Robert Steindl

Know-how ist blau.

Extensions für TYPO3

So entwickeln Sie maßgeschneiderte TYPO3-Erweiterungen

- Die TYPO3-Programmierschnittstelle beherrschen
- Extensions sauber programmieren und der Community zur Verfügung stellen
- Das bringen TYPO3 5.0 und FLOW3

FRANZIS

Inhaltsübersicht

1 Vorbereitung

1.1 Einleitung

2 Planung der Erweiterung

2.1 Anforderungen sammeln

2.2 Umsetzung des Entwurfs in TYPO3

3 Struktur einer Extension

3.1 Kategorien von Extensions

3.2 Dateistruktur einer Extension

4 Der Extension Kickstarter

4.1 Aufbau des Kickstarters

4.2 Dateien des Kickstarters

5 Das Frontend-Plugin

5.1 Ausgangsbasis

5.2 Frontend-Funktionen

5.3 Plugin-Konfiguration

5.4 Lokalisierung

5.5 Caching in Plugins

5.6 Der Plugin-Code

6 Backend-Programmierung

6.1 Das Backend-Formular

6.2 Das Backend-Modul

7 Fertigstellen der Extension

7.1 Code Review

7.2 Dokumentation

7.3 Veröffentlichen der Erweiterung

8 Project Coding Guidelines

8.1 Project Coding Guidelines

9 Die TYPO3-API

9.1 Die Programmierschnittstelle von TYPO3

10 Ausblick

10.1 Einschätzung

10.2 MVC als Entwurfsmuster

10.3 FLOW3 und Extbase

10.4 Fluid

10.5 Kickstarter

10.6 Zusammenfassung

Stichwortverzeichnis

1 Vorbereitung

1.1 Einleitung

Dieses Buch ist eine praxisorientierte Hilfe für alle, die den Schritt vom reinen TYPO3-Anwender hin zum Entwickler gehen wollen.

Im Laufe dieses Buches wird der Werdegang einer TYPO3-Extension Schritt für Schritt nachvollzogen – vom Einrichten einer passenden Entwicklungsumgebung bis hin zur Veröffentlichung im TYPO3-Repository.

Basis dieses Buches ist TYPO3 in der Version 4.2.10.

Voraussetzungen

Folgende Voraussetzungen und Kenntnisse sollten Sie als Leser dieses Buches mitbringen:

Sicherer Umgang mit TYPO3

Sie sollten TYPO3 als Administrator kennen und verstehen. Grundwissen zum Thema Typoscript und TYPO3-Extensions sollte vorhanden sein.

Programmierkenntnisse

Die Programmiersprache für TYPO3 und seine Extensions ist PHP. Sie sollten daher solide Kenntnisse im Umgang mit PHP im Allgemeinen sowie mit Klassen und Objekten im Besonderen haben.

HTML-Kenntnisse

Die meisten Extensions erzeugen Ausgaben im Frontend. Sicherer Umgang mit HTML und CSS wird daher vorausgesetzt.

Einrichten von TYPO3 als Entwicklungsumgebung

Grundsätzlich lässt sich eine TYPO3-Extension mit nicht mehr als einem simplen Texteditor programmieren. Allerdings würde das nicht nur unnötige Zeit kosten, sondern auch viele Fehlerquellen öffnen. Eine sinnvoll eingerichtete TYPO3-Umgebung ist daher für die Extension-Entwicklung äußerst hilfreich. Im Folgenden werden einige Erweiterungen und Dokumentationen beschrieben, die Ihnen helfen, den Weg zur eigenen Extension zu ebnen.

Der Extension Kickstarter

Der Extension Kickstarter (*kickstarter*) ist der Ausgangspunkt fast jeder Erweiterung. Er hilft, ein sauberes Grundgerüst für eine Extension aufzubauen – allerdings auch nicht mehr. Es handelt sich nämlich nicht um einen Editor, mit dem sich eine Extension nach dem Erstellen dauerhaft weiter pflegen lässt.

Wir werden den Kickstarter im Verlauf der Entwicklung unserer Erweiterung noch genauer kennenlernen.

Der Extension Development Evaluator

Die zweite hilfreiche TYPO3-Extension ist der Extension Development Evaluator (*extdeveval*). Diese Erweiterung stellt eine Reihe von Funktionen zur Verfügung, um Ihre Erweiterung zu säubern und an die TYPO3-Richtlinien anzupassen. Außerdem stellt sie eine Dokumentation der TYPO3-Funktionen zur Verfügung.

Auch diese Erweiterung werden wir beim Aufbau unserer Extension einsetzen.

T3Dev

Die Erweiterung T3Dev (*t3dev*) stellt ebenfalls einen Zugang zur Dokumentation zur Verfügung. Darüber hinaus enthält sie ein Tool zum Erstellen der XML-Strukturen für eine Flexform.

Dokumentationen

Die wichtigsten Dokumentationen zu TYPO3 stehen im Repository als »Extensions« zur Verfügung, auch wenn sie keinerlei Code enthalten. Stattdessen bringen sie die Dokumentation in Form einer OpenOffice-Datei auf Ihren Rechner, sodass die Informationen auch ohne Internet-Verbindung jederzeit zur Verfügung stehen.

Die Dokumentationen lassen sich mithilfe des Extension Managers problemlos installieren, indem Sie nach »doc_« suchen. Die wichtigsten Dokumentationen sind:

Dokumentation	Extension Key	Beschreibung
TSconfig	`doc_core_tsconfig`	Dokumentation der Typoscript-Konfiguration für Page und User TSConfig.

Dokumentation	Extension Key	Beschreibung
TSref	`doc_core_tsref`	Die Typoscript-Referenz
TYPO3 Coding Guidelines	`doc_core_cgl`	Guidelines für die Code-Entwicklung von TYPO3-Extensions
TYPO3 Core API	`doc_core_api`	Dokumentation der Programmierschnittstellen (APIs) von TYPO3
TypoScript Syntax and In-depth Study	`doc_core_ts`	Einführung in Typoscript

Tabelle 1.1: TYPO3-Dokumentationen

2 Planung der Erweiterung

Am Anfang jedes Projekts steht eine sorgfältige Planung, und eine TYPO3-Extension macht hier keine Ausnahme – im Gegenteil. Da jede Erweiterung Bestandteil des umfangreichen und komplexen TYPO3-Systems wird, muss sie über die »normalen« Anforderungen an ein Software-Projekt hinaus auch noch die Regeln erfüllen, die TYPO3 vorgibt. Ohne diese Vorgaben wäre es kaum möglich, die Stabilität und Flexibilität von TYPO3 zu gewährleisten.

2.1 Anforderungen sammeln

Am Beginn jeder Planung steht das Sammeln der Anforderungen – noch unabhängig vom Weg der Implementierung. So banal das klingen mag, so häufig wird dieser wesentliche Schritt versäumt oder zumindest zu leichtfertig angegangen. Stattdessen entwickelt sich das endgültige Konzept oft erst bei der Implementierung bzw. der Vorstellung der ersten Funktionen. Allerdings bedeutet das meist unnötige Verzögerungen und führt zu nicht optimalen Ergebnissen, weil an bereits fertigen Funktionen nachträglich Anpassungen erfolgen müssen. Im schlimmsten Fall stellt sich heraus, dass die grundlegende Struktur falsch ist, was dann umfangreiche Umbauarbeit nach sich zieht.

Eine gute erste Planungsphase hilft auch, unterschiedliche Denkweisen auszugleichen. Einfaches Beispiel: Denkt ein Bankangestellter an eine Transaktion, dann meint er meistens den Transfer von Geld. Ein Programmierer hat dagegen bei diesem Begriff eher Datenbanken im Sinn. Es

lohnt sich daher, solche Missverständnisse von vornherein auszuschließen und eine gemeinsame Sicht der Dinge zu schaffen.

Das Pflichtenheft der Beispiel-Extension

Hier kommt nun das erste Mal unsere Beispiel-Extension ins Spiel. Im Laufe des Buches wollen wir einen Event Manager entwickeln, eine Erweiterung also, mit der sich Veranstaltungen verwalten lassen. Die Ausgangssituation soll folgendermaßen aussehen:

- Auf der Website sollen Veranstaltungen angekündigt werden, zu denen sich Besucher registrieren können.
- Eine Veranstaltung wird beschrieben mit Titel, Ort und Zeit. Außerdem soll es eine ausführliche Beschreibung geben sowie eine Agenda und einen Anreißtext, beispielsweise für die Homepage.
- Zu einer Veranstaltung gibt es Sponsoren. Auf der Seite einer Veranstaltung soll eine Liste der Sponsoren mit Name und Logo zu sehen sein, die jeweils mit einem kurzen Firmenprofil verlinkt werden.
- Es gibt Haupt- und Nebensponsoren.
- Die Registrierung der Besucher erfolgt per Online-Formular. Dabei ist zu berücksichtigen, dass eine Veranstaltung nur eine begrenzte Zahl von Plätzen hat.
- Jede Registrierung soll eine Mail an einen Mitarbeiter der Firma auslösen. Darüber hinaus soll in der Website-Verwaltung zu jeder Veranstaltung eine Teilnehmerliste abrufbar sein.

Als ergänzende Faktoren kommen dazu:

- Ein Sponsor kann mehrere Veranstaltungen bezuschussen. Das Firmenprofil soll in diesem Fall nicht jedes Mal neu erstellt werden.

- Besucher, die sich bereits einmal registriert haben, sollen ihre Daten nicht neu eingeben müssen.

Es gibt natürlich noch weitere naheliegende Funktionen, die zu unserer Erweiterung passen würden. Wir werden sie hier aber nicht umsetzen, um den Rahmen nicht zu sprengen. Denkbare zusätzliche Funktionen wären:

- Veranstaltungsreihen, also sich wiederholende Termine
- Online-Bezahlfunktionen
- Registrierte Besucher können sich nach der Veranstaltung online anmelden, um zusätzliche Daten abzurufen, etwa die Präsentationen.

Technische Anforderungen

Über das funktionelle Pflichtenheft hinaus gibt es einige technische Details, die häufig nicht zur Sprache kommen, aber auf jeden Fall berücksichtigt werden sollten. Dazu gehören:

- Einfache Anpassbarkeit des Layouts
- Einfache Konfiguration im Backend
- Unterstützung für mehrsprachige Websites
- Dokumentation

Insbesondere der letzte Punkt gilt als undankbare Aufgabe und wird oft nur kurz oder lieblos erfüllt. Erfahrene Auftraggeber bestehen auf einer guten Dokumentation, doch auch aus Sicht des Programmierers ist sie wichtig. Denn nahezu jedes Projekt holt die Entwickler irgendwann wieder ein, beispielsweise weil neue Funktionen implementiert werden sollen. Und auch wenn man den Code selbst geschrieben hat – ein Jahr später wirkt er so fremd, als hätte man ihn nie gesehen.

Die Datenbankstruktur

Die obigen Anforderungen müssen nun in eine Datenbankstruktur umgesetzt werden – zunächst noch unabhängig von TYPO3. Folgende Tabellen sollen die Grundlage bilden:

Veranstaltungen

Feld	Typ	Beschreibung
`title`	`varchar(100)`	Titel der Veranstaltung
`location`	`varchar(100)`	Ort der Veranstaltung
`date`	`date`	Termin
`teaser`	`tinytext`	Anreißtext
`description`	`text`	Beschreibung
`agenda`	`text`	Agenda
`seats`	`int`	Anzahl Plätze

Tabelle 2.1: Datenbanktabelle für die Veranstaltungen

Sponsoren

Feld	Typ	Beschreibung
`name`	`varchar(100)`	Name des Sponsors
`logo`	`tinytext`	Dateiname des Logos
`url`	`tinytext`	Link zur Website
`profile`	`text`	Firmenprofil

Tabelle 2.2: Datenbanktabelle für die Sponsoren

Teilnehmer

Feld	Typ	Beschreibung
`name`	`tinytext`	Name des Teilnehmers
`email`	`varchar(200)`	E-Mail-Adresse
`street`	`varchar(150)`	Straße / Hausnummer
`zipcode`	`varchar(10)`	Postleitzahl
`city`	`varchar(100)`	Ort
`country`	`varchar(100)`	Land

Tabelle 2.3: Datenbanktabelle für die Teilnehmer

Zu den reinen Datentabellen kommen noch zwei Kreuztabellen, in denen gespeichert wird, welcher Sponsor zu welcher Veranstaltung gehört und welcher Teilnehmer sich wo registriert hat. Kreuztabellen wählen wir deshalb, weil zwischen Veranstaltungen und Sponsoren eine m-zu-n-Beziehung besteht, d. h. eine Veranstaltung kann mehrere Sponsoren haben, ein Sponsor mehrere Veranstaltungen fördern. Das Gleiche gilt für Veranstaltungen und Teilnehmer. Gemäß den Normalisierungsregeln für Datenbanken sind in solchen Fällen Kreuztabellen das Mittel der Wahl.

Eine grafische Darstellung des Datenbankmodells sieht dann etwa so aus:

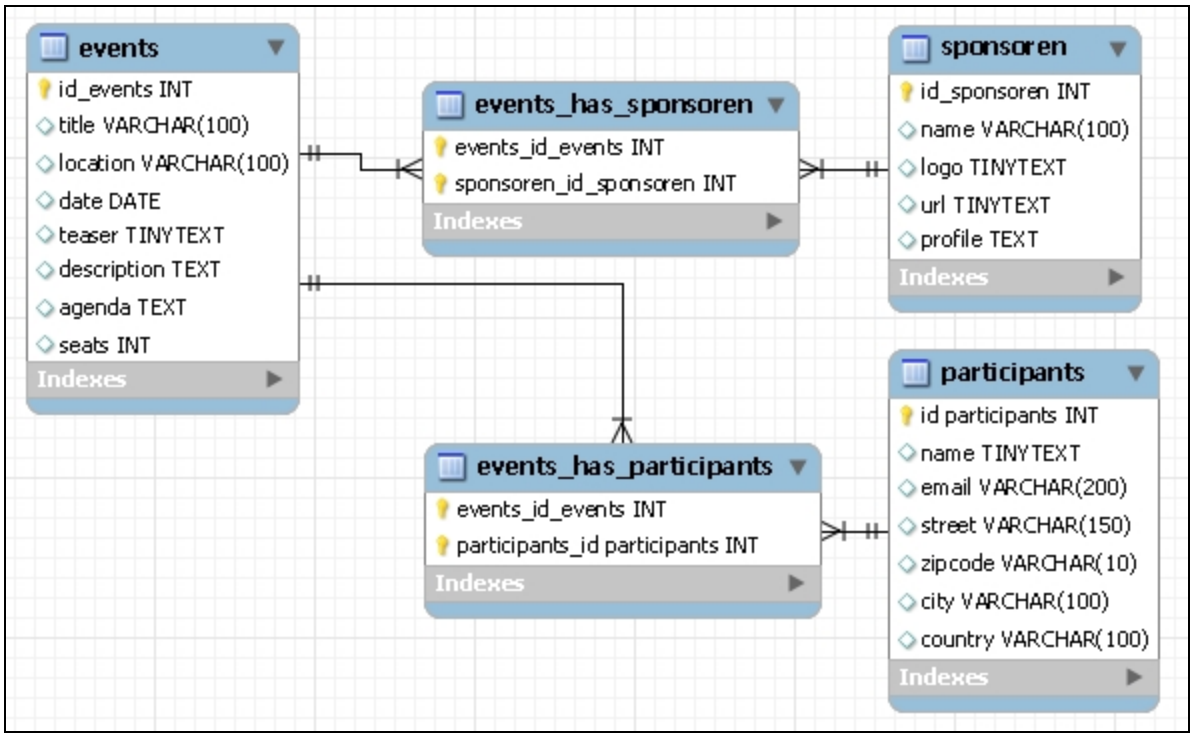

Bild 2.1 Der Entwurf für das Datenbankmodell

2.2 Umsetzung des Entwurfs in TYPO3

Nachdem die Datenstruktur feststeht, soll nun die Anpassung an die TYPO3-Umgebung erfolgen. Das beginnt mit der Suche nach dem richtigen Namen.

Der Extension Key

Erweiterungen werden von TYPO3 nicht zuletzt anhand des Namens verwaltet. Daher muss jede TYPO3-Extension eine eindeutige Bezeichnung erhalten, den *Extension Key* . Bei der Benennung von Extensions sind daher einige Regeln einzuhalten.

Für Extensions, die nur für eine Installation verwendet werden sollen, muss die Eindeutigkeit des Namens nur innerhalb des Systems hergestellt sein. Wenn Sie sicher sind, dass eine Erweiterung nur für ein bestimmtes Projekt von Interesse ist und nicht allgemein verfügbar sein soll, nehmen Sie daher einfach einen Namen Ihrer Wahl und setzen das Präfix `user_` davor. Damit ist klar, dass diese Extension nicht aus dem öffentlichen Repository für Erweiterungen stammen kann. Vom Extension Key abgeleitete Bezeichner für Tabellen usw. ergeben sich dann wie in der folgenden Tabelle angegeben:

	Regel	**Beispiel**
Extension Key	Frei wählbar, aber mit Präfix `user_`	`user_myextension`
Datenbank-Tabellen und -Felder	mit dem Präfix der Erweiterung	`user_myextension_images` `user_myextension_categories`
Backend-Modul	enthält keine Unterstriche, beginnt mit `u`	`umyextension`
PHP-Klassen	Klassennamen werden gebildet wie Datenbank-Tabellen und Felder, die Dateien dazu erhalten den Vorsatz `class.`	`class.myextension_pi1.php`

Tabelle 2.4: User Extension Keys und abgeleitete Bezeichner

Soll eine Erweiterung dagegen der TYPO3-Gemeinde zur Verfügung gestellt, sprich ins TYPO3 Extension Repository (TER) geladen werden, muss der Extension Key weltweit eindeutig sein.

Um dies sicherzustellen, werden Extension Keys im TER registriert. Dabei werden neben der Eindeutigkeit auch noch einige formale Kriterien für den Aufbau des Keys geprüft.

Diese sind:

- Der Key darf nur aus Buchstaben, Ziffern und dem Unterstrich (_) bestehen.
- Der Key darf nicht mit einem reservierten Präfix wie `tx`, `u`, `tt_` oder `sys_` beginnen – auch wenn im Repository aus historischen Gründen noch Extensions wie `tt_news` oder `tt_address` zu finden sind.
- Da aus dem Key Bezeichner für Module abgeleitet werden, indem die Unterstriche entfernt werden, muss der Key auch in diesem Fall eindeutig bleiben. Unter anderem deswegen wird für neue Extensions empfohlen, von vornherein keine Unterstriche im Extension Key zu verwenden.
- Da aus dem Extension Key auch der Verzeichnisname in TYPO3 abgeleitet wird, sollten keine Großbuchstaben verwendet werden. Dies würde je nach Betriebssystem zu Komplikationen führen.
- Damit ergeben sich folgende Bezeichner in der Extension:

	Regel	**Beispiel**
Extension Key	Vom TER zugewiesen bzw. akzeptiert	`myextension`
Datenbank-Tabellen und -Felder	Name der Erweiterung, versehen mit dem Präfix `tx_`	`tx_myextension_images` `tx_myextension_categories`
Backend-Modul	enthält keine Unterstriche, beginnt mit `tx`	`txmyextension`
PHP-Klassen	Klassennamen werden gebildet wie Datenbank-Tabellen und Felder, die Dateien dazu erhalten den Vorsatz `class.`	`class.tx_myextension_pi1.php`

Tabelle 2.5: TER Extension Keys und abgeleitete Bezeichner

Neben den formalen Kriterien gibt es auch einige »weiche« Anforderungen. So soll der Key den Zweck der Extension beschreiben, gleichzeitig aber nicht zu lang sein. Im Repository finden Sie viele Extensions, die mit einem Kürzel für den Autor beginnen. Das wird nicht empfohlen, lässt sich aber manchmal nicht vermeiden, wenn man eine Erweiterung in einer Kategorie schreibt, in der es schon einige andere gibt. Besser ist es aber, wenn Ihre Extension eine Eigenart aufweist, die im Extension Key benutzt werden kann.

Für unseren Event Manager wären also gute Extension Keys:

- `eventmanager`
- `eventmgr`
- `eventmanagement`

Nicht zu empfehlen wären dagegen:

- `emgr` (zu kurz)
- `my_manager` (keine Beschreibung, Unterstrich)
- `event_and_sponsor_management` (lang, viele Unterstriche)

Veröffentlichen von Extensions

Selbst wenn beim Schreiben einer Erweiterung zunächst nicht geplant ist, die Extension im TER zu veröffentlichen, sollten Sie diesen Schritt auf jeden Fall aus zwei Gründen mit einplanen:

Zum einen ist es mit viel Aufwand verbunden, den Namen einer Extension nachträglich zu ändern. Der Extension Key wird für Verzeichnis- und Dateinamen ebenso benutzt wie für Datenbanktabellen. Und selbst wenn die Erweiterung zu speziell ist, um für andere von Nutzen zu sein: Sie können

sie als privat oder als Test markieren und sie so im TER ablegen, ohne den Zugriff für andere freizuschalten.

Zum anderen lebt TYPO3 nicht zuletzt von der Vielzahl an Erweiterungen. Wenn eine Extension also von allgemeinem Interesse sein könnte, sollte sie auch im TER vorhanden sein.

Alle Erweiterungen, die Sie schreiben, unterliegen wie TYPO3 selbst automatisch der GPL (GNU General Public License). Das zwingt Sie zwar nicht zur Veröffentlichung, doch sollten Sie sich im Klaren sein, dass der Quellcode Ihrer Extension von jedem benutzt und weiterbearbeitet werden darf, der Zugriff darauf hat.

Um eine Extension im Repository veröffentlichen zu können, benötigen Sie einen Login auf der Website *TYPO3.org*. Hier können Sie ein persönliches Profil hinterlegen, sind aber nicht dazu gezwungen. Nach der Anmeldung können Sie auf der Seite https://TYPO3.org/extensions/extension-keys/ einen Extension Key registrieren.

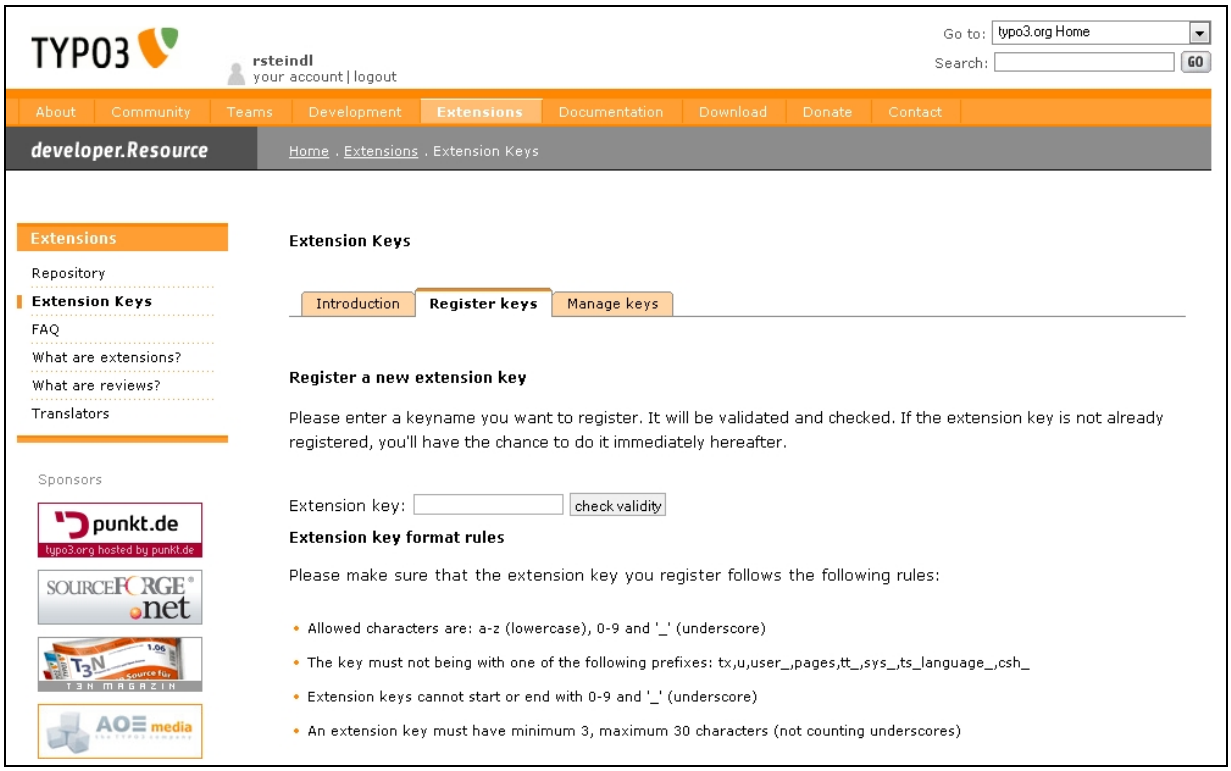

Bild 2.2 Das Registrieren eines Extension Keys geht schnell und einfach

Das Repository prüft, ob der Key die Vorgaben erfüllt und ob er frei ist. Falls ja, wird er eingetragen, andernfalls erhalten Sie eine Fehlermeldung.

Tipp: Wenn Sie einen Extension Key reservieren wollen, recherchieren Sie vorher im Repository, welche Namen bereits benutzt werden. Allerdings kann es passieren, dass ein Extension Key reserviert ist, obwohl keine Extension dazu im Repository angeboten wird.

Einmal reservierte Extension Keys können Sie über die oben genannte Website im Abschnitt »Manage keys« auch verwalten. Sie sehen dort, wann der letzte Upload der Extension erfolgt ist. Auch eine Löschfunktion für Ihre Extension Keys ist hier zu finden. Allerdings lässt sich ein

Key nur dann komplett löschen, wenn er noch nie benutzt wurde, d. h. wenn noch keine Extension mit diesem Key ins Repository eingestellt wurde.

Wenn Sie eine Extension nicht mehr weiter verwalten wollen, aber ein anderer User sich dazu bereit erklärt hat, können Sie die Rechte an einem Extension Key auch einem anderen User übertragen.

Von den oben genannten brauchbaren Keys waren bereits einige vergeben, für die hier vorgestellte Erweiterung wurde daher der Key `eventmanagement` reserviert. Nach den Empfehlungen von TYPO3 ist er mit 15 Zeichen etwas zu lang, doch abgesehen davon erfüllt er alle Anforderungen.

Anpassung der Datenbankstruktur

Wir wissen bereits, dass unsere Extension eine Datenbank benötigt, und haben die gewünschte Struktur bereits bestimmt. Da aber diese Struktur in die bereits bestehende von TYPO3 integriert werden soll, müssen einige Anforderungen erfüllt werden.

TYPO3-eigene Felder

Jede TYPO3-Tabelle hat einige Felder, die für die Verwaltung durch TYPO3 wichtig sind. Zwingend sind hierbei zunächst nur `uid` und `pid`.

`uid` (Unique ID) ist die eindeutige Kennzeichnung jedes Eintrags. Dabei handelt es sich um ein sogenanntes Autoincrement-Feld, das automatisch hochzählt.

`pid` (Page ID) bestimmt, auf welcher Seite des TYPO3-Baums der Inhalt platziert wird. Eine 0 steht für den Beginn des

Baums, den `root level`.

Wie wir später sehen werden, lassen sich beim Anlegen der Tabelle im Kickstarter weitere Optionen aktivieren, die in folgenden Feldern verwaltet werden:

Feld	Typ	Beschreibung
`tstamp`	Integer	Datum und Uhrzeit der letzten Änderung des Datensatzes als Unix-Timestamp.
`crdate`	Integer	Datum und Uhrzeit der Erstellung des Datensatzes als Unix-Timestamp.
`cruser_id`	Integer	Die ID des Backend-Users (entspricht dem Feld `uid` in der Tabelle `beuser`), der den Eintrag erstellt hat. Wurde ein Eintrag über das Frontend angelegt, steht hier meist eine 0.
`sys_language_uid`	Integer	Die `uid` der Sprache, in der der Eintrag angelegt wurde. Dieses Feld ist zusammen mit `l10_parent` und `l10_diffsource` wichtig für die Lokalisierung von Datensätzen.
`l10n_parent`	Integer	Die `uid` des Datensatzes, der durch diesen Eintrag lokalisiert wird.
`l10n_diffsource`	Medium Text	Die Originaldaten, die die Grundlage der Lokalisierung bilden.
`deleted`	Tiny Integer	Flag, das zeigt, ob der Datensatz gelöscht ist.
`hidden`	Tiny Integer	Flag, das zeigt, ob der Datensatz auf »verborgen« gesetzt ist.
`starttime`	Integer	Zeitpunkt, ab dem der Datensatz online gezeigt wird, als Unix-Timestamp.
`endtime`	Integer	Zeitpunkt, ab dem der Datensatz nicht mehr gezeigt wird, als Unix-Timestamp.
`fe_group`	Integer	`uid` bzw. Liste von `uid`s der Frontend-Usergruppen, die auf diesen Eintrag zugreifen dürfen.

Tabelle 2.6: Zusätzliche Felder in TYPO3

Tipp: In einigen (älteren) Extensions finden Sie statt der Felder `l10n_parent` und `l10n_diffsource` die Felder `l18n_parent` und `l18n_diffsource`. Das hat historische Gründe – und ist eigentlich ein Bug. Denn `l10n` ist die übliche Abkürzung für das Wort *Localisation* (weil zwischen »l« und »n« 10 weitere Buchstaben kommen). Das Gegenstück dazu ist *Internationalisation*, was abgekürzt `i18n` ergibt.
Der Prozess der Übersetzung, der mithilfe der Datenbankfelder realisiert wird, gehört allgemein zur Lokalisierung, darum ist `l10n...` die korrekte Bezeichnung. Als Internationalisierung wird eher die Vorbereitung und grundlegende Struktur bezeichnet, auf der die Lokalisierung aufsetzt.

Wenn eine Extension das Workspace-Konzept bzw. die Versionierung von Datensätzen unterstützt, kommen das Feld `t3_origuid` sowie eine Reihe weiterer Felder hinzu, die alle mit `t3ver_` beginnen.

Die Felder unserer Beispieldatenbank kommen mit den genannten Feldern nicht in einen Namenskonflikt. Wäre das der Fall, sollten Sie die Benennung der Felder entsprechend ändern. Theoretisch lassen sich auch die Namen der TYPO3-Felder ändern, doch das erfordert zum einen Handarbeit und erschwert zum anderen das Verständnis der Extension.

Datenbankrelationen in TYPO3

TYPO3 kennt unterschiedliche Methoden, um Beziehungen zwischen zwei Tabellen herzustellen.

Die einfachste Methode (und die erste, die in TYPO3 vorhanden war) besteht darin, in einem Feld einer Tabelle

eine kommaseparierte Liste von `uid`-Werten der zweiten Tabelle zu speichern. In manchen Fällen wird statt der `uid` als Zahl ein String aus Tabellenname und `uid` in der Form `tabelle_uid` gespeichert. Dies sorgt dafür, dass wirklich jedes Inhaltselement ansprechbar wird.

Der Vorteil dieser Methode ist die relativ einfache Verständlichkeit und die Verarbeitungsgeschwindigkeit.

Aus Sicht der Datenbanktheorie ist eine solche Kopplung allerdings für die m-zu-n-Relationen, die wir in unserer Extension verwenden wollen, nicht korrekt. Stattdessen fordern die Normalisierungsregeln in solchen Fällen eine Kreuztabelle. Eine solche Tabelle nimmt Paarungen von ID-Werten auf.

Das folgende Beispiel zeigt den prinzipiellen Aufbau einer Datenbankstruktur mit Kreuztabelle:

Unsere Event-Tabelle soll folgende Einträge haben:

uid	name
1	TYPO3-Seminar
2	PHP-Kurs
3	Datenbankentwicklung mit MySQL

Tabelle 2.7: Beispieldaten Events

Die Liste der Sponsoren sieht so aus:

uid	name
1	IBM
2	MySQL
3	Oracle
4	Franzis-Verlag

Tabelle 2.8: Beispieldaten Sponsoren

Wenn nun (nur als Beispiel) IBM das PHP-Seminar sponsort, MySQL das Datenbank-Seminar und der Franzis-Verlag bei TYPO3 und PHP als Sponsor auftritt, sieht die Kreuztabelle wie folgt aus:

event_uid	sponsor_uid
2	1
2	4
3	2
1	4

Tabelle 2.9: Beispiel Kreuztabelle

Eine solche Konstruktion ist auf den ersten Blick komplexer und erfordert tatsächlich auch einen sorgfältigeren Umgang mit den Daten. Im Gegenzug erhält man damit aber eine robustere Struktur mit weniger Redundanzen. Und für Datenbank-Abfragen kann sozusagen jede der beiden Spalten die Führung übernehmen, d. h. wenn man die Sponsoren eines Events wissen will, sucht man alle Einträge mit einer bestimmten `event_uid`. Für die Frage, welche Events ein Sponsor bezuschusst, wird die `sponsor_uid` festgelegt.

In TYPO3 ist dieses Konzept ebenfalls implementiert und kann im Kickstarter beim Anlegen einer Relation aktiviert werden. Die Kreuztabelle wird dann automatisch angelegt. Die Besonderheiten der M-M-Relation (Many-to-Many) in TYPO3 werden wir beim Aufbau der Extension genauer behandeln.

TYPO3 kennt auch noch eine dritte Methode, Beziehungen zwischen zwei Datenbank-Tabellen herzustellen – sie lautet kurz IRRE. Die Abkürzung steht für Inline Relational Record

Editing und wurde mit der Absicht eingeführt, das Editieren verknüpfter Datensätze zu erleichtern. Wenn beispielsweise eine Person mehrere Adressen hat, lässt sich mit IRRE ein Backend-Formular gestalten, das sowohl die Personendaten als auch die Adressen bearbeitbar macht, ohne auf neue Seiten zu wechseln. Das Anlegen und Bearbeiten neuer Adressen wird dabei mittels AJAX gesteuert. Für IRRE gibt es im TYPO3 Extension Repository das *Tutorial for Inline Relational Record Editing IRRE* (*irre_tutorial*), das mit Beispieldaten das Konzept näher erläutert.

Für unseren Event-Manager werden wir die Event- und die Sponsoren-Tabelle mit einer Kreuztabelle realisieren, die Kopplung der Teilnehmer zu den Events dagegen mit der klassischen Methode, um beides vorzustellen.

Verwendung vorhandener Tabellen

Die Anpassung der Datenbankstruktur an TYPO3 erfordert aber noch eine weitere Überlegung: Die geplante Extension soll die Adressen der Teilnehmer speichern, wofür in unserem Entwurf eine entsprechende Tabelle definiert wurde. Allerdings gibt es in TYPO3 bereits Möglichkeiten, Teilnehmer bzw. Adressen zu speichern – zum einen die Extension `tt_address`, zum anderen die Tabelle für die Frontend-User (`fe_users`).

Tipp: Lassen sich Funktionen einer Erweiterung unter Zuhilfenahme der vorhandenen Tabellen durchführen, so ist das meist der bessere Weg, weil sich ähnliche Daten nicht verteilt über das ganze System finden. Im schlimmsten Fall könnte ein angemeldeter User über `fe_users` verwaltet werden, seine Adresse in `tt_address` zu

finden sein und seine Reservierungen in einer dritten Tabelle.

`tt_address` wird von einigen weiteren Extensions unterstützt, sodass man damit beispielsweise eine gemeinsame Adressdatenbank für Mailings und Events aufbauen kann. Da sich Adressen in hierarchisch gegliederte Gruppen einteilen lassen, ist eine saubere Trennung nach Funktion problemlos möglich.

Die Teilnehmer als Frontend-User zu registrieren, ist vor allem dann sinnvoll, wenn mit der Registrierung auch Funktionen der Website freigeschaltet werden. So könnte ein Teilnehmer Zugang zu den Präsentationen einer Veranstaltung erhalten oder sich bei der Buchung weiterer Seminare mit seinem Login anmelden und sich so die erneute Eingabe aller Daten ersparen.

Allerdings ist die Struktur der `fe_users`-Tabelle für unsere Zwecke nicht optimal, vor allem für deutschsprachige Webseiten. Es fehlen Felder für die Anrede und akademische Titel, Name und Adresse sind nicht aufgeteilt in Vor- und Nachname bzw. Straße und Hausnummer. Außerdem wollen wir speichern, zu welchen Veranstaltungen sich jemand angemeldet hat.

Trotzdem lässt sich `fe_users` für unsere Zwecke verwenden. In TYPO3 ist es kein Problem für eine Extension, bestehende Tabellen zu erweitern. Unser Event Manager wird daher einfach die `fe_users`-Tabelle um die nötigen Felder ergänzen.

3 Struktur einer Extension

Nachdem die Planung abgeschlossen ist und die grundsätzliche Datenstruktur feststeht, müssen wir im nächsten Schritt festlegen, wie die gewünschten Funktionen der Erweiterung in TYPO3 realisiert werden können. Um diese Entscheidung zu treffen, sollen hier die verschiedenen möglichen Bestandteile einer Extension vorgestellt werden.

3.1 Kategorien von Extensions

Im Extension Manager von TYPO3 finden wir 10 verschiedene Kategorien:

- Backend
- Backend Modules
- Frontend
- Frontend Plugins
- Services
- Documentation
- Examples
- Templates
- Miscellaneous

- Kategorie []

Bild 3.1 Die Kategorien im Extension Manager

Backend

Die Kategorie `Backend` umfasst Erweiterungen, die die Funktion des TYPO3-Backends erweitern, ohne einen eigenen Eintrag in der Modulliste zu benötigen. Beispiele hierfür sind der Rich Text Editor `rtehtmlarea` oder `cms` , das eigentliche Content Management von TYPO3.

Backend Modules

In der Kategorie `Backend Modules` finden sich Module, also Extensions, die die Backend-Funktionalität von TYPO3 erweitern. Sie sind typischerweise links im Modulbaum mit Einträgen gelistet. Beispiele hierfür sind die Grundmodule von TYPO3 oder Erweiterungen wie Full Backup (`w4x_backup`).

Frontend

Unter `Frontend` finden sich Erweiterungen, die Funktionalität für das Frontend bereitstellen, aber nicht notwendigerweise selbst HTML-Ausgaben erzeugen. Vertreter dieser Kategorie sind etwa `css_styled_content` oder `RealUrl` .

Frontend Plugins

`Frontend Plugins` erzeugen Ausgaben auf der Website. Sie stellen neue Inhaltsobjekte zur Verfügung oder erweitern bestehende. Beispiele hierfür sind Erweiterungen wie `tt_news` oder `felogin` . Dieser Erweiterungstyp ist vermutlich derjenige, der am häufigsten für selbstgeschriebene Extensions benutzt wird.

Services

Services sind spezielle Extensions, die sozusagen Dienstleistungen für andere Erweiterungen bereitstellen, beispielsweise die Authentifizierung von Usern. In einem TYPO3-System kann eine Service-Funktion mehrfach angeboten werden, etwa einmal durch Aufruf eines externen Programms und einmal durch ein PHP-Skript. Welche Funktion genutzt wird, kann beispielsweise vom Betriebssystem abhängen.

Tipp: Services sollten nicht mit Frontend- oder Backend-Funktionen in eine Extension gepackt werden. Lediglich für die Funktion notwendige Tabellen oder Tabellenfelder werden Bestandteil der Service-Erweiterung.