Hinweis: diese Datei listet die Beiträge bis #104 im Beitrag *Tagebuch einer Extension-Entwicklung* im Forum *Entwickler-Tutorials* der Contao-Community auf.

Allerdings sind dabei nicht nur die Lobhudel-Beiträge (damit es keine Missverständnisse gibt: auch ich finde Stefans Tutorial toll!) entfernt, sondern auch manche Anrede und Grußformel. Ich wollt's so kurz wie möglich halten. Auch die Fotos der Poster und den ganzen Forumskram habe ich entfernt.

Ich habe keine Sorgfalt darauf verwendet, dass

- die Seitenumbrüche ordentlich sind und
- die Zeilenumbrüche im Code korrekt sind,

da mir das

- nicht so wichtig und vor allem
- zuviel Arbeit

war. Für meine Zwecke langt es so wie es ist.

Inhaltsverzeichnis

Schritt 1: Anforderungen & Randbedingungen....4

Schritt 2: Extension-Creator....10

Schritt 3: Backend-Modul registrieren und SQL-Tabellenstruktur anlegen....12

Schritt 3b: SQL reloaded....18

Schritt 4: Wir wagen uns in das DCA-Land....20

Schritt 4b: Verwirrung im DCA-Land....27

Schritt 4c: Leichte Entwirrung im DCA-Land....28

Schritt 4d: DCA-Polishing....30

Schritt 4f: Nochmal anders....36

Schritt 4g: DCA - Almost there....37

Schritt 5: Language-Files....44

Schritt 6: Von Callbacks und Subpaletten....49

Schritt 7: Endlich Frontend!....55

Schritt 7b: Ein wenig Finetuning....60

Schritt 8: Parameter fürs Modul....61

Schritt 8b: Parameter für das Modul, die zweite....63

Schritt 9: Details, Details, Details!....66

Schritt 10: Etwas Feinschliff....74

Schritt 11: Die zweite Tabelle....80

--> gelöst!!! 94

Schritt 12: Callback-Magie....97

Schritt 13: Frontendmodul Meldeliste....112

Schritt 13b: JOIN-Power....121

Detailseiten 122

Hooks 123

dl1ely

Tagebuch einer Extension-Entwicklung

AKTUELLER STAND DES QUELLTEXTES:
Aktueller Stand des Quelltextes vom 29.5.
Bitte das ZIP innerhalb des /system/modules/-Ordners entpacken!

Ich möchte hier gerne eine Art "Tagebuch" einer Extension-Entwicklung schreiben. Ich besitze das TL-Buch, ich kenne das "Hello, World"-Beispiel, ich kenne das CD-Collection-Beispiel, ich kenne das FlowPlayer-Beispiel. Trotzdem ist mir weiterhin nicht glasklar, welche Schritte man bis zur eigenen Extension gehen muss. Ich bin PHP-erfahren, habe aber noch nie eine TL-Extension geschrieben. Ich möchte mit diesem Tagebuch anderen "Anfängern" die Möglichkeit geben, an einem realen Beispiel zu lernen, und auch meine Entscheidungen und Gedanken entlang des Wegs kennen zulernen. In vielen Tutorials wird nur vorgegeben, was in welche Datei geschrieben wird, und was Zeile X oder Zeile Y dort tut. Der Prozess hin zu Zeile X oder Y bleibt leider zu oft im Dunkeln, und erschwert mir die Übertragung der vorgestellten Tutorial-Inhalte auf meine eigenen Probleme. Ich möchte hier auch Irrwege dokumentieren, wie sie für Extension-Anfänger wahrscheinlich typisch sind, und nicht nur ein Tutorial zum Endprodukt abliefern. Ich hoffe, die Foren-Admin akzeptieren so etwas als "Tutorial", falls nicht, dann bitte ich um Verschiebung in die "Fragen"-Sektion. Wobei ich eher berichten als fragen möchte. Natürlich freue ich mich auch über Hinweise, wenn ich vielleicht ganz in die falsche Richtung denke, oder ich im Rahmen meines "Tutorials" selbst nicht mehr weiter weiß.

Vorneweg:
TL = TYPOlight
FE = Frontend (Seite, die jeder Surfer sieht)
BE = Backend (Seite, die nur eingeloggte "Redakteure" sehen unter der /typolight URL)

Alles soll sich unter TL 2.8.0 abspielen, und ich werde den Extension Creator verwenden, um mir das "Skelett" der Extension zu erstellen.

Die Extension wird sehr speziell auf die Bedürfnisse der von mir betreuten Seite zugeschnitten sein. Ich glaube nicht, dass mit dem engen Scope jemand anders sie nutzen kann. Darum werde ich sie wohl auch nicht im Extension-Repository veröffentlichen. Ich kann und werde den Quelltext aber sicherlich zur Verfügung stellen, sei es zur Benutzung oder zum "Studium".

Schritt 1: Anforderungen & Randbedingungen

Zunächst mal zum Hintergrund:

Ich bin Webmaster der Seite http://www.gruen-weiss-aachen.de , der Webpräsenz des momentan größten Tanzsportvereins in NRW, und einem der größten in Deutschland. Wie man unschwer erkennen kann, bedarf die Seite dringend eines Relaunchs, der auf Basis von TYPOlight erfolgen soll. Dazu erfolgt auf einer Subdomain parallel zum weiteren Betrieb der bisherigen Webseite der Aufbau einer TL-basierten Seite.

Das Design steht noch nicht, erstmal soll die technische Funktionalität geschaffen werden. 95% der bisherigen Seiteninhalte lassen sich mit "TL-Bordmitteln" oder schon verfügbaren Extensions wie efg erschlagen. Allerdings gibt es auf der alten Seite einige "handgestrickte" PHP-Skripte, deren Funktionalität mehr oder weniger auf der neuen Seite erhalten bleiben soll.

Eine Funktionalität, die sich mit verfügbaren Extensions meiner Meinung nach nicht nachbilden lässt, und für die ich deshalb eine eigene Extension entwickeln möchte/muss, ist folgende:

Der Verein besitzt eine Menge an Tanzturnierpaaren. Zu diesen Turnierpaaren gehören Daten wie Name(n), Start- und Alterklassen, und ein Flag, ob es noch aktiv ist. Diese Daten werden von einer berechtigten Person (Sportwart) im BE gepflegt. Zusätzlich gehört zu einem Turnierpaar eine Art "Visitenkarte" mit Bild, Freitext, Email-Adresse, Telefonnummer...Diese Daten sollen auch im Backend pflegbar sein, zusätzlich sollen diese Daten aber auch von dem Turnierpaar selbst unter Kenntnis eines individuellen Passworts veränderbar sein. Dazu möchte ich aber KEINE Frontend-User-Verwaltung nutzen.

Für jedes Turnierpaar gibt es N "Meldungen", dies sind Datensätze, die enthalten, zu welchem Tanzturnier ein Paar fahren wird/gefahren ist. Turnierpaare informieren den Sportwart darüber, bei welchen Turnieren sie starten wollen. Der Sportwart führt die Anmeldung beim Turnierveranstalter durch (So ist das Procedere im Tanzsport), und trägt danach die "Meldung" in der Datenbank ein. Auf der sog. "Meldeliste" kann das Tanzpaar dann selbst sehen, ob es zu einem Turnier gemeldet worden ist. Die Liste der Meldungen (1:N-Beziehung zu den Turnierpaaren) wird vom Sportwart im BE gepflegt. Auch hier gibt es die Möglichkeit, dass ein Turnierpaar unter Kenntnis seines Passworts seinen erzielten Platz auf einem Turnier und einen freien Kommentar zu einer Meldung hinzufügen kann.

Sollte das Tanzpaar das Turnier seiner Leistungsklasse gewonnen haben, darf es direkt im Anschluss das Turnier der nächsthöheren Klasse mittanzen (Ohne Meldung durch den Sportwart). Hier muss es die Möglichkeit geben, dass das Paar selbst einen Meldungs-Eintrag in der Datenbank vornehmen kann (Das sog. Folgeturnier).

Die Meldeliste dient also mehreren Zwecken:
- Rückmeldung vom Sportwart an die Paare, dass die Meldung beim Veranstalter erfolgt ist
- Außenwirkung für den Verein: Wie viele Paare haben wir, und wo starten sie
- Außenwirkung für die Tanzpaare selbst: Wie gut haben wir abgeschnitten

Übertragen auf die Tabellenstruktur handelt es sich also um die Tabelle der Turnierpaare, klassisch im BE durch ein BE-Modul gepflegt. Die Meldungen sind "Childs" der Turnierpaare, und sollten so auch im Backend angezeigt werden (Wie Artikel - Inhaltselemente). Zugriff auf das BE-Modul hat nur der Sportwart.

Zusätzlich soll es einen Mechanismus geben, unter Angabe eines Tanzpaar-spezifischen Passworts bestimmte Felder in der Tanzpaar-Tabelle und in der Melde-Tabelle durch ein Frontend-Formular zu verändern. Außerdem soll es die Möglichkeit geben, einen neuen Satz in der Melde-Tabelle anzulegen.

Die Ausgabe der Daten im Frontend soll auf verschiedene Weisen geschehen:

- Modul "Aktive Turnierpaare"
- Modul "Ehemalige Turnierpaare"

Für beide Module jeweils eine Detail-Ansicht des gewählten Turnierpaares mit Bild, Freitext, freigegebenen Kontaktdaten wie Telefon oder Email, und einer Liste aller Meldungen dieses Paares

- Modul "Meldeliste" - Eine chronologisch absteigend sortierte Liste aller Melde-Einträge, Neueste also zuerst.

- Module/Formulare für Änderung der Paardaten im Frontend und zum Hinzufügen von Platzierung/Kommentar bei Meldungen

Da das Ergänzen/Ändern der einzelnen Tabellenfelder mithilfe des Paar-Passworts ohne die Einführung von Frontend-Usern doch ziemlich an der TL-Philosophie vorbeigeht, tendiere ich momentan dazu, dafür Formulare zu verwenden, deren Weiterleitungsziel eine Seite ist, die per insert-Tag ein stand-alone-PHP-File einbindet, was nach Prüfung des Passworts die entsprechenden Datenbankfelder updatet.

Gibt es hier eigentlich Erfahrungen mit zeitgleichem Zugriff auf Tabellen bei zwei eingeloggten BE-Usern, oder wie hier in meinem Fall, wenn ein Skript auf FE-Seite potentiell Daten in der Datenbank ändert, die vielleicht gerade ein BE-User betrachtet? Aber damit werde ich leben, die Kollisionswahrscheinlichkeit ist sehr gering.

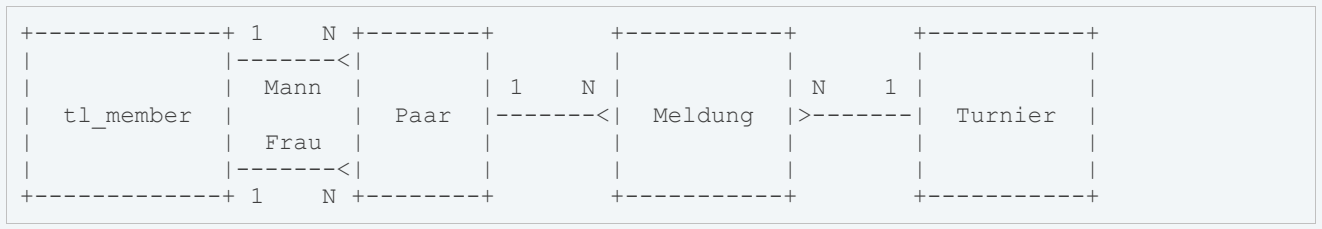
Coming up next: Extension-"Skelett" anlegen mit dem Extension-Creator.

---

deerwood

kannst Du bitte begründen, warum Du die Frontend-Benutzer/Member Funktionalität nicht nutzen willst? Ich sähe nämlich (aus der Hüfte) in etwa dies Datenmodell:

Code:



Dann könnte man z.B. leicht prüfen, ob ein angemeldeter FE Nutzer Felder eines Paar-Datensatzes bearbeiten darf usw.: seine ID muss == Mann/Frau ID sein. Direkte Personendaten wären in tl_member, nur echte Paardaten wären in Paar. Turnierdaten (Termin, Ort usw.) wären sauber/normalisiert in Turnier, Meldung enthielte nur noch wenig, wie etwa Anmeldetermin, Anmelder (Sportwart oder Paar-Partner), erreichter Platz. In Tabelle Turnier könnte man auch eine Selbstreferenz auf das Folgeturnier haben und so verhindern, dass Paar-Partner sich selbst für beliebige Turniere anmelden. Man könnte auch herausbekommen, wenn Partner im Laufe der Zeit in verschiedenen Paarungen tanzen.

Eventuell könnte man diese Erweiterung auch generalisieren: statt "Paar" etwa "Mannschaft/Gruppe" und nicht 2 Verweise auf tl_member, sondern ein serialisiertes Array mit Mitgliedern und ihrer Rolle in der Mannschaft (Mann/Frau, Torwart, Schlagzeuger, Vorschoter) und z.B. dem Start/Ende-Datum der Mitgliedschaft in der Gruppe ... oder *klassisch* sogar eine N:M Tabelle zwischen tl_member und "Mannschaft/Gruppe". Besonderheiten wie "Anmeldung normalerweise nur durch den Sportwart" könnte man eventuell in eine Erweiterung zum Modul auslagern (Hook vorsehen bei der Rechteprüfung z.B.), dann hätte man ein Basismodul, das auch andere verwenden können und eben eine Tanzturnier-Speziallösung. Ich will nicht unnötig verkomplizieren, aber ich denke man sollte im Vorfeld schon versuchen, das Modul wiederverwendbar auszulegen und möglichst auf das TL Framework setzen.

Ich bin auch kein erfahrener Modul-Programmierer und werde diesen Thread mit Interesse verfolgen.

---

deerwood

nochmals etwas weiter gedacht:

Mitglieder-Gruppen (tl_member_group) gibt es ja schon in TL mit allem Drum und Dran / Verwaltung. Vielleicht muss man für Mannschaften/Paare nur noch einige Felder und Logik ergänzen?

Und was sind denn Turniere/Veranstaltungen? Das sind doch "Events" (tl_calendar_events), für die es ein Basis-Modul in TL gibt, und, soweit ich sehen kann, auch schon diverse Erweiterungen, die zusätzliche Felder bieten.

Was fehlt also noch? Eigentlich nur die "Mitte" (Meldung) zwischen member_group und calendar_event.

Das ist halt eine zusätzliche/besondere Beziehung/Relation zwischen member_group und calendar_event, die eine zusätzliche Logik erfordert, sobald jemand die Beziehung herstellen/bearbeiten möchte, implementiert als Modul (und wohl mit einer zusätzlichen N:M Tabelle "Meldung/Teilnahme").

Bin ich abgeflogen, oder versteht jemand diesen Ansatz oder bin ich einfach dumm und alle machen das bereits so?

---

dl1ely

vielen Dank für deinen Input. Habe ihn gestern Abend noch gelesen, aber keine Zeit/Ruhe zum Antworten gehabt. Prinzipiell hast du mit deiner Idee sicherlich recht, auch wenn ich aufgrund von (Noch-)Unkenntnis der schon implementierten Tabellenstruktur nicht sagen kann, ob da irgendwo noch ein Haken ist. Für ein abstraktes "Lehrbuchbeispiel" einer Extension wäre das wohl genau der richtige Weg.

Folgende Gründe (Die alle nicht "zwingend" sind, aber die Summe macht es) machen es aber für mich unattraktiv:

- Eine Normalisierung in Meldung und Turnier ist in 90% der Fälle Overkill. Häufig gibt es zu einem Turnier genau eine Meldung. Häufig wird es sich also um eine 1:1-Beziehung handeln.

- Handling der "Personen" als Front-Member: Werde ich mir nochmal anschauen, aber ein großes Erbe in meinem konkreten Fall ist leider die Notwendigkeit, schon bestehende Daten zu migrieren, insbesondere die Liste der Turnierpaare, die bis in die 90er Jahre zurückreicht. Man kann sich vorstellen, dass da bei aktuell 45 Turnierpaaren (=90 Personen AKTIV) über die Jahre geschätzt 300-400 beteiligte Personen angesammelt haben, die ich alle in die tl_member-Tabelle migrieren müsste, obwohl davon eben nur 90 überhaupt noch potentiell einloggen.

  Die "Alten" nutzen die Seite eh nicht mehr, sollen sowieso nicht mehr ihre alten Einträge editieren dürfen, und sind (ehrlich gesagt) inzwischen teilweise verstorben. Für einen Großteil der Personen, die (inaktive) Paare bilden, müsste ich aus dem alten "Paar"-Datensatz zwei Datensätze für die tl_member-Tabelle erstellen, und mir dabei größtenteils die Inhalte der Datenfelder (email, passwort, whatever) ausdenken.

  Natürlich sehe ich die von dir geschilderten Vorteile der Abstraktion/Normalisierung, der Verfolgbarkeit des Wechsels von Paarzusammenstellungen, usw...Aber ich erkaufe es mir auch mit dem oben geschilderten Overhead.

- Im Tanzpaar übernimmt meistens einer das Eintragen der Ergebnisse (meist der Herr :-). Wenn ich die Damen auch als Member anlegen würde, würden 45% der Member-Einträge selbst der aktiven Paare wahrscheinlich nie genutzt werden.

- Vorschlag der Generalisierung: Sicherlich erstrebenswert, aber ich sehe die Gefahr, mich vor lauter Generalisierung "zu verzetteln", und 90% meiner Arbeit in "what-ifs" zu stecken, die in meinem Fall gar nicht benötigt werden.

Wenn ich mir den Source der efg-Extension angucke, die ja eine eierlegende Wollmilchsau ist, dann wird mir schwindelig, obwohl ich von mir behaupte, ganz fit in PHP zu sein. Ich verstehe nur einen Bruchteil der Dinge, die da abgehen. Auch wenn es für den Launch der neuen Webseite keinen fixen Termin gibt, möchte ich eigentlich schnell zu Ergebnissen kommen, und kein "never-ending Project" draus machen, an dem ich mich verhebe. Schließlich ist das meine allererste Extension.

Vielleicht wird eine v2 daraus, die dann Anspruch hat, eine allgemeine "Sport-Mannschaften-Turnierwesen"-Verwaltungs-Extension zu sein. Erstmal möchte ich aber mein Problem lösen. So clean wie möglich, aber auch so dirty wie nötig.

Im Endeffekt will ich in meinem Tutorial zeigen, wie man Informationen mit Vater-Kind-Beziehungen ablegt, im BE editierbar macht, und in verschiedenen "Views" im Frontend anzeigt. Meine "Spezialsauerei" der Editierbarkeit eines Teils der Daten aus dem Frontend heraus darf jemand, der das hier nachvollziehen möchte, gerne gedanklich filtern. Da ist mir klar, dass es der reinen TL-Lehre etwas widerspricht. Ich werde diese Funktionalität auch als Letztes implementieren, und vielleicht sogar aus diesem "Bericht" her ganz rauslassen. Dann wird es nur eine Art "extended CD-Collection"-Beispiel (http://dev.typolight.org/projects/ty...rialsExtension), was schon fast in meine Richtung geht, was mir nur _zu_ Hoppla-Hopp durch alle Schritte durchgeht.

- Ich muss bedenken, dass alle Anwender (Der Sportwart im Backend, aber vor Allem die Turnierpaare) nun jahrelang einen gewissen Workflow bezüglich Meldeliste und Eintragen der Turnierergebnisse gewöhnt sind. Ein großer Teil unserer Turnierpaare befindet sich im mittleren bis hohen Seniorenalter, und schafft es trotzdem seine Turnierergebnisse online einzutragen. Ich kann hier keine großen Revolutionen veranstalten und möchte den technischen Ablauf deshalb möglichst nachbilden.

  Da es bisher möglich war, durch Angabe eines Paar-Passworts seine Turnierergebnisse in einem Formular einzutragen, würde ich das gerne beibehalten. Ein formelles "Einloggen" in die Seite wäre für manche Anwender wahrscheinlich schon zuviel der Veränderung :-). Außerdem schürt so etwas bei Nicht-Turnierpaaren das Gefühl, dass es für eingeloggte Personen "Geheimseiten" gibt, die ihnen nicht zur Verfügung stehen. In diesem Fall wäre das ja auch so, wäre aber nicht transparent.

  Wenn jeder Benutzer die Formularseite "Turnierergebnis für Paar XY eintragen, aber nur wenn man das Paarpasswort kennt" sieht, wird das viel weniger Fragen aufwerfen, als wenn es auf einmal "Login-Möglichkeiten" im FE geben wird, die es bisher nicht gab. Das hat immer den Geruch von Privilegisierung, die bei den "Nicht-Privilegierten" auf Ablehnung stößt.

  Ich weiß, das klingt übertrieben, ist aber ein Erfahrungswert. Mache ich ein Frontend-Login auf die Seite, wird es sofort 10-20 Anfragen geben, ob man (als nicht-Turnierpaar) nicht auch einen Login bekommen könnte, weil man sonst ja bestimmt was verpasst...Ich will da keine Begehrlichkeiten wecken.

- Das Problem der Datenmigration habe ich schon mal angesprochen. Sowohl in der Turnierpaarliste als auch in der Meldeliste gibt es Einträge bis in die 90er Jahre zurück, die ich natürlich ins neue System übernehmen muss.

  Jede Normalisierung über die Beziehung "Turnierpaar<- 1:N ->Meldung" hinaus, so sinnvoll sie akademisch auch sein mag, macht mir sehr viel Arbeit beim Schreiben eines Migrationsskriptes, weil ich da nämlich dann Normalisierung algorithmisch durchführen muss, die in den alten Daten nicht vorhanden ist. Von daher würde ich gerne auch unter Missachtung von "clean-Design"-Grundsätzen bei der Konzeption der Struktur der Datenbanktabellen möglichst nah an der etablierten Form bleiben.

  Die Nachteile erscheinen mir erträglich, die Vorteile beim Arbeitsaufwand immens. Ich habe auch schon gesehen, wie mit sklavischer Normalisierungswut sich selbst in den Fuß schießt, weil man vor lauter Tabellen, die N:M-Beziehungen abbilden, kein vernünftiges SELECT mehr schreiben kann...

  Man darf da gerne anderer Meinung sein, für mein Beispiel würde ich gerne bei zwei Tabellen, eine für Turnierpaare, eine für Meldungen, bleiben. Für die Turnierpaare schaue ich mir nochmal die

tl_member-Tabelle an, ob ich die Personen dort nicht ablege, aber aus den o.g. psychologischen Gründen würde ich ungerne ein Frontend-Login einführen (müssen).

- Auf PHP-Seite möchte ich natürlich so weit wie möglich (und so weit es meine bisherigen Kenntnisse des Frameworks es zulassen) auf das TL-Framework setzen. Ist doch klar!

Auf jeden Fall vielen, vielen Dank für das Feedback so weit, genau so hatte ich mir den Thread/die Diskussion eigentlich vorgestellt. Mit Vorschlägen von Alternativen, Anbringen von Zweifeln, usw...

---

dl1ely

Sehr gerne, danke für den Tipp. Ich werde die Posts, wo es in der Extension auch wirklich vorwärts geht zusätzlich mit roten großen Überschriften versehen, damit man später beim Lesen des ganzen Threads die "Nebendiskussion" leichter überspringen kann.

FloB

> Zitat von **dl1ely**
>
> *Handling der "Personen" als Front-Member: Werde ich mir nochmal anschauen, aber ein großes Erbe in meinem konkreten Fall ist leider die Notwendigkeit, schon bestehende Daten zu migrieren, insbesondere die Liste der Turnierpaare, die bis in die 90er Jahre zurückreicht. […]*
>
> *Die "Alten" nutzen die Seite eh nicht mehr, sollen sowieso nicht mehr ihre alten Einträge editieren dürfen, und sind (ehrlich gesagt) inzwischen teilweise verstorben. Für einen Großteil der Personen, die (inaktive) Paare bilden, müsste ich aus dem alten "Paar"-Datensatz zwei Datensätze für die tl_member-Tabelle erstellen, und mir dabei größtenteils die Inhalte der Datenfelder (email, passwort, whatever) ausdenken.*

Nun, aus was bestehen denn die bestehenden Daten? Name, Geburtsdatum ist klar, dann noch Tanzpartner (am besten UserID – oder "Gruppen-ID", sprich zwei Partner bekommen eine neue Nummer? Damit wäre auch der Grundstein gelegt, ganze Tanzgruppen ohne Mehraufwand anzulegen), Turnierklasse, aktive Jahre. Diese Felder kann man ja problemlos zu tl_member hinzufügen.

Wenn du diese Daten in einem bestimmten Format hast (CSV, XML o. ä.) kannst du sie per SQL importieren. Die Daten, die du von manchen Personen nicht hast (E-Mail etc.) musst du da nicht angeben – nur wenn man versucht diese Einträge über das Backend / Frontend zu ändern, meckert TL. Frontendanzeige sollte problemlos funktionieren. Somit hast du die saubere Integration von Nutzer und Paaren.

OK, ich verstehe dein Argument, dass das möglicherweise ein wenig Overkill für diese Art von Daten ist. Dann wäre es sinnvoll für Paare eine zugehörige "Verwalter-ID" zu geben, die auf einen Member-Account verweist. Dieser Member kann dann alle Einträge ändern, auf dessen die Paare / Personen verweisen. Ein zusätzliches Feld für die Trainer-ID (hinter dem auch ein Member steckt) in der Paarliste ermöglicht dann dem Trainer zusätzliche Einstellungen an der Paarung vorzunehmen (z. B. Turnier bestätigen).

---

dl1ely

Hehe, ihr blast das Ding schon viel zu weit auf...Trainer-ID o.ä. ist nicht notwendig, Trainer spielen in diesem "Prozess" keine Rolle. Ich denke es wird klarer (vielleicht auch enttäuschender), wenn ich die SQL-Files für meine Daten zeige. Es ist eigentlich recht billig, gerade für einen "TL-Profi". Aber das bin ich noch nicht ;-).

Die Daten liegen noch in einer (anderen) MySQL-Datenbank. Der technische Vorgang der Migration ist kein Problem, aber ich werde per Skript manche Felder der bisherigen Datenbankstruktur verändern müssen, um es

meiner geplanten neuen Struktur anzupassen.

Leider komme ich momentan nicht dazu, diesen Thread täglich weiterzuführen, es ist nur Hobby. Aber ich mache weiter, keine Sorge :-).

Schritt 2: Extension-Creator

Also, los geht es mit dem Skelett für die geplante Extension. Ich verwende den Extension-Creator aus dem Extension-Repository.

- Titel: Es geht um Turnierpaar-"Verwaltung", und um Namenskonflikte zu vermeiden möchte ich gerne einen spezifischen Präfix nutzen: gw angelehnt an den Vereinsnamen "Grün-Weiß". Also: Titel = gw_turnierpaare. An dieser Stelle bin ich mir noch nicht so sicher, wo dieser "Titel" überall erscheinen wird, und ob es deshalb ein beliebiger Text sein soll, oder eher ein "identifier", also z.b. ohne Leerzeichen u.ä. Sicherheitshalber gehe ich den Identifier-Weg. Besser hässlich als nicht-funktionierend.

- Ordnername: Ebenfalls gw_turnierpaare.

- Autor, Copyright und Lizenz: Selbsterklärend

- Paket: Ein Paketname ist gefragt. Der Hilfetext unter dem Eingabefeld schlägt hilfreich "z.B. meinEigenesModul" vor. Ist das Modul jetzt das Paket? Was ist ein Paket? Sowas wie ein Namensraum? Da ich noch nicht weiß, ob ich für die Vereinsseite noch andere Extensions pogrammieren werde, nehme ich die Vereinsabkürzung als "Paketname", also "GW". Weitere vereinsspezifische Extensions würde ich dann in dasselbe Paket stecken.

- Ein Backend-Modul hinzufügen: Ja klar, schließlich sollen Daten im Backend bearbeitet werden. Also dort ein Häkchen.

- Backend-Klassen: Wenig hilfreicher Erklärungstext: "Hier können Sie eine kommagetrennte Liste der zu erstellenden Backend-Klassen eingeben." - Was sind Backend-Klassen? Wofür brauche ich die? Eigentlich müsste doch alles, was ich im Backend vorhabe, durch Einträge im DCA-File realisierbar sein, schließlich geht es nur um Pflege von zwei abhängigen Datenbanktabellen. Also mal mutig leer gelassen, falls das falsch ist kann man es später noch hinzufügen.

- Backend-Tabellen: Das sind wohl meine Datentabellen, ich will eine für Turnierpaare, eine für Meldungen, also: "tl_gw_turnierpaare,tl_gw_meldungen".

- Backend-Templates: Auch hier wieder wenig erhellender Hilfetext. Auch das TL-Buch beschränkt sich da leider fast auf das Abschreiben der Hilfetexte unter den Eingabefeldern. Ich kenne Templates nur für das Frontend, also beschließe ich mutig, dass ich das wohl nicht brauche. Sollte sich das später als Irrtum herausstellen, wird es sich hoffentlich noch korrigieren lassen.

- Ein Frontend-Modul hinzufügen: Aber klar, die Daten sollen schließlich im Frontend visualisiert werden. Also Haken dran.

- Frontend-Klassen: So weit wie ich es bisher verstanden habe, braucht jedes Modul eine eigene Klasse. Nach meiner bisherigen Planung brauche ich ein Modul "Turnierpaarliste" inklusive Detail-Ansicht der einzelnen Paar-Einträge. Die Unterscheidung aktiv/nicht aktiv würde ich gerne über einen Parameter im Modul lösen, so dass dasselbe Modul die Liste der aktiven und der ehemaligen Paare anzeigen kann. Außerdem benötige ich ein Modul "Meldeliste" mit der chronologisch sortierten Übersicht der Meldungen. Also: "gwTurnierpaarliste,gwMeldeliste" als Klassennnamen.

- Frontend-Tabellen: Ratlosigkeit. Was unterscheidet Frontend-Tabellen von Backend-Tabellen? Da ich meine Tabellen schon in den Backend-Tabellen abgehandelt habe, lasse ich das Feld leer.

- Frontend-Templates: Natürlich! "gw_turnierpaarliste,gw_turnierpaarliste_detail,gw _meldeliste" fallen mir sofort ein, vielleicht genügt das schon. Falls nicht, kann ich später noch welche hinzufügen.

- Sprachpakete erstellen: Natürlich. Auch wenn es wahrscheinlich niemand in Englisch benutzen wird, tut es mir aber auch nicht weh, also Sprachen = "en,de".

Dann "speichern und schließen", und auf den grünen Haken am Ende der neuen Zeile "gw_turnierpaare" im Extension-Creator geklickt. Die Warnung bestätigt, und der Extension-Creator hat mir erstmal den Grundstock an Files in /system/modules/gw_turnierpaare/ erzeugt.

Und zwar:

- gwMeldeliste.php,gwTurnierpaarliste.php: meine Frontendklassen
- config/config.php und config/database.sql (Letzteres für meine SQL-Tabellenstruktur)
- dca/tl_gw_turnierpaare.php und dca/tl_gw_meldungen.php: Die DCA-Definitionen für meine Tabellen zur Bearbeitung im Backend
- languages/...: Die Sprachfiles in en und de-Variante
- templates/...: Die drei Frontend-Templates, die ich angegeben hatte

Im nächsten Schritt orientieren wir uns etwas und beginnen, die vorgegebenen Files zu modifizieren.

P.S.: Wenn die erfahrenen TL-Programmierer jetzt schon Gänsehaut haben: Sorry. Ich mache das zum ersten Mal, und stelle mich nicht künstlich dumm an. Ich versuche so zu schildern, wie ich als Einsteiger die Sachen sehe, was mich verwirrt usw...Bin für Verbesserungsvorschläge z.B. zu Namens-Schemata usw immer zu haben. Genauso freue ich mich über Aufklärung zu Dingen, die ich selbst nicht verstehe...Backendklassen, Backend-Templates, Frontend-Tabellen, Paket...

Schritt 3: Backend-Modul registrieren und SQL-Tabellenstruktur anlegen

Jetzt geht es also an die Files, die uns der Extension-Generator im letzten Schritt erzeugt hat. Ich vewende WinSCP unter Windows 7 und PSPad als Editor.

Zunächst will ich mein Backend-Modul in TYPOlight bekannt machen. Dafür öffne ich /system/modules/gw_turnierpaare/config/config.php .

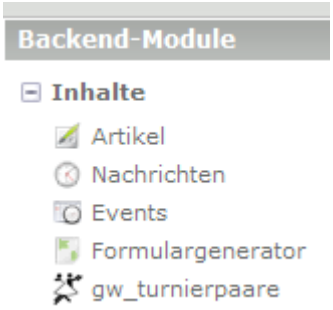
Im Skelett dieser Datei sind schon Abschnitte für die Eintragung von Backend-Modulen, Front-Modulen, Content Elementen, Hooks und noch viel mehr vorgesehen. Ich orientiere mich am cd-collection-Tutorial (http://dev.typolight.org/projects/ty...rialsExtension) und trage im Abschnitt für Backend-Module folgendes ein:

PHP-Code:

```
$GLOBALS['BE_MOD']['content']['gw_turnierpaare'] = array
(
    'tables' => array('tl_gw_turnierpaare','tl_gw_meldungen'),
    'icon'   => 'system/modules/gw_turnierpaare/icons/turnierpaare.png'
);
```

Hiermit registriere ich ein Modul mit dem Bezeichner "gw_turnierpaare", das sich auf die Datenbanktabellen "tl_gw_turnierpaare" und "tl_gw_meldungen" stützt (wie im Extension-Generator angegeben). In der Liste der Backend-Module (linke Spalte im Backend) soll ein Icon vor dem Bezeichner angezeigt werden, dessen Pfad ich unter 'icon' angegeben habe. Das Icon, was ich mir ausgesucht habe, ist an diesen Post angehängt. Ich habe mich für ein eigenes Unterverzeichnis für mögliche weitere Icons entschieden, und deshalb manuell das Unterverzeichnis "icons" angelegt und mein Icon dort hochgeladen. Im Backend-Modul soll der Berechtigte (also der Sportwart) die Daten Anlegen/Löschen/Ändern dürfen.

Ich speichere die Datei zunächst, und lade meine Backend-Ansicht (als Administrator!) neu. Nun sehe ich in der linken Spalte unter "Inhalte" den neuen Eintrag "gw_turnierpaare" mit meinem Icon. Ein Klick darauf führt leider noch zu einer Fehlermeldung, da die SQL-Tabellen tl_gw_turnierpaare und tl_gw_meldungen noch nicht angelegt sind.



Also, die Tabellen anlegen: Ich öffne /system/modules/gw_turnierpaare/config/database.sql , in der mir der Extension-Generator schon ein Skelett für meine beiden Tabellen vorgegeben hat. In beiden Tabellen sind id, pid, sorting und tstamp vorgegeben, sowie der primary key id und der key pid. Ich vermute, dass pid die "Parent ID" ist. Meine Turnierpaare haben keinen parent, darum lösche ich die Definition von "pid" und die Festlegung von "pid" als Key.

Ich füge meine restlichen Felder hinzu, ohne so recht zu wissen, mit welcher Syntax genau. Ich habe mal was von SQL92-Syntax gelesen, aber die kenne ich nicht. Gibt es BOOLEAN-Datentypen? Char vs. Varchar? Sicherheitshalber halte ich mich erstmal an die MySQL-Syntax. Im Endeffekt sieht der Abschnitt für tl_gw_turnierpaare in database.sql so aus:

Code:

```
CREATE TABLE `tl_gw_turnierpaare` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `sorting` int(10) unsigned NOT NULL default '0',
  `tstamp` int(10) unsigned NOT NULL default '0',
  `partnernachname` varchar(64) NOT NULL default '_',
  `partnervorname` varchar(64) NULL,
  `partnerinnachname` varchar(64) NULL,
  `partnerinvorname` varchar(64) NULL,
  `startgruppe` varchar(32) NOT NULL default '_',
  `startklasselatein` varchar(12) NULL,
  `startklassestandard` varchar(12) NULL,
  `aktiv` int(1) NOT NULL default '0',
  `aktivseit` int(4) NULL,
  `aktivbis` int(4) NULL,
  `password` varchar(32) NULL,
  `bild` varchar(255) NULL,
  `anschrift` text NULL,
  `zeigeanschrift` int(1) NOT NULL default '0',
  `telefon` varchar(32) NULL,
  `zeigetelefon` int(1) NOT NULL default '0',
  `fax` varchar(32) NULL,
  `zeigefax` int(1) NOT NULL default '0',
  `mobil` varchar(32) NULL,
  `zeigemobil` int(1) NOT NULL default '0',
  `email` varchar(128) NULL,
  `zeigeemail` int(1) NOT NULL default '0',
  `homepage` varchar(128) NULL,
  `zeigehomepage` int(1) NOT NULL default '0',
  `beschreibung` text NULL,
  PRIMARY KEY  (`id`),
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

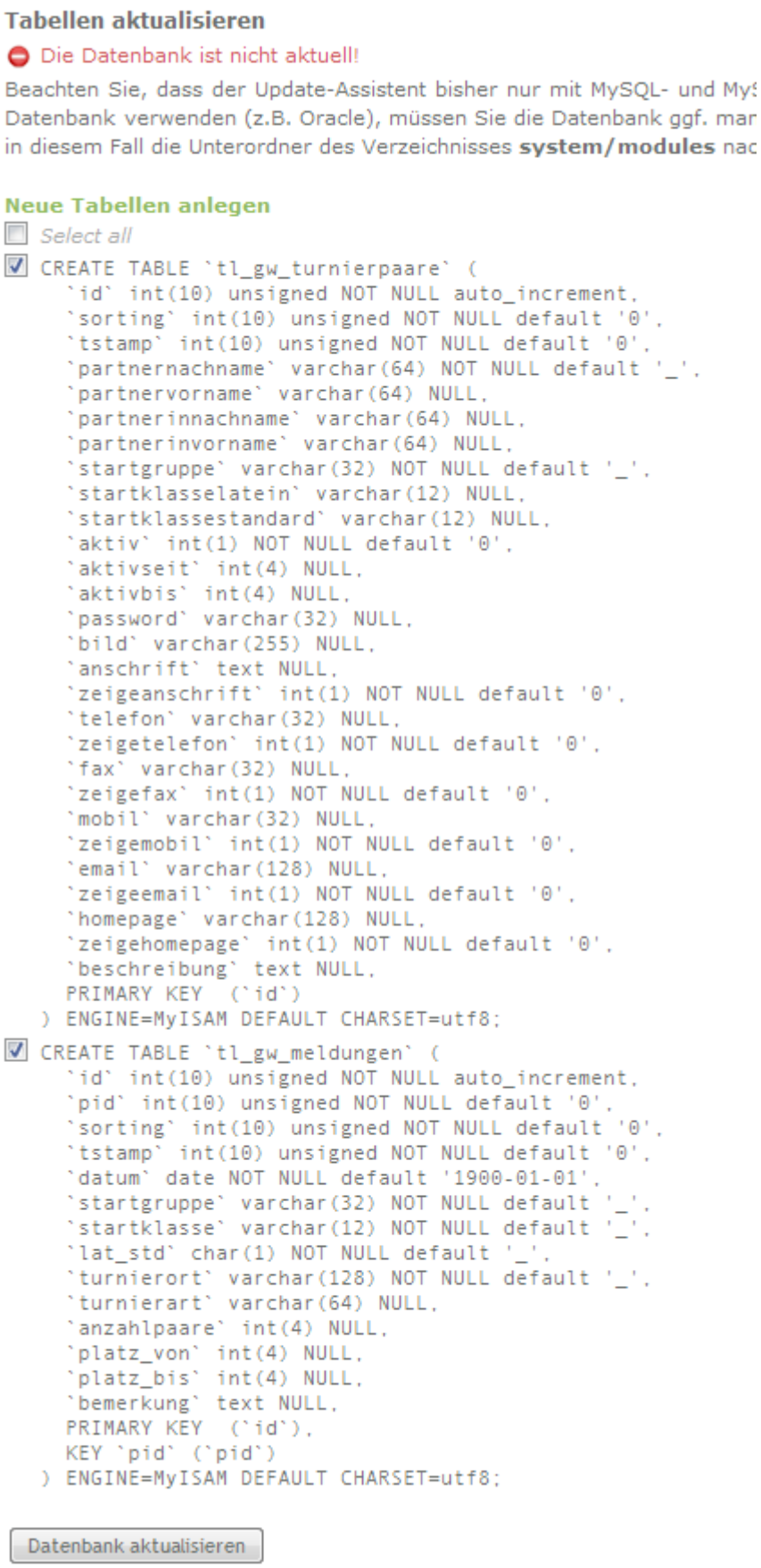
Man kann sich hier jetzt beliebige Gedanken über die Datenstruktur, Abstraktion, Normalisierung usw machen. Ich möchte es jetzt _so_ lösen  .

Meine Meldungen in tl_gw_meldungen sollen parents haben (nämlich die Turnierpaare), darum lasse ich das Skelett so, und erweitere um meine eigenen Felder. Für meinen Fall kommt das hier heraus:

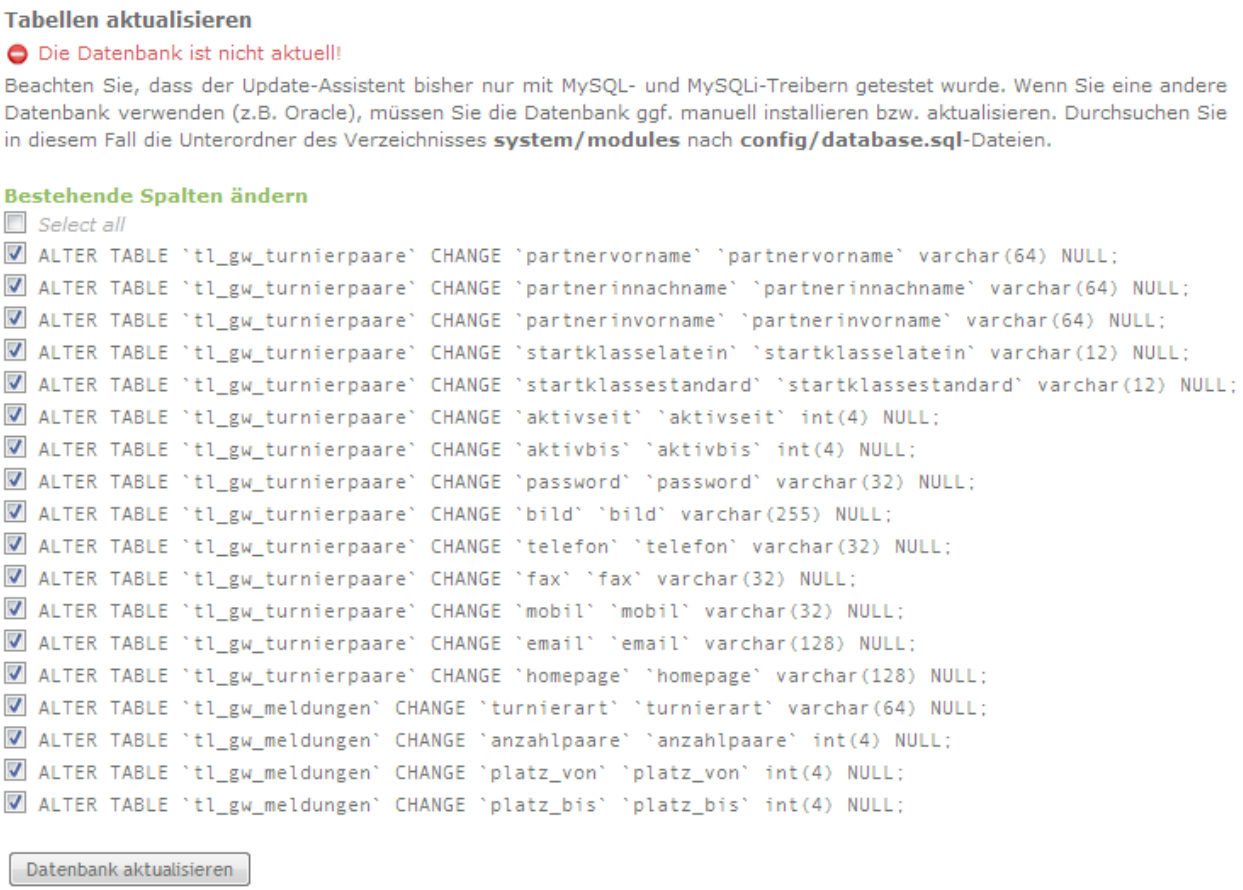
Code:

```
CREATE TABLE `tl_gw_meldungen` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `pid` int(10) unsigned NOT NULL default '0',
  `sorting` int(10) unsigned NOT NULL default '0',
  `tstamp` int(10) unsigned NOT NULL default '0',
  `datum` date NOT NULL default '1900-01-01',
  `startgruppe` varchar(32) NOT NULL default '_',
  `startklasse` varchar(12) NOT NULL default '_',
  `lat_std` char(1) NOT NULL default '_',
  `turnierort` varchar(128) NOT NULL default '_',
  `turnierart` varchar(64) NULL,
  `anzahlpaare` int(4) NULL,
  `platz_von` int(4) NULL,
  `platz_bis` int(4) NULL,
  `bemerkung` text NULL,
  PRIMARY KEY  (`id`),
  KEY `pid` (`pid`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Ich speichere database.sql und rufe /typolight/install.php in meiner TYPOlight-Installation auf. TYPOlight bemerkt, daß die Datenbankstruktur nicht mehr aktuell ist, und schlägt mir vor, meine Tabellen anzulegen. Ich bestätige das.



Was dann kommt, erstaunt mich dann aber doch etwas: Die Datenbankstruktur soll weiterhin nicht aktuell sein:



Auch wenn ich diese Vorschläge bestätige, ändert sich nichts. Ein Blick in die Datenbank zeigt aber, dass die Tabellen wie von mir gewünscht angelegt wurden. Irgendwie kommt TYPOlight dort ins Schleudern. Ein Klick auf gw_turnierpaare im Backend verläuft jetzt aber ohne Fehlermeldung, auch wenn in diesem Backend-Modul noch nichts "passiert".

Meine Frage an dieser Stelle also an die erfahrenen Entwickler: Was ist an meiner SQL-Definition "falsch", dass die Tabellen zwar richtig angelegt werden, das TYPOlight-Install-Tool aber damit nicht zurecht kommt?

Seitengestalter

Hallo, jeweils am Ende der Zeile das >NULL< durch >"< (das sind zwei einfache Striche, nicht der Doppelte - und ohne die spitzen Klammern) ersetzen.

---

dl1ely

leider Nein...
Das Install-Tool schlägt dann ein Anpassen der Datenbankstruktur z.B. wie folgt vor:

Code:

```
ALTER TABLE `tl_gw_turnierpaare` CHANGE `partnervorname` `partnervorname` varchar(64) '';
```

Wenn man das bestätigt, erntet man nur eine SQL-Fehlermeldung (von MySQL).

Code:

```
Fatal error: Uncaught exception Exception  with message Query error: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '''' at line 1 (ALTER TABLE `tl_gw_turnierpaare` CHANGE `partnervorname` `partnervorname` varchar(64) '';)
```

Noch jemand eine Idee?
Im o.g. CD-Collection-Tutorial wird es in einer Zeile der database.sql bei einem NULL-Feld auch so gemacht (also mit "NULL" am Ende der Zeile). Lasse ich "NULL" komplett weg, will mir das Installationstool weiterhin die Datenbankstruktur anpassen, obwohl sie eigentlich schon korrekt ist.

Seitengestalter

Sorry, mein Fehler. Muss heißen >default "<.
Meiner Erfahrung nach mag Tl kein varchar mit dem Default NULL

BugBuster

Siehe hier zum Thema wie erstelle ich die sql richtig:
http://www.typolight-community.de/showthread.php?t=41

Vereinfacht gesagt, vollständige MySQL Syntax.

---

dl1ely

OK, wir kommen der Sache näher, aber für mich bleibt es mysteriös...

- NOT NULL Felder sind kein Problem - Man gibt den Default an, und alles ist OK.
- varchar-Felder mit NULL werden geschluckt (also funktionieren), wenn man "default "" anhängt, also z.b.

Code:

```
`name` varchar(32) NULL default ''
```

- int- oder text-Felder mit NULL werden vom Install-Tool nicht geschluckt, wenn "default '0'" (für int) oder "default "" (für text) anhängt, also z.B.

Code:

```
`flag` int(1) default '0'
```

Das Install-Tool meint weiterhin, die Datenbankstruktur würde nicht stimmen, und lässt sich auch nicht davon abhalten.

Ich habe mal in das database.sql der Extension "twitterreader" geschaut, dort steht ein NON-NULL text-Feld als

Code:

```
`feld` text NULL,
```

eingetragen, und dort funktioniert das offensichtlich auch. Warum bei mir nicht? Help!

P.S.: Meine verbleibenden Problem-Felder sind:

Code:

```
ALTER TABLE `tl_gw_turnierpaare` CHANGE `aktivseit` `aktivseit` int(4) NULL default
'0000';
ALTER TABLE `tl_gw_turnierpaare` CHANGE `aktivbis` `aktivbis` int(4) NULL default '0000';
ALTER TABLE `tl_gw_turnierpaare` CHANGE `anschrift` `anschrift` text NULL default '';
ALTER TABLE `tl_gw_turnierpaare` CHANGE `beschreibung` `beschreibung` text NULL default
'';
ALTER TABLE `tl_gw_meldungen` CHANGE `bemerkung` `bemerkung` text NULL default '';
```

BugBuster

> Zitat von **Seitengestalter**
> *Meiner Erfahrung nach mag Tl kein varchar mit dem Default NULL*

Doch, sieht komisch aus, aber so gehts:

Code:

```
`varchar_null_demo` varchar(32) NULL default NULL,
```

Siehe Link von mir weiter oben.

BugBuster

Mal einfach gefragt, wenn ein Feld NULL sein darf, wozu dann ein Default?
Wenn ein Default eingetragen werden soll, wenn nichts übergeben wurde, dann muss NOT NULL definiert werden.
Textfelder dürfen kein Default haben, bei INT und NULL sieht nächsten Eintrag.

BugBuster

Code:

```
ALTER TABLE `tl_gw_turnierpaare` CHANGE `aktivseit` `aktivseit` int(4) NULL default NULL;
ALTER TABLE `tl_gw_turnierpaare` CHANGE `aktivbis` `aktivbis` int(4) NULL default NULL;
ALTER TABLE `tl_gw_turnierpaare` CHANGE `anschrift` `anschrift` text NULL;
ALTER TABLE `tl_gw_turnierpaare` CHANGE `beschreibung` `beschreibung` text NULL;
ALTER TABLE `tl_gw_meldungen` CHANGE `bemerkung` `bemerkung` text NULL;
```

---

dl1ely

Halleluja, es ist gelöst!

Leider gabs etwas zeitliche Überschneidung mit meinen Posts und denen von Glen. Wenn man genau nach http://www.typolight-community.de/showthread.php?t=41 arbeitet, dann geht es:

- NOT NULL-Felder mit default
- NULL-Felder mit "NULL default NULL"
- NULL-text-Felder ohne default!

---

dl1ely

> Zitat von **BugBuster**
> *Mal einfach gefragt, wenn ein Feld NULL sein darf, wozu dann ein Default?*

Einfache Antwort: Weil Roland mich drauf brachte und es (komischerweise) funktioniert. Ich hätte mir sowas nicht ausgedacht ;-). "NULL default NULL" ist aber auch eine gewisse Tautologie. Anyway, es funktioniert...und "NULL default NULL" gefällt mir auch besser, darum habe ich es jetzt darauf geändert.

---

dl1ely

Schritt 3b: SQL reloaded

Nach kleinem Kampf mit dem Install-Tool sieht das SQL für meine beiden Tabellen in database.sql nun so aus:

Code:

```
CREATE TABLE `tl_gw_turnierpaare` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `sorting` int(10) unsigned NOT NULL default '0',
  `tstamp` int(10) unsigned NOT NULL default '0',
  `partnernachname` varchar(64) NOT NULL default '',
  `partnervorname` varchar(64) NULL default NULL,
  `partnerinnachname` varchar(64) NULL default NULL,
  `partnerinvorname` varchar(64) NULL default NULL,
  `startgruppe` varchar(32) NOT NULL default '',
  `startklasselatein` varchar(12) NULL default NULL,
  `startklassestandard` varchar(12) NULL default NULL,
  `aktiv` int(1) NOT NULL default '0',
  `aktivseit` int(4) NULL default NULL,
  `aktivbis` int(4) NULL default NULL,
  `password` varchar(32) NULL default NULL,
  `bild` varchar(255) NULL default NULL,
  `anschrift` text NULL,
  `zeigeanschrift` int(1) NOT NULL default '0',
  `telefon` varchar(32) NULL default NULL,
  `zeigetelefon` int(1) NOT NULL default '0',
  `fax` varchar(32) NULL default NULL,
  `zeigefax` int(1) NOT NULL default '0',
  `mobil` varchar(32) NULL default NULL,
  `zeigemobil` int(1) NOT NULL default '0',
  `email` varchar(128) NULL default NULL,
  `zeigeemail` int(1) NOT NULL default '0',
  `homepage` varchar(128) NULL default NULL,
  `zeigehomepage` int(1) NOT NULL default '0',
  `beschreibung` text NULL,
  PRIMARY KEY  (`id`),
```

```
) ENGINE=MyISAM DEFAULT CHARSET=utf8;

CREATE TABLE `tl_gw_meldungen` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `pid` int(10) unsigned NOT NULL default '0',
  `sorting` int(10) unsigned NOT NULL default '0',
  `tstamp` int(10) unsigned NOT NULL default '0',
  `datum` date NOT NULL default '1900-01-01',
  `startgruppe` varchar(32) NOT NULL default '',
  `startklasse` varchar(12) NOT NULL default '',
  `lat_std` char(1) NOT NULL default '',
  `turnierort` varchar(128) NOT NULL default '',
  `turnierart` varchar(64) NULL default NULL,
  `anzahlpaare` int(4) NULL default NULL,
  `platz_von` int(4) NULL default NULL,
  `platz_bis` int(4) NULL default NULL,
  `bemerkung` text NULL,
  PRIMARY KEY  (`id`),
  KEY `pid` (`pid`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

und wir merken uns:

- NOT NULL-Felder mit default (logisch)
- NULL-"text"-Felder OHNE default
- Sonstige NULL-Felder mit "default NULL"
- default _KLEIN_ schreiben
- Und falls man da was verändert: _2_ Leerzeichen zwischen PRIMARY KEY und (`id`)

Und hier nochmal der Link zum Thread, der das erklärt:
http://www.typolight-community.de/showthread.php?t=41.

So, das Install-Tool ist zufrieden, die Datenbank-Tabellen angelegt - es kann weiter gehen!

Schritt 4: Wir wagen uns in das DCA-Land

Jetzt kommt es zu einem (vermutlich) harten Brocken. Mein backend-Modul wird links in der Navigation des Backends angezeigt, aber man kann noch keine Datensätze anlegen oder verändern. Dafür müssen wir einen passenden "DCA-Record" anlegen. Ich werde mich wieder vom CD-Collection-Tutorial und der Referenz zu den DCA-Records leiten lassen. Die Referenz ist schon mal erschlagend-beeindruckend.

Mithilfe der DCA-Records erstellt TYPOlight die Masken, mit denen man im Backend die Tabellen füllen, verändern und löschen kann. In /system/modules/gw_turnierpaare/dca/tl_gw_turnierpaare.php hat der Extension-Generator freundlicherweise schon ein Skelett für einen DCA-Record für die Tabelle tl_gw_turnierpaare angelegt.

PHP-Code:

```
$GLOBALS['TL_DCA']['tl_gw_turnierpaare'] = array
(

    // Config
    'config' => array
    (
        'dataContainer'               => 'Table',
        'enableVersioning'            => true
    ),
...
```

Der zweite Array-Key in $GLOBALS ist der Name unserer Tabelle. Im darauffolgenden mehrfach verschachtelten Array gibt es zunächst die "config"-Sektion. Hier wird zunächst festgehalten, dass es sich bei der Datenquelle um eine Tabelle handelt. Laut Referenz sind auch noch File und Folder vorgesehen. Sicherlich sind Tabellen der am häufigsten gebrauchte Datacontainer. enableVersioning erlaubt die Versionierung der Einträge - das ist OK und passt mir ins Konzept. Die Referenz verrät mir, daß ich eine "child Table" angeben kann. Da die Turniermeldungen Childs der Turnierpaare werden soll, ergänze ich also

PHP-Code:

```
        'ctable'                      => 'tl_gw_meldungen'
```

Die verbleibenden Optionen in "config" erscheinen mir nicht weiter von Bedeutung. Weiter geht es mit dem Abschnitt "list", und dort mit "sorting":

PHP-Code:

```
    // List
    'list' => array
    (
        'sorting' => array
        (
            'mode'                    => 1,
            'fields'                  => array(''),
            'flag'                    => 1
        ),
```

Sortierart und Sortierreihenfolge sind für mich erstmal ok, da ich gerne nach Nachnamen von Herrn und Dame sortieren würde, verändere ich die Zeile mit 'fields' auf:

PHP-Code:

```
            'fields'                    => array('partnernachname','partnerinnachnam
e'),
```

Als nächstes kommt ein Block "'label":

PHP-Code:

```
        'label' => array
        (
            'fields'                    => array(''),
            'format'                    => '%s'
        ),
```

Hier scheint es wohl darum zu gehen, was in der Liste der schon bestehenden Tabelleneinträge eingezeigt wird. Ich verändere die Zeilen auf

PHP-Code:

```
            'fields'                    => array('partnernachname','partnervorname',
'partnerinnachname','partnerinvorname','startgruppe','startklassestd','startklasse
lat'),
            'format'                    => '%s, %s und %s, %s - %s %s LAT / %s STD'
```

Etwas "domain-specific knowledge": Startgruppe ist im Prinzip die Altersklasse, Startklasse ist die Leistungsklasse (Die "Liga"), in der das Paar tanzt, und zwar unterschieden nach lateinamerikanischen und Standardtänzen. Diese Infos sind für den Sportwart interessant und sollten in der Übersichtsliste vorhanden sein. Die "%s" im format-String werden in der Reihenfolge mit Feldinhalten befüllt, wie wir sie obendrüber im Array angegeben haben. Der Aufbau des format-Strings sollte PHP- (oder C-)Programmierern bekannt sein.

Dann kommt ein Abschnitt "global_operations" und "operations", den ich aber gar nicht verändern will:

PHP-Code:

```
        'global_operations' => array
        (
            'all' => array
            (
                'label'                 => &$GLOBALS['TL_LANG']['MSC']['all'],
                'href'                  => 'act=select',
                'class'                 => 'header_edit_all',
                'attributes'            => 'onclick="Backend.getScrollOffset();"'
            )
        ),
        'operations' => array
        (
            'edit' => array
            (
                'label'                 => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['edit'],
                'href'                  => 'act=edit',
                'icon'                  => 'edit.gif'
            ),
            'copy' => array
            (
                'label'                 => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['copy'],
                'href'                  => 'act=copy',
                'icon'                  => 'copy.gif'
            ),
            'delete' => array
            (
```

```
                'label'                 => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['delete'],
                'href'                  => 'act=delete',
                'icon'                  => 'delete.gif',
                'attributes'            => 'onclick="if (!confirm(\'' . $GLOBAL
S['TL_LANG']['MSC']['deleteConfirm'] . '\')) return false; Backend.getScrollOffset
();"'
            ),
            'show' => array
            (
                'label'                 => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['show'],
                'href'                  => 'act=show',
                'icon'                  => 'show.gif'
            )
        )
    ),
```

Laut Referenz sollte "global_operations" ein Unterpunkt von "operations" sein, im Skelett-File des Extension-Generators stehen sie aber auf gleicher Ebene. Bin etwas verwundert, aber wird schon funktionieren.

Nächster Abschnitt im vorgegeben File sind "palettes" und "subpalettes". Leider stehen die nicht in der Referenz, und auch die Seite über "palettes" macht mich nicht so richtig schlauer.

PHP-Code:

```
    // Palettes
    'palettes' => array
    (
        '__selector__'                  => array(''),
        'default'                       => ''
    ),

    // Subpalettes
    'subpalettes' => array
    (
        ''                              => ''
    ),
```

Ein Blick ins CD-Collection-Tutorial verrät, dass man unter "default" die Felder angeben kann, die in Paletten sortiert werden sollen: Felder innerhalb der Palette mit Komma getrennt, Beginn einer neuen Palette durch ein Semikolon.
Da ich es zeitlich für diesen Post nicht schaffen werde, alle Felder meiner tl_gw_turnierpaare-Tabelle zu definieren, will ich zunächst nur die Namensfelder definieren, und zum Testen 2 Paletten benutzen. Ich editiere den "palettes"-Eintrag also in

PHP-Code:

```
    // Palettes
    'palettes' => array
    (
        '__selector__'                  => array(''),
        'default'                       => 'partnernachname,partnervorname;partnerin
nachname,partnerinvorname'
    ),
```

"Subpalettes" lässt mich weiterhin ratlos, also Finger weg davon.

Der letzte Teil der Skelett-Datei (und hier wird es richtig spannend!) ist das "fields"-Array:

PHP-Code:

```
    // Fields
    'fields' => array
    (
        '' => array
        (
            'label'                 => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare'][''],
            'exclude'               => true,
            'inputType'             => 'text',
            'eval'                  => array('mandatory'=>true, 'maxlength'=>255
)
        )
    )
);
```

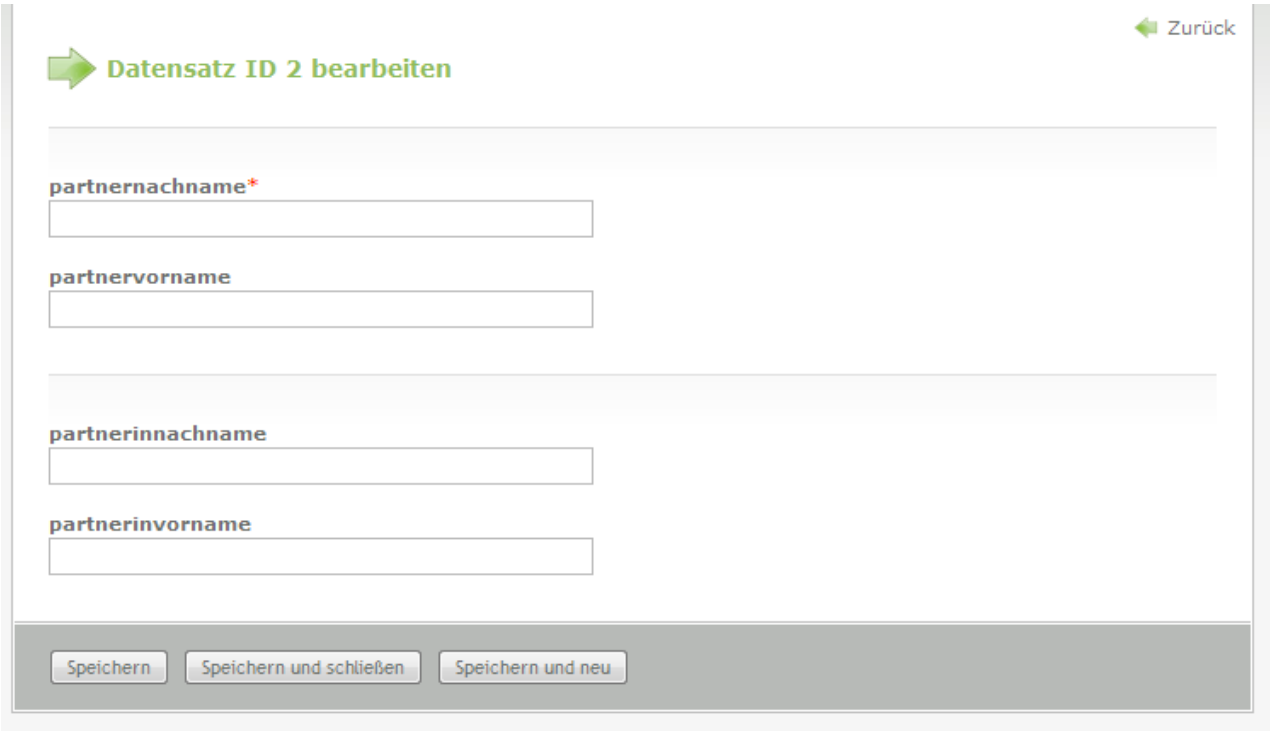
Ich halte mich erstmal an das Skelett, und füge nur die Feldnamen hinzu, und vervielfältige den Block auf insgesamt 4 Stück:

PHP-Code:

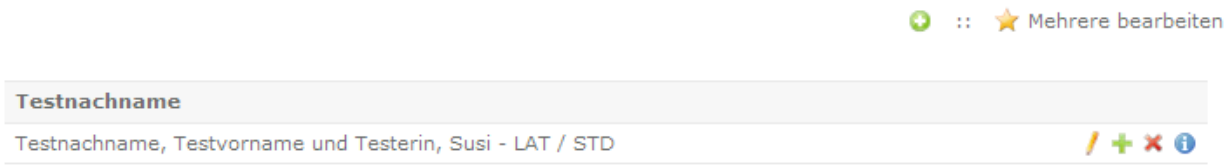
```
    // Fields
    'fields' => array
    (
        'partnernachname' => array
        (
            'label'                 => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnernachname'],
            'exclude'               => false,
            'inputType'             => 'text',
            'eval'                  => array('mandatory'=>true, 'maxlength'=>64)

        ),
        'partnervorname' => array
        (
            'label'                 => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnervorname'],
            'exclude'               => false,
            'inputType'             => 'text',
            'eval'                  => array('mandatory'=>false, 'maxlength'=>64
)
        ),
        'partnerinnachname' => array
        (
            'label'                 => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnerinnachname'],
            'exclude'               => false,
            'inputType'             => 'text',
            'eval'                  => array('mandatory'=>false, 'maxlength'=>64
)
        ),
        'partnerinvorname' => array
        (
            'label'                 => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnerinvorname'],
            'exclude'               => false,
            'inputType'             => 'text',
            'eval'                  => array('mandatory'=>false, 'maxlength'=>64
)
        )
    )
```

Die Labels müssen wir später noch in den Sprachfiles definieren, "exclude" = true bedeutet, dass nur Admins das Feld sehen können. Da später ein Nicht-Admin die Tebelle pflegen können soll, setze ich es also überall auf "false". Ich hoffe mein Gedankengang ist da richtig. Wir setzen für jedes Feld die Maximallänge auf 64 Zeichen, und nur der Partner-Nachname ist verpflichtend. Warum nicht auch Vorname und der Namen der Partnerin? Ich brauche für meine Anwendung EINE klitzekleine Ausnahme, in der ich gerne eine Mannschaft in die Startliste eintragen würde. Deren Name würde dann in 'partnernachname' stehen, die restlichen Felder wären leer.

Für den nächsten Post wird das alles noch verfeinert, weitere Optionen für die Felder hinzugefügt und vor allem alle Felder der Tabelle im DCA-Record definiert. Aber erstmal ein kleines, bescheidenes Zwischenergebnis zur Motivation:



Und man kann auch schon was eingeben:



Da Startgruppe und Klasse(n) noch nicht einzugeben sind, bleiben die in der Übersichtsliste noch leer. Aber: Grundlegend funktioniert das schonmal, und auch den Begriff "palette" habe ich jetzt (anhand des Screenshots) verstanden.

Im nächsten Post wird das alles erweitert und "poliert".

Zur Übersicht nochmal mein aktueller Stand der Datei /system/modules/gw_turnierpaare/dca/tl_gw_turnierpaare.php:

PHP-Code:

```
/**
 * Table tl_gw_turnierpaare
 */
$GLOBALS['TL_DCA']['tl_gw_turnierpaare'] = array
```

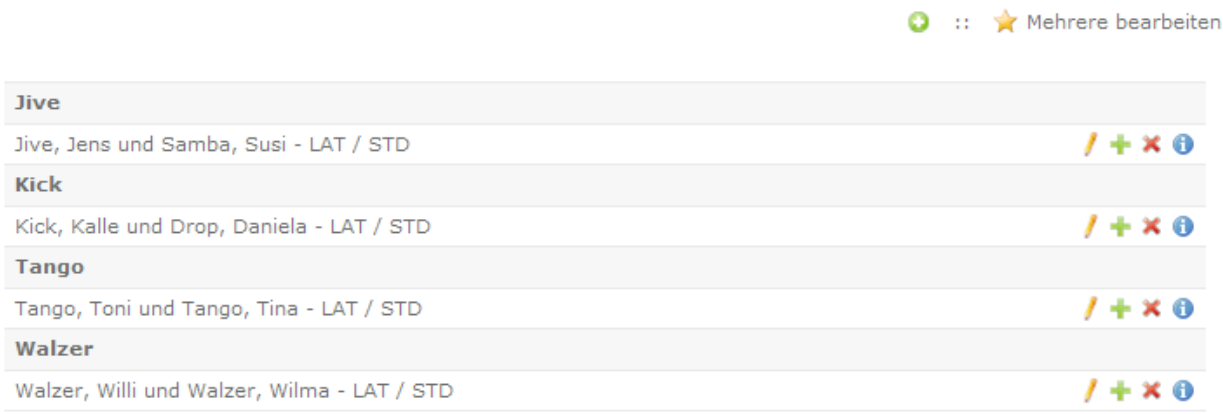
```
(

    // Config
    'config' => array
    (
        'dataContainer'               => 'Table',
        'enableVersioning'            => true,
        'ctable'                      => 'tl_gw_meldungen'
    ),

    // List
    'list' => array
    (
        'sorting' => array
        (
            'mode'                    => 1,
            'fields'                  => array('partnernachname','partnerinnachnam
e'),
            'flag'                    => 1
        ),
        'label' => array
        (
            'fields'                  => array('partnernachname','partnervorname',
'partnerinnachname','partnerinvorname','startgruppe','startklassestd','startklasse
lat'),
            'format'                  => '%s, %s und %s, %s - %s %s LAT / %s STD'
        ),
        'global_operations' => array
        (
            'all' => array
            (
                'label'               => &$GLOBALS['TL_LANG']['MSC']['all'],
                'href'                => 'act=select',
                'class'               => 'header_edit_all',
                'attributes'          => 'onclick="Backend.getScrollOffset();"'
            )
        ),
        'operations' => array
        (
            'edit' => array
            (
                'label'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['edit'],
                'href'                => 'act=edit',
                'icon'                => 'edit.gif'
            ),
            'copy' => array
            (
                'label'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['copy'],
                'href'                => 'act=copy',
                'icon'                => 'copy.gif'
            ),
            'delete' => array
            (
                'label'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['delete'],
                'href'                => 'act=delete',
                'icon'                => 'delete.gif',
                'attributes'          => 'onclick="if (!confirm(\'' . $GLOBAL
S['TL_LANG']['MSC']['deleteConfirm'] . '\')) return false; Backend.getScrollOffset
();"'
            ),
            'show' => array
```

```
            (
                'label'                 => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['show'],
                'href'                  => 'act=show',
                'icon'                  => 'show.gif'
            )
        )
    ),

    // Palettes
    'palettes' => array
    (
        '__selector__'                  => array(''),
        'default'                       => 'partnernachname,partnervorname;partnerin
nachname,partnerinvorname'
    ),

    // Subpalettes
    'subpalettes' => array
    (
        ''                              => ''
    ),

    // Fields
    'fields' => array
    (
        'partnernachname' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnernachname'],
            'exclude'                   => false,
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>true, 'maxlength'=>64)

        ),
        'partnervorname' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnervorname'],
            'exclude'                   => false,
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'maxlength'=>64
)
        ),
        'partnerinnachname' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnerinnachname'],
            'exclude'                   => false,
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'maxlength'=>64
)
        ),
        'partnerinvorname' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnerinvorname'],
            'exclude'                   => false,
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'maxlength'=>64
)
        )
    )
```

```
);
```

Schritt 4b: Verwirrung im DCA-Land

Nachdem ich einige Testpaare in meine "Minimalmaske" eingetragen habe, stelle ich fest, dass es nicht ganz so aussieht, wie ich es gerne hätte.



Für jeden Herrennachnamen gibt es eine eigene Gruppenüberschrift. Das ist irgendwie suboptimal, und verschwendet Platz. Ich hätte gerne keine Gruppenüberschriften, oder nur "A", "B", "C", usw...

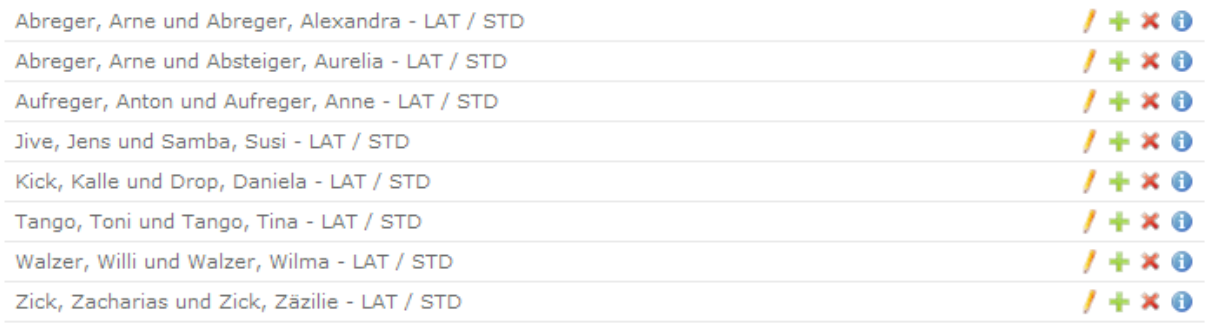
Ich vermute, dass das mit dem Eintrag ['list']['sorting']['flag'] zusammenhängt, den ich auf "1" hatte, laut Referenz "Sort by initial letter ascending":

PHP-Code:

```
    // List
    'list' => array
    (
        'sorting' => array
        (
            'mode'                    => 1,
            'fields'                  => array('partnernachname', 'partnervorname'
, 'partnerinnachname', 'partnerinvorname'),
            'flag'                    => 1
        ),
```

Ich spiele also etwas mit 'flag' herum, und stelle fest: Irgendwie beeinflusst das garnix. 2 = "Sort by initial letter descending", 3 = "Sort by initial two letters ascending", 4 = "Sort by initial two letters descending", 11 = "Sort ascending" oder 12 = "Sort descending" machen in meiner Auflistung nirgendwo irgendeinen Unterschied. Immer wird stur "ascending" in der Reihenfolge meiner Sortierfelder sortiert, und für jeden "Unique" Herrennachnamen gibt es eine Gruppenüberschrift.

In meiner Verzweifelung setze ich ['list']['sorting']['mode'] auf 0, laut Referenz "Records are not sorted". Das Ergebnis sieht so aus:



Immerhin die Gruppenüberschriften weg...und stur "ascending" nach meinen Sortierfeldern sortiert. Fast schon unnötig zu erwähnen, dass 'flag' auch hier keine Wirkung zu haben scheint.

Um das Sortieren vielleicht "von Hand" steuern zu können, füge ich streng nach Referenz die Zeile

PHP-Code:

```
            'panelLayout'             => 'search,sort,filter'
```

hinzu in der Erwartung, dass mir dann in der Übersicht die entsprechenden Optionen angeboten werden. Leider - nichts. Die Übersichtsliste der Turnierpaare verändert sich überhaupt nicht.

Meine ['list']['sorting']-Sektion sieht nun so aus:

PHP-Code:

```
        'sorting' => array
        (
            'mode'                    => 0,
            'fields'                  => array('partnernachname', 'partnervorname'
, 'partnerinnachname', 'partnerinvorname'),
            'flag'                    => 1,
            'panelLayout'             => 'search,sort,filter'
        ),
```

Jemand eine Ahnung, warum das Verhalten so ist, bzw. warum mich die Referenz für die DCA-Records so im Stich lässt?

Danke...

Schritt 4c: Leichte Entwirrung im DCA-Land

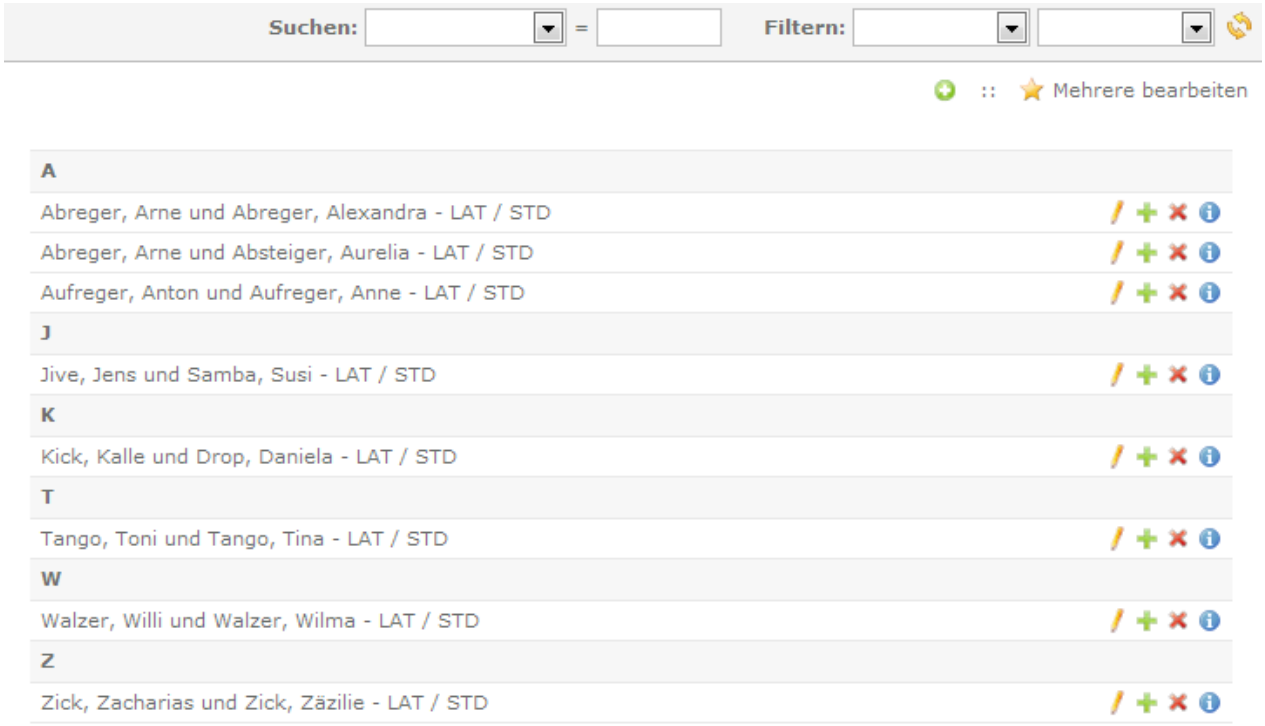
Lösung gefunden! Bei den einzelnen 'field'-Beschreibungen muss noch die Freigabe zum Sortieren, Filtern und Suchen gegeben werden. Das sieht jetzt so aus:

PHP-Code:

```
    // Fields
    'fields' => array
    (
        'partnernachname' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnernachname'],
            'exclude'                 => false,
            'inputType'               => 'text',
            'search'                  => true,
```

```
            'sorting'                   => true,
            'filter'                    => true,
            'flag'                      => 1,
            'eval'                      => array('mandatory'=>true, 'maxlength'=>64)

        ),
        'partnervorname' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnervorname'],
            'exclude'                   => false,
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'maxlength'=>64
)
        ),
        'partnerinnachname' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnerinnachname'],
            'exclude'                   => false,
            'inputType'                 => 'text',
            'search'                    => true,
            'sorting'                   => true,
            'filter'                    => true,
            'flag'                      => 1,
            'eval'                      => array('mandatory'=>false, 'maxlength'=>64
)
        ),
        'partnerinvorname' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnerinvorname'],
            'exclude'                   => false,
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'maxlength'=>64
)
        )
    )
```

Mit Ergebnis:



Schon besser, auch wenn die Dropdown-Liste hinter "Suchen:" noch leer ist. Vielleicht liegt das an den noch fehlenden Feld-Labels in den Sprachdateien. Nur warum man 'flag' bei den einzelnen Fields und nochmal global angeben muss, das will ich noch nicht verstehen...

Ergänzung: Und wenn ich ['sorting']['mode'] auf 2 setze, dann kann ich sogar mein Sortierfeld auswählen....sehr schön...

Schritt 4d: DCA-Polishing

Nachdem also die leichten Verwirrungen rund um den DCA-Record beseitigt sind, geht es weiter damit, die Backend-"Maske" für die tl_gw_turnierpaare-Tabelle zu definieren und zu "polieren".

Zu jedem Feld lege ich einen Verweis auf den Erklärungs-Text an, der unter dem Eingabefeld angezeigt wird, z.B. für das 'partnernachname'-Feld im Abschnitt ['fields']['partnernachname']:

PHP-Code:

```
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnernachname_explanation'],
```

Der entsprechende Text muss in den Sprachfiles natürlich noch eingetragen werden - später.

Außerdem ergänze ich den 'eval'-Wert meiner bisherigen 4 Eingabefelder um den Wert 'minlength' => 1, um bei den namen eine Mindestlänge zu erzwingen (Beim Wert 1 wahrscheinlich überflüssig, aber egal).

Bei dem Nachnamen des Partners und der Partnerin ergänze ich außerdem 'tl_class' => 'w50'. Das sorgt dafür, dass zwei Felder nebeneinander dargestellt werden. Das Feld mit der w50-Klasse links, das darauffolgende rechts. Dadurch werden Nachname und Vorname jeder Person nebeneinander in einer Zeile dargestellt.

Meine Einstellungen für das "partnernachname"-Feld sehen jetzt so aus:

PHP-Code:

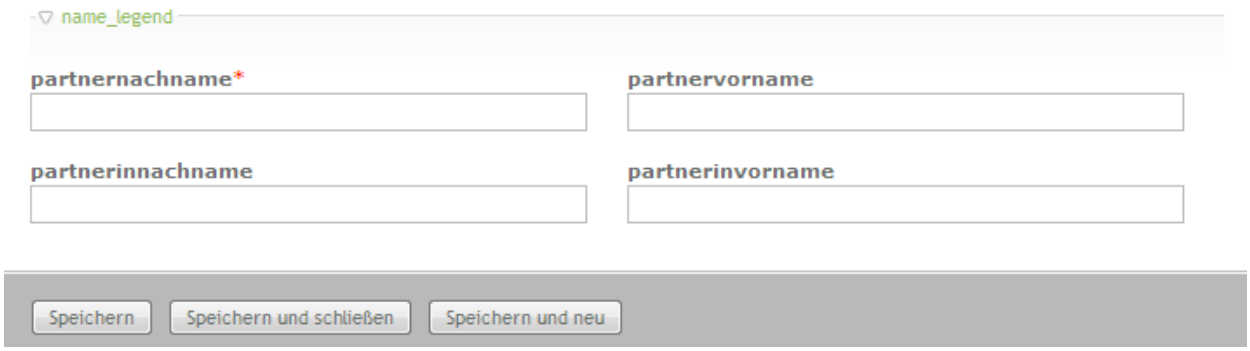
```
    // Fields
    'fields' => array
    (
        'partnernachname' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnernachname'],
            'explanation'             => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnernachname_explanation'],
            'exclude'                 => false,
            'inputType'               => 'text',
            'search'                  => true,
            'sorting'                 => true,
            'filter'                  => true,
            'flag'                    => 1,
            'eval'                    => array('mandatory'=>true, 'minlength' => 1
, 'maxlength'=>64, 'tl_class' => 'w50')
        ),
```

Wo ich gerade noch optisch etwas aufräume, baue ich den Eintrag 'default' unter 'palettes' so um:

PHP-Code:

```
    // Palettes
    'palettes' => array
    (
        '__selector__'                => array(''),
        'default'                     => '{name_legend},partnernachname,partnervor
name,partnerinnachname,partnerinvorname;'
    ),
```

{name_legend} legt die Überschrift für die "Palette" fest (also die Felder bis zum nächsten Semikolon). Der Wert muss später im Sprachenfile definiert werden. Ich habe nun alle 4 Textfelder in einer Palette. Optisches Ergebnis:



Wenn man sich "vernünftige" Überschriften aus dem Sprachfile dazu vorstellt, schon mal ganz OK :-).

Nun geht es um die noch fehlenden Tabellenfelder.

Zunächst kommt "startgruppe", das bezeichnet die Altersklasse des Paars. Das Feld soll mandatory sein, aber auch eine "leere Option" erlauben. Ich will als Ausnahme auch eine Mannschaft in die Paarliste eingeben können, und Mannschaften haben keine Altersklasse. Mein Code für das field sieht so aus:

PHP-Code:

```
        'startgruppe' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']
```

```
['tl_gw_turnierpaare']['startgruppe'],
            'explanation'             => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['startgruppe_explanation'],
            'exclude'                 => false,
            'inputType'               => 'select',
            'options'                 => array('KIN I','KIN II', 'JUN I', 'JUN II'
, 'JUG', 'HGR', 'HGR II', 'SEN I', 'SEN II', 'SEN III', 'SEN IV'),
            'eval'                    => array('mandatory'=>true, 'includeBlankOpt
ion' => true)
        ),
```

Ich wähle also ein "select", also eine Drop-Down-Box. In "options" liste ich die möglichen Altersgruppen auf. im "eval"-Bereich gebe ich noch an, dass eine leere Option hinzugefügt werden soll.

Dann kommen startklasselatein und startklassestandard. Inhaltlich kann in beiden Feldern dasselbe drinstehen, darum ist es fast nur Copy&Paste für das zweite Feld. Die Definition sieht so aus:

PHP-Code:

```
        'startklasselatein' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['startklasselatein'],
            'explanation'             => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['startklasselatein_explanation'],
            'exclude'                 => false,
            'inputType'               => 'select',
            'options'                 => array('E','D', 'C', 'B', 'A', 'S', 'PRO',
'LL', 'OL', 'RL', '2. BL', '1. BL'),
            'eval'                    => array('mandatory'=>true, 'includeBlankOpt
ion' => true, 'tl_class' => 'w50')
        ),
```

Auch hier wieder eine Drop-Down-Box mit Optionen und Möglichkeit der "leeren Option". Durch tl_class => w50 wird die Drop-Down-Box nach links gerückt, so dass rechts daneben noch die gleichartige Box für startklassestandard passt. Die hat natürlich KEIN tl_class => w50!

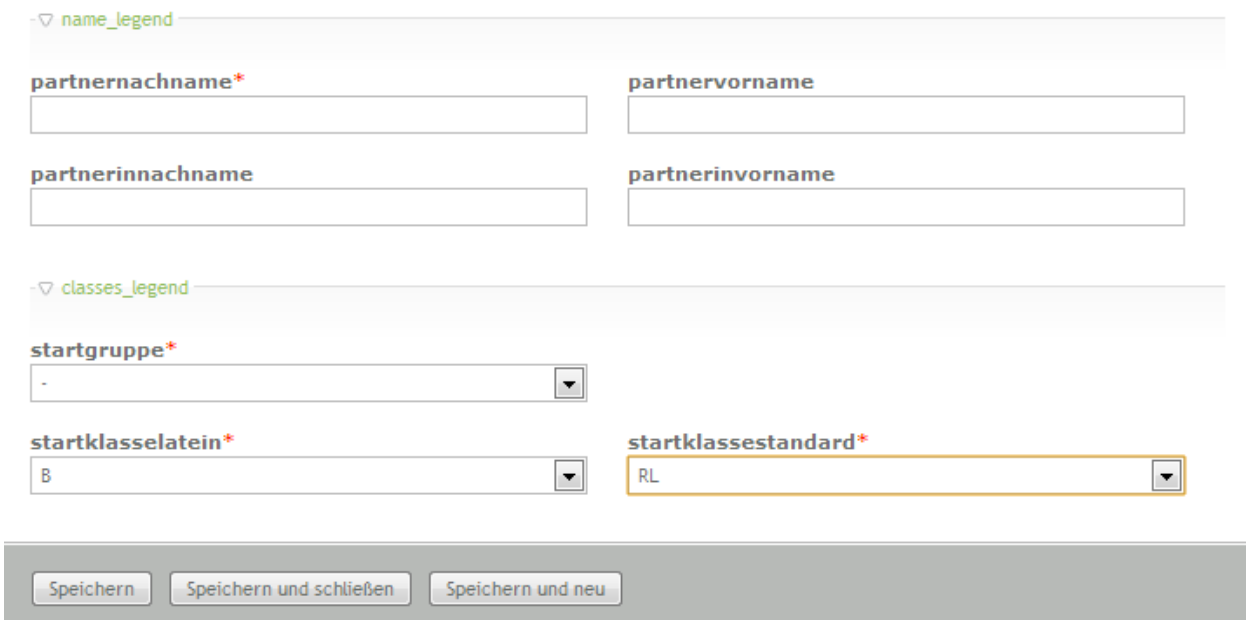
Eigentlich müsste ich prüfen, dass entweder in startklasselatein oder startklassestandard ein Wert ausgewählt ist (also nicht in beiden Feldern die leere Option gewählt wurde), aber das bürde ich zunächst mal dem User auf, vielleicht ergänze ich hier später eine Validation durch einen Hook.

Zur Motivation will ich meine drei neuen Felder auch in im backend sehen, dazu muss ich sie zur Liste der Paletten hinzufügen. Ich packe sie in eine eigene Palette mit Überschrift.

PHP-Code:

```
    // Palettes
    'palettes' => array
    (
        '__selector__'                => array(''),
        'default'                     => '{name_legend},partnernachname,partnervor
name,partnerinnachname,partnerinvorname;
{classes_legend},startgruppe,startklasselatein,startklassestandard'
    ),
```

Ergebnis:



Weiter geht, jetzt folgen die Felder aktiv, aktivseit und aktivbis.

PHP-Code:

```
        'aktiv' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['aktiv'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['aktiv_explanation'],
            'exclude'                   => false,
            'inputType'                 => 'checkbox',
            'eval'                      => array('mandatory'=>true, 'isBoolean' => t
rue)
        ),
```

Aktiv wird eine Checkbox. Ich weiß zwar nicht, was es für eine Bedeutung hat, aber da eine CheckBox immer "Boolean" ist, setze ich in 'eval' isBoolean => true.

PHP-Code:

```
        'aktivseit' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['aktivseit'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['aktivseit_explanation'],
            'exclude'                   => false,
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'minlength' =>
4, 'maxlength' => 4, 'rgxp' => 'digit', 'tl_class' => 'w50')
        ),
        'aktivbis' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['aktivbis'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['aktivbis_explanation'],
            'exclude'                   => false,
```

```
            'inputType'                => 'text',
            'eval'                     => array('mandatory'=>false, 'minlength' =>
4, 'maxlength' => 4, 'rgxp' => 'digit')
        ),
```

aktivseit und aktivbis sollen nur eine Jahreszahl enthalten. Darum setze ich Minimal- und Maximallänge auf 4 und lasse durch 'rgxp' nur Zahlen zu. Man hätte auch eine Dropdown-Box mit Jahreszahlen drin nehmen können. Ich denke beides hat Vor- und Nachteile. Sich durch DropDown-Boxen zu scrollen, die bei "1900" anfangen, wenn man nach "2004" will, ist auch kein Vergnügen. Das erste Feld setze ich mit tl_class => w50 nach links, um das zweite Feld daneben darstellen zu können.

Ich ergänze die Paletten-Definition um

PHP-Code:

```
{aktiv_legend:hide},aktiv,aktivseit,aktivbis;
```

Da die Felder nicht so oft editiert werden, schließe ich die Palette defaultmäßig.

Nun kommt schon ein kleiner Sonderfall: password. Dies soll das Paar-Passwort sein, was zum Eintragen von Turnierergebnissen oder geänderten persönlichen Daten im Frontend dient. Ich will das nur in eigenen PHP-Skripten nutzen, von daher habe ich hier alle Freiheiten, wie ich das realisiere.

Ich möchte gerne, dass der Sportwart in diesem Feld ein Klartextpasswort eingeben kann. In die Datenbank soll aber nur der MD5-Hash des Passworts gelangen. Momentan plane ich, dass in dem Textfeld einfach der MD5-Hash angezeigt wird, wenn man aber etwas in dieses Feld eingibt, dass es dann aber durch einen Hook in den Hash umgewandelt wird, bevor es in der Datenbank gespeichert wird. Um die Realisation kümmere ich mich später. Erstmal soll es ein ganz normales Text-Feld sein:

PHP-Code:

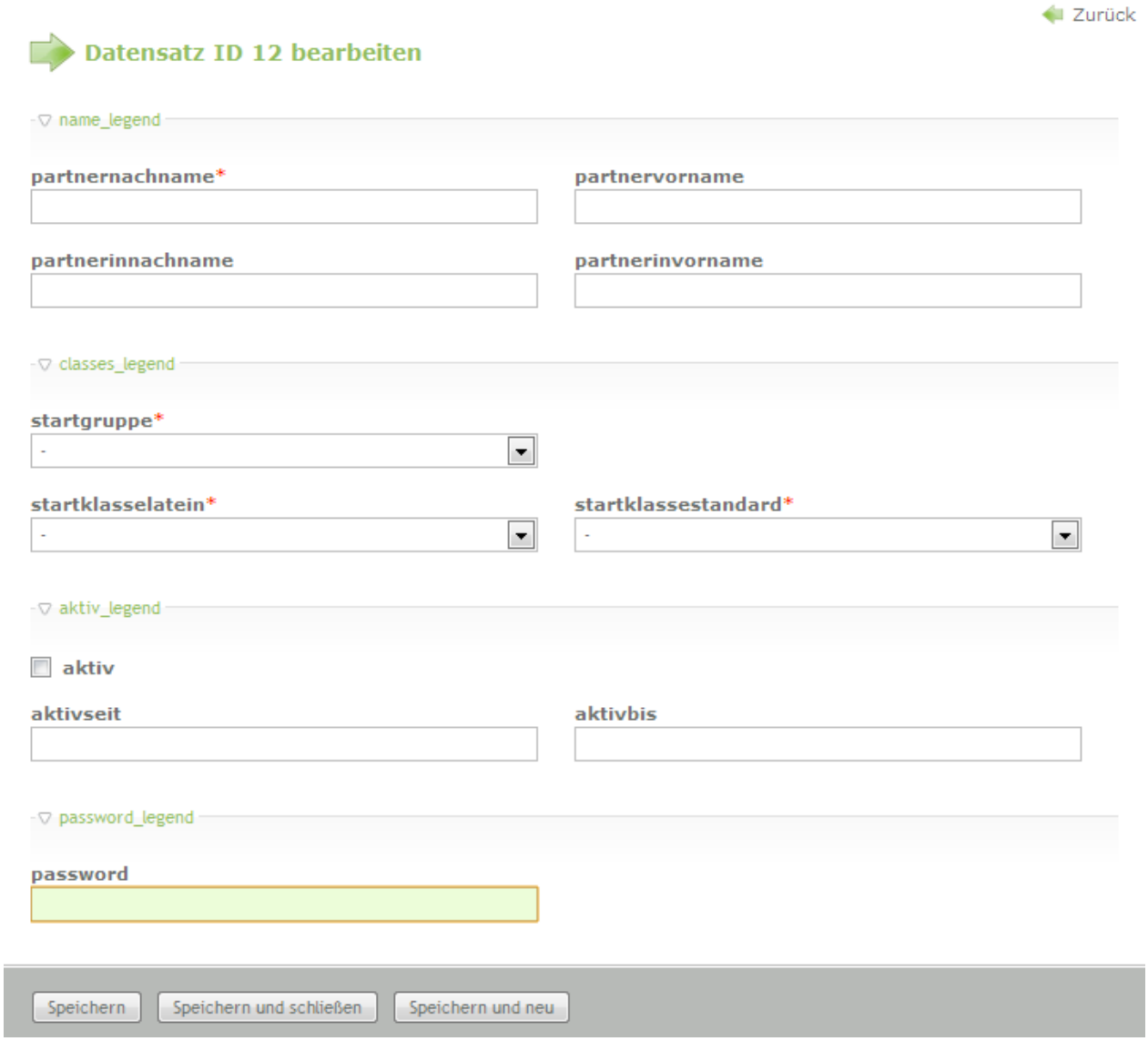
```
        'password' => array
        (
            'label'                    => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['password'],
            'explanation'              => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['password_explanation'],
            'exclude'                  => false,
            'inputType'                => 'text',
            'eval'                     => array('mandatory'=>false, 'minlength' =>
1, 'maxlength' => 64)
        )
```

Dieses Feld soll (alleine) in einer eigenen Palette stehen, darum ergänze ich die Palettendefinition um:

PHP-Code:

```
{password_legend:hide},password;
```

Zwischenstand der Backend-Maske:



Und nochmal die gesamte Paletten-Definition:

PHP-Code:

```
    // Palettes
    'palettes' => array
    (
        '__selector__'                => array(''),
        'default'                     => '{name_legend},partnernachname,partnervor
name,partnerinnachname,partnerinvorname;
{classes_legend},startgruppe,startklasselatein,startklassestandard;
{aktiv_legend:hide},aktiv,aktivseit,aktivbis;{password_legend:hide},password;'
    ),
```

Leider ist damit schon wieder das Ende meiner zur Verfügung stehenden Zeit erreicht (Sorry, wenn es zu langsam voran geht). Bald geht es weiter.

Schritt 4e: Ein Bug, ein Bug!

Nachdem man jetzt Startklassen eingeben, entdecke ich natürlich gleich einen Bug, und zwar in ['list']['label'] ['fields'].

VORHER (falsch):

PHP-Code:

```
            'fields'                  => array('partnernachname','partnervorname',
'partnerinnachname','partnerinvorname','startgruppe','startklassestd','startklasse
lat'),
```

NACHHER (richtig):

PHP-Code:

```
            'fields'                  => array('partnernachname','partnervorname',
'partnerinnachname','partnerinvorname','startgruppe','startklassestandard','startk
lasselatein'),
```

D.h. die beiden Felder für die Startklasse Standard und Latein waren falsch benannt. Sorry.

Und weiter geht es mit Bugs:
Wenn ich bei Feldern, die eine "leere Option" zulassen, trotzdem in 'eval' mandatory => true fordere, kann die leere Option nicht ausgewählt werden. Gut, irgendwie auch logisch. Aus diesem Grund setze ich bei startgruppe, startklasselatein und startklassestandard "'mandatory' => false", um auch die leere Option zuzulassen.

Und letzter kleiner Fehler:
Im format-String für die Zeilen in der Übersicht der Turnierpaare ['list']['label']['format'] ist LAT und STD vertauscht, ich drehe das um. Neu:

PHP-Code:

```
            'format'                  => '%s, %s und %s, %s - %s %s STD / %s LAT'
```

Schritt 4f: Nochmal anders

Tjaja, wie das bei so einem Tagebuch im Gegensatz zum "durchgeplanten und polierten" Tutorial so ist: Ich habe mir nochmal was anders überlegt.

Ich habe mich entschlossen, die Felder startklasselatein und startklassestandard doch mandatory zu machen, aber als Option eine Leer-Option "-" hinzuzufügen. So ist der Benutzer gezwungen, explizit anzugeben, dass ein Paar keine Startklasse in einer der beiden Sektionen hat, und in meiner Paarübersicht sieht es besser aus, wenn z.B. vor "LAT" noch ein Strich steht, statt einfach garnichts.

Bei beiden Feldern sieht der Eintrag in ['fields'] jetzt also so aus:

PHP-Code:

```
            'options'                 => array('-', 'E','D', 'C', 'B', 'A', 'S', '
PRO', 'LL', 'OL', 'RL', '2. BL', '1. BL'),
            'eval'                    => array('mandatory'=>true)
```

Dann habe ich noch Entdeckt, dass man eine Checkbox nicht mandatory machen darf, weil dann MUSS sie nämlich angehakt werden. Ist irgendwie suboptimal. Also nochmal den Eintrag ['fields']['aktiv']['eval'] geändert auf:

PHP-Code:

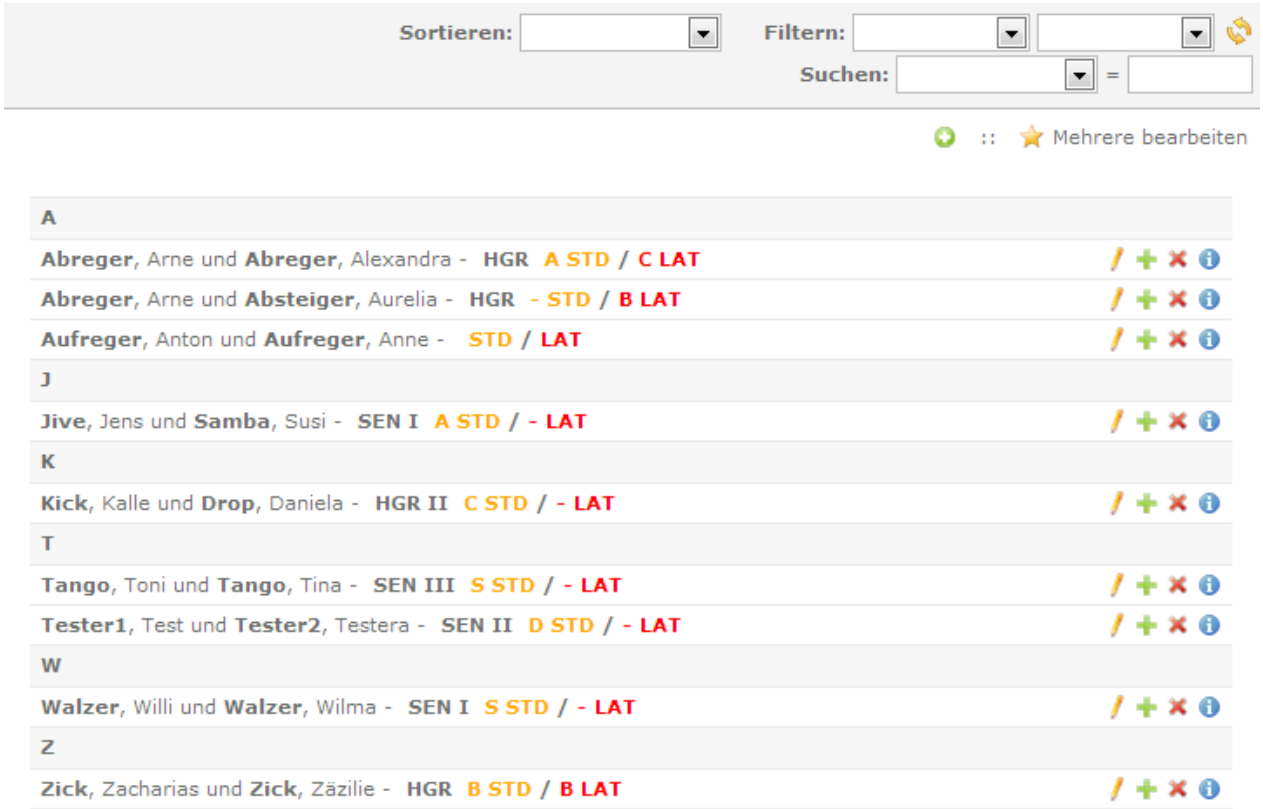
```
            'eval'                    => array('mandatory'=>false, 'isBoolean' =>
true)
```

Und schließlich habe ich den Format-String für die Turnierpaar-Übersicht nochmal überarbeitet. Damit die relevantesten Elemente hervorstechen gebe ich die Nachnamen der Partner fett aus, ebenso die Startgruppe. Die Startklassen zusätzlich in orange (Standard) und rot (Latein). Die Startpässe der Paare in der jeweiligen Sektion haben die gleichen Farben, so dass dies für den Eingeweihten eine natürliche Assoziation ist. ['list']['label']['format'] lautet jetzt:

PHP-Code:

```
            'format'                    => '<span style="font-weight: bold;">
%s</span>, %s und <span style="font-weight: bold;">%s</span>, %s - <span style="fo
nt-weight: bold; margin-left: 5px">%s <span style="color: orange; margin-left: 5px
;">%s STD</span> / <span style="color: red;">%s LAT</span></span>'
```

Und sieht so aus:



Aber im nächsten Post geht es endlich mit den restlichen Feldern der tl_gw_turnierpaare-Tabelle weiter.

Schritt 4g: DCA - Almost there

Schritt 4 scheint kein Ende zu nehmen. Wie erwartet erweist sich das Thema "DCA" als harter Brocken.

Zunächst geht es weiter mit den restlichen Feldern der tl_gw_turnierpaare-Tabelle.
Für Anschrift, Telefonnummer, Fax, Mobilnummer, Email-Adresse und Homepage gibt es jeweils ein Flag, ob es im öffentlichen Profil angezeigt werden soll. Ist es nicht gesetzt, sind die Daten nur im Backend sichtbar. So kann der Sportwart das Paar evtl. erreichen, falls notwendig.
Statt die Anschrift in Straße, PLZ, Ort, usw. aufzusplitten, habe ich hierfür eine Textarea vorgesehen. Im DCA sieht das so aus:

PHP-Code:

```
        'anschrift' => array
        (
```

```
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['anschrift'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['anschrift_explanation'],
            'inputType'                 => 'textarea',
            'eval'                      => array('mandatory'=>false, 'cols' => 40, '
rows' => 5)
        ),
```

Alles wie gehabt, zusätzlich geben cols und rows die Spalten und Zeilen des Eingabebereichs an.

Die Definition für das Flag, ob die Anschrift öffentlich angezeigt werden soll, ist so wie beim Feld "aktiv":

PHP-Code:

```
        'zeigeanschrift' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['zeigeanschrift'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['zeigeanschrift_explanation'],
            'inputType'                 => 'checkbox',
            'eval'                      => array('mandatory'=>false, 'isBoolean' =>
true)
        ),
```

Nicht Besonderes!

Die Felder für Telefon, Fax, Mobilnummer, EMail und Homepage sind jeweils Textfelder, denen ich je nach Art die passende Regular Expression zur Überprüfung der Inhalte zuweise. Zusätzlich hat jedes Feld die "Anzeigen"-Checkbox:

PHP-Code:

```
        'telefon' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['telefon'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['telefon_explanation'],
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'maxlength' =>
32, 'rgxp' => 'phone')
        ),
        'zeigetelefon' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['zeigetelefon'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['zeigetelefon_explanation'],
            'inputType'                 => 'checkbox',
            'eval'                      => array('mandatory'=>false, 'isBoolean' =>
true, 'tl_class' => 'clr m12 w50')
        ),
        'fax' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['fax'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['fax_explanation'],
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'maxlength' =>
```

```
32, 'rgxp' => 'phone')
        ),
        'zeigefax' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['zeigefax'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['zeigefax_explanation'],
            'inputType'                 => 'checkbox',
            'eval'                      => array('mandatory'=>false, 'isBoolean' =>
true, 'tl_class' => 'clr m12 w50')
        ),
        'mobil' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['mobil'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['mobil_explanation'],
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'maxlength' =>
32, 'rgxp' => 'phone')
        ),
        'zeigemobil' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['zeigemobil'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['zeigemobil_explanation'],
            'inputType'                 => 'checkbox',
            'eval'                      => array('mandatory'=>false, 'isBoolean' =>
true, 'tl_class' => 'clr m12 w50')
        ),
        'email' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['email'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['email_explanation'],
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'maxlength' =>
32, 'rgxp' => 'email')
        ),
        'zeigeemail' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['zeigeemail'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['zeigeemail_explanation'],
            'inputType'                 => 'checkbox',
            'eval'                      => array('mandatory'=>false, 'isBoolean' =>
true, 'tl_class' => 'clr m12 w50')
        ),
        'homepage' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['homepage'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['homepage_explanation'],
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'maxlength' =>
32, 'rgxp' => 'url')
        ),
        'zeigehomepage' => array
        (
```

```
            'label'                     => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['zeigehomepage'],
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['zeigehomepage_explanation'],
            'inputType'                 => 'checkbox',
            'eval'                      => array('mandatory'=>false, 'isBoolean' =>
true, 'tl_class' => 'clr m12 w50')
        ),
```

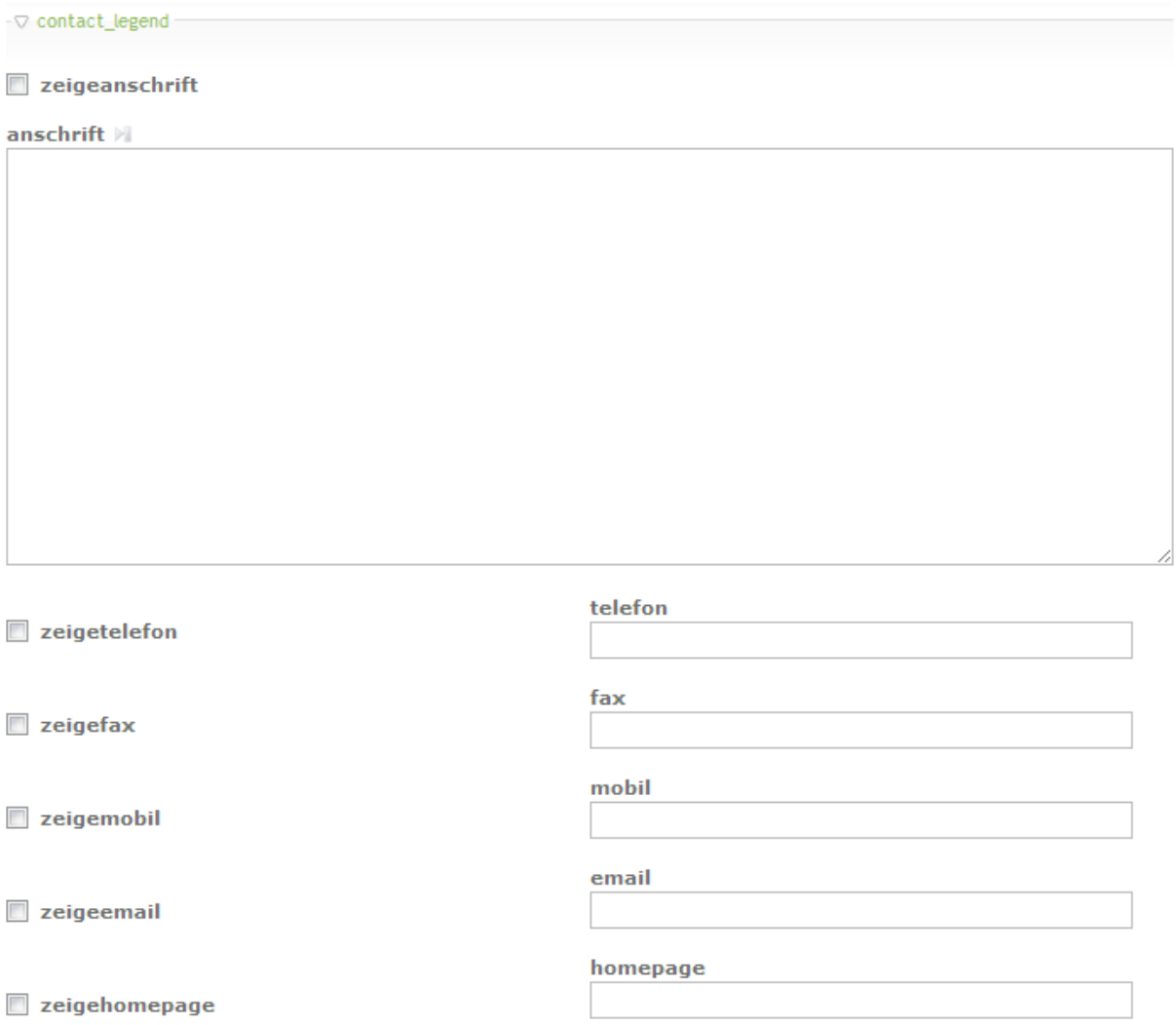
Die tl_class-Werte sind so gewählt, dass jede "Anzeigen"-Checkbox links in einer Reihe mit dem entsprechenden Textfeld (rechts) in einer Zeile steht. Ich hätte es gerne andersrum gehabt, also Textfeld links, Checkbox rechts, aber trotz viel experimentieren mit den tl_class-Werten ist es mir nicht gelungen, das Layout sah immer "zerschossen" aus.

Den Wert ['palettes']['default'] ergänze ich noch um

PHP-Code:

```
'{contact_legend:hide},zeigeanschrift,anschrift,zeigetelefon,telefon,zeigefax,fax,
zeigemobil,mobil,zeigeemail,email,zeigehomepage,homepage;'
```

Ergebnis:



"Beschreibung" ist eine Textarea, die in eigener Palette angezeigt werden soll. Einzige Besonderheit ist hier, dass ich HTML im Inhalt zulassen will.

PHP-Code:

```
        'beschreibung' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['beschreibung'],
            'explanation'             => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['beschreibung_explanation'],
            'inputType'               => 'textarea',
            'eval'                    => array('mandatory'=>false, 'cols' => 80, '
rows' => 20, 'allowHtml' => true)
        ),
```

Schließlich fehlt noch das Bild. Hier wollte ich eine Bilder-Auswahl wie im Content-Element "Bild" haben. Ich habe eine Weile herumexperimentiert, insbesondere mit dem Feldtyp "radioTable" (Der aber ganz falsch ist, wie mir jetzt klar ist). Lösung brachte dann ein Blick in das CD-Collection-Tutorial, wo auch so eine Bilderauswahl drin ist. Der richtige DCA-Eintrag lautet:

PHP-Code:

```
        'bild' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['bild'],
            'explanation'             => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['bild_explanation'],
      'inputType'               => 'fileTree',
            'eval'                    => array('mandatory'=>false, 'files'=>true,
'fieldType'=>'radio', 'filesOnly' => true, 'extensions' => 'jpg,jpeg,png,gif', 'pa
th' => 'tl_files/GW/Bilder_Turnierpaare/')
        )
```

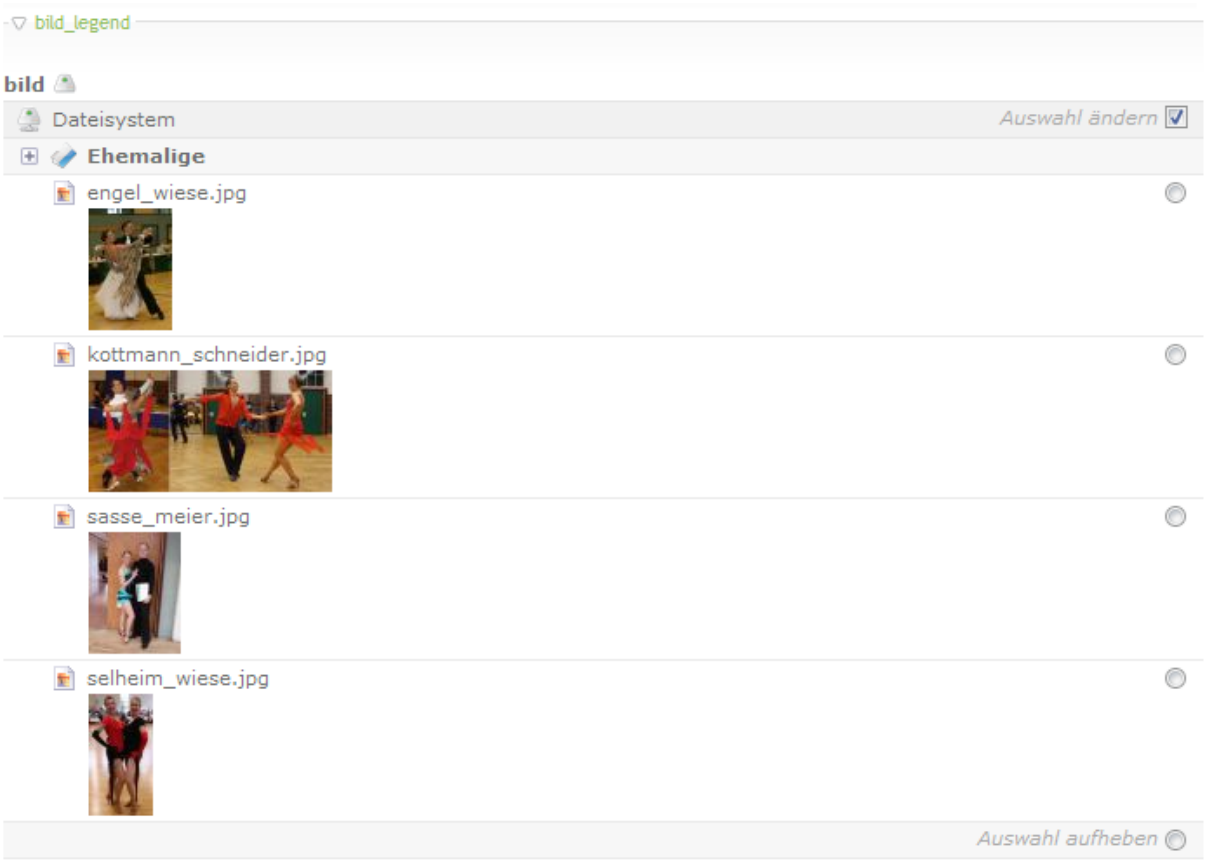
Typ ist also ein "fileTree", in dem man mittels Radiobutton EIN File auswählen kann (fieldType => radio), in dem Dateien mit den Erweiterungen jpeg,jpg,png und gif angezeigt werden (extensions), in dem Unterverzeichnisse UND Dateien angezeigt werden (files => true, sonst werden NUR Verzeichnisse angezeigt), und in dem man mittels des Radiobuttons auch ausschließlich Dateien auswählen kann, KEINE Unterverzeichnisse (filesOnly => true). Zusätzlich gebe ich den "Basis-Pfad" an, aus dem man auswählen kann (path). Wobei der natürlich bei Jedem anders heißen kann...Also eigentlich nicht so toll. Muss nochmal drüber nachdenken.

Noch die Palettendefinition erweitern um

PHP-Code:

```
'{beschreibung_legend:hide},beschreibung;{bild_legend:hide},bild;'
```

und wir landen hier:



Damit bin ich im Prinzip mit der Maskendefinition für diese Tabelle fertig, abgesehen vom Hook für das MD5-Hashing meines Passworts. Das verschiebe ich erstmal auf später :-).

Folgende "Probleme" habe ich noch: Meine Palettendefinition sieht insgesamt so aus:

PHP-Code:

```
    // Palettes
    'palettes' => array
    (
        '__selector__'                => array(''),
        'default'                     => '{name_legend},partnernachname,partnervor
name,partnerinnachname,partnerinvorname;'
                                      .'{classes_legend},startgruppe,startklassela
tein,startklassestandard;'
                                      .'{aktiv_legend:hide},aktiv,aktivseit,aktivb
is;{password_legend:hide},password;'
                                      .'{contact_legend:hide},zeigeanschrift,ansch
rift,zeigetelefon,telefon,zeigefax,fax,zeigemobil,mobil,zeigeemail,email,zeigehome
page,homepage;'
                                      .'{beschreibung_legend:hide},beschreibung;
{bild_legend:hide},bild;'
    ),
```

Eigentlich erwarte ich, dass nur die oberste Palette geöffnet ist, und alle folgenden geschlossen. Komischerweise sind beim Editieren bestehender Einträge und auch bei der Neuanlage alle geöffnet bis auf "password" und "beschreibung". Ich kann nicht verstehen, wieso. Kann mich jemand schlau machen?

Die Eigenschaft 'exclude' im Abschnitt 'fields' soll steuern, ob das jeweilige Feld in der Usergruppenverwaltung spezifisch für einzelne Gruppen (de)aktivierbar ist. Bei exclude => true soll das Feld in der Usergruppenverwaltung erscheinen, bei false soll es dort nicht erscheinen und immer sichtbar sein (für die Gruppen, die das Backend-Modul überhaupt freigegeben haben).

Das Verhalten scheint aber ein anderes zu sein: Egal ob ich exclude true oder false zuweise, erscheint das Feld in der Usergruppenverwaltung. Nur wenn ich exclude ganz weglasse, erscheint es dort nicht. Ein Bug? Keine Ahnung. Zumindest ist es in der Referenz anders beschrieben. Da ich diese Steuerung auf Feldebene nicht brauche, sondern das ganze Backendmodul nur einer bestimmten Usergruppe freischalten möchte, habe ich die 'exclude'-Eigenschaft aus allen Felddefinitionen entfernt.

Mein "Traum" wäre, dass man, falls ein Bild schon ausgewählt ist im Filetree man sofort dieses Bild sieht, statt erst den Filetree öffnen zu müssen, um zu sehen ob irgendwo der Radiobutton gesetzt ist. So kann man beim Öffnen eines Datensatzes nicht schnell sehen, ob ein Paar ein Bild zugewiesen hat, oder nicht. Aber ich glaube, das ist so (noch) nicht vorgesehen.

So, das sollte das Ende von Schritt 4 gewesen (puh),. Um die Maske zu vervollständigen werde ich mit der Definition der Texte in den Sprachfiles weitermachen.

Schritt 5: Language-Files

Nun geht es an die Sprachfiles, um das Backend-Modul "hübsch" zu machen.
Ich demonstriere es für die deutschen Sprachfiles im /system/modules/gw_turnierpaare/languages/de/-Verzeichnis. Englisch geht genau analog :-).

Zunächst definieren wir die Namen der Back- und Frontendmodule, und einen kurzen Erklärungstext dazu. Das wird in modules.php gemacht:

PHP-Code:

```
/**
 * Back end modules
 */
$GLOBALS['TL_LANG']['MOD']['gw_turnierpaare'] = array('Turnierpaare', 'Verwaltung
der Turnierpaare und der Meldeliste.');


/**
 * Front end modules
 */
$GLOBALS['TL_LANG']['FMD']['gw_turnierpaarliste'] = array('Turnierpaarliste', 'Die
ses Modul zeigt die Turnierpaarliste an');
$GLOBALS['TL_LANG']['FMD']['gw_meldeliste'] = array('Meldeliste', 'Dieses Modul ze
igt die Meldeliste an');
```

Die Bezeichner hinter 'MOD' und 'FMD' müssen die sein, dir wir im config/config.php der Extension definiert haben. Die entsprechenden Texte für die beiden geplanten Frontendmodule habe ich hier auch schon mal eingetragen, auch wenn es die Module noch nicht gibt...

Die Texte für die Backendfelder sind in tl_gw_turnierpaare.php definiert, entsprechend dem Namen der Datenbanktabelle. Die in der DCA-Record-Definition deklarierten Felder müssen wir mit Text füllen. Das ist ziemlich straight-forward:

PHP-Code:

```
/**
 * Fields
 */
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['partnernachname']    = array('Nachname
des Partners', 'Bitte den Nachnamen des (m&auml;nnlichen) Partners eingeben');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['partnervorname']     = array('Vorname d
es Partners', 'Bitte den Vornamen des (m&auml;nnlichen) Partners eingeben');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['partnerinnachname']  = array('Nachname
der Partnerin', 'Bitte den Nachnamen des (weiblichen) Partners eingeben');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['partnerinvorname']   = array('Vorname d
er Partnerin', 'Bitte den Vornamen des (weiblichen) Partners eingeben');

$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['startgruppe']        = array
('Startgruppe', 'Bitte die Startgruppe (JUG, HGR, SEN, ...) des Paares eingeben');

$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['startklasselatein']  = array
('Startklasse Latein', 'Bitte die Startklasse (Latein) des Paares eingeben. Kein L
ateinstartbuch = "-"');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['startklassestandard'] = array
('Startklasse Standard', 'Bitte die Startklasse (Standard) des Paares eingeben. Ke
in Standardstartbuch = "-"');

$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['aktiv']              = array('Aktiv', '
Bitte angeben, ob das Paar noch aktiv ist');
```

```
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['aktivseit']          = array('Aktiv sei
t', 'Bitte Jahreszahl des ersten Starts angeben (z.B. 2005)');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['aktivbis']           = array('Aktiv bis
', 'Bitte Jahreszahl des letzten Starts angeben, wenn das Paar nicht mehr aktiv is
t (z.B. 2008)');

$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['password']           = array
('Passwort', 'Bitte ein Passwort f&uuml;r das Paar anlegen');

$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['zeigeanschrift']     = array('Anschrift
anzeigen', 'Anschrift in der Visitenkarte sichtbar?');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['anschrift']          = array
('Anschrift', 'Bitte Anschrift des Paares eingeben');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['zeigetelefon']       = array
('Telefonnummer anzeigen', 'Telefonnummer in der Visitenkarte sichtbar?');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['telefon']            = array
('Telefonnummer', 'Bitte Telefonnummer des Paares eingeben');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['zeigefax']           = array('Faxnummer
anzeigen', 'Faxnummer in der Visitenkarte sichtbar?');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['fax']                = array
('Faxnummer', 'Bitte Faxnummer des Paares eingeben');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['zeigemobil']         = array
('Mobilnummer anzeigen', 'Mobilnummer in der Visitenkarte sichtbar?');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['mobil']              = array
('Mobilnummer', 'Bitte Mobilnummer des Paares eingeben');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['zeigeemail']         = array('Email-
Adresse anzeigen', 'Email-Adresse in der Visitenkarte sichtbar?');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['email']              = array('Email-
Adresse', 'Bitte Emailadresse des Paares eingeben');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['zeigehomepage']      = array('Homepage-
Adresse anzeigen', 'Homepage-Adresse in der Visitenkarte sichtbar?');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['homepage']           = array('Homepage-
Adresse', 'Bitte Homepage-Adresse des Paares eingeben');

$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['beschreibung']       = array
('Beschreibungstext', 'Bitte Beschreibungstext des Paares eingeben');

$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['bild']               = array
('Paarbild', 'Bitte ein Paarbild ausw&auml;hlen');

/**
 * Reference
 */
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['name_legend']          = 'Namen';
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['classes_legend']       = 'Startdaten';
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['aktiv_legend']         = 'Aktiv';
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['password_legend']      = 'Passwort';
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['contact_legend']       = 'Kontakt';
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['beschreibung_legend']  = 'Beschreibung
';
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['bild_legend']          = 'Bild';

/**
 * Buttons
 */
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['new']    = array('Neues Paar', 'Ein neu
es Turnierpaar anlegen');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['edit']   = array('Editieren', 'Das Turn
ierpaar editieren');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['copy']   = array('Paar kopieren', 'Das
Turnierpaar in die Zwischenablage kopieren');
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['delete'] = array('Paar löschen', 'Das T
urnierpaar aus der Liste entfernen');
```

```
$GLOBALS['TL_LANG']['tl_gw_turnierpaare']['show']   = array('Details', 'Die Detail
ansicht des Turnierpaars anzeigen');
```

Unterschieden werden die Texte für die Eingabefelder, die aus Überschrift/Bezeichner für das Feld und dem darunter angezeigten Beschreibungstext bestehen, den Texten für Palettenüberschriften (mittlerer Teil) und den Texten für die Buttons in den Masken (letzter Teil). Umlaute müssen HTML-üblich durch ihre Ersatzcodes (ä = ä usw.) dargestellt werden. Die Beschreibungstexte sollten auch nicht zu lang werden.

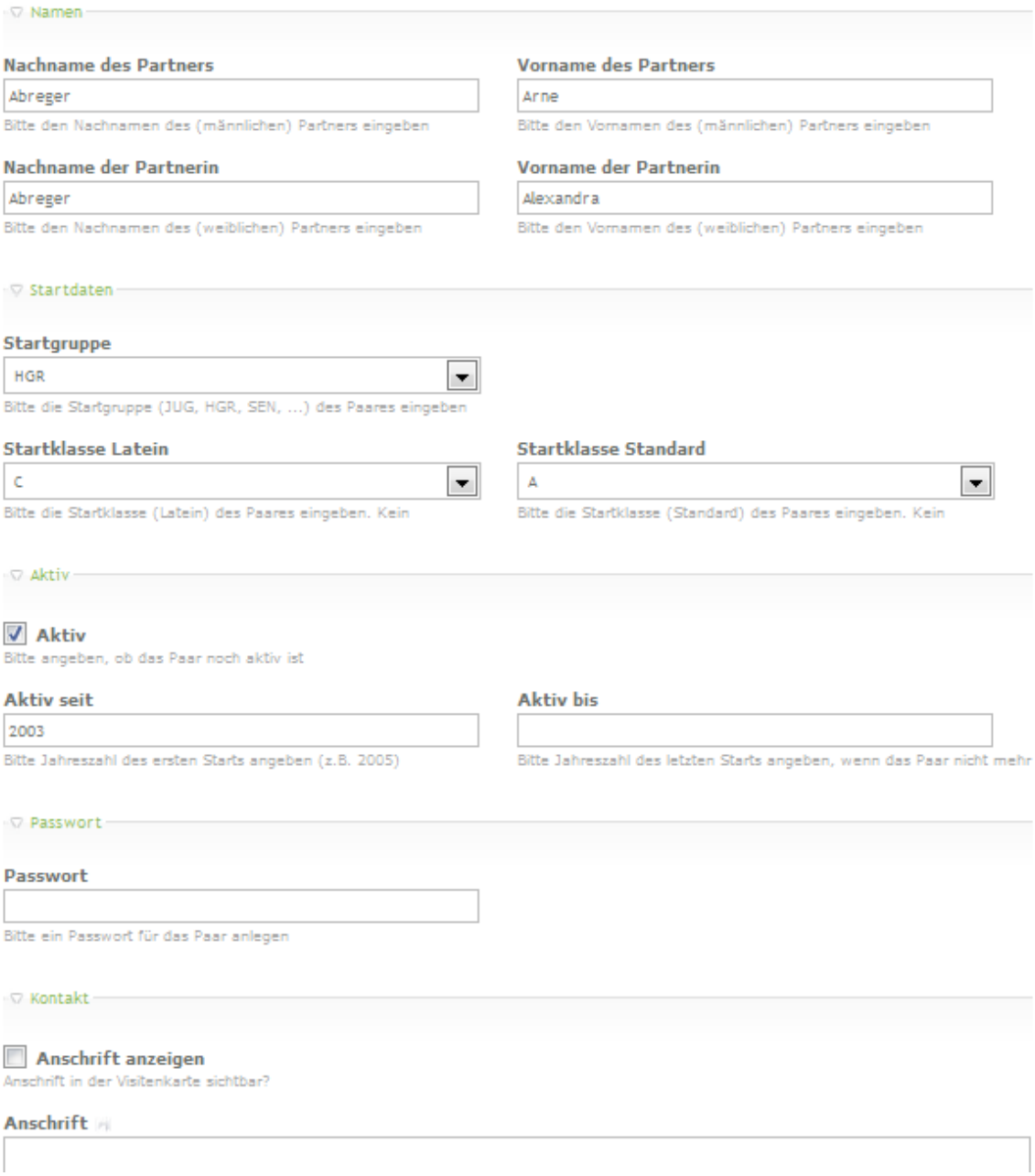
Ich habe in der DCA-Definition auch Referenzen auf Sprach-Strings für die Eigenschaft "explanation" angegeben, z.B. so:

PHP-Code:

```
            'explanation'               => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['partnernachname_explanation'],
```

allerdings habe ich nicht erkennen können, wo das benutzt wird. Vielleicht nur in bestimmten Situatuionen/Konfigurationen. Ich habe all diese Referenzen bei jedem Feld aus meiner DCA-Konfiguration also wieder entfernt.

Die Backend-Maske sieht jetzt so aus:



---

deerwood

Umlaute müssen HTML-üblich durch ihre Ersatzcodes (ä = ä usw.) dargestellt werden

Nein. UTF8 Zeichen/Editoren sind heutzutage angesagt . Z.B. Notepad++ oder Eclipse oder ... oder ...

Macht Spass, dies Tutorial einfach zu verfolgen, ich habe auch noch mehr Kommentare, mache aber bitte erst mal ungestört weiter. Es ist lehrreich, Deine Arbeit zu verfolgen. Ab und zu wäre es schön, wenn Du Deinen Stand des Codes als ZIP (etwa immer im ersten Beitrag und mit einer Versions-Nummer versehen) anbieten würdest, dann könnte man das als interessierter Mitleser ohne all zu viel Tipp-Arbeit in eine Test-Installation übernehmen und selbst ein wenig herumprobieren.

---

dl1ely

vielen Dank für den Hinweis. Ich nutze PSPad, den ich eigentlich als "vernünftigen" Editor ansehe, aber trotzdem stand der Zeichensatz auf "ANSI". Mit UTF8 geht es natürlich auch mit Umlauten ;-). Da muss ich in Zukunft mehr drauf achten.

Ich bitte sehr um Kommentare, genau deshalb mache ich das hier auch. Wenn keine kommen, mache ich "naiv" weiter...Also, bitte keine Zurückhaltung.

Zum Download: In der Tat wird es langsam sehr viel, und es ist schwer nachzuvollziehen. Das hatte ich mir auch anders vorgestellt. Ich denke ich werde also zukünftig den aktuellen Stand des Codes in der Tat zum Download bereitstellen. Danke für die Anregung.

Schritt 6: Von Callbacks und Subpaletten

Nachdem auch das englische Sprachfile fertig ist, geht es nun an die letzte fehlende Funktionalität der Backend-Maske. Zunächst gibt es aber noch einige Detailkorrekturen.

Da MySQL keinen "Boolean"-Datentyp bietet, hatte ich die Felder, die nur true/false sein können, als int(1) angelegt, mit den möglichen Werten 0/1. Das klappt auch prinzipiell, die Stati der Checkboxen werden gespeichert und wieder aus der Datenbank ausgelesen, aber ein Problem zeigt sich, wenn man nach so einem Feld filtern will: Ich will nach dem "aktiv"-Feld filtern können. Häufig sind die die "aktiven" Turnierpaare von Interesse, die nicht mehr aktiven verstopfen aber die Liste.

Wählt man dieses Feld nun als Filter-Feld aus, werden in der DropDown-Liste als Filtermöglichkeiten aber nur "Ja" und nochmals "Ja" angezeigt. Das Filtern klappt damit auch, bei dem einen "Ja" werden nur die aktiven Paare angezeigt, beim anderen "Ja" die inaktiven. Aber das ist natürlich nicht so gewollt.

Durch Abschauen bei anderen Extensions bin ich darauf gekommen, die Checkbox-Felder durch "char(1)" statt "int(1)" abzubilden. Das klappt genau so gut, und auch das Filtern funktioniert mit "Ja" und "Nein". Alle int(1)-Felder wurden entsprechend in char(1) verändert. Nach dem Anpassen der database.sql muss natürlich das Install-Tool ausgeführt werden, um die Änderungen in der Datenbank durchzuführen.

Die database.sql sieht jetzt so aus:

Code:

```
CREATE TABLE `tl_gw_turnierpaare` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `sorting` int(10) unsigned NOT NULL default '0',
  `tstamp` int(10) unsigned NOT NULL default '0',
  `partnernachname` varchar(64) NOT NULL default '',
  `partnervorname` varchar(64) NULL default NULL,
  `partnerinnachname` varchar(64) NULL default NULL,
  `partnerinvorname` varchar(64) NULL default NULL,
  `startgruppe` varchar(32) NOT NULL default '',
  `startklasselatein` varchar(12) NULL default NULL,
  `startklassestandard` varchar(12) NULL default NULL,
  `aktiv` char(1) NOT NULL default '',
  `aktivseit` int(4) NULL default NULL,
  `aktivbis` int(4) NULL default NULL,
  `resetpassword` char(1) NULL default '',
  `password` varchar(64) NULL default NULL,
  `bild` varchar(255) NULL default NULL,
  `anschrift` text NULL,
  `zeigeanschrift` char(1) NOT NULL default '',
  `telefon` varchar(32) NULL default NULL,
  `zeigetelefon` char(1) NOT NULL default '',
  `fax` varchar(32) NULL default NULL,
  `zeigefax` char(1) NOT NULL default '',
  `mobil` varchar(32) NULL default NULL,
  `zeigemobil` char(1) NOT NULL default '',
  `email` varchar(128) NULL default NULL,
  `zeigeemail` char(1) NOT NULL default '',
  `homepage` varchar(128) NULL default NULL,
  `zeigehomepage` char(1) NOT NULL default '',
  `beschreibung` text NULL,
  PRIMARY KEY  (`id`),
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Wer aufgepasst hat, dem ist auch noch ein neues Feld aufgefallen:

Code:

```
  `resetpassword` char(1) NULL default '',
```

Das werde ich gleich für das Aktivieren einer Subpalette benötigen. Der Inhalt des Feldes in der Datenbank wird später nicht gebraucht, aber leider muss das Feld vorhanden sein, um es so nutzen zu können, wie ich es vorhabe.

Für das Passwortfeld habe ich ich vor, dass ein dort eingegebenes Passwort SHA1-gehasht in der Datenbank abgelegt wird, nicht im Klartext. Dabei wird ein eventuell schon vorhandes Passwort natürlich überschrieben. Um Fehleingaben zu verhindern, möchte ich eine Checkbox anzeigen, die defaultmäßig "aus" ist. Erst wenn die Checkbox aktiviert ist, soll per AJAX das Passwortfeld angezeigt werden.

Zunächst fügen die wir Definition für das Checkbox-Feld in die "field"-Sektion des DCA-Records ein:

PHP-Code:

```
        'resetpassword' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['resetpassword'],
            'inputType'               => 'checkbox',
            'default'                 => '',
            'eval'                    => array('mandatory'=>false, 'isBoolean' =>
true, 'submitOnChange' => true),
        ),
```

submitOnChange bewirkt, dass das Formular neu geladen wird, wenn das Feld angeklickt wird. nur dann wird das Passwortfeld nachgeladen.

Dafür benötigen wir eine sogenannte "Subpalette". Die Checkbox (bei mir "resetpassword") muss als "__selector__" angegeben werden im DCA-Record:

PHP-Code:

```
    // Palettes
    'palettes' => array
    (
        '__selector__'                => array('resetpassword'),
...
```

In der "default"-Sektion sieht die Palettendefinition so aus:

PHP-Code:

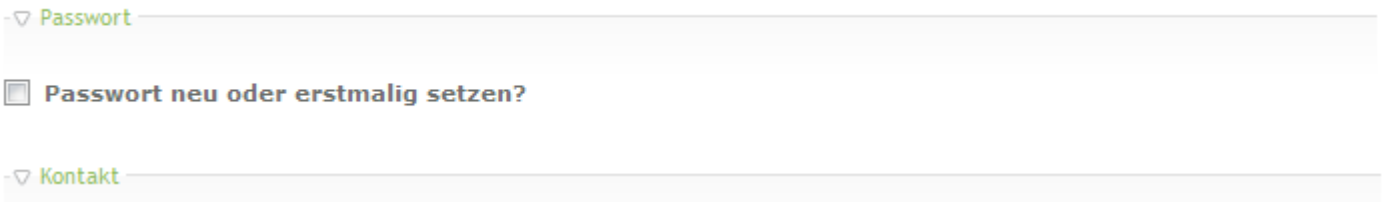
```
                                     .'{aktiv_legend:hide},aktiv,aktivseit,aktivbis;
{password_legend:hide},resetpassword;'
```

Hier steht also der Name der Subpalette. Der Name des Passwortfeldes steht hier nicht mehr. Das wird in der "subpalettes"-Sektion angegeben:

PHP-Code:

```
    // Subpalettes
    'subpalettes' => array
    (
        'resetpassword'               => 'password'
    ),
```

Dies bedeutet, dass das Feld "password" in die Subpalette "resetpassword" eingeblendet wird, wenn "resetpassword" aktiviert ist. Wird es deaktiviert, verschwindet die Subpalette wieder.



Die Texte hierzu wurden in den Sprachfiles entsprechend erweitert und angepasst.

Für beide Felder, resetpassword und password, benötige ich besondere Funktionalitäten. resetpassword soll bei Öffnen der Backendmaske IMMER deaktiviert, das Passwort-Feld also versteckt sein - egal was in der Datenbank für das Feld steht. Dafür setze ich zunächst

PHP-Code:

```
            'default'                 => '',
```

was dafür sorgt, dass beim Anlegen eines neuen Datensatzes die Checkbox deaktiviert ist. Und dann gibt es noch die Option "load_callback", in der eine Funktion angegeben werden kann, die beim Laden des Feldes aufgerufen wird (Zu den Details von Callbacks gleich mehr).

Hier habe ich versucht, durch "return ";" immer den Defaultwert zurückzugeben. Leider klappt das nicht richtig, wenn der Datensatz mit "speichern" gespeichert wird, aber geöffnet bleibt. Obwohl die Checkbox dann deaktiviert dargestellt wird, bleibt das Passwort-Feld trotzdem angezeigt und wird nicht versteckt. Erst durch manuelles Aktivieren und erneutes Deaktivieren verschwindet das Passwortfeld wieder. Ich weiß nicht, ob das ein Bug oder gewollt ist, zumindest gefiel es mir nicht.

Ein weiterer Versuch scheiterte mit dem save_callback: Eine Funktion, die aufgerufen wird, bevor das Feld in die Datenbank gespeichert wird. Hier versuchte ich ebenfalls durch ein "return ";" zu erzwingen, dass immer ein deaktiviertes Feld gespeichert wird, und damit auch beim erneuten Anzeigen des Formulars deaktiviert bleibt. Das funktioniert optisch auch sehr gut. Leider werden dann aber keine Eingaben im Passwort-Feld gespeichert.

Der Grund wird sehr wahrscheinlich sein, dass der save_callback auf dem resetpassword-Feld ausgeführt wird, bevor der Input im password-Feld verarbeitet wird. Mein Save-Callback deaktiviert die Checkbox, und damit auch die Subpalette, und das Feld in der Subpalette wird gar nicht mehr ausgewertet oder gespeichert. Auch die Reihenfolge der Felddefinitionen hat darauf keinen Einfluss, es kommt wohl auf die Reihenfolge in der Palettendefinition an, und die kann ich nicht umdrehen.

An der Stelle war tiefe Frustration angesagt, aber ich habe das Problem anders lösen können. Beim resetpassword-Feld werden jetzt keine Callbacks verwendet.

Dafür aber beim password-Feld, was jetzt in der Definition so aussieht:

PHP-Code:

```
        'password' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['password'],
            'inputType'               => 'text',
            'eval'                    => array('mandatory'=>false, 'minlength' =>
1, 'maxlength' => 64),
            'load_callback'           => array(array('tl_gw_turnierpaare','passwor
d_load_callback')),
            'save_callback'           => array(array('tl_gw_turnierpaare','passwor
d_save_callback'))
        ),
```

Durch diese Definition wird beim Laden des Feldes die Funktion "password_load_callback" und beim Speichern "password_save_callback" aufgerufen, die sich in der Klasse "tl_gw_turnierpaare" befinden. Diese Klasse lege ich in der DCA-Definitions-Datei /modules/gw_turnierpaare/dca/tl_gw_turnierpaare.php an.

PHP-Code:

```
class tl_gw_turnierpaare extends Backend
{

    /**
     * Import the back end user object
     */
    public function __construct()
    {
        parent::__construct();
        $this->import('BackendUser', 'User');
    }


  public function password_load_callback()
  {
 ...
 }

  public function password_save_callback($var, $dc)
  {
...
  }

}
```

Das Gerüst habe ich mir bei anderen Extensions abgeschaut, es scheint zumindest klug zu sein, von "Backend" zu erben, und den Konstruktur zu überschreiben. Vielleicht ist es auch nicht nötig, ich habe es nicht probiert

Ob der load_callback Parameter übergeben bekommt weiss ich nicht, aber ich benötige keinen Parameter.

Der save_callback erhält den Wert des Feldes, das gespeichert werden soll ($var), und den DataContainer ($dc), der zu dem Formular gehört. Meist (wie auch hier) ist es DC_Table, der DataContainer für Datenbanktabellen.

Mein password-Feld in der Datenbank wird den SHA1-Hash, also einen langen String unverständlicher hexadezimaler Zahlen enthalten. Es nützt nichts, wenn ich den im Backend im Passwort-Feld anzeige. Dort will der Admin Klartext-Passwörter eingeben. Im Load-Callback setze ich den Wert des password-Felds also auf einen leeren String, egal was in der Datenbank steht:

PHP-Code:

```
  public function password_load_callback()
  {
    // Passwort-Feld immer leer anzeigen (Damit der User SHA1-Hash nicht sieht)
    return '';
  }
```

Damit wird der User schon mal nicht vom "Müll" aus der Datenbank belästigt. Umgekehrt müssen wir aber nicht das Klartext-Passwort, sondern den Hash in die Datenbank schreiben. Das macht der save_callback:

PHP-Code:

```
  public function password_save_callback($var, $dc)
  {
    // Kein neues PW angegeben: Feld nicht ändern
    if(strlen($var) < 1) return '';

    // Aktuellen Datensatz aus DB holen
    $row = $this->Database->prepare("SELECT * FROM tl_gw_turnierpaare WHERE id=?")

                          ->execute($dc->id);

    // PW in Passwort und Salt aufspalten
    list($strPassword, $strSalt) = explode(':', $row->password);

    // Falls kein Salt vorhanden, dann erzeugen
        if (!strlen($strSalt))
        {
            $strSalt = substr(md5(uniqid('', true)), 0, 23);
        }

    // SHA1-Hash aus Salt+neuem Passwort berechnen, Salt anhängen
    $pwd = sha1($strSalt . $var) . ':' . $strSalt;

    // Das resetpassword-Feld löschen
    $this->Database->prepare("UPDATE tl_gw_turnierpaare SET resetpassword='' WHERE
id=?")
                          ->executeUncached($dc->id);

    return $pwd;
  }
```

Falls kein Passwort angegeben wurde, macht der save_callback garnichts, und gibt einen leeren String zurück. Das Feld "Id" des DataContainers enthält die id des aktuellen Datensatzes in der Datenbank. Um den Hash berechnen zu können, benötige ich das "alte" Passwort in der Datenbank. Darum hole ich mir erstmal den gesamten Datensatz mit der ID ab.

Die Hash-Erzeugung habe ich mir beim File /system/libraries/User.php abgeschaut und funktioniert genauso wie in der Userverwaltung von TYPOlight: Der Hash wird zusammen mit einem "Salt" erzeugt, der zusammen mit dem Hash (durch Doppelpunkt getrennt) im Passwort-Feld abgespeichert wird. Existiert noch kein Salt, wird er erzeugt. Existiert der Salt schon, wird er weiterverwendet (und dafür muss ich das alte Passwort aus der Datenbank auslesen - um an den evtl. schon vorhandenen Salt zu kommen). Die Hash-Erzeugung läuft dann ziemlich straight-forward ab.

Und fast ganz am Ende nochmal der Knackpunkt: Hier setze ich das resetpassword-Feld in der Datenbank auf ". Das (und leider nur das) sorgt in allen Fällen dafür, dass beim Öffnen von Datensätzen die Subpalette für das Passwort-Feld geschlossen ist.

Abschließend gebe ich den berechneten Hash zurück. Er wird dann in die Datenbank eingetragen.

Wichtiger Hinweis noch: load_callback und save_callback müssen doppelt geschachtelte Arrays sein, weil es

mehrere Callbacks geben kann, die nacheinander aufgerufen werden, der Feldwert wird jeweils durch alle durchgeschleust. Falls man aber z.B. einen onSubmitCallback für das ganze Formular vorgeben möchte, ist das nur ein einfaches Array mit Klassennamen und Methodenname, weil es hier nur einen Callback gibt. Das ist so leider in der Referenz der möglichen Callbacks nicht dokumentiert, und hat mir kurz graue Haare beschert.

Der aktuelle Codestand wird gleich oben im ersten Post aktualisiert.

Damit ist das Backend-Modul für die Turnierpaar-Tabelle erstmal fertig.
Weiter wird es dann (endlich) mit dem Frontendmodul für die Turnierpaarliste gehen.

Schritt 7: Endlich Frontend!

nach soviel Kampf hinter den Kulissen wird jetzt endlich was im Frontend angezeigt und - um das vorweg zu nehmen - das klappt ziemlich reibungslos.

Zunächst mache ich eine "Simpel-Version" des Frontends, die erstmal was anzeigt. Verfeinerungsschritte kommen dann nach und nach dazu. Erstmal muss das schnelle Erfolgserlebnis her.

Zunächst mal müssen die Frontendmodule in system/modules/gw_turnierpaare/config/config.php "registriert" werden:

PHP-Code:

```
// Front end module
array_insert($GLOBALS['FE_MOD']['turnierpaare'], 0, array
(
    'gw_turnierpaarliste' => 'gwTurnierpaarliste',
    'gw_meldeliste' => 'gwMeldeliste'
));
```

'turnierpaare' scheint die Zwischenüberschrift zu sein, die man in der Dropdownliste der zur Verfügung stehenden Module zu sehen kriegt. Die Keys in dem Array sind einfach nur die Bezeichner, die durch die Languagefiles auch übersetzt werden. Die Values dahinter sind die Namen der Frontendklassen. Zu jeder Frontendklasse existiert /system/modules/gw_turnierpaare/*.php mit der entsprechenden Klassendefinition.

Damit in der Modul-Dropdown-Liste nicht das hässliche 'turnierpaare' steht, muss die Languagedatei erweiter werden, z.b. system/modules/gw_turnierpaare/languages/de/modules.php um:

PHP-Code:

```
$GLOBALS['TL_LANG']['FMD']['turnierpaare'] = array('Turnierpaare');
```

Nun kommt das eine große Puzzle-Teil für die Frontendausgabe: Die PHP-Klasse, die unsere Inhalte aus der Datenbank holt, aufbereitet und an das Template weiterreicht:

PHP-Code:

```
class gwTurnierpaarliste extends Module
{

    /**
     * Template
     * @var string
     */
    protected $strTemplate = 'gw_turnierpaarliste';

    /**
     * Generate module
     */
    protected function compile()
    {
    $arrPaare = array();
    $objPaare = $this->Database->execute("SELECT * FROM tl_gw_turnierpaare ORDER B
Y partnernachname, partnerinnachname");

    while ($objPaare->next())
    {
      $newArr = array
      (
```

```
        'partnernachname' => trim($objPaare->partnernachname),
        'partnervorname' => trim($objPaare->partnervorname),
        'partnerinnachname' => trim($objPaare->partnerinnachname),
        'partnerinvorname' => trim($objPaare->partnerinvorname),
        'startgruppe' => $objPaare->startgruppe,
        'startklasselatein' => $objPaare->startklasselatein,
        'startklassestandard' => $objPaare->startklassestandard,
        'aktiv' => $objPaare->aktiv,
        'aktivseit' => $objPaare->aktivseit,
        'aktivbis' => $objPaare->aktivbis,
      );

      if(strlen($objPaare->bild) == 0)
      {
        $newArr['bild'] = '/system/modules/gw_turnierpaare/icons/default.png';
      }
      else
      {
        $newArr['bild'] = $this->getImage($objPaare->bild, '30', '30');
      }

      $arrPaare[] = $newArr;
    }

    $this->Template->paare = $arrPaare;
    }
}
```

Wir leiten von Module ab, und $strTemplate enthält den Namen des Templates, was wir füllen wollen. Die Funktion, die die Werte für das Template liefert, muss compile() heißen.

Wir holen uns alle Turnierpaardatensätze aus der Datenbank, sortiert nach den Nachnamen (Andere Sortierungen und Filter werden später hinzugefügt!), durchlaufen diese in einer Schleife und füllen pro Eintrag ein Array $newArr mit den Werten der Datenbankfelder.

Das 'bild'-Feld wird anders behandelt: Ist es leer (also kein Bild ausgewählt), wird als Bild-URL ein Default-Bild übergeben, was ich von Hand zum icons-Unterverzeichnis hinzufüge (Das benutzte Bild hänge ich hier an). Falls ein Bild angegeben wurde, hole ich mir über die in TL eingebaute getImage()-Funktion ein Thumbnail des Bildes, was maximal 30 mal 30 Pixel groß ist.

Dann wird das das Array, was die Daten eines Paares enthält zum Array mit allen Paardaten hinzugefügt, und ganz am Ende die Template-Variable 'paare' mit unserem Ergebnis (Also den Daten aller Paare in der Liste) gefüllt.

Nun das Ausgabetemplate system/modules/gw_turnierpaare/templates/gw_turnierpaarliste.tpl:

Code:

```
<div class="<?php echo $this->class; ?>"<?php echo $this->cssID; ?><?php if ($this-
>style): ?> style="<?php echo $this->style; ?>"<?php endif; ?>>
<?php if ($this->headline): ?>
<<?php echo $this->hl; ?>><?php echo $this->headline; ?></<?php echo $this->hl; ?>>
<?php endif; ?>

<table cellpadding="4" cellspacing="0" summary="Turnierpaarliste">
  <thead>
    <tr>
      <th> </th>
      <th>Name</th>
      <th>Startgruppe</th>
      <th>Std</th>
      <th>Lat</th>
```

```
      <th>Aktiv seit</th>
      <th> Aktiv bis</th>
    </tr>
  </thead>
  <tbody>
<?php foreach ($this->paare as $paar): ?>

  <tr<?php if($paar['aktiv'] != '1') { echo ' style="color: #888;"'; } ?>>
  <td>
  <img src="<?php echo $paar['bild']; ?>">
  </td>
  <td><?php echo $paar['partnernachname']; ?>
<?php if($paar['partnervorname']): ?>
<?php   echo ', '.$paar['partnervorname']; ?>
<?php endif; ?>
<?php if($paar['partnerinnachname']): ?>
<?php   echo ' und '.$paar['partnerinnachname']; ?>
<?php   if($paar['partnerinvorname']): ?>
<?php     echo ', '.$paar['partnerinvorname']; ?>
<?php   endif; ?>
<?php endif; ?>
  </td>
  <td><?php echo $paar['startgruppe']; ?>
  </td>
  <td><?php echo $paar['startklassestandard']; ?>
  </td>
  <td><?php echo $paar['startklasselatein']; ?>
  </td>
  <td><?php echo $paar['aktivseit']; ?>
  </td>
  <td><?php echo $paar['aktivbis']; ?>
  </td>
  </tr><?php endforeach; ?>
  </tbody>
</table>

</div>
```

Den Teil bis zum Beginn der Table habe ich aus einem anderen Template übernommen, und enthält den "Standardheader" mit der Möglichkeit, spezielle Klassen, IDs und Styles anzugeben. Auch eine Überschrift wird zugelassen, falls sie gesetzt ist.

In der Tabelle selbst wird ein Kopfbereich definiert mit den Spaltenüberschriften.
Dann werden alle Paare in der Paarliste (Variable $this->paare) durchlaufen. Falls das Paar NICHT aktiv ist, setze ich einen style für die Tabellenzeile (graue Schrift). Das ist so nicht "sauber" und wird auch so nicht bleiben, sondern durch richtiges CSS-Markup ersetzt. Erstmal dient es zur Visualisierung.

Aus den Namen von Herr und Dame wird ein "schöner" Namensstring zusammengebaut und ausgegeben. Die rechtlichen Felder werden ziemlich straight-forward ausgegeben.

Nun wären wir eigentlich "fast" fertig. Aber um unser neues Frondendmodul nutzen zu können, muss es im Backend unter "Module" angelegt werden. Das übernimmt die Backend-Klasse tl_module, und die weiss noch nichts detailliertes über unser neues Frontendmodul.

Wir müssen die DCA-Konfiguration für die Backend-Klasse tl_module so erweitern, dass die richtige Palette angezeigt wird, wenn bei Modul-Typ unser neues Frontendmodul ausgewählt wird. Dafür legen wir in system/modules/gw_turnierpaare/dca/ eine neue Datei tl_module.php an. Dort können wir die Definition erweitern. Das tun wir NICHT im tl_module-Backendmodul selbst. Da die DCA-Definitionen aller Extensions nacheinander eingelesen werden (und auf jeden Fall nach Frontend und Backend) können wir den fehlenden Eintrag für tl_module bei uns im Modul nachholen.

In die tl_module.php kommt:

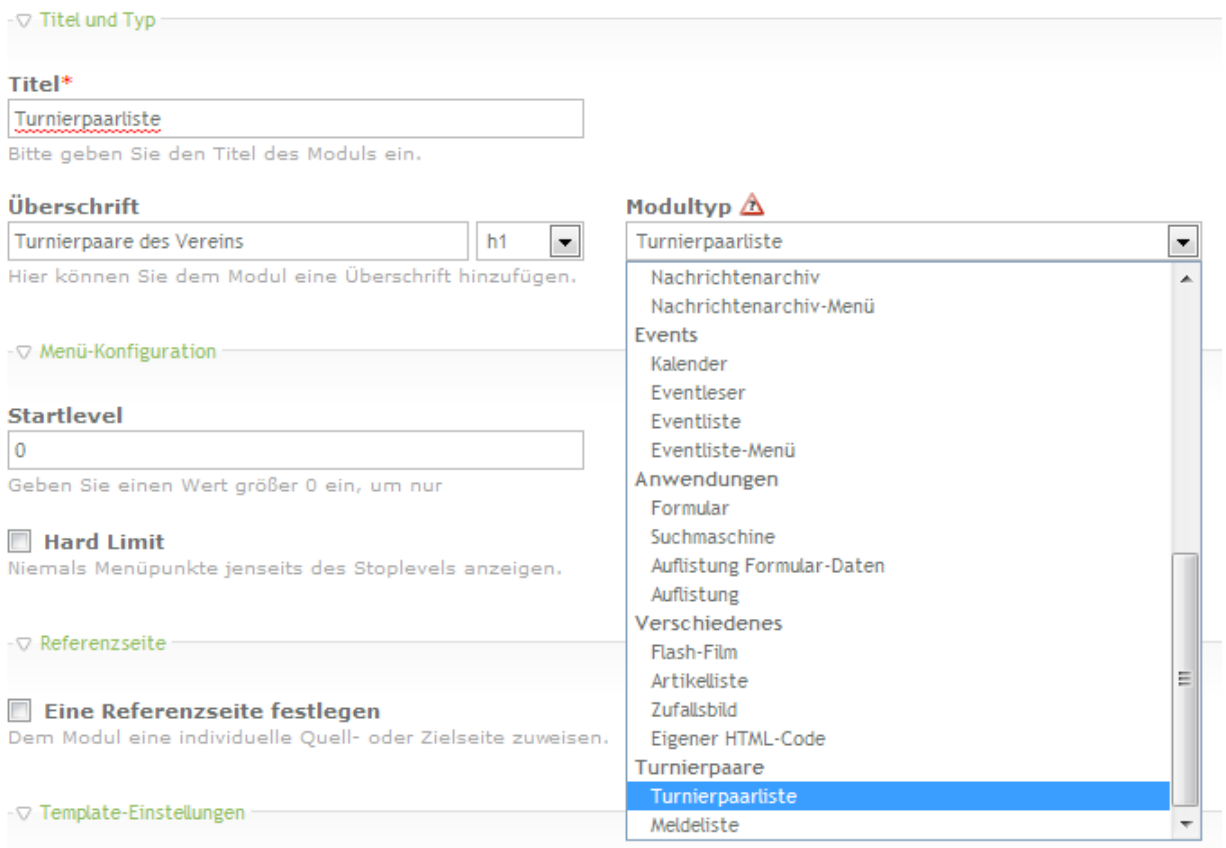
PHP-Code:

```php
<?php
// Add a palette to tl_module

$GLOBALS['TL_DCA']['tl_module']['palettes']['gw_turnierpaarliste'] = 'name,type,headline;align,space,cssID';
?>
```

Wenn in der Modulverwaltung in der Drop-Down-Liste der zur Verfügung stehenden Module unseres ('gw_turnierpaare') ausgewählt wird, wird diese Palette aktiviert, und als Modul-Optionen Name, Typ und Überschrift, Ausrichtung, Abstände und css-Optionen angezeigt. Später werden wir hier noch weitere Optionen einfügen.

Der letzte Key der Definition ['gw_turnierpaarliste'] muss die ID unseres Moduls sein, so wie in config/config.php definiert. Hier hatte ich zuerst 'tl_gw_turnierpaare' benutzt - funktioniert nicht!

Jetzt können wir das neue Modul in der Modulverwaltung anlegen:



Hier sehen wir unsere Frontendmodule (das zweite kommt noch) in der Auswahlliste und bei Selektion des oberen Moduls die geänderte Palette (durch die Erweiterung des DCAs von tl_module). "align" scheint aber irgendwie nicht zu funktionieren. Ich ignoriere das erstmal.

Jetzt können wir das Modul einem Artikel hinzufügen, und bei mir sieht es dann so aus:

	Name	Startgruppe	Std	Lat	Aktiv seit	Aktiv bis
	Abreger, Arne und Abreger, Alexandra	HGR	A	C	2003	
	Abreger, Arne und Absteiger, Aurelia	HGR	-	B	1999	2001
	Aufreger, Anton und Aufreger, Anne	JUG	C	A	1999	2004
	E, Dirk und W, Susanne	HGR	A	D	2002	
	Jive, Jens und Samba, Susi	SEN I	A	-		
	Kick, Kalle und Drop, Daniela	HGR II	C	-		
	Standardformation		RL	-	2004	
	Tango, Toni und Tango, Tina	SEN III	S	-	2000	
	Walzer, Willi und Walzer, Wilma	SEN I	S	-		
	Zick, Zacharias und Zick, Zäzilie	HGR	B	B	2002	

Gut, das Icon ist auf meinem Hintergrund noch nicht "hübsch", aber die zugrunde liegende Funktionalität (Default-Bild wenn keins gesetzt), usw... ist erkennbar. Auch die Thumbnailerstellung eines Paarbildes funktioniert, und nicht-aktive Paare werden grau dargestellt.

Damit genug für heute, ein erstes Ergebnis ist erreicht, aber das Frontendmodul wird noch deutlich aufgebohrt werden. In den nächsten Folgen ;-).

Schritt 7b: Ein wenig Finetuning

Zunächst mal habe ich mir die Palettendefinitionen in system/modules/backend/dca/tl_module.php angeschaut und festgestellt, dass der von mir benutzte Code aus einem Tutorial wohl etwas veraltet ist. Ich orientiere mich an den anderen Einträgen und ändere meine Palettendefinition für die Modul-Verwaltung in system/modules/gw_turnierpaare/dca/tl_module.php auf:

PHP-Code:

```
<?php
// Add a palette to tl_module

$GLOBALS['TL_DCA']['tl_module']['palettes']['gw_turnierpaarliste'] = '{title_legen
```

```
d},name,headline,type;{protected_legend:hide},protected;
{expert_legend:hide},guests,cssID,space';
?>
```

Damit haben wir schöne Palettenüberschriften und den "Standardsatz" an Eigenschaften für Module.

Dann hatte ich das hier entdeckt: http://www.typolight.org/blog-leser/...zugreifen.html, und erinnerte mich an den save_callback für das Passwort-Feld der Paare, der genau sowas (Datensatz mit aktueller ID aus Datenbank einlesen) macht. Da ich sowieso nur 2.8 einsetze, wo das Feature des "activeRecord" verfügbar ist, klang das gut, und ich wollte es so umsetzen.

Aber: Der activeRecord enthält im save_callback wirklich die aktuellen Werte der Felder in der Eingabemaske. Im Falle des Passwortfelds also den leeren String, wenn nichts eingegeben wurde, bzw. das neue Klartextpasswort. Im save_callback brauche ich aber das alte, in der DB vorliegende Passwort (um den salt dort herauszuholen). Das geht mit dem activeRecord nicht, und ich muss mit der alten Methode, das Feld aus der Datenbank zu holen leben. Wenn man aber wirklich die aktuell in der Maske vorliegenden Felder benötigt, ist der activeRecord bestimmt ein super Feature.

Toflar

Als kleine Ergänzung. Ich finde das passt gerade schön hier rein:

Oftmals muss man für ältere TL Versionen mit dem Code ein bisschen anders umgehen. Aber man möchte die Erweiterung trotzdem für mehrere Versionen freigeben. Dann muss man halt an diversen Orten die TL-Version prüfen:

PHP-Code:

```
if(version_compare(VERSION . '.' . BUILD, '2.8.0', '<'))
{
   // mach für Versionen unter 2.8.0 das
}
else
{
   // ab Version 2.8.0 mach das
}
```

Schritt 8: Parameter fürs Modul

Die Ausgabe im Frontendmodul möchte ich gerne in der Modulverwaltung parametrisieren können. Einerseits soll man auswählen können, ob das Modul nur aktive oder nur inaktive, oder alle Paare auflisten soll. Andererseits soll der Sortiermodus zwischen "Nachnamen alphabetisch" und "Nach Altersgruppe aufsteigend" wählbar sein.

Dafür ist der etablierte Weg eine Erweiterung der Tabelle tl_module um die benötigten Felder. Man soll bestehende Fehler wenn möglich recyclen, aber man recycelt dann auch die Beschreibungstexte usw mit, das war für mich nicht passend.

Also habe ich neue Felder angelegt. In der config/database.php meiner Extension:

Code:

```
--
-- Extend table 'tl_module'
--

CREATE TABLE `tl_module` (
  `gw_tp_showonlyactive` char(1) NOT NULL default 'B',
  `gw_tp_couplesorting` char(1) NOT NULL default 'A',
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Hier füge ich also zwei "CHAR"-Felder hinzu, die Buchstabencodes enthalten (Beim Filter: B = Alle Paare [Both], A = Nur aktive Paare, I = Nur inaktive Paare; Beim Sortieren: A = Alphabetisch, C = Nach Altersklasse [Class]), die ich dann später im Frontend-Ausgabemodul auslesen und interpretieren muss. WICHTIG: Obwohl ich nur eine schon bestehende Tabelle ERWEITERN will, muss ich hier "CREATE TABLE" verwenden. Das Installtool macht das Richtige daraus...

Danach muss ich das Installtool aufrufen, um die neuen Felder anlegen zu lassen.

Dann muss der DCA-Record für tl_module so erweitert werden, dass meine Felder angezeigt werden, wenn in der Dropdown-Liste der zur Verfügung stehenden Module die Turnierpaarliste ausgewählt wird.

Dafür nehmen wir uns system/modules/gw_turnierpaare/dca/tl_module.php vor:

PHP-Code:

```
$GLOBALS['TL_DCA']['tl_module']['palettes']['gw_turnierpaarliste'] = '{title_legend},name,headline,type;{sort_legend},gw_tp_showonlyactive,gw_tp_couplesorting;{protected_legend:hide},protected;{expert_legend:hide},guests,cssID,space';
```

In der Subpalette für mein Modul füge ich einen neuen Abschnitt mit Überschrift "sort_legend" ein. Darin werden meine beiden neuen Datenbankfelder angezeigt.

Die müssen noch im DCA definiert werden:

PHP-Code:

```
$GLOBALS['TL_DCA']['tl_module']['fields']['gw_tp_showonlyactive'] = array
(
    'label'                   => &$GLOBALS['TL_LANG']['tl_module']['gw_tp_showonlyactive'],
    'default'                 => 'B',
    'exclude'                 => true,
    'inputType'               => 'select',
    'options'                 => array('B','A','I'),
```

```
    'reference'               => &$GLOBALS['TL_LANG']['tl_module']
['gw_tp_filteroptions'],
    'eval'                    => array('mandatory'=>true, 'tl_class' => 'w50')
);

$GLOBALS['TL_DCA']['tl_module']['fields']['gw_tp_couplesorting'] = array
(

    'label'                   => &$GLOBALS['TL_LANG']['tl_module']
['gw_tp_couplesorting'],
    'default'                 => 'A',
    'exclude'                 => true,
    'inputType'               => 'select',
    'options'                 => array('A','C'),
    'reference'               => &$GLOBALS['TL_LANG']['tl_module']
['gw_tp_sortoptions'],
    'eval'                    => array('mandatory'=>true)
);
```

Der Typ ist "select", also eine Dropdownbox. Die "options" sind die Werte, die in die Datenbank geschrieben werden sollen. Die "reference" sind im Gegensatz dazu die Werte, die in der Dropdown-Box angezeigt werden sollen. Hier war mir anfangs unklar, wie genau die Übersetzungen definiert werden müssen. Dazu gleich mehr. Hier setze ich erstmal die Referenz auf ein Array, was in den Sprachfiles definiert wird. Da die Werte in der Modulverwaltung angezeigt werden, habe ich mich dazu entschlossen, sie unter ['tl_module'] einzusortieren, aber mit meinem Präfix "gw_tp", um Namenskollisionen unwahrscheinlich zu machen.

Nun noch die Definition der Strings in der deutschen Sprachdatei (englisch geht genauso). Da die String zur Modulverwaltung gehören, habe ich mich dazu entschlossen, sie in die Datei system/modules/gw_turnierpaare/languages/de/modules.php zu schreiben. Man hätte auch eine neue Datei tl_module.php anlegen können. Ich weiss nicht, was da "best practice" ist.

Wir ergänzen also die modules.php um:

PHP-Code:

```
/**
 * Back end fields for tl_module
 */
$GLOBALS['TL_LANG']['tl_module']['sort_legend']     = 'Filter und Sortierung';

$GLOBALS['TL_LANG']['tl_module']['gw_tp_showonlyactive'] = array('Aktive/Inaktive
Paare auflisten?', 'Bitte wählen Sie aus, ob nur aktive, inaktive oder alle Paare
gelistet werden sollen');
$GLOBALS['TL_LANG']['tl_module']['gw_tp_filteroptions'] = array('B' => 'Alle Paare
', 'A' => 'Nur aktive Paare', 'I' => 'Nur inaktive Paare');

$GLOBALS['TL_LANG']['tl_module']['gw_tp_couplesorting'] = array('Sortiermodus', 'B
itte wählen Sie aus, wie die Liste der Turnierpaare sortiert sein soll');
$GLOBALS['TL_LANG']['tl_module']['gw_tp_sortoptions'] = array('A' => 'Alphabetisch
', 'C' => 'Nach Altersklasse aufsteigend');
```

Hier kommt nun der Knackpunkt der "reference"-Arrays, der mir nicht klar war, und erst durch Abgucken bei anderen Extensions geklärt wurde:

Das reference-Array darf nicht einfach ein aufsteigendes Array der Texte zu den Optionen sein wie das "options"-Array (also "array('text1','text2','text3');"), sondern es muss ein assoziatives Array mit der Beziehung 'option1' => 'Text1' sein, sonst funktioniert es nicht. Die DCA-Referenz bleibt da leider sehr schwammig.

Auch die Abschnittsüberschrift "sort_legend" definieren wir hier.

Damit haben wir dieses Ergebnis:

Leider schaffe ich es in diesem Schritt nicht mehr, die neuen Felder in der Frontendausgabe auch noch auszuwerten (Der Kampf mit dem reference-Array hat doch etwas Zeit gekostet), das muss bis zum nächsten Mal (übernächste Woche) warten, sorry.

Schritt 8b: Parameter für das Modul, die zweite

Nun geht es daran, die Modulparameter aus Schritt 7 im Frontendmodul auch wirklich zu nutzen.
Dafür muss ich sie mir zuerst beschaffen. Das geschieht zu Beginn der compile()-Funktion in /system/modules/gw_turnierpaare/gwTurnierpaarliste.php:

PHP-Code:

```
        $moduleParams = $this->Database->prepare("SELECT * FROM tl_module WHERE id
=?")
                        ->limit(1)
                        ->execute($this->id);
```

$this->id enthält die Modul-ID, und über einen Blick in die Datenbank bekommen wir alle Felder der entsprechenden Zeile und damit auch unsere Parameter.

Die muss ich nun auswerten uns entsprechend reagieren. Zunächst das Flag, das mir nur die aktiven Paare, nur die inaktiven oder beide liefert:

PHP-Code:

```
    $whereClause = '';
    if($moduleParams->gw_tp_showonlyactive == 'A')
    {
      $whereClause = "WHERE aktiv='1'";
    }
    else
    {
      if($moduleParams->gw_tp_showonlyactive == 'I')
      {
        $whereClause = "WHERE aktiv=''";
      }
    }
```

Die Variable $whereClause baue ich später in meinen SELECT der Turnierpaare ein.

Für die Sortierung muss ich die alphabetische Sortierung nach Nachname Herr und Nachname Dame einerseits

und die Sortierung nach Startgruppe (und dann Startklasse und erst dann Nachnamen) unterscheiden.

Der erste Fall ist einfach ("ORDER BY partnernachname, partnerinnachname"), der andere aber komplizierter, da die Sortierrreihenfolge nicht alphabetisch ist, sondern sich nach den Altersklassen richten soll, die Jüngsten zuerst. Aber ein wenig Googeln hilft, das Problem mit einem SQL-Query zu lösen. Die FIELD()-Funktion liefert mir den Index des Inhalts eines Datenbankfeldes innerhalb einer Liste von Werten. Das kann ich benutzen, um in beliebiger Reihenfolge zu sortieren. Will ich das Feld "field" in der Reihenfolge "BCA" sortieren, leistet das "ORDER BY FIELD(field,'B','C','A')".

PHP-Code:

```
    $orderClause = 'partnernachname, partnerinnachname';
    if($moduleParams->gw_tp_couplesorting == 'C')
    {
      // Nach Altersgruppen sortieren
      $fieldstartgruppe="''";
       foreach(array('KIN I','KIN II', 'JUN I', 'JUN II', 'JUG', 'HGR', 'HGR II',
'SEN I', 'SEN II', 'SEN III', 'SEN IV') as $gruppe)
      {
        $fieldstartgruppe .= ",'".$gruppe."'";
      }

      $fieldstartklasse="''";
      foreach(array('-', 'E','D', 'C', 'B', 'A', 'S', 'PRO', 'LL', 'OL', 'RL', '2.
BL', '1. BL') as $klasse)
      {
        $fieldstartklasse .= ",'".$klasse."'";
      }

      $orderClause = "FIELD(startgruppe,".$fieldstartgruppe."), FIELD(startklasses
tandard,".$fieldstartklasse."), FIELD(startklasselatein,".$fieldstartklasse."), pa
rtnernachname, partnerinnachname";
    }
```

Defaultmäßig wird nach den Nachnamen alphabetisch sortiert. Wenn aber das gw_tp_couplesorting-Feld 'C' ist, dann wird aus den möglichen Werten der Altersgruppe und der Startklasse jeweils ein String zusammengebaut, der dann in $orderClause zum "ORDER BY"-Teil zusammengebaut wird.

So wird zuerst nach der Altersgruppe (in vorgegebener Reihenfolge), dann nach den Startklassen Standard und Latein (in vorgegebener Reihenfolge) und schließlich nach den Nachnamen sortiert. Natürlich hätte ich die Liste der Gruppen und Klassen direkt als String definieren können, statt ein Array anzulegen, mit einer Variable drüberzulaufen, um daraus wieder einen String zu machen.

Aber der Gedanke war, das 'options"-Feld der DCA-Definition der entsprechenden Felder im DCA-Record zu nutzen. Eine Änderung der möglichen Optionen dort würde dann auch gleich an dieser Stelle genutzt werden. Leider scheint die DCA-Definition des Backend-Moduls an dieser Stelle (Frontend-Modul!) leider nicht geladen zu sein, $GLOBALS['TL_DCA'] ist zumindest nicht vorhanden. Schade! Aber vielleicht lagere ich die Arrays noch in eine gemeinsame, geteilte Include-Datei aus, die ich in DCA-Definition und im Frontendmodul nutzen kann. Darum bleibt es erstmal bei der Schleife über den Arrayelementen.

Schließlich muss das ursprüngliche SELECT-Statement noch um die neuen Variablen $whereClause und $orderClause erweitert werden:

PHP-Code:

```
    $objPaare = $this->Database->execute("SELECT * FROM tl_gw_turnierpaare " . $wh
ereClause . "ORDER BY " . $orderClause);
```

Ich definiere nun zwei Module, einmal die Liste der aktiven Turnierpaare, die eben nur die aktiven Paare anzeigt, sortiert nach Startgruppen und Klassen, und eine Liste der inaktiven Paare, alphabetisch sortiert. Beide

Module füge in in eine Seite als Inhaltselemente ein.

Das Ergebnis:

Turnierpaare des Vereins

	Name	Startgruppe	Std	Lat	Aktiv seit	Aktiv bis	
	Standardformation		RL	-	2004		Detail
	Zick, Zacharias und Zick, Zäzilie	HGR	B	B	2002		Detail
	E, Dirk und W, Susanne	HGR	A	D	2002		Detail
	Abreger, Arne und Abreger, Alexandra	HGR	A	C	2003		Detail
	Kick, Kalle und Drop, Daniela	HGR II	C	-			Detail
	Jive, Jens und Samba, Susi	SEN I	A	-			Detail
	Walzer, Willi und Walzer, Wilma	SEN I	S	-			Detail
	Tango, Toni und Tango, Tina	SEN III	S	-	2000		Detail

Ehemalige Turnierpaare des Vereins

	Name	Startgruppe	Std	Lat	Aktiv seit	Aktiv bis	
	Abreger, Arne und Absteiger, Aurelia	HGR	-	B	1999	2001	Detail
	Aufreger, Anton und Aufreger, Anne	JUG	C	A	1999	2004	Detail

Oben das Modul, das die aktiven Turnierpaare sortiert nach Altersgruppe anzeigt, unten das Modul, das die inaktiven Paare alphabetisch sortiert anzeigt. Die graue Farbe bei den Inaktiven wird auch noch geändert, und im nächsten Schritt wird es an den "Detail"-Link gehen.

Schritt 9: Details, Details, Details!

In der Listenübersicht der Turnierpaare ist als Link bereits vorgesehen, dass man beim Klick auf den "Detail"-Link eines Paares zu einer Detail-Seite kommen kann. Ich möchte das mit demselben Modul regeln, also ohne Weiterleitungsseite auf eine (versteckte) Seite mit einem "DetailViewer"-Modul.

Ich habe mir das schamlos beim EFG abgeguckt. Meine Turnierpaarliste liegt unter turnierpaarliste.html. Wenn man als URL z.B. turnierpaarliste/info/8.html verwendet, wird trotzdem die Seite "turnierpaarliste" aufgerufen, aber jetzt hat der GET-Parameter "info" den Wert 8. Sehr praktisch.

Prüft man im Modulcode auf diesen Parameter, kann man dann gegebenenfalls auf das Detail-Template wechseln und anderen Code ausführen.

Nochmal der entsprechende Abschnitt aus dem Template der /system/modules/gw_turnierpaare/templates/gw_turnierpaarliste.tpl:

PHP-Code:

```
  </td>
  <td><?php echo '<a href="/turnierpaarliste/info/'.
$paar['id'].'.html">Detail</a>'; ?>
  </td>
```

Hier wird aus der Paar-ID der Detail-Link gebaut. Mich stört noch, dass mein Seitenname "turnierpaarliste" hardkodiert ist. Das werde ich noch ändern müssen, habe gerade aber keine Idee, wie ich an diese Information komme. Außerdem wäre es schöner, statt mit der numerischen ID mit einem alphanumerischen Alias zu arbeiten, so wie bei Seiten oder Artikeln. Das werde ich noch in einem zukünftigen Schritt implementieren.

Das Template für die Detailseite /system/modules/gw_turnierpaare/templates/gw_turnierpaarliste_detail.tpl sieht so aus:

PHP-Code:

```
<div class="<?php echo $this->class; ?> block paarvisitenkarte"<?php echo $this-
>cssID; ?>
<?php if ($this->style): ?> style="<?php echo $this->style; ?>"<?php endif; ?>>

<h3><?php echo $this->paar['partnernachname']; ?>
<?php if($this->paar['partnervorname']) { echo ", ".$this->paar['partnervorname'];
}; ?>
<?php if($this->paar['partnerinnachname']) { echo " und ".$this-
>paar['partnerinnachname']; }; ?>
<?php if($this->paar['partnerinvorname']) { echo ", ".$this-
>paar['partnerinvorname']; }; ?>
</h3>

<?php if ($this->paar['bild']): ?>
<div class="left">
<?php if($this->paar['bildfullsize']): ?>
<a href="<?php echo $this->paar['bildfullsize']; ?>" rel="lightbox[bild]">
<?php endif; ?>
<img src="<?php echo $this->paar['bild']; ?>">
<?php if($this->paar['bildfullsize']): ?>
</a>
<?php endif; ?>

</div>
<?php endif; ?>
```

```
<div class="right">
<ul>
<?php if ($this->paar['startklassestandard'] != '-'): ?>
<li>
  <?php echo $this->paar['startgruppe']." ".$this->paar['startklassestandard']." S
tandard"; ?>
</li>
<?php endif; ?>

<?php if ($this->paar['startklasselatein'] != '-'): ?>
<li>
  <?php echo $this->paar['startgruppe']." ".$this->paar['startklasselatein']." Lat
ein"; ?>
</li>
<?php endif; ?>

<li>
   Aktiv: <?php echo $this->paar['aktivseit']; ?> - <?php if($this-
>paar['aktivbis']) { echo $this->paar['aktivbis']; } else { echo "heute"; }; ?>
</li>

<?php if ($this->paar['anschrift']): ?>
<li>
   <?php echo nl2br($this->paar['anschrift']); ?>
</li>
<?php endif; ?>

<?php if ($this->paar['telefon']): ?>
<li>
   Tel.: <?php echo $this->paar['telefon']; ?>
</li>
<?php endif; ?>


<?php if ($this->paar['fax']): ?>
<li>
   Fax: <?php echo $this->paar['fax']; ?>
</li>
<?php endif; ?>


<?php if ($this->paar['mobil']): ?>
<li>
   Mob.: <?php echo $this->paar['mobil']; ?>
</li>
<?php endif; ?>


<?php if ($this->paar['email']): ?>
<li>
   EMail: {{email::<?php echo $this->paar['email']; ?>}}
</li>
<?php endif; ?>

<?php if ($this->paar['homepage']): ?>
<li>
   Homepage: <a href="<?php echo $this->paar['homepage']; ?>"><?php echo $this-
>paar['homepage']; ?></a>
</li>
<?php endif; ?>

</ul>
</div>
```

```
<div class="twocol">
<?php echo nl2br($this->paar['beschreibung']); ?>
</div>

</div>
```

Im Prinzip nichts besonderes, ich bastele aus den Namen eine schöne Überschrift, Bild und die anderen Texte werden in DIVs verpackt, die ich per CSS dann "hübsch" anordne. Beim Bild wird unser Modulcode in $paar['bild'] eine verkleinerte Version (180px breit max.) zurückliefern, in $paar['bildfullsize'] befindet sich das Originalbild, das hier im Template dann über eine Lightbox großklickbar ist. Das Array $paar im Template entspricht ansonsten den Datenbankfeldern in tl_gw_turnierpaare.

Die Hauptarbeit passiert nun in der Frontend-Modul-Klasse /system/modules/gw_turnierpaare/gwTurnierpaarliste.php:

PHP-Code:

```
class gwTurnierpaarliste extends Module
{

    /**
     * Template
     * @var string
     */
    protected $strTemplate = 'gw_turnierpaarliste';

  protected $strDetailKey = 'info';
```

Zunächst definiere ich wie üblich den "Default"-Templatenamen, das ist der für die gesamte Liste. Zusätzlich definiere ich hier das URL-Fragment, was meinen "Detail-Link" ausmachen soll.

PHP-Code:

```
  function obj2Arr($objPaar)
  {
    $newArr = array
      (
        'partnernachname' => trim($objPaar->partnernachname),
        'partnervorname' => trim($objPaar->partnervorname),
        'partnerinnachname' => trim($objPaar->partnerinnachname),
        'partnerinvorname' => trim($objPaar->partnerinvorname),
        'startgruppe' => $objPaar->startgruppe,
        'startklasselatein' => $objPaar->startklasselatein,
        'startklassestandard' => $objPaar->startklassestandard,
        'aktiv' => $objPaar->aktiv,
        'aktivseit' => $objPaar->aktivseit,
        'aktivbis' => $objPaar->aktivbis,
        'id' => $objPaar->id,
        'beschreibung' => $objPaar->beschreibung,
      );

      if($objPaar->zeigeanschrift == '1')
      {
        $newArr['anschrift'] = $objPaar->anschrift;
      }

      if($objPaar->zeigetelefon == '1')
      {
        $newArr['telefon'] = $objPaar->telefon;
      }
```

```
      if($objPaar->zeigefax == '1')
      {
        $newArr['fax'] = $objPaar->fax;
      }

      if($objPaar->zeigemobil == '1')
      {
        $newArr['mobil'] = $objPaar->mobil;
      }

      if($objPaar->zeigeemail == '1')
      {
        $newArr['email'] = $objPaar->email;
      }

      if($objPaar->zeigehomepage == '1')
      {
        $newArr['homepage'] = $objPaar->homepage;
      }

    return $newArr;
  }
```

Diese Utility-Funktion füllt mir eine Zeile des Resultsets der Datenbank in ein Array. Ich brauche das an zwei Stellen, darum habe ich das hierhin ausgelagert. Hier werden auch schon die "zeige"-Flags berücksichtigt: Sind die nicht angehakt, landet auch nichts im Array. Damit spare ich mir die Abfragen im Template.

PHP-Code:

```
    protected function compile()
    {
        if ( strlen($this->Input->get($this->strDetailKey)) )
        {
      // A "detail"-URL is given -> Output turnierpaarliste_detail template
      $this->compileDetailTemplate();

    }
    else
    {
      // Output the turnierpaarliste template
      $this->compileListTemplate();
      }
}
```

Die alte "Compile"-Funktion wurde angenehm kurz. Falls der GET-Parameter mit dem Namen "info" gesetzt ist (über unsere URL turnierpaarliste/info/8.html), dann wird eine Funktion zur Ausgabe des Detail-Templates aufgerufen, ansonsten eine für das Listentemplate.

PHP-Code:

```
  // Compiles the turnierpaarliste template
  protected function compileListTemplate()
  {
        $moduleParams = $this->Database->prepare("SELECT * FROM tl_module WHERE id
=?")
                ->limit(1)
                ->execute($this->id);

    $whereClause = '';
    if($moduleParams->gw_tp_showonlyactive == 'A')
    {
```

```
      $whereClause = "WHERE aktiv='1'";
    }
    else
    {
      if($moduleParams->gw_tp_showonlyactive == 'I')
      {
        $whereClause = "WHERE aktiv=''";
      }
    }

    $orderClause = 'partnernachname, partnerinnachname';
    if($moduleParams->gw_tp_couplesorting == 'C')
    {
      // Nach Altersgruppen sortieren
      $fieldstartgruppe="''";
       foreach(array('KIN I','KIN II', 'JUN I', 'JUN II', 'JUG', 'HGR', 'HGR II',
'SEN I', 'SEN II', 'SEN III', 'SEN IV') as $gruppe)
      {
        $fieldstartgruppe .= ",'".$gruppe."'";
      }

      $fieldstartklasse="''";
      foreach(array('-', 'E','D', 'C', 'B', 'A', 'S', 'PRO', 'LL', 'OL', 'RL', '2.
BL', '1. BL') as $klasse)
      {
        $fieldstartklasse .= ",'".$klasse."'";
      }

      $orderClause = "FIELD(startgruppe,".$fieldstartgruppe."), FIELD(startklasses
tandard,".$fieldstartklasse."), FIELD(startklasselatein,".$fieldstartklasse."), pa
rtnernachname, partnerinnachname";
    }

    $arrPaare = array();

    $objPaare = $this->Database->execute("SELECT * FROM tl_gw_turnierpaare " . $wh
ereClause . "ORDER BY " . $orderClause);

    while ($objPaare->next())
    {
      $newArr = $this->obj2Arr($objPaare);

      if(strlen($objPaare->bild) == 0)
      {
        $newArr['bild'] = '/system/modules/gw_turnierpaare/icons/default.png';
      }
      else
      {
        $newArr['bild'] = $this->getImage($objPaare->bild, '', '48');
      }

      $arrPaare[] = $newArr;
    }

    $this->Template->paare = $arrPaare;
  }
```

Bei der Ausgabe des Listentemplates hat sich nicht viel gegenüber dem letzten Schritt getan, nur die Utility-Funktion obj2Arr wird jetzt benutzt, um den Großteil der Felder aus der Datenbank ins Array zu packen. Als Paarbild wird ein Thumbnail von max. 48 Pixeln Höhe ausgegeben.

PHP-Code:

```
  // Compiles the data for the turnierparliste_detail template
  protected function compileDetailTemplate()
  {
    $coupleRow = $this->Database->prepare("SELECT * FROM tl_gw_turnierpaare WHERE
id=?")
              ->limit(1)
              ->execute($this->Input->get($this->strDetailKey));

    $this->strTemplate = 'gw_turnierpaarliste_detail';
    $this->Template = new FrontendTemplate($this->strTemplate);

    $newArr = $this->obj2Arr($coupleRow);

    if(strlen($coupleRow->bild) == 0)
    {
      $newArr['bild'] = '/system/modules/gw_turnierpaare/icons/default.png';
    }
    else
    {
      $newArr['bild'] = $this->getImage($coupleRow->bild, '180', '');
      $newArr['bildfullsize'] = $coupleRow->bild;
    }

    $this->Template->paar = $newArr;
  }
```

Hier die Ausgabe des Detail-Templates: Zunächst holen wir uns das Paar aus der Datenbank, dessen Detailseite ausgegeben werden soll. Hier wäre wohl zukünftig noch eine Fehlerabfrage ("ID existiert nicht") notwendig. *Hüstel*

Dann müssen wir das andere Template wählen. Dafür ersetze ich den Namen durch den Neuen. Das alleine reichte aber nicht: Es erschien immer noch das alte Template, weil es schon initialisiert war, als compile() aufgerufen wurde.

Eine Änderung von $strTemplate bleibt dann unbemerkt. Darum muss jetzt in der nächsten Zeile ein neues Frontendtemplate instanziert werden. Ich hätte den Namen auch direkt als Parameter bei der Instanzierung angeben können. Dann fülle ich mit obj2Arr() mein Paar-Array, und gebe (falls vorhanden) das Paarbild-Thumbnail mit maximal 180 Pixel Breite aus, und zusätzlich das Originalbild.

Das Ergebnis ist das Folgende – Turnierpaarliste:

Ich habe das Default-Bild noch ausgetauscht, das angezeigt wird, wenn kein Paarbild hinterlegt ist (Ich hänge

es auch hier nochmal an). Das "richtige" Paarbild ist übrigens ein freies aus Wikimedia Commons.

Nach Klick auf einen Detaillink:

Turnierpaare des Vereins

	Name	Startgruppe	Std	Lat	Aktiv seit	Aktiv bis	
	Standardformation		RL	-	2004		Detail
	Zick, Zacharias und Zick, Zäzilie	HGR	B	B	2002		Detail
	E, Dirk und W, Susanne	HGR	A	D	2002		Detail
	Abreger, Arne und Abreger, Alexandra	HGR	A	C	2003		Detail
	Kick, Kalle und Drop, Daniela	HGR II	C	-			Detail
	Jive, Jens und Samba, Susi	SEN I	A	-			Detail
	Walzer, Willi und Walzer, Wilma	SEN I	S	-			Detail
	Tango, Toni und Tango, Tina	SEN III	S	-	2000		Detail

Schritt 10: Etwas Feinschliff

Einige Details will ich noch verbessern, bevor es endlich an die noch fehlenden Turniermeldungen geht.

Insgesamt ist es ein Mischmasch von vielen Kleinigkeiten, die aber teilweise in mehreren Dateien Auswirkungen haben. Es könnte also chaotisch werden :-).

Beginnen wir wir mit /system/modules/gw_turnierpaare/geTurnierpaarliste.php , der Klasse in der die Frontendtemplates gefüllt werden.

PHP-Code:

```
class gwTurnierpaarliste extends Module
{

    /**
     * Template
     * @var string
     */
    protected $strTemplate = 'gw_turnierpaarliste';

    protected $strDetailTemplate = 'gw_turnierpaarliste_detail';

    protected $strDetailErrorTemplate = 'gw_turnierpaarliste_error';


  public static $strDetailKey = 'info';

  public static $StartGruppen = array('KIN I','KIN II', 'JUN I', 'JUN II', 'JUG',
'HGR', 'HGR II', 'SEN I', 'SEN II', 'SEN III', 'SEN IV');

  public static $StartKlassen = array('-', 'E','D', 'C', 'B', 'A', 'S', 'PRO', 'LL
', 'OL', 'RL', '2. BL', '1. BL');
```

Zunächst definiere ich mir neben meinem "Defaulttemplate" noch einen Templatenamen für die Detailansicht, und ein "Error-Template", das angezigt wird, wenn das für die Detailansicht gewünschte Tanzpaar nicht gefunden wird.

Dann folgen 3 statische Variablendeklarationen, dies sind also Klassenvariablen. Die werde ich Klassenübergreifend nutzen, weil die entsprechenden Einträge in mehreren Files genutzt werden, und ich sie nur an einer Stelle editieren möchte. $strDetailKey benötigen wir im Listentemplate, die beiden anderen Variablen im DCA-Record für die Tabelle tl_gw_turnierpaare und für die sortierte Ausgabe nach Startgruppe und -klasse.

Hier verwenden wir sie dann auch (Funktion compileListTemplate()):

PHP-Code:

```
...
      // Nach Altersgruppen sortieren
      $fieldstartgruppe="''";
       foreach(gwTurnierpaarliste::$StartGruppen as $gruppe)
      {
        $fieldstartgruppe .= ",'".$gruppe."'";
      }

      $fieldstartklasse="''";
      foreach(gwTurnierpaarliste::$StartKlassen as $klasse)
      {
```

```
        $fieldstartklasse .= ",'".$klasse."'";
      }
...
```

Und auch die Verwendung von $strDetailKey wird angepasst:

PHP-Code:

```
...
        if ( strlen($this->Input->get(gwTurnierpaarliste::$strDetailKey)) )
...
```

Im DCA-Record für die Felder startgruppe, startklasselatein und startklassestandard sinngemäß (Datei /system/modules/gw_turnierpaare/dca/tl_gw_turnierpaare.php):

PHP-Code:

```
...
        'startgruppe' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['startgruppe'],
            'inputType'               => 'select',
            'sorting'                 => true,
            'options'                 => gwTurnierpaarliste::$StartGruppen,
            'eval'                    => array('mandatory'=>false, 'includeBlankOp
tion' => true)
        ),
        'startklasselatein' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['startklasselatein'],
            'inputType'               => 'select',
            'sorting'                 => true,
            'options'                 => gwTurnierpaarliste::$StartKlassen,
            'eval'                    => array('mandatory'=>true, 'tl_class' => 'w
50')
        ),
        'startklassestandard' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['startklassestandard'],
            'inputType'               => 'select',
            'sorting'                 => true,
            'options'                 => gwTurnierpaarliste::$StartKlassen,
            'eval'                    => array('mandatory'=>true)
        ),
...
```

und schließlich noch im Template für die Übersichtsliste /system/modules/gw_turnierpaare/templates/gw_turnierpaarliste.tpl:

PHP-Code:

```
...
  <td><?php echo '<a href="/{{env::page_alias}}/'.gwTurnierpaarliste::
$strDetailKey.'/'.$paar['alias'].'.html">Detail</a>'; ?>
...
```

Hier benutzen wir auch das Insert-Tag {{env:age_alias}}, um den Seitenalias nicht wie bisher hart im Template kodiert zu haben. Das funktioniert jetzt, egal wie die Seite heißt, auf der das Frondendmodul

eingebunden wird.

Hier sieht man auch noch eine andere Änderung: $paar['alias']. Ich möchte die Detailseite nicht nur über die numerische ID des Paares aufrufen können, sondern eleganter über einen Seitenalias, z.b. /turnierpaarliste/info/mueller.html .

Dafür definiere ich ein neues varchar(128)-Feld in der tl_gw_turnierpaare-Tabelle (/system/modules/gw_turnierpaare/config/database.sql):

Code:

```
...
  `partnerinvorname` varchar(64) NULL default NULL,
  `alias` varchar(128) NOT NULL default '',
  `startgruppe` varchar(32) NOT NULL default '',
...
```

Die Definitionen für den DCA-Record klaue ich mir bei tl_article.php aus dem Backend (/system/modules/gw_turnierpaare/dca/tl_gw_turnierpaare.php):

PHP-Code:

```
...
        'alias' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']
['tl_gw_turnierpaare']['alias'],
            'exclude'                 => true,
            'inputType'               => 'text',
            'eval'                    => array('rgxp'=>'alnum', 'doNotCopy'=>true,
'spaceToUnderscore'=>true, 'maxlength'=>128, 'tl_class'=>'w50'),
            'save_callback' => array
            (
                array('tl_gw_turnierpaare', 'generateAlias')
            )
    ),
...
```

In die Palette tragen wir das Feld auch noch ein, damit es manuell editierbar bleibt:

PHP-Code:

```
    // Palettes
    'palettes' => array
    (
        '__selector__'                => array('resetpassword'),
        'default'                     => '{name_legend},partnernachname,partnervor
name,partnerinnachname,partnerinvorname,alias;'
                                      .'{classes_legend},startgruppe,startklassela
tein,startklassestandard;'
...
```

Und wir müssen den Callback schreiben

PHP-Code:

```
  /**
   * Generate alias from couples names
```

```
   */
    public function generateAlias($varValue, DataContainer $dc)
    {
        $autoAlias = false;

        // Generate alias if there is none
        if (!strlen($varValue))
        {
            $autoAlias = true;
            $key = $dc->activeRecord->partnernachname;
            if(strlen($dc->activeRecord->partnerinnachname) > 0 && strcmp($dc-
>activeRecord->partnernachname,$dc->activeRecord->partnerinnachname))
            {
        $key = $key.'_'.$dc->activeRecord->partnerinnachname;
      }
            $varValue = standardize($key);
        }

        $objAlias = $this->Database->prepare("SELECT id FROM tl_gw_turnierpaare WH
ERE id=? OR alias=?")
                                   ->execute($dc->id, $varValue);

        // Check whether the page alias exists
        if ($objAlias->numRows > 1)
        {
            if (!$autoAlias)
            {
                throw new Exception(sprintf($GLOBALS['TL_LANG']['ERR']
['aliasExists'], $varValue));
            }

            $varValue .= '-' . $dc->id;
        }

        return $varValue;
    }
```

Der Seitenalias ist nachnameherr_nachnamedame, wenn die Nachnamen verschieden sind, falls sie gleich sind nur nachname. Ein Paar Müller/Schulze wäre also mueller_schulze, ein Ehepaar Maier nur maier. Das ganze wird mit einer eingebauten TL-Funktion standardisiert. Sollte der Alias schon existieren (eher unwahrscheinlich), dann wird die ID hintenan gehängt. Das ist 1:1 so wie bei Seitenaliasen in TL auch.

Im Frontendmodul /system/modules/gw_turnierpaare/gwTurnierpaarliste.php erfolgen dafür nun auch noch Anpassungen. Zunächst in der Funktion obj2Arr, damit das neue Feld auch im Template zur Verfügung steht:

PHP-Code:

```
...
        'id' => $objPaar->id,
        'alias' => $objPaar->alias,
        'beschreibung' => $objPaar->beschreibung,
...
```

und schließlich in compileDetailTemplate:

PHP-Code:

```
  // Compiles the data for the turnierparliste_detail template
  protected function compileDetailTemplate()
  {
    $coupleRow = $this->Database->prepare("SELECT * FROM tl_gw_turnierpaare WHERE
id=? OR alias=?")
```

```
            ->limit(1)
            ->execute($this->Input->get(self::$strDetailKey),$this->Input-
>get(self::$strDetailKey));

    if($coupleRow->numRows == 0)
    {
          $this->Template = new FrontendTemplate($this->strDetailErrorTemplate);
    }
    else
    {
      $this->Template = new FrontendTemplate($this->strDetailTemplate);

      $newArr = $this->obj2Arr($coupleRow);

      if(strlen($coupleRow->bild) == 0)
      {
        $newArr['bild'] = '/system/modules/gw_turnierpaare/icons/default.png';
      }
      else
      {
        $newArr['bild'] = $this->getImage($coupleRow->bild, '180', '');
        $newArr['bildfullsize'] = $coupleRow->bild;
      }

      $this->Template->paar = $newArr;
    }
  }
```

Hier wird entweder nach ID oder nach Alias gesucht. Aufruf der Detailseite würde also sowohl über /info/12.html als auch info/mueller.html funktionieren. Wird kein Datensatz mit dieser ID gefunden, wird das Error-Template ausgegeben, ansonsten wie üblich der Detail-Datensatz.

Schließlich noch das sehr simple neue Template system/modules/gw_turnierpaare/templates/gw_turnierpaarliste_error.tpl:

PHP-Code:

```
<div class="<?php echo $this->class; ?> block paarvisitenkarte"<?php echo $this-
>cssID; ?>
<?php if ($this->style): ?> style="<?php echo $this->style; ?>"<?php endif; ?>>

<h3>Dieses Paar existiert leider nicht!</h3>
```

Da sich rein optisch außer einem Extra Feld im Backend für den Alias nichts getan hat, verzichte ich in diesem Schritt auf Screenshots. Leider ging es auch etwas kreuz und quer mit kleinen Änderungen in mehreren Dateien. Ich hoffe trotzdem man konnte noch folgen...

Jürgen

Eine Frage zu der Demoanwendung am Anfang. Diese habe ich installiert und das Backend funktioniert auch gut. Nur das erstellen eines Moduls macht mir Schwierigkeiten. Da scheint Typolight sich aufzuhängen. Der Bildschirm bleibt einfach weiß.

Ich hab es local unter winxp pro oder win7 pro unter xampp laufen. Ich hab mal displayErrors aktiviert und als erstes kommt die Fehlermeldung

Warning: Cannot modify header information - headers already sent by (output started at G:\xampplite\htdocs\test\system\modules\gw_turnier paare\dca\tl_module.php:32) in G:\xampplite\htdocs\test\system\libraries\Template .php on line 174

---

dl1ely

bitte öffne /system/modules/gw_turnierpaare/dca/tl_module.php in einem Editor. Zeilen 31 und 32 sind Leerzeilen, bitte entferne die. Die letzte Zeile muss Zeile 30 mit "?>" sein.

Ich denke, daran wird es liegen. Ich vermute, das hängt mit den unterschiedlichen Zeilenumbrüchen bei Linux/Windows zusammen, bei mir tritt das Problem zumindest nicht auf. Dadurch werden wohl zwei Leerzeilen ausgegeben, bevor dann in Zeile 174 von system/libraries/Template.php der Header für die Seite gesetzt wird. Das schlägt dann natürlich fehl. In Zukunft werde ich mehr auf die Leerzeilen-Problematik achten. In der nächsten "Version" des Quelltexts wird das behoben sein.

Schritt 11: Die zweite Tabelle

Ich habe mich lange davor gedrückt, aber jetzt muss es an meine zweite Tabelle gehen: tl_gw_meldungen.

Meldungen sind Teilnahmen von Turnierpaaren an Turnieren. Sie bestehen aus dem Datum der Turnierteilnahme, dem Ort, der Startgruppe (Altersklasse) und der Startklasse (aus Regeltechnischen Gründen müssen die NICHT zwingend mit den entsprechenden Werten, die beim Turnierpaar eingetragen sind übereinstimmen), der Tanzart (Standard/Lateinamerikanisch), der Turnierart (Hier unterscheidet man "offene Turniere", "Einladungsturniere", "Landesmeisterschaften", "Deutsche Meisterschaften" usw), der Anzahl der gestarteten Paare, dem Platz (Da es geteilte Plätze wie 5.-7. geben kann gibt es "platz_von" und "platz_bis"), und einem Freitext als Bemerkung.

Soviel zur Domain specific knowledge.

Schauen wir uns nochmal die SQL-Definition in config/database.sql an:

Code:

```
--
-- Table `tl_gw_meldungen`
--

CREATE TABLE `tl_gw_meldungen` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `pid` int(10) unsigned NOT NULL default '0',
  `sorting` int(10) unsigned NOT NULL default '0',
  `tstamp` int(10) unsigned NOT NULL default '0',
  `datum` date NOT NULL default '2000-01-01',
  `startgruppe` varchar(32) NOT NULL default '',
  `startklasse` varchar(12) NOT NULL default '',
  `lat_std` char(32) NOT NULL default '',
  `turnierort` varchar(128) NOT NULL default '',
  `turnierart` varchar(64) NULL default NULL,
  `anzahlpaare` int(4) NULL default NULL,
  `platz_von` int(4) NULL default NULL,
  `platz_bis` int(4) NULL default NULL,
  `bemerkung` text NULL,
  PRIMARY KEY  (`id`),
  KEY `pid` (`pid`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Die ersten 4 Felder sind ja so "default". Hier macht pid (Parent-ID) aber auch Sinn. Die Meldungen sind "Childs" der Turnierpaare. Im Backend wollte ich auch so vorgehen, also bei den Turnierpaaren auf einen Tree-View umstellen. Im DCA-Record für tl_gw_turnierpaare habe ich also unter "sort" den "mode" auf 5 gestellt, und im DCA-Record für tl_gw_meldungen auf 6.

Dann straft mich TYPOlight aber mit der SQL-Fehlermeldung, dass es in der Tabelle tl_gw_turnierpaare kein Feld "pid" gäbe. Richtig, gibt es auch nicht. Da das die Parent-Tabelle ist, braucht die auch kein "pid". Nun gut, offensichtlich erfordert die Logik das so, also erfülle ich den Wunsch und spendiere auch tl_gw_turnierpaare ein Feld "pid".

Beim Betrachten im Backend fällt mir aber auf, dass das unpraktisch ist: Der Sportwart, der das im Backend pflegt, den interessiert nur die nach dem Turnierdatum sortierte Liste, neueste ganz oben. Der will gar nicht erst das Turnierpaar suchen, um dort eine Meldung einzugeben.

Ich entscheide mich, Turnierpaare und Meldungen im Backend getrennt zu verwalten, aber bei den Meldungen dann eine foreign-key-Beziehung über die pid zur Turnierpaartabelle zu haben. Wenn der Sportwart eine neue Meldung anlegt, soll er über ein Dropdown das Turnierpaar auswählen, zu dem diese Meldung gehört, die Liste selbst soll aber nach dem Datum absteigend sortiert sein.

Ich brauche also eine weitere Seite im Backend, und modifiziere /system/modules/gw_turnierpaare/config/config.php auf:

PHP-Code:

```
// Back end module
$GLOBALS['BE_MOD']['content']['gw_turnierpaare'] = array
(
    'tables' => array('tl_gw_turnierpaare'),
    'icon'   => 'system/modules/gw_turnierpaare/icons/turnierpaare.png'
);

$GLOBALS['BE_MOD']['content']['gw_meldungen'] = array
(
    'tables' => array('tl_gw_meldungen'),
    'icon'   => 'system/modules/gw_turnierpaare/icons/meldeliste.png'
);
```

Also zwei hier formal getrennte Tabellen mit getrennten Backendseiten und verschiedenen Icons (meldeliste.png hänge ich hier an, das soll ein Siegerpodest darstellen).

Dann gehts an DCA für die Meldungstabelle, in /system/modules/gw_turnierpaare/dca/tl_gw_meldungen.php:

PHP-Code:

```
/**
 * Table tl_gw_meldungen
 */
$GLOBALS['TL_DCA']['tl_gw_meldungen'] = array
(

    // Config
    'config' => array
    (
        'dataContainer'               => 'Table',
        'enableVersioning'            => true,
    ),
```

Hier keine Besonderheiten, es ist eine Tabelle mit Versionierung...

PHP-Code:

```
    // List
    'list' => array
    (
        'sorting' => array
        (
            'mode'                    => 1,
            'fields'                  => array('datum DESC', 'turnierort'),
            'panelLayout'             => '',
            'flag'                    => 8,
        ),
```

Sortierung nach festem Feld, nämlich dem absteigenden Datum, und dann nach dem Turnierort alphabetisch. Das Panel für Sortieren, Filtern usw lasse ich erstmal weg (Das sorgte nämlich für eine kryptische Fehlermeldung - ich kümmere mich später darum). flag 8 ist das absteigende Sortieren nach dem Monat. Zu meinem Problem damit komme ich später.

PHP-Code:

```
        'label' => array
        (
            'fields'                  => array('datum', 'turnierort', 'turnierart'
, 'startgruppe','startklasse','lat_std','pid'),
            'format'                  => '<span style="font-weight: bold;">
%s</span>, %s (%s), %s %s %s - %s'
        ),
```

Hier bastele ich mir die Ausgabezeile, mit dem Datum in fett, dann dem Ort, der Turnierart, Startgruppe und Klasse und der Info, ob Standard oder Lateinamerikanisch. Dann müsste dort eigentlich noch der Name des Tanzpaares hin, aber der steht ja nicht (direkt) in der Tabelle, sondern nur die pid. Darum nehme ich erstmal die - besser als Nix. Ich vermute, ich muss/kann da mit einem Label-Callback arbeiten und die pid selbst in die Namen auflösen.

PHP-Code:

```
        'global_operations' => array
        (
            'all' => array
            (
                'label'               => &$GLOBALS['TL_LANG']['MSC']['all'],
                'href'                => 'act=select',
                'class'               => 'header_edit_all',
                'attributes'          => 'onclick="Backend.getScrollOffset();"'
            )
        ),
        'operations' => array
        (
            'edit' => array
            (
                'label'               => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['edit'],
                'href'                => 'act=edit',
                'icon'                => 'edit.gif'
            ),
            'copy' => array
            (
                'label'               => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['copy'],
                'href'                => 'act=copy',
                'icon'                => 'copy.gif'
            ),
            'delete' => array
            (
                'label'               => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['delete'],
                'href'                => 'act=delete',
                'icon'                => 'delete.gif',
                'attributes'          => 'onclick="if (!confirm(\'' . $GLOBAL
S['TL_LANG']['MSC']['deleteConfirm'] . '\')) return false; Backend.getScrollOffset
();"'
            ),
            'show' => array
            (
                'label'               => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['show'],
                'href'                => 'act=show',
                'icon'                => 'show.gif'
            )
        )
    ),
```

Hier bleibt erstmal alles so, wie vom Extension Creator vorgegeben.

PHP-Code:

```
    // Palettes
    'palettes' => array
    (
        '__selector__'                => array(''),
        'default'                     => '{couple_legend},pid;
{tournament_legend},datum,turnierort,turnierart,startgruppe,startklasse,lat_std;'
                                      .'{result_legend},anzahlpaare,platz_von,platz_
bis,bemerkung;'
    ),

    // Subpalettes
    'subpalettes' => array
    (
        ''                            => ''
    ),
```

Auch hier nichts spannendes: Eine normale Palette für alle Felder.

PHP-Code:

```
    // Fields
    'fields' => array
    (
        'pid' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['pid'],
            'inputType'               => 'select',
            'foreignKey'              => 'tl_gw_turnierpaare.partnernachname',
            'search'                  => true,
            'sorting'                 => true,
            'eval'                    => array('mandatory'=>true)
        ),
```

pid soll ein Foreign Key in die Turnierpaare-Tabelle sein. mir der foreignKey-Option wird das Dropdown-Feld mit den Partnernachnamen aus der Turnierpaar-Tabelle gefüllt. Zum ersten Testen ist das ganz OK, aber eigentlich stelle ich mir das anders vor: Es sollen dort nur AKTIVE Paare auswählbar sein, und ich hätte dort gerne die kompletten Namen des Turnierpaares stehen, nicht nur den Nachnamen des Herrn. Auch da wird wohl ein Callback hermüssen - später.

PHP-Code:

```
        'datum' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['datum'],
            'inputType'               => 'text',
            'search'                  => true,
            'sorting'                 => true,
            'flag'                    => 11,
            'eval'                    => array('mandatory'=>true, 'datepicker'=>$t
his->getDatePickerString(), 'tl_class'=>'w50 wizard', 'minlength' => 1, 'maxlength
'=>64, 'rgxp' => 'date')
        ),
```

Abgeguckt beim DCA-Record für Artikel: Das Datumsfeld. Insbesondere den getDatePickerString() verstehe ich nicht - muss ich aber auch erstmal nicht. Kommt Zeit, kommt Erleuchtung.

PHP-Code:

```
        'turnierort' => array
        (
            'label'                 => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['turnierort'],
            'inputType'             => 'text',
            'search'                => true,
            'sorting'               => true,
            'flag'                  => 1,
            'eval'                  => array('mandatory'=>true, 'minlength' => 1
, 'maxlength'=>128, 'tl_class' => 'w50')
        ),
```

Nichts Besonderes hier...

PHP-Code:

```
        'turnierart' => array
        (
            'label'                 => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['turnierart'],
            'inputType'             => 'select',
            'sorting'               => false,
            'options'               => gwTurnierpaarliste::$TurnierArten,
            'eval'                  => array('mandatory'=>false, 'includeBlankOp
tion' => true, 'tl_class' => 'w50')
        ),
```

Ähnlich wie bei den Startgruppen und Klassen lagere ich die Optionen für "Turnierart" in meine Frontendklasse aus. Gefällt mir besser so, und ich kann es "Re-usen".

PHP-Code:

```
        'startgruppe' => array
        (
            'label'                 => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['startgruppe'],
            'inputType'             => 'select',
            'sorting'               => true,
            'flag'                  => 1,
            'options'               => gwTurnierpaarliste::$StartGruppen,
            'eval'                  => array('mandatory'=>false, 'includeBlankOp
tion' => true, 'tl_class' => 'w50')
        ),
        'startklasse' => array
        (
            'label'                 => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['startklasse'],
            'inputType'             => 'select',
            'sorting'               => false,
            'options'               => gwTurnierpaarliste::$StartKlassen,
            'eval'                  => array('mandatory'=>true, 'tl_class' => 'w
50')
        ),
        'lat_std' => array
        (
            'label'                 => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['lat_std'],
            'inputType'             => 'select',
            'sorting'               => false,
            'options'               => gwTurnierpaarliste::$TanzArten,
```

```
            'eval'                      => array('mandatory'=>true, 'tl_class' => 'w
50')
        ),
```

Wie zuvor, normale Dropdownfelder, deren Optionen ich in der Frontendklasse gwTurnierpaarliste ablege, um sie von verschiedenen Bereichen aus nutzen zu können.

PHP-Code:

```
        'anzahlpaare' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['anzahlpaare'],
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'minlength' =>
1, 'maxlength'=>4, 'rgxp' => 'digit')
        ),
        'platz_von' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['platz_von'],
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'minlength' =>
1, 'maxlength'=>4, 'rgxp' => 'digit', 'tl_class' => 'w50')
        ),
        'platz_bis' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['platz_bis'],
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'minlength' =>
1, 'maxlength'=>4, 'rgxp' => 'digit', 'tl_class' => 'w50')
        ),
        'bemerkung' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['bemerkung'],
            'inputType'                 => 'textarea',
            'eval'                      => array('mandatory'=>false, 'cols' => 80, '
rows' => 20, 'allowHtml' => false)
        ),
    )
);
```

Auch hier eigentlich "Hausmannskost": 3 Felder für Zahlen und ein Textfeld für die Bemerkung.

/system/modules/gw_turnierpaare/gwTurnierpaarliste.php (das Frontend-Modul) erweitere ich noch um die Optionenlisten für Tanzart und Turnierart:

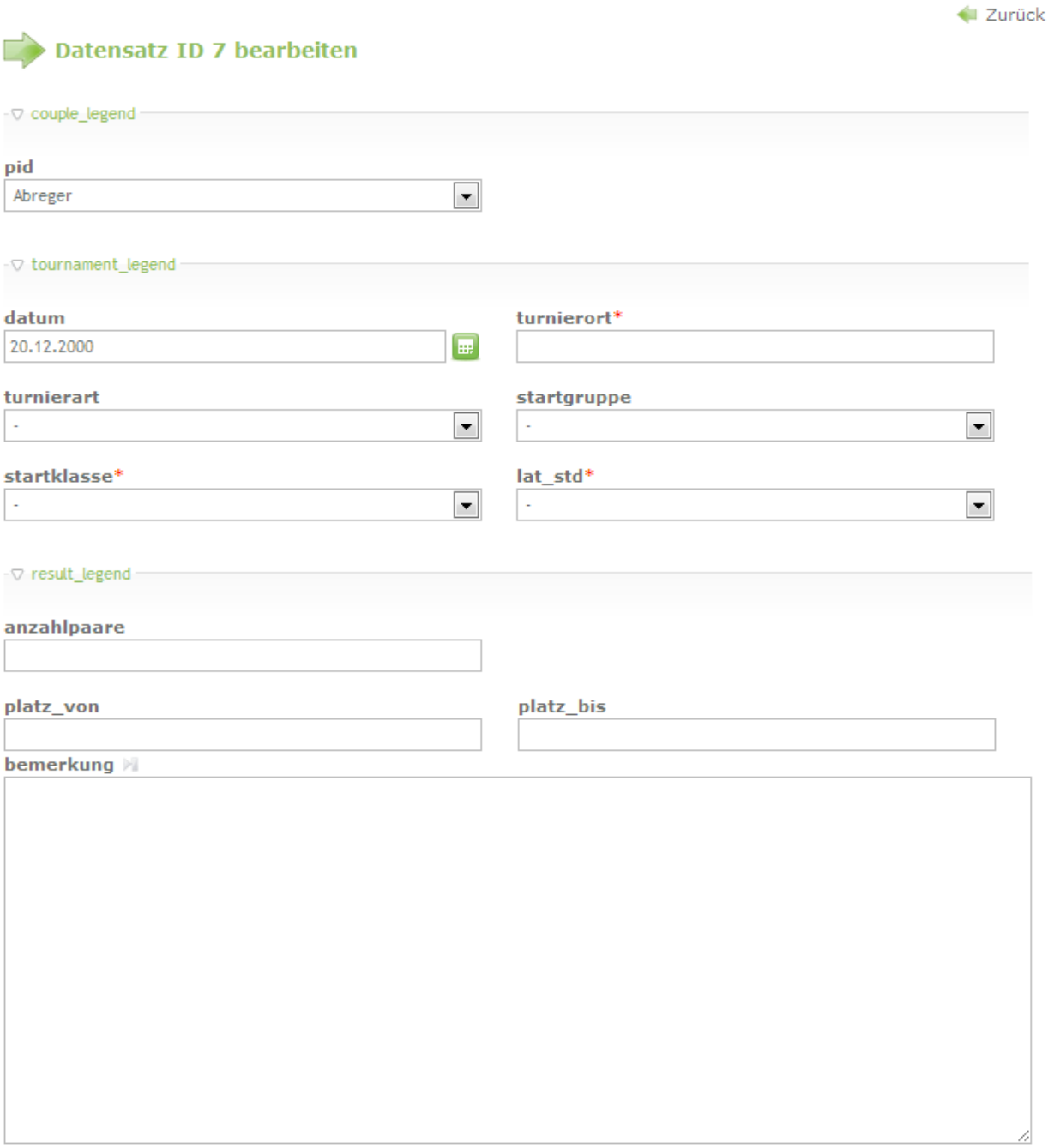
PHP-Code:

```
  public static $TurnierArten = array('-', 'OT','ET', 'LM', 'DM', 'EM', 'WM');

  public static $TanzArten = array('-', 'Std','Lat');
```

Zum ersten Testen füge ich zwei Testdatensätze in die Meldungstabelle ein. Ergebnis:

Gut, die Tabellenheader sehen noch nicht so aus wie gewünscht, aber das ist Feinschliff. Die gewünschten Informationen (mit pid 7) werden angezeigt.

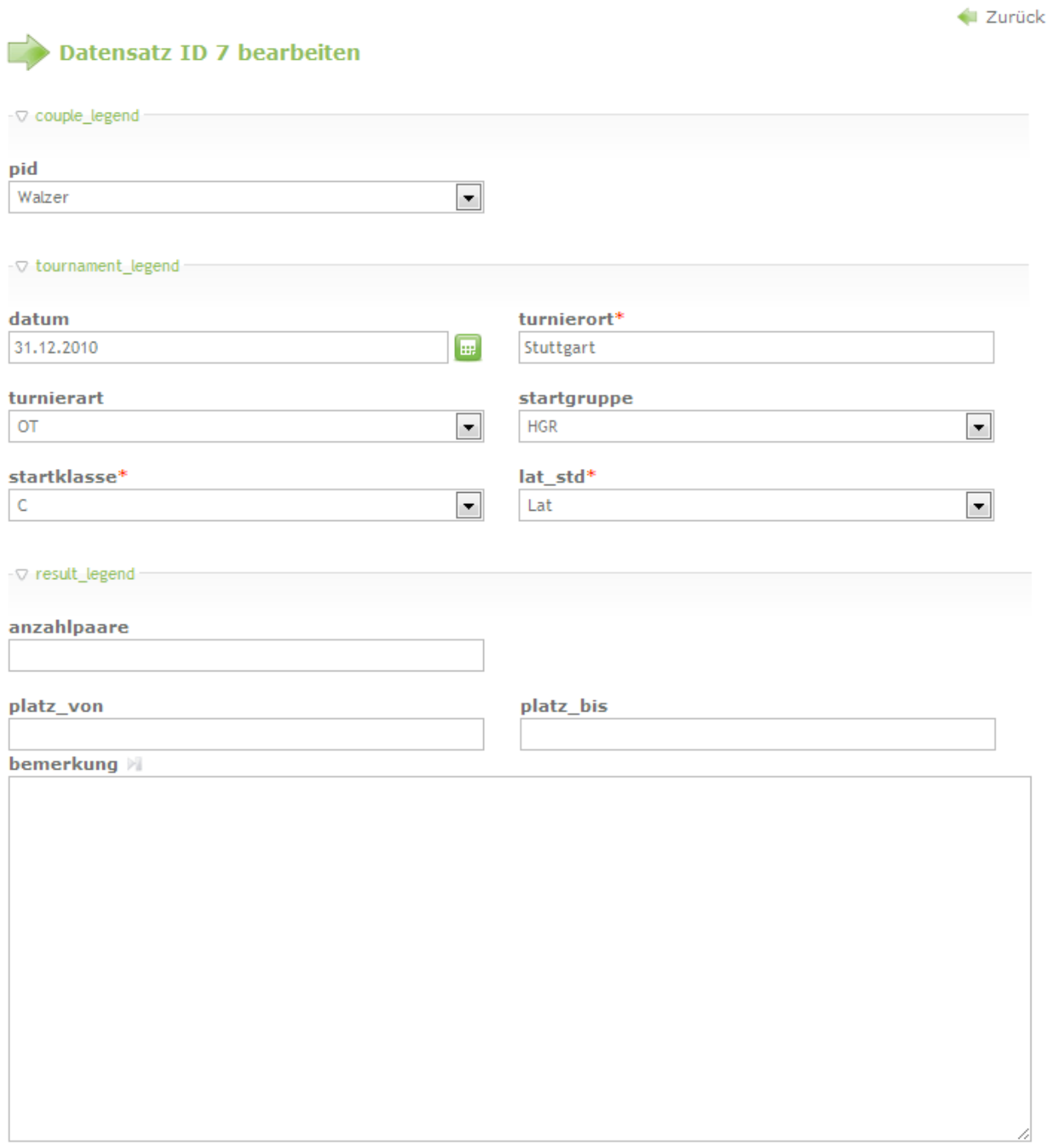
Wenn ich einen neuen Eintrag hinzufüge, sieht das so aus:



Durch das Fehlen der Language-Einträge natürlich noch sehr unschön, aber alle Felder sind da. Die Foreign-Key-Beziehung in die Turnierpaar-Tabelle klappt (rudimentär) auch.

Mich wundert der "20.12.2000" als Default im Datumsfeld, aber ich habe auch keinen Default vorgegeben. Es wäre wohl praktisch, durch einen Load-Callback das aktuelle Datum dort als Default vorzugeben.

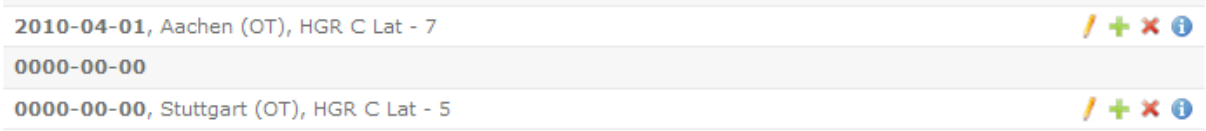
Ich gebe also mal Daten ein:



Und drücke auf Speichern- was sehe ich beim Datum:

30.11.1999? Das habe ich im Datepicker aber nicht ausgewählt....auch eine manuelle Eingabe sorgt für das selbe 1999er-Ergebnis.

Und in der Übersichtstabelle dann das:



0000-00-00...hm.

Das ist alles nicht so ermutigend. Ich habe die Konfiguration für das Datumsfeld bei tl_article.php abgeschaut, da funktioniert ja auch alles.

Bevor noch mehr Leute da Zeit reinstecken: Das Problem bei meinem Datumsfeld ist einfach, dass ich instinktiv in MYSQL ein "date"-Feld angelegt habe. Es muss aber ein varchar(10) sein. Logisch ist das nicht. Also bitte keine weiteren Hinweise zur Fehlerbehebung, lindesbs war der Schnellste, danke :-).

Hätte ich mehr Zeit gehabt, hätte ich mir selbst mal die DB-Struktur von tl_article angeschaut, um den Unterschied zu finden, Aber Zeit ist gerade knapp, und ich wollte endlich die nächste "Folge" rausschicken. Im nächsten Schritt wird das Datumsfeld also voraussichtlich hervorragend funktionieren, indem wir es in ein "varchar(10)" verwandeln :-).

---

deerwood

Abgeguckt beim DCA-Record für Artikel: Das Datumsfeld.

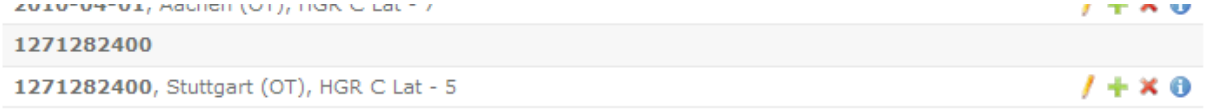
Ich habe gerade mal schnell gecheckt: DB Datentyp DATE() wird in TL offenbar nirgendwo verwendet, alle Felder, die im DCA den DatePicker verwenden, haben in der DB den Datentyp VARCHAR(10) (bis auf tl_news.date, das ist merkwürdigerweise INT(10)). Vielleicht ist das das Problem?

---

dl1ely

Ja, ich vermute sehr stark, dass das das Problem ist. Habe es einfach intuitiv (falsch) gemacht. Leider fehlte mir die Zeit, vorher "im Core" abzugucken. Eins habe ich sowieso schon gelernt: Das Einzige, auf was man sich in Sachen "TL-Entwickler-Doku" verlassen kann, ist der Core-Source ;-).

EDIT: habe es schnell geändert und das Feld "datum" auf varchar(10) umgestellt. Jetzt klappt der Datepicker. Aber jetzt wird das Feld mit einem Unix-Timestamp gefüllt. Gebe ich das Feld also in meinen Tabellenzeilen aus oder als Sortierheader, steht dort nur Zahlenwust statt eines Datums. Ich wusste schon, warum ich lieber "date" haben wollte ;-). Ich kriege graue Haare. Da muss ich mir also definitiv noch was einfallen lassen...



BugBuster

der Timestamp hat aber den Vorteil, das du diesen einfach wandeln kannst, genauer in die Form die der Nutzer im Backend (Einstellungen / Startseite) definiert hat.
Mein Modul hat z.B. solch eine Zeile:

PHP-Code:

```
$VisitorsStartDate = date($GLOBALS['TL_CONFIG']['dateFormat'],$objVisitors->visitors_startdate);
```

$objVisitors->visitors_startdate ist hier der Timestamp aus der DB.

---

dl1ely

Sicherlich richtig, aber wenn ich per DCA-Record im Backend damit arbeiten will, z.b. als Sortier-, bzw. Gruppierfeld, dann habe ich nicht die Möglichkeit, dort ein "date()" zu injecten. Aber ich schaue mal weiter, was mit Callbacks geht und wie ich meiner Anforderung näherkomme. :-)

TLight

Zuerst einmal herzichen Dank für dieses hervorragende Tutorial!!! So etwas habe ich mir schon sehr lange gewünscht.

Ich versuche nun, nach Deiner Vorlage eine Verwaltung für unseren Vereinsbekleidungs-Verleih zu programmieren. Ich habe den Extension-Creator, die Datenbank-Einrichtung und die DCA-Programmierung fürs Erste recht gut hinbekommen und alle kleinen Fehler ziemlich schnell lokalisieren und beheben können. Ich arbeite auf einer lokalen Installation und habe die Fehlermeldung eingeschaltet.

Wenn ich mich richtig erinnere, so kommt seitdem ich das language file "modules.php" angefasst habe, folgende Fehlermeldung:

```
Warning: Cannot modify header information - headers already sent by (output started at
mein_pfad\system\modules\to_bekleidungsverleih\languages\de \modules.php:1) in
mein_pfad\system\libraries\Template.php on line 174

#0 [internal function]: __error(2, 'Cannot modify h...', 'mein_pfad...', 174, Array)
#1 mein_pfad\system\libraries\Template.php(174): header('Content-Type: t...')
#2 mein_pfad\system\modules\backend\BackendTemplate.php(135): Template->output()
#3 mein_pfad\typolight\main.php(286): BackendTemplate->output()
#4 mein_pfad\typolight\main.php(102): Main->output()
#5 mein_pfad\typolight\main.php(295): Main->run()
#6 {main}
```

Leider bin ich nun mit meinem Latein am Ende... Weiß jemand Rat?

Toflar

Bei Dir ist irgend ein Zeichen vor der ersten php Anweisung Also wahrscheinlich " <?php" statt "<?php" oder sowas

---

TLight

Ich habe in modules.php nachgesehen - leider ist dort kein Freizeichen. Auch in keiner anderen der von mir angefassten Dateien (soweit ich mich erinnern kann).

Zu allem Überfluss bemerke ich gerade, dass beim Laden der Startseite des Backends folgender Fehler 2x vor dem o. g. Fehler gelistet wird:

```
Warning: Illegal offset type in mein_pfad\typolight\main.php on line 183
```

#0 mein_pfad\typolight\main.php(183): __error(2, 'Illegal offset ...', 'mein_pfad...', 183, Array)
#1 mein_pfad\typolight\main.php(93): Main->welcomeScreen()
#2 mein_pfad\typolight\main.php(295): Main->run()
#3 {main}

---

xchs

> Zitat von **TLight**
> *...leider ist dort kein Freizeichen. Auch in keiner anderen der von mir angefassten Dateien (soweit ich mich erinnern kann).*

Das kann aber unter Umständen auch ein verstecktes Zeichen sein, je nachdem welchen Editor Du verwendest (UTF-8 vs. UTF-8 BOM usw.)

TLight

Nein, leider nichts zu finden. Auch im stinknormalen Editor ist da kein Zeichen. Ich verwende UTF-8-Codierung, aber auch da konnte ich nichts finden...

FloB

Schau alle Dateien durch. Mit welchem Editor bearbeitest du die Dateien? Schau mal da nach eine Option bezüglich des BOM.

Einen BOM "siehst" du eigentlich auch nur in einem Hex-Editor.

---

dl1ely

> Zitat von **TLight**
> *Nein, leider nichts zu finden. Auch im stinknormalen Editor ist da kein Zeichen. Ich verwende UTF-8-Codierung, aber auch da konnte ich nichts finden...*

Hmm, es muss aber definitiv was mit modules.php, Zeile 1 zu tun haben.

Wenn du magst, kannst du mir dein File schicken (zip, damit beim Versenden nix passiert), an stefan@fam-pfeiffer.de

---

Jürgen

Ich hatte heute auch das Problem mit diesem Fehler. Wollte zwei Ankreuzfelder ergänzen. Hab dann die Codierung von UTF-8 auf UTF-8 ohne BOM umgestellt und siehe, da seit dem klappt es. Benutze das Notepad++ unter Windows.

Wenn ich noch eine Frage stellen darf. Wenn ich calender_events um zwei Checkboxen erweitern möchte und das der Titel_legende zuweise wird ein Checkboxfeld verdoppelt und die Legende verschwindet. Erweitere ich aber z.B. die Publish_legende werden wie gewünscht nur zwei Felder angezeigt und auch die Legende bleibt erhalten.

---

dl1ely

> Zitat von **TLight**

Wow! Das ging ja schnell...

Ich habe in modules.php nachgesehen - leider ist dort kein Freizeichen. Auch in keiner anderen der von mir angefassten Dateien (soweit ich mich erinnern kann).

Zu allem Überfluss bemerke ich gerade, dass beim Laden der Startseite des Backends folgender Fehler 2x vor dem o. g. Fehler gelistet wird:

Wenn jemand noch was weiß...
...wär's toll!

Schick mir mal bitte deine /lang/de/modules.php Sprachdatei (Oder zeige sie uns hier, die ist ja nicht so lang). Ich weiß ja nicht, ob du es genauso wie ich gemacht hast, aber ich bin über diesen Fehler auch gestolpert, aber das gehört zum nächsten Teil des Tagesbuchs. Und bevor wir hier in die falsche Richtung rennen, würde ich gerne sehen, wie diese Datei aussieht. Falls es "mein" Fehler ist, kann ich dir dann sagen woran es liegt ;-).

Oder um es doch hier vorweg zu nehmen: Wenn du wie ich in der aktuellen Version dein(e) Backendmodul(e) unter eine eigene Obergruppe gesetzt hast (bei mir gw_paarverwaltung), dann darf der entsprechende Übersetzungseintrag $GLOBALS['TL_LANG']['MOD']['gw_paarverwaltung'] NICHT ein Array sein (erster Eintrag Text fürs Menü, zweiter Eintrag der Erklärungstext), sondern einfach nur ein String, der den Menüeintrag darstellt. Das ist anders als bei den Einträgen für die Module selbst. Bei meinem Modul ist das jetzt so:

PHP-Code:

```
/**
 * Back end modules
 */
$GLOBALS['TL_LANG']['MOD']['gw_turnierpaare'] = array('Turnierpaare', 'Verwaltung
der Turnierpaare.');

$GLOBALS['TL_LANG']['MOD']['gw_meldungen'] = array('Meldungen', 'Verwaltung der Tu
rniermeldungen (Meldeliste).');

$GLOBALS['TL_LANG']['MOD']['gw_paarverwaltung'] = 'Paarverwaltung';
```

Die letzte Zeil ist der Menüeintrag für die "Gruppe" im Backendmenü, die ersten beiden Einträge die Werte für meine beiden Backendmodule. Hatte auch für die "Gruppe" ein Array aus Menütext und Erklärungstext, und das brachte mir den von dir beschriebenen Fehler. Ist das dein Problem?

---

dl1ely

Zitat von **Jürgen**
Wenn ich noch eine Frage stellen darf. Wenn ich calender_events um zwei Checkboxen erweitern möchte und das der Titel_legende zuweise wird ein Checkboxfeld verdoppelt und die Legende verschwindet. Erweitere ich aber z.B. die Publish_legende werden wie gewünscht nur zwei Felder angezeigt und auch die Legende bleibt erhalten.

Mach ich da was falsch oder hat da TL ein Problem.

Grüße
Jürgen

Mal ein Schuss ins Blaue: Macht dein str_replace für den Paletteneintrag wirklich das richtige? Check das lieber mal durch ein Debugging-Echo oder ähnlich. Beim richtigen str_replace() zur Modifikation der bestehenden Palette kann man sich böse vertun.

OK, hier Kommando zurück. Der von mir beschriebene Fehler kann erst zuschlagen, wenn man eine Modifikation der /config/config.php macht, die ich bisher hier noch gar nicht beschrieben habe, sondern die erst in der nächsten Tagebuchfolge erscheint. Wenn du also bisher alles so gemacht hast wie ich, kann es nicht dein Problem sein. Trotzdem hat das Problem definitiv mit falschen Einträgen in den Language-Dateien (und zwar modules.php) zu suchen. Entweder hast du irgendwo ein Array, wo ein String erwartet wird, oder umgekehrt.

Falls du es selbst nicht findest, bitte deine config/config.php und language/de/modules.php zu mir, dann schaue ich mir das an.

Sorry für die Verwirrung, das ist das Problem, wenn der eigentliche Code schon einen Schritt weiter ist als die aktuelle Tagebuchberichterstattung...

TLight

@dl1ely: wahrscheinlich habe ich genau den von Dir beschriebenen Fehler gemacht!

/config/config.php:

PHP-Code:

```
$GLOBALS['BE_MOD']['to_bekleidungsverleih']['to_magazin'] = array
(
    'tables' => array('tl_to_bkvs_magazin'),
    'icon'   => 'system/modules/to_bekleidungsverleih/icons/trikot.png'
);
$GLOBALS['BE_MOD']['to_bekleidungsverleih']['to_verleih'] = array
(
    'tables' => array('tl_to_bkvs_verleih'),
    'icon'   => 'system/modules/to_bekleidungsverleih/icons/trikot.png'
);
```

/languages/de/modules.php:

PHP-Code:

```
/**
 * Back end modules
 */
$GLOBALS['TL_LANG']['MOD']['to_bekleidungsverleih'] = array('Bekleidungsverleih',
'');
$GLOBALS['TL_LANG']['MOD']['to_magazin']            = array('Magazin', '');
$GLOBALS['TL_LANG']['MOD']['to_verleih']            = array('Verleih', '');
```

Leider habe ich Deine Lösung nicht ganz verstanden. Aber ich warte einfach auf Deinen neuen Tagebucheintrag. Will Dir nicht noch mehr Arbeit machen als Du eh schon hast.

Wenn ich die /languages/de/modules wie folgt ändere:

PHP-Code:

```
/**
 * Back end modules
 */
$GLOBALS['TL_LANG']['MOD']['to_bekleidungsverleih'] ='Bekleidungsverleih';
$GLOBALS['TL_LANG']['MOD']['to_magazin']            = array('Magazin', '');
$GLOBALS['TL_LANG']['MOD']['to_verleih']            = array('Verleih', '');
```

verschwinden zwar die zwei Fehler der Startseite und das Modul wird in der Mittelspalte angezeigt:

Warning: Illegal offset type in mein_pfad\typolight\main.php on line 183

#0 mein_pfad\typolight\main.php(183): __error(2, 'Illegal offset ...', 'mein_pfad...', 183, Array)
#1 mein_pfad\typolight\main.php(93): Main->welcomeScreen()
#2 mein_pfad\typolight\main.php(295): Main->run()
#3 {main}

Der Ursprungsfehler bleibt aber:

Warning: Cannot modify header information - headers already sent by (output started at mein_pfad\system\modules\to_bekleidungsverleih\lan guages\de \modules.php:1) in mein_pfad\system\libraries\Template.php on line 174

#0 [internal function]: __error(2, 'Cannot modify h...', 'mein_pfad...', 174, Array)
#1 mein_pfad\system\libraries\Template.php(174): header('Content-Type: t...')
#2 mein_pfad\system\modules\backend\BackendTemplate.p hp(135): Template->output()
#3 mein_pfad\typolight\main.php(286): BackendTemplate->output()
#4 mein_pfad\typolight\main.php(102): Main->output()
#5 mein_pfad\typolight\main.php(295): Main->run()
#6 {main}

Angehängte Dateien

- bekleidung.zip (5,9 KB, 3x aufgerufen)

--> gelöst!!!

Bevor sich nun zu viele mit meinem Problem beschäftigen, hier die Lösung:

Habe meine Dateien in Notepad++ in UTF-8 ohne BOM konvertiert - und das Problem hat sich erledigt. Was zum T. ist BOM? Nun ja, vielen Dank für all Eure Hilfe. Toll, wie schnell das geklappt hat!

Jürgen

Zitat von **dl1ely**
Mal ein Schuss ins Blaue: Macht dein str_replace für den Paletteneintrag wirklich das richtige? Check das lieber mal durch ein Debugging-Echo oder ähnlich. Beim richtigen str_replace() zur Modifikation der bestehenden Palette kann man sich böse vertun.

Stefan
Zitat von **TLight**
Bevor sich nun zu viele mit meinem Problem beschäftigen, hier die Lösung:

Habe meine Dateien in Notepad++ in UTF-8 ohne BOM konvertiert - und das Problem hat sich erledigt. Was zum T. ist BOM? Nun ja, vielen Dank für all Eure Hilfe. Toll, wie schnell das geklappt hat!

Gruß, Torsten
@Torsten: Gerne. Ich war auch schon kurz vor dem verzweifeln mit dem BOM.

@Stefan: Ich hab das Beispiel "CalenderFreeEntry" aus dem Wiki benutzt. Auch dort tritt das Problem auf. Genauer gesagt wird beim Hinzufügen von einer Checkbox die Legende nicht mehr angezeigt (nur bei Titel_legende). Wenn ich nun eine zweite Checkbox hinzufüge erscheint eine von diesen doppelt. Sobald ich es auf eine andere Legende lege funktioniert alles wie erwartet.

---

dl1ely

> Zitat von **TLight**
> *Vielen Dank für die zahlreichen Hilfsangebote!*
>
> *@dl1ely: wahrscheinlich habe ich genau den von Dir beschriebenen Fehler gemacht!*
>
> */config/config.php:*
>
> *PHP-Code:*
> ```
> $GLOBALS['BE_MOD']['to_bekleidungsverleih']['to_magazin'] = array
> (
>     'tables' => array('tl_to_bkvs_magazin'),
>     'icon'   => 'system/modules/to_bekleidungsverleih/icons/trikot.png'
> );
> $GLOBALS['BE_MOD']['to_bekleidungsverleih']['to_verleih'] = array
> (
>     'tables' => array('tl_to_bkvs_verleih'),
>     'icon'   => 'system/modules/to_bekleidungsverleih/icons/trikot.png'
> );
> ```
> */languages/de/modules.php:*
>
> *PHP-Code:*
> ```
> /**
>  * Back end modules
>  */
> $GLOBALS['TL_LANG']['MOD']['to_bekleidungsverleih'] = array('Bekleidungsverleih',
> '');
> $GLOBALS['TL_LANG']['MOD']['to_magazin']            = array('Magazin', '');
> $GLOBALS['TL_LANG']['MOD']['to_verleih']            = array('Verleih', '');
> ```
> *Leider habe ich Deine Lösung nicht ganz verstanden. Aber ich warte einfach auf Deinen neuen Tagebucheintrag. Will Dir nicht noch mehr Arbeit machen als Du eh schon hast.*
>
> *Vielen Dank!*

Naja, "Lösung nicht verstanden". Eine Lösung ist es eigentlich nicht, eher eine Erklärung. to_bekleidungsverleih ist die "Obergruppe" im Backend-Modul-Menü für deine beiden Backend-Module to_magazin und to_verleih. Und bei TL ist es nunmal so, dass dann in der Language-Datei bei der Obergruppe nur ein einzelner Übersetzungsstring stehen darf, während bei den Backendmodulen selbst ein Array bestehend aus Übersetzung des Menüeintrags und des Erklärungstextes stehen muss.

Witzigerweise bist du in genau dasselbe Problem wie ich gelaufen, aber ich hätte erst in der nächsten Tagebuchausgabe davon berichtet. Jetzt ist ja alles geklärt.

---

dl1ely

> Zitat von **Jürgen**
> *@Stefan: Ich hab das Beispiel "CalenderFreeEntry" aus dem Wiki benutzt. Auch dort tritt das Problem auf. Genauer gesagt wird beim Hinzufügen von einer Checkbox die Legende nicht mehr angezeigt (nur bei Titel_legende). Wenn ich nun eine zweite Checkbox hinzufüge erscheint eine von diesen doppelt. Sobald ich es auf eine andere Legende lege funktioniert alles wie erwartet.*
> *Das versteh ich nicht.* *Wenn es bei allen Legenden funktioniert und bei der Titel_legende nicht. Vielleicht liegt es daran das Titel_legende die erste beim Calendar_event ist?*
>
> *Grüße*
> *Jürgen*

Wie gesagt, der einzige Tip, den ich erstmal habe, wäre per echo o.ä. sich den String der Palettendefinition im DCA nach der Modifikation anzugucken. Ich vermute einfach, da läuft beim str_replace oder wie auch immer der bestehende String modifziert wird was schief. Ohne den String zu sehen ist das aber schwer zu sagen...

FloB

> Zitat von **TLight**
> *Was zum T. ist BOM?*
> http://de.wikipedia.org/wiki/Byte_Order_Mark

---

dl1ely

> Zitat von **FloB**
> *http://de.wikipedia.org/wiki/Byte_Order_Mark*

Übrigens, aus meiner persönlichen Sicht ist die mangelnde Kenntnis dieser Abkürzung keine Schande - ich kannte sie bis vor Kurzem auch noch nicht und habe sie erst in der Wikipedia kennengelernt. Und das obwohl ich mir eigentlich einbilde, in der Informatik-Welt schon umfangreiche Kenntnisse zu besitzen. Bei UTF habe ich aber wohl noch Lücken ;-)

---

klabog

ausgezeichnetes Tutorial! Das motiviert wirklich, sich ein bisschen umfassender mit Tl zu beschäftigen. So was ähnliches wollte ich mal mit der Anpassung von Templates machen, muss aber zu meiner Schande gestehen, dass ich nicht die Ausdauer (und Kenntnisse) hatte das durchzuziehen.
Aber grundsätzlich finde ich diese Methode - ein "Anfänger" -> oder besser "Nicht-Experte" - schildert wie er ein Projekt durchführt, mit allen Unsicherheiten und Fehlern die dabei entstehen und die Community begleitet mit Kommentaren und Hinweisen, die beste Form sich Wissen anzueignen.
Am Schluss sollte dabei allerdings ein HowTo herauskommen, das die Entwicklung Schritt für Schritt abbildet und auf die möglichen Fehler an entsprechender Stelle hinweist.
Das muss nicht notwendigerweise von Dir kommen, Du machst Dir schon genug Arbeit, sondern vielleicht von einem Team das Deine Vorlage redaktionell bearbeitet und zusammen mit Dir vielleicht eine Extension bastelt die vom speziellen Fall auf eine universelle Variante zielt.
Ich bin gerne dabei.

---

dl1ely

Naja gut, schauen wir mal...Vielleicht kann man das in der Tat "am Ende" in ein "normales", leichter lesbares Tutorial umwandeln. Leider geht es momentan nicht voran, weil ich meine Prioritäten außerhalb von TL leider höher setzen muss. Der Tag hat nur 24 Stunden, und ich brauche momentan auch ohne das Tagebuch gefühlt mindestens 48h.

Schritt 12: Callback-Magie

Ja, nach langer Pause geht es wirklich weiter. Diesmal wird die Backendpflege der "Meldungen"-Tabelle komplettiert. Gegenüber dem letzten Schritt haben sich nach fruchtbarer Diskussion einige Änderungen ergeben.

Zunächst die Datenbankdefinition der Tabelle tl_gw_meldungen in system/modules/tl_gw_turnierpaare/config/database.sql:

Code:

```
--
-- Table `tl_gw_meldungen`
--

CREATE TABLE `tl_gw_meldungen` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `pid` int(10) unsigned NOT NULL default '0',
  `sorting` int(10) unsigned NOT NULL default '0',
  `tstamp` int(10) unsigned NOT NULL default '0',
  `datum` varchar(10) NOT NULL default '',
  `startgruppe` varchar(32) NOT NULL default '',
  `startklasse` varchar(12) NOT NULL default '',
  `lat_std` char(32) NOT NULL default '',
  `turnierort` varchar(128) NOT NULL default '',
  `turnierart` varchar(64) NULL default NULL,
  `anzahlpaare` int(4) NULL default NULL,
  `platz_von` int(4) NULL default NULL,
  `platz_bis` int(4) NULL default NULL,
  `bemerkung` text NULL,
  PRIMARY KEY  (`id`),
  KEY `pid` (`pid`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Das Feld datum war vorher vom Typ "date", aber das funktioniert so nicht, wenn man es als Datumsfeld im Backend verwenden will. Darum jetzt ein varchar(10). Der Rest blieb unverändert.

Auch das Einfügen der Backendmodule in die Auswahlleiste auf der linken Seite des Backends ändere ich in /system/modules/tl_gw_turnierpaare/config/config.php ab:

PHP-Code:

```
// Back end module
array_insert($GLOBALS['BE_MOD'], 0, array(
    'gw_paarverwaltung' => array(
      'gw_turnierpaare' => array
      (
        'tables' => array('tl_gw_turnierpaare'),
        'icon'   => 'system/modules/gw_turnierpaare/icons/turnierpaare.png'
      ),

      'gw_meldungen' => array
      (
        'tables' => array('tl_gw_meldungen'),
        'icon'   => 'system/modules/gw_turnierpaare/icons/meldeliste.png'
      )
    )
  )
);
```

Durch das array_insert kann ich mit zweiten Parameter steuern, an welcher Stelle das als letzter Parameter angegebene Array eingefügt werden soll. Bei einer "0" wie hier wandern meine Arrayeinträge ganz nach oben, bei einer "1" an zweite Stelle, usw. Da in der Praxis auf der Webseite häufiger neue Turnierpaarmeldungen eingegeben werden als neue Artikel, mächte ich die Turnierpaarverwaltung gerne ganz oben haben.

Und dadurch, dass ich in dem Array was ich einfüge eine zusätzliche Ebene "gw_paarverwaltung" habe, gruppiere ich die beiden Einträge gw_turnierpaare und gw_meldungen in eine Untergruppe innerhalb des Menüs. Das schafft mehr Übersicht.

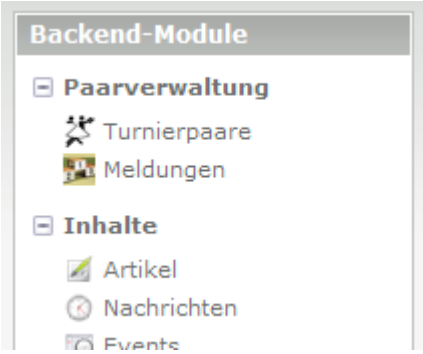
Um die Language-Datei für diese Einträge muss ich mich jetzt auch einmal kümmern (/system/modules/tl_gw_turnierpaare/languages/modules.php):

PHP-Code:

```
/**
 * Back end modules
 */
$GLOBALS['TL_LANG']['MOD']['gw_turnierpaare'] = array('Turnierpaare', 'Verwaltung
der Turnierpaare.');

$GLOBALS['TL_LANG']['MOD']['gw_meldungen'] = array('Meldungen', 'Verwaltung der Tu
rniermeldungen (Meldeliste).');

$GLOBALS['TL_LANG']['MOD']['gw_paarverwaltung'] = 'Paarverwaltung';
```

ACHTUNG: Während die Language-Einträge für die Module selbst ein Array bestehend aus dem Label des Menüeintrags und dem Beschreibungstext des Moduls sein müssen, ist der Label für die Überschrift der Gruppe im Menü (gw_paarverwaltung) nur ein einfacher String!

Wie sieht das nun aus?



Wo wir schonmal bei den Language-Dateien sind, machen wir auch noch die Label für die Felder des Backendmoduls (/system/modules/tl_gw_turnierpaare/languages/de/tl_gw_meldungen.php):

PHP-Code:

```
<?php if (!defined('TL_ROOT')) die('You can not access this file directly!');
/**
 * Fields
 */
$GLOBALS['TL_LANG']['tl_gw_meldungen']['pid'] = array('Paar', 'Turnierpaar auswähl
en');

$GLOBALS['TL_LANG']['tl_gw_meldungen']['datum'] = array('Turnierdatum', 'Turnierda
tum eingeben');
$GLOBALS['TL_LANG']['tl_gw_meldungen']['turnierort'] = array('Turnierort', 'Turnie
rort eingeben');
```

```
$GLOBALS['TL_LANG']['tl_gw_meldungen']['turnierart'] = array('Turnierform', 'Turnierform auswählen');
$GLOBALS['TL_LANG']['tl_gw_meldungen']['startgruppe'] = array('Startgruppe', 'Startgruppe auswählen');
$GLOBALS['TL_LANG']['tl_gw_meldungen']['startklasse'] = array('Startklasse', 'Startklasse auswählen');
$GLOBALS['TL_LANG']['tl_gw_meldungen']['lat_std'] = array('Latein/Standard', 'Tanzart auswählen');

$GLOBALS['TL_LANG']['tl_gw_meldungen']['anzahlpaare'] = array('Anzahl gestarteter Paare', 'Die Paaranzahl des Turniers eingeben');
$GLOBALS['TL_LANG']['tl_gw_meldungen']['platz_von'] = array('Platz', 'Erzielter Platz');
$GLOBALS['TL_LANG']['tl_gw_meldungen']['platz_bis'] = array('Platz bis', 'Geteilter Platz: Platz bis... - sonst leer lassen');
$GLOBALS['TL_LANG']['tl_gw_meldungen']['bemerkung'] = array('Bemerkung', 'Bemerkung eingeben');

/**
 * Reference
 */
$GLOBALS['TL_LANG']['tl_gw_meldungen']['couple_legend'] = 'Paar';
$GLOBALS['TL_LANG']['tl_gw_meldungen']['tournament_legend'] = 'Turnier';
$GLOBALS['TL_LANG']['tl_gw_meldungen']['result_legend'] = 'Ergebnis';

/**
 * Buttons
 */
$GLOBALS['TL_LANG']['tl_gw_meldungen']['new']    = array('Neue Meldung', 'Eine neue Turniermeldung anlegen');
$GLOBALS['TL_LANG']['tl_gw_meldungen']['edit']   = array('Editieren', 'Die Turniermeldung editieren');
$GLOBALS['TL_LANG']['tl_gw_meldungen']['copy']   = array('Meldung kopieren', 'Die Turniermeldung in die Zwischenablage kopieren');
$GLOBALS['TL_LANG']['tl_gw_meldungen']['delete'] = array('Meldung löschen', 'Die Turniermeldung aus der Liste entfernen');
$GLOBALS['TL_LANG']['tl_gw_meldungen']['show']   = array('Details', 'Die Detailansicht der Turniermeldung anzeigen');
?>
```

Da muss ich wohl nicht mehr viel zu schreiben: Der erste Block sind Label und Beschreibungen für die Eingabefelder, der zweite Block sind die Label für die Palettenüberschriften, und die dritte Gruppe schließlich die Label und Beschreibungen für die "Aktionsbuttons".

Kommen wir zum spannenden Teil, mir dem ich im Schritt 11 ja noch einen Kampf zu fechten hatte: der DCA-Record (/system/modules/tl_gw_turnierpaare/dca/tl_gw_meldungen.php):

PHP-Code:

```
<?php if (!defined('TL_ROOT')) die('You can not access this file directly!');
/**
 * Table tl_gw_meldungen
 */
$GLOBALS['TL_DCA']['tl_gw_meldungen'] = array
(

    // Config
    'config' => array
    (
        'dataContainer'               => 'Table',
        'enableVersioning'            => true,
    ),
```

Hier so weit nichts Spannendes: Weiterhin wird eine Tabelle bearbeitet, und ich aktiviere Versionierung.

PHP-Code:

```
    // List
    'list' => array
    (
        'sorting' => array
        (
            'mode'                    => 2,
            'fields'                  => array('datum DESC', 'turnierort', 'pid'),

            'panelLayout'             => 'filter;sort,search,limit'
        ),
```

mode = 2 bedeutet, dass das Sortierfeld im Header der Tabelle wählbar ist, defaultmäßig sortiere ich nach Turnierdatum (Jüngste ganz oben), dann dem Ort, und schließlich der Paar-ID.

Mit panelLayout aktiviere ich die Paletten oben am Kopf der Tabelle, um das Sortierfeld wählbar zu machen, das Suchfeld zu aktivieren und die Anzahl der Treffer limitieren zu können.

PHP-Code:

```
        'label' => array
        (
            'fields'                  => array('datum','turnierort', 'turnierart',
'startgruppe','startklasse','lat_std'),
            'format'                  => '%s - #name# - <span style="font-weight:
bold;">%s</span> - <span style="color: #section_colour#;">%s %s %s %s</span>',
            'label_callback'          => array('tl_gw_meldungen', 'lookup_pid')
        ),
```

So, hier beginnt etwas die Magie...ich bastele mir das Format der Ausgabezeilen für die Backendansicht.

Leider habe ich hier nur die Felder der aktuellen Tabelle (tl_gw_meldungen) zur Verfügung, ich möchte aber gerne den Namen des Paares anzeigen, was zur "pid" gehört, die im Datensatz steht (pid ist der Foreign Key in die tl_gw_turnierpaare-Tabelle. Im klassischen SQL würde ich hier also einen JOIN machen).

So direkt geht das leider nicht, darum definiere ich einen "label_callback", der weiter unten in dieser Datei als Funktion "lookup_pid()" in der Klasse "tl_gw_meldungen" definiert wird. Die Felder, die ich jetzt schon habe, definiere ich auch direkt im String. Für den Namen des Paares setze ich zunächst einen Platzhalter "#name#" in den String.

Außerdem gibt es noch einen Platzhalter "#section_colour#". Ich möchte Turnierstarts in der Tanzform Standardtänze in orange darstellen, und solche in der Tanzform Lateinamerikanische Tänze in rot. Das kann ich an dieser Stelle im DCA-Record nicht entscheiden, weil es vom Inhalt des Felds "lat_std" abhängt. Auch das wird im label_callback geregelt.

Die Platzhalter werden im label_callback durch ein str_replace mit den gewünschten Werten ersetzt. Schön ist das nicht, aber funktioniert.

PHP-Code:

```
        'global_operations' => array
        (
            'all' => array
            (
                'label'               => &$GLOBALS['TL_LANG']['MSC']['all'],
                'href'                => 'act=select',
                'class'               => 'header_edit_all',
```

```
                'attributes'            => 'onclick="Backend.getScrollOffset();"'
            )
        ),
        'operations' => array
        (
            'edit' => array
            (
                'label'                 => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['edit'],
                'href'                  => 'act=edit',
                'icon'                  => 'edit.gif'
            ),
            'copy' => array
            (
                'label'                 => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['copy'],
                'href'                  => 'act=copy',
                'icon'                  => 'copy.gif'
            ),
            'delete' => array
            (
                'label'                 => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['delete'],
                'href'                  => 'act=delete',
                'icon'                  => 'delete.gif',
                'attributes'            => 'onclick="if (!confirm(\'' . $GLOBAL
S['TL_LANG']['MSC']['deleteConfirm'] . '\')) return false; Backend.getScrollOffset
();"'
            ),
            'show' => array
            (
                'label'                 => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['show'],
                'href'                  => 'act=show',
                'icon'                  => 'show.gif'
            )
        )
    ),
```

Zu den globalen Operationen gibt es eigentlich nichts zu sagen. Die bleiben so, wie der Extensioncreator sie angelegt hat...

PHP-Code:

```
    // Palettes
    'palettes' => array
    (
        '__selector__'                  => array(''),
        'default'                       => '{couple_legend},pid;
{tournament_legend},datum,turnierort,turnierart,startgruppe,startklasse,lat_std;'
                                        .'{result_legend},anzahlpaare,platz_von,platz_
bis,bemerkung;'
    ),

    // Subpalettes
    'subpalettes' => array
    (
        ''                              => ''
    ),
```

Die Palettendefinition hat sich nicht geändert, und Subpaletten gibt es weiterhin nicht.

PHP-Code:

```
    // Fields
    'fields' => array
    (
        'pid' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['pid'],
            'inputType'                 => 'select',
            'options_callback'          => array('tl_gw_meldungen', 'getActiveCouple
s'),
            'search'                    => true,
            'sorting'                   => true,
            'eval'                      => array('mandatory'=>true)
        ),
```

Das Feld "pid" war bisher ein foreignKey-Feld, aber damit konnte ich z.B. nur mit dem Partner-Nachnamen verknüpfen, nicht mit dem Paarnamen. Außerdem wurden hier immer ALLE Paare angezeigt, hier sollen aber nur AKTIVE Paare auswählbar sein. Um das besser machen zu können, definiere ich die Möglichkeiten in der Dropdown-Box selbst durch den options_callback "getActiveCouples()" in der Klasse tl_gw_meldungen, die gleich noch folgt.

PHP-Code:

```
        'datum' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['datum'],
            'inputType'                 => 'text',
            'search'                    => true,
            'sorting'                   => true,
            'flag'                      => 6,
            'eval'                      => array('mandatory'=>true, 'datepicker'=>$t
his->getDatePickerString(), 'tl_class'=>'w50 wizard', 'minlength' => 1, 'maxlength
'=>10, 'rgxp' => 'date')
        ),
```

"flag" = 6 ist hier der große Trick, damit das Timestamp-Format von "datum" richtig als Datum im Format TT.MM.JJJJ angezeigt wird. Abgeguckt habe ich das übrigens im Backend in /system/modules/backend/dca/tl_log.php .

PHP-Code:

```
        'turnierort' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['turnierort'],
            'inputType'                 => 'text',
            'search'                    => true,
            'sorting'                   => true,
            'flag'                      => 11,
            'eval'                      => array('mandatory'=>true, 'minlength' => 1
, 'maxlength'=>128, 'tl_class' => 'w50')
        ),
        'turnierart' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['turnierart'],
            'inputType'                 => 'select',
            'search'                    => true,
```

```
            'sorting'                   => true,
            'flag'                      => 11,
            'options'                   => gwTurnierpaarliste::$TurnierArten,
            'eval'                      => array('mandatory'=>false, 'tl_class' => '
w50')
        ),
        'startgruppe' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['startgruppe'],
            'inputType'                 => 'select',
            'search'                    => true,
            'sorting'                   => true,
            'flag'                      => 11,
            'options'                   => gwTurnierpaarliste::$StartGruppen,
            'eval'                      => array('mandatory'=>false, 'includeBlankOp
tion' => true, 'tl_class' => 'w50')
        ),
        'startklasse' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['startklasse'],
            'inputType'                 => 'select',
            'search'                    => true,
            'sorting'                   => true,
            'flag'                      => 11,
            'options'                   => gwTurnierpaarliste::$StartKlassen,
            'eval'                      => array('mandatory'=>true, 'tl_class' => 'w
50')
        ),
        'lat_std' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['lat_std'],
            'inputType'                 => 'select',
            'sorting'                   => false,
            'options'                   => gwTurnierpaarliste::$TanzArten,
            'eval'                      => array('mandatory'=>true, 'tl_class' => 'w
50')
        ),
        'anzahlpaare' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['anzahlpaare'],
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'minlength' =>
1, 'maxlength'=>4, 'rgxp' => 'digit')
        ),
        'platz_von' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['platz_von'],
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'minlength' =>
1, 'maxlength'=>4, 'rgxp' => 'digit', 'tl_class' => 'w50')
        ),
        'platz_bis' => array
        (
            'label'                     => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['platz_bis'],
            'inputType'                 => 'text',
            'eval'                      => array('mandatory'=>false, 'minlength' =>
1, 'maxlength'=>4, 'rgxp' => 'digit', 'tl_class' => 'w50')
        ),
```

```
        'bemerkung' => array
        (
            'label'                   => &$GLOBALS['TL_LANG']['tl_gw_meldungen']
['bemerkung'],
            'inputType'               => 'textarea',
            'eval'                    => array('mandatory'=>false, 'cols' => 80, '
rows' => 20, 'allowHtml' => false)
        ),
    )
);
```

Hier gibt es keine großen Besonderheiten mehr. Gegenüber Schritt 11 habe ich die durchsuchbaren und sortierbaren Felder etwas erweitert bzw. geändert.

Nun kommt die Backendklasse, in der ich die Callbacks unterbringe:

PHP-Code:

```
class tl_gw_meldungen extends Backend
{

    /**
     * Import the back end user object
     */
    public function __construct()
    {
        parent::__construct();
        $this->import('BackendUser', 'User');
    }
```

Auftaktgeplänkel...

Zunächst der options_callback, der mir die Liste der aktiven Turnierpaare für die Auswahl in der Dropdown-Box liefert:

PHP-Code:

```
  public function getActiveCouples()
  {
      $couples = array();

        // Get all the active couples
        $objCouples = $this->Database->prepare("SELECT id,partnernachname,partnerv
orname,partnerinnachname,partnerinvorname FROM tl_gw_turnierpaare WHERE aktiv='1'
ORDER by partnernachname, partnervorname, partnerinnachname, partnerinvorname")
                            ->execute();

        while ($objCouples->next())
        {
            $k = $objCouples->id;
            $v = $objCouples->partnernachname;
            if($objCouples->partnervorname)
            {
               $v .= ', '.$objCouples->partnervorname;
      }
      if($objCouples->partnerinnachname)
      {
        $v .= ' und '.$objCouples->partnerinnachname;
        if($objCouples->partnerinvorname)
              {
                 $v .= ', '.$objCouples->partnerinvorname;
```

```
        }
      }

          $couples[$k] =$v;
        }

        return $couples;
  }
```

Der größte Teil der Funktion ist eigentlich das "Zusammenbasteln" des Paarnamens aus den Einzelbestandteilen der Namen von Partner und Partnerin. Zunächst werden aus der tl_gw_turnierpaare-Tabelle die Namen der aktiven Paare selektiert und der Ergebnisstring daraus zusammengesetzt. Das Ergebnis wird dann einem Array zugewiesen, wobei die ID des Paares der Array-Key ist, und der Paarname der Value - also z.B. $couples[47] = "Wupp, Willi und Wupp, Sieglinde".

In der Dropdownbox steht der String, und wenn ich den dort auswähle, landet die ID 47 als "pid" in der Datenbanktabelle tl_gw_meldungen. Und, was noch vieeeeel cooler ist: Wenn ich nach dem pid-Feld sortiere, dann werden auch die Sortierheader durch die entsprechenden Strings ersetzt, statt dass dort der numerische pid-Wert steht. Leider erfolgt die Sortierung natürlich nach dem Key, nicht nach dem Value...aber das wäre ja fast zuviel verlangt :-).

Weiter geht es mit label_callback. Als erstes Argument bekommt der die aktuelle Zeile aus der Datenbank, als zweites Argument den Label, wie er weiter oben im DCA-Record definiert wurde, hier also mit dem Platzhaltern #name# und #section_colour#.

PHP-Code:

```
    public function lookup_pid($row, $label)
    {
        $pid = $row['pid'];

        // Datensatz mit ID pid aus Tabelle tl_gw_turnierpaare holen
    $prow = $this->Database->prepare("SELECT * FROM tl_gw_turnierpaare WHERE id=?
")
                               ->execute($pid);
```

Mit der pid aus dem aktuellen Datensatz holen wir den Datensatz des Turnierpaares...

PHP-Code:

```
    $name = '<span style="font-weight: bold;">'.$prow->partnernachname.'</span>';
    if($prow->partnervorname)
    {
      $name .= ', '.$prow->partnervorname;
    };
    if($prow->partnerinnachname)
    {
      $name .= ' und <span style="font-weight: bold;">'.$prow-
>partnerinnachname.'</span>';
      if($prow->partnerinvorname)
      {
        $name .= ', '.$prow->partnerinvorname;
      }
    };
```

...und bauen aus den Namensbestandteilen den Paarnamen zusammen, wobei die Nachnamen fett erscheinen sollen.

PHP-Code:

```
    $colour = 'black';
    switch($row['lat_std'])
    {
      case 'Std':
        $colour = 'orange';
        break;
      case 'Lat':
        $colour = 'red';
        break;
    }
```

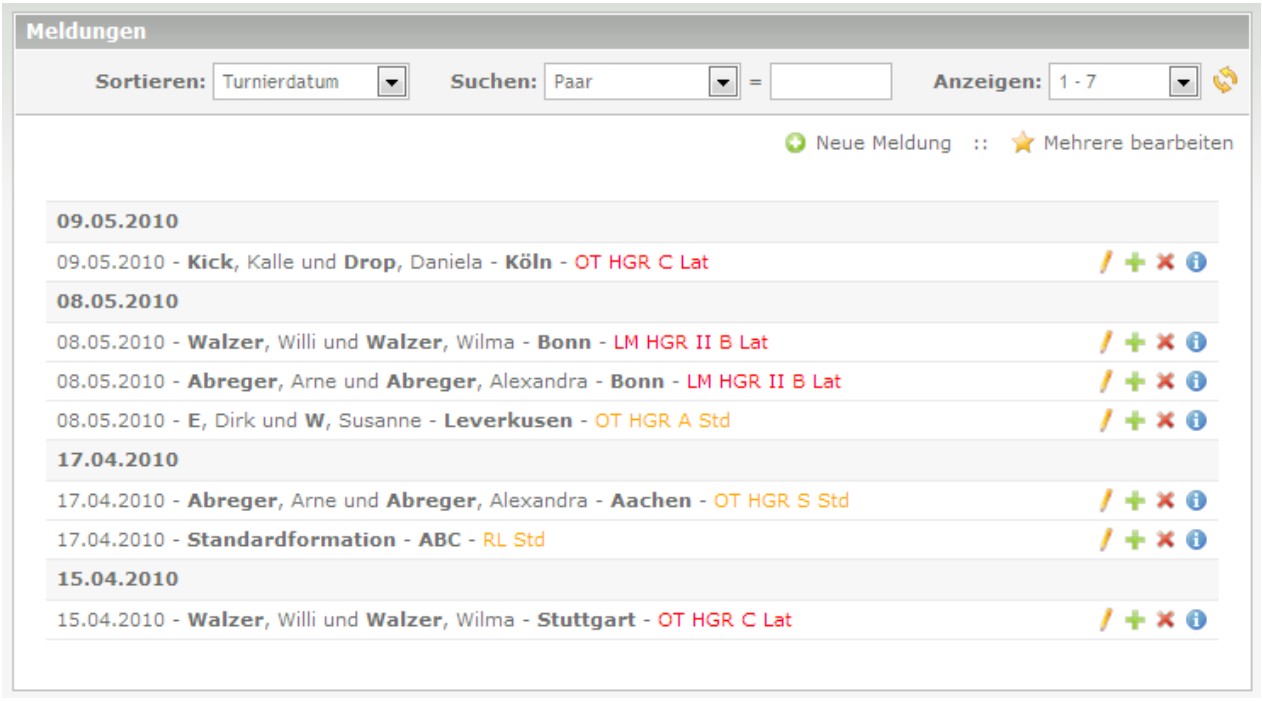
In Abhängigkeit vom Inhalt des lat_std-Feldes wird eine Farbe zugewiesen. Falls der Inhalt unbekannt ist, bleibt die Schrift schwarz.

PHP-Code:

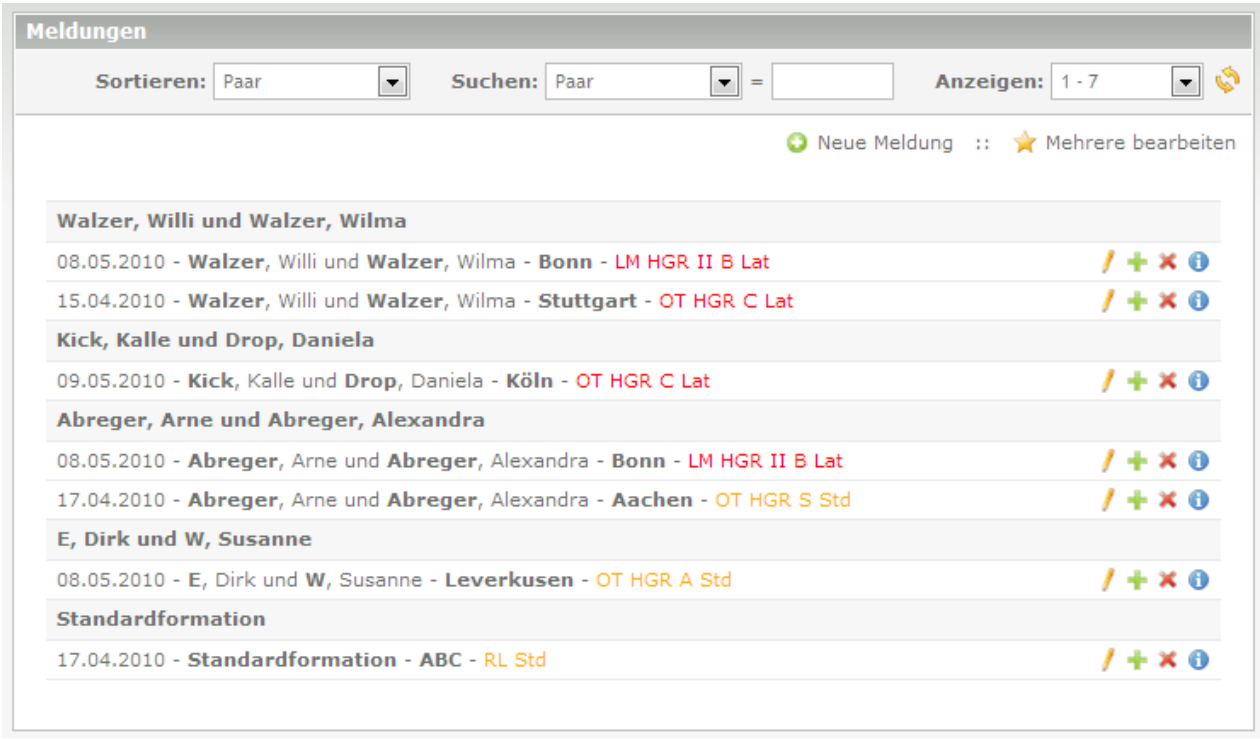
```
    $label = str_replace('#section_colour#', $colour, $label);

    $label = str_replace('#name#', $name, $label);

    return $label;
  }

};
?>
```

Wir ersetzen die Platzhalter #name# und #section_colour# im Label-String durch die neu berechneten Werte, und geben den Label zurück.

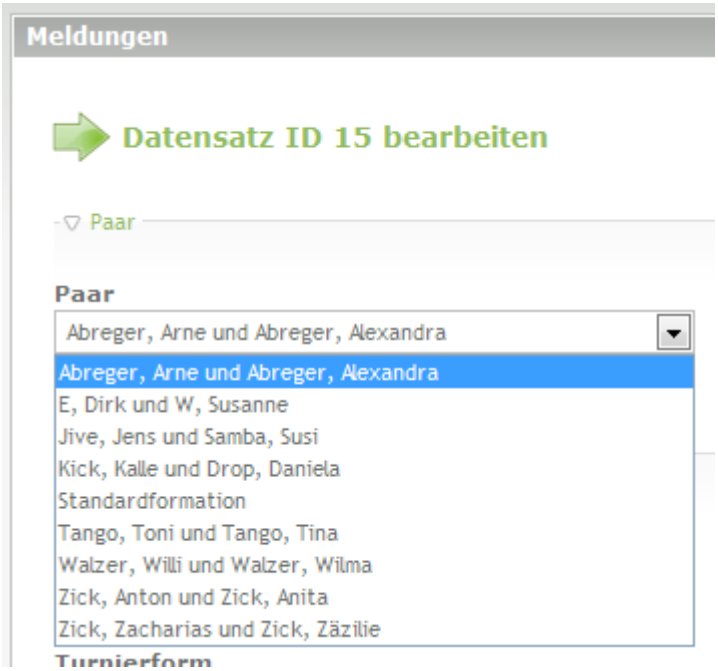
Was haben wir damit nun erreicht:



Unsere Startmeldungen sind nach dem Datum sortiert, die Nachnamen im Paarnamen sind fett, die Stadt auch, und die "Kategorisierung" des Turniers ist orange oder rot, je nachdem ob es um Standardtänze oder lateinamerikanische Tänze geht.



Sortieren wir nach den Paaren, dann wird die numerische pid in den Sortierheader durch unseren options_callback in die entsprechenden Strings umgesetzt - sehr nett. Leider erfolgt die Sortierung weiterhin numerisch nach der pid, und nicht alphabetisch nach den Strings. Vielleicht hat jemand noch eine Idee, wie man das lösen könnte?



Und bei Neuanlage einer Turniermeldung werden in der Dropdown-Liste nurnoch die aktiven Paare aufgeführt, nicht mehr ALLE.

Im nächsten Schritt geht es ans Frontendmodul für die Meldeliste...

Ich bin gerade in der DCA-Referenz über das Feld "findInSet" unter "eval" gestolpert:
"Sort by the actual option values instead of their labels (available from version 2.7.RC1)."

Das klingt wie das, was ich gerne für mein "pid"-Feld hätte, d.h. Sortierung nach den Paarnamen, die sich aus

der pid ergeben anstatt numerisch nach dem pid-Wert. Leider wirft das einen Fehler in DC_Table.php (Zeile 3468 bei TL 2.8.3), wo $keys nicht definiert ist. Wenn ich das richtig sehe, funktioniert das nämlich nur mit hardcodierten "options" im DCA-Record (Zeile 3461), ein options_callback wird dort nicht ausgewertet.

Kann das einer der "Profis" bestätigen? Wäre das dann nicht evtl. ein kleiner lohnenswerter Punkt für einen Feature-Request? Sieht für mich nach einem Zweizeiler der Marke "Falls options nicht gesetzt ist options_callback ausführen, dann weitermachen" aus.

EDIT: Ich habe mal ein Ticket dafür gemacht: http://dev.typolight.org/issues/1914
Habe meine DC_Table.php entsprechend angepasst, und damit funktioniert es bei mir wie gewünscht.

Setsunaa

Ich Entwickel seit zwei Monaten an einem Webseiten Katalog und wusste echt nicht was da auf mich zu kommt. Leider sind solche Art von "Katalogen" eher weniger vertreten und auch die catalouge extension für meine Dienste doch etwas zu überladen ist(da ich nix verkaufe) musste ich mich mit den gleichen Sachen asueinandersetzen und dein Tagebuch war eine große hilfe.

auch wenn ich oft geflucht ahbe, denn bei dir funktionierten Sachen, die ich etwas umständlciher machen musste. Was aber auch an meiner Datenbank struktur liegt. Hällt man sich nicht an die gedachten Strukturen, muss man einiges nachimplementieren. Um genauer zu sein ist es für mein vorhaben nicht so simpel nur 1:n beziheungen zu finden.

Zum Verständniss: eine Webseite hat einen ersteller/empfehler und einen Webmaster/Kontaktperson. Diese können gleich aber auch unterschiedlcih sein, sind aber beide in der tl_member gespeichert. Klar könnte ich miene Tabelle noch etwas "zerlegen" und schauen das ich den webmaster in eine kleine Tabelle auslagere. Bloß die Frage ist wie sinnvoll es für mich ist. Ich fahre nebenbei kleine Geschwindigkeitstests, da ich die komplette Seite dynamisch lassen muss...naja egal man könnte darauf eingehen, wenn es gewollt ist.

ich wollte darauf hinaus, je spezieller der wunsch ist, desto mehr eigenarbeit ist dabei.
Als Beispiel nehme man das save_callback für ein field. Funktioniert einwandfrei, außer ich definiere vorher ein input_field_callback, dann wird dieses Callback nicht mehr angerührt und ich muss einen Globalen callback ausführen. Und das nur, weil es eigentlich nur ein "anzeigefeld" sein sollte, was automatisch befüllt wird und auch nicht änderbar ist. (Vielleicht ist aber auch genau das der Knackpunkt warum es nicht geht)

so aber lange rede kurzer sinn 

> Zitat von **dl1ely**
> *Das klingt wie das, was ich gerne für mein "pid"-Feld hätte, d.h. Sortierung nach den Paarnamen, die sich aus der pid ergeben anstatt numerisch nach dem pid-Wert. Leider wirft das einen Fehler in DC_Table.php (Zeile 3468 bei TL 2.8.3), wo $keys nicht definiert ist. Wenn ich das richtig sehe, funktioniert das nämlich nur mit hardcodierten "options" im DCA-Record (Zeile 3461), ein options_callback wird dort nicht ausgewertet.*

mir ist auch aufgefallen und ich hoffe ich lehne mich nicht zu weit aus dem fenster, aber dieser Treiber scheint älter zu sein. Denn ich finde man merkt bei Typolight die struktur und die überlegungen dahinter. Es gibt zwei möglichkeiten die DC_table zu erweitern. du kannst(solange es keine sortierung oder filter sein soll) kannst du dort wo du das label definierts und die Felder dafür, Tabellenübergreifend arbeiten.

Denn im Code wird der String nochmal zerlegt. du kannst sozusagen <feldname>:<table_name>.<table_field> sagen, dann macht er nochmal einen DB aufruf und holt diese Information für dich.
Willst du es beim Sortieren haben, gibt es auch dort ein hook, welcher aber leider erst aufgerufen wurde nachdem die DB abgefragt wurde(und auch leider schon sortiert). Nun bekomme ich schon das Ergebniss und kann dieses abändern. An sich ja auch nicht schlecht immerhin noch besser als ids dort stehen zu haben, aber dadurch ist es nur eine "gruppierung" und keine Sortierung mehr. Aber bei genauem überlegen(da ich mir dachte ok...dann halt mein eigener Treiber) glaube ich, das es leider dann nicht mehr trivial und performant sortiert werden kann...ist aber noch nicht ins detail durchdacht.

also bei mir sieht es nun so aus:

PHP-Code:

```
'label' => array (
            'fields'            => array ('id', 'title', 'url', 'ersteller_id:tl_m
ember.username'),
            'format'            => '<span style="font-weight: bold;">#%s</span> WC
ard von <span style="font-weight: bold;">%s</span>'
                                    . ' (%s) Empfohlen von: <span style="font-
weight: bold;">%s</span>',
            'group_callback'    => array('WCards', 'listViewGroupCallback')
        ),
```

dass Callback ist für die Nachbearbeitung wenn Sortiert wird. Obwohl ich deine Lösung auch sehr interessant finde und auch heute gleich mal testen werde, ob ich es auch nicht so lösen kann.

Nochmal Danke für deine mühe...auch wenn es schade ist, dass nun "erst"(ist nicht abwertend gemeint) dort bist, da ich gerade nachlesen wollte, was du bezüglich des frontend Inputs tust, da ich mir dort gerade den Kopf zerbreche.

---

dl1ely

Leider kann ich dir mit deinen Ausführungen in der zweiten Hälfte eines Posts nicht so ganz folgen.
Die Notation "feld_A:Tabelle_B.feld_B" im DCA-Record für Tabelle A ist mir gestern erst versehentlich unter die Augen gekommen (Ticket-ID #88). In meinem Fall würde es mir nicht soviel helfen, da ich einen String brauche, der sich aus mehreren Feldern aus Tabelle B zusammensetzt, und das auch noch abhängig von den Inhalten. Ohne eigenen PHP-Code geht es da nicht, darum muss der Label-Callback bleiben.

Bezüglich der Sortierung nach den Values aus dem options_callback habe ich ein Ticket aufgemacht (#1914), und den dort von mir vorgeschlagenene Code in meine eigene DC_Table.php übernommen. Funktioniert bei mir einwandfrei. Ich hoffe, dass Leo meinen Codevorschlag übernimmt und die nächste TL-Version das kann.

Mit meiner Entwicklung wirds noch weitergehen, auch der User-Input im Frontend kommt noch. Leider ist die Freizeit begrenzt, und es ist auch nur ein Freizeitprojekt. Von daher kann ich es leider nicht "runterrattern". Außerdem dauert das Erstellen jedes Schrittes ca. 1h, mit Tippen, Pasten vom Code in den Post und dem Anfertigen der Screenshots. Aber egal, da muss ich jetzt durch. Das Ende ist in Sicht :-).

Setsunaa

sorry das ich mich unverständlich ausgedrückt habe. Dies ist ein Problem, was häufiger auftritt, da ich ein, wie sagt man, "ein in ein schwarzes Loch hineindenker" bin . Aber ich versuche mein bestes, also bitte nicht übel nehmen...

Stimmt du setzt deinen String zusammen, dies dachte ich mir auch, als ich dir den Vorschlag geschrieben hatt, aber wollte diesen nicht rausnehmen, da es ja vielleicht anderen helfen kann.

Das mit dem Hook ist glaube ich eher etwas, was nur auftritt, wenn man nach etwas sortieren möchte, wo der eigentliche wert aber in einer anderen Tabelle steht. Wie es bei mir der Fall ist, da es gewünscht ist. Also zumindest wenn man sich nicht an die notation hällt. Da ich dies nicht tun kann(zumindest nicht in der Form, die ich jetzt habe), da ich zwei mal die id eines members in der Tabelle habe.

Also in "meiner" Tabelle steht ein ersteller für die Webseiten Karte, dies ist aber nur die id der tl_member. Wenn ich danach sortieren möchte, wird nur die Zahl aus "meiner" tabelle sortiert und ich habe keine möglichkeit gefunden zu sagen, das er diesen wert vorher mit dem usernamen aus tl_member zu ersetzen.

Leider nur danach, dadurch ist es halt nur eine Gruppierung.

Die Stelle in der DC_Table wo du gerne die Änderung hättest hatte ich schon gesehen, aber als für mich unbrauchbar abgestempelt, da dies mir nicht weiterhilft. Aber vielleicht habe ich auch nur etwas übersehen...

Naja ich muss eh viel selber schreiben, dank (an manchen stellen wirklich unnötige) Abhängigkeiten und Wünsche.

---

dl1ely

> Zitat von **Setsunaa**
> *Das mit dem Hook ist glaube ich eher etwas, was nur auftritt, wenn man nach etwas sortieren möchte, wo der eigentliche wert aber in einer anderen Tabelle steht. Wie es bei mir der Fall ist, da es gewünscht ist. Also zumindest wenn man sich nicht an die notation hällt. Da ich dies nicht tun kann(zumindest nicht in der Form, die ich jetzt habe), da ich zwei mal die id eines members in der Tabelle habe.*
>
> *Also in "meiner" Tabelle steht ein ersteller für die Webseiten Karte, dies ist aber nur die id der tl_member. Wenn ich danach sortieren möchte, wird nur die Zahl aus "meiner" tabelle sortiert und ich habe keine möglichkeit gefunden zu sagen, das er diesen wert vorher mit dem usernamen aus tl_member zu ersetzen. Leider nur danach, dadurch ist es halt nur eine Gruppierung.*

Und das findInSet zusammen mit einem Options-Callback hilft dir nicht? Kann deinen konkreten Fall da gerade nicht überschauen. Wie gesagt, bei mir löst es mein Problem...

Setsunaa

hmm...wäre möglich, aber ich empfinde es als unschön und mein erster Gedanke dabei war, wie performant ist es, ab einer gewissen User Zahl. Denn dieses Callback würde wenn dann bei jedem einzelnen ausgeführt werden müssen. Aber es könnte auch ganz einfach sein, ich schau mir deine Lösung noch an. Danke für den Hinweiß.

Naja ist ersteinmal nicht so wichtig, das kann ich noch später ändern...und wenn es soll eh mehr über das Frontend gemacht werden, dafür muss ich eh noch einen "Admin" Bereich bauen...
Welches mir heute wirklich Kopfschmerzen bereitet....vorallem durch den Spruch "keep it simpel" und dann hat man kleinigkeiten, die echt nicht leicht zu lösen sind...naja einfach nur speziell

---

dl1ely

> Zitat von **dl1ely**
> *Bezüglich der Sortierung nach den Values aus dem options_callback habe ich ein Ticket aufgemacht (#1914), und den dort von mir vorgeschlagenene Code in meine eigene DC_Table.php übernommen. Funktioniert bei mir einwandfrei. Ich hoffe, dass Leo meinen Codevorschlag übernimmt und die nächste TL-Version das kann.*

Juchu:
* Status wurde geändert von New zu Accepted.
* Zielversion wurde gesetzt auf 2.9.0.

Schritt 13: Frontendmodul Meldeliste

So, weiter geht es mit dem vorerst letzten Schritt: Das Frontendmodul für die Meldeliste.
Besondere Herausforderungen sind hier die Möglichkeit der Verknüpfung einzelner Einträge mit dem Frontendmodul für die Turnierpaardetails, und eine konfigurierbare seitenweise Anzeige der Einträge der Meldeliste (Da diese im Lauf der Zeit sehr umfangreich werden kann).

Die Klasse des Frontendmoduls heisst "gwMeldeliste", und da ich sie schon bei der Erstellung im Extension Creator angegeben hatte, ist diese schon als Frontendmodul registriert. Trotzdem checken wir das nochmal in /system/modules/gw_turnierpaare/config.php :

PHP-Code:

```
// Front end module
array_insert($GLOBALS['FE_MOD']['turnierpaare'], 0, array
(
    'gw_turnierpaarliste' => 'gwTurnierpaarliste',
    'gw_meldeliste' => 'gwMeldeliste'
));
```

Für die Konfiguration des Frontend-Moduls für die Darstellung benötige ich zwei Parameter: Die Anzahl der Datensätze pro Seite und die URL, auf die weitergeleitet werden soll, wenn man auf einen Tanzpaarnamen klickt (Das sollte eine URL sein, auf der das Frontendmodul der Turnierpaarliste eingebunden ist). Darum muss /system/modules/gw_turnierpaare/config/database.sql erweitert werden:

Code:

```
--
-- Extend table 'tl_module'
--

CREATE TABLE `tl_module` (
  `gw_tp_showonlyactive` char(1) NOT NULL default 'B',
  `gw_tp_couplesorting` char(1) NOT NULL default 'A',
  `gw_ml_pagesize` int(4) NOT NULL default '50',
  `gw_ml_coupledetails` varchar(255) NULL default NULL
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

gw_ml_pagesize ist also ein int mit maximal 4 Stellen, gw_ml_coupledetails ein string.

Damit die beim Anlegen des Moduls auch angezeigt und editiert werden können, müssen wir an das DCA von tl_module ran, also in /system/modules/gw_turnierpaare/dca/tl_module.php einfügen:

PHP-Code:

```
$GLOBALS['TL_DCA']['tl_module']['palettes']['gw_meldeliste'] = '{title_legend},nam
e,headline,type;{size_legend},gw_ml_pagesize, gw_ml_coupledetails;
{protected_legend:hide},protected;{expert_legend:hide},guests,cssID,space';

$GLOBALS['TL_DCA']['tl_module']['fields']['gw_ml_pagesize'] = array
(

    'label'                   => &$GLOBALS['TL_LANG']['tl_module']
['gw_ml_pagesize'],
    'default'                 => '50',
    'exclude'                 => true,
    'inputType'               => 'text',
  'eval'                    => array('mandatory'=>true, 'minlength' => 1, 'maxleng
```

```
th'=>4, 'rgxp' => 'digit', 'tl_class' => 'w50')
);

$GLOBALS['TL_DCA']['tl_module']['fields']['gw_ml_coupledetails'] = array
(

    'label'                   => &$GLOBALS['TL_LANG']['tl_module']
['gw_ml_coupledetails'],
    'exclude'                 => true,
    'inputType'               => 'text',
  'eval'                    => array('mandatory'=>false, 'tl_class' => 'w50')
);
```

Zunächst legen wir die neue Palette an, in der neben einigen Standardfeldern unsere beiden neuen Datenbankfelder stehen, und anschließend definieren wir diese Felder im DCA. Für die pagesize nehme ich einen Default von 50, ansonsten sind die Optionen inzwischen wohlbekannt.

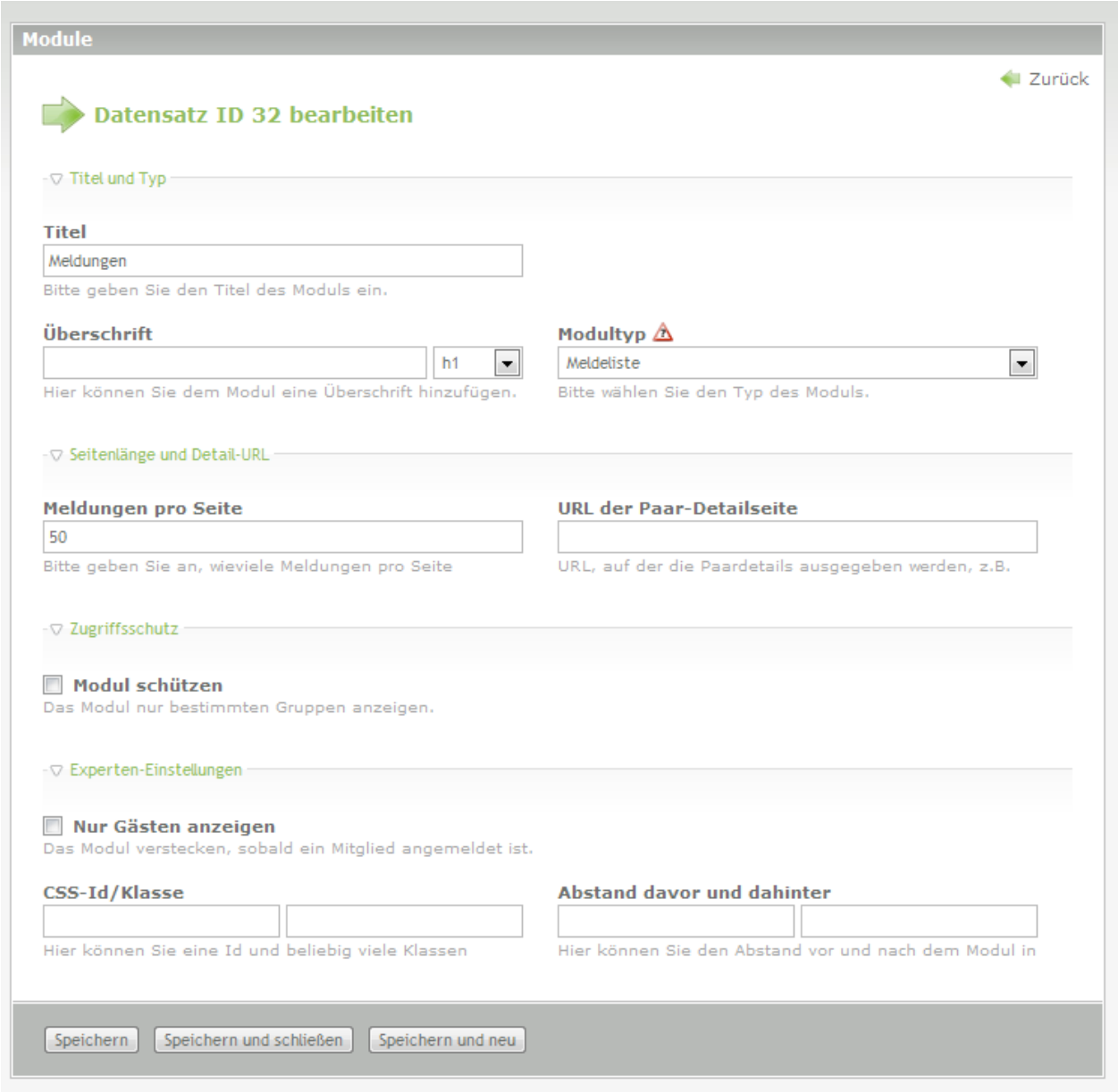
Nun brauchen wir noch hübsche Labels für die neuen Felder im Backend.
/system/modules/gw_turnierpaare/languages/de/modules.php:

PHP-Code:

```
$GLOBALS['TL_LANG']['tl_module']['size_legend']     = 'Seitenlänge und Detail-
URL';

$GLOBALS['TL_LANG']['tl_module']['gw_ml_pagesize'] = array('Meldungen pro Seite',
'Bitte geben Sie an, wieviele Meldungen pro Seite ausgegeben werden sollen');
$GLOBALS['TL_LANG']['tl_module']['gw_ml_coupledetails'] = array('URL der Paar-
Detailseite' , 'URL, auf der die Paardetails ausgegeben werden, z.B. /turnierpaarl
iste/info/');
```

Entsprechend die englische Variante natürlich genauso.

Damit sind wir im Backend hier angekommen:



Nun das Frontendmodul /system/modules/gw_turnierpaare/gwMeldeliste.php :

PHP-Code:

```
<?php if (!defined('TL_ROOT')) die('You can not access this file directly!');

/**
 * Class gwMeldeliste
 *
 * @copyright  (C) 2010
 * @author     Stefan Pfeiffer
 * @package    Controller
 */
class gwMeldeliste extends Module
{

    /**
     * Template
     * @var string
     */
    protected $strTemplate = 'gw_meldeliste';
```

```
    protected $strErrorTemplate = 'gw_meldeliste_error';
```

Zwei Templatenamen: Einer für die Ausgabe der Liste, der andere wenn etwas schiefgegangen ist...

PHP-Code:

```
  public static $strPageKey = 'page';
```

Der URL-Bestandteil, der beim Blättern in der Liste unsere aktuelle Seite mitzählt. Mit 'page' also z.B. .../meldeliste/page/3.html um die dritte Seite anzeigen zu lassen.

PHP-Code:

```
    /**
     * Generate module
     */
    protected function compile()
    {
        $moduleParams = $this->Database->prepare("SELECT gw_ml_pagesize, gw_ml_cou
pledetails FROM tl_module WHERE id=?")
                ->limit(1)
                ->execute($this->id);

    $pagesize = $moduleParams->gw_ml_pagesize;

    $this->Template->coupledetails = $moduleParams->gw_ml_coupledetails;
```

Zunächst holen wir uns unsere beiden Modulparameter aus der Datenbank. die Detail-URL schreiben wir gleich ins Template, die pagesize merken wir uns erstmal.

PHP-Code:

```
    if ( strlen($this->Input->get(gwMeldeliste::$strPageKey)) )
        {
      if(is_numeric($this->Input->get(gwMeldeliste::$strPageKey)))
      {
        $page = $this->Input->get(gwMeldeliste::$strPageKey);
      }
      else
      {
        // Error
        $this->Template = new FrontendTemplate($this->strErrorTemplate);
        return;
      }
        }
        else
        {
      $page = 0;
    }

    $this->Template->page = $page;
```

Falls eine Seitenzahl in der URL angegeben, wird diese in $page gespeichert. Falls die Angabe nicht-numerisch ist, wird das Fehlertemplate ausgegeben. Und wenn keine Seitenzahl angegeben wird, tragen wir 0 ins Template ein.

PHP-Code:

```
    $limit = " LIMIT ".($pagesize*$page).",".$pagesize;

    $testNextPage = $this->Database->execute("SELECT * from tl_gw_meldungen ORDER
BY datum DESC LIMIT ".($pagesize*($page+1)).",1");
    if($testNextPage->numRows < 1)
    {
      $this->Template->nextpage = -1;
    }
    else
    {
      $this->Template->nextpage = $page+1;
    }
```

Hier basteln wir uns das passende "LIMIT"-Statement für die aktuell ausgewählte Seite zusammen. Ausserdem machen wir einen Testselect auf die erste row der darauffolgenden Seite. Falls diese existiert, wird die Nummer der nächsten Seite ins Template geschrieben, ansonsten eine -1 als Hinweis fürs Template.

Ich weiss, dass man diesen Check, ob eine weitere Seite existiert auch anders lösen kann, wahrscheinlich sogar eleganter. Aber erstmal funktioniert es :-).

PHP-Code:

```
    $arrMeldungen = array();
```

Hier kommen die ganzen Datensätze des Ergebnis rein...

PHP-Code:

```
    $objMeldungen = $this->Database->execute("SELECT * FROM tl_gw_meldungen ORDER
BY datum DESC".$limit);

    if($objMeldungen->numRows == 0)
    {
      // Error
      $this->Template = new FrontendTemplate($this->strErrorTemplate);
      return;
    }
```

Ergebnisdatensätze aus der DB holen - wenn keine gefunden wurden, dann Error-Template ausgeben.

PHP-Code:

```
    while($newArr = $objMeldungen->fetchAssoc())
    {
```

Jede Row in ein assoziatives Array umwandeln...

PHP-Code:

```
      $objPaar = $this->Database->execute("SELECT * FROM tl_gw_turnierpaare WHERE
id=".$newArr['pid']);

      $name =   $objPaar->partnernachname;
      if($objPaar->partnervorname)
      {
        $name .= ', '.$objPaar->partnervorname;
      }
      if($objPaar->partnerinnachname)
      {
```

```
        $name .= ' und '.$objPaar->partnerinnachname;
      }
      if($objPaar->partnerinvorname)
      {
        $name .= ', '.$objPaar->partnerinvorname;
      }
      $newArr['name'] = $name;
      $newArr['paaralias'] = $objPaar->alias;
```

Und für jede Row aus der pid den Namen und den Alias des Turnierpaares bestimmen und mit ins assoziative Array aufnehmen.

PHP-Code:

```
      $arrMeldungen[] = $newArr;
    }

    $this->Template->meldungen = $arrMeldungen;
    }
}
?>
```

Schließlich wird jeder Datensatz an das große Ergebnisarray angehangen und dann ins Template geschrieben.

Womit wir zu den Templates kommen. Zunächst /system/modules/gw_turnierpaare/templates/gw_meldeliste_error.tpl :

PHP-Code:

```
<div class="<?php echo $this->class; ?> block meldeliste"<?php echo $this->cssID;
?>

<?php if ($this->style): ?> style="<?php echo $this->style; ?>"<?php endif; ?>>



<h3>Es trat ein Fehler auf!</h3>
```

Erstmal easy as this, hier könnte man sich natürlich beliebig weiter austoben als nur mit diesem lapidaren Satz.

Spannender ist /system/modules/gw_turnierpaare/templates/gw_meldeliste.tpl :

PHP-Code:

```
<div class="<?php echo $this->class; ?>"<?php echo $this->cssID; ?><?php if
($this->style): ?> style="<?php echo $this->style; ?>"<?php endif; ?>>
<?php if ($this->headline): ?>
<<?php echo $this->hl; ?>><?php echo $this->headline; ?></<?php echo $this->hl; ?
>>
<?php endif; ?>
```

Standardauftakt...

PHP-Code:

```
<table cellpadding="4" cellspacing="0" summary="Meldeliste">
  <thead>
    <tr>
      <th class="centered">Datum</th>
```

```
      <th class="centered">Name</th>
      <th class="centered">Ort</th>
      <th class="centered">Turnier</th>
      <th class="centered">Turnierart</th>
      <th class="centered">Platz</th>
      <th class="centered">Paare</th>
      <th class="centered">Bemerkung</th>
    </tr>
  </thead>
```

Header der Tabelle...

PHP-Code:

```
  <tbody>
<?php foreach ($this->meldungen as $meldung): ?>
```

Wir gehen durch alle rows im Array durch.

PHP-Code:

```
  <tr>
  <td class="centered">
  <?php echo date('d.m.Y', $meldung['datum']); ?>
  </td>
```

Datum "richtig" formatieren...

PHP-Code:

```
  <td><a href="<?php echo $this->coupledetails; ?><?php echo $meldun
g['paaralias']; ?>.html"><?php echo $meldung['name']; ?></a>
  </td>
```

Hier wird der Paarname und als hinterlegter Link die Detail-URL mit dem Paaralias ausgegeben.

PHP-Code:

```
  <td class="centered"><strong><?php echo $meldung['turnierort']; ?></strong>
  </td>
  <td class="centered<?php if ($meldung['lat_std'] == 'Std'){echo ' std';} else {
if ($meldung['lat_std'] == 'Lat'){echo ' lat';}}?>"><?php echo $meldun
g['startgruppe']; ?> <?php echo $meldung['startklasse']; ?> <?php echo $meldun
g['lat_std']; ?>
  </td>
```

In Abhängigkeit von der Tanzart wird dem TD eine CSS-Klasse zugewiesen (zur farblichen Absetzung).

PHP-Code:

```
  <td class="centered"><?php echo $meldung['turnierart']; ?>
  </td>
  <td class="centered"><?php echo $meldung['platz_von']; ?>
  <?php if (strlen($meldung['platz_bis']) > 0): ?>
  <?php if ($meldung['platz_von'] != $meldung['platz_bis']): ?>
  <?php echo " - ".$meldung['platz_bis']; ?>
  <?php endif; ?>
  <?php endif; ?>

  </td>
```

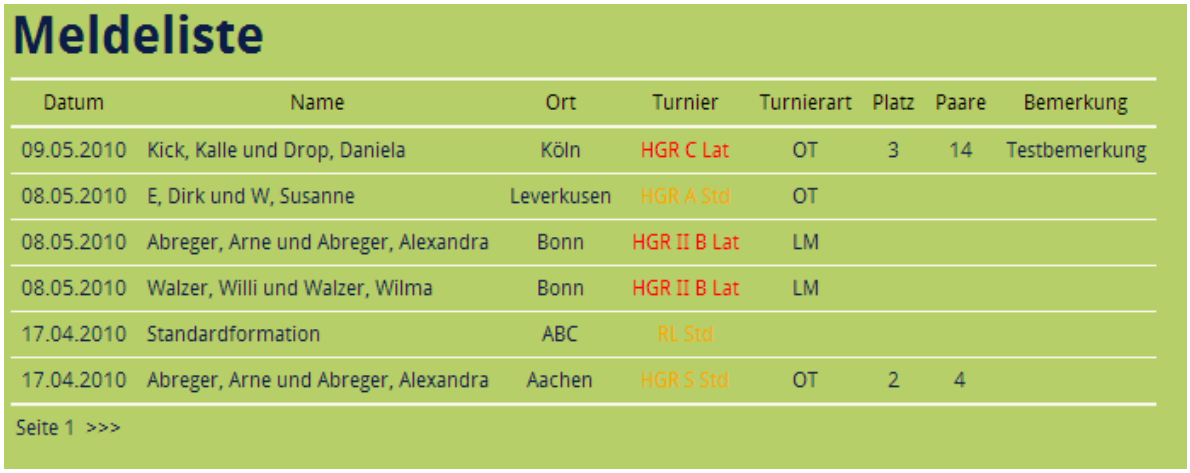
```
  <td class="centered"><?php echo $meldung['anzahlpaare']; ?>
  </td>
  <td><?php echo $meldung['bemerkung']; ?>
  </td>
  </tr><?php endforeach; ?>
  </tbody>
</table>
```

So weit unsere Tabelle...Es fehlen noch die Links, um eine Seite vor oder zurück zu springen...wenn es dann davor oder dahinter noch Seiten gibt.

PHP-Code:

```
<?php if ($this->page > 1): ?>
<a id="prev" href="/{{env::page_alias}}/<?php echo gwMeldeliste::$strPageKey; ?
>/<?php echo ($this->page-1); ?>.html"<<<<</a>
<?php elseif ($this->page == 1): ?>
<a href="/{{env::page_alias}}.html"<<<<</a>
<?php endif; ?>

 Seite <?php echo ($this->page+1); ?> 

<?php if ($this->nextpage >= 0): ?>
<a id="next" href="/{{env::page_alias}}/<?php echo gwMeldeliste::$strPageKey; ?
>/<?php echo $this->nextpage; ?>.html">>>>></a>
<?php endif; ?>

</div>
```

Das liefert uns im Frontend (ohne weitere CSS-Modifikationen) bei 6 Meldungen (nur zur Demo) pro Seite:

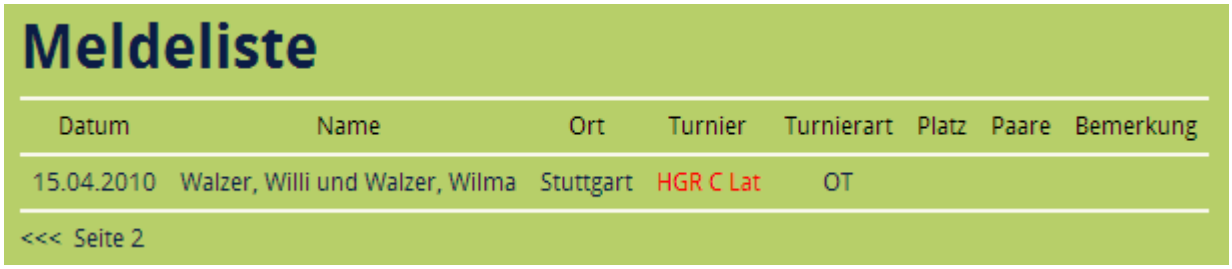


Meldeliste

Datum	Name	Ort	Turnier	Turnierart	Platz	Paare	Bemerkung
09.05.2010	Kick, Kalle und Drop, Daniela	Köln	HGR C Lat	OT	3	14	Testbemerkung
08.05.2010	E, Dirk und W, Susanne	Leverkusen	HGR A Std	OT			
08.05.2010	Abreger, Arne und Abreger, Alexandra	Bonn	HGR II B Lat	LM			
08.05.2010	Walzer, Willi und Walzer, Wilma	Bonn	HGR II B Lat	LM			
17.04.2010	Standardformation	ABC	RL Std				
17.04.2010	Abreger, Arne und Abreger, Alexandra	Aachen	HGR S Std	OT	2	4	

Seite 1 >>>

und beim Weiterschalten auf die zweite Seite (.../page/1.html):

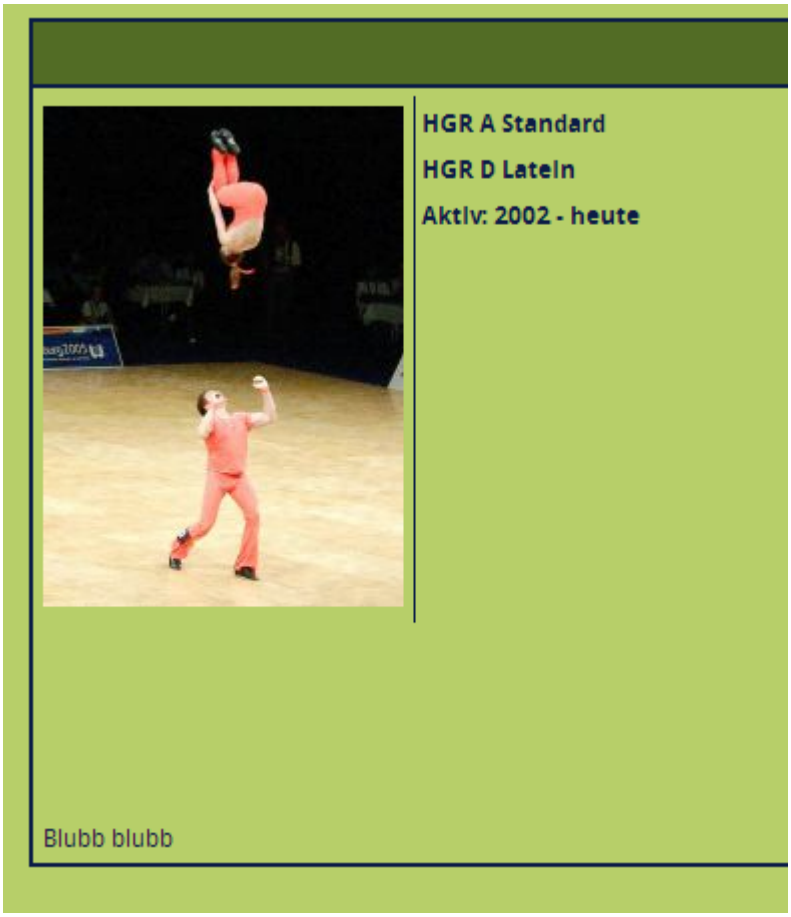


Meldeliste

Datum	Name	Ort	Turnier	Turnierart	Platz	Paare	Bemerkung
15.04.2010	Walzer, Willi und Walzer, Wilma	Stuttgart	HGR C Lat	OT			

<<< Seite 2

Ein Klick auf den Turnierpaarnamen springt zur Detail-URL, bei mir /turnierpaarliste/info/<alias>.html:



So, das soll es erstmal gewesen sein. Ich habe eine Menge gelernt, und würde den Code JETZT ganz anders aufziehen. Und das werde ich auch tun (haha). Allerdings ist der Zeithorizont unklar, und ich werde es auch nicht mehr schaffen "nebenbei" das Tagebuch zu führen. Pro Folge war es doch ca. eine Stunde Arbeit - mehr als gedacht! Mir fehlen für meinen Einsatz des Moduls auch noch einige kleine Features, aber das ist wirklich sehr speziell. Zunächst werde ich gemeinsam genutzte Funktionen der Back- und Frontendmodule wohl in eine gemeinsame Klasse packen, um die Module und auch die Templates etwas zu entschlacken.

Vielleicht werde ich zu gegebener Zeit zum Thema "AJAX" in Frontendmodulen nochmal etwas schreiben, aber dann nur speziell auf diesen Teilbereich bezogen.

Es tut mir leid, dass es hier die letzten Wochen etwas zäh und "langweilig" wurde, aber unerwarteter Weise war meine Freizeit (in der ich das hier tue) knapper als gedacht. Ich hoffe, Einige konnten teilweise Kleinigkeiten aus meinem Lernprozess mitnehmen , egal wie chaotisch er war, und für sich sinnvoll nutzen.

---

deerwood

Zum Schritt 13 habe ich noch 2 Anmerkungen:

1. Statt Deinem Test für eine nächste Seite würde ich ein "SELECT COUNT(*) AS count FROM tl_gw_meldungen" machen, das kann MySQL sehr schnell, ohne wirklich auf die Datensätze zugreifen zu müssen. Basierend auf der Anzahl aller Datensätze und unter Berücksichtigung der Sätze pro Seite kann man dann eine beliebige Paginierung erzeugen (etwa auch "Erste/Letzte Seite" oder "5 Seiten weiter/zurück").
2. Statt die Meldungen erst zu selektieren und dann in der WHILE Schleife die Paar-Infos (für die Namen) einzeln nachzuselektieren solltest Du vor der Schelife einen JOIN zwischen 'tl_gw_meldungen' und 'tl_gw_turnierpaare' machen und in der Schleife dann nur noch die Namen zusammensetzen. Das ist erheblich performanter, es wird nur 1 statt 51 Statements abgesetzt und JOIN ist ja nun gerade eine der Stärken von relationalen Datenbanken.

Jedenfalls wünsche ich Dir sehr viel Erfolg bei Deiner geplanten Überarbeitung und ich würde mich SEHR freuen, wenn Du auch noch etwas zu AJAX schreibst.

---

dl1ely

> Zitat von **deerwood**
>
> *Zum Schritt 13 habe ich noch 2 Anmerkungen:*
>
> > *1. Statt Deinem Test für eine nächste Seite würde ich ein "SELECT COUNT(*) AS count FROM tl_gw_meldungen" machen, das kann MySQL sehr schnell, ohne wirklich auf die Datensätze zugreifen zu müssen. Basierend auf der Anzahl aller Datensätze und unter Berücksichtigung der Sätze pro Seite kann man dann eine beliebige Paginierung erzeugen (etwa auch "Erste/Letzte Seite" oder "5 Seiten weiter/zurück").*

Ja, das geht sicherlich auch, und mit der Gesamtzahl der Datensätze kann man die von dir beschriebenen Dinge umsetzen. Ob das bezüglich der Performance wirklich so viel Unterschied macht? Ich glaube nicht, dass das wirklich messbar ist, wenn es einmal Seitenabruf aufgerufen wird. Ich überlasse das mal als Übung dem interessierten Leser ;-).

> Zitat von **deerwood**
>
> > *2. Statt die Meldungen erst zu selektieren und dann in der WHILE Schleife die Paar-Infos (für die Namen) einzeln nachzuselektieren solltest Du vor der Schelife einen JOIN zwischen 'tl_gw_meldungen' und 'tl_gw_turnierpaare' machen und in der Schleife dann nur noch die Namen zusammensetzen. Das ist erheblich performanter, es wird nur 1 statt 51 Statements abgesetzt und JOIN ist ja nun gerade eine der Stärken von relationalen Datenbanken.*
>
> *Jedenfalls wünsche ich Dir sehr viel Erfolg bei Deiner geplanten Überarbeitung und ich würde mich SEHR freuen, wenn Du auch noch etwas zu AJAX schreibst.*

Ja, da hast Du natürlich völlig recht. Ein Join ist deutlich besser. So wie ich das gemacht habe, war das quick'n'dirty, und sollte nicht als Vorbild dienen. Mir war erstmal unklar, welche Keys im assoziativen Array genutzt werden, wenn ich zwei Tabellen joine. Muss ich ein Feld wie "tabelle2.partnernachname" dann immer mit "AS" umbenennen? Ich muss mir mal anschauen, wie das in anderen Modulen gemacht wird, und dann schiebe ich nochmal eine Version nach, die es besser macht, und die Anzahl der Queries in der Tat drastisch reduziert :-).

Schritt 13b: JOIN-Power

Wie oben besprochen habe ich die Schleife über die Meldungen durch einen Join ersetzt. Im assoziativen Array werden einfach die Feldnamen aus jeder Tabelle benutzt, bei Konflikten gewinnt vermutlich die letztgenannte Tabelle(?).

/system/modules/gw_turnierpaare/gwMeldeliste.php sieht jetzt im unteren Teil so aus:

PHP-Code:

```
    $objMeldungen = $this->Database->execute("SELECT * FROM tl_gw_meldungen m, tl_
gw_turnierpaare p WHERE m.pid = p.id ORDER BY datum DESC".$limit);

    if($objMeldungen->numRows == 0)
    {
      // Error
      $this->Template = new FrontendTemplate($this->strErrorTemplate);
      return;
```

```
    }

    $arrMeldungen = $objMeldungen->fetchAllAssoc();
    $templateResult = array();

    foreach($arrMeldungen as $meldung)
    {
      $name = $meldung['partnernachname'];
      if($meldung['partnervorname'])
      {
        $name .= ', '.$meldung['partnervorname'];
      }
      if($meldung['partnerinnachname'])
      {
        $name .= ' und '.$meldung['partnerinnachname'];
      }
      if($meldung['partnerinvorname'])
      {
        $name .= ', '.$meldung['partnerinvorname'];
      }
      $meldung['name'] = $name;
      $templateResult[] = $meldung;
    }
    $this->Template->meldungen = $templateResult;
    }
}
```

Also statt einer Schleife über alle Rows mit neuem SELECT für den Paarnamen nurnoch eine Schleife über alle Result-Rows, um aus den Namensbestandteilen den Anzeigenamen zu basteln...Wesentlich performanter!

MBM

Detailseiten

ich habe mir Dein Modul mal runtergeladen und es mal installiert und getestet. Es funktioniert auch soweit bis auf die Ansicht der Detailseite. Bekomme immer ein *Page not found...*

Ich frage mich wie Du das gelöst hast. Hat es vielleicht mit meiner Konfiguration des Apache Webserver zu tun?

Ich benutze in der Apache Konfiguration mod_rewrite und das Contao CMS 2.8.3 liegt in einem ALIAS Ordner. Auch ist inder CMS Konfiguration die index.php Anzeige ausgeschaltet (eben mod_rewrite).

URL: http://webserver/contao/

Code:

```
<alias "/<pfad>/contao_tuts">
RewriteBase /contao
```

---

dl1ely

Da wird bestimmt der Hund im Thema "Wie baue ich die Detail-URL zusammen" begraben sein. Wie sieht denn die "normale" URL der Meldeliste aus, und welche URL wird versucht aufzurufen, wenn Du den "Detail"-Link anklickst? Kannst den Domainnamen ja gerne weglassen.

Es kann gut sein, dass mein Ansatz für die Detail-URL nur bei bestimmten Randbedingungen funktioniert, dann müsste ich das nochmal nachbessern.

MBM

Hooks

> Zitat von **dl1ely**
> *Hallo!*
> *Da wird bestimmt der Hund im Thema "Wie baue ich die Detail-URL zusammen" begraben sein.*

Hunde begraben ? Mich würde ja interessieren unter welcher Konfiguration das bei Dir läuft.
Nun ja, ich habe jetzt auch eine Lösung entwickelt. Ich weiß zwar auch nicht ob das Best Practise ist und ob es nicht einen einfacheren Weg gibt, aber es funktioniert.

> Zitat von **dl1ely**
> *Wie sieht denn die "normale" URL der Meldeliste aus, und welche URL wird versucht aufzurufen, wenn Du den "Detail"-Link anklickst? Kannst den Domainnamen ja gerne weglassen.*

Meine Konfigurationsparameter

Im Backend ist URL's umschreiben auf aktiv gestellt.

Code:

```
Hauptlink: http://cms/contao/turnierpaareliste.html
Detaillink: http://cms/contao/turnierpaareliste/info/test_testin.htm

# htaccess
RewriteBase /contao
RewriteRule ^(.*)/info/(.*)$ $1.html?info=$2
RewriteRule .*\.html$ index.php [L]
```

> Zitat von **dl1ely**
> *Es kann gut sein, dass mein Ansatz für die Detail-URL nur bei bestimmten Randbedingungen funktioniert, dann müsste ich das nochmal nachbessern.*

Ich habe nun ein wenig experimentiert und zwei Lösungen gefunden.
Die erste war Quick & Dirty und ist **nicht update sicher** und die zweite könnte gehen.

Lösung 1: index.php ergänzen (Nicht zu empfehlen)

PHP-Code:

```
    public function run()
    {
        global $objPage;
        // Get page ID
        $pageId = $this->getPageIdFromUrl();
                // filter  alias
        if (stripos($pageId, '/info') >0 )
        {
            $pageId = substr($pageId,0,stripos($pageId, '/info'));
        }
}
```

Lösung 2: Hook

2.1 Ergänzung des TL-Hook array.

config.php - /system/modules/gw_turnierpaare/config/

PHP-Code:

```
$GLOBALS['TL_HOOKS']['getPageIdFromUrl'][] = array('gwHook', 'gwGetPageIdFromUrl
');
```

2.2 Hook Klassendatei erstellen.

gwHook.php - /system/modules/gw_turnierpaare/

PHP-Code:

```
class gwHook extends Controller
{
 public function  gwGetPageIdFromUrl($arrFragments)
 {
    // get alias (-url)
     $url = explode('/', $arrFragments[0]) ;
    // filter or cut, seo parameter and set only alias
     $arrFragments[0] = $url[0] ;
       return array_unique($arrFragments);
}
}
```

---

dl1ely

Hallo MBM,

sorry für die verspätete Antwort, aber ich war leider ziemlich beschäftigt. Ich kann dir auf Anhieb nicht sagen, warum das nicht so funktioniert wie bei mir.

In meiner .htaccess ist "RewriteBase /", da die Contao-Installation nicht in einem Unterverzeichnis liegt. Sonst habe ich auch "nur"

Code:

```
RewriteRule .*\.html$ index.php [L]
```

in der .htaccess .

Bei der Detail-URL schreibst du nur was von ".htm" am Ende der URL. Das ist aber sicherlich auch ".html", genau wie bei der Übersichts-URL, oder?

Bis auf die Rewritebase sehe ich erstmal keinen zwingenden Unterschied. Aber es wundert mich, dass es daran haken soll.

Dass du schon eine Menge gelernt hast zeigt ja dein Lösungsansatz mit dem Hook, das zeugt vom Lernerfolg ;-).