uverein

Handbuch für Entwickler

Version: 0.2.3

letzte Änderung: 05.10.2008

Inhaltsverzeichnis

database.inc (class cl_database)

Diese Klasse ist für den Datenbankzugriff verantwortlich. Die Initialisierung erfolgt durch Übergabe des Datenbanknamens.

```
datenbank = new cl_database (beispieldb);
```

uverein
database.inc (class cl_database)

error_name

Function error_name ($key , $inhalt)

$key Fehlercode lt. untenstehender Tabelle

$inhalt Text, der ausgegeben werden soll

Mit dieser Funktion können Fehlermeldungen eingetragen werden. Die folgenden Fehlercodes sind vorhanden und können geändert werden:

Fehlercode:	Voreinstellung:	Bedeutung:
CL_DATABASE_OK	OK	Es sind keine Fehler aufgetreten
CL_DATABASE_ERROR_CANTOPEN	Can't open Database	Datenbank kann nicht geöffnet werden
CL_DATABASE_ERROR_SQL	Error while executing SQL-query	Fehler bei der Ausführung eines SQL-Befehls
CL_DATABASE_ERROR_UNKNOWN	Unknown Error	Es ist ein Fehler aufgetreten, für den kein anderer Fehlercode vorhanden ist.

```
$datenbank->error_name (CL_DATABASE_OK,“Alles OK“);
```

Die Funktion gibt kein Ergebnis zurück.

uverein
database.inc (class cl_database)

error_text

Function error_text ()

Diese Funktion gibt den Status der letzten Aktion zurück. Ausgegeben wird der durch die Funktion error_name voreingestellte Text.

```
echo $datenbank->error_text();
```

Die Funktion gibt die letzte Fehlermeldung im Klartext zurück.

uverein
database.inc (class cl_database)

fieldnames

function fieldnames ([$qu])

$qu	optionale Angabe des Antwortzeigers

Die Funktion liefert ein Array mit den Spaltenüberschriften = Feldnamen zur letzten oder optional der durch den Antwortzeiger definierten SQL-Abfrage zurück.

```
$qu = $datenbank->sql_befehl („SELECT * FROM belege“);
$FELDER = $datenbank->fieldnames();
foreach ($FELDER as $key => $inhalt)
{
  echo „$key = $inhalt <BR>“;
}
```

Der Rückgabewert ist ein Array mit den Feldnamen. Wenn die letzte Anfrage keine Felder lieferte wird ein leerer String zurück geliefert.

result_array

function result_array ([$qu])

$qu	optionale Angabe des Antwortzeigers

Die Funktion gibt das Ergebnis der letzten oder als Antwortzeiger übergebenen Anfrage als zweidimensionales Array zurück.

Die Adressierung im Array geschieht durch Angabe von Zeilen und Spalten-Nr. oder alternativ durch Zeilen-Nr. und Feldname.

```
$qu = $datenbank->sql_befehl („SELECT * FROM belege“);
$data = $datenbank->result_array ();
echo „In Feld 0 der Spalte 0 steht: „.
     $data[0][0];
echo „Im Feld kontonr der Spalte 2 steht: „.
     $data[2][„kontonr“];
```

Wenn die letzte Abfrage kein Ergebnis liefert ist die Rückgabe nicht definiert.

sql_1_value

function sql_1_value ($tabelle, $wert, $where)

$tabelle	Welche Datenbank-Tabelle soll abgefragt werden ?
$wert	Welches Feld soll geholt werden ?
$where	Abfrage um nur ein Resultat zu erhalten

Die Funktion holt einen Wert aus der Datenbank. Die WHERE-Formulierung sollte daher ganz genau überdacht werden.

```
$nr = $datenbank->sql_1_value („belege“,“max(nr) + 1“,““);
// gibt die nächste Beleg-Nr. zurück
echo „Die nächste Beleg-Nr. ist $nr“;
```

Als Resultat wird das gesuchte Feld zurück geliefert. Wenn WHERE kein Ergebnis lieferte, wird ein leerer String zurück gegeben.

uverein
database.inc (class cl_database)

sql_abfrage

function sql_abfrage ($tabelle, $where, [[$order], $group])

$tabelle	Welche Datenbank-Tabelle soll abgefragt werden ?
$where	Abfrage
$order	Optionale Sortierung der Rückgabe
$group	Optionale Gruppierung. Wenn hier Angaben gemacht werden muss beachtet werden, dass alle Felder abgefragt werden – im Normalfall bleibt dieses Feld daher leer

Die Funktion setzt aus den übergebenen Daten einen SQL-Befehl zusammen und gibt diesen an die Datenbank weiter.

```
$qu = $datenbank->sql_abfrage („belege“,“nr>10“);
// gibt alle Belege mit Beleg-Nr. > 10 zurück
$i = 0;
while ($i < $datenbank->sql_lines())
{
  $data = $datenbank->sql_line ($i);
  echo „$data->nr  $data->datum $data->text <BR>“;
  $i++;
}
```

Es wird ein Zeiger auf die Antwort zurück geliefert.

uverein
database.inc (class cl_database)

sql_befehl

function sql_befehl ($befehl)

$befehl	SQL-Befehl, der ausgeführt werden soll

Die Funktion schickt den übergebenen SQL-Befehl direkt an die Datenbank.

```
$qu = $datenbank->sql_befehl („SELECT * FROM belege“);
$i = 0;
while ($i < $datenbank->sql_lines($qu))
{
  $data = $datenbank->sql_line ($i,$qu);
  echo „$data->nr  $data->datum $data->text <BR>“;
  $i++;
}
```

Es wird ein Zeiger auf die Antwort zurück geliefert.

sql_fields

function sql_fields ([$qu])

$qu	optionale Angabe des Antwortzeigers

Die Funktion liefert die Anzahl der Spalten zur letzten oder optional der durch den Antwortzeiger definierten SQL-Abfrage zurück.

```
$qu = $datenbank->sql_befehl („SELECT * FROM belege“);
echo „Die Anfrage hat „;
echo $datenbank->sql_fields());
echo „ Spalten zurück gegeben!“;
```

Der Rückgabewert ist eine Zahl >= 0

sql_insert

function sql_insert ($tabelle, $was, $inhalt)

$tabelle	In welche Datenbank-Tabelle soll eingefügt werden ?
$was	Welche Felder werden eingetragen ?
$inhalt	Die VALUEs, die eingetragen werden

Die Funktion fügt eine neue Zeile in die Datenbank ein. Die Anzahl der Felder muss mit den in $inhalt angegebenen Feldern übereinstimmen. Die Funktion prüft nicht ob die Zeile tatsächlich eingefügt werden kann. Konnte die Zeile nicht eingefügt werden ist der Errorcode auf CL_DATABASE_ERROR_SQL gesetzt.

```
$datenbank->sql_insert
(„belege“,“nr,datum,betrag“,“'21','20.10.2007','100.99');
```

Die Funktion gibt kein Resultat zurück. Ob ein Fehler aufgetreten ist kann über den Errorcode abgefragt werden.

```
$datenbank->sql_insert
(„belege“,“nr,datum,betrag“,“'21','20.10.2007','100.99');
if ($datenbank->ERRORCODE != CL_DATABASE_OK)
  echo $datenbank->error_text();
```

sql_lines

function sql_lines ([$qu])

$qu	optionale Angabe des Antwortzeigers

Die Funktion liefert die Anzahl der Zeilen zur letzten oder optional der durch den Antwortzeiger definierten SQL-Abfrage zurück.

```
$qu = $datenbank->sql_befehl („SELECT * FROM belege“);
echo „Die Anfrage hat „;
echo $datenbank->sql_lines());
echo „ Ergebnisse gebracht !“;
```

Der Rückgabewert ist eine Zahl >= 0

sql_line

function sql_line ($nr,[$qu])

$nr	Zeilen-Nr. die geholt werden soll
$qu	optionale Angabe des Antwortzeigers

Die Funktion gibt die gewählte Antwortzeile zur SQL-Abfrage zurück. $nr wird geprüft und nur gültige Zeilen ausgewählt.

- Wenn $nr < 0 ist wird die erste Spalte ausgeben.
- Wenn $nr > Anzahl der Spalten der gewählten SQL-Abrage, dann wird die letzte Spalte ausgegeben.

```
$qu = $datenbank->sql_befehl („SELECT * FROM belege“);
$data = $datenbank->sql_line (0,$qu);
echo „$data->nr  $data->datum $data->text <BR>“;
```

Die Rückgabe ist ein Array mit der gewünschten Zeile.

sql_update

function sql_update ($tabelle, $set, [$condition])

$tabelle	In welche Datenbank-Tabelle soll eingefügt werden ?
$set	Update-Einträge
$condition	Optionale WHERE - Angabe

Mit dieser Funktion können ein oder mehrere Felder mit geänderten Werten überschrieben werden. Die Funktion prüft nicht ob die Änderung tatsächlich durchgeführt werden kann. Konnte die Änderung nicht erfolgen ist der Errorcode auf CL_DATABASE_ERROR_SQL gesetzt.

```
$datenbank->sql_update („belege“,“nr='10'“,“nr='99'“);
if ($datenbank->ERRORCODE != CL_DATABASE_OK)
  echo $datenbank->error_text();
```

Die Funktion gibt kein Resultat zurück. Ob ein Fehler aufgetreten ist kann über den Errorcode abgefragt werden.

formular.inc (class cl_formular)

cl_formular ([$ziel], [$target])

Diese Klasse ist für die Verwaltung und Auswertung der Formulare zuständig. Alle Eingaben sollen durch diese Klasse abgebildet werden.

Zur Initialiserung können verschiedene optionale Parameter übergeben werden.

(1) Ohne Parameter wird die aktuelle Datei und der aktuelle Frame als CGI-Script für die Auswertung verwendet.

```
FORMULAR = new cl_formular ();
```

(2) Wenn nur das Ziel angegeben werden soll, aber der aktuelle Frame beibehalten wird, kann die Initialisierung mit Angabe des CGI-Scripts für die Auswerung erfolgen

```
FORMULAR = new cl_formular („auswerten.php“);
```

(3) Es können aber auch Ziel und Frame angegeben werden. Die Initialisierung erfolgt dann mit Angabe des CGI-Scripts für die Auswertung und als zweitem Parameter der für die Auswertung zu verwendenden Frame.

```
FORMULAR = new cl_formular („auswerten.php“,“_blank“);
```

(4) Es kann aber auch nur ein Frame übergeben und als CGI-Script die aktuelle Datei benutzt werden. Die Initialisierung erfolgt dann mit NULL als erstem Parameter und dem Ziel-Frame als zweitem Parameter.

```
FORMULAR = new cl_formular (NULL,“_blank“);
```

uverein
formular.inc (class cl_formular)

checkbox

function checkbox ($name, $wert, $anzeige, [$checked])

$name	Name der Checkbox
$wert	Rückgabewert
$anzeige	Text für die Beschriftung
$checked	optionaler Wert. Wenn etwas übergeben wird ist die Box ausgewaehlt – ansonsten wird der Wert aus der letzten Übermittlung des Formulars genommen

Die Funktion stellt eine Checkbox mit Beschriftung zur Verfügung. Die Auswertung erfolgt nach Formularübermittlung über die Funktionen returnwert bzw. returnzahl.

```
$FORMULAR->checkbox („box“,“OK“,“bitte anklicken“);
// gibt
//      ☐ bitte anklicken
// aus
```

Die Funktion hat keinen Rückgabewert.

uverein
formular.inc (class cl_formular)

eingabe

function eingabe ($name, [$len], [$vorgabe])

$name	Name des Formular-Feldes
$len	optionaler Parameter, der die Feldlänge definiert Voreinstellung = 80
$vorgabe	optionaler Parameter, der eine Vorgabe für das Formular-feld enthält. Voreinstellung = Inhalt lt. letztem Sendevorgang

Diese Funktion stellt ein Eingabefeld zur Verfügung. Die Breite des Feldes wird durch $len festgelegt. Falls $vorgabe mit übergeben wird ist dies die Vorbelegung für das Eingabefeld. Falls $vorgabe nicht übergeben wird ist die Vorbelegung der Eintrag beim letzten Sendevorgang oder falls noch kein Sendevorgang stattgefunden hat ein leerer String.

```
$FORMULAR->eingabe („eingabefeld“);
```

Die Funktion hat keinen Rückgabewert.

uverein
formular.inc (class cl_formular)

feld

function feld ($name, [$breite], [$hoehe], [$vorgabe])

$name	Name des Formular-Feldes
$breite	optionaler Parameter, der die Feldbreite definiert Voreinstellung = 80 Zeichen
$hoehe	optionaler Parameter, der die Feldhöhe definiert Voreinstellung = 10 Zeilen
$vorgabe	optionaler Parameter, der eine Vorgabe für das Formular-feld enthält. Voreinstellung = Inhalt lt. letztem Sendevorgang

Diese Funktion stellt ein Eingabefeld für mehrzeiligen Text zur Verfügung. Die Breite und Höhe des Feldes werden durch $breite und $hoehe festgelegt. Falls $vorgabe mit übergeben wird ist dies die Vorbelegung für das Feld. Falls $vorgabe nicht übergeben wird ist die Vorbelegung der Eintrag beim letzten Sendevorgang oder falls noch kein Sendevorgang stattgefunden hat ein leerer String.

```
$FORMULAR->feld („eingabefeld“);
```

Die Funktion hat keinen Rückgabewert.

grafikbutton

function grafikbutton ($value, $pic)

$value	Wert, der zurück gegeben werden soll
$pic	Pfad zum anzuzeigenden Bild

Die Funktion stellt einen Sendebutton bereit, der statt Text eine Grafik anzeigt. Wenn der Button angeklickt wird erfolgt die Rückgabe des als $value übergebenen Wertes. Die Rückgabe erfolgt über die Funktion sende_ergebnis.

```
$FORMULAR->grafikbutton („OK“,“../icons/ok.png“);
```

Die Funktion hat keinen Rückgabewert.

hide

function hide ($name, [$wert])

Mit dieser Funktion kann eine Variable unsichtbar mit übermittelt werden. Wenn kein Wert angegeben wird, liest die Funktion den Inhalt der Variablen aus den übermittelten Daten aus.

```
$FORMULAR->hide („geschuetzt“,“Ja“);
```

Die Funktion hat keine Rückgabeparameter. Auch im Fehlerfall erfolgt keine Rückgabe.

uverein
formular.inc (class cl_formular)

inhalt

function inhalt ($text)

$text	Text, der im Formular ausgeben werden soll

Diese Funktion fügt Text und ggf. HTML-Tags in das Formular ein. Dadurch wird der Vordruck insgesamt übersichtlicher.

```
$FORMULAR->inhalt („<HR>Bemerkungsfeld:<BR>“);
$FORMULAR->feld   („bemerkungen“);
$FORMULAR->inhalt („<BR><HR>“);
```

Diese Funktion hat keinen Rückgabewert und wird beim Sendevorgang nicht berücksichtigt.

uverein
formular.inc (class cl_formular)

numeingabe2anzeige

function numeingabe2anzeige ($zahl, $nachkomma)

$zahl	Zahl, die für die Anzeige optimiert werden soll
$nachkomma	darzustellende Nachkommastellen

Diese Funktion wandelt die übergebene Zahl für die Darstellung am Bildschirm um. Die Zahl kann bereits formatiert oder unformatiert übergeben werden. Das Ergebnis ist jeweils gleich.

```
echo $FORMULAR->(„10.9“,2); // gibt 10,90 aus
echo $FORMULAR->(„10,9“,2); // gibt 10,90 aus
```

Die Funktion liefert als Rückgabe die formatierte Zahl.

returnglobal

function returnglobal ($name)

$name	Name des Formular-Feldes

Mit dieser Funktion wird das Formular nach der Übermittlung ausgelesen – falls der Wert nicht im Formular übergeben wurde wird zusätzlich geprüft ob das Feld über die Browser-Funktion direkt übergeben wurde und ggf. eingelesen.

```
$FORMULAR = new cl_formular ();
$eingabe = FORMULAR->returnglobal („eingabe“);
```

Die Funktion gibt den Inhalt des Formularfeldes bzw. alternativ einen direkt übergebenen Wert zurück. Ist das Feld leer kommt als Ergebnis ein leerer String zurück.

uverein
formular.inc (class cl_formular)

returnwert

function returnwert ($name)

$name	Name des Formular-Feldes

Mit dieser Funktion wird das Formular nach der Übermittlung ausgelesen.

```
$FORMULAR = new cl_formular ();
$eingabe = FORMULAR->returnwert („eingabe“);
```

Die Funktion gibt den Inhalt des Formularfeldes zurück. Ist das Feld leer oder wurde das Formular noch nicht abgeschickt kommt als Ergebnis ein leerer String zurück.

returnzahl

function returnzahl ($name)

$name	Name des Formular-Feldes

Mit dieser Funktion wird das Formular nach der Übermittlung ausgelesen.

```
$FORMULAR = new cl_formular ();
$eingabe = FORMULAR->returnzahl („eingabe“);
```

Die Funktion gibt den Inhalt des Formularfeldes zurück. Ist das Feld leer oder wurde das Formular noch nicht abgeschickt kommt als Ergebnis die Zahl „0“ zurückgegeben.

schliessen

function schliessen ()

Diese Funktion beendet das Formular und liefert das Formular als Rückgabe.

```
$FORMULAR = new cl_formular();
$FORMULAR->text („<HR>Was tun ?</BR>“);
$FORMULAR->senden („Nichts“);
$FORMULAR->text („  „);
$FORMULAR->senden („Etwas“);
echo $FORMULAR->schliessen();
```

Die Funktion gibt das komplette Formular zurück, das weiterverarbeitet oder ausgegeben werden kann.

uverein
formular.inc (class cl_formular)

selection

function selection ($name, $sel, [$selected], [$size])

$name	Name des Feldes
$sel	Array mit den Selektionsfeldern
$selected	Optionaler Parameter, der den Wert des selektierten Feldes enthält Voreinstellung = Wert des letzten Sendevorganges, wenn noch keine Sendevorgang stattgefunden hat 1. Element
$size	Anzahl der anzuzeigenden Selektionen Voreinstellung = 1

Diese Funktion stellt ein Selektionsfeld zur Verfügung. Die Daten für die einzelnen Selektionszeilen werden als Array übergeben. Der Schlüssel des Array's wird hierbei als Rückgabewert und der Array-Inhalt für die Anzeige verwandt.

```
$SEL = array („OK“ => „Alles OK“, „KAPUT“ => „zerstört“);
$FORMULAR->selection („was“,$SEL);
```

Die Funktion hat keinen Rückgabewert. Das Ergebnis des Sendevorganges kann über die Funktion returnwert oder returnzahl ausgelesen werden.

uverein
formular.inc (class cl_formular)

sende_ergebnis

function sende_ergebnis ()

Diese Funktion gibt das Ergebnis der letzten Übermittlung des Formulars zurück. Welcher Wert zurück gegeben werden kann wird in den Funktionen

- senden

oder

- grafikbutton

definiert.

```
$Auswahl = $FORMULAR->sende_ergebnis();
```

Wenn kein Button gedrückt wurde ist das Rückgabe-Ergebnis nicht definiert und es wird ein leerer String zurück gegeben.

senden

function senden ($name)

$name	Text der als Button angezeigt wird

Diese Funktion stellt einen Sende-Button bereit, der mit dem übergebenen Text beschriftet wird. Der Text ist gleichzeitig der Rückgabewert, wenn der Button angeklickt wird. Die Rückgabe erfolgt über die Funktion sende_ergebnis.

```
$FORMULAR->senden („OK“);
$FORMULAR->senden („Abbruch“);
```

Die Funktion hat keinen Rückgabewert.

uverein
formular.inc (class cl_formular)

test

function test ()

Diese Funktion gibt alle Formularfelder mit Vorgabewert sowie alle durch den letzten Sendevorgang übermittelten Felder aus.

Die Funktion ist nur für den Formular-Test bestimmt und sollte im endgültigen Dokument auskommentiert oder entfernt werden.

```
echo $FORMULAR->test();
```

uverein
formular.inc (class cl_formular)

Globale Variablen

$tausend

Diese Variable nimmt den Trenner für die Darstellung der Tausender-Stellen im numerischen Format auf.

Voreinstellung: . (Punkt)

$trenner

Diese Variable nimmt den Trenner zwischen Ganzzahl und Nachkommastellen im numerischen Format auf.

Voreinstellung: , (Komma)

uvereinpdf.inc (class cl_uvereinpdf)

Die Klasse ermöglicht es PDF-Dokumente zu erstellen.

Grundlage ist die Bibliothek FPDF:

- Projekt-Seite: http://www.fpdf.org/
- Debian-Paket: php-fpdf

Die Initialiserung erfolgt durch die Klassendefinition

```
$PDF = new cl_uvereinpdf („Kopfzeile“,“Fusszeile“);
```

Die Ausgabe des Dokuments erfolgt durch Aufruf von

```
$PDF->anzeige();
```

uverein
uvereinpdf.inc (class cl_uvereinpdf)

table_init

function table_init ($Spalten,
$Font, $Fontsize,
$Rahmen,
[$Spaltenbreiten], [$Spaltenausrichtungen])

$Spalten	Anzahl der Tabellen-Spalten
$Font	Schriftart, z.B. Arial
$Fontsize	Schriftgrösse in Punkt
$Rahmen	Rahmen anzeigen (0 = nein)
optionale Werte:	
$Spaltenbreiten	Array mit den relativen Spaltenbreiten Voreinstellung: gleich große Spalten
$Spaltenausrichtung	Array mit den Spaltenausrichtungen R rechtsbündig L linksbündig = Voreinstellung C zentriert

Die Funktion initialisiert eine Tabelle und muss daher immer dann aufgerufen werden, wenn eine neue Tabelle beginnen soll.

```
$PDF = new cl_uvereinpdf („Testseite“,“Fusszeile“);
$Breite = array (1,2,1);
$PDF->table_init (3,“Courier“,10,1,$Breite,array(„L“,“R“,“C“));
$PDF->table_line (array (
  0 => "Beschriftung Spalte 1",
  1 => "Beschriftung Spalte 2 ist laenger als die anderen",
  2 => "Spalte 3"));
$PDF->anzeige ();
```

Die Funktion hat keinen Rückgabewert.

table_head

function table_head ($ueberschrift)

$ueberschrift Array mit den Spaltenüberschriften

Die Funktion gibt den Inhalt des übergebenen Arrays in Fettschrift zentriert aus.

```
$ueberschriften =
  array (1 => „Spalte 1“, 2 => „Spalte 2“, 3 => „Spalte 3“);
$PDF->table_head ($ueberschriften);
```

Die Funktion hat keinen Rückgabewert.

table_line

function table_line ($inhalt)

$inhalt Array mit einer Tabellenzeile

Die Funktion gibt den Inhalt des übergebenen Arrays aus.

```
$inhalt = array (
  1 => „Inhalt der Spalte 1“,
  2 => „Inhalt der Spalte 2“,
  3 => „Inhalt der Spalte 3“);
$PDF->table_line ($inhalt);
```

Die Funktion hat keinen Rückgabewert.

uverein
uvereinpdf.inc (class cl_uvereinpdf)

textzeile

function textzeile ($Text, [$Schrift], [$Schriftgroesse], [$Ausrichtung])

$Text	Text, der ausgegeben werden soll
$Schrift	In welcher Schrift soll der Text ausgeben werden. Voreinstellung: Arial
$Schriftgroesse	Schriftgroesse in Punkt Voreinstellung: 10
$Ausrichtung	Textausrichtung Voreinstellung: linksbündig Mögliche Werte: *R rechtsbündig* *L linksbündig = Voreinstellung* *C zentriert*

Die Funktion gibt eine Text-Zeile aus.

```
$pdf->textzeile ("Dies ist eine Zeile","Courier",15,'C');
```

Datenbank-Definition

Die Datenbank besteht aus folgenden Tabellen:

uverein
Datenbank-Definition

beitrag

In diese Tabelle werden die Beitragsbuchungen eingetragen.

Name	Daten-Typ	Informationen
bu_nr	INTEGER	Interne Buchungs-Nr. für die Beitragsbuchungen, wird automatisch vergeben.
v_name	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
m_nr	INTEGER	Mitglieds-Nr., referenziert personen.nr
jahr	VARCHAR 10	Buchungsjahr
f_konto	NUMERIC 10,0	Finanzkonto
datum	DATE	Buchungsdatum
soll	NUMERIC 30,2	Soll-Buchung. Diese Buchung wird nur für die Mitgliederverwaltung benötigt
haben	NUMERIC 30,2	Haben-Buchung. Diese Buchung fließt automatisch in die Finanzbuchhaltung ein.
bu_text	TEXT	Freier Erläuterungstext für den Eintrag.

uverein
Datenbank-Definition

beitrags_art

Diese Tabelle enthält die Beitragsarten. Z.B. Ehrenmitglied, passives Mitglied usw.

Name	Daten-Typ	Informationen
nr	INTEGER	Nummer für die Beitragsart. Diese Nr. wird vereinsübergreifend nur einmal vergeben.
verein	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
art	TEXT	Beschreibung der Beitragsart. Diese Beschreibung wird in der Mitgliederverwaltung angezeigt.
jahresbeitrag	NUMERIC 30,2	Jahresbeitrag

beitrags_konto

Diese Tabelle enthält Angaben zu Beitragsfälligkeiten der Mitglieder.

Name	Daten-Typ	Informationen
v_name	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
nr	SERIAL	Fortlaufende Nr des Eintrages – gilt nur für diese Tabelle
m_nr	INTEGER	Mitglieds-Nummer – referenziert mit der Tabelle „personen“
faellig_ab	DATE	1. Fälligkeitsdatum
art	INTEGER	Beitragsart – referenziert mit der Tabelle „beitrags_art“
zahlungsweise	INTEGER	Zahlungsweise als Zeiger auf die Tabelle „zahlungsweise“

uverein
Datenbank-Definition

bilanz

Diese Tabelle enthält Informationen zur Gewinn- und Verlustrechnung

Name	Daten-Typ	Informationen
Verein	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
jahr	VARCHAR 10	Buchungsjahr
summe	NUMERIC 30,0	Endsumme der Gewinn- und Verlustrechung = Gesamtübertrag zum nächsten Jahr
waehrung	VARCHAR 10	Währung, in der das betroffene Jahr gebucht ist (z.B. EUR)
schlusstxt	TEXT	Schlusstext zur Gewinn- und Verlustrechnung. Z.B. „Erstellt durch den Schatzmeister des Vereins“
prueftxt	TEXT	Informationstext zur Prüfung der Gewinn- und Verlustrechnung. Z.B. „Geprüft durch die Finanzkommision des Vereins“
datum	DATE	Datum, an dem die Buchungen geprüft wurden

uverein
Datenbank-Definition

briefe

Diese Tabelle enthält Brieftexte für verschiedene Anlässe.

Name	Daten-Typ	Informationen
verein	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
nr	INTEGER	Fortlaufende Nr. des Eintrags – gilt nur für diese Tabelle
info	VARCHAR 20	Kurzbeschreibung. Dieser Text wird angezeigt wenn der Brief einer Aktion zugeordnet werden soll. Aktionen sind in der Tabelle „virtual_konto“ erläutert.
inhalt	TEXT	Brieftext incl. Platzhaltern für Datenbankauszüge

Platzhalter für Datenbankauszüge

Platzhalter	Was steht im Brief
%%BEITRAG%%	Stelle eine Tabelle in den Brief mit allen Beitragsbuchungen incl. Buchungstext zum Mitglied.
%%TERMIN28%%	Wird im Brieftext ersetzt durch das aktuelle Datum + 28 Tage

bu_belege

Diese Tabelle enthält die Buchungsbelege für alle Vereine.

Name	Daten-Typ	Informationen
b_verein	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
idx	SERIAL	Interne Nr., die nur für diese Tabelle gültig ist.
b_nr	NUMERIC 10,0	Buchungs-Nr. ACHTUNG: Die Nr. 1 ist für den Übertrag reserviert und wird beim Jahreswechsel ggf. überschrieben.
b_jahr	VARCHAR 10	Buchungsjahr
b_datum	DATE	Datum des Buchungsbeleges
b_konto	NUMERIC 10,0	Buchungskonto – referenziert mit Tabelle „bu_konten“
f_konto	NUMERIC 10,0	Finanzkonto – referenziert mit Tabelle „bu_konten“
b_betrag	NUMERIC 20,2	Buchungsbetrag – brutto
b_steuer	NUMERIC 20,2	Im Buchungsbetrag enthaltene Vorsteuer
b_text	TEXT	Buchungstext
b_kst	NUMERIC 10,0	Kostenstelle – referenziert mit Tabelle „bu_kst“

uverein
Datenbank-Definition

bu_klassen

Diese Tabelle enthält die einzelnen Bereiche, in die die Konten aufgeteilt werden.

Name	Daten-Typ	Informationen
Verein	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
kl_nr	NUMERIC 10,0	Nummer des Bereichs – wird für die referenzierenden Tabelle benötigt
kl_name	TEXT	Beschreibung des Bereichs. Z.B. „Ideeler Bereich“, „Zweckbetrieb“ usw.

uverein
Datenbank-Definition

bu_konten

Diese Tabelle enthält die Buchungskonten

Name	Daten-Typ	Informationen
Verein	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
kto_nr	NUMERIC 10,0	Konto-Nr.
kto_name	TEXT	Beschreibung des Kontos
kto_klasse	NUMERIC 10,0	Bereich, zu dem das Konto gehört – referenziert mit Tabelle „bu_klassen“
kto_ziel	NUMERIC 10,0	Auf welches Konto wird dieses Konto abgeschlossen? Wenn kto_ziel und kto_nr identisch sind erfolgt der Abschluss in der Gewinn- und Verlustrechung, ansonsten handelt es sich um ein Unterkonto des Konto mit der Nr. „kto_ziel“
kto_stand	NUMERIC 25,2	Kontostand ACHTUNG: Diese Spalte wird evlt. aus der Tabelle entfernt.

uverein
Datenbank-Definition

bu_kst

Diese Tabelle enthält die Kostenstellen

Name	Daten-Typ	Informationen
verein	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
kst	NUMERIC 10,0	Nummer der Kostenstelle
kst_name	TEXT	Beschreibung der Kostenstelle

key_anrede

Diese Tabelle enthält die Anredeschlüssel für die Tabelle „personen“

Name	Daten-Typ	Informationen
schluessel	SMALLINT	Schlüssel für die Anrede, auf den referenziert wird
brief	TEXT	Anrede für den Briefkopf. Z.B. ● Herr ● Frau
inhalt	TEXT	Anrede im Brief. Z.B. ● Sehr geehrter Herr ● Sehr geehrte Frau

uverein
Datenbank-Definition

key_banken

Diese Tabelle enthält Bankleitzahlen und Banknamen. Diese Angaben werden u.a. in den Tabellen „*_einzug“ und „vereine“ benötigt.

Name	Daten-Typ	Informationen
blz	NUMERIC 8,0	Bankleitzahl
bank	TEXT	Name der Bank

uverein
Datenbank-Definition

key_zahlungsarten

Diese Tabelle enthält die Zahlungsarten für alle Vereine.

Name	Daten-Typ	Informationen
nr	SMALLINT	Schlüssel auf den von anderen Tabellen referenziert wird.
bezeichnung	TEXT	Beschreibung. Z.B. • per Rechnung • Einzugsermächtigung

uverein
Datenbank-Definition

muster = Kurzname des Vereins

Diese Tabelle existiert für jeden Verein und enthält die Informationen über die dem Verein zugeordneten Mitglieder

Name	Daten-Typ	Informationen
nr	BIGINT	Mitglieds-Nr. - referenziert mit der Tabelle „personen“
eintritt	DATE	Eintrittsdatum
ende	DATE	Vereinsaustritt
kommentar	TEXT	Freier Kommentar zum Mitglied.
zahlungsart	SMALLINT	Zahlungsart des Mitglieds – referenziert mit Tabelle „key_zahlungsarten“
aktiv	BOOLEAN	Differenzierung von ausgeschiedenen und aktiven Mitgliedern (ausgeschieden = false) Die Spalte wird durch einen Cronjob automatisch einmal pro Stunde aktualisiert.

uverein
Datenbank-Definition

muster_einzug = Kurzname des Vereins + _einzug

Diese Tabelle verwaltet die Einzugsermächtigungen und existiert für jeden Verein.

Name	Daten-Typ	Informationen
nr	BIGINT	Mitglieds-Nr. - referenziert mit der Tabelle „personen“
blz	NUMERIC 8,0	Bankleitzahl – referenziert mit der Tabelle „key_banken“
konto	NUMERIC 10,0	Konto-Nr. von dem abgebucht werden soll

uverein
Datenbank-Definition

opt_tabledef

Definitionstabelle für weitere Felder in der Mitglieder-Verwaltung

Name	Daten-Typ	Informationen
nr	SERIAL	Eindeutiges Kennzeichen als Querverweismöglichkeit für die Tabelle „opt_tabledata“
verein	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
bezeichnung	VARCHAR 50	Kurzbezeichnung des optionalen Feldes - Pflichtfeld
beschreibung	TEXT	Ausführliche Beschreibung des optionalen Feldes – kein Pflichtfeld

opt_tabledata

Tabelle mit den Inhalten der optionalen Felder

Name	Daten-Typ	Informationen
opt_table	BIGINT	Referenz auf „opt_tabledef.nr“
person	BIGINT	Personen-Referenz auf „personen.nr“
inhalt	TEXT	Inhalt des optionalen Feldes

uverein
Datenbank-Definition

personen

Diese Tabelle enthält alle vereinsübergreifenden Angaben zu den einzelnen Mitgliedern.

Name	Daten-Typ	Informationen
nr	SERIAL	Personen-Nr. - auf diesen Wert referenzieren die vereinsinternen Tabellen.
anrede	SMALLINT	Anredeschlüssel – referenziert mit Tabelle „key_anrede“
titel	TEXT	Akademischer Titel
name	TEXT	Nachname
vorname	TEXT	Vorname(n)
geboren	DATE	Geburts-Datum. Wenn kein Geburtsdatum bekannt ist wird dieses Feld mit „01.01.1800“ gefüllt
strasse	TEXT	Strasse
hausnr	TEXT	Hausnummer
land	TEXT	Land – z.B. „Deutschland“
plz	NUMERIC 5,0	Postleitzahl
ort	TEXT	Wohnort
telefon	TEXT	Telefon-Nr.
email	TEXT	EMail-Adresse

uverein
Datenbank-Definition

tax

Diese Tabelle enthält Daten zur Steuererklärung.

Name	Daten-Typ	Informationen
verein	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
jahr	NUMERIC 4,0	Buchungsjahr
datum	DATE	Datum, an dem die Steuererklärung oder Voranmeldung ans Finanzamt gesandt wurde
art	INTEGER	Steuerart – referenziert Tabelle „tax_base.nr“
umsatz	NUMERIC 20,2	Netto-Umsatz
steuer	NUMERIC 20,2	Steuer-Betrag
bu_nr	NUMERIC 10,0	Buchungs-Nr. unter der die Steuerabrechung in der Buchhaltung eingetragen wurde. Referenziert Tabelle „bu_belege.b_nr“

tax_base

Diese Tabelle enthält Informationen über die Steuer-Arten

Name	Daten-Typ	Informationen
verein	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
nr	INTEGER	Nr. der Steuerart – auf dieses Feld referenzieren andere Tabellen
name	TEXT	Bezeichnung der Steuerart
steuersatz	NUMERIC 8,4	Steuersatz
rundung	INTEGER	Auf wieviele Stellen hinter dem Komma wird die Steuer gerundet?
Kurztext	TEXT	Hinweise – optionale Angabe

uverein
Datenbank-Definition

tax_konten

Diese Tabelle enthält die Zuordnung der Buchungskonten zu den einzelnen Steuerarten.

Name	Daten-Typ	Informationen
verein	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
steuersatz	INTEGER	Steuerart – referenziert Tabelle „tax_base“
kto	NUMERIC 10,0	Konto-Nr. die der Steuerart zugeordnet wird. Referenziert Tabelle „bu_konten“

uverein
Datenbank-Definition

vereine

Diese Tabelle enthält Angaben zu den vorhanden Vereinen.

Name	Daten-Typ	Informationen
kurzname	VARCHAR 20	Verein-Kurzname, wird von fast allen anderen Tabellen referenziert.
name	TEXT	Name des Vereins
strasse	TEXT	Strasse und Hausnummer
ort	TEXT	PLZ und Ort, ggf. auch Land
blz	VARCHAR 10	Bankleitzahl des Beitragskontos
bank	TEXT	Name der Bank für das Beitragskonto
kontonr	NUMERIC 15,0	Konto-Nr. des Beitragskontos
email	TEXT	Email-Adresse des Vereins
version	TEXT	Versions-Nr. unter der der Verein angelegt wurde oder die Datenbank komplett geändert wurde. Dieses Feld wird bei Anpassungen benötigt und ggf. nach einer Anpassung mit der neuen Versions-Nr. überschrieben.

uverein
Datenbank-Definition

virtual_konto

Diese Tabelle enthält Informationen über vereinsintern benötigte Werte und Zuordnungen. Diese Tabelle wird von verschiedenen Programmteilen benötigt um Werte korrekt zuordnen zu können.

Name	Daten-Typ	Informationen
verein	VARCHAR 20	Verein-Kurzname, referenziert vereine.kurzname
art	VARCHAR 20	Schlüssel
konto	NUMERIC 10,0	Siehe unten
fkonto	NUMERIC 10,0	Siehe unten
kst	NUMERIC 10,0	Siehe unten

Welche Daten stehen in den Feldern?

Schlüssel	Feld	Welche Information steht hier?
BEITRAG	*Zuordnung der Beitragszahlungen aus der Mitgliederverwaltung zur Buchhaltung*	
	konto	Beitragskonto - referenziert Tabelle „bu_konten“
	fkonto	Finanzkonto, das in der Mitgliederverwaltung als Voreinstellung eingetragen wird - referenziert Tabelle „bu_konten“
	kst	Kostenstelle unter der die Beitragsbuchungen eingetragen werden – referenziert Tabelle „bu_kst“
BEITRAG_1 BEITRAG_2 BEITRAG_3	*Beitragserinnerungen 1 - 3*	
	konto	Nummer des zu verwendenden Briefes. Referenziert mit Tabelle „briefe“
	fkonto	Höhe der anfallenden Mahnkosten
	kst	-
BEITRAG_LEV	*Brief nach einer Lastschrift-Retoure*	
	konto	Nummer des zu verwendenden Briefes. Referenziert mit Tabelle „briefe“
	fkonto	Höhe der anfallenden Mahnkosten
	kst	-
FINANZKONTO	*Welcher Kontenart gehören die Finanzkonten an*	
	konto	Kontenart – referenziert Tabelle „bu_klassen“
	fkonto	-
	kst	-
SCHWEBEKONTO	*Welches Konto wird als Schwebekonto verwandt?*	

uverein
Datenbank-Definition

Schlüssel	Feld	Welche Information steht hier?
	konto	Konto-Nr. - referenziert Tabelle „bu_konten“
	fkonto	-
	kst	-
UEBERTRAG	*Wie wird der Übertrag zugeordnet?*	
	konto	Kontenart – referenziert Tabelle „bu_klassen“
	fkonto	Konto-Nr. - referenziert Tabelle „bu_konten“
	kst	Kostenstelle – referenziert Tabelle „bu_kst“
VORSTEUER	*Wie wird die in bu_belege.b_steuer eingetragene Vorsteuer in die Buchhaltung eingetragen?*	
	konto	Buchungskonto - referenziert Tabelle „bu_konten“
	fkonto	Finanzkonto - referenziert Tabelle „bu_konten“
	kst	Kostenstelle – referenziert Tabelle „bu_kst“

zahlungsweise

Diese Tabelle enthält Informationen über die definierten Zahlungsweisen. Auf diese Tabelle referenzieren die für die Beitragszahlung zuständigen Tabellen.

Name	Daten-Typ	Informationen
nr	INTEGER	Schlüssel, auf den von anderen Tabellen referenziert wird.
art	TEXT	Bezeichnung im Klartext
einmalig	BOOLEAN	Einmalzahlung (ja / nein)
teiler	INTEGER	Teiler für den Jahresbeitrag. Z.B. 12 für monatlich

uverein

Anlagen

Ab hier folgen Original-Dateien aus dem Projekt:

php/klassen/database.inc

```
<?php

// ----------------------------------------------------------
// Klasse definieren um den Zugriff auf die Datenbank vom
// Quell-Code abzukapseln
//
// P O S T Q R E S
// ----------------------------------------------------------

// ----------------------------------------------------------
// Konstanten
// ----------------------------------------------------------
define ('CL_DATABASE_OK',                0);
define ('CL_DATABASE_ERROR_CANTOPEN',    1);
define ('CL_DATABASE_ERROR_SQL',         2);
define ('CL_DATABASE_ERROR_UNKNOWN',     3);
define ('CL_DATABASE_MAX_ERROR',         4);

// ----------------------------------------------------------
// Klassendefinition
// ----------------------------------------------------------
class cl_database
{
  var $DATENBANK;             // Datenbank-Resource
  var $DBNAME;                // Name der Datenbank
  var $ERRORCODE;             // Fehlercode
  var $ERRORTEXT;             // Array mit Fehlertext
  var $qu;                    // Zeiger auf Query
  var $TESTMODE;              // Testmode ein- oder aus

  // ----------------------------------------------------------
  // Fehlermeldungen
  // ----------------------------------------------------------

  function error_name ($key, $inhalt) // Textmeldung eintragen
  {
    $this->ERRORTEXT[$key] = $inhalt;
  }

  // Fehlermeldung ausgeben
  function error_text ()
  {
    $x = "";
    $i = ($this->ERRORCODE > CL_DATABASE_OK) ? $this->ERRORCODE : CL_DATABASE_OK;
    $i = ($i <= CL_DATABASE_MAX_ERROR) ? $i : CL_DATABASE_MAX_ERROR;
    return ($this->ERRORTEXT[$i]);
  }

  // ----------------------------------------------------------
  // Ausgabe fuer Testmode
  // ----------------------------------------------------------
  function testmode ($info)
  {
    if ($this->TESTMODE)
    {
      echo "<HR>$info".
           "<BR>RETURN: ".$this->error_text().
           "<BR>LINES:  ".@pg_num_rows($this->qu).
           "<BR>";
    }
  }

  // ----------------------------------------------------------
  // SQL-Abfrage
  // ----------------------------------------------------------

  // Abfrage
```

```
function sql_befehl ($befehl)
{
  $this->ERRORCODE = CL_DATABASE_OK;
  $this->qu = @pg_query ($this->DATENBANK, "$befehl");
  if (!$this->qu) $this->ERRORCODE = CL_DATABASE_ERROR_SQL;
  $this->testmode ($befehl);
  return ($this->qu);
}

// Anzahl der Zeilen zur letzten SQL-Abfrage
function sql_lines ($qu = NULL)
{
  $query = ($qu == NULL) ? $this->qu : $qu;
  if ($query) $x = @pg_num_rows ($query);
  else        $x = 0;
  return ($x);
}

// Anzahl der Spalten zur letzten SQL-Abfrage
function sql_fields ($qu = NULL)
{
  $query = ($qu == NULL) ? $this->qu : $qu;
  if ($query) $x = @pg_num_fields ($query);
  else        $x = 0;
  return ($x);
}

// Spaltenbeschriftung auslesen und als Array zurück geben
function fieldnames ($qu = NULL)
{
  $query = ($qu == NULL) ? $this->qu : $qu;
  $max = $this->sql_fields($query);
  if ($max > 0)
  {
    $i = 0;
    while ($i < $max)
    {
      $ARR[$i] = @pg_field_name($query,$i);
      $i++;
    }
  }
  else $ARR[0] = "";
  return ($ARR);
}

// Zeile auslesen
function sql_line ($nr,$qu = NULL)
{
  $query = ($qu == NULL) ? $this->qu : $qu;
  $nr = ($nr < 0) ? 0 : $nr;
  $nr = ($nr > $this->sql_lines($query))
      ? $this->sql_lines($query) - 1
      : $nr;
  if ($query) $data = @pg_fetch_object ($query,$nr);
  else        $data = "";
  return ($data);
}

// Abfrageergebnis als Array zurueck geben
function result_array ($qu = NULL)
{
  $qu = ($qu == NULL) ? $this->qu : $qu;
  return (@pg_fetch_array($qu));
}

// Sonderabfrage
function sql_abfrage ($tabelle, $where, $order = "", $group = "")
{
  $WHERE = ($where == "") ? "" : "WHERE " . $where;
  $ORDER = ($order == "") ? "" : "ORDER BY " . $order;
  $GROUP = ($group == "") ? "" : "GROUP BY " . $group;
```

```
    $this->sql_befehl ("SELECT * FROM ".$tabelle." ".$WHERE." ".$ORDER." ".$GROUP);
  }

  // 1 bestimmter Wert soll ausgelesen werden
  function sql_1_value ($tabelle, $wert, $where)
  {
    $qu = $this->qu;
    $WHERE = ($where != "") ? " WHERE $where" : "";
    $this->sql_befehl ("SELECT $wert AS abfrage FROM $tabelle $WHERE");
    $data = $this->sql_line(0);
    $this->qu = $qu;
    return ($data->abfrage);
  }

  // Neuer Tabelleneintrag
  function sql_insert ($tabelle, $was, $inhalt)
  {
    $this->sql_befehl ("INSERT INTO " . $tabelle . "(" . $was . ")" . $inhalt);
  }

  // Daten updaten / aendern
  function sql_update ($tabelle, $set, $condition = "")
  {
    $condition = ($condition=="") ? "" : " WHERE " . $condition;
    $this->sql_befehl ("UPDATE " . $tabelle . " SET " . $set . $condition);
  }

  // ------------------------------------------------------
  // Datenbank oeffnen
  // ------------------------------------------------------
  function datenbank_oeffnen ()
  {
    $this->DATENBANK = @pg_connect ("dbname='".$this->DBNAME."'");
    $this->ERRORCODE = (!$this->DATENBANK)
                     ? CL_DATABASE_ERROR_CANTOPEN
                     : CL_DATABASE_OK;
    $this->testmode ("pg_connect(\"dbname='\".$this->DBNAME.\"'\"");
  }

  // ------------------------------------------------------
  // Klasse initialisieren
  // ------------------------------------------------------
  function cl_database ($name)
  {
    // Datenbankname merken
    $this->DBNAME = $name;

    // Fehlertexte eintragen
    $this->error_name (CL_DATABASE_OK,"OK");
    $this->error_name (CL_DATABASE_ERROR_CANTOPEN,
                       "Can't open Database ".$this->DBNAME);
    $this->error_name (CL_DATABASE_ERROR_SQL,
                       "Error while executing SQL-query");
    $this->error_name (CL_DATABASE_ERROR_UNKNOWN,"Unknown Error");

    // Postgres-Datenbank oeffnen
    $this->TESTMODE = false; // Testmode ausschalten
    $this->datenbank_oeffnen ();
  }
}

?>
```

php/klassen/formular.inc

```
<?php

// ----------------------------------------------------------------------
// Klassendefinition fuer Formulare
// ----------------------------------------------------------------------
class cl_formular
{
  var $Name;                    // Name des Formulars
  var $formular_ziel;
  var $formular_target;
  var $FORMULAR;
  var $INTERNE_VARIABLEN;
                                // Zahlen formatieren
  var $tausend;                 // Trennzeichen fuer Tausender
  var $trenner;                 // Trennzeichen fuer Nachkommastellen


  // ----------------------------------------------------------------------
  // Klasse initialisieren
  // $ziel      Formular-Ziel, wenn nichts angegeben ist dann PHP_SELF
  // $target    Frame in dem das Formular-Ergebnis dargestellt wird
  // ----------------------------------------------------------------------
  function cl_formular ($ziel = NULL, $target = NULL)
  {
    $this->formular_ziel   = ($ziel == NULL) ? $_SERVER['PHP_SELF'] : $ziel;
    $this->formular_target = ($target == NULL) ? "_self" : $target;
    $this->Name = time();
    $this->FORMULAR = "<FORM action=\"".$this->formular_ziel."\"".
                      " name=\"".$this->Name."\"".
                      " method=\"POST\"".
                      " enctype=\"multipart/form-data\"".
                      " target=\"".$this->formular_target."\">";
    $this->tausend = ".";
    $this->trenner = ",";
  }

  // ----------------------------------------------------------------------
  // Formular-Rueckgabewert aus den Superglobals auslesen
  // $name    Name der Variablen
  // ----------------------------------------------------------------------
  function returnwert ($name)
  {
    $checkup = explode ("[",$name);     // Soll ein Array ausgelesen werden ?
    if (count ($checkup) > 1)           // Ja - Array auslesen
    {
      $v_name = $checkup[0];
      $checkup = explode ("]",$checkup[1]);
      $v_element = $checkup[0];
      $v_wert = (empty($_POST[$v_name])) ? "" : $_POST[$v_name];
      $wert = (@empty($v_wert[$v_element])) ? "" : $v_wert[$v_element];
    }
    else                                // Nein - kein Array
    {
      $wert = (empty($_POST[$name])) ? "" : $_POST[$name];
    }
    return($wert);
  }

  // ----------------------------------------------------------------------
  // Formular-Rueckgabewert als Zahl aus den Superglobals auslesen
  // $name    Name der Variablen
  // ----------------------------------------------------------------------
  function returnzahl ($name)
  {
    $zahl = $this->returnwert($name);
    $zahl = ($zahl == "") ? 0 : $zahl;
    $zahl = str_replace("|",".",
```

```
          str_replace(",",".",
          str_replace(".","",trim($zahl))));
  $zahl = (is_numeric ($zahl)) ? $zahl : 0;
  return($zahl);
}


// ------------------------------------------------------------------
// Wert aus Formular bzw. alternativ aus REQUEST lesen
// $name     Name der Variablen
// ------------------------------------------------------------------
function returnglobal ($name)
{
  $wert = $this->returnwert($name);
  if ($wert == "")
  {
    $wert = (empty($_REQUEST[$name])) ? $wert : $_REQUEST[$name];
  }
  return ($wert);
}

// ------------------------------------------------------------------
// Variablenwerte in Formular einfuegen, die nicht angezeigt werden
// $name    Name der Variablen
// $wert    Inhalt der Variablen (wenn kein Wert uebergeben wird,
//          wird der Wert aus den Superglobals direkt ausgelesen)
// ------------------------------------------------------------------
function hide ($name, $wert = NULL)
{
  $wert = ($wert == NULL) ? $this->returnwert($name) : $wert;
  $this->FORMULAR .= "<INPUT type='hidden' name='$name' value='$wert'></input>";
  $this->INTERNE_VARIABLEN[$name] = $wert;
}

// ------------------------------------------------------------------
// Ergebnis des Sende-Buttons zurueck geben
// ------------------------------------------------------------------
function sende_ergebnis ()
{
  return($this->returnwert("senden"));
}

// ------------------------------------------------------------------
// Sende-Button ausgeben
// $name     Text, der angezeigt wird = gleichzeitig Ergebnis
// ------------------------------------------------------------------
function senden ($name)
{
  $this->FORMULAR .= "<INPUT type='submit' name='senden' value='$name'></input>";
}

// ------------------------------------------------------------------
// Grafischen Sende-Button ausgeben
// $value     Rueckgabewert, wenn Button gedrueckt wurde
// $pic       Grafik
// ------------------------------------------------------------------
function grafikbutton ($value, $pic)
{
  $this->FORMULAR .= "<INPUT type='image' src='$pic' alt='$pic' ".
                     "name='senden' value='$value' width='25'>";
}

// ------------------------------------------------------------------
// Eingabefeld einfuegen
// $name        Name der Variablen
// $len         Laenge des Feldes (nichts = 80)
// $vorgabe     Inhalt der Variablen (nichts = aus Superglobals holen)
// ------------------------------------------------------------------
function eingabe ($name, $len = NULL, $vorgabe = NULL)
{
  $len = ($len == NULL) ? 80 : $len;
```

```
    $maxlen = $len + 20;
    $vorgabe = ($vorgabe == NULL) ? $this->returnwert($name) : $vorgabe;
    $vorgabe = str_replace("\\\"", '"', $vorgabe);
    $this->FORMULAR .= "<INPUT type='text'".
                       " name='$name'".
                       " size='$len'".
                       " maxlength='$maxlen'".
                       " value='$vorgabe'>";
    $this->INTERNE_VARIABLEN[$name] = $vorgabe;
  }

  // ---------------------------------------------------------------
  // Textfeld einfuegen
  // $name        Name der Variablen
  // $breite      Breite Feldes (nichts = 80)
  // $hoehe       Hoehe des Feldes (nichts = 10)
  // $vorgabe     Inhalt der Variablen (nichts = aus Superglobals holen)
  // ---------------------------------------------------------------
  function feld ($name, $breite = NULL, $hoehe = NULL, $vorgabe = NULL)
  {
    $breite = ($breite == NULL) ? 80 : $breite;
    $hoehe  = ($hoehe  == NULL) ? 10 : $hoehe;
    $vorgabe = ($vorgabe == NULL) ? $this->returnwert($name) : $vorgabe;
    $vorgabe = str_replace("\\\"", '"', $vorgabe);
    $this->FORMULAR .= "<TEXTAREA name='$name' rows='$hoehe' cols='$breite'>".
                       $vorgabe .
                       "</TEXTAREA>";
    $this->INTERNE_VARIABLEN[$name] = $vorgabe;
  }

  // ---------------------------------------------------------------
  // Selektion ausgeben
  // $name      Name der Variablen
  // $sel       Array mit Selectionsangaben
  //            Beispiel:
  //            $sel["auswahlname"] = "Was wird angezeigt"
  // $selected  Welches Element ist selektiert ?
  //            (Voreinstellung = 1. Element bzw. lt. Superglobals)
  // $size      Anzahl der angezeigten Selection (Voreinstellung = 1)
  // ---------------------------------------------------------------
  function selection ($name, $sel, $selected = NULL, $size = NULL)
  {
    $size = ($size == NULL) ? 1 : $size;
    $selected = ($selected == NULL) ? $this->returnwert($name) : $selected;
    $this->INTERNE_VARIABLEN[$name] = $selected;
    $this->FORMULAR .= "<SELECT name='$name' size='$size'>";
    $i = 0; foreach ($sel as $key => $bezeichnung)
    {
      $SEL = ($selected == "")
           ? (($i == 0) ? "selected" : "")
          : (($key == $selected) ? "selected" : "");
      $i++;
      $this->FORMULAR .= "<OPTION value='$key' $SEL>$bezeichnung</OPTION>";
    }
    $this->FORMULAR .= "</SELECT>";
  }


// ---------------------------------------------------------------
// Checkbox ausgeben
// name       = Name der Checkbox
// wert       = Rueckgabewert
// anzeige    = das was angezeigt wird
// checked    = ausgewaehlt ?
//              Voreinstellung = Nein bzw. Superglobals
//              Jede Angabe bedeutet ja
// ---------------------------------------------------------------
  function checkbox ($name,$wert,$anzeige,$checked = NULL)
  {
    $checked = ($checked == NULL) ? $this->returnwert($name) : $checked;
    $checked = ($checked == "") ? "" : "checked=\"checked\"";
```

```
    $this->INTERNE_VARIABLEN[$name] = ($checked == "") ? "" : $wert;
    $this->FORMULAR .= "<INPUT type=\"checkbox\"".
                       " name=\"$name\"".
                       " value=\"$wert\"".
                       " ".$checked.
                       ">$anzeige</INPUT>";
  }

// ---------------------------------------------------------------------
// Radio-Button ausgeben
// name       = Name der des Feldes mit Radio-Buttons
// wert       = Rueckgabewert
// anzeige    = das was angezeigt wird
// checked    = ausgewaehlt ?
//              Voreinstellung = Nein bzw. Superglobals
//              Jede Angabe bedeutet ja
// ---------------------------------------------------------------------
  function radio_button ($name,$wert,$anzeige,$checked = NULL)
  {
    if ($checked == NULL)
    {
      $checked = $this->returnwert($name);
    }
    else
    {
      $checked = ($this->returnwert($name) == "") ? $wert : $this->returnwert($name);
    }
    $checked = ($checked == $wert) ? "checked=\"checked\"" : "";
    $this->FORMULAR .= "<INPUT type=\"radio\" name=\"$name\" value=\"$wert\" ";
    $this->FORMULAR .= $checked;
    $this->FORMULAR .= ">$anzeige</INPUT>";
    $this->INTERNE_VARIABLEN[$name] = ($checked == "") ? "" : $wert;
  }

  // ---------------------------------------------------------------------
  // Eingabe des Zwischentextes in das Formular
  // $text  auszugebender Text
  // ---------------------------------------------------------------------
  function inhalt ($text)
  {
    $this->FORMULAR .= $text;
  }

  // ---------------------------------------------------------------------
  // Wert fuer die Darstellung im Eingabefeld umrechnen
  // ---------------------------------------------------------------------
  function numeingabe2anzeige ($zahl, $nachkomma)
  {
        $zahl = trim($zahl);
    $zahl = str_replace ($this->trenner,".",$zahl);  // Nachkommatrenner aendern
    $zahl = str_replace ("|",".",$zahl);             // Nachkommatrenner am Nummernblock
    $zahl = number_format($zahl,$nachkomma,$this->trenner,$this->tausend); // umrechnen
    return $zahl;
  }

  // ---------------------------------------------------------------------
  // Formular abschliessen und komplettes Formular an aufrufendes
  // Programm zurueck geben
  // ---------------------------------------------------------------------
  function schliessen ()
  {
    $this->FORMULAR .= "</FORM>";
    return ($this->FORMULAR);
  }

  // ---------------------------------------------------------------------
  // Nur fuer Testzwecke
  // ---------------------------------------------------------------------
  function test ()
  {
    $ret  = "<HR><H1>TEST-INFO</H1>class cl_formular ($this->formular_ziel)<HR>";
```

```
    $ret .= "<TABLE border=\"1\"><TR><TH>Name</TH><TH>Inhalt</TH></TR>\n";
    $ret .= "<TR><TD colspan=\"2\"><B>\$_POST</B></TD></TR>";
    if (! empty($_POST)) {
    foreach ($_POST as $key => $inhalt)
    {
      $ret .= "<TR><TD>$key</TD><TD>$inhalt</TD></TR>\n";
    }}
    $ret .= "<TR><TD colspan=\"2\"><B>Interne Werte</B></TD></TR>";
    if (! empty($this->INTERNE_VARIABLEN)) {
    foreach ($this->INTERNE_VARIABLEN as $key => $inhalt)
    {
      $ret .= "<TR><TD>$key</TD><TD>$inhalt</TD></TR>\n";
    }}
    $ret .= "</TABLE>\n";
    return ($ret);
  }
}

?>
```

php/klassen/view.inc

```
<?php

require ("fpdf/fpdf.php");

// ----------------------------------------------------------------------
// Konstanten definieren
// ----------------------------------------------------------------------

// *** Art der Ansicht
define ('VIEW_PDF',  0);             // PDF-Ansicht
define ('VIEW_HTML', 1);             // HTML-Ansicht

// *** Definition der PDF-Seite
define ('A4_BREITE', 190);           // Sichtbare Breite der PDF-Seite
define ('A4_LAENGE', 260);           // Sichtbare Laenge der PDF-Seite

// *** Definition der Array-Bestandteile fuer die Uebersetzung der
// *** HTML-Tags in die PDF-Darstellung
define ('VIEW_TAGFONT',   'FONT');    // Welcher Font wird gewaehlt
define ('VIEW_COLOR',     'COLOR');   // Farbe der Schrift
define ('VIEW_BACKGROUND','BACK');    // Hintergrund-Farbe
define ('VIEW_TAGWIDTH', 'WIDTH');    // Schriftgroesse
define ('VIEW_TAGHIGTH', 'HIGHT');    // Zeilenhoehe
define ('VIEW_ALIGN',    'ALIGN');    // Ausrichtung

// ----------------------------------------------------------------------
// Klassendefinition
// ----------------------------------------------------------------------
class cl_view extends FPDF
{
  var $VIEW_ART;            // Art der Anzeige (PDF oder HTML)
  var $ANZEIGE;             // Seitentext fuer HTML-Anzeige
  var $PDF_Tag;             // Uebersetzung von HTML-Tags

// ----------------------------------------------------------------------
// PDF-Defnition fuer HTML-TAGS
// ----------------------------------------------------------------------
function tag_definition ($name,       // Name, z.B. H1 fuer <H1>
                         $font,       // Welcher Font wird gewaehlt
                         $color,      // Farbe der Schrift
                         $background, // Hintergrund-Farbe
                         $groesse,    // Schriftgroesse
                         $hoehe,      // Zeilenhoehe
                         $aus         // Ausrichtung
                        )
{
  $name = strtoupper ($name);
  $this->PDF_Tag["$name"][VIEW_TAGFONT]    = $font;
  $this->PDF_Tag["$name"][VIEW_COLOR]      = $color;
  $this->PDF_Tag["$name"][VIEW_BACKGROUND] = $background;
  $this->PDF_Tag["$name"][VIEW_TAGWIDTH]   = $groesse;
  $this->PDF_Tag["$name"][VIEW_TAGHIGTH]   = $hoehe;
  $this->PDF_Tag["$name"][VIEW_ALIGN]      = $aus;
}

// ----------------------------------------------------------------------
// HTML-Tags fuer die PDF-Darstellung definieren - diese Funktion
// muss ggf. in einer abgeleiteten Klasse ueberschrieben werden um
// eigene Definitionen zu integrieren
// ----------------------------------------------------------------------
function define_tag ()
{
  $this->tag_definition ("BASIS" ,"Arial",0,255,10,5,'L');    // Grundwert
  $this->tag_definition ("HEADER","Arial",0,220,20,10,'C');   // Seiten-Ueberschrift
  $this->tag_definition ("FOOTER","Arial",0,250,8,5,'L');     // Fusszeile
```

```
  $this->tag_definition ("H1","Arial",0,255,20,10,'C');       // <H1>
  $this->tag_definition ("H2","Arial",0,255,15,8,'C');        // <H2>
  $this->tag_definition ("H3","Arial",0,255,10,5,'C');        // <H3>
  $this->tag_definition ("BR","Arial",0,255,10,5,'C');        // <BR> Zeilenumbruch
}

// ----------------------------------------------------------------------
// PDF-Textausgabe
// ----------------------------------------------------------------------
function pdf_line ($s,$TAG,$FULL = 0)
{
  $TAG = strtoupper($TAG);
  $this->SetFont ($this->PDF_Tag["$TAG"][VIEW_TAGFONT],
                  '',
                  $this->PDF_Tag["$TAG"][VIEW_TAGWIDTH]);
  $this->SetTextColor($this->PDF_Tag["$TAG"][VIEW_COLOR]);
  $this->SetFillColor($this->PDF_Tag["$TAG"][VIEW_BACKGROUND]);
  $Breite = ($FULL == 0) ? $this->GetStringWidth($s) : 0;
  $this->Cell($Breite,
              $this->PDF_Tag["$TAG"][VIEW_TAGHIGTH],
              $s,
              0,
              0,
              $this->PDF_Tag["$TAG"][VIEW_ALIGN],
              ($this->PDF_Tag["$TAG"][VIEW_BACKGROUND] == 255) ? 0 : 1);
  // Zeilenumbruch muss immer gleich aktueller Zeichenhoehe sein
  if ($TAG == "BR")
    $this->Cell(0,$this->PDF_Tag["$TAG"][VIEW_TAGHIGTH],"",0,1,'L',
                ($this->PDF_Tag["$TAG"][VIEW_BACKGROUND] == 255) ? 0 : 1);
}

// ----------------------------------------------------------------------
// Klasse initialisieren
// ----------------------------------------------------------------------

function cl_view ($art,              // Art der Anzeige
                  $titel = NULL,     // Titel der HTML-Seite
                                     // bzw. Header der PDF-Seite
                       $body = NULL,      // Hintergrundfarbe oder CSS-Class
                                          // bzw. Footer der PDF-Seite
                       $css = NULL,       // Stylesheet
                       $background = NULL // Hintergrund-Bild
                      )
{
  switch ($art)
  {
    case VIEW_PDF :
      $this->FPDF("P","mm","A4");
      $this->SetHeader = ($titel == NULL) ? "" : $titel;
           $this->SetFooter = ($body == NULL)  ? "" : $body;
           $this->SetLeftMargin  (20);
           $this->SetRightMargin (20);
           $this->define_tag ();
           $this->FontName = ($css == NULL) ? 'Arial' : $css;
           $this->AddPage();
           $this->VIEW_ART = VIEW_PDF;
      break;
    default :
      $this->VIEW_ART = VIEW_HTML;
           $this->ANZEIGE  = "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\" ".
                             "\"http://www.w3.org/TR/html4/loose.dtd\">";
      $this->ANZEIGE .= "<HTML><HEAD><META http-equiv=\"content-type\" ".
                        "content=\"text/html; charset=utf-8\">";
           $this->ANZEIGE .= "<TITLE>";
           $this->ANZEIGE .= ($titel == NULL) ? basename($_SERVER["PHP_SELF"]) : $titel;
           $this->ANZEIGE .= "</TITLE>";
           $this->ANZEIGE .= ($css == NULL)
                           ? ""
                           : "<LINK rel=\"stylesheet\" type=\"text/css\" href=\"$css\">";
           $this->ANZEIGE .= "</HEAD>";
      $BODY  = ($body == NULL) ? ""
```

```
             : ($css == NULL)  ? "bgcolor=\"$body\"" : "class=\"$body\"";
      $BODY .= " ";
      $BODY .= ($background == NULL) ? "" : "background=\"$background\"";
      $this->ANZEIGE .= "<BODY $BODY>";
      break;
  }
}

// ----------------------------------------------------------------------
// Seite abschliessen
// ----------------------------------------------------------------------
function ende ()
{
  if ($this->VIEW_ART == VIEW_PDF)
  {
    $this->Close();
    $this->Output("","I");
  }
  else
  {
    $this->ANZEIGE .= "</BODY></HTML>";
    echo $this->ANZEIGE;
  }
}

// **********************************************************************
// Definition fuer
//
// P D F - D a r s t e l l u n g
//
// **********************************************************************

// ----------------------------------------------------------------------
// Kopfzeile der Blaetter - ueberschreibt leere Funktion aus FPDF
// ----------------------------------------------------------------------
function Header ()
{
  $this->SetY(10);
  $this->pdf_line ($this->SetHeader,"HEADER",1);
  $this->SetY($this->GetY() + 15);
}

// ----------------------------------------------------------------------
// Fusszeile der Blaetter - ueberschreibt leere Funktion aus FPDF
// ----------------------------------------------------------------------
function Footer ()
{
  $this->SetLineWidth (0.1);
  $this->AliasNbPages ("{MS}");
  $this->SetY (-15);
  $this->pdf_line ($this->SetFooter,"FOOTER",1);
  $this->Cell (0,$this->PDF_Tag["FOOTER"][VIEW_TAGHIGTH],$this->PageNo() . "/{MS}",'',1,'R');
}

function pdf_anzeige ($s,$art = "BASIS")
{
//
// Auswertung der HTML-Tags und ersetzen durch definierte Angaben
//
  $this->pdf_line ($s,strtoupper($art));
}

// **********************************************************************
// Definition fuer
//
// H T M L - D a r s t e l l u n g
//
// **********************************************************************
function html_anzeige ($s)
{
  $this->ANZEIGE .= $s;
```

```
}

// ********************************************************************
//
// Allgemeine Einstellungen
//
// ********************************************************************

function auswertung ($s,$art)
{
  if ($this->VIEW_ART == VIEW_PDF)
  {
    $this->pdf_anzeige ($s,$art);
  }
  else
  {
    $this->html_anzeige ("<$art>$s</$art>");
  }
}

function L_ine ($s ,             // auszugebender Text
                $art = "BASIS",  // HTML-Tag
                $nl = false      // Zeilenumbruch nach Textausgabe
               )
{
  $art = strtoupper($art);
  $this->auswertung ($s,$art);
  if ($nl == true)
  {
    $this->auswertung("","BR");
  }
}

}  // Ende der Klasse

?>
```

php/klassen/uvereinpdf.inc

```
<?php

require ("fpdf/fpdf.php");

// ----------------------------------------------------------
// Konstanten
// ----------------------------------------------------------
define ('CL_UVEREINPDF_TABSPALTEN',       0);    // Anzahl der Tabellenspalten
define ('CL_UVEREINPDF_TABSCHRIFT',       1);    // Tabellen-Schriftart, z.B. Arial
define ('CL_UVEREINPDF_TABSCHRIFTGROESSE',2);    // Tabellen-Schriftgroesse
define ('CL_UVEREINPDF_TABSPALTENBREITE', 3);    // Spaltenbreite jeder einzelnen Spalte
define ('CL_UVEREINPDF_TABRAHMEN',        4);    // Rahmen um die Tabelle ?
define ('CL_UVEREINPDF_TABAUSRICHTUNG',   5);    // Ausrichtung jeder einzelnen Spalte

class cl_uvereinpdf extends FPDF
{

var $TabColor;             // Hintergrundfarbe der Tabellenzeile
var $NewPage;              // Seitenwechsel anzeigen
var $SetTempDir;           // Pfad zum temporaeren Verzeichnis
var $PDFName;              // Name des temporaeren Dateinamens
var $Tabelle;              // Array mit Tabelleneigenschaften

// ----------------------------------------------------------------------
// Nicht darstellbare Zeichen in darstellbare konvertieren (intern)
// ----------------------------------------------------------------------
function uvereinpdf_caseconvert ($S_OLD)
{

  $S_OLD = str_replace ("&auml;" ,"ae",$S_OLD);
  $S_OLD = str_replace ("&Auml;" ,"Ae",$S_OLD);
  $S_OLD = str_replace ("&ouml;" ,"oe",$S_OLD);
  $S_OLD = str_replace ("&Ouml;" ,"Oe",$S_OLD);
  $S_OLD = str_replace ("&uuml;" ,"ue",$S_OLD);
  $S_OLD = str_replace ("&Uuml;" ,"Ue",$S_OLD);
  $S_OLD = str_replace ("&szlig;","ss",$S_OLD);
  return ($S_OLD);
}

// ----------------------------------------------------------------------
// Kopfzeile - wird automatisch von FPDF aufgerufen
// ----------------------------------------------------------------------
function Header ()
{
  $this->SetY (10);
  $this->SetFont ('Arial','',20);
  $this->SetTextColor (0);
  $this->SetFillColor (220);
  $this->SetHeader = $this->uvereinpdf_caseconvert($this->SetHeader);
  $this->Cell (0,10,$this->SetHeader,0,1,'C',1);
}

// ----------------------------------------------------------------------
// Fusszeile - wird automatisch von FPDF aufgerufen
// ----------------------------------------------------------------------
function Footer ()
{
  $this->SetLineWidth (0.1);
  $this->AliasNbPages ('{MS}');
  $this->SetY(-15);
  $this->SetFont ('Arial','',8);
  $this->SetTextColor (127,127,127);
  $this->SetFooter = $this->uvereinpdf_caseconvert($this->SetFooter);
  $this->Cell (0,5,$this->SetFooter,'T',0,'L');
  $this->SetFont ('Arial','',10);
  $this->Cell (0,5,$this->PageNo() . "/{MS}",'T',1,'R');
  $this->NewPage = true;
```

```
}

// ------------------------------------------------------------------
// Tabellenfunktionen
// ------------------------------------------------------------------

//
// Initialisieren
//

function table_init ($SPALTEN,              // Anzahl der Tabellen-Spalten
                     $FONT,$FONTSIZE,       // Schriftart und Groesse
                     $RAHMEN = 0,           // Rahmen ja/nein
                     $TABELLE = NULL,       // Array mit den Spaltenbreiten
                     $AUSRICHTUNG = NULL)   // Array mit den Spaltenausrichtungen
{
  $this->Tabelle[CL_UVEREINPDF_TABSPALTEN] = ($SPALTEN > 0) ? $SPALTEN : 1;
  $this->Tabelle[CL_UVEREINPDF_TABSCHRIFT] = $FONT;
  $this->Tabelle[CL_UVEREINPDF_TABSCHRIFTGROESSE] = $FONTSIZE;
  $this->Tabelle[CL_UVEREINPDF_TABRAHMEN] = $RAHMEN;
  // Spaltenbreiten auswerten
  if ($TABELLE != NULL)
  {
    if (count($TABELLE) < $this->Tabelle[CL_UVEREINPDF_TABSPALTEN])
    {
      $i = count($TABELLE); while ($i < $this->Tabelle[CL_UVEREINPDF_TABSPALTEN])
      {
        $TABELLE[$i] = 1;
        $i++;
      }
    }
    $i = 0; foreach ($TABELLE as $key => $inhalt)
    {
      $i += $inhalt;
    }
    $x = 0; foreach ($TABELLE as $key => $inhalt)
    {
      $this->Tabelle[CL_UVEREINPDF_TABSPALTENBREITE][$x] = intval(170 * $inhalt / $i);
      $x++;
    }
  }
  else
  {
    $x = intval(170 / $this->Tabelle[CL_UVEREINPDF_TABSPALTEN]);
    $i = 0; while ($i < $this->Tabelle[CL_UVEREINPDF_TABSPALTEN])
    {
      $this->Tabelle[CL_UVEREINPDF_TABSPALTENBREITE][$i] = $x; $i++;
    }
  }
  // Spaltenausrichtung auswerten (gilt nicht fuer Kopfzeile)
  if ($AUSRICHTUNG != NULL)
  {
    if (count($AUSRICHTUNG) < $this->Tabelle[CL_UVEREINPDF_TABSPALTEN])
    {
      $i = count($AUSRICHTUNG); while ($i < $this->Tabelle[CL_UVEREINPDF_TABSPALTEN])
      {
        $AUSRICHTUNG[$i] = "L";
        $i++;
      }
    }
    $i = 0; foreach ($AUSRICHTUNG as $key => $inhalt)
    {
      $WERTE = array ("L","l","R","r","C","c");
      if (in_array($inhalt,$WERTE))
        $this->Tabelle[CL_UVEREINPDF_TABAUSRICHTUNG][$i] = $inhalt;
      else
        $this->Tabelle[CL_UVEREINPDF_TABAUSRICHTUNG][$i] = "L";
      $i++;
    }
  }
  else
```

```
  {
    $i = 0; while ($i < $this->Tabelle[CL_UVEREINPDF_TABSPALTEN])
    {
      $this->Tabelle[CL_UVEREINPDF_TABAUSRICHTUNG][$i] = "L"; $i++;
    }
  }
}

//
// Ausgabe einer Tabellenzeile - Klassenintern
//

function internal_tabline ($line,$bold) {
  $this->SetFont ($this->Tabelle[CL_UVEREINPDF_TABSCHRIFT],$bold,
                  $this->Tabelle[CL_UVEREINPDF_TABSCHRIFTGROESSE]);
  $baselen = intval($this->GetStringWidth ("A"))+1;
  $i = 0; while ($i < count($line)) {
    $maxlen = $this->Tabelle[CL_UVEREINPDF_TABSPALTENBREITE][$i];
    $xlen = intval($maxlen / $baselen) - 5;
    $INHALT = explode (" ",$this->uvereinpdf_caseconvert($line[$i]),$xlen);
    $iii = 0; $S[$i][$iii] = "";
    $ii = 0; while ($ii < count($INHALT)) {
      $txtlen = $this->GetStringWidth($S[$i][$iii]) +
                $this->GetStringWidth($INHALT[$ii]);
      if ($txtlen >= $maxlen) {
        $iii++; $S[$i][$iii] = "";
      }
      $S[$i][$iii] .= $INHALT[$ii] . " ";
      $ii++;
    }
    $i++;
  }
  $maxlines = 1; $i = 0;
  while ($i < $this->Tabelle[CL_UVEREINPDF_TABSPALTEN]) {
    $maxlines = ($maxlines > count($S[$i])) ? $maxlines : count($S[$i]);
    $i++;
  }
  $i = 0; while ($i < $maxlines)   {
    $ii = 0; while ($ii < $this->Tabelle[CL_UVEREINPDF_TABSPALTEN]) {
      if ($this->Tabelle[CL_UVEREINPDF_TABRAHMEN] != 0) {
        if ($i == 0) $Border = (($i + 1) <  $maxlines) ? "TLR" : 1;
        else         $Border = (($i + 1) == $maxlines) ? "BLR" : "LR";
      }
      else $Border = 0;
      if ($i < count($S[$ii])) {
        $this->Cell ($this->Tabelle[CL_UVEREINPDF_TABSPALTENBREITE][$ii],
                     intval($this->Tabelle[CL_UVEREINPDF_TABSCHRIFTGROESSE]/2),
                     $S[$ii][$i],$Border,0,
                     $this->Tabelle[CL_UVEREINPDF_TABAUSRICHTUNG][$ii],0);
      }
      else {
        $this->Cell (intval($this->Tabelle[CL_UVEREINPDF_TABSPALTENBREITE][$ii]),
                     intval($this->Tabelle[CL_UVEREINPDF_TABSCHRIFTGROESSE]/2),
                     " ",$Border,0,
                     $this->Tabelle[CL_UVEREINPDF_TABAUSRICHTUNG][$ii],0);
      }
      $this->Cell (1,
                   intval($this->Tabelle[CL_UVEREINPDF_TABSCHRIFTGROESSE]/2),
                   " ",$Border,0,'',0);
      $ii++;
    }
    $this->Cell (0,intval($this->Tabelle[CL_UVEREINPDF_TABSCHRIFTGROESSE]/2)," ",0,1,'',0);
    $i++;
  }
}
//
// Kopfzeile
//
function table_head ($TABELLE)
{
  $i = 0; while ($i < $this->Tabelle[CL_UVEREINPDF_TABSPALTEN])
```

```
  {
    $MERKER [$i] = $this->Tabelle[CL_UVEREINPDF_TABAUSRICHTUNG][$i];
    $this->Tabelle[CL_UVEREINPDF_TABAUSRICHTUNG][$i] = "C";
    $i++;
  }
  $this->internal_tabline ($TABELLE,"B");
  $i = 0; while ($i < $this->Tabelle[CL_UVEREINPDF_TABSPALTEN])
  {
    $this->Tabelle[CL_UVEREINPDF_TABAUSRICHTUNG][$i] = $MERKER[$i];
    $i++;
  }
}

//
// Normale Tabellenzeile
//
function table_line ($TABELLE)
{
  $this->internal_tabline ($TABELLE,"");
}

// ----------------------------------------------------------------------
// Normalen Text ausgeben
// ----------------------------------------------------------------------
function textzeile ($S,$FONT = "Arial",$FONTSIZE = 10, $AUSRICHTUNG = "L")
{
  $this->SetFont ($FONT,'',$FONTSIZE);
  $S = $this->uvereinpdf_caseconvert($S);
  $this->MultiCell (0,intval($FONTSIZE/2),$S,0,$AUSRICHTUNG,0);
}

// ----------------------------------------------------------------------
// Initialisierung
// ----------------------------------------------------------------------
function cl_uvereinpdf ($Head, $Foot)
{
  $this->FPDF ("P","mm","A4");
  $this->SetHeader = $Head;
  $this->SetFooter = $Foot;
  $this->NewPage = false;
  $this->SetLeftMargin(20);
  $this->SetRightMargin(20);
  $this->SetTempDir = "../tmp";
  $this->AddPage();
  $this->SetFont ('Arial','',18);
  $this->PDFName = "uverein" . time() . ".pdf";
}

function anzeige ()
{
  $this->Close ();
  $this->Output ($this->SetTempDir."/".$this->PDFName,"F");
  echo "<SCRIPT language=\"JavaScript\">";
  echo "self.location.href=\"".$this->SetTempDir."/".$this->PDFName."\";";
  echo "</SCRIPT>";
}

}

?>
```

uverein
Anlagen