

PHP OTServ Toolkit



Contents

POT	1
PHP 5.0	3
POT class preview	5
Quick start	6
DAO objects	9
Guilds	13
Guild action drivers	15
Account number hack	18
Server online status	19
Package POT Procedural Elements	22
E_OTS_NoDriver.php	22
E_OTS_NotLoaded.php	23
IOTS_DAO.php	24
IOTS_DB.php	25
IOTS_GuildAction.php	26
OTS.php	27
OTS_Account.php	28
OTS_Accounts_List.php	29
OTS_Container.php	30
OTS_DB_MySQL.php	31
OTS_DB_ODBC.php	32
OTS_DB_PostgreSQL.php	33
OTS_DB_SQLite.php	34
OTS_Group.php	35
OTS_Groups_List.php	36
OTS_Guild.php	37
OTS_GuildRank.php	38
OTS_GuildRanks_List.php	39
OTS_Guilds_List.php	40
OTS_InfoRespond.php	41
OTS_Item.php	42
OTS_Player.php	43
OTS_Players_List.php	44
OTS_SQLite_Results.php	45
Package POT Classes	46
Class E_OTS_NoDriver	46
Class E_OTS_NotLoaded	46
Class IOTS_DAO	47
Constructor __construct	47
Class IOTS_DB	48
Constructor __construct	48

Method fieldName	48
Method lastInsertId	49
Method limit	49
Method SQLquery	50
Method SQLquote	50
Method tableName	50
Class IOTS GuildAction	51
Constructor construct	51
Method addRequest	52
Method deleteRequest	52
Method listRequests	53
Method submitRequest	53
Class OTS Account	54
Constructor construct	54
Method ban	54
Method block	55
Method create	55
example: account.php	55
Method createEx	56
example: account.php	56
Method find	58
Method getCustomField	58
Method getEmail	59
Method getGroup	59
Method getId	59
Method getPACCDays	60
Method getPassword	60
Method getPlayers	61
Method isBanned	61
Method isBlocked	61
Method isLoaded	62
Method load	62
Method save	62
Method setCustomField	63
Method setEmail	63
Method setGroup	64
Method setPACCDays	64
Method setPassword	65
Method unban	65
Method unblock	66
Method clone	66
Method set state	66
Method sleep	67
Method wakeup	67
Class OTS Accounts List	68
Constructor construct	68
Method count	68
Method current	69
Method deleteAccount	69

Method key	69
Method next	70
Method rewind	70
Method setLimit	70
Method setOffset	71
Method valid	71
Method set state	71
Method sleep	72
Method wakeup	72
Class OTS Container	73
Method addItem	73
Method count	74
Method current	74
Method key	74
Method next	75
Method removeItem	75
Method rewind	75
Method valid	76
Class OTS DB MySQL	76
Constructor construct	77
Method fieldName	77
Method limit	78
Method SQLquery	78
Method SQLquote	78
Method tableName	79
Class OTS DB ODBC	79
Constructor construct	80
Method fieldName	80
Method limit	81
Method SQLquery	81
Method SQLquote	82
Method tableName	82
Class OTS DB PostgreSQL	83
Constructor construct	83
Method fieldName	84
Method limit	84
Method SQLquery	85
Method SQLquote	85
Method tableName	86
Class OTS DB SQLite	86
Constructor construct	87
Method fieldName	87
Method limit	88
Method SQLquery	88
Method SQLquote	89
Method tableName	89
Class OTS Group	90
Constructor construct	90
Method getAccess	90

Method getCustomField	91
Method getFlags	91
Method getId	92
Method getMaxDepotItems	92
Method getMaxVIPList	92
Method getName	93
Method getPlayers	93
Method isLoaded	93
Method load	94
Method save	94
Method setAccess	94
Method setCustomField	95
Method setFlags	96
Method setMaxDepotItems	96
Method setMaxVIPList	96
Method setName	97
Method clone	97
Method set state	98
Method sleep	98
Method wakeup	99
Class OTS_Groups_List	99
Constructor construct	99
Method count	100
Method current	100
Method deleteGroup	100
Method key	101
Method next	101
Method rewind	101
Method setLimit	102
Method setOffset	102
Method valid	103
Method set state	103
Method sleep	103
Method wakeup	104
Class OTS_Guild	104
Constructor construct	105
Method acceptInvite	105
Method acceptRequest	106
Method deleteInvite	106
Method deleteRequest	107
Method find	107
Method getCreationData	108
Method getCustomField	108
Method getGuildRanks	108
Method getId	109
Method getName	109
Method getOwner	110
Method invite	110
Method isLoaded	110

Method listInvites	111
Method listRequests	111
Method load	111
Method request	112
Method save	112
Method setCreationData	113
Method setCustomField	113
Method setInvitesDriver	114
Method setName	114
Method setOwner	115
Method setRequestsDriver	115
Method clone	116
Method set state	116
Method sleep	117
Method wakeup	117
Class OTS_GuildRank	117
Constructor construct	118
Method find	118
Method getCustomField	119
Method getGuild	119
Method getId	120
Method getLevel	120
Method getName	120
Method getPlayers	121
Method isLoaded	121
Method load	121
Method save	122
Method setCustomField	122
Method setGuild	123
Method setLevel	123
Method setName	124
Method clone	124
Method set state	125
Method sleep	125
Method wakeup	125
Class OTS_GuildRanks_List	126
Constructor construct	126
Method count	127
Method current	127
Method deleteGuildRank	127
Method key	128
Method next	128
Method rewind	128
Method setLimit	129
Method setOffset	129
Method valid	130
Method set state	130
Method sleep	131
Method wakeup	131

<u>Class OTS_Guilds_List</u>	131
<u>Constructor construct</u>	132
<u>Method count</u>	132
<u>Method current</u>	132
<u>Method deleteGuild</u>	133
<u>Method key</u>	133
<u>Method next</u>	134
<u>Method rewind</u>	134
<u>Method setLimit</u>	134
<u>Method setOffset</u>	135
<u>Method valid</u>	135
<u>Method set state</u>	135
<u>Method sleep</u>	136
<u>Method wakeup</u>	136
<u>Class OTS_InfoRespond</u>	137
<u>Method getClientVersion</u>	137
<u>Method getEmail</u>	137
<u>Method getIP</u>	138
<u>Method getLocation</u>	138
<u>Method getMapAuthor</u>	138
<u>Method getMapHeight</u>	139
<u>Method getMapName</u>	139
<u>Method getMapWidth</u>	139
<u>Method getMaxPlayers</u>	140
<u>Method getMonstersCount</u>	140
<u>Method getMOTD</u>	140
<u>Method getName</u>	141
<u>Method getOnlinePlayers</u>	141
<u>Method getOwner</u>	141
<u>Method getPlayersPeak</u>	142
<u>Method getPort</u>	142
<u>Method getServer</u>	142
<u>Method getServerVersion</u>	143
<u>Method getTSPQVersion</u>	143
<u>Method getUptime</u>	143
<u>Method getURL</u>	144
<u>Class OTS_Item</u>	144
<u>Constructor construct</u>	144
<u>Method count</u>	145
<u>Method getAttributes</u>	145
<u>Method getCount</u>	145
<u>Method getId</u>	146
<u>Method setAttributes</u>	146
<u>Method setCount</u>	147
<u>Class OTS_Player</u>	147
<u>Constructor construct</u>	147
<u>Method ban</u>	148
<u>Method find</u>	148
<u>Method getAccount</u>	149

Method getCap	149
Method getConditions	150
Method getCustomField	150
Method getDepot	151
Method getDirection	151
Method getExperience	151
Method getGroup	152
Method getGuildNick	152
Method getHealth	153
Method getHealthMax	153
Method getId	153
Method getLastIP	154
Method getLastLogin	154
Method getLevel	154
Method getLookAddons	155
Method getLookBody	155
Method getLookFeet	155
Method getLookHead	156
Method getLookLegs	156
Method getLookType	156
Method getLossExperience	157
Method getLossMana	157
Method getLossSkills	158
Method getMagLevel	158
Method getMana	158
Method getManaMax	159
Method getManaSpent	159
Method getName	159
Method getPosX	160
Method getPosY	160
Method getPosZ	160
Method getPremiumEnd	161
Method getRank	161
Method getRankId	161
Method getRedSkullTime	162
Method getSex	162
Method getSkill	163
Method getSkillTries	163
Method getSlot	164
Method getSoul	164
Method getTownId	165
Method getVocation	165
Method hasRedSkull	165
Method isBanned	166
Method isLoaded	166
Method isSaveSet	166
Method load	167
Method save	167
Method setAccount	167

Method setCap	168
Method setConditions	168
Method setCustomField	169
Method setDepot	169
Method setDirection	170
Method setExperience	171
Method setGroup	171
Method setGuildNick	171
Method setHealth	172
Method setHealthMax	172
Method setLastIP	173
Method setLastLogin	173
Method setLevel	173
Method setLookAddons	174
Method setLookBody	174
Method setLookFeet	175
Method setLookHead	175
Method setLookLegs	176
Method setLookType	176
Method setLossExperience	176
Method setLossMana	177
Method setLossSkills	177
Method setMagLevel	178
Method setMana	178
Method setManaMax	178
Method setManaSpent	179
Method setName	179
Method setPosX	180
Method setPosY	180
Method setPosZ	181
Method setPremiumEnd	181
Method setRank	181
Method setRankId	182
Method setRedSkull	182
Method setRedSkullTime	183
Method setSave	183
Method setSex	183
Method setSkill	184
Method setSkillTries	184
Method setSlot	185
Method setSoul	185
Method setTownId	186
Method setVocation	186
Method unban	187
Method unsetRedSkull	187
Method unsetSave	187
Method clone	188
Method set state	188
Method sleep	188

Method wakeup	189
Class OTS Players List	189
Constructor construct	190
Method count	190
Method current	190
Method deletePlayer	191
Method key	191
Method next	191
Method rewind	192
Method setLimit	192
Method setOffset	192
Method valid	193
Method set state	193
Method sleep	194
Method wakeup	194
Class POT	195
Class Constant BAN ACCOUNT	195
Class Constant BAN IP	195
Class Constant BAN PLAYER	196
Class Constant DB MYSQL	196
Class Constant DB ODBC	196
Class Constant DB PGSQL	197
Class Constant DB SQLITE	197
Class Constant DEPOT_SID_FIRST	197
Class Constant DIRECTION EAST	198
Class Constant DIRECTION NORTH	198
Class Constant DIRECTION SOUTH	198
Class Constant DIRECTION WEST	199
Class Constant SEX FEMALE	199
Class Constant SEX MALE	199
Class Constant SKILL AXE	200
Class Constant SKILL CLUB	200
Class Constant SKILL DISTANCE	200
Class Constant SKILL FISHING	201
Class Constant SKILL FIST	201
Class Constant SKILL SHIELDING	201
Class Constant SKILL SWORD	202
Class Constant SLOT AMMO	202
Class Constant SLOT ARMOR	203
Class Constant SLOT BACKPACK	203
Class Constant SLOT FEET	203
Class Constant SLOT HEAD	204
Class Constant SLOT LEFT	204
Class Constant SLOT LEGS	204
Class Constant SLOT NECKLACE	205
Class Constant SLOT RIGHT	205
Class Constant SLOT RING	206
Class Constant VOCATION DRUID	206
Class Constant VOCATION KNIGHT	206

Class Constant VOCATION_NONE	207
Class Constant VOCATION_PALADIN	207
Class Constant VOCATION_SORCERER	207
Method banIP	208
Method connect	208
example: connect.php	208
Method createObject	209
Method getDBHandle	210
Method getInstance	210
Method getVocationID	211
Method getVocationName	211
Method getVocationsList	212
Method isIPBanned	212
Method loadClass	213
Method loadVocations	213
Method serverStatus	214
example: example	214
Method setPOTPath	215
example: fakeroot.php	215
Method unbanIP	215
compat.php	217
Appendices	218
Appendix A - Class Trees	219
POT	219
Appendix B - README/CHANGELOG/INSTALL	223
CHANGELOG	224
README	224
INSTALL	226
NEWS	226
Appendix D - Todo List	228

POT

This is documenation 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.

Type hinting

In PHP5 new feature was introduced - [type hinting](#). In PHP 5.0 only class names are supported, array type is supported since PHP 5.1. So if you want to use POT with PHP 5.0 you must remove all array hinted types in classes methods.

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.

Basics

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 through guild ranks. To list guild members you first need to list its ranks. Here is an example solution to list members in original 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 ' . $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 [IOTS_GuildAction interface](#). When the driver is assigned to guild object, each time a method of [OTS_Guild](#) object is called, it will forward this to action driver.

Sample driver

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

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

```

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

```

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

Account number hack

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

Walkaround

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

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

Server online status

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

Such a simple way

[POT class](#) contains [serverStatus\(\) method](#) which sends 'info' packet to OTS and handles results. It returns object of class [OTS_InfoRespond](#) which provides access methods for all 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 POT Procedural Elements

E_OTS_NoDriver.php

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

E_OTS_NotLoaded.php

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

IOTS_DAO.php

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

OTS.php

This file contains main toolkit class.

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

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Copyright** 2007 (C) by Wrzasq
- **TODO** Spawns support (OTBM support -> cache).
- **TODO** Items list (items.xml + items.otb -> cache).
- **TODO** Implement __get()/__set()/__call()/__toString(); Iterator, ArrayAccess, Countable interfaces.
- **TODO** List objects sorting/criteria-based loading.
- **TODO** More detailed documentation, better examples, more detailed phpUnit tests.
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Account.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.1
- **Version** 0.0.4
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Accounts_List.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com >
- **Version** 0.0.1
- **Version** 0.0.4
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Container.php

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

OTS_DB_MySQL.php

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

OTS_DB_ODBC.php

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

OTS_DB_PostgreSQL.php

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

OTS_DB_SQLite.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.1
- **Version** 0.0.4
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Group.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.1
- **Version** 0.0.4
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Groups_List.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.1
- **Version** 0.0.4
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_Guild.php

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

OTS_GuildRank.php

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

OTS_GuildRanks_List.php

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

OTS_Guilds_List.php

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

OTS_InfoRespond.php

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

OTS_Item.php

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

OTS_Player.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.1
- **Version** 0.0.4+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 >
- **Version** 0.0.1
- **Version** 0.0.4
- **Copyright** 2007 (C) by Wrzasq
- **License** [GNU Lesser General Public License, Version 3](#)

OTS_SQLite_Results.php

- **Package** POT
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.1
- **Copyright** 2007 (C) by Wrzasq
- **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** 0.0.4
- **Since** 0.0.4

Class E_OTS_NotLoaded

[line 20]

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

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

- **Package** POT

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

Class 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 32]

Guild action interface.

Guild action interface.

This interface indicates that class can handle OTServ guild action.

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

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

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

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

Function Parameters:

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

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

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

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

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

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

Function Parameters:

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

Adds new request.

Adds new request.

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

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

Function Parameters:

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

Deletes request.

Deletes request.

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

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

List of saved pending actions.

List of saved pending actions.

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

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

Function Parameters:

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

Finalizes request.

Finalizes request.

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

- **Access** public

Class OTS_Account

[line 21]

OTServ account abstraction.

OTServ account abstraction.

- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.4

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::ban([\$time = 0]) *[line 524]*

Function Parameters:

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

Bans current account.

Bans current account.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN
- **Access** public

void function OTS_Account::block() [*line 401*]

Blocks account.

Blocks account.

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

int function OTS_Account::create([\$min = 1], [\$max = 9999999]) [*line 140*]

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 0.0.4 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** 0.0.4
- **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 166*]

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** 0.0.4
- **Version** 0.0.1
- **Throws** Exception When there are no free account numbers.
- **Since** 0.0.4
- **Access** public
- **Example**

void function OTS_Account::find(\$email) [*line 233*]

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 448*]

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 353*]

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 297*]

Returns group of this account.

Returns group of this account.

- **Version** 0.0.4
- **Version** 0.0.1
- **Throws** E_OTS_NotLoaded If account is not loaded.
- **Since** 0.0.4
- **Access** public

int function OTS_Account::getId() [*line 279*]

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 414*]

PACC days.

PACC days.

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

string function OTS_Account::getPassword() [*line 326*]

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 497*]

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::isBanned() [*line 559*]

Checks if account is banned.

Checks if account is banned.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN
- **Access** public

bool function OTS_Account::isBlocked() [*line 380*]

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 250*]

Checks if object is loaded.

Checks if object is loaded.

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

void function OTS_Account::load(\$id) [*line 220*]

Function Parameters:

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

Loads account with given number.

Loads account with given number.

- **Version** 0.0.4
- **Version** 0.0.1
- **Access** public

void function OTS_Account::save() [*line 261*]

Updates account in database.

Updates account in database.

- **Version** 0.0.4

- **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 474]

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 368]

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 314*]

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 431*]

Function Parameters:

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

Sets PACC days count.

Sets PACC days count.

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

void function OTS_Account::setPassword(\$password) [*line 341*]

Function Parameters:

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

Sets account's password.

Sets account's password.

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

void function OTS_Account::unban() [*line 541*]

Deletes ban from current account.

Deletes ban from current account.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN
- **Access** public

void function OTS_Account::unlock() [line 393]

Unblocks account.

Unblocks account.

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

void function OTS_Account::__clone() [line 85]

Creates clone of object.

Creates clone of object.

Copy of object needs to have different ID.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

void function OTS_Account::__set_state(\$properties) [line 100]

Function Parameters:

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

Magic PHP5 method.

Magic PHP5 method.

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

- **Version** 0.0.4
- **Version** 0.0.1
- **Static**
- **Since** 0.0.4
- **Access** public

array function OTS_Account::__sleep() [*line 57*]

Magic PHP5 method.

Magic PHP5 method.

Allows object serialisation.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

void function OTS_Account::__wakeup() [*line 71*]

Magic PHP5 method.

Magic PHP5 method.

Allows object unserialisation.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

Class OTS_Accounts_List

[line 21]

List of accounts.

List of accounts.

- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.4

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 220]

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 170*]

Returns current row.

Returns current row.

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

void function OTS_Accounts_List::deleteAccount(\$account) [*line 160*]

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 192*]

Current cursor position.

Current cursor position.

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

void function OTS_Accounts_List::next() [line 182]

Moves to next row.

Moves to next row.

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

void function OTS_Accounts_List::rewind() [line 210]

Select accounts from database.

Select accounts from database.

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

void function OTS_Accounts_List::setLimit([\$limit = false]) [line 125]

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 142*]

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 202*]

Checks if there are any rows left.

Checks if there are any rows left.

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

void function OTS_Accounts_List::__set_state(\$properties) [*line 100*]

Function Parameters:

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

Magic PHP5 method.

Magic PHP5 method.

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

- **Version** 0.0.4
- **Version** 0.0.1
- **Static**
- **Since** 0.0.4
- **Access** public

array function OTS_Accounts_List::__sleep() [*line 71*]

Magic PHP5 method.

Magic PHP5 method.

Allows object serialisation.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

void function OTS_Accounts_List::__wakeup() [*line 85*]

Magic PHP5 method.

Magic PHP5 method.

Allows object unserialisation.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **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 its 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 against 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** 0.0.4
- **Since** 0.0.4

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** 0.0.4
- **See** [POT::connect\(\)](#)
- **Since** 0.0.4
- **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** 0.0.4
- **Since** 0.0.4
- **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** 0.0.4
- **Since** 0.0.4
- **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** 0.0.4
- **Since** 0.0.4
- **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 againsts other values.

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

Class OTS_DB_PostgreSQL

[line 20]

PostgreSQL connection interface.

PostgreSQL connection interface.

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

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** 0.0.4
- **See** [POT::connect\(\)](#)
- **Since** 0.0.4
- **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** 0.0.4
- **Since** 0.0.4
- **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** 0.0.4
- **Since** 0.0.4
- **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** 0.0.4
- **Since** 0.0.4
- **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 against other values.

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

Class OTS_DB_SQLite

[*line 21*]

SQLite connection interface.

SQLite connection interface.

- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.4

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** 0.0.4

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 240]

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 327*]

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 213*]

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 169*]

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 267*]

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 294*]

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 186*]

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 376*]

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::isLoaded() [*line 136*]

Checks if object is loaded.

Checks if object is loaded.

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

void function OTS_Group::load(\$id) [line 125]

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 144]

Saves account in database.

Saves account in database.

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

void function OTS_Group::setAccess(\$access) [line 255]

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 353]`

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 228]

Function Parameters:

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

Sets rights flags.

Sets rights flags.

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

void function OTS_Group::setMaxDepotItems(\$maxdepotitems) [line 282]

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 309]

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 201]

Function Parameters:

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

Sets group's name.

Sets group's name.

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

void function OTS_Group::__clone() [line 85]

Creates clone of object.

Creates clone of object.

Copy of object needs to have different ID.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4

- **Access** public

void function OTS_Group::__set_state(\$properties) [line 100]

Function Parameters:

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

Magic PHP5 method.

Magic PHP5 method.

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

- **Version** 0.0.4
- **Version** 0.0.1
- **Static**
- **Since** 0.0.4
- **Access** public

array function OTS_Group::__sleep() [line 57]

Magic PHP5 method.

Magic PHP5 method.

Allows object serialisation.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

void function OTS_Group::__wakeup() [*line 71*]

Magic PHP5 method.

Magic PHP5 method.

Allows object unserialisation.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

Class OTS_Groups_List

[*line 21*]

List of groups.

List of groups.

- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.4

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 220*]

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 170*]

Returns current row.

Returns current row.

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

void function OTS_Groups_List::deleteGroup(\$group) [*line 160*]

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 192*]

Current cursor position.

Current cursor position.

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

void function OTS_Groups_List::next() [*line 182*]

Moves to next row.

Moves to next row.

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

void function OTS_Groups_List::rewind() [*line 210*]

Select groups from database.

Select groups from database.

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

void function OTS_Groups_List::setLimit([\$limit = false]) [*line 125*]

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 142*]

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 202*]

Checks if there are any rows left.

Checks if there are any rows left.

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

void function OTS_Groups_List::__set_state(\$properties) [*line 100*]

Function Parameters:

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

Magic PHP5 method.

Magic PHP5 method.

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

- **Version** 0.0.4
- **Version** 0.0.1
- **Static**
- **Since** 0.0.4
- **Access** public

array function OTS_Groups_List::__sleep() [*line 71*]

Magic PHP5 method.

Magic PHP5 method.

Allows object serialisation.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

void function OTS_Groups_List::__wakeup() [line 85]

Magic PHP5 method.

Magic PHP5 method.
Allows object unserialisation.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

Class OTS_Guild

[line 20]

OTServ guild abstraction.
OTServ guild abstraction.

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

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

void function OTS_Guild::acceptInvite(\$player) [*line 459*]

Function Parameters:

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

Finalise invitation.

Finalise invitation.

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

void function OTS_Guild::acceptRequest(\$player) [line 551]

Function Parameters:

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

Accepts player.

Accepts player.

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

void function OTS_Guild::deleteInvite(\$player) [line 436]

Function Parameters:

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

Deletes invitation for player to guild.

Deletes invitation for player to guild.

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

void function OTS_Guild::deleteRequest(\$player) [line 528]

Function Parameters:

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

Deletes request from player.

Deletes request from player.

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

void function OTS_Guild::find(\$name) [line 167]

Function Parameters:

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

Loads guild by it's name.

Loads guild by it's name.

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

int function OTS_Guild::getCreationData() [*line 286*]

Guild creation data.

Guild creation data.

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

string function OTS_Guild::getCustomField(\$field) [*line 317*]

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

array function OTS_Guild::getGuildRanks() [*line 363*]

Reads all ranks that are in this guild.

Reads all ranks that are in this guild.

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

int function OTS_Guild::getId() [*line 216*]

Guild ID.

Guild ID.

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

string function OTS_Guild::getName() [*line 232*]

Guild name.

Guild name.

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

OTS_Player function OTS_Guild::getOwner() [*line 258*]

Returns owning player of this player.

Returns owning player of this player.

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

void function OTS_Guild::invite(\$player) [*line 413*]

Function Parameters:

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

Invites player to guild.

Invites player to guild.

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

bool function OTS_Guild::isLoaded() [*line 184*]

Checks if object is loaded.

Checks if object is loaded.

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

array function OTS_Guild::listInvites() [*line 390*]

Returns list of invited players.

Returns list of invited players.

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

array function OTS_Guild::listRequests() [*line 482*]

Returns list of players that requested membership.

Returns list of players that requested membership.

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

void function OTS_Guild::load(\$id) [*line 156*]

Function Parameters:

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

Loads guild with given id.

Loads guild with given id.

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

void function OTS_Guild::request(\$player) [*line 505*]

Function Parameters:

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

Requests membership in guild for player player.

Requests membership in guild for player player.

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

void function OTS_Guild::save() [*line 192*]

Saves guild in database.

Saves guild in database.

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

void function OTS_Guild::setCreationData(\$creationdata) [line 301]

Function Parameters:

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

Sets guild creation data.

Sets guild creation data.

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

void function OTS_Guild::setCustomField(\$field, \$value) [line 341]

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

void function OTS_Guild::setInvitesDriver([\$invites = null]) [line 136]

Function Parameters:

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

Assigns invites handler.

Assigns invites handler.

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

void function OTS_Guild::setName(\$name) [line 247]

Function Parameters:

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

Sets players's name.

Sets players's name.

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

void function OTS_Guild::setOwner(\$owner) [line 275]

Function Parameters:

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

Assigns guild to owner.

Assigns guild to owner.

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

void function OTS_Guild::setRequestsDriver([\$requests = null]) [line 146]

Function Parameters:

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

Assigns requests handler.

Assigns requests handler.

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

void function OTS_Guild::__clone() [line 92]

Creates clone of object.

Creates clone of object.

Copy of object needs to have different ID.

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

void function OTS_Guild::__set_state(\$properties) [line 111]

Function Parameters:

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

Magic PHP5 method.

Magic PHP5 method.

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

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

- **Access** public

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

Magic PHP5 method.

Magic PHP5 method.

Allows object serialisation.

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

void function OTS_Guild::__wakeup() *[line 80]*

Magic PHP5 method.

Magic PHP5 method.

Allows object unserialisation.

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

Class OTS_GuildRank *[line 20]*

OTServ guild rank abstraction.

OTServ guild rank abstraction.

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

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

void function OTS_GuildRank::find(\$name, [\$guild = null]) [*line 130*]

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

string function OTS_GuildRank::getCustomField(\$field) [*line 288*]

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

OTS_Guild function OTS_GuildRank::getGuild() [*line 229*]

Returns guild of this rank.

Returns guild of this rank.

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

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

int function OTS_GuildRank::getId() [*line 187*]

Rank ID.

Rank ID.

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

int function OTS_GuildRank::getLevel() [*line 257*]

Rank's access level.

Rank's access level.

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

string function OTS_GuildRank::getName() [*line 203*]

Rank name.

Rank name.

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

array function OTS_GuildRank::getPlayers() [*line 334*]

Reads all players who has this rank set.

Reads all players who has this rank set.

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

bool function OTS_GuildRank::isLoading() [*line 155*]

Checks if object is loaded.

Checks if object is loaded.

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

void function OTS_GuildRank::load(\$id) [*line 116*]

Function Parameters:

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

Loads rank with given id.

Loads rank with given id.

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

void function OTS_GuildRank::save() [line 163]

Saves rank in database.

Saves rank in database.

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

void function OTS_GuildRank::setCustomField(\$field, \$value) [line 312]

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 improper 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 a string ('1') instead of passing it as an integer.

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

void function OTS_GuildRank::setGuild(\$guild) [line 246]

Function Parameters:

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

Assigns rank to guild.

Assigns rank to guild.

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

void function OTS_GuildRank::setLevel(\$level) [line 272]

Function Parameters:

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

Sets rank's access level within guild.

Sets rank's access level within guild.

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

void function OTS_GuildRank::setName(\$name) [line 218]

Function Parameters:

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

Sets rank's name.

Sets rank's name.

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

void function OTS_GuildRank::__clone() [line 78]

Creates clone of object.

Creates clone of object.

Copy of object needs to have different ID.

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

- **Access** public

void function OTS_GuildRank::__set_state(\$properties) [line 91]

Function Parameters:

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

Magic PHP5 method.

Magic PHP5 method.

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

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

array function OTS_GuildRank::__sleep() [line 54]

Magic PHP5 method.

Magic PHP5 method.

Allows object serialisation.

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

void function OTS_GuildRank::__wakeup() [line 66]

Magic PHP5 method.

Magic PHP5 method.

Allows object unserialisation.

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

Class OTS_GuildRanks_List

[line 20]

List of guild ranks.

List of guild ranks.

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

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

int function OTS_GuildRanks_List::count() [*line 212*]

Returns number of ranks on list in current criterium.

Returns number of ranks on list in current criterium.

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

OTS_GuildRank function OTS_GuildRanks_List::current() [*line 162*]

Returns current row.

Returns current row.

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

void function OTS_GuildRanks_List::deleteGuildRank(\$guildRank) [*line 152*]

Function Parameters:

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

Deletes guild rank.

Deletes guild rank.

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

mixed function OTS_GuildRanks_List::key() [*line 184*]

Current cursor position.

Current cursor position.

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

void function OTS_GuildRanks_List::next() [*line 174*]

Moves to next row.

Moves to next row.

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

void function OTS_GuildRanks_List::rewind() [*line 202*]

Select ranks from database.

Select ranks from database.

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

void function OTS_GuildRanks_List::setLimit([\$limit = false]) [line 118]

Function Parameters:

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

Sets LIMIT.

Sets LIMIT.

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

void function OTS_GuildRanks_List::setOffset([\$offset = false]) [line 135]

Function Parameters:

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

Sets OFFSET.

Sets OFFSET.

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

bool function OTS_GuildRanks_List::valid() [*line 194*]

Checks if there are any rows left.

Checks if there are any rows left.

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

void function OTS_GuildRanks_List::__set_state(\$properties) [*line 93*]

Function Parameters:

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

Magic PHP5 method.

Magic PHP5 method.

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

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

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

Magic PHP5 method.

Magic PHP5 method.

Allows object serialisation.

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

void function OTS_GuildRanks_List::__wakeup() [*line 80*]

Magic PHP5 method.

Magic PHP5 method.

Allows object unserialisation.

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

Class OTS_Guilds_List

[*line 20*]

List of guilds.

List of guilds.

- **Package** POT

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

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

int function OTS_Guilds_List::count() [*line 212*]

Returns number of guilds on list in current criterium.

Returns number of guilds on list in current criterium.

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

OTS_Guild function OTS_Guilds_List::current() [*line 162*]

Returns current row.

Returns current row.

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

void function OTS_Guilds_List::deleteGuild(\$guild) [*line 152*]

Function Parameters:

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

Deletes guild.

Deletes guild.

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

mixed function OTS_Guilds_List::key() [*line 184*]

Current cursor position.

Current cursor position.

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

void function OTS_Guilds_List::next() [line 174]

Moves to next row.

Moves to next row.

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

void function OTS_Guilds_List::rewind() [line 202]

Select guilds from database.

Select guilds from database.

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

void function OTS_Guilds_List::setLimit([\$limit = false]) [line 118]

Function Parameters:

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

Sets LIMIT.

Sets LIMIT.

- **Version** 0.0.4

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

void function OTS_Guilds_List::setOffset([\$offset = false]) [*line 135*]

Function Parameters:

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

Sets OFFSET.

Sets OFFSET.

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

bool function OTS_Guilds_List::valid() [*line 194*]

Checks if there are any rows left.

Checks if there are any rows left.

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

void function OTS_Guilds_List::__set_state(\$properties) [*line 93*]

Function Parameters:

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

Magic PHP5 method.

Magic PHP5 method.

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

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

array function OTS_Guilds_List::__sleep() [line 68]

Magic PHP5 method.

Magic PHP5 method.

Allows object serialisation.

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

void function OTS_Guilds_List::__wakeup() [line 80]

Magic PHP5 method.

Magic PHP5 method.

Allows object unserialisation.

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

- **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** 0.0.4+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::ban([\$time = 0]) [line 1721]

Function Parameters:

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

Bans current player.

Bans current player.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN
- **Access** public

void function OTS_Player::find(\$name) [line 157]

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 259*]

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 914*]

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 1028*]

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 1327*]

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 1602*]

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** 0.0.4
- **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 644*]

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 401*]

Experience points.

Experience points.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTTS_NotLoaded If player is not loaded.
- **Access** public

OTS_Group function OTS_Player::getGroup() [*line 288*]

Returns group of this player.

Returns group of this player.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTTS_NotLoaded If player is not loaded.
- **Access** public

string function OTS_Player::getGuildNick() [*line 1115*]

Guild nick.

Guild nick.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTTS_NotLoaded If player is not loaded.
- **Access** public

int function OTS_Player::getHealth() [*line 482*]

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 509*]

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 215*]

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 968*]

Last login IP.

Last login IP.

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

int function OTS_Player::getLastLogin() [*line 941*]

Last login timestamp.

Last login timestamp.

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

int function OTS_Player::getLevel() [*line 428*]

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 806*]

Addons.

Addons.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTTS_NotLoaded If player is not loaded.
- **Access** public

int function OTS_Player::getLookBody() [*line 671*]

Body color.

Body color.

- **Version** 0.0.3
- **Version** 0.0.1
- **Throws** E_OTTS_NotLoaded If player is not loaded.
- **Access** public

int function OTS_Player::getLookFeet() [*line 698*]

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 725*]

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 752*]

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 779*]

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 1238*]

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 1265*]

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 1292*]

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 455*]

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 536*]

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 563*]

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 590*]

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 232*]

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 833*]

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 860*]

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 887*]

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 318*]

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 1159*]

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 1143*]

Guild rank ID.

Guild rank ID.

- **Version** 0.0.3
- **Version** 0.0.1
- **Deprecated** 0.0.4 Use getRank().
- **Throws** E_OTS_NotLoaded If player is not loaded.
- **Access** public

int function OTS_Player::getRedSkullTime() [*line 1055*]

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 347*]

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 1382*]

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 1414*]

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 1467*]

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** 0.0.4
- **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 617*]

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 1211*]

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 374*]

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 1082*]

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::isBanned() [*line 1756*]

Checks if player is banned.

Checks if player is banned.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN
- **Access** public

bool function OTS_Player::isLoading() [*line 174*]

Checks if object is loaded.

Checks if object is loaded.

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

bool function OTS_Player::isSaveSet() [*line 995*]

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 136]

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 184]

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 276]

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 929]
Function Parameters:

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

Sets capacity.
Sets capacity.

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

void function OTS_Player::setConditions(\$conditions) [line 1043]
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 1357]

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 1657]

Function Parameters:

- *int* **\$depot** Depot ID to save items.
- [OTS_Item](#) **\$item** Item (can be a container with content) for given depot. Leave this parameter blank to clear depot.
- *int* **\$pid** Deprecated, not used anymore.
- *int* **\$depot_id** Internal, for further use.

Sets depot content.

Sets depot content.

- **Version** 0.0.4
- **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 659*]

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 416]

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 305]

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 1130]

Function Parameters:

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

Sets guild nick.

Sets guild nick.

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

void function OTS_Player::setHealth(\$health) [line 497]

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 524]

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 983]

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 956]

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 443]

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 821*]

Function Parameters:

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

Sets addons.

Sets addons.

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

void function OTS_Player::setLookBody(\$lookbody) [*line 686*]

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 713]

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 740]

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 767]

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 794]

Function Parameters:

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

Sets outfit.

Sets outfit.

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

void function OTS_Player::setLossExperience(\$loss_experience) [line 1253]

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 1280]
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 1307]
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 470]

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 551]

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 578]

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 605*]

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 247*]

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 848]

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 875]

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 902]

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 335]

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 1192]

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 1182]*

Function Parameters:

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

Sets guild rank ID.

Sets guild rank ID.

- **Version** 0.0.1
- **Deprecated** 0.0.4 Use setRank().
- **Access** public

void function OTS_Player::setRedSkull() *[line 1103]*

Sets red skull flag.

Sets red skull flag.

- **Version** 0.0.1

- **Access** public

void function OTS_Player::setRedSkullTime(\$redskulltime) [line 1070]

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 1016]

Sets save flag.

Sets save flag.

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

void function OTS_Player::setSex(\$sex) [line 362]

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 1400]

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 1432]

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 1521]

Function Parameters:

- *int* **\$slot** Slot to save items.
- [*OTS_Item*](#) **\$item** Item (can be a container with content) for given slot. Leave this parameter blank to clear slot.
- *int* **\$pid** Deprecated, not used anymore.

Sets slot content.

Sets slot content.

- **Version** 0.0.4
- **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 632]

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 1226]

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 389]

Function Parameters:

- *int* **\$vocation** Player proffesion.

Sets player proffesion.

Sets player proffesion.

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

void function OTS_Player::unban() [line 1738]

Deletes ban from current player.

Deletes ban from current player.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN
- **Access** public

void function OTS_Player::unsetRedSkull() [line 1095]

Unsets red skull flag.

Unsets red skull flag.

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

void function OTS_Player::unsetSave() [line 1008]

Unsets save flag.

Unsets save flag.

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

void function OTS_Player::__clone() [*line 95*]

Creates clone of object.

Creates clone of object.

Copy of object needs to have different ID.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

void function OTS_Player::__set_state(\$properties) [*line 110*]

Function Parameters:

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

Magic PHP5 method.

Magic PHP5 method.

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

- **Version** 0.0.4
- **Version** 0.0.1
- **Static**
- **Since** 0.0.4
- **Access** public

array function OTS_Player::__sleep() [*line 67*]

Magic PHP5 method.

Magic PHP5 method.
Allows object serialisation.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

void function OTS_Player::__wakeup() [line 81]

Magic PHP5 method.

Magic PHP5 method.
Allows object unserialisation.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

Class OTS_Players_List

[line 21]

List of players.

List of players.

- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.4

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 220]

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 170]

Returns current row.

Returns current row.

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

void function OTS_Players_List::deletePlayer(\$player) [*line 160*]

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 192*]

Current cursor position.

Current cursor position.

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

void function OTS_Players_List::next() [*line 182*]

Moves to next row.

Moves to next row.

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

void function OTS_Players_List::rewind() [line 210]

Select players from database.

Select players from database.

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

void function OTS_Players_List::setLimit([\$limit = false]) [line 125]

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 142]

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 202*]

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

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

void function OTS_Players_List::__set_state(\$properties) [*line 100*]

Function Parameters:

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

Magic PHP5 method.
Magic PHP5 method.
Allows object importing from [var_export\(\)](#).

- **Version** 0.0.4
- **Version** 0.0.1

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

array function OTS_Players_List::__sleep() [*line 71*]

Magic PHP5 method.

Magic PHP5 method.

Allows object serialisation.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

void function OTS_Players_List::__wakeup() [*line 85*]

Magic PHP5 method.

Magic PHP5 method.

Allows object unserialisation.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

Class POT

[line 28]

Main POT class.

Main POT class.

- **Package** POT
- **Version** 0.0.1
- **Version** 0.0.4+SVN

POT::BAN_ACCOUNT

= 3 [line 261]

Account ban.

Account ban.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN

POT::BAN_IP

= 1 [line 245]

IP ban.

IP ban.

- **Version** 0.0.4+SVN
- **Version** 0.0.1

- **Since** 0.0.4+SVN

POT::BAN_PLAYER

= 2 [*line 253*]

Player ban.

Player ban.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN

POT::DB_MYSQL

= 1 [*line 33*]

MySQL driver.

MySQL driver.

- **Version** 0.0.1

POT::DB_ODBC

= 4 [*line 51*]

ODBC driver.

ODBC driver.

- **Version** 0.0.4

- **Version** 0.0.1
- **Since** 0.0.4

POT::DB_PGSQL

= 3 *[line 44]*

PostgreSQL driver.
PostgreSQL driver.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4

POT::DB_SQLITE

= 2 *[line 37]*

SQLite driver.
SQLite driver.

- **Version** 0.0.1

POT::DEPOT_SID_FIRST

= 100 *[line 237]*

First depot item sid.
First depot item sid.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4

POT::DIRECTION_EAST

= 1 [*line 100*]

East.

East.

- **Version** 0.0.1

POT::DIRECTION_NORTH

= 0 [*line 96*]

North.

North.

- **Version** 0.0.1

POT::DIRECTION_SOUTH

= 2 [*line 104*]

South.

South.

- **Version 0.0.1**

POT::DIRECTION_WEST

= 3 [*line 108*]

West.

West.

- **Version 0.0.1**

POT::SEX_FEMALE

= 0 [*line 56*]

Female gender.

Female gender.

- **Version 0.0.1**

POT::SEX_MALE

= 1 [*line 60*]

Male gender.

Male gender.

- **Version 0.0.1**

POT::SKILL_AXE

= 3 [*line 137*]

Axe fighting.
Axe fighting.

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

POT::SKILL_CLUB

= 1 [*line 123*]

Club fighting.
Club fighting.

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

POT::SKILL_DISTANCE

= 4 [*line 144*]

Distance fighting.
Distance fighting.

- **Version** 0.0.2

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

POT::SKILL_FISHING

= 6 [*line 158*]

Fishing.

Fishing.

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

POT::SKILL_FIST

= 0 [*line 116*]

Fist fighting.

Fist fighting.

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

POT::SKILL_SHIELDING

= 5 [*line 151*]

Shielding.

Shielding.

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

POT::SKILL_SWORD

= 2 [*line 130*]

Sword fighting.
Sword fighting.

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

POT::SLOT_AMMO

= 10 [*line 229*]

Ammunition slot.
Ammunition slot.

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

POT::SLOT_ARMOR

= 4 *[line 187]*

Armor slot.

Armor slot.

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

POT::SLOT_BACKPACK

= 3 *[line 180]*

Backpack slot.

Backpack slot.

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

POT::SLOT_FEET

= 8 *[line 215]*

Boots slot.

Boots slot.

- **Version** 0.0.3

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

POT::SLOT_HEAD

= 1 [*line 166*]

Head slot.

Head slot.

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

POT::SLOT_LEFT

= 6 [*line 201*]

Left hand slot.

Left hand slot.

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

POT::SLOT_LEGS

= 7 [*line 208*]

Legs slot.

Legs slot.

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

POT::SLOT_NECKLACE

= 2 [*line 173*]

Necklace slot.

Necklace slot.

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

POT::SLOT_RIGHT

= 5 [*line 194*]

Right hand slot.

Right hand slot.

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

POT::SLOT_RING

= 9 *[line 222]*

Ring slot.

Ring slot.

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

POT::VOCATION_DRUID

= 2 *[line 79]*

Druid.

Druid.

- **Version** 0.0.1
- **Deprecated** 0.0.4+SVN Vocations are now loaded dynamicly from vocations.xml file.

POT::VOCATION_KNIGHT

= 4 *[line 91]*

Knight.

Knight.

- **Version** 0.0.1
- **Deprecated** 0.0.4+SVN Vocations are now loaded dynamicly from vocations.xml file.

POT::VOCATION_NONE

= 0 *[line 67]*

None vocation.

None vocation.

- **Version** 0.0.1
- **Deprecated** 0.0.4+SVN Vocations are now loaded dynamicly from vocations.xml file.

POT::VOCATION_PALADIN

= 3 *[line 85]*

Paladin.

Paladin.

- **Version** 0.0.1
- **Deprecated** 0.0.4+SVN Vocations are now loaded dynamicly from vocations.xml file.

POT::VOCATION_SORCERER

= 1 *[line 73]*

Sorcerer.

Sorcerer.

- **Version** 0.0.1

- **Deprecated** 0.0.4+SVN Vocations are now loaded dynamicly from vocations.xml file.

void function POT::banIP(\$ip, [\$mask = '255.255.255.255'], [\$time = 0]) [line 602]

Function Parameters:

- *string* **\$ip** IP to ban.
- *string* **\$mask** Mask for ban (by default bans only given IP).
- *int* **\$time** Time for time until expires (0 - forever).

Bans given IP number.

Bans given IP number.

Adds IP/mask ban. You can call this function with only one parameter to ban only given IP address without expiration.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN
- **Access** public

void function POT::connect(\$driver, \$params) [line 384]

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 than one OTServ installed on one database.

- **Version** 0.0.1
- **Version** 0.0.4
- **Throws** Exception When driver is not supported.
- **Access** public
- **Example**

IOTS_DAO function POT::createObject(\$class) [line 435]

Function Parameters:

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

Creates OTServ DAO class instance.

Creates OTServ DAO class instance.

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

LOTS_DB function POT::getDBHandle() [*line 510*]

Returns database connection handle.

Returns database connection handle.

At all you shouldn't use this method and work with database using POT classes, but it may be sometime necessary to use direct database access (mainly until POT won't provide many important features).

It is also important as serialised objects after unserialisation needs to be re-initialised with database connection.

- **Version** 0.0.4
- **Version** 0.0.1
- **Since** 0.0.4
- **Access** public

POT function POT::getInstance() [*line 268*]

Singleton.

Singleton.

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

int|bool function POT::getVocationID(\$name) [*line 554*]

Function Parameters:

- *string* **\$name** Vocation.

Returns vocation's ID.

Returns vocation's ID.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN
- **Access** public

string|bool function POT::getVocationName(\$id) [*line 567*]

Function Parameters:

- *int* **\$id** Vocation ID.

Returns name of given vocation's ID.

Returns name of given vocation's ID.

- **Version** 0.0.4+SVN

- **Version** 0.0.1
- **Since** 0.0.4+SVN
- **Access** public

array function POT::getVocationsList() [*line 586*]

Returns list (id => name) of loaded vocations.

Returns list (id => name) of loaded vocations.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN
- **Access** public

bool function POT::isIPBanned(\$ip) [*line 670*]

Function Parameters:

- *string* **\$ip** IP to ban.

Checks if given IP is banned.

Checks if given IP is banned.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN
- **Access** public

void function POT::loadClass(\$class) [line 343]

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

void function POT::loadVocations(\$file) [line 533]

Function Parameters:

- *string* **\$file** vocations.xml file location.

Loads vocations list.

Loads vocations list.

Loads vocations list from given file.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN

- **Access public**

OTS_InfoRespond|bool function POT::serverStatus(\$server, \$port) [line 453]

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 299]

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**

void function POT::unbanIP(\$ip, [\$mask = '255.255.255.255']) [line 637]

Function Parameters:

- *string* **\$ip** IP to ban.
- *string* **\$mask** Mask for ban (by default 255.255.255.255).

Deletes ban from given IP number.

Deletes ban from given IP number.

Removes given IP/mask ban.

- **Version** 0.0.4+SVN
- **Version** 0.0.1
- **Since** 0.0.4+SVN
- **Access** public

compat.php

POT compatibility assurance package.

POT compatibility assurance package.

This package makes you sure that POT scripts won't cause FATAL errors on PHP older PHP 5.x versions. However remember that some PHP features won't be enabled with it. For example if you have PHP 5.0.x, this package will define Countable interface for you so PHP will know it, but it won't allow you to use count(\$countableObject) structure.

- **Package** POT
- **Sub-Package** compat
- **Author** Wrzasq < wrzasq@gmail.com>
- **Version** 0.0.2
- **Copyright** 2007 (C) by Wrzasq
- **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](#)

Appendix B - README/CHANGELOG/INSTALL

CHANGELOG

[0.0.4+SVN]

- * Added support for vocations.xml file. <wrzasq>
- * Added support for bans. <wrzasq>

[0.0.4]

- * Added guild system support (guilds, ranks, invitations and requests drivers mechanisms). <wrzasq>
- * Added account group support. <wrzasq>
- * Added support for depot_id field (it is reserved in OTServ for further use). <wrzasq>
- * Added PostgreSQL and ODBC drivers. <wrzasq>
- * Added __sleep() and __wakeup() methods to allow POT objects to be stored in sessions. <wrzasq>
- * Added __clone() methods to allow save ID-losing cloning of POT objects. <wrzasq>
- * Added __set_state() methods. <wrzasq>
- * Updated players table structure. <wrzasq>
- * Dropped REGEXP operator bindings - not used anywhere. <wrzasq>
- * Fixed items loading and saving. <wrzasq>
- * Fixed typos. <wrzasq>

[0.0.3]

- * Added custom fields support. <wrzasq>
- * Added items and depots support. <wrzasq>
- * Added support for players PACC timestamps. <wrzasq>
- * Fixed loading skills. <wrzasq>
- * Replaced E_USER_* with exceptions. <wrzasq>
- * Uses fetchAll() in loops to prevent MySQL buffering problems. <wrzasq>
- * Restricted access to POT class constructor to make sure it won't be instantiated directly. <wrzasq>

[0.0.2]

- * Added "compat" library for POT. <wrzasq>
- * Added skills support in OTS_Player class. <wrzasq>
- * Added 'info' serverStatus() method and respond handler for server status protocol. <wrzasq>
- * Fixed `redskulltime` field name in OTS_Player. <wrzasq>
- * Fixed 'password' parameter for DB_MYSQL driver. <wrzasq>
- * Added find() to OTS_Account class to load accounts by their's e-mail addresses. <wrzasq>
- * POT class now automatically binds own __autoload() handler with spl_autoload_register(). <wrzasq>

[0.0.1]

- * Initial release. <wrzasq>

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.
CHANGELOG - changes history.
INSTALL - installation tutorial.
LICENSE - POT license (GNU LGPL v3), if you don't accept it - don't use any of those scripts.
NEWS - changes in current release.
README - this readme file.
RULES - rules to be followed during developing contributed code.
Makefile - make input, for documentation generation.
documentation.pdf - phpDocumentor-generator documentation in PDF format.
compat.php - Compatibility assurance library.
test.php - phpUnit test suite.

===== Makefile =====

Makefile contains some targets for make that can help in development. Makefile requires following command-line commands:

php: PHP CLI interface.
phpdoc: phpDocumentor.
phpunit: PHPUnit testing framework.

Possible targets:

all: default one, runs all other targets (in order: clean, check, documentation, pdf, online, test, package).
clean: deletes documentation.
check: checks syntax of all PHP files.
documentation: generates HTML documentation.
pdf: generates PDF documentation.
online: OTServ-AAC website documentation template used.
test: runs test suite.
package: creates pot.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.

For more info see AUTHORS file in OTServ tree.

INSTALL

POT is a toolkit which means you don't literally install it. You copy it's files and write code for it. All source files are located in classes/ subdirectory. Copy them to your script directory.

You can put main file - OTS.php in different directory then other files.

For information about how to include POT in your code see the documentation.

NEWS

What's new in 0.0.4 version?

- * Added guild system support (guilds, ranks, invitations and requests drivers mechanisms).

Main feature of new update - includes full support for guilds system.

- * Added account group support.

Support for new accounts table structure.

- * Added support for depot_id field (it is reserved in OTServ for further use).

Even though depot_id field in OTServ database is not used, it was moved back as it is reserved for further use. POT also supports it.

- * Added PostgreSQL and ODBC drivers.

POT supports new revdbsys database drivers.

- * Added __sleep() and __wakeup() methods to allow POT objects to be stored in sessions.

Allows you to store POT objects in sessions.

- * Added __clone() methods to allow save ID-losing cloning of POT objects.

Allows you cloning of POT objects while deleting their's IDs for re-inserting.

- * Added __set_state() methods.

Magic PHP5 method - you can read more in PHP Manual.

- * Updated players table structure.

As always POT keeps your scripts up-to-date with support for latest structure of database.

- * Dropped REGEXP operator bindings - not used anywhere.

This internal feature wasn't used anywhere so it would be a waste to keep it out there.

- * Fixed items loading and saving.

Items saving (both body slots and depot items) in previous versions was coded wrong way. Fixed now.

- * Fixed typos.

Like always there were some bugs which of course were fixed.

Appendix D - Todo List

In Package POT

In [OTS.php](#)

- Implement `__get()/__set()/__call()/__toString()`; Iterator, ArrayAccess, Countable interfaces.
- Items list (items.xml + items.otb -> cache).
- List objects sorting/criteria-based loading.
- More detailed documentation, better examples, more detailed phpUnit tests.
- Spawns support (OTBM support -> cache).

Index

A

[Account number hack](#) 18

C

[constructor OTS Guilds List:: construct\(\)](#) 132
 Sets database connection handler.
[constructor OTS GuildRanks List:: construct\(\)](#) 126
 Sets database connection handler.
[constructor OTS GuildRank:: construct\(\)](#) 118
 Sets database connection handler.
[constructor OTS Guild:: construct\(\)](#) 105
 Sets database connection handler.
[constructor OTS Item:: construct\(\)](#) 144
 Creates item of given ID.
[constructor OTS Player:: construct\(\)](#) 147
 Sets database connection handler.
[CHANGELOG](#) 224
[compat.php](#) 217
 POT compatibility assurance package.
[constructor OTS Players List:: construct\(\)](#) 190
 Sets database connection handler.
[constructor OTS Groups List:: construct\(\)](#) 99
 Sets database connection handler.
[constructor OTS Group:: construct\(\)](#) 90
 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 OTS Accounts List:: construct\(\)](#) 68
 Sets database connection handler.
[constructor OTS DB MySQL:: construct\(\)](#) 77
 Creates database connection.
[constructor OTS DB SQLite:: construct\(\)](#) 87
 Creates database connection.
[constructor OTS DB PostgreSQL:: construct\(\)](#) 83
 Creates database connection.
[constructor OTS DB ODBC:: construct\(\)](#) 80
 Creates database connection.
[constructor IOTS DAO:: construct\(\)](#) 47
 DAO objects must be initialized with a database.

D

DAO objects	9
-----------------------------	---

E

E_OTs_NotLoaded	46
<i>Occurs when code attempts to access property of not loaded object.</i>	
E_OTs_NoDriver	46
<i>Occurs when code attempts to execute driven action that has no assigned driver to handle it.</i>	
E_OTs_NotLoaded.php	23
E_OTs_NoDriver.php	22

G

Guild action drivers	15
Guilds	13

I

IOTS_GuildAction::addRequest()	52
<i>Adds new request.</i>	
IOTS_GuildAction	51
<i>Guild action interface.</i>	
IOTS_DB::tableName()	50
<i>Query-quoted table name.</i>	
IOTS_GuildAction::deleteRequest()	52
<i>Deletes request.</i>	
IOTS_GuildAction::listRequests()	53
<i>List of saved pending actions.</i>	
INSTALL	226
IOTS_GuildAction::submitRequest()	53
<i>Finalizes request.</i>	
IOTS_DB::SQLquote()	50
<i>Query-quoted string value.</i>	
IOTS_DB::SQLquery()	50
<i>Evaluates query.</i>	
IOTS_DAO	47
<i>OTServ database object.</i>	
IOTS_GuildAction.php	26
IOTS_DB.php	25
IOTS_DB	48
<i>OTServ database handler interface.</i>	
IOTS_DB::fieldName()	48
<i>Query-quoted field name.</i>	
IOTS_DB::limit()	49
<i>LIMIT/OFFSET clause for queries.</i>	
IOTS_DB::lastInsertId()	49
<i>ID of last created record.</i>	
IOTS_DAO.php	24

N

NEWS	226
--------------------------------	-----

O

OTS_Player::ban()	148
<i>Bans current player.</i>	
OTS_Player	147
<i>OTServ character abstraction.</i>	
OTS_Item::setCount()	147
<i>Sets count of item.</i>	
OTS_Item::setAttributes()	146
<i>Sets item attributes.</i>	
OTS_Player::find()	148
<i>Loads player by it's name.</i>	
OTS_Player::getAccount()	149
<i>Returns account of this player.</i>	
OTS_Player::getCustomField()	150
<i>Reads custom field.</i>	
OTS_Player::getConditions()	150
<i>Conditions.</i>	
OTS_Player::getCap()	149
<i>Capacity.</i>	
OTS_Item::getId()	146
<i>Returns item type.</i>	
OTS_Item::getCount()	145
<i>Returns count of item.</i>	
OTS_InfoRespond::getTSPQVersion()	143
<i>Returns version of root element.</i>	
OTS_InfoRespond::getServerVersion()	143
<i>Returns server version.</i>	
OTS_InfoRespond::getServer()	142
<i>Returns server attribute.</i>	
OTS_InfoRespond::getPort()	142
<i>Returns server port.</i>	
OTS_InfoRespond::getUptime()	143
<i>Returns server uptime.</i>	
OTS_InfoRespond::getURL()	144
<i>Returns server website.</i>	
OTS_Item::getAttributes()	145
<i>Returns item custom attributes.</i>	
OTS_Item::count()	145
<i>Count value for current item.</i>	
OTS_Item	144
<i>Single item representation.</i>	
OTS_Player::getDepot()	151
<i>Returns items tree from given depot.</i>	
OTS_Player::getDirection()	151
<i>Looking direction.</i>	
OTS_Player::getLookType()	156
<i>Outfit.</i>	

OTS_Player::getLookLegs()	156
<i>Legs color.</i>	
OTS_Player::getLookHead()	156
<i>Hair color.</i>	
OTS_Player::getLookFeet()	155
<i>Boots color.</i>	
OTS_Player::getLossExperience()	157
<i>Percentage of experience lost after dead.</i>	
OTS_Player::getLossMana()	157
<i>Percentage of used mana lost after dead.</i>	
OTS_Player::getMana()	158
<i>Current mana.</i>	
OTS_Player::getMagLevel()	158
<i>Magic level.</i>	
OTS_Player::getLossSkills()	158
<i>Percentage of skills lost after dead.</i>	
OTS_Player::getLookBody()	155
<i>Body color.</i>	
OTS_Player::getLookAddons()	155
<i>Addons.</i>	
OTS_Player::getHealth()	153
<i>Current HP.</i>	
OTS_Player::getGuildNick()	152
<i>Guild nick.</i>	
OTS_Player::getGroup()	152
<i>Returns group of this player.</i>	
OTS_Player::getExperience()	151
<i>Experience points.</i>	
OTS_Player::getHealthMax()	153
<i>Maximum HP.</i>	
OTS_Player::getId()	153
<i>Player ID.</i>	
OTS_Player::getLevel()	154
<i>Experience level.</i>	
OTS_Player::getLastLogin()	154
<i>Last login timestamp.</i>	
OTS_Player::getLastIP()	154
<i>Last login IP.</i>	
OTS_InfoRespond::getPlayersPeak()	142
<i>Returns record of online players.</i>	
OTS_InfoRespond::getOwner()	141
<i>Returns owner name.</i>	
OTS_GuildRanks_List::wakeup()	131
<i>Magic PHP5 method.</i>	
OTS_GuildRanks_List::sleep()	131
<i>Magic PHP5 method.</i>	
OTS_GuildRanks_List::set_state()	130
<i>Magic PHP5 method.</i>	
OTS_GuildRanks_List::valid()	130
<i>Checks if there are any rows left.</i>	
OTS_Guilds_List	131
<i>List of guilds.</i>	
OTS_Guilds_List::count()	132

<i>Returns number of guilds on list in current criterium.</i>	
OTS_Guilds_List::key()	133
<i>Current cursor position.</i>	
OTS_Guilds_List::deleteGuild()	133
<i>Deletes guild.</i>	
OTS_Guilds_List::current()	132
<i>Returns current row.</i>	
OTS_GuildRanks_List::setOffset()	129
<i>Sets OFFSET.</i>	
OTS_GuildRanks_List::setLimit()	129
<i>Sets LIMIT.</i>	
OTS_GuildRanks_List::count()	127
<i>Returns number of ranks on list in current criterium.</i>	
OTS_GuildRanks_List	126
<i>List of guild ranks.</i>	
OTS_GuildRank::wakeup()	125
<i>Magic PHP5 method.</i>	
OTS_GuildRank::sleep()	125
<i>Magic PHP5 method.</i>	
OTS_GuildRanks_List::current()	127
<i>Returns current row.</i>	
OTS_GuildRanks_List::deleteGuildRank()	127
<i>Deletes guild rank.</i>	
OTS_GuildRanks_List::rewind()	128
<i>Select ranks from database.</i>	
OTS_GuildRanks_List::next()	128
<i>Moves to next row.</i>	
OTS_GuildRanks_List::key()	128
<i>Current cursor position.</i>	
OTS_Guilds_List::next()	134
<i>Moves to next row.</i>	
OTS_Guilds_List::rewind()	134
<i>Select guilds from database.</i>	
OTS_InfoRespond::getMapWidth()	139
<i>Returns map width.</i>	
OTS_InfoRespond::getMapName()	139
<i>Returns map name.</i>	
OTS_InfoRespond::getMapHeight()	139
<i>Returns map height.</i>	
OTS_InfoRespond::getMapAuthor()	138
<i>Returns map author.</i>	
OTS_InfoRespond::getMaxPlayers()	140
<i>Returns maximum amount of players online.</i>	
OTS_InfoRespond::getMonstersCount()	140
<i>Returns number of all monsters on map.</i>	
OTS_InfoRespond::getOnlinePlayers()	141
<i>Returns current amount of players online.</i>	
OTS_InfoRespond::getName()	141
<i>Returns server name.</i>	
OTS_InfoRespond::getMOTD()	140
<i>Returns server's Message Of The Day</i>	
OTS_InfoRespond::getLocation()	138
<i>Returns server location.</i>	

OTS InfoRespond::getIP()	138
<i>Returns server IP.</i>	
OTS Guilds List::set_state()	135
<i>Magic PHP5 method.</i>	
OTS Guilds List::valid()	135
<i>Checks if there are any rows left.</i>	
OTS Guilds List::setOffset()	135
<i>Sets OFFSET.</i>	
OTS Guilds List::setLimit()	134
<i>Sets LIMIT.</i>	
OTS Guilds List::sleep()	136
<i>Magic PHP5 method.</i>	
OTS Guilds List::wakeup()	136
<i>Magic PHP5 method.</i>	
OTS InfoRespond::getEmail()	137
<i>Returns owner e-mail.</i>	
OTS InfoRespond::getClientVersion()	137
<i>Returns dedicated version of client.</i>	
OTS InfoRespond	137
<i>Wrapper for 'info' respond's DOMDocument.</i>	
OTS Player::getManaMax()	159
<i>Maximum mana.</i>	
OTS Player::getManaSpent()	159
<i>Mana spent.</i>	
OTS Player::setSave()	183
<i>Sets save flag.</i>	
OTS Player::setRedSkullTime()	183
<i>Sets red skulled time remained.</i>	
OTS Player::setRedSkull()	182
<i>Sets red skull flag.</i>	
OTS Player::setRankId()	182
<i>Sets guild rank ID.</i>	
OTS Player::setSex()	183
<i>Sets player gender.</i>	
OTS Player::setSkill()	184
<i>Sets skill value.</i>	
OTS Player::setSoul()	185
<i>Sets soul points.</i>	
OTS Player::setSlot()	185
<i>Sets slot content.</i>	
OTS Player::setSkillTries()	184
<i>Sets skill's tries for next level.</i>	
OTS Player::setRank()	181
<i>Assigns guild rank.</i>	
OTS Player::setPremiumEnd()	181
<i>Sets player's Premium Account expiration timestamp.</i>	
OTS Player::setManaMax()	178
<i>Sets maximum mana.</i>	
OTS Player::setMana()	178
<i>Sets current mana.</i>	
OTS Player::setMagLevel()	178
<i>Sets magic level.</i>	
OTS Player::setLossSkills()	177

<i>Sets percentage of skills lost after dead.</i>	
OTS_Player::setManaSpent()	179
<i>Sets mana spent.</i>	
OTS_Player::setName()	179
<i>Sets players's name.</i>	
OTS_Player::setPosZ()	181
<i>Sets Z map coordinate.</i>	
OTS_Player::setPosY()	180
<i>Sets Y map coordinate.</i>	
OTS_Player::setPosX()	180
<i>Sets X map coordinate.</i>	
OTS_Player::setTownId()	186
<i>Sets residence town's ID.</i>	
OTS_Player::setVocation()	186
<i>Sets player proffesion.</i>	
OTS_Players_List::setLimit()	192
<i>Sets LIMIT.</i>	
OTS_Players_List::rewind()	192
<i>Select players from database.</i>	
OTS_Players_List::next()	191
<i>Moves to next row.</i>	
OTS_Players_List::key()	191
<i>Current cursor position.</i>	
OTS_Players_List::setOffset()	192
<i>Sets OFFSET.</i>	
OTS_Players_List::valid()	193
<i>Checks if there are any rows left.</i>	
OTS_Players_List::__wakeup()	194
<i>Magic PHP5 method.</i>	
OTS_Players_List::__sleep()	194
<i>Magic PHP5 method.</i>	
OTS_Players_List::__set_state()	193
<i>Magic PHP5 method.</i>	
OTS_Players_List::deletePlayer()	191
<i>Deletes player.</i>	
OTS_Players_List::current()	190
<i>Returns current row.</i>	
OTS_Player::__clone()	188
<i>Creates clone of object.</i>	
OTS_Player::unsetSave()	187
<i>Unsets save flag.</i>	
OTS_Player::unsetRedSkull()	187
<i>Unsets red skull flag.</i>	
OTS_Player::unban()	187
<i>Deletes ban from current player.</i>	
OTS_Player::__set_state()	188
<i>Magic PHP5 method.</i>	
OTS_Player::__sleep()	188
<i>Magic PHP5 method.</i>	
OTS_Players_List::count()	190
<i>Returns number of characters on list in current criterium.</i>	
OTS_Players_List	189
<i>List of players.</i>	

OTS_Player::wakeup()	189
<i>Magic PHP5 method.</i>	
OTS_Player::setLossMana()	177
<i>Sets percentage of used mana lost after dead.</i>	
OTS_Player::setLossExperience()	176
<i>Sets percentage of experience lost after dead.</i>	
OTS_Player::getVocation()	165
<i>Player profession.</i>	
OTS_Player::getTownId()	165
<i>Residence town's ID.</i>	
OTS_Player::getSoul()	164
<i>Soul points.</i>	
OTS_Player::getSlot()	164
<i>Returns items tree from given slot.</i>	
OTS_Player::hasRedSkull()	165
<i>Checks if player has red skull.</i>	
OTS_Player::isBanned()	166
<i>Checks if player is banned.</i>	
OTS_Player::load()	167
<i>Loads player with given id.</i>	
OTS_Player::isSaveSet()	166
<i>Checks if save flag is set.</i>	
OTS_Player::isLoading()	166
<i>Checks if object is loaded.</i>	
OTS_Player::getSkillTries()	163
<i>Returns player's skill's tries for next level.</i>	
OTS_Player::getSkill()	163
<i>Returns player's skill.</i>	
OTS_Player::getPosZ()	160
<i>Z map coordinate.</i>	
OTS_Player::getPosY()	160
<i>Y map coordinate.</i>	
OTS_Player::getPosX()	160
<i>X map coordinate.</i>	
OTS_Player::getName()	159
<i>Player name.</i>	
OTS_Player::getPremiumEnd()	161
<i>Player's Premium Account expiration timestamp.</i>	
OTS_Player::getRank()	161
<i>Assigned guild rank.</i>	
OTS_Player::getSex()	162
<i>Player gender.</i>	
OTS_Player::getRedSkullTime()	162
<i>Red skulled time remained.</i>	
OTS_Player::getRankId()	161
<i>Guild rank ID.</i>	
OTS_Player::save()	167
<i>Saves player in database.</i>	
OTS_Player::setAccount()	167
<i>Assigns character to account.</i>	
OTS_Player::setLookAddons()	174
<i>Sets addons.</i>	
OTS_Player::setLevel()	173

<i>Sets experience level.</i>	
OTS_Player::setLastLogin()	173
<i>Sets last login timestamp.</i>	
OTS_Player::setLastIP()	173
<i>Sets last login IP.</i>	
OTS_Player::setLookBody()	174
<i>Sets body color.</i>	
OTS_Player::setLookFeet()	175
<i>Sets boots color.</i>	
OTS_Player::setLookType()	176
<i>Sets outfit.</i>	
OTS_Player::setLookLegs()	176
<i>Sets legs color.</i>	
OTS_Player::setLookHead()	175
<i>Sets hair color.</i>	
OTS_Player::setHealthMax()	172
<i>Sets maximum HP.</i>	
OTS_Player::setHealth()	172
<i>Sets current HP.</i>	
OTS_Player::setCustomField()	169
<i>Writes custom field.</i>	
OTS_Player::setConditions()	168
<i>Sets conditions.</i>	
OTS_Player::setCap()	168
<i>Sets capacity.</i>	
OTS_Player::setDepot()	169
<i>Sets depot content.</i>	
OTS_Player::setDirection()	170
<i>Sets looking direction.</i>	
OTS_Player::setGuildNick()	171
<i>Sets guild nick.</i>	
OTS_Player::setGroup()	171
<i>Assigns character to group.</i>	
OTS_Player::setExperience()	171
<i>Sets experience points.</i>	
OTS_GuildRank:: set_state()	125
<i>Magic PHP5 method.</i>	
OTS_GuildRank:: clone()	124
<i>Creates clone of object.</i>	
OTS_Accounts_List:: set_state()	71
<i>Magic PHP5 method.</i>	
OTS_Accounts_List::valid()	71
<i>Checks if there are any rows left.</i>	
OTS_Accounts_List::setOffset()	71
<i>Sets OFFSET.</i>	
OTS_Accounts_List::setLimit()	70
<i>Sets LIMIT.</i>	
OTS_Accounts_List:: sleep()	72
<i>Magic PHP5 method.</i>	
OTS_Accounts_List:: wakeup()	72
<i>Magic PHP5 method.</i>	
OTS_Container::count()	74
<i>Number of items inside container.</i>	

OTS Container::addItem()	73
<i>Adds item to container.</i>	
OTS Container	73
<i>Container item representation.</i>	
OTS Accounts List::rewind()	70
<i>Select accounts from database.</i>	
OTS Accounts List::next()	70
<i>Moves to next row.</i>	
OTS Account:: wakeup()	67
<i>Magic PHP5 method.</i>	
OTS Account:: sleep()	67
<i>Magic PHP5 method.</i>	
OTS Account:: set_state()	66
<i>Magic PHP5 method.</i>	
OTS Account:: clone()	66
<i>Creates clone of object.</i>	
OTS Accounts List	68
<i>List of accounts.</i>	
OTS Accounts List::count()	68
<i>Returns number of accounts on list in current criterium.</i>	
OTS Accounts List::key()	69
<i>Current cursor position.</i>	
OTS Accounts List::deleteAccount()	69
<i>Deletes account.</i>	
OTS Accounts List::current()	69
<i>Returns current row.</i>	
OTS Container::current()	74
<i>Returns current item.</i>	
OTS Container::key()	74
<i>Current cursor position.</i>	
OTS DB ODBC::SQLquote()	82
<i>IOTS_DB method.</i>	
OTS DB ODBC::SQLquery()	81
<i>IOTS_DB method.</i>	
OTS DB ODBC::limit()	81
<i>LIMIT/OFFSET clause for queries.</i>	
OTS DB ODBC::fieldName()	80
<i>Query-quoted field name.</i>	
OTS DB ODBC::tableName()	82
<i>Query-quoted table name.</i>	
OTS DB PostgreSQL	83
<i>PostgreSQL connection interface.</i>	
OTS DB PostgreSQL::SQLquery()	85
<i>IOTS_DB method.</i>	
OTS DB PostgreSQL::limit()	84
<i>LIMIT/OFFSET clause for queries.</i>	
OTS DB PostgreSQL::fieldName()	84
<i>Query-quoted field name.</i>	
OTS DB ODBC	79
<i>ODBC connection interface.</i>	
OTS DB MySQL::tableName()	79
<i>Query-quoted table name.</i>	
OTS Container::valid()	76

<i>Checks if there are any items left.</i>	
OTS_Container::rewind()	75
<i>Resets internal items array pointer.</i>	
OTS_Container::removeItem()	75
<i>Removes given item from current container.</i>	
OTS_Container::next()	75
<i>Moves to next item.</i>	
OTS_DB_MySQL	76
<i>MySQL connection interface.</i>	
OTS_DB_MySQL::fieldName()	77
<i>Query-quoted field name.</i>	
OTS_DB_MySQL::SQLquote()	78
<i>IOTS_DB method.</i>	
OTS_DB_MySQL::SQLquery()	78
<i>IOTS_DB method.</i>	
OTS_DB_MySQL::limit()	78
<i>LIMIT/OFFSET clause for queries.</i>	
OTS_Account::unblock()	66
<i>Unblocks account.</i>	
OTS_Account::unban()	65
<i>Deletes ban from current account.</i>	
OTS_Item.php	42
OTS_InfoRespond.php	41
OTS_Guilds_List.php	40
OTS_GuildRanks_List.php	39
OTS_Player.php	43
OTS_Players_List.php	44
OTS_Account::ban()	54
<i>Bans current account.</i>	
OTS_Account	54
<i>OTServ account abstraction.</i>	
OTS_SQLite_Results.php	45
OTS_GuildRank.php	38
OTS_Guild.php	37
OTS_DB_MySQL.php	31
OTS_Container.php	30
OTS_Accounts_List.php	29
OTS_Account.php	28
OTS_DB_ODBC.php	32
OTS_DB_PostgreSQL.php	33
OTS_Groups_List.php	36
OTS_Group.php	35
OTS_DB_SQLite.php	34
OTS_Account::block()	55
<i>Blocks account.</i>	
OTS_Account::create()	55
<i>Creates new account.</i>	
OTS_Account::save()	62
<i>Updates account in database.</i>	
OTS_Account::load()	62
<i>Loads account with given number.</i>	
OTS_Account::isLoading()	62
<i>Checks if object is loaded.</i>	

OTS Account::isBlocked()	61
<i>Checks if account is blocked.</i>	
OTS Account::setCustomField()	63
<i>Writes custom field.</i>	
OTS Account::setEMail()	63
<i>Sets account's email.</i>	
OTS Account::setPassword()	65
<i>Sets account's password.</i>	
OTS Account::setPACCDays()	64
<i>Sets PACC days count.</i>	
OTS Account::setGroup()	64
<i>Assigns account to group.</i>	
OTS Account::isBanned()	61
<i>Checks if account is banned.</i>	
OTS Account::getPlayers()	61
<i>List of characters on account.</i>	
OTS Account::getCustomField()	58
<i>Reads custom field.</i>	
OTS Account::find()	58
<i>Loads account by it's e-mail address.</i>	
OTS Account::createEx()	56
<i>Creates new account.</i>	
OTS Account::getEMail()	59
<i>E-mail address.</i>	
OTS Account::getGroup()	59
<i>Returns group of this account.</i>	
OTS Account::getPassword()	60
<i>Account's password.</i>	
OTS Account::getPACCDays()	60
<i>PACC days.</i>	
OTS Account::getId()	59
<i>Account number.</i>	
OTS DB PostgreSQL::SQLquote()	85
<i>IOTS_DB method.</i>	
OTS DB PostgreSQL::tableName()	86
<i>Query-quoted table name.</i>	
OTS Guild::load()	111
<i>Loads guild with given id.</i>	
OTS Guild::listRequests()	111
<i>Returns list of players that requested membership.</i>	
OTS Guild::listInvites()	111
<i>Returns list of invited players.</i>	
OTS Guild::isLoaded()	110
<i>Checks if object is loaded.</i>	
OTS Guild::request()	112
<i>Requests membership in guild for player player.</i>	
OTS Guild::save()	112
<i>Saves guild in database.</i>	
OTS Guild::setInvitesDriver()	114
<i>Assigns invites handler.</i>	
OTS Guild::setCustomField()	113
<i>Writes custom field.</i>	
OTS Guild::setCreationData()	113

<i>Sets guild creation data.</i>	
OTS_Guild::invite()	110
<i>Invites player to guild.</i>	
OTS_Guild::getOwner()	110
<i>Returns owning player of this player.</i>	
OTS_Guild::find()	107
<i>Loads guild by it's name.</i>	
OTS_Guild::deleteRequest()	107
<i>Deletes request from player.</i>	
OTS_Guild::deleteInvite()	106
<i>Deletes invitation for player to guild.</i>	
OTS_Guild::acceptRequest()	106
<i>Accepts player.</i>	
OTS_Guild::getCreationData()	108
<i>Guild creation data.</i>	
OTS_Guild::getCustomField()	108
<i>Reads custom field.</i>	
OTS_Guild::getName()	109
<i>Guild name.</i>	
OTS_Guild::getId()	109
<i>Guild ID.</i>	
OTS_Guild::getGuildRanks()	108
<i>Reads all ranks that are in this guild.</i>	
OTS_Guild::setName()	114
<i>Sets players's name.</i>	
OTS_Guild::setOwner()	115
<i>Assigns guild to owner.</i>	
OTS_GuildRank::load()	121
<i>Loads rank with given id.</i>	
OTS_GuildRank::isLoading()	121
<i>Checks if object is loaded.</i>	
OTS_GuildRank::getPlayers()	121
<i>Reads all players who has this rank set.</i>	
OTS_GuildRank::getName()	120
<i>Rank name.</i>	
OTS_GuildRank::save()	122
<i>Saves rank in database.</i>	
OTS_GuildRank::setCustomField()	122
<i>Writes custom field.</i>	
OTS_GuildRank::setName()	124
<i>Sets rank's name.</i>	
OTS_GuildRank::setLevel()	123
<i>Sets rank's access level within guild.</i>	
OTS_GuildRank::setGuild()	123
<i>Assigns rank to guild.</i>	
OTS_GuildRank::getLevel()	120
<i>Rank's access level.</i>	
OTS_GuildRank::getId()	120
<i>Rank ID.</i>	
OTS_Guild::__sleep()	117
<i>Magic PHP5 method.</i>	
OTS_Guild::__set_state()	116
<i>Magic PHP5 method.</i>	

OTS_Guild::clone()	116
<i>Creates clone of object.</i>	
OTS_Guild::setRequestsDriver()	115
<i>Assigns requests handler.</i>	
OTS_Guild::wakeup()	117
<i>Magic PHP5 method.</i>	
OTS_GuildRank	117
<i>OTServ guild rank abstraction.</i>	
OTS_GuildRank::getGuild()	119
<i>Returns guild of this rank.</i>	
OTS_GuildRank::getCustomField()	119
<i>Reads custom field.</i>	
OTS_GuildRank::find()	118
<i>Loads rank by it's name.</i>	
OTS_Guild::acceptInvite()	105
<i>Finalise invitation.</i>	
OTS_Guild	104
<i>OTServ guild abstraction.</i>	
OTS_Group::getPlayers()	93
<i>List of characters in given group.</i>	
OTS_Group::getName()	93
<i>Group name.</i>	
OTS_Group::getMaxVIPList()	92
<i>Maximum count of players in VIP list.</i>	
OTS_Group::getMaxDepotItems()	92
<i>Maximum count of items in depot.</i>	
OTS_Group::isLoading()	93
<i>Checks if object is loaded.</i>	
OTS_Group::load()	94
<i>Loads group with given id.</i>	
OTS_Group::setCustomField()	95
<i>Writes custom field.</i>	
OTS_Group::setAccess()	94
<i>Sets access level.</i>	
OTS_Group::save()	94
<i>Saves account in database.</i>	
OTS_Group::getId()	92
<i>Group ID.</i>	
OTS_Group::getFlags()	91
<i>Rights flags.</i>	
OTS_DB_SQLite::SQLquery()	88
<i>IOTS_DB method.</i>	
OTS_DB_SQLite::limit()	88
<i>LIMIT/OFFSET clause for queries.</i>	
OTS_DB_SQLite::fieldName()	87
<i>Query-quoted field name.</i>	
OTS_DB_SQLite	86
<i>SQLite connection interface.</i>	
OTS_DB_SQLite::SQLquote()	89
<i>IOTS_DB method.</i>	
OTS_DB_SQLite::tableName()	89
<i>Query-quoted table name.</i>	
OTS_Group::getCustomField()	91

<i>Reads custom field.</i>	
OTS_Group::getAccess()	90
<i>Access level.</i>	
OTS_Group	90
<i>OTServ user group abstraction.</i>	
OTS_Group::setFlags()	96
<i>Sets rights flags.</i>	
OTS_Group::setMaxDepotItems()	96
<i>Sets maximum count of items in depot.</i>	
OTS_Groups_List::setLimit()	102
<i>Sets LIMIT.</i>	
OTS_Groups_List::rewind()	101
<i>Select groups from database.</i>	
OTS_Groups_List::next()	101
<i>Moves to next row.</i>	
OTS_Groups_List::key()	101
<i>Current cursor position.</i>	
OTS_Groups_List::setOffset()	102
<i>Sets OFFSET.</i>	
OTS_Groups_List::valid()	103
<i>Checks if there are any rows left.</i>	
OTS_Groups_List::wakeup()	104
<i>Magic PHP5 method.</i>	
OTS_Groups_List::sleep()	103
<i>Magic PHP5 method.</i>	
OTS_Groups_List::set_state()	103
<i>Magic PHP5 method.</i>	
OTS_Groups_List::deleteGroup()	100
<i>Deletes group.</i>	
OTS_Groups_List::current()	100
<i>Returns current row.</i>	
OTS_Group::clone()	97
<i>Creates clone of object.</i>	
OTS_Group::setName()	97
<i>Sets group's name.</i>	
OTS_Group::setMaxVIPList()	96
<i>Sets maximum count of players in VIP list.</i>	
OTS_Group::set_state()	98
<i>Magic PHP5 method.</i>	
OTS_Group::sleep()	98
<i>Magic PHP5 method.</i>	
OTS_Groups_List::count()	100
<i>Returns number of groups on list in current criterium.</i>	
OTS_Groups_List	99
<i>List of groups.</i>	
OTS_Group::wakeup()	99
<i>Magic PHP5 method.</i>	
OTS.php	27
<i>This file contains main toolkit class.</i>	

P

POT::VOCATION_KNIGHT	206
<i>Knight.</i>	
POT::VOCATION_DRUID	206
<i>Druid.</i>	
POT::VOCATION_NONE	207
<i>None vocation.</i>	
POT::VOCATION_PALADIN	207
<i>Paladin.</i>	
POT::VOCATION_SORCERER	207
<i>Sorcerer.</i>	
POT::SLOT_RING	206
<i>Ring slot.</i>	
POT::SLOT_RIGHT	205
<i>Right hand slot.</i>	
POT::SLOT_HEAD	204
<i>Head slot.</i>	
POT::SLOT_FEET	203
<i>Boots slot.</i>	
POT::SLOT_LEFT	204
<i>Left hand slot.</i>	
POT::SLOT_LEGS	204
<i>Legs slot.</i>	
POT::SLOT_NECKLACE	205
<i>Necklace slot.</i>	
POT::banIP()	208
<i>Bans given IP number.</i>	
POT::connect()	208
<i>Connects to database.</i>	
POT::loadVocations()	213
<i>Loads vocations list.</i>	
POT::loadClass()	213
<i>Loads POT class file.</i>	
POT::serverStatus()	214
<i>Queries server status.</i>	
POT::setPOTPath()	215
<i>Set POT directory.</i>	
POT::unbanIP()	215
<i>Deletes ban from given IP number.</i>	
POT::isIPBanned()	212
<i>Checks if given IP is banned.</i>	
POT::getVocationsList()	212
<i>Returns list (id => name) of loaded vocations.</i>	
POT::getDBHandle()	210
<i>Returns database connection handle.</i>	
POT::createObject()	209
<i>Creates OTServ DAO class instance.</i>	
POT::getInstance()	210
<i>Singleton.</i>	
POT::getVocationID()	211
<i>Returns vocation's ID.</i>	
POT::getVocationName()	211
<i>Returns name of given vocation's ID.</i>	
POT::SLOT_BACKPACK	203

<i>Backpack slot.</i>	203
POT::SLOT_ARMOR	203
<i>Armor slot.</i>	
POT::DB_PGSQL	197
<i>PostgreSQL driver.</i>	
POT::DB_ODBC	196
<i>ODBC driver.</i>	
POT::DB_SQLITE	197
<i>SQLite driver.</i>	
POT::DEPOT_SID_FIRST	197
<i>First depot item sid.</i>	
POT::DIRECTION_EAST	198
<i>East.</i>	
POT::DB_MYSQL	196
<i>MySQL driver.</i>	
POT::BAN_PLAYER	196
<i>Player ban.</i>	
POT class preview	5
PHP 5.0	3
POT	195
<i>Main POT class.</i>	
POT::BAN_ACCOUNT	195
<i>Account ban.</i>	
POT::BAN_IP	195
<i>IP ban.</i>	
POT::DIRECTION_NORTH	198
<i>North.</i>	
POT::DIRECTION_SOUTH	198
<i>South.</i>	
POT::SKILL_FIST	201
<i>Fist fighting.</i>	
POT::SKILL_FISHING	201
<i>Fishing.</i>	
POT::SKILL_SHIELDING	201
<i>Shielding.</i>	
POT::SKILL_SWORD	202
<i>Sword fighting.</i>	
POT::SLOT_AMMO	202
<i>Ammunition slot.</i>	
POT::SKILL_DISTANCE	200
<i>Distance fighting.</i>	
POT::SKILL_CLUB	200
<i>Club fighting.</i>	
POT::DIRECTION_WEST	199
<i>West.</i>	
POT::SEX_FEMALE	199
<i>Female gender.</i>	
POT::SEX_MALE	199
<i>Male gender.</i>	
POT::SKILL_AXE	200
<i>Axe fighting.</i>	
POT	1

Q

Quick start	6
---------------------------------------	---

R

README	224
----------------------------------	-----

S

Server online status	19
------------------------------------------------	----