

Bericht: Update der Website kh-berlin.de von Typo3 6.2 auf 7.6 und von PHP 5.5 auf 7.2

Oktober 2019 bis Januar 2020

Ausgangssituation

a) Typo3

- Typo3 6.2
- Das letzte Update wurde 2014 (vor 5 Jahren) durchgeführt
- Es gibt eine mächtige Extension "khberlin", die die spezifischen Funktionalitäten und die meisten Anpassungen und das Design darstellt.
- zwei weitere Extensions, deren Funktion unklar ist
- Die Entwickler der Extension sind nicht mehr erreichbar
- Die Grundfunktionen der Extension geht noch auf 2012 zurück
- Es gibt einige bekannte Bugs, die seit Jahren nicht behoben wurden

b) Daten

- Hochgeladene statische Dateien: 62GB
- MySQL-Datenbank: Größe schwankt zwischen 580 und 1100MB

c) Entwicklungsumgebung

- Gehostet ist die Seite bei einem klassischen Provider für PHP und statischen Content
- Deploys gibt es in dem Sinne gar nicht. Änderungen werden direkt im live code durchgeführt
- Es gibt kein Repository

Schritt 1: Website für Entwicklung handhabbar machen

Ziel: Es soll möglich sein, einfach ein Testsystem aufzusetzen, dort Änderungen vorzunehmen, diese zu versionieren und automatisiert zu deployen

- Tools: Docker, GIT, composer, bash

a) Docker

- Entwicklung einer Docker-Umgebung, die die Versionen von PHP, Apache und MySQL des Servers abbildet und sich verändern lässt. Grundlage: <https://github.com/webdevops/TYPO3-docker-boilerplate> (thanks a lot)

b) bash

- Script zum Download von Daten & Datenbank
- Scripte zum Aufsetzen von Docker, composer, etc

- Script zum Import von Datenbanken in MySQL auf Docker

c) GIT & composer

- Anlegen eine Monorepos, das Typo3 enthält, die Extensions und die Entwicklungsumgebung, also Dockerfiles & scripts
- Typo3 wird über composer installiert
- Anlegen von Dokumentationen (README.md)

Ergebnis: In wenigen Schritten und automatisiert lässt sich die Website auf lokalen Rechnern installieren:

1. \$ git clone ...
2. \$ bin/download-db && bin/download-data
3. \$ docker-compose up
4. \$ make composer install
5. \$ make mysql-import-latest
6. Fertig. Seite läuft <http://localhost:8000>

Dabei waren noch einige Performance-Optimierungen nötig. Den entscheidenden Schub hat der Ausschluss von typo3temp aus den gespiegelten Pfaden gebracht. Dort wird offenbar viel geschrieben und gelesen.

Schritt 2: Automatisierte Tests

Um die Funktionen der Seite künftig und während der kommenden Umbauten automatisiert testen zu können, ist ein Test-Framework nützlich. Um in der PHP-Welt zu bleiben, wähle ich

- phpunit 7
- php-webdriver für selenium von facebook
- selenium-cluster mit docker-compose

Leider ist hier vieles tückisch und schlecht dokumentiert. Am Ende aber gibt es aber Tests, die automatisch User einloggen, Projekte anlegen, Bilder hochladen, Seiten aufrufen und das alles mit Screenshots dokumentieren. Das ganze läuft parallel in 8 Containern auf Firefox und Chrome und ist daher auch recht fix.

Diese Funktionen werden sich wohl erst mittelfristig so recht auszahlen.

Schritt 3: Upgrade von Typo3 und PHP

Dieser Schritt erfordert viel Kleinarbeit, Recherche, Kenntnisse in Typo3 und PHP

In vielen Dateien müssen Pfade zu Modulen eingefügt und geändert werden. Dafür waren Reguläre Ausdrücke nützlich, die dann aber manchmal auch zu viel ersetzen.

Einige Extensions müssen entfernt werden, andere kommen hinzu. An vielen Stellen muss die Extension khberlin angefasst werden. Eine der ominösen Extensions stellt sich als überflüssig heraus.

Datenbankabfragen ändern sich, veraltete Funktionen müssen ersetzt werden, veraltete Techniken werden aufgepeppt.

Die Umstellung auf PHP7.2 klappt hingegen erstaunlich gut. Insgesamt ist der Prozess aber sehr langwierig. Dabei muss immer wieder getestet und geprüft werden.

Am Schluss gibt es eine Version, in der keine Bugs ersichtlich sind und fast alles genau so wie vorher aussieht.

Durch weitere Optimierungsschritte lässt sich die Datenbank auf 280MB zusammenschrumpfen, was das Testen schon um einiges erleichtert und beschleunigt.

Schritt 3: Automatisierte und sichere Deployments

Ziele:

- Deployments sollen ohne riskante Handarbeit laufen
- Es soll ein Testsystem geben
- Es wird aus dem Repository durch "git push" deployed
- Versionen von PHP, Typo3 etc. können im Repository eingestellt werden

Im Prinzip existieren diese ganzen Funktionen schon aus Schritt 1. in der Docker-Umgebung.

Allerdings sollen die Seite sie erstmal auf dem vorhandenen Centos-Server laufen.

Die Lösung besteht aus Scripten, die als Git-Hooks auf einem Bare Repository auf dem Zielsystem ausgeführt werden. Die Schritte sind:

- git checkout
- composer install
- Linken der Verzeichnisse für fileadmin und uploads
- Erzeugen von AdditionalConfiguration.php
- Verschieben ins Webroot

Das ganze wird flexibel über eine Konfigurationsdatei konfiguriert, in der Datenbankconfig, Verzeichnispfade usw. stehen. Davon gibt es zwei Stück: Live und Stage.

Live und Stage können unterschiedliche PHP-Versionen haben, was über .user.ini gesteuert wird.

Schritt 4: Feedback und Umschalten

Nachdem die neue Version auf dem Testsystem eine Weile lief, begutachtet wurde, und keine weiteren Probleme gefunden wurden, ist es Zeit für die Umstellung.

Da bei der Umstellung die nun aktuellen Daten migriert werden müssen, stand am Ende von Schritt 2 eine Anleitung, mit der die komplette Seite und Datenbank Schritt für Schritt von 6.2 auf 7.8 migriert werden kann. Dies habe ich schon geübt.

Jetzt muss nur noch der Wartungsmodus angeschaltet, die Daten kopiert und die Anleitung durchlaufen werden.

Danach kommt das nächste Update auf Version 8.