

# PHP OTServ Toolkit



# Contents

<a href="#">POT</a>	1
<a href="#">PHP 5.0</a>	3
<a href="#">POT class preview</a>	5
<a href="#">Quick start</a>	6
<a href="#">DAO objects</a>	9
<a href="#">Guilds</a>	13
<a href="#">Guild action drivers</a>	15
<a href="#">Account number hack</a>	16
<a href="#">Server online status</a>	17
<a href="#">Package default Classes</a>	19
<a href="#">Class InvitesDriver</a>	19
<a href="#">Constructor construct</a>	19
<a href="#">Method addRequest</a>	19
<a href="#">Method deleteRequest</a>	20
<a href="#">Method listRequests</a>	20
<a href="#">Method submitRequest</a>	20
<a href="#">Package POT Procedural Elements</a>	22
<a href="#">E OTS NoDriver.php</a>	22
<a href="#">E OTS NotLoaded.php</a>	23
<a href="#">IOTS DAO.php</a>	24
<a href="#">IOTS DB.php</a>	25
<a href="#">IOTS GuildAction.php</a>	26
<a href="#">OTS.php</a>	27
<a href="#">OTS Account.php</a>	28
<a href="#">OTS Accounts List.php</a>	29
<a href="#">OTS Container.php</a>	30
<a href="#">OTS DB MySQL.php</a>	31
<a href="#">OTS DB ODBC.php</a>	32
<a href="#">OTS DB PostgreSQL.php</a>	33
<a href="#">OTS DB SQLite.php</a>	34
<a href="#">OTS Group.php</a>	35
<a href="#">OTS Groups List.php</a>	36
<a href="#">OTS Guild.php</a>	37
<a href="#">OTS GuildRank.php</a>	38
<a href="#">OTS GuildRanks List.php</a>	39
<a href="#">OTS Guilds List.php</a>	40
<a href="#">OTS InfoRespond.php</a>	41
<a href="#">OTS Item.php</a>	42
<a href="#">OTS Player.php</a>	43
<a href="#">OTS Players List.php</a>	44
<a href="#">OTS SQLite Results.php</a>	45

<a href="#"><u>Package POT Classes</u></a>	46
<a href="#"><u>Class E OTS NoDriver</u></a>	46
<a href="#"><u>Class E OTS NotLoaded</u></a>	46
<a href="#"><u>Class IOTS DAO</u></a>	47
<a href="#"><u>Constructor construct</u></a>	47
<a href="#"><u>Class IOTS DB</u></a>	48
<a href="#"><u>Constructor construct</u></a>	48
<a href="#"><u>Method fieldName</u></a>	48
<a href="#"><u>Method lastInsertId</u></a>	49
<a href="#"><u>Method limit</u></a>	49
<a href="#"><u>Method SQLquery</u></a>	50
<a href="#"><u>Method SQLquote</u></a>	50
<a href="#"><u>Method tableName</u></a>	50
<a href="#"><u>Class IOTS GuildAction</u></a>	51
<a href="#"><u>Constructor construct</u></a>	51
<a href="#"><u>Method addRequest</u></a>	52
<a href="#"><u>Method deleteRequest</u></a>	52
<a href="#"><u>Method listRequests</u></a>	53
<a href="#"><u>Method submitRequest</u></a>	53
<a href="#"><u>Class OTS Account</u></a>	54
<a href="#"><u>Constructor construct</u></a>	54
<a href="#"><u>Method block</u></a>	54
<a href="#"><u>Method create</u></a>	55
<a href="#"><u>example: account.php</u></a>	55
<a href="#"><u>Method createEx</u></a>	56
<a href="#"><u>example: account.php</u></a>	56
<a href="#"><u>Method find</u></a>	57
<a href="#"><u>Method getCustomField</u></a>	57
<a href="#"><u>Method getEmail</u></a>	58
<a href="#"><u>Method getGroup</u></a>	58
<a href="#"><u>Method getId</u></a>	59
<a href="#"><u>Method getPACCDays</u></a>	59
<a href="#"><u>Method getPassword</u></a>	60
<a href="#"><u>Method getPlayers</u></a>	60
<a href="#"><u>Method isBlocked</u></a>	60
<a href="#"><u>Method isLoaded</u></a>	61
<a href="#"><u>Method load</u></a>	61
<a href="#"><u>Method save</u></a>	61
<a href="#"><u>Method setCustomField</u></a>	62
<a href="#"><u>Method setEmail</u></a>	62
<a href="#"><u>Method setGroup</u></a>	63
<a href="#"><u>Method setPACCDays</u></a>	63
<a href="#"><u>Method setPassword</u></a>	64
<a href="#"><u>Method unblock</u></a>	64
<a href="#"><u>Class OTS Accounts List</u></a>	65
<a href="#"><u>Constructor construct</u></a>	65
<a href="#"><u>Method count</u></a>	65
<a href="#"><u>Method current</u></a>	66
<a href="#"><u>Method deleteAccount</u></a>	66

<a href="#">Method key</a>	66
<a href="#">Method next</a>	67
<a href="#">Method rewind</a>	67
<a href="#">Method setLimit</a>	67
<a href="#">Method setOffset</a>	68
<a href="#">Method valid</a>	68
<a href="#">Class OTS Container</a>	69
<a href="#">Method addItem</a>	69
<a href="#">Method count</a>	69
<a href="#">Method current</a>	70
<a href="#">Method key</a>	70
<a href="#">Method next</a>	70
<a href="#">Method removeItem</a>	71
<a href="#">Method rewind</a>	71
<a href="#">Method valid</a>	71
<a href="#">Class OTS DB MySQL</a>	72
<a href="#">Constructor construct</a>	72
<a href="#">Method fieldName</a>	73
<a href="#">Method limit</a>	73
<a href="#">Method SQLquery</a>	74
<a href="#">Method SQLquote</a>	74
<a href="#">Method tableName</a>	75
<a href="#">Class OTS DB ODBC</a>	75
<a href="#">Constructor construct</a>	75
<a href="#">Method fieldName</a>	76
<a href="#">Method limit</a>	77
<a href="#">Method SQLquery</a>	77
<a href="#">Method SQLquote</a>	78
<a href="#">Method tableName</a>	78
<a href="#">Class OTS DB PostgreSQL</a>	79
<a href="#">Constructor construct</a>	79
<a href="#">Method fieldName</a>	80
<a href="#">Method limit</a>	80
<a href="#">Method SQLquery</a>	81
<a href="#">Method SQLquote</a>	81
<a href="#">Method tableName</a>	81
<a href="#">Class OTS DB SQLite</a>	82
<a href="#">Constructor construct</a>	82
<a href="#">Method fieldName</a>	83
<a href="#">Method limit</a>	83
<a href="#">Method SQLquery</a>	84
<a href="#">Method SQLquote</a>	84
<a href="#">Method tableName</a>	85
<a href="#">Class OTS Group</a>	85
<a href="#">Constructor construct</a>	86
<a href="#">Method getAccess</a>	86
<a href="#">Method getCustomField</a>	86
<a href="#">Method getFlags</a>	87
<a href="#">Method getId</a>	87

<a href="#">Method getMaxDepotItems</a>	88
<a href="#">Method getMaxVIPList</a>	88
<a href="#">Method getName</a>	88
<a href="#">Method getPlayers</a>	89
<a href="#">Method isLoaded</a>	89
<a href="#">Method load</a>	89
<a href="#">Method save</a>	90
<a href="#">Method setAccess</a>	90
<a href="#">Method setCustomField</a>	91
<a href="#">Method setFlags</a>	91
<a href="#">Method setMaxDepotItems</a>	92
<a href="#">Method setMaxVIPList</a>	92
<a href="#">Method setName</a>	93
<a href="#">Class OTS_Groups_List</a>	93
<a href="#">Constructor construct</a>	93
<a href="#">Method count</a>	94
<a href="#">Method current</a>	94
<a href="#">Method deleteGroup</a>	94
<a href="#">Method key</a>	95
<a href="#">Method next</a>	95
<a href="#">Method rewind</a>	95
<a href="#">Method setLimit</a>	96
<a href="#">Method setOffset</a>	96
<a href="#">Method valid</a>	97
<a href="#">Class OTS_Guild</a>	97
<a href="#">Constructor construct</a>	97
<a href="#">Method acceptInvite</a>	98
<a href="#">Method acceptRequest</a>	98
<a href="#">Method deleteInvite</a>	99
<a href="#">Method deleteRequest</a>	99
<a href="#">Method find</a>	100
<a href="#">Method getCreationData</a>	100
<a href="#">Method getCustomField</a>	101
<a href="#">Method getGuildRanks</a>	101
<a href="#">Method getID</a>	102
<a href="#">Method getName</a>	102
<a href="#">Method getOwner</a>	102
<a href="#">Method invite</a>	103
<a href="#">Method isLoaded</a>	103
<a href="#">Method listInvites</a>	104
<a href="#">Method listRequests</a>	104
<a href="#">Method load</a>	104
<a href="#">Method request</a>	105
<a href="#">Method save</a>	105
<a href="#">Method setCreationData</a>	106
<a href="#">Method setCustomField</a>	106
<a href="#">Method setInvitesDriver</a>	107
<a href="#">Method setName</a>	107
<a href="#">Method setOwner</a>	108

<a href="#">Method setRequestsDriver</a>	108
<a href="#">Class OTS_GuildRank</a>	109
<a href="#">Constructor construct</a>	109
<a href="#">Method find</a>	109
<a href="#">Method getCustomField</a>	110
<a href="#">Method getGuild</a>	111
<a href="#">Method getId</a>	111
<a href="#">Method getLevel</a>	111
<a href="#">Method getName</a>	112
<a href="#">Method getPlayers</a>	112
<a href="#">Method isLoaded</a>	112
<a href="#">Method load</a>	113
<a href="#">Method save</a>	113
<a href="#">Method setCustomField</a>	113
<a href="#">Method setGuild</a>	114
<a href="#">Method setLevel</a>	115
<a href="#">Method setName</a>	115
<a href="#">Class OTS_GuildRanks_List</a>	116
<a href="#">Constructor construct</a>	116
<a href="#">Method count</a>	116
<a href="#">Method current</a>	117
<a href="#">Method deleteGuildRank</a>	117
<a href="#">Method key</a>	117
<a href="#">Method next</a>	118
<a href="#">Method rewind</a>	118
<a href="#">Method setLimit</a>	118
<a href="#">Method setOffset</a>	119
<a href="#">Method valid</a>	119
<a href="#">Class OTS_Guilds_List</a>	120
<a href="#">Constructor construct</a>	120
<a href="#">Method count</a>	121
<a href="#">Method current</a>	121
<a href="#">Method deleteGuild</a>	121
<a href="#">Method key</a>	122
<a href="#">Method next</a>	122
<a href="#">Method rewind</a>	122
<a href="#">Method setLimit</a>	123
<a href="#">Method setOffset</a>	123
<a href="#">Method valid</a>	124
<a href="#">Class OTS_InfoRespond</a>	124
<a href="#">Method getClientVersion</a>	124
<a href="#">Method getEmail</a>	125
<a href="#">Method getIP</a>	125
<a href="#">Method getLocation</a>	125
<a href="#">Method getMapAuthor</a>	126
<a href="#">Method getMapHeight</a>	126
<a href="#">Method getMapName</a>	126
<a href="#">Method getMapWidth</a>	127
<a href="#">Method getMaxPlayers</a>	127

<a href="#">Method getMonstersCount</a>	127
<a href="#">Method getMOTD</a>	128
<a href="#">Method getName</a>	128
<a href="#">Method getOnlinePlayers</a>	128
<a href="#">Method getOwner</a>	129
<a href="#">Method getPlayersPeak</a>	129
<a href="#">Method getPort</a>	129
<a href="#">Method getServer</a>	130
<a href="#">Method getServerVersion</a>	130
<a href="#">Method getTSPQVersion</a>	130
<a href="#">Method getUptime</a>	131
<a href="#">Method getURL</a>	131
<a href="#">Class OTS_Item</a>	131
<a href="#">Constructor construct</a>	132
<a href="#">Method count</a>	132
<a href="#">Method getAttributes</a>	132
<a href="#">Method getCount</a>	133
<a href="#">Method getId</a>	133
<a href="#">Method setAttributes</a>	133
<a href="#">Method setCount</a>	134
<a href="#">Class OTS_Player</a>	134
<a href="#">Constructor construct</a>	135
<a href="#">Method find</a>	135
<a href="#">Method getAccount</a>	136
<a href="#">Method getCap</a>	136
<a href="#">Method getConditions</a>	136
<a href="#">Method getCustomField</a>	137
<a href="#">Method getDepot</a>	137
<a href="#">Method getDirection</a>	138
<a href="#">Method getExperience</a>	138
<a href="#">Method getGroup</a>	139
<a href="#">Method getGuildNick</a>	139
<a href="#">Method getHealth</a>	139
<a href="#">Method getHealthMax</a>	140
<a href="#">Method getId</a>	140
<a href="#">Method getLastIP</a>	140
<a href="#">Method getLastLogin</a>	141
<a href="#">Method getLevel</a>	141
<a href="#">Method getLookAddons</a>	141
<a href="#">Method getLookBody</a>	142
<a href="#">Method getLookFeet</a>	142
<a href="#">Method getLookHead</a>	143
<a href="#">Method getLookLegs</a>	143
<a href="#">Method getLookType</a>	143
<a href="#">Method getLossExperience</a>	144
<a href="#">Method getLossMana</a>	144
<a href="#">Method getLossSkills</a>	144
<a href="#">Method getMagLevel</a>	145
<a href="#">Method getMana</a>	145

<a href="#">Method getManaMax</a>	145
<a href="#">Method getManaSpent</a>	146
<a href="#">Method getName</a>	146
<a href="#">Method getPosX</a>	146
<a href="#">Method getPosY</a>	147
<a href="#">Method getPosZ</a>	147
<a href="#">Method getPremiumEnd</a>	148
<a href="#">Method getRank</a>	148
<a href="#">Method getRankId</a>	148
<a href="#">Method getRedSkullTime</a>	149
<a href="#">Method getSex</a>	149
<a href="#">Method getSkill</a>	149
<a href="#">Method getSkillTries</a>	150
<a href="#">Method getSlot</a>	150
<a href="#">Method getSoul</a>	151
<a href="#">Method getTownId</a>	151
<a href="#">Method getVocation</a>	152
<a href="#">Method hasRedSkull</a>	152
<a href="#">Method isLoaded</a>	152
<a href="#">Method isSaveSet</a>	153
<a href="#">Method load</a>	153
<a href="#">Method save</a>	153
<a href="#">Method setAccount</a>	154
<a href="#">Method setCap</a>	154
<a href="#">Method setConditions</a>	155
<a href="#">Method setCustomField</a>	155
<a href="#">Method setDepot</a>	156
<a href="#">Method setDirection</a>	156
<a href="#">Method setExperience</a>	157
<a href="#">Method setGroup</a>	157
<a href="#">Method setGuildNick</a>	158
<a href="#">Method setHealth</a>	158
<a href="#">Method setHealthMax</a>	159
<a href="#">Method setLastIP</a>	159
<a href="#">Method setLastLogin</a>	159
<a href="#">Method setLevel</a>	160
<a href="#">Method setLookAddons</a>	160
<a href="#">Method setLookBody</a>	161
<a href="#">Method setLookFeet</a>	161
<a href="#">Method setLookHead</a>	162
<a href="#">Method setLookLegs</a>	162
<a href="#">Method setLookType</a>	162
<a href="#">Method setLossExperience</a>	163
<a href="#">Method setLossMana</a>	163
<a href="#">Method setLossSkills</a>	164
<a href="#">Method setMagLevel</a>	164
<a href="#">Method setMana</a>	164
<a href="#">Method setManaMax</a>	165
<a href="#">Method setManaSpent</a>	165



<a href="#">Method setName</a>	166
<a href="#">Method setPosX</a>	166
<a href="#">Method setPosY</a>	167
<a href="#">Method setPosZ</a>	167
<a href="#">Method setPremiumEnd</a>	167
<a href="#">Method setRank</a>	168
<a href="#">Method setRankId</a>	168
<a href="#">Method setRedSkull</a>	169
<a href="#">Method setRedSkullTime</a>	169
<a href="#">Method setSave</a>	169
<a href="#">Method setSex</a>	170
<a href="#">Method setSkill</a>	170
<a href="#">Method setSkillTries</a>	171
<a href="#">Method setSlot</a>	171
<a href="#">Method setSoul</a>	172
<a href="#">Method setTownId</a>	172
<a href="#">Method setVocation</a>	173
<a href="#">Method unsetRedSkull</a>	173
<a href="#">Method unsetSave</a>	173
<a href="#">Class OTS Players List</a>	174
<a href="#">Constructor construct</a>	174
<a href="#">Method count</a>	174
<a href="#">Method current</a>	175
<a href="#">Method deletePlayer</a>	175
<a href="#">Method key</a>	176
<a href="#">Method next</a>	176
<a href="#">Method rewind</a>	176
<a href="#">Method setLimit</a>	176
<a href="#">Method setOffset</a>	177
<a href="#">Method valid</a>	177
<a href="#">Class POT</a>	178
<a href="#">Class Constant DB_MYSQL</a>	178
<a href="#">Class Constant DB_ODBC</a>	178
<a href="#">Class Constant DB_PGSQL</a>	179
<a href="#">Class Constant DB_SQLITE</a>	179
<a href="#">Class Constant DIRECTION_EAST</a>	179
<a href="#">Class Constant DIRECTION_NORTH</a>	180
<a href="#">Class Constant DIRECTION_SOUTH</a>	180
<a href="#">Class Constant DIRECTION_WEST</a>	180
<a href="#">Class Constant SEX_FEMALE</a>	181
<a href="#">Class Constant SEX_MALE</a>	181
<a href="#">Class Constant SKILL_AXE</a>	181
<a href="#">Class Constant SKILL_CLUB</a>	182
<a href="#">Class Constant SKILL_DISTANCE</a>	182
<a href="#">Class Constant SKILL_FISHING</a>	182
<a href="#">Class Constant SKILL_FIST</a>	183
<a href="#">Class Constant SKILL_SHIELDING</a>	183
<a href="#">Class Constant SKILL_SWORD</a>	183
<a href="#">Class Constant SLOT_AMMO</a>	184

<a href="#">Class Constant SLOT_ARMOR</a>	184
<a href="#">Class Constant SLOT_BACKPACK</a>	185
<a href="#">Class Constant SLOT_FEET</a>	185
<a href="#">Class Constant SLOT_HEAD</a>	185
<a href="#">Class Constant SLOT_LEFT</a>	186
<a href="#">Class Constant SLOT_LEGS</a>	186
<a href="#">Class Constant SLOT_NECKLACE</a>	186
<a href="#">Class Constant SLOT_RIGHT</a>	187
<a href="#">Class Constant SLOT_RING</a>	187
<a href="#">Class Constant VOCATION_DRUID</a>	188
<a href="#">Class Constant VOCATION_KNIGHT</a>	188
<a href="#">Class Constant VOCATION_NONE</a>	188
<a href="#">Class Constant VOCATION_PALADIN</a>	188
<a href="#">Class Constant VOCATION_SORCERER</a>	189
<a href="#">Method connect</a>	189
<a href="#">example: connect.php</a>	189
<a href="#">Method createObject</a>	190
<a href="#">Method getInstance</a>	191
<a href="#">Method loadClass</a>	191
<a href="#">Method serverStatus</a>	192
<a href="#">example: example</a>	192
<a href="#">Method setPOTPath</a>	193
<a href="#">example: fakeroot.php</a>	193
<a href="#">compat.php</a>	195
<a href="#">Appendices</a>	196
<a href="#">Appendix A - Class Trees</a>	197
<a href="#">POT</a>	197
<a href="#">default</a>	200
<a href="#">Appendix B - README/CHANGELOG/INSTALL</a>	201
<a href="#">CHANGELOG</a>	202
<a href="#">README</a>	202
<a href="#">INSTALL</a>	203
<a href="#">NEWS</a>	204

# POT

*This is documentation 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 are (to be honest...) idiots - they don't know what PHP 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 information 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 doesn't provide any additional universal functionality. All it's classes are strictly connected with OTServ database.

## What about XML?

Sorry to say, XML guys - go out. OTServ will never leave XML - it is good to store some flat parts of database there. But not for main database which requires more advanced relationship between data. However of course maybe someone would want to create DB\_XML driver for POT? If you really are a masochist - you're welcome, we will be glad to contribute with you ;).

If you are interested in why XML so sux, and you with it, check out [OTFans thread](#).

## 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://www.otserv-aac.info/pot/" >
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>
46     <td>' . $player> getName(), '</td>
47 </tr>' ;
48     $first= false;
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 [LOTS\\_GuildAction interface](#). When the driver is assigned to guild object, each time a method of {@list OTS\_Guild 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:

# 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 OTServ 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.

# Package default Classes

## Class InvitesDriver *[line 18]*

- **Package** default

Constructor *void* function InvitesDriver::\_\_construct(\$guild) *[line 24]*

### **Function Parameters:**

- [OTS\\_Guild](#) **\$guild**

- **Access** public

*void* function InvitesDriver::addRequest(\$player) *[line 42]*

### **Function Parameters:**

- [OTS\\_Player](#) **\$player**

- **Access** public

*void function InvitesDriver::deleteRequest(\$player) [line 48]*

***Function Parameters:***

- [\*OTS\\_Player\*](#) **\$player**

- **Access** public

*void function InvitesDriver::listRequests() [line 32]*

- **Access** public

*void function InvitesDriver::submitRequest(\$player) [line 54]*

***Function Parameters:***

- [\*OTS\\_Player\*](#) **\$player**

- **Access** public



# Package POT Procedural Elements

## E\_OTS\_NoDriver.php

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



# E\_OTS\_NotLoaded.php

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

# IOTS\_DAO.php

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

# IOTS\_DB.php

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

# IOTS\_GuildAction.php

- **Package** POT
- **Author** Wrzasq < [wrzasq@gmail.com](mailto:wrzasq@gmail.com)>
- **Version** 0.0.3+SVN
- **Copyright** 2007 (C) by Wrzasq
- **Since** 0.0.3+SVN
- **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](mailto:wrzasq@gmail.com)>
- **Version** 0.0.1
- **Version** 0.0.3+SVN
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

# OTS\_Account.php

- **Package** POT
- **Author** Wrzasq < [wrzasq@gmail.com](mailto:wrzasq@gmail.com)>
- **Version** 0.0.1
- **Version** 0.0.3+SVN
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

# OTS\_Accounts\_List.php

- **Package** POT
- **Author** Wrzasq < [wrzasq@gmail.com](mailto:wrzasq@gmail.com)>
- **Version** 0.0.1
- **Version** 0.0.3
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

# OTS\_Container.php

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

# OTS\_DB\_ODBC.php

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

# OTS\_DB\_PostgreSQL.php

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

# OTS\_DB\_SQLite.php

- **Package** POT
- **Author** Wrzasq < [wrzasq@gmail.com](mailto:wrzasq@gmail.com) >
- **Version** 0.0.1
- **Version** 0.0.3+SVN
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

# OTS\_Group.php

- **Package** POT
- **Author** Wrzasq < [wrzasq@gmail.com](mailto:wrzasq@gmail.com)>
- **Version** 0.0.1
- **Version** 0.0.3
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

# OTS\_Groups\_List.php

- **Package** POT
- **Author** Wrzasq < [wrzasq@gmail.com](mailto:wrzasq@gmail.com)>
- **Version** 0.0.1
- **Version** 0.0.3
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

# OTS\_Guild.php

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

# OTS\_GuildRank.php

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



# OTS\_GuildRanks\_List.php

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

# OTS\_Guilds\_List.php

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

# OTS\_InfoRespond.php

- **Package** POT
- **Author** Wrzasq < [wrzasq@gmail.com](mailto: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](mailto: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\_Player.php

- **Package** POT
- **Author** Wrzasq < [wrzasq@gmail.com](mailto:wrzasq@gmail.com)>
- **Version** 0.0.1
- **Version** 0.0.3+SVN
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

# OTS\_Players\_List.php

- **Package** POT
- **Author** Wrzasq < [wrzasq@gmail.com](mailto:wrzasq@gmail.com)>
- **Version** 0.0.1
- **Version** 0.0.3
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

# OTS\_SQLite\_Results.php

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

# Package POT Classes

## 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** 0.0.3+SVN
- **Since** 0.0.3+SVN

## 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 IOTS\_DAO

*[line 21]*

### **OTServ database object.**

OTServ database object.

This interface indicates that class is a OTServ DAO class.

- **Package** POT
- **Version** 0.0.1

Constructor *void* function IOTS\_DAO::\_\_construct(\$db) *[line 28]*

#### **Function Parameters:**

- [\*IOTS\\_DB\*](#) **\$db** Database connection object.

### **DAO objects must be initialized with a database.**

DAO objects must be initialized with a database.

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

# Class IOTS\_DB

[line 21]

## **OTServ database handler interface.**

OTServ database handler interface.

This interface specifies routines requires by DAO classes.

- **Package** POT
- **Version** 0.0.1

Constructor *void* function IOTS\_DB::\_\_construct(\$params) [line 28]

### **Function Parameters:**

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

## **Connection parameters.**

Connection parameters.

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

*string* function IOTS\_DB::fieldName(\$name) [line 36]

### **Function Parameters:**

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

**Query-quoted field name.**  
Query-quoted field name.

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

*int* function IOTS\_DB::lastInsertId() [*line 63*]

**ID of last created record.**  
ID of last created record.

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

*string* function IOTS\_DB::limit([\$limit = false], [\$offset = false]) [*line 71*]

**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
- **Access** public

*mixed* function IOTS\_DB::SQLquery(\$query) [*line 57*]

**Function Parameters:**

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

**Evaluates query.**

Evaluates query.

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

*string* function IOTS\_DB::SQLquote(\$value) [*line 50*]

**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
- **Access** public

*string* function IOTS\_DB::tableName(\$name) [*line 43*]

**Function Parameters:**

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

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

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

## Class IOTS\_GuildAction

[line 24]

### **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.

- **Package** POT
- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN

Constructor *void* function IOTS\_GuildAction::\_\_construct(\$guild) [line 33]

#### **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** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function IOTS\_GuildAction::addRequest(\$player) [line 46]*

***Function Parameters:***

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

**Adds new request.**

Adds new request.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function IOTS\_GuildAction::deleteRequest(\$player) [line 52]*

***Function Parameters:***

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

**Deletes request.**

Deletes request.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*array* function IOTS\_GuildAction::listRequests() [*line 40*]

#### **List of saved pending actions.**

List of saved pending actions.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void* function IOTS\_GuildAction::submitRequest(\$player) [*line 58*]

#### **Function Parameters:**

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

#### **Finalizes request.**

Finalizes request.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

# Class OTS\_Account

[line 21]

## OTServ account abstraction.

OTServ account abstraction.

- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.3+SVN

Constructor *void* function OTS\_Account::\_\_construct(\$db) [line 42]

### **Function Parameters:**

- [\*IOTS\\_DB\*](#) **\$db** Database connection object.

## Sets database connection handler.

Sets database connection handler.

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

*void* function OTS\_Account::block() [line 328]

## **Blocks account.**

Blocks account.

- **Version** 0.0.1



- **Access public**

*int* function `OTS_Account::create([ $min = 1 ], [ $max = 9999999 ])` [line 67]

## 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->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 ?>

```

### 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!

**IMPORTANT:** Since 0.0.3+SVN there is `group_id` field which this method does not support. Account's `group_id` is set to first one found in database. You should use [createEx\(\)](#) method if you want to set `group_id` field during creation.

- **Version** 0.0.3+SVN
- **Version** 0.0.1
- **Throws** Exception When there are no free account numbers.
- **Access** public
- **Example**

*int* function `OTS_Account::createEx($group, [$min = 1], [$max = 9999999])` [*line 93*]  
**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 // group for account
18 $group = $ots->createObject('Group');
19
20 // loads group with id 1
21 $group->load(1);
22
23 // generates new account number
24 $number = $account->createEx($group);
25
26 // give user his number
27 echo 'Your account number is: ', $number;
28
29 ?>

```

#### **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!

- **Version** 0.0.3+SVN
- **Version** 0.0.1
- **Throws** Exception When there are no free account numbers.
- **Since** 0.0.3+SVN
- **Access** public
- **Example**

*void* function OTS\_Account::find(\$email) [*line 160*]

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

*string* function OTS\_Account::getCustomField(\$field) [*line 375*]

**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** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If account is not loaded.
- **Since** 0.0.3
- **Access** public

*string* function OTS\_Account::getEmail() [*line 280*]

### **E-mail address.**

E-mail address.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If account is not loaded.
- **Access** public

*OTS\_Group* function OTS\_Account::getGroup() [*line 224*]

### **Returns group of this account.**

Returns group of this account.

- **Version** 0.0.3+SVN
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If account is not loaded.
- **Since** 0.0.3+SVN
- **Access** public

*int* function OTS\_Account::getId() [*line 206*]

**Account number.**

Account number.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If account is not loaded.
- **Access** public

*int* function OTS\_Account::getPACCDays() [*line 341*]

**PACC days.**

PACC days.

- **Version** 0.0.3+SVN
- **Version** 0.0.1
- **Deprecated** 0.0.3 There is no more premdays field in accounts table.
- **Throws** E\_OTS\_NotLoaded If account is not loaded.
- **Access** public

*string* function OTS\_Account::getPassword() [*line 253*]

**Account's password.**

Account's password.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If account is not loaded.
- **Access** public

*array* function OTS\_Account::getPlayers() [*line 424*]

**List of characters on account.**

List of characters on account.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If account is not loaded.
- **Access** public

*bool* function OTS\_Account::isBlocked() [*line 307*]

**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.
- **Access** public

*bool* function OTS\_Account::isLoaded() [*line 177*]

**Checks if object is loaded.**

Checks if object is loaded.

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

*void* function OTS\_Account::load(\$id) [*line 147*]

**Function Parameters:**

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

**Loads account with given number.**

Loads account with given number.

- **Version** 0.0.3+SVN
- **Version** 0.0.1
- **Access** public

*void* function OTS\_Account::save() [*line 188*]

**Updates account in database.**

Updates account in database.

- **Version** 0.0.3+SVN

- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded False if account doesn't have ID assigned.
- **Access** public

*void function OTS\_Account::setCustomField(\$field, \$value) [line 401]*

**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** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If account is not loaded.
- **Since** 0.0.3
- **Access** public

*void function OTS\_Account::setEMail(\$email) [line 295]*

**Function Parameters:**

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



### **Sets account's email.**

Sets account's email.

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

*void function* OTS\_Account::setGroup(\$group) [*line 241*]

#### **Function Parameters:**

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

### **Assigns account to group.**

Assigns account to group.

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

*void function* OTS\_Account::setPACCDays(\$premdays, \$pacc) [*line 358*]

#### **Function Parameters:**

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

### **Sets PACC days count.**

Sets PACC days count.

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

*void function* OTS\_Account::setPassword(\$password) [*line 268*]

**Function Parameters:**

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

**Sets account's password.**

Sets account's password.

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

*void function* OTS\_Account::unblock() [*line 320*]

**Unblocks account.**

Unblocks account.

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

# Class OTS\_Accounts\_List

[line 21]

## List of accounts.

List of accounts.

- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.3

Constructor *void* function OTS\_Accounts\_List::\_\_construct(\$db) [line 56]

### **Function Parameters:**

- [\*IOTS\\_DB\*](#) **\$db** Database connection object.

## Sets database connection handler.

Sets database connection handler.

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

*int* function OTS\_Accounts\_List::count() [line 161]

## Returns number of accounts on list in current criterium.

Returns number of accounts on list in current criterium.

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

*OTS\_Account* function *OTS\_Accounts\_List::current()* [*line 111*]

**Returns current row.**

Returns current row.

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

*void* function *OTS\_Accounts\_List::deleteAccount(\$account)* [*line 101*]

**Function Parameters:**

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

**Deletes account.**

Deletes account.

- **Version** 0.0.3
- **Version** 0.0.1
- **Access** public

*mixed* function *OTS\_Accounts\_List::key()* [*line 133*]

**Current cursor position.**

Current cursor position.

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

*void function OTS\_Accounts\_List::next() [line 123]*

**Moves to next row.**

Moves to next row.

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

*void function OTS\_Accounts\_List::rewind() [line 151]*

**Select accounts from database.**

Select accounts from database.

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

*void function OTS\_Accounts\_List::setLimit([\$limit = false]) [line 66]*

**Function Parameters:**

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

**Sets LIMIT.**

Sets LIMIT.

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

*void* function OTS\_Accounts\_List::setOffset([\$offset = false]) [*line 83*]

**Function Parameters:**

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

**Sets OFFSET.**

Sets OFFSET.

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

*bool* function OTS\_Accounts\_List::valid() [*line 143*]

**Checks if there are any rows left.**

Checks if there are any rows left.

- **Version** 0.0.1
- **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 19]

**MySQL connection interface.**

MySQL connection interface.

- **Package** POT
- **Version** 0.0.1

Constructor *void* function OTS\_DB\_MySQL::\_\_construct(\$params) [line 46]

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

*string* function OTS\_DB\_MySQL::fieldName(\$name) [*line 101*]

**Function Parameters:**

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

**Query-quoted field name.**

Query-quoted field name.

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

*string* function OTS\_DB\_MySQL::limit([\$limit = false], [\$offset = false]) [*line 152*]

**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
- **Access** public

*PDOStatement|bool* function OTS\_DB\_MySQL::SQLquery(\$query) [*line 140*]

**Function Parameters:**

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

**IOTS\_DB method.**

IOTS\_DB method.

Overwrites PDO method.

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

*string* function OTS\_DB\_MySQL::SQLquote(\$string) [*line 126*]

**Function Parameters:**

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

**IOTS\_DB method.**

IOTS\_DB method.

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

- **Version** 0.0.1

- **Access** public

*string* function OTS\_DB\_MySQL::tableName(\$name) [*line 112*]

**Function Parameters:**

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

**Query-quoted table name.**

Query-quoted table name.

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

## Class OTS\_DB\_ODBC

[*line 20*]

**ODBC connection interface.**

ODBC connection interface.

- **Package** POT
- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN

Constructor *void* function OTS\_DB\_ODBC::\_\_construct(\$params) [*line 47*]

**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** 0.0.3+SVN
- **See** [POT::connect\(\)](#)
- **Since** 0.0.3+SVN
- **Access** public

*string* function OTS\_DB\_ODBC::fieldName(\$name) [*line 95*]

#### **Function Parameters:**

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

### **Query-quoted field name.**

Query-quoted field name.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*string* function OTS\_DB\_ODBC::limit([\$limit = false], [\$offset = false]) [*line 146*]

**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** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*PDOStatement|bool* function OTS\_DB\_ODBC::SQLquery(\$query) [*line 134*]

**Function Parameters:**

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

**IOTS\_DB method.**

IOTS\_DB method.

Overwrites PDO method.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*string* function OTS\_DB\_ODBC::SQLquote(\$string) [*line 120*]

**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** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*string* function OTS\_DB\_ODBC::tableName(\$name) [*line 106*]

**Function Parameters:**

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

**Query-quoted table name.**

Query-quoted table name.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public



# Class OTS\_DB\_PostgreSQL

[line 20]

## PostgreSQL connection interface.

PostgreSQL connection interface.

- **Package** POT
- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN

Constructor *void* function OTS\_DB\_PostgreSQL::\_\_construct(\$params) [line 47]

### **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** 0.0.3+SVN
- **See** [POT::connect\(\)](#)
- **Since** 0.0.3+SVN

- **Access** public

*string* function OTS\_DB\_PostgreSQL::fieldName(\$name) [*line 102*]

**Function Parameters:**

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

**Query-quoted field name.**

Query-quoted field name.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*string* function OTS\_DB\_PostgreSQL::limit([\$limit = false], [\$offset = false]) [*line 153*]

**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** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*PDOStatement|bool* function OTS\_DB\_PostgreSQL::SQLquery(\$query) [*line 141*]

**Function Parameters:**

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

**IOTS\_DB method.**

IOTS\_DB method.

Overwrites PDO method.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*string* function OTS\_DB\_PostgreSQL::SQLquote(\$string) [*line 127*]

**Function Parameters:**

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

**IOTS\_DB method.**

IOTS\_DB method.

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

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*string* function OTS\_DB\_PostgreSQL::tableName(\$name) [*line 113*]

#### **Function Parameters:**

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

#### **Query-quoted table name.**

Query-quoted table name.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

## Class OTS\_DB\_SQLite

[line 21]

#### **SQLite connection interface.**

SQLite connection interface.

- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.3+SVN

Constructor *void* function OTS\_DB\_SQLite::\_\_construct(\$params) [line 44]

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

*string* function OTS\_DB\_SQLite::fieldName(\$name) [*line 64*]

#### **Function Parameters:**

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

### **Query-quoted field name.**

Query-quoted field name.

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

*string* function OTS\_DB\_SQLite::limit([\$limit = false], [\$offset = false]) [*line 115*]

#### **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
- **Access** public

*PDOStatement|bool* function OTS\_DB\_SQLite::SQLquery(\$query) [*line 103*]

### **Function Parameters:**

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

## **IOTS\_DB method.**

IOTS\_DB method.

Overwrites PDO method.

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

*string* function OTS\_DB\_SQLite::SQLquote(\$string) [*line 89*]

### **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
- **Access** public

*string* function OTS\_DB\_SQLite::tableName(\$name) [*line 75*]

**Function Parameters:**

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

**Query-quoted table name.**

Query-quoted table name.

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

## Class OTS\_Group

[*line 21*]

**OTServ user group abstraction.**

OTServ user group abstraction.

- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.3

Constructor *void* function OTS\_Group::\_\_construct(\$db) [line 42]

**Function Parameters:**

- [\*IOTS\\_DB\*](#) \$db Database connection object.

**Sets database connection handler.**

Sets database connection handler.

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

*int* function OTS\_Group::getAccess() [line 167]

**Access level.**

Access level.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If group is not loaded.
- **Access** public

*string* function OTS\_Group::getCustomField(\$field) [line 254]

**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** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If group is not loaded.
- **Since** 0.0.3
- **Access** public

*int* function OTS\_Group::getFlags() [*line 140*]

#### **Rights flags.**

Rights flags.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If group is not loaded.
- **Access** public

*int* function OTS\_Group::getId() [*line 96*]

#### **Group ID.**

Group ID.

- **Version** 0.0.3

- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If group is not loaded.
- **Access** public

*int* function OTS\_Group::getMaxDepotItems() [*line 194*]

**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.
- **Access** public

*int* function OTS\_Group::getMaxVIPList() [*line 221*]

**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.
- **Access** public

*string* function OTS\_Group::getName() [*line 113*]

**Group name.**

Group name.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If group is not loaded.
- **Access** public

*array* function OTS\_Group::getPlayers() [*line 303*]

**List of characters in given group.**

List of characters in given group.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If group is not loaded.
- **Access** public

*bool* function OTS\_Group::isLoading() [*line 63*]

**Checks if object is loaded.**

Checks if object is loaded.

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

*void* function OTS\_Group::load(\$id) [*line 52*]

**Function Parameters:**

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

### **Loads group with given id.**

Loads group with given id.

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

*void function OTS\_Group::save() [line 71]*

### **Saves account in database.**

Saves account in database.

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

*void function OTS\_Group::setAccess(\$access) [line 182]*

#### ***Function Parameters:***

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

### **Sets access level.**

Sets access level.

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

*void function OTS\_Group::setCustomField(\$field, \$value) [line 280]*

**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** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If group is not loaded.
- **Since** 0.0.3
- **Access** public

*void function OTS\_Group::setFlags(\$flags) [line 155]*

**Function Parameters:**

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

**Sets rights flags.**

Sets rights flags.

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

*void function OTS\_Group::setMaxDepotItems(\$maxdepotitems) [line 209]*

**Function Parameters:**

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

**Sets maximum count of items in depot.**

Sets maximum count of items in depot.

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

*void function OTS\_Group::setMaxVIPList(\$maxviplist, \$maxdepotitems) [line 236]*

**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
- **Access** public

*void* function OTS\_Group::setName(\$name) [*line 128*]

**Function Parameters:**

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

**Sets group's name.**

Sets group's name.

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

## Class OTS\_Groups\_List

[*line 21*]

**List of groups.**

List of groups.

- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.3

Constructor *void* function OTS\_Groups\_List::\_\_construct(\$db) [*line 56*]

**Function Parameters:**

- [\*IOTS\\_DB\*](#) **\$db** Database connection object.

### **Sets database connection handler.**

Sets database connection handler.

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

*int* function OTS\_Groups\_List::count() [*line 161*]

### **Returns number of groups on list in current criterium.**

Returns number of groups on list in current criterium.

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

*OTS\_Group* function OTS\_Groups\_List::current() [*line 111*]

### **Returns current row.**

Returns current row.

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

*void* function OTS\_Groups\_List::deleteGroup(\$group) [*line 101*]

### **Function Parameters:**

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



### **Deletes group.**

Deletes group.

- **Version** 0.0.3
- **Version** 0.0.1
- **Access** public

*mixed* function OTS\_Groups\_List::key() [*line 133*]

### **Current cursor position.**

Current cursor position.

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

*void* function OTS\_Groups\_List::next() [*line 123*]

### **Moves to next row.**

Moves to next row.

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

*void* function OTS\_Groups\_List::rewind() [*line 151*]

### **Select groups from database.**

Select groups from database.

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

*void function* OTS\_Groups\_List::setLimit([\$limit = false]) [*line 66*]

**Function Parameters:**

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

**Sets LIMIT.**

Sets LIMIT.

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

*void function* OTS\_Groups\_List::setOffset([\$offset = false]) [*line 83*]

**Function Parameters:**

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

**Sets OFFSET.**

Sets OFFSET.

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

*bool* function OTS\_Groups\_List::valid() [*line 143*]

**Checks if there are any rows left.**

Checks if there are any rows left.

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

## Class OTS\_Guild

[*line 20*]

**OTServ guild abstraction.**

OTServ guild abstraction.

- **Package** POT
- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN

Constructor *void* function OTS\_Guild::\_\_construct(\$db) [*line 55*]

**Function Parameters:**

- [\*IOTS\\_DB\*](#) **\$db** Database connection object.

**Sets database connection handler.**

Sets database connection handler.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_Guild::acceptInvite(\$player) [line 388]*

**Function Parameters:**

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

**Finalise invitation.**

Finalise invitation.

- **Version** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Throws** E\_OTS\_NoDriver If there is no invites driver assigned.
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_Guild::acceptRequest(\$player) [line 480]*

**Function Parameters:**

- [\*OTS\\_Player\*](#) **\$player** Player to be accepted.

**Accepts player.**

Accepts player.

- **Version** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Throws** E\_OTS\_NoDriver If there is no requests driver assigned.
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_Guild::deleteInvite(\$player) [line 365]*

**Function Parameters:**

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

**Deletes invitation for player to guild.**

Deletes invitation for player to guild.

- **Version** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Throws** E\_OTS\_NoDriver If there is no invites driver assigned.
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_Guild::deleteRequest(\$player) [line 457]*

**Function Parameters:**

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

**Deletes request from player.**

Deletes request from player.

- **Version** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Throws** E\_OTS\_NoDriver If there is no requests driver assigned.
- **Since** 0.0.3+SVN
- **Access** public

*void* function OTS\_Guild::find(\$name) [*line 96*]

**Function Parameters:**

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

**Loads guild by it's name.**

Loads guild by it's name.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*int* function OTS\_Guild::getCreationData() [*line 215*]

**Guild creation data.**

Guild creation data.

- **Version** 0.0.3+SVN

- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Since** 0.0.3+SVN
- **Access** public

*string* function OTS\_Guild::getCustomField(\$field) [*line 246*]

**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** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Since** 0.0.3+SVN
- **Access** public

*array* function OTS\_Guild::getGuildRanks() [*line 292*]

**Reads all ranks that are in this guild.**

Reads all ranks that are in this guild.

- **Version** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Since** 0.0.3+SVN

- **Access** public

*int* function OTS\_Guild::getId() [*line 145*]

**Guild ID.**

Guild ID.

- **Version** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Since** 0.0.3+SVN
- **Access** public

*string* function OTS\_Guild::getName() [*line 161*]

**Guild name.**

Guild name.

- **Version** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Since** 0.0.3+SVN
- **Access** public

*OTS\_Player* function OTS\_Guild::getOwner() [*line 187*]

**Returns owning player of this player.**

Returns owning player of this player.

- **Version** 0.0.3+SVN



- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Since** 0.0.3+SVN
- **Access** public

*void* function OTS\_Guild::invite(\$player) [*line 342*]

**Function Parameters:**

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

**Invites player to guild.**

Invites player to guild.

- **Version** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Throws** E\_OTS\_NoDriver If there is no invites driver assigned.
- **Since** 0.0.3+SVN
- **Access** public

*bool* function OTS\_Guild::isLoaded() [*line 113*]

**Checks if object is loaded.**

Checks if object is loaded.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*array* function OTS\_Guild::listInvites() [*line 319*]

**Returns list of invited players.**

Returns list of invited players.

- **Version** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Throws** E\_OTS\_NoDriver If there is no invites driver assigned.
- **Since** 0.0.3+SVN
- **Access** public

*array* function OTS\_Guild::listRequests() [*line 411*]

**Returns list of players that requested membership.**

Returns list of players that requested membership.

- **Version** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Throws** E\_OTS\_NoDriver If there is no requests driver assigned.
- **Since** 0.0.3+SVN
- **Access** public

*void* function OTS\_Guild::load(\$id) [*line 85*]

**Function Parameters:**

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

**Loads guild with given id.**

Loads guild with given id.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_Guild::request(\$player) [line 434]*

**Function Parameters:**

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

**Requests membership in guild for player player.**  
Requests membership in guild for player player.

- **Version** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Throws** E\_OTS\_NoDriver If there is no requests driver assigned.
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_Guild::save() [line 121]*

**Saves guild in database.**  
Saves guild in database.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN

- **Access** public

*void function OTS\_Guild::setCreationData(\$creationdata) [line 230]*

**Function Parameters:**

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

**Sets guild creation data.**

Sets guild creation data.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_Guild::setCustomField(\$field, \$value) [line 270]*

**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** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If guild is not loaded.
- **Since** 0.0.3+SVN
- **Access** public

*void function* OTS\_Guild::setInvitesDriver([\$invites = null]) [*line 65*]

**Function Parameters:**

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

**Assigns invites handler.**

Assigns invites handler.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function* OTS\_Guild::setName(\$name) [*line 176*]

**Function Parameters:**

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

**Sets players's name.**

Sets players's name.

- **Version** 0.0.3+SVN

- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_Guild::setOwner(\$owner) [line 204]*

**Function Parameters:**

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

**Assigns guild to owner.**

Assigns guild to owner.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_Guild::setRequestsDriver([\$requests = null]) [line 75]*

**Function Parameters:**

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

**Assigns requests handler.**

Assigns requests handler.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

# Class OTS\_GuildRank

[line 20]

**OTServ guild rank abstraction.**

OTServ guild rank abstraction.

- **Package** POT
- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN

Constructor *void* function OTS\_GuildRank::\_\_construct(\$db) [line 41]

**Function Parameters:**

- [\*IOTS\\_DB\*](#) **\$db** Database connection object.

**Sets database connection handler.**

Sets database connection handler.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void* function OTS\_GuildRank::find(\$name, [\$guild = null]) [line 65]

**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** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*string* function OTS\_GuildRank::getCustomField(\$field) [*line 223*]

#### **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** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If rank is not loaded.
- **Since** 0.0.3+SVN
- **Access** public



*OTS\_Guild* function *OTS\_GuildRank::getGuild()* [*line 164*]

**Returns guild of this rank.**

Returns guild of this rank.

- **Version** 0.0.3+SVN
- **Throws** *E\_OTS\_NotLoaded* If rank is not loaded.
- **Since** 0.0.3+SVN
- **Access** public

*int* function *OTS\_GuildRank::getId()* [*line 122*]

**Rank ID.**

Rank ID.

- **Version** 0.0.3+SVN
- **Throws** *E\_OTS\_NotLoaded* If rank is not loaded.
- **Since** 0.0.3+SVN
- **Access** public

*int* function *OTS\_GuildRank::getLevel()* [*line 192*]

**Rank's access level.**

Rank's access level.

- **Version** 0.0.3+SVN
- **Throws** *E\_OTS\_NotLoaded* If rank is not loaded.

- **Since** 0.0.3+SVN
- **Access** public

*string* function OTS\_GuildRank::getName() [*line 138*]

**Rank name.**

Rank name.

- **Version** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If rank is not loaded.
- **Since** 0.0.3+SVN
- **Access** public

*array* function OTS\_GuildRank::getPlayers() [*line 269*]

**Reads all players who has this rank set.**

Reads all players who has this rank set.

- **Version** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If rank is not loaded.
- **Since** 0.0.3+SVN
- **Access** public

*bool* function OTS\_GuildRank::isLoaded() [*line 90*]

**Checks if object is loaded.**

Checks if object is loaded.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_GuildRank::load(\$id) [line 51]*

**Function Parameters:**

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

**Loads rank with given id.**

Loads rank with given id.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_GuildRank::save() [line 98]*

**Saves rank in database.**

Saves rank in database.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_GuildRank::setCustomField(\$field, \$value) [line 247]*

**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** 0.0.3+SVN
- **Throws** E\_OTS\_NotLoaded If rank is not loaded.
- **Since** 0.0.3+SVN
- **Access** public

*void* function OTS\_GuildRank::setGuild(\$guild) [*line 181*]

#### **Function Parameters:**

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

### **Assigns rank to guild.**

Assigns rank to guild.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN

- **Access** public

*void function OTS\_GuildRank::setLevel(\$level) [line 207]*

**Function Parameters:**

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

**Sets rank's access level within guild.**

Sets rank's access level within guild.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_GuildRank::setName(\$name) [line 153]*

**Function Parameters:**

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

**Sets rank's name.**

Sets rank's name.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

# Class OTS\_GuildRanks\_List

[line 20]

## List of guild ranks.

List of guild ranks.

- **Package** POT
- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN

Constructor *void* function OTS\_GuildRanks\_List::\_\_construct(\$db) [line 55]

### **Function Parameters:**

- [\*IOTS\\_DB\*](#) **\$db** Database connection object.

## Sets database connection handler.

Sets database connection handler.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*int* function OTS\_GuildRanks\_List::count() [line 159]

## Returns number of ranks on list in current criterium.

Returns number of ranks on list in current criterium.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*OTS\_GuildRank* function OTS\_GuildRanks\_List::current() [*line 109*]

**Returns current row.**

Returns current row.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void* function OTS\_GuildRanks\_List::deleteGuildRank(\$guildRank) [*line 99*]

**Function Parameters:**

- [\*OTS\\_GuildRank\*](#) **\$guildRank** Rank to be deleted.

**Deletes guild rank.**

Deletes guild rank.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*mixed* function OTS\_GuildRanks\_List::key() [*line 131*]

**Current cursor position.**  
Current cursor position.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_GuildRanks\_List::next() [line 121]*

**Moves to next row.**

Moves to next row.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_GuildRanks\_List::rewind() [line 149]*

**Select ranks from database.**

Select ranks from database.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_GuildRanks\_List::setLimit([\$limit = false]) [line 65]*

**Function Parameters:**



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

### **Sets LIMIT.**

Sets LIMIT.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void* function OTS\_GuildRanks\_List::setOffset([\$offset = false]) [*line 82*]

#### **Function Parameters:**

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

### **Sets OFFSET.**

Sets OFFSET.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*bool* function OTS\_GuildRanks\_List::valid() [*line 141*]

#### **Checks if there are any rows left.**

Checks if there are any rows left.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

## Class OTS\_Guilds\_List

*[line 20]*

**List of guilds.**  
List of guilds.

- **Package** POT
- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN

Constructor *void* function OTS\_Guilds\_List::\_\_construct(\$db) *[line 55]*  
**Function Parameters:**

- [\*IOTS\\_DB\*](#) **\$db** Database connection object.

**Sets database connection handler.**  
Sets database connection handler.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*int* function OTS\_Guilds\_List::count() [*line 159*]

**Returns number of guilds on list in current criterium.**

Returns number of guilds on list in current criterium.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*OTS\_Guild* function OTS\_Guilds\_List::current() [*line 109*]

**Returns current row.**

Returns current row.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void* function OTS\_Guilds\_List::deleteGuild(\$guild) [*line 99*]

**Function Parameters:**

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

**Deletes guild.**

Deletes guild.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*mixed* function OTS\_Guilds\_List::key() [*line 131*]

**Current cursor position.**

Current cursor position.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void* function OTS\_Guilds\_List::next() [*line 121*]

**Moves to next row.**

Moves to next row.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void* function OTS\_Guilds\_List::rewind() [*line 149*]

**Select guilds from database.**

Select guilds from database.

- **Version** 0.0.3+SVN

- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_Guilds\_List::setLimit([\$limit = false]) [line 65]*

***Function Parameters:***

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

**Sets LIMIT.**

Sets LIMIT.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*void function OTS\_Guilds\_List::setOffset([\$offset = false]) [line 82]*

***Function Parameters:***

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

**Sets OFFSET.**

Sets OFFSET.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **Access** public

*bool* function OTS\_Guilds\_List::valid() [*line 141*]

**Checks if there are any rows left.**

Checks if there are any rows left.

- **Version** 0.0.3+SVN
- **Since** 0.0.3+SVN
- **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\_Player**

[*line 21*]

### **OTServ character abstraction.**

OTServ character abstraction.



- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.3+SVN

Constructor *void* function OTS\_Player::\_\_construct(\$db) [*line 52*]

**Function Parameters:**

- [\*IOTS\\_DB\*](#) **\$db** Database connection object.

**Sets database connection handler.**

Sets database connection handler.

- **Version** 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** 0.0.1
- **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.
- **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.
- **Access** public

*mixed* function OTS\_Player::getConditions() [*line 955*]

**Conditions.**

Conditions.

- **Version** 0.0.3
- **Version** 0.0.1

- **Throws** E\_OTS\_NotLoaded If player is not loaded.
- **Access** public

*string* function OTS\_Player::getCustomField(\$field) [line 1254]

**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** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If player is not loaded.
- **Since** 0.0.3
- **Access** public

*OTS\_Item|null* function OTS\_Player::getDepot(\$depot) [line 1523]

**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** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If player is not loaded.
- **Since** 0.0.3
- **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.
- **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.
- **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.
- **Access** public

*string* function OTS\_Player::getGuildNick() [*line 1042*]

**Guild nick.**

Guild nick.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If player is not loaded.
- **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.
- **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.
- **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.
- **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\_OTTS\_NotLoaded If player is not loaded.
- **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\_OTTS\_NotLoaded If player is not loaded.
- **Access** public

*int* function OTS\_Player::getLevel() [*line 355*]

### **Experience level.**

Experience level.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTTS\_NotLoaded If player is not loaded.
- **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.
- **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.
- **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.
- **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.
- **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.
- **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.
- **Access** public

*int* function OTS\_Player::getLossExperience() [*line 1165*]

**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.
- **Access** public

*int* function OTS\_Player::getLossMana() [*line 1192*]

**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.
- **Access** public

*int* function OTS\_Player::getLossSkills() [*line 1219*]

**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.
- **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.
- **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.
- **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.
- **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.
- **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.
- **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.
- **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.
- **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.
- **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
- **Access** public

*OTS\_GuildRank|null* function OTS\_Player::getRank() [*line 1086*]

**Assigned guild rank.**

Assigned guild rank.

- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If player is not loaded.
- **Access** public

*int* function OTS\_Player::getRankId() [*line 1070*]

**Guild rank ID.**

Guild rank ID.

- **Version** 0.0.3
- **Version** 0.0.1
- **Deprecated** 0.0.3+SVN Use getRank().
- **Throws** E\_OTS\_NotLoaded If player is not loaded.

- **Access** public

*int* function OTS\_Player::getRedSkullTime() [*line 982*]

**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.
- **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.
- **Access** public

*int* function OTS\_Player::getSkill(\$skill) [*line 1309*]

**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
- **Access** public

*int* function OTS\_Player::getSkillTries(\$skill) [*line 1341*]

**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
- **Access** public

*OTS\_Item|null* function OTS\_Player::getSlot(\$slot) [*line 1394*]

**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** 0.0.3+SVN
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If player is not loaded.
- **Since** 0.0.3
- **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.
- **Access** public

*int* function OTS\_Player::getTownId() [*line 1138*]

### 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.
- **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.
- **Access** public

*bool* function OTS\_Player::hasRedSkull() [*line 1009*]

**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.
- **Access** public

*bool* function OTS\_Player::isLoading() [*line 101*]

**Checks if object is loaded.**

Checks if object is loaded.

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

*bool* function OTS\_Player::isSaveSet() [*line 922*]

**Checks if save flag is set.**

Checks if save flag is set.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If player is not loaded.
- **Access** public

*void* function OTS\_Player::load(\$id) [*line 63*]

**Function Parameters:**

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

**Loads player with given id.**

Loads player with given id.

- **Version** 0.0.2
- **Version** 0.0.1
- **Access** public

*void* function OTS\_Player::save() [*line 111*]

**Saves player in database.**

Saves player in database.

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

*void* function OTS\_Player::setCap(\$cap) [*line 856*]

**Function Parameters:**

- *int* **\$cap** Capacity.

**Sets capacity.**

Sets capacity.

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

*void function OTS\_Player::setConditions(\$conditions) [line 970]*

**Function Parameters:**

- *mixed* **\$conditions** Condition binary field.

**Sets conditions.**

Sets conditions.

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

*void function OTS\_Player::setCustomField(\$field, \$value) [line 1284]*

**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** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If player is not loaded.
- **Since** 0.0.3
- **Access** public

*void function OTS\_Player::setDepot(\$depot, [\$item = null], [\$pid = 0], [\$depot\_id = 0]) [line 1578]*

**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** For internal recursive insertion.
- *int* **\$depot\_id** Internal, for further use.

**Sets slot content.**

Sets slot content.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If player is not loaded.
- **Since** 0.0.3
- **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
- **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
- **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
- **Access** public

*void function OTS\_Player::setGuildNick(\$guildnick) [line 1057]*

***Function Parameters:***

- *string* **\$guildnick** Name.

**Sets guild nick.**

Sets guild nick.

- **Version** 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
- **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
- **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
- **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
- **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
- **Access** public

*void function OTS\_Player::setLookAddons(\$lookaddons) [line 748]*

#### ***Function Parameters:***

- *int* **\$lookaddons** Addons.

### **Sets addons.**

Sets addons.

- **Version** 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
- **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
- **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
- **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
- **Access** public

*void function OTS\_Player::setLookType(\$looktype) [line 721]*

**Function Parameters:**

- *int* **\$looktype** Outfit.

**Sets outfit.**

Sets outfit.

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

*void function OTS\_Player::setLossExperience(\$loss\_experience) [line 1180]*

**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
- **Access** public

*void function OTS\_Player::setLossMana(\$loss\_mana) [line 1207]*

**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

- **Access** public

*void function OTS\_Player::setLossSkills(\$loss\_skills) [line 1234]*

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

*void function OTS\_Player::setRank([\$guildRank = null]) [line 1119]*

#### ***Function Parameters:***

- [\*OTS\\_GuildRank\*](#)*|null* **\$guildRank** Guild rank (null to clear assign).

### **Assigns guild rank.**

Assigns guild rank.

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

*void function OTS\_Player::setRankId(\$rank\_id) [line 1109]*

#### ***Function Parameters:***

- *int* **\$rank\_id** Guild rank ID.

### **Sets guild rank ID.**

Sets guild rank ID.

- **Version** 0.0.1
- **Deprecated** 0.0.3+SVN Use setRank().
- **Access** public

*void function OTS\_Player::setRedSkull() [line 1030]*

#### **Sets red skull flag.**

Sets red skull flag.

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

*void function OTS\_Player::setRedSkullTime(\$redskulltime) [line 997]*

#### **Function Parameters:**

- *int* **\$redskulltime** Red skulled time remained.

#### **Sets red skulled time remained.**

Sets red skulled time remained.

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

*void function OTS\_Player::setSave() [line 943]*

#### **Sets save flag.**

Sets save flag.

- **Version** 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
- **Access** public

*void function OTS\_Player::setSkill(\$skill, \$value) [line 1327]*

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

void function OTS\_Player::setSkillTries(\$skill, \$tries) [*line 1359*]

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

void function OTS\_Player::setSlot(\$slot, [\$item = null], [\$pid = 0]) [*line 1448*]

**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** For internal use in case of containers.

**Sets slot content.**

Sets slot content.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E\_OTS\_NotLoaded If player is not loaded.
- **Since** 0.0.3
- **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
- **Access** public

*void function OTS\_Player::setTownId(\$town\_id) [line 1153]*

***Function Parameters:***

- *int* **\$town\_id** Residence town's ID.

**Sets residence town's ID.**

Sets residence town's ID.

- **Version** 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
- **Access** public

*void function OTS\_Player::unsetRedSkull() [line 1022]*

**Unsets red skull flag.**

Unsets red skull flag.

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

*void function OTS\_Player::unsetSave() [line 935]*

**Unsets save flag.**

Unsets save flag.

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

## Class OTS\_Players\_List

*[line 21]*

### List of players.

List of players.

- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.3

Constructor *void* function OTS\_Players\_List::\_\_construct(\$db) *[line 56]*

#### **Function Parameters:**

- [\*IOTS\\_DB\*](#) **\$db** Database connection object.

### Sets database connection handler.

Sets database connection handler.

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

*int* function OTS\_Players\_List::count() *[line 161]*



**Returns number of characters on list in current criterium.**

Returns number of characters on list in current criterium.

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

*OTS\_Player* function OTS\_Players\_List::current() [*line 111*]

**Returns current row.**

Returns current row.

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

*void* function OTS\_Players\_List::deletePlayer(\$player) [*line 101*]

**Function Parameters:**

- [\*OTS\\_Player\*](#) **\$player** Player to be deleted.

**Deletes player.**

Deletes player.

- **Version** 0.0.3
- **Version** 0.0.1
- **Access** public

*mixed* function OTS\_Players\_List::key() [*line 133*]

**Current cursor position.**

Current cursor position.

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

*void* function OTS\_Players\_List::next() [*line 123*]

**Moves to next row.**

Moves to next row.

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

*void* function OTS\_Players\_List::rewind() [*line 151*]

**Select players from database.**

Select players from database.

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

*void* function OTS\_Players\_List::setLimit([\$limit = false]) [*line 66*]

**Function Parameters:**

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

### Sets LIMIT.

Sets LIMIT.

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

*void* function OTS\_Players\_List::setOffset([\$offset = false]) [*line 83*]

#### **Function Parameters:**

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

### Sets OFFSET.

Sets OFFSET.

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

*bool* function OTS\_Players\_List::valid() [*line 143*]

#### **Checks if there are any rows left.**

Checks if there are any rows left.

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

# Class POT

*[line 23]*

## **Main POT class.**

Main POT class.

- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.3+SVN

## **POT::DB\_MYSQL**

= 1 *[line 28]*

## **MySQL driver.**

MySQL driver.

- **Version** 0.0.1

## **POT::DB\_ODBC**

= 4 *[line 46]*

## **ODBC driver.**

ODBC driver.

- **Version** 0.0.3+SVN
- **Version** 0.0.1
- **Since** 0.0.3+SVN

#### **POT::DB\_PGSQL**

= 3 [*line 39*]

#### **PostgreSQL driver.**

PostgreSQL driver.

- **Version** 0.0.3+SVN
- **Version** 0.0.1
- **Since** 0.0.3+SVN

#### **POT::DB\_SQLITE**

= 2 [*line 32*]

#### **SQLite driver.**

SQLite driver.

- **Version** 0.0.1

#### **POT::DIRECTION\_EAST**

= 1 [*line 85*]

#### **East.**

East.

- **Version 0.0.1**

**POT::DIRECTION\_NORTH**

= 0 *[line 81]*

**North.**  
North.

- **Version 0.0.1**

**POT::DIRECTION\_SOUTH**

= 2 *[line 89]*

**South.**  
South.

- **Version 0.0.1**

**POT::DIRECTION\_WEST**

= 3 *[line 93]*

**West.**  
West.

- **Version 0.0.1**

## POT::SEX\_FEMALE

= 0 *[line 51]*

### **Female gender.**

Female gender.

- **Version** 0.0.1

## POT::SEX\_MALE

= 1 *[line 55]*

### **Male gender.**

Male gender.

- **Version** 0.0.1

## POT::SKILL\_AXE

= 3 *[line 122]*

### **Axe fighting.**

Axe fighting.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.2

## POT::SKILL\_CLUB

= 1 *[line 108]*

**Club fighting.**  
Club fighting.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.2

## POT::SKILL\_DISTANCE

= 4 *[line 129]*

**Distance fighting.**  
Distance fighting.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.2

## POT::SKILL\_FISHING

= 6 *[line 143]*

**Fishing.**  
Fishing.



- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.2

#### **POT::SKILL\_FIST**

= 0 [*line 101*]

**Fist fighting.**  
Fist fighting.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.2

#### **POT::SKILL\_SHIELDING**

= 5 [*line 136*]

**Shielding.**  
Shielding.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.2

#### **POT::SKILL\_SWORD**

= 2 [*line 115*]

**Sword fighting.**

Sword fighting.

- **Version** 0.0.2
- **Version** 0.0.1
- **Since** 0.0.2

#### **POT::SLOT\_AMMO**

= 10 [*line 214*]

**Ammunition slot.**  
Ammunition slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.3

#### **POT::SLOT\_ARMOR**

= 4 [*line 172*]

**Armor slot.**  
Armor slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.3

## POT::SLOT\_BACKPACK

= 3 *[line 165]*

### Backpack slot.

Backpack slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.3

## POT::SLOT\_FEET

= 8 *[line 200]*

### Boots slot.

Boots slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.3

## POT::SLOT\_HEAD

= 1 *[line 151]*

### Head slot.

Head slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.3

#### **POT::SLOT\_LEFT**

= 6 [*line 186*]

#### **Left hand slot.**

Left hand slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.3

#### **POT::SLOT\_LEGS**

= 7 [*line 193*]

#### **Legs slot.**

Legs slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.3

#### **POT::SLOT\_NECKLACE**

= 2 [*line 158*]

#### **Necklace slot.**

Necklace slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.3

**POT::SLOT\_RIGHT**

= 5 [*line 179*]

**Right hand slot.**  
Right hand slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.3

**POT::SLOT\_RING**

= 9 [*line 207*]

**Ring slot.**  
Ring slot.

- **Version** 0.0.3
- **Version** 0.0.1
- **Since** 0.0.3

## POT::VOCATION\_DRUID

= 2 *[line 68]*

### Druid.

Druid.

- **Version 0.0.1**

## POT::VOCATION\_KNIGHT

= 4 *[line 76]*

### Knight.

Knight.

- **Version 0.0.1**

## POT::VOCATION\_NONE

= 0 *[line 60]*

### None vocation.

None vocation.

- **Version 0.0.1**

## POT::VOCATION\_PALADIN

= 3 *[line 72]*

**Paladin.**  
Paladin.

- **Version 0.0.1**

**POT::VOCATION\_SORCERER**

= 1 *[line 64]*

**Sorcerer.**  
Sorcerer.

- **Version 0.0.1**

*void function POT::connect(\$driver, \$params) [line 338]*

**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' => 'otserv'
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 then one OTServ installed on one database.

- **Version** 0.0.1
- **Version** 0.0.3+SVN
- **Throws** Exception When driver is not supported.
- **Access** public
- **Example**

*LOTS\_DAO* function POT::createObject(\$class) [*line 389*]

### **Function Parameters:**

- *string* **\$class** Class name.

### **Creates OTServ DAO class instance.**

Creates OTServ DAO class instance.



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

POT function POT::getInstance() [*line 221*]

### **Singleton.**

Singleton.

- **Version** 0.0.1
- **Static**
- **Access** public

void function POT::loadClass(\$class) [*line 297*]

### **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
- **Access** public
- **Example** example not found

OTS\_InfoRespond|bool function POT::serverStatus(\$server, \$port) [line 407]  
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 0.0.1**
- **Version 0.0.2**
- **Since 0.0.2**
- **Access public**
- **Example**

void function POT::setPOTPath(\$path) [line 252]

## 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**
- **Access public**
- **Example**



## 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](mailto:wrzasq@gmail.com)>
- **Version** 0.0.2
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

# Appendices

# Appendix A - Class Trees

## Package POT

### E\_OTS\_NoDriver

- Exception
  - [E\\_OTS\\_NoDriver](#)

### E\_OTS\_NotLoaded

- Exception
  - [E\\_OTS\\_NotLoaded](#)

### IOTS\_DAO

- [IOTS\\_DAO](#)

### IOTS\_DB

- [IOTS\\_DB](#)

### IOTS\_GuildAction

- [IOTS\\_GuildAction](#)

### OTS\_Account

- [OTS\\_Account](#)

## OTS\_Accounts\_List

- [OTS\\_Accounts\\_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\_Group

- [OTS\\_Group](#)



## OTS\_Groups\_List

- [OTS\\_Groups\\_List](#)

## OTS\_Guild

- [OTS\\_Guild](#)

## OTS\_GuildRank

- [OTS\\_GuildRank](#)

## OTS\_GuildRanks\_List

- [OTS\\_GuildRanks\\_List](#)

## OTS\_Guilds\_List

- [OTS\\_Guilds\\_List](#)

## OTS\_InfoRespond

- DOMDocument
  - [OTS\\_InfoRespond](#)

## OTS\_Item

- [OTS\\_Item](#)
  - [OTS\\_Container](#)

## OTS\_Player

- [OTS\\_Player](#)

## OTS\_Players\_List

- [OTS\\_Players\\_List](#)

## POT

- [POT](#)

Package default

## InvitesDriver

- [InvitesDriver](#)

# Appendix B - README/CHANGELOG/INSTALL

# CHANGELOG

## [0.0.3+SVN]

- \* 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>
- \* Updated players table structure. <wrzasq>
- \* Dropped REGEXP operator bindings - not used anywhere. <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>

# 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 language 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.  
BUGS - known bugs.  
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.  
TODO - list of things to be done.  
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.zip file for distribution purposes.

For more readable output of phpUnit test run:  
php test.php

#### ===== Credits =====

\* Wrzasq <wrzasq@gmail.com> - project initiator, main developer.

## 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 0.0.3 version?

- \* Added custom fields support.

You can now use POT with non-standard SVN database structure (however it is not as comfortable as with standard SVN fields). You have to save your standard record before saving custom fields.

- \* Added items and depots support.

OTS\_Item and OTS\_Container classes. OTS\_Player now has getSlot(), setSlot(), getDepot(), setDepot() methods. You can manage items tables as objects trees.

- \* Added support for players PACC timestamps.

In current OTServ SVN premium time is not stored in accounts table, but in players table also not as days, but as ending moment timestamp. Account PACC methods are now obsolete.

- \* Fixed loading skills.

Small typo.

- \* Replaced E\_USER\_\* with exceptions.

No more error messages between text on website, everything is now thrown as exceptions.

- \* Uses fetchAll() in loops to prevent MySQL buffering problems.

PDO is really fucked up in some places and MySQL driver queries buffering is one of them. This change should prevent POT from producing some errors in very particular situations.

# Index

## A

[Account number hack](#) . . . . . 16

## C

[constructor OTS Guilds List:: construct\(\)](#) . . . . . 120  
    *Sets database connection handler.*  
[constructor OTS GuildRanks List:: construct\(\)](#) . . . . . 116  
    *Sets database connection handler.*  
[constructor OTS GuildRank:: construct\(\)](#) . . . . . 109  
    *Sets database connection handler.*  
[constructor OTS Guild:: construct\(\)](#) . . . . . 97  
    *Sets database connection handler.*  
[constructor OTS Item:: construct\(\)](#) . . . . . 132  
    *Creates item of given ID.*  
[constructor OTS Player:: construct\(\)](#) . . . . . 135  
    *Sets database connection handler.*  
[CHANGELOG](#) . . . . . 202  
[compat.php](#) . . . . . 195  
    *POT compatibility assurance package.*  
[constructor OTS Players List:: construct\(\)](#) . . . . . 174  
    *Sets database connection handler.*  
[constructor OTS Groups List:: construct\(\)](#) . . . . . 93  
    *Sets database connection handler.*  
[constructor OTS Group:: construct\(\)](#) . . . . . 86  
    *Sets database connection handler.*  
[constructor OTS Account:: construct\(\)](#) . . . . . 54  
    *Sets database connection handler.*  
[constructor IOTS GuildAction:: construct\(\)](#) . . . . . 51  
    *Objects are initialized with a guild that they are assigned to.*  
[constructor IOTS DB:: construct\(\)](#) . . . . . 48  
    *Connection parameters.*  
[constructor IOTS DAO:: construct\(\)](#) . . . . . 47  
    *DAO objects must be initialized with a database.*  
[constructor OTS Accounts List:: construct\(\)](#) . . . . . 65  
    *Sets database connection handler.*  
[constructor OTS DB MySQL:: construct\(\)](#) . . . . . 72  
    *Creates database connection.*  
[constructor OTS DB SQLite:: construct\(\)](#) . . . . . 82  
    *Creates database connection.*  
[constructor OTS DB PostgreSQL:: construct\(\)](#) . . . . . 79  
    *Creates database connection.*  
[constructor OTS DB ODBC:: construct\(\)](#) . . . . . 75  
    *Creates database connection.*

<a href="#">constructor InvitesDriver::construct()</a>	19
--------------------------------------------------------	----

## D

<a href="#">DAO objects</a>	9
-----------------------------	---

## E

<a href="#">E_OTS_NotLoaded</a>	46
<i>Occurs when code attempts to access property of not loaded object.</i>	
<a href="#">E_OTS_NoDriver</a>	46
<i>Occurs when code attempts to execute driven action that has no assigned driver to handle it.</i>	
<a href="#">E_OTS_NotLoaded.php</a>	23
<a href="#">E_OTS_NoDriver.php</a>	22

## G

<a href="#">Guild action drivers</a>	15
<a href="#">Guilds</a>	13

## I

<a href="#">IOTS_DB::tableName()</a>	50
<i>Query-quoted table name.</i>	
<a href="#">IOTS_DB::SQLquote()</a>	50
<i>Query-quoted string value.</i>	
<a href="#">IOTS_DB::SQLquery()</a>	50
<i>Evaluates query.</i>	
<a href="#">IOTS_DB::limit()</a>	49
<i>LIMIT/OFFSET clause for queries.</i>	
<a href="#">IOTS_GuildAction</a>	51
<i>Guild action interface.</i>	
<a href="#">IOTS_GuildAction::addRequest()</a>	52
<i>Adds new request.</i>	
<a href="#">INSTALL</a>	203
<a href="#">IOTS_GuildAction::submitRequest()</a>	53
<i>Finalizes request.</i>	
<a href="#">IOTS_GuildAction::listRequests()</a>	53
<i>List of saved pending actions.</i>	
<a href="#">IOTS_GuildAction::deleteRequest()</a>	52
<i>Deletes request.</i>	
<a href="#">IOTS_DB::lastInsertId()</a>	49
<i>ID of last created record.</i>	
<a href="#">IOTS_DB::fieldName()</a>	48
<i>Query-quoted field name.</i>	
<a href="#">InvitesDriver::submitRequest()</a>	20
<a href="#">InvitesDriver::listRequests()</a>	20
<a href="#">InvitesDriver::deleteRequest()</a>	20
<a href="#">InvitesDriver::addRequest()</a>	19



<a href="#">IOTS_DAO.php</a>	24
<a href="#">IOTS_DB.php</a>	25
<a href="#">IOTS_DB</a>	48
<i>OTServ database handler interface.</i>	
<a href="#">IOTS_DAO</a>	47
<i>OTserv database object.</i>	
<a href="#">IOTS_GuildAction.php</a>	26
<a href="#">InvitesDriver</a>	19

## N

<a href="#">NEWS</a>	204
----------------------	-----

## O

<a href="#">OTS_Player::getConditions()</a>	136
<i>Conditions.</i>	
<a href="#">OTS_Player::getCap()</a>	136
<i>Capacity.</i>	
<a href="#">OTS_Player::getAccount()</a>	136
<i>Returns account of this player.</i>	
<a href="#">OTS_Player::getCustomField()</a>	137
<i>Reads custom field.</i>	
<a href="#">OTS_Player::getDepot()</a>	137
<i>Returns items tree from given depot.</i>	
<a href="#">OTS_Player::getGroup()</a>	139
<i>Returns group of this player.</i>	
<a href="#">OTS_Player::getExperience()</a>	138
<i>Experience points.</i>	
<a href="#">OTS_Player::getDirection()</a>	138
<i>Looking direction.</i>	
<a href="#">OTS_Player::find()</a>	135
<i>Loads player by it's name.</i>	
<a href="#">OTS_Player</a>	134
<i>OTServ character abstraction.</i>	
<a href="#">OTS_Item::count()</a>	132
<i>Count value for current item.</i>	
<a href="#">OTS_Item</a>	131
<i>Single item representation.</i>	
<a href="#">OTS_InfoRespond::getURL()</a>	131
<i>Returns server website.</i>	
<a href="#">OTS_Item::getAttributes()</a>	132
<i>Returns item custom attributes.</i>	
<a href="#">OTS_Item::getCount()</a>	133
<i>Returns count of item.</i>	
<a href="#">OTS_Item::setCount()</a>	134
<i>Sets count of item.</i>	
<a href="#">OTS_Item::setAttributes()</a>	133
<i>Sets item attributes.</i>	
<a href="#">OTS_Item::getId()</a>	133
<i>Returns item type.</i>	

<a href="#">OTS_Player::getGuildNick()</a>	139
<i>Guild nick.</i>	
<a href="#">OTS_Player::getHealth()</a>	139
<i>Current HP.</i>	
<a href="#">OTS_Player::getLossExperience()</a>	144
<i>Percentage of experience lost after dead.</i>	
<a href="#">OTS_Player::getLookType()</a>	143
<i>Outfit.</i>	
<a href="#">OTS_Player::getLookLegs()</a>	143
<i>Legs color.</i>	
<a href="#">OTS_Player::getLossMana()</a>	144
<i>Percentage of used mana lost after dead.</i>	
<a href="#">OTS_Player::getLossSkills()</a>	144
<i>Percentage of skills lost after dead.</i>	
<a href="#">OTS_Player::getManaMax()</a>	145
<i>Maximum mana.</i>	
<a href="#">OTS_Player::getMana()</a>	145
<i>Current mana.</i>	
<a href="#">OTS_Player::getMagLevel()</a>	145
<i>Magic level.</i>	
<a href="#">OTS_Player::getLookHead()</a>	143
<i>Hair color.</i>	
<a href="#">OTS_Player::getLookFeet()</a>	142
<i>Boots color.</i>	
<a href="#">OTS_Player::getLastIP()</a>	140
<i>Last login IP.</i>	
<a href="#">OTS_Player::getId()</a>	140
<i>Player ID.</i>	
<a href="#">OTS_Player::getHealthMax()</a>	140
<i>Maximum HP.</i>	
<a href="#">OTS_Player::getLastLogin()</a>	141
<i>Last login timestamp.</i>	
<a href="#">OTS_Player::getLevel()</a>	141
<i>Experience level.</i>	
<a href="#">OTS_Player::getLookBody()</a>	142
<i>Body color.</i>	
<a href="#">OTS_Player::getLookAddons()</a>	141
<i>Addons.</i>	
<a href="#">OTS_InfoRespond::getUptime()</a>	131
<i>Returns server uptime.</i>	
<a href="#">OTS_InfoRespond::getTSPQVersion()</a>	130
<i>Returns version of root element.</i>	
<a href="#">OTS_Guilds_List::key()</a>	122
<i>Current cursor position.</i>	
<a href="#">OTS_Guilds_List::deleteGuild()</a>	121
<i>Deletes guild.</i>	
<a href="#">OTS_Guilds_List::current()</a>	121
<i>Returns current row.</i>	
<a href="#">OTS_Guilds_List::next()</a>	122
<i>Moves to next row.</i>	
<a href="#">OTS_Guilds_List::rewind()</a>	122
<i>Select guilds from database.</i>	
<a href="#">OTS_Guilds_List::valid()</a>	124

<i>Checks if there are any rows left.</i>	
<a href="#">OTS_Guilds_List::setOffset()</a>	123
<i>Sets OFFSET.</i>	
<a href="#">OTS_Guilds_List::setLimit()</a>	123
<i>Sets LIMIT.</i>	
<a href="#">OTS_Guilds_List::count()</a>	121
<i>Returns number of guilds on list in current criterium.</i>	
<a href="#">OTS_Guilds_List</a>	120
<i>List of guilds.</i>	
<a href="#">OTS_GuildRanks_List::next()</a>	118
<i>Moves to next row.</i>	
<a href="#">OTS_GuildRanks_List::key()</a>	117
<i>Current cursor position.</i>	
<a href="#">OTS_GuildRanks_List::deleteGuildRank()</a>	117
<i>Deletes guild rank.</i>	
<a href="#">OTS_GuildRanks_List::rewind()</a>	118
<i>Select ranks from database.</i>	
<a href="#">OTS_GuildRanks_List::setLimit()</a>	118
<i>Sets LIMIT.</i>	
<a href="#">OTS_GuildRanks_List::valid()</a>	119
<i>Checks if there are any rows left.</i>	
<a href="#">OTS_GuildRanks_List::setOffset()</a>	119
<i>Sets OFFSET.</i>	
<a href="#">OTS_InfoRespond</a>	124
<i>Wrapper for 'info' respond's DOMDocument.</i>	
<a href="#">OTS_InfoRespond::getClientVersion()</a>	124
<i>Returns dedicated version of client.</i>	
<a href="#">OTS_InfoRespond::getOnlinePlayers()</a>	128
<i>Returns current amount of players online.</i>	
<a href="#">OTS_InfoRespond::getName()</a>	128
<i>Returns server name.</i>	
<a href="#">OTS_InfoRespond::getMOTD()</a>	128
<i>Returns server's Message Of The Day</i>	
<a href="#">OTS_InfoRespond::getOwner()</a>	129
<i>Returns owner name.</i>	
<a href="#">OTS_InfoRespond::getPlayersPeak()</a>	129
<i>Returns record of online players.</i>	
<a href="#">OTS_InfoRespond::getServerVersion()</a>	130
<i>Returns server version.</i>	
<a href="#">OTS_InfoRespond::getServer()</a>	130
<i>Returns server attribute.</i>	
<a href="#">OTS_InfoRespond::getPort()</a>	129
<i>Returns server port.</i>	
<a href="#">OTS_InfoRespond::getMonstersCount()</a>	127
<i>Returns number of all monsters on map.</i>	
<a href="#">OTS_InfoRespond::getMaxPlayers()</a>	127
<i>Returns maximum amount of players online.</i>	
<a href="#">OTS_InfoRespond::getLocation()</a>	125
<i>Returns server location.</i>	
<a href="#">OTS_InfoRespond::getIP()</a>	125
<i>Returns server IP.</i>	
<a href="#">OTS_InfoRespond::getEmail()</a>	125
<i>Returns owner e-mail.</i>	

<a href="#">OTS_InfoRespond::getMapAuthor()</a>	126
<i>Returns map author.</i>	
<a href="#">OTS_InfoRespond::getMapHeight()</a>	126
<i>Returns map height.</i>	
<a href="#">OTS_InfoRespond::getMapWidth()</a>	127
<i>Returns map width.</i>	
<a href="#">OTS_InfoRespond::getMapName()</a>	126
<i>Returns map name.</i>	
<a href="#">OTS_Player::getManaSpent()</a>	146
<i>Mana spent.</i>	
<a href="#">OTS_Player::getName()</a>	146
<i>Player name.</i>	
<a href="#">OTS_Player::setPremiumEnd()</a>	167
<i>Sets player's Premium Account expiration timestamp.</i>	
<a href="#">OTS_Player::setPosZ()</a>	167
<i>Sets Z map coordinate.</i>	
<a href="#">OTS_Player::setPosY()</a>	167
<i>Sets Y map coordinate.</i>	
<a href="#">OTS_Player::setRank()</a>	168
<i>Assigns guild rank.</i>	
<a href="#">OTS_Player::setRankId()</a>	168
<i>Sets guild rank ID.</i>	
<a href="#">OTS_Player::setSave()</a>	169
<i>Sets save flag.</i>	
<a href="#">OTS_Player::setRedSkullTime()</a>	169
<i>Sets red skulled time remained.</i>	
<a href="#">OTS_Player::setRedSkull()</a>	169
<i>Sets red skull flag.</i>	
<a href="#">OTS_Player::setPosX()</a>	166
<i>Sets X map coordinate.</i>	
<a href="#">OTS_Player::setName()</a>	166
<i>Sets players's name.</i>	
<a href="#">OTS_Player::setLossMana()</a>	163
<i>Sets percentage of used mana lost after dead.</i>	
<a href="#">OTS_Player::setLossExperience()</a>	163
<i>Sets percentage of experience lost after dead.</i>	
<a href="#">OTS_Player::setLookType()</a>	162
<i>Sets outfit.</i>	
<a href="#">OTS_Player::setLossSkills()</a>	164
<i>Sets percentage of skills lost after dead.</i>	
<a href="#">OTS_Player::setMagLevel()</a>	164
<i>Sets magic level.</i>	
<a href="#">OTS_Player::setManaSpent()</a>	165
<i>Sets mana spent.</i>	
<a href="#">OTS_Player::setManaMax()</a>	165
<i>Sets maximum mana.</i>	
<a href="#">OTS_Player::setMana()</a>	164
<i>Sets current mana.</i>	
<a href="#">OTS_Player::setSex()</a>	170
<i>Sets player gender.</i>	
<a href="#">OTS_Player::setSkill()</a>	170
<i>Sets skill value.</i>	
<a href="#">OTS_Players_List::key()</a>	176

Current cursor position.	175
<a href="#">OTS Players List::deletePlayer()</a>	175
Deletes player.	
<a href="#">OTS Players List::current()</a>	175
Returns current row.	
<a href="#">OTS Players List::next()</a>	176
Moves to next row.	
<a href="#">OTS Players List::rewind()</a>	176
Select players from database.	
<a href="#">OTS Players List::valid()</a>	177
Checks if there are any rows left.	
<a href="#">OTS Players List::setOffset()</a>	177
Sets OFFSET.	
<a href="#">OTS Players List::setLimit()</a>	176
Sets LIMIT.	
<a href="#">OTS Players List::count()</a>	174
Returns number of characters on list in current criterium.	
<a href="#">OTS Players List</a>	174
List of players.	
<a href="#">OTS Player::setSoul()</a>	172
Sets soul points.	
<a href="#">OTS Player::setSlot()</a>	171
Sets slot content.	
<a href="#">OTS Player::setSkillTries()</a>	171
Sets skill's tries for next level.	
<a href="#">OTS Player::setTownId()</a>	172
Sets residence town's ID.	
<a href="#">OTS Player::setVocation()</a>	173
Sets player proffesion.	
<a href="#">OTS Player::unsetSave()</a>	173
Unsets save flag.	
<a href="#">OTS Player::unsetRedSkull()</a>	173
Unsets red skull flag.	
<a href="#">OTS Player::setLookLegs()</a>	162
Sets legs color.	
<a href="#">OTS Player::setLookHead()</a>	162
Sets hair color.	
<a href="#">OTS Player::getSoul()</a>	151
Soul points.	
<a href="#">OTS Player::getSlot()</a>	150
Returns items tree from given slot.	
<a href="#">OTS Player::getSkillTries()</a>	150
Returns player's skill's tries for next level.	
<a href="#">OTS Player::getTownId()</a>	151
Residence town's ID.	
<a href="#">OTS Player::getVocation()</a>	152
Player proffesion.	
<a href="#">OTS Player::isSaveSet()</a>	153
Checks if save flag is set.	
<a href="#">OTS Player::isLoading()</a>	152
Checks if object is loaded.	
<a href="#">OTS Player::hasRedSkull()</a>	152
Checks if player has red skull.	

<a href="#">OTS_Player::getSkill()</a>	149
<i>Returns player's skill.</i>	
<a href="#">OTS_Player::getSex()</a>	149
<i>Player gender.</i>	
<a href="#">OTS_Player::getPosZ()</a>	147
<i>Z map coordinate.</i>	
<a href="#">OTS_Player::getPosY()</a>	147
<i>Y map coordinate.</i>	
<a href="#">OTS_Player::getPosX()</a>	146
<i>X map coordinate.</i>	
<a href="#">OTS_Player::getPremiumEnd()</a>	148
<i>Player's Premium Account expiration timestamp.</i>	
<a href="#">OTS_Player::getRank()</a>	148
<i>Assigned guild rank.</i>	
<a href="#">OTS_Player::getRedSkullTime()</a>	149
<i>Red skulled time remained.</i>	
<a href="#">OTS_Player::getRankId()</a>	148
<i>Guild rank ID.</i>	
<a href="#">OTS_Player::load()</a>	153
<i>Loads player with given id.</i>	
<a href="#">OTS_Player::save()</a>	153
<i>Saves player in database.</i>	
<a href="#">OTS_Player::setLastIP()</a>	159
<i>Sets last login IP.</i>	
<a href="#">OTS_Player::setHealthMax()</a>	159
<i>Sets maximum HP.</i>	
<a href="#">OTS_Player::setHealth()</a>	158
<i>Sets current HP.</i>	
<a href="#">OTS_Player::setLastLogin()</a>	159
<i>Sets last login timestamp.</i>	
<a href="#">OTS_Player::setLevel()</a>	160
<i>Sets experience level.</i>	
<a href="#">OTS_Player::setLookFeet()</a>	161
<i>Sets boots color.</i>	
<a href="#">OTS_Player::setLookBody()</a>	161
<i>Sets body color.</i>	
<a href="#">OTS_Player::setLookAddons()</a>	160
<i>Sets addons.</i>	
<a href="#">OTS_Player::setGuildNick()</a>	158
<i>Sets guild nick.</i>	
<a href="#">OTS_Player::setGroup()</a>	157
<i>Assigns character to group.</i>	
<a href="#">OTS_Player::setConditions()</a>	155
<i>Sets conditions.</i>	
<a href="#">OTS_Player::setCap()</a>	154
<i>Sets capacity.</i>	
<a href="#">OTS_Player::setAccount()</a>	154
<i>Assigns character to account.</i>	
<a href="#">OTS_Player::setCustomField()</a>	155
<i>Writes custom field.</i>	
<a href="#">OTS_Player::setDepot()</a>	156
<i>Sets slot content.</i>	
<a href="#">OTS_Player::setExperience()</a>	157

<i>Sets experience points.</i>	
<a href="#">OTS_Player::setDirection()</a>	156
<i>Sets looking direction.</i>	
<a href="#">OTS_GuildRanks_List::current()</a>	117
<i>Returns current row.</i>	
<a href="#">OTS_GuildRanks_List::count()</a>	116
<i>Returns number of ranks on list in current criterium.</i>	
<a href="#">OTS_Container</a>	69
<i>Container item representation.</i>	
<a href="#">OTS_Accounts_List::valid()</a>	68
<i>Checks if there are any rows left.</i>	
<a href="#">OTS_Accounts_List::setOffset()</a>	68
<i>Sets OFFSET.</i>	
<a href="#">OTS_Container::addItem()</a>	69
<i>Adds item to container.</i>	
<a href="#">OTS_Container::count()</a>	69
<i>Number of items inside container.</i>	
<a href="#">OTS_Container::next()</a>	70
<i>Moves to next item.</i>	
<a href="#">OTS_Container::key()</a>	70
<i>Current cursor position.</i>	
<a href="#">OTS_Container::current()</a>	70
<i>Returns current item.</i>	
<a href="#">OTS_Accounts_List::setLimit()</a>	67
<i>Sets LIMIT.</i>	
<a href="#">OTS_Accounts_List::rewind()</a>	67
<i>Select accounts from database.</i>	
<a href="#">OTS_Accounts_List</a>	65
<i>List of accounts.</i>	
<a href="#">OTS_Account::unblock()</a>	64
<i>Unblocks account.</i>	
<a href="#">OTS_Account::setPassword()</a>	64
<i>Sets account's password.</i>	
<a href="#">OTS_Accounts_List::count()</a>	65
<i>Returns number of accounts on list in current criterium.</i>	
<a href="#">OTS_Accounts_List::current()</a>	66
<i>Returns current row.</i>	
<a href="#">OTS_Accounts_List::next()</a>	67
<i>Moves to next row.</i>	
<a href="#">OTS_Accounts_List::key()</a>	66
<i>Current cursor position.</i>	
<a href="#">OTS_Accounts_List::deleteAccount()</a>	66
<i>Deletes account.</i>	
<a href="#">OTS_Container::removeItem()</a>	71
<i>Removes given item from current container.</i>	
<a href="#">OTS_Container::rewind()</a>	71
<i>Resets internal items array pointer.</i>	
<a href="#">OTS_DB_ODBC::SQLquote()</a>	78
<i>IOTS_DB method.</i>	
<a href="#">OTS_DB_ODBC::SQLquery()</a>	77
<i>IOTS_DB method.</i>	
<a href="#">OTS_DB_ODBC::limit()</a>	77
<i>LIMIT/OFFSET clause for queries.</i>	

<a href="#">OTS_DB_ODBC::tableName()</a>	78
<i>Query-quoted table name.</i>	
<a href="#">OTS_DB_PostgreSQL</a>	79
<i>PostgreSQL connection interface.</i>	
<a href="#">OTS_DB_PostgreSQL::SQLquery()</a>	81
<i>IOTS_DB method.</i>	
<a href="#">OTS_DB_PostgreSQL::limit()</a>	80
<i>LIMIT/OFFSET clause for queries.</i>	
<a href="#">OTS_DB_PostgreSQL::fieldName()</a>	80
<i>Query-quoted field name.</i>	
<a href="#">OTS_DB_ODBC::fieldName()</a>	76
<i>Query-quoted field name.</i>	
<a href="#">OTS_DB_ODBC</a>	75
<i>ODBC connection interface.</i>	
<a href="#">OTS_DB_MySQL::fieldName()</a>	73
<i>Query-quoted field name.</i>	
<a href="#">OTS_DB_MySQL</a>	72
<i>MySQL connection interface.</i>	
<a href="#">OTS_Container::valid()</a>	71
<i>Checks if there are any items left.</i>	
<a href="#">OTS_DB_MySQL::limit()</a>	73
<i>LIMIT/OFFSET clause for queries.</i>	
<a href="#">OTS_DB_MySQL::SQLquery()</a>	74
<i>IOTS_DB method.</i>	
<a href="#">OTS_DB_MySQL::tableName()</a>	75
<i>Query-quoted table name.</i>	
<a href="#">OTS_DB_MySQL::SQLquote()</a>	74
<i>IOTS_DB method.</i>	
<a href="#">OTS_Account::setPACCDays()</a>	63
<i>Sets PACC days count.</i>	
<a href="#">OTS_Account::setGroup()</a>	63
<i>Assigns account to group.</i>	
<a href="#">OTS_GuildRanks_List.php</a>	39
<a href="#">OTS_GuildRank.php</a>	38
<a href="#">OTS_Guild.php</a>	37
<a href="#">OTS_Guilds_List.php</a>	40
<a href="#">OTS_InfoRespond.php</a>	41
<a href="#">OTS_Players_List.php</a>	44
<a href="#">OTS_Player.php</a>	43
<a href="#">OTS_Item.php</a>	42
<a href="#">OTS_Groups_List.php</a>	36
<a href="#">OTS_Group.php</a>	35
<a href="#">OTS_Container.php</a>	30
<a href="#">OTS_Accounts_List.php</a>	29
<a href="#">OTS_Account.php</a>	28
<a href="#">OTS_DB_MySQL.php</a>	31
<a href="#">OTS_DB_ODBC.php</a>	32
<a href="#">OTS_DB_SQLite.php</a>	34
<a href="#">OTS_DB_PostgreSQL.php</a>	33
<a href="#">OTS_SQLite_Results.php</a>	45
<a href="#">OTS_Account</a>	54
<i>OTServ account abstraction.</i>	
<a href="#">OTS_Account::isBlocked()</a>	60



<i>Checks if account is blocked.</i>	
<a href="#">OTS Account::getPlayers()</a>	60
<i>List of characters on account.</i>	
<a href="#">OTS Account::getPassword()</a>	60
<i>Account's password.</i>	
<a href="#">OTS Account::isLoading()</a>	61
<i>Checks if object is loaded.</i>	
<a href="#">OTS Account::load()</a>	61
<i>Loads account with given number.</i>	
<a href="#">OTS Account::setEMail()</a>	62
<i>Sets account's email.</i>	
<a href="#">OTS Account::setCustomField()</a>	62
<i>Writes custom field.</i>	
<a href="#">OTS Account::save()</a>	61
<i>Updates account in database.</i>	
<a href="#">OTS Account::getPACCDays()</a>	59
<i>PACC days.</i>	
<a href="#">OTS Account::getId()</a>	59
<i>Account number.</i>	
<a href="#">OTS Account::createEx()</a>	56
<i>Creates new account.</i>	
<a href="#">OTS Account::create()</a>	55
<i>Creates new account.</i>	
<a href="#">OTS Account::block()</a>	54
<i>Blocks account.</i>	
<a href="#">OTS Account::find()</a>	57
<i>Loads account by it's e-mail address.</i>	
<a href="#">OTS Account::getCustomField()</a>	57
<i>Reads custom field.</i>	
<a href="#">OTS Account::getGroup()</a>	58
<i>Returns group of this account.</i>	
<a href="#">OTS Account::getEMail()</a>	58
<i>E-mail address.</i>	
<a href="#">OTS DB PostgreSQL::SQLquote()</a>	81
<i>IOTS_DB method.</i>	
<a href="#">OTS DB PostgreSQL::tableName()</a>	81
<i>Query-quoted table name.</i>	
<a href="#">OTS Guild::load()</a>	104
<i>Loads guild with given id.</i>	
<a href="#">OTS Guild::listRequests()</a>	104
<i>Returns list of players that requested membership.</i>	
<a href="#">OTS Guild::listInvites()</a>	104
<i>Returns list of invited players.</i>	
<a href="#">OTS Guild::request()</a>	105
<i>Requests membership in guild for player player.</i>	
<a href="#">OTS Guild::save()</a>	105
<i>Saves guild in database.</i>	
<a href="#">OTS Guild::setInvitesDriver()</a>	107
<i>Assigns invites handler.</i>	
<a href="#">OTS Guild::setCustomField()</a>	106
<i>Writes custom field.</i>	
<a href="#">OTS Guild::setCreationData()</a>	106
<i>Sets guild creation data.</i>	

<a href="#">OTS_Guild::isLoaded()</a>	103
<i>Checks if object is loaded.</i>	
<a href="#">OTS_Guild::invite()</a>	103
<i>Invites player to guild.</i>	
<a href="#">OTS_Guild::getCreationData()</a>	100
<i>Guild creation data.</i>	
<a href="#">OTS_Guild::find()</a>	100
<i>Loads guild by it's name.</i>	
<a href="#">OTS_Guild::deleteRequest()</a>	99
<i>Deletes request from player.</i>	
<a href="#">OTS_Guild::getCustomField()</a>	101
<i>Reads custom field.</i>	
<a href="#">OTS_Guild::getGuildRanks()</a>	101
<i>Reads all ranks that are in this guild.</i>	
<a href="#">OTS_Guild::getOwner()</a>	102
<i>Returns owning player of this player.</i>	
<a href="#">OTS_Guild::getName()</a>	102
<i>Guild name.</i>	
<a href="#">OTS_Guild::getId()</a>	102
<i>Guild ID.</i>	
<a href="#">OTS_Guild::setName()</a>	107
<i>Sets players's name.</i>	
<a href="#">OTS_Guild::setOwner()</a>	108
<i>Assigns guild to owner.</i>	
<a href="#">OTS_GuildRank::save()</a>	113
<i>Saves rank in database.</i>	
<a href="#">OTS_GuildRank::load()</a>	113
<i>Loads rank with given id.</i>	
<a href="#">OTS_GuildRank::isLoaded()</a>	112
<i>Checks if object is loaded.</i>	
<a href="#">OTS_GuildRank::setCustomField()</a>	113
<i>Writes custom field.</i>	
<a href="#">OTS_GuildRank::setGuild()</a>	114
<i>Assigns rank to guild.</i>	
<a href="#">OTS_GuildRanks_List</a>	116
<i>List of guild ranks.</i>	
<a href="#">OTS_GuildRank::setName()</a>	115
<i>Sets rank's name.</i>	
<a href="#">OTS_GuildRank::setLevel()</a>	115
<i>Sets rank's access level within guild.</i>	
<a href="#">OTS_GuildRank::getPlayers()</a>	112
<i>Reads all players who has this rank set.</i>	
<a href="#">OTS_GuildRank::getName()</a>	112
<i>Rank name.</i>	
<a href="#">OTS_GuildRank::find()</a>	109
<i>Loads rank by it's name.</i>	
<a href="#">OTS_GuildRank</a>	109
<i>OTServ guild rank abstraction.</i>	
<a href="#">OTS_Guild::setRequestsDriver()</a>	108
<i>Assigns requests handler.</i>	
<a href="#">OTS_GuildRank::getCustomField()</a>	110
<i>Reads custom field.</i>	
<a href="#">OTS_GuildRank::getGuild()</a>	111

<i>Returns guild of this rank.</i>	
<a href="#">OTS_GuildRank::getLevel()</a>	111
<i>Rank's access level.</i>	
<a href="#">OTS_GuildRank::getId()</a>	111
<i>Rank ID.</i>	
<a href="#">OTS_Guild::deleteInvite()</a>	99
<i>Deletes invitation for player to guild.</i>	
<a href="#">OTS_Guild::acceptRequest()</a>	98
<i>Accepts player.</i>	
<a href="#">OTS_Group::getMaxDepotItems()</a>	88
<i>Maximum count of items in depot.</i>	
<a href="#">OTS_Group::getId()</a>	87
<i>Group ID.</i>	
<a href="#">OTS_Group::getFlags()</a>	87
<i>Rights flags.</i>	
<a href="#">OTS_Group::getMaxVIPList()</a>	88
<i>Maximum count of players in VIP list.</i>	
<a href="#">OTS_Group::getName()</a>	88
<i>Group name.</i>	
<a href="#">OTS_Group::load()</a>	89
<i>Loads group with given id.</i>	
<a href="#">OTS_Group::isLoaded()</a>	89
<i>Checks if object is loaded.</i>	
<a href="#">OTS_Group::getPlayers()</a>	89
<i>List of characters in given group.</i>	
<a href="#">OTS_Group::getCustomField()</a>	86
<i>Reads custom field.</i>	
<a href="#">OTS_Group::getAccess()</a>	86
<i>Access level.</i>	
<a href="#">OTS_DB_SQLite::limit()</a>	83
<i>LIMIT/OFFSET clause for queries.</i>	
<a href="#">OTS_DB_SQLite::fieldName()</a>	83
<i>Query-quoted field name.</i>	
<a href="#">OTS_DB_SQLite</a>	82
<i>SQLite connection interface.</i>	
<a href="#">OTS_DB_SQLite::SQLquery()</a>	84
<i>IOTS_DB method.</i>	
<a href="#">OTS_DB_SQLite::SQLquote()</a>	84
<i>IOTS_DB method.</i>	
<a href="#">OTS_Group</a>	85
<i>OTServ user group abstraction.</i>	
<a href="#">OTS_DB_SQLite::tableName()</a>	85
<i>Query-quoted table name.</i>	
<a href="#">OTS_Group::save()</a>	90
<i>Saves account in database.</i>	
<a href="#">OTS_Group::setAccess()</a>	90
<i>Sets access level.</i>	
<a href="#">OTS_Groups_List::rewind()</a>	95
<i>Select groups from database.</i>	
<a href="#">OTS_Groups_List::next()</a>	95
<i>Moves to next row.</i>	
<a href="#">OTS_Groups_List::key()</a>	95
<i>Current cursor position.</i>	

<a href="#">OTS Groups List::setLimit()</a>	96
<i>Sets LIMIT.</i>	
<a href="#">OTS Groups List::setOffset()</a>	96
<i>Sets OFFSET.</i>	
<a href="#">OTS Guild::acceptInvite()</a>	98
<i>Finalise invitation.</i>	
<a href="#">OTS Guild</a>	97
<i>OTServ guild abstraction.</i>	
<a href="#">OTS Groups List::valid()</a>	97
<i>Checks if there are any rows left.</i>	
<a href="#">OTS Groups List::deleteGroup()</a>	94
<i>Deletes group.</i>	
<a href="#">OTS Groups List::current()</a>	94
<i>Returns current row.</i>	
<a href="#">OTS Group::setMaxDepotItems()</a>	92
<i>Sets maximum count of items in depot.</i>	
<a href="#">OTS Group::setFlags()</a>	91
<i>Sets rights flags.</i>	
<a href="#">OTS Group::setCustomField()</a>	91
<i>Writes custom field.</i>	
<a href="#">OTS Group::setMaxVIPList()</a>	92
<i>Sets maximum count of players in VIP list.</i>	
<a href="#">OTS Group::setName()</a>	93
<i>Sets group's name.</i>	
<a href="#">OTS Groups List::count()</a>	94
<i>Returns number of groups on list in current criterium.</i>	
<a href="#">OTS Groups List</a>	93
<i>List of groups.</i>	
<a href="#">OTS.php</a>	27
<i>This file contains main toolkit class.</i>	

## P

<a href="#">POT::SLOT_LEGS</a>	186
<i>Legs slot.</i>	
<a href="#">POT::SLOT_NECKLACE</a>	186
<i>Necklace slot.</i>	
<a href="#">POT::SLOT_RIGHT</a>	187
<i>Right hand slot.</i>	
<a href="#">POT::SLOT_RING</a>	187
<i>Ring slot.</i>	
<a href="#">POT::SLOT_LEFT</a>	186
<i>Left hand slot.</i>	
<a href="#">POT::SLOT_HEAD</a>	185
<i>Head slot.</i>	
<a href="#">POT::SLOT_ARMOR</a>	184
<i>Armor slot.</i>	
<a href="#">POT::SLOT_BACKPACK</a>	185
<i>Backpack slot.</i>	
<a href="#">POT::SLOT_FEET</a>	185
<i>Boots slot.</i>	
<a href="#">POT::VOCATION_DRUID</a>	188

<a href="#"><i>Druid.</i></a>	
<a href="#">POT::VOCATION_KNIGHT</a>	188
<a href="#"><i>Knight.</i></a>	
<a href="#">POT::getInstance()</a>	191
<a href="#"><i>Singleton.</i></a>	
<a href="#">POT::loadClass()</a>	191
<a href="#"><i>Loads POT class file.</i></a>	
<a href="#">POT::serverStatus()</a>	192
<a href="#"><i>Queries server status.</i></a>	
<a href="#">POT::setPOTPath()</a>	193
<a href="#"><i>Set POT directory.</i></a>	
<a href="#">POT::createObject()</a>	190
<a href="#"><i>Creates OTServ DAO class instance.</i></a>	
<a href="#">POT::connect()</a>	189
<a href="#"><i>Connects to database.</i></a>	
<a href="#">POT::VOCATION_NONE</a>	188
<a href="#"><i>None vocation.</i></a>	
<a href="#">POT::VOCATION_PALADIN</a>	188
<a href="#"><i>Paladin.</i></a>	
<a href="#">POT::VOCATION_SORCERER</a>	189
<a href="#"><i>Sorcerer.</i></a>	
<a href="#">POT::SLOT_AMMO</a>	184
<a href="#"><i>Ammunition slot.</i></a>	
<a href="#">POT::SKILL_SWORD</a>	183
<a href="#"><i>Sword fighting.</i></a>	
<a href="#">POT::DB_PGSQL</a>	179
<a href="#"><i>PostgreSQL driver.</i></a>	
<a href="#">POT::DB_SQLITE</a>	179
<a href="#"><i>SQLite driver.</i></a>	
<a href="#">POT::DIRECTION_EAST</a>	179
<a href="#"><i>East.</i></a>	
<a href="#">POT::DIRECTION_NORTH</a>	180
<a href="#"><i>North.</i></a>	
<a href="#">POT::DB_ODBC</a>	178
<a href="#"><i>ODBC driver.</i></a>	
<a href="#">POT::DB_MYSQL</a>	178
<a href="#"><i>MySQL driver.</i></a>	
<a href="#">PHP 5.0</a>	3
<a href="#">POT class preview</a>	5
<a href="#">POT</a>	178
<a href="#"><i>Main POT class.</i></a>	
<a href="#">POT::DIRECTION_SOUTH</a>	180
<a href="#"><i>South.</i></a>	
<a href="#">POT::DIRECTION_WEST</a>	180
<a href="#"><i>West.</i></a>	
<a href="#">POT::SKILL_FISHING</a>	182
<a href="#"><i>Fishing.</i></a>	
<a href="#">POT::SKILL_FIST</a>	183
<a href="#"><i>Fist fighting.</i></a>	
<a href="#">POT::SKILL_SHIELDING</a>	183
<a href="#"><i>Shielding.</i></a>	
<a href="#">POT::SKILL_DISTANCE</a>	182
<a href="#"><i>Distance fighting.</i></a>	

<a href="#">POT::SKILL CLUB</a>	182
<i>Club fighting.</i>	
<a href="#">POT::SEX FEMALE</a>	181
<i>Female gender.</i>	
<a href="#">POT::SEX MALE</a>	181
<i>Male gender.</i>	
<a href="#">POT::SKILL AXE</a>	181
<i>Axe fighting.</i>	
<a href="#">POT</a>	1

## Q

<a href="#">Quick start</a>	6
-----------------------------	---

## R

<a href="#">README</a>	202
------------------------	-----

## S

<a href="#">Server online status</a>	17
--------------------------------------	----