Dokumentation für die KIMBdbf PHP-Klasse

KIMB-technologies

12. Dezember 2016

Inhaltsverzeichnis

 © KIMB-technologies

 © KIMB-technologies

1 Allgemein

Die KIMBdbf PHP-Klasse (KIMB databasefile) für PHP ersetzt eine Datenbank, sie kann aber auch in ergänzug zu einer Datenbank verwendet werden, z.B. zur Speicherung der Zugangsdaten.
Alle Datensätze werden komplett in Dateien direkt auf dem Server gespeichert.
Die Klasse ist komplett in PHP geschrieben und jedes Skript muss nur die Klassendatei laden per `require` oder mit einem autoload.
Die Klasse beinhaltet viele Methoden zur Datenspeicherung, so ist lesen, schreiben und Suchen von bestimmten Datensätzen möglich. Es Funktionen die den elementaren Methoden entsprechen.
Es wird PHP 5 (größer 5.5.0) oder PHP 7 benötigt. Für verschlüsselte Dateien ist PHP Mcrypt erforderlich.

1.1 Dateiaufbau

1.1.1 Beispieldatei

```
<[about:doc]>KIMB dbf V4.10 - KIMB-technologies<[about:doc]>
<[user]>mmuster<[user]>
<[name]>Max Mustermann<[name]>
<[aboutme]>Ich bin der Max.<[aboutme]>
<[passwort]>2f4cdd81258da5f104bb343ab9d13ef<[passwort]>
<[groups]>chemie<[groups]>
<[groups]>physik<[groups]>
<[membership1]>newsletter chemie<[membership1]>
<[membership2]>newsletter physik<[membership2]>
<[membership3]>newsletter uni<[membership3]>
<[105]>note==abschluesse<[105]>
<[105-note]>1<[105-note]>
<[105-abschluesse]>Chemie<[105-abschluesse]>
```

1.1.2 Datensatzarten

Es gibt drei verschiedenen Arten von Datensätzen:

Normal

Es wird einem Tag ein Inhalt zugeordnet.
Tags können einmal oder mehrfach vorhanden sein.
Ein Tag kann nur ersetzt oder gelöscht werden, wenn dieser nur einmal in einer Datei vorhanden ist.
Es kann aber in den Inhalten mehrfach vorhandener Tags gesucht werden.
Beispielanwendung: Logdatei, Benutzerdatei, Konfigurationsdatei

Aufzählung

Es werden einem Tag mehrere Inhalte zugeordnet.
Es können einzelne Inhalte gelöscht werden.
Beispielanwendung: Zugriffsrechte, Newsletter, Gruppen

ID-Zuordnung

Es wird eine ID erstellt, welche eigenen Untergruppen erhält und diese mit Inhalten füllen kann.
Es können alle Inhalte einer ID bestimmt werden, genauso können alle IDs nach einem bestimmten Inhalt durchsucht werden.
Es können einzelne Inhalte bearbeitet werden.
Genau wie eine Tabelle einer Datenbank.
Beispielanwendung: Auflistungen mit Attributen, Benutzerdatei, Konfigurationsdatei
Alle IDs in einer Datei werden automatisch der Aufzählung `allidslist` hinzugefügt.

1.2 Lizenz

Die KIMBMySQL PHP-Klasse ist unter der GPLv3 veröffentlicht.
http://www.gnu.org/licenses/gpl-3.0.txt

2 Einbau

Die KIMBdbf PHP-Klasse kann per autoload in ein Projekt eingebunden werden. Sie ist unter dem `namespace KIMBdbf;` zu erreichen.
Die Klassendatei erhalten Sie (hier).

Zur Speicherung der Daten werden Dateien genutzt. Ohne besondere Konfiguration wird automatisch ein Unterordner „/kimb-data/“ bei der Klassendatei zur Speicherung der Dateien gewählt. (Dieser kann bei der Initialisierung des Objektes anders gewählt werden.)
Dieser Ordner muss vorhanden und für PHP schreibbar sein, aus Sicherheitsgründen sollte er nicht mit einem Browser zu erreichen sein.

Seit Version 4.5 werden die KIMB-Dateien beim Schreiben durch die KIMBdbf-Klasse gesperrt, es ist somit kein paralleles Schreiben verschiedener Prozesse/ Objekte möglich.
Sollte das Schreiben einmal nicht möglich sein, führt die KIMBdbf 10 Versuche durch, doch noch Schreibrechte zu erhalten (mit je 0,75 Sekunden Pause). Sollte sie dabei keine Rechte erhalten, wird das Skript beendet!
Gleichzeitiges Lesen in einer Datei von verschiedenen Prozessen/ Objekten aus stellt kein Problem dar!

 © KIMB-technologies

3 Verwendung

3.1 Kodierung, Vergebene Zeichen

Die KIMBdbf verlangt keine bestimmte Kodierung, da PHP Datensätze immer so in Dateien speichert, wie es diese erhält. Das bedeutet, man muss sich für eine Kodierung entschieden und alle Datensätze kommen dann so zurück wie diese übergeben wurden.
Es ist sicherlich sinnvoll weit verbeitete Kodierungen zu verenden, z.B. UTF-8.

Dateiname

Dateinamen dürfen nur aus Zeichen von A-Z (groß und klein) sowie Ziffern bestehen. Außderdem sind die folgenden Sonderzeichen erlaubt: `_ . - /`
Deutsche Umlaute, das Sz sowie Punkte und Leerzeichen, werden automatisch umgeschrieben. Alle anderen verbotenen Zeichen werden herausgeschnitten.

Tag

Die Tags dürfen aus allen Zeichen bestehen, mit Ausnahmen von `<[` , `]>` und `about:doc` , welche zu `<` , `>` und `aboutdoc` umgeschrieben werden.
Auch Zeilenumbrüche sowie Tabulatoren sind verboten, diese werden herausgeschnitten.

Inhalt

Inhalte können alle Zeichen haben.

Intern werden einige Zeichen bzw. Zeichenketten kodiert gespeichert und bei der Ausgabe zurückverwandelt. (Umbrüche, Tabulatoren, `<[` , `]>` und `==`) Die kodierte Speicherung benötigt etwas mehr Speicherplatz.

Ab Version 4.5 ist es möglich neben Strings als Inhalten auch Arrays as Inhalte zu speichern.

 © KIMB-technologies

Die Arrays werden bei der Speicherung automatisch erkannt und als JSON abgelegt.
Für die Ausgabe wird das JSON dann wieder zu PHP-Arrays konvertiert.

In der Dokumentation ist weiterhin String als Typ für die Parameter und Rückgaben der Inhalte angeben. Sie können hier aber auch Arrays übergeben, bzw. müssen mit Rückgaben als Array rechnen.
(Arrays kommen natürlich nur zurück, wenn diese auch bei der Speicherung übegeben wurden.)

 © KIMB-technologies

3.2 Objekt erstellen

Erstellen Sie zuerst ein Objekt der KIMBdbf PHP-Klasse.

```
$dbf = new KIMBdbf\KIMBdbf(
          $datei [, $encryptkey = 'off' [, $path = __DIR__ ]]
       );
```

Parameter		
$datei	String	*Name der KIMB-Datei, in der das Objekt arbeiten soll. (üblicherweise *.kimb) (Unterordner möglich) (Datei muss nicht vorhanden sein, wird automatisch erstellt)*
$encryptkey	Sting	*Passwort, mit dem die Datei verschlüsselt wurde/ verschlüsselt werden soll. (Verschlüsselung über PHP Mcrypt;* `MCRYPT_BLOWFISH , MCRYPT_MODE_CBC` *) (nicht übergeben =>* `off` *=> keine Verschlüsselung)*
$path	String	*Pfad zum Ordner „/kimb-data/“. (nicht übergeben =>* `__DIR__` *=> Unterordner direkt bei der Klassendatei)*
Rückgabe		
KIMBdbf Objekt		

Anschließend können sie gleich mit dem Objekt arbeiten. Sie können natürlich auch mehrere KIMBdbf Objekte Parallel verwenden.

3.3 Methodenübersicht

Die Übersicht zeigt die verschiedenen Methoden, sortiert nach den Datensatzarten.

	Lesen & Suchen	Schreiben
Normal	$this->read_kimb_one $this->read_kimb_all $this->read_kimb_search	$this->write_kimb_new $this->write_kimb_replace $this->write_kimb_delete $this->write_kimb_one
Aufzählung	$this->read_kimb_all_teilpl $this->read_kimb_search_teilpl	$this->write_kimb_teilpl $this->write_kimb_teilpl_del_all
ID-Zuordnung	$this->read_kimb_id $this->read_kimb_all_xxxid $this->read_kimb_id_all $this->search_kimb_id $this->search_kimb_xxxid	$this->next_kimb_id $this->write_kimb_id $this->write_kimb_id_array
Weitere	$this->show_kimb_file $this->delete_kimb_file	

Klicken Sie auf eine Methode und sie werden zur genauen Erklärung geleitet.

3.4 Methoden

3.4.1 $this->read_kimb_one

Lesen des Inhalts eines Tags. (wenn mehrfach vorhanden, erster Inhalt)

```
$dbf->read_kimb_one( $teil );
```

Parameter		
$teil	String	*Tag des Inhalts*
Rückgabe		
	String	*Inhalt*

Zur Methodenübersicht

3.4.2 $this->read_kimb_all

Lesen aller Inhalte eines Tags.

```
$dbf->read_kimb_all( $teil );
```

Parameter		
$teil	String	*Tag der Inhalte*
Rückgabe		
	Array	*Inhalte*

Zur Methodenübersicht

3.4.3 $this->read_kimb_search

Suchen nach einem Tag mit bestimmtem Inhalt, bei mehrfach vorhanden Tags.

```
$dbf->read_kimb_search( $teil, $search );
```

Parameter		
$teil	String	*Tag der zu durchsuchenden Inhalte*
$search	String	*Gesuchter Inhalt*
Rückgabe		
	Boolean	*Gefunden?*

Zur Methodenübersicht

3.4.4 $this->write_kimb_new

Einen Inhalt in einen neuen Tag schreiben.

```
$dbf->write_kimb_new( $teil, $inhalt );
```

Parameter		
$teil	String	*Tag für den neuen Inhalt*
$inhalt	String	*Neuer Inhalt*
Rückgabe		
	Boolean	*Erfolgreich?*

Zur Methodenübersicht

3.4.5 $this->write_kimb_replace

Einen Inhalt in einem vorhanden Tag ersetzen. (Tag darf nur einmal vorhanden sein)

```
$dbf->write_kimb_replace( $teil, $inhalt );
```

Parameter		
$teil	String	*Tag mit dem alten Inhalt*
$inhalt	String	*Neuer Inhalt*
Rückgabe		
	Boolean	*Erfolgreich?*

Zur Methodenübersicht

3.4.6 $this->write_kimb_delete

Einen Tag löschen. (Tag darf nur einmal vorhanden sein)

```
$dbf->write_kimb_delete( $teil );
```

Parameter		
$teil	String	*Zu löschender Tag*
Rückgabe		
	Boolean	*Erfolgreich?*

Zur Methodenübersicht

3.4.7 $this->write_kimb_one

Inhalt einfach unter einem Tag ablegen. (Tag wird erstellt, sofern dieser noch nicht vorhanden ist, andernfalls wird der Inhalt ersetzt.)

```
$dbf->write_kimb_one( $teil, $inhalt );
```

Parameter		
$teil	String	*Tag für den Inhalt*
$inhalt	String	*Neuer Inhalt*
Rückgabe		
	Boolean	*Erfolgreich?*

 © KIMB-technologies

Zur Methodenübersicht

3.4.8 $this->read_kimb_search_teilpl

Inhalt unter einem Tag in einer Aufzählung suchen.

```
$dbf->read_kimb_search_teilpl( $teil, $search );
```

Parameter		
$teil	String	*Tag der zu durchsuchenden Inhalte*
$search	String	*Gesuchter Inhalt*
Rückgabe		
	Boolean	*Gefunden?*

Zur Methodenübersicht

3.4.9 $this->read_kimb_all_teilpl

Alle Inhalte unter einem Tag in einer Aufzählung ausgeben.

```
$dbf->read_kimb_all_teilpl( $teil );
```

Parameter		
$teil	String	*Tag der Inhalte*
Rückgabe		
	Array	*Inhalte*

Zur Methodenübersicht

3.4.10 $this->write_kimb_teilpl

Inhalte einer Aufzählung hinzufügen, ändern und löschen.

```
$dbf->write_kimb_teilpl( $teil, $inhalt, $todo );
```

Parameter		
$teil	String	*Tag des Inhalts/ der Aufzählung*
$inhalt	String	*Neuer oder zu löschender Inhalt*
$todo	String	`add` *Inhalt neu hinzufügen oder verändern*
		`del` *Inhalt aus der Aufzählung löschen*
Rückgabe		
	Boolean	*Erfolgreich?*

Zur Methodenübersicht

3.4.11 $this->write_kimb_teilpl_del_all

Alle Inhalte und die gesamte Aufzählung löschen.

```
$dbf->write_kimb_teilpl_del_all( $teil );
```

Parameter		
$teil	String	*Tag der Inhalte/ Aufzählung*
Rückgabe		
	Boolean	*Erfolgreich?*

Zur Methodenübersicht

3.4.12 $this->read_kimb_id

Inhalte einer Untergruppe einer ID lesen oder alle Untergruppen einer ID lesen.

```
$dbf->read_kimb_id( $id [, $xxxid = '---all---'] );
```

Parameter		
$id	Int	*ID mit Inhalten und Untergruppen*
$xxxid	String	*Name der Untergruppe, deren Inhalt ausgeben werden soll (nicht übergeben => Array mit Inhalten aller Untergruppen)*
Rückgabe		
	String	*Inhalt der gewählten Untergruppe*
	Array	*Inhalt aller Untergruppen der ID* abhängig von $xxxid

 © KIMB-technologies

Zur Methodenübersicht

3.4.13 $this->read_kimb_all_xxxid

Alle Untergruppen einer ID ausgeben.

```
$dbf->read_kimb_all_xxxid( $id );
```

Parameter		
$id	Int	*ID, von der man die Untergruppen wissen will*
Rückgabe		
	Array	*Array mit allen Untergruppen der gewählten ID*

Zur Methodenübersicht

3.4.14 $this->read_kimb_id_all

Alle Inhalte aller IDs in der Datei ausgeben.

```
$dbf->read_kimb_id_all();
```

Parameter	
keine	
Rückgabe	

Array	*Array mit allen IDs, Untergruppen und Inhalten*
	`array( ID =>`
	`array( XXXID => Inhalt, ... ),`
	`ID => .... )`

Zur Methodenübersicht

3.4.15 $this->search_kimb_id

Suchen nach einem Inhalt in allen Untergruppen einer ID.

```
$dbf->search_kimb_id( $search , $id );
```

Parameter		
$search	String	*Gesuchter Inhalt einer Untergruppe*
$id	Int	*ID, deren Untergruppen durchsucht werden*
Rückgabe		
	Boolean/ String	*Wenn Inhalt gefunden, dann Name der Untergruppe, sonst false*

Zur Methodenübersicht

3.4.16 $this->search_kimb_xxxid

Durchsuchen von allen IDs nach einem Inhalt in einer Untergruppe.

```
$dbf->search_kimb_xxxid( $search, $xxxid );
```

Parameter		
$search	String	*Gesuchter Inhalt einer Untergruppe*
$xxxid	String	*Untergruppe, welche durchsucht werden soll*
Rückgabe		
	Boolean/ Int	*Wenn Inhalt gefunden, dann ID, sonst false*

Zur Methodenübersicht

3.4.17 $this->next_kimb_id

Herausfinden der nächsten freien ID in der aktuellen KIMBdbf.

```
$dbf->next_kimb_id();
```

Parameter
keine
Rückgabe
Int *Nächste freie ID*

Zur Methodenübersicht

3.4.18 $this->write_kimb_id

Schreiben und löschen von Inhalten in IDs und Untergruppen.

```
$dbf->write_kimb_id(
  $id, $todo [, $xxxid = '---none---'
               [, $inhalt = '---none---' [, $oldmode = false ]]]
);
```

Parameter		
$id	Int	*ID, in der geschrieben werden soll (wird automatisch erstellt, sollte sie leer sein)* *(Bei `0` wird automatisch eine freie ID gesucht. [Auto_Increment])* *(Unter `public KIMBdbf::last_written_id` finden Sie die zuletzt veränderte ID.)*
$todo	String	`add` *Untergruppe neu hinzufügen oder Inhalt verändern* `del` *Untergruppe oder ID löschen*
$xxxid	String	*Untergruppe, in der gearbeitet werden soll (nicht übergeben & $todo `del` => gesamte ID löschen)*
$inhalt	String	*Inhalt, der geschrieben werden soll (nicht übergeben nur sinnvoll wenn ID oder Untergruppe gelöscht werden soll)*
$oldmode	Boolean	*Auto_Increment deaktivieren, dadurch lässt sich die ID `0` schreiben.*
Rückgabe		
	Boolean	*Erfolgreich?*

Zur Methodenübersicht

3.4.19 $this->write_kimb_id_array

Ein Array mit mehreren IDs in einer KIMBdbf ablegen.
Quasi das schreibende Gegenstück zu `$dbf->read_kimb_id_all()`.

```
$dbf->write_kimb_id_array( $array );
```

Parameter		

$array	Array	*Array mit IDs, Untergruppen und Inhalten*
		`array( ID =>`
		`array( XXXID => Inhalt, ... ),`
		`ID => .... )`
		Achtung, die Inhalte der IDs/ Untergruppen werden überschrieben.

Rückgabe		
	Boolean	*Erfolgreich?*

Zur Methodenübersicht

3.4.20 $this->delete_kimb_file

Die gesamte KIMB-Datei löschen.

```
$dbf->delete_kimb_file();
```

Parameter	
keine	
Rückgabe	
Boolean	*Erfolgreich?*

Zur Methodenübersicht

3.4.21 $this->show_kimb_file

Die gesamte KIMB-Datei, wie sie auf dem Dateisystem liegt, zurückgeben.

```
$dbf->show_kimb_file();
```

Parameter	
keine	
Rückgabe	
String	*Dateiinhalt*

Zur Methodenübersicht

3.5 Funktionen

Bei den Funktionen ist nur der Zugriff auf die Datensatzart „Normal“ möglich. Außerdem ist keine Wahl des „/kimb-data/“ Pfades möglich. Auch die Verschlüsselung ist immer deaktviert.
Die Funktionen arbeiten technisch gesehen auch mit der KIMBdbf Klasse.

3.5.1 read_kimb_one

Lesen des Inhalts eines Tags. (wenn mehrfach vorhanden, erster Inhalt)

```
read_kimb_one( $datei, $teil );
```

Parameter		
$datei	String	*Name der KIMB-Datei (wie für das Objekt)*
$teil	String	*Tag des Inhalts*
Rückgabe		
	String	*Inhalt*

3.5.2 read_kimb_search

Suchen nach einem Tag mit bestimmtem Inhalt, bei mehrfach vorhanden Tags.

```
read_kimb_search( $datei, $teil, $search );
```

Parameter		
$datei	String	*Name der KIMB-Datei (wie für das Objekt)*
$teil	String	*Tag der zu durchsuchenden Inhalte*
$search	String	*Gesuchter Inhalt*
Rückgabe		
	Boolean	*Gefunden?*

3.5.3 write_kimb_new

Einen Inhalt in einen neuen Tag schreiben.

```
write_kimb_new( $datei, $teil, $inhalt );
```

Parameter		
$datei	String	*Name der KIMB-Datei (wie für das Objekt)*
$teil	String	*Tag für den neuen Inhalt*
$inhalt	String	*Neuer Inhalt*
Rückgabe		
	Boolean	*Erfolgreich?*

3.5.4 write_kimb_replace

Einen Inhalt in einem vorhanden Tag ersetzen. (Tag darf nur einmal vorhanden sein)

```
write_kimb_replace( $datei, $teil, $inhalt );
```

Parameter		
$datei	String	*Name der KIMB-Datei (wie für das Objekt)*
$teil	String	*Tag mit dem alten Inhalt*
$inhalt	String	*Neuer Inhalt*
Rückgabe		
	Boolean	*Erfolgreich?*

3.5.5 write_kimb_delete

Einen Tag löschen. (Tag darf nur einmal vorhanden sein)

```
write_kimb_delete( $datei, $teil );
```

Parameter		
$datei	String	*Name der KIMB-Datei (wie für das Objekt)*
$teil	String	*Zu löschender Tag*
Rückgabe		
	Boolean	*Erfolgreich?*

3.5.6 delete_kimb_datei

Die gesamte KIMB-Datei löschen.

```
delete_kimb_datei( $datei );
```

Parameter		
$datei	String	*Name der KIMB-Datei (wie für das Objekt)*
Rückgabe		
	Boolean	*Erfolgreich?*

3.6 Überlagerung

Bei den KIMB-Dateien ist kein vollständiger Schutz vor Überschreiben/ Mehrfachbelegung eingebaut, so kann man mit einer Methode für einen normalen Datensatz auch Aufzählung oder ID- Zuordnungen umschreiben. Dies ist kein Fehler, sollte aber jedem Entwickler im Hinterkopf bleiben. Dadurch ermöglicht sich ein einfacher Aufbau eigener Funktionen oder auch das Bearbeiten eines Inhalts ohne die dafür vorgesehene Funktion.
Um Überschreiben zu verhindern bietet sich folgendes an:

1. Alle ID's sind Zahlen
2. Keine Tags fangen mit Zahlen an (Schutz für ID-Zuordnung)
3. Keine Tags enden mit Zahlen (Schutz für Aufzählungen)
4. Aufzählungen erhalten einen Präfix, z.B.: „tp-“
5. Untergruppen bestehen nur aus Buchstaben
6. Normale Datensätze erhalten einen Präfix, z.B.: „nt-“

 © KIMB-technologies

4 Links

KIMBdbf

https://github.com/kimbtech/KIMBdbf

KIMB-technologies

CMS, Downloader und Co.

https://kimbtech.github.io/

GIT

https://bitbucket.org/kimbtech/

https://gtihub.com/kimbtech/

Lizenz

http://www.gnu.org/licenses/gpl-3.0.txt