

PHP OTServ Toolkit



Contents

POT	1
PHP 5.0	3
POT class preview	5
Quick start	6
DAO objects	9
Guilds	13
Guild action drivers	15
Account number hack	18
Server online status	19
About OTServ AAC scripts	21
Package POT Procedural Elements	25
E OTS ErrorCode.php	25
E OTS FileLoaderError.php	26
E OTS NoDriver.php	27
E OTS NotLoaded.php	28
E OTS OTBMEError.php	29
E OTS OutOfBuffer.php	30
IOTS DAO.php	31
IOTS DB.php	32
IOTS FileCache.php	33
IOTS GuildAction.php	34
OTS.php	35
OTS Account.php	36
OTS Accounts List.php	37
OTS Base DAO.php	38
OTS Base List.php	39
OTS Container.php	40
OTS DB MySQL.php	41
OTS DB ODBC.php	42
OTS DB PostgreSQL.php	43
OTS DB SQLite.php	44
OTS FileLoader.php	45
OTS FileNode.php	46
OTS Group.php	47
OTS Groups List.php	48
OTS Guild.php	49
OTS GuildRank.php	50
OTS GuildRanks List.php	51
OTS Guilds List.php	52
OTS InfoRespond.php	53
OTS Item.php	54

OTS_MapCoords.php	55
OTS_Monster.php	56
OTS_OTBMFile.php	57
OTS_Player.php	58
OTS_Players_List.php	59
OTS_SQLField.php	60
OTS_SQLFilter.php	61
OTS_SQLite_Results.php	62
Package POT Classes	63
Class E_OTS_ErrorCode	63
Constructor __construct	63
Class E_OTS_FileLoaderError	64
Class Constant ERROR_CAN_NOT_OPEN	64
Class Constant ERROR_EOF	64
Class Constant ERROR_INVALID_FILE_VERSION	65
Class Constant ERROR_INVALID_FORMAT	65
Class Constant ERROR_NOT_OPEN	65
Class Constant ERROR_SEEK_ERROR	66
Class Constant ERROR_TELL_ERROR	66
Class E_OTS_NoDriver	67
Class E_OTS_NotLoaded	67
Class E_OTS_OTBMErrors	68
Class Constant LOADMAPERROR_OUTDATEDHEADER	68
Class Constant LOADMAPERROR_UNKNOWNNODETYPE	68
Class E_OTS_OutOfBuffer	69
Class IOTS_DAO	69
Constructor __construct	69
Class IOTS_DB	70
Constructor __construct	70
Method fieldName	71
Method lastInsertId	71
Method limit	72
Method SQLquery	72
Method SQLquote	73
Method tableName	73
Class IOTS_FileCache	74
Method readCache	74
Method writeCache	74
Class IOTS_GuildAction	75
Constructor __construct	76
Method addRequest	76
Method deleteRequest	77
Method listRequests	77
Method submitRequest	77
Class OTS_Account	78
Method ban	78
Method block	79
Method count	79
Method create	80

example: account.php	80
Method createEx	81
Method delete	81
Method find	82
Method getCustomField	82
Method getEmail	83
Method getGroup	83
Method getId	84
Method getIterator	84
Method getPACCDays	85
Method getPassword	85
Method getPlayers	86
Method getPlayersList	86
Method isBanned	86
Method isBlocked	87
Method isLoaded	87
Method load	88
Method save	88
Method setCustomField	88
Method setEmail	89
Method setGroup	90
Method setPACCDays	90
Method setPassword	91
Method unban	91
Method unblock	92
Class OTS_Accounts_List	92
Method deleteAccount	92
Method init	93
Class OTS_Base_DAO	93
Var \$db	94
Constructor construct	94
Method clone	95
Method set_state	95
Method sleep	95
Method wakeup	96
Class OTS_Base_List	96
Var \$class	97
Var \$table	97
Constructor construct	97
Method count	98
Method current	98
Method init	99
Method key	99
Method next	99
Method orderBy	100
Method resetOrder	100
Method rewind	100
Method setFilter	101
Method setLimit	101

Method setOffset	102
Method valid	102
Method set state	102
Method sleep	103
Method wakeup	103
Class OTS Container	104
Method addItem	104
Method count	105
Method current	105
Method key	105
Method next	106
Method removeItem	106
Method rewind	106
Method valid	107
Class OTS DB MySQL	107
Constructor construct	108
Method fieldName	108
Method limit	109
Method SQLquery	109
Method SQLquote	110
Method tableName	110
Class OTS DB ODBC	111
Constructor construct	111
Method fieldName	112
Method limit	112
Method SQLquery	113
Method SQLquote	113
Method tableName	114
Class OTS DB PostgreSQL	114
Constructor construct	115
Method fieldName	115
Method limit	116
Method SQLquery	116
Method SQLquote	117
Method tableName	117
Class OTS DB SQLite	118
Constructor construct	118
Method fieldName	119
Method limit	119
Method SQLquery	120
Method SQLquote	120
Method tableName	121
Class OTS FileLoader	122
Class Constant ESCAPE_CHAR	122
Class Constant NODE_END	122
Class Constant NODE_START	123
Var \$root	123
Method loadFile	123
Method setCacheDriver	124

Method clone	124
Method set_state	125
Method sleep	125
Class OTS_FileNode	126
Method getBuffer	126
Method getChar	126
Method getChild	127
Method getLong	127
Method getNext	127
Method getShort	128
Method getString	128
Method getType	128
Method isValid	129
Method setBuffer	129
Method setChild	130
Method setNext	130
Method setType	130
Method skip	131
Method clone	131
Method set_state	132
Class OTS_Group	132
Method count	133
Method delete	133
Method getAccess	134
Method getCustomField	134
Method getFlags	135
Method getId	135
Method getIterator	135
Method getMaxDepotItems	136
Method getMaxVIPList	136
Method getName	137
Method getPlayers	137
Method getPlayersList	137
Method isLoading	138
Method load	138
Method save	139
Method setAccess	139
Method setCustomField	139
Method setFlags	140
Method setMaxDepotItems	141
Method setMaxVIPList	141
Method setName	142
Class OTS_Groups_List	142
Method deleteGroup	143
Method init	143
Class OTS_Guild	144
Method acceptInvite	144
Method acceptRequest	144
Method count	145

Method delete	145
Method deleteInvite	146
Method deleteRequest	146
Method find	147
Method getCreationData	147
Method getCustomField	148
Method getGuildRanks	148
Method getGuildRanksList	149
Method getId	149
Method getIterator	150
Method getName	150
Method getOwner	150
Method invite	151
Method isLoading	151
Method listInvites	152
Method listRequests	152
Method load	152
Method request	153
Method save	153
Method setCreationData	154
Method setCustomField	154
Method setInvitesDriver	155
Method setName	155
Method setOwner	156
Method setRequestsDriver	156
Method clone	157
Method sleep	157
Class OTS_GuildRank	158
Method count	158
Method delete	158
Method find	159
Method getCustomField	159
Method getGuild	160
Method getId	160
Method getIterator	161
Method getLevel	161
Method getName	161
Method getPlayers	162
Method getPlayersList	162
Method isLoading	163
Method load	163
Method save	164
Method setCustomField	164
Method setGuild	165
Method setLevel	165
Method setName	166
Class OTS_GuildRanks_List	166
Method deleteGuildRank	166
Method init	167

Class OTS_Guilds_List	167
Method deleteGuild	168
Method init	168
Class OTS_InfoRespond	169
Method getClientVersion	169
Method getEmail	169
Method getIP	170
Method getLocation	170
Method getMapAuthor	170
Method getMapHeight	171
Method getMapName	171
Method getMapWidth	171
Method getMaxPlayers	172
Method getMonstersCount	172
Method getMOTD	172
Method getName	173
Method getOnlinePlayers	173
Method getOwner	173
Method getPlayersPeak	174
Method getPort	174
Method getServer	174
Method getServerVersion	175
Method getTSPQVersion	175
Method getUptime	175
Method getURL	176
Class OTS_Item	176
Constructor construct	176
Method count	177
Method getAttributes	177
Method getCount	177
Method getId	178
Method setAttributes	178
Method setCount	179
Class OTS_MapCoords	179
Constructor construct	179
Method getX	180
Method getY	180
Method getZ	181
Method set_state	181
Class OTS_Monster	182
Method getArmor	182
Method getAttacks	182
Method getDefense	183
Method getDefenses	183
Method getExperience	183
Method getFlag	184
Method getFlags	184
Method getHealth	184
Method getImmunities	185

Method getLoot	185
Method getManaCost	185
Method getName	186
Method getRace	186
Method getSpeed	186
Method getVoices	187
Method hasImmunity	187
Class OTS_OTBMFile	187
Class Constant OTBM_ATTR_ACTION_ID	188
Class Constant OTBM_ATTR_DEPOT_ID	188
Class Constant OTBM_ATTR_DESC	189
Class Constant OTBM_ATTR_DESCRIPTION	189
Class Constant OTBM_ATTR_EXT_FILE	189
Class Constant OTBM_ATTR_EXT_HOUSE_FILE	190
Class Constant OTBM_ATTR_EXT_SPAWN_FILE	190
Class Constant OTBM_ATTR_HOUSEDOORID	190
Class Constant OTBM_ATTR_ITEM	191
Class Constant OTBM_ATTR_RUNE_CHARGES	191
Class Constant OTBM_ATTR_TELE_DEST	191
Class Constant OTBM_ATTR_TEXT	192
Class Constant OTBM_ATTR_TILE_FLAGS	192
Class Constant OTBM_ATTR_UNIQUE_ID	192
Class Constant OTBM_NODE_HOUSEDOTILE	193
Class Constant OTBM_NODE_ITEM	193
Class Constant OTBM_NODE_ITEM_DEF	193
Class Constant OTBM_NODE_MAP_DATA	194
Class Constant OTBM_NODE_MONSTER	194
Class Constant OTBM_NODE_ROOTV1	194
Class Constant OTBM_NODE_SPAWNS	195
Class Constant OTBM_NODE_SPAWN_AREA	195
Class Constant OTBM_NODE_TILE	195
Class Constant OTBM_NODE_TILE_AREA	196
Class Constant OTBM_NODE_TILE_REF	196
Class Constant OTBM_NODE_TILE_SQUARE	196
Class Constant OTBM_NODE_TOWN	197
Class Constant OTBM_NODE_TOWNS	197
Method getDescription	197
Method getHeight	198
Method getTownID	198
Method getTownName	198
Method getTownsList	199
Method getTownTemple	199
Method getWidth	200
Method loadFile	200
Method set_state	200
Method wakeup	201
Class OTS_Player	201
Method ban	202
Method delete	202

Method find	203
Method getAccount	203
Method getCap	204
Method getConditions	204
Method getCustomField	204
Method getDepot	205
Method getDirection	206
Method getExperience	206
Method getGroup	206
Method getGuildNick	207
Method getHealth	207
Method getHealthMax	208
Method getId	208
Method getLastIP	208
Method getLastLogin	209
Method getLevel	209
Method getLookAddons	210
Method getLookBody	210
Method getLookFeet	210
Method getLookHead	211
Method getLookLegs	211
Method getLookType	212
Method getLossExperience	212
Method getLossMana	212
Method getLossSkills	213
Method getMagLevel	213
Method getMana	214
Method getManaMax	214
Method getManaSpent	214
Method getName	215
Method getPosX	215
Method getPosY	215
Method getPosZ	216
Method getPremiumEnd	216
Method getRank	217
Method getRankId	217
Method getRedSkullTime	217
Method getSave	218
Method getSex	218
Method getSkill	219
Method getSkillTries	219
Method getSlot	220
Method getSoul	220
Method getTownId	221
Method getVocation	221
Method getVocationName	222
Method hasRedSkull	222
Method isBanned	222
Method isLoaded	223

Method isSaveSet	223
Method load	224
Method save	224
Method setAccount	224
Method setCap	225
Method setConditions	225
Method setCustomField	226
Method setDepot	227
Method setDirection	227
Method setExperience	228
Method setGroup	228
Method setGuildNick	229
Method setHealth	229
Method setHealthMax	229
Method setLastIP	230
Method setLastLogin	230
Method setLevel	231
Method setLookAddons	231
Method setLookBody	232
Method setLookFeet	232
Method setLookHead	233
Method setLookLegs	233
Method setLookType	234
Method setLossExperience	234
Method setLossMana	234
Method setLossSkills	235
Method setMagLevel	235
Method setMana	236
Method setManaMax	236
Method setManaSpent	237
Method setName	237
Method setPosX	238
Method setPosY	238
Method setPosZ	238
Method setPremiumEnd	239
Method setRank	239
Method setRankId	240
Method setRedSkull	240
Method setRedSkullTime	241
Method setSave	241
Method setSex	242
Method setSkill	242
Method setSkillTries	243
Method setSlot	243
Method setSoul	244
Method setTownId	244
Method setVocation	245
Method unban	245
Method unsetRedSkull	246

Method unsetSave	246
Method sleep	246
Class OTS_Players_List	247
Method deletePlayer	247
Method init	248
Class OTS_SQLField	248
Constructor construct	249
Method getName	249
Method getTable	249
Class OTS_SQLFilter	250
Class Constant CRITERIUM_AND	250
Class Constant CRITERIUM_OR	250
Class Constant OPERATOR_EQUAL	251
Class Constant OPERATOR_GREATER	251
Class Constant OPERATOR_LIKE	251
Class Constant OPERATOR_LOWER	252
Class Constant OPERATOR_NEQUAL	252
Class Constant OPERATOR_NGREATER	253
Class Constant OPERATOR_NLIKE	253
Class Constant OPERATOR_NLOWER	253
Method addFilter	254
Method compareField	254
Method getTables	255
Method sleep	255
Method toString	255
Class POT	256
Class Constant BAN_ACCOUNT	256
Class Constant BAN_IP	257
Class Constant BAN_PLAYER	257
Class Constant DB_MYSQL	258
Class Constant DB_ODBC	258
Class Constant DB_PGSQL	258
Class Constant DB_SQLITE	259
Class Constant DEPOT_SID_FIRST	259
Class Constant DIRECTION_EAST	259
Class Constant DIRECTION_NORTH	260
Class Constant DIRECTION_SOUTH	260
Class Constant DIRECTION_WEST	260
Class Constant ORDER_ASC	261
Class Constant ORDER_DESC	261
Class Constant SEX_FEMALE	262
Class Constant SEX_MALE	262
Class Constant SKILL_AXE	262
Class Constant SKILL_CLUB	263
Class Constant SKILL_DISTANCE	263
Class Constant SKILL_FISHING	264
Class Constant SKILL_FIST	264
Class Constant SKILL_SHIELDING	264
Class Constant SKILL_SWORD	265

Class Constant SLOT_AMMO	265
Class Constant SLOT_ARMOR	266
Class Constant SLOT_BACKPACK	266
Class Constant SLOT_FEET	266
Class Constant SLOT_HEAD	267
Class Constant SLOT_LEFT	267
Class Constant SLOT_LEGS	268
Class Constant SLOT_NECKLACE	268
Class Constant SLOT_RIGHT	268
Class Constant SLOT_RING	269
Class Constant VOCATION_DRUID	269
Class Constant VOCATION_KNIGHT	270
Class Constant VOCATION_NONE	270
Class Constant VOCATION_PALADIN	270
Class Constant VOCATION_SORCERER	271
Method banIP	271
Method connect	272
example: connect.php	272
Method createFilter	273
Method createObject	273
Method getDBHandle	274
Method getInstance	274
Method getMonster	275
Method getMonstersList	275
Method getVocationID	276
Method getVocationName	276
Method getVocationsList	277
Method isIPBanned	277
Method loadClass	278
Method loadMonsters	278
Method loadVocations	279
Method serverStatus	279
example: example	279
Method setPOTPath	280
example: fakeroot.php	280
Method unbanIP	281
compat.php	283
Appendices	284
Appendix A - Class Trees	285
POT	285
Appendix B - README/CHANGELOG/INSTALL	290
LICENSE	291
CHANGELOG	293
RULES	294
README	296
INSTALL	297
NEWS	297
Appendix D - Todo List	299

POT

This is documentenation of POT - official toolkit for OTServ AAC scripts.

PHP OTServ Toolkit

There are several reasons why POT was created:

- Just because it was needed - OTServ should have had that long time ago.
- To unify AAC scripts - there are tons of them, and you never know how to write even a single line of code to them as each of them are created different way.
- To provide reliable way of database accessing - most of people who create AAC scripts don't know what PHP really is, how to use it, they just "want to make own AAC script".
- To provide easy interface - people who write in PHP want to write in PHP, not using SQL, XML and many other languages. POT provides abstract PHP interface for data stored in database.

POT has been created for latest SVN release, it will work best with pure SVN servers. However it provides routines to access custom database structure elements. However it won't work with broken database - it relies on database foreign key constraints, triggers etc.

System requirements

To use POT you need [PHP](#) version at least 5.0 with [PDO extension installed](#) (so it means you will mostly need PHP 5.1, but it is possible to download PDO as external libraries for PHP 5.0.x).

What POT is

POT is a toolkit/library for accessing OTServ database from PHP. It provides PHP classes that represents OTServ database inforation as an objects.

What POT is not

- It is not AAC script - this is a toolkit for making them, but you can't directly run it as website. It has only programming interface.
- It is not application/system framework - you won't create website with only POT. POT has only functionality connected with OTServ database, it doesn't contain for example templates engine. You also won't be able to use it as an ordinary database connection engine - it makes use of [PDO](#) so you can use PDO by itself, POT doesnt provide any additional universal functionality. All it's classes are strictly connected with OTServ database.

How to use

This is toolkit - set of classes/methods for OTServ database. It abstracts database mechanisms for you so you can work on "physical" PHP objects. But you must know how to use them. This documentation describes some basic steps and toolkit API, but you must know PHP in order to make use of them - the best place to get some knowledge is [PHP manual](#).

Don't copy any of included examples, neither codes provided as examples - they probably won't work. Mainly it's because you have to put your database configuration into them and your script paths. But it's not enough. If you have your own `__autoload()` mechanism you won't be able to just include example codes - you would need to redefine `__autoload()` function, which PHP doesn't allow to (but you should know that very well). Example codes are examples - write your own (if you want them to work the best way for you).

Link

If you use POT in your script and want to show that you can put this image on your website:

You can use following code for that:

```
1 <a href="http://otserv-aac.info/" >
2   
3 </a>
```

PHP 5.0

Some things that you should know if you use POT under PHP 5.0.x.

PHP 5.0

PHP5 was a huge step in PHP history. It is completely other language than PHP4 (and older versions). POT is written for PHP5 but currently most PHP5 installations are done with PHP 5.1 and higher versions. PHP 5.0 differs from next versions in few details (or even not details, but huge changes, but those mostly doesn't affect POT). There are some important things you should know if you use POT with PHP 5.0.

PDO

POT requires [PDO extension](#). It is bundled with PHP since 5.1 version. If you use PHP 5.0 you still can install PDO, but you need to do that using [PECL extensions](#). Detailed information about how to do that are in [PHP manual PDO page](#).

Sub package "compat"

If you use PHP 5.0 you should include special [compatibility assurance library](#). POT uses some mechanisms that exists since PHP 5.1 like [Countable interface](#). It doesn't disallow you using POT with PHP 5.0. Compatibility library will create unexisting interfaces, classes, functions, constants etc. However keep in mind that you won't be able to use PHP 5.1 and newer language mechanisms as it is not possible to redefine PHP behaviour. Here is an example:

```
1  <?php
2
3  /**
4   * @ignore
5   * @package examples
6   * @author Wrzasq <wrzasq@gmail.com>
7   * @copyright 2007 (C) by Wrzasq
8   * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9   */
10
11 // do that before any POT operations!
12 include('compat.php');
13
14 // to not repeat all that stuff
15 include('quickstart.php');
16
17 // STEP 1: no error here - even though we loaded class that implements Countable interface which does not
18 // exists in PHP 5.0 SPL library, because 'compat' library defines it.
19 $list= POT::getInstance()-> createObject('Players_List');
20
21 // STEP 2: we can do that in every version - count() is in fact just a public method
22 echo $list-> count();
23
24 // STEP 3: it won't work correctly in PHP 5.0 - PHP won't call internal count() method of object, will print trivial
25 // count() evaluation result on object
26 echo count( $list);
```


25
26 ?>

Nothin new

Compatibility library makes you sure, that POT scripts won't cause FATAL errors if you run them on older versions of PHP. However it doesn't introduce any new mechanisms so you won't find anything new in this package. It is safe to include compat.php file even if you work with PHP version 5.1 or newer, but there is no point in doing that.

__autoload()

POT registers own __autoload() handler with [spl_autoload_register\(\)](#). This function exists since PHP 5.1.2. Compatibility library defines this function as definer of another function - ordinary __autoload(). If you have own __autoload() function, compat's spl_autoload_register() won't redefine __autoload() to avoid E_ERROR. You then need to bind [POT::loadClass\(\) method](#) to your __autoload() function manually.

What about older PHP versions?

No way. POT was written using new PHP5 object engine - you cant use it with PHP4 and older versions of PHP, PHP/FI.

POT class preview

Here main POT class will be described in more guided way.

What it is

[POT](#) class is main class of this toolkit. You will access any other classes using this one. It creates for you instances of other classes when you call it's methods and handles class files loading.

Creating instance of POT class

To get POT object you have to use [POT::getInstance\(\)](#) static method. You should never ever create POT class instances directly! [POT::getInstance\(\)](#) will save static instance and return it globally so you won't need to re-create instances of this class. It is important, as object of this class contains another resources like database connection, or classes directory path so after creating new instance it would not contain them from previous one.

[__autoload\(\)](#) and POT classes

PHP5 provides nice [autoloading mechanism](#). POT makes use of [spl_autoload_register\(\) function](#) to bind own mechanism with it automatically. If you have your own [__autoload](#) function defined, after including POT class you have to register your function with [spl_autoload_register\(\)](#) as well.

DAO classes

Key part of this toolbox are Data Access Objects which provides abstraction layer in PHP for plain database data. You create them via main POT class using [createObject\(\) method](#).

Quick start

Quick start guide.

Putting this all together

To set POT up for using you have to create it's instance and connect to database (it will automatically bind [POT classes loading mechanism](#) to `__autoload()` function. Here is a startup code example:

```
1  <?php
2
3  /**
4   * @ignore
5   * @package examples
6   * @author Wrzasq <wrzasq@gmail.com>
7   * @copyright 2007 (C) by Wrzasq
8   * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9   */
10
11 // binds your __autoload code
12 if( function_exists('__autoload') )
13 {
14     spl_autoload_register('__autoload');
15 }
16
17 // includes POT main file
18 include( './classes/OTS.php' );
19
20 // database configuration - can be simply moved to external file, eg. config.php
21 $config= array(
22     'driver' =>  POT::DB_MYSQL,
23     'host' =>    'localhost',
24     'user' =>    'wrzasq',
25     'database' => 'otserv'
26 );
27
28 // creates POT instance (or get existing one)
29 $ots= POT::getInstance();
30 $ots-> connect(null, $config;
31
32 ?>
```

Account creation

It is very simple to create account with POT. Here is example code that is self-explainable:

```
1  <?php
2
3  /**
4   * @ignore
5   * @package examples
6   * @author Wrzasq <wrzasq@gmail.com>
7   * @copyright 2007 (C) by Wrzasq
8   * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
```

```

9  */
10
11 // to not repeat all that stuff
12 include('quickstart.php');
13
14 // creates new OTS_Account object
15 $account= $ots->    createObject('Account');
16
17 // generates new account number
18 $number= $account->    create();
19
20 /*
21 to generate number from 111111 to 999999 use:
22 $number = $account->create(111111, 999999);
23 */
24
25 // sets account info
26 $account->    setPassword('secret');// $account->setPassword( md5('secret') );
27 $account->    setEmail('foo@example.com');
28 $account->    unblock();// remember to unblock!
29 $account->    setPACCDays(0);
30 $account->    save();
31
32 // give user his number
33 echo 'Your account number is: ',    $number
34
35 ?>

```

It is important to remember that [create\(\) method](#) sets `blocked` field of record to true by default, so for smaller projects where you, for example, wouldn't need e-mail activation unblock it after creation.

Character reading

Here comes also simple example for character search:

```

1  <?php
2
3  /**
4   * @ignore
5   * @package examples
6   * @author Wrzasq <wrzasq@gmail.com>
7   * @copyright 2007 (C) by Wrzasq
8   * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9   */
10
11 // to not repeat all that stuff
12 include('quickstart.php');
13
14 // creates new OTS_Player object
15 $player= $ots->    createObject('Player');
16
17 // loads player
18 $player->    find('Wrzasq');
19
20 // checks if player exists
21 if( $player->    isLoading() )
22 {
23     // prints character info

```

```

24     echo 'Player \'' . $player> getName() . '\' has ' . $player> getLevel() . ' level.', "\n"
25
26     // example of associated objects retrieving
27     echo 'Player \'' . $player> getName() . '\' is member of ' . $player> getGroup()-> getName() . '
group.', "\n"
28 }
29 else
30 {
31     echo 'Player does not exists.', "\n"
32 }
33
34 ?>

```

Objects listings

There are also classes for entire sets of records. For each of row classes there is list class. Throught list object you can read single objects and/or delete them from database. Also you can set limitation (for example for pagination). All list classes implements Countable and Iterator interfaces:

```

1  <?php
2
3  /**
4   * @ignore
5   * @package examples
6   * @author Wrzasq <wrzasq@gmail.com>
7   * @copyright 2007 (C) by Wrzasq
8   * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9   */
10
11 // to not repeat all that stuff
12 include('quickstart.php');
13
14 // creates new OTS_Player object
15 $players= $ots> createObject('Players_List');
16
17 // count of all players - Countable interface implemented
18 echo 'There are ' . count( $players) . ' players in our database.', "\n"
19
20 // sets limitation
21 $players> setLimit(10);
22 $players> setOffset(2);
23
24 // iterates through selected players
25 foreach($playersas $index=> $player)
26 {
27     // each returned item is instance of OTS_Player class
28     echo (2 + $index) . ': ' . $player> getName(), "\n"
29 }
30
31 ?>

```

DAO objects

Main part of POT are Data Access Objects objects

What are DAO objects?

DAO stands for Data Access Objects. Those are objects which you use mostly - players, accounts, groups, objects lists. They use database resource to fetch/store data and provides you programming interface to access that data without using additional languages like SQL, or XML.

Why this way?

PHP is a PHP. When you write a code in PHP each element has a meaning. While using SQL you have to use database queries. In code they are simply a strings which doesn't represent any particular data for programming environment. DAO objects wraps database operations in objective aspect, so "dead" string queries becomes a fully functional objects which you can control more strictly, allows you to assign relations and automate some parts.

Basic operations

Most basic operations are loading, editing and saving data. To see examples of this, see [Quick start guide](#).

Lists objects

For each table there exist single object class and objects list class. List classes implements [Iterator interface](#) so to list their's content you must use [foreach\(\) loop](#). Each element returned for this loop will be instance of single DAO object. You also use lists to delete items.

Custom fields

POT was created for basic SVN database structure. However you can access custom fields with POT. You do that with `getCustomField()` and `setCustomField()` methods of DAO objects (single, not lists).

While accessing custom fields you have to remember about using proper PHP types of passed values. POT doesn't know anything about those fields so it uses value type to check the way it should serve it for a query. Don't worry about safety - it doesn't create any hole for SQL injections. But you must remember, that 1 (integer) is not same as '1' (string), or 1.0 (float). POT will quote strings to fit SQL query and to prevent from SQL injections so make sure you [cast](#) your values to type that represents field type to prevent (mainly) from quoting numeric fields.

You should use those methods only to access custom fields that are not accessible through standard POT API. Those methods executes SQL query each time you call them so it would be a huge effectivity loss to access standard fields with `getCustomField()/setCustomField()`.

Also it is important that in difference to fields accessible with standard setters you can set custom field value

on not loaded/saved object. You must either load object from database, or save standard record before using custom fields as they need record primary key assigned to object for queries. Here is an example:

```
1  <?php
2
3  /**
4   * @ignore
5   * @package examples
6   * @author Wrzasq <wrzasq@gmail.com>
7   * @copyright 2007 (C) by Wrzasq
8   * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9   */
10
11 // to not repeat all that stuff
12 include('quickstart.php');
13
14 // creates new OTS_Player object
15 $player= $ots-> createObject('Player');
16
17 // sets basic fields
18 $player-> setName('Wrzasq');
19 $player-> setSex(POT::SEX_MALE);
20 $player-> setVocation(POT::VOCATION_KNIGHT);
21 /* etc... */
22
23 /*
24  this is bad! we can't call this now as we dont have object ID assinged yet
25
26  $player->setCustomField('my_field', 2);
27
28  must save before that to get automatic ID:
29  */
30 $player-> save();
31
32 // now we can call that:
33 // 2 won't be quoted - it's integer
34 $player-> setCustomField('my_field', 2);
35 // 3 will be quoted - '3' is a string!
36 $player-> setCustomField('another_field', '3');
37
38 ?>
```

Player items

POT provides also objective way of browsing/editing player items (body slots and depot items with all containers). You have [OTS_Item](#) and [OTS_Container](#) classes for that. OTS_Item represents single item, OTS_Container can contain sub-items (either OTS_Item objects, or next level OTS_Container objects).

There is important thing to mention - POT doesn't know anything about item types! Items tree only contains item IDs from database, it doesn't load any information from items.otb, nor items.xml files.

Detailed API you will find in documentation of those classes. Here are examples of how you use slot and depot items fetching and saving:

```
1  <?php
2
3  /**
4   * @ignore
```

```

5  * @package examples
6  * @author Wrzasq <wrzasq@gmail.com>
7  * @copyright 2007 (C) by Wrzasq
8  * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9  */
10
11 // to not repeat all that stuff
12 include('quickstart.php');
13
14 // creates new OTS_Player object
15 $player= $ots-> createObject('Player');
16 $player-> find('Wrzasq');
17
18 /*
19  Items loading example.
20 */
21
22 // loading item from ammunition slot
23 $item= $player-> getSlot(POT::SLOT_AMMO);
24
25 echo $player-> getName(), ' has item with id ', $item-> getId(), ' in his/her ammo slot.', "\n" ;
26
27 // checks if item is a container
28 if($item instanceof OTS_Container)
29 {
30     // list backpack content
31     foreach($item as $inside)
32     {
33         echo 'Container contains item with id ', $inside-> getId(), ' ', "\n" ;
34     }
35 }
36
37 /*
38  Items tree composing example.
39 */
40
41 // creates container - here it would be a depot locker (we pass ID of item to create)
42 $container= new OTS_Container(2590);
43
44 // now let's create depot chest
45 $chest= new OTS_Container(2594);
46
47 // let's put chest inside locker
48 $container-> addItem($chest);
49
50 // now let's put something deeper - into the chest
51 $item1= new OTS_Item(3015);
52 $chest-> addItem($item1);
53
54 // and more...
55 $item2= new OTS_Item(3013);
56 $chest-> addItem($item2);
57
58 // let's set count for an item
59 $item2-> setCount(2);
60
61 /*
62  Here is a tree of items which we created:
63

```



```

64 $container [depot locker]
65 `-- $chest [depot chest]
66     |-- $item1 [first item inserted into chest]
67     `-- $item2 [second item inserted into chest] count=2
68 */
69
70 /*
71     Items saving example.
72 */
73
74 // now we simply put those items into players depot (2 is depot ID)
75 $player-> setDepot(2, $container);
76
77 ?>

```

Important thing - OTS_Container class is subclass of OTS_Item. Each container is also an item.

Guilds

Guilds system basics.

Baiscs

Like for most other data types, for guilds and ranks there are two kinds of classes - single object class and list class. For guilds those are [OTS_Guild](#) and [OTS_Guilds_List](#), for ranks - [OTS_GuildRank](#) and [OTS_GuildRanks_List](#).

Guild management

Listing guilds is simple so there is no need to explain it more. More complex is listing guild members. Guild membership is not assigned directly - it is done throught guild ranks. To list guild members you first need to list it's ranks. Here is an example solution to list members in oryiginal Tibia-like way:

```
1  <?php
2
3  /**
4   * @ignore
5   * @package examples
6   * @author Wrzasq <wrzasq@gmail.com>
7   * @copyright 2007 (C) by Wrzasq
8   * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9   */
10
11 // to not repeat all that stuff
12 include('quickstart.php');
13
14 // loads guild
15 $guild= $ots-> createObject('Guild');
16 $guild-> load(1);
17
18 $color= '#FFFFCC';
19
20 echo '<h1>Members of ' , htmlspecialchars( $guild->getName() ), '</h1>' ;
21
22 ?>
23 <table>
24     <thead>
25         <tr>
26             <th>Rank</th>
27             <th>Members</th>
28         </tr>
29     </thead>
30     <tbody>
31 <?php
32
33 // lists members of all ranks
34 foreach( $guild-> getGuildRanks()as $guildRank)
35 {
36     // display rank in first row
37     $first= true;
```

```

38 // switches rank rows color
39 $color= $color== '#FFFFCC' ? '#FFCCFF' : '#FFFFCC';
40
41 // list members of this rank
42 foreach( $guildRank> getPlayers() as $player)
43 {
44     echo '<tr style="background-color: ' . $color .
45     '<td>' . $first?htmlspecialchars( $guildRank> getName() ) : "<td>' . $player> getName(), '</td>
46     '</tr>' . $first= false;
47     }
48 }
49 }
50 }
51
52 ?>
53 </tbody>
54 </table>

```

Guild action drivers

Handling invites/requests system for guilds.

How does it work?

OTServ database contains all guilds contents. But it is very common in AAC world to create invites system (or also requests system, but invitations are more common). It is not provided by standard OTServ database, though nearly all AAC scripts contains such mechanisms. POT classes allows you to set own drivers for invitations and requests to extend basic OTS functionality.

You have to write a driver class and assign it's object to guild object - then guild object will call requested actions on driver which will execute action code dependent on your script.

Driver structure

Both invites and requests drivers are similar - they must implement [IOTS_GuildAction interface](#). When the driver is assigned to guild object, each time a method of [OTS_Guild](#) object is called, it will forward this to action driver.

Sample driver

Driver implements your logic for invites (or membership requests). Here is sample code that you can base on:

```
1  <?php
2
3  /**
4   * @ignore
5   * @package examples
6   * @author Wrzasq <wrzasq@gmail.com>
7   * @copyright 2007 (C) by Wrzasq
8   * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9   */
10
11 // to not repeat all that stuff
12 include('quickstart.php');
13
14 /**
15  POT guilds invites driver.
16  */
17
18 /**
19  * @ignore
20  */
21 class InvitesDriver implements IOTS_GuildAction
22 {
23     // assigned guild
24     private $guild;
25
26     // initializes driver
27     public function __construct(OTS_Guild $guild)
28     {
```

```

29     $this>    guild= $guild
30     // this line automates the process - you can call it manually from outside, but why?
31     $this>    guild>    setInvitesDriver$this;
32 }
33
34 // returns all invited players to current guild
35 public functionlistRequests()
36 {
37     $invites= array();
38
39     /* here you must create OTS_Player object for each invited player */
40
41     return$invites
42 }
43
44 // invites player to current guild
45 public functionaddRequest(OTS_Player $player)
46 {
47     /* here you must save invitation for given player */
48 }
49
50 // un-invites player
51 public functiondeleteRequest(OTS_Player $player)
52 {
53     /* here you must delete invitation for given player */
54 }
55
56 // commits invitation
57 public functionsubmitRequest(OTS_Player $player)
58 {
59     $rank= null;
60
61     // finds normal member rank
62     foreach( $this>    guild>    getGuildRanks($s $guildRank)
63     {
64         if( $guildRank>    getLevel() == 1)
65         {
66             $rank= $guildRank
67             break;
68         }
69     }
70
71     $player>    setRank($rank);
72     $player>    save();
73
74     // clears invitation
75     $this>    deleteRequest($player);
76 }
77 }
78
79 /*
80 Parts of this class driver has been taken from OTSCMS (http://otscms.sourceforge.net/) project source
81 code.
82 */
83 // loads player wiht ID 1
84 $player= $ots>    createObject('Player');
85 $player>    load(1);
86

```

```
87 // loads guild with ID 1
88 $guild= $ots->    createObject('Guild');
89 $guild->    load(1);
90
91 // creates invitation logic driver for your implementation for current guild
92 new InvitesDriver( $guild);
93
94 // note that you call guild method!
95 $guild->    invite($player);
96
97 ?>
```

Account number hack

Example code of how to use prepared account number instead of random.

Walkaround

POT always generates random account number - [it is the way your script should work](#). It is done that way with premeditation. However you can walk around it with simple code:

```
1  <?php
2
3  /**
4   * @ignore
5   * @package examples
6   * @author Wrzasq <wrzasq@gmail.com>
7   * @copyright 2007 (C) by Wrzasq
8   * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9   */
10
11 // to not repeat all that stuff
12 include('quickstart.php');
13
14 // your non-random number
15 $number= 123456;
16
17 // creates new OTS_Account object
18 $account= $ots->createObject('Account');
19 $account->load($number);
20
21 // number is busy
22 if( $account->isLoaded() )
23 {
24     echo 'Account number ', $number, 'is used.', "\n" ;
25 }
26 // it is not
27 else
28 {
29     // generate number from exactly $number - $number range
30     $number= $account->create($number, $number);
31     echo 'Your account number is: ', $number, "\n" ;
32 }
33
34 ?>
```

Server online status

This tutorial will describe how to test server status with POT.

Such a simple way

[POT class](#) contains [serverStatus\(\) method](#) which sends 'info' packet to OTS and handles results. It returns object of class [OTS_InfoRespond](#) which provides access methods for all OTSserv respond info. It will return false if server is offline. Here is a simple example of this method usage:

```
1  <?php
2
3  /**
4   * @ignore
5   * @package examples
6   * @author Wrzasq <wrzasq@gmail.com>
7   * @copyright 2007 (C) by Wrzasq
8   * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9   */
10
11 // to not repeat all that stuff
12 include('quickstart.php');
13
14 // server and port
15 $server= '127.0.0.1';
16 $port= 7171;
17
18 // queries server of status info
19 $status= $ots-> serverStatus($server, $port);
20
21 // offline
22 if(!$status)
23 {
24     echo 'Server ', $server, ' is offline.', "\n" ;
25 }
26 // displays various info
27 else
28 {
29     echo 'Server name: ', $status-> getName(), "\n" ;
30     echo 'Server owner: ', $status-> getOwner(), "\n" ;
31     echo 'Players online: ', $status-> getOnlinePlayers(), "\n" ;
32     echo 'Maximum allowed number of players: ', $status-> getMaxPlayers(), "\n" ;
33     echo 'Required client version: ', $status-> getClientVersion(), "\n" ;
34     echo 'All monsters: ', $status-> getMonstersCount(), "\n" ;
35     echo 'Server message: ', $status-> getMOTD(), "\n" ;
36 }
37
38 ?>
```

DOM way

In case you would want to use this method for some non-SVN server which contains custom fields in respond packet you can still use it. OTS_InfoRespond class is child of DOMDocument class and doesn't overwrite it's

interface neither behaviour in any way. Returned object is standard DOM document so you can work with it in standard DOM-way.

About OTServ AAC scripts

This small article describes general info about OTServ AAC scripts.

Basics

Welcome! On this website you will find info about OTServ accmakers. This website is dedicated both for people who don't know anything about that and authors of such scripts. Beginners will find here basics and clues about how to use such scripts, as well as ready solutions for their's AAC. AAC creators should follow instructions on this site in order to make those scripts well.

Many people want to start using AAC scripts without knowledge. They spam forums, IRC channels and people IMs. If you don't know anything about AAC scripts, then this site is perfect for you. Before you will ask any question, read this website. If you will still don't know the answer, think before ask.

Mainly last times there went out plenty new AAC scripts. Too bad they are very poor and people who make them don't know anything about their's job. People started to think that if they made "own AAC" (which usually means to copy other script and sign with own nickname) they are cool - sorry guys - you only show how stupid your codes are. This website provides information about how to make good scripts and promotes ready solutions for safe and stable websites. We hope this website will change the situation and people who are creating accmakers will correct their's works or leave publishing bad scripts.

What is AAC?

AAC stands for Automatic Account Creator also called accmaker. Most generally it is a program (application, or script) that automates account creation process. However for a long time already simple accmakers aren't enough - nowadays average AAC should have additional options like account management, statistics and character lookup.

Types of AAC

Basically there two types of accmakers: websites and in-game. In-game AACs are NPCs that ask user for account and character information. To use such AAC person must log into special account (usually 1/1). Website AACs provides much more features - you can browse web from every place and from many devices. You can access it globally. Also those accmakers aren't restricted by Tibia client and can be extended in many ways. Usually website accmakers are PHP scripts and works on various HTTP servers.

Why not ingame

First when people of OTS world weren't familiar with PHP, HTTP servers they were just adding some code to server and account used to be created after logging in on special password where was NPC to complete the process. As OTS community were extending and new ideas came out, there appeared first website scripts - they provided at least so much functionality as NPC accmaker. With time website AACs were extending and now they usually contains many features that NPC would never have. Website AAC is accessible from every device where you have the Internet and browser so it means nearly every computer all over the world in this days. But those are all advantages of website AAC - there is one more reason which simply disqualifies in-game AACs: they are in fact impossible. Why? It is possible to create such AAC only if you add your server to many lists and links. To use in-game AAC people need to know your IP to connect and have Tibia client to use AAC. Normally they wouldn't know

that and Tibia client is not a standard application that is installed in every computer. The only way to provide accessibility for users is a website.

Website AAC HOWTO

Website AAC is most commonly used type of accmaker. People who want to use it first time find it hard to install and maintain. In fact it can be - you really need to know what you are doing with it and how does it work. Usually this type of AAC is a PHP script so we won't discuss other cases. To run PHP script you need a HTTP server - program which will provide website for people from outside, with installed PHP - interpreter of PHP scripts that executes them. It is quite easy to install Apache and PHP manually, but it is described all over the net, so we won't describe it here.

Main features

Of course basic AAC script feature, as the name says, must be account creation. But from the time when the first website accmaker was made (about 2004) scripts of that kind were extended and now "just AAC" is never enough. Empty site with only form for account creation shows that server is poor, that administrator doesn't care about it (and users) and he is probably a noob that just wanted to have "my own masta OTS". Currently even simple accmaker must provide some basic features.

Account creation is of course the most basic AAC feature. But this is also the point which is made wrong in nearly every script. Account number has to be random and generated during account creation, not during entering website by user. That's most important criterion which we used for our recommended scripts. This is for safety reasons, but not only. It is simply only possible way of correct implementation - people who make it other way simply don't know what they are doing as this is very unstable realisation.

On the beginning first accmaker was just a website form for creating account and character. But someone who made script was just lazy and finished work in that point. First more extended script was OTSCMS which was first that introduced login mechanism and allowed users to manage account from website. It means that they could create many characters on one account, or for example change password. Currently accmakers with only account creation form are not even worth to downloading (except PVP servers) and account managers are now standard.

Other important features of website AAC scripts are ideas based on [original Tibia](#) website. Many script contains statistics page where are listed players with highest scores, character view page where it is possible to check information about given player. Also lastly guilds system is very popular. Very important is, to merge website with OTS world and create some kind of community, it means that on website player should be affected, or at least connected with character in game.

Also very important is, to provide easy way of changing website behaviors and/or layout without editing script engine. Some scripts contain template engines, multilanguage support and modular structure. All those things give user ability to make website to look just like he wants. Everyone wants to have original website which will impress visitor.

Don't touch!

Before using, every AAC needs to be installed. It requires to put information about OTS as it needs to work on its database. Many dumb people create scripts and just put configuration file there so people will edit it. Too bad they don't even know about how to distribute PHP scripts, so how average user should know it? Script is a code - user mustn't touch the code. Code is a hermetic environment - when user will edit it and type something wrong it will crash. Every PHP script that needs to be edited in any way, includes installer, or just configuration editor. With

such script user fills settings on website and installer checks and validates them and then creates configuration file with saved settings.

Important notes

This website presents some ways of AAC script development that should be followed. It is not just our wish, but we presents clever and considered ideas. Those are just real points of view.

We want to promote "good scripts". If you have a script that fits our requirements you can contact us - we will add it to list of our recommended scripts.

Ready scripts

Here are links to some major AAC scripts:

- [OTSCMS](#) - uses POT.
- [SmartAss](#).
- [Nicaw CMS](#).
- [TauAccmaker](#).

Link



If you want to help us you can put following image on your website:

You can use following code for that:

```
1 <a href="http://otserv-aac.info/" >
2 
3 </a>
```


Package POT Procedural Elements

E_OTS_ErrorCode.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com >
- **Version** 0.0.6
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.6
- **License** [GNU Lesser General Public License, Version 3](#)

E_OTS_FileLoaderError.php

Code in this file bases on oryiginal OTServ binary format loading C++ code (fileloader.

Code in this file bases on oryiginal OTServ binary format loading C++ code (fileloader.h, fileloader.cpp).

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.6
- **License** [GNU Lesser General Public License, Version 3](#)

E_OTS_NoDriver.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.4
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.4
- **License** [GNU Lesser General Public License, Version 3](#)

E_OTS_NotLoaded.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.3
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.3
- **License** [GNU Lesser General Public License, Version 3](#)

E_OTS_OTBMErrror.php

Code in this file bases on oryiginal OTServ OTBM format loading C++ code (iomapotbm.

Code in this file bases on oryiginal OTServ OTBM format loading C++ code (iomapotbm.h, iomapotbm.cpp).

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.6
- **License** [GNU Lesser General Public License, Version 3](#)

E_OTS_OutOfBuffer.php

Code in this file bases on oryiginal OTServ binary format loading C++ code (fileloader.

Code in this file bases on oryiginal OTServ binary format loading C++ code (fileloader.h, fileloader.cpp).

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.6
- **License** [GNU Lesser General Public License, Version 3](#)

IOTS_DAO.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.5
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.1
- **License** [GNU Lesser General Public License, Version 3](#)

IOTS_DB.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Version** 0.0.1
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.1
- **License** [GNU Lesser General Public License, Version 3](#)

IOTS_FileCache.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.6
- **License** [GNU Lesser General Public License, Version 3](#)

IOTS_GuildAction.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.4
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.4
- **License** [GNU Lesser General Public License, Version 3](#)

OTS.php

This file contains main toolkit class.

This file contains main toolkit class. Please read README file for quick startup guide and/or tutorials for more info.

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.5
- **Version** 0.0.1
- **Copyright** 2007 (C) by Wrzasq
- **TODO** 0.1.0: Get rid of POT::getInstance()->create*() calls - use POT::getInstance()->getDBHandle() in constructors.
- **TODO** 0.0.7: Spells.
- **TODO** 0.0.8: Items list (items.xml + items.otb -> cache).
- **TODO** 0.1.0: Implement __get()/__set()/__call()/__toString(); ArrayAccess interface.
- **TODO** 1.0.0: Main POT class as database instance.
- **TODO** 1.0.0: Complete phpUnit test.
- **TODO** 1.0.0: More detailed documentation and tutorials, also update examples and tutorials.
- **Since** 0.0.1
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Account.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Version** 0.0.1
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.1
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Accounts_List.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.5
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.1
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Base_DAO.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com >
- **Version** 0.0.6
- **Version** 0.0.5
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.5
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Base_List.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Version** 0.0.5
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.5
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Container.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.3
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.3
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_DB_MySQL.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com >
- **Version** 0.0.6
- **Version** 0.0.1
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.1
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_DB_ODBC.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Version** 0.0.4
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.4
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_DB_PostgreSQL.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Version** 0.0.4
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.4
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_DB_SQLite.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com >
- **Version** 0.0.6
- **Version** 0.0.1
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.1
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_FileLoader.php

Code in this file bases on oryiginal OTServ binary format loading C++ code (fileloader.

Code in this file bases on oryiginal OTServ binary format loading C++ code (fileloader.h, fileloader.cpp).

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.6
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_FileNode.php

Code in this file bases on oryiginal OTServ binary format loading C++ code (fileloader.

Code in this file bases on oryiginal OTServ binary format loading C++ code (fileloader.h, fileloader.cpp).

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.6
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Group.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.5
- **Version** 0.0.1
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.1
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Groups_List.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.5
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.1
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Guild.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.5
- **Version** 0.0.4
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.4
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_GuildRank.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.5
- **Version** 0.0.4
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.4
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_GuildRanks_List.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.5
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.4
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Guilds_List.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.5
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.4
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_InfoRespond.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com >
- **Version** 0.0.2
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.2
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Item.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.3
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.3
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_MapCoords.php

Code in this file bases on oryiginal OTServ OTBM format loading C++ code (iomapotbm.

Code in this file bases on oryiginal OTServ OTBM format loading C++ code (iomapotbm.h, iomapotbm.cpp).

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.6
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Monster.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.6
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_OTBMFile.php

Code in this file bases on oryiginal OTServ OTBM format loading C++ code (iomapotbm.

Code in this file bases on oryiginal OTServ OTBM format loading C++ code (iomapotbm.h, iomapotbm.cpp).

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.6
- **Copyright** 2007 (C) by Wrzasq
- **TODO** 0.1.0: Houses support.
- **TODO** 1.0.0: Complete OTBM support: link tiles with items, spawns and houses.
- **TODO** 1.0.0: Spawns support.
- **Since** 0.0.6
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Player.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.5
- **Version** 0.0.1
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.1
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Players_List.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.5
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.1
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_SQLField.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.5
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.5
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_SQLFilter.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.5
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.5
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_SQLite_Results.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.1
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.1
- **License** [GNU Lesser General Public License, Version 3](#)

Package POT Classes

Class E_OTS_ErrorCode

[line 20]

Generic exception class for error codes.

Generic exception class for error codes.

- **Package** POT
- **Version** 0.0.6
- **Since** 0.0.6

Constructor *void* function E_OTS_ErrorCode::__construct(\$code) [line 27]

Function Parameters:

- *int* **\$code** Error code.

Sets error code.

Sets error code.

- **Version** 0.0.6
- **Since** 0.0.6

- **Access** public

Class E_OTS_FileLoaderError

[line 22]

Error during reading OTServ binary file.
Error during reading OTServ binary file.

- **Package** POT
- **Version** 0.0.6
- **Since** 0.0.6

E_OTS_FileLoaderError::ERROR_CAN_NOT_OPEN

= 2 [line 31]

Could not open file.
Could not open file.

- **Version** 0.0.6
- **Since** 0.0.6

E_OTS_FileLoaderError::ERROR_EOF

= 4 [line 35]

Unexpected end of file.
Unexpected end of file.

- **Version** 0.0.6
- **Since** 0.0.6

E_OTS_FileLoaderError::ERROR_INVALID_FILE_VERSION

= 1 [*line 27*]

Unsupported file version.
Unsupported file version.

- **Version** 0.0.6
- **Since** 0.0.6

E_OTS_FileLoaderError::ERROR_INVALID_FORMAT

= 8 [*line 47*]

File corrupted.
File corrupted.

- **Version** 0.0.6
- **Since** 0.0.6

E_OTS_FileLoaderError::ERROR_NOT_OPEN

= 6 [*line 43*]

Attempted to execute operation on not opened file.

Attempted to execute operation on not opened file.

- **Version** 0.0.6
- **Since** 0.0.6

E_OTS_FileLoaderError::ERROR_SEEK_ERROR

= 5 [*line 39*]

Failed to seek in given position in file.
Failed to seek in given position in file.

- **Version** 0.0.6
- **Since** 0.0.6

E_OTS_FileLoaderError::ERROR_TELL_ERROR

= 9 [*line 51*]

Failed to read position in file.
Failed to read position in file.

- **Version** 0.0.6
- **Since** 0.0.6

Class E_OTS_NoDriver

[line 20]

Occurs when code attempts to execute driven action that has no assigned driver to handle it.

Occurs when code attempts to execute driven action that has no assigned driver to handle it.

- **Package** POT
- **Version** 0.0.4
- **Since** 0.0.4

Class E_OTS_NotLoaded

[line 20]

Occurs when code attempts to access property of not loaded object.

Occurs when code attempts to access property of not loaded object.

- **Package** POT
- **Version** 0.0.3
- **Since** 0.0.3

Class E_OTS_OTBMEError

[line 22]

OTBM map loading error.
OTBM map loading error.

- **Package** POT
- **Version** 0.0.6
- **Since** 0.0.6

E_OTS_OTBMErrror::LOADMAPERROR_OUTDATEDHEADER

= 3 *[line 27]*

Unsupported file version.
Unsupported file version.

- **Version** 0.0.6
- **Since** 0.0.6

E_OTS_OTBMErrror::LOADMAPERROR_UNKNOWNNODETYPE

= 8 *[line 31]*

Unknown node type.
Unknown node type.

- **Version** 0.0.6
- **Since** 0.0.6

Class E_OTS_OutOfBuffer

[line 22]

Occurs when properties stream has ended and there is still read attempt.

Occurs when properties stream has ended and there is still read attempt.

- **Package** POT
- **Version** 0.0.6
- **Since** 0.0.6

Class IOTS_DAO

[line 22]

OTServ database object.

OTServ database object.

This interface indicates that class is a OTServ DAO class.

- **Package** POT
- **Version** 0.0.5
- **Since** 0.0.1

Constructor *void* function IOTS_DAO::__construct(\$db) *[line 30]*

Function Parameters:

- **PDO \$db** Database connection object.

DAO objects must be initialized with a database.

DAO objects must be initialized with a database.

- **Version** 0.0.5
- **Deprecated** 0.0.5 This constructor convention won't be part of interface in future.
- **Since** 0.0.1
- **Access** public

Class IOTS_DB

[line 25]

OTServ database handler interface.

OTServ database handler interface.

This interface specifies routines requires by DAO classes.

- **Package** POT
- **Version** 0.0.6
- **Version** 0.0.1
- **Deprecated** 0.0.5 Don't rely on this interface - it is for backward compatibility only. Check POT instance instead.
- **Since** 0.0.1

Constructor *void* function IOTS_DB::__construct(\$params) *[line 33]*

Function Parameters:

- **array \$params** Connection configuration.

Connection parameters.

Connection parameters.

- **Version** 0.0.6
- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

string function IOTS_DB::fieldName(\$name) [*line 41*]

Function Parameters:

- *string* **\$name** Field name.

Query-quoted field name.

Query-quoted field name.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

int function IOTS_DB::lastInsertId() [*line 68*]

ID of last created record.

ID of last created record.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

string function IOTS_DB::limit([\$limit = false], [\$offset = false]) [*line 76*]

Function Parameters:

- *int|bool* **\$limit** Limit of rows to be affected by query (false if no limit).
- *int|bool* **\$offset** Number of rows to be skipped before applying query effects (false if no offset).

LIMIT/OFFSET clause for queries.

LIMIT/OFFSET clause for queries.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

mixed function IOTS_DB::SQLquery(\$query) [*line 62*]

Function Parameters:

- *string* **\$query** Database query.

Evaluates query.

Evaluates query.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

string function IOTS_DB::SQLquote(\$value) [*line 55*]

Function Parameters:

- *string* **\$value** Value to be quoted to be suitable for database query.

Query-quoted string value.

Query-quoted string value.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

string function IOTS_DB::tableName(\$name) [*line 48*]

Function Parameters:

- *string* **\$name** Table name.

Query-quoted table name.

Query-quoted table name.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

Class IOTS_FileCache

[line 20]

This interface describe binary files cache control drivers.

This interface describe binary files cache control drivers.

- **Package** POT
- **Version** 0.0.6
- **Since** 0.0.6

OTS_FileNode | *null* function IOTS_FileCache::readCache(\$md5) [line 28]

Function Parameters:

- *string* **\$md5** MD5 hash of file.

Returns cache.

Returns cache.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

void function IOTS_FileCache::writeCache(\$md5, \$root) [line 35]

Function Parameters:

- *string* **\$md5** MD5 checksum of current file.
- [OTS_FileNode](#) **\$root** Root node of file which should be cached.

Writes node cache.

Writes node cache.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

Class IOTS_GuildAction

[line 32]

Guild action interface.

Guild action interface.

This interface indicates that class can handle OTServ guild action.

You can use it for example to handle invites or membership requests.

If you want to serialise (for example save in session) your guild objects with assigned drivers you need to implement also `__sleep()` and `__wakeup()` methods in your drivers, as assigned drivers are also serialised.

- **Package** POT
- **Version** 0.0.4

- **Since** 0.0.4

Constructor *void* function IOTS_GuildAction::__construct(\$guild) [*line 41*]

Function Parameters:

- [*OTS_Guild*](#) **\$guild** Guild that this driver is assigned to.

Objects are initialized with a guild that they are assigned to.

Objects are initialized with a guild that they are assigned to.

It is recommended that your implementations calls assignment functions of \$guild to automatically assign itself as action handler.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function IOTS_GuildAction::addRequest(\$player) [*line 54*]

Function Parameters:

- [*OTS_Player*](#) **\$player** Player which is object of request.

Adds new request.

Adds new request.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function IOTS_GuildAction::deleteRequest(\$player) [line 60]

Function Parameters:

- [OTS Player](#) **\$player** Player which is object of request.

Deletes request.

Deletes request.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

array function IOTS_GuildAction::listRequests() [line 48]

List of saved pending actions.

List of saved pending actions.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function IOTS_GuildAction::submitRequest(\$player) [line 66]

Function Parameters:

- [OTS Player](#) **\$player** Player which is object of request.

Finalizes request.

Finalizes request.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

Class OTS_Account

[line 22]

OTServ account abstraction.

OTServ account abstraction.

- **Package** POT
- **Version** 0.0.6
- **Version** 0.0.1
- **Since** 0.0.1

void function OTS_Account::ban([\$time = 0]) [line 464]

Function Parameters:

- *int* **\$time** Time for time until expires (0 - forever).

Bans current account.

Bans current account.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

void function OTS_Account::block() [*line 310*]

Blocks account.

Blocks account.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

int function OTS_Account::count() [*line 555*]

Returns number of player within.

Returns number of player within.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If account is not loaded.
- **Since** 0.0.5
- **Since** 0.0.1
- **Access** public

int function OTS_Account::create([\$min = 1], [\$max = 9999999]) [*line 47*]

account.php

```
1  <?php
2
3  /**
4   * @ignore
5   * @package examples
6   * @author Wrzasq <wrzasq@gmail.com>
7   * @copyright 2007 (C) by Wrzasq
8   * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9   */
10
11 // to not repeat all that stuff
12 include('quickstart.php');
13
14 // creates new OTS_Account object
15 $account = $ots->createObject('Account');
16
17 // generates new account number
18 $number = $account->create();
19
20 /*
21 to generate number from 111111 to 999999 use:
22 $number = $account->create(111111, 999999);
23 */
24
25 // sets account info
26 $account->setPassword('secret'); // $account->setPassword( md5('secret') );
27 $account->setEMail('foo@example.com');
28 $account->unlock(); // remember to unlock!
29 $account->setPACCDays(0);
30 $account->save();
31
32 // give user his number
33 echo 'Your account number is: ', $number;
34
35 ?>
```

Function Parameters:

- *int* **\$min** Minimum number.
- *int* **\$max** Maximum number.

Creates new account.

Creates new account.

Create new account in given range (1 - 9999999 by default).

Remember! This method sets blocked flag to true after account creation!

- **Version** 0.0.6
- **Version** 0.0.1
- **Throws** Exception When there are no free account numbers.

- **Since** 0.0.1
- **Access** public
- **Example**

int function OTS_Account::createEx(\$group, [\$min = 1], [\$max = 9999999]) [*line 115*]

Function Parameters:

- [OTS_Group](#) **\$group** Group to be assigned to account.
- *int* **\$min** Minimum number.
- *int* **\$max** Maximum number.

Creates new account.

Creates new account.

Create new account in given range (1 - 9999999 by default) in given group.

Remember! This method sets blocked flag to true after account creation!

IMPORTANT: Since 0.0.6 there isn't group_id field which this method was created for. You should use [create\(\)](#) method.

- **Version** 0.0.6_SVN
- **Version** 0.0.1
- **Deprecated** 0.0.6 There is no more group_id field in database, use create().
- **Since** 0.0.4
- **Since** 0.0.1
- **Access** public

void function OTS_Account::delete() [*line 518*]

Deletes account.

Deletes account.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If account is not loaded.
- **Since** 0.0.5
- **Since** 0.0.1
- **Access** public

void function OTS_Account::find(\$email) [*line 139*]

Function Parameters:

- *string* **\$email** Account's e-mail address.

Loads account by it's e-mail address.

Loads account by it's e-mail address.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.2
- **Access** public

string function OTS_Account::getCustomField(\$field) [*line 357*]

Function Parameters:

- *string* **\$field** Field name.

Reads custom field.

Reads custom field.

Reads field by it's name. Can read any field of given record that exists in database.

Note: You should use this method only for fields that are not provided in standard setters/getters (SVN fields). This method runs SQL query each time you call it so it highly overloads used resources.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTs_NotLoaded If account is not loaded.
- **Since** 0.0.3
- **Since** 0.0.1
- **Access** public

string function OTS_Account::getEmail() [*line 262*]

E-mail address.

E-mail address.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTs_NotLoaded If account is not loaded.
- **Since** 0.0.1
- **Access** public

OTS_Group function OTS_Account::getGroup() [*line 204*]

Returns group of this account.

Returns group of this account.

- **Version** 0.0.6
- **Version** 0.0.1
- **Deprecated** 0.0.6 There is no more group_id field in database.
- **Throws** E_OTS_NotLoaded If account is not loaded.
- **Since** 0.0.1
- **Since** 0.0.4
- **Access** public

int function OTS_Account::getId() [*line 185*]

Account number.

Account number.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If account is not loaded.
- **Since** 0.0.1
- **Access** public

Iterator function OTS_Account::getIterator() [*line 542*]

Returns players iterator.

Returns players iterator.

There is no need to implement entire Iterator interface since we have [players list class](#) for it.

- **Version** 0.0.5

- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If account is not loaded.
- **Since** 0.0.5
- **Since** 0.0.1
- **Access** public

int function OTS_Account::getPACCDays() [*line 323*]

PACC days.

PACC days.

- **Version** 0.0.4
- **Version** 0.0.1
- **Deprecated** 0.0.3 There is no more premdays field in accounts table.
- **Since** 0.0.1
- **Throws** E_OTS_NotLoaded If account is not loaded.
- **Access** public

string function OTS_Account::getPassword() [*line 235*]

Account's password.

Account's password.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If account is not loaded.
- **Since** 0.0.1
- **Access** public

array function OTS_Account::getPlayers() [*line 407*]

List of characters on account.

List of characters on account.

- **Version** 0.0.5
- **Version** 0.0.1
- **Deprecated** 0.0.5 Use getPlayersList().
- **Since** 0.0.1
- **Throws** E_OTS_NotLoaded If account is not loaded.
- **Access** public

OTS_Players_List function OTS_Account::getPlayersList() [*line 437*]

List of characters on account.

List of characters on account.

In difference to [getPlayers\(\) method](#) this method returns filtered [OTS_Players_List](#) object instead of array of [OTS_Player](#) objects. It is more effective since OTS_Player_List doesn't perform all rows loading at once.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If account is not loaded.
- **Since** 0.0.5
- **Since** 0.0.1
- **Access** public

bool function OTS_Account::isBanned() [*line 499*]

Checks if account is banned.

Checks if account is banned.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

bool function OTS_Account::isBlocked() [*line 289*]

Checks if account is blocked.

Checks if account is blocked.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If account is not loaded.
- **Since** 0.0.1
- **Access** public

bool function OTS_Account::isLoaded() [*line 156*]

Checks if object is loaded.

Checks if object is loaded.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Account::load(\$id) [line 126]

Function Parameters:

- **int \$id** Account number.

Loads account with given number.

Loads account with given number.

- **Version** 0.0.6
- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Account::save() [line 167]

Updates account in database.

Updates account in database.

- **Version** 0.0.6
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded False if account doesn't have ID assigned.
- **Since** 0.0.1
- **Access** public

void function OTS_Account::setCustomField(\$field, \$value) [line 383]

Function Parameters:

- *string* **\$field** Field name.
- *mixed* **\$value** Field value.

Writes custom field.

Writes custom field.

Write field by it's name. Can write any field of given record that exists in database.

Note: You should use this method only for fields that are not provided in standard setters/getters (SVN fields). This method runs SQL query each time you call it so it highly overloads used resources.

Note: Make sure that you pass \$value argument of correct type. This method determinates whether to quote field name. It is safe - it makes you sure that no unproper queries that could lead to SQL injection will be executed, but it can make your code working wrong way. For example: \$object->setCustomField('foo', '1'); will quote 1 as as string ('1') instead of passing it as a integer.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If account is not loaded.
- **Since** 0.0.3
- **Since** 0.0.1
- **Access** public

void function OTS_Account::setEMail(\$email) [*line 277*]

Function Parameters:

- *string* **\$email** E-mail address.

Sets account's email.

Sets account's email.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Account::setGroup(\$group) [line 224]

Function Parameters:

- [*OTS Group*](#) **\$group** Group to be a member.

Assigns account to group.

Assigns account to group.

- **Version** 0.0.6
- **Version** 0.0.1
- **Deprecated** 0.0.6 There is no more group_id field in database.
- **Since** 0.0.1
- **Access** public

void function OTS_Account::setPACCDays(\$premdays, \$pacc) [line 340]

Function Parameters:

- *int* **\$pacc** PACC days.
- **\$premdays**

Sets PACC days count.

Sets PACC days count.

- **Version** 0.0.4
- **Version** 0.0.1
- **Deprecated** 0.0.3 There is no more premdays field in accounts table.
- **Since** 0.0.1
- **Access** public

void function OTS_Account::setPassword(\$password) [line 250]

Function Parameters:

- *string* **\$password** Password.

Sets account's password.

Sets account's password.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Account::unban() [line 481]

Deletes ban from current account.

Deletes ban from current account.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5

- **Access** public

void function OTS_Account::unlock() [line 302]

Unblocks account.

Unblocks account.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

Class OTS_Accounts_List

[line 21]

List of accounts.

List of accounts.

- **Package** POT
- **Version** 0.0.5
- **Since** 0.0.1

void function OTS_Accounts_List::deleteAccount(\$account) [line 30]

Function Parameters:

- [OTS_Account](#) **\$account** Account to be deleted.

Deletes account.

Deletes account.

- **Version** 0.0.5
- **Deprecated** 0.0.5 Use `OTS_Account->delete()`.
- **Since** 0.0.1
- **Access** public

void function `OTS_Accounts_List::init()` [*line 43*]

Sets list parameters.

Sets list parameters.

This method is called at object creation.

- **Version** 0.0.5
- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

Class OTS_Base_DAO

[*line 22*]

Basic data access object routines.

Basic data access object routines.

- **Package** POT
- **Version** 0.0.6
- **Version** 0.0.5
- **Abstract Element**
- **Since** 0.0.5

OTS_Base_DAO::\$db

PDO = [line 29]

Database connection.

Database connection.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** protected

Constructor *void* function OTS_Base_DAO::__construct(\$db) *[line 36]*

Function Parameters:

- *PDO* **\$db** Database connection object.

Sets database connection handler.

Sets database connection handler.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

void function OTS_Base_DAO::__clone() [line 73]

Creates clone of object.

Creates clone of object.

Copy of object needs to have different ID.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

void function OTS_Base_DAO::__set_state(\$properties) [line 87]

Function Parameters:

- **array \$properties** List of object properties.

Magic PHP5 method.

Magic PHP5 method.

Allows object importing from [var_export\(\)](#).

- **Version** 0.0.6
- **Version** 0.0.5
- **Static**
- **Since** 0.0.5
- **Access** public

array function OTS_Base_DAO::__sleep() [line 49]

Magic PHP5 method.

Magic PHP5 method.
Allows object serialisation.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

void function OTS_Base_DAO::__wakeup() [*line 61*]

Magic PHP5 method.

Magic PHP5 method.
Allows object unserialisation.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

Class OTS_Base_List

[*line 22*]

Basic list class routines.

Basic list class routines.

- **Package** POT
- **Version** 0.0.6

- **Version** 0.0.5
- **Abstract Element**
- **Since** 0.0.5

OTS_Base_List::\$class

string = [line 78]

Class of generated objects.

Class of generated objects.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** protected

OTS_Base_List::\$table

string = [line 71]

Default table name for queries.

Default table name for queries.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** protected

Constructor *void* function OTS_Base_List::__construct(\$db) *[line 85]*

Function Parameters:

- **PDO \$db** Database connection object.

Sets database connection handler.

Sets database connection handler.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

int function OTS_Base_List::count() [*line 240*]

Returns number of accounts on list in current criterium.

Returns number of accounts on list in current criterium.

- **Version** 0.0.5
- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

/OTS_DAO function OTS_Base_List::current() [*line 189*]

Returns current row.

Returns current row.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

void function OTS_Base_List::init() [*line 94*]

Sets list parameters.

Sets list parameters.

- **Version** 0.0.5
- **Abstract Element**
- **Since** 0.0.5
- **Access** public

mixed function OTS_Base_List::key() [*line 219*]

Current cursor position.

Current cursor position.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

void function OTS_Base_List::next() [*line 209*]

Moves to next row.

Moves to next row.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

void function OTS_Base_List::orderBy(\$filed, [\$order = POT::ORDER_ASC], \$field) [line 272]

Function Parameters:

- *string* **\$field** Field name.
- *int* **\$order** Sorting order (ascending by default).
- **\$filed**

Appends sorting rule.

Appends sorting rule.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

void function OTS_Base_List::resetOrder() [line 261]

Clears ORDER BY clause.

Clears ORDER BY clause.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

void function OTS_Base_List::rewind() [line 201]

Select rows from database.

Select rows from database.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

void function OTS_Base_List::setFilter([\$filter = null]) [*line 253*]

Function Parameters:

- [*OTS_SQLFilter*](#)|*null* **\$filter** Filter for list.

Sets filter on list.

Sets filter on list.

Call without argument to reset filter.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

void function OTS_Base_List::setLimit([\$limit = false]) [*line 155*]

Function Parameters:

- *int|bool* **\$limit** Limit for SELECT (false to reset).

Sets LIMIT.

Sets LIMIT.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

void function OTS_Base_List::setOffset([\$offset = false]) [line 172]

Function Parameters:

- *int|bool* **\$offset** Offset for SELECT (false to reset).

Sets OFFSET.

Sets OFFSET.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

bool function OTS_Base_List::valid() [line 229]

Checks if there are any rows left.

Checks if there are any rows left.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

void function OTS_Base_List::__set_state(\$properties) [line 130]

Function Parameters:

- **array \$properties** List of object properties.

Magic PHP5 method.

Magic PHP5 method.

Allows object importing from [var_export\(\)](#).

- **Version** 0.0.6
- **Version** 0.0.5
- **Static**
- **Since** 0.0.5
- **Access** public

array function OTS_Base_List::__sleep() [line 104]

Magic PHP5 method.

Magic PHP5 method.

Allows object serialisation.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

void function OTS_Base_List::__wakeup() [line 116]

Magic PHP5 method.

Magic PHP5 method.

Allows object unserialisation.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

Class OTS_Container

[line 20]

Container item representation.

Container item representation.

- **Package** POT
- **Version** 0.0.3
- **Since** 0.0.3

void function OTS_Container::addItem(\$item) *[line 34]*

Function Parameters:

- [OTS_Item](#) \$item Item.

Adds item to container.

Adds item to container.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

int function `OTS_Container::count()` [*line 65*]

Number of items inside container.

Number of items inside container.

OTS_Container implementation of Countable interface differs from OTS_Item implementation. [OTS_Item::count\(\)](#) returns count of given item, `OTS_Container::count()` returns number of items inside container. If somehow it would be possible to make container items with more than 1 in one place, you can use [OTS_Item::getCount\(\)](#) and [OTS_Item::setCount\(\)](#) in code where you are not sure if working with regular item, or container.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

OTS_Item function `OTS_Container::current()` [*line 75*]

Returns current item.

Returns current item.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

mixed function `OTS_Container::key()` [*line 93*]

Current cursor position.

Current cursor position.

- **Version** 0.0.3

- **Since** 0.0.3
- **Access** public

void function OTS_Container::next() [line 83]

Moves to next item.

Moves to next item.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

void function OTS_Container::removeItem(\$item) [line 46]

Function Parameters:

- [*OTS_Item*](#) **\$item** Item.

Removes given item from current container.

Removes given item from current container.

Passed item must be exactly instance of item which is stored in container, not it's copy.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

void function OTS_Container::rewind() [line 111]

Resets internal items array pointer.

Resets internal items array pointer.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

bool function OTS_Container::valid() [*line 103*]

Checks if there are any items left.

Checks if there are any items left.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

Class OTS_DB_MySQL

[*line 22*]

MySQL connection interface.

MySQL connection interface.

- **Package** POT
- **Version** 0.0.6
- **Version** 0.0.1
- **Since** 0.0.1

Constructor *void* function OTS_DB_MySQL::__construct(\$params) [*line 50*]

Function Parameters:

- *array* **\$params** Connection parameters.

Creates database connection.

Creates database connection.

Connects to MySQL database on given arguments.

List of parameters for this drivers:

- *host* - database server.
- *port* - port (optional, also it is possible to use host:port in *host* parameter).
- *database* - database name.
- *user* - user login.
- *password* - user password.

- **Version** 0.0.6
- **Version** 0.0.1
- **See** [POT::connect\(\)](#)
- **Since** 0.0.1
- **Access** public

string function OTS_DB_MySQL::fieldName(\$name) [*line 105*]

Function Parameters:

- *string* **\$name** Field name.

Query-quoted field name.

Query-quoted field name.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

string function OTS_DB_MySQL::limit([\$limit = false], [\$offset = false]) [*line 158*]

Function Parameters:

- *int|bool* **\$limit** Limit of rows to be affected by query (false if no limit).
- *int|bool* **\$offset** Number of rows to be skipped before applying query effects (false if no offset).

LIMIT/OFFSET clause for queries.

LIMIT/OFFSET clause for queries.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

PDOStatement|bool function OTS_DB_MySQL::SQLquery(\$query) [*line 146*]

Function Parameters:

- *string* **\$query** SQL query.

IOTS_DB method.

IOTS_DB method.

Overwrites PDO method.

- **Version** 0.0.1
- **Deprecated** 0.0.5 Use PDO::query().
- **Since** 0.0.1
- **Access** public

string function OTS_DB_MySQL::SQLquote(\$string) [*line 131*]

Function Parameters:

- *string* **\$string** String to be quoted.

IOTS_DB method.

IOTS_DB method.

Overwrites PDO method - we won't use quoting against other values.

- **Version** 0.0.1
- **Deprecated** 0.0.5 Use PDO::quote().
- **Since** 0.0.1
- **Access** public

string function OTS_DB_MySQL::tableName(\$name) [*line 116*]

Function Parameters:

- *string* **\$name** Table name.

Query-quoted table name.

Query-quoted table name.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

Class OTS_DB_ODBC

[line 22]

ODBC connection interface.
ODBC connection interface.

- **Package** POT
- **Version** 0.0.6
- **Version** 0.0.4
- **Since** 0.0.4

Constructor *void* function OTS_DB_ODBC::__construct(\$params) [line 50]

Function Parameters:

- *array* **\$params** Connection parameters.

Creates database connection.

Creates database connection.

Connects to ODBC data source on given arguments.

List of parameters for this drivers:

- *host* - database host.
- *port* - ODBC driver.
- *database* - database name.
- *user* - user login.

- *password* - user password.

- **Version** 0.0.6
- **Version** 0.0.4
- **See** [POT::connect\(\)](#)
- **Since** 0.0.4
- **Access** public

string function OTS_DB_ODBC::fieldName(\$name) [*line 98*]

Function Parameters:

- *string* **\$name** Field name.

Query-quoted field name.

Query-quoted field name.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

string function OTS_DB_ODBC::limit([\$limit = false], [\$offset = false]) [*line 151*]

Function Parameters:

- *int|bool* **\$limit** Limit of rows to be affected by query (false if no limit).
- *int|bool* **\$offset** Number of rows to be skipped before applying query effects (false if no offset).

LIMIT/OFFSET clause for queries.

LIMIT/OFFSET clause for queries.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

PDOStatement|bool function OTS_DB_ODBC::SQLquery(\$query) [*line 139*]

Function Parameters:

- *string* **\$query** SQL query.

IOTS_DB method.

IOTS_DB method.

Overwrites PDO method.

- **Version** 0.0.4
- **Deprecated** 0.0.5 Use PDO::query().
- **Since** 0.0.4
- **Access** public

string function OTS_DB_ODBC::SQLquote(\$string) [*line 124*]

Function Parameters:

- *string* **\$string** String to be quoted.

IOTS_DB method.

IOTS_DB method.

Overwrites PDO method - we won't use quoting against other values.

- **Version** 0.0.4
- **Deprecated** 0.0.5 Use PDO::quote().
- **Since** 0.0.4
- **Access** public

string function OTS_DB_ODBC::tableName(\$name) [*line 109*]
Function Parameters:

- *string* **\$name** Table name.

Query-quoted table name.
Query-quoted table name.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

Class OTS_DB_PostgreSQL

[*line 22*]

PostgreSQL connection interface.
PostgreSQL connection interface.

- **Package** POT
- **Version** 0.0.6
- **Version** 0.0.4
- **Since** 0.0.4

Constructor *void* function OTS_DB_PostgreSQL::__construct(\$params) [*line 50*]

Function Parameters:

- *array* **\$params** Connection parameters.

Creates database connection.

Creates database connection.

Connects to PgSQL database on given arguments.

List of parameters for this drivers:

- *host* - database server.
- *port* - port (optional, also it is possible to use host:port in *host* parameter).
- *database* - database name.
- *user* - user login.
- *password* - user password.

- **Version** 0.0.6
- **Version** 0.0.4
- **See** [POT::connect\(\)](#)
- **Since** 0.0.4
- **Access** public

string function OTS_DB_PostgreSQL::fieldName(\$name) [*line 105*]

Function Parameters:

- *string* **\$name** Field name.

Query-quoted field name.

Query-quoted field name.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

string function OTS_DB_PostgreSQL::limit([\$limit = false], [\$offset = false]) [*line 158*]

Function Parameters:

- *int|bool* **\$limit** Limit of rows to be affected by query (false if no limit).
- *int|bool* **\$offset** Number of rows to be skipped before applying query effects (false if no offset).

LIMIT/OFFSET clause for queries.

LIMIT/OFFSET clause for queries.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

PDOStatement|bool function OTS_DB_PostgreSQL::SQLquery(\$query) [*line 146*]

Function Parameters:

- *string* **\$query** SQL query.

IOTS_DB method.

IOTS_DB method.
Overwrites PDO method.

- **Version** 0.0.4
- **Deprecated** 0.0.5 Use PDO::query().
- **Since** 0.0.4
- **Access** public

string function OTS_DB_PostgreSQL::SQLquote(\$string) [*line 131*]

Function Parameters:

- *string* **\$string** String to be quoted.

IOTS_DB method.

IOTS_DB method.
Overwrites PDO method - we won't use quoting agains other values.

- **Version** 0.0.4
- **Deprecated** 0.0.5 Use PDO::quote().
- **Since** 0.0.4
- **Access** public

string function OTS_DB_PostgreSQL::tableName(\$name) [*line 116*]

Function Parameters:

- *string* **\$name** Table name.

Query-quoted table name.

Query-quoted table name.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

Class OTS_DB_SQLite

[line 22]

SQLite connection interface.

SQLite connection interface.

- **Package** POT
- **Version** 0.0.6
- **Version** 0.0.1
- **Since** 0.0.1

Constructor *void* function OTS_DB_SQLite::__construct(\$params) [line 46]

Function Parameters:

- *array* **\$params** Connection parameters.

Creates database connection.

Creates database connection.

Connects to SQLite database on given arguments.

List of parameters for this drivers:

- *database* - database name.

- **Version** 0.0.6
- **Version** 0.0.1
- **See** [POT::connect\(\)](#)
- **Since** 0.0.1
- **Access** public

string function OTS_DB_SQLite::fieldName(\$name) [*line 66*]

Function Parameters:

- *string* **\$name** Field name.

Query-quoted field name.

Query-quoted field name.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

string function OTS_DB_SQLite::limit([\$limit = false], [\$offset = false]) [*line 119*]

Function Parameters:

- *int|bool* **\$limit** Limit of rows to be affected by query (false if no limit).
- *int|bool* **\$offset** Number of rows to be skipped before applying query effects (false if no offset).

LIMIT/OFFSET clause for queries.

LIMIT/OFFSET clause for queries.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

PDOStatement|bool function OTS_DB_SQLite::SQLquery(\$query) [*line 107*]

Function Parameters:

- *string* **\$query** SQL query.

IOTS_DB method.

IOTS_DB method.

Overwrites PDO method.

- **Version** 0.0.1
- **Deprecated** 0.0.5 Use PDO::query().
- **Since** 0.0.1
- **Access** public

string function OTS_DB_SQLite::SQLquote(\$string) [*line 92*]

Function Parameters:

- *string* **\$string** String to be quoted.

IOTS_DB method.

IOTS_DB method.

Overwrites PDO method - we won't use quoting against other values.

- **Version** 0.0.1
- **Deprecated** 0.0.5 Use PDO::quote().
- **Since** 0.0.1
- **Access** public

string function OTS_DB_SQLite::tableName(\$name) [line 77]

Function Parameters:

- *string* **\$name** Table name.

Query-quoted table name.

Query-quoted table name.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

Class OTS_FileLoader

[line 22]

Universal OTServ binary formats reader.

Universal OTServ binary formats reader.

- **Package** POT
- **Version** 0.0.6
- **Since** 0.0.6

OTS_FileLoader::ESCAPE_CHAR

= 0xFD *[line 35]*

Escape another special byte.

Escape another special byte.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_FileLoader::NODE_END

= 0xFF *[line 31]*

End of node.

End of node.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_FileLoader::NODE_START

= 0xFE [*line 27*]

Start of node.

Start of node.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_FileLoader::\$root

OTS_FileNode = [*line 49*]

Root node.

Root node.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** protected

void function OTS_FileLoader::loadFile(\$file) [*line 127*]

Function Parameters:

- *string* **\$file** Filepath.

Opens file.

Opens file.

- **Version** 0.0.6
- **Throws** `E_OTS_FileLoaderError` When error occurs during file operation.
- **Since** 0.0.6
- **Access** public

void function OTS_FileLoader::setCacheDriver([\$cache = null]) [line 116]

Function Parameters:

- [*`IOTS_FileCache`*](#) **\$cache** Cache handler (leave this parameter if you want to unset caching).

Sets cache handler.

Sets cache handler.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

void function OTS_FileLoader::__clone() [line 82]

Creates clone of object.

Creates clone of object.

Copy of object needs to have different ID.

- **Version** 0.0.6
- **Version** 0.0.6
- **Since** 0.0.6
- **Since** 0.0.6

- **Access** public

void function OTS_FileLoader::__set_state(\$properties) [*line 98*]

Function Parameters:

- *array* **\$properties** List of object properties.

Magic PHP5 method.

Magic PHP5 method.

Allows object importing from [var_export\(\)](#).

- **Version** 0.0.6
- **Version** 0.0.6
- **Static**
- **Since** 0.0.6
- **Since** 0.0.6
- **Access** public

array function OTS_FileLoader::__sleep() [*line 68*]

Magic PHP5 method.

Magic PHP5 method.

Allows object serialisation.

- **Version** 0.0.6
- **Version** 0.0.6
- **Since** 0.0.6
- **Since** 0.0.6
- **Access** public

Class OTS_FileNode

[line 22]

OTServ binary file node representation.

OTServ binary file node representation.

- **Package** POT
- **Version** 0.0.6
- **Since** 0.0.6

string function OTS_FileNode::getBuffer() [line 102]

Returns properties stream.

Returns properties stream.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

int function OTS_FileNode::getChar() [line 207]

Returns single byte.

Returns single byte.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

OTS_FileNode function *OTS_FileNode::getChild()* [*line 143*]

Returns first child.

Returns first child.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

int function *OTS_FileNode::getLong()* [*line 237*]

Returns quater byte.

Returns quater byte.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

OTS_FileNode function *OTS_FileNode::getNext()* [*line 123*]

Returns next sibling.

Returns next sibling.

- **Version** 0.0.6

- **Since** 0.0.6
- **Access** public

int function OTS_FileNode::getShort() [*line 222*]

Returns double byte.

Returns double byte.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

string function OTS_FileNode::getString([*\$length* = false]) [*line 255*]

Function Parameters:

- *int|bool* **\$length** String length.

Returns string from buffer.

Returns string from buffer.

If length is not given then treats first byte from current buffer as string length.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

int function OTS_FileNode::getType() [*line 163*]

Returns node type.

Returns node type.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

bool function OTS_FileNode::isValid() [*line 183*]

Checks if there is anything left in stream.

Checks if there is anything left in stream.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

void function OTS_FileNode::setBuffer(\$buffer) [*line 112*]

Function Parameters:

- *string* **\$buffer** Properties stream.

Sets properties stream.

Sets properties stream.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

void function OTS_FileNode::setChild(\$child) [line 153]

Function Parameters:

- [*OTS_FileNode*](#) **\$child** Child node.

Sets first child.

Sets first child.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

void function OTS_FileNode::setNext(\$next) [line 133]

Function Parameters:

- [*OTS_FileNode*](#) **\$next** Sibling node.

Sets next sibling.

Sets next sibling.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

void function OTS_FileNode::setType(\$type) [line 173]

Function Parameters:

- *int* **\$type** Node type.

Sets node type.

Sets node type.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

void function OTS_FileNode::skip(\$n) [*line 277*]

Function Parameters:

- *int* **\$n** Bytes to skip.

Skips given amount of bytes.

Skips given amount of bytes.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

void function OTS_FileNode::__clone() [*line 62*]

Creates clone of object.

Creates clone of object.

Copy of object needs to have different ID.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

`void function OTS_FileNode::__set_state($properties) [line 84]`

Function Parameters:

- `array $properties` List of object properties.

Magic PHP5 method.

Magic PHP5 method.

Allows object importing from [var_export\(\)](#).

- **Version** 0.0.6
- **Static**
- **Since** 0.0.6
- **Access** public

Class OTS_Group

[line 22]

OTServ user group abstraction.

OTServ user group abstraction.

- **Package** POT

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1

int function OTS_Group::count() [*line 385*]

Returns number of player within.

Returns number of player within.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If group is not loaded.
- **Since** 0.0.5
- **Since** 0.0.1
- **Access** public

void function OTS_Group::delete() [*line 348*]

Deletes group.

Deletes group.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If group is not loaded.
- **Since** 0.0.5
- **Since** 0.0.1
- **Access** public

int function OTS_Group::getAccess() [*line 154*]

Access level.

Access level.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If group is not loaded.
- **Since** 0.0.1
- **Access** public

string function OTS_Group::getCustomField(\$field) [*line 241*]

Function Parameters:

- *string* **\$field** Field name.

Reads custom field.

Reads custom field.

Reads field by it's name. Can read any field of given record that exists in database.

Note: You should use this method only for fields that are not provided in standard setters/getters (SVN fields). This method runs SQL query each time you call it so it highly overloads used resources.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If group is not loaded.
- **Since** 0.0.3
- **Since** 0.0.1
- **Access** public

int function OTS_Group::getFlags() [*line 127*]

Rights flags.

Rights flags.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If group is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Group::getId() [*line 83*]

Group ID.

Group ID.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If group is not loaded.
- **Since** 0.0.1
- **Access** public

Iterator function OTS_Group::getIterator() [*line 372*]

Returns players iterator.

Returns players iterator.

There is no need to implement entire Iterator interface since we have [players list class](#) for it.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If group is not loaded.
- **Since** 0.0.5
- **Since** 0.0.1
- **Access** public

int function OTS_Group::getMaxDepotItems() [*line 181*]

Maximum count of items in depot.

Maximum count of items in depot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If group is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Group::getMaxVIPList() [*line 208*]

Maximum count of players in VIP list.

Maximum count of players in VIP list.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If group is not loaded.
- **Since** 0.0.1
- **Access** public

string function OTS_Group::getName() [*line 100*]

Group name.

Group name.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If group is not loaded.
- **Since** 0.0.1
- **Access** public

array function OTS_Group::getPlayers() [*line 291*]

List of characters in given group.

List of characters in given group.

- **Version** 0.0.5
- **Version** 0.0.1
- **Deprecated** 0.0.5 Use getPlayersList().
- **Since** 0.0.1
- **Throws** E_OTS_NotLoaded If group is not loaded.
- **Access** public

OTS_Players_List function OTS_Group::getPlayersList() [*line 321*]

List of characters in group.

List of characters in group.

In difference to [getPlayers\(\) method](#) this method returns filtered [OTS Players List](#) object instead of array of [OTS Player](#) objects. It is more effective since OTS_Player_List doesn't perform all rows loading at once.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If group is not loaded.
- **Since** 0.0.5
- **Since** 0.0.1
- **Access** public

bool function OTS_Group::isLoaded() [*line 48*]

Checks if object is loaded.

Checks if object is loaded.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Group::load(\$id) [*line 37*]

Function Parameters:

- *int* **\$id** Group number.

Loads group with given id.

Loads group with given id.

- **Version** 0.0.5

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Group::save() [line 58]

Saves account in database.

Saves account in database.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Group::setAccess(\$access) [line 169]

Function Parameters:

- *int* **\$access** Access level.

Sets access level.

Sets access level.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Group::setCustomField(\$field, \$value) [line 267]

Function Parameters:

- *string* **\$field** Field name.
- *mixed* **\$value** Field value.

Writes custom field.

Writes custom field.

Write field by it's name. Can write any field of given record that exists in database.

Note: You should use this method only for fields that are not provided in standard setters/getters (SVN fields). This method runs SQL query each time you call it so it highly overloads used resources.

Note: Make sure that you pass \$value argument of correct type. This method determinates whether to quote field name. It is safe - it makes you sure that no unproper queries that could lead to SQL injection will be executed, but it can make your code working wrong way. For example: \$object->setCustomField('foo', '1'); will quote 1 as as string ('1') instead of passing it as a integer.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If group is not loaded.
- **Since** 0.0.3
- **Since** 0.0.1
- **Access** public

void function OTS_Group::setFlags(\$flags) [*line 142*]

Function Parameters:

- *int* **\$flags** Flags.

Sets rights flags.

Sets rights flags.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Group::setMaxDepotItems(\$maxdepotitems) [line 196]

Function Parameters:

- *int* **\$maxdepotitems** Maximum value.

Sets maximum count of items in depot.

Sets maximum count of items in depot.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Group::setMaxVIPList(\$maxviplist, \$maxdepotitems) [line 223]

Function Parameters:

- *int* **\$maxdepotitems** Maximum value.
- **\$maxviplist**

Sets maximum count of players in VIP list.

Sets maximum count of players in VIP list.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Group::setName(\$name) [*line 115*]
Function Parameters:

- *string* **\$name** Name.

Sets group's name.
 Sets group's name.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

Class OTS_Groups_List

[*line 21*]

List of groups.
 List of groups.

- **Package** POT
- **Version** 0.0.5
- **Since** 0.0.1

void function OTS_Groups_List::deleteGroup(\$group) [line 30]

Function Parameters:

- [*OTS_Group*](#) **\$group** Group to be deleted.

Deletes group.

Deletes group.

- **Version** 0.0.5
- **Deprecated** 0.0.5 Use OTS_Group->delete().
- **Since** 0.0.1
- **Access** public

void function OTS_Groups_List::init() [line 43]

Sets list parameters.

Sets list parameters.

This method is called at object creation.

- **Version** 0.0.5
- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

Class OTS_Guild

[line 22]

OTServ guild abstraction.

OTServ guild abstraction.

- **Package** POT
- **Version** 0.0.5
- **Version** 0.0.4
- **Since** 0.0.4

void function OTS_Guild::acceptInvite(\$player) [line 442]

Function Parameters:

- [OTS_Player](#) **\$player** Player to be joined.

Finalise invitation.

Finalise invitation.

- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If guild is not loaded.
- **Throws** E_OTS_NoDriver If there is no invites driver assigned.
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::acceptRequest(\$player) [line 534]

Function Parameters:

- [OTS_Player](#) \$player Player to be accepted.

Accepts player.

Accepts player.

- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If guild is not loaded.
- **Throws** E_OTS_NoDriver If there is no requests driver assigned.
- **Since** 0.0.4
- **Access** public

int function OTS_Guild::count() [*line 594*]

Returns number of ranks within.

Returns number of ranks within.

- **Version** 0.0.5
- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If guild is not loaded.
- **Since** 0.0.5
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::delete() [*line 557*]

Deletes guild.

Deletes guild.

- **Version** 0.0.5
- **Version** 0.0.4
- **Throws** E_OTTS_NotLoaded If guild is not loaded.
- **Since** 0.0.5
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::deleteInvite(\$player) [line 419]

Function Parameters:

- [*OTS_Player*](#) **\$player** Player to be un-invited.

Deletes invitation for player to guild.

Deletes invitation for player to guild.

- **Version** 0.0.4
- **Throws** E_OTTS_NotLoaded If guild is not loaded.
- **Throws** E_OTTS_NoDriver If there is no invites driver assigned.
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::deleteRequest(\$player) [line 511]

Function Parameters:

- [*OTS_Player*](#) **\$player** Player to be rejected.

Deletes request from player.

Deletes request from player.

- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If guild is not loaded.
- **Throws** E_OTS_NoDriver If there is no requests driver assigned.
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::find(\$name) [*line 114*]

Function Parameters:

- *string* **\$name** Guild's name.

Loads guild by it's name.

Loads guild by it's name.

- **Version** 0.0.5
- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

int function OTS_Guild::getCreationData() [*line 235*]

Guild creation data.

Guild creation data.

- **Version** 0.0.4
- **Throws** E_OTTS_NotLoaded If guild is not loaded.
- **Since** 0.0.4
- **Access** public

string function OTS_Guild::getCustomField(\$field) [*line 267*]

Function Parameters:

- *string* **\$field** Field name.

Reads custom field.

Reads custom field.

Reads field by it's name. Can read any field of given record that exists in database.

Note: You should use this method only for fields that are not provided in standard setters/getters (SVN fields). This method runs SQL query each time you call it so it highly overloads used resources.

- **Version** 0.0.5
- **Version** 0.0.4
- **Throws** E_OTTS_NotLoaded If guild is not loaded.
- **Since** 0.0.4
- **Access** public

array function OTS_Guild::getGuildRanks() [*line 316*]

Reads all ranks that are in this guild.

Reads all ranks that are in this guild.

- **Version** 0.0.5

- **Version** 0.0.4
- **Deprecated** 0.0.5 Use `getGuildRanksList()`.
- **Since** 0.0.4
- **Throws** `E_OTS_NotLoaded` If guild is not loaded.
- **Access** public

OTS_GuildRanks_List function `OTS_Guild::getGuildRanksList()` [*line 346*]

List of ranks in guild.

List of ranks in guild.

In difference to [getGuildRanks\(\) method](#) this method returns filtered [OTS_GuildRanks_List](#) object instead of array of [OTS_GuildRank](#) objects. It is more effective since `OTS_GuildRanks_List` doesn't perform all rows loading at once.

- **Version** 0.0.5
- **Version** 0.0.4
- **Throws** `E_OTS_NotLoaded` If guild is not loaded.
- **Since** 0.0.5
- **Since** 0.0.4
- **Access** public

int function `OTS_Guild::getId()` [*line 165*]

Guild ID.

Guild ID.

- **Version** 0.0.4
- **Throws** `E_OTS_NotLoaded` If guild is not loaded.
- **Since** 0.0.4
- **Access** public

Iterator function OTS_Guild::getIterator() [*line 581*]

Returns ranks iterator.

Returns ranks iterator.

There is no need to implement entire Iterator interface since we have [ranks list class](#) for it.

- **Version** 0.0.5
- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If guild is not loaded.
- **Since** 0.0.5
- **Since** 0.0.4
- **Access** public

string function OTS_Guild::getName() [*line 181*]

Guild name.

Guild name.

- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If guild is not loaded.
- **Since** 0.0.4
- **Access** public

OTS_Player function OTS_Guild::getOwner() [*line 207*]

Returns owning player of this player.

Returns owning player of this player.

- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If guild is not loaded.
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::invite(\$player) [*line 396*]

Function Parameters:

- [*OTS_Player*](#) **\$player** Player to be invited.

Invites player to guild.

Invites player to guild.

- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If guild is not loaded.
- **Throws** E_OTS_NoDriver If there is no invites driver assigned.
- **Since** 0.0.4
- **Access** public

bool function OTS_Guild::isLoaded() [*line 131*]

Checks if object is loaded.

Checks if object is loaded.

- **Version** 0.0.4
- **Since** 0.0.4

- **Access** public

array function OTS_Guild::listInvites() [*line 373*]

Returns list of invited players.

Returns list of invited players.

- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If guild is not loaded.
- **Throws** E_OTS_NoDriver If there is no invites driver assigned.
- **Since** 0.0.4
- **Access** public

array function OTS_Guild::listRequests() [*line 465*]

Returns list of players that requested membership.

Returns list of players that requested membership.

- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If guild is not loaded.
- **Throws** E_OTS_NoDriver If there is no requests driver assigned.
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::load(\$id) [*line 102*]

Function Parameters:

- *int* **\$id** Guild's ID.

Loads guild with given id.

Loads guild with given id.

- **Version** 0.0.5
- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::request(\$player) [line 488]

Function Parameters:

- [*OTS_Player*](#) **\$player** Player that requested membership.

Requests membership in guild for player player.

Requests membership in guild for player player.

- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If guild is not loaded.
- **Throws** E_OTS_NoDriver If there is no requests driver assigned.
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::save() [line 141]

Saves guild in database.

Saves guild in database.

- **Version** 0.0.5
- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::setCreationData(\$creationdata) [line 250]

Function Parameters:

- *int* **\$creationdata** Guild creation data.

Sets guild creation data.

Sets guild creation data.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::setCustomField(\$field, \$value) [line 292]

Function Parameters:

- *string* **\$field** Field name.
- *mixed* **\$value** Field value.

Writes custom field.

Writes custom field.

Write field by it's name. Can write any field of given record that exists in database.

Note: You should use this method only for fields that are not provided in standard setters/getters (SVN fields). This method runs SQL query each time you call it so it highly overloads used resources.

Note: Make sure that you pass \$value argument of correct type. This method determinates whether to quote field value. It is safe - it makes you sure that no unproper queries that could lead to SQL injection will be executed, but it can make your code working wrong way. For example: \$object->setCustomField('foo', '1'); will quote 1 as as string ('1') instead of passing it as a integer.

- **Version** 0.0.5
- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If guild is not loaded.
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::setInvitesDriver([\$invites = null]) [line 81]

Function Parameters:

- [*IOTS_GuildAction*](#) **\$invites** Invites driver (don't pass it to clear driver).

Assigns invites handler.

Assigns invites handler.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::setName(\$name) [line 196]

Function Parameters:

- *string* **\$name** Name.

Sets players's name.

Sets players's name.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::setOwner(\$owner) [line 224]

Function Parameters:

- [*OTS_Player*](#) **\$owner** Owning player.

Assigns guild to owner.

Assigns guild to owner.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::setRequestsDriver([\$requests = null]) [line 91]

Function Parameters:

- [*IOTS_GuildAction*](#) **\$requests** Membership requests driver (don't pass it to clear driver).

Assigns requests handler.

Assigns requests handler.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function OTS_Guild::__clone() [*line 65*]

Creates clone of object.

Creates clone of object.

Copy of object needs to have different ID.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

array function OTS_Guild::__sleep() [*line 53*]

Magic PHP5 method.

Magic PHP5 method.

Allows object serialisation.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

Class OTS_GuildRank

[line 22]

OTServ guild rank abstraction.

OTServ guild rank abstraction.

- **Package** POT
- **Version** 0.0.5
- **Version** 0.0.4
- **Since** 0.0.4

int function OTS_GuildRank::count() [line 356]

Returns number of player within.

Returns number of player within.

- **Version** 0.0.5
- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If rank is not loaded.
- **Since** 0.0.5
- **Since** 0.0.4
- **Access** public

void function OTS_GuildRank::delete() [line 319]

Deletes guild rank.

Deletes guild rank.

- **Version** 0.0.5
- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If guild rank is not loaded.
- **Since** 0.0.5
- **Since** 0.0.4
- **Access** public

void function OTS_GuildRank::find(\$name, [\$guild = null]) [*line 52*]

Function Parameters:

- *string* **\$name** Rank's name.
- [*OTS_Guild*](#) **\$guild** Guild in which rank should be found.

Loads rank by it's name.

Loads rank by it's name.

As there can be several ranks with same name in different guilds you can pass optional second parameter to specify in which guild script should look for rank.

- **Version** 0.0.5
- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

string function OTS_GuildRank::getCustomField(\$field) [*line 213*]

Function Parameters:

- *string* **\$field** Field name.

Reads custom field.

Reads custom field.

Reads field by it's name. Can read any field of given record that exists in database.

Note: You should use this method only for fields that are not provided in standard setters/getters (SVN fields). This method runs SQL query each time you call it so it highly overloads used resources.

- **Version** 0.0.5
- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If rank is not loaded.
- **Since** 0.0.4
- **Access** public

OTS_Guild function OTS_GuildRank::getGuild() [*line 153*]

Returns guild of this rank.

Returns guild of this rank.

- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If rank is not loaded.
- **Since** 0.0.4
- **Access** public

int function OTS_GuildRank::getId() [*line 111*]

Rank ID.

Rank ID.

- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If rank is not loaded.

- **Since** 0.0.4
- **Access** public

Iterator function OTS_GuildRank::getIterator() [*line 343*]

Returns players iterator.

Returns players iterator.

There is no need to implement entire Iterator interface since we have [players list class](#) for it.

- **Version** 0.0.5
- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If rank is not loaded.
- **Since** 0.0.5
- **Since** 0.0.4
- **Access** public

int function OTS_GuildRank::getLevel() [*line 181*]

Rank's access level.

Rank's access level.

- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If rank is not loaded.
- **Since** 0.0.4
- **Access** public

string function OTS_GuildRank::getName() [*line 127*]

Rank name.

Rank name.

- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If rank is not loaded.
- **Since** 0.0.4
- **Access** public

array function OTS_GuildRank::getPlayers() [*line 262*]

Reads all players who has this rank set.

Reads all players who has this rank set.

- **Version** 0.0.5
- **Version** 0.0.4
- **Deprecated** 0.0.5 Use getPlayersList().
- **Since** 0.0.4
- **Throws** E_OTS_NotLoaded If rank is not loaded.
- **Access** public

OTS_Players_List function OTS_GuildRank::getPlayersList() [*line 292*]

List of characters with current rank.

List of characters with current rank.

In difference to [getPlayers\(\) method](#) this method returns filtered [OTS Players List](#) object instead of array of [OTS Player](#) objects. It is more effective since OTS_Player_List doesn't perform all rows loading at once.

- **Version** 0.0.5
- **Version** 0.0.4

- **Throws** `E_OTS_NotLoaded` If rank is not loaded.
- **Since** 0.0.5
- **Since** 0.0.4
- **Access** public

bool function `OTS_GuildRank::isLoaded()` [*line 77*]

Checks if object is loaded.

Checks if object is loaded.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function `OTS_GuildRank::load($id)` [*line 37*]

Function Parameters:

- *int* **\$id** Rank's ID.

Loads rank with given id.

Loads rank with given id.

- **Version** 0.0.5
- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function OTS_GuildRank::save() [line 87]

Saves rank in database.

Saves rank in database.

- **Version** 0.0.5
- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function OTS_GuildRank::setCustomField(\$field, \$value) [line 238]

Function Parameters:

- *string* **\$field** Field name.
- *mixed* **\$value** Field value.

Writes custom field.

Writes custom field.

Write field by it's name. Can write any field of given record that exists in database.

Note: You should use this method only for fields that are not provided in standard setters/getters (SVN fields). This method runs SQL query each time you call it so it highly overloads used resources.

Note: Make sure that you pass \$value argument of correct type. This method determinates whether to quote field value. It is safe - it makes you sure that no unproper queries that could lead to SQL injection will be executed, but it can make your code working wrong way. For example: \$object->setCustomField('foo', '1'); will quote 1 as as string ('1') instead of passing it as a integer.

- **Version** 0.0.5
- **Version** 0.0.4
- **Throws** E_OTS_NotLoaded If rank is not loaded.

- **Since** 0.0.4
- **Access** public

void function OTS_GuildRank::setGuild(\$guild) [line 170]

Function Parameters:

- [*OTS_Guild*](#) **\$guild** Owing guild.

Assigns rank to guild.

Assigns rank to guild.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function OTS_GuildRank::setLevel(\$level) [line 196]

Function Parameters:

- *int* **\$level** access level within guild.

Sets rank's access level within guild.

Sets rank's access level within guild.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

void function OTS_GuildRank::setName(\$name) [*line 142*]

Function Parameters:

- *string* **\$name** Name.

Sets rank's name.

Sets rank's name.

- **Version** 0.0.4
- **Since** 0.0.4
- **Access** public

Class OTS_GuildRanks_List

[*line 21*]

List of guild ranks.

List of guild ranks.

- **Package** POT
- **Version** 0.0.5
- **Since** 0.0.4

void function OTS_GuildRanks_List::deleteGuildRank(\$guildRank) [*line 30*]

Function Parameters:

- [OTS_GuildRank](#) \$guildRank Rank to be deleted.

Deletes guild rank.

Deletes guild rank.

- **Version** 0.0.5
- **Deprecated** 0.0.5 Use OTS_GuildRank->delete().
- **Since** 0.0.4
- **Access** public

void function OTS_GuildRanks_List::init() [*line 43*]

Sets list parameters.

Sets list parameters.

This method is called at object creation.

- **Version** 0.0.5
- **Since** 0.0.4
- **Since** 0.0.5
- **Access** public

Class OTS_Guilds_List

[*line 21*]

List of guilds.

List of guilds.

- **Package** POT
- **Version** 0.0.5
- **Since** 0.0.4

void function OTS_Guilds_List::deleteGuild(\$guild) [*line 30*]

Function Parameters:

- [*OTS_Guild*](#) **\$guild** Guild to be deleted.

Deletes guild.

Deletes guild.

- **Version** 0.0.5
- **Deprecated** 0.0.5 Use OTS_Guild->delete().
- **Since** 0.0.4
- **Access** public

void function OTS_Guilds_List::init() [*line 43*]

Sets list parameters.

Sets list parameters.

This method is called at object creation.

- **Version** 0.0.5
- **Since** 0.0.4
- **Since** 0.0.5

- **Access** public

Class OTS_InfoRespond

[line 22]

Wrapper for 'info' respond's DOMDocument.

Wrapper for 'info' respond's DOMDocument.

Note: as this class extends DOMDocument class and contains exactly respond XML tree you can work on it as on normal DOM tree.

- **Package** POT
- **Version** 0.0.2
- **Since** 0.0.2

string function OTS_InfoRespond::getClientVersion() *[line 121]*

Returns dedicated version of client.

Returns dedicated version of client.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

string function OTS_InfoRespond::getEmail() *[line 141]*

Returns owner e-mail.

Returns owner e-mail.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

string function OTS_InfoRespond::getIP() [*line 49*]

Returns server IP.

Returns server IP.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

string function OTS_InfoRespond::getLocation() [*line 79*]

Returns server location.

Returns server location.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

string function OTS_InfoRespond::getMapAuthor() [*line 202*]

Returns map author.

Returns map author.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

int function OTS_InfoRespond::getMapHeight() [*line 222*]

Returns map height.

Returns map height.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

string function OTS_InfoRespond::getMapName() [*line 191*]

Returns map name.

Returns map name.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

int function OTS_InfoRespond::getMapWidth() [*line 212*]

Returns map width.

Returns map width.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

int function OTS_InfoRespond::getMaxPlayers() [*line 161*]

Returns maximum amount of players online.

Returns maximum amount of players online.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

int function OTS_InfoRespond::getMonstersCount() [*line 181*]

Returns number of all monsters on map.

Returns number of all monsters on map.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

string function OTS_InfoRespond::getMOTD() [*line 232*]

Returns server's Message Of The Day

Returns server's Message Of The Day

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

string function OTS_InfoRespond::getName() [*line 59*]

Returns server name.

Returns server name.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

int function OTS_InfoRespond::getOnlinePlayers() [*line 151*]

Returns current amount of players online.

Returns current amount of players online.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

string function OTS_InfoRespond::getOwner() [*line 131*]

Returns owner name.

Returns owner name.

- **Version** 0.0.2

- **Since** 0.0.2
- **Access** public

int function OTS_InfoRespond::getPlayersPeak() [*line 171*]

Returns record of online players.

Returns record of online players.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

int function OTS_InfoRespond::getPort() [*line 69*]

Returns server port.

Returns server port.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

string function OTS_InfoRespond::getServer() [*line 101*]

Returns server attribute.

Returns server attribute.

I have no idea what the hell is it representing :P.

- **Version** 0.0.2

- **Since** 0.0.2
- **Access** public

string function OTS_InfoRespond::getServerVersion() [*line 111*]

Returns server version.

Returns server version.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

string function OTS_InfoRespond::getTSPQVersion() [*line 29*]

Returns version of root element.

Returns version of root element.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

int function OTS_InfoRespond::getUptime() [*line 39*]

Returns server uptime.

Returns server uptime.

- **Version** 0.0.2
- **Since** 0.0.2

- **Access** public

string function OTS_InfoRespond::getURL() [*line 89*]

Returns server website.

Returns server website.

- **Version** 0.0.2
- **Since** 0.0.2
- **Access** public

Class OTS_Item

[*line 20*]

Single item representation.

Single item representation.

- **Package** POT
- **Version** 0.0.3
- **Since** 0.0.3

Constructor *void* function OTS_Item::__construct(\$id) [*line 48*]

Function Parameters:

- *int* **\$id** Item ID.

Creates item of given ID.
Creates item of given ID.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

int function OTS_Item::count() [*line 108*]

Count value for current item.
Count value for current item.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

string function OTS_Item::getAttributes() [*line 88*]

Returns item custom attributes.
Returns item custom attributes.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

int function OTS_Item::getCount() [*line 68*]

Returns count of item.

Returns count of item.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

int function OTS_Item::getId() [*line 58*]

Returns item type.

Returns item type.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

void function OTS_Item::setAttributes(\$attributes) [*line 98*]

Function Parameters:

- *string* **\$attributes** Item Attributes.

Sets item attributes.

Sets item attributes.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

void function OTS_Item::setCount(\$count) [*line 78*]

Function Parameters:

- **int \$count** Count.

Sets count of item.

Sets count of item.

- **Version** 0.0.3
- **Since** 0.0.3
- **Access** public

Class OTS_MapCoords

[*line 22*]

Map position point.

Map position point.

- **Package** POT
- **Version** 0.0.6
- **Since** 0.0.6

Constructor *void function* OTS_MapCoords::__construct(\$x, \$y, \$z) [*line 52*]

Function Parameters:

- *int* **\$x** X.
- *int* **\$y** Y.
- *int* **\$z** Z.

Sets coords for point.

Sets coords for point.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

int function OTS_MapCoords::getX() [*line 77*]

Returns X.

Returns X.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

int function OTS_MapCoords::getY() [*line 87*]

Returns Y.

Returns Y.

- **Version** 0.0.6

- **Since** 0.0.6
- **Access** public

int function OTS_MapCoords::getZ() [*line 97*]

Returns Z.

Returns Z.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

void function OTS_MapCoords::__set_state(\$properties) [*line 67*]

Function Parameters:

- *array* **\$properties** List of object properties.

Magic PHP5 method.

Magic PHP5 method.

Allows object importing from [var_export\(\)](#).

- **Version** 0.0.6
- **Static**
- **Since** 0.0.6
- **Access** public

Class OTS_Monster

[line 22]

Wrapper for monsters files DOMDocument.

Wrapper for monsters files DOMDocument.

Note: as this class extends DOMDocument class and contains exactly respond XML tree you can work on it as on normal DOM tree.

- **Package** POT
- **Version** 0.0.6
- **Since** 0.0.6

int function OTS_Monster::getArmor() [line 268]

Returns monster armor.

Returns monster armor.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

array function OTS_Monster::getAttacks() [line 309]

Returns list of monster attacks.

Returns list of monster attacks.

- **Version** 0.0.6
- **Since** 0.0.6

- **Access** public

int function OTS_Monster::getDefense() [*line 250*]

Returns monster defense rate.

Returns monster defense rate.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

array function OTS_Monster::getDefenses() [*line 286*]

Returns list of special defenses.

Returns list of special defenses.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

int function OTS_Monster::getExperience() [*line 49*]

Returns amount of experience for killing this monster.

Returns amount of experience for killing this monster.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

int|bool function OTS_Monster::getFlag(\$flag) [*line 118*]

Function Parameters:

- *string* **\$flag** Flag.

Returns specified flag value.

Returns specified flag value.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

array function OTS_Monster::getFlags() [*line 97*]

Returns all monster flags (in format flagname => value).

Returns all monster flags (in format flagname => value).

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

int function OTS_Monster::getHealth() [*line 87*]

Returns monster HP.

Returns monster HP.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

array function OTS_Monster::getImmunities() [*line 193*]

Returns all monster immunities.

Returns all monster immunities.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

array function OTS_Monster::getLoot() [*line 163*]

Returns all possible loot.

Returns all possible loot.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

int|bool function OTS_Monster::getManaCost() [*line 69*]

Returns amount of mana required to summon this monster.

Returns amount of mana required to summon this monster.

- **Version** 0.0.6

- **Since** 0.0.6
- **Access** public

string function OTS_Monster::getName() [*line 29*]

Returns monster name.

Returns monster name.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

string function OTS_Monster::getRace() [*line 39*]

Returns monster race.

Returns monster race.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

int function OTS_Monster::getSpeed() [*line 59*]

Returns monster speed.

Returns monster speed.

- **Version** 0.0.6
- **Since** 0.0.6

- **Access** public

array function OTS_Monster::getVoices() [*line 139*]

Returns voices that monster can sound.

Returns voices that monster can sound.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

bool function OTS_Monster::hasImmunity(\$name) [*line 224*]

Function Parameters:

- *string* **\$name** Immunity to check.

Checks if monster has given immunity.

Checks if monster has given immunity.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

Class OTS_OTBMFile

[*line 25*]

OTBM format reader.

OTBM format reader.

- **Package** POT
- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_ACTION_ID

= 4 [*line 42*]

Action ID.

Action ID.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_DEPOT_ID

= 10 [*line 66*]

Depot ID.

Depot ID.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_DESC

= 7 *[line 54]*

Description.

Description.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_DESCRIPTION

= 1 *[line 30]*

Description attribute.

Description attribute.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_EXT_FILE

= 2 *[line 34]*

External file.

External file.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_EXT_HOUSE_FILE

= 13 [*line 78*]

External houses file.

External houses file.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_EXT_SPAWN_FILE

= 11 [*line 70*]

External spawns file.

External spawns file.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_HOUSEDOORID

= 14 [*line 82*]

ID of doors.

ID of doors.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_ITEM

= 9 *[line 62]*

Item.

Item.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_RUNE_CHARGES

= 12 *[line 74]*

Rune changes amount.

Rune changes amount.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_TELE_DEST

= 8 *[line 58]*

Teleport destination.

Teleport destination.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_TEXT

= 6 *[line 50]*

Text.

Text.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_TILE_FLAGS

= 3 *[line 38]*

Tile flags.

Tile flags.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_ATTR_UNIQUE_ID

= 5 *[line 46]*

Unique ID.

Unique ID.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_HOUSED_TILE

= 14 [*line 139*]

Tile of house.

Tile of house.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_ITEM

= 6 [*line 107*]

Item.

Item.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_ITEM_DEF

= 3 [*line 95*]

Item definition.

Item definition.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_MAP_DATA

= 2 [*line 91*]

Map data container.

Map data container.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_MONSTER

= 11 [*line 127*]

Monster.

Monster.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_ROOTV1

= 1 [*line 87*]

Root node.

Root node.

- **Version** 0.0.6

- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_SPAWNS

= 9 [*line 119*]

Spawns container.

Spawns container.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_SPAWN_AREA

= 10 [*line 123*]

Spawn.

Spawn.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_TILE

= 5 [*line 103*]

Single tile.

Single tile.

- **Version** 0.0.6

- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_TILE_AREA

= 4 [*line 99*]

Map tiles fragment.

Map tiles fragment.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_TILE_REF

= 8 [*line 115*]

Tile reference.

Tile reference.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_TILE_SQUARE

= 7 [*line 111*]

Tile.

Tile.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_TOWN

= 13 [*line 135*]

Town.

Town.

- **Version** 0.0.6
- **Since** 0.0.6

OTS_OTBMFile::OTBM_NODE_TOWNS

= 12 [*line 131*]

Towns container.

Towns container.

- **Version** 0.0.6
- **Since** 0.0.6

string function OTS_OTBMFile::getDescription() [*line 350*]

Returns map description.

Returns map description.

- **Version** 0.0.6

- **Since** 0.0.6
- **Access** public

int function OTS_OTBMFile::getHeight() [*line 340*]

Returns map height.

Returns map height.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

int|bool function OTS_OTBMFile::getTownID(\$name) [*line 361*]

Function Parameters:

- *string* **\$name** Town.

Returns town's ID.

Returns town's ID.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

string|bool function OTS_OTBMFile::getTownName(\$id) [*line 372*]

Function Parameters:

- *int* **\$id** Town ID.

Returns name of given town's ID.

Returns name of given town's ID.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

array function OTS_OTBMFile::getTownsList() [*line 389*]

Returns list (id => name) of loaded towns.

Returns list (id => name) of loaded towns.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

OTS_MapCoords|bool function OTS_OTBMFile::getTownTemple(\$id) [*line 400*]

Function Parameters:

- *int* **\$id** Town id.

Returns town's temple position.

Returns town's temple position.

- **Version** 0.0.6

- **Since** 0.0.6
- **Access** public

int function OTS_OTBMFile::getWidth() [*line 330*]

Returns map width.

Returns map width.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

void function OTS_OTBMFile::loadFile(\$file) [*line 215*]

Function Parameters:

- *string* **\$file** Filename.

Loads OTBM file content.

Loads OTBM file content.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

void function OTS_OTBMFile::__set_state(\$properties) [*line 197*]

Function Parameters:

- *array* **\$properties** List of object properties.

Magic PHP5 method.

Magic PHP5 method.

Allows object importing from [var_export\(\)](#).

- **Version** 0.0.6
- **Static**
- **Since** 0.0.6
- **Access** public

void function OTS_OTBMFile::__wakeup() [line 183]

Magic PHP5 method.

Magic PHP5 method.

Allows object unserialisation.

- **Version** 0.0.6
- **Since** 0.0.6
- **Access** public

Class OTS_Player

[line 22]

OTServ character abstraction.

OTServ character abstraction.

- **Package** POT
- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1

void function OTS_Player::ban([\$time = 0]) [line 1673]

Function Parameters:

- *int* **\$time** Time for time until expires (0 - forever).

Bans current player.

Bans current player.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

void function OTS_Player::delete() [line 1727]

Deletes player.

Deletes player.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.

- **Since** 0.0.5
- **Since** 0.0.1
- **Access** public

void function OTS_Player::find(\$name) *[line 84]*

Function Parameters:

- *string* **\$name** Player's name.

Loads player by it's name.

Loads player by it's name.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.2
- **Access** public

OTS_Account function OTS_Player::getAccount() *[line 186]*

Returns account of this player.

Returns account of this player.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1

- **Access** public

int function OTS_Player::getCap() [*line 841*]

Capacity.

Capacity.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

mixed function OTS_Player::getConditions() [*line 980*]

Conditions.

Conditions.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

string function OTS_Player::getCustomField(\$field) [*line 1279*]

Function Parameters:

- *string* **\$field** Field name.

Reads custom field.

Reads custom field.

Reads field by it's name. Can read any field of given record that exists in database.

Note: You should use this method only for fields that are not provided in standard setters/getters (SVN fields). This method runs SQL query each time you call it so it highly overloads used resources.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.3
- **Since** 0.0.1
- **Access** public

OTS_Item|null function OTS_Player::getDepot(\$depot) [*line 1554*]

Function Parameters:

- *int* **\$depot** Depot ID to get items.

Returns items tree from given depot.

Returns items tree from given depot.

Note: OTS_Player class has no information about item types. It returns all items as OTS_Item, unless they have any contained items in database, so empty container will be instanced as OTS_Item object, not OTS_Container.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.

- **Since** 0.0.3
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getDirection() [*line 571*]

Looking direction.

Looking direction.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getExperience() [*line 328*]

Experience points.

Experience points.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

OTS_Group function OTS_Player::getGroup() [*line 215*]

Returns group of this player.

Returns group of this player.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

string function OTS_Player::getGuildNick() [*line 1067*]

Guild nick.

Guild nick.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getHealth() [*line 409*]

Current HP.

Current HP.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.

- **Since** 0.0.1
- **Access** public

int function OTS_Player::getHealthMax() [*line 436*]

Maximum HP.

Maximum HP.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getId() [*line 142*]

Player ID.

Player ID.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getLastIP() [*line 895*]

Last login IP.

Last login IP.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getLastLogin() [*line 868*]

Last login timestamp.

Last login timestamp.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getLevel() [*line 355*]

Experience level.

Experience level.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1

- **Access** public

int function OTS_Player::getLookAddons() [*line 733*]

Addons.

Addons.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getLookBody() [*line 598*]

Body color.

Body color.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getLookFeet() [*line 625*]

Boots color.

Boots color.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getLookHead() [*line 652*]

Hair color.

Hair color.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getLookLegs() [*line 679*]

Legs color.

Legs color.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getLookType() [*line 706*]

Outfit.

Outfit.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getLossExperience() [*line 1190*]

Percentage of experience lost after dead.

Percentage of experience lost after dead.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getLossMana() [*line 1217*]

Percentage of used mana lost after dead.

Percentage of used mana lost after dead.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getLossSkills() [*line 1244*]

Percentage of skills lost after dead.

Percentage of skills lost after dead.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getMagLevel() [*line 382*]

Magic level.

Magic level.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getMana() [*line 463*]

Current mana.

Current mana.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getManaMax() [*line 490*]

Maximum mana.

Maximum mana.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getManaSpent() [*line 517*]

Mana spent.

Mana spent.

- **Version** 0.0.3
- **Version** 0.0.1

- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

string function OTS_Player::getName() [*line 159*]

Player name.

Player name.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getPosX() [*line 760*]

X map coordinate.

X map coordinate.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getPosY() [*line 787*]

Y map coordinate.

Y map coordinate.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getPosZ() [*line 814*]

Z map coordinate.

Z map coordinate.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getPremiumEnd() [*line 245*]

Player's Premium Account expiration timestamp.

Player's Premium Account expiration timestamp.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.

- **Since** 0.0.3
- **Since** 0.0.1
- **Access** public

OTS_GuildRank | null function OTS_Player::getRank() [*line 1111*]

Assigned guild rank.

Assigned guild rank.

- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getRankId() [*line 1095*]

Guild rank ID.

Guild rank ID.

- **Version** 0.0.3
- **Version** 0.0.1
- **Deprecated** 0.0.4 Use getRank().
- **Since** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Access** public

int function OTS_Player::getRedSkullTime() [*line 1007*]

Red skulled time remained.

Red skulled time remained.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getSave() [*line 952*]

Save counter.

Save counter.

- **Version** 0.0.6
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.6
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getSex() [*line 274*]

Player gender.

Player gender.

- **Version** 0.0.3
- **Version** 0.0.1

- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getSkill(\$skill) [*line 1334*]

Function Parameters:

- *int* **\$skill** Skill ID.

Returns player's skill.

Returns player's skill.

- **Version** 0.0.2
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.2
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getSkillTries(\$skill) [*line 1366*]

Function Parameters:

- *int* **\$skill** Skill ID.

Returns player's skill's tries for next level.

Returns player's skill's tries for next level.

- **Version** 0.0.2
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.2
- **Since** 0.0.1
- **Access** public

OTS_Item|null function OTS_Player::getSlot(\$slot) [*line 1419*]
Function Parameters:

- *int* **\$slot** Slot to get items.

Returns items tree from given slot.

Returns items tree from given slot.

Note: OTS_Player class has no information about item types. It returns all items as OTS_Item, unless they have any contained items in database, so empty container will be instanced as OTS_Item object, not OTS_Container.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.3
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getSoul() [*line 544*]

Soul points.

Soul points.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getTownId() [*line 1163*]

Residence town's ID.

Residence town's ID.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

int function OTS_Player::getVocation() [*line 301*]

Player proffesion.

Player proffesion.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

string/bool function OTS_Player::getVocationName() [*line 1749*]

Player profession name.

Player proffesion name.

- **Version** 0.0.6
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.6
- **Since** 0.0.1
- **Access** public

bool function OTS_Player::hasRedSkull() [*line 1034*]

Checks if player has red skull.

Checks if player has red skull.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.1
- **Access** public

bool function OTS_Player::isBanned() [*line 1708*]

Checks if player is banned.

Checks if player is banned.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

bool function OTS_Player::isLoaded() [*line 101*]

Checks if object is loaded.

Checks if object is loaded.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

bool function OTS_Player::isSaveSet() [*line 923*]

Checks if save flag is set.

Checks if save flag is set.

- **Version** 0.0.3
- **Version** 0.0.1
- **Deprecated** 0.0.6 In database save field is now integer.
- **Since** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Access** public

void function OTS_Player::load(\$id) [line 62]

Function Parameters:

- *int* **\$id** Player's ID.

Loads player with given id.

Loads player with given id.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::save() [line 111]

Saves player in database.

Saves player in database.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setAccount(\$account) [line 203]

Function Parameters:

- [*OTS_Account*](#) **\$account** Owning account.

Assigns character to account.

Assigns character to account.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setCap(\$cap) [line 856]

Function Parameters:

- *int* **\$cap** Capacity.

Sets capacity.

Sets capacity.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setConditions(\$conditions) [line 995]

Function Parameters:

- *mixed* **\$conditions** Condition binary field.

Sets conditions.

Sets conditions.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

`void function OTS_Player::setCustomField($field, $value) [line 1309]`

Function Parameters:

- *string* **\$field** Field name.
- *mixed* **\$value** Field value.

Writes custom field.

Writes custom field.

Write field by it's name. Can write any field of given record that exists in database.

Note: You should use this method only for fields that are not provided in standard setters/getters (SVN fields). This method runs SQL query each time you call it so it highly overloads used resources.

Note: Make sure that you pass \$value argument of correct type. This method determinates whether to quote field value. It is safe - it makes you sure that no unproper queries that could lead to SQL injection will be executed, but it can make your code working wrong way. For example: `$object->setCustomField('foo', '1');` will quote 1 as as string ('1') instead of passing it as a integer.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.3
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setDepot(\$depot, [\$item = null], [\$pid = 0], [\$depot_id = 0]) [line 1609]

Function Parameters:

- *int* **\$depot** Depot ID to save items.
- [OTS_Item](#) **\$item** Item (can be a container with content) for given depot. Leave this parameter blank to clear depot.
- *int* **\$pid** Deprecated, not used anymore.
- *int* **\$depot_id** Internal, for further use.

Sets depot content.

Sets depot content.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.3
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setDirection(\$direction) [line 586]

Function Parameters:

- *int* **\$direction** Looking direction.

Sets looking direction.

Sets looking direction.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setExperience(\$experience) [line 343]

Function Parameters:

- *int* **\$experience** Experience points.

Sets experience points.

Sets experience points.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setGroup(\$group) [line 232]

Function Parameters:

- [*OTS_Group*](#) **\$group** Group to be a member.

Assigns character to group.

Assigns character to group.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setGuildNick(\$guildnick) [line 1082]

Function Parameters:

- *string* **\$guildnick** Name.

Sets guild nick.

Sets guild nick.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setHealth(\$health) [line 424]

Function Parameters:

- *int* **\$health** Current HP.

Sets current HP.

Sets current HP.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setHealthMax(\$healthmax) [line 451]

Function Parameters:

- *int* **\$healthmax** Maximum HP.

Sets maximum HP.

Sets maximum HP.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setLastIP(\$lastip) [*line 910*]

Function Parameters:

- *int* **\$lastip** Last login IP.

Sets last login IP.

Sets last login IP.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setLastLogin(\$lastlogin) [*line 883*]

Function Parameters:

- *int* **\$lastlogin** Last login timestamp.

Sets last login timestamp.

Sets last login timestamp.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setLevel(\$level) [line 370]

Function Parameters:

- *int* **\$level** Experience level.

Sets experience level.

Sets experience level.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setLookAddons(\$lookaddons) [line 748]

Function Parameters:

- *int* **\$lookaddons** Addons.

Sets addons.

Sets addons.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setLookBody(\$lookbody) [line 613]

Function Parameters:

- *int* **\$lookbody** Body color.

Sets body color.

Sets body color.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setLookFeet(\$lookfeet) [line 640]

Function Parameters:

- *int* **\$lookfeet** Boots color.

Sets boots color.

Sets boots color.

- **Version** 0.0.1

- **Since** 0.0.1
- **Access** public

void function OTS_Player::setLookHead(\$lookhead) [line 667]

Function Parameters:

- *int* **\$lookhead** Hair color.

Sets hair color.

Sets hair color.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setLookLegs(\$looklegs) [line 694]

Function Parameters:

- *int* **\$looklegs** Legs color.

Sets legs color.

Sets legs color.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setLookType(\$looktype) [line 721]

Function Parameters:

- *int* **\$looktype** Outfit.

Sets outfit.

Sets outfit.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setLossExperience(\$loss_experience) [line 1205]

Function Parameters:

- *int* **\$loss_experience** Percentage of experience lost after dead.

Sets percentage of experience lost after dead.

Sets percentage of experience lost after dead.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setLossMana(\$loss_mana) [line 1232]

Function Parameters:

- *int* **\$loss_mana** Percentage of used mana lost after dead.

Sets percentage of used mana lost after dead.

Sets percentage of used mana lost after dead.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setLossSkills(\$loss_skills) [*line 1259*]

Function Parameters:

- *int* **\$loss_skills** Percentage of skills lost after dead.

Sets percentage of skills lost after dead.

Sets percentage of skills lost after dead.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setMagLevel(\$maglevel) [*line 397*]

Function Parameters:

- *int* **\$maglevel** Magic level.

Sets magic level.

Sets magic level.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setMana(\$mana) [line 478]

Function Parameters:

- *int* **\$mana** Current mana.

Sets current mana.

Sets current mana.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setManaMax(\$manamax) [line 505]

Function Parameters:

- *int* **\$manamax** Maximum mana.

Sets maximum mana.

Sets maximum mana.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setManaSpent(\$manaspent) [line 532]

Function Parameters:

- *int* **\$manaspent** Mana spent.

Sets mana spent.

Sets mana spent.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setName(\$name) [line 174]

Function Parameters:

- *string* **\$name** Name.

Sets players's name.

Sets players's name.

- **Version** 0.0.1
- **Since** 0.0.1

- **Access** public

void function OTS_Player::setPosX(\$posx) [line 775]

Function Parameters:

- *int* **\$posx** X map coordinate.

Sets X map coordinate.

Sets X map coordinate.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setPosY(\$posy) [line 802]

Function Parameters:

- *int* **\$posy** Y map coordinate.

Sets Y map coordinate.

Sets Y map coordinate.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setPosZ(\$posz) [line 829]

Function Parameters:

- *int* **\$posz** Z map coordinate.

Sets Z map coordinate.

Sets Z map coordinate.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setPremiumEnd(\$premend) [*line 262*]

Function Parameters:

- *int* **\$premend** PACC expiration timestamp.

Sets player's Premium Account expiration timestamp.

Sets player's Premium Account expiration timestamp.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.3
- **Access** public

void function OTS_Player::setRank([\$guildRank = null]) [*line 1144*]

Function Parameters:

- [OTS_GuildRank](#)|null **\$guildRank** Guild rank (null to clear assign).

Assigns guild rank.

Assigns guild rank.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setRankId(\$rank_id) [line 1134]

Function Parameters:

- *int* **\$rank_id** Guild rank ID.

Sets guild rank ID.

Sets guild rank ID.

- **Version** 0.0.1
- **Deprecated** 0.0.4 Use setRank().
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setRedSkull() [line 1055]

Sets red skull flag.

Sets red skull flag.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setRedSkullTime(\$redskulltime) [line 1022]

Function Parameters:

- *int* **\$redskulltime** Red skulled time remained.

Sets red skulled time remained.

Sets red skulled time remained.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setSave([\$save = 1]) [line 968]

Function Parameters:

- *int* **\$save** Save counter (this parameter is optional for backward compatibility when save was just a flag).

Sets save counter.

Sets save counter.

- **Version** 0.0.6

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setSex(\$sex) [*line 289*]

Function Parameters:

- *int* **\$sex** Player gender.

Sets player gender.

Sets player gender.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setSkill(\$skill, \$value) [*line 1352*]

Function Parameters:

- *int* **\$skill** Skill ID.
- *int* **\$value** Skill value.

Sets skill value.

Sets skill value.

- **Version** 0.0.2
- **Version** 0.0.1

- **Since** 0.0.1
- **Since** 0.0.2
- **Access** public

void function OTS_Player::setSkillTries(\$skill, \$tries) [line 1384]

Function Parameters:

- *int* **\$skill** Skill ID.
- *int* **\$tries** Skill tries.

Sets skill's tries for next level.

Sets skill's tries for next level.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.2
- **Access** public

void function OTS_Player::setSlot(\$slot, [\$item = null], [\$pid = 0]) [line 1473]

Function Parameters:

- *int* **\$slot** Slot to save items.
- [*OTS_Item*](#) **\$item** Item (can be a container with content) for given slot. Leave this parameter blank to clear slot.
- *int* **\$pid** Deprecated, not used anymore.

Sets slot content.

Sets slot content.

- **Version** 0.0.5
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Since** 0.0.3
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setSoul(\$soul) [line 559]

Function Parameters:

- *int* **\$soul** Soul points.

Sets soul points.

Sets soul points.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setTownId(\$town_id) [line 1178]

Function Parameters:

- *int* **\$town_id** Residence town's ID.

Sets residence town's ID.

Sets residence town's ID.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::setVocation(\$vocation) [line 316]

Function Parameters:

- *int* **\$vocation** Player proffesion.

Sets player proffesion.

Sets player proffesion.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::unban() [line 1690]

Deletes ban from current player.

Deletes ban from current player.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5

- **Access** public

void function OTS_Player::unsetRedSkull() [*line 1047*]

Unsets red skull flag.

Unsets red skull flag.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function OTS_Player::unsetSave() [*line 939*]

Unsets save flag.

Unsets save flag.

- **Version** 0.0.6
- **Version** 0.0.1
- **Deprecated** 0.0.6 In database save field is now integer.
- **Since** 0.0.1
- **Access** public

array function OTS_Player::__sleep() [*line 51*]

Magic PHP5 method.

Magic PHP5 method.

Allows object serialisation.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.4
- **Access** public

Class OTS_Players_List

[line 21]

List of players.
List of players.

- **Package** POT
- **Version** 0.0.5
- **Since** 0.0.1

void function OTS_Players_List::deletePlayer(\$player) *[line 30]*

Function Parameters:

- [*OTS_Player*](#) **\$player** Player to be deleted.

Deletes player.
Deletes player.

- **Version** 0.0.5

- **Deprecated** 0.0.5 Use OTS_Player->delete().
- **Since** 0.0.1
- **Access** public

void function OTS_Players_List::init() [*line 43*]

Sets list parameters.

Sets list parameters.

This method is called at object creation.

- **Version** 0.0.5
- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

Class OTS_SQLField

[*line 20*]

SQL identifier representation.

SQL identifier representation.

- **Package** POT
- **Version** 0.0.5
- **Since** 0.0.5

Constructor *void* function OTS_SQLField::__construct(\$name, [\$table = "]) [*line 41*]

Function Parameters:

- *string* **\$name** Field name.
- *string* **\$table** Table name.

Creates new field representation.

Creates new field representation.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

string function OTS_SQLField::getName() [*line 52*]

Returns field name.

Returns field name.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

string function OTS_SQLField::getTable() [*line 62*]

Returns table name.

Returns table name.

- **Version** 0.0.5

- **Since** 0.0.5
- **Access** public

Class OTS_SQLFilter

[line 20]

SQL WHERE clause object.
SQL WHERE clause object.

- **Package** POT
- **Version** 0.0.5
- **Since** 0.0.5

OTS_SQLFilter::CRITERIUM_AND

= 1 *[line 58]*

AND sibling.
AND sibling.

- **Version** 0.0.5
- **Since** 0.0.5

OTS_SQLFilter::CRITERIUM_OR

= 2 *[line 62]*

OR sibling.
OR sibling.

- **Version** 0.0.5
- **Since** 0.0.5

OTS_SQLFilter::OPERATOR_EQUAL

= 1 [*line 25*]

Equal operator.
Equal operator.

- **Version** 0.0.5
- **Since** 0.0.5

OTS_SQLFilter::OPERATOR_GREATER

= 3 [*line 33*]

Greater-then operator.
Greater-then operator.

- **Version** 0.0.5
- **Since** 0.0.5

OTS_SQLFilter::OPERATOR_LIKE

= 7 [*line 49*]

LIKE operator.
LIKE operator.

- **Version** 0.0.5
- **Since** 0.0.5

OTS_SQLFilter::OPERATOR_LOWER

= 2 [*line 29*]

Lower-then operator.
Lower-then operator.

- **Version** 0.0.5
- **Since** 0.0.5

OTS_SQLFilter::OPERATOR_NEQUAL

= 4 [*line 37*]

Not-equal operator.
Not-equal operator.

- **Version** 0.0.5
- **Since** 0.0.5

OTS_SQLFilter::OPERATOR_NGREATER

= 6 *[line 45]*

Not-greater-then operator.

Not-greater-then operator.

- **Version** 0.0.5
- **Since** 0.0.5

OTS_SQLFilter::OPERATOR_NLIKE

= 8 *[line 53]*

Not-LIKE operator.

Not-LIKE operator.

- **Version** 0.0.5
- **Since** 0.0.5

OTS_SQLFilter::OPERATOR_NLOWER

= 5 *[line 41]*

Not-lower-then operator.

Not-lower-then operator.

- **Version** 0.0.5
- **Since** 0.0.5

void function OTS_SQLFilter::addFilter(\$left, [\$right = null], [\$operator = self::OPERATOR_EQUAL], [\$criterium =

self::CRITERIUM_AND)) [line 238]

Function Parameters:

- *mixed* **\$left** Left side ([OTS_SQLField class](#) object, or literal value).
- *mixed* **\$right** Right side ([OTS_SQLField class](#) object, or literal value).
- *int* **\$operator** Operator used for comparsion (equal check by default).
- *int* **\$criterium** Criterium merging method (AND by default).

General-purpose filter.

General-purpose filter.

Appends new filter in universal way.

To append subset of another filters us `addFilter($OTS_SQLFilterObject)`.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

`void function OTS_SQLFilter::compareField($field, $value, [$operator = self::OPERATOR_EQUAL], [$criterium = self::CRITERIUM_AND])` [line 251]

Function Parameters:

- *string* **\$field** Field name.
- *mixed* **\$value** Literal value.
- *int* **\$operator** Operator used for comparsion (equal by default).
- *int* **\$criterium** Criterium merging method (AND by default).

Compares field with a literal value.

Compares field with a literal value.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

array function OTS_SQLFilter::getTables() [*line 263*]

Returns list of all tables used by filter.

Returns list of all tables used by filter.
This is required for FROM clause.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

array function OTS_SQLFilter::__sleep() [*line 79*]

Magic PHP5 method.

Magic PHP5 method.
Allows object serialisation.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

string function OTS_SQLFilter::__toString() [*line 92*]

Returns string representation of WHERE clause.

Returns string representation of WHERE clause.

Returned string can be easily inserted into SQL query.

- **Version** 0.0.5
- **Since** 0.0.5
- **Access** public

Class POT

[line 31]

Main POT class.
Main POT class.

- **Package** POT
- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1

POT::BAN_ACCOUNT

= 3 *[line 262]*

Account ban.
Account ban.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5

POT::BAN_IP

= 1 *[line 248]*

IP ban.

IP ban.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5

POT::BAN_PLAYER

= 2 *[line 255]*

Player ban.

Player ban.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5

POT::DB_MYSQL

= 1 *[line 36]*

MySQL driver.
MySQL driver.

- **Version** 0.0.1
- **Since** 0.0.1

POT::DB_ODBC

= 4 *[line 54]*

ODBC driver.
ODBC driver.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.4

POT::DB_PGSQL

= 3 *[line 47]*

PostgreSQL driver.
PostgreSQL driver.

- **Version** 0.0.4

- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.4

POT::DB_SQLITE

= 2 [*line 40*]

SQLite driver.
SQLite driver.

- **Version** 0.0.1
- **Since** 0.0.1

POT::DEPOT_SID_FIRST

= 100 [*line 240*]

First depot item sid.
First depot item sid.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.4

POT::DIRECTION_EAST

= 1 [*line 103*]

East.

East.

- **Version** 0.0.1
- **Since** 0.0.1

POT::DIRECTION_NORTH

= 0 [*line 99*]

North.
North.

- **Version** 0.0.1
- **Since** 0.0.1

POT::DIRECTION_SOUTH

= 2 [*line 107*]

South.
South.

- **Version** 0.0.1
- **Since** 0.0.1

POT::DIRECTION_WEST

= 3 [*line 111*]

West.
West.

- **Version** 0.0.1
- **Since** 0.0.1

POT::ORDER_ASC

= 1 [*line 270*]

Ascencind sorting order.
Ascencind sorting order.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5

POT::ORDER_DESC

= 2 [*line 277*]

Descending sorting order.
Descending sorting order.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1

- **Since** 0.0.5

POT::SEX_FEMALE

= 0 [*line 59*]

Female gender.
Female gender.

- **Version** 0.0.1
- **Since** 0.0.1

POT::SEX_MALE

= 1 [*line 63*]

Male gender.
Male gender.

- **Version** 0.0.1
- **Since** 0.0.1

POT::SKILL_AXE

= 3 [*line 140*]

Axe fighting.
Axe fighting.

- **Version** 0.0.2

- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.2

POT::SKILL_CLUB

= 1 [*line 126*]

Club fighting.
Club fighting.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.2

POT::SKILL_DISTANCE

= 4 [*line 147*]

Distance fighting.
Distance fighting.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.2

POT::SKILL_FISHING

= 6 *[line 161]*

Fishing.

Fishing.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.2

POT::SKILL_FIST

= 0 *[line 119]*

Fist fighting.

Fist fighting.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.2

POT::SKILL_SHIELDING

= 5 *[line 154]*

Shielding.

Shielding.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.2

POT::SKILL_SWORD

= 2 [*line 133*]

Sword fighting.

Sword fighting.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.2

POT::SLOT_AMMO

= 10 [*line 232*]

Ammunition slot.

Ammunition slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.3

POT::SLOT_ARMOR

= 4 *[line 190]*

Armor slot.

Armor slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.3

POT::SLOT_BACKPACK

= 3 *[line 183]*

Backpack slot.

Backpack slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.3

POT::SLOT_FEET

= 8 *[line 218]*

Boots slot.

Boots slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.3

POT::SLOT_HEAD

= 1 [*line 169*]

Head slot.

Head slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.3

POT::SLOT_LEFT

= 6 [*line 204*]

Left hand slot.

Left hand slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.3

POT::SLOT_LEGS

= 7 [*line 211*]

Legs slot.

Legs slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.3

POT::SLOT_NECKLACE

= 2 [*line 176*]

Necklace slot.

Necklace slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.3

POT::SLOT_RIGHT

= 5 [*line 197*]

Right hand slot.

Right hand slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.3

POT::SLOT_RING

= 9 [*line 225*]

Ring slot.

Ring slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.3

POT::VOCATION_DRUID

= 2 [*line 82*]

Druid.

Druid.

- **Version** 0.0.1
- **Deprecated** 0.0.5 Vocations are now loaded dynamicly from vocations.xml file.
- **Since** 0.0.1

POT::VOCATION_KNIGHT

= 4 *[line 94]*

Knight.
Knight.

- **Version** 0.0.1
- **Deprecated** 0.0.5 Vocations are now loaded dynamicly from vocations.xml file.
- **Since** 0.0.1

POT::VOCATION_NONE

= 0 *[line 70]*

None vocation.
None vocation.

- **Version** 0.0.1
- **Deprecated** 0.0.5 Vocations are now loaded dynamicly from vocations.xml file.
- **Since** 0.0.1

POT::VOCATION_PALADIN

= 3 *[line 88]*

Paladin.
Paladin.

- **Version** 0.0.1
- **Deprecated** 0.0.5 Vocations are now loaded dynamicly from vocations.xml file.
- **Since** 0.0.1

POT::VOCATION_SORCERER

= 1 [*line 76*]

Sorcerer.

Sorcerer.

- **Version** 0.0.1
- **Deprecated** 0.0.5 Vocations are now loaded dynamicly from vocations.xml file.
- **Since** 0.0.1

*void function POT::banIP(\$ip, [\$mask = '255.255.255.255'], [\$time = 0]) [*line 618*]*

Function Parameters:

- *string* **\$ip** IP to ban.
- *string* **\$mask** Mask for ban (by default bans only given IP).
- *int* **\$time** Time for time until expires (0 - forever).

Bans given IP number.

Bans given IP number.

Adds IP/mask ban. You can call this function with only one parameter to ban only given IP address without expiration.

- **Version** 0.0.5

- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

`void function POT::connect($driver, $params) [line 400]`

connect.php

```

1  <?php
2
3  /**
4   * @ignore
5   * @package examples
6   * @author Wrzasq <wrzasq@gmail.com>
7   * @copyright 2007 (C) by Wrzasq
8   * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9   */
10
11 // includes POT main file
12 include('../classes/OTS.php');
13
14 // you can easily store such structure in config.php
15 $config = array(
16     'driver' => POT::DB_MYSQL,
17     'prefix' => '',
18     'host' => 'localhost',
19     'user' => 'wrzasq',
20     'password' => '',
21     'database' => 'otserve'
22 );
23
24 // connects to database
25 $ots = POT::getInstance();
26 $ots->connect(null, $config);
27 // could be: $ots->connect(POT::DB_MYSQL, $config);
28
29 ?>

```

Function Parameters:

- `int|null $driver` Database driver type.
- `array $params` Connection info.

Connects to database.

Connects to database.

Creates OTServ database connection object.

First parameter is one of database driver constants values. Currently MySQL, SQLite, PostgreSQL and ODBC drivers are supported.

This parameter can be null, then you have to specify *'driver'* parameter.

Such way is comfortable to store entire database configuration in one array and possibly runtime evaluation and/or configuration file saving.

For parameters list see driver documentation. Common parameters for all drivers are:

- *driver* - optional, specifies driver, applies when *\$driver* method parameter is *null*
- *prefix* - optional, prefix for database tables, use if you have more than one OTServ installed on one database.

- **Version** 0.0.4
- **Version** 0.0.1
- **Throws** Exception When driver is not supported.
- **Since** 0.0.1
- **Access** public
- **Example**

OTS_SQLFilter function POT::createFilter() [line 710]

Creates lists filter.

Creates lists filter.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

IOTS_DAO function POT::createObject(\$class) [line 451]

Function Parameters:

- *string* **\$class** Class name.

Creates OTServ DAO class instance.
Creates OTServ DAO class instance.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

PDO function POT::getDBHandle() [line 526]

Returns database connection handle.

Returns database connection handle.

At all you shouldn't use this method and work with database using POT classes, but it may be sometime necessary to use direct database access (mainly until POT won't provide many important features).

It is also important as serialised objects after unserialisation needs to be re-initialised with database connection.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.4
- **Access** public

POT function POT::getInstance() [line 284]

Singleton.

Singleton.

- **Version** 0.0.1
- **Static**
- **Since** 0.0.1
- **Access** public

OTS_Monster function POT::getMonster(\$name) [*line 781*]

Function Parameters:

- *string* **\$name** Monster name.

Returns loaded data of given monster.

Returns loaded data of given monster.

- **Version** 0.0.6
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.6
- **Access** public

array function POT::getMonstersList() [*line 768*]

Returns list of loaded monsters.

Returns list of loaded monsters.

- **Version** 0.0.6
- **Version** 0.0.1
- **Since** 0.0.1

- **Since** 0.0.6
- **Access** public

int|bool function POT::getVocationID(\$name) [*line 570*]

Function Parameters:

- *string* **\$name** Vocation.

Returns vocation's ID.

Returns vocation's ID.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

string|bool function POT::getVocationName(\$id) [*line 583*]

Function Parameters:

- *int* **\$id** Vocation ID.

Returns name of given vocation's ID.

Returns name of given vocation's ID.

- **Version** 0.0.5
- **Version** 0.0.1

- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

array function POT::getVocationsList() [*line 602*]

Returns list (id => name) of loaded vocations.

Returns list (id => name) of loaded vocations.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

bool function POT::isIPBanned(\$ip) [*line 686*]

Function Parameters:

- *string* **\$ip** IP to ban.

Checks if given IP is banned.

Checks if given IP is banned.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5

- **Access** public

void function POT::loadClass(\$class) [line 359]

Function Parameters:

- *string* **\$class** Class name.

Loads POT class file.

Loads POT class file.

Runtime class loading on demand - usefull for `__autoload()` function.

Note: Since 0.0.2 version this function is suitable for `spl_autoload_register()`.

Note: Since 0.0.3 version this function handles also exceptions.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public

void function POT::loadMonsters(\$path) [line 740]

Function Parameters:

- *string* **\$path** Monsters directory.

Loads monsters mapping file.

Loads monsters mapping file.

- **Version** 0.0.6
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.6
- **Access** public

void function POT::loadVocations(\$file) [*line 549*]

Function Parameters:

- *string* **\$file** vocations.xml file location.

Loads vocations list.

Loads vocations list.

Loads vocations list from given file.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

OTS_InfoRespond|bool function POT::serverStatus(\$server, \$port) [*line 469*]

example

```

1  <?php
2
3  /**
4   * @ignore
5   * @package examples
6   * @author Wrzasq <wrzasq@gmail.com>
7   * @copyright 2007 (C) by Wrzasq
8   * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9   */
10
11 // to not repeat all that stuff
12 include('quickstart.php');
13
14 // server and port

```

```

15  $server = '127.0.0.1';
16  $port = 7171;
17
18  // queries server of status info
19  $status = $ots-> serverStatus($server, $port);
20
21  // offline
22  if(!$status)
23  {
24      echo 'Server ', $server, ' is offline.', "\n" ;
25  }
26  // displays various info
27  else
28  {
29      echo 'Server name: ', $status-> getName(), "\n" ;
30      echo 'Server owner: ', $status-> getOwner(), "\n" ;
31      echo 'Players online: ', $status-> getOnlinePlayers(), "\n" ;
32      echo 'Maximum allowed number of players: ', $status-> getMaxPlayers(), "\n" ;
33      echo 'Required client version: ', $status-> getClientVersion(), "\n" ;
34      echo 'All monsters: ', $status-> getMonstersCount(), "\n" ;
35      echo 'Server message: ', $status-> getMOTD(), "\n" ;
36  }
37
38  ?>

```

Function Parameters:

- *string* **\$server** Server IP/domain.
- *int* **\$port** OTServ port.

Queries server status.

Queries server status.

Sends 'info' packet to OTS server and return output.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.2
- **Access** public
- **Example**

void function POT::setPOTPath(\$path) [line 315]

fakeroot.php

```

1  <?php
2
3  /**

```

```

4      * @ignore
5      * @package examples
6      * @author Wrzasq <wrzasq@gmail.com>
7      * @copyright 2007 (C) by Wrzasq
8      * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
9      */
10
11     // this is the way you should work with POT if you moved main OTS.php file outside POT's directory
12     include('path/to/OTS.php');
13
14     // dont use 'new POT()'!!!
15     $ots = POT::getInstance();
16     $ots-> setPOTPath('../classes/');
17
18     /*
19         here comes your stuff...
20     */
21
22     ?>

```

Function Parameters:

- *string* **\$path** POT files path.

Set POT directory.

Set POT directory.

Use this method if you keep your POT package in different directory then this file.

- **Version** 0.0.1
- **Since** 0.0.1
- **Access** public
- **Example**

void function POT::unbanIP(\$ip, [\$mask = '255.255.255.255']) [*line 653*]

Function Parameters:

- *string* **\$ip** IP to ban.
- *string* **\$mask** Mask for ban (by default 255.255.255.255).

Deletes ban from given IP number.

Deletes ban from given IP number.

Removes given IP/mask ban.

- **Version** 0.0.5
- **Version** 0.0.1
- **Since** 0.0.1
- **Since** 0.0.5
- **Access** public

compat.php

POT compatibility assurance package.

POT compatibility assurance package.

This package makes you sure that POT scripts won't cause FATAL errors on PHP older PHP 5.x versions. However remember that some PHP features won't be enabled with it. For example if you have PHP 5.0.x, this package will define Countable interface for you so PHP will know it, but it won't allow you to use count(\$countableObject) structure.

- **Package** POT
- **Sub-Package** compat
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.2
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.2
- **License** [GNU Lesser General Public License, Version 3](#)

Appendices

Appendix A - Class Trees

Package POT

E_OTS_ErrorCode

- [E_OTS_ErrorCode](#)
 - [E_OTS_FileLoaderError](#)
 - [E_OTS_OTBMEError](#)

E_OTS_NoDriver

- Exception
 - [E_OTS_NoDriver](#)

E_OTS_NotLoaded

- Exception
 - [E_OTS_NotLoaded](#)

E_OTS_OutOfBuffer

- Exception
 - [E_OTS_OutOfBuffer](#)

IOTS_DAO

- [IOTS_DAO](#)

IOTS_DB

- [IOTS_DB](#)

IOTS_FileCache

- [IOTS_FileCache](#)

IOTS_GuildAction

- [IOTS_GuildAction](#)

OTS_Base_DAO

- [OTS_Base_DAO](#)
 - [OTS_Account](#)
 - [OTS_Group](#)
 - [OTS_Guild](#)
 - [OTS_GuildRank](#)
 - [OTS_Player](#)
 - [OTS_SQLFilter](#)

OTS_Base_List

- [OTS_Base_List](#)
 - [OTS_Accounts_List](#)
 - [OTS_Groups_List](#)
 - [OTS_GuildRanks_List](#)
 - [OTS_Guilds_List](#)
 - [OTS_Players_List](#)

OTS_DB_MySQL

- PDO
 - [OTS_DB_MySQL](#)

OTS_DB_ODBC

- PDO
 - [OTS_DB_ODBC](#)

OTS_DB_PostgreSQL

- PDO
 - [OTS_DB_PostgreSQL](#)

OTS_DB_SQLite

- PDO
 - [OTS_DB_SQLite](#)

OTS_FileLoader

- [OTS_FileLoader](#)
 - [OTS_OTBMFile](#)

OTS_FileNode

- [OTS_FileNode](#)

OTS_InfoRespond

- DOMDocument
 - [OTS_InfoRespond](#)

OTS_Item

- [OTS_Item](#)
 - [OTS_Container](#)

OTS_MapCoords

- [OTS_MapCoords](#)

OTS_Monster

- DOMDocument
 - [OTS_Monster](#)

OTS_SQLField

- [OTS_SQLField](#)

POT

- [POT](#)

Appendix B - README/CHANGELOG/INSTALL

LICENSE

GNU LESSER GENERAL PUBLIC LICENSE Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates
the terms and conditions of version 3 of the GNU General Public
License, supplemented by the additional permissions listed below.

0. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser
General Public License, and the "GNU GPL" refers to version 3 of the GNU
General Public License.

"The Library" refers to a covered work governed by this License,
other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided
by the Library, but which is not otherwise based on the Library.
Defining a subclass of a class defined by the Library is deemed a mode
of using an interface provided by the Library.

A "Combined Work" is a work produced by combining or linking an
Application with the Library. The particular version of the Library
with which the Combined Work was made is also called the "Linked
Version".

The "Minimal Corresponding Source" for a Combined Work means the
Corresponding Source for the Combined Work, excluding any source code
for portions of the Combined Work that, considered in isolation, are
based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the
object code and/or source code for the Application, including any data
and utility programs needed for reproducing the Combined Work from the
Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License
without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a
facility refers to a function or data to be supplied by an Application
that uses the facility (other than as an argument passed when the

facility is invoked), then you may convey a copy of the modified version:

- a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or
- b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

- a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

- a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the Combined Work with a copy of the GNU GPL and this license document.
- c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.
- d) Do one of the following:
 - 0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.
 - 1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time

a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.

e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.
- b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

CHANGELOG

[0.0.6]

- * Updated for last database changes. <wrzasq>
- * Increased PHP 5.0 compatibility. <wrzasq>
- * Added generic binary formats reader with cache drivers. <wrzasq>
- * Added OTBM files basic support. <wrzasq>
- * Added monsters support. <wrzasq>
- * Added OTS_Player::getVocationName() method. <wrzasq>

[0.0.5]

- * Added support for vocations.xml file. <wrzasq>
- * Added support for bans. <wrzasq>
- * Added sorting and filtering for lists. <wrzasq>
- * Code grouped into base classes. <wrzasq>
- * Some code optimisation. <wrzasq>
- * Fixed typos. <wrzasq>

[0.0.4]

- * Added guild system support (guilds, ranks, invitations and requests drivers mechanisms). <wrzasq>
- * Added account group support. <wrzasq>
- * Added support for depot_id field (it is reserved in OTServ for further use). <wrzasq>
- * Added PostgreSQL and ODBC drivers. <wrzasq>
- * Added __sleep() and __wakeup() methods to allow POT objects to be stored in sessions. <wrzasq>
- * Added __clone() methods to allow save ID-losing cloning of POT objects. <wrzasq>
- * Added __set_state() methods. <wrzasq>
- * Updated players table structure. <wrzasq>
- * Dropped REGEXP operator bindings - not used anywhere. <wrzasq>
- * Fixed items loading and saving. <wrzasq>
- * Fixed typos. <wrzasq>

[0.0.3]

- * Added custom fields support. <wrzasq>
- * Added items and depots support. <wrzasq>
- * Added support for players PACC timestamps. <wrzasq>
- * Fixed loading skills. <wrzasq>
- * Replaced E_USER_* with exceptions. <wrzasq>
- * Uses fetchAll() in loops to prevent MySQL buffering problems. <wrzasq>
- * Restricted access to POT class constructor to make sure it won't be instantiated directly. <wrzasq>

[0.0.2]

- * Added "compat" library for POT. <wrzasq>
- * Added skills support in OTS_Player class. <wrzasq>
- * Added 'info' serverStatus() method and respond handler for server status protocol. <wrzasq>
- * Fixed 'redskulltime' field name in OTS_Player. <wrzasq>
- * Fixed 'password' parameter for DB_MYSQL driver. <wrzasq>
- * Added find() to OTS_Account class to load accounts by their's e-mail addresses. <wrzasq>
- * POT class now automatically binds own __autoload() handler with spl_autoload_register(). <wrzasq>

[0.0.1]

- * Initial release. <wrzasq>

RULES

Zero rule: We use Unicode (UTF-8).

Of course we should handle input encoding respectively, but output and internal data/code are all written in UTF-8.

I. Coding rules to be followed:

[1] Never ever use global!

It's just the worst thing you can do in PHP scripts.

[2] Avoid using define - use class constants.

To group code better, to allow classes __autoload() handling.

[3] Use !isset() instead of is_null().

That has exactly same effect and we should follow the most simple methods. Just to make code cleaner (however remember that isset() is a PHP language structure and has its limitations!).

[4] Don't use functions - use class methods (except Compat package).

This will allow __autoload() handling for all routines as they will be members of classes.

[5] Use 4 spaces as tabulation.

Tabulator character can be differently displayed and generally four spaces makes code more readable.

[6] Always use brackets for blocks and leave them in separated lines in same nesting level that block instruction:

```
if(condition)
{
    for($i = 0; $i < $j; $i++)
    {
        statement;
    }
}
```

[7] Use single quotes instead of double ones.

' are, in standard way, faster than " and it keeps code cleaner if you simply concat everything rather than inserting something like placeholders into string.

[8] Use spaces between parenthesis and operators (except object member accessing operator):

```
$foo = $lol . $rotfl;
$foo .= $bar;
$obj = new Class( substr( str_replace( implode('.', $array), ',', '.'), 2) );
echo $obj->field;
echo $obj->method( rand() );
echo $obj->method($value);
```

[9] Use <?php opening tag.

It is the most reliable and standard way for starting PHP code.

[10] Use isset(array[offset]) instead of array_key_exists().

It saves alot of resources (relatively).

II. Versioning:

- * Major - Major toolkit milestone.
- * Minor - Toolkit noticeable step.
- * Release - Independent package version.

Package version shouldn't be increased if package itself wasn't changed - but it can't increase it's major/minor numbers over current toolkit release.

III. File naming:

[1] Use lowercase names for directories.

[2] Use fiels and directories in code in case-sensitive way.

Remember that probably this code will be mostly run on non-Windows platforms.

[3] Use existing directories structure.

Put classes into classes directory, tutorials into tutorials directory etc.

README

POT (PHP OTServ Toolkit) is a PHP toolkit for scripts that work with OTServ database.

===== About =====

This toolkit provides a way for PHP programmers that don't know SQL langauge to work with OTServ database.

For installation help check INSTALL file.

For usage tutorial/API documentation check <http://www.otserv-aac.info/pot/> or documentation.pdf file.

===== Contact =====

In case of any contact needed, please use following e-mail address: wrzasq@gmail.com.

===== Files =====

classes/ - POT class files.

examples/ - example files for learning.

tutorials/ - phpDocumentor directory.

CHANGELOG - changes history.

INSTALL - installation tutorial.

LICENSE - POT license (GNU LGPL v3), if you don't accept it - don't use any of those scripts.

NEWS - changes in current release.
README - this readme file.
RULES - rules to be followed during developing contributed code.
Makefile - make input, for documentation generation.
documentation.pdf - phpDocumentor-generator documentation in PDF format.
compat.php - Compatibility assurance library.
test.php - phpUnit test suite.

===== Makefile =====

Makefile contains some targets for make that can help in development. Makefile requires following command-line commands:

php: PHP CLI interface.
phpdoc: phpDocumentor.
phpunit: PHPUnit testing framework.

Possible targets:

all: default one, runs all other targets (in order: clean, check, documentation, pdf, online, test, package).
clean: deletes documentation.
check: checks syntax of all PHP files.
documentation: generates HTML documentation.
pdf: generates PDF documentation.
online: OTServ-AAC website documentation template used.
test: runs test suite.
package: creates pot.tar.gz file for distribution purposes.

For more readable output of phpUnit test run:
php test.php

===== Credits =====

* Wrzasq <wrzasq@gmail.com> - project initiator, main developer.

For more info see AUTHORS file in OTServ tree.

INSTALL

POT is a toolkit which means you don't literally install it. You copy it's files and write code for it. All source files are located in classes/ subdirectory. Copy them to your script directory.

You can put main file - OTS.php in different directory then other files.

For information about how to include POT in your code see the documentation.

NEWS

What's new in 0.0.6 version?

- * Updated for last database changes.

There were minor database changes like save counter which this POT version of course supports.

- * Increased PHP 5.0 compatibility.

Dropper array type hints. POT should now run on PHP 5.0 easily.

- * Added generic binary formats reader with cache drivers.

Universal class for reading OTServ binary file formats (OTB-based).

- * Added OTBM files basic support.

Support for OTBM map files. Currently it saves only spawns points, but as it contains entire map info you can easily add map tiles reading for example. It also supports cache mechanism, so you it won't slow down your scripts if you have big OTBM files.

- * Added monsters support.

You can now load list of monsters and easily get info about particular monsters using OTS_Monster wrapper for DOMDocument.

- * Added OTS_Player::getVocationName() method.

Wrapper for POT::getVocationName() with current player vocation ID.

Appendix D - Todo List

In Package POT

In [OTS.php](#)

- 0.0.7: Spells.
- 0.0.8: Items list (items.xml + items.otb -> cache).
- 0.1.0: Get rid of POT::getInstance()->create*() calls - use POT::getInstance()->getDBHandle() in constructors.
- 0.1.0: Implement __get()/__set()/__call()/__toString(); ArrayAccess interface.
- 1.0.0: Complete PHPUnit test.
- 1.0.0: Main POT class as database instance.
- 1.0.0: More detailed documentation and tutorials, also update examples and tutorials.

In [OTS_OTBMFile.php](#)

- 0.1.0: Houses support.
- 1.0.0: Complete OTBM support: link tiles with items, spawns and houses.
- 1.0.0: Spawns support.

Index

A

About OTServ AAC scripts	21
Account number hack	18

C

constructor OTS_Item::__construct()	176
Creates item of given ID.	
constructor OTS_DB_SQLite::__construct()	118
Creates database connection.	
constructor OTS_MapCoords::__construct()	179
Sets coords for point.	
constructor OTS_SQLField::__construct()	249
Creates new field representation.	
CHANGELOG	293
compat.php	283
POT compatibility assurance package.	
constructor OTS_DB_PostgreSQL::__construct()	115
Creates database connection.	
constructor OTS_DB_ODBC::__construct()	111
Creates database connection.	
constructor IOTS_DB::__construct()	70
Connection parameters.	
constructor IOTS_DAO::__construct()	69
DAO objects must be initialized with a database.	
constructor IOTS_GuildAction::__construct()	76
Objects are initialized with a guild that they are assigned to.	
constructor OTS_Base_DAO::__construct()	94
Sets database connection handler.	
constructor OTS_DB_MySQL::__construct()	108
Creates database connection.	
constructor OTS_Base_List::__construct()	97
Sets database connection handler.	
constructor E_OTS_ErrorCode::__construct()	63
Sets error code.	

D

DAO objects	9
-----------------------------	---

E

E_OTS_NoDriver	67
<i>Occurs when code attempts to execute driven action that has no assigned driver to handle it.</i>	
E_OTS_FileLoaderError::ERROR_TELL_ERROR	66
<i>Failed to read position in file.</i>	
E_OTS_FileLoaderError::ERROR_SEEK_ERROR	66
<i>Failed to seek in given position in file.</i>	
E_OTS_FileLoaderError::ERROR_NOT_OPEN	65
<i>Attempted to execute operation on not opened file.</i>	
E_OTS_NotLoaded	67
<i>Occurs when code attempts to access property of not loaded object.</i>	
E_OTS_OTBMErrors	68
<i>OTBM map loading error.</i>	
E_OTS_OutOfBuffer	69
<i>Occurs when properties stream has ended and there is still read attempt.</i>	
E_OTS_OTBMErrors::LOADMAPERROR_UNKNOWNNODETYPE	68
<i>Unknown node type.</i>	
E_OTS_OTBMErrors::LOADMAPERROR_OUTDATEDHEADER	68
<i>Unsupported file version.</i>	
E_OTS_FileLoaderError::ERROR_INVALID_FORMAT	65
<i>File corrupted.</i>	
E_OTS_FileLoaderError::ERROR_INVALID_FILE_VERSION	65
<i>Unsupported file version.</i>	
E_OTS_OTBMErrors.php	29
<i>Code in this file bases on oryiginal OTServ OTBM format loading C++ code (iomapotbm.</i>	
E_OTS_NotLoaded.php	28
E_OTS_NoDriver.php	27
E_OTS_FileLoaderError.php	26
<i>Code in this file bases on oryiginal OTServ binary format loading C++ code (fileloader.</i>	
E_OTS_OutOfBuffer.php	30
<i>Code in this file bases on oryiginal OTServ binary format loading C++ code (fileloader.</i>	
E_OTS_ErrorCode	63
<i>Generic exception class for error codes.</i>	
E_OTS_FileLoaderError::ERROR_EOF	64
<i>Unexpected end of file.</i>	
E_OTS_FileLoaderError::ERROR_CAN_NOT_OPEN	64
<i>Could not open file.</i>	
E_OTS_FileLoaderError	64
<i>Error during reading OTServ binary file.</i>	
E_OTS_ErrorCode.php	25

G

Guild action drivers	15
Guilds	13

I

IOTS_GuildAction	75
<i>Guild action interface.</i>	
IOTS_FileCache::writeCache()	74
<i>Writes node cache.</i>	

IOTS_FileCache::readCache()	74
<i>Returns cache.</i>	
IOTS_FileCache	74
<i>This interface describe binary files cache control drivers.</i>	
IOTS_GuildAction::addRequest()	76
<i>Adds new request.</i>	
IOTS_GuildAction::deleteRequest()	77
<i>Deletes request.</i>	
INSTALL	297
IOTS_GuildAction::submitRequest()	77
<i>Finalizes request.</i>	
IOTS_GuildAction::listRequests()	77
<i>List of saved pending actions.</i>	
IOTS_DB::tableName()	73
<i>Query-quoted table name.</i>	
IOTS_DB::SQLquote()	73
<i>Query-quoted string value.</i>	
IOTS_DAO	69
<i>OTServ database object.</i>	
IOTS_GuildAction.php	34
IOTS_FileCache.php	33
IOTS_DB.php	32
IOTS_DB	70
<i>OTServ database handler interface.</i>	
IOTS_DB::fieldName()	71
<i>Query-quoted field name.</i>	
IOTS_DB::SQLquery()	72
<i>Evaluates query.</i>	
IOTS_DB::limit()	72
<i>LIMIT/OFFSET clause for queries.</i>	
IOTS_DB::lastInsertId()	71
<i>ID of last created record.</i>	
IOTS_DAO.php	31

L

LICENSE	291
-------------------------	-----

N

NEWS	297
----------------------	-----

O

OTS_OTBMFile::OTBM_NODE_ITEM_DEF	193
<i>Item definition.</i>	
OTS_OTBMFile::OTBM_NODE_MAP_DATA	194
<i>Map data container.</i>	
OTS_OTBMFile::OTBM_NODE_ITEM	193
<i>Item.</i>	

OTS OTBMFile::OTBM_NODE_HOSETILE	193
<i>Tile of house.</i>	
OTS OTBMFile::OTBM_ATTR_UNIQUE_ID	192
<i>Unique ID.</i>	
OTS OTBMFile::OTBM_NODE_MONSTER	194
<i>Monster.</i>	
OTS OTBMFile::OTBM_NODE_ROOTV1	194
<i>Root node.</i>	
OTS OTBMFile::OTBM_NODE_TILE_AREA	196
<i>Map tiles fragment.</i>	
OTS OTBMFile::OTBM_NODE_TILE_REF	196
<i>Tile reference.</i>	
OTS OTBMFile::OTBM_NODE_TILE	195
<i>Single tile.</i>	
OTS OTBMFile::OTBM_NODE_SPAWN_AREA	195
<i>Spawn.</i>	
OTS OTBMFile::OTBM_NODE_SPAWNS	195
<i>Spawns container.</i>	
OTS OTBMFile::OTBM_ATTR_TILE_FLAGS	192
<i>Tile flags.</i>	
OTS OTBMFile::OTBM_ATTR_TEXT	192
<i>Text.</i>	
OTS OTBMFile::OTBM_ATTR_DESC	189
<i>Description.</i>	
OTS OTBMFile::OTBM_ATTR_DESCRIPTION	189
<i>Description attribute.</i>	
OTS OTBMFile::OTBM_ATTR_DEPOT_ID	188
<i>Depot ID.</i>	
OTS OTBMFile::OTBM_ATTR_ACTION_ID	188
<i>Action ID.</i>	
OTS OTBMFile	187
<i>OTBM format reader.</i>	
OTS OTBMFile::OTBM_ATTR_EXT_FILE	189
<i>External file.</i>	
OTS OTBMFile::OTBM_ATTR_EXT_HOUSE_FILE	190
<i>External houses file.</i>	
OTS OTBMFile::OTBM_ATTR_RUNE_CHARGES	191
<i>Rune charges amount.</i>	
OTS OTBMFile::OTBM_ATTR_TELE_DEST	191
<i>Teleport destination.</i>	
OTS OTBMFile::OTBM_ATTR_ITEM	191
<i>Item.</i>	
OTS OTBMFile::OTBM_ATTR_HOUSEDOORID	190
<i>ID of doors.</i>	
OTS OTBMFile::OTBM_ATTR_EXT_SPAWN_FILE	190
<i>External spawns file.</i>	
OTS OTBMFile::OTBM_NODE_TILE_SQUARE	196
<i>Tile.</i>	
OTS OTBMFile::OTBM_NODE_TOWN	197
<i>Town.</i>	
OTS Player::getConditions()	204
<i>Conditions.</i>	
OTS Player::getCustomField()	204

<i>Reads custom field.</i>	
OTS_Player::getCap()	204
<i>Capacity.</i>	
OTS_Player::getAccount()	203
<i>Returns account of this player.</i>	
OTS_Player::find()	203
<i>Loads player by it's name.</i>	
OTS_Player::getDepot()	205
<i>Returns items tree from given depot.</i>	
OTS_Player::getDirection()	206
<i>Looking direction.</i>	
OTS_Player::getHealth()	207
<i>Current HP.</i>	
OTS_Player::getHealthMax()	208
<i>Maximum HP.</i>	
OTS_Player::getGuildNick()	207
<i>Guild nick.</i>	
OTS_Player::getGroup()	206
<i>Returns group of this player.</i>	
OTS_Player::getExperience()	206
<i>Experience points.</i>	
OTS_Player::delete()	202
<i>Deletes player.</i>	
OTS_Player::ban()	202
<i>Bans current player.</i>	
OTS_OTBMFile::getTownID()	198
<i>Returns town's ID.</i>	
OTS_OTBMFile::getTownName()	198
<i>Returns name of given town's ID.</i>	
OTS_OTBMFile::getHeight()	198
<i>Returns map height.</i>	
OTS_OTBMFile::getDescription()	197
<i>Returns map description.</i>	
OTS_OTBMFile::OTBM_NODE_TOWNS	197
<i>Towns container.</i>	
OTS_OTBMFile::getTownsList()	199
<i>Returns list (id => name) of loaded towns.</i>	
OTS_OTBMFile::getTownTemple()	199
<i>Returns town's temple position.</i>	
OTS_OTBMFile::__wakeup()	201
<i>Magic PHP5 method.</i>	
OTS_Player	201
<i>OTServ character abstraction.</i>	
OTS_OTBMFile::__set_state()	200
<i>Magic PHP5 method.</i>	
OTS_OTBMFile::loadFile()	200
<i>Loads OTBM file content.</i>	
OTS_OTBMFile::getWidth()	200
<i>Returns map width.</i>	
OTS_Monster::hasImmunity()	187
<i>Checks if monster has given immunity.</i>	
OTS_Monster::getVoices()	187
<i>Returns voices that monster can sound.</i>	

OTS_InfoRespond::getOnlinePlayers()	173
<i>Returns current amount of players online.</i>	
OTS_InfoRespond::getOwner()	173
<i>Returns owner name.</i>	
OTS_InfoRespond::getName()	173
<i>Returns server name.</i>	
OTS_InfoRespond::getMOTD()	172
<i>Returns server's Message Of The Day</i>	
OTS_InfoRespond::getMonstersCount()	172
<i>Returns number of all monsters on map.</i>	
OTS_InfoRespond::getPlayersPeak()	174
<i>Returns record of online players.</i>	
OTS_InfoRespond::getPort()	174
<i>Returns server port.</i>	
OTS_InfoRespond::getUptime()	175
<i>Returns server uptime.</i>	
OTS_InfoRespond::getURL()	176
<i>Returns server website.</i>	
OTS_InfoRespond::getTSPQVersion()	175
<i>Returns version of root element.</i>	
OTS_InfoRespond::getServerVersion()	175
<i>Returns server version.</i>	
OTS_InfoRespond::getServer()	174
<i>Returns server attribute.</i>	
OTS_InfoRespond::getMaxPlayers()	172
<i>Returns maximum amount of players online.</i>	
OTS_InfoRespond::getMapWidth()	171
<i>Returns map width.</i>	
OTS_Guilds_List::init()	168
<i>Sets list parameters.</i>	
OTS_InfoRespond	169
<i>Wrapper for 'info' respond's DOMDocument.</i>	
OTS_Guilds_List::deleteGuild()	168
<i>Deletes guild.</i>	
OTS_Guilds_List	167
<i>List of guilds.</i>	
OTS_GuildRanks_List::init()	167
<i>Sets list parameters.</i>	
OTS_InfoRespond::getClientVersion()	169
<i>Returns dedicated version of client.</i>	
OTS_InfoRespond::getEmail()	169
<i>Returns owner e-mail.</i>	
OTS_InfoRespond::getMapHeight()	171
<i>Returns map height.</i>	
OTS_InfoRespond::getMapName()	171
<i>Returns map name.</i>	
OTS_InfoRespond::getMapAuthor()	170
<i>Returns map author.</i>	
OTS_InfoRespond::getLocation()	170
<i>Returns server location.</i>	
OTS_InfoRespond::getIP()	170
<i>Returns server IP.</i>	
OTS_Item	176

Single item representation.	
OTS_Item::count()	177
Count value for current item.	
OTS_Monster::getFlag()	184
Returns specified flag value.	
OTS_Monster::getFlags()	184
Returns all monster flags (in format flagname => value).	
OTS_Monster::getExperience()	183
Returns amount of experience for killing this monster.	
OTS_Monster::getDefenses()	183
Returns list of special defenses.	
OTS_Monster::getDefense()	183
Returns monster defense rate.	
OTS_Monster::getHealth()	184
Returns monster HP.	
OTS_Monster::getImmunities()	185
Returns all monster immunities.	
OTS_Monster::getRace()	186
Returns monster race.	
OTS_Monster::getSpeed()	186
Returns monster speed.	
OTS_Monster::getName()	186
Returns monster name.	
OTS_Monster::getManaCost()	185
Returns amount of mana required to summon this monster.	
OTS_Monster::getLoot()	185
Returns all possible loot.	
OTS_Monster::getAttacks()	182
Returns list of monster attacks.	
OTS_Monster::getArmor()	182
Returns monster armor.	
OTS_Item::setAttributes()	178
Sets item attributes.	
OTS_Item::setCount()	179
Sets count of item.	
OTS_Item::getId()	178
Returns item type.	
OTS_Item::getCount()	177
Returns count of item.	
OTS_Item::getAttributes()	177
Returns item custom attributes.	
OTS_MapCoords	179
Map position point.	
OTS_MapCoords::getX()	180
Returns X.	
OTS_Monster	182
Wrapper for monsters files DOMDocument.	
OTS_MapCoords::__set_state()	181
Magic PHP5 method.	
OTS_MapCoords::getZ()	181
Returns Z.	
OTS_MapCoords::getY()	180
Returns Y.	

OTS_Player::getId()	208
<i>Player ID.</i>	
OTS_Player::getLastIP()	208
<i>Last login IP.</i>	
OTS_Player::setRedSkull()	240
<i>Sets red skull flag.</i>	
OTS_Player::setRedSkullTime()	241
<i>Sets red skulled time remained.</i>	
OTS_Player::setRankId()	240
<i>Sets guild rank ID.</i>	
OTS_Player::setRank()	239
<i>Assigns guild rank.</i>	
OTS_Player::setPremiumEnd()	239
<i>Sets player's Premium Account expiration timestamp.</i>	
OTS_Player::setSave()	241
<i>Sets save counter.</i>	
OTS_Player::setSex()	242
<i>Sets player gender.</i>	
OTS_Player::setSoul()	244
<i>Sets soul points.</i>	
OTS_Player::setTownId()	244
<i>Sets residence town's ID.</i>	
OTS_Player::setSlot()	243
<i>Sets slot content.</i>	
OTS_Player::setSkillTries()	243
<i>Sets skill's tries for next level.</i>	
OTS_Player::setSkill()	242
<i>Sets skill value.</i>	
OTS_Player::setPosZ()	238
<i>Sets Z map coordinate.</i>	
OTS_Player::setPosY()	238
<i>Sets Y map coordinate.</i>	
OTS_Player::setLossExperience()	234
<i>Sets percentage of experience lost after dead.</i>	
OTS_Player::setLossMana()	234
<i>Sets percentage of used mana lost after dead.</i>	
OTS_Player::setLookType()	234
<i>Sets outfit.</i>	
OTS_Player::setLookLegs()	233
<i>Sets legs color.</i>	
OTS_Player::setLookHead()	233
<i>Sets hair color.</i>	
OTS_Player::setLossSkills()	235
<i>Sets percentage of skills lost after dead.</i>	
OTS_Player::setMagLevel()	235
<i>Sets magic level.</i>	
OTS_Player::setName()	237
<i>Sets players's name.</i>	
OTS_Player::setPosX()	238
<i>Sets X map coordinate.</i>	
OTS_Player::setManaSpent()	237
<i>Sets mana spent.</i>	
OTS_Player::setManaMax()	236

<i>Sets maximum mana.</i>	
OTS_Player::setMana()	236
<i>Sets current mana.</i>	
OTS_Player::setVocation()	245
<i>Sets player proffesion.</i>	
OTS_Player::unban()	245
<i>Deletes ban from current player.</i>	
OTS_SQLFilter::OPERATOR_NEQUAL	252
<i>Not-equal operator.</i>	
OTS_SQLFilter::OPERATOR_NGREATER	253
<i>Not-greater-then operator.</i>	
OTS_SQLFilter::OPERATOR_LOWER	252
<i>Lower-then operator.</i>	
OTS_SQLFilter::OPERATOR_LIKE	251
<i>LIKE operator.</i>	
OTS_SQLFilter::OPERATOR_GREATER	251
<i>Greater-then operator.</i>	
OTS_SQLFilter::OPERATOR_NLIKE	253
<i>Not-LIKE operator.</i>	
OTS_SQLFilter::OPERATOR_NLOWER	253
<i>Not-lower-then operator.</i>	
OTS_SQLFilter::sleep()	255
<i>Magic PHP5 method.</i>	
OTS_SQLFilter::toString()	255
<i>Returns string representation of WHERE clause.</i>	
OTS_SQLFilter::getTables()	255
<i>Returns list of all tables used by filter.</i>	
OTS_SQLFilter::compareField()	254
<i>Compares field with a literal value.</i>	
OTS_SQLFilter::addFilter()	254
<i>General-purpose filter.</i>	
OTS_SQLFilter::OPERATOR_EQUAL	251
<i>Equal operator.</i>	
OTS_SQLFilter::CRITERIUM_OR	250
<i>OR sibling.</i>	
OTS_Players_List	247
<i>List of players.</i>	
OTS_Players_List::deletePlayer()	247
<i>Deletes player.</i>	
OTS_Player::sleep()	246
<i>Magic PHP5 method.</i>	
OTS_Player::unsetSave()	246
<i>Unsets save flag.</i>	
OTS_Player::unsetRedSkull()	246
<i>Unsets red skull flag.</i>	
OTS_Players_List::init()	248
<i>Sets list parameters.</i>	
OTS_SQLField	248
<i>SQL identifier representation.</i>	
OTS_SQLFilter::CRITERIUM_AND	250
<i>AND sibling.</i>	
OTS_SQLFilter	250
<i>SQL WHERE clause object.</i>	

OTS SQLField::getTable()	249
<i>Returns table name.</i>	
OTS SQLField::getName()	249
<i>Returns field name.</i>	
OTS Player::setLookFeet()	232
<i>Sets boots color.</i>	
OTS Player::setLookBody()	232
<i>Sets body color.</i>	
OTS Player::getPosY()	215
<i>Y map coordinate.</i>	
OTS Player::getPosZ()	216
<i>Z map coordinate.</i>	
OTS Player::getPosX()	215
<i>X map coordinate.</i>	
OTS Player::getName()	215
<i>Player name.</i>	
OTS Player::getManaSpent()	214
<i>Mana spent.</i>	
OTS Player::getPremiumEnd()	216
<i>Player's Premium Account expiration timestamp.</i>	
OTS Player::getRank()	217
<i>Assigned guild rank.</i>	
OTS Player::getSex()	218
<i>Player gender.</i>	
OTS Player::getSkill()	219
<i>Returns player's skill.</i>	
OTS Player::getSave()	218
<i>Save counter.</i>	
OTS Player::getRedSkullTime()	217
<i>Red skulled time remained.</i>	
OTS Player::getRankId()	217
<i>Guild rank ID.</i>	
OTS Player::getManaMax()	214
<i>Maximum mana.</i>	
OTS Player::getMana()	214
<i>Current mana.</i>	
OTS Player::getLookBody()	210
<i>Body color.</i>	
OTS Player::getLookFeet()	210
<i>Boots color.</i>	
OTS Player::getLookAddons()	210
<i>Addons.</i>	
OTS Player::getLevel()	209
<i>Experience level.</i>	
OTS Player::getLastLogin()	209
<i>Last login timestamp.</i>	
OTS Player::getLookHead()	211
<i>Hair color.</i>	
OTS Player::getLookLegs()	211
<i>Legs color.</i>	
OTS Player::getLossSkills()	213
<i>Percentage of skills lost after dead.</i>	
OTS Player::getMagLevel()	213

<i>Magic level.</i>	212
OTS_Player::getLossMana()	212
<i>Percentage of used mana lost after dead.</i>	
OTS_Player::getLossExperience()	212
<i>Percentage of experience lost after dead.</i>	
OTS_Player::getLookType()	212
<i>Outfit.</i>	
OTS_Player::getSkillTries()	219
<i>Returns player's skill's tries for next level.</i>	
OTS_Player::getSlot()	220
<i>Returns items tree from given slot.</i>	
OTS_Player::setExperience()	228
<i>Sets experience points.</i>	
OTS_Player::setGroup()	228
<i>Assigns character to group.</i>	
OTS_Player::setDirection()	227
<i>Sets looking direction.</i>	
OTS_Player::setDepot()	227
<i>Sets depot content.</i>	
OTS_Player::setCustomField()	226
<i>Writes custom field.</i>	
OTS_Player::setGuildNick()	229
<i>Sets guild nick.</i>	
OTS_Player::setHealth()	229
<i>Sets current HP.</i>	
OTS_Player::setLevel()	231
<i>Sets experience level.</i>	
OTS_Player::setLookAddons()	231
<i>Sets addons.</i>	
OTS_Player::setLastLogin()	230
<i>Sets last login timestamp.</i>	
OTS_Player::setLastIP()	230
<i>Sets last login IP.</i>	
OTS_Player::setHealthMax()	229
<i>Sets maximum HP.</i>	
OTS_Player::setConditions()	225
<i>Sets conditions.</i>	
OTS_Player::setCap()	225
<i>Sets capacity.</i>	
OTS_Player::getVocationName()	222
<i>Player proffesion name.</i>	
OTS_Player::hasRedSkull()	222
<i>Checks if player has red skull.</i>	
OTS_Player::getVocation()	221
<i>Player proffesion.</i>	
OTS_Player::getTownId()	221
<i>Residence town's ID.</i>	
OTS_Player::getSoul()	220
<i>Soul points.</i>	
OTS_Player::isBanned()	222
<i>Checks if player is banned.</i>	
OTS_Player::isLoading()	223
<i>Checks if object is loaded.</i>	

OTS Player::setAccount()	224
<i>Assigns character to account.</i>	
OTS Player::save()	224
<i>Saves player in database.</i>	
OTS Player::load()	224
<i>Loads player with given id.</i>	
OTS Player::isSaveSet()	223
<i>Checks if save flag is set.</i>	
OTS GuildRanks List::deleteGuildRank()	166
<i>Deletes guild rank.</i>	
OTS GuildRanks List	166
<i>List of guild ranks.</i>	
OTS Base List::next()	99
<i>Moves to next row.</i>	
OTS Base List::orderBy()	100
<i>Appends sorting rule.</i>	
OTS Base List::key()	99
<i>Current cursor position.</i>	
OTS Base List::init()	99
<i>Sets list parameters.</i>	
OTS Base List::current()	98
<i>Returns current row.</i>	
OTS Base List::resetOrder()	100
<i>Clears ORDER BY clause.</i>	
OTS Base List::rewind()	100
<i>Select rows from database.</i>	
OTS Base List::valid()	102
<i>Checks if there are any rows left.</i>	
OTS Base List:: __set_state()	102
<i>Magic PHP5 method.</i>	
OTS Base List::setOffset()	102
<i>Sets OFFSET.</i>	
OTS Base List::setLimit()	101
<i>Sets LIMIT.</i>	
OTS Base List::setFilter()	101
<i>Sets filter on list.</i>	
OTS Base List::count()	98
<i>Returns number of accounts on list in current criterium.</i>	
OTS Base List::\$table	97
<i>Default table name for queries.</i>	
OTS Accounts List::init()	93
<i>Sets list parameters.</i>	
OTS Base DAO	93
<i>Basic data access object routines.</i>	
OTS Accounts List::deleteAccount()	92
<i>Deletes account.</i>	
OTS Accounts List	92
<i>List of accounts.</i>	
OTS Account::unblock()	92
<i>Unblocks account.</i>	
OTS Base DAO::\$db	94
<i>Database connection.</i>	
OTS Base DAO:: __clone()	95

Creates clone of object.	
OTS Base List	96
Basic list class routines.	
OTS Base List::\$class	97
Class of generated objects.	
OTS Base DAO:: wakeup()	96
Magic PHP5 method.	
OTS Base DAO:: sleep()	95
Magic PHP5 method.	
OTS Base DAO:: set_state()	95
Magic PHP5 method.	
OTS Base List:: sleep()	103
Magic PHP5 method.	
OTS Base List:: wakeup()	103
Magic PHP5 method.	
OTS DB ODBC::fieldName()	112
Query-quoted field name.	
OTS DB ODBC::limit()	112
LIMIT/OFFSET clause for queries.	
OTS DB ODBC	111
ODBC connection interface.	
OTS DB MySQL::tableName()	110
Query-quoted table name.	
OTS DB MySQL::SQLquote()	110
IOTS_DB method.	
OTS DB ODBC::SQLquery()	113
IOTS_DB method.	
OTS DB ODBC::SQLquote()	113
IOTS_DB method.	
OTS DB PostgreSQL::limit()	116
LIMIT/OFFSET clause for queries.	
OTS DB PostgreSQL::SQLquery()	116
IOTS_DB method.	
OTS DB PostgreSQL::fieldName()	115
Query-quoted field name.	
OTS DB PostgreSQL	114
PostgreSQL connection interface.	
OTS DB ODBC::tableName()	114
Query-quoted table name.	
OTS DB MySQL::SQLquery()	109
IOTS_DB method.	
OTS DB MySQL::limit()	109
LIMIT/OFFSET clause for queries.	
OTS Container::current()	105
Returns current item.	
OTS Container::key()	105
Current cursor position.	
OTS Container::count()	105
Number of items inside container.	
OTS Container::addItem()	104
Adds item to container.	
OTS Container	104
Container item representation.	

OTS Container::next()	106
<i>Moves to next item.</i>	
OTS Container::removeItem()	106
<i>Removes given item from current container.</i>	
OTS DB MySQL::fieldName()	108
<i>Query-quoted field name.</i>	
OTS DB MySQL	107
<i>MySQL connection interface.</i>	
OTS Container::valid()	107
<i>Checks if there are any items left.</i>	
OTS Container::rewind()	106
<i>Resets internal items array pointer.</i>	
OTS Account::unban()	91
<i>Deletes ban from current account.</i>	
OTS Account::setPassword()	91
<i>Sets account's password.</i>	
OTS InfoRespond.php	53
OTS Item.php	54
OTS Guilds_List.php	52
OTS GuildRanks_List.php	51
OTS GuildRank.php	50
OTS MapCoords.php	55
<i>Code in this file bases on oryiginal OTServ OTBM format loading C++ code (iomapotbm).</i>	
OTS Monster.php	56
OTS SQLField.php	60
OTS SQLFilter.php	61
OTS Players_List.php	59
OTS Player.php	58
OTS OTBMFile.php	57
<i>Code in this file bases on oryiginal OTServ OTBM format loading C++ code (iomapotbm).</i>	
OTS Guild.php	49
OTS Groups_List.php	48
OTS Base_List.php	39
OTS Container.php	40
OTS Base DAO.php	38
OTS Accounts_List.php	37
OTS Account.php	36
OTS DB MySQL.php	41
OTS DB ODBC.php	42
OTS FileNode.php	46
<i>Code in this file bases on oryiginal OTServ binary format loading C++ code (fileloader).</i>	
OTS Group.php	47
OTS FileLoader.php	45
<i>Code in this file bases on oryiginal OTServ binary format loading C++ code (fileloader).</i>	
OTS DB SQLite.php	44
OTS DB PostgreSQL.php	43
OTS SQLite_Results.php	62
OTS Account	78
<i>OTServ account abstraction.</i>	
OTS Account::isBanned()	86
<i>Checks if account is banned.</i>	
OTS Account::isBlocked()	87
<i>Checks if account is blocked.</i>	

OTS Account::getPlayersList()	86
<i>List of characters on account.</i>	
OTS Account::getPlayers()	86
<i>List of characters on account.</i>	
OTS Account::getPassword()	85
<i>Account's password.</i>	
OTS Account::isLoading()	87
<i>Checks if object is loaded.</i>	
OTS Account::load()	88
<i>Loads account with given number.</i>	
OTS Account::setGroup()	90
<i>Assigns account to group.</i>	
OTS Account::setPACCDays()	90
<i>Sets PACC days count.</i>	
OTS Account::setEMail()	89
<i>Sets account's email.</i>	
OTS Account::setCustomField()	88
<i>Writes custom field.</i>	
OTS Account::save()	88
<i>Updates account in database.</i>	
OTS Account::getPACCDays()	85
<i>PACC days.</i>	
OTS Account::getIterator()	84
<i>Returns players iterator.</i>	
OTS Account::create()	80
<i>Creates new account.</i>	
OTS Account::createEx()	81
<i>Creates new account.</i>	
OTS Account::count()	79
<i>Returns number of player within.</i>	
OTS Account::block()	79
<i>Blocks account.</i>	
OTS Account::ban()	78
<i>Bans current account.</i>	
OTS Account::delete()	81
<i>Deletes account.</i>	
OTS Account::find()	82
<i>Loads account by it's e-mail address.</i>	
OTS Account::getId()	84
<i>Account number.</i>	
OTS Account::getGroup()	83
<i>Returns group of this account.</i>	
OTS Account::getEMail()	83
<i>E-mail address.</i>	
OTS Account::getCustomField()	82
<i>Reads custom field.</i>	
OTS DB PostgreSQL::SQLquote()	117
<i>IOTS_DB method.</i>	
OTS DB PostgreSQL::tableName()	117
<i>Query-quoted table name.</i>	
OTS Guild::getName()	150
<i>Guild name.</i>	
OTS Guild::getOwner()	150

<i>Returns owning player of this player.</i>	
OTS_Guild::getIterator()	150
<i>Returns ranks iterator.</i>	
OTS_Guild::getId()	149
<i>Guild ID.</i>	
OTS_Guild::getGuildRanksList()	149
<i>List of ranks in guild.</i>	
OTS_Guild::invite()	151
<i>Invites player to guild.</i>	
OTS_Guild::isLoading()	151
<i>Checks if object is loaded.</i>	
OTS_Guild::request()	153
<i>Requests membership in guild for player player.</i>	
OTS_Guild::save()	153
<i>Saves guild in database.</i>	
OTS_Guild::load()	152
<i>Loads guild with given id.</i>	
OTS_Guild::listRequests()	152
<i>Returns list of players that requested membership.</i>	
OTS_Guild::listInvites()	152
<i>Returns list of invited players.</i>	
OTS_Guild::getGuildRanks()	148
<i>Reads all ranks that are in this guild.</i>	
OTS_Guild::getCustomField()	148
<i>Reads custom field.</i>	
OTS_Guild	144
<i>OTServ guild abstraction.</i>	
OTS_Guild::acceptInvite()	144
<i>Finalise invitation.</i>	
OTS_Groups_List::init()	143
<i>Sets list parameters.</i>	
OTS_Groups_List::deleteGroup()	143
<i>Deletes group.</i>	
OTS_Groups_List	142
<i>List of groups.</i>	
OTS_Guild::acceptRequest()	144
<i>Accepts player.</i>	
OTS_Guild::count()	145
<i>Returns number of ranks within.</i>	
OTS_Guild::find()	147
<i>Loads guild by it's name.</i>	
OTS_Guild::getCreationData()	147
<i>Guild creation data.</i>	
OTS_Guild::deleteRequest()	146
<i>Deletes request from player.</i>	
OTS_Guild::deleteInvite()	146
<i>Deletes invitation for player to guild.</i>	
OTS_Guild::delete()	145
<i>Deletes guild.</i>	
OTS_Guild::setCreationData()	154
<i>Sets guild creation data.</i>	
OTS_Guild::setCustomField()	154
<i>Writes custom field.</i>	

OTS_GuildRank::getPlayers()	162
<i>Reads all players who has this rank set.</i>	
OTS_GuildRank::getPlayersList()	162
<i>List of characters with current rank.</i>	
OTS_GuildRank::getName()	161
<i>Rank name.</i>	
OTS_GuildRank::getLevel()	161
<i>Rank's access level.</i>	
OTS_GuildRank::getIterator()	161
<i>Returns players iterator.</i>	
OTS_GuildRank::isLoading()	163
<i>Checks if object is loaded.</i>	
OTS_GuildRank::load()	163
<i>Loads rank with given id.</i>	
OTS_GuildRank::setLevel()	165
<i>Sets rank's access level within guild.</i>	
OTS_GuildRank::setName()	166
<i>Sets rank's name.</i>	
OTS_GuildRank::setGuild()	165
<i>Assigns rank to guild.</i>	
OTS_GuildRank::setCustomField()	164
<i>Writes custom field.</i>	
OTS_GuildRank::save()	164
<i>Saves rank in database.</i>	
OTS_GuildRank::getId()	160
<i>Rank ID.</i>	
OTS_GuildRank::getGuild()	160
<i>Returns guild of this rank.</i>	
OTS_Guild::setRequestsDriver()	156
<i>Assigns requests handler.</i>	
OTS_Guild::__clone()	157
<i>Creates clone of object.</i>	
OTS_Guild::setOwner()	156
<i>Assigns guild to owner.</i>	
OTS_Guild::setName()	155
<i>Sets players's name.</i>	
OTS_Guild::setInvitesDriver()	155
<i>Assigns invites handler.</i>	
OTS_Guild::__sleep()	157
<i>Magic PHP5 method.</i>	
OTS_GuildRank	158
<i>OTServ guild rank abstraction.</i>	
OTS_GuildRank::getCustomField()	159
<i>Reads custom field.</i>	
OTS_GuildRank::find()	159
<i>Loads rank by it's name.</i>	
OTS_GuildRank::delete()	158
<i>Deletes guild rank.</i>	
OTS_GuildRank::count()	158
<i>Returns number of player within.</i>	
OTS_Group::setName()	142
<i>Sets group's name.</i>	
OTS_Group::setMaxViplist()	141

<i>Sets maximum count of players in VIP list.</i>	
OTS_FileNode::getBuffer()	126
<i>Returns properties stream.</i>	
OTS_FileNode::getChar()	126
<i>Returns single byte.</i>	
OTS_FileNode	126
<i>OTServ binary file node representation.</i>	
OTS_FileLoader::sleep()	125
<i>Magic PHP5 method.</i>	
OTS_FileLoader::set_state()	125
<i>Magic PHP5 method.</i>	
OTS_FileNode::getChild()	127
<i>Returns first child.</i>	
OTS_FileNode::getLong()	127
<i>Returns quater byte.</i>	
OTS_FileNode::getType()	128
<i>Returns node type.</i>	
OTS_FileNode::isValid()	129
<i>Checks if there is anything left in stream.</i>	
OTS_FileNode::getString()	128
<i>Returns string from buffer.</i>	
OTS_FileNode::getShort()	128
<i>Returns double byte.</i>	
OTS_FileNode::getNext()	127
<i>Returns next sibling.</i>	
OTS_FileLoader::clone()	124
<i>Creates clone of object.</i>	
OTS_FileLoader::setCacheDriver()	124
<i>Sets cache handler.</i>	
OTS_DB_SQLite::SQLquery()	120
<i>IOTS_DB method.</i>	
OTS_DB_SQLite::SQLquote()	120
<i>IOTS_DB method.</i>	
OTS_DB_SQLite::limit()	119
<i>LIMIT/OFFSET clause for queries.</i>	
OTS_DB_SQLite::fieldName()	119
<i>Query-quoted field name.</i>	
OTS_DB_SQLite	118
<i>SQLite connection interface.</i>	
OTS_DB_SQLite::tableName()	121
<i>Query-quoted table name.</i>	
OTS_FileLoader	122
<i>Universal OTServ binary formats reader.</i>	
OTS_FileLoader::\$root	123
<i>Root node.</i>	
OTS_FileLoader::loadFile()	123
<i>Opens file.</i>	
OTS_FileLoader::NODE_START	123
<i>Start of node.</i>	
OTS_FileLoader::NODE_END	122
<i>End of node.</i>	
OTS_FileLoader::ESCAPE_CHAR	122
<i>Escape another special byte.</i>	

OTS FileNode::setBuffer()	129
<i>Sets properties stream.</i>	
OTS FileNode::setChild()	130
<i>Sets first child.</i>	
OTS Group::getPlayers()	137
<i>List of characters in given group.</i>	
OTS Group::getPlayersList()	137
<i>List of characters in group.</i>	
OTS Group::getName()	137
<i>Group name.</i>	
OTS Group::getMaxVIPList()	136
<i>Maximum count of players in VIP list.</i>	
OTS Group::getMaxDepotItems()	136
<i>Maximum count of items in depot.</i>	
OTS Group::isLoading()	138
<i>Checks if object is loaded.</i>	
OTS Group::load()	138
<i>Loads group with given id.</i>	
OTS Group::setFlags()	140
<i>Sets rights flags.</i>	
OTS Group::setMaxDepotItems()	141
<i>Sets maximum count of items in depot.</i>	
OTS Group::setCustomField()	139
<i>Writes custom field.</i>	
OTS Group::setAccess()	139
<i>Sets access level.</i>	
OTS Group::save()	139
<i>Saves account in database.</i>	
OTS Group::getIterator()	135
<i>Returns players iterator.</i>	
OTS Group::getId()	135
<i>Group ID.</i>	
OTS FileNode:: clone()	131
<i>Creates clone of object.</i>	
OTS FileNode:: set_state()	132
<i>Magic PHP5 method.</i>	
OTS FileNode::skip()	131
<i>Skips given amount of bytes.</i>	
OTS FileNode::setType()	130
<i>Sets node type.</i>	
OTS FileNode::setNext()	130
<i>Sets next sibling.</i>	
OTS Group	132
<i>OTServ user group abstraction.</i>	
OTS Group::count()	133
<i>Returns number of player within.</i>	
OTS Group::getFlags()	135
<i>Rights flags.</i>	
OTS Group::getCustomField()	134
<i>Reads custom field.</i>	
OTS Group::getAccess()	134
<i>Access level.</i>	
OTS Group::delete()	133

<i>Deletes group.</i>	
OTS.php	35
<i>This file contains main toolkit class.</i>	

P

POT::VOCATION_PALADIN	270
<i>Paladin.</i>	
POT::VOCATION_NONE	270
<i>None vocation.</i>	
POT::VOCATION_SORCERER	271
<i>Sorcerer.</i>	
POT::banIP()	271
<i>Bans given IP number.</i>	
POT::createFilter()	273
<i>Creates lists filter.</i>	
POT::connect()	272
<i>Connects to database.</i>	
POT::VOCATION_KNIGHT	270
<i>Knight.</i>	
POT::VOCATION_DRUID	269
<i>Druid.</i>	
POT::SLOT_LEFT	267
<i>Left hand slot.</i>	
POT::SLOT_HEAD	267
<i>Head slot.</i>	
POT::SLOT_LEGS	268
<i>Legs slot.</i>	
POT::SLOT_NECKLACE	268
<i>Necklace slot.</i>	
POT::SLOT_RING	269
<i>Ring slot.</i>	
POT::SLOT_RIGHT	268
<i>Right hand slot.</i>	
POT::createObject()	273
<i>Creates OTServ DAO class instance.</i>	
POT::getDBHandle()	274
<i>Returns database connection handle.</i>	
POT::loadMonsters()	278
<i>Loads monsters mapping file.</i>	
POT::loadClass()	278
<i>Loads POT class file.</i>	
POT::loadVocations()	279
<i>Loads vocations list.</i>	
POT::serverStatus()	279
<i>Queries server status.</i>	
POT::unbanIP()	281
<i>Deletes ban from given IP number.</i>	
POT::setPOTPath()	280
<i>Set POT directory.</i>	
POT::isIPBanned()	277
<i>Checks if given IP is banned.</i>	

POT::getVocationsList()	277
<i>Returns list (id => name) of loaded vocations.</i>	
POT::getMonster()	275
<i>Returns loaded data of given monster.</i>	
POT::getInstance()	274
<i>Singleton.</i>	
POT::getMonstersList()	275
<i>Returns list of loaded monsters.</i>	
POT::getVocationID()	276
<i>Returns vocation's ID.</i>	
POT::getVocationName()	276
<i>Returns name of given vocation's ID.</i>	
POT::SLOT_FEET	266
<i>Boots slot.</i>	
POT::SLOT_BACKPACK	266
<i>Backpack slot.</i>	
POT::DB_PGSQL	258
<i>PostgreSQL driver.</i>	
POT::DB_ODBC	258
<i>ODBC driver.</i>	
POT::DB_SQLITE	259
<i>SQLite driver.</i>	
POT::DEPOT_SID_FIRST	259
<i>First depot item sid.</i>	
POT::DIRECTION_NORTH	260
<i>North.</i>	
POT::DIRECTION_EAST	259
<i>East.</i>	
POT::DB_MYSQL	258
<i>MySQL driver.</i>	
POT::BAN_PLAYER	257
<i>Player ban.</i>	
POT class preview	5
PHP 5.0	3
POT	256
<i>Main POT class.</i>	
POT::BAN_ACCOUNT	256
<i>Account ban.</i>	
POT::BAN_IP	257
<i>IP ban.</i>	
POT::DIRECTION_SOUTH	260
<i>South.</i>	
POT::DIRECTION_WEST	260
<i>West.</i>	
POT::SKILL_FIST	264
<i>Fist fighting.</i>	
POT::SKILL_FISHING	264
<i>Fishing.</i>	
POT::SKILL_SHIELDING	264
<i>Shielding.</i>	
POT::SKILL_SWORD	265
<i>Sword fighting.</i>	
POT::SLOT_ARMOR	266

<i>Armor slot.</i>	
POT::SLOT_AMMO	265
<i>Ammunition slot.</i>	
POT::SKILL_DISTANCE	263
<i>Distance fighting.</i>	
POT::SKILL_CLUB	263
<i>Club fighting.</i>	
POT::ORDER_DESC	261
<i>Descending sorting order.</i>	
POT::ORDER_ASC	261
<i>Ascencind sorting order.</i>	
POT::SEX_FEMALE	262
<i>Female gender.</i>	
POT::SEX_MALE	262
<i>Male gender.</i>	
POT::SKILL_AXE	262
<i>Axe fighting.</i>	
POT	1

Q

Quick start	6
-----------------------------	---

R

README	296
RULES	294

S

Server online status	19
--------------------------------------	----