TYPOlight-Entwicklerworkshop

Ziel des Workshops war es, allgemeine Programmierungsfragen zu diskutieren, die sich nicht nur im Zusammenhang mit TYPOlight, sondern bei allen Webapplikationen stellen.

1. Best practice für "Speaking-URLs“

Wie generiert man suchmaschinenfreundliche URLs, die nicht zu tief verschachtelt sind, und wie verhindert man, dass Google dieselbe Seite mehrfach indiziert (Duplicate Content) wenn sich z.B. nur das Datum des Mini-Kalenders ändert?

Verwendung eines Suffix

Die Verwendung eines .html-Suffixes reduziert die Verschachtelungstiefe, was sich positiv auf den Pagerank auswirkt.

http://domain.de/alias/key/value ← gesehen als index.html im Ordner value (4. Level)
http://domain.de/alias/key/value.html ← gesehen als value.html im Ordner key (3. Level)

Verwendung von *key:value*-Paaren

Die Verwendung von *key:value*-Paaren anstatt *key/value*-Paaren reduziert die Verschachtelungstiefe und ermöglicht das Hinzufügen von Parametern, ohne auf deren Reihenfolge achten zu müssen.

http://domain.de/alias/key/value.html ← gesehen als value.html im Ordner key (3. Level)
http://domain.de/alias/key:value.html ← gesehen als key:value.html im Ordner alias (2. Level)

2. Best practice für mehrsprachige Webseiten

Wie fügt man die Sprachinformation in die URL ein, so dass Google alle Sprache indizieren kann und man trotzdem flexible Aliase oder sogar denselben Alias für eine Seite in mehreren Sprachen verwenden kann?

Es gibt keine allgemeingültige Lösung

- Sich auf die Browsersprache zu verlassen würde bedeuten, dass Google nichts indiziert, das nicht auf Englisch ist
- Ein Aliasprefix (z.B. /en/alias.html) erhöht die Verschachtelungstiefe
- Ein URL-Suffix (z.B. alias.en.html) kann problematisch sein, wenn bestehende URLs erhalten werden sollen

Mögliche Lösung

Man könnte die Position des Sprachtags in der URL auswählbar machen, so dass es entweder vor dem Alias, vor dem URL-Suffix oder überhaupt nicht hinzugefügt wird. Dabei sollte das „ein Alias pro Sprache“-Prinzip eingehalten werden, wobei derselbe Alias mehrfach verwendet werden kann, wenn die Sprachinformation in der URL vorhanden ist.

3. InnoDB vs. MyISAM

Um hierarchische Kategorien bzw. Strukturen abzubilden, sollten „Nested Sets" verwendet werden. Da diese während des Abspeicherns in der Datenbank sehr fehleranfällig sind, sollten Transaktionen eingesetzt werden, um die referentielle Integrität sicherzustellen. MyISAM-Tabellen (der Standard-Tabellentyp in MySQL) unterstützt Transaktionen jedoch nicht vor Version 5.3.

Können wir InnoDB voraussetzen?

- Nicht alle Provider unterstützen InnoDB-Tabellen
- Es ist nicht sicher, dass InnoDB ein Teil des MySQL-Projektes bleibt

Mögliche Lösung

- Transaktionen und InnoDB werden verwendet wenn vorhanden
- Fallback-Lösung für MyISAM-Tabellen: keine Transaktionen verwenden
- Driver-Methoden für spezifische Funktionen wie z.B. die MySQL-Volltextsuche

4. Meta-Informationen für Bilder und Dateien

Wie speichert man Meta-Informationen für Bilder und Dateien in der Datenbank, ohne das Bild selbst in der Datenbank abzulegen? Wie stellt man die referentielle Integrität sicher, wenn ein Anwender eine Datei via FTP umbenennt oder verschiebt?

Mögliche Lösung

Mit *mdf5_file()* lässt sich ein eindeutiger Hashwert erstellen, der anstatt der Datei in der Datenbank gespeichert werden kann. Ein automatischer Wartungsjob müsste das Dateisystem regelmäßig nach verschobenen oder umbenannten (eventuell auch gelöschten) Dateien durchsuchen und die Datenbank entsprechend aktualisieren.

Saved by the bell

Leider war die Zeit ziemlich schnell vorbei, so dass wir nicht alle Themen auf der Liste diskutieren konnten. Folgende Punkte blieben offen:

- Wie speichert man wiederkehrende Events in der Datenbank?
- MooTools vs. jQuery