mySQL DAO

Generated by Doxygen 1.8.6

Thu Jul 3 2014 12:03:12

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

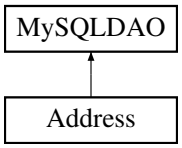
Chapter 3

Class Documentation

3.1 Address Class Reference

example class demonstrating the use of MySQLDAO

Inheritance diagram for Address:



Public Member Functions

- __construct ($newId, $newName, $newStreet1, $newStreet2, $newCity, $newState, $newZip)
  constructor for Address
- getId ()
  accessor method for id
- setId ($newId)
  mutator method for id
- getName ()
  accessor method for name
- setName ($newName)
  mutator method for name
- getStreet1 ()
  accessor method for first line of street address
- setStreet1 ($newStreet1)
  mutator method for first line of street address
- getStreet2 ()
  accessor method for second line of street address
- setStreet2 ($newStreet2)
  mutator method for second line of street address
- getCity ()
  accessor method for city
- setCity ($newCity)
  mutator method for city
- getState ()

accessor method for state

- setState ($newState)

  mutator method for state

- getZip ()

  accessor method for zip

- setZip ($newZip)

  mutator method for zip

Protected Attributes

- $id

  id of the address; this is the primary key

- $name

  name of the person or company

- $street1

  first line of the street address

- $street2

  second line of the street address

- $city

  city of the street address

- $state

  state of the street address

- $zip

  ZIP code of the street address.

Static Protected Attributes

- static $dao_primaryKey = "id"

  MySQLDAO's name of the primary key.

- static $dao_tableName = "address"

  MySQLDAO's name of the table.

- static $dao_typeMap

  MySQLDAO's map of the data types.

Additional Inherited Members

3.1.1 Detailed Description

example class demonstrating the use of MySQLDAO

This is a typical example of a class that represents a row in a mySQL table. Notice the class is simple and clean and consists of only typical accessor and mutator methods. All one needs to do is extend the MySQLDAO class and include the static DAO related variables to deploy the DAO pattern.

Author

Dylan McDonald dylanm@deepdivecoders.com

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Address::__construct (*$newId, $newName, $newStreet1, $newStreet2, $newCity, $newState, $newZip*)

constructor for Address

Parameters

integer	new value of id
string	new value of name
string	new value of first line of street address
string	new value of second line of street address
string	new value of city
string	new value of state
string	new value of zip

Exceptions

RuntimeException	if invalid inputs detected

3.1.3 Member Function Documentation

3.1.3.1 Address::getCity ( )

accessor method for city

Returns

string value of city

3.1.3.2 Address::getId ( )

accessor method for id

Returns

integer value of id

3.1.3.3 Address::getName ( )

accessor method for name

Returns

string value of name

3.1.3.4 Address::getState ( )

accessor method for state

Returns

string value of state

3.1.3.5 Address::getStreet1 ( )

accessor method for first line of street address

Returns

string value of first line of street address

3.1.3.6 Address::getStreet2 ( )

accessor method for second line of street address

Returns

string value of second line of street address

3.1.3.7 Address::getZip ( )

accessor method for zip

Returns

string value of zip

3.1.3.8 Address::setCity (*$newCity*)

mutator method for city

Parameters

string	new value of city

Exceptions

UnexpectedValueException	if location is empty

3.1.3.9 Address::setId (*$newId*)

mutator method for id

Parameters

integer	new value of id

Exceptions

RangeException	if id is negative
UnexpectedValueException	if id is invalid

3.1.3.10 Address::setName (*$newName*)

mutator method for name

Parameters

string	new value of name

Exceptions

UnexpectedValueException	if name is empty

3.1.3.11 Address::setState (*$newState*)

mutator method for state

Parameters

string	new value of state

Exceptions

UnexpectedValueException	if state is invalid

3.1.3.12 Address::setStreet1 (*$newStreet1*)

mutator method for first line of street address

Parameters

string	new value of first line of street address

Exceptions

UnexpectedValueException	if location is empty

3.1.3.13 Address::setStreet2 (*$newStreet2*)

mutator method for second line of street address

Parameters

string	new value of second line of street address

3.1.3.14 Address::setZip (*$newZip*)

mutator method for zip

Parameters

string	new value of zip

Exceptions

UnexpectedValueException	if zip is invalid

3.1.4 Member Data Documentation

3.1.4.1 Address::$dao_typeMap [static],[protected]

Initial value:

```
= array("id"  => "i", "name"  => "s", "street1" => "s", "street2" => "s",
                                      "city" => "s", "state" => "s", "zip"     => "s")
```

MySQLDAO's map of the data types.

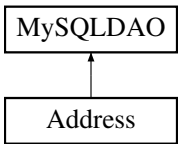
The documentation for this class was generated from the following file:

- address.php

3.2 MySQLDAO Class Reference

abstract class for the mySQL DAO design pattern

Inheritance diagram for MySQLDAO:



Public Member Functions

- delete (&$mysqli)

  deletes this object from mySQL
- insert (&$mysqli)

  inserts this object to mySQL
- update (&$mysqli)

  updates this object in mySQL

Protected Member Functions

- dao_generateBindParameters ($fieldList)

  generates an array to pass to mysqli_stmt::bind_param()
- dao_generateFieldList ($omitPrimaryKey)

  generates a comma separated field list
- dao_generateParameters ($numFields)

  generates placeholders for an INSERT query template
- dao_generateUpdateParameters ($fieldList)

  generates placeholders for an UPDATE query template

3.2.1 Detailed Description

abstract class for the mySQL DAO design pattern

This class is the abstract class for the mySQL Data Access Object (DAO) pattern. Each subclass of this class will receive the following public methods:

- delete(&$mysqli);
- insert(&$mysqli);
- update(&$mysqli);

In order to use the DAO pattern, subclasses must have the following protected static variables:

- dao_primaryKey: string containing the primary key's field name
- dao_tableName: string containing the mySQL table name
- dao_typeMap: array containing "field" => "type" pairs, where field is the field name and type is the data type according to mysqli_stmt::bind_param().

Author

Dylan McDonald dylanm@deepdivecoders.com

See Also

http://www.php.net/manual/en/mysqli-stmt.bind-param.php

3.2.2 Member Function Documentation

3.2.2.1 MySQLDAO::dao_generateBindParameters (*$fieldList*) [protected]

generates an array to pass to mysqli_stmt::bind_param()

Parameters

string	comma separated field list

Returns

array arguments to pass to mysqli_stmt::bind_param()

See Also

http://www.php.net/manual/en/mysqli-stmt.bind-param.php

3.2.2.2 MySQLDAO::dao_generateFieldList (*$omitPrimaryKey*) [protected]

generates a comma separated field list

Parameters

boolean	whether to omit the primary key

Returns

string comma separated field list

3.2.2.3 MySQLDAO::dao_generateParameters (*$numFields*) [protected]

generates placeholders for an INSERT query template

Parameters

integer	number of fields to template

Returns

string templated placeholders

3.2.2.4 MySQLDAO::dao_generateUpdateParameters (*$fieldList*) [protected]

generates placeholders for an UPDATE query template

Parameters

string	comma separated field list

Returns

string templated placeholders

3.2.2.5 MySQLDAO::delete (& *$mysqli*)

deletes this object from mySQL

Parameters

mysqli	pointer to valid mysqli connection

Exceptions

mysqli_sql_exception	if a mySQL error occurs

3.2.2.6 MySQLDAO::insert (& *$mysqli*)

inserts this object to mySQL

Parameters

mysqli	pointer to valid mysqli connection

Exceptions

mysqli_sql_exception	if a mySQL error occurs

3.2.2.7 MySQLDAO::update (& *$mysqli*)

updates this object in mySQL

Parameters

mysqli	pointer to valid mysqli connection

Exceptions

mysqli_sql_exception	if a mySQL error occurs

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

- mysqldao.php

Index