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 block . . . 54
Method create . . . 55
example: account.php . . . 55
Method createEx . . . 56
example: account.php . . . 56
Method find . . . 57
Method getCustomField . . . 58
Method getEMail . . . 58
Method getGroup . . . 59
Method getId . . . 59
Method getPACCDays . . . 59
Method getPassword . . . 60
Method getPlayers . . . 60
Method isBlocked . . . 60
Method isLoaded . . . 61
Method load . . . 61
Method save . . . 61
Method setCustomField . . . 62
Method setEMail . . . 63
Method setGroup . . . 63
Method setPACCDays . . . 63
Method setPassword . . . 64
Method unblock . . . 64
Method __clone . . . 65
Method __set_state . . . 65
Method __sleep . . . 66
Method __wakeup . . . 66
Class OTS_Accounts_List . . . 66
Constructor __construct . . . 67
Method count . . . 67
Method current . . . 67
Method deleteAccount . . . 68
Method key . . . 68
Method next . . . 69
Method rewind . . . 69

Method setLimit . . . 69
Method setOffset . . . 70
Method valid . . . 70
Method __set_state . . . 70
Method __sleep . . . 71
Method __wakeup . . . 71
Class OTS_Container . . . 72
Method addItem . . . 72
Method count . . . 72
Method current . . . 73
Method key . . . 73
Method next . . . 74
Method removeItem . . . 74
Method rewind . . . 74
Method valid . . . 75
Class OTS_DB_MySQL . . . 75
Constructor __construct . . . 75
Method fieldName . . . 76
Method limit . . . 76
Method SQLquery . . . 77
Method SQLquote . . . 77
Method tableName . . . 78
Class OTS_DB_ODBC . . . 78
Constructor __construct . . . 79
Method fieldName . . . 79
Method limit . . . 80
Method SQLquery . . . 80
Method SQLquote . . . 81
Method tableName . . . 81
Class OTS_DB_PostgreSQL . . . 82
Constructor __construct . . . 82
Method fieldName . . . 83
Method limit . . . 83
Method SQLquery . . . 84
Method SQLquote . . . 84
Method tableName . . . 85
Class OTS_DB_SQLite . . . 85
Constructor __construct . . . 86
Method fieldName . . . 86
Method limit . . . 87
Method SQLquery . . . 87
Method SQLquote . . . 87
Method tableName . . . 88
Class OTS_Group . . . 88
Constructor __construct . . . 89
Method getAccess . . . 89
Method getCustomField . . . 90
Method getFlags . . . 90
Method getId . . . 90

Method getMaxDepotItems . . . 91
Method getMaxVIPList . . . 91
Method getName . . . 92
Method getPlayers . . . 92
Method isLoaded . . . 92
Method load . . . 93
Method save . . . 93
Method setAccess . . . 93
Method setCustomField . . . 94
Method setFlags . . . 94
Method setMaxDepotItems . . . 95
Method setMaxVIPList . . . 95
Method setName . . . 96
Method __clone . . . 96
Method __set_state . . . 96
Method __sleep . . . 97
Method __wakeup . . . 97
Class OTS_Groups_List . . . 98
Constructor __construct . . . 98
Method count . . . 99
Method current . . . 99
Method deleteGroup . . . 99
Method key . . . 100
Method next . . . 100
Method rewind . . . 100
Method setLimit . . . 101
Method setOffset . . . 101
Method valid . . . 101
Method __set_state . . . 102
Method __sleep . . . 102
Method __wakeup . . . 103
Class OTS_Guild . . . 103
Constructor __construct . . . 103
Method acceptInvite . . . 104
Method acceptRequest . . . 104
Method deleteInvite . . . 105
Method deleteRequest . . . 105
Method find . . . 106
Method getCreationData . . . 106
Method getCustomField . . . 107
Method getGuildRanks . . . 107
Method getId . . . 108
Method getName . . . 108
Method getOwner . . . 108
Method invite . . . 109
Method isLoaded . . . 109
Method listInvites . . . 110
Method listRequests . . . 110
Method load . . . 110

Method request . . . 111
Method save . . . 111
Method setCreationData . . . 112
Method setCustomField . . . 112
Method setInvitesDriver . . . 113
Method setName . . . 113
Method setOwner . . . 114
Method setRequestsDriver . . . 114
Method __clone . . . 115
Method __set_state . . . 115
Method __sleep . . . 116
Method __wakeup . . . 116
Class OTS_GuildRank . . . 116
Constructor __construct . . . 117
Method find . . . 117
Method getCustomField . . . 118
Method getGuild . . . 118
Method getId . . . 119
Method getLevel . . . 119
Method getName . . . 119
Method getPlayers . . . 120
Method isLoaded . . . 120
Method load . . . 120
Method save . . . 121
Method setCustomField . . . 121
Method setGuild . . . 122
Method setLevel . . . 122
Method setName . . . 123
Method __clone . . . 123
Method __set_state . . . 123
Method __sleep . . . 124
Method __wakeup . . . 124
Class OTS_GuildRanks_List . . . 125
Constructor __construct . . . 125
Method count . . . 126
Method current . . . 126
Method deleteGuildRank . . . 126
Method key . . . 127
Method next . . . 127
Method rewind . . . 127
Method setLimit . . . 128
Method setOffset . . . 128
Method valid . . . 129
Method __set_state . . . 129
Method __sleep . . . 129
Method __wakeup . . . 130
Class OTS_Guilds_List . . . 130
Constructor __construct . . . 131
Method count . . . 131

Method current 131
Method deleteGuild 132
Method key 132
Method next 132
Method rewind 133
Method setLimit 133
Method setOffset 133
Method valid 134
Method __set_state 134
Method __sleep 135
Method __wakeup 135
Class OTS_InfoRespond 136
Method getClientVersion 136
Method getEMail 136
Method getIP 137
Method getLocation 137
Method getMapAuthor 137
Method getMapHeight 138
Method getMapName 138
Method getMapWidth 138
Method getMaxPlayers 139
Method getMonstersCount 139
Method getMOTD 139
Method getName 139
Method getOnlinePlayers 140
Method getOwner 140
Method getPlayersPeak 140
Method getPort 141
Method getServer 141
Method getServerVersion 141
Method getTSPQVersion 142
Method getUptime 142
Method getURL 142
Class OTS_Item 143
Constructor __construct 143
Method count 144
Method getAttributes 144
Method getCount 144
Method getId 145
Method setAttributes 145
Method setCount 145
Class OTS_Player 146
Constructor __construct 146
Method find 147
Method getAccount 147
Method getCap 148
Method getConditions 148
Method getCustomField 148
Method getDepot 149

Method getDirection . . . 149
Method getExperience . . . 150
Method getGroup . . . 150
Method getGuildNick . . . 151
Method getHealth . . . 151
Method getHealthMax . . . 151
Method getId . . . 152
Method getLastIP . . . 152
Method getLastLogin . . . 152
Method getLevel . . . 153
Method getLookAddons . . . 153
Method getLookBody . . . 153
Method getLookFeet . . . 154
Method getLookHead . . . 154
Method getLookLegs . . . 154
Method getLookType . . . 155
Method getLossExperience . . . 155
Method getLossMana . . . 156
Method getLossSkills . . . 156
Method getMagLevel . . . 156
Method getMana . . . 157
Method getManaMax . . . 157
Method getManaSpent . . . 157
Method getName . . . 158
Method getPosX . . . 158
Method getPosY . . . 158
Method getPosZ . . . 159
Method getPremiumEnd . . . 159
Method getRank . . . 160
Method getRankId . . . 160
Method getRedSkullTime . . . 160
Method getSex . . . 161
Method getSkill . . . 161
Method getSkillTries . . . 161
Method getSlot . . . 162
Method getSoul . . . 163
Method getTownId . . . 163
Method getVocation . . . 163
Method hasRedSkull . . . 164
Method isLoaded . . . 164
Method isSaveSet . . . 164
Method load . . . 165
Method save . . . 165
Method setAccount . . . 165
Method setCap . . . 166
Method setConditions . . . 166
Method setCustomField . . . 167
Method setDepot . . . 167
Method setDirection . . . 168

Method setExperience . . . 169
Method setGroup . . . 169
Method setGuildNick . . . 169
Method setHealth . . . 170
Method setHealthMax . . . 170
Method setLastIP . . . 171
Method setLastLogin . . . 171
Method setLevel . . . 171
Method setLookAddons . . . 172
Method setLookBody . . . 172
Method setLookFeet . . . 173
Method setLookHead . . . 173
Method setLookLegs . . . 174
Method setLookType . . . 174
Method setLossExperience . . . 174
Method setLossMana . . . 175
Method setLossSkills . . . 175
Method setMagLevel . . . 176
Method setMana . . . 176
Method setManaMax . . . 176
Method setManaSpent . . . 177
Method setName . . . 177
Method setPosX . . . 178
Method setPosY . . . 178
Method setPosZ . . . 179
Method setPremiumEnd . . . 179
Method setRank . . . 179
Method setRankId . . . 180
Method setRedSkull . . . 180
Method setRedSkullTime . . . 181
Method setSave . . . 181
Method setSex . . . 181
Method setSkill . . . 182
Method setSkillTries . . . 182
Method setSlot . . . 183
Method setSoul . . . 183
Method setTownId . . . 184
Method setVocation . . . 184
Method unsetRedSkull . . . 185
Method unsetSave . . . 185
Method __clone . . . 185
Method __set_state . . . 186
Method __sleep . . . 186
Method __wakeup . . . 187
Class OTS_Players_List . . . 187
Constructor __construct . . . 187
Method count . . . 188
Method current . . . 188
Method deletePlayer . . . 188

Method key . . . 189
Method next . . . 189
Method rewind . . . 189
Method setLimit . . . 190
Method setOffset . . . 190
Method valid . . . 190
Method __set_state . . . 191
Method __sleep . . . 191
Method __wakeup . . . 192
Class POT . . . 192
Class Constant DB_MYSQL . . . 193
Class Constant DB_ODBC . . . 193
Class Constant DB_PGSQL . . . 193
Class Constant DB_SQLITE . . . 194
Class Constant DEPOT_SID_FIRST . . . 194
Class Constant DIRECTION_EAST . . . 194
Class Constant DIRECTION_NORTH . . . 195
Class Constant DIRECTION_SOUTH . . . 195
Class Constant DIRECTION_WEST . . . 195
Class Constant SEX_FEMALE . . . 196
Class Constant SEX_MALE . . . 196
Class Constant SKILL_AXE . . . 196
Class Constant SKILL_CLUB . . . 196
Class Constant SKILL_DISTANCE . . . 197
Class Constant SKILL_FISHING . . . 197
Class Constant SKILL_FIST . . . 198
Class Constant SKILL_SHIELDING . . . 198
Class Constant SKILL_SWORD . . . 198
Class Constant SLOT_AMMO . . . 199
Class Constant SLOT_ARMOR . . . 199
Class Constant SLOT_BACKPACK . . . 199
Class Constant SLOT_FEET . . . 200
Class Constant SLOT_HEAD . . . 200
Class Constant SLOT_LEFT . . . 201
Class Constant SLOT_LEGS . . . 201
Class Constant SLOT_NECKLACE . . . 201
Class Constant SLOT_RIGHT . . . 202
Class Constant SLOT_RING . . . 202
Class Constant VOCATION_DRUID . . . 202
Class Constant VOCATION_KNIGHT . . . 203
Class Constant VOCATION_NONE . . . 203
Class Constant VOCATION_PALADIN . . . 203
Class Constant VOCATION_SORCERER . . . 204
Method connect . . . 204
example: connect.php . . . 204
Method createObject . . . 205
Method getDBHandle . . . 206
Method getInstance . . . 206
Method loadClass . . . 206

Method serverStatus 207
example: example 207
Method setPOTPath 208
example: fakeroot.php 208
compat.php 210
Appendices 211
Appendix A - Class Trees 212
POT 212
Appendix B - README/CHANGELOG/INSTALL 216
CHANGELOG 217
README 217
INSTALL 219
NEWS 219

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 ralies on database foreign key contraints, triggers etc.

System requirements

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

What POT is

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

What POT is not

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

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 realy 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 enought. If you have your own __autoload() mechanism you won't be able to just inlude example codes - you would need to redefine __autoload() function, which PHP doesnt 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:

```
<a href="http://www.otserv-aac.info/pot/">
    <img alt="This site was smoked" src="http://www.otserv-aac.info/pot.png" />
</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 histroy. It is completly other language then 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:

```php
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// do that before any POT operations!
include( '../compat.php');

// to not repeat all that stuff
include( 'quickstart.php');

// STEP 1: no error here - even thought we loaded class that implements Countable interface which does not exists in PHP 5.0 SPL library, because 'compat' library defines it.
$list= POT::getInstance()->createObject('Players_List');

// STEP 2: we can do that in every version - count() is in fact just a public method
echo $list->count();

// STEP 3: it won't work correctly in PHP 5.0 - PHP won't call internaly count() method of object, will print trivial count() evaluation result on object
echo count( $list);
```

```

?>
```

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 manualy.

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 globaly 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 automaticly. If you have your own __autoload function defined, after including POT class you have to register your function with spl_autoload_register() aswell.

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 automaticly bind POT classes loading mechanism to __autoload() function. Here is a startup code example:

```
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// binds your __autoload code
if( function_exists('__autoload') )
{
    spl_autoload_register('__autoload');
}

// includes POT main file
include( '../classes/OTS.php');

// database configuration - can be simply moved to external file, eg. config.php
$config = array(
    'driver' =>     POT::DB_MYSQL,
    'host' =>     'localhost',
    'user' =>     'wrzasq',
    'database' =>     'otserv'
);

// creates POT instance (or get existing one)
$ots = POT::getInstance();
$ots->     connect(null, $config);

?>
```

Account creation

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

```
<?php

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

```
*/

// to not repeat all that stuff
include( 'quickstart.php');

// creates new OTS_Account object
$account= $ots->      createObject('Account');

// generates new account number
$number= $account->       create();

/*
to generate number from 111111 to 999999 use:
$number = $account->create(111111, 999999);
*/

// sets account info
$account->      setPassword('secret');// $account->setPassword( md5('secret') );
$account->      setEMail('foo@example.com');
$account->      unblock();// remember to unblock!
$account->      setPACCDays(0);
$account->      save();

// give user his number
echo 'Your account number is: ',      $number;

?>
```

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:

```
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// to not repeat all that stuff
include( 'quickstart.php');

// creates new OTS_Player object
$player= $ots->      createObject('Player');

// loads player
$player->      find('Wrzasq');

// checks if player exists
if( $player->      isLoaded() )
{
    // prints character info
```

```php
    echo 'Player \"' . $player->getName() . '\' has ' . $player->getLevel() . ' level.', "\n";

    // example of associated objects retriving
    echo 'Player \"' . $player->getName() . '\' is member of ' . $player->getGroup()->getName() . ' group.', "\n";
}
else
{
    echo 'Player does not exists.', "\n";
}

?>
```

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:

```php
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// to not repeat all that stuff
include('quickstart.php');

// creates new OTS_Player object
$players = $ots->createObject('Players_List');

// count of all players - Countable interface implemented
echo 'There are ' . count($players) . ' players in our database.', "\n";

// sets limitation
$players->setLimit(10);
$players->setOffset(2);

// iterates throught selected players
foreach($players as $index => $player)
{
    // each returned item is instance of OTS_Player class
    echo (2 + $index) . ': ' . $player->getName(), "\n";
}

?>
```

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 langauges 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 enviroment. 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 quide.

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 throught 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:

```php
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// to not repeat all that stuff
include('quickstart.php');

// creates new OTS_Player object
$player = $ots->createObject('Player');

// sets basic fields
$player->setName('Wrzasq');
$player->setSex(POT::SEX_MALE);
$player->setVocation(POT::VOCATION_KNIGHT);
/* etc... */

/*
this is bad! we can't call this now as we dont have object ID assinged yet

$player->setCustomField('my_field', 2);

must save before that to get automatic ID:
*/
$player->save();

// now we can call that:
// 2 won't be quoted - it's integer
$player->setCustomField('my_field', 2);
// 3 will be quoted - '3' is a string!
$player->setCustomField('another_field', '3');

?>
```

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:

```php
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

 // to not repeat all that stuff
 include('quickstart.php');

 // creates new OTS_Player object
 $player = $ots->createObject('Player');
 $player->find('Wrzasq');

 /*
     Items loading example.
 */

 // loading item from ammunition slot
 $item = $player->getSlot(POT::SLOT_AMMO);

 echo $player->getName(), ' has item with id ', $item->getId(), ' in his/her ammo slot.', "\n";

 // checks if item is a container
 if($item instanceof OTS_Container)
 {
     // list backpack content
     foreach($item as $inside)
     {
         echo 'Container contains item with id ', $inside->getId(), '.', "\n";
     }
 }

 /*
     Items tree composing example.
 */

 // creates container - here it would be a depot locker (we pass ID of item to create)
 $container = new OTS_Container(2590);

 // now let's create depot chest
 $chest = new OTS_Container(2594);

 // let's put chest inside locker
 $container->addItem($chest);

 // now let's put something deeper - into the chest
 $item1 = new OTS_Item(3015);
 $chest->addItem($item1);

 // and more...
 $item2 = new OTS_Item(3013);
 $chest->addItem($item2);

 // let's set count for an item
 $item2->setCount(2);

 /*
 Here is a tree of items which we created:

```

```
$container [depot locker]
`-- $chest [depot chest]
    |-- $item1 [first item inserted into chest]
    `-- $item2 [second item inserted into chest] count=2
*/

/*
    Items saving example.
*/

// now we simply put those items into players depot (2 is depot ID)
$player->setDepot(2, $container);

?>
```

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

Guilds

Guilds system basics.

Baiscs

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

Guild management

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

```
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// to not repeat all that stuff
include('quickstart.php');

// loads guild
$guild = $ots->createObject('Guild');
$guild->load(1);

$color = '#FFFFCC';

echo '<h1>Members of ' , htmlspecialchars( $guild->getName() ), '</h1>';

?>
<table>
    <thead>
        <tr>
            <th>Rank</th>
            <th>Members</th>
        </tr>
    </thead>
    <tbody>
<?php

// lists members of all ranks
foreach( $guild->getGuildRanks() as $guildRank)
{
    // display rank in first row
    $first = true;
    // switches rank rows color
    $color= $color==   '#FFFFCC' ? '#FFCCFF' : '#FFFFCC';

    // list members of this rank
    foreach( $guildRank->        getPlayers() as $player)
    {
        echo '<tr style="background-color: '            ,        $color, '">
    <td>'       ,   $first?htmlspecialchars(     $guildRank->   getName() ) : '', '</td>
    <td>'       ,   $player->    getName(), '</td>
</tr>'       ;
        $first=  false;
    }
}

?>
    </tbody>
</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, thought 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:

```
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// to not repeat all that stuff
include( 'quickstart.php');

/*
    POT guilds invites driver.
*/

/**
 * @ignore
 */
class InvitesDriver implements IOTS_GuildAction
{
    // assigned guild
    private $guild;

    // initializes driver
    public function __construct(OTS_Guild $guild)
    {
        $this->guild = $guild;
        // this line automates the process - you can call it manualy from outside, but why?
        $this->guild->setInvitesDriver($this);
    }

    // returns all invited players to current guild
    public function listRequests()
    {
        $invites = array();

        /* here you must create OTS_Player object for each invited player */

        return $invites;
    }

    // invites player to current guild
    public function addRequest(OTS_Player $player)
    {
        /* here you must save invitation for given player */
    }

    // un-invites player
    public function deleteRequest(OTS_Player $player)
    {
        /* here you must delete invitation for given player */
    }

    // commits invitation
    public function submitRequest(OTS_Player $player)
    {
        $rank = null;

        // finds normal member rank
        foreach( $this->guild->getGuildRanks() as $guildRank)
        {
            if( $guildRank->getLevel() == 1)
            {
                $rank = $guildRank;
                break;
            }
        }

        $player->setRank($rank);
        $player->save();

        // clears invitation
        $this->deleteRequest($player);
    }
}

/*
    Parts of this class driver has been taken from OTSCMS (http://otscms.sourceforge.net/) project source
code.
*/

// loads player wiht ID 1
$player = $ots->createObject('Player');
$player->load(1);

```

```php
// loads guild with ID 1
$guild = $ots->createObject('Guild');
$guild->load(1);

// creates invitation logic driver for your implementation for current guild
new InvitesDriver( $guild);

// note that you call guild method!
$guild->invite($player);

?>
```

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 aroud it with simple code:

```
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// to not repeat all that stuff
include('quickstart.php');

// your non-random number
$number = 123456;

// creates new OTS_Account object
$account = $ots->createObject('Account');
$account->load($number);

// number is busy
if( $account->isLoaded() )
{
    echo 'Account number ', $number, ' is used.', "\n";
}
// it is not
else
{
    // generate number from exacly $number - $number range
    $number = $account->create($number, $number);
    echo 'Your account number is: ', $number, '.', "\n";
}

?>
```

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:

```php
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// to not repeat all that stuff
include('quickstart.php');

// server and port
$server = '127.0.0.1';
$port = 7171;

// queries server of status info
$status = $ots->serverStatus($server, $port);

// offline
if(!$status)
{
    echo 'Server ', $server, ' is offline.', "\n";
}
// displays various info
else
{
    echo 'Server name: ', $status->getName(), "\n";
    echo 'Server owner: ', $status->getOwner(), "\n";
    echo 'Players online: ', $status->getOnlinePlayers(), "\n";
    echo 'Maximum allowed number of players: ', $status->getMaxPlayers(), "\n";
    echo 'Required client version: ', $status->getClientVersion(), "\n";
    echo 'All monsters: ', $status->getMonstersCount(), "\n";
    echo 'Server message: ', $status->getMOTD(), "\n";
}

?>
```

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** 0.0.1
- **Version** 0.0.4
- **Copyright** 2007 (C) by Wrzasq
- **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
- **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 insterface 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 insterface 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 obejcts 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 recommeded that your implementations calls assignment functions of $guild to automaticly 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::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

```
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// to not repeat all that stuff
include('quickstart.php');

// creates new OTS_Account object
$account = $ots->   createObject('Account');

// generates new account number
$number = $account->   create();

/*
to generate number from 111111 to 999999 use:
$number = $account->create(111111, 999999);
*/

// sets account info
$account->   setPassword('secret'); // $account->setPassword( md5('secret') );
$account->   setEMail('foo@example.com');
$account->   unblock(); // remember to unblock!
$account->   setPACCDays(0);
$account->   save();

// give user his number
echo 'Your account number is: ', $number;

?>
```

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

```php
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// to not repeat all that stuff
include('quickstart.php');

// creates new OTS_Account object
$account = $ots->   createObject('Account');

// group for account
$group = $ots->   createObject('Group');

// loads group with id 1
$group->   load(1);

// generates new account number
$number = $account->   createEx($group);

// give user his number
echo 'Your account number is: ', $number;

?>
```

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::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::unblock() *[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 implemention. 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 then 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 exacly instance of item which is stored in container, not it's copy.

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

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

Resets internal items array pointer.

Resets internal items array pointer.

- **Version** 0.0.3

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

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

Checks if there are any items left.

Checks if there are any items left.

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

Class OTS_DB_MySQL

[line 19]

MySQL connection interface.

MySQL connection interface.

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

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

Function Parameters:

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

Creates database connection.

Creates database connection.

Connects to MySQL database on given arguments.

List of parameters for this drivers:

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

- **Version** 0.0.1
- **See** POT::connect()
- **Access** public

string function OTS_DB_MySQL::fieldName($name) *[line 101]*

Function Parameters:

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

Query-quoted field name.

Query-quoted field name.

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

string function OTS_DB_MySQL::limit([$limit = false], [$offset = false]) *[line 152]*

Function Parameters:

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

LIMIT/OFFSET clause for queries.
LIMIT/OFFSET clause for queries.

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

PDOStatement|bool function OTS_DB_MySQL::SQLquery($query) *[line 140]*
Function Parameters:

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

IOTS_DB method.
IOTS_DB method.
Overwrites PDO method.

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

string function OTS_DB_MySQL::SQLquote($string) *[line 126]*
Function Parameters:

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

IOTS_DB method.
IOTS_DB method.

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

- **Version** 0.0.1
- **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:

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

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

- **Version** 0.0.4
- **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:

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

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

- **Version** 0.0.4
- **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:

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

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

- **Version** 0.0.1
- **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::isLoaded() *[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 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 rank is not loaded.
- **Since** 0.0.4
- **Access** public

void function OTS_GuildRank::setGuild($guild) *[line 246]*
Function Parameters:

- *OTS_Guild* **$guild** Owning 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 exacly 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

Constructor *void* function OTS_Player::__construct($db) *[line 52]*
Function Parameters:

- *IOTS_DB* **$db** Database connection object.

Sets database connection handler.
Sets database connection handler.

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

void function OTS_Player::find($name) *[line 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_OTS_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_OTS_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_OTS_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_OTS_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_OTS_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_OTS_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::isLoaded() *[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::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 23]

Main POT class.

Main POT class.

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

POT::DB_MYSQL

= 1 *[line 28]*

MySQL driver.
MySQL driver.

- **Version** 0.0.1

POT::DB_ODBC

= 4 *[line 46]*

ODBC driver.
ODBC driver.

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

POT::DB_PGSQL

= 3 *[line 39]*

PostgreSQL driver.
PostgreSQL driver.

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

POT::DB_SQLITE

= 2 *[line 32]*

SQLite driver.

SQLite driver.

- **Version** 0.0.1

POT::DEPOT_SID_FIRST

= 100 *[line 222]*

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 85]*

East.

East.

- **Version** 0.0.1

POT::DIRECTION_NORTH

= 0 *[line 81]*

North.
North.

- **Version** 0.0.1

POT::DIRECTION_SOUTH

= 2 *[line 89]*

South.
South.

- **Version** 0.0.1

POT::DIRECTION_WEST

= 3 *[line 93]*

West.
West.

- **Version** 0.0.1

POT::SEX_FEMALE

= 0 *[line 51]*

Female gender.
Female gender.

- **Version** 0.0.1

POT::SEX_MALE

= 1 *[line 55]*

Male gender.
Male gender.

- **Version** 0.0.1

POT::SKILL_AXE

= 3 *[line 122]*

Axe fighting.
Axe fighting.

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

POT::SKILL_CLUB

= 1 *[line 108]*

Club fighting.
Club fighting.

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

POT::SKILL_DISTANCE

= 4 *[line 129]*

Distance fighting.
Distance fighting.

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

POT::SKILL_FISHING

= 6 *[line 143]*

Fishing.
Fishing.

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

POT::SKILL_FIST

= 0 *[line 101]*

Fist fighting.
Fist fighting.

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

POT::SKILL_SHIELDING

= 5 *[line 136]*

Shielding.
Shielding.

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

POT::SKILL_SWORD

= 2 *[line 115]*

Sword fighting.
Sword fighting.

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

POT::SLOT_AMMO

= 10 *[line 214]*

Ammunition slot.

Ammunition slot.

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

POT::SLOT_ARMOR

= 4 *[line 172]*

Armor slot.

Armor slot.

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

POT::SLOT_BACKPACK

= 3 *[line 165]*

Backpack slot.
Backpack slot.

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

POT::SLOT_FEET

= 8 *[line 200]*

Boots slot.
Boots slot.

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

POT::SLOT_HEAD

= 1 *[line 151]*

Head slot.
Head slot.

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

POT::SLOT_LEFT

= 6 *[line 186]*

Left hand slot.
Left hand slot.

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

POT::SLOT_LEGS

= 7 *[line 193]*

Legs slot.
Legs slot.

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

POT::SLOT_NECKLACE

= 2 *[line 158]*

Necklace slot.
Necklace slot.

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

POT::SLOT_RIGHT

= 5 *[line 179]*

Right hand slot.

Right hand slot.

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

POT::SLOT_RING

= 9 *[line 207]*

Ring slot.

Ring slot.

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

POT::VOCATION_DRUID

= 2 *[line 68]*

Druid.
Druid.

- **Version** 0.0.1

POT::VOCATION_KNIGHT

= 4 *[line 76]*

Knight.
Knight.

- **Version** 0.0.1

POT::VOCATION_NONE

= 0 *[line 60]*

None vocation.
None vocation.

- **Version** 0.0.1

POT::VOCATION_PALADIN

= 3 *[line 72]*

Paladin.
Paladin.

- **Version** 0.0.1

POT::VOCATION_SORCERER

= 1 *[line 64]*

Sorcerer.

Sorcerer.

- **Version** 0.0.1

void function POT::connect($driver, $params) *[line 346]*

connect.php

```
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// includes POT main file
include('../classes/OTS.php');

// you can easily store such structure in config.php
$config = array(
    'driver' =>    POT::DB_MYSQL,
    'prefix' =>    '',
    'host' =>    'localhost',
    'user' =>    'wrzasq',
    'password' =>    '',
    'database' =>    'otserv'
);

// connects to database
$ots = POT::getInstance();
$ots->   connect(null, $config);
// could be: $ots->connect(POT::DB_MYSQL, $config);

?>
```

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, aplies when *$driver* method parameter is *null*
- *prefix* - optional, prefix for database tables, use if you have more then one OTServ installed on one database.

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

IOTS_DAO function POT::createObject($class) *[line 397]*

Function Parameters:

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

Creates OTServ DAO class instance.

Creates OTServ DAO class instance.

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

IOTS_DB function POT::getDBHandle() *[line 472]*

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 229]*

Singleton.

Singleton.

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

void function POT::loadClass($class) *[line 305]*

Function Parameters:

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

Loads POT class file.

Loads POT class file.

Runtime class loading on demand - usefull for __autoload() function.

Note: Since 0.0.2 version this function is suitable for spl_autoload_register().

Note: Since 0.0.3 version this function handles also exceptions.

- **Version** 0.0.3
- **Version** 0.0.1
- **Access** public
- **Example** example not found

OTS_InfoRespond|*bool* function POT::serverStatus($server, $port) *[line 415]*

example

```php
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// to not repeat all that stuff
include('quickstart.php');

// server and port
$server = '127.0.0.1';
$port = 7171;

// queries server of status info
$status = $ots->   serverStatus($server, $port);

// offline
if(!$status)
{
    echo 'Server ', $server, ' is offline.', "\n"          ;
}
// displays various info
else
{
    echo 'Server name: ', $status->   getName(), "\n"          ;
    echo 'Server owner: ', $status->   getOwner(), "\n"          ;
    echo 'Players online: ', $status->   getOnlinePlayers(), "\n"          ;
    echo 'Maximum allowed number of players: ', $status->   getMaxPlayers(), "\n"          ;
    echo 'Required client version: ', $status->   getClientVersion(), "\n"          ;
    echo 'All monsters: ', $status->   getMonstersCount(), "\n"          ;
    echo 'Server message: ', $status->   getMOTD(), "\n"          ;
}

?>
```

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 260]*

fakeroot.php

```php
<?php

/**
 * @ignore
 * @package examples
 * @author Wrzasq <wrzasq@gmail.com>
 * @copyright 2007 (C) by Wrzasq
 * @license http://www.gnu.org/licenses/lgpl-3.0.txt GNU Lesser General Public License, Version 3
 */

// this is the way you should work with POT if you moved main OTS.php file outside POT's directory
include('path/to/OTS.php');

// dont use 'new POT()'!!!
$ots = POT::getInstance();
$ots->   setPOTPath('../classes/');

/*
    here comes your stuff...
*/

```

```
?>
```

Function Parameters:

- *string* **$path** POT files path.

Set POT directory.

Set POT directory.

Use this method if you keep your POT package in different directory then this file.

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

compat.php

POT compatibility assurance package.

POT compatibility assurance package.

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

- **Package** POT
- **Sub-Package** compat
- **Author** Wrzasq < wrzasq@gmail.com>
- **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]
* 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 reserverd in OTServ for futher 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 instanced 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 automaticly 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 langauge to work with OTServ database.

For installation help check INSTALL file.

For usage tutorial/API documentation check http://www.otserv-aac.info/pot/ or documentation.pdf file.

===== Contact =====

In case of any contact needed, please use following e-mail address: wrzasq@gmail.com.

===== Files =====

classes/ - POT class files.
examples/ - example files for learning.
tutorials/ - phpDocumentor directory.
BUGS - known bugs.
CHANGELOG - changes history.
INSTALL - installation tutorial.
LICENSE - POT license (GNU LGPL v3), if you don't accept it - don't use any of those scripts.
NEWS - changes in current release.
README - this readme file.
RULES - rules to be followed during developing contributed code.
TODO - list of things to be done.
Makefile - make input, for documentation generation.
documentation.pdf - phpDocumentor-generater documentation in PDF format.
compat.php - Compatibility assurance library.
test.php - phpUnit test suite.

===== Makefile =====

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

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

Possible targets:

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

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

===== Credits =====

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

INSTALL

POT is a toolkit which means you don't literaly 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 reserverd in OTServ for futher use).

    Even thought 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.

Index

A

Account number hack 18

C

constructor OTS_Guilds_List::__construct() 131
Sets database connection handler.

constructor OTS_GuildRanks_List::__construct() 125
Sets database connection handler.

constructor OTS_GuildRank::__construct() 117
Sets database connection handler.

constructor OTS_Guild::__construct() 103
Sets database connection handler.

constructor OTS_Item::__construct() 143
Creates item of given ID.

constructor OTS_Player::__construct() 146
Sets database connection handler.

CHANGELOG 217

compat.php 210
POT compatibility assurance package.

constructor OTS_Players_List::__construct() 187
Sets database connection handler.

constructor OTS_Groups_List::__construct() 98
Sets database connection handler.

constructor OTS_Group::__construct() 89
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() 67
Sets database connection handler.

constructor OTS_DB_MySQL::__construct() 75
Creates database connection.

constructor OTS_DB_SQLite::__construct() 86
Creates database connection.

constructor OTS_DB_PostgreSQL::__construct() 82
Creates database connection.

constructor OTS_DB_ODBC::__construct() 79
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
Occurs when code attempts to access property of not loaded object.
E_OTS_NoDriver . . . 46
Occurs when code attempts to execute driven action that has no assigned driver to handle it.
E_OTS_NotLoaded.php . . . 23
E_OTS_NoDriver.php . . . 22

G

Guild action drivers . . . 15
Guilds . . . 13

I

IOTS_GuildAction::addRequest() . . . 52
Adds new request.
IOTS_GuildAction . . . 51
Guild action interface.
IOTS_DB::tableName() . . . 50
Query-quoted table name.
IOTS_GuildAction::deleteRequest() . . . 52
Deletes request.
IOTS_GuildAction::listRequests() . . . 53
List of saved pending actions.
INSTALL . . . 219
IOTS_GuildAction::submitRequest() . . . 53
Finalizes request.
IOTS_DB::SQLquote() . . . 50
Query-quoted string value.
IOTS_DB::SQLquery() . . . 50
Evaluates query.
IOTS_DAO . . . 47
OTserv database object.
IOTS_GuildAction.php . . . 26
IOTS_DB.php . . . 25
IOTS_DB . . . 48
OTServ database handler interface.
IOTS_DB::fieldName() . . . 48
Query-quoted field name.
IOTS_DB::limit() . . . 49
LIMIT/OFFSET clause for queries.
IOTS_DB::lastInsertId() . . . 49
ID of last created record.
IOTS_DAO.php . . . 24

N

NEWS 219

O

OTS_Player 146
OTServ character abstraction.

OTS_Item::setCount() 145
Sets count of item.

OTS_Item::setAttributes() 145
Sets item attributes.

OTS_Item::getId() 145
Returns item type.

OTS_Player::find() 147
Loads player by it's name.

OTS_Player::getAccount() 147
Returns account of this player.

OTS_Player::getCustomField() 148
Reads custom field.

OTS_Player::getConditions() 148
Conditions.

OTS_Player::getCap() 148
Capacity.

OTS_Item::getCount() 144
Returns count of item.

OTS_Item::getAttributes() 144
Returns item custom attributes.

OTS_InfoRespond::getServerVersion() 141
Returns server version.

OTS_InfoRespond::getServer() 141
Returns server attribute.

OTS_InfoRespond::getPort() 141
Returns server port.

OTS_InfoRespond::getPlayersPeak() 140
Returns record of online players.

OTS_InfoRespond::getTSPQVersion() 142
Returns version of root element.

OTS_InfoRespond::getUptime() 142
Returns server uptime.

OTS_Item::count() 144
Count value for current item.

OTS_Item 143
Single item representation.

OTS_InfoRespond::getURL() 142
Returns server website.

OTS_Player::getDepot() 149
Returns items tree from given depot.

OTS_Player::getDirection() 149
Looking direction.

OTS_Player::getLookLegs() 154
Legs color.

OTS_Player::getLookHead() 154
Hair color.
OTS_Player::getLookFeet() 154
Boots color.
OTS_Player::getLookBody() 153
Body color.
OTS_Player::getLookType() 155
Outfit.
OTS_Player::getLossExperience() 155
Percentage of experience lost after dead.
OTS_Player::getMagLevel() 156
Magic level.
OTS_Player::getLossSkills() 156
Percentage of skills lost after dead.
OTS_Player::getLossMana() 156
Percentage of used mana lost after dead.
OTS_Player::getLookAddons() 153
Addons.
OTS_Player::getLevel() 153
Experience level.
OTS_Player::getGuildNick() 151
Guild nick.
OTS_Player::getGroup() 150
Returns group of this player.
OTS_Player::getExperience() 150
Experience points.
OTS_Player::getHealth() 151
Current HP.
OTS_Player::getHealthMax() 151
Maximum HP.
OTS_Player::getLastLogin() 152
Last login timestamp.
OTS_Player::getLastIP() 152
Last login IP.
OTS_Player::getId() 152
Player ID.
OTS_InfoRespond::getOwner() 140
Returns owner name.
OTS_InfoRespond::getOnlinePlayers() 140
Returns current amount of players online.
OTS_GuildRanks_List::__wakeup() 130
Magic PHP5 method.
OTS_GuildRanks_List::__sleep() 129
Magic PHP5 method.
OTS_GuildRanks_List::__set_state() 129
Magic PHP5 method.
OTS_GuildRanks_List::valid() 129
Checks if there are any rows left.
OTS_Guilds_List 130
List of guilds.
OTS_Guilds_List::count() 131
Returns number of guilds on list in current criterium.
OTS_Guilds_List::key() 132

Current cursor position.
OTS_Guilds_List::deleteGuild() 132
Deletes guild.
OTS_Guilds_List::current() 131
Returns current row.
OTS_GuildRanks_List::setOffset() 128
Sets OFFSET.
OTS_GuildRanks_List::setLimit() 128
Sets LIMIT.
OTS_GuildRanks_List::count() 126
Returns number of ranks on list in current criterium.
OTS_GuildRanks_List 125
List of guild ranks.
OTS_GuildRank::__wakeup() 124
Magic PHP5 method.
OTS_GuildRank::__sleep() 124
Magic PHP5 method.
OTS_GuildRanks_List::current() 126
Returns current row.
OTS_GuildRanks_List::deleteGuildRank() 126
Deletes guild rank.
OTS_GuildRanks_List::rewind() 127
Select ranks from database.
OTS_GuildRanks_List::next() 127
Moves to next row.
OTS_GuildRanks_List::key() 127
Current cursor position.
OTS_Guilds_List::next() 132
Moves to next row.
OTS_Guilds_List::rewind() 133
Select guilds from database.
OTS_InfoRespond::getMapName() 138
Returns map name.
OTS_InfoRespond::getMapHeight() 138
Returns map height.
OTS_InfoRespond::getMapAuthor() 137
Returns map author.
OTS_InfoRespond::getLocation() 137
Returns server location.
OTS_InfoRespond::getMapWidth() 138
Returns map width.
OTS_InfoRespond::getMaxPlayers() 139
Returns maximum amount of players online.
OTS_InfoRespond::getName() 139
Returns server name.
OTS_InfoRespond::getMOTD() 139
Returns server's Message Of The Day
OTS_InfoRespond::getMonstersCount() 139
Returns number of all monsters on map.
OTS_InfoRespond::getIP() 137
Returns server IP.
OTS_InfoRespond::getEMail() 136
Returns owner e-mail.

OTS_Guilds_List::valid() 134
Checks if there are any rows left.
OTS_Guilds_List::setOffset() 133
Sets OFFSET.
OTS_Guilds_List::setLimit() 133
Sets LIMIT.
OTS_Guilds_List::__set_state() 134
Magic PHP5 method.
OTS_Guilds_List::__sleep() 135
Magic PHP5 method.
OTS_InfoRespond::getClientVersion() 136
Returns dedicated version of client.
OTS_InfoRespond 136
Wrapper for 'info' respond's DOMDocument.
OTS_Guilds_List::__wakeup() 135
Magic PHP5 method.
OTS_Player::getMana() 157
Current mana.
OTS_Player::getManaMax() 157
Maximum mana.
OTS_Player::setSave() 181
Sets save flag.
OTS_Player::setRedSkullTime() 181
Sets red skulled time remained.
OTS_Player::setRedSkull() 180
Sets red skull flag.
OTS_Player::setRankId() 180
Sets guild rank ID.
OTS_Player::setSex() 181
Sets player gender.
OTS_Player::setSkill() 182
Sets skill value.
OTS_Player::setSoul() 183
Sets soul points.
OTS_Player::setSlot() 183
Sets slot content.
OTS_Player::setSkillTries() 182
Sets skill's tries for next level.
OTS_Player::setRank() 179
Assigns guild rank.
OTS_Player::setPremiumEnd() 179
Sets player's Premium Account expiration timestamp.
OTS_Player::setManaMax() 176
Sets maximum mana.
OTS_Player::setMana() 176
Sets current mana.
OTS_Player::setMagLevel() 176
Sets magic level.
OTS_Player::setLossSkills() 175
Sets percentage of skills lost after dead.
OTS_Player::setManaSpent() 177
Sets mana spent.
OTS_Player::setName() 177

Sets players's name.
OTS_Player::setPosZ() . . . 179
Sets Z map coordinate.
OTS_Player::setPosY() . . . 178
Sets Y map coordinate.
OTS_Player::setPosX() . . . 178
Sets X map coordinate.
OTS_Player::setTownId() . . . 184
Sets residence town's ID.
OTS_Player::setVocation() . . . 184
Sets player proffesion.
OTS_Players_List::setLimit() . . . 190
Sets LIMIT.
OTS_Players_List::rewind() . . . 189
Select players from database.
OTS_Players_List::next() . . . 189
Moves to next row.
OTS_Players_List::key() . . . 189
Current cursor position.
OTS_Players_List::setOffset() . . . 190
Sets OFFSET.
OTS_Players_List::valid() . . . 190
Checks if there are any rows left.
OTS_Players_List::__wakeup() . . . 192
Magic PHP5 method.
OTS_Players_List::__sleep() . . . 191
Magic PHP5 method.
OTS_Players_List::__set_state() . . . 191
Magic PHP5 method.
OTS_Players_List::deletePlayer() . . . 188
Deletes player.
OTS_Players_List::current() . . . 188
Returns current row.
OTS_Player::__clone() . . . 185
Creates clone of object.
OTS_Player::unsetSave() . . . 185
Unsets save flag.
OTS_Player::unsetRedSkull() . . . 185
Unsets red skull flag.
OTS_Player::__set_state() . . . 186
Magic PHP5 method.
OTS_Player::__sleep() . . . 186
Magic PHP5 method.
OTS_Players_List::count() . . . 188
Returns number of characters on list in current criterium.
OTS_Players_List . . . 187
List of players.
OTS_Player::__wakeup() . . . 187
Magic PHP5 method.
OTS_Player::setLossMana() . . . 175
Sets percentage of used mana lost after dead.
OTS_Player::setLossExperience() . . . 174
Sets percentage of experience lost after dead.

OTS_Player::getTownId() . . . 163
Residence town's ID.
OTS_Player::getSoul() . . . 163
Soul points.
OTS_Player::getSlot() . . . 162
Returns items tree from given slot.
OTS_Player::getSkillTries() . . . 161
Returns player's skill's tries for next level.
OTS_Player::getVocation() . . . 163
Player proffesion.
OTS_Player::hasRedSkull() . . . 164
Checks if player has red skull.
OTS_Player::load() . . . 165
Loads player with given id.
OTS_Player::isSaveSet() . . . 164
Checks if save flag is set.
OTS_Player::isLoaded() . . . 164
Checks if object is loaded.
OTS_Player::getSkill() . . . 161
Returns player's skill.
OTS_Player::getSex() . . . 161
Player gender.
OTS_Player::getPosY() . . . 158
Y map coordinate.
OTS_Player::getPosX() . . . 158
X map coordinate.
OTS_Player::getName() . . . 158
Player name.
OTS_Player::getManaSpent() . . . 157
Mana spent.
OTS_Player::getPosZ() . . . 159
Z map coordinate.
OTS_Player::getPremiumEnd() . . . 159
Player's Premium Account expiration timestamp.
OTS_Player::getRedSkullTime() . . . 160
Red skulled time remained.
OTS_Player::getRankId() . . . 160
Guild rank ID.
OTS_Player::getRank() . . . 160
Assigned guild rank.
OTS_Player::save() . . . 165
Saves player in database.
OTS_Player::setAccount() . . . 165
Assigns character to account.
OTS_Player::setLookAddons() . . . 172
Sets addons.
OTS_Player::setLevel() . . . 171
Sets experience level.
OTS_Player::setLastLogin() . . . 171
Sets last login timestamp.
OTS_Player::setLastIP() . . . 171
Sets last login IP.
OTS_Player::setLookBody() . . . 172

Sets body color.
OTS_Player::setLookFeet() 173
Sets boots color.
OTS_Player::setLookType() 174
Sets outfit.
OTS_Player::setLookLegs() 174
Sets legs color.
OTS_Player::setLookHead() 173
Sets hair color.
OTS_Player::setHealthMax() 170
Sets maximum HP.
OTS_Player::setHealth() 170
Sets current HP.
OTS_Player::setCustomField() 167
Writes custom field.
OTS_Player::setConditions() 166
Sets conditions.
OTS_Player::setCap() 166
Sets capacity.
OTS_Player::setDepot() 167
Sets depot content.
OTS_Player::setDirection() 168
Sets looking direction.
OTS_Player::setGuildNick() 169
Sets guild nick.
OTS_Player::setGroup() 169
Assigns character to group.
OTS_Player::setExperience() 169
Sets experience points.
OTS_GuildRank::__set_state() 123
Magic PHP5 method.
OTS_GuildRank::__clone() 123
Creates clone of object.
OTS_Container 72
Container item representation.
OTS_Accounts_List::__wakeup() 71
Magic PHP5 method.
OTS_Accounts_List::__sleep() 71
Magic PHP5 method.
OTS_Accounts_List::__set_state() 70
Magic PHP5 method.
OTS_Container::addItem() 72
Adds item to container.
OTS_Container::count() 72
Number of items inside container.
OTS_Container::next() 74
Moves to next item.
OTS_Container::key() 73
Current cursor position.
OTS_Container::current() 73
Returns current item.
OTS_Accounts_List::valid() 70
Checks if there are any rows left.

[OTS_Accounts_List::setOffset()](#) 70
Sets OFFSET.
[OTS_Accounts_List::current()](#) 67
Returns current row.
[OTS_Accounts_List::count()](#) 67
Returns number of accounts on list in current criterium.
[OTS_Accounts_List](#) 66
List of accounts.
[OTS_Account::__wakeup()](#) 66
Magic PHP5 method.
[OTS_Accounts_List::deleteAccount()](#) 68
Deletes account.
[OTS_Accounts_List::key()](#) 68
Current cursor position.
[OTS_Accounts_List::setLimit()](#) 69
Sets LIMIT.
[OTS_Accounts_List::rewind()](#) 69
Select accounts from database.
[OTS_Accounts_List::next()](#) 69
Moves to next row.
[OTS_Container::removeItem()](#) 74
Removes given item from current container.
[OTS_Container::rewind()](#) 74
Resets internal items array pointer.
[OTS_DB_PostgreSQL](#) 82
PostgreSQL connection interface.
[OTS_DB_ODBC::tableName()](#) 81
Query-quoted table name.
[OTS_DB_ODBC::SQLquote()](#) 81
IOTS_DB method.
[OTS_DB_ODBC::SQLquery()](#) 80
IOTS_DB method.
[OTS_DB_PostgreSQL::fieldName()](#) 83
Query-quoted field name.
[OTS_DB_PostgreSQL::limit()](#) 83
LIMIT/OFFSET clause for queries.
[OTS_DB_PostgreSQL::tableName()](#) 85
Query-quoted table name.
[OTS_DB_PostgreSQL::SQLquote()](#) 84
IOTS_DB method.
[OTS_DB_PostgreSQL::SQLquery()](#) 84
IOTS_DB method.
[OTS_DB_ODBC::limit()](#) 80
LIMIT/OFFSET clause for queries.
[OTS_DB_ODBC::fieldName()](#) 79
Query-quoted field name.
[OTS_DB_MySQL::fieldName()](#) 76
Query-quoted field name.
[OTS_DB_MySQL](#) 75
MySQL connection interface.
[OTS_Container::valid()](#) 75
Checks if there are any items left.
[OTS_DB_MySQL::limit()](#) 76

LIMIT/OFFSET clause for queries.
OTS_DB_MySQL::SQLquery() 77
IOTS_DB method.
OTS_DB_ODBC 78
ODBC connection interface.
OTS_DB_MySQL::tableName() 78
Query-quoted table name.
OTS_DB_MySQL::SQLquote() 77
IOTS_DB method.
OTS_Account::__sleep() 66
Magic PHP5 method.
OTS_Account::__set_state() 65
Magic PHP5 method.
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::block() 54
Blocks account.
OTS_Account 54
OTServ account abstraction.
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::create() 55
Creates new account.
OTS_Account::createEx() 56
Creates new account.
OTS_Account::setEMail() 63
Sets account's email.
OTS_Account::setCustomField() 62
Writes custom field.
OTS_Account::save() 61
Updates account in database.
OTS_Account::load() 61
Loads account with given number.
OTS_Account::setGroup() 63
Assigns account to group.
OTS_Account::setPACCDays() 63
Sets PACC days count.
OTS_Account::__clone() 65
Creates clone of object.

OTS_Account::unblock() 64
Unblocks account.
OTS_Account::setPassword() 64
Sets account's password.
OTS_Account::isLoaded() 61
Checks if object is loaded.
OTS_Account::isBlocked() 60
Checks if account is blocked.
OTS_Account::getEMail() 58
E-mail address.
OTS_Account::getCustomField() 58
Reads custom field.
OTS_Account::find() 57
Loads account by it's e-mail address.
OTS_Account::getGroup() 59
Returns group of this account.
OTS_Account::getId() 59
Account number.
OTS_Account::getPlayers() 60
List of characters on account.
OTS_Account::getPassword() 60
Account's password.
OTS_Account::getPACCDays() 59
PACC days.
OTS_DB_SQLite 85
SQLite connection interface.
OTS_DB_SQLite::fieldName() 86
Query-quoted field name.
OTS_Guild::request() 111
Requests membership in guild for player player.
OTS_Guild::load() 110
Loads guild with given id.
OTS_Guild::listRequests() 110
Returns list of players that requested membership.
OTS_Guild::listInvites() 110
Returns list of invited players.
OTS_Guild::save() 111
Saves guild in database.
OTS_Guild::setCreationData() 112
Sets guild creation data.
OTS_Guild::setName() 113
Sets players's name.
OTS_Guild::setInvitesDriver() 113
Assigns invites handler.
OTS_Guild::setCustomField() 112
Writes custom field.
OTS_Guild::isLoaded() 109
Checks if object is loaded.
OTS_Guild::invite() 109
Invites player to guild.
OTS_Guild::getCreationData() 106
Guild creation data.
OTS_Guild::find() 106

Loads guild by it's name.
OTS_Guild::deleteRequest() . . . 105
Deletes request from player.
OTS_Guild::deleteInvite() . . . 105
Deletes invitation for player to guild.
OTS_Guild::getCustomField() . . . 107
Reads custom field.
OTS_Guild::getGuildRanks() . . . 107
Reads all ranks that are in this guild.
OTS_Guild::getOwner() . . . 108
Returns owning player of this player.
OTS_Guild::getName() . . . 108
Guild name.
OTS_Guild::getId() . . . 108
Guild ID.
OTS_Guild::setOwner() . . . 114
Assigns guild to owner.
OTS_Guild::setRequestsDriver() . . . 114
Assigns requests handler.
OTS_GuildRank::load() . . . 120
Loads rank with given id.
OTS_GuildRank::isLoaded() . . . 120
Checks if object is loaded.
OTS_GuildRank::getPlayers() . . . 120
Reads all players who has this rank set.
OTS_GuildRank::getName() . . . 119
Rank name.
OTS_GuildRank::save() . . . 121
Saves rank in database.
OTS_GuildRank::setCustomField() . . . 121
Writes custom field.
OTS_GuildRank::setName() . . . 123
Sets rank's name.
OTS_GuildRank::setLevel() . . . 122
Sets rank's access level within guild.
OTS_GuildRank::setGuild() . . . 122
Assigns rank to guild.
OTS_GuildRank::getLevel() . . . 119
Rank's access level.
OTS_GuildRank::getId() . . . 119
Rank ID.
OTS_Guild::__sleep() . . . 116
Magic PHP5 method.
OTS_Guild::__set_state() . . . 115
Magic PHP5 method.
OTS_Guild::__clone() . . . 115
Creates clone of object.
OTS_Guild::__wakeup() . . . 116
Magic PHP5 method.
OTS_GuildRank . . . 116
OTServ guild rank abstraction.
OTS_GuildRank::getGuild() . . . 118
Returns guild of this rank.

[OTS_GuildRank::getCustomField()](#) 118
Reads custom field.
[OTS_GuildRank::find()](#) 117
Loads rank by it's name.
[OTS_Guild::acceptRequest()](#) 104
Accepts player.
[OTS_Guild::acceptInvite()](#) 104
Finalise invitation.
[OTS_Group::isLoaded()](#) 92
Checks if object is loaded.
[OTS_Group::getPlayers()](#) 92
List of characters in given group.
[OTS_Group::getName()](#) 92
Group name.
[OTS_Group::getMaxVIPList()](#) 91
Maximum count of players in VIP list.
[OTS_Group::load()](#) 93
Loads group with given id.
[OTS_Group::save()](#) 93
Saves account in database.
[OTS_Group::setFlags()](#) 94
Sets rights flags.
[OTS_Group::setCustomField()](#) 94
Writes custom field.
[OTS_Group::setAccess()](#) 93
Sets access level.
[OTS_Group::getMaxDepotItems()](#) 91
Maximum count of items in depot.
[OTS_Group::getId()](#) 90
Group ID.
[OTS_DB_SQLite::SQLquote()](#) 87
IOTS_DB method.
[OTS_DB_SQLite::SQLquery()](#) 87
IOTS_DB method.
[OTS_DB_SQLite::limit()](#) 87
LIMIT/OFFSET clause for queries.
[OTS_DB_SQLite::tableName()](#) 88
Query-quoted table name.
[OTS_Group](#) 88
OTServ user group abstraction.
[OTS_Group::getFlags()](#) 90
Rights flags.
[OTS_Group::getCustomField()](#) 90
Reads custom field.
[OTS_Group::getAccess()](#) 89
Access level.
[OTS_Group::setMaxDepotItems()](#) 95
Sets maximum count of items in depot.
[OTS_Group::setMaxVIPList()](#) 95
Sets maximum count of players in VIP list.
[OTS_Groups_List::setOffset()](#) 101
Sets OFFSET.
[OTS_Groups_List::setLimit()](#) 101

Sets LIMIT.
OTS_Groups_List::rewind() 100
Select groups from database.
OTS_Groups_List::next() 100
Moves to next row.
OTS_Groups_List::valid() 101
Checks if there are any rows left.
OTS_Groups_List::__set_state() 102
Magic PHP5 method.
OTS_Guild 103
OTServ guild abstraction.
OTS_Groups_List::__wakeup() 103
Magic PHP5 method.
OTS_Groups_List::__sleep() 102
Magic PHP5 method.
OTS_Groups_List::key() 100
Current cursor position.
OTS_Groups_List::deleteGroup() 99
Deletes group.
OTS_Group::__set_state() 96
Magic PHP5 method.
OTS_Group::__clone() 96
Creates clone of object.
OTS_Group::setName() 96
Sets group's name.
OTS_Group::__sleep() 97
Magic PHP5 method.
OTS_Group::__wakeup() 97
Magic PHP5 method.
OTS_Groups_List::current() 99
Returns current row.
OTS_Groups_List::count() 99
Returns number of groups on list in current criterium.
OTS_Groups_List 98
List of groups.
OTS.php 27
This file contains main toolkit class.

P

POT::SLOT_NECKLACE 201
Necklace slot.
POT::SLOT_RIGHT 202
Right hand slot.
POT::SLOT_RING 202
Ring slot.
POT::VOCATION_DRUID 202
Druid.
POT::SLOT_LEGS 201
Legs slot.
POT::SLOT_LEFT 201
Left hand slot.

POT::SLOT_ARMOR 199
Armor slot.
POT::SLOT_BACKPACK 199
Backpack slot.
POT::SLOT_FEET 200
Boots slot.
POT::SLOT_HEAD 200
Head slot.
POT::VOCATION_KNIGHT 203
Knight.
POT::VOCATION_NONE 203
None vocation.
POT::getInstance() 206
Singleton.
POT::loadClass() 206
Loads POT class file.
POT::serverStatus() 207
Queries server status.
POT::setPOTPath() 208
Set POT directory.
POT::getDBHandle() 206
Returns database connection handle.
POT::createObject() 205
Creates OTServ DAO class instance.
POT::VOCATION_PALADIN 203
Paladin.
POT::VOCATION_SORCERER 204
Sorcerer.
POT::connect() 204
Connects to database.
POT::SLOT_AMMO 199
Ammunition slot.
POT::SKILL_SWORD 198
Sword fighting.
POT::DB_PGSQL 193
PostgreSQL driver.
POT::DB_SQLITE 194
SQLite driver.
POT::DEPOT_SID_FIRST 194
First depot item sid.
POT::DIRECTION_EAST 194
East.
POT::DB_ODBC 193
ODBC driver.
POT::DB_MYSQL 193
MySQL driver.
PHP 5.0 3
POT class preview 5
POT 192
Main POT class.
POT::DIRECTION_NORTH 195
North.
POT::DIRECTION_SOUTH 195

South.
POT::SKILL_DISTANCE 197
Distance fighting.
POT::SKILL_FISHING 197
Fishing.
POT::SKILL_FIST 198
Fist fighting.
POT::SKILL_SHIELDING 198
Shielding.
POT::SKILL_CLUB 196
Club fighting.
POT::SKILL_AXE 196
Axe fighting.
POT::DIRECTION_WEST 195
West.
POT::SEX_FEMALE 196
Female gender.
POT::SEX_MALE 196
Male gender.
POT 1

Q
Quick start 6

R
README 217

S
Server online status 19