Einführung

in die Web-Programmierung (mit PHP/MySQL)

Vorstellungsrunde - Dozenten

FABIO DUO

Eidg. dipl. Wirtschaftsinformatiker FH

- Managing Partner des freihandlabor (seit 2011)
- Fachdozent für Web-Programmierung an der ZHAW (seit 2013)
- E-Commerce Manager bei Bradford Exchange Ltd. (2007-2011)
  - Verantwortlich für E-Shops in 12 Ländern
  - Durchführung und Koordination von Verkaufskampagnen
  - Positionierung und Überwachung von SEO, SEM
  - Analyse und Reporting
  - Beratung und Schulung der Ländermiederlassungen
  - Konzeption und Planung von neuen Features der E-Shops
  - Koordination mit 4 Länderniederlassungen und Zulieferern

JAIME LUIS OBERLE

B.A. FHNW in Produkt- und Industriedesign

- Managing Partner des freihandlabor (seit 2009)
- Fachdozent für Web-Programmierung an der ZHAW (seit 2013)
- System- und Web-Entwickler an der ETH Zürich (2005-2009)
  - Automatisierung der Print-Services
  - Software-Packaging und Provisioning (4 Betriebssysteme)
  - Server- und Client-Administration (ca. 800 Hosts)
  - Entwicklung und Leitung von Web-Projekten
  - Kursleiter für studentische Weiterbildungskurse
  - Führung eines kleinen Teams für Maintenance-Arbeiten

Vorstellungsrunde - Studenten

- Name(n)?
- Job? Wo?
- Hintergrund (Lehre/Studium)?
- Vorkenntnisse Web-Projekte?
- Programmiersprachen?
- Erwartungen an den Kurs?

Termine

- Lektionsplanung unter http://goo.gl/QRQe8R
  - 2.9.14 und 9.9.14 - Einführung
  - ab 16.9.14 - PHP/MySQL - Vertiefung
  - 7.10.14 - Repetition und Zwischenprüfung
  - 14.10.14 - Unterrichtsfreie Woche (Selbststudium)
  - 20.01.15 - Modulprüfung

Web-Applikationen

- Zentrale Daten (Monitoring, Measuring, Security)
- Zentrales Deployment (keine Installation auf Clients)
- Keine Umgebungs-/Versions-Überprüfung auf Clients nötig (latest)
- Updates/Bugfixes viel einfacher und effizienter
- Theoretisch Zugang von überall aus dem www
- Plattform-Unabhängig (≠Browser-Unabhängig :)
- Weniger Aufwand für Support und Wartung
- Acces für Mobiles/Tablets besteht so bereits
- -> User Interface/Experience auch hier immer entscheidend
  - mehr Freiraum, als bei Desktop-Applikationen
  - erfordert gute Kenntnisse des View-Layers (HTML/CSS/JS und Browser)

Warum eine Vorlesung über PHP/MySQL

- PHP wurde seit 1995 als Sprache für das Web entwickelt
  - Open Source
- Sehr viele Web-Projekte schon damit realisiert
- MySQL hat sich als offenes DB-System als de facto Standard etabliert
- Konsequenz
  - Grosses Angebot an Hosting und Platform-Providern
  - Viele (gute) Beispiel-Projekte, Doku, Klassen, Module
- Einfach erlernbar, trotzdem sehr mächtig
  - sehr viele Tutorials und Literatur
  - interpretierte Scriptsprache, C-like Syntax, effizient

PHP - Entstehung I

- PHP 1 wurde 1995 durch Rasmus Lerdorf entwickelt als Sammlung von Perl-Scripts ("Personal Home Page Tools")
- PHP 2 bzw. PHP/FI (Form Interpreter)
  - Portierung von PHP nach C (auch heute noch)
  - sehr Perl-ähnlich, aber eingeschränkt
- PHP 3 wurde dann von Andi Gutmans und Zeev Suraski (Zend) entwickelt, da PHP/FI für eCommerce-Lösungen zu schwach war
- PHP 4 wurde dann von Zend weiter entwickelt und PHP/FI wurde eingestellt

PHP - Entstehung II

- PHP 4 verbesserte und führte ein
  - Ausführungsgeschwindigkeit
  - Session-Management
  - Output-Buffering
  - primitives OOP
- Ablösung des de facto Standards Perl
  - www wuchs Ende 90er sehr stark
  - damit wuchs der Bedarf an einfach erlernbaren Scriptsprachen, die dynamische Websites ermöglichten

PHP - Entstehung III

- PHP 5 erschien im 2004
  - durch weitere Sprachkonstrukte OOP möglich
  - Exceptions
  - mysqli_* Funktionen (i = improved :)
  - SQLite-Support
  - SimpleXML
  - DOM-Extension für dyn. Manipulationen

PHP - Entstehung IV

- PHP 5.1 (2005) - DB-Abstraction mit PDO
- PHP 5.2 (2006)
  - schnellere/effizienteres Memory-Management
  - Filter-Extensions (Sanitization/Validation)
  - JSON-Support (Serialisierung von vars/arrays, AJAX)
  - ZIP-Support
  - OOP Datums-und Zeit-Funktionen
  - Entwicklung seit Anfang 2011 eingestellt
- PHP 5.3 (2009)
  - Namespace-Support
  - optionale Garbage Collection
  - wird heute noch weiter entwickelt und ist sehr verbreitet
    - Verbreitung PHP allg. per Jan 2013: ca. 244 Mio. (80% aller Sites im www)

PHP - Entstehung V

- PHP 5.4 (2012) - "old stable"
  - Default Charset nun UTF-8
  - Simplere Array-Syntax
  - Trait-Support
  - Upload-Process lässt sich über Session monitoren
  - Eingebauter Web-Server für Entwicklung
- PHP 5.5 (2013)- ursprünglich als 6.0 gedacht - "stable"
  - Password-Hashing API
  - Bundled mit OpCode cache
  - Generators (effizientere Iterationen)

Funktionsweise I

- PHP verarbeitet den Code serverseitig
- Quelltext gelangt nicht an Browser, sondern an den Interpreter auf dem Webserver (≠JavaScript)
- Danach schickt der Webserver die Ausgaben aus den Scripten an den Browser (HTML, Bilder, PDFs...)
- Webserver benötigen somit eine Schnittstelle für die Ausführung von PHP (ISAPI, CGI, FCGI, etc.)
  - Supported: Apache, NGINX, IIS, PWS, OmniHTTPD

Funktionsweise II

- Vorteile
  - Client (Browser) benötigt keine Interpreter-Fähigkeiten
  - Quellcode und Daten bleiben auf dem Server
- Nachteile
  - Jede Aktion durch User kann erst bei einem HTTP-Request erfasst werden (AJAX!)
  - Quellcode muss bei jedem Aufruf interpretiert werden, was die Performance mindert (caching, compiling mögl.)

Mängel und Kritik

- PHP ist über lange Zeit unkontrolliert gewachsen
- schwache Typisierung (weak typecasting)
- ähnliche Funktionen wurden unterschiedlich benannt
- logische Reihenfolge von Funktionsparametern ähnlicher Funktionen fehlt zum Teil
- OOP-Fähigkeiten da, aber viele Libraries sind prozedural aufgebaut

Beispiele von PHP im www

- Facebook
- Wikipedia
- Yahoo
- Vimeo
- Flickr

Populäre OSS in PHP

- phpMyAdmin (!), phpPgAdmin (DB-Verwaltung)
- phpBB, vBulletin, BurningBoard (Foren)
- Wordpress, Typo3, Drupal, Contao (CMS)
- Zend, Horde, Cake, Symfony, Yii (Frameworks)
- SugarCRM, Dolibarr (Business Software)
- MediaWiki (Wiki)

Lizenz und Bezug

- PHP 3 - GNU General Public License
- PHP 4 - PHP License (von Zend)
  - erlaubt die freie Verwendung und Veränderung der Sourcen, somit kostenlos
  - OSX/Linux führen PHP standarmässig mit
- [W|M|L]AMP - Komplettpakete für Win/OSX/Linux
  - schwerfällig
  - unterschiedliche Versionen, Umgebungen
  - Provisioning und Config-Management mühsam
  - Deployment-Workflow muss daran angepasst werden

Tech-Stack - Apache

- populärster Web-Server soweit
- bei den meisten Hosting-Anbietern installiert
- mittels Virtual Hosting mehrer Sites betreiben
- Konfiguration über httpd.conf (und includes)
  - Overrides im DocumentRoot mit .htaccess
- Modular aufgebaut, verschiedenste Interpreter

Tech-Stack - PHP I

- Serverseitige Ausführung/Interpretation
  - als (F)CGI
  - mod_php für Apache
  - CLI
- Konfiguration über php.ini, Overrides abhängig von der Schnittstelle zum Web-Server
- php -i in Terminal oder <?php phpinfo(); ?> gibt alle Auskünfte über Konfiguration, Module und Version

Tech-Stack - PHP II

- von der Performance her meistens hinter anderen (vorkompilierten) Interpretern (Python, Perl)
  - Bytecode-Caching möglich
    - OpCode
    - APC
    - Zend Optimizer
    - eAccelerator
  - Kompilierung
    - PHC
    - HipHop (Facebook)
  - Programmierung von GUI-Applikationen auch möglich
    - praktisch keine Verbreitung
    - GTK2 Extensions
    - Qt-Extension (dead)

Tech-Stack - PHP Extensions

- Extensions vergrössern den Funktionsumfang
  - in Form von *.dlls oder *.so Modulen
  - Konfiguration über *.ini Dateien
- PECL - PHP Extension Community Library
  - nur Nutzen, wenn Server/Hoster diese supporten
- PEAR
  - als PHP CLI Script meistens mitinstalliert
  - gute Klassen-Library
  - wird bspw. von Frameworks wie horde verwendet

Tech-Stack - MySQL

- (vom Web-Server) unabhängiger Server
- Open Source Datenbank-System
- kann auf dedizierten Servern laufen (Performance)
- Konfiguration über my.cnf
- kann für die Entwicklung ohne weitere Konfiguration genutzt werden
- Verwaltung optimalerweise mit phpMyAdmin
- Pro Datenbank immer separaten User erstellen, der nur darauf Usage-Rechte besitzt (Credentials stehen immer plaintext im PHP-Source)

Tech-Stack - Debugging I

- Variabeln können mit var_dump() inspiziert werden
- print_r() eignet sich auch limitiert zum einfachen Debugging mittels Ausgabe
- Debugging von interpretierten Sprachen ist meistens eine Herausforderung
  - loose Typing (keine Typensicherheit)
  - evtl. Verwendung anspruchsvoller Funktionen
  - Side-Effects

Tech-Stack - Debugging II

- ein guter Debugger kann hier sehr viel helfen, bedeutet aber Mehraufwand
  - propietärer Debugger von Zend
  - Open Source: Xdebug
- Xdebug
  - als PHP-Modul auf Server installieren
  - auf dem Client (IDE) läuft ein Listener
  - Webserver gibt nun bei Requests nicht nur auf den Web-Ports antwort, sondern auch auf 9000 Debugging-Output
  - Verbinden der IDE mit diesem Port erlaubt schrittweises Debugging darin

Tech-Stack - IDE / Editors

- Netbeans (recommended)
- Aptana (bzw. Eclipse)
- Zend Studio ($$$)
- JetBrains PhpStorm ($$$)
- oder irgendwelche Texteditoren, dann verzichtet man aber auf Code-Completion und Debugging
  - OSX: Coda, SublimeText
  - Linux: vim, emacs, Bluefish, SublimeText
  - Win: Notepad++, SublimeText

Literatur

- http://www.php.net
- http://www.phptherightway.com
- http://httpd.apache.org
- http://www.mysql.com/products/community
- http://xdebug.org
- http://www.vagrantup.com