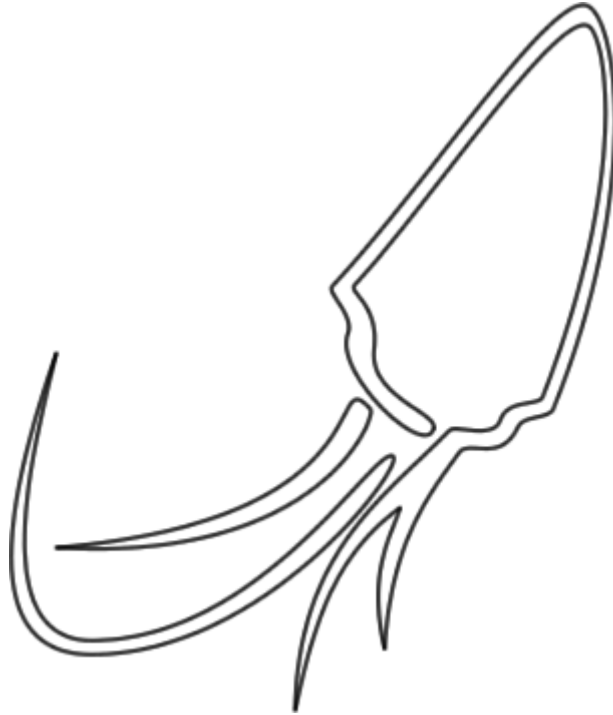


Mimesis v1.02n

Function Reference



Edition 1

desanitize

A function that desanitizes data

desanitize(*\$entry*)

parameter	type	description
<i>\$entry</i>	<i>string</i>	The serialized data to be desanitized

Returns

type	description
<i>array</i>	Desanitized data or FALSE on failure

file_cull_contents

A function that reads/writes content to a file

```
file_cull_contents($filename, $offset = 0, $bytes = null, $whence = SEEK_SET, $data = null)
```

parameter	type	description
\$filename	<i>string</i>	The file to be read/written
\$offset	<i>integer</i>	The offset from where to begin the read/write operation
\$bytes	<i>integer</i>	The number of bytes to be read
\$whence	<i>integer</i>	The location from where to compute offset for fseek
\$data	<i>string</i>	The data to be written

Returns

type	description
<i>mixed</i>	The number of bytes written, the bytes read, or FALSE on failure

file_place_contents

A function that writes content to a file

`file_place_contents($filename, $data)`

parameter	type	description
<code>\$filename</code>	<i>string</i>	The file to be written
<code>\$data</code>	<i>string</i>	The data to be written to the file

Returns

type	description
<i>integer</i>	The number of bytes written or FALSE on failure

Mimesis::__construct

The constructor sets the table and structural file uris, their existence, and the universal file permissions

Mimesis(*\$cwd*, *\$table*, *\$struct*)

parameter	type	description
<i>\$cwd</i>	<i>string</i>	A URI denoting the directory where the database tables are to be created/stored
<i>\$table</i>	<i>string</i>	The name of a database table
<i>\$struct</i>	<i>string</i>	The name of the structural file

Mimesis::table

A method that returns the table of interest

```
table($path = null)
```

parameter	type	description
<i>\$path</i>	<i>boolean</i>	Return table path or just the table name (name returned by default)

Returns

type	description
<i>string</i>	Table name or table path

Mimesis::struct

A method that returns the structural file of interest

```
struct($path = null)
```

parameter	type	description
<i>\$path</i>	<i>boolean</i>	Return the structure path or just the structure name (name returned by default)

Returns

type	description
<i>string</i>	Structural file name or path

Mimesis::tableExists

A method that returns the existence of the table of interest

`tableExists()`

Returns

type	description
<i>boolean</i>	Existence of table

Mimesis::structExists

A method that returns the existence of the structural file

`structExists()`

Returns

type	description
<i>boolean</i>	Structure existence

Mimesis::getRow

A method that retrieves rows from a table. With one parameter passed the method assumes that a regular expression pattern has been given, and all rows matching that pattern will be returned. With more than one parameter listed, those specific rows will be looked for without performing a pattern search. If no parameters are listed FALSE is returned and an E_USER_NOTICE level warning is issued.

getRow(...)

Returns

type	description
<i>array</i>	Array or FALSE on failure

Mimesis::query

A method that retrieves all rows in a table

query()

Returns

type	description
<i>array</i>	An array of tabular rows or FALSE on failure

Mimesis::insertRow

A method that inserts rows into a table (if the table does not exist it attempts to create it)

```
insertRow($data, $atomic = true)
```

parameter	type	description
\$data	<i>array</i>	An array of tabular rows to be inserted
\$atomic	<i>boolean</i>	Whether or not the file modifications should be atomic

Returns

type	description
<i>boolean</i>	TRUE on success, FALSE on failure

Mimesis::lock

A method that locks a table

```
lock($polling = null)
```

parameter	type	description
<i>\$polling</i>	<i>integer</i>	Specifies the sleep time (seconds) for the lock to wait in order to reacquire the lock if it fails. If not set, the mutex will only attempt to acquire once.

Returns

type	description
<i>boolean</i>	TRUE on success FALSE on failure

Mimesis::release

A method that releases the lock on a table

release()

Returns

type	description
<i>boolean</i>	TRUE on success FALSE on failure

Mimesis::deleteRow

A method that deletes rows within a table based on a regular expression pattern search

```
deleteRow($row, $atomic = true)
```

parameter	type	description
\$row	<i>string</i>	Regular expression pattern of rows to search for
\$atomic	<i>boolean</i>	Whether or not file modifications should be atomic

Returns

type	description
<i>boolean</i>	TRUE on success, FALSE on failure

Mimesis::deleteTable

A method that deletes a table and its structural file

```
deleteTable($atomic = true)
```

parameter	type	description
<i>\$atomic</i>	<i>boolean</i>	Whether or not file modifications should be atomic

Returns

type	description
<i>boolean</i>	TRUE on success, FALSE on failure

Mimesis::renameRow

A method that reassigns a label to a row

```
renameRow($old, $new, $atomic = true)
```

parameter	type	description
\$old	<i>string</i>	Regular expression pattern of rows to search for
\$new	<i>string</i>	The new label for the renamed rows
\$atomic	<i>boolean</i>	Whether or not file modifications should be atomic

Returns

type	description
<i>boolean</i>	TRUE on success, FALSE on failure

Mimesis::refresh

A method that refreshes a table by removing its row history

```
refresh($sort, $atomic = true)
```

parameter	type	description
\$sort	<i>boolean</i>	If refresh should also sort (ascending) by row labels
\$atomic	<i>boolean</i>	Whether or not file modifications should be atomic

Returns

type	description
<i>boolean</i>	TRUE on success, FALSE on failure

Mimesis::entries

A method that returns the historical number of entries and the unique number of entries within the table

entries()

Returns

type	description
<i>array</i>	An array whose first value is the historical count and the second value is the unique count

Mutex::_construct

The constructor sets up all the parameters to create the lock. The lock's timeout will always be at least twice that of the `max_execution_time`.

`Mutex($filename)`

parameter	type	description
<code>\$filename</code>	<i>string</i>	File to be locked

Mutex::acquireLock

A method that sets the lock on a file

```
acquireLock($polling = null)
```

parameter	type	description
<i>\$polling</i>	<i>integer</i>	Specifies the sleep time (seconds) for the lock to wait in order to reacquire the lock if it fails. If not set, the mutex will only attempt to acquire once.

Returns

type	description
<i>boolean</i>	TRUE on success FALSE on failure

Mutex::releaseLock

A method that releases the lock on a file

```
releaseLock()
```

Returns

type	description
<i>boolean</i>	TRUE on success FALSE on failure

Mutex::lockTime

A method that returns the timeout value set to a lock

lockTime()

Returns

type	description
<i>integer</i>	Seconds on success FALSE on failure

Mutex::lockID

A method that returns the mutex id set to a lock

lockID()

Returns

type	description
<i>string</i>	Mutex id on success FALSE on failure

Mutex::lockAbort

A method that ensures lock release upon an abnormal script termination condition

```
lockAbort()
```

Polarizer::__construct

The constructor determines the procedure to follow on how to parse the serialized input strings dependent on one or two parameters being passed to it. If one parameter is passed then it assumes a serialized array and will thus split the serialized array string into two strings of keys and values respectively. If two parameters are passed then the constructor assumes that it is being given serialized keys and serialized values and therefore recombines them into a serialized array.

`Polarizer($keys, $values = null)`

parameter	type	description
<code>\$keys</code>	<i>mixed</i>	Array or serialized key(s)
<code>\$values</code>	<i>string</i>	Serialized values(s)

Polarizer::getKeys

A method that returns serialized keys from an array

getKeys()

Returns

type	description
<i>string</i>	Serialized key(s) or FALSE on failure

Polarizer::getValues

A method that returns serialized values from an array

getValues()

Returns

type	description
<i>string</i>	Serialized value(s) or FALSE on failure

Polarizer::getArr

A method that returns an array from serialized keys and serialized values

getArr()

Returns

type	description
<i>array</i>	Array or FALSE on failure

sanitize

A function that sanitizes data

sanitize(*\$entry*)

parameter
\$entry

type
mixed

description
The data to be sanitized

Returns

type
string

description
Sanitized data

singularID

A function that generates unique ids

```
singularID($prefix = 'sid', $tempdir = '_cache')
```

parameter	type	description
<i>\$prefix</i>	<i>string</i>	The 3 character prefix for the id
<i>\$tempdir</i>	<i>string</i>	The temporary directory to use for id generation

Returns

type	description
<i>string</i>	Unique id or FALSE on failure

str_offsets

A function that returns the offsets of all substrings within a string

```
str_offsets($haystack, $needle, $offset = 0)
```

parameter	type	description
<i>\$haystack</i>	<i>string</i>	The string to be searched
<i>\$needle</i>	<i>string</i>	The string to search for
<i>\$offset</i>	<i>boolean</i>	Location in haystack to begin searching from

Returns

type	description
<i>array</i>	An array of offsets, or FALSE on none found

str_r_pos

A function that returns the rightmost offset of a substring within a string

```
str_r_pos($haystack, $needle, $offset = 0)
```

parameter	type	description
\$haystack	<i>string</i>	The string to be searched
\$needle	<i>string</i>	The string to search for
\$offset	<i>boolean</i>	Location in haystack to begin searching from

Returns

type	description
<i>integer</i>	The substring offset, or FALSE if not found