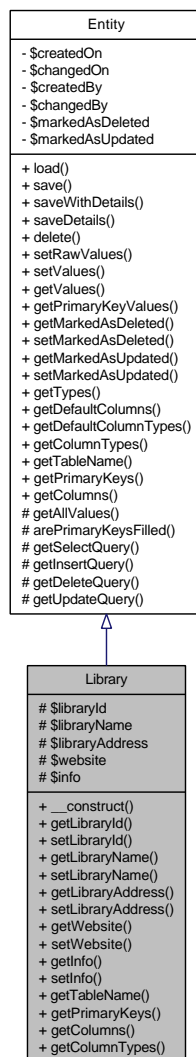
Class representing a language entity list.

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

- [www/backend/models/language/languagelist.php](http://www/backend/models/language/languagelist.php)

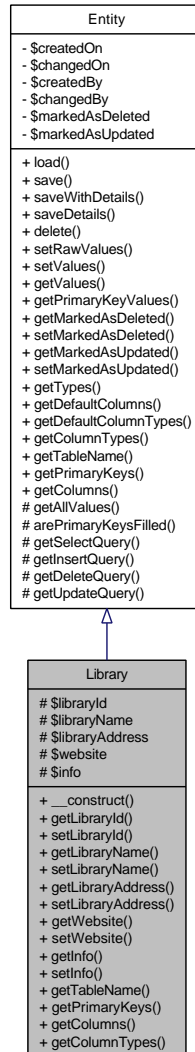
### 3.51 Library Class Reference

Inheritance diagram for Library:





Collaboration diagram for Library:



## Public Member Functions

- [\\_\\_construct](#) (\$libraryId=null)
- [getLibraryId](#) ()
- [setLibraryId](#) (\$libraryId)
- [getLibraryName](#) ()
- [setLibraryName](#) (\$libraryName)
- [getLibraryAddress](#) ()
- [setLibraryAddress](#) (\$libraryAddress)
- [getWebsite](#) ()
- [setWebsite](#) (\$website)

- **getInfo** ()
- **setInfo** (\$info)

### Static Public Member Functions

- static [getTableName](#) ()
- static [getPrimaryKeys](#) ()
- static [getColumns](#) ()
- static [getColumnTypes](#) ()

### Protected Attributes

- [\\$libraryId](#)
- [\\$libraryName](#)
- [\\$libraryAddress](#)
- [\\$website](#)
- [\\$info](#)

## 3.51.1 Detailed Description

Class representing a library entity.

## 3.51.2 Constructor & Destructor Documentation

### 3.51.2.1 Library::\_\_construct (\$ *libraryId* = null)

Constructs a library by id.

#### Parameters

*\$id* Id of the library. Default (null) will create a new library.

## 3.51.3 Member Function Documentation

### 3.51.3.1 static Library::getColumns () [static]

Gets all the columns.

#### Returns

Array of all columns, except primary keys.

Reimplemented from [Entity](#).

**3.51.3.2 static Library::getColumnTypes () [static]**

Gets all the column types, per column, including primary keys.

**Returns**

Array of all column types.

Reimplemented from [Entity](#).

**3.51.3.3 Library::getLibraryId ()**

Getters and setters.

**3.51.3.4 static Library::getPrimaryKeys () [static]**

Gets the primary keys.

**Returns**

Array of all primary keys.

Reimplemented from [Entity](#).

**3.51.3.5 static Library::getTableName () [static]**

Gets the table name.

**Returns**

The table name.

Reimplemented from [Entity](#).

**3.51.4 Member Data Documentation****3.51.4.1 Library::\$info [protected]**

Info text about this library.

**3.51.4.2 Library::\$libraryAddress [protected]**

[Library](#) address

**3.51.4.3 Library::\$libraryId [protected]**

Id of this library.

**3.51.4.4 Library::\$libraryName [protected]**

[Library](#) name.

**3.51.4.5 Library::\$website [protected]**

[Library](#) website.

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

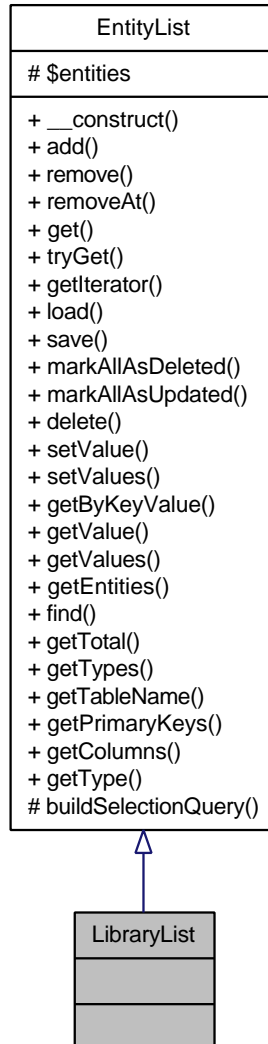
- [www/backend/models/library/library.php](#)

## 3.52 LibraryList Class Reference

Inheritance diagram for LibraryList:



Collaboration diagram for LibraryList:



### 3.52.1 Detailed Description

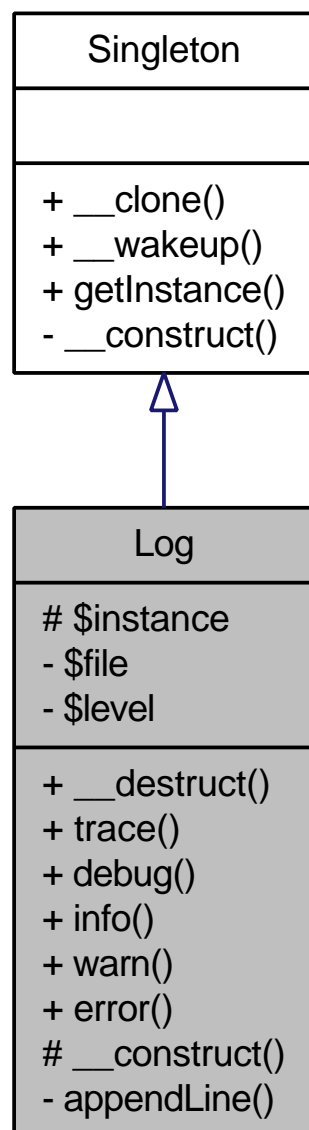
Class representing a library entity list.

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

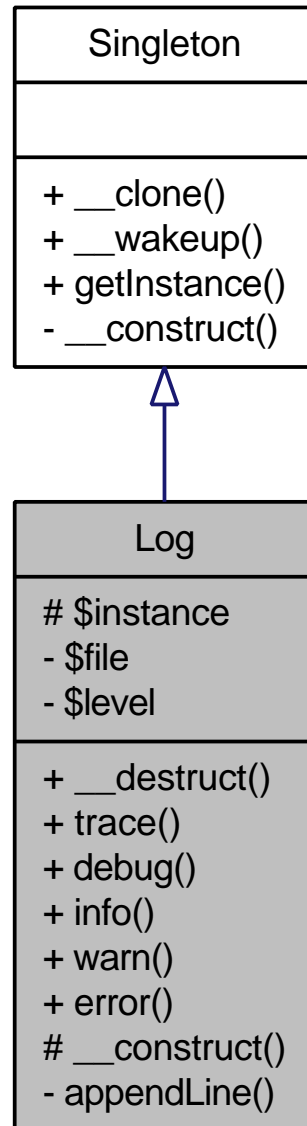
- [www/backend/models/library/librarylist.php](http://www/backend/models/library/librarylist.php)

## 3.53 Log Class Reference

Inheritance diagram for Log:



Collaboration diagram for Log:



### Public Member Functions

- `__destruct ()`

### Static Public Member Functions

- static `trace ($format)`
- static `debug ($format)`



- static [info](#) (\$format)
- static [warn](#) (\$format)
- static [error](#) (\$format)

## Protected Member Functions

- [\\_\\_construct](#) ()

## Static Protected Attributes

- static [\\$instance](#)

### 3.53.1 Detailed Description

[Log](#) class.

### 3.53.2 Constructor & Destructor Documentation

#### 3.53.2.1 [Log::\\_\\_construct](#) () [protected]

Constructs a log class instance.

Reimplemented from [Singleton](#).

#### 3.53.2.2 [Log::\\_\\_destruct](#) ()

Closes log file.

### 3.53.3 Member Function Documentation

#### 3.53.3.1 [static Log::debug](#) (\$format) [static]

Adds a debug message.

##### Parameters

*\$format* The format of the message.

... The arguments of the format.

#### 3.53.3.2 [static Log::error](#) (\$format) [static]

Adds an error message.

**Parameters**

*\$format* The format of the message.  
... The arguments of the format.

**3.53.3.3 static Log::info (\$format) [static]**

Adds an info message.

**Parameters**

*\$format* The format of the message.  
... The arguments of the format.

**3.53.3.4 static Log::trace (\$format) [static]**

Adds a trace message.

**Parameters**

*\$format* The format of the message.  
... The arguments of the format.

**3.53.3.5 static Log::warn (\$format) [static]**

Adds a warning message.

**Parameters**

*\$format* The format of the message.  
... The arguments of the format.

**3.53.4 Member Data Documentation****3.53.4.1 Log::\$instance [static, protected]**

Unique instance.

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

- [www/backend/framework/util/log.php](http://www/backend/framework/util/log.php)

## 3.54 Mailer Class Reference

### Static Public Member Functions

- static [sendMail](#) (\$recipient, \$subject, \$message)
- static [sendActivationMail](#) (\$puser)
- static [sendUserDeclinedMail](#) (\$puser)
- static [sendPasswordRestorationMail](#) (\$user)

### 3.54.1 Detailed Description

General helper class for sending e-mails.

### 3.54.2 Member Function Documentation

#### 3.54.2.1 static Mailer::sendActivationMail (\$puser) [static]

Sends a standard activation mail to the specified [PendingUser](#) containing his or her activation code.

#### Parameters

[PendingUser](#) \$puser The pending user entity, its values should be set.

#### Exceptions

[MailerException](#) When something goes wrong, like the e-mail not being accepted for delivery.

#### 3.54.2.2 static Mailer::sendMail (\$recipient, \$subject, \$message) [static]

Sends an e-mail. Opens up and closes an SMTP connection and therefore is not very suitable for 'mass mailing' a large numbers of users at once.

#### Parameters

*string* \$recipient The e-mail address of the recipient.

*string* \$subject The subject of the mail.

*string* \$message The message in the body of the mail.

#### Exceptions

[MailException](#) When an e-mail is not accepted for delivery.

**3.54.2.3 static Mailer::sendPasswordRestorationMail (\$user) [static]**

Sends a standard password restoration mail to the specified [User](#).

**Parameters**

[User](#) \$user The user entity, its values should be set.

**Exceptions**

[MailerException](#) When something goes wrong, like the e-mail not being accepted for delivery.

**3.54.2.4 static Mailer::sendUserDeclinedMail (\$puser) [static]**

Sends a standard registration declined mail to the specified [PendingUser](#).

**Parameters**

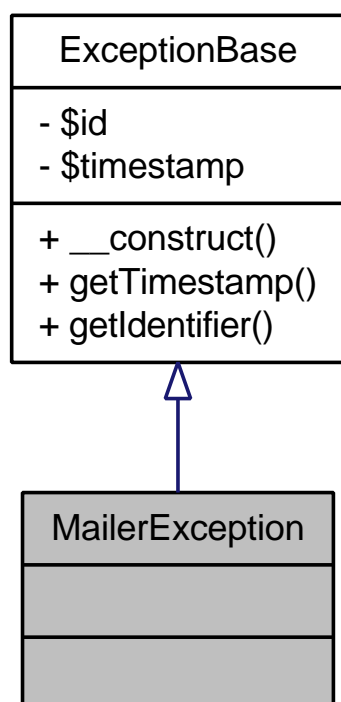
[PendingUser](#) \$puser The pending user entity, its values should be set.

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

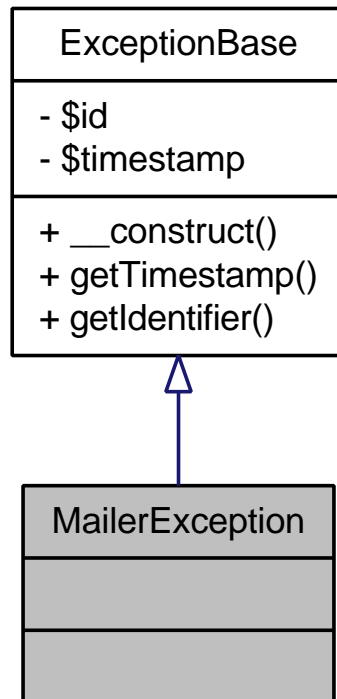
- [www/backend/util/mailer.php](#)

## 3.55 MailerException Class Reference

Inheritance diagram for MailerException:



Collaboration diagram for MailerException:



### 3.55.1 Detailed Description

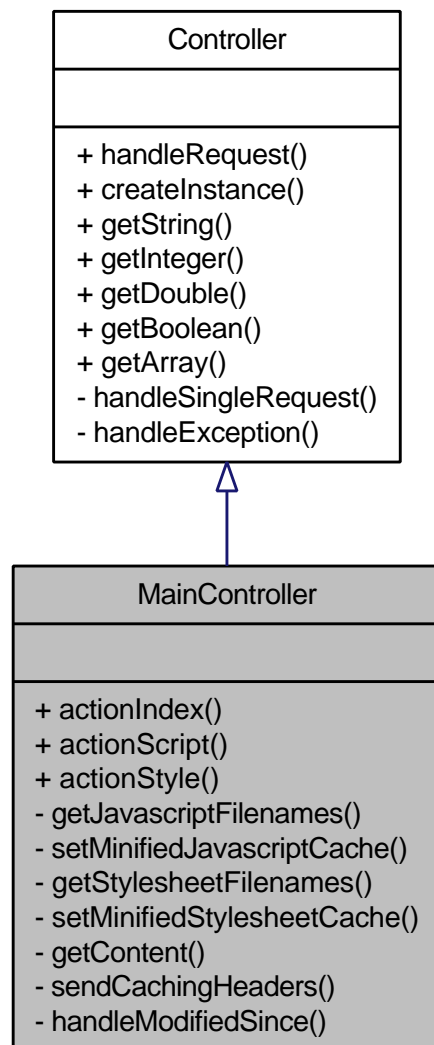
Exceptions.

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

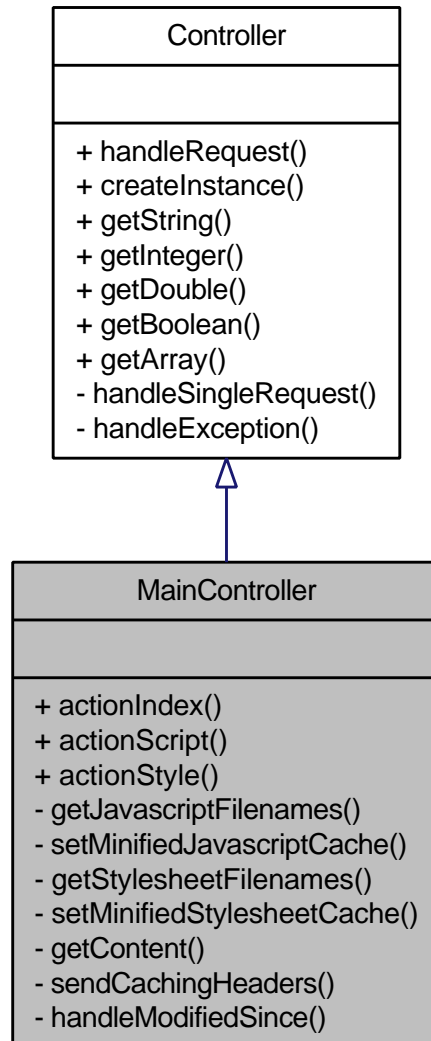
- [www/backend/util/mailer.php](http://www/backend/util/mailer.php)

## 3.56 MainController Class Reference

Inheritance diagram for MainController:



Collaboration diagram for MainController:



### Public Member Functions

- [actionIndex](#) (\$data)
- [actionScript](#) (\$data)
- [actionStyle](#) (\$data)

### 3.56.1 Detailed Description

Main controller class.



## 3.56.2 Member Function Documentation

### 3.56.2.1 MainController::actionIndex (\$ *data*)

Loads index page.

### 3.56.2.2 MainController::actionScript (\$ *data*)

Loads Javascript.

### 3.56.2.3 MainController::actionStyle (\$ *data*)

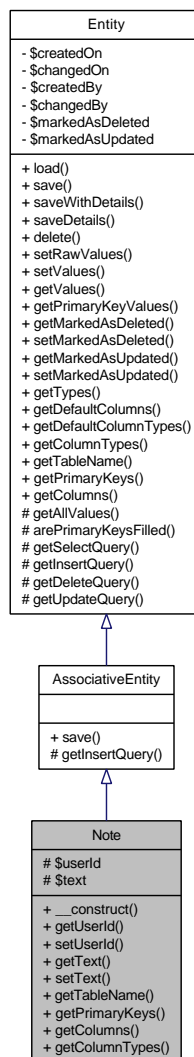
Loads stylesheets.

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

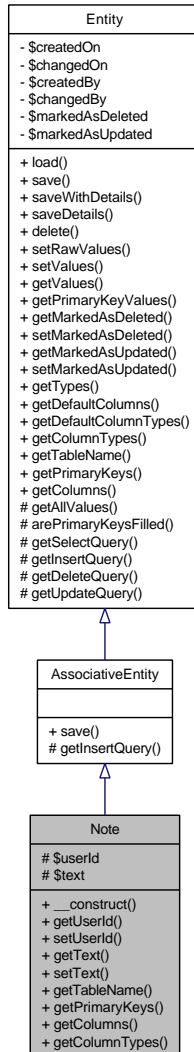
- [www/backend/controllers/maincontroller.php](#)

### 3.57 Note Class Reference

Inheritance diagram for Note:



Collaboration diagram for Note:



## Public Member Functions

- [\\_\\_construct](#) (\$id=null)
- [getUserId](#) ()
- [setUserId](#) (\$userId)
- [getText](#) ()
- [setText](#) (\$text)

## Static Public Member Functions

- static [getTableName](#) ()

- static [getPrimaryKeys \(\)](#)
- static [getColumns \(\)](#)
- static [getColumnTypes \(\)](#)

## Protected Attributes

- [\\$userId](#)
- [\\$text](#)

### 3.57.1 Detailed Description

Class representing a note entity.

### 3.57.2 Constructor & Destructor Documentation

#### 3.57.2.1 `Note::__construct ($ id = null)`

Constructs a note by user id.

#### Parameters

*\$id* Id of the user. Default (null) will create a new note.

### 3.57.3 Member Function Documentation

#### 3.57.3.1 `static Note::getColumns () [static]`

Gets all the columns.

#### Returns

Array of all columns, except primary keys.

Reimplemented from [Entity](#).

#### 3.57.3.2 `static Note::getColumnTypes () [static]`

Gets all the column types, per column, including primary keys.

#### Returns

Array of all column types.

Reimplemented from [Entity](#).

### 3.57.3.3 static Note::getPrimaryKeys () [static]

Gets the primary keys.

#### Returns

Array of all primary keys.

Reimplemented from [Entity](#).

### 3.57.3.4 static Note::getTableName () [static]

Gets the table name.

#### Returns

The table name.

Reimplemented from [Entity](#).

### 3.57.3.5 Note::getUserId ()

Getters and setters.

## 3.57.4 Member Data Documentation

### 3.57.4.1 Note::\$text [protected]

Text inside the note.

### 3.57.4.2 Note::\$userId [protected]

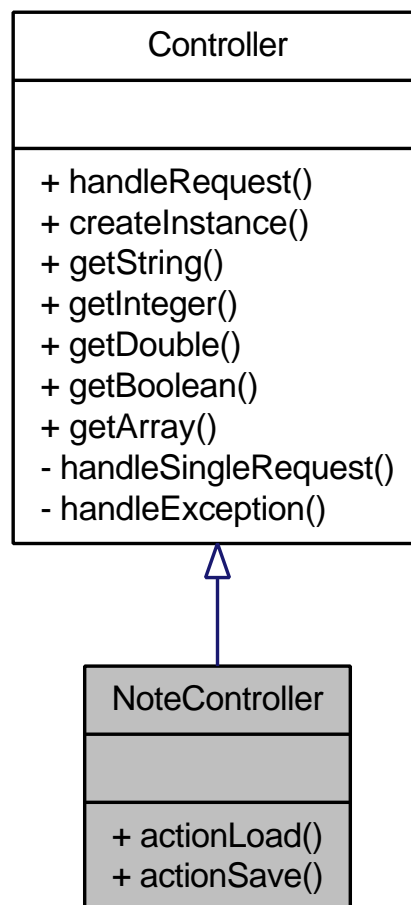
Id of the user who made the notes.

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

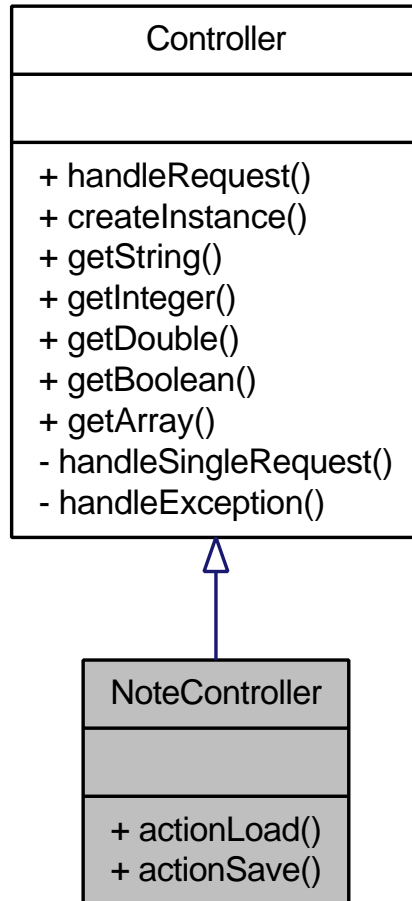
- `www/backend/models/note/note.php`

### 3.58 NoteController Class Reference

Inheritance diagram for NoteController:



Collaboration diagram for NoteController:



### Public Member Functions

- [actionLoad](#) (\$data)
- [actionSave](#) (\$data)

### 3.58.1 Detailed Description

[Note](#) controller class.

### 3.58.2 Member Function Documentation

#### 3.58.2.1 NoteController::actionLoad (\$ data)

Loads a note.

### 3.58.2.2 NoteController::actionSave (\$ *data*)

Saves a note.

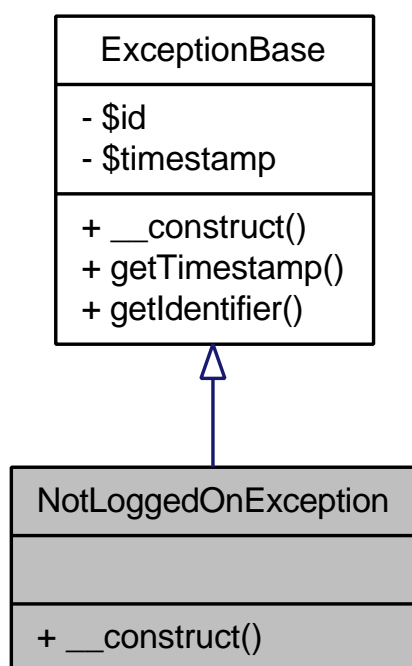
The documentation for this class was generated from the following file:

- [www/backend/controllers/notecontroller.php](http://www/backend/controllers/notecontroller.php)

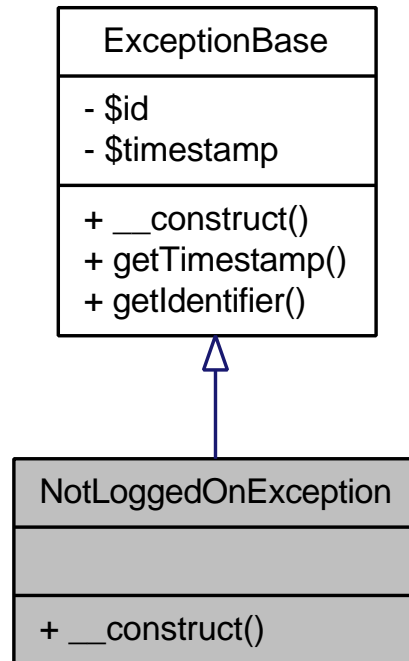


## 3.59 NotLoggedOnException Class Reference

Inheritance diagram for NotLoggedOnException:



Collaboration diagram for NotLoggedOnException:



### 3.59.1 Detailed Description

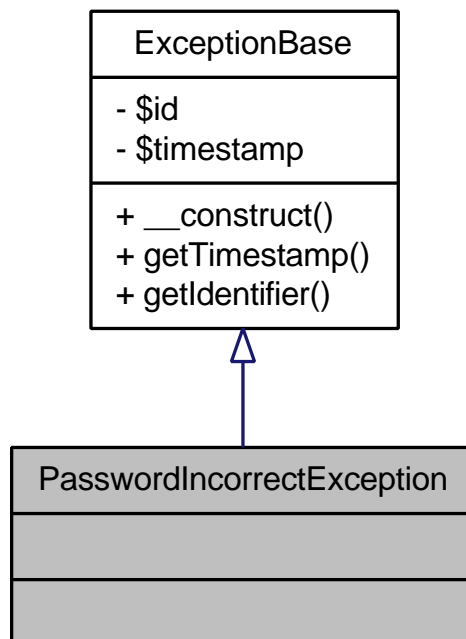
Exceptions.

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

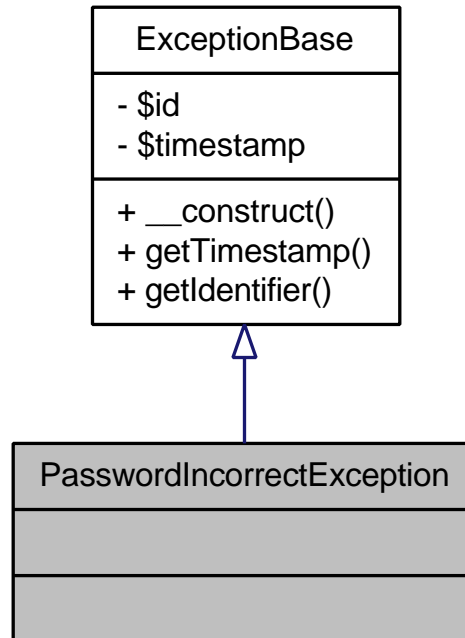
- [www/backend/util/authentication.php](http://www/backend/util/authentication.php)

## 3.60 PasswordIncorrectException Class Reference

Inheritance diagram for PasswordIncorrectException:



Collaboration diagram for PasswordIncorrectException:



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

- [www/backend/controllers/usercontroller.php](#)

## 3.61 Pdf Class Reference

### Public Member Functions

- `__construct` ([Binding](#) \$binding, [CacheEntry](#) \$cacheEntry, \$range=null, \$transcriptions=null, \$annotations=false)

#### 3.61.1 Detailed Description

A PDF exporting class. Exports bindings or a subset thereof to the PDF format.

This class has no public methods other than its constructor: it immediately exports to the given [CacheEntry](#).

All dimensions are in points, equalling 1/72 inch, unless indicated otherwise.

#### 3.61.2 Constructor & Destructor Documentation

**3.61.2.1 Pdf::\_\_construct** ([Binding](#) \$ *binding*, [CacheEntry](#) \$ *cacheEntry*, \$ *range* = null, \$ *transcriptions* = null, \$ *annotations* = false)

Creates a new PDF based on the given scan.

##### Parameters

***Binding*** \$binding The [Binding](#) to export.

***CacheEntry*** \$cacheEntry The [CacheEntry](#) to export to.

***mixed*** \$range The range of pages to export. This may be null for all, a number for a single page and an array of two numbers for a range.

***callback*** \$transcriptions Function that extracts required transcription languages from Annotations.

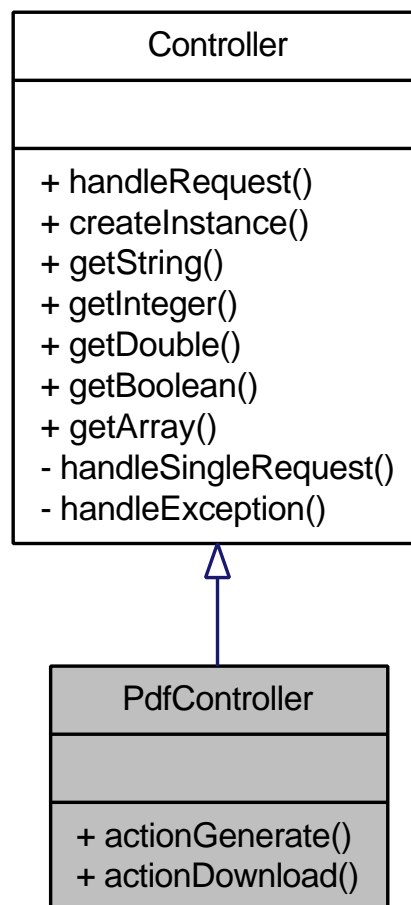
***boolean*** \$annotations Whether to include annotations on the scans. Only valid on combination with transcriptions parameter set.

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

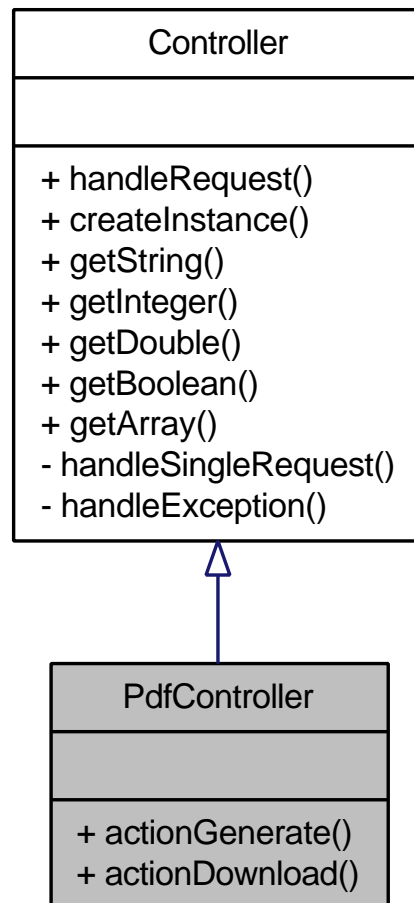
- `www/backend/util/pdf.php`

### 3.62 PdfController Class Reference

Inheritance diagram for PdfController:



Collaboration diagram for PdfController:



### Public Member Functions

- [actionGenerate](#) (\$data)
- [actionDownload](#) (\$data)

#### 3.62.1 Detailed Description

PDF controller class.

## 3.62.2 Member Function Documentation

### 3.62.2.1 PdfController::actionDownload (\$ *data*)

Outputs the PDF as a file for downloading. Also sets the correct headers for file download.

### 3.62.2.2 PdfController::actionGenerate (\$ *data*)

Generates the PDF and stores it in a cache file.

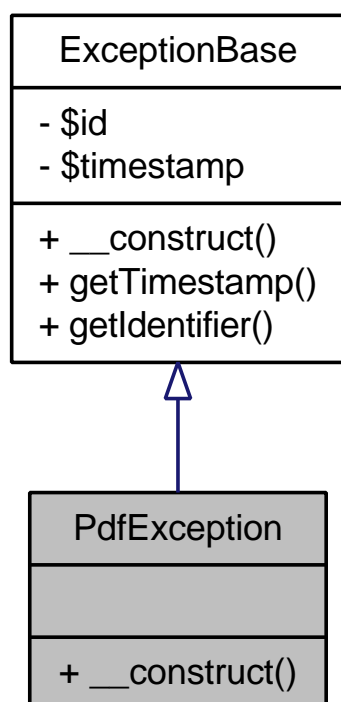
The documentation for this class was generated from the following file:

- [www/backend/controllers/pdfcontroller.php](http://www/backend/controllers/pdfcontroller.php)

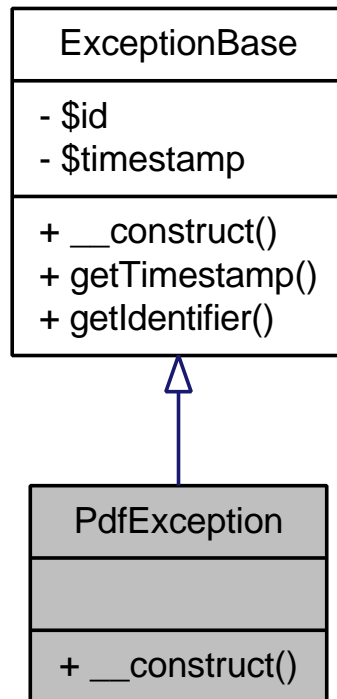


## 3.63 PdfException Class Reference

Inheritance diagram for PdfException:



Collaboration diagram for PdfException:

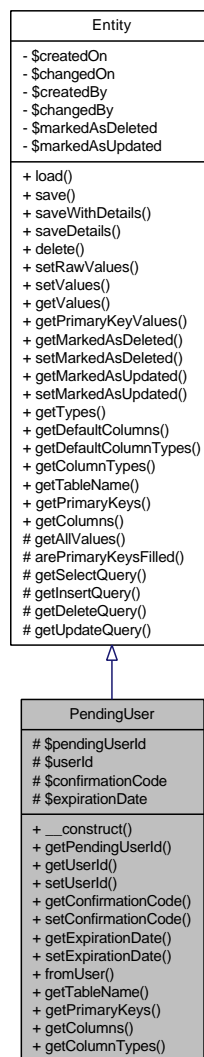


The documentation for this class was generated from the following file:

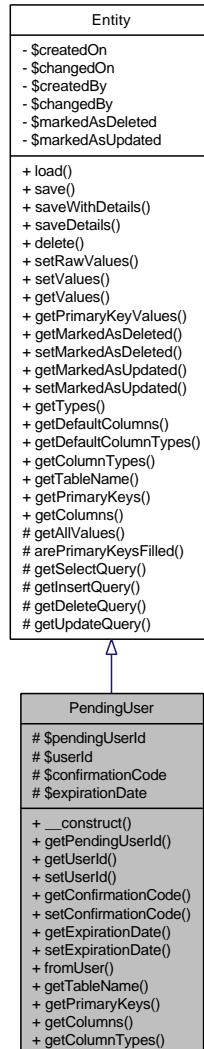
- [www/backend/util/pdf.php](http://www/backend/util/pdf.php)

## 3.64 PendingUser Class Reference

Inheritance diagram for PendingUser:



Collaboration diagram for PendingUser:



## Public Member Functions

- [\\_\\_construct](#) (\$id=null)
- [getPendingUserId](#) ()
- [getUserId](#) ()
- [setUserId](#) (\$id)
- [getConfirmationCode](#) ()
- [setConfirmationCode](#) (\$code)
- [getExpirationDate](#) ()
- [setExpirationDate](#) (\$date)

## Static Public Member Functions

- static [fromUser](#) (\$user)
- static [getTableName](#) ()
- static [getPrimaryKeys](#) ()
- static [getColumns](#) ()
- static [getColumnTypes](#) ()

## Protected Attributes

- [\\$pendingUserId](#)
- [\\$userId](#)
- [\\$confirmationCode](#)
- [\\$expirationDate](#)

### 3.64.1 Detailed Description

Class representing a yet unactivated user entity.

### 3.64.2 Constructor & Destructor Documentation

#### 3.64.2.1 PendingUser::\_\_construct (\$id = null)

Constructs a pending user.

##### Parameters

*\$id* Id of the pending user. Default (null) will create a new pending user.

### 3.64.3 Member Function Documentation

#### 3.64.3.1 static PendingUser::fromUser (\$user) [static]

Constructs a new [PendingUser](#) based on an existing [User](#). This method will also generate a confirmation code and set a expiration date. Accepted will be set to NULL.

This new pending user can subsequently be saved to create a new database entry or update an existing one.

##### Returns

[PendingUser](#) The new pending user.

#### 3.64.3.2 static PendingUser::getColumns () [static]

Gets all the columns that are not primary keys as an array.

Reimplemented from [Entity](#).

**3.64.3.3 static PendingUser::getColumnTypes () [static]**

Gets all the column types, per column, including primary keys.

**Returns**

Array of all column types.

Reimplemented from [Entity](#).

**3.64.3.4 PendingUser::getPendingUserId ()**

Getters and setters.

**3.64.3.5 static PendingUser::getPrimaryKeys () [static]**

Get an array with the primary keys.

Reimplemented from [Entity](#).

**3.64.3.6 static PendingUser::getTableName () [static]**

Get the name of the corresponding table.

Reimplemented from [Entity](#).

**3.64.4 Member Data Documentation****3.64.4.1 PendingUser::\$confirmationCode [protected]**

Confirmation code.

**3.64.4.2 PendingUser::\$expirationDate [protected]**

The date of when this pending user will expire. Will be a timestamp.

**3.64.4.3 PendingUser::\$pendingUserId [protected]**

Pending user id

**3.64.4.4 PendingUser::\$userId [protected]**

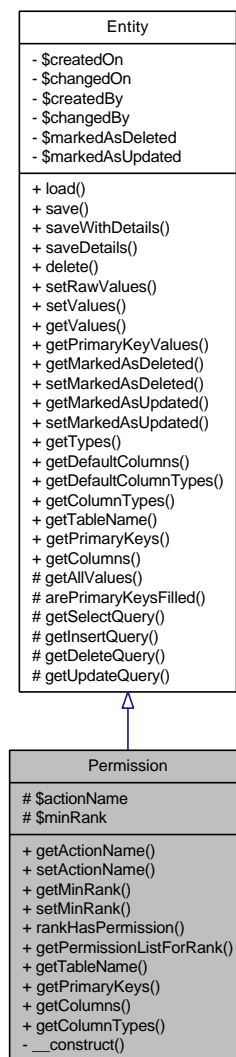
[User](#) id.

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

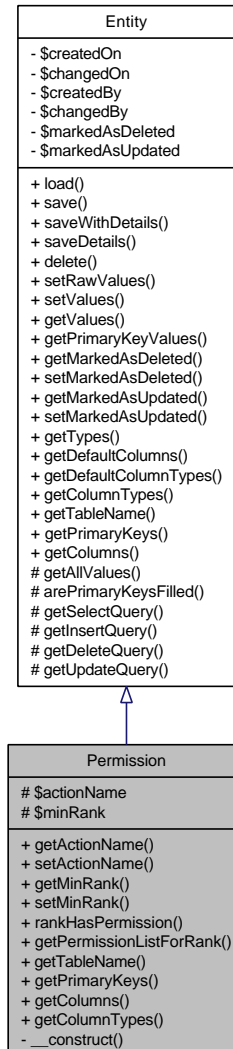
- [www/backend/models/user/pendinguser.php](#)

## 3.65 Permission Class Reference

Inheritance diagram for Permission:



Collaboration diagram for Permission:



## Public Member Functions

- [getActionName](#) ()
- [setActionName](#) (\$name)
- [getMinRank](#) ()
- [setMinRank](#) (\$rank)

## Static Public Member Functions

- static [rankHasPermission](#) (\$action, \$rank)
- static [getPermissionListForRank](#) (\$rank)



- static [getTableName \(\)](#)
- static [getPrimaryKeys \(\)](#)
- static [getColumns \(\)](#)
- static [getColumnTypes \(\)](#)

## Protected Attributes

- [\\$actionName](#)
- [\\$minRank](#)

### 3.65.1 Detailed Description

[Permission](#) entity. This represents actions restricted to specific users.

### 3.65.2 Member Function Documentation

#### 3.65.2.1 [Permission::getActionName \(\)](#)

Getters and setters.

#### 3.65.2.2 [static Permission::getColumns \(\)](#) [**static**]

Gets all the columns that are not primary keys as an array.

Reimplemented from [Entity](#).

#### 3.65.2.3 [static Permission::getColumnTypes \(\)](#) [**static**]

Gets all the column types, per column, including primary keys.

#### Returns

Array of all column types.

Reimplemented from [Entity](#).

#### 3.65.2.4 [static Permission::getPermissionListForRank \(\\$rank\)](#) [**static**]

Returns an array of all actions a user with the specified rank is allowed to perform.

#### Parameters

*int* \$rank The rank of the kind of user for which to receive the permission list.

#### Returns

array An array of strings representing action names.

### 3.65.2.5 static `Permission::getPrimaryKeys ()` [`static`]

Get an array with the primary keys.

Reimplemented from [Entity](#).

### 3.65.2.6 static `Permission::getTableName ()` [`static`]

Get the name of the corresponding table.

Reimplemented from [Entity](#).

### 3.65.2.7 static `Permission::rankHasPermission ($action, $rank)` [`static`]

Indicates whether some user with the provided rank has permission to undertake the action with a certain name.

#### Parameters

*string* `$action` The name of the action.

*int* `$rank` The user rank for which to test the permission.

#### Returns

bool true if the rank is high enough for this action, false otherwise.

#### Exceptions

[EntityException](#) If no action with this name could be found in the database.

## 3.65.3 Member Data Documentation

### 3.65.3.1 `Permission::$actionName` [`protected`]

Name of the associated action.

### 3.65.3.2 `Permission::$minRank` [`protected`]

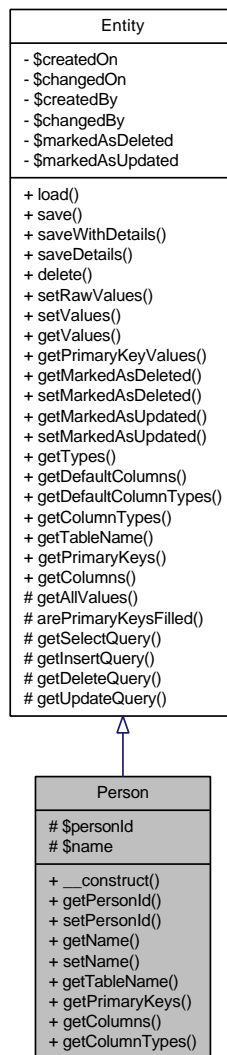
Minimal rank required to undertake this action.

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

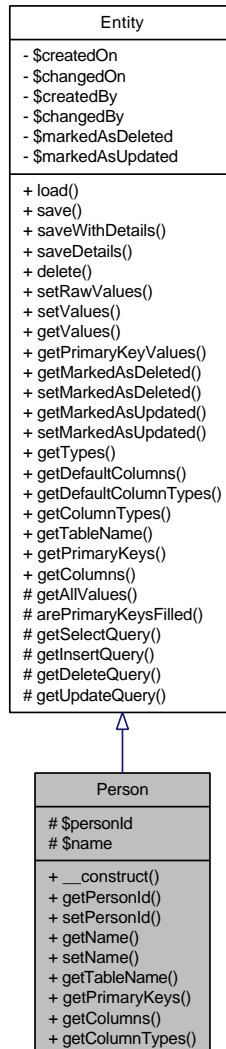
- `www/backend/models/permission/permission.php`

## 3.66 Person Class Reference

Inheritance diagram for Person:



Collaboration diagram for Person:



## Public Member Functions

- `__construct` (\$personId=null)
- `getPersonId` ()
- `setPersonId` (\$personId)
- `getName` ()
- `setName` (\$name)

## Static Public Member Functions

- static `getTableName` ()

- static [getPrimaryKeys](#) ()
- static [getColumns](#) ()
- static [getColumnTypes](#) ()

### Protected Attributes

- [\\$personId](#)
- [\\$name](#)

### 3.66.1 Detailed Description

Class representing a person entity.

### 3.66.2 Constructor & Destructor Documentation

#### 3.66.2.1 `Person::__construct ($personId = null)`

Constructs a person by id.

#### Parameters

*\$id* Id of the person. Default (null) will create a new person.

### 3.66.3 Member Function Documentation

#### 3.66.3.1 `static Person::getColumns () [static]`

Gets all the columns.

#### Returns

Array of all columns, except primary keys.

Reimplemented from [Entity](#).

#### 3.66.3.2 `static Person::getColumnTypes () [static]`

Gets all the column types, per column, including primary keys.

#### Returns

Array of all column types.

Reimplemented from [Entity](#).

### 3.66.3.3 `Person::getPersonId ()`

Getters and setters.

### 3.66.3.4 `static Person::getPrimaryKeys () [static]`

Gets the primary keys.

#### Returns

Array of all primary keys.

Reimplemented from [Entity](#).

### 3.66.3.5 `static Person::getTableName () [static]`

Gets the table name.

#### Returns

The table name.

Reimplemented from [Entity](#).

## 3.66.4 Member Data Documentation

### 3.66.4.1 `Person::$name [protected]`

Full name of the person.

### 3.66.4.2 `Person::$personId [protected]`

[Person](#) id.

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

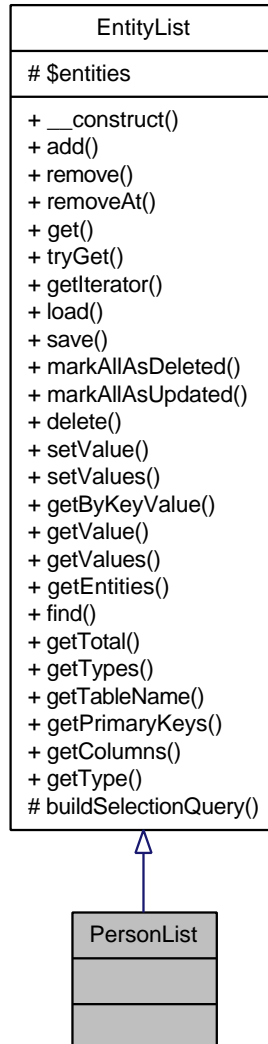
- `www/backend/models/person/person.php`

## 3.67 PersonList Class Reference

Inheritance diagram for PersonList:



Collaboration diagram for PersonList:



### 3.67.1 Detailed Description

Class representing a person entity list.

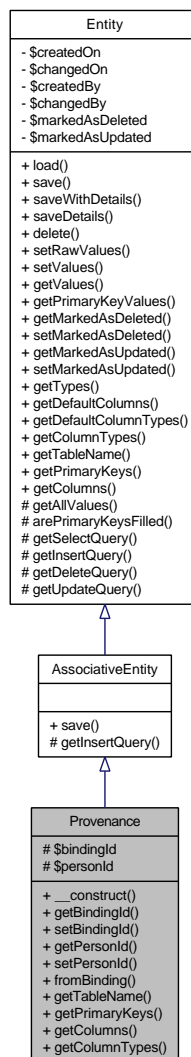
The documentation for this class was generated from the following file:

- [www/backend/models/person/personlist.php](http://www/backend/models/person/personlist.php)

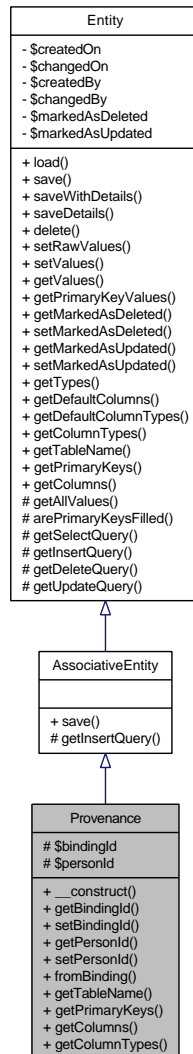


## 3.68 Provenance Class Reference

Inheritance diagram for Provenance:



Collaboration diagram for Provenance:



## Public Member Functions

- `__construct` (\$bindingId=null, \$personId=null)
- `getBindingId` ()
- `setBindingId` (\$bindingId)
- `getPersonId` ()
- `setPersonId` (\$personId)

## Static Public Member Functions

- static `fromBinding` (\$binding)

- static [getTableName \(\)](#)
- static [getPrimaryKeys \(\)](#)
- static [getColumns \(\)](#)
- static [getColumnTypes \(\)](#)

## Protected Attributes

- [\\$bindingId](#)
- [\\$personId](#)

### 3.68.1 Detailed Description

Class representing a provenance entity. Associative between binding and person.

### 3.68.2 Constructor & Destructor Documentation

#### 3.68.2.1 Provenance::\_\_construct (*\$ bindingId* = null, *\$ personId* = null)

Constructs a provenance by binding and person ids.

#### Parameters

*\$bindingId* Id of the binding. Default (null) will create a new provenance.

*\$personId* Id of the person. Default (null) will create a new provenance.

### 3.68.3 Member Function Documentation

#### 3.68.3.1 static Provenance::fromBinding (*\$ binding*) [static]

Returns all the readers of one binding

#### Parameters

*\$binding* The binding model

#### Returns

Array of provenance models

#### 3.68.3.2 Provenance::getBindingId ()

Getters and setters.

### 3.68.3.3 static Provenance::getColumns () [static]

Gets all the columns.

#### Returns

Array of all columns, except primary keys.

Reimplemented from [Entity](#).

### 3.68.3.4 static Provenance::getColumnTypes () [static]

Gets all the column types, per column, including primary keys.

#### Returns

Array of all column types.

Reimplemented from [Entity](#).

### 3.68.3.5 static Provenance::getPrimaryKeys () [static]

Gets the primary keys.

#### Returns

Array of all primary keys.

Reimplemented from [Entity](#).

### 3.68.3.6 static Provenance::getTableName () [static]

Gets the table name.

#### Returns

The table name.

Reimplemented from [Entity](#).

## 3.68.4 Member Data Documentation

### 3.68.4.1 Provenance::\$bindingId [protected]

[Binding](#) id.

#### 3.68.4.2 Provenance::\$personId [protected]

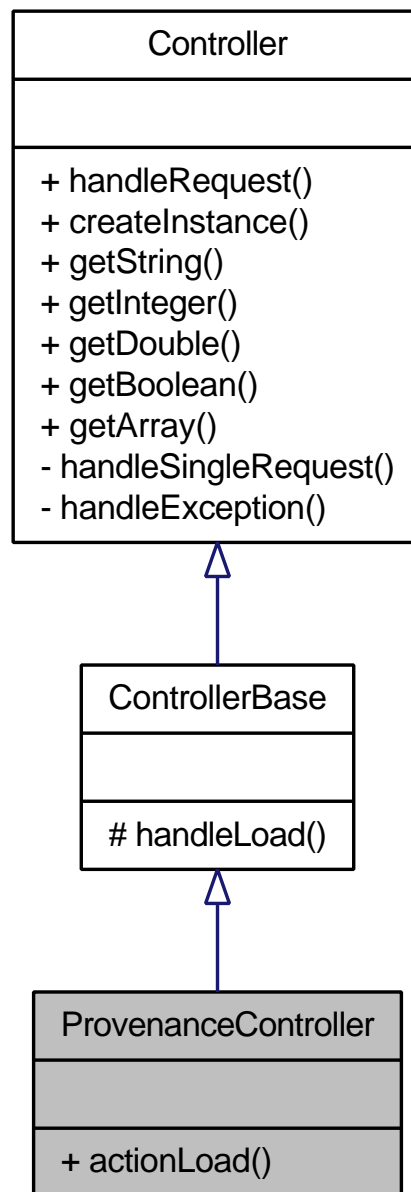
[Person](#) id.

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

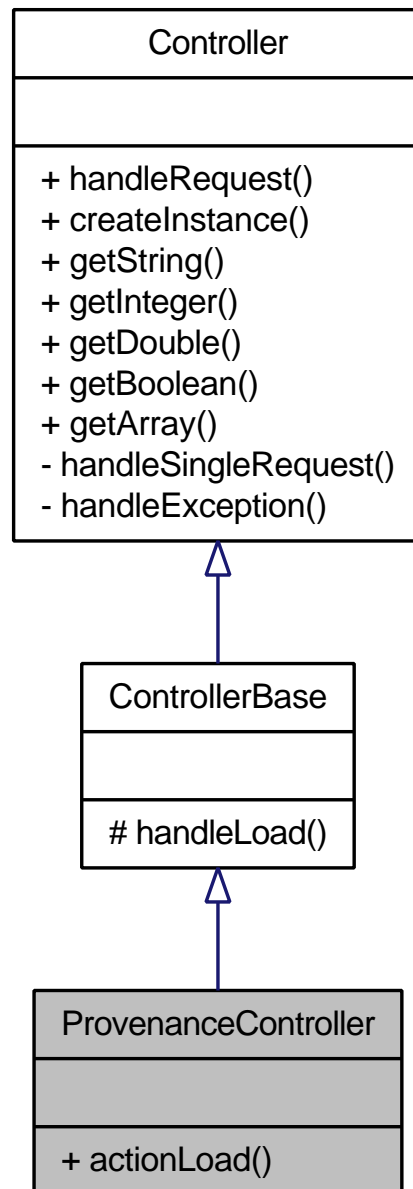
- [www/backend/models/provenance/provenance.php](#)

### 3.69 ProvenanceController Class Reference

Inheritance diagram for ProvenanceController:



Collaboration diagram for ProvenanceController:



## Public Member Functions

- [actionLoad](#) (\$data)

### 3.69.1 Detailed Description

[Provenance](#) controller class.

### 3.69.2 Member Function Documentation

#### 3.69.2.1 ProvenanceController::actionLoad (\$ *data*)

Loads provenances.

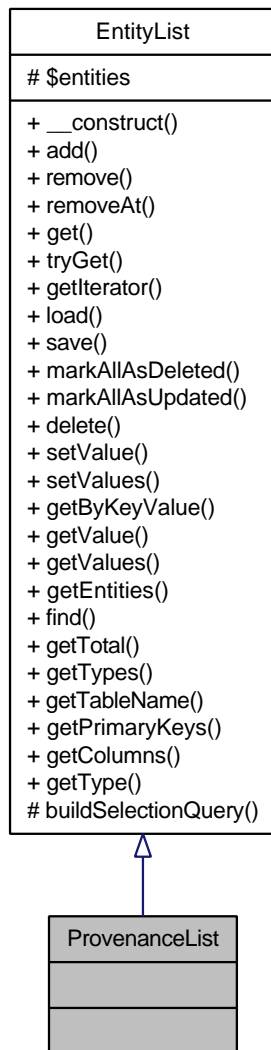
The documentation for this class was generated from the following file:

- [www/backend/controllers/provenancecontroller.php](#)

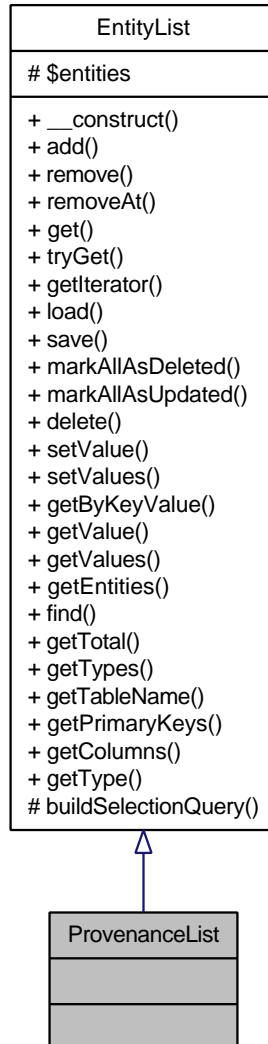


## 3.70 ProvenanceList Class Reference

Inheritance diagram for ProvenanceList:



Collaboration diagram for ProvenanceList:



### 3.70.1 Detailed Description

Class representing a provenance entity list.

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

- [www/backend/models/provenance/provenancelist.php](http://www/backend/models/provenance/provenancelist.php)

## 3.71 Query Class Reference

### Public Member Functions

- [clear](#) (\$fields=null)
- [columns](#) ()
- [count](#) (\$column= '\*', \$as= 'total')
- [aggregate](#) (\$function, \$as= 'aggregate', \$column= '\*')
- [headline](#) (\$columns, \$query, \$as)
- [returning](#) ()
- [from](#) ()
- [join](#) (\$table, \$conditions= '', \$type= '')
- [where](#) ()
- [whereOr](#) ()
- [having](#) ()
- [havingOr](#) ()
- [whereFulltext](#) (\$columns, \$query, \$orNull=false, \$isVector=false)
- [limit](#) (\$limit=null, \$offset=0)
- [groupBy](#) ()
- [orderBy](#) (\$column, \$direction= 'asc')
- [execute](#) (\$arguments=array(), \$types=null)

### Static Public Member Functions

- static [select](#) ()
- static [insert](#) (\$table, \$values)
- static [update](#) (\$table, \$values)
- static [delete](#) (\$table)

#### 3.71.1 Detailed Description

EXAMPLES:

Complex example:

```
$query = Query::select('u.userId')-> from('Users u')-> count('id', 'grantTotal')->
> aggregate('MAX', 'maximum')-> where('username = :username', 'password-Hash = :hash')->
whereOr('username = :username', 'passwordHash = :hash')->
join('OtherTable o', array('o.id = u.id', 'o.name = u.name'), 'LEFT')-> limit(0, 1)->
> orderBy('u.username', 'desc')-> groupBy('u.name');
```

```
$rowSet = $query->execute(array(':username' => $username, ':hash' => self::secureHash($password)));
```

Simple example:

```
Query::delete('Users')->where('userId = :id')->execute(array('id' => $userId);
```

See also the [User](#) model.

## 3.71.2 Member Function Documentation

### 3.71.2.1 Query::clear (\$fields = null)

Clears all query elements that have been set.

#### Parameters

**array** \$fields The elements to clear. When null (default), will clear all elements.

### 3.71.2.2 static Query::delete (\$table) [static]

Determine the query should be a DELETE-statement. Should generally be followed by a where.

#### Parameters

**string** \$table The table from which to delete.

### 3.71.2.3 Query::from ()

When constructing a SELECT-statement, specifies which table(s) to select from.

#### Parameters

**string/array** \$tables An array of SQL identifiers representing tables to select from. It's also possible to specify a single string, which will be interpreted as an array containing only that.

### 3.71.2.4 static Query::insert (\$table, \$values) [static]

Determines the query should be an INSERT-statement and specifies its parameters.

#### Parameters

**string** \$table The name of the table in which to insert.

**array** \$vals An associative array with column names as keys and as values parameter marks or SQL values.

### 3.71.2.5 Query::returning ()

Specifies a column to be added to the returning clause.

Returning is a very useful PostgreSQL feature. See also the PostgreSQL documentation at <http://www.postgresql.org/docs/9.1/static/sql-insert.html>

**Parameters**

*string* ... The names of the column of which the value should be added to the result set.

**3.71.2.6 static Query::select () [static]**

Determines the query should be a SELECT-statement, if not yet the case, and specifies which columns should be selected. If already called on the same object before, it will extend the columns to be selected. If a single select with no argument or an empty array has been made, all columns will be selected.

**Parameters**

... SQL Identifiers representing columns to limit the result of the selection to.

**3.71.2.7 static Query::update (\$ table, \$ values) [static]**

Determines the query should be an UPDATE-statement and specifies its parameters. Can (and usually should) be followed by a where.

**Parameters**

*string* \$table The name of the table in which to update rows.

*array* \$vals An associative array with column names as keys and as values parameter marks or SQL values.

**3.71.2.8 Query::where ()**

Specify the a condition of a query. Each argument should be a string containing a predicate that will be directly placed in the where-statement of the query. [Note](#) that its contents will NOT be escaped or validated, so make sure they do not directly depend on user input. Parameter markers should be used instead.

The currently supported operators are '<','>','>=','<=','=','==','is','like','ilike','!=','<>','not is','not like','not ilike','overlaps','in' and 'not in'. Operators are case insensitive.

The arguments are separated by a logical AND's. Use whereOr to separate them with OR's.

**3.71.2.9 Query::whereFulltext (\$ columns, \$ query, \$ orNull = false, \$ isVector = false)**

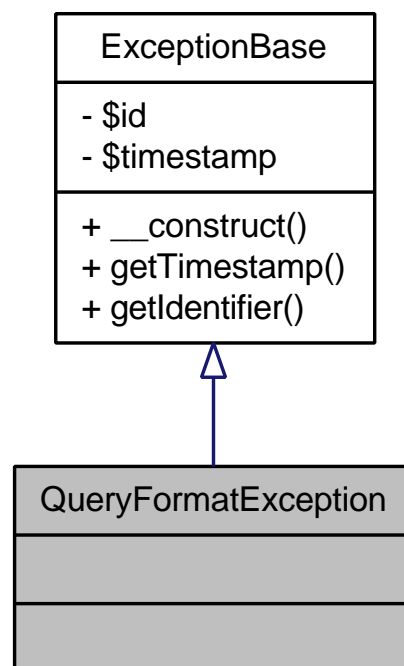
A specific where implementation for fulltext searches.

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

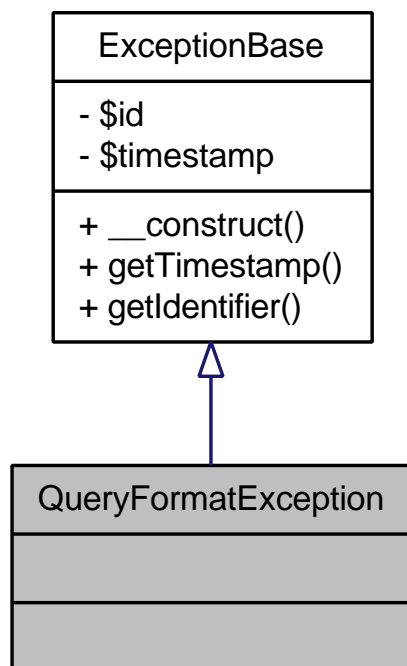
- [www/backend/framework/database/query.php](http://www/backend/framework/database/query.php)

### 3.72 QueryFormatException Class Reference

Inheritance diagram for QueryFormatException:



Collaboration diagram for QueryFormatException:

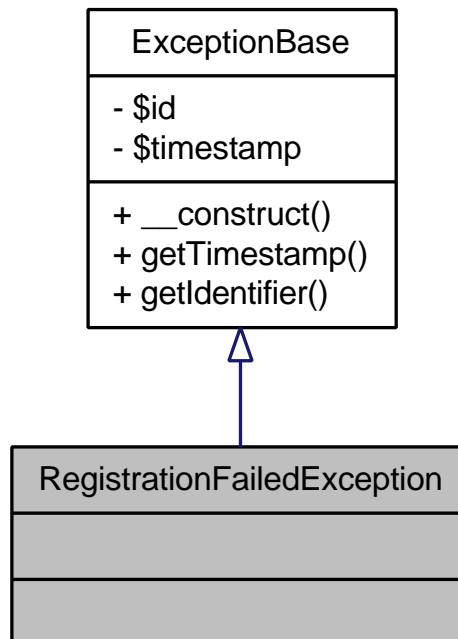


The documentation for this class was generated from the following file:

- [www/backend/framework/database/database.php](http://www/backend/framework/database/database.php)

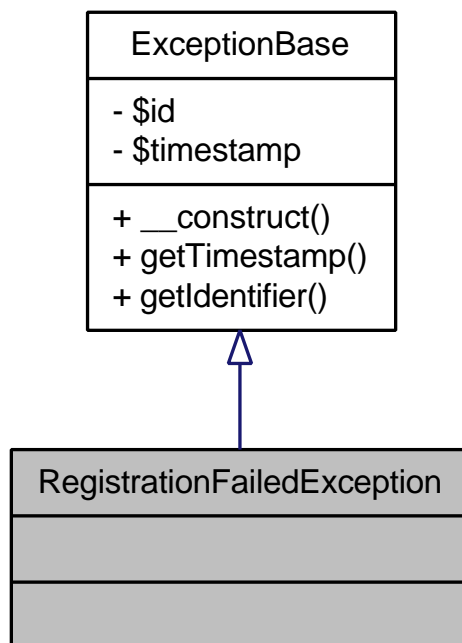
### 3.73 RegistrationFailedException Class Reference

Inheritance diagram for RegistrationFailedException:





Collaboration diagram for RegistrationFailedException:



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

- [www/backend/controllers/usercontroller.php](http://www/backend/controllers/usercontroller.php)

## 3.74 ResultSet Class Reference

### Public Member Functions

- [\\_\\_construct](#) (\$statement)
- [\\_\\_destruct](#) ()
- [getIterator](#) ()
- [getAmount](#) ()
- [tryGetFirstRow](#) ()
- [getFirstRow](#) ()
- [asArrays](#) ()

### 3.74.1 Detailed Description

An IteratorAggregate for the results of a query.

Can be used to iterate over the rows in the result set of a query.

Example:

```
* $results = $dbc->query('SELECT user_name FROM users WHERE birth_date = %a', unixtojd());
* foreach($results as $result)
* {
*     print 'Happy birthday, ' . $result->getValue('user_name') . '!\n'
* }
*
```

### 3.74.2 Constructor & Destructor Documentation

#### 3.74.2.1 ResultSet::\_\_construct (\$ statement)

Constructs a [ResultSet](#) from a prepared PDO statement.

##### Parameters

***PDOStatement*** \$statement The PDO statement. Should already have been executed.

#### 3.74.2.2 ResultSet::\_\_destruct ()

Destructs a [ResultSet](#).

##### Parameters

***PDOStatement*** \$statement The PDO statement. Should already have been executed.

### 3.74.3 Member Function Documentation

#### 3.74.3.1 ResultSet::asArrays ()

Returns an array of the ResultSet's contents

#### 3.74.3.2 ResultSet::getAmount ()

Returns the amount of rows in a [ResultSet](#).

#### 3.74.3.3 ResultSet::getFirstRow ()

Returns first row of a [ResultSet](#) or throws an exception.

#### 3.74.3.4 ResultSet::getIterator ()

Returns an Iterator of a [ResultSet](#).

#### 3.74.3.5 ResultSet::tryGetFirstRow ()

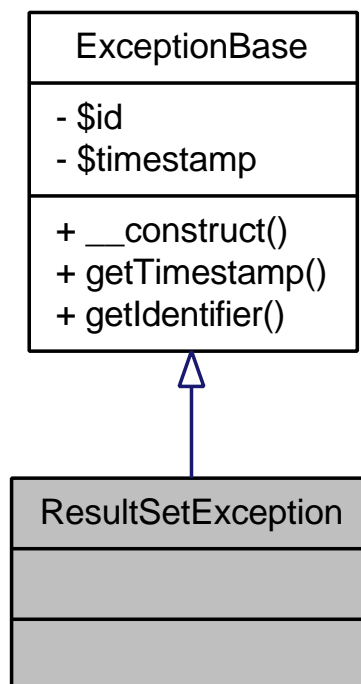
Returns first row of a [ResultSet](#) or returns null.

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

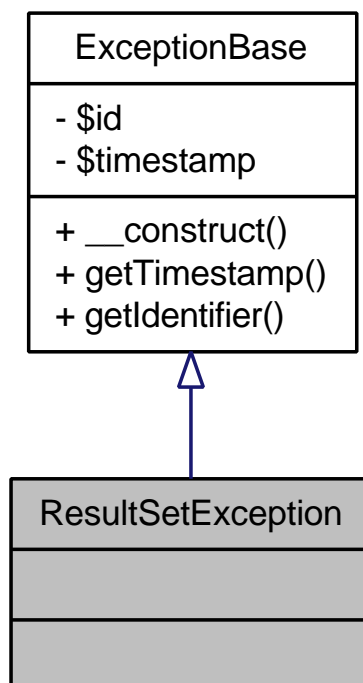
- [www/backend/framework/database/resultset.php](#)

### 3.75 ResultSetException Class Reference

Inheritance diagram for ResultSetException:



Collaboration diagram for ResultSetException:



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

- [www/backend/framework/database/resultset.php](http://www/backend/framework/database/resultset.php)

## 3.76 ResultSetIterator Class Reference

### Public Member Functions

- **\_\_construct** (\$rset, \$stat)
- [next](#) ()
- [key](#) ()
- [current](#) ()
- [valid](#) ()
- [rewind](#) ()

### 3.76.1 Detailed Description

An iterator of ResultSetRows used by [ResultSet](#).

### 3.76.2 Member Function Documentation

#### 3.76.2.1 ResultSetIterator::current ()

Returns the current value.

#### 3.76.2.2 ResultSetIterator::key ()

Returns the current key.

#### 3.76.2.3 ResultSetIterator::next ()

Returns the next value.

#### 3.76.2.4 ResultSetIterator::rewind ()

Sets the current index to zero.

#### 3.76.2.5 ResultSetIterator::valid ()

Checks whether there exists another value.

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

- [www/backend/framework/database/resultset.php](#)

## 3.77 ResultSetRow Class Reference

### Public Member Functions

- [\\_\\_construct](#) (\$row)
- [getValue](#) (\$name, \$type=null)
- [getValues](#) (\$types=null)

### 3.77.1 Detailed Description

Contains the items in a result set from a single row.

### 3.77.2 Constructor & Destructor Documentation

#### 3.77.2.1 ResultSetRow::\_\_construct (\$ row)

Constructs a [ResultSetRow](#).

### 3.77.3 Member Function Documentation

#### 3.77.3.1 ResultSetRow::getValue (\$ name, \$ type = null)

Get the value from the column with the specified name as a string.

#### Parameters

*string* \$name The name of the column, is case incensitive.

*array* \$type Type of the value returned

#### Returns

A string representing the value stored at the specified column, or null if there is no column with this name.

#### 3.77.3.2 ResultSetRow::getValues (\$ types = null)

Returns an associative array with the column names and corresponding values.

#### Parameters

*\$types* Types of the values returned by column name.

#### Returns

The array of row values.

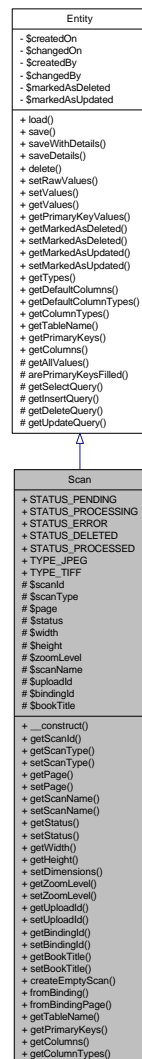
The documentation for this class was generated from the following file:

- [www/backend/framework/database/resultset.php](http://www/backend/framework/database/resultset.php)

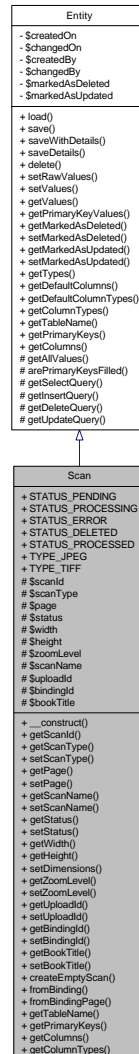


## 3.78 Scan Class Reference

Inheritance diagram for Scan:



Collaboration diagram for Scan:



## Public Member Functions

- [\\_\\_construct](#) (\$id=null)
- [getScanId](#) ()
- [getScanType](#) ()
- [setScanType](#) (\$type)
- [getPage](#) ()
- [setPage](#) (\$page)
- [getScanName](#) ()
- [setScanName](#) (\$scanName)
- [getStatus](#) ()

- **setStatus** (\$status)
- **getWidth** ()
- **getHeight** ()
- **setDimensions** (\$width, \$height)
- **getZoomLevel** ()
- **setZoomLevel** (\$level)
- **getUploadId** ()
- **setUploadId** (\$id)
- **getBindingId** ()
- **setBindingId** (\$id)
- **getBookTitle** ()
- **setBookTitle** (\$bookTitle)

### Static Public Member Functions

- static **createEmptyScan** (\$scanType)
- static **fromBinding** (\$binding)
- static **fromBindingPage** (\$binding, \$range)
- static **getTableName** ()
- static **getPrimaryKeys** ()
- static **getColumns** ()
- static **getColumnTypes** ()

### Public Attributes

- const **STATUS\_PENDING** = 1
- const **STATUS\_PROCESSING** = 2
- const **STATUS\_ERROR** = 3
- const **STATUS\_DELETED** = 6
- const **STATUS\_PROCESSED** = 5
- const **TYPE\_JPEG** = "jpeg"
- const **TYPE\_TIFF** = "tiff"

### Protected Attributes

- **\$scanId**
- **\$scanType**
- **\$page**
- **\$status**
- **\$width**
- **\$height**
- **\$zoomLevel**
- **\$scanName**
- **\$uploadId**
- **\$bindingId**
- **\$bookTitle**

### 3.78.1 Detailed Description

Class representing a scan entity.

### 3.78.2 Constructor & Destructor Documentation

#### 3.78.2.1 Scan::\_\_construct (\$ *id* = null)

Constructs a scan entity.

##### Parameters

*\$id* Id of the scan. Default (null) will create a new scan.

### 3.78.3 Member Function Documentation

#### 3.78.3.1 static Scan::createEmptyScan (\$ *scanType*) [static]

Creates a new empty scan with its status set to enqueued and saves it.

##### Parameters

*\$scanType* The image type of the input image, should be either "tiff" or "jpeg".

##### Returns

The entity of the newly created scan, which is saved and therefor has its primary key set.

#### 3.78.3.2 static Scan::fromBinding (\$ *binding*) [static]

Returns all the books belonging to one binding

##### Parameters

*\$binding* The binding model

##### Returns

Array of scan models

#### 3.78.3.3 static Scan::fromBindingPage (\$ *binding*, \$ *range*) [static]

Returns all the scans of a certain page range which belong to one binding

##### Parameters

*\$binding* The binding model

*\$range* The page range

### Returns

Array of scan models

#### 3.78.3.4 static Scan::getColumns () [static]

Gets all the columns that are not primary keys as an array.

Reimplemented from [Entity](#).

#### 3.78.3.5 static Scan::getColumnTypes () [static]

Gets all the column types, per column, including primary keys.

### Returns

Array of all column types.

Reimplemented from [Entity](#).

#### 3.78.3.6 static Scan::getPrimaryKeys () [static]

Get an array with the primary keys.

Reimplemented from [Entity](#).

#### 3.78.3.7 Scan::getScanId ()

Getters and setters.

#### 3.78.3.8 static Scan::getTableName () [static]

Get the name of the corresponding table.

Reimplemented from [Entity](#).

### 3.78.4 Member Data Documentation

#### 3.78.4.1 Scan::\$bindingId [protected]

The Id of the binding to which this scan belongs.

#### 3.78.4.2 Scan::\$bookTitle [protected]

Joined attributes.

**3.78.4.3 Scan::\$height [protected]**

The height of this scan.

**3.78.4.4 Scan::\$page [protected]**

The page number of this scan.

**3.78.4.5 Scan::\$scanId [protected]**

The Id of this scan.

**3.78.4.6 Scan::\$scanName [protected]**

The filename of this scan.

**3.78.4.7 Scan::\$scanType [protected]**

The type of this scan.

**3.78.4.8 Scan::\$status [protected]**

The status of this scan.

**3.78.4.9 Scan::\$uploadId [protected]**

The Id of the upload corresponding to this scan.

**3.78.4.10 Scan::\$width [protected]**

The width of this scan.

**3.78.4.11 Scan::\$zoomLevel [protected]**

The maximum zoom level of this scan.

**3.78.4.12 const Scan::STATUS\_PENDING = 1**

[Scan](#) status constants.

#### 3.78.4.13 `const Scan::TYPE_JPEG = "jpeg"`

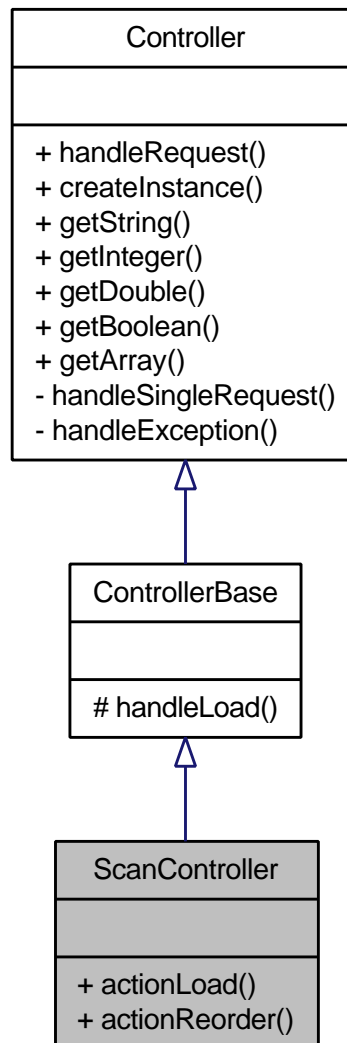
[Scan](#) type constants.

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

- [www/backend/models/scan/scan.php](http://www/backend/models/scan/scan.php)

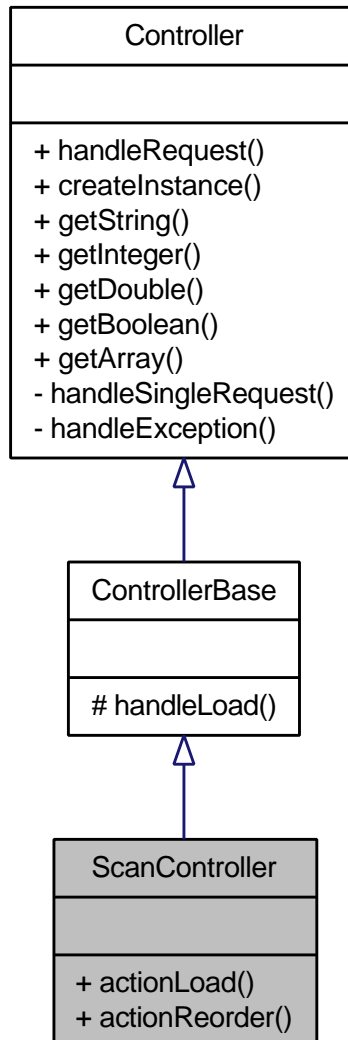
### 3.79 ScanController Class Reference

Inheritance diagram for ScanController:





Collaboration diagram for ScanController:



## Public Member Functions

- [actionLoad](#) (\$data)
- [actionReorder](#) (\$data)

### 3.79.1 Detailed Description

[Scan](#) controller class.

## 3.79.2 Member Function Documentation

### 3.79.2.1 ScanController::actionLoad (\$ *data*)

Loads scans.

### 3.79.2.2 ScanController::actionReorder (\$ *data*)

Reorders the scans of a binding

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

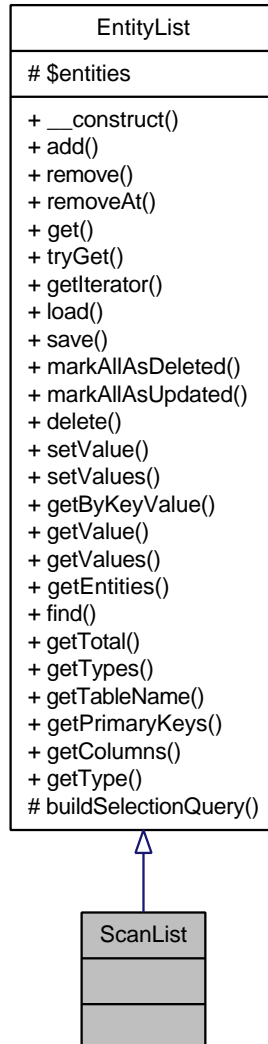
- [www/backend/controllers/scancontroller.php](http://www/backend/controllers/scancontroller.php)

## 3.80 ScanList Class Reference

Inheritance diagram for ScanList:



Collaboration diagram for ScanList:



### 3.80.1 Detailed Description

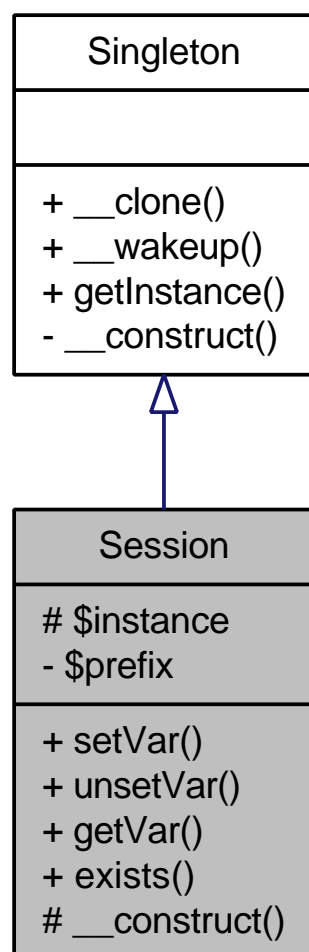
Class representing a scan entity list.

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

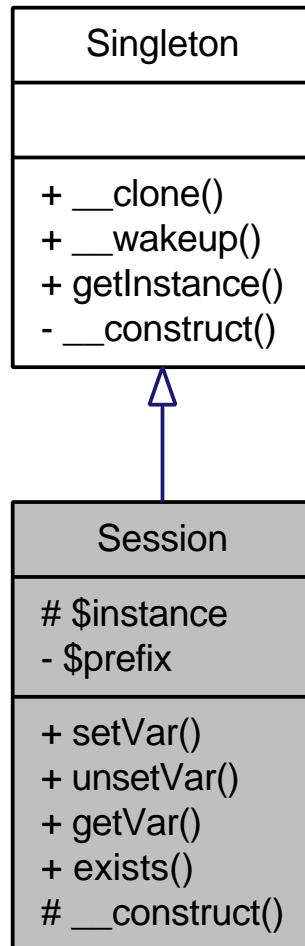
- [www/backend/models/scan/scanlist.php](http://www/backend/models/scan/scanlist.php)

## 3.81 Session Class Reference

Inheritance diagram for Session:



Collaboration diagram for Session:



## Public Member Functions

- `setVar` (\$name, \$value)
- `unsetVar` (\$name)
- `getVar` (\$name)
- `exists` (\$name)

## Protected Member Functions

- `__construct` ()

## Static Protected Attributes

- static [\\$instance](#)

### 3.81.1 Detailed Description

Represents the current user session.

### 3.81.2 Constructor & Destructor Documentation

#### 3.81.2.1 Session::\_\_construct () [protected]

Constructs a session class instance.

Reimplemented from [Singleton](#).

### 3.81.3 Member Function Documentation

#### 3.81.3.1 Session::exists (\$ name)

Checks whether a variable exists in the session.

##### Parameters

*\$name* Name of the variable.

##### Returns

Whether the variable exists.

#### 3.81.3.2 Session::getVar (\$ name)

Sets a session variable.

##### Parameters

*\$name* Name of the variable.

##### Returns

The value of the variable or the empty string if it does not exist.

#### 3.81.3.3 Session::setVar (\$ name, \$ value)

Sets a session variable.

**Parameters**

*\$name* Name of the variable.

*\$value* Value of the variable.

**3.81.3.4 Session::unsetVar (\$ name)**

Unsets a session variable.

**Parameters**

*\$name* Name of the variable.

**3.81.4 Member Data Documentation****3.81.4.1 Session::\$instance [static, protected]**

Unique instance.

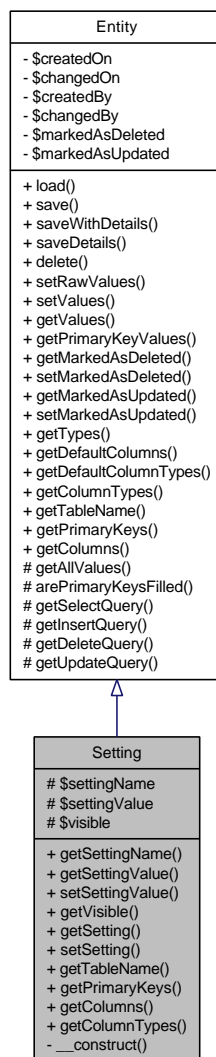
The documentation for this class was generated from the following file:

- [www/backend/framework/util/session.php](http://www/backend/framework/util/session.php)

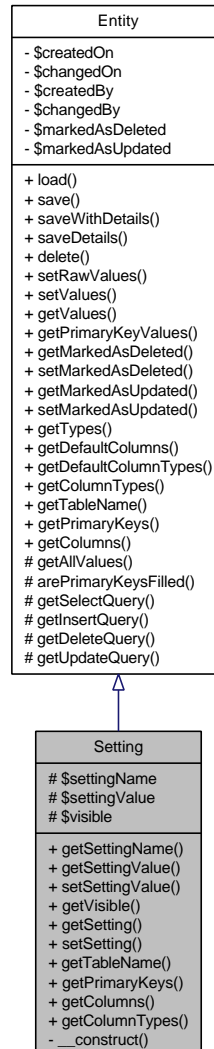


## 3.82 Setting Class Reference

Inheritance diagram for Setting:



Collaboration diagram for Setting:



## Public Member Functions

- [getSettingName](#) ()
- [getSettingValue](#) ()
- [setSettingValue](#) (\$value)
- [getVisible](#) ()

## Static Public Member Functions

- static [getSetting](#) (\$name, \$default=null, \$onlyVisible=false)
- static [setSetting](#) (\$name, \$value)

- static [getTableName \(\)](#)
- static [getPrimaryKeys \(\)](#)
- static [getColumns \(\)](#)
- static [getColumnTypes \(\)](#)

## Protected Attributes

- [\\$settingName](#)
- [\\$settingValue](#)
- [\\$visible](#)

### 3.82.1 Detailed Description

[Setting](#) entity. This represents settings that can be set in the database.

### 3.82.2 Member Function Documentation

#### 3.82.2.1 static [Setting::getColumns \(\)](#) [**static**]

Gets all the columns that are not primary keys as an array.

Reimplemented from [Entity](#).

#### 3.82.2.2 static [Setting::getColumnTypes \(\)](#) [**static**]

Gets all the column types, per column, including primary keys.

#### Returns

Array of all column types.

Reimplemented from [Entity](#).

#### 3.82.2.3 static [Setting::getPrimaryKeys \(\)](#) [**static**]

Get an array with the primary keys.

Reimplemented from [Entity](#).

#### 3.82.2.4 static [Setting::getSetting \(\\$ name, \\$ default = null, \\$ onlyVisible = false\)](#) [**static**]

Returns the value associated with the provided setting name. If such a setting does not exist, the default argument is returned (if one is defined; otherwise an exception is thrown).

### Parameters

*string* \$name The name of the setting.

*string* \$default The result if the setting does not exist.

*bool* \$onlyVisible Only publicly visible settings are returned.

### Returns

string The value of the setting.

### Exceptions

[SettingException](#) If default is null and the setting is not present.

#### 3.82.2.5 Setting::getSettingName ()

Getters and setters.

#### 3.82.2.6 static Setting::getTableName () [static]

Get the name of the corresponding table.

Reimplemented from [Entity](#).

#### 3.82.2.7 static Setting::setSetting (\$ name, \$ value) [static]

Sets the value of the specified setting, creating it if it does not yet exist.

### Parameters

*string* \$name The name of the setting.

*string* \$value The new value of the setting.

### 3.82.3 Member Data Documentation

#### 3.82.3.1 Setting::\$settingName [protected]

Name of the setting.

#### 3.82.3.2 Setting::\$settingValue [protected]

Value of the setting.

**3.82.3.3 Setting::\$visible [protected]**

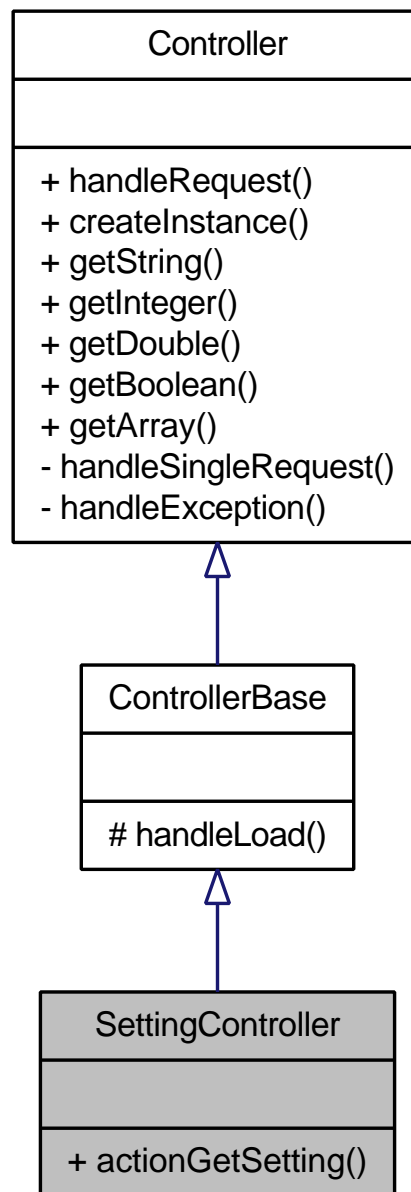
Whether the setting is visible to the client.

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

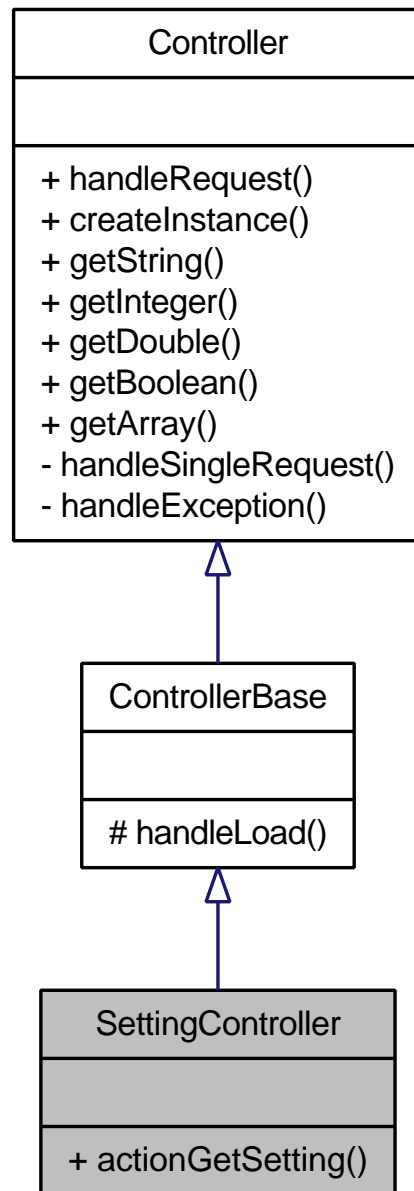
- [www/backend/models/setting/setting.php](http://www/backend/models/setting/setting.php)

### 3.83 SettingController Class Reference

Inheritance diagram for SettingController:



Collaboration diagram for SettingController:



## Public Member Functions

- [actionGetSetting](#) (\$data)

### 3.83.1 Detailed Description

Settings controller. Can be used to view the values of publicly visible settings.

### 3.83.2 Member Function Documentation

#### 3.83.2.1 `SettingController::actionGetSetting ($ data)`

Gets the value of a setting with the specified name. Only visible settings can be viewed by this.

#### Parameters

*\$data* Should contain a string 'setting', containing the name of the setting.

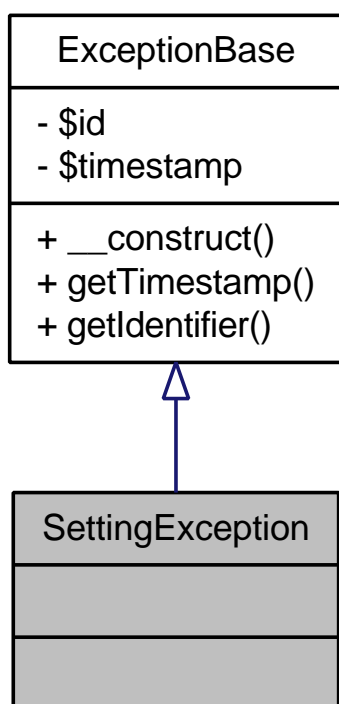
The documentation for this class was generated from the following file:

- `www/backend/controllers/settingcontroller.php`

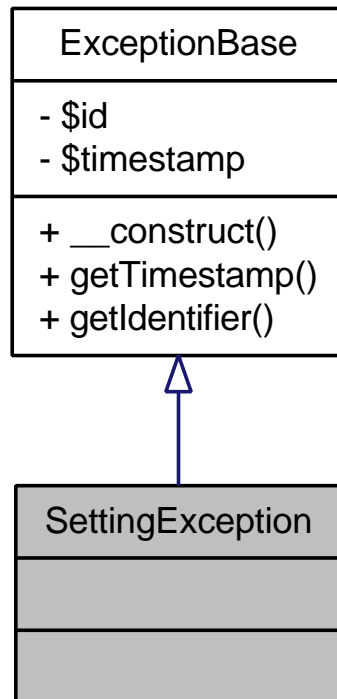


## 3.84 SettingException Class Reference

Inheritance diagram for SettingException:



Collaboration diagram for SettingException:

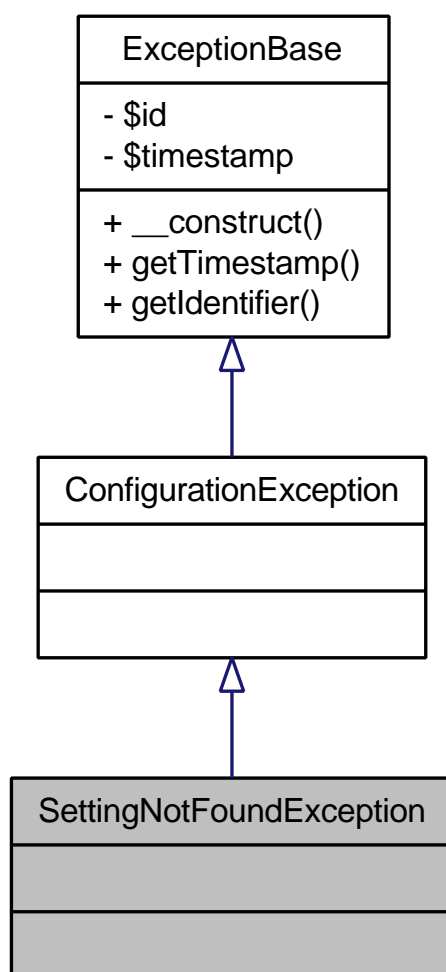


The documentation for this class was generated from the following file:

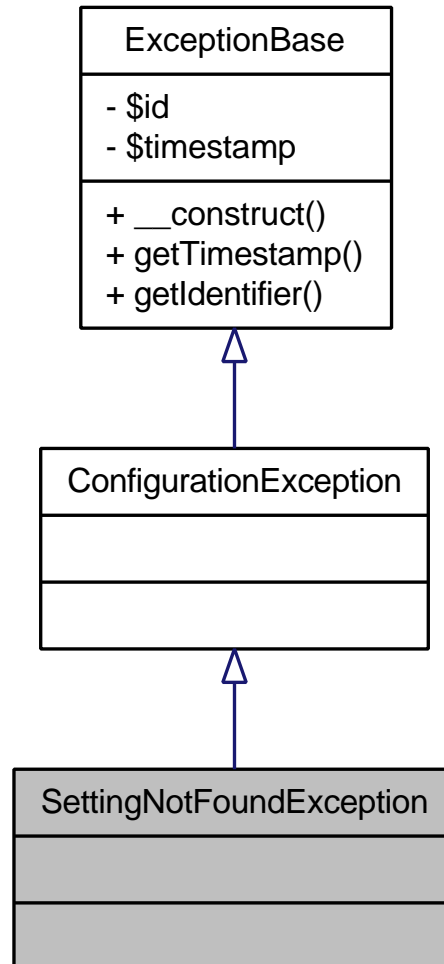
- [www/backend/models/setting/setting.php](http://www/backend/models/setting/setting.php)

## 3.85 SettingNotFoundException Class Reference

Inheritance diagram for SettingNotFoundException:



Collaboration diagram for SettingNotFoundException:

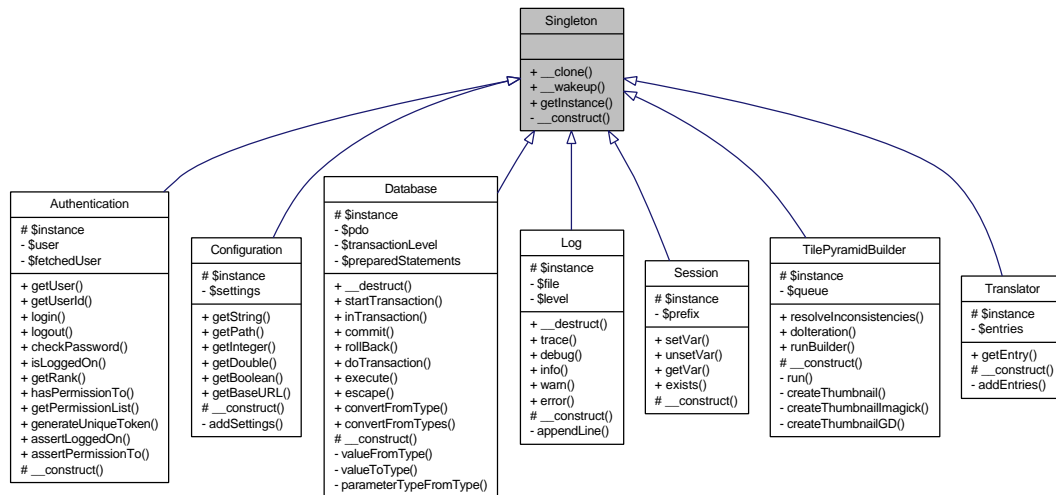


The documentation for this class was generated from the following file:

- [www/backend/framework/util/configuration.php](http://www/backend/framework/util/configuration.php)

## 3.86 Singleton Class Reference

Inheritance diagram for Singleton:



### Public Member Functions

- `__clone ()`
- `__wakeup ()`

### Static Public Member Functions

- static `getInstance ()`

#### 3.86.1 Detailed Description

Base class for all singleton classes.

#### 3.86.2 Member Function Documentation

##### 3.86.2.1 static Singleton::getInstance () [static]

Gets the unique singleton instance.

##### Returns

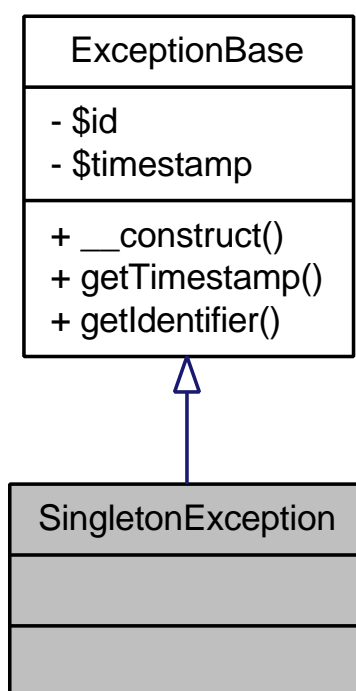
The unique instance of the subclass.

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

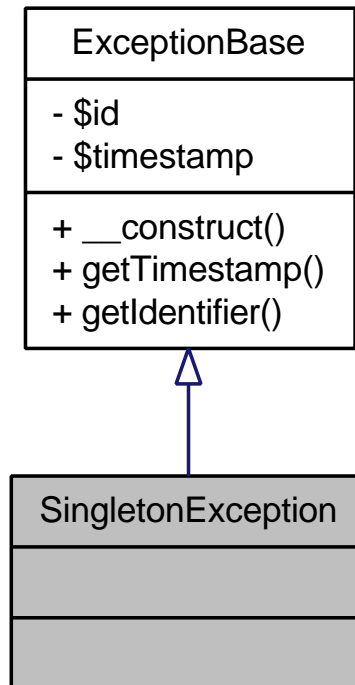
- [www/backend/framework/util/singleton.php](http://www/backend/framework/util/singleton.php)

## 3.87 SingletonException Class Reference

Inheritance diagram for SingletonException:



Collaboration diagram for SingletonException:



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

- [www/backend/framework/util/singleton.php](http://www/backend/framework/util/singleton.php)



## 3.88 StylesheetMinifier Class Reference

### Public Member Functions

- `__construct` (\$input, \$filename)
- `min` ()

### Static Public Member Functions

- static `minify` (\$css, \$location)

### 3.88.1 Detailed Description

Stylesheet minifier class.

### 3.88.2 Constructor & Destructor Documentation

#### 3.88.2.1 StylesheetMinifier::\_\_construct (\$input, \$filename)

Constructor

##### Parameters

- string* \$input CSS to be minified.
- string* \$filename Location of the minified file.

### 3.88.3 Member Function Documentation

#### 3.88.3.1 StylesheetMinifier::min ()

Minifies CSS.

##### Returns

Minified CSS.

#### 3.88.3.2 static StylesheetMinifier::minify (\$css, \$location) [static]

Minifies CSS.

##### Parameters

- string* \$css CSS to be minified.

##### Returns

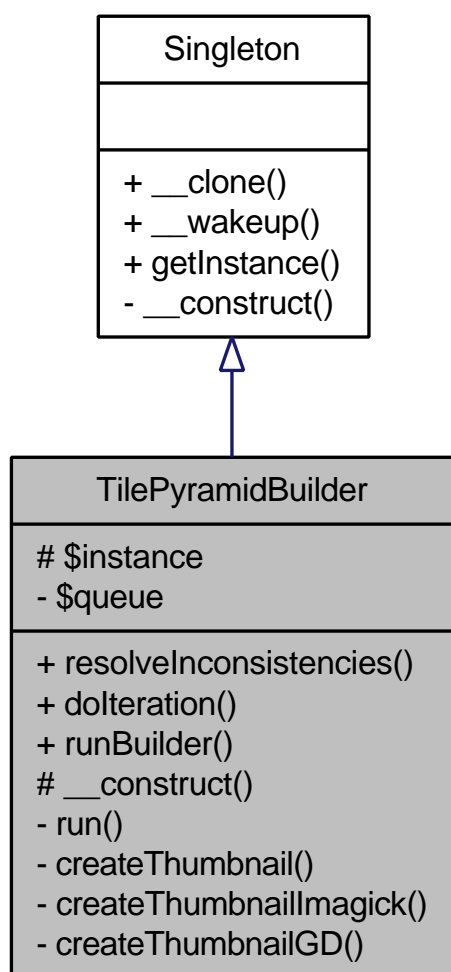
string Minified CSS.

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

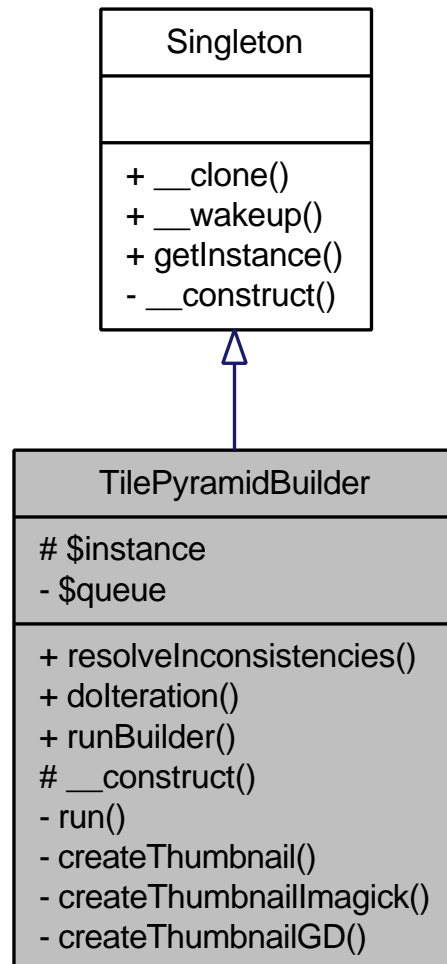
- [www/backend/util/cssmin.php](http://www/backend/util/cssmin.php)

## 3.89 TilePyramidBuilder Class Reference

Inheritance diagram for TilePyramidBuilder:



Collaboration diagram for TilePyramidBuilder:



## Public Member Functions

- [resolveInconsistencies](#) ()
- [doIteration](#) ()
- [runBuilder](#) (\$iterationPause=5)

## Static Protected Attributes

- static [\\$instance](#)

### 3.89.1 Detailed Description

A class that can be used to register newly uploaded scans which will be put in a queue to be processed.

### 3.89.2 Member Function Documentation

#### 3.89.2.1 TilePyramidBuilder::doIteration ()

First checks whether the queue is empty and, if so, fills it with new scans.

Then all scans in the queue are processed one by one.

#### Exceptions

***Exception*** If processing a single scan fails or another error occurs while doing so. The method will try to set that scan's status to error. There might be other scans left in the queue, so calling doIteration again will make it continue with the rest.

#### 3.89.2.2 TilePyramidBuilder::resolveInconsistencies ()

When the builder crashes while processing a scan but does not succeed to set its error flag, this method can be used to change the processing status of every scan to the error status.

NOTE: Only call this when there are no other threads running the tile pyramid builder simultaneously.

#### 3.89.2.3 TilePyramidBuilder::runBuilder (\$ iterationPause = 5)

Continuously keep running the pyramid builder.

NOTE: This method resolves inconsistencies every 20 rounds, so make sure only a single thread is executing this method at a time.

#### Parameters

*int* \$iterationPause The timeout in seconds after each iteration.

### 3.89.3 Member Data Documentation

#### 3.89.3.1 TilePyramidBuilder::\$instance [static, protected]

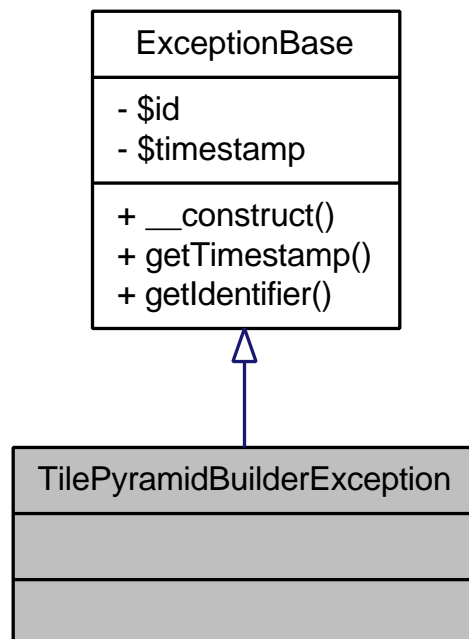
[Singleton](#) instance.

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

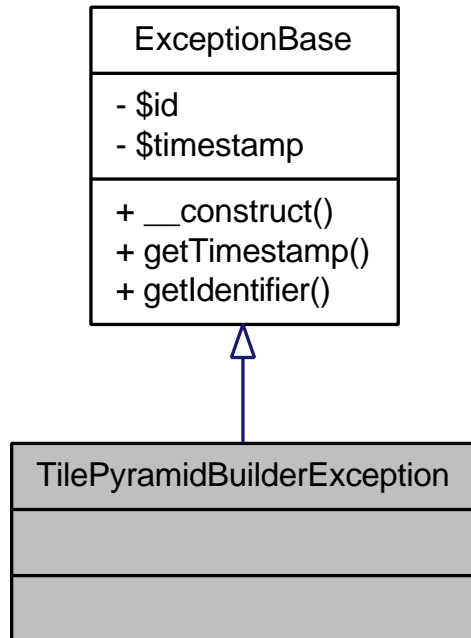
- [www/backend/util/tilepyramidbuilder.php](http://www/backend/util/tilepyramidbuilder.php)

### 3.90 TilePyramidBuilderException Class Reference

Inheritance diagram for TilePyramidBuilderException:



Collaboration diagram for TilePyramidBuilderException:



### 3.90.1 Detailed Description

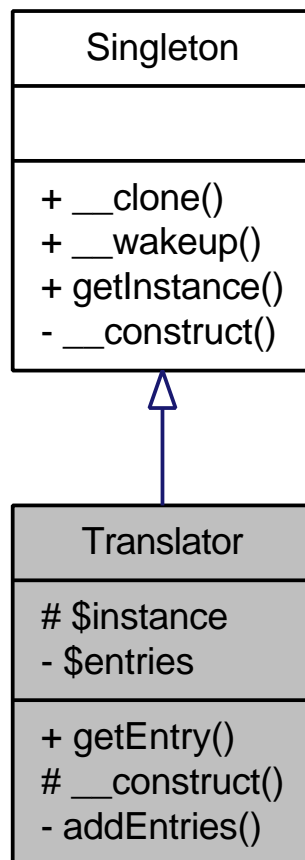
An exception thrown when the tile pyramid builder fails for some reason.

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

- [www/backend/util/tilepyramidbuilder.php](http://www/backend/util/tilepyramidbuilder.php)

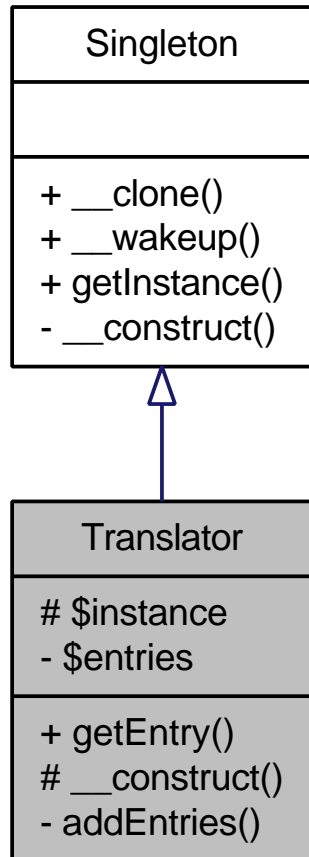
### 3.91 Translator Class Reference

Inheritance diagram for Translator:





Collaboration diagram for Translator:



### Public Member Functions

- [getEntry](#) (\$id, \$args=array())

### Protected Member Functions

- [\\_\\_construct](#) ()

### Static Protected Attributes

- static [\\$instance](#)

### 3.91.1 Detailed Description

[Translator](#) class.

### 3.91.2 Constructor & Destructor Documentation

#### 3.91.2.1 Translator::\_\_construct () [protected]

Constructs a configuration class instance.

Reimplemented from [Singleton](#).

### 3.91.3 Member Function Documentation

#### 3.91.3.1 Translator::getEntry (\$id, \$args = array ())

Gets a language entry.

##### Parameters

*\$id* The id of the entry.

*\$args* The arguments to apply to the entry. Format is that of printf.

##### Returns

The language entry.

### 3.91.4 Member Data Documentation

#### 3.91.4.1 Translator::\$instance [static, protected]

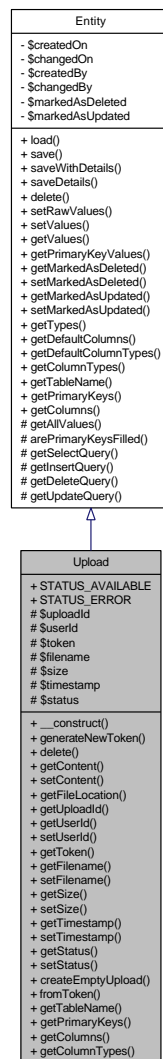
Unique instance.

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

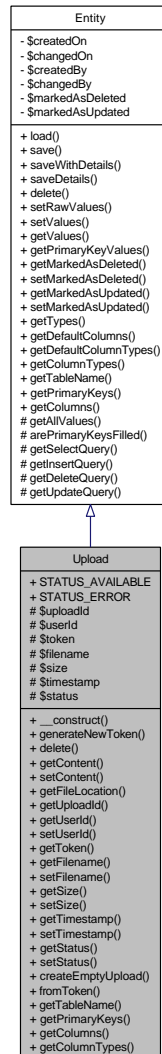
- [www/backend/framework/util/translator.php](#)

## 3.92 Upload Class Reference

Inheritance diagram for Upload:



Collaboration diagram for Upload:



## Public Member Functions

- [\\_\\_construct](#) (\$id=null)
- [generateNewToken](#) ()
- [delete](#) ()
- [getContent](#) ()
- [setContent](#) (\$filename, \$move=false)
- [getFileLocation](#) ()
- [getUploadId](#) ()
- [getUserId](#) ()
- [setUserId](#) (\$id)

- **getToken ()**
- **getFilename ()**
- **setFilename (\$filename)**
- **getSize ()**
- **setSize (\$size)**
- **getTimestamp ()**
- **setTimestamp (\$timestamp)**
- **getStatus ()**
- **setStatus (\$status)**

### Static Public Member Functions

- static [createEmptyUpload](#) (\$userId, \$filename, \$size)
- static [fromToken](#) (\$token)
- static [getTableName](#) ()
- static [getPrimaryKeys](#) ()
- static [getColumns](#) ()
- static [getColumnTypes](#) ()

### Public Attributes

- const [STATUS\\_AVAILABLE](#) = 0
- const [STATUS\\_ERROR](#) = 1

### Protected Attributes

- [\\$uploadId](#)
- [\\$userId](#)
- [\\$token](#)
- [\\$filename](#)
- [\\$size](#)
- [\\$timestamp](#)
- [\\$status](#)

#### 3.92.1 Detailed Description

Class representing an upload entity.

#### 3.92.2 Constructor & Destructor Documentation

##### 3.92.2.1 Upload::\_\_construct (\$id = null)

Constructs an upload entity.

##### Parameters

*\$id* Id of the upload. Default (null) will create a new upload.

### 3.92.3 Member Function Documentation

#### 3.92.3.1 static Upload::createEmptyUpload (\$ *userId*, \$ *filename*, \$ *size*) [static]

Creates a new empty upload with default settings.

##### Parameters

*\$userId*

*\$filename*

*\$size*

##### Returns

The entity of the newly created upload, with its primary key set.

#### 3.92.3.2 Upload::delete ()

Deletes the entity from the database.

Reimplemented from [Entity](#).

#### 3.92.3.3 static Upload::fromToken (\$ *token*) [static]

Gets an upload by a token.

##### Parameters

*\$token* Token to lookup upload by.

##### Returns

[Upload](#) of that token or null.

##### Exceptions

[EntityException](#) If upload could not be found.

#### 3.92.3.4 Upload::generateNewToken ()

Sets a newly generated token.

#### 3.92.3.5 static Upload::getColumns () [static]

Gets all the columns that are not primary keys as an array.

Reimplemented from [Entity](#).

### 3.92.3.6 static Upload::getColumnTypes () [static]

Gets all the column types, per column, including primary keys.

#### Returns

Array of all column types.

Reimplemented from [Entity](#).

### 3.92.3.7 Upload::getContent ()

Get the contents of the upload.

#### Returns

[Upload](#) file contents.

### 3.92.3.8 Upload::getFileLocation ()

Get the file location of this upload.

### 3.92.3.9 static Upload::getPrimaryKeys () [static]

Get an array with the primary keys.

Reimplemented from [Entity](#).

### 3.92.3.10 static Upload::getTableName () [static]

Get the name of the corresponding table.

Reimplemented from [Entity](#).

### 3.92.3.11 Upload::getUploadId ()

Getters and setters.

### 3.92.3.12 Upload::setContent (\$filename, \$move = false)

Set the contents of the upload.

#### Parameters

*\$filename* Filename with new content.

*\$move* Whether to move file. Otherwise a copy is assumed.

### 3.92.4 Member Data Documentation

#### 3.92.4.1 Upload::\$uploadId [protected]

Fields.

#### 3.92.4.2 const Upload::STATUS\_AVAILABLE = 0

[Upload](#) status constants.

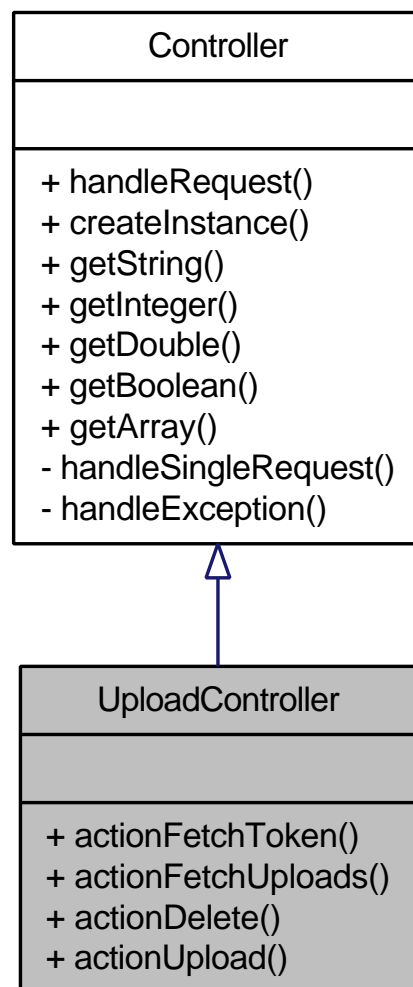
The documentation for this class was generated from the following file:

- [www/backend/models/upload/upload.php](#)

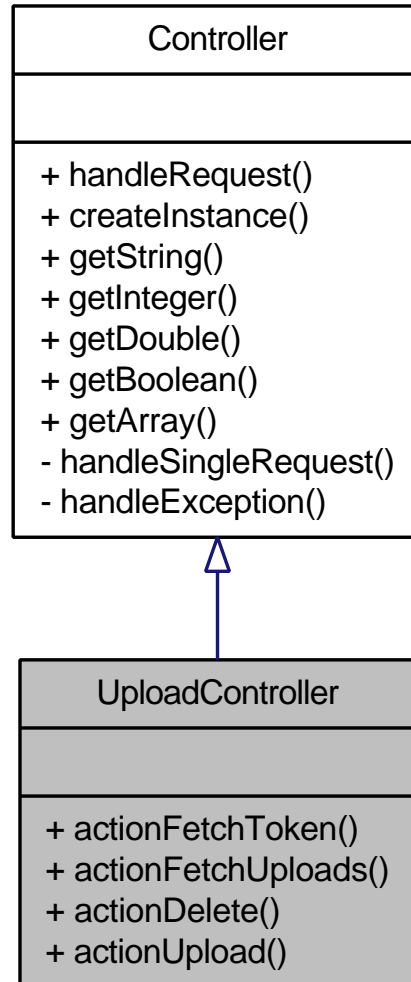


## 3.93 UploadController Class Reference

Inheritance diagram for UploadController:



Collaboration diagram for UploadController:



### Public Member Functions

- [actionFetchToken](#) (\$data)
- [actionFetchUploads](#) (\$data)
- [actionDelete](#) (\$data)
- [actionUpload](#) (\$data)

### 3.93.1 Detailed Description

[Upload](#) controller class.

## 3.93.2 Member Function Documentation

### 3.93.2.1 UploadController::actionDelete (\$ *data*)

Removed an upload

### 3.93.2.2 UploadController::actionFetchToken (\$ *data*)

Fetches an upload token.

### 3.93.2.3 UploadController::actionFetchUploads (\$ *data*)

Fetches an upload token.

### 3.93.2.4 UploadController::actionUpload (\$ *data*)

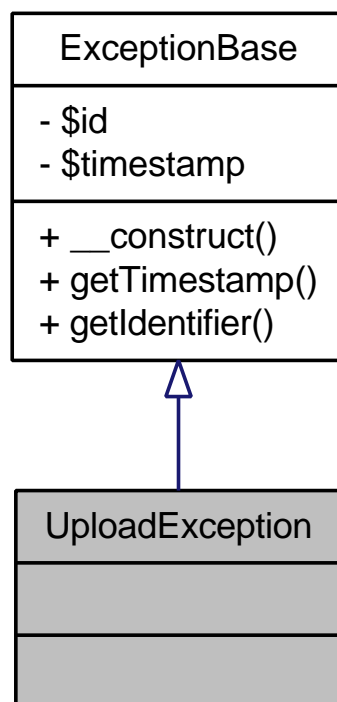
[Upload](#) a file.

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

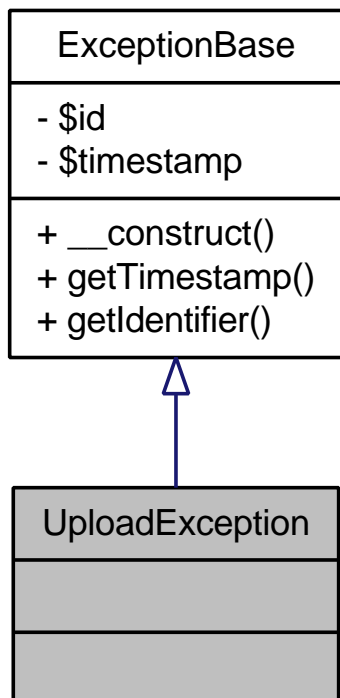
- [www/backend/controllers/uploadcontroller.php](#)

### 3.94 UploadException Class Reference

Inheritance diagram for UploadException:



Collaboration diagram for UploadException:

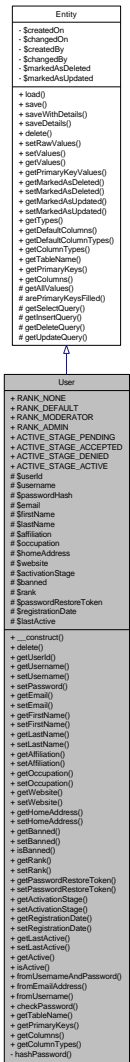


The documentation for this class was generated from the following file:

- [www/backend/models/upload/upload.php](http://www/backend/models/upload/upload.php)

3.95 User Class Reference

Inheritance diagram for User:



Collaboration diagram for User:



## Public Member Functions

- [\\_\\_construct](#) (\$id=null)
- [delete](#) ()
- [getUserId](#) ()
- [getUsername](#) ()
- [setUsername](#) (\$username)
- [setPassword](#) (\$password)
- [getEmail](#) ()
- [setEmail](#) (\$email)
- [getFirstName](#) ()

- **setFirstName** (\$firstName)
- **getLastName** ()
- **setLastName** (\$lastName)
- **getAffiliation** ()
- **setAffiliation** (\$affiliation)
- **getOccupation** ()
- **setOccupation** (\$occupation)
- **getWebsite** ()
- **setWebsite** (\$website)
- **getHomeAddress** ()
- **setHomeAddress** (\$address)
- **getBanned** ()
- **setBanned** (\$banned)
- **isBanned** ()
- **getRank** ()
- **setRank** (\$rank)
- **getPasswordRestoreToken** ()
- **setPasswordRestoreToken** (\$token)
- **getActivationStage** ()
- **setActivationStage** (\$stage)
- **getRegistrationDate** ()
- **setRegistrationDate** (\$registrationDate)
- **getLastActive** ()
- **setLastActive** (\$lastActive)
- **getActive** ()
- **isActive** ()

## Static Public Member Functions

- static [fromUsernameAndPassword](#) (\$username, \$password)
- static [fromEmailAddress](#) (\$email)
- static [fromUsername](#) (\$username)
- static [checkPassword](#) (\$userId, \$password)
- static [getTableName](#) ()
- static [getPrimaryKeys](#) ()
- static [getColumns](#) ()
- static [getColumnTypes](#) ()

## Public Attributes

- const [RANK\\_NONE](#) = 0
- const **RANK\_DEFAULT** = 10
- const **RANK\_MODERATOR** = 40
- const **RANK\_ADMIN** = 50
- const [ACTIVE\\_STAGE\\_PENDING](#) = 0
- const **ACTIVE\_STAGE\_ACCEPTED** = 1
- const **ACTIVE\_STAGE\_DENIED** = 2
- const **ACTIVE\_STAGE\_ACTIVE** = 3



## Protected Attributes

- [\\$userId](#)
- [\\$username](#)
- [\\$passwordHash](#)
- [\\$email](#)
- [\\$firstName](#)
- [\\$lastName](#)
- [\\$affiliation](#)
- [\\$occupation](#)
- [\\$homeAddress](#)
- [\\$website](#)
- [\\$activationStage](#)
- [\\$banned](#)
- [\\$rank](#)
- [\\$passwordRestoreToken](#)
- [\\$registrationDate](#)
- [\\$lastActive](#)

### 3.95.1 Detailed Description

Class representing a user entity.

### 3.95.2 Constructor & Destructor Documentation

#### 3.95.2.1 User::\_\_construct (\$ *id* = null)

Constructs a user by id.

##### Parameters

*\$id* Id of the user. Default (null) will create a new user.

### 3.95.3 Member Function Documentation

#### 3.95.3.1 static User::checkPassword (\$ *userId*, \$ *password*) [static]

Checks the password of the *userId*.

##### Parameters

*\$userId* UserId of the user.

*\$password* Password of the user.

##### Returns

Correctness of password.

### 3.95.3.2 User::delete ()

Deletes this user. References to this user will be referred to by a dummy user.

The user id of the entity should be set before calling this.

Reimplemented from [Entity](#).

### 3.95.3.3 static User::fromEmailAddress (\$ email) [static]

Creates a user entity based on an e-mail address.

#### Parameters

*string* \$email An e-mail address.

#### Returns

[User](#) The user entity with this e-mail address, or null if there is no user with this e-mail.

### 3.95.3.4 static User::fromUsername (\$ username) [static]

Loads the user with the specified username.

#### Parameters

*string* \$username The name of the user to load.

#### Returns

[User](#) A fully loaded user entity of the user with this name.

#### Exceptions

[EntityException](#) If the user does not exist.

### 3.95.3.5 static User::fromUsernameAndPassword (\$ username, \$ password) [static]

Constructs a user by username and password. Also check whether the user is allowed to log in. If that is not the case, an exception is thrown.

#### Parameters

*\$username* Username of the user.

*\$password* Password of the user.

#### Returns

The matching user.

## Exceptions

*[UserBannedException](#)* If the user is banned.

*[UserPendingException](#)* If the user is not yet activated.

*[UserNotFoundException](#)* If no user with the provided username/password combination exists.

### 3.95.3.6 static User::getColumns () [static]

Gets all the columns.

#### Returns

Array of all columns, except primary keys.

Reimplemented from [Entity](#).

### 3.95.3.7 static User::getColumnTypes () [static]

Gets all the column types, per column, including primary keys.

#### Returns

Array of all column types.

Reimplemented from [Entity](#).

### 3.95.3.8 static User::getPrimaryKeys () [static]

Gets the primary keys.

#### Returns

Array of all primary keys.

Reimplemented from [Entity](#).

### 3.95.3.9 static User::getTableName () [static]

Gets the table name.

#### Returns

The table name.

Reimplemented from [Entity](#).

**3.95.3.10 User::getId ()**

Getters and setters.

**3.95.4 Member Data Documentation****3.95.4.1 User::\$activationStage [protected]**

The activation stage of the user.

**3.95.4.2 User::\$affiliation [protected]**

Affiliation.

**3.95.4.3 User::\$banned [protected]**

Whether this user is banned.

**3.95.4.4 User::\$email [protected]**

Email address.

**3.95.4.5 User::\$firstName [protected]**

First name.

**3.95.4.6 User::\$homeAddress [protected]**

Home address.

**3.95.4.7 User::\$lastActive [protected]**

Timestamp this user has last logged in.

**3.95.4.8 User::\$lastName [protected]**

Last name.

**3.95.4.9 User::\$occupation [protected]**

Occupation.

**3.95.4.10 User::\$passwordHash [protected]**

Password hash.

**3.95.4.11 User::\$passwordRestoreToken [protected]**

Password restoration token.

**3.95.4.12 User::\$rank [protected]**

Rank number.

**3.95.4.13 User::\$registrationDate [protected]**

Registration date of this user.

**3.95.4.14 User::\$userId [protected]**

User id.

**3.95.4.15 User::\$username [protected]**

Username.

**3.95.4.16 User::\$website [protected]**

Website.

**3.95.4.17 const User::ACTIVE\_STAGE\_PENDING = 0**

Activation stage constants.

**3.95.4.18 const User::RANK\_NONE = 0**

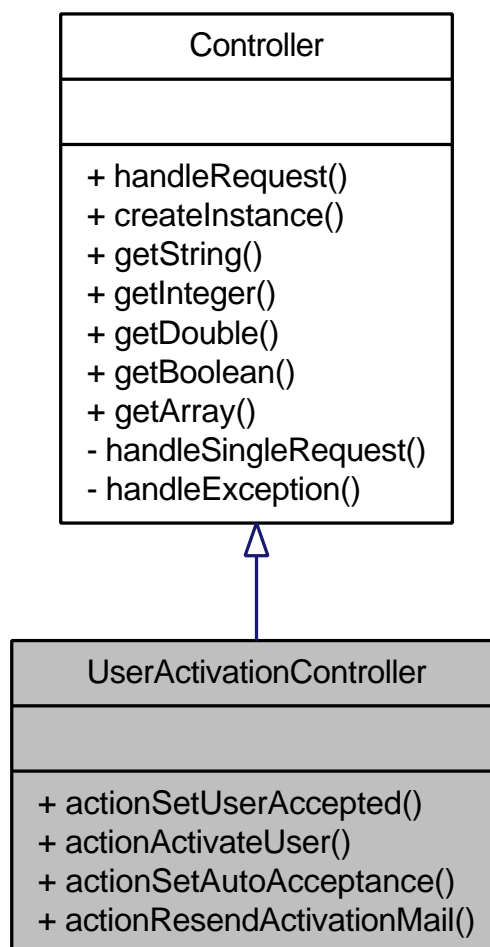
Rank constants: gaps in numbers are intentional to add more ranks if needed.

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

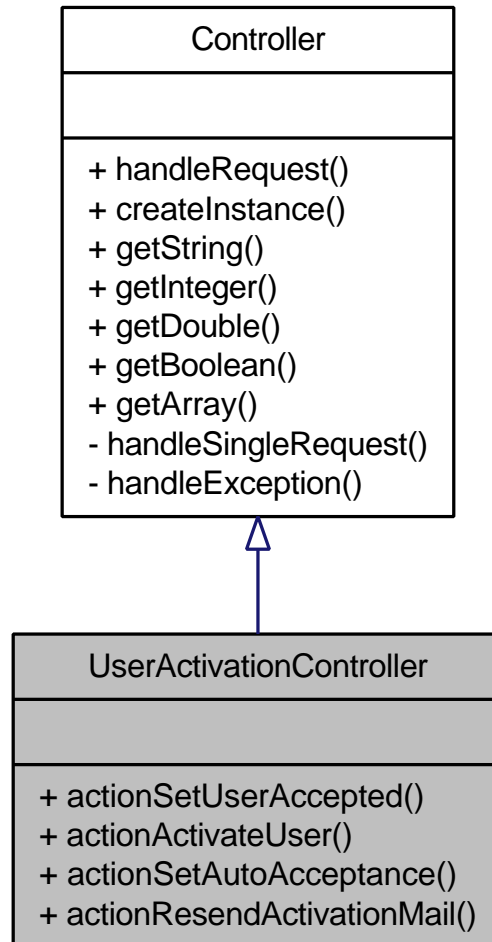
- `www/backend/models/user/user.php`

### 3.96 UserActivationController Class Reference

Inheritance diagram for UserActivationController:



Collaboration diagram for UserActivationController:



### Public Member Functions

- [actionSetUserAccepted](#) (\$data)
- [actionActivateUser](#) (\$data)
- [actionSetAutoAcceptance](#) (\$data)
- [actionResendActivationMail](#) (\$data)

#### 3.96.1 Detailed Description

[User](#) activation controller class.

## 3.96.2 Member Function Documentation

### 3.96.2.1 `UserActivationController::actionActivateUser ($ data)`

Activates a user based on a activation token. If the code matches with an existing pending user, this sets the active bit of the user to true and deletes the pending user entry.

#### Parameters

*\$data* Should contain a 'token', which is the confirmation code of the activation.

### 3.96.2.2 `UserActivationController::actionResendActivationMail ($ data)`

Sends another activation mail in case a user did not receive or lost the initial one.

#### Parameters

*\$data* Should contain the 'username' of the user to whom to send a new e-mail.

### 3.96.2.3 `UserActivationController::actionSetAutoAcceptance ($ data)`

Turns automatic user acceptance on or off. When off, administrators will manually have to accept or decline registrations. If on, registrations are accepted automatically.

In both cases users will still require to click an activation link though.

#### Parameters

*\$data* Should contain a boolean 'autoAccept', which is true if the setting should be turned on and false if it should be turned off. Additionally, if autoAccept is true, there should be a boolean 'acceptAllPending' that indicates whether all users currently waiting for activation should also be activated.

### 3.96.2.4 `UserActivationController::actionSetUserAccepted ($ data)`

Accept or decline a pending user. If either is succesfull, the user will be send an e-mail. If the user was accepted the e-mail will also contain a confirmation code.

#### Parameters

*\$data* Should have 'username' and 'accepted' entries; the latter indicated whether to accept or decline.

#### Exceptions

*[UserActivationException](#)*

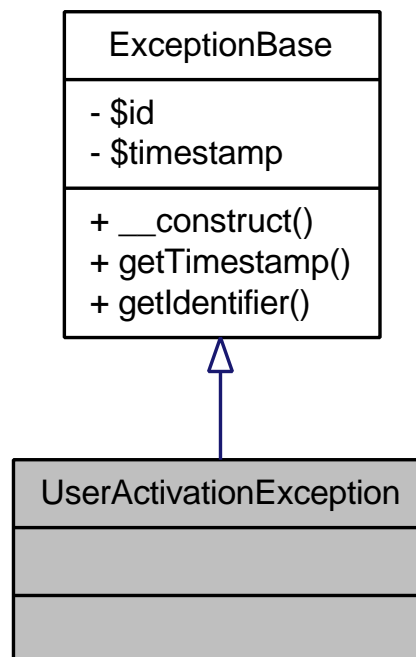


The documentation for this class was generated from the following file:

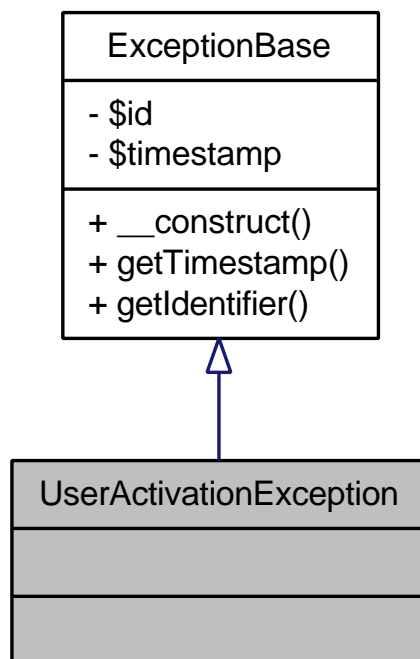
- [www/backend/controllers/useractivationcontroller.php](#)

### 3.97 UserActivationException Class Reference

Inheritance diagram for UserActivationException:



Collaboration diagram for UserActivationException:

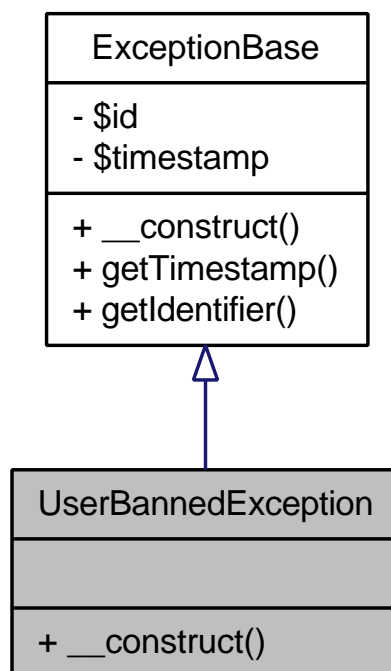


The documentation for this class was generated from the following file:

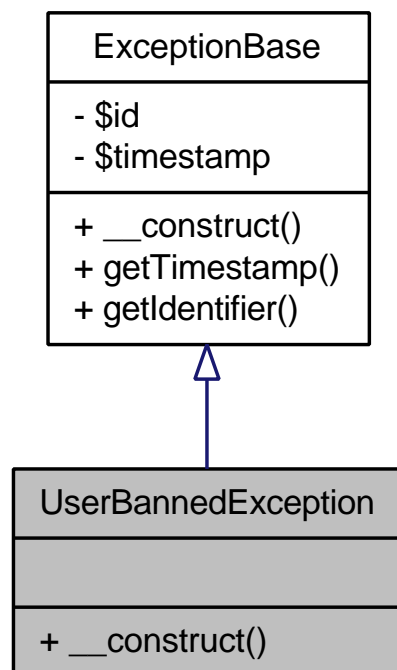
- `www/backend/controllers/useractivationcontroller.php`

### 3.98 UserBannedException Class Reference

Inheritance diagram for UserBannedException:



Collaboration diagram for UserBannedException:



### Public Member Functions

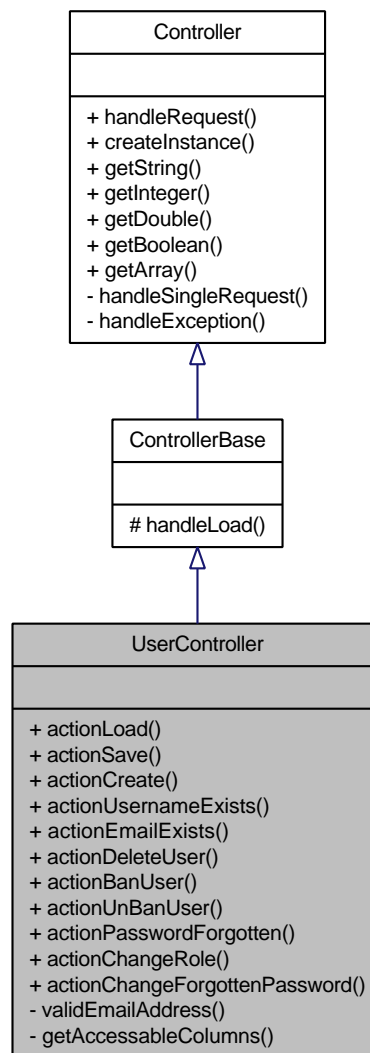
- `__construct` (\$username)

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

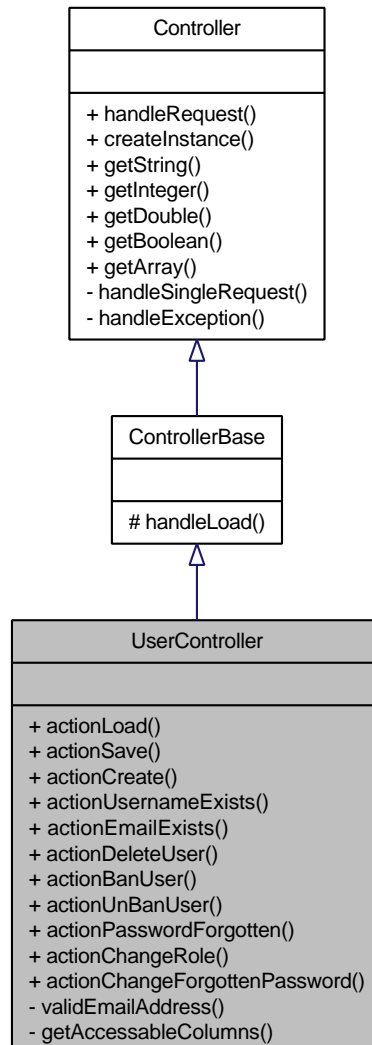
- `www/backend/models/user/user.php`

### 3.99 UserController Class Reference

Inheritance diagram for UserController:



Collaboration diagram for UserController:



## Public Member Functions

- [actionLoad](#) (\$data)
- [actionSave](#) (\$data)
- [actionCreate](#) (\$data)
- [actionUsernameExists](#) (\$data)
- [actionEmailExists](#) (\$data)
- [actionDeleteUser](#) (\$data)
- [actionBanUser](#) (\$data)
- [actionUnBanUser](#) (\$data)
- [actionPasswordForgotten](#) (\$data)

- [actionChangeRole](#) (\$data)
- [actionChangeForgottenPassword](#) (\$data)

### 3.99.1 Detailed Description

[User](#) controller class.

### 3.99.2 Member Function Documentation

#### 3.99.2.1 UserController::actionBanUser (\$ *data*)

Bans a user.

#### 3.99.2.2 UserController::actionChangeForgottenPassword (\$ *data*)

Changes a user's forgotten password based on a token.

#### 3.99.2.3 UserController::actionChangeRole (\$ *data*)

Change the role (rank) of a user.

#### Parameters

*\$data* Should contain 'username' and 'role', the latter being a number indicating the new rank of this user.

#### 3.99.2.4 UserController::actionCreate (\$ *data*)

Creates a user.

#### 3.99.2.5 UserController::actionDeleteUser (\$ *data*)

Deletes a user.

#### 3.99.2.6 UserController::actionEmailExists (\$ *data*)

Checks whether an email address already exists as an email address or username.

#### 3.99.2.7 UserController::actionLoad (\$ *data*)

Loads users.



**3.99.2.8 UserController::actionPasswordForgotten (\$ *data*)**

Sends an e-mail to the user containing a link with which he/she can enter a new password. The only information a user needs to specify for this is an e-mail address.

**3.99.2.9 UserController::actionSave (\$ *data*)**

Saves a user.

**3.99.2.10 UserController::actionUnBanUser (\$ *data*)**

Unbans a user.

**3.99.2.11 UserController::actionUsernameExists (\$ *data*)**

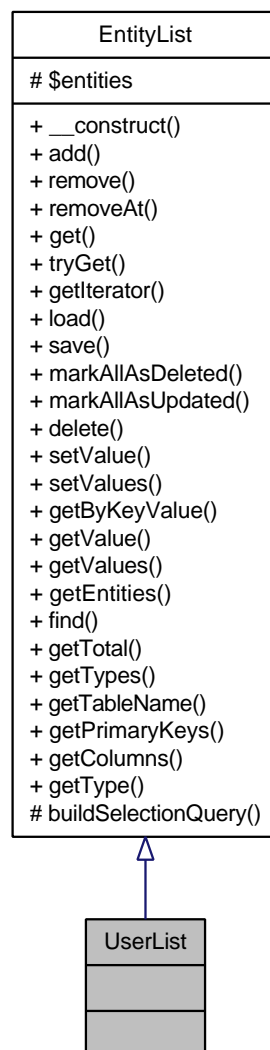
Checks whether a username already exists as an username or email address.

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

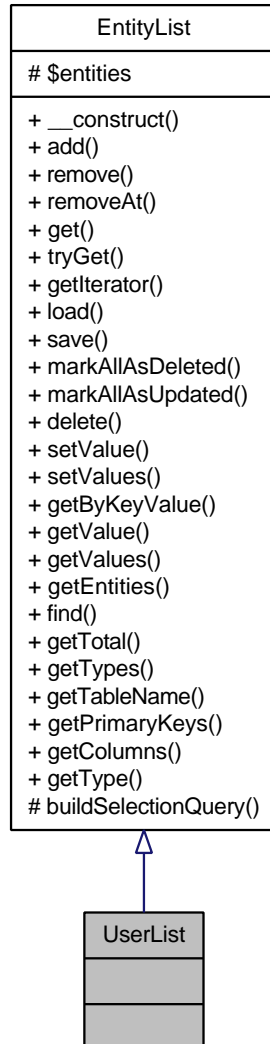
- [www/backend/controllers/usercontroller.php](#)

### 3.100 UserList Class Reference

Inheritance diagram for UserList:



Collaboration diagram for UserList:



### 3.100.1 Detailed Description

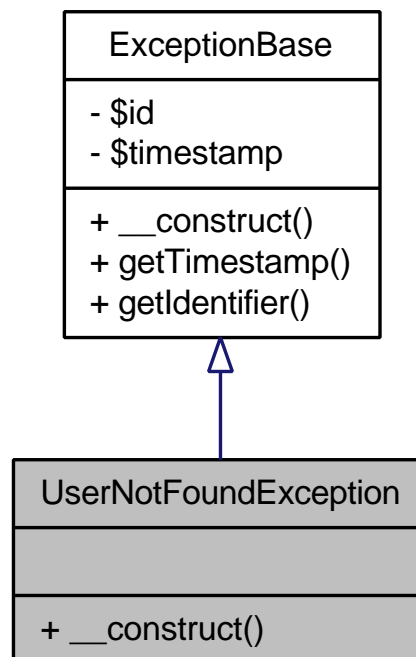
Class representing a user entity list.

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

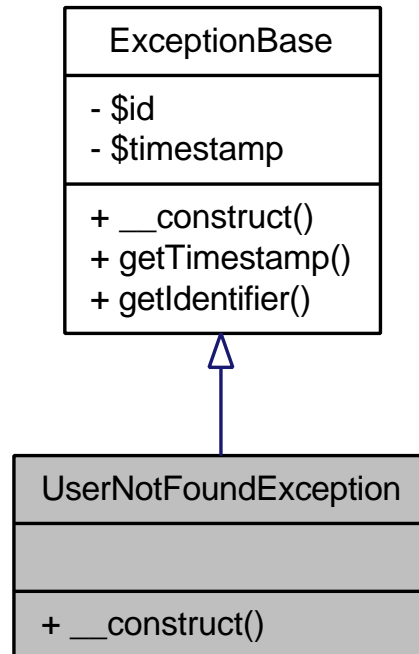
- [www/backend/models/user/userlist.php](http://www/backend/models/user/userlist.php)

### 3.101 UserNotFoundException Class Reference

Inheritance diagram for UserNotFoundException:



Collaboration diagram for UserNotFoundException:



## Public Member Functions

- `__construct ($username)`

### 3.101.1 Detailed Description

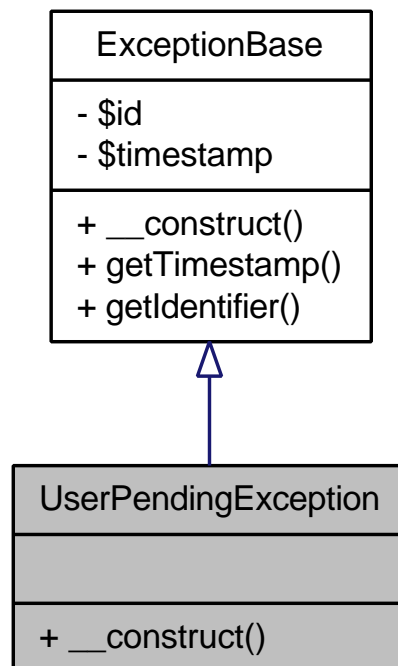
Exceptions.

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

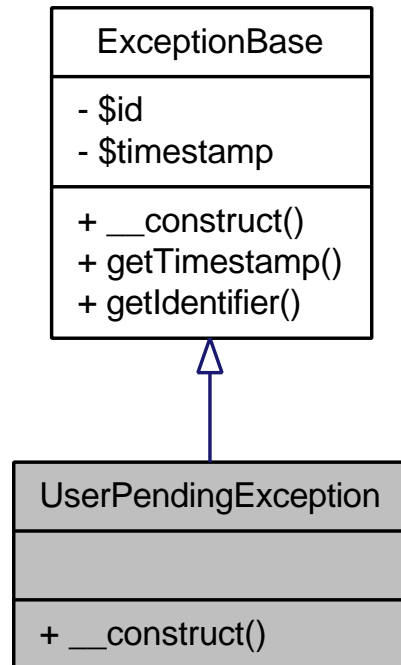
- `www/backend/models/user/user.php`

### 3.102 UserPendingException Class Reference

Inheritance diagram for UserPendingException:



Collaboration diagram for UserPendingException:



### Public Member Functions

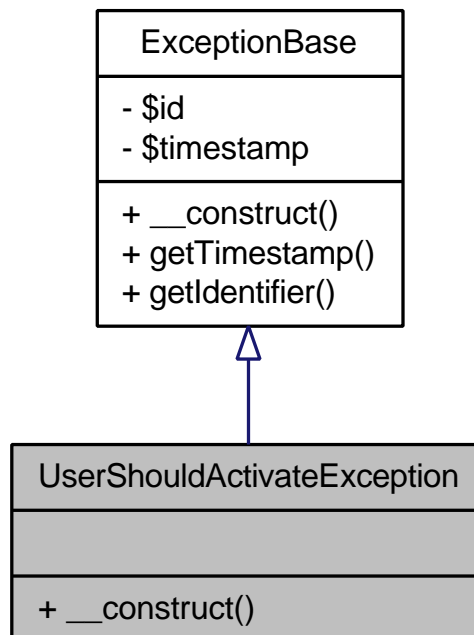
- `__construct ($username)`

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

- `www/backend/models/user/user.php`

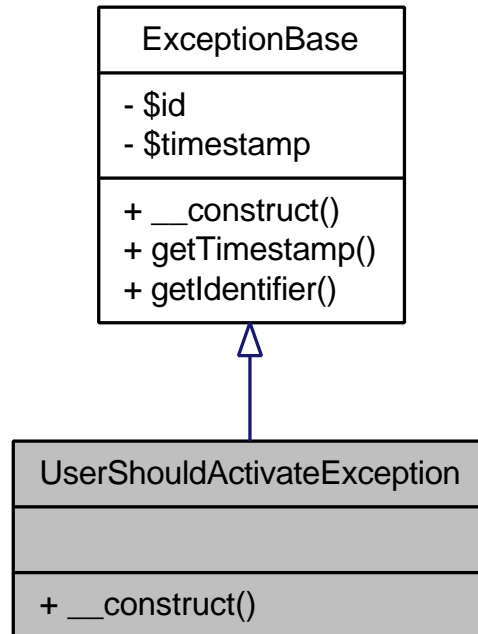
### 3.103 UserShouldActivateException Class Reference

Inheritance diagram for UserShouldActivateException:





Collaboration diagram for UserShouldActivateException:



### Public Member Functions

- `__construct` (\$username)

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

- `www/backend/models/user/user.php`

# Index

\$actionName  
    HelpParagraph, [157](#)  
    Permission, [216](#)  
\$activationStage  
    User, [306](#)  
\$affiliation  
    User, [306](#)  
\$annotationId  
    Annotation, [15](#)  
\$authorList  
    Book, [77](#)  
\$banned  
    User, [306](#)  
\$bindingId  
    Binding, [50](#)  
    BindingLanguage, [58](#)  
    Book, [77](#)  
    Provenance, [226](#)  
    Scan, [251](#)  
\$bindingLanguageList  
    Binding, [50](#)  
\$bookId  
    Author, [39](#)  
    Book, [77](#)  
    BookLanguage, [86](#)  
\$bookLanguageList  
    Book, [77](#)  
\$bookList  
    Binding, [50](#)  
\$bookTitle  
    Scan, [251](#)  
\$changedUserId  
    Annotation, [15](#)  
\$confirmationCode  
    PendingUser, [212](#)  
\$createdUserId  
    Annotation, [15](#)  
\$email  
    User, [306](#)  
\$expirationDate  
    PendingUser, [212](#)  
\$firstName  
    User, [306](#)  
\$firstPage  
    Book, [77](#)  
\$height  
    Scan, [251](#)  
\$helpPageId  
    HelpPage, [149](#)  
    HelpParagraph, [157](#)  
\$helpParagraphId  
    HelpParagraph, [157](#)  
\$helpType  
    HelpPage, [149](#)  
\$homeAddress  
    User, [306](#)  
\$info  
    Library, [177](#)  
\$instance  
    Authentication, [32](#)  
    Configuration, [104](#)  
    Database, [118](#)  
    Log, [184](#)  
    Session, [262](#)  
    TilePyramidBuilder, [283](#)  
    Translator, [288](#)  
\$languageId  
    BindingLanguage, [58](#)  
    BookLanguage, [86](#)  
    Language, [168](#)  
\$languageName  
    Language, [168](#)  
\$lastActive  
    User, [306](#)  
\$lastName  
    User, [306](#)  
\$lastPage  
    Book, [77](#)  
\$libraryAddress  
    Library, [177](#)  
\$libraryId  
    Binding, [50](#)

- Library, [177](#)
- \$libraryName
  - Library, [177](#)
- \$maxYear
  - Book, [77](#)
- \$minRank
  - Permission, [216](#)
- \$minYear
  - Book, [77](#)
- \$name
  - Person, [220](#)
- \$occupation
  - User, [306](#)
- \$order
  - Annotation, [15](#)
- \$page
  - Scan, [252](#)
- \$pageName
  - HelpPage, [149](#)
- \$paragraphParentId
  - HelpParagraph, [157](#)
- \$passwordHash
  - User, [306](#)
- \$passwordRestoreToken
  - User, [307](#)
- \$pendingUserId
  - PendingUser, [212](#)
- \$personId
  - Author, [39](#)
  - Person, [220](#)
  - Provenance, [226](#)
- \$placePublished
  - Book, [77](#)
- \$polygon
  - Annotation, [15](#)
- \$preciseDate
  - Book, [77](#)
- \$printVersion
  - Book, [77](#)
- \$provenanceList
  - Binding, [50](#)
- \$publisher
  - Book, [78](#)
- \$rank
  - User, [307](#)
- \$registrationDate
  - User, [307](#)
- \$scanId
  - Annotation, [15](#)
  - Scan, [252](#)
- \$scanList
  - Binding, [50](#)
- \$scanName
  - Scan, [252](#)
- \$scanType
  - Scan, [252](#)
- \$settingName
  - HelpParagraph, [158](#)
  - Setting, [266](#)
- \$settingValue
  - Setting, [266](#)
- \$signature
  - Binding, [50](#)
- \$status
  - Binding, [51](#)
  - Scan, [252](#)
- \$summary
  - Binding, [51](#)
- \$text
  - Note, [195](#)
- \$timeChanged
  - Annotation, [15](#)
- \$timeCreated
  - Annotation, [15](#)
- \$title
  - Book, [78](#)
  - HelpParagraph, [158](#)
- \$transcriptionEng
  - Annotation, [15](#)
- \$transcriptionOrig
  - Annotation, [15](#)
- \$uploadId
  - Scan, [252](#)
  - Upload, [294](#)
- \$userId
  - Binding, [51](#)
  - Note, [195](#)
  - PendingUser, [212](#)
  - User, [307](#)
- \$username
  - User, [307](#)
- \$visible
  - Setting, [266](#)
- \$website
  - Library, [178](#)
  - User, [307](#)
- \$width
  - Scan, [252](#)
- \$zoomLevel
  - Scan, [252](#)

- \_\_construct
  - Annotation, 13
  - Authentication, 29
  - Author, 38
  - Binding, 49
  - BindingLanguage, 57
  - Book, 75
  - BookLanguage, 85
  - CacheEntry, 99
  - Configuration, 102
  - EntityList, 130
  - HelpPage, 147
  - HelpParagraph, 155
  - JavascriptMinifier, 160
  - Language, 167
  - Library, 176
  - Log, 183
  - Note, 194
  - Pdf, 203
  - PendingUser, 211
  - Person, 219
  - Provenance, 225
  - ResultSet, 240
  - ResultSetRow, 245
  - Scan, 250
  - Session, 261
  - StylesheetMinifier, 279
  - Translator, 288
  - Upload, 291
  - User, 303
- \_\_destruct
  - Log, 183
  - ResultSet, 240
- AccessDeniedException, 9
- action
  - JavascriptMinifier, 160
- actionActivateUser
  - UserActivationController, 310
- actionBanUser
  - UserController, 318
- actionChangeForgottenPassword
  - UserController, 318
- actionChangeRole
  - UserController, 318
- actionCreate
  - UserController, 318
- actionDelete
  - BindingController, 54
  - UploadController, 297
- actionDeleteUpload
  - BindingUploadController, 71
- actionDeleteUser
  - UserController, 318
- actionDownload
  - PdfController, 206
- actionEmailExists
  - UserController, 318
- actionFetchToken
  - UploadController, 297
- actionFetchUploads
  - UploadController, 297
- actionFirstLastPages
  - BookController, 81
- actionGenerate
  - PdfController, 206
- actionGetBindingStatus
  - BindingUploadController, 71
- actionGetGuestPermissionList
  - AuthenticationController, 35
- actionGetPermissionList
  - AuthenticationController, 35
- actionGetSetting
  - SettingController, 270
- actionHasPermissionTo
  - AuthenticationController, 35
- actionIndex
  - MainController, 191
- actionKeepAlive
  - AuthenticationController, 35
- actionLoad
  - AnnotationController, 19
  - AuthorController, 43
  - BindingController, 54
  - BindingLanguageController, 62
  - BookController, 81
  - BookLanguageController, 90
  - HelpController, 144
  - LanguageController, 171
  - NoteController, 197
  - ProvenanceController, 230
  - ScanController, 256
  - UserController, 318
- actionLogin
  - AuthenticationController, 35
- actionLogout
  - AuthenticationController, 35
- actionPasswordForgotten
  - UserController, 318
- actionReorder

- ScanController, 256
- actionResendActivationMail
  - UserActivationController, 310
- actionSave
  - AnnotationController, 19
  - NoteController, 197
  - UserController, 319
- actionScript
  - MainController, 191
- actionSearch
  - BookController, 81
- actionSetAutoAcceptance
  - UserActivationController, 310
- actionSetUserAccepted
  - UserActivationController, 310
- actionStyle
  - MainController, 191
- actionUnBanUser
  - UserController, 319
- actionUniqueLibrarySignature
  - BindingUploadController, 71
- actionUpload
  - BindingUploadController, 71
  - UploadController, 297
- actionUsernameExists
  - UserController, 319
- ACTIVE\_STAGE\_PENDING
  - User, 307
- add
  - EntityList, 130
  - HelpPageList, 152
- addFulltext
  - BookController, 81
- Annotation, 11
  - \$annotationId, 15
  - \$changedUserId, 15
  - \$createdUserId, 15
  - \$order, 15
  - \$polygon, 15
  - \$scanId, 15
  - \$timeChanged, 15
  - \$timeCreated, 15
  - \$transcriptionEng, 15
  - \$transcriptionOrig, 15
  - \_\_construct, 13
  - fromScan, 14
  - getAnnotationId, 14
  - getColumns, 14
  - getColumnTypes, 14
  - getPrimaryKeys, 14
  - getTableName, 14
- AnnotationController, 17
  - actionLoad, 19
  - actionSave, 19
  - polygonEqual, 19
  - textEqual, 19
- AnnotationList, 20
- APCCacheEntry, 22
- arePrimaryKeysFilled
  - Entity, 122
- asArrays
  - ResultSet, 241
- assertLoggedOn
  - Authentication, 29
- assertPermissionTo
  - Authentication, 29
- AssociativeEntity, 24
  - getInsertQuery, 26
  - save, 26
- Authentication, 27
  - \$instance, 32
  - \_\_construct, 29
  - assertLoggedOn, 29
  - assertPermissionTo, 29
  - checkPassword, 29
  - generateUniqueToken, 30
  - getPermissionList, 30
  - getRank, 30
  - getUser, 30
  - getUserId, 30
  - hasPermissionTo, 31
  - isLoggedInOn, 31
  - login, 31
  - logout, 31
- AuthenticationController, 33
  - actionGetGuestPermissionList, 35
  - actionGetPermissionList, 35
  - actionHasPermissionTo, 35
  - actionKeepAlive, 35
  - actionLogin, 35
  - actionLogout, 35
- Author, 36
  - \$bookId, 39
  - \$personId, 39
  - \_\_construct, 38
  - fromBook, 38
  - getBookId, 38
  - getColumns, 38
  - getColumnTypes, 39
  - getPrimaryKeys, 39

- getTableName, 39
- AuthorController, 41
  - actionLoad, 43
- AuthorList, 44
- Binding, 46
  - \$bindingId, 50
  - \$bindingLanguageList, 50
  - \$bookList, 50
  - \$libraryId, 50
  - \$provenanceList, 50
  - \$scanList, 50
  - \$signature, 50
  - \$status, 51
  - \$summary, 51
  - \$userId, 51
  - \_\_construct, 49
  - getBindingId, 49
  - getColumns, 49
  - getColumnTypes, 49
  - getPrimaryKeys, 49
  - getTableName, 49
  - loadDetails, 50
  - saveDetails, 50
  - STATUS\_UPLOADED, 51
- BindingController, 52
  - actionDelete, 54
  - actionLoad, 54
- BindingLanguage, 55
  - \$bindingId, 58
  - \$languageId, 58
  - \_\_construct, 57
  - createOrGet, 57
  - getBindingId, 57
  - getColumns, 57
  - getColumnTypes, 58
  - getPrimaryKeys, 58
  - getTableName, 58
- BindingLanguageController, 60
  - actionLoad, 62
- BindingLanguageList, 63
- BindingList, 65
- BindingStatusException, 67
- BindingUploadController, 69
  - actionDeleteUpload, 71
  - actionGetBindingStatus, 71
  - actionUniqueLibrarySignature, 71
  - actionUpload, 71
  - uniqueLibrarySignature, 71
- Book, 72
  - \$authorList, 77
  - \$bindingId, 77
  - \$bookId, 77
  - \$bookLanguageList, 77
  - \$firstPage, 77
  - \$lastPage, 77
  - \$maxYear, 77
  - \$minYear, 77
  - \$placePublished, 77
  - \$preciseDate, 77
  - \$printVersion, 77
  - \$publisher, 78
  - \$title, 78
  - \_\_construct, 75
  - fromBinding, 75
  - fromBindingPage, 75
  - getBookId, 75
  - getColumns, 75
  - getColumnTypes, 76
  - getPrimaryKeys, 76
  - getTableName, 76
  - loadDetails, 76
  - saveDetails, 76
- BookController, 79
  - actionFirstLastPages, 81
  - actionLoad, 81
  - actionSearch, 81
  - addFulltext, 81
  - splitFulltextQuery, 81
- BookLanguage, 83
  - \$bookId, 86
  - \$languageId, 86
  - \_\_construct, 85
  - createOrGet, 85
  - getBookId, 85
  - getColumns, 85
  - getColumnTypes, 86
  - getPrimaryKeys, 86
  - getTableName, 86
- BookLanguageController, 88
  - actionLoad, 90
- BookLanguageList, 91
- BookList, 93
- BookNotFoundException, 95
- buildSelectionQuery
  - EntityList, 130
- Cache, 97
  - getEntry, 97
  - getFileEntry, 97

- CacheEntry, 98
  - \_\_construct, 99
- checkPassword
  - Authentication, 29
  - User, 303
- clear
  - Query, 234
- commit
  - Database, 116
- Configuration, 100
  - \$instance, 104
  - \_\_construct, 102
  - getBaseURL, 102
  - getBoolean, 102
  - getDouble, 102
  - getInteger, 103
  - getPath, 103
  - getString, 104
- ConfigurationException, 105
- Controller, 107
  - createInstance, 107
  - getArray, 108
  - getBoolean, 108
  - getDouble, 108
  - getInteger, 109
  - getString, 109
  - handleRequest, 109
- ControllerBase, 110
  - handleLoad, 111
- ControllerException, 112
- convertFromType
  - Database, 116
- convertFromTypes
  - Database, 116
- createEmptyScan
  - Scan, 250
- createEmptyUpload
  - Upload, 292
- createInstance
  - Controller, 107
- createOrGet
  - BindingLanguage, 57
  - BookLanguage, 85
- current
  - ResultSetIterator, 244
- Database, 114
  - \$instance, 118
  - commit, 116
  - convertFromType, 116
  - convertFromTypes, 116
  - doTransaction, 116
  - execute, 117
  - inTransaction, 117
  - rollBack, 117
  - startTransaction, 118
- DatabaseException, 119
- debug
  - Log, 183
- delete
  - Entity, 122
  - EntityList, 130
  - Query, 234
  - Upload, 292
  - User, 303
- doIteration
  - TilePyramidBuilder, 283
- doTransaction
  - Database, 116
- Entity, 121
  - arePrimaryKeysFilled, 122
  - delete, 122
  - getAllValues, 122
  - getColumns, 122
  - getColumnTypes, 123
  - getDefaultColumns, 123
  - getDefaultColumnTypes, 123
  - getDeleteQuery, 123
  - getInsertQuery, 123
  - getMarkedAsDeleted, 124
  - getPrimaryKeys, 124
  - getPrimaryKeyValues, 124
  - getSelectQuery, 124
  - getTableName, 124
  - getTypes, 125
  - getUpdateQuery, 125
  - getValues, 125
  - load, 125
  - save, 125
  - saveDetails, 125
  - saveWithDetails, 126
  - setRawValues, 126
  - setValues, 126
- EntityException, 127
- EntityList, 129
  - \_\_construct, 130
  - add, 130
  - buildSelectionQuery, 130
  - delete, 130

- find, [130](#)
- get, [131](#)
- getByKeyValue, [131](#)
- getColumns, [131](#)
- getEntities, [132](#)
- getIterator, [132](#)
- getPrimaryKeys, [132](#)
- getTableName, [132](#)
- getTotal, [132](#)
- getType, [132](#)
- getTypes, [133](#)
- getValue, [133](#)
- getValues, [133](#)
- load, [133](#)
- markAllAsDeleted, [133](#)
- markAllAsUpdated, [133](#)
- remove, [134](#)
- removeAt, [134](#)
- save, [134](#)
- setValue, [134](#)
- setValues, [134](#)
- tryGet, [134](#)
- error
  - Log, [183](#)
- ExceptionBase, [136](#)
  - getIdentifier, [136](#)
  - getTimestamp, [136](#)
- execute
  - Database, [117](#)
- exists
  - Session, [261](#)
- FileCacheEntry, [137](#)
- find
  - EntityList, [130](#)
- FormatException, [140](#)
- from
  - Query, [234](#)
- fromBinding
  - Book, [75](#)
  - Provenance, [225](#)
  - Scan, [250](#)
- fromBindingPage
  - Book, [75](#)
  - Scan, [250](#)
- fromBook
  - Author, [38](#)
- fromEmailAddress
  - User, [304](#)
- fromScan
  - Annotation, [14](#)
- fromToken
  - Upload, [292](#)
- fromUser
  - PendingUser, [211](#)
- fromUsername
  - User, [304](#)
- fromUsernameAndPassword
  - User, [304](#)
- generateNewToken
  - Upload, [292](#)
- generateUniqueToken
  - Authentication, [30](#)
- get
  - EntityList, [131](#)
  - JavascriptMinifier, [160](#)
- getActionName
  - Permission, [215](#)
- getAllValues
  - Entity, [122](#)
- getAmount
  - ResultSet, [241](#)
- getAnnotationId
  - Annotation, [14](#)
- getArray
  - Controller, [108](#)
- getBaseURL
  - Configuration, [102](#)
- getBindingId
  - Binding, [49](#)
  - BindingLanguage, [57](#)
  - Provenance, [225](#)
- getBookId
  - Author, [38](#)
  - Book, [75](#)
  - BookLanguage, [85](#)
- getBoolean
  - Configuration, [102](#)
  - Controller, [108](#)
- getByKeyValue
  - EntityList, [131](#)
- getChildren
  - HelpPage, [147](#)
  - HelpParagraph, [156](#)
- getColumns
  - Annotation, [14](#)
  - Author, [38](#)
  - Binding, [49](#)
  - BindingLanguage, [57](#)



- Book, [75](#)
- BookLanguage, [85](#)
- Entity, [122](#)
- EntityList, [131](#)
- HelpPage, [147](#)
- HelpParagraph, [156](#)
- Language, [167](#)
- Library, [176](#)
- Note, [194](#)
- PendingUser, [211](#)
- Permission, [215](#)
- Person, [219](#)
- Provenance, [225](#)
- Scan, [251](#)
- Setting, [265](#)
- Upload, [292](#)
- User, [305](#)
- getColumnTypes
  - Annotation, [14](#)
  - Author, [39](#)
  - Binding, [49](#)
  - BindingLanguage, [58](#)
  - Book, [76](#)
  - BookLanguage, [86](#)
  - Entity, [123](#)
  - HelpPage, [148](#)
  - HelpParagraph, [156](#)
  - Language, [167](#)
  - Library, [176](#)
  - Note, [194](#)
  - PendingUser, [211](#)
  - Permission, [215](#)
  - Person, [219](#)
  - Provenance, [226](#)
  - Scan, [251](#)
  - Setting, [265](#)
  - Upload, [292](#)
  - User, [305](#)
- getContent
  - HelpParagraph, [156](#)
  - Upload, [293](#)
- getDefaultColumns
  - Entity, [123](#)
- getDefaultColumnTypes
  - Entity, [123](#)
- getDeleteQuery
  - Entity, [123](#)
- getDouble
  - Configuration, [102](#)
  - Controller, [108](#)
- getEntities
  - EntityList, [132](#)
- getEntry
  - Cache, [97](#)
  - Translator, [288](#)
- getFileEntry
  - Cache, [97](#)
- getFileLocation
  - Upload, [293](#)
- getFirstRow
  - ResultSet, [241](#)
- getHelpPageId
  - HelpPage, [148](#)
- getHelpParagraphId
  - HelpParagraph, [156](#)
- getIdentifier
  - ExceptionBase, [136](#)
- getInsertQuery
  - AssociativeEntity, [26](#)
  - Entity, [123](#)
- getInstance
  - Singleton, [275](#)
- getInteger
  - Configuration, [103](#)
  - Controller, [109](#)
- getIterator
  - EntityList, [132](#)
  - ResultSet, [241](#)
- getLanguageId
  - Language, [167](#)
- getLibraryId
  - Library, [177](#)
- getMarkedAsDeleted
  - Entity, [124](#)
- getPath
  - Configuration, [103](#)
- getPendingUserId
  - PendingUser, [212](#)
- getPermissionList
  - Authentication, [30](#)
- getPermissionListForRank
  - Permission, [215](#)
- getPersonId
  - Person, [219](#)
- getPrimaryKeys
  - Annotation, [14](#)
  - Author, [39](#)
  - Binding, [49](#)
  - BindingLanguage, [58](#)
  - Book, [76](#)

- BookLanguage, 86
- Entity, 124
- EntityList, 132
- HelpPage, 148
- HelpParagraph, 156
- Language, 167
- Library, 177
- Note, 194
- PendingUser, 212
- Permission, 215
- Person, 220
- Provenance, 226
- Scan, 251
- Setting, 265
- Upload, 293
- User, 305
- getPrimaryKeyValues
  - Entity, 124
- getRank
  - Authentication, 30
- getScanId
  - Scan, 251
- getSelectQuery
  - Entity, 124
- getSetting
  - Setting, 265
- getSettingName
  - Setting, 266
- getString
  - Configuration, 104
  - Controller, 109
- getTableName
  - Annotation, 14
  - Author, 39
  - Binding, 49
  - BindingLanguage, 58
  - Book, 76
  - BookLanguage, 86
  - Entity, 124
  - EntityList, 132
  - HelpPage, 148
  - HelpParagraph, 157
  - Language, 168
  - Library, 177
  - Note, 195
  - PendingUser, 212
  - Permission, 216
  - Person, 220
  - Provenance, 226
  - Scan, 251
  - Setting, 266
  - Upload, 293
  - User, 305
- getTimestamp
  - ExceptionBase, 136
- getTotal
  - EntityList, 132
- getType
  - EntityList, 132
- getTypes
  - Entity, 125
  - EntityList, 133
- getUpdateQuery
  - Entity, 125
- getUploadId
  - Upload, 293
- getUser
  - Authentication, 30
- getUserId
  - Authentication, 30
  - Note, 195
  - User, 305
- getValue
  - EntityList, 133
  - ResultSetRow, 245
- getValues
  - Entity, 125
  - EntityList, 133
  - HelpPage, 148
  - HelpParagraph, 157
  - ResultSetRow, 245
- getVar
  - Session, 261
- handleLoad
  - ControllerBase, 111
- handleRequest
  - Controller, 109
- hasPermissionTo
  - Authentication, 31
- HelpController, 142
  - actionLoad, 144
  - setLeaves, 144
- HelpPage, 145
  - \$helpPageId, 149
  - \$helpType, 149
  - \$pageName, 149
  - \_\_construct, 147
  - getChildren, 147
  - getColumns, 147

- getColumnTypes, 148
- getHelpPageId, 148
- getPrimaryKeys, 148
- getTableName, 148
- getValues, 148
- HelpPageList, 150
  - add, 152
- HelpParagraph, 153
  - \$actionName, 157
  - \$helpPageId, 157
  - \$helpParagraphId, 157
  - \$paragraphParentId, 157
  - \$settingName, 158
  - \$title, 158
  - \_\_construct, 155
  - getChildren, 156
  - getColumns, 156
  - getColumnTypes, 156
  - getContent, 156
  - getHelpParagraphId, 156
  - getPrimaryKeys, 156
  - getTableName, 157
  - getValues, 157
- info
  - Log, 184
- insert
  - Query, 234
- inTransaction
  - Database, 117
- isAlphaNum
  - JavascriptMinifier, 160
- isLoggedIn
  - Authentication, 31
- JavascriptMinifier, 159
  - \_\_construct, 160
  - action, 160
  - get, 160
  - isAlphaNum, 160
  - min, 160
  - minify, 161
  - next, 161
  - peek, 161
- JavascriptMinifierException, 162
- key
  - ResultSetIterator, 244
- Language, 165
  - \$languageId, 168
  - \$languageName, 168
  - \_\_construct, 167
  - getColumns, 167
  - getColumnTypes, 167
  - getLanguageId, 167
  - getPrimaryKeys, 167
  - getTableName, 168
- LanguageController, 169
  - actionLoad, 171
- LanguageList, 172
- Library, 174
  - \$info, 177
  - \$libraryAddress, 177
  - \$libraryId, 177
  - \$libraryName, 177
  - \$website, 178
  - \_\_construct, 176
  - getColumns, 176
  - getColumnTypes, 176
  - getLibraryId, 177
  - getPrimaryKeys, 177
  - getTableName, 177
- LibraryList, 179
- load
  - Entity, 125
  - EntityList, 133
- loadDetails
  - Binding, 50
  - Book, 76
- Log, 181
  - \$instance, 184
  - \_\_construct, 183
  - \_\_destruct, 183
  - debug, 183
  - error, 183
  - info, 184
  - trace, 184
  - warn, 184
- login
  - Authentication, 31
- logout
  - Authentication, 31
- Mailer, 185
  - sendActivationMail, 185
  - sendMail, 185
  - sendPasswordRestorationMail, 185
  - sendUserDeclinedMail, 186
- MailerException, 187

- MainController, 189
  - actionIndex, 191
  - actionScript, 191
  - actionStyle, 191
- markAllAsDeleted
  - EntityList, 133
- markAllAsUpdated
  - EntityList, 133
- min
  - JavascriptMinifier, 160
  - StylesheetMinifier, 279
- minify
  - JavascriptMinifier, 161
  - StylesheetMinifier, 279
- next
  - JavascriptMinifier, 161
  - ResultSetIterator, 244
- Note, 192
  - \$text, 195
  - \$userId, 195
  - \_\_construct, 194
  - getColumns, 194
  - getColumnTypes, 194
  - getPrimaryKeys, 194
  - getTableName, 195
  - getUserId, 195
- NoteController, 196
  - actionLoad, 197
  - actionSave, 197
- NotLoggedInException, 199
- PasswordIncorrectException, 201
- Pdf, 203
  - \_\_construct, 203
- PdfController, 204
  - actionDownload, 206
  - actionGenerate, 206
- PdfException, 207
- peek
  - JavascriptMinifier, 161
- PendingUser, 209
  - \$confirmationCode, 212
  - \$expirationDate, 212
  - \$pendingUserId, 212
  - \$userId, 212
  - \_\_construct, 211
  - fromUser, 211
  - getColumns, 211
  - getColumnTypes, 211
  - getPendingUserId, 212
  - getPrimaryKeys, 212
  - getTableName, 212
- Permission, 213
  - \$actionName, 216
  - \$minRank, 216
  - getActionName, 215
  - getColumns, 215
  - getColumnTypes, 215
  - getPermissionListForRank, 215
  - getPrimaryKeys, 215
  - getTableName, 216
  - rankHasPermission, 216
- Person, 217
  - \$name, 220
  - \$personId, 220
  - \_\_construct, 219
  - getColumns, 219
  - getColumnTypes, 219
  - getPersonId, 219
  - getPrimaryKeys, 220
  - getTableName, 220
- PersonList, 221
- polygonEqual
  - AnnotationController, 19
- Provenance, 223
  - \$bindingId, 226
  - \$personId, 226
  - \_\_construct, 225
  - fromBinding, 225
  - getBindingId, 225
  - getColumns, 225
  - getColumnTypes, 226
  - getPrimaryKeys, 226
  - getTableName, 226
- ProvenanceController, 228
  - actionLoad, 230
- ProvenanceList, 231
- Query, 233
  - clear, 234
  - delete, 234
  - from, 234
  - insert, 234
  - returning, 234
  - select, 235
  - update, 235
  - where, 235
  - whereFulltext, 235
- QueryFormatException, 236

- RANK\_NONE
  - User, [307](#)
- rankHasPermission
  - Permission, [216](#)
- RegistrationFailedException, [238](#)
- remove
  - EntityList, [134](#)
- removeAt
  - EntityList, [134](#)
- resolveInconsistencies
  - TilePyramidBuilder, [283](#)
- ResultSet, [240](#)
  - \_\_construct, [240](#)
  - \_\_destruct, [240](#)
  - asArrays, [241](#)
  - getAmount, [241](#)
  - getFirstRow, [241](#)
  - getIterator, [241](#)
  - tryGetFirstRow, [241](#)
- ResultSetException, [242](#)
- ResultSetIterator, [244](#)
  - current, [244](#)
  - key, [244](#)
  - next, [244](#)
  - rewind, [244](#)
  - valid, [244](#)
- ResultSetRow, [245](#)
  - \_\_construct, [245](#)
  - getValue, [245](#)
  - getValues, [245](#)
- returning
  - Query, [234](#)
- rewind
  - ResultSetIterator, [244](#)
- rollBack
  - Database, [117](#)
- runBuilder
  - TilePyramidBuilder, [283](#)
- save
  - AssociativeEntity, [26](#)
  - Entity, [125](#)
  - EntityList, [134](#)
- saveDetails
  - Binding, [50](#)
  - Book, [76](#)
  - Entity, [125](#)
- saveWithDetails
  - Entity, [126](#)
- Scan, [247](#)
  - \$bindingId, [251](#)
  - \$bookTitle, [251](#)
  - \$height, [251](#)
  - \$page, [252](#)
  - \$scanId, [252](#)
  - \$scanName, [252](#)
  - \$scanType, [252](#)
  - \$status, [252](#)
  - \$uploadId, [252](#)
  - \$width, [252](#)
  - \$zoomLevel, [252](#)
  - \_\_construct, [250](#)
  - createEmptyScan, [250](#)
  - fromBinding, [250](#)
  - fromBindingPage, [250](#)
  - getColumns, [251](#)
  - getColumnTypes, [251](#)
  - getPrimaryKeys, [251](#)
  - getScanId, [251](#)
  - getTableName, [251](#)
  - STATUS\_PENDING, [252](#)
  - TYPE\_JPEG, [252](#)
- ScanController, [254](#)
  - actionLoad, [256](#)
  - actionReorder, [256](#)
- ScanList, [257](#)
- select
  - Query, [235](#)
- sendActivationMail
  - Mailer, [185](#)
- sendMail
  - Mailer, [185](#)
- sendPasswordRestorationMail
  - Mailer, [185](#)
- sendUserDeclinedMail
  - Mailer, [186](#)
- Session, [259](#)
  - \$instance, [262](#)
  - \_\_construct, [261](#)
  - exists, [261](#)
  - getVar, [261](#)
  - setVar, [261](#)
  - unsetVar, [262](#)
- setContent
  - Upload, [293](#)
- setLeaves
  - HelpController, [144](#)
- setRawValues
  - Entity, [126](#)
- setSetting

- Setting, 266
- Setting, 263
  - \$settingName, 266
  - \$settingValue, 266
  - \$visible, 266
  - getColumns, 265
  - getColumnTypes, 265
  - getPrimaryKeys, 265
  - getSetting, 265
  - getSettingName, 266
  - getTableName, 266
  - setSetting, 266
- SettingController, 268
  - actionGetSetting, 270
- SettingException, 271
- SettingNotFoundException, 273
- setValue
  - EntityList, 134
- setValues
  - Entity, 126
  - EntityList, 134
- setVar
  - Session, 261
- Singleton, 275
  - getInstance, 275
- SingletonException, 277
- splitFulltextQuery
  - BookController, 81
- startTransaction
  - Database, 118
- STATUS\_AVAILABLE
  - Upload, 294
- STATUS\_PENDING
  - Scan, 252
- STATUS\_UPLOADED
  - Binding, 51
- StylesheetMinifier, 279
  - \_\_construct, 279
  - min, 279
  - minify, 279
- textEqual
  - AnnotationController, 19
- TilePyramidBuilder, 281
  - \$instance, 283
  - doIteration, 283
  - resolveInconsistencies, 283
  - runBuilder, 283
- TilePyramidBuilderException, 284
- trace
  - Log, 184
- Translator, 286
  - \$instance, 288
  - \_\_construct, 288
  - getEntry, 288
- tryGet
  - EntityList, 134
- tryGetFirstRow
  - ResultSet, 241
- TYPE\_JPEG
  - Scan, 252
- uniqueLibrarySignature
  - BindingUploadController, 71
- unsetVar
  - Session, 262
- update
  - Query, 235
- Upload, 289
  - \$uploadId, 294
  - \_\_construct, 291
  - createEmptyUpload, 292
  - delete, 292
  - fromToken, 292
  - generateNewToken, 292
  - getColumns, 292
  - getColumnTypes, 292
  - getContent, 293
  - getFileLocation, 293
  - getPrimaryKeys, 293
  - getTableName, 293
  - getUploadId, 293
  - setContent, 293
  - STATUS\_AVAILABLE, 294
- UploadController, 295
  - actionDelete, 297
  - actionFetchToken, 297
  - actionFetchUploads, 297
  - actionUpload, 297
- UploadException, 298
- User, 300
  - \$activationStage, 306
  - \$affiliation, 306
  - \$banned, 306
  - \$email, 306
  - \$firstName, 306
  - \$homeAddress, 306
  - \$lastActive, 306
  - \$lastName, 306
  - \$occupation, 306

- [\\$passwordHash, 306](#)
  - [\\$passwordRestoreToken, 307](#)
  - [\\$rank, 307](#)
  - [\\$registrationDate, 307](#)
  - [\\$userId, 307](#)
  - [\\$username, 307](#)
  - [\\$website, 307](#)
  - [\\_\\_construct, 303](#)
  - [ACTIVE\\_STAGE\\_PENDING, 307](#)
  - [checkPassword, 303](#)
  - [delete, 303](#)
  - [fromEmailAddress, 304](#)
  - [fromUsername, 304](#)
  - [fromUsernameAndPassword, 304](#)
  - [getColumns, 305](#)
  - [getColumnTypes, 305](#)
  - [getPrimaryKeys, 305](#)
  - [getTableName, 305](#)
  - [getUserId, 305](#)
  - [RANK\\_NONE, 307](#)
- [UserActivationController, 308](#)
  - [actionActivateUser, 310](#)
  - [actionResendActivationMail, 310](#)
  - [actionSetAutoAcceptance, 310](#)
  - [actionSetUserAccepted, 310](#)
- [UserActivationException, 312](#)
- [UserBannedException, 314](#)
- [UserController, 316](#)
  - [actionBanUser, 318](#)
  - [actionChangeForgottenPassword, 318](#)
  - [actionChangeRole, 318](#)
  - [actionCreate, 318](#)
  - [actionDeleteUser, 318](#)
  - [actionEmailExists, 318](#)
  - [actionLoad, 318](#)
  - [actionPasswordForgotten, 318](#)
  - [actionSave, 319](#)
  - [actionUnBanUser, 319](#)
  - [actionUsernameExists, 319](#)
- [UserList, 320](#)
- [UserNotFoundException, 322](#)
- [UserPendingException, 324](#)
- [UserShouldActivateException, 326](#)
- [valid](#)
  - [ResultSetIterator, 244](#)
- [warn](#)
  - [Log, 184](#)
- [where](#)
  - [Query, 235](#)
- [whereFulltext](#)
  - [Query, 235](#)